



# MiR Fleet Enterprise Documentation

Date: 01/2025

Version: 1.2 (en)



## Copyright and disclaimer

Mobile Industrial Robots A/S (MiR) makes no warranties, expressed or implied, in respect of this document or its contents. In addition, the contents of this document are subject to change without prior notice. Every precaution has been taken in the preparation of this document. Nevertheless, MiR assumes no responsibility for errors or omissions or any damages resulting from the use of the information contained.

MiR authorizes you to view, copy, print, and distribute materials available in this document provided that:

- The materials are used for internal informational purposes only.
- A MiR copyright notice appears on every copy of the material and any portion thereof.
- No materials or related graphics are modified or altered in any way. Any rights not expressly granted herein are reserved by MiR.

Copyright © 2024–2025 by Mobile Industrial Robots A/S.

Original instructions (English)

Contact the manufacturer:

Mobile Industrial Robots A/S  
Energivej 51  
DK-5260 Odense S

[www.mobile-industrial-robots.com](http://www.mobile-industrial-robots.com)

Phone: +45 20 377 577

Email: [support@mir-robots.com](mailto:support@mir-robots.com)

CVR: 35251235

# Table of contents

Copyright and disclaimer .....	2
Table of contents .....	3
1. Version history .....	6
2. MiR Fleet features .....	8
2.1 MiR system .....	8
2.2 Alerts .....	9
2.3 Auto Charging and Auto Staging .....	13
2.4 Carts .....	15
2.5 Evacuations .....	20
2.6 Dashboards .....	21
2.7 Footprints .....	26
2.8 Groups .....	30
2.9 Interfaces and modes .....	32
2.10 I/O modules .....	37
2.11 Logging .....	40
2.12 Maps .....	45
2.13 Marker types .....	74
2.14 Missions .....	77
2.15 Mission scheduler .....	82
2.16 Resource handling .....	87
2.17 Robot states and fleet behavior .....	91
2.18 Roles and users .....	95
2.19 Security .....	101
2.20 Sites .....	102
2.21 Sounds .....	104
2.22 Synchronization .....	105
2.23 System settings .....	106
2.24 Transitions .....	107

3. Robot interface .....	110
3.1 Mutual trust page .....	110
3.2 Setup .....	110
3.3 Monitoring .....	114
3.4 System .....	121
3.5 Help .....	142
3.6 Hook .....	146
4. Commissioning .....	160
4.1 Commissioning checklist .....	162
4.2 Migrate data from SW 2.x or 3.x .....	174
4.3 Plan solution .....	175
4.4 Set up MiR Fleet .....	208
4.5 Set up users .....	256
4.6 Connect robots .....	267
4.7 Set up navigation .....	293
4.8 Program robot tasks .....	331
4.9 Configure path planning .....	402
4.10 Assess safety .....	437
4.11 Ensure security and stability .....	463
4.12 Prepare handover .....	471
5. APIs and integration .....	484
5.1 MiR Fleet Integration API .....	484
5.2 Top Module API .....	582
5.3 Shop Floor Connector .....	596
5.4 Robot APIs .....	620
6. Maintenance .....	621
6.1 Upgrade .....	621
6.2 Restart .....	621
6.3 Data backups and site files .....	621
6.4 Error logs .....	622
6.5 Robot hardware .....	622

7. Where to find more information .....	623
7.1 Documentation .....	623
7.2 Models and drawings .....	624
7.3 Resources .....	624

## 1. Version history

This table shows current and previous versions of this document.

Revision	Description
1.2	<p><b>Date:</b> 2025-01-20</p> <p><b>MiR Fleet Enterprise version:</b> 1.1.0</p> <ul style="list-style-type: none"><li>• Updated for Shop Floor Connector 1.1.2 Affects section: Shop Floor Connector</li></ul>
1.1	<p><b>Date:</b> 2024-12-18</p> <p><b>MiR Fleet Enterprise version:</b> 1.1.0</p> <ul style="list-style-type: none"><li>• Updated descriptions for mutual trust and Standalone mode for new interface. Affect sections: Add robots and Interface and modes.</li><li>• Added Resource queue feature. Affects sections: Maps and Resource handling.</li><li>• Improved MiR Fleet Integration API with examples and use cases. Affects sections: MiR Fleet Integration API: Subscriptions, Serial-orders, and Use cases.</li><li>• Added groups to MiR Fleet Integration API. Affects sections: MiR Fleet Integration API: Subscriptions, Schemas, and Endpoints.</li><li>• Added access to Swagger page for Shop Floor Connector and updated for version 1.1.1. Affects section: Shop Floor Connector</li><li>• Added instruction for upgrading and downgrading the robot software. Affects section: Upgrade software.</li><li>• Added description of pallet jack control page in robot interface. Affects section: System.</li></ul>

Revision	Description
	<ul style="list-style-type: none"><li>• Describes how to update Entry positions for MiR1200 Pallet Jack. Affects section: Create positions and markers.</li></ul>
1.0	<p><b>Date:</b> 2024-10-18</p> <p><b>MiR Fleet Enterprise version:</b> 1.0.0</p> <p>First edition.</p>

## 2. MiR Fleet features

The following sections describe the MiR Fleet Enterprise components, features, and tools you can use in your MiR system.

### 2.1 MiR system

A MiR system can consist of many MiR products and compatible modules. The base MiR components are:

- MiR robots with top modules to automate various tasks.
- MiR charging stations to charge robots between tasks.
- MiR Fleet for centralized control to manage robot tasks, charging, and traffic.

MiR Fleet and MiR robots must be connected to the same network and communicate over Wi-Fi or 5G—see "[Meet system requirements](#)" on page 209.

#### 2.1.1 MiR robots

Use MiR robots as a mobile base for any compatible top module. Design or purchase a top module suited for the task you want your MiR application to execute, then program missions to define the tasks your MiR robots will execute.

Each robot has its own network that you can connect the robot's top module or other devices to. You can use the following methods to enable communication between a robot and its top module:

- The [MiR Top Module API](#)
- I/O modules using 24V signals through the top compartment interfaces
- Modbus
- Safety interfaces, intended for safety-related communications

#### 2.1.2 MiR Fleet

MiR Fleet offers centralized control of robots on the same network. It is intended to manage all communication from users and integration systems to MiR robots.



You can operate and set up your MiR system using the web-based user interface. Your entire site is set up, stored in, and shared with robots through MiR Fleet. MiR Fleet can also handle limited traffic and resource management, assign scheduled missions to suitable robots, evacuate robots in case of emergencies, and automatically send robots to charging stations and staging areas. You must define and program all behaviors you want MiR Fleet to apply.

To integrate external systems into your MiR system, use MiR Fleet Integration API with MiR Fleet—see ["MiR Fleet Integration API" on page 484](#).

MiR Fleet **cannot**:

- Manage other vehicles than MiR robots.
- Act as a warehouse management system.
- Collect and analyze historical data. This is done in MiR Insights instead or through custom external integrations—see ["Monitor system" on page 470](#).

### 2.1.3 MiR charging stations

MiR Fleet sends any robots that are not running missions to the nearest MiR charging station. This keeps robots charged and ready for future missions.

MiR Charge 24V is compatible with MiR100 and MiR200.

The MiR Charge 48V models are compatible with MiR250, MiR500, MiR600, MiR1000, MiR1350, and MiR1200 Pallet Jack. The 105A model charges MiR1200 Pallet Jack faster than the 35A model. For all other robots, the charging speed is the same regardless of the charger model.

### 2.1.4 External systems

Integrate any number of additional systems and devices. Use the API options to set up communications to the MiR system—see ["APIs and integration" on page 484](#).

## 2.2 Alerts

The alerts page notifies and logs the following main events on the MiR system:

- Two or more robots are in a deadlocks.
- A robot enters Emergency or Protective stop.
- A robot is in error state.

- A robot is not compatible with MiR Fleet.
- A mission is invalid.

Alerts are intended for internal communication about the MiR system status. It is optional to what extent and how you want to use the alert notations.

For each alert, you can:

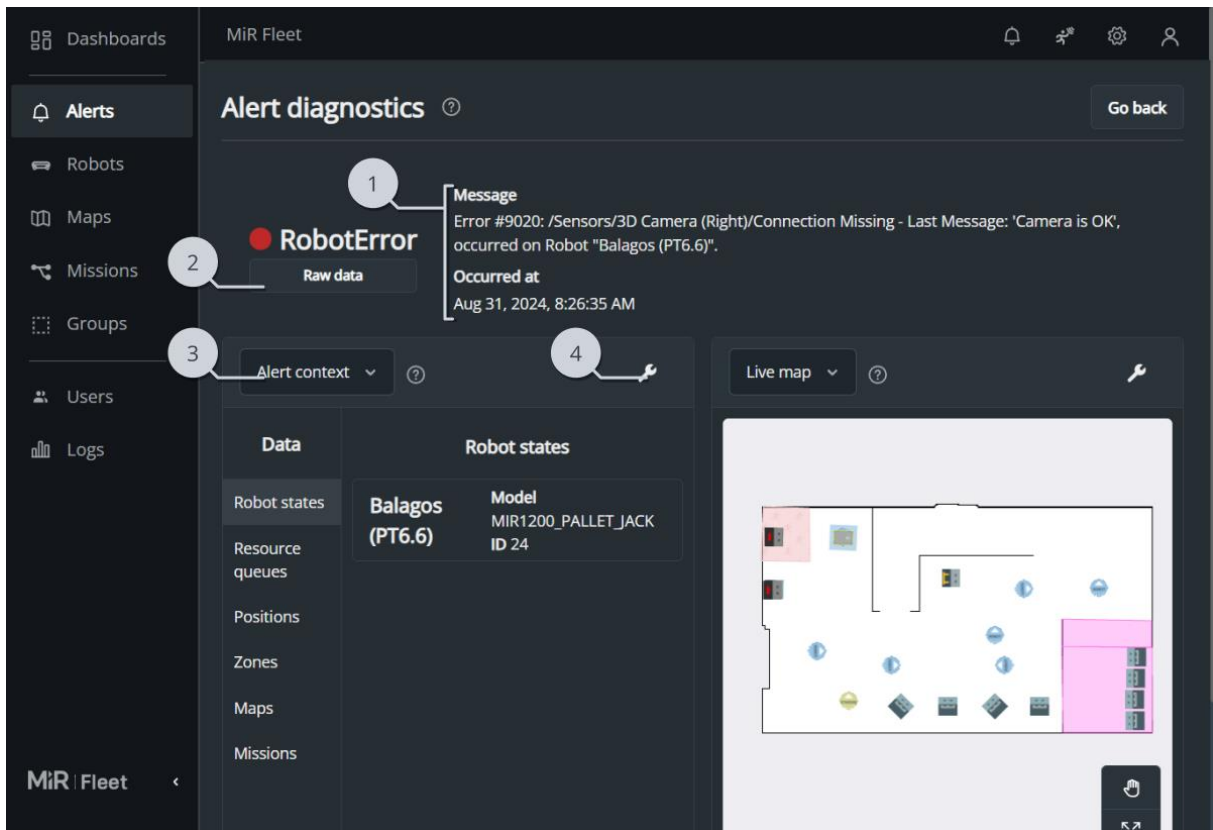
- Assign a priority
- Set the alert status
- Write relevant notes to each alert

None of the alert settings you change have an affect on the rest of the MiR system. When you change the status of an alert to **Resolved**, it is expected that you have resolved the alerted issue manually first.

For more information about different monitoring options and how to use them, see "[Monitor system](#)" on page 470.

### 2.2.1 Alert diagnostics

For more context information for an alert, you can open the alert diagnostics page.



Description	Description
<p>1 <b>Alert basic information</b></p> <p>The base information regarding the alert.</p>	<p>2 <b>Alert data</b></p> <p>View all the alert data in its raw code format.</p>
<p>3 <b>Alert widget selector</b></p> <p>Toggle between the alert widgets—see "<a href="#">Alert widgets</a>" below.</p>	<p>4 <b>Widget settings</b></p> <p>Filter which data is shown in the selected widget. The setting only affect the selected widget.</p>

### 2.2.2 Alert widgets

Use the alert widgets for a visualization of the data recorded when the alert was triggered.

## Alert context

Alert context displays all the data the MiR system collects that is relevant to the alert. Select any of the identified elements to display the raw alert data. The amount of available data varies depending on the alert type:

- **Deadlock detected:** The context identifies which robots are part of the deadlock, the mission they were executing, the map the deadlock occurred on, information about the resources involved, and which zones were activated by the robots.
- **Robot error:** The context identifies which robot is affected and which map it was on when the error occurred. For more information about the error, open the robot's interface and go to **Monitoring > Hardware health** or **System log**.
- **Robot E-stop:** The context identifies which robot is affected and which map it was on when the Emergency or Protective stop occurred. For more information about the robot's state, open the robot's interface and go to **Monitoring > Hardware health** or **System log**.
- **Robot compatibility:** The context identifies which robot is not running a software version that is fully compatible with MiR Fleet.
- **Invalid mission:** The context identifies the robot that tried to execute the mission and the invalid mission.

## Map

Displays the map in its current state. The widget defaults to the map where the alert was triggered. You can change which map is shown in the widget settings.

Use this to see where all robots connected to MiR Fleet are currently located to help you resolve the alert. If you have many alerts that are triggered from the same area, it indicates you should review what may be causing the issue in that area. Location-based issues can often be solved with zones.

## Robot data

Displays the state of each robot when the alert occurred. You can use the widget settings to toggle between all robots being shown or just the alert-related robots.

If you are looking at a deadlock alert, the data also includes which resources each robot was occupying at the time.

If you have a Map widget open at the same time, you can select robots and positions in the Robot data widget and they will be highlighted on the map.

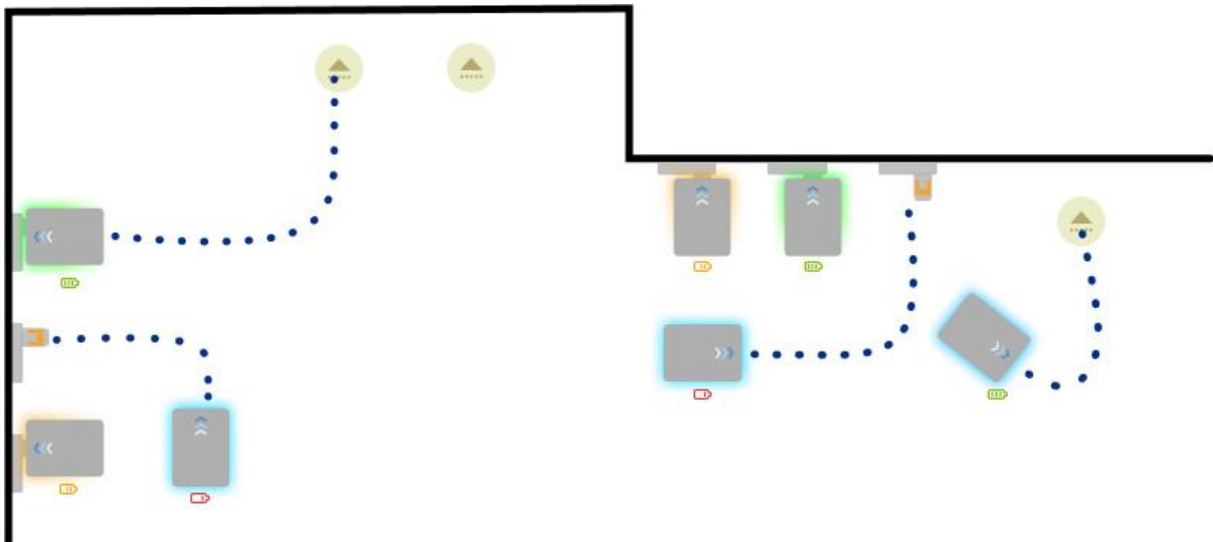
## 2.3 Auto Charging and Auto Staging

When Auto charging and Auto staging are enabled—see ["System settings" on page 106](#)—, MiR Fleet sends idle robots to charging stations and Staging positions automatically. MiR Fleet only manages robots and resources that are not currently used in any user-defined missions. MiR Fleet optimizes the charging flow of robots by prioritizing the following:

- **Emergency battery percentage:** Any robots with a battery percentage below 7% have the highest priority to be sent to a charging station, and can overrule some of the Auto charging settings to be assigned to a charging station.
- **Availability:** Resources that are not assigned to any robots have a higher priority to be assigned than occupied resources.
- **Battery percentage:** Robots with lower battery percentages have a higher priority to be sent to charging stations.
- **Distance:** Robots are assigned to resources so the total distance traveled by all robots is minimized.

There is no priority for MiR1200 Pallet Jack to use Charger 48V 105A. You must use groups to define this behavior—see ["Groups" on page 30](#).

Use the Auto charging system settings to change how MiR Fleet sends and releases robots from charging stations—see ["System settings" on page 106](#).



### 2.3.1 Behavior

MiR Fleet controls robots by assigning them Auto charging or Auto staging missions. Auto charging missions send robots to charging stations and Auto staging missions send robots to Staging positions.

Auto charging and Auto staging missions are predefined and cannot be altered. They appear in the mission queue like all other missions and can be paused and aborted in the same way.

For Auto charging and Auto staging to work optimally, the total number of Staging positions and charging stations must be one more than the total number of robots.

### 2.3.2 Evaluated robots and resources

When MiR Fleet considers which robots to evaluate for Auto charging and staging it does **not** include:

- Robots that are executing a mission. This includes Auto charging and Auto staging missions.
- Robots that are assigned a user-defined order that is pending or outbound.
- Robots that have a battery percentage below 7% and are already charging.

When MiR Fleet considers which resources to evaluate for Auto charging and staging it does **not** include:

- Charging stations or Staging positions that are queued or occupied to be used in a user-defined mission.
- Charging stations or Staging positions that are queued to be used in a Auto charging or Auto staging mission.
- Charging stations where a robot below 7% is already charging.

### 2.3.3 Sequence

The Auto charging and Auto staging sequence can be triggered by:

- The battery percentage on a charging robot.
- A charging station or Staging position is added.
- A group is changed.
- The resource queue is changed.

- An Auto charging mission fails.
- A robot has been idle for longer than the set **Idle time**.
- The Auto charging and Auto staging settings are changed.

If any of the changes above have an effect on whether MiR Fleet is fulfilling the optimal charging setup, the Auto charging and Auto staging sequence starts. During the sequence, MiR Fleet evaluates all of the potential pairs of robots and charging stations or Staging positions to determine the optimal solution for all robots connected to MiR Fleet.

## 2.4 Carts

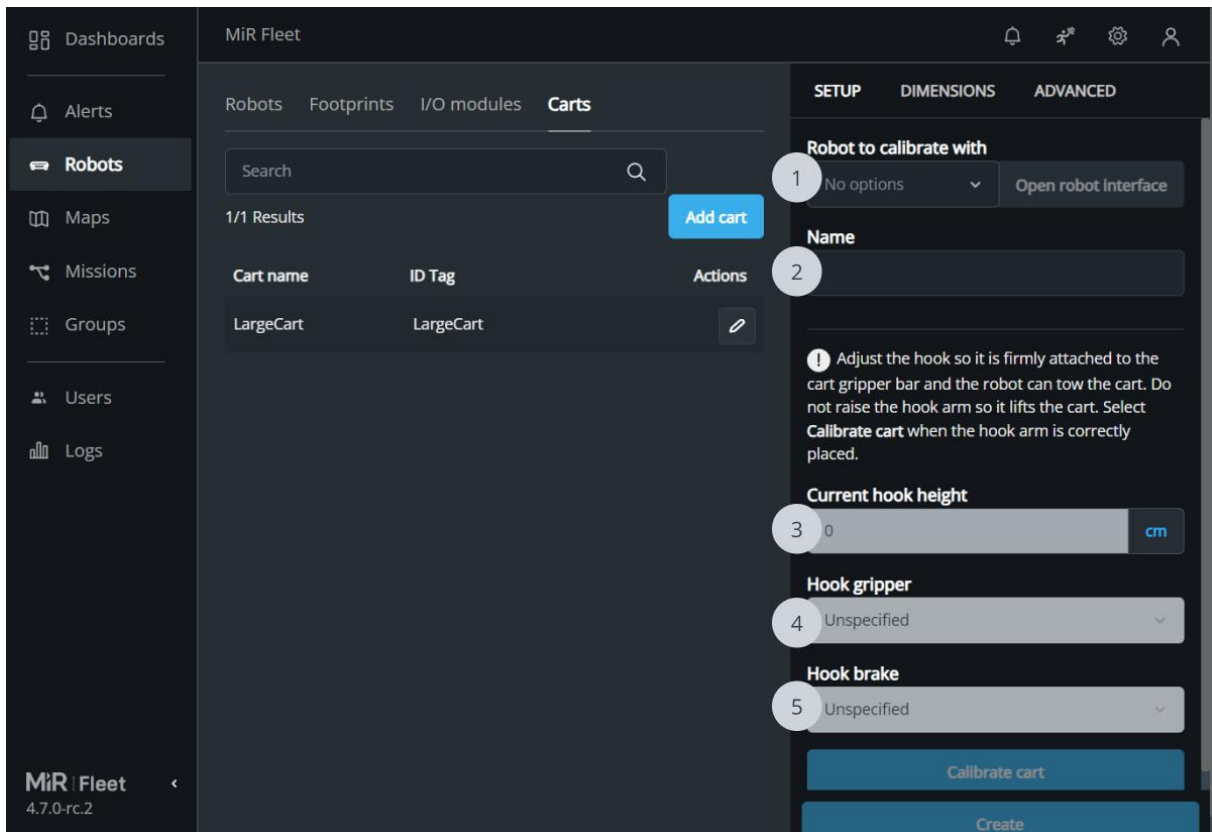
A cart in the user interface is a description of a physical cart that MiR robots can pick up and place. You must define every individual cart that you want the robot to be able to transport.

A cart in the interface is used to provide the information MiR robots need to pick up, drive, and place the cart correctly. When you send the robot to a Cart position to pick up a cart, the robot scans the ID tag on the cart to link the corresponding cart description in the interface that describes the dimensions, calibrations, and docking offsets of the cart.

### 2.4.1 Setup

Use the setup page to calibrate the height of the hook and placement of the ID tag for a specific cart.

When you calibrate a cart, configure the hook so the gripper is attached to the cart gripping bar and is at the hook height you want while it tows the cart. Select **Calibrate cart** to make the robot automatically detect all the necessary values. The determined values are shown under **Advanced**.



Description	Description
<p>1 <b>Robot to calibrate with</b></p> <p>Select a robot with a hook top module that you want to use to calibrate the cart.</p> <p>Place the robot in front of the cart and attach the cart to the gripper at the height you want the robot to transport the cart. You must control the robot and hook from the robot interface.</p>	<p>2 <b>Name</b></p> <p>Enter the ID tag of the cart. The name must be the same as the string of the ID tag.</p> <p>This is used to identify the cart.</p>
<p>3 <b>Hook height</b></p> <p>Displays the current height of the hook. Adjust the value to move the</p>	<p>4 <b>Hook gripper</b></p> <p>Displays the current state of the gripper. Use the drop-down to open and close</p>



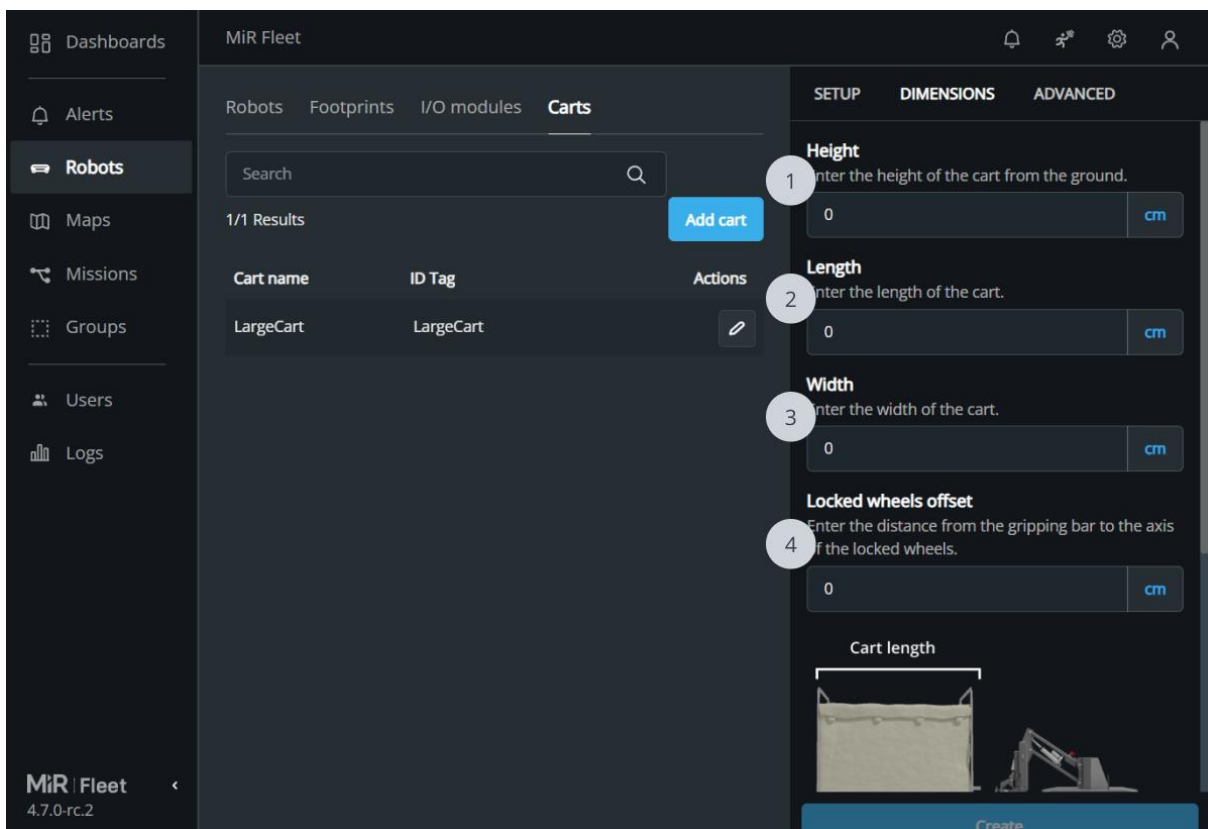
Description	Description
hook up and down.	the gripper.

5 **Hook brake**

Displays the current state of the hook brake. Use the drop-down to activate and deactivate the brake. When the brake is deactivated, the hook arm can swing freely.

### 2.4.2 Dimensions

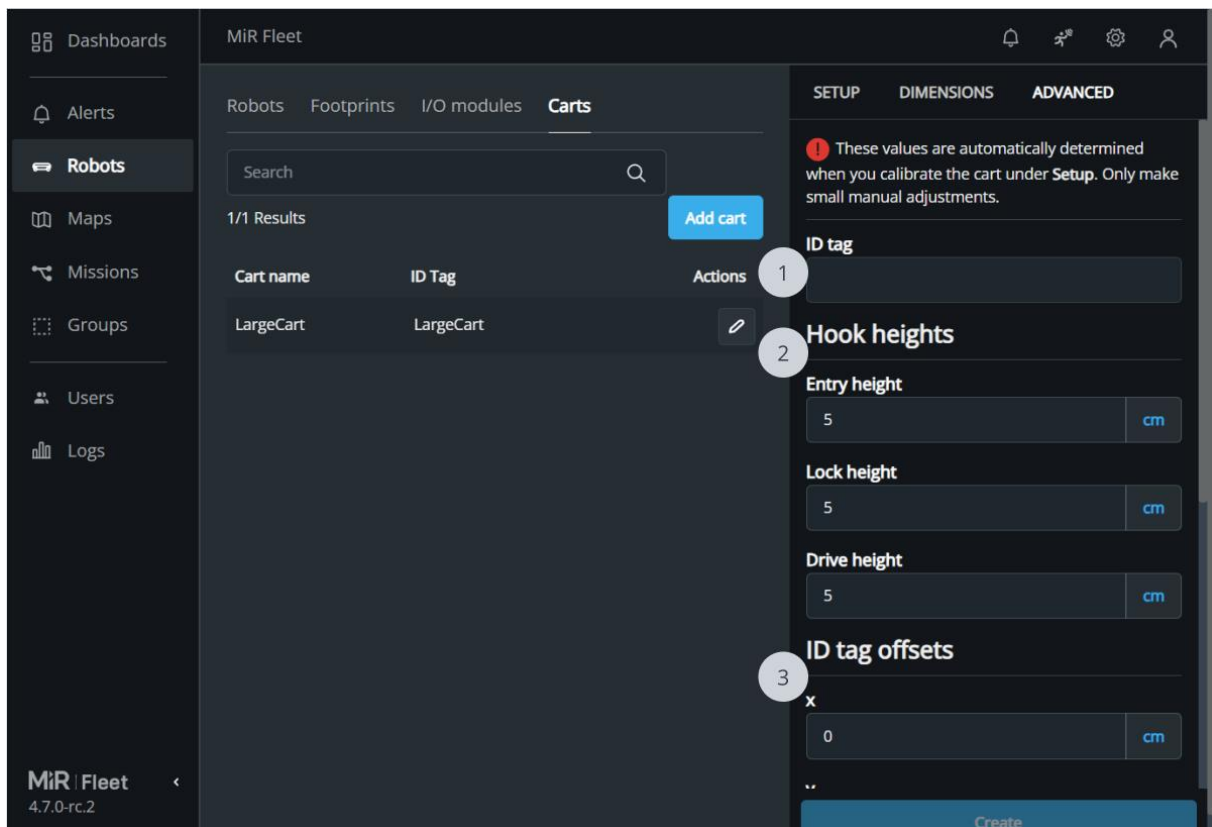
Use the dimension page to specify how large the cart is. The robot uses this information to plan its paths so the cart does not collide with obstacles and to correctly reverse with the cart.


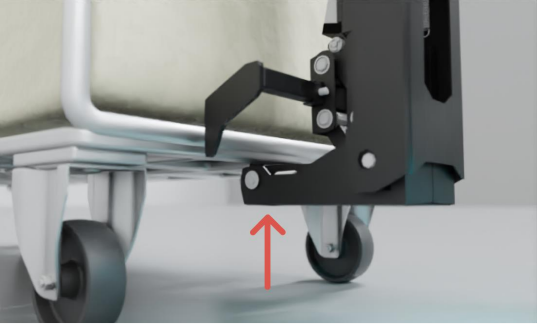
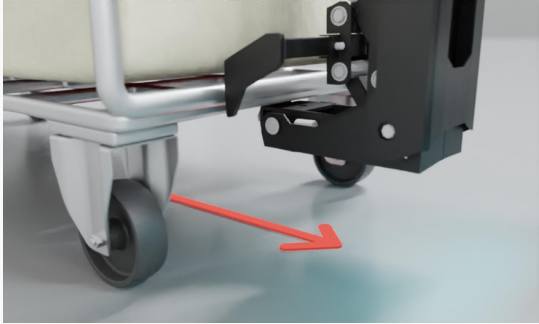


Description	Description
<p>1 <b>Height</b></p> <p>Enter the highest point of the cart measured from the ground.</p>	<p>2 <b>Length</b></p> <p>Enter the length of the cart from the gripper bar to the point furthest from the robot.</p>
<p>3 <b>Width</b></p> <p>Enter the width of the cart.</p>	<p>4 <b>Locked wheels offset</b></p> <p>Enter the distance from the gripper bar to the locked wheels axis.</p>

### 2.4.3 Advanced

Use the advanced tab to see values that the hook detected when you ran the cart calibration. You can adjust these values if necessary.



Description	Description
<p><b>1 ID tag</b></p> <p>The unique tag string. This is the text the ID tag reads when the hook scans it.</p>	<p><b>2 Entry height</b></p> <p>The hook height where the gripper can slide under the gripper bar.</p> 
<p><b>3 Lock height</b></p> <p>The hook height where the gripper can attach to the gripper bar.</p> 	<p><b>4 Drive height</b></p> <p>The hook height where the robot can tow the cart.</p> 
<p><b>5 ID tag offsets</b></p> <p>Values describing the position of the ID tag so the hook camera knows where to detect the tag.</p>	

## 2.5 Evacuations

In the MiR Fleet interface, you can trigger an evacuation to clear all connected robots from specific areas. Use Evacuation zones and Evacuation positions to define how robots should behave when you trigger an evacuation.

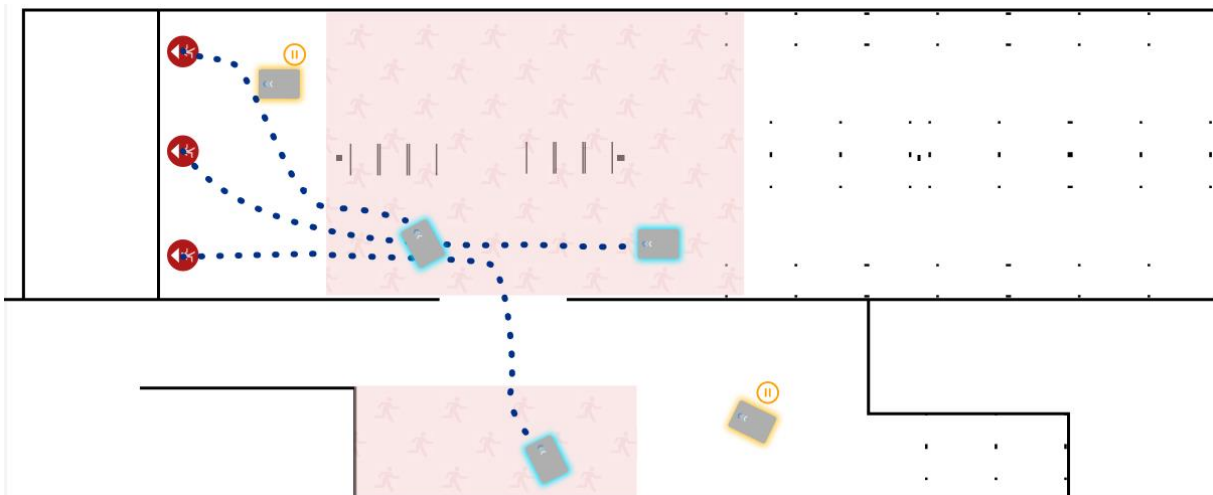
For more information about preparing for emergency events, see ["Define emergency plans" on page 466](#).

For more information about creating zones, see ["Zones" on page 62](#) and ["Create zones" on page 416](#).

For more information about creating positions, see ["Positions" on page 50](#) and ["Create positions and markers" on page 337](#).

When you initiate an evacuation:

- All robots inside an Evacuation zone drive to the nearest Evacuation position.
- Any robot that is not in an Evacuation zone is paused.
- All missions are discontinued or paused.
- When the evacuation is complete, an error log is automatically generated.
- When an evacuation is ongoing, you cannot schedule missions.
- If you scheduled a mission before an evacuation, it will be assigned once the evacuation ends, and a valid robot is in Ready state.



It is only possible to evacuate all Evacuation zones at once.

For a robot to be evacuated, it must part of MiR Fleet and either be in Ready state or executing a mission.

To trigger an evacuation, select **Evacuate all zones** <sup>38</sup> in the upper-right corner in the MiR Fleet interface.

You can also trigger and subscribe to evacuations using MiR Fleet Integration API—see ["Endpoints" on page 525](#) and ["Event subscriptions" on page 496](#).

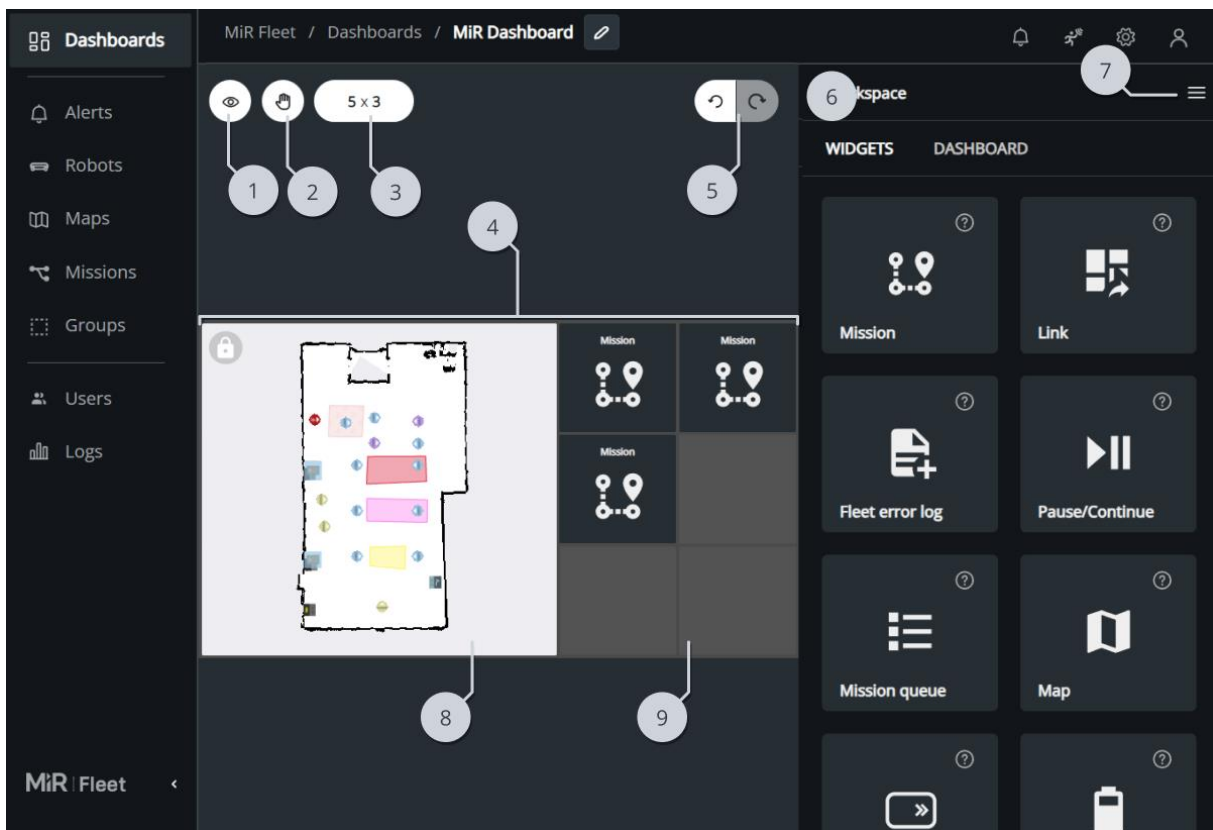
## 2.6 Dashboards

A dashboard is a customizable collection of widgets representing MiR Fleet and robot features.

A dashboard is used to collect the key functions a user needs on a single page.

For guidelines when planning, designing, and creating dashboards for different use cases, see ["Create dashboards" on page 264](#).

### 2.6.1 Editor Overview



Description	Description
<p>1 <b>Preview</b></p> <p>Open a preview of the dashboard. You can view and interact with the dashboard to test all functionality and appearance.</p>	<p>2 <b>Move</b></p> <p>Enable your cursor to drag and move the dashboard canvas around.</p>
<p>3 <b>Grid size</b></p> <p>Change the number of squares in the grid.</p>	<p>4 <b>Dashboard canvas</b></p> <p>Create dashboards on a grid based canvas you can drag and modify widgets on.</p> <p>Change the grid size by dragging the left or bottom edge of the dashboard canvas.</p>
<p>5 <b>Undo and Redo</b></p> <p>Undo the last change you applied. If you regret an Undo, use Redo to reapply the change.</p>	<p>6 <b>Editor dialog box</b></p> <p>Edit the component you are working on.</p> <p>When nothing is selected, you can add widgets to the dashboards or change the overall dashboard aspect ratio and background color.</p> <p>When you select a widget on the dashboard canvas, you can edit the widget settings.</p>
<p>7 <b>Dashboard settings</b></p> <p>Open options to:</p> <ul style="list-style-type: none"> <li>• Save a copy of the dashboard.</li> <li>• Open a preview of the dashboard.</li> <li>• Delete the dashboard.</li> </ul>	<p>8 <b>Placed widget</b></p> <p>Widgets you have placed on the dashboard canvas can be resized, moved, modified, and duplicated.</p>

Description	Description
9 <b>Empty grid spaces</b>	Available space to put widgets on.

## 2.6.2 Widgets

Widgets can be organized and resized according to the canvas grid:

- Select a widget by left-clicking it.
- Delete or duplicate a widget by right-clicking it.
- Resize a widget by dragging the edges or corners of the widget.
- Reposition a widget by dragging it around the dashboard canvas. When you drag a widget around the shadow indicates where it will be placed. A green shadow indicates that the widget uses its default size.
- Select multiple widgets by holding down Shift or Ctrl while selecting widgets.

The following section describe the widgets you can use.

For some widget options, you can enable **User choice**. This lets the user change the value when they activate the widget.

Many widgets also include a **Visual** tab that lets you customize the displayed text, icon, and color of the widget.

You can use the following widgets:

- **Mission:**

The Mission widget adds a preselected mission to the mission queue when activated.

Select the robot you want to execute the mission or select **Any robot** for the mission to be assigned to the next valid robot.

Select a priority to influence in which order the queued missions are executed.

Toggle **Fixed value** to make the value selected in the editor always apply when the widget is activated, or **Request value** to request a value when the widget is activated.

- **Mission queue:**

The Mission queue widget displays the list of queued missions.

Select which mission list you want displayed: Executing, Pending, or Scheduled missions.

Select if you can abort single missions, all missions at once, or cannot abort any missions through the widget.

- **Map:**

The Map widget displays the selected map with all the active robots on that map.

Toggling the map lock enables you to use the tools that are available when viewing the widget in a dashboard. Modify the map position and displayed data to set the default state when the dashboard is opened.

If the map widget follows a selected robot, the widget shows the robot on the map it is currently on. You have the following tools and display options for the widget:

- Schedule a Go-to mission
- Toggle laser scanner data
- Toggle obstacle cloud data
- Toggle path

If the map widget does not follow a robot, the widget only shows the selected map. You have the following tools for the widget:

- Pan or move the map
- Scale and position to map size
- Zoom in and out

- **Fleet Log:**

The Fleet error log widget generates a Fleet error log and saves it under Logs when you activate it.

- **Continue/Pause:**

The Continue/Pause widget continues or pauses the selected robot's current mission when the widget is activated.

If you select **Use robot selector**, and the Robot selector widget is set to **Any robot**, you are prompted to select a robot when you activate the widget.

Toggle **Fixed value** to make the robot selected in the editor always apply when the widget is activated, or **Request value** to make you select the robot when you activate the widget.

- **Battery:**

The Battery widget displays the current battery percentage for the selected robot.



- **Robot Selector:**

The Robot selector widget allows you to select which robot the dashboard applies to.

When you select a robot in the Robot selector widget, all other widgets where you have selected **Use robot selector** are automatically set to the selected robot.

If you add multiple Robot selector widgets, they will copy the chosen robot.

- **Link:**

The Link widget opens the defined link when activated. If the link is to another MiR Fleet dashboard, you jump to the link directly. If the link is to an external page, the link is opened in a new tab.

### 2.6.3 Keyboard shortcuts

Keys	Description
Space	Drag the workspace
Middle mouse button	
ctrl + z	Undo
ctrl + shift + z	Redo
ctrl + y	Redo
ctrl + s	Save
esc	Deselect widget
ctrl + select	Select multiple widgets
shift + select	

Keys	Description
ctrl + d	Duplicate the selected widget
Delete	Delete the selected widget
Backspace	Delete the selected widget
End	Toggle preview
Home	Fit to screen
Scroll	Zoom

## 2.7 Footprints

A footprint defines the amount of space the robot needs around it to operate in the work environment. It consists of a horizontal shape around the robot, slightly bigger than the robot itself, and a maximum height that includes the robot's top module and payload measured from the floor.

The footprint is used to ensure that the robot plans its route to avoid collisions. The robot plans its route so the footprint does not collide with any mapped or detected obstacles. The larger the footprint, the more space the robot ensures is kept between it and all obstacles when planning its path. If the footprint is too small, the robot may plan its path too close to obstacles and affect its driving behavior. If the footprint is smaller than the robot's Protective field, the robot will drive too close to obstacles and trigger a Protective stop.

Robot can only detect obstacle heights with the 3D cameras. This means robots will only detect low hanging obstacles in front of the robot and check if the footprint height fits under the obstacle.

The footprint is only used for planning paths. Footprints do not affect the Personnel detection safety function. Keep the footprint 10-15% larger than the smallest Protective fields to avoid the robot entering Protective stop from driving too close to static obstacles.

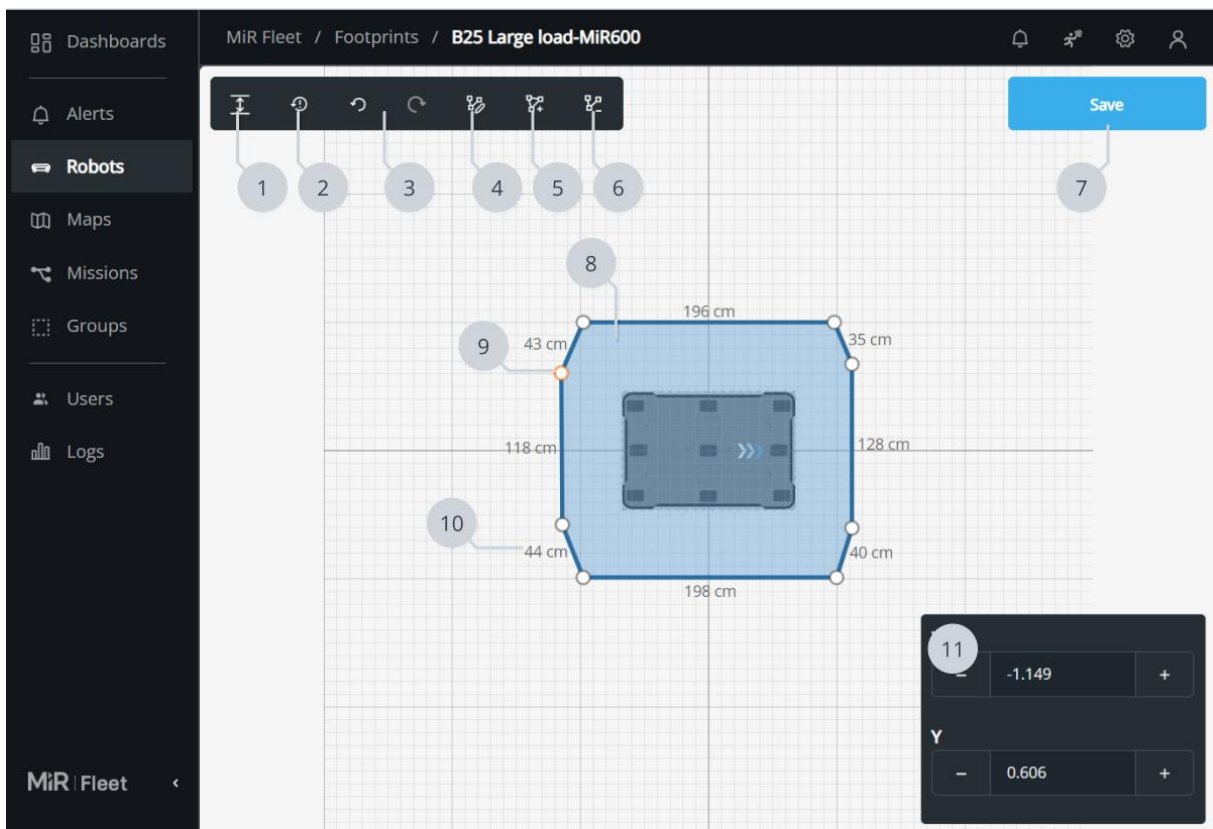
The size of the robot Protective fields depends on the SICK configuration. For the default sizes, see the user guide or integrator manual for your robot application.

For guidelines when planning, designing, and creating footprints for your site, and coordinating footprints with the robot's Protective fields, see ["Create footprints" on page 404](#).

To change the size of the Protective fields, see ["Adjust Protective fields" on page 457](#).

If you modify a footprint that is currently used by a robot, the robot must execute a Set footprint action to apply the changes.

### 2.7.1 Editor overview



Description	Description
1 <b>Edit height</b> Open the footprint height editor. The	2 <b>Reset</b> Reset the footprint to the default

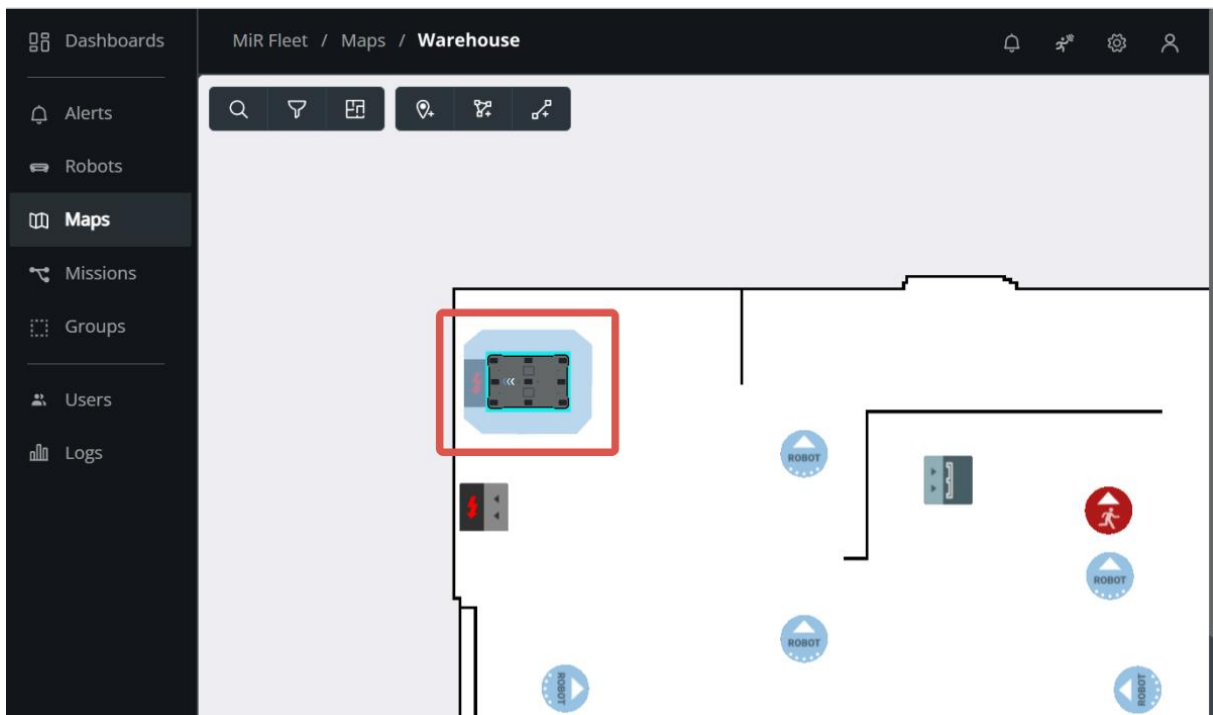
Description	Description
<p>height is measured from the floor.</p>	<p>footprint for the robot type.</p>
<p><b>3 Undo and Redo</b></p> <p>Undo the last change you applied. If you regret an Undo, use Redo to reapply the change.</p>	<p><b>4 Select</b></p> <p>Reposition points you select on the footprint's edge. You can either drag the point to where you want it to be or edit the coordinates.</p>
<p><b>5 Add point</b></p> <p>Add more points to the footprint.</p>	<p><b>6 Remove point</b></p> <p>Remove points you select from the footprint.</p>
<p><b>7 Save</b></p> <p>Save the item you are editing.</p>	<p><b>8 Footprint</b></p> <p>The editor shows a graphical presentation of the footprint you are editing. For reference, the robot is shown in the editor. The footprint must be a convex shape.</p>
<p><b>9 Point</b></p> <p>Modify the point with the selected cursor tool:</p> <ul style="list-style-type: none"> <li>• Drag points around to reposition them.</li> <li>• Double-click a point to remove it.</li> <li>• Click on a line to create a new point.</li> </ul> <p>When you select a point, its X and Y-coordinates are shown in the Point coordinates dialogue and can be</p>	<p><b>10 Line</b></p> <p>Point are visually connected with line segments. The length of each line is displayed next to it.</p> <p>Click on a line to create a new point.</p>

Description	Description
<p>modified for precise adjustments.</p>	
<p>11 <b>Point coordinates</b></p> <p>When you select a point, the X and Y-coordinates of the point are displayed here. You can modify these coordinates to adjust the placement of the point.</p>	

Drag the points to change the size and shape of the footprint, or select a point and enter the X and Y values at the bottom-left corner of the editor.

### 2.7.2 Active and default footprint

The active footprint is the footprint that is currently applied to the robot. The robot's active footprint is shown as a blue shape around the robot.



To change the active footprint, use the Set footprint action found under the Move action group.

The default footprint is the footprint the robot reverts to when it executes a Set footprint action where **Default** is enabled. If you mount a top module to the robot so its physical dimensions are always extended, you should set this as the default footprint.

To change the default footprint, open the robot interface, go to **System > Settings > Planner**, and select a new footprint under **Robot footprint**.

The default footprint should be the footprint you want to easily be able to revert the robot to.

When you change the footprint in a mission, the robot only returns to its default footprint when it executes a Set footprint action that applies the default footprint. The footprint does **not** reset when a mission is completed.

## 2.8 Groups

Use groups to link chargers, staging positions, and mission groups to selected robots.

Groups consist of robots, mission groups, charging stations, and Staging positions.

If you use groups:

- MiR Fleet only sends robots to charging stations that are in the same group as the robot.
- MiR Fleet only sends robots to Staging positions that are in the same group as the robot.
- MiR Fleet only assigns missions to robots that are in the same group as the mission's mission group.

If a robot is not part of a group, it can only use resources and missions that have not been assigned to a group.

MiR robots are never sent to charging stations they are not compatible with, even if they are in the same group.

Robots can be part of multiple groups and then have access to resources in all of the groups it is a part of.

Use groups if you:

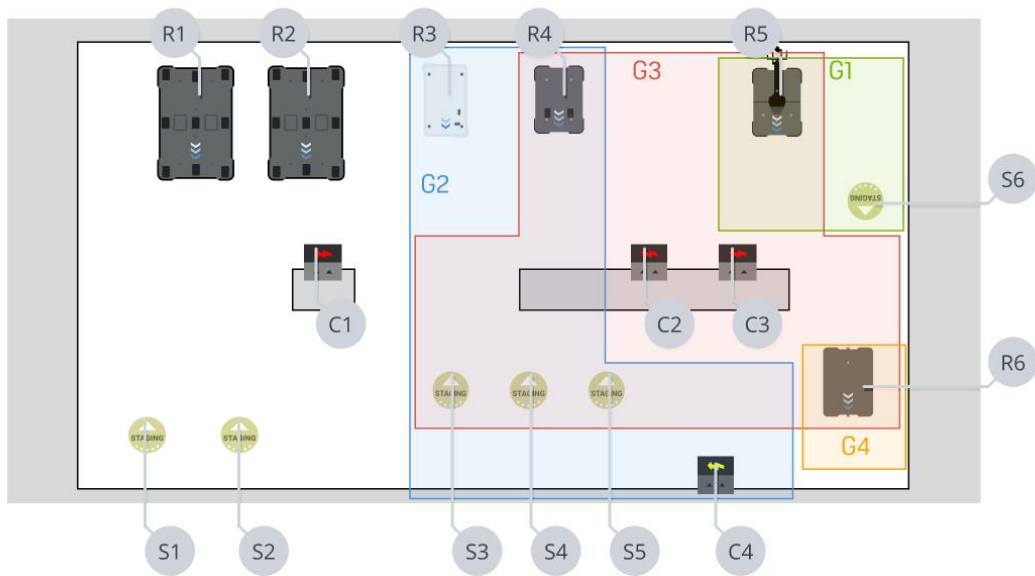
- Have a large site and you want to restrict robots to certain charging and staging areas.
- Have robots with different top modules for specific tasks.

For guidelines when planning and adding components to groups, see ["Create groups" on page 288](#).

### 2.8.1 Group example

The following figure illustrates how you can use groups to manage which Staging positions and charging stations each robot can use and which missions the robots can be assigned to.

Groups					Actions
Name	Positions	Missions	Robots		
G1 Hook	3	1	1		
G2 Deckload	6	1	2		
G3 MiR250	5	1	2		
G4 Shelf	0	1	1		



In this case, each group has a mission group assigned to it with the same group name. The robots in this example can use the following resources:

Robot	Group	Resources	Mission group
R1 (MiR600)	None	C1, S1, and S2	All unassigned mission groups

Robot	Group	Resources	Mission group
R2 (MiR600)	None	C1, S1, and S2	All unassigned mission groups
R3 (MiR100)	G2 (Flat top)	S3, S4, S5, and C4	MiR100 missions
R4 (MiR250)	G2 (Flat top), G3 (MiR250)	S3, S4, S5, C2, and C3	MiR250 missions
R5 (MiR250 Hook)	G1 (Hook), G3 (MiR250)	S3, S4, S5, S6, C2, and C3	Hook missions and MiR250 missions
R6 (MiR250 Shelf Carrier)	G4 (Shelf), G3 (MiR250)	S3, S4, S5, C2, and C3	Shelf missions and MiR250 missions

## 2.9 Interfaces and modes

Robots and MiR Fleet can be accessed and used through different interfaces and modes. The following sections describe the robot interfaces and modes, how to access or enable them, and the affect.

### 2.9.1 MiR Fleet interface

Use the MiR Fleet interface as the main interface to:

- Create, configure, and edit your site.
- Program missions and robot behavior.
- Schedule missions and send commands to robots.
- Monitor MiR robots.

To access the MiR Fleet interface, see ["Sign in" on page 254](#).



## 2.9.2 Application platform interfaces (API)

Use the APIs to integrate external systems such as management applications or top modules to your MiR system—see ["APIs and integration" on page 484](#).

## 2.9.3 Robot interface

Use the basic robot interface to:

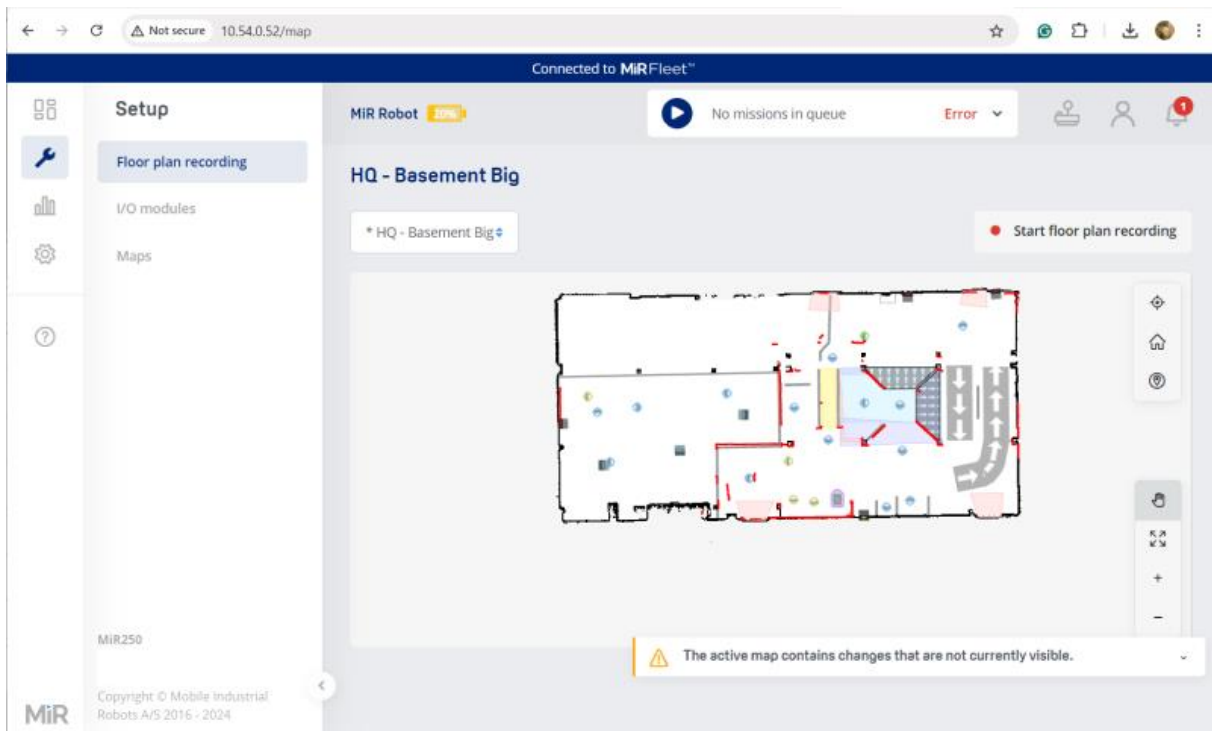
- Record new floor plans that are uploaded to MiR Fleet immediately.
- Move the robot in Manual mode with the joystick.
- Connect the robot to the site Wi-Fi network.
- Adjust the robot settings.
- See the robot's hardware health and other metrics.
- Upgrade the robot software.
- View the robot's system log.
- Set up PLC registers on the robot.
- Generate robot error logs.

For more information about each page, see ["Robot interface" on page 110](#).

When you use the basic interface, the robot is still mainly controlled by MiR Fleet.

The first time you access the basic robot interface, connect to the robot's network using an Ethernet cable or Access point dongle and go to 192.168.12.20—see the user guide or interface manual for your robot application.

After connecting the robot to your site network, you can access the basic interface from the network by entering the robot's IP address into your preferred browser: `http://<robot_ip>`.



### 2.9.4 Standalone robot

Use the Standalone robot mode when you want to use the robot without it being connected to MiR Fleet. Use this when creating and testing the site during commissioning.

When the robot is in Standalone mode, it is no longer controlled by or connected to your main MiR Fleet. The robot creates a local MiR Fleet that runs on the robot itself and only the robot is connected to it.

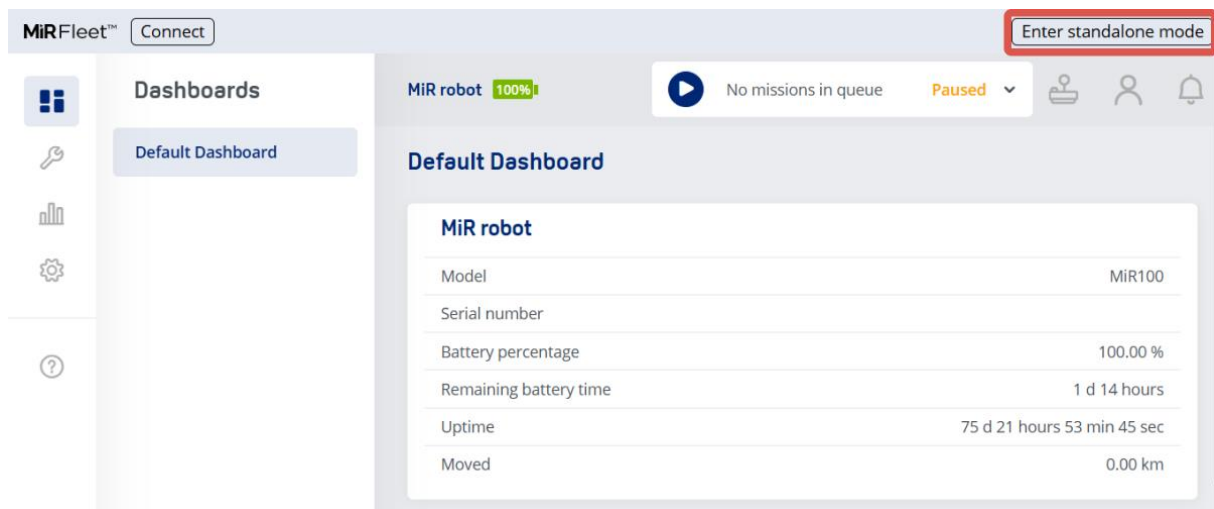
You can assign missions and create site components directly on the robot. None of this data is shared automatically with your main MiR Fleet server. When you are satisfied with the site, export a .site file from the robot's MiR Fleet and import it to your main MiR Fleet to transfer all of the site data.

The site data you create on the robot's local MiR Fleet will continue to be saved on the robot. If you exit Standalone mode and reconnect the robot to the main MiR Fleet, and later reactivate Standalone mode, the robot's local MiR Fleet contains the same site data that was left last time the robot was in Standalone mode.

When the robot is in Standalone mode, it will send itself to charging stations and Staging positions automatically when it is idle. If you have other robots operating on the site connected to MiR Fleet, it does not coordinate charging or staging with the other robots. Occupied resources are handled in the same way as if a physical obstacle is blocking the position.

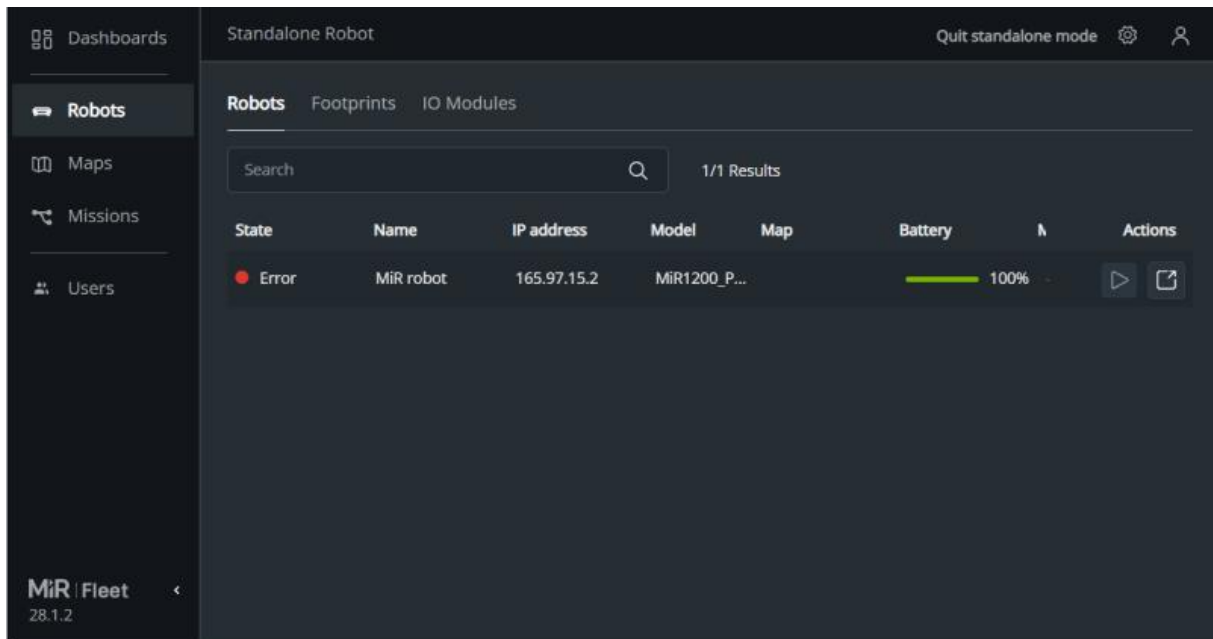
You cannot merge two site files. If you already have some site configurations on MiR Fleet you want to save, you must manually apply these changes again after importing the new .site file.

To enable Standalone mode, go to the robot interface of your robot and select Enter standalone mode.



If this is the first time you use the Standalone interface on the robot, use the default credentials—see ["Sign in" on page 254](#). You will be prompted to enter a new password that you must use next time you sign into the robot with the Standalone interface.

To disable Standalone mode, select **Quit standalone mode** at any time in the upper-right corner of the Standalone interface.



### 2.9.5 Robot platform interface

Use the robot's platform interface to manage backups, software versions, and data storage on the robot—see the guide *How to use backups and recover data on MiR robots*.

To access the Platform interface go to 192.168.12.20:8888 while directly connected to the robot's network with an Ethernet cable or access point dongle.



**SYSTEM OVERVIEW**

Hostname	mir-edb-intel-gen1
Rootfs Partition	(A)
Product Config Overlay	ACTIVE
Product Type	ROBOT
Product Model	MIR100
Product Revision	MK4
Applied Application	
App Launcher Status	FAILED
Application Status	FAILED
Pending Application	Restore
Application Version	
Platform Version	3.2.0.0-g3a95b0899557-4207-develop
Current Time	16/08/2023, 08:09:29 (UTC)

Reboot Platform

Restart Application

Stop Application

Robot Settings

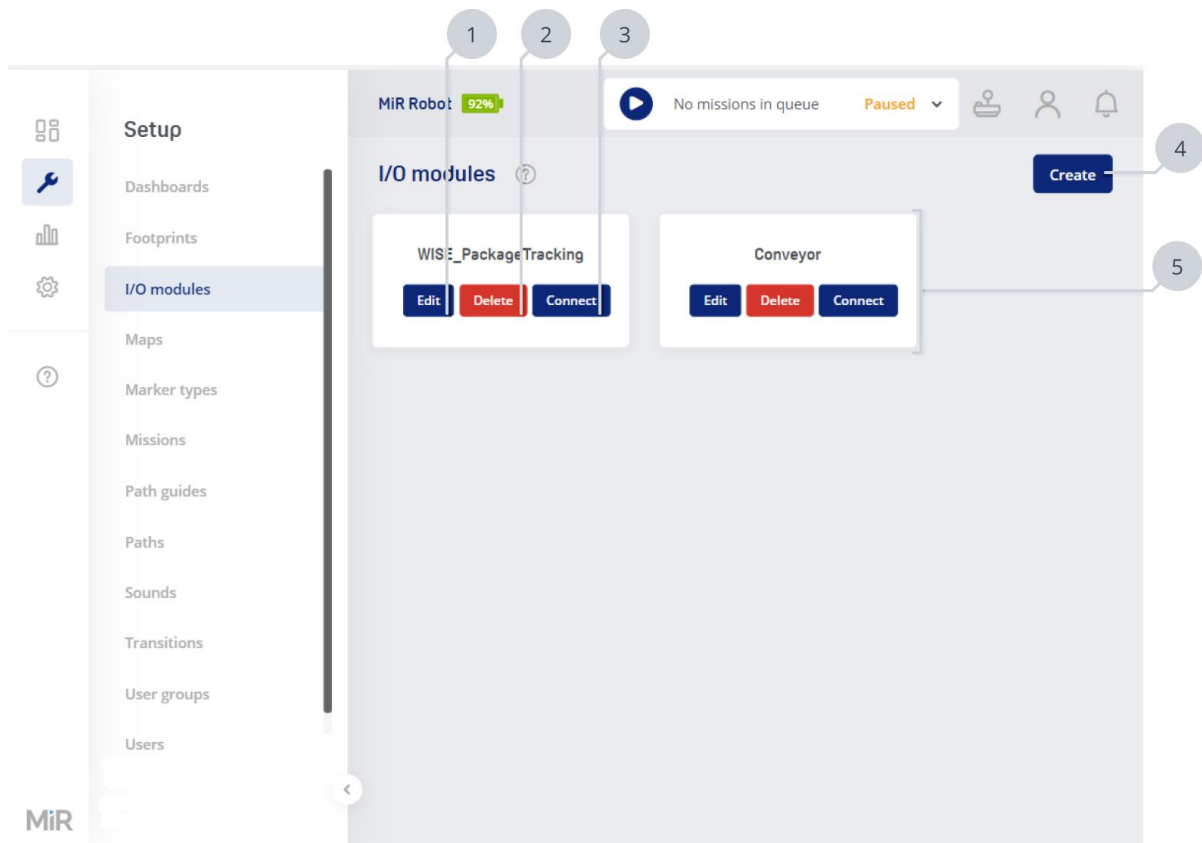
## 2.10 I/O modules

An I/O module defines and controls the robot's connection to an I/O device that the robot can communicate with. The device must be connected to the same network as the robot and is identified using its IP address.

I/O modules are used for receiving inputs and sending outputs with external devices, enabling the robot and the connecting device to send signals that can be linked to certain actions or missions.

Control or read I/O registers using I/O actions in missions or Integration zones on maps.

For more information about setting up WISE modules, see the guide *How to use WISE modules*.

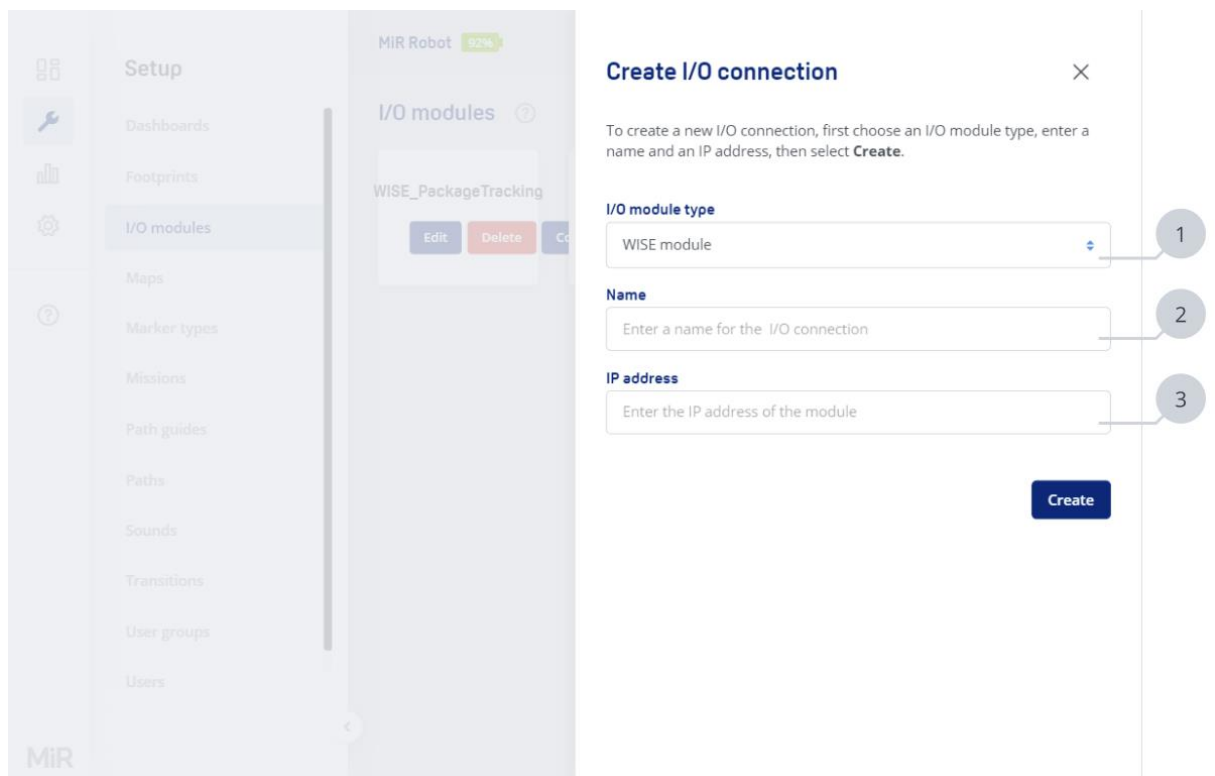


Description	Description
<p>1 <b>Edit</b></p> <p>Opens the dialog box to edit the selected I/O module.</p>	<p>2 <b>Delete</b></p> <p>Removes the selected I/O module.</p>
<p>3 <b>Connect</b></p> <p>Tries to connect the robot to the I/O device. Once connected, you can see the inputs received from the device and control which outputs from the robot are active or inactive. You can use this to test any connected I/O device.</p>	<p>4 <b>Create</b></p> <p>Opens the dialog box to create a new I/O module—see "<a href="#">Create I/O connection</a>" on page 112.</p>

Description	Description
<p>5 <b>I/O modules</b> All of the I/O modules you create to define a connection to an I/O device are displayed here.</p>	

### 2.10.1 Create I/O connection

To create a new I/O connection, first choose an I/O module type, enter a name and an IP address, then select **Create**.



Description	Description
<p>1 <b>I/O module type</b></p>	<p>2 <b>Name</b> Enter a name for the I/O module.</p>

Description	Description
Select the type of I/O device you want to connect to the robot.	
3 <b>IP address</b>  Enter the IP address of the I/O device you want to connect to the robot.	

## 2.11 Logging

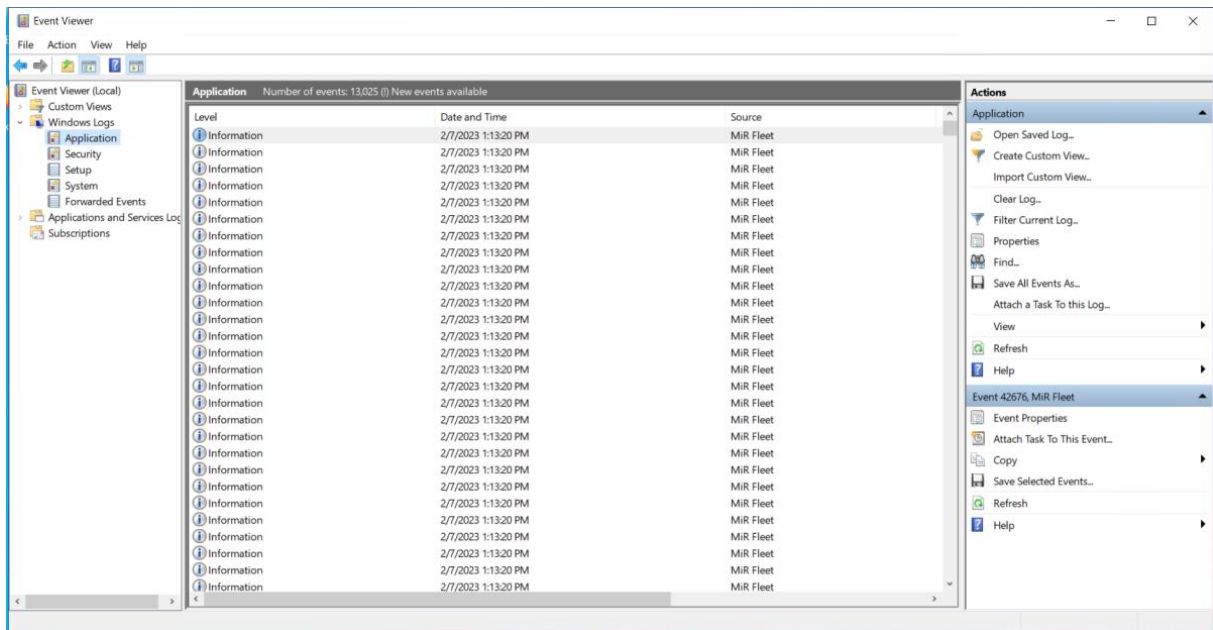
MiR Fleet logs the following data sets:

- **Functional data logs:** All regular MiR Fleet events are logged in the Windows Event Viewer.
- **Audit log:** Security and access related data is logged in the audit log. This is an auto-generated .log file that can be saved anywhere on the MiR Fleet server.
- **Error logs:** More detailed and encrypted error logs that are only intended for MiR Technical Support to help troubleshoot issues. There are both error logs for robots and MiR Fleet.

### 2.11.1 Functional log

MiR Fleet logs all general behavior in Windows Event Viewer. You can find all MiR Fleet related log entries under **Windows Logs > Application**. Filter for events where the source is `MiR Fleet`.





### 2.11.2 Audit log

The audit log is intended for establishing an audit trail and record events relevant to security and forensic analysis. You can set up an application to constantly analyze and monitor the .log file—see "[Monitor system](#)" on page 470.

The audit log is never rotated automatically. You must implement a protocol to clear the log after a given size or date to avoid running out of disk space. If you delete, rename, or move the audit log file, a new one is automatically generated.

#### File location and configuration

By default, the audit log file is saved under `logs/audit.log` in the MiR Fleet installation file.

You can configure the log file path and disable audit logging in the `custom_settings.json` file.

```
{
  "Mir.AuditLogger.AuditLoggerOptions":
  {
    "AuditSink": "File",
    "FilePath": "logs/audit.log"
  }
}
```

- `AuditSink`: Use "File" to log entries into a .log file. Use "None" to disable the audit log.
- `FilePath`: Enter the path you want the audit log to be stored.

## Structure

Each audit log entry follows the structure in the following code snippet.

```
{
  "@t": "",
  "@mt": "",
  "@tr": "",
  "@sp": "",
  "Message": "",
  "AuditLoggerData": {
    "Category": "",
    "Type": "",
    "EventId": "",
    "EventResult": "",
    "$type": ""
  },
  "User": {
    "Name": "",
    "IsAuthenticated": boolean,
    "$type": ""
  }
}
```

- `@t`: Timestamp of the entry
- `@mt`: Message template
- `@tr`: Trace ID of the entry
- `@sp`: Unique identifier that traces related logs
- `Message`: Logged message of one of the following types:
  - `Request`: Highlights a REST endpoint being called on MiR Fleet  
Has the format "Request: [REST Method] [Endpoint route]"
  - `SignalR`: Highlights a SignalR endpoint being called on MiR Fleet  
Has the format "SignalR: [SignalR Endpoint name] called"

- `AuditLoggerData`: A structure containing the following fields about the logged data:
  - `Category`: The type of audit log event. Is always `AuditLogEvent` in this context.
  - `Type`: The application that triggered the data entry. Can be of the following types:
    - `AuditLoggingFilter`: The entry comes from SignalR
    - `HttpRequest`: The entry comes from the REST API
  - `EventId`: The endpoint that triggered the audit log
  - `EventResult`: Informs if the access attempt was done while authenticated:
    - **When authenticated**: `Authenticated access attempt`
    - **When unauthenticated**: `Anonymous access attempt`
  - `$type`: The data type. Is always `AuditLoggerData` in this context
- `User`: A structure describing the relevant user. Contains the following fields about the user:
  - `Name`:
    - If authenticated, this will be a GUID pointing back to the user
    - If unauthenticated, this will say "Unknown"
  - `IsAuthenticated`: A Boolean indicating if the user was authenticated
  - `$type`: The data type. Is always `User` in this context

### 2.11.3 Fleet error logs

Error logs are used by MiR Technical Support to diagnose the cause of any issues occurring with your system. Error logs are generated automatically if MiR Fleet detects an error.

MiR Fleet error logs only include information recorded on MiR Fleet. If a MiR robot has a fleet related issue, always include an error log from the affected robot as well.

You can view, download, and generate a MiR Fleet log from the interface under **Logs**.

Robots can also generate error logs—see ["Error logs" on page 115](#).

#### Configure logging

You can customize how error log data is logged in the file `appsetting.json`. This file is located in the directory where MiR Fleet is installed (by default **C:/Program Files/MiR Fleet/fleet**).

The error log data is only intended for MiR Technical Support. Avoid making changes unless asked to or you have issues storing the fleet log data with the default settings.

The standard settings are:

```
"Serilog": {
  "Using": [
    "Serilog.Sinks.Console",
    "Serilog.Sinks.Elasticsearch"
  ],
  "MinimumLevel": {
    "Default": "Warning",
    "Override": {
      "Mir": "Warning",
      "Microsoft": "Warning",
      "System": "Warning",
      "Microsoft.AspNetCore.SignalR": "Warning",
      "Microsoft.AspNetCore.Http.Connections": "Warning"
    }
  },
  "Enrich": [
    "FromLogContext"
  ],
  "WriteTo": [
    {
      "Name": "Console",
      "Args": {
        "outputTemplate": "{Timestamp:G} [{Level:u3}] [{SourceContext}] {Message}"
      }
    }
  ]
}
```

Common aspects you may want to configure are:

- **Logging system data to files:** If you want to log all the system data to any number of external file locations you can specify any number of logging destinations. This is useful for sites where the built-in logging features are not sufficient for the amount of the log data you want to save for MiR Technical Support.

To set up an additional logging destination, add an item to the `WriteTo` array.

For example, to configure a destination where MiR Fleet logs 500 files of 100 MB each to `C:\Windows\ServiceProfiles\MiRFleet\AppData\Local\MiRFleet\logs\log.jsonl`, use the following item entry:

```
"WriteTo": [
  {
    ...
  },
  {
    "Name": "File",
    "Args": {
      "path": "%localappdata%\MiRFleet\logs\log.jsonl",
      "rollOnFileSizeLimit": "true",
      "fileSizeLimitBytes": 100000000, 100MB
      "retainedFileCountLimit": 500, keep 500 files
      "formatter": "Serilog.Formatting.Elasticsearch.ElasticsearchJsonFormatter, Serilog.Formatting.Elasticsearch"
    }
  }
]
```

- **Limiting the log file size:** Add a memory sink limit to define the maximum number of lines to be logged by adding the `MaxQueueEntries` key to `custom-settings.json`:

```
{
  "Mir.Fleet.ClassLib.InMemorySink.InMemorySinkOptions": {
    "MaxQueueEntries": 5000 // The maximum number of permitted entries for the
    inmemorysink queue
  }
}
```

## 2.12 Maps

A map is a representation of the operating area of the robot. A basic map contains walls and floor that indicate where the robot can drive. You can add various map elements to determine how you want the robot to drive.

A map is used by the robot to navigate. The robot uses the map to determine its current location, where its goal destination is, and how it should get from its current location to the goal.

All maps belong to a site, which is the overall container for one or more maps used in the same facility.

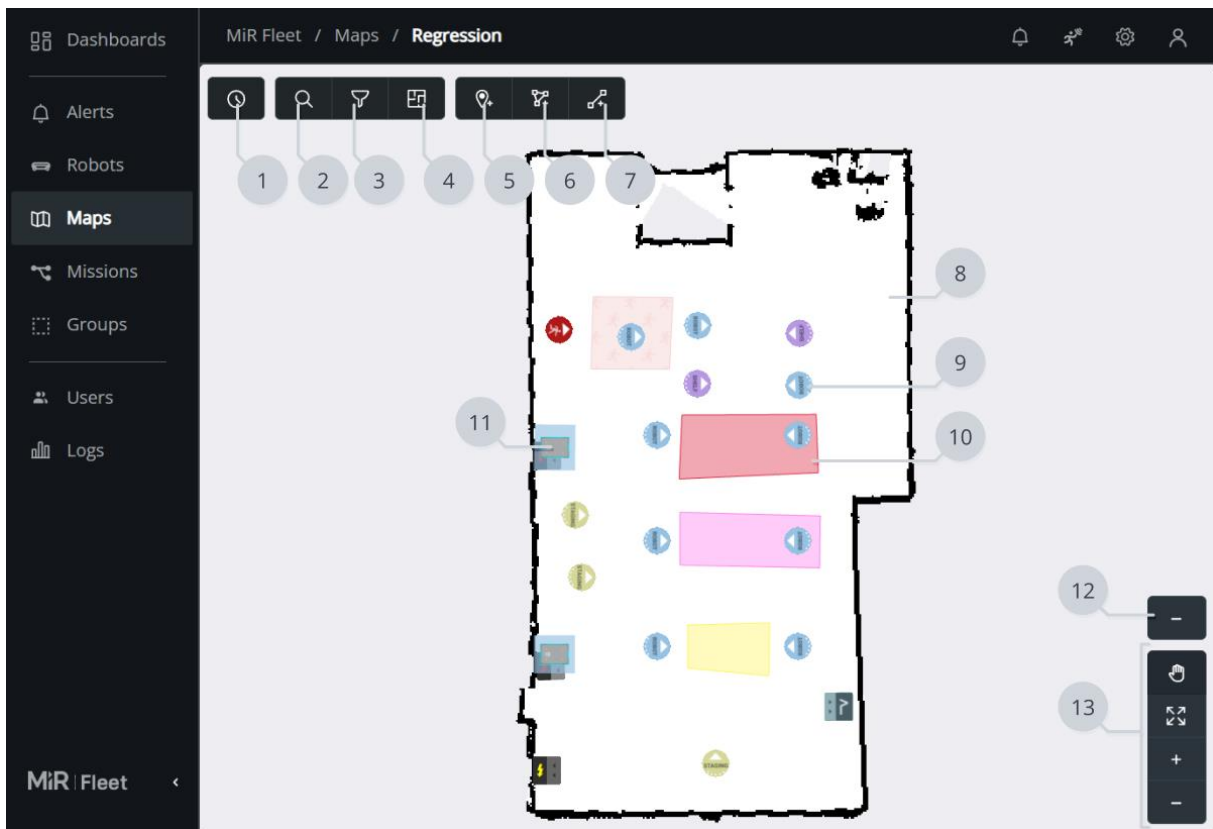
A map consists of two layers:

- A **floor plan** that identifies permanent obstacles and open areas and indicate where the robot can drive.
- **Map components** that you add to the map to control where and how MiR robots drive.

For help creating floor plans by recording or uploading, see ["Create floor plans" on page 294](#).

For help creating maps by connecting floor plans together, see ["Create maps" on page 311](#).

### 2.12.1 Editor Overview



Description	Description
<p>1    <b>Ressource queue</b></p> <p>Displays a list of all resources on the map. Select a resource to see which robots are queued for that resource.</p>	<p>2    <b>Find positions</b></p> <p>Enter the name of the position or marker you want to find. You can also see the list of all markers and positions on the map.</p>
<p>3    <b>Filters</b></p> <p>Apply filters to change which components are displayed on the map.</p>	<p>4    <b>Floor plan</b></p> <p>Edit, download, upload, or append to the floor plan. The floor plan is the base layer of the map including the walls and floor to mark permanent structures in the robot's work environment—see "<a href="#">Floor plan</a>" on the next page.</p>
<p>5    <b>Add position</b></p> <p>Select a <a href="#">marker</a> or <a href="#">position</a> from the list that you want to add to the map.</p>	<p>6    <b>Draw a new shape</b></p> <p>Select a <a href="#">zone type</a> you want to add to the map, and draw a polygon shape to cover the area you want the zone to cover.</p>
<p>7    <b>Draw a line</b></p> <p>Select a <a href="#">zone type</a> you want to add to the map, and draw a path with that zone. You can modify the width of the zone path while creating it.</p>	<p>8    <b>Floor plan</b></p> <p>The floor plan is the bottom layer of the map that shows the walls and floor of the operating area.</p>
<p>9    <b>Position</b></p> <p>All positions and markers you add to the map</p>	<p>10    <b>Zone</b></p> <p>All zones you add to the map</p>

Description		Description	
	are displayed with icons that represent the type of position or marker.		are displayed with different colors that represent the type of zone.
11	<p><b>Robot</b></p> <p>All robots connected to MiR Fleet that are on the active map are displayed. As the robots drive around, their positions on the map update. The blue shape around the robot matches the shape of the robot's active footprint. The colored cloud around the robot matches the robot's status light and indicates the robot's current state.</p>	12	<p><b>Grid</b></p> <p>Toggle the grid overlay.</p>
13	<p><b>Navigation</b></p> <p>Use the navigation tools to pan, zoom in and zoom out of the map, or fit the view to the map.</p>		

### 2.12.2 Floor plan

The floor plan is the bottom layer of the map and represents the operating area of the robot. It consists only of:

- **Walls:** Walls mark areas on the map where there are permanent physical obstacles the robot cannot drive through and can use to localize itself on the map. You can add straight lines to create a more legible map and decide how thick the walls should be by editing the stroke width.
- **Floor:** The floor on the map marks the areas where the robot is able to drive. When mapping, the floor is created automatically. You can use the Floor tool to touch up the existing floor if there are areas where the floor is missing.
- **Unknown space:** The robot cannot drive into areas that are not marked as floor. The robot does not include unknown space when localizing.

You can [record](#) or [upload](#) any number of floor plans which are always available in the floor plan selector. You can use floor plans to create new maps and to append to existing maps.



For guidelines when planning, designing, and creating floor plans for your site, and to help determine whether to record or upload your floor plan, see ["Create floor plans" on page 294](#).

The conversion scale from the floor plan to the real work environment is 20 px to 1 m.

## Record

For guidelines on how to best record a map, see ["Record floor plan" on page 300](#).

Recording is only possible on the robot interface. After you make a floor plan recording, it is automatically saved on MiR Fleet and can be used to create a new map or be appended to an existing map.

When you start a floor plan recording, the mapping engine starts. The mapping engine uses input from the safety laser scanners and motor encoders to create a 2D representation of the robot's operating environment. All physical obstacles that the scanners detect are recorded on the floor plan as walls. The floor plan only includes obstacles that the robot detects at the laser scanner height (200 mm from the floor).

The 3D cameras turn off while mapping and post-processing the floor plan. The only data that is saved while mapping is the created floor plan.

The robot will drive at medium speed while mapping to ensure better coverage.

## Upload

You can upload a .png file to MiR Fleet of a floor plan you have saved on your device. The floor plan can be one you have exported from a MiR system, or you can upload a floor plan created from a CAD file. When you upload a map, the scale must be 20 pixels to 1 m.

For guidelines on uploading maps, see ["Upload floor plan" on page 305](#).

## Append

Keeps the current floor plan on the map and lets you append the selected floor plan to it. Once you have appended a floor plan, you cannot separate them again, though each floor plan is still saved individually in your collection of floor plans.

All floor plans you have previously uploaded or recorded can be appended.

For guidelines on appending floor plans to existing maps, see ["Combine floor plans to make a map" on page 312](#).

## Adjust

Adjust the placement of the floor plan by changing its X-Y placement on the map or rotating it.

## Download

You can download a black and white .png file of the floor plan. You can make modifications to the floor plan in your preferred editor and upload it to MiR Fleet with your changes. Keep the correction ratio: 20 pixels to 1 m.

### 2.12.3 Positions

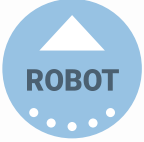

Positions are defined as X-Y coordinates on a map that mark locations where you want the robot to travel to.



Positions are used either as destination positions or as waypoints on a route that you want to use in missions. With positions, the robot does not compare its position to a physical entity, making them less accurate than markers.



The final orientation of the robot is indicated by the arrow on the position icon.

To drive to a position, the robot must be correctly localized on the map—see "[Evaluate localization](#)" on page 321.

For more information about creating positions, see "[Create positions and markers](#)" on page 337.

Graphic	Description
	<p><b>Robot position</b></p> <p>A Robot position marks a location where the robot can be sent to. It has no special features.</p>
	<p><b>Cart position</b></p> <p>A Cart position marks where a robot with a hook can pick up and place carts.</p> <p>When a robot picks up a cart on a Cart position, you must select a Cart type that describes the dimensions of the cart the robot is docking to.</p>

Graphic	Description
	<p>When a robot places a cart on a Cart position, you must select how the robot should park the cart.</p> <p><b>Shelf position</b></p> <p>A Shelf position marks where a robot with a shelf top module can pick up and place shelves.</p> <p>A robot can place shelves on other positions also, but then it cannot pick up the shelf again autonomously.</p> <p>Shelf positions act both as markers and positions. When there is a shelf on the position, the robot uses the shelf as a marker to dock to. When there are no shelves on the position, the robot treats the position like a Robot position since there is nothing physical to dock to.</p> <p>When a robot docks to a shelf on a Shelf position, you must select a marker type that describes the dimensions of the shelf the robot is docking to.</p>
	<p><b>Pallet position</b></p> <p>A Pallet position marks where MiR1200 Pallet Jack can pick up and place pallets.</p> <p>MiR1200 Pallet Jack can place pallets on other positions also, but then it cannot pick up the pallet again autonomously.</p> <p>Pallet positions act both as markers and positions. When there is a pallet on the position, the robot uses the pallet as a marker to dock to. When there is no pallet on the position, the robot treats the position like a Robot position since there is nothing physical to dock to.</p>

Graphic	Description
	<b>Evacuation position</b>  An Evacuation position marks where a robot must go when commanded to evacuate all Evacuation zones from MiR Fleet. This position is only used in MiR Fleet.
	<b>Staging position</b>  A Staging position marks where a robot can wait if it is idle. This position is only used in MiR Fleet.



Practice using positions in the lesson *Define positions*. You can find this lesson on [MiR Academy](#).

## 2.12.4 Markers

Markers are defined as X-Y coordinates on a map that mark locations where you want the robot to travel to. Markers are points on the map that mark a physical entity, such as a charging station or a pallet rack, and enable the robot to position itself accurately relative to this entity.

You should always use markers when it is important that the robot is positioned accurately relative to an object in the work environment, such as load transfer stations and work stations.

To drive to a marker, the robot must be correctly localized on the map—see "[Evaluate localization](#)" on page 321.

For more information about creating markers, see "[Create positions and markers](#)" on page 337.

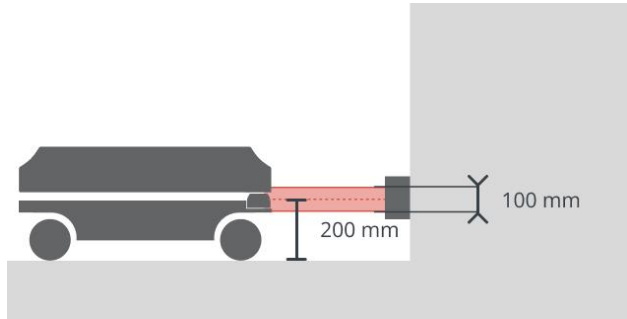


Practice creating markers in the lesson *Add markers*. You can find this lesson on [MiR Academy](#).

## Marker specifications and types

Markers must be:



- Coated with matte and neutral colored paint. We recommend matte iron gray (RAL 7011).
- Placed 200 mm from the ground and at least 100 mm tall. This corresponds to the robot's laser scanner detection area.

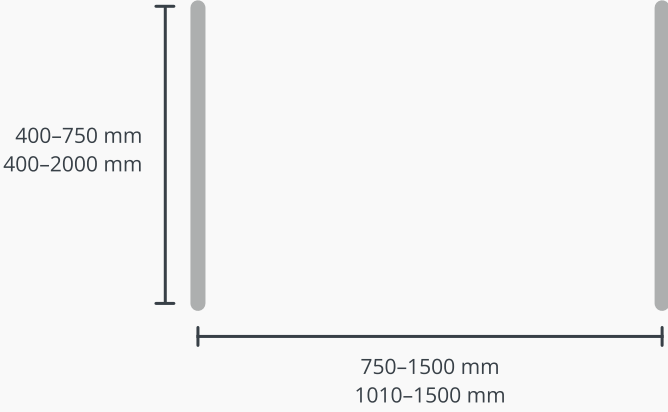

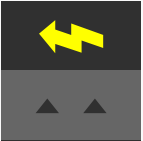
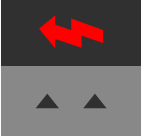


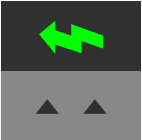
- Meet the dimensions specified in [Table 2.1](#) for the marker type.

**Table 2.1** Marker types

Graphic	Description
	<p><b>L-marker</b></p> <p>Robots can dock to L-markers in several different ways. Robots can both dock to the inside and outside of an L-marker, and the marker can be on any side of the robot.</p> <p>400 mm × 600 mm (±1 mm) with a 90° (±1°) angle between the plates.</p> <p>The diagram shows an L-shaped marker. The vertical leg is labeled with a dimension of 400 mm. The horizontal leg is labeled with a dimension of 600 mm. The two legs meet at a 90-degree angle.</p>

Graphic	Description
	<p><b>VL-marker</b></p> <p>VL-markers are similar to V-markers but have an additional plate that enables the robot to dock more accurately to the marker. They are also designed for the robot to either dock so its front or its rear is facing the marker.</p> <p>150 (<math>\pm 1</math> mm) long with a <math>120^\circ</math> (<math>\pm 1^\circ</math>) angle and 350 mm (<math>\pm 25</math> mm) long plate to the right</p> 
	<p><b>V-marker</b></p> <p>V-markers are designed for the robot to dock with its front or its rear is facing the marker.</p> <p>150 mm (<math>\pm 1</math> mm) long with a <math>120^\circ</math> (<math>\pm 1^\circ</math>) angle</p> 
	<p><b>Bar-marker</b></p> <p>Bar-markers are two long bars with enough space for the robot to fit between them.</p>

Graphic	Description
	<p>They can be used if you design custom pallet racks—see <a href="#">"Custom pallet racks" on the next page</a>.</p> <p>MiR100, MiR200, MiR250: 400–750 mm long and 750–1500 mm between</p> <p>MiR500, MiR600, MiR1000, MiR1350: 400–2 000 mm long and 1 010–1 500 mm between</p> 
	<p><b>Pallet rack</b></p> <p>Intended for physical instances of MiR EU Pallet Rack, MiR Pallet Rack, or a custom rack with the same dimensions—see <a href="#">"Custom pallet racks" on the next page</a>.</p>
	<p><b>MiR Charge 24V</b></p> <p>Intended for physical instances of MiR Charge 24V.</p>
	<p><b>MiR Charge 48V 35A</b></p> <p>Represents where the robot can find a physical entity of MiR Charge 48V 35A.</p>

Graphic	Description
	MiR Charge 48V 35A can be used by MiR250, MiR500, MiR600, MiR1000, and MiR1350. It can also be used by MiR1200 Pallet Jack, but will charge the robot slower than MiR Charge 48V 105A.
	<p><b>MiR Charge 48V 105A</b></p> <p>Represents where the robot can find a physical entity of MiR Charge 48V 105A.</p> <p>MiR Charge 48V 105A is intended for MiR1200 Pallet Jack and charges the robot faster than MiR Charge 48V 35A.</p> <p>It can also be used by MiR250, MiR500, MiR600, MiR1000, and MiR1350, but there is no difference in the charging speed between the 35A and 105A variants.</p>



For more information about marker dimensions and how the robots docks to them, see *How to create and dock to V-markers, VL-markers, L-markers, and Bar-markers*. You can find this guide on [MiR Support Portal](#).

## Custom pallet racks

If you do not use one of MiR's pallet rack products with the robot and choose to create custom pallet racks, you can either use a Bar-marker or a Pallet rack marker to mark it on the map.

### Pallet rack marker

Use a Pallet rack marker if the dimensions of your pallet racks are the same as the MiR pallet racks—see MiR's website for the pallet rack specifications.

This enables the robot to use the Check position status action to check if there is a pallet on top of a pallet rack before docking to it. This feature is not available for Bar-markers.

### Bar-marker

Use a Bar-marker if the dimensions of your custom pallet racks differ from the MiR pallet racks.



Make sure the height of your pallet rack is suitable for the height specifications and tolerances for your top application.



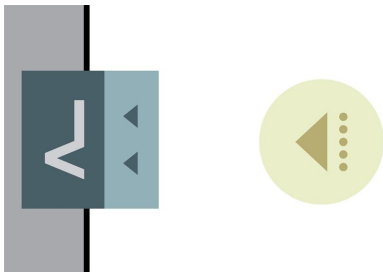
For more information about creating custom pallet racks, see *How to create custom pallet racks*. You can find this guide on [MiR Support Portal](#).

## Docking to markers

When robots position themselves at a set point relative to the marker, the robot must do a docking sequence. When the robot is docking, it uses its safety laser scanners to detect the marker and drives itself to the correct position relative to the detected marker. The robot begins docking to a marker from the marker's Entry position—see [Figure 2.1](#). The Entry position is automatically created approximately one meter in front of the marker and can be moved in the map editor.

The Entry position can be manually moved, however, moving the entry position does not guarantee that the robot is able to dock to the marker or dock within the claimed specifications. It is the responsibility of the person who moves the Entry position to validate and ensure that the docking can still be performed.

**Figure 2.1** A VL-marker with its Entry position from the robot interface



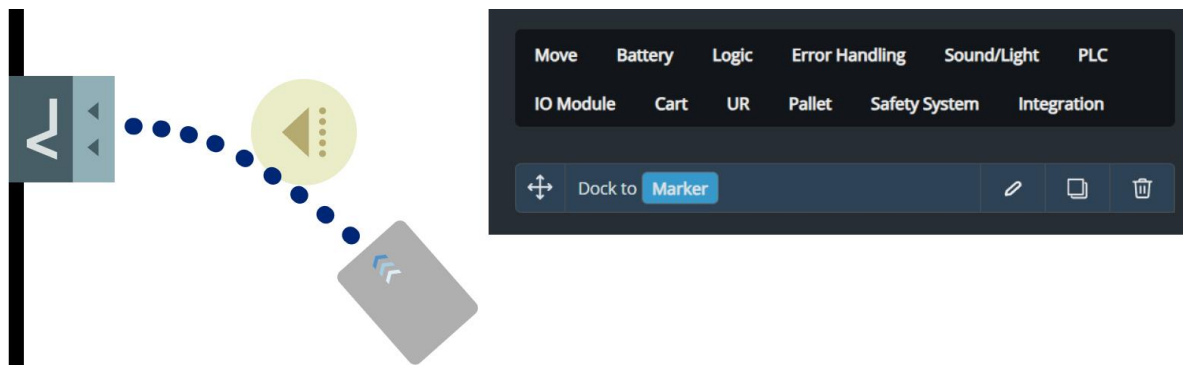
L-markers are the only markers where the Protective fields are not muted automatically. If you want the robot to mute the Protective fields while docking to an L-marker, you must use a Mute Protective fields action.

## Fast docking and Slow docking

Depending on how you create the docking action in a mission, the robot will either dock quickly to the marker in one motion, or it will drive to the Entry position and dock from there.

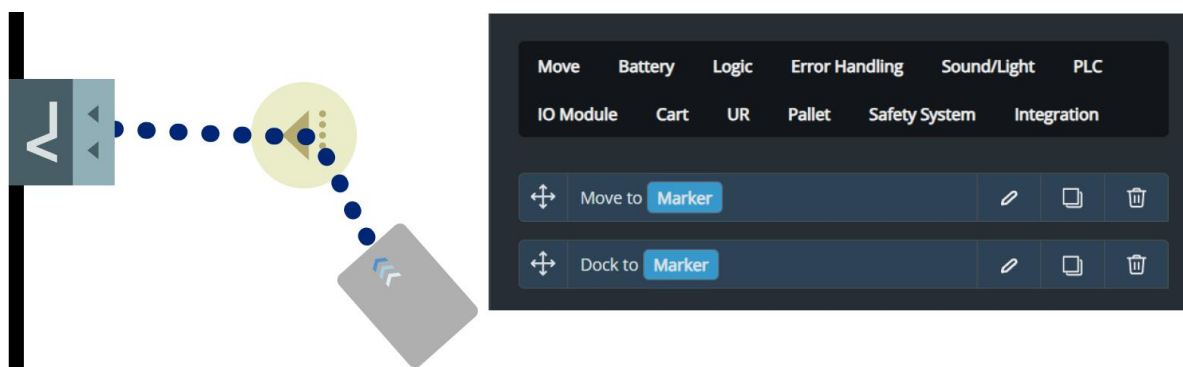
If you only use a Dock to action to move the robot to the marker, it will aim for the Entry position but will begin docking as soon as it detects the marker. This means the robot might never stop at the Entry position before beginning to dock and will instead dock to the marker in one smooth motion.

**Figure 2.2** Illustration of how the robot Fast docks when the mission only uses a Dock action



If you add a Move action to the same marker before the Dock action, the robot will first drive to the Entry position and stop there before beginning the docking sequence. This can take longer than Fast docking, but it increases how precisely the robot docks to the marker. If the robot often does not dock precisely to a marker, we recommend modifying the mission to make the robot stop at the Entry position before docking.

**Figure 2.3** Illustration of how the robot Slow docks when the mission uses a Move action followed by Dock action to the same marker



## Undocking from markers

A MiR robot can undock from markers automatically (except L-markers). When undocking, the robot reverses from the marker until it is outside of the undocking area—see [Figure 2.4](#). The robot will keep muting its Protective fields while it undocks. Once the robot begins a new action

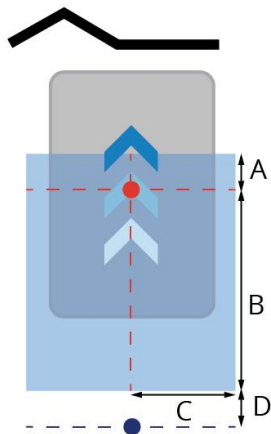
that requires it to plan a new path, the Protective fields are activated again, and the robot continues normal operation.

To undock from L-markers the robot needs to use a Relative move as part of its mission. The Relative move action makes the robot drive an appropriate distance from the marker either forward or backward, depending on the robot's orientation. The distance depends on the docking offsets.

Enter a value in the X field of the Relative move action to define how the robot should move. A positive value moves the robot forward, and a negative value moves it backward.

The undocking sequence will always occur if a robot begins an action where it plans a new route while inside the undocking area, even if the robot was not previously docked to the marker. This means the sequence is also initiated on robots that have entered the area via a Relative move action or Manual control.

**Figure 2.4** The red dot represents the center of the robot when it is docked to the marker, the blue dot represents how far back the robot intends to move to undock from the marker, and the blue area represents the undocking area

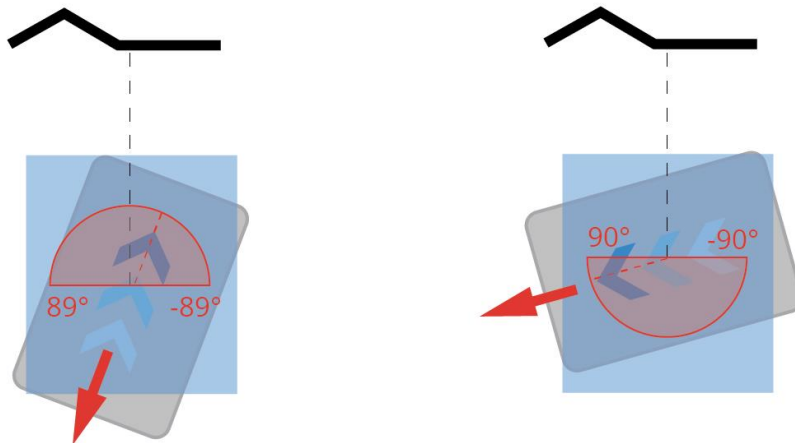


**Table 2.2** Identification of the dimensions in [Figure 2.4](#)

Pos.	Description	Pos.	Description
A	Always 100 mm. This is the distance in front of the robot's center when docked to the marker. This ensures that even if the robot is pushed forward a bit, it will still undock when leaving the marker.	B	The distance entered under <b>Docked at marker reverse distance</b> . This is the distance from the center of the robot when it is docked to the edge of the undocking area furthest from the marker.
C	The distance entered under <b>Docked at marker side threshold</b> . This is the distance to either side of the center line from the robot's docked position.	D	Always 100 mm. To make sure the robot's center has exited the undocking area, the robot aims to move back an additional 100 mm.

The robot will reverse from the marker as long as its orientation to the marker is below  $90^\circ$ . If the orientation difference is greater than  $90^\circ$ , the robot will move forward to exit the undocking area instead. This means that if the robot reverses into the area or is perpendicular to the marker, it will undock by moving forward.

**Figure 2.5** Robots that stop inside the undocking area at an angle less than  $90^\circ$  to the marker undock by driving backward. Robots that stop with an angle greater than  $90^\circ$  undock by driving forward



L-markers are the only markers that do not support automatic undocking. If you want a robot to undock from an L-marker, you must use a Relative move action.

The only case where the robot will not execute the undocking sequence is if you use a Relative move action right after the Dock action. The robot will always execute the Relative move action first, and if the robot is outside of the undocking area after finishing the Relative move action, it will continue to the next action without undocking.

### Marker accuracy

If you have more than one robot, you may experience that robots do not all dock to markers with the same accuracy.

Robots will dock with a high precision, meaning that the same robot will dock to almost the same position to the same marker.

If you want to increase precision, ensure that you are using Slow docking—see "[Fast docking and Slow docking](#)" on page 57.

If you need to ensure that all robots dock to all markers with the same accuracy, you can use the docking parameters to adjust the robot's final docking positions. There are two sets of docking parameters you can modify:

- The marker's docking parameters. Adjust these to modify how all robots dock to a specific marker.
- The robot's global docking parameters. Adjust these to modify how a specific robot docks to all markers of a specific type.

It is also possible to create multiple markers, one for each robot, for the same physical entity, for example, a Charging station, and then adjust the markers' docking parameters until all the robots dock with the desired accuracy.



For more information about the various docking parameters, see *How to change docking parameters*. You can find this guide on [MiR Support Portal](#).

### 2.12.5 Zones

Zones mark an entire area where you want to influence how the robot behaves or plans its paths.

For more information about creating zones, see "[Create zones](#)" on page 416.

For zones to affect robots, the zone must exist before the robot plans its paths. If a zone is created while the robot is driving between two points, the zone is ignored.

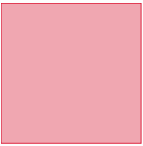
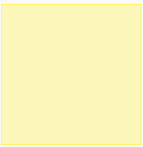
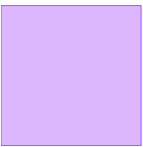
#### Action zones


Action zones are zones that have an affect on robots' behavior **when the robot's center is inside the zone**.

Use these zones to:




- Alert personnel of the robot with Sound and light zones in areas with limited visibility—see "[Operating hazard zones](#)" on page 440.
- Reduce the speed of the robot in operating hazard zones—see "[Operating hazard zones](#)" on page 440.
- Reduce the risk of robots blocking each other with Limit-robot zones—see "[Manage traffic](#)" on page 411.

- Change the robot's driving behavior with Planner zones.
- Evacuate robots from areas during an emergency with Evacuation zones—see "[Define emergency plans](#)" on page 466.
- Enable communication to external devices with Integration zones.

Graphic	Description
	<p><b>Speed zone</b></p> <p>A robot in a Speed zone tries to drive at the speed selected for the zone. Use a reduced speed if the robot is driving in a zone with many people, and an increased speed to quickly drive through a clear area.</p> <p>This is an action zone and only affects the robot when the robot is inside the zone.</p>
	<p><b>Sound and light zone</b></p> <p>A robot in a Sound and Light zone can play a sound and blink its status lights when driving in this zone. This zone can be used to warn people about the presence of the robot.</p> <p>This is an action zone, and only affects the robot when it is inside the zone.</p>
	<p><b>Planner zone</b></p> <p>A robot in a Planner zone can drive with different planner settings. You can choose between:</p> <ul style="list-style-type: none"> <li>• <b>No-localization:</b> Ignore data from the laser scanners and localize with encoders and IMU only. Use this in repetitive or featureless areas where the scanner data does not help the robot localize, for example long straight corridors.</li> <li>• <b>Look-ahead:</b> Decrease the field of view to prevent the robots from reacting to obstacles in the distance. Use this in areas with many dynamic obstacles.</li> </ul>

Graphic	Description
	<ul style="list-style-type: none"><li>• <b>Wait for obstacle:</b> Change how long robots remains stationary before planning a new global path. Use this with Path deviation to configure how strictly robots must follow their original path.</li><li>• <b>Path deviation:</b> Change how far robots can deviate from its path. Use this with Wait for obstacle to configure how strictly robots must follow their original path.</li><li>• <b>Ignore camera obstacle data:</b> Ignore data from cameras where robots detect non-existent camera data. Use this in areas with lighting conditions that interfere with the camera data.</li><li>• <b>Obstacle history clearing:</b> Change how robots remembers camera obstacle data. Use this in areas with dynamic obstacles that affect how the robot replans its path.</li></ul> <p>This is an action zone, and only affects the robot when it is inside the zone.</p>
	<h3 data-bbox="488 1106 711 1137">Integration zone</h3> <p data-bbox="488 1173 919 1205">A robot in an Integration zone can:</p> <ul style="list-style-type: none"><li>• Trigger a Payload event in the Top Module API or the MiR Fleet Integration API when the robot enters or exits the zone.</li><li>• Activate a port on an I/O module when it enters the zone. When the robot exits the zone, the port is deactivated.</li><li>• Change a PLC register when it enters the zone. When the robot exits the zone, you can change the same or another PLC register to a chosen value. Registers 1 to 100 are reserved for integers and registers 101 to 200 are reserved for floating point numbers.</li></ul> <p data-bbox="488 1624 1337 1693">This is an action zone and only affects the robot when it is inside the zone.</p>



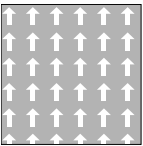

Graphic	Description
	<p><b>Limit-robots zone</b></p> <p>When a robot plans a path through a Limit-robots zone, it may only enter the zone if the number of robots inside the zone is less than the set limit. This zone only applies to robots controlled by MiR Fleet. If a robot cannot enter a zone because the limit has been reached, the robot waits outside the edge of the zone until another robot exits the zone.</p> <p>This is an action zone, and only affects when the robot may enter the zone.</p>
	<p><b>Evacuation zone</b></p> <p>A robot in an Evacuation zone will go to an Evacuation position if an evacuation is initiated. If there are not enough positions for all robots, all remaining robots will be sent to their nearest Evacuation positions and pause as close to them as possible.</p> <p>If an evacuation is triggered, all robots outside Evacuation zones are paused.</p> <p>This is an action zone, and only affects the robot when it is inside the zone.</p>
	<p><b>Lock zone</b></p> <p>When a robot plans a path through a Lock zone, it may only enter the zone if the zone is unlocked. If a zone is locked, the robot waits outside the edge of the zone until the zone unlocks. The robot will never try to plan around a locked zone. You can control the Lock state of the zone through the MiR Fleet interface or the MiR Fleet Integration API.</p> <p>If a robot is already assigned to the Lock zone when you lock it, the robot completes its movement through the zone. If another robot that has not been assigned the zone requests it, it must wait in the queue</p>




Graphic	Description
	<p>until the zone is unlocked.</p> <p>This is an action zone, and only affects when the robot may enter the zone.</p>

## Path planning zones

Path planning zones affect **how the robot plans its path**.

For guidelines on how to use zones for path planning, see "[Manage traffic](#)" on page 411.

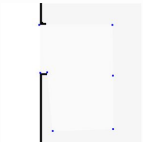
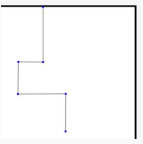
Graphic	Description
	<p><b>Directional zone</b></p> <p>When a robot plans a path through a Directional zone, the path is always <math>\pm 90^\circ</math> in the direction specified in the zone.</p> <p>For each Directional zone, you specify its direction, and the map shows the direction with arrows drawn on the zone.</p> <p>In Directional zone shapes, the same direction is applied across the whole shape. In Directional zone lines, the direction follows the direction of the line.</p> <p>All positions or markers inside or close to Directional zones must be oriented <math>\pm 90^\circ</math> to the zone direction so the robot can plan according to the zone.</p> <p>This is a path planning zone, and only effects how the robot plans its path.</p>
	<p><b>Preferred zone</b></p> <p>When a robot plans a path, it tries to go through Preferred zones as much as possible unless there is a significantly faster path. The robot</p>

Graphic	Description
	<p>can drive out of the zone to avoid obstacles.</p> <p>This is a path planning zone, and only effects how the robot plans its path.</p>
	<p><b>Unpreferred zone</b></p> <p>When a robot plans its path, it tries to avoid Unpreferred zones. Use this to guide the robot away from busy areas or routes it should avoid unless there are no other options. The robot only drives into Unpreferred zones if all other paths are blocked or is the path is significantly faster.</p> <p>If a robot does plan into an Unpreferred zone, it tries to minimize the distance where it travels in the zone.</p> <p>This is a path planning zone, and only effects how the robot plans its path.</p>
	<p><b>Forbidden zone</b></p> <p>When a robot plans a path, the robot ensures that its footprint never enters a Forbidden zone. Unless specified in the action parameters, robots never enter Forbidden zones.</p> <p>This is a path planning zone and affects how the robot plans its path and executes actions.</p>
	<p><b>Access zone</b></p> <p>When the robot plans a path, the robot ignores mapped obstacles that are in Access zones. This allows the robot to plan paths through areas that it otherwise would not. Use this zone to make robots plan up ramps and through doors that are marked as walls. If a robot detects an obstacle in the zone, it will wait for the obstacle to clear before continuing along the path.</p>

Graphic	Description
	This is a path planning zone, and only effects how the robot plans its path.

## Floor plan zones

Besides editing the floor plan by drawing on the floor plan—see ["Floor plan" on page 48](#)—, you can also add floors and walls as map components. This allows you to edit the vertices that define the shape of the floor or wall more easily. If you export the floor plan from the interface, any walls or floor zones are not included.

Graphic	Description
	<p><b>Floor zone</b></p> <p>When mapping, the floor is created automatically. The floor on the map marks the areas where the robot is able to drive. You can use the Floor tool to touch up the existing floor if there are areas where the floor is missing.</p>
	<p><b>Wall zone</b></p> <p>Walls mark areas on the map where there are permanent physical obstacles the robot cannot drive through. You can add straight lines to create a more legible map and decide how thick the walls should be by editing the stroke width.</p>

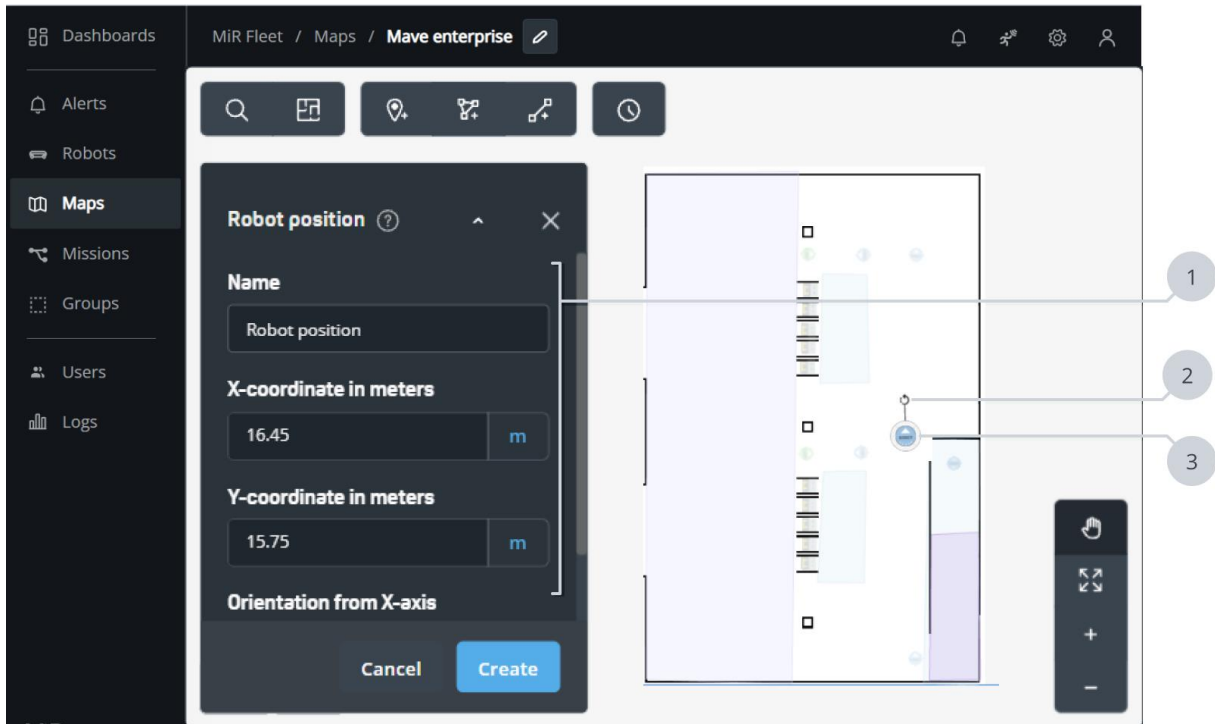
### 2.12.6 Map component tools

There are three tools you can use to add map components:

- **Add position** is used to add either a marker or a position to the map—see ["Add position" below](#).
- **Draw a new shape** is used to add a zone that covers a large area to the map—see ["Draw a new shape" on the next page](#). With this tool, you define the vertices of the shape where the zone is applied.
- **Draw a line** is used to add a zone in the form of a path—see ["Draw a line" on page 72](#). With this tool, you define consecutive points and the width of the line between the points where the zone is applied.

### Add position

To add a new position, you must first select which position or marker type you want to create—see ["Markers" on page 52](#) and ["Positions" on page 50](#). The chosen marker or position is then added to the map viewer and you can reposition it where you want it on the map.



Description	Description
1 <b>Position settings</b>	2 <b>Rotate position</b>

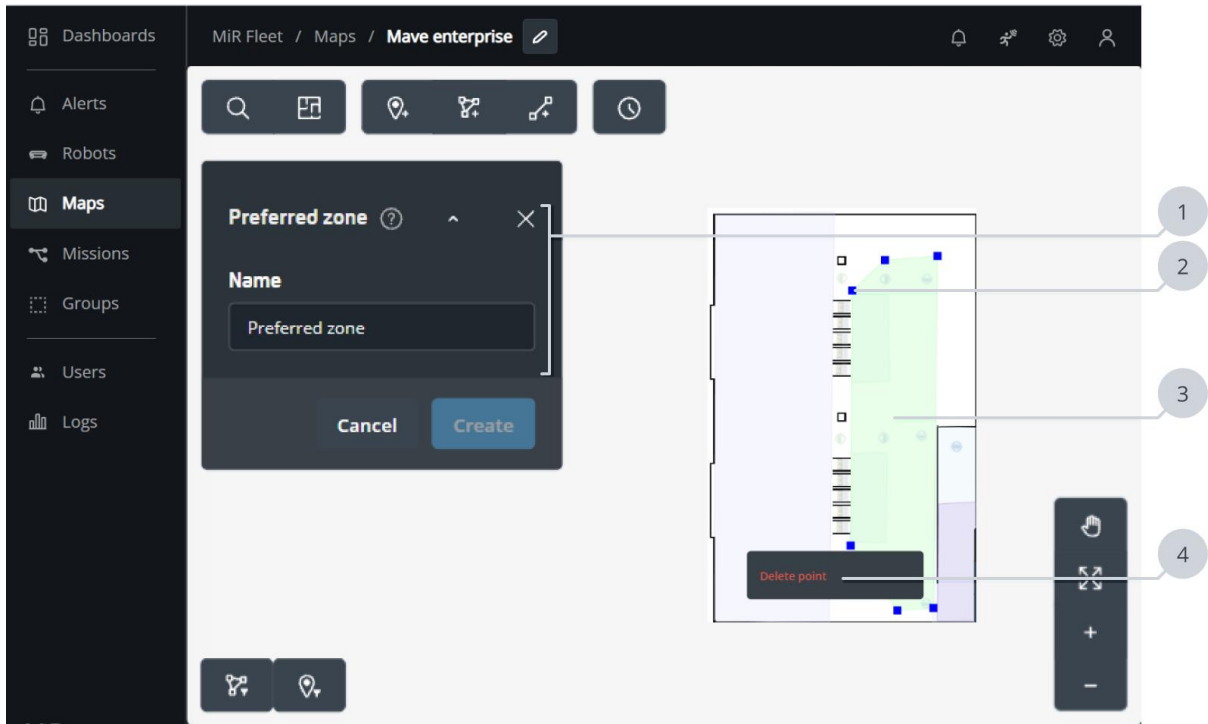
Description	Description
<p>In the position settings you can adjust where the position is on the map and its orientation.</p> <p>If you are creating a marker, you can also adjust the offsets to modify where the robot positions itself relative to the marker.</p> <p>If you have opened the active map and your robot is close to the marker, you can select <b>Detect</b> to make the robot automatically fill out the values so that when it docks to the marker, it ends up in the position it is currently in, unless it is a charging station or pallet rack marker.</p>	<p>Pull the rotate tool around to change the orientation of the position.</p> <p>If you change the orientation numerically in the position settings, the orientation of the position in the map viewer will update.</p>
<p>3 <b>New position</b></p> <p>The new position is shown on the map. Pull the position around on the map to reposition it.</p> <p>If you change the position coordinates numerically in the position settings, the location of the position in the map viewer will update.</p>	

### Draw a new shape

To add a new shape, you must first select which type of zone you want to create—see ["Zones" on page 62](#).

To draw a shape on the map, select a starting point for the shape and continue adding points to create the shape you want. The interface will automatically generate a shape with an edge that passes through each point you make. While creating the shape, you can:

- Drag, add, and delete points to modify the line.
- Adjust the zone settings specific to your zone.
- Rearrange which pairs of points an edge is generated between.



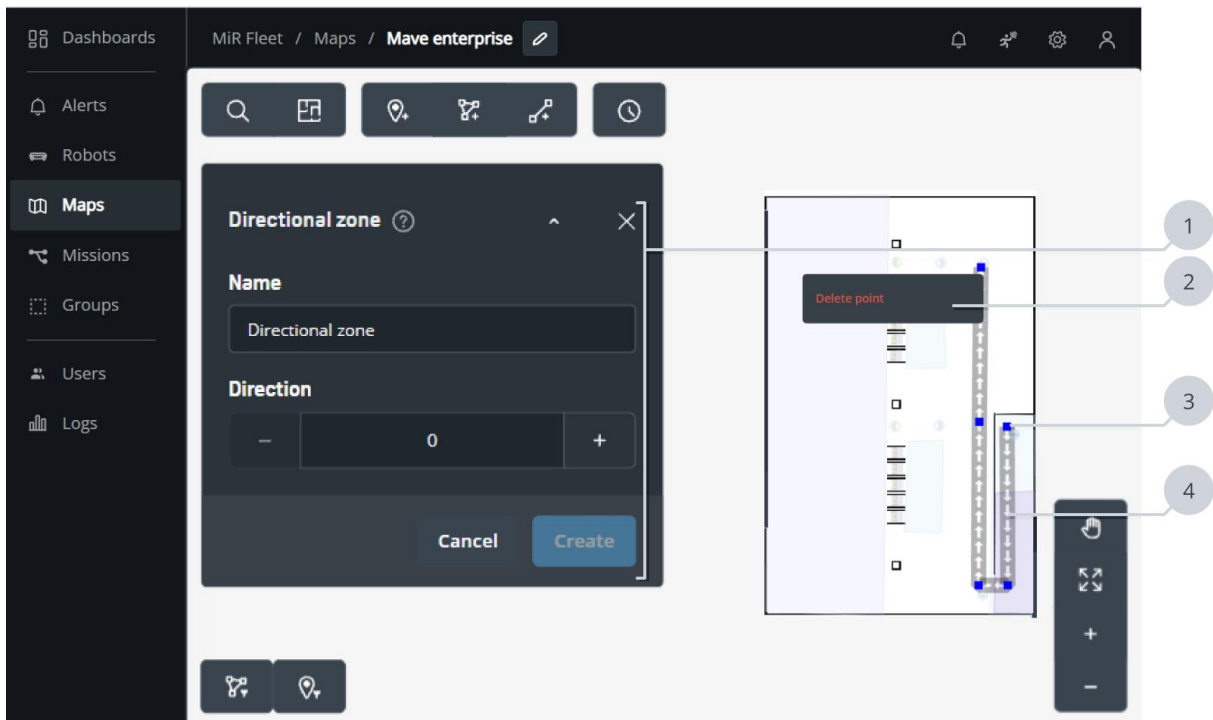
Description	Description
<p>1 <b>Zone settings</b></p> <p>In the zone settings, you can:</p> <ul style="list-style-type: none"> <li>• Enter the name of the zone.</li> <li>• Modify any zone specific settings.</li> </ul>	<p>2 <b>Point</b></p> <p>You can drag points around to modify the shape. If you select between points, a new point is generated.</p>
<p>3 <b>Shape area</b></p> <p>The area inside all of the created points is shaded with the selected zone color to indicate the area the zone will cover.</p>	<p>4 <b>Delete point</b></p> <p>When you select a point, the option to delete the point appears.</p>

### Draw a line

To add a new line, select which type of Path planning zone you want to create—see "[Path planning zones](#)" on page 66.

To draw the line on the map, select a starting point for the line and continue adding points to create the line you want. The interface will automatically draw a line to the next point you create. While creating the line, you can:

- Drag, add, and delete points to modify the line.
- Adjust the line width.
- Adjust the zone settings specific to your zone.
- Rearrange between which pairs of points an edge is generated.



Description	Description
<p>1 <b>Zone settings</b></p> <p>In the zone settings, you can:</p>	<p>2 <b>Line</b></p> <p>A line is automatically generated</p>

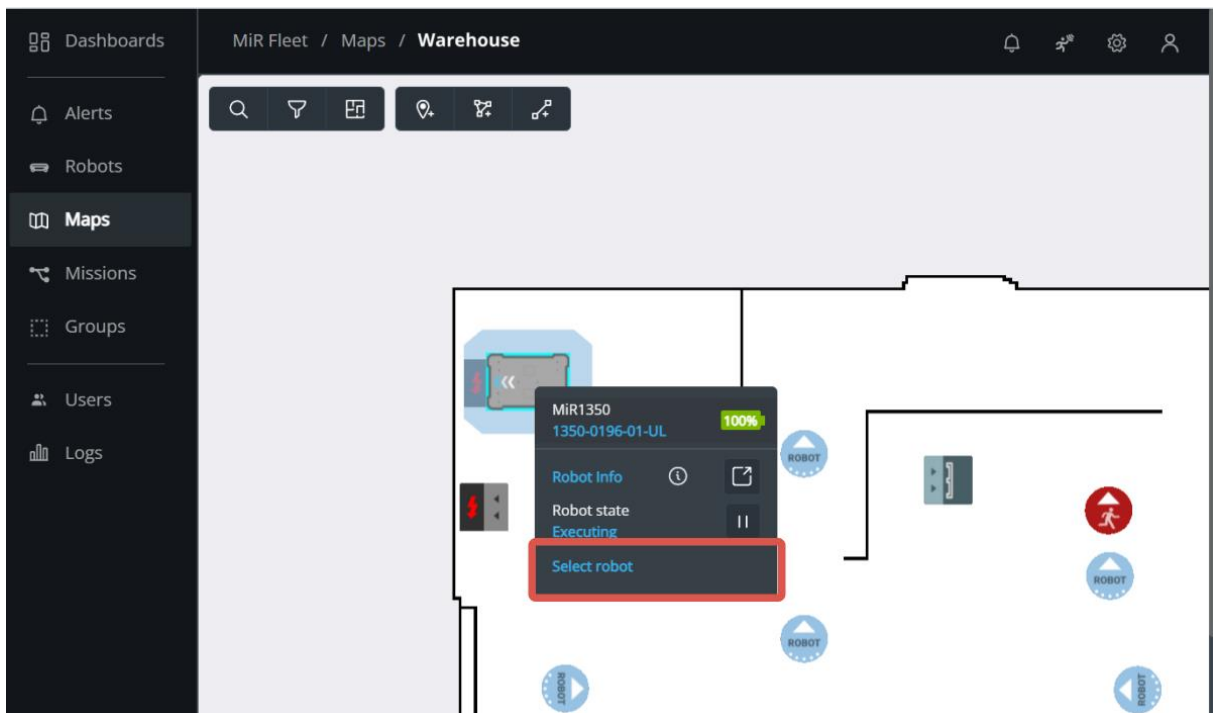


Description	Description
<ul style="list-style-type: none"> <li>• Enter the name of the zone.</li> <li>• Adjust the line width.</li> <li>• Modify any zone specific settings.</li> </ul>	<p>between every pair of consecutively created points.</p>
<p>3 <b>Point</b></p> <p>You can drag points around to modify the shape. To create a new point, select an empty area without any existing points.</p>	<p>4 <b>Delete point</b></p> <p>When you select a point, the option to delete the point appears.</p>






### 2.12.7 Select robot

When you are viewing a map with a connected robot, you can select the robot to enable robot data to be propagated to map view and to control the robot.

To select a robot, open the robot's context menu and **Select robot**.



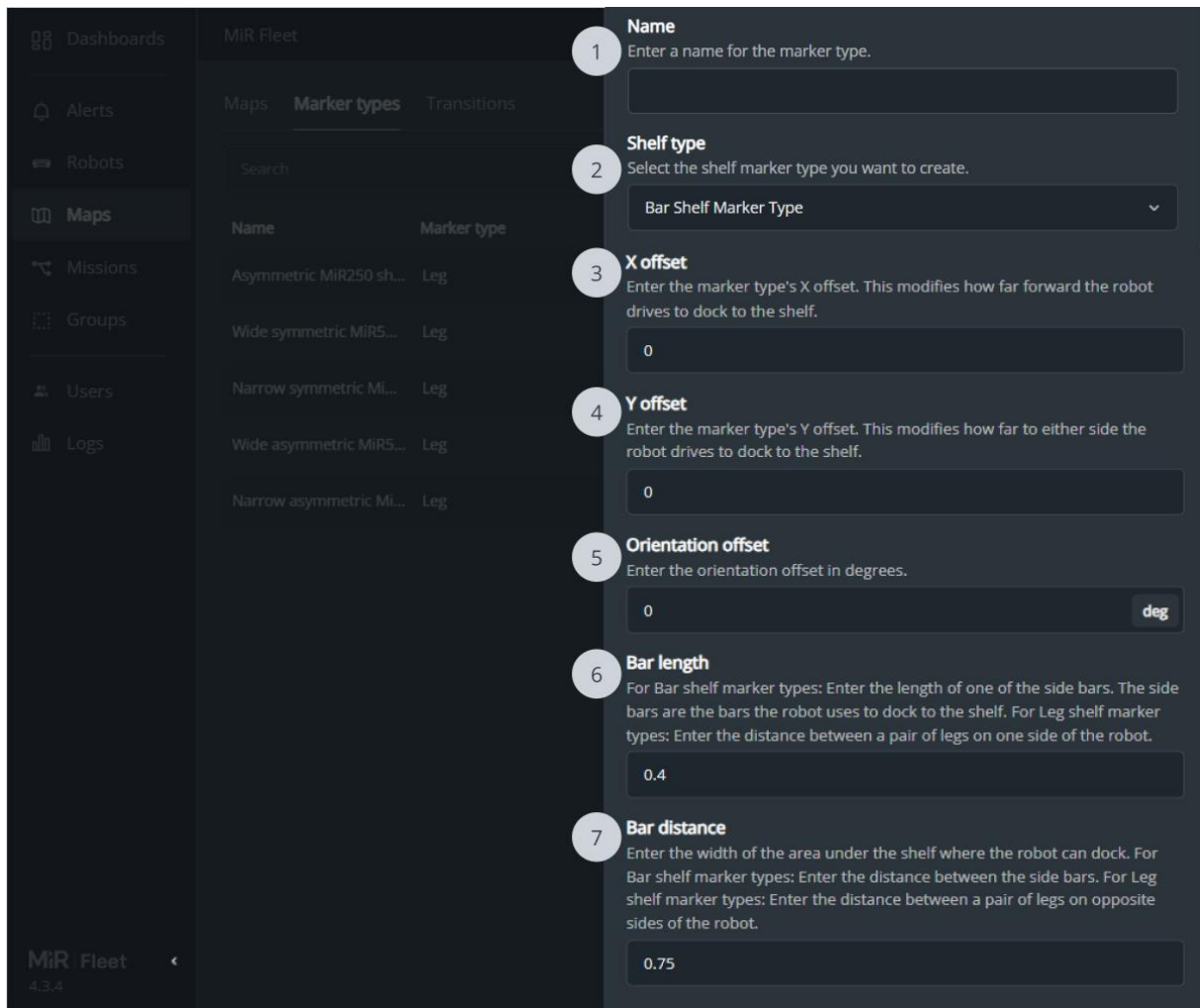
When the robot is selected you can use tools to control the robot or toggle data visibility:

- **Keep robot in center**    
Makes the map automatically follow the robot to keep the robot in the center of the map.
- **Send robot to target**    
Send the robot to any point on the map. MiR Fleet assigns a mission to the robot to drive to the chosen point. You can choose to queue the mission after the current mission, or to clear the current mission queue to make the robot go immediately.
- **Scanner data**    
Displays or hides the selected robot's laser scanner data. The data is represented as red dots.
- **Obstacle cloud**    
Displays or hides the selected robot's obstacle cloud. The data is a combination of all sensor data when the robot is moving. The data is represented with blue clouds and is only shown when the robot is moving.
- **Path**    
Displays or hides the selected robot's planned path. The path is represented as consecutive blue dots.

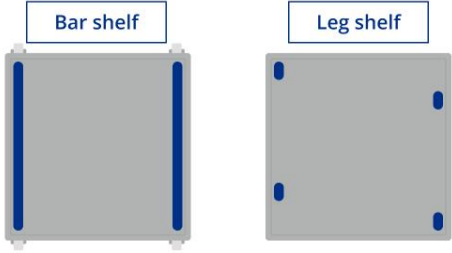



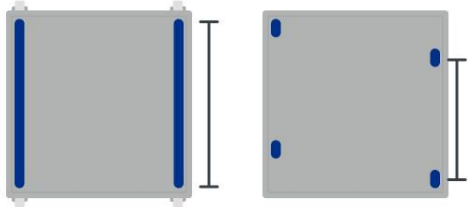
## 2.13 Marker types

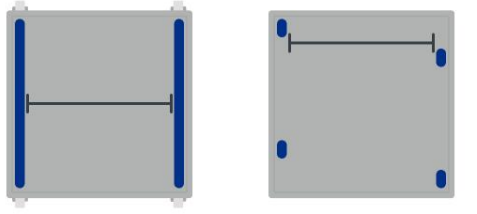
A marker type is a description of a shelf that MiR robots can dock to. You must have a marker type for each type and size of shelf you want your robot to be able to transport.

A marker type is used to provide the information MiR robots need to dock to a shelf correctly. When you send the robot to a Shelf position to pick up a shelf, you must also select the marker type corresponding to the shelf.



Description		Description	
1	<b>Name</b> Enter a name for the marker type.	2	<b>Shelf type</b> Select the shelf marker type you want to create.  Bar-markers are for MiR100 and MiR200.  Leg-markers are for MiR250, MiR500, MiR600, MiR1000, and MiR1350.

Description	Description
	 <p>The image shows two diagrams of marker types. On the left is a 'Bar shelf' marker, represented as a grey rectangle with two vertical blue bars on the left and right sides. On the right is a 'Leg shelf' marker, represented as a grey rectangle with four blue dots at the corners, representing legs.</p>
<p>3 <b>Orientation offset</b></p> <p>Enter the orientation offset in degrees.</p>  <p>The diagram shows a grey rectangle representing a robot. A blue arrow points upwards from the center. A curved double-headed arrow indicates a rotation from the vertical. The text '+ Orientation -' is written below the arrow.</p>	<p>4 <b>X-offset</b></p> <p>Enter the marker type's X-offset. This modifies how far forward the robot drives to dock to the shelf.</p>  <p>The diagram shows a grey rectangle representing a robot. A blue arrow points upwards from the center. A vertical line with a double-headed arrow indicates a distance from the center to the top edge. The text '+ X-offset -' is written next to the arrow.</p>
<p>5 <b>Y-offset</b></p> <p>Enter the marker type's Y-offset. This modifies how far to either side the robot drives to dock to the shelf.</p>  <p>The diagram shows a grey rectangle representing a robot. A blue arrow points upwards from the center. A horizontal line with a double-headed arrow indicates a distance from the center to the left and right edges. The text '+ Y-offset -' is written above the arrow.</p>	<p>6 <b>Bar length</b></p> <p>For Bar shelf markers: Enter the length of one of the side bars. The side bars are to the left and right sides of the robot.</p> <p>For Leg shelf markers: Enter the distance between a pair of legs on one side of the robot.</p>  <p>The image shows two diagrams. On the left is a 'Bar shelf' marker with two vertical blue bars. A vertical line with a double-headed arrow indicates the length of one bar. On the right is a 'Leg shelf' marker with four blue dots. A vertical line with a double-headed arrow indicates the distance between a pair of legs on one side.</p>

Description	Description
<p>7 <b>Bar distance</b></p> <p>Enter the width of the area under the shelf where the robot can dock.</p> <p>For Bar shelf markers: Enter the distance between the side bars. The side bars are to the left and right sides of the robot.</p> <p>For Leg shelf markers: Enter the distance between a pair of legs on opposite sides of the robot.</p> 	

## 2.14 Missions

A mission is a user-defined series of actions the robot can be set to perform on demand. A mission can be a simple transportation task between defined positions or a more complex job that includes both moving between positions and performing actions, such as unloading a pallet, moving to a charging station when the battery is low, or sending an email on arrival at a position.

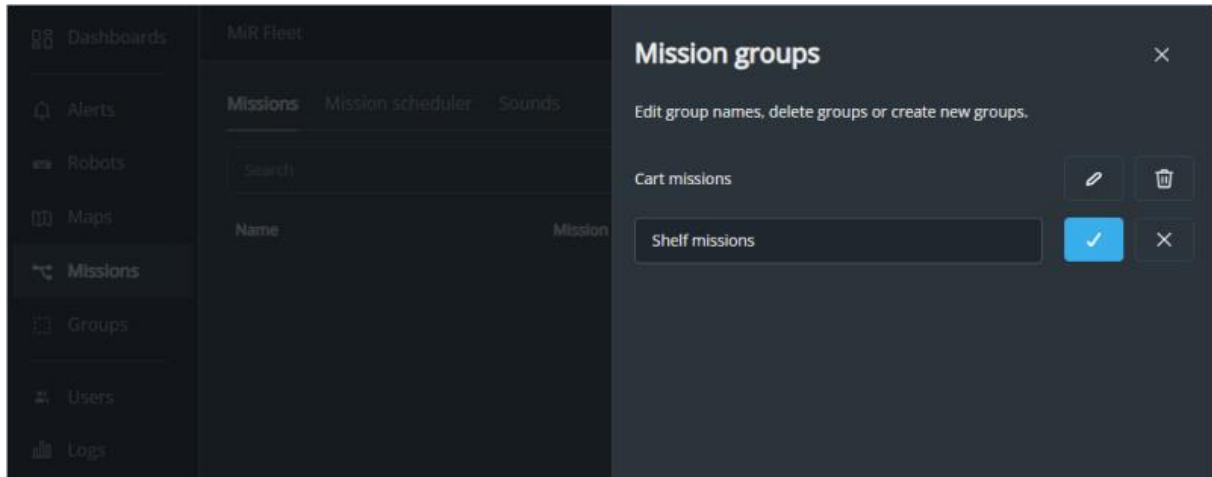
You start a mission by adding the mission to the mission queue. Missions will be run in the order they are added, or you can rearrange them as needed.

In MiR Fleet, missions are controlled in **Mission scheduler**.

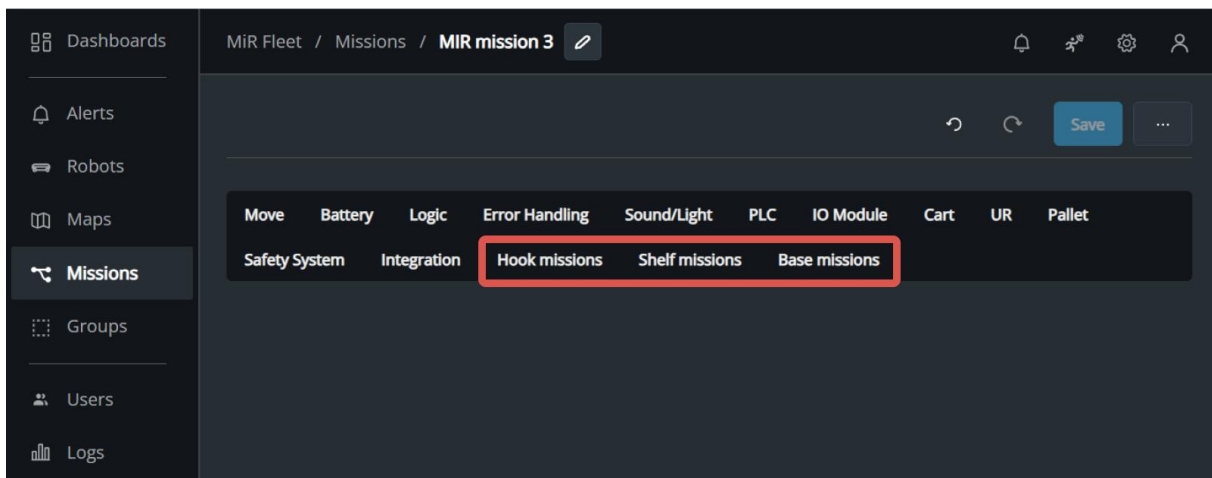
For guidelines on planning and creating missions, see ["Create missions and mission groups" on page 347](#).

### 2.14.1 Mission groups

Mission groups are used to organize your missions into various groups. You can use groups to define which mission groups contain missions the robot can execute—see ["Groups" on page 30](#).




In the mission editor, the mission groups are displayed at the top of the editor after the action groups. This lets you nest missions from the mission groups into the mission you are creating.

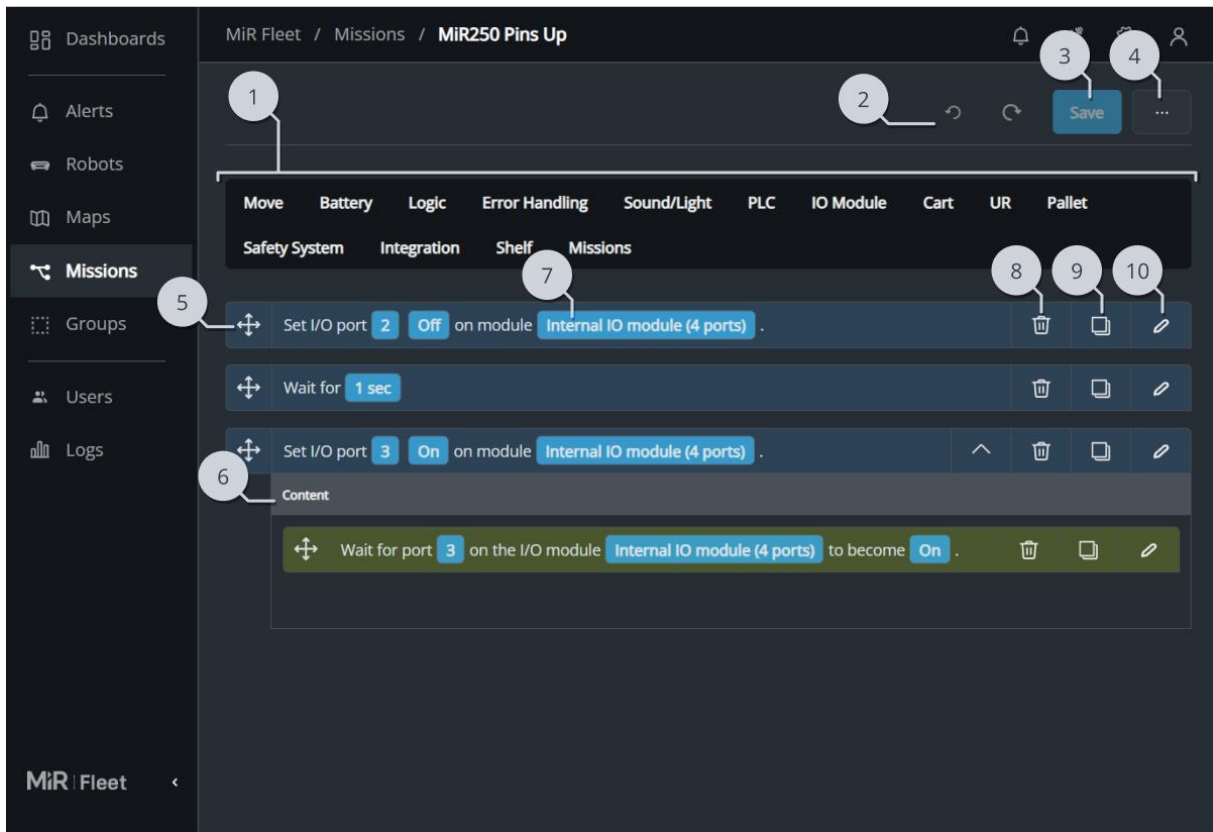


### 2.14.2 Edit mission

A mission is built from actions that you pick from the action and mission group menus in the top bar. You can also pick already created missions and nest them in new missions.

When you have picked the actions or missions you want in your mission, do the following:

- Drag the actions to sort them in the desired order. The actions are executed in a top-to-bottom order.
- Set the parameters for the selected action by selecting **Edit**  at the far right of the action line.



Description	Description
<p>1 <b>Action menus</b></p> <p>All of the actions and missions you can add to your mission are divided into the action menus at the top of the editor. These action menus correspond to the mission groups you have created for the site.</p>	<p>2 <b>Undo and Redo</b></p> <p>Undo the last change you applied. If you regret an Undo, use Redo to reapply the change.</p>
<p>3 <b>Save</b></p>	<p>4 <b>Mission settings</b></p>

Description	Description
Save the item you are editing.	Allows you to: <ul style="list-style-type: none"> <li>• Edit the name of the mission and which mission group and site the mission is part of.</li> <li>• Save a copy of the mission.</li> <li>• Delete the mission.</li> </ul>
5 <b>Added action or mission</b>  When you add an action or mission from the top menus, it is displayed in the editor below. Here, you can reorder and edit each mission or action. The mission is always executed from top to bottom, meaning that when the mission is run, it starts with the action at the top, and once the robot has completed that action, it continues to the next action below.	6 <b>Scope</b>  Some actions have a scope where you can add other actions. To add an action to an action scope, either drag and drop the action into the scope, or select the scope and then add the action from the action menu.
7 <b>Entered parameter</b>  The most important custom parameters are shown in the overview in blue. When you change the parameter value, the value in the overview will also update. If you assign a variable to the parameter, it is displayed in purple.	8 <b>Delete</b>  Deletes the action or mission from the current mission.
9 <b>Copy</b>  Copies the action or mission below the existing instance.	10 <b>Edit</b>  Opens the action or mission editor dialog box.



### 2.14.3 Arguments

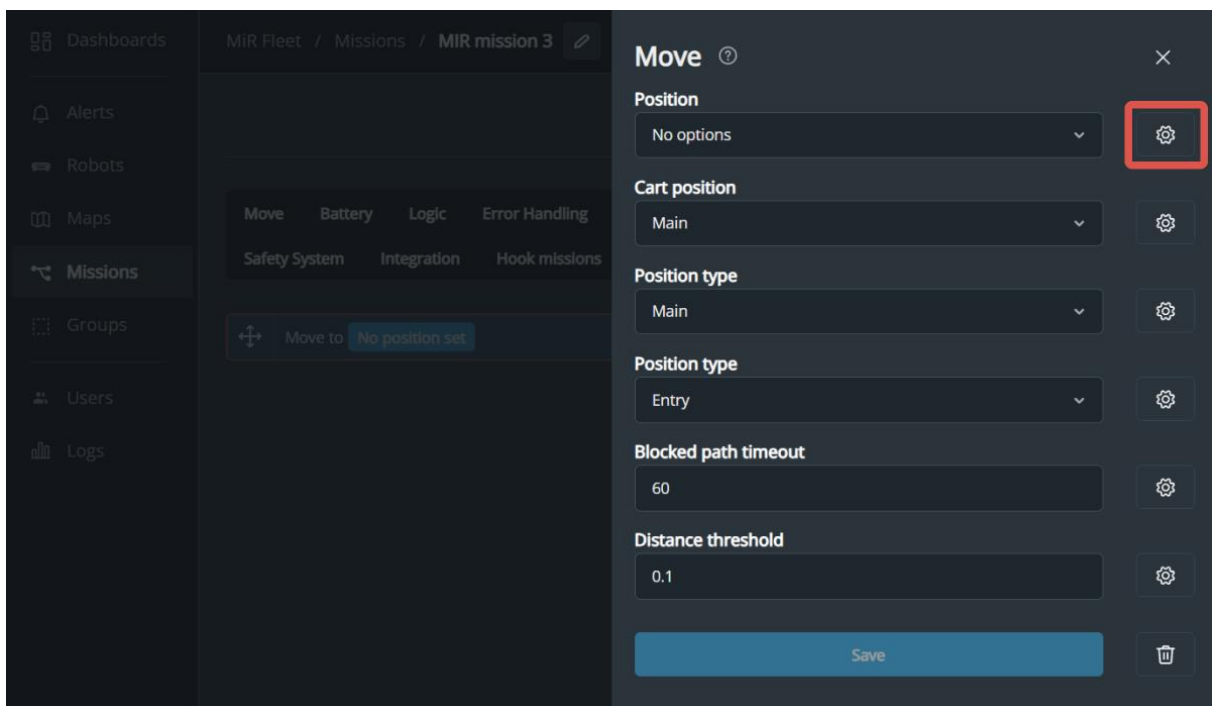
An argument is a flexible value that can be set each time you run a mission or nest the mission in another mission. Use arguments when you want to reuse the same mission for similar robot tasks.

To use an argument for this action parameter, enable **Use an argument**, name the argument, then select a default value.

If there are any existing arguments in this mission that are used for the same parameter type, they are listed under **Available arguments**. If you use the same argument name, the same value is passed to the arguments used in both instances.

When you schedule or nest a mission with an argument, you will be prompted to enter a value for the argument.

If you have already defined an argument, you can reuse the argument in other parameters that accept the same argument. All existing arguments within the same mission and for the same parameter type are available for reuse. For each available argument, you can also see the other actions where the argument is used.



### 2.14.4 Mission action limiter

If you experience that a mission is executed slower than expected, the action limiter may have been activated.

If the robot executes many actions within a short time frame, it can use a significant amount of the robot's processing power. This can have a negative effect on the robot's other automatic systems. To avoid this, the robot introduces an action limit if necessary.

If more than 100 actions have been performed in the last 10 seconds, the robot begins limiting the number of actions to two actions per second. It will keep this limit until less than four actions are executed in the last 10 seconds. Starting a new mission does not reset the action limit count or remove the two actions per second limit.

## 2.15 Mission scheduler

Missions—see ["Missions" on page 77](#)—are scheduled and monitored from the Mission scheduler menu.

The Mission scheduler is a list of missions that MiR Fleet will assign to connected robots. When MiR Fleet assigns a mission, it considers:

- Which robot is part of the same group as the mission—see ["Groups" on page 30](#).
- Which robot is closest to the start of the mission.
- The priority of the mission.
- The scheduled time for the mission.

MiR Fleet uses the scheduler to assign Auto charging and Auto staging missions to robots that meet the criteria to be sent to a charging station or Staging position—see ["Auto Charging and Auto Staging" on page 13](#). These missions will not have a priority or ID since their behavior is predefined.


You can add missions to the Mission scheduler. When you schedule a mission:

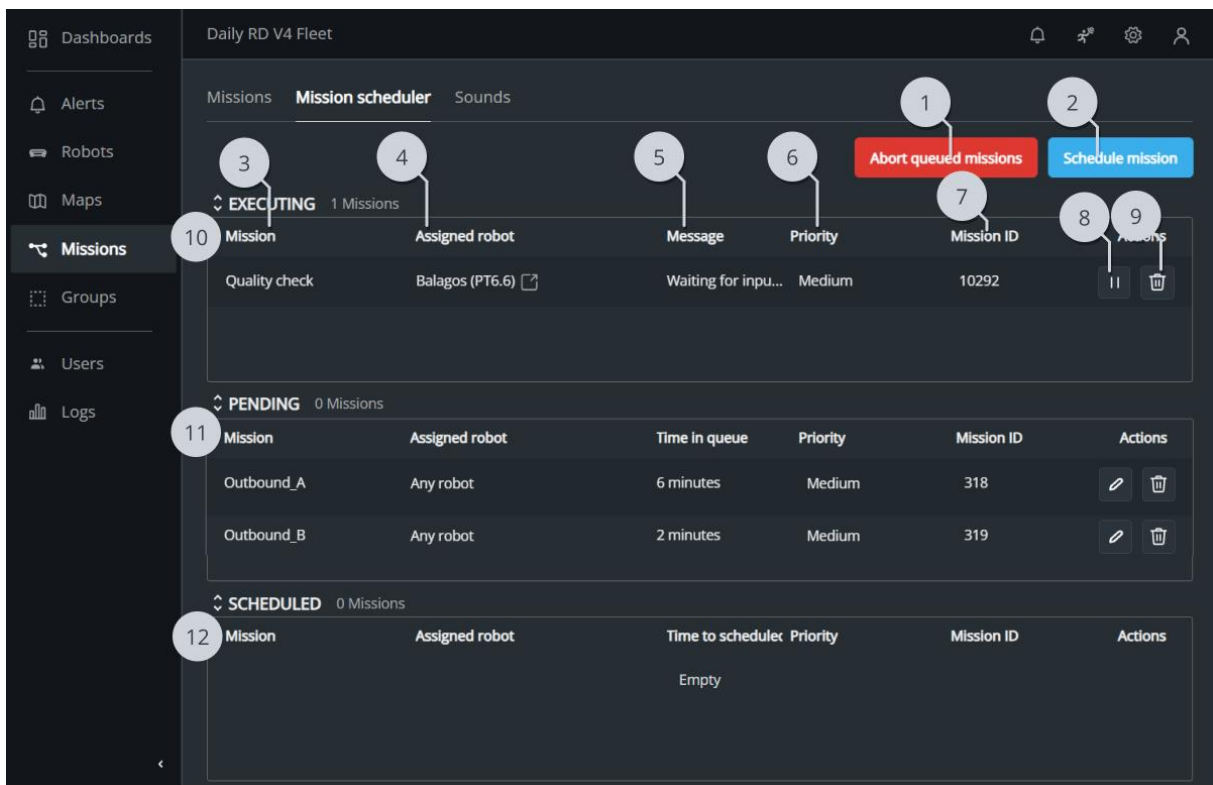
- Set the values of the arguments in the mission.
- Set the start time of the mission.
- Set the priority of the mission.

Use orders in the MiR Fleet Integration API—see ["Serial-orders" on page 521](#)—to schedule missions and assign arguments in the same way. If you use the API, you can set up a system that enables recurring missions and missions that are triggered by events automatically.

Missions that you schedule or have scheduled will fall into one of the following categories:

- **Aborted**  
Scheduled missions that were canceled or aborted or the robot failed to complete.
- **Pending**  
Missions that you want to be executed as soon as a suitable robot is available.
- **Scheduled**  
Missions that have been scheduled for a later time.
- **Done**  
Missions that have been completed successfully.
- **Executing**  
Missions that are currently being executed by a robot connected to MiR Fleet.

Missions are displayed under the category where they belong. If a mission has not yet been executed, you can remove it from the scheduler by selecting **Delete** .



Description		Description	
1	<p><b>Cancel queued missions</b></p> <p>Cancel all Pending and Scheduled missions.</p>	2	<p><b>Schedule mission</b></p> <p>Opens the scheduler dialog box—see <a href="#">"Schedule mission" on the next page.</a></p>
3	<p><b>Mission name</b></p> <p>The name of the mission.</p>	4	<p><b>Assigned robot</b></p> <p>The robot that the mission is assigned to.</p>
5	<p><b>Current task</b></p> <p>The current action the robot is executing from the mission.</p>	6	<p><b>Priority</b></p> <p>The priority of the mission. Missions with higher priority are executed before missions with a lower priority. You can rearrange the order of scheduled and pending missions to change their priority.</p>
7	<p><b>Mission ID</b></p> <p>Each mission has an ID. You must use the ID when scheduling missions with MiR Fleet Integration API.</p>	8	<p><b>Pause/Continue</b></p> <p>Pause or continue the mission execution.</p>
9	<p><b>Abort/Cancel</b></p> <p>If the mission is being executed, MiR Fleet sends an abort request to the robot. When the robot receives the request, it aborts the mission it is currently executing and sends a confirmation back to MiR Fleet.</p> <p>If the mission is not assigned to a robot yet, MiR Fleet cancels the mission and removes it from the queue.</p>	10	<p><b>Executing missions</b></p> <p>Lists all missions that are currently being executed by a robot on MiR Fleet.</p>

	Description		Description
11	<p><b>Pending missions</b></p> <p>Lists all missions that are ready to be assigned to the next available robot. Missions can be rearranged to change their priority.</p>	12	<p><b>Scheduled missions</b></p> <p>Lists all missions that have been scheduled for a later start time.</p>

### 2.15.1 Mission order

Missions can only be assigned to robots when they are in the Pending missions list. The mission's earliest start time determines when a mission moves from the Scheduled mission list to the Pending missions list.

When you schedule a mission, there are two options for the start time setting:

- Enter a specific date and time to add the mission to Pending missions.
- Select **Run as soon as possible** to add the mission to Pending missions immediately.

Pending missions are assigned to robots in the order they are displayed in the MiR Fleet interface. The list is ordered based on priority and order the length of time in the queue. By default, new missions are added after any other pending missions of the same priority.

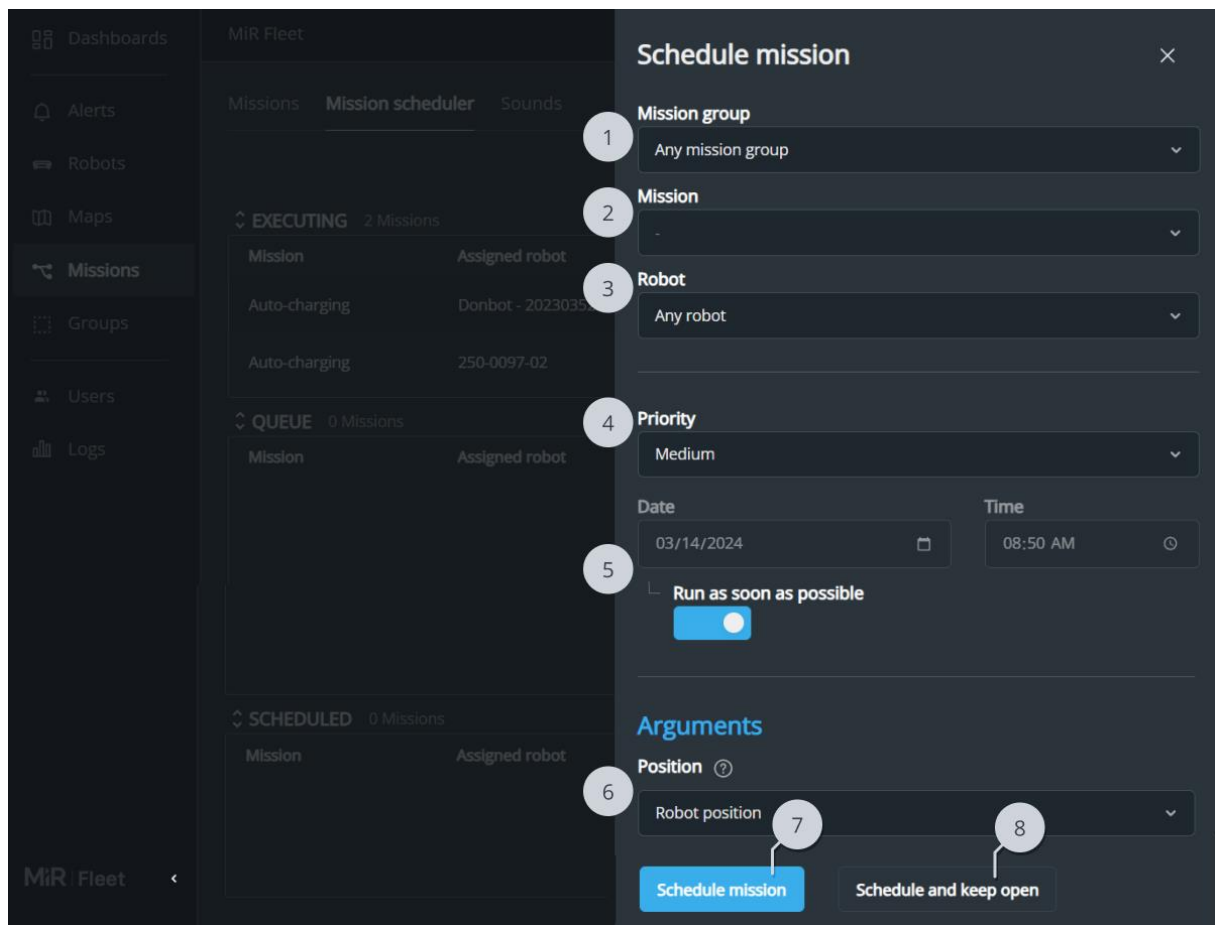
You can change the mission order by dragging the pending missions. This automatically changes the priority to match the new position you have placed the mission in the queue.

When a robot finishes a mission, aborts a mission, or a new mission is scheduled, MiR Fleet evaluates which of the pending missions can be assigned to available and compatible robots.

### 2.15.2 Schedule mission

To schedule a mission, select a mission, enter the time and date you want the mission to start, select which robot should execute the mission, and select a priority. If the selected mission uses any arguments, select or enter a value for each argument. Then select **Schedule mission**.

Select any of the missions from the list that you want to schedule. You can filter the missions based on mission groups.



Description	Description
<p>1 <b>Mission group</b></p> <p>Select a mission group to filter the available missions.</p>	<p>2 <b>Mission</b></p> <p>Select the mission you want to schedule.</p>
<p>3 <b>Robot</b></p> <p>Select the robot you want to execute the mission.</p>	<p>4 <b>Priority</b></p> <p>Set the mission priority. Missions with higher priority are listed higher in the Pending missions list.</p>
<p>5 <b>Scheduling options</b></p>	<p>6 <b>Arguments</b></p>

Description	Description
<ul style="list-style-type: none"> <li>• Enter a specific date and time to add the mission to the list of queued missions.</li> <li>• Select <b>Run as soon as possible</b> to add the mission to the Pending mission list immediately.</li> </ul>	<p>If the mission uses any arguments, you are prompted to select values for the arguments.</p>
<p>7 <b>Schedule mission</b></p> <p>Schedule the mission and close the Schedule mission dialogue box.</p>	<p>8 <b>Schedule and keep open</b></p> <p>Schedule the mission and keep the Schedule mission dialogue box open to schedule another mission.</p>

## 2.16 Resource handling

Resources in MiR Fleet is an umbrella term for map features that can only be occupied by a limited number of robots and where it is the responsibility of MiR Fleet to manage which robot has priority.

All types of positions and markers, Limit-robots zones, and Lock zones are resources in MiR Fleet.

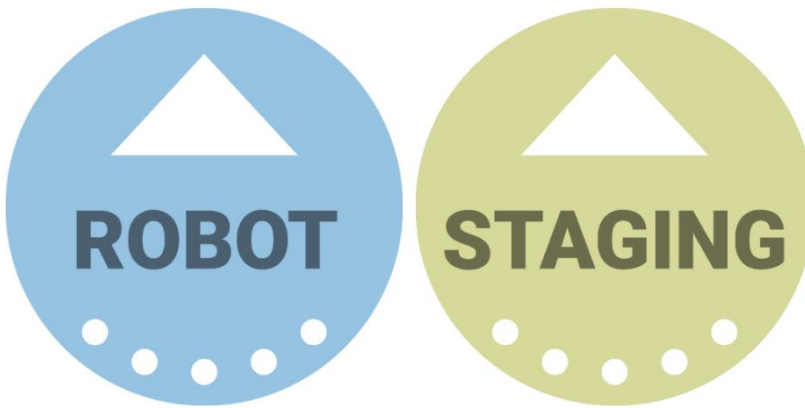
MiR Fleet handles the resources shared between the robots connected to the fleet. For each resource, MiR Fleet manages:

- When the resource is available or occupied.
- Which robot is currently occupying the resource.
- Which robots are in queue to occupy the resource as soon as it is available.

There is no limit to how many robots can be queued to a resource. You can see which robots are queued for each resource in the map viewer in the MiR Fleet interface. You can either open the resource queue for the whole map—see ["Maps" on page 45](#)—or select a specific resource to see the queue for the selected resource alone.

### 2.16.1 Positions

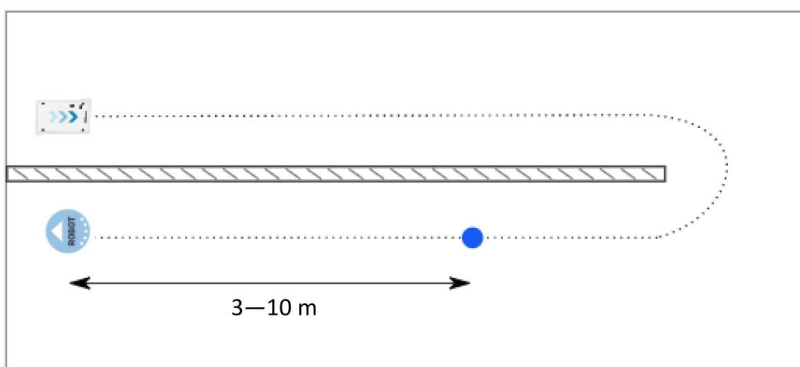
A position can only be assigned to one robot at any given time. However, there is no limit to how many robots can be queued to a position.



The robot can be assigned to a position if it fulfills the following requirements:

- The position is not assigned to another robot.
- The position is the goal of the robot's current driving action.
- The position is within the Resource assignment distance of the robot's center along its global path (meaning not through obstacles)—see "[Resource assignment distance](#)" on page 90.

**Figure 2.6** Robots can be assigned a resource as soon as they are 3–10 m from the resource (not through obstacles)

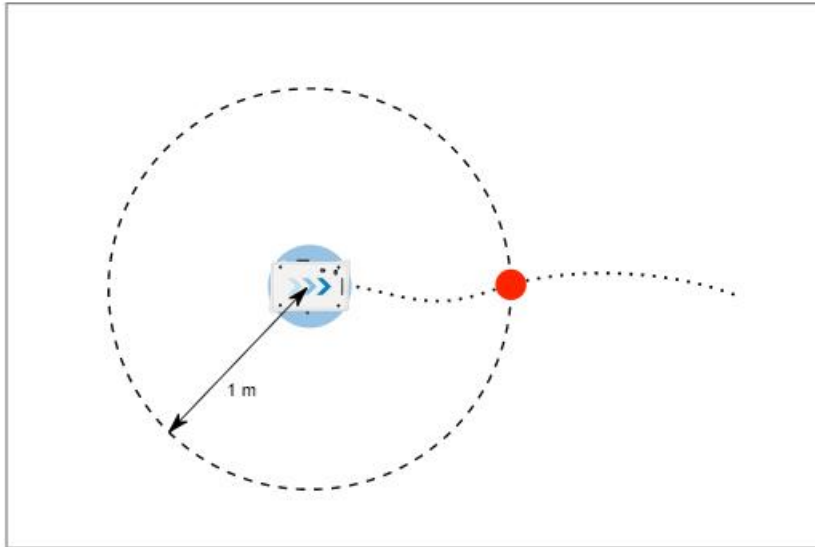


The robot releases a position if both of the following requirements are met:



- The robot is following a new plan where the goal is no longer the assigned position resource.
- The distance from the center of the position to the center of the robot is more than one meter.

**Figure 2.7** The robot releases the position when the robot's center is more than one meter away from the center of the position



### 2.16.2 Markers

Every marker has an Entry position that is linked to the marker. The robot must be assigned to the Entry position before being able to dock to the marker.

When the robot is docked to a marker, it continues to occupy the Entry position.

Entry positions are handled by MiR Fleet the same way as other positions. For the robot to dock successfully to a marker, the requirements for the marker's Entry position has to be met, as described in ["Positions" on the previous page](#).

**Figure 2.8** Charging station and VL-marker with Entry positions



When a robot docks to a marker or undocks from a marker, it first checks that it does not enter any occupied Limit-robots zones by doing so.

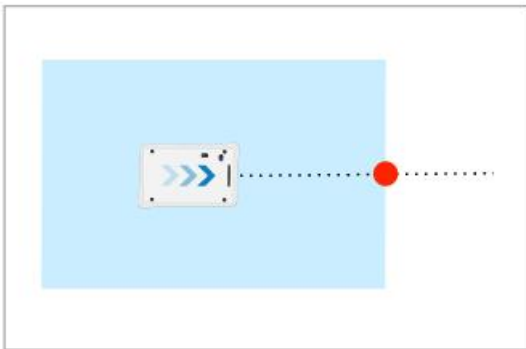
### 2.16.3 Zones

A robot can be assigned to a Limit-robots zone or Lock zone if the following requirements are fulfilled:

- The number of robots assigned to the Limit-robots zone is fewer than the maximum capacity of the zone.
- The Lock zone is not locked.
- The robot's path intersects the edge of the zone.

A robot releases a zone when the robot's center crosses the edge of the zone on its way out of the zone, making the resource available for other robots.

**Figure 2.9** The robot releases the zone when its center reaches the edge of the zone



### 2.16.4 Resource assignment distance

By default, robots are assigned a resource when the resource is 10 m from the robot along its current global path to the resource. You can reduce this value to as low as three meters, using the Resource assignment setting under **Settings** ⚙️. Reducing this value can resolve deadlocks in certain areas of your map where you use Limit-robots zones. Changing this setting modifies the behavior of all robots on MiR Fleet. We do not recommend modifying the Resource assignment distance without first revising if you can modify the Limit-robots zones to resolve issues.

If you want to change the value, consider how it may affect every resource and robots with various footprint sizes in your site. After you change the value, test that the robot behavior is as desired afterward.

## 2.17 Robot states and fleet behavior

This section describes the states MiR robots can be in when connected to MiR Fleet and how they behave in these cases.

### 2.17.1 Adding robot to MiR Fleet

When you add a robot to MiR Fleet, MiR Fleet goes through the following steps:

- 1 The system checks that the robot is compatible. The robot is compatible if:
  - The robot software version is compatible with MiR Fleet.
  - The given IP address is for a MiR robot.
  - The robot is not connected to another fleet.
  - The robot's mutual trust certificate is registered in MiR Fleet—see "[Mutual trust page](#)" on page 110.
- 2 If the robot is compatible, it is added to the fleet. When adding it, the existing site on the robot is replaced with the one on MiR Fleet. The initial synchronization takes a few seconds.
- 3 Once the site data has been received by the robot, and the robot has communicated its status info to MiR Fleet, the robot is considered active on the fleet.

If an error occurs during the process, an error is reported in the MiR Fleet interface.

### 2.17.2 Restart or shut down MiR Fleet

If the MiR Fleet host device or application shuts down or restarts, the following occurs:

- 1 All resources for all robots are released.
- 2 Robots in the middle of executing a mission continue the mission until it is completed or until the robot reaches a new resource.

- 3 When MiR Fleet starts up again, all robots are re-evaluated and resynchronized before being considered as active robots. Then they can resume their previous states.
- 4 All missions that were scheduled are still in the mission scheduler and will be assigned to robots accordingly.

### 2.17.3 Restart or shut down connected MiR robot

If you restart or shut down a robot connected to MiR Fleet, the robot is treated as an Unavailable robot by MiR Fleet. When the robot is turned on again, it automatically reconnects to MiR Fleet, provided it has not been deleted from the fleet.

### 2.17.4 Incompatible robots

Robots are incompatible if they are not running a software version where the data base structures match. Incompatible robots are connected to MiR Fleet but cannot do anything and are not considered in any MiR Fleet features. Once the robot software is updated and becomes compatible, the robot can be used like any other compatible robot in the fleet.

### 2.17.5 Unavailable robots

When MiR Fleet cannot communicate with a connected robot, the robot is considered Unavailable. In this state, MiR Fleet continues to try to re-establish connection with the robot. Until MiR Fleet receives a response from the robot, it is assumed that the robot is not receiving any fleet information. This means that Unavailable robots:

- Keep all resources they had when they entered Unavailable state.
- Cannot be assigned to new resources.
- Cannot be assigned new missions.
- Cannot be sent to chargers or staging positions.
- Cannot exchange Collision avoidance data. Any displayed data for an Unavailable robot is outdated.

- Continue executing missions they were assigned before losing connection.  
If the robot needs a resource that it was not assigned before losing connection, it waits until it either reconnects to MiR Fleet and is assigned the resource, or until five minutes without any communication with MiR Fleet passes and the robot is permitted to continue the mission without waiting to be assigned resources. If the communication is ever re-established, the robot will revert to waiting for fleet resources like all other active fleet robots.

When you turn off a robot connected to MiR Fleet, it becomes Unavailable. Before turning off a robot connected to MiR Fleet:

- Move the robot away from fleet resources to ensure the robot is not physically blocking other robots from using fleet resources.
- Delete the robot from MiR Fleet so that it releases all resources.

### 2.17.6 Deleted robots

When you delete a robot on MiR Fleet, you suspend that robot from all MiR Fleet features. When a robot is deleted:

- All resources that it occupied are released, and the robot is removed from all resource queues.
- The robot cannot occupy new resources, and the fleet will no longer try to assign any new resources to the robot.
- The robot no longer synchronizes information with MiR Fleet.
- The robot cannot receive any new missions from MiR Fleet.
- Collision avoidance information is not shared with the robot.
- The robot is no longer considered in Auto charging and Auto staging.

If you re-add the robot to MiR Fleet, you must re-add it to any groups it was part of.

Provided the robot has the same IP address, it will use the same ID in the API and can reuse the mutual trust setup.

### 2.17.7 Robot fleet states overview

You can see the fleet state of each robot in the MiR Fleet interface under **Robots**.

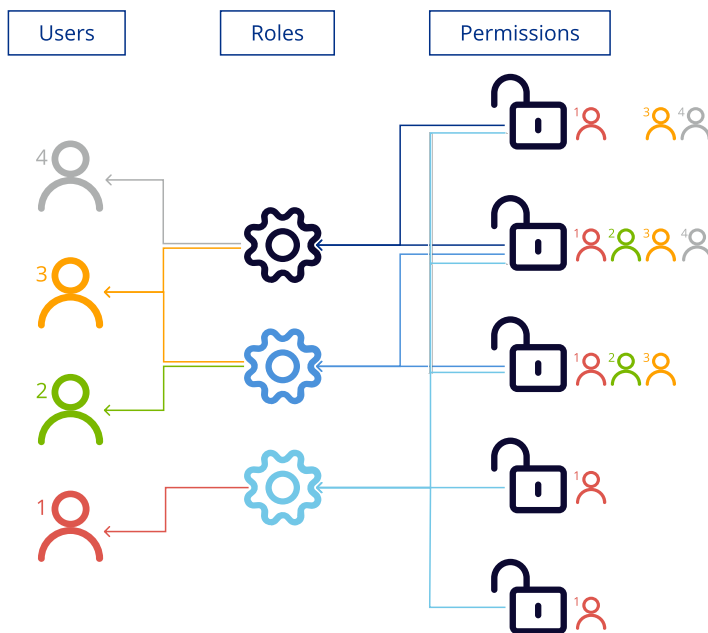
There are three main fleet states robots can be in when they are connected to MiR Fleet:

- **Idle:** The robot is not executing any missions and is in Ready state. Idle robots are automatically assigned to charging stations and Staging positions—see "[Auto Charging and Auto Staging](#)" on page 13.
- **Operational:** The robot is executing missions assigned by MiR Fleet.
- **Not operational:** The robot is in a state where it cannot be assigned missions by MiR Fleet. This includes:
  - **Cable charger connected:** A charging cable is connected to the robot's charging interface.
  - **Error:** The robot is in Error state. Resolve the error and clear the state in the interface to bring the robot into operation.
  - **Manual control:** The robot is being controlled manually with the joystick.
  - **Operating mode key state:** The operating mode is changed to something other than Automatic mode:
    - **Locked mode:** The robot's Operating mode key is set to Locked. Put the key to Automatic mode and press the Resume button to bring the robot into operation.
    - **Manual mode:** The robot's Operating mode key is set to Manual. Either activate Manual control in the interface and press the Resume button, or put the key to Automatic mode and press the Resume button to bring the robot into operation.
  - **Paused:** The robot has been Paused. Toggle the Paused/Continue state to bring the robot into operation.
  - **System busy:** The robot system is completing a task and the robot cannot be controlled until it is complete:
    - **Shutting down:** The robot is shutting down.
    - **Starting up:** The robot is starting up. Wait for the robot to finish starting up and press Resume to bring the robot into operation.
  - **Safety-related stop:** These are states where the robot is stopped due to the safety system:
    - **Scanners blocked:** The safety laser scanners have detected an obstacle within their active Protective field—see Personnel detection in the user guide or integrator manual for your robot application. The robot will resumes operation once the field is clear for two seconds.
    - **Safety restart required:** A safety function was triggered. Press the Restart button to bring the robot into operation.
    - **Emergency stop:** The robot is in Emergency stop. Release the Emergency stop button and press Resume to bring the robot out of Emergency stop.
    - **Brakes released:** The Manual brake release switch is active. The brakes are released and the robot is in Protective stop until the brakes are engaged.

## 2.18 Roles and users

MiR Fleet uses the flat RBAC (Role Based Access Control) paradigm for user management. This means that each user in the system is assigned permissions through roles. This is done based on the following principles:

- You can define any number of users and roles in the system.
- You can assign multiple roles to each user.
- You can assign the same role to multiple users.
- Each role can enable any number of permissions.
- The same permissions can be enabled in multiple roles.



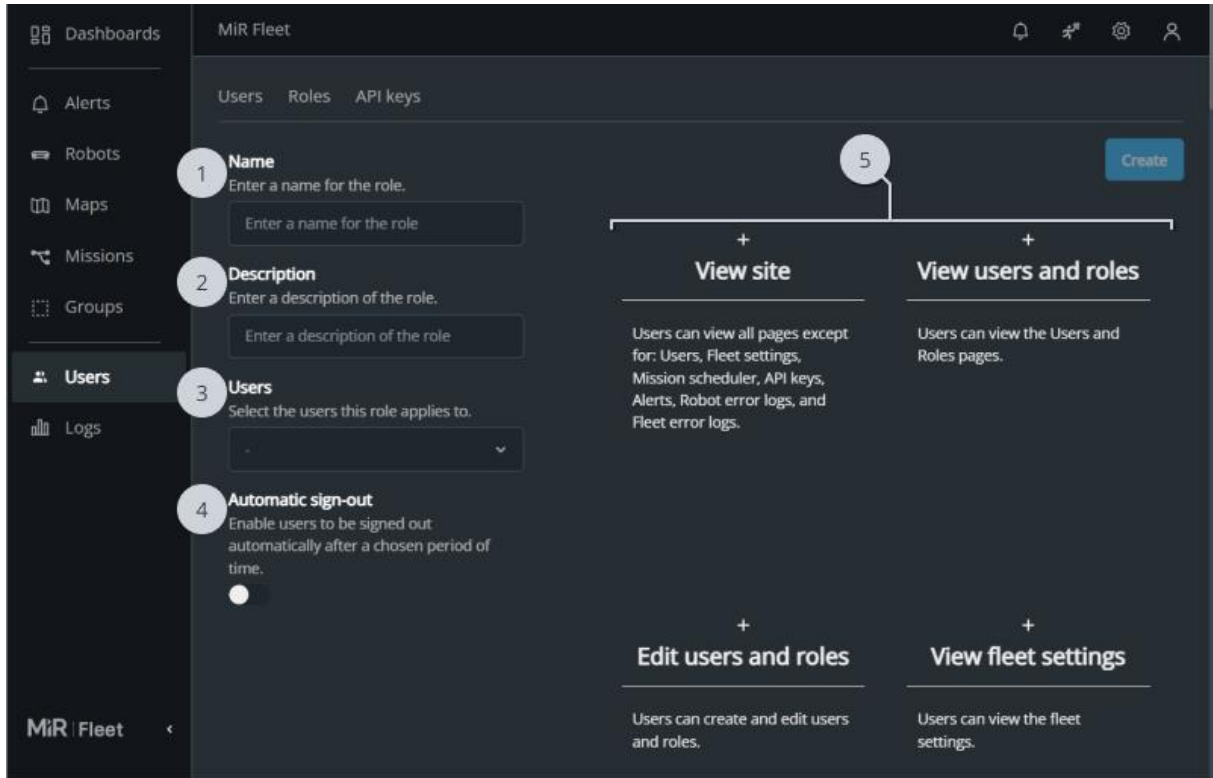
Users may edit elements if the following applies:

- The role the user is assigned to has permission to edit the element type.
- The element was created by yourself or a user that you created. This also includes any child-users created by a user you created.

By default, MiR Fleet starts with one user that is granted all permissions in the system. This user cannot be deleted or modified. This ensures that there is always a user that can access the system.

Before creating any users or roles, we recommend mapping your users and their tasks to limit the number of permissions each user has accordingly—see ["Create users and roles"](#) on page 257.

### 2.18.1 Editor overview



Description	Description
<p>1 <b>Name</b></p> <p>Enter the name of the role.</p>	<p>2 <b>Description</b></p> <p>Enter a description of the role. Use a description that clarifies which users should be assigned to the role.</p>
<p>3 <b>Users</b></p> <p>Select which users the role applies to.</p>	<p>4 <b>Automatic sign out</b></p> <p>Enable to make MiR Fleet automatically sign out the user if they have been idle for the set timeout.</p>



Description	Description
<p>5 <b>Permissions</b></p> <p>Select the permissions for all users assigned to the role. All enabled permissions are highlighted with blue.</p>	

### 2.18.2 Password requirements

Passwords must contain at least:

- Eight characters
- One special character
- One lower-case letter
- One upper-case letter
- One number

### 2.18.3 Permissions

	View	Edit
Site	Enables users to see the pages for the site components listed below that refer the Site view permissions.	There is no collected permission to edit a site. You must enable each site component individually.
Dashboards	Is enabled with Site view permission.	<p>Users can create and edit dashboards.</p> <p>Users must also have access to <b>View site</b>.</p> <p>To use certain widgets, you</p>

	View	Edit
		<p>must enable additional permissions:</p> <ul style="list-style-type: none"> <li>• To assign or abort missions through mission or map widgets: <b>Schedule missions</b></li> <li>• To view the mission queue <b>View scheduled missions</b></li> <li>• To continue or pause the selected robot's mission: <b>Manage robots</b></li> <li>• To create fleet error logs: <b>Generate fleet error log</b></li> </ul>
Marker types	Is enabled with Site view permission.	<p>Users can create and edit marker types.</p> <p>Users do not need this permission to edit the marker calibration in the robot settings.</p>
Footprints	Is enabled with Site view permission.	<p>Users can create and edit footprints.</p> <p>Users do not need this permission to set the robot footprint.</p>
Groups	Is enabled with Site view permission.	Users can create and edit groups.
Maps	Is enabled with Site view permission.	Users can create and edit maps.

	<b>View</b>	<b>Edit</b>
Missions	Is enabled with Site view permission.	Users can create and edit missions.  To let users schedule missions, enable <b>Schedule missions</b> .
Sounds	Is enabled with Site view permission.	Users can create and edit sounds.
Carts	Is enabled with Site view permission.	Users can create and edit carts for MiR hook robots.
Robots	Is enabled with Site view permission.	Users can add and remove robots from MiR Fleet and continue or pause the selected robot's mission.  On dashboards, users can use the Continue/Pause widget.
I/O modules	Is enabled with Site view permission.	Users can create and edit I/O modules.
Transitions	Is enabled with Site view permission.	Users can create and edit map transitions.
Fleet settings	Users can view the fleet settings.	Users can edit the fleet settings.  To let users import or export site settings, enable <b>Import and export site</b> .

	<b>View</b>	<b>Edit</b>
Schedule missions	<p>Users can view the Mission scheduler page.</p> <p>On dashboards, users can view the Mission queue widget.</p>	<p>Users can schedule missions from the mission scheduler.</p> <p>On dashboards, users can cancel missions using the Mission queue widget.</p>
Users and roles	Users can view the Users and Roles pages.	Users can create and edit users and roles.
API keys	<p>Users can view the API keys.</p> <p>Users cannot see the authentication string for an API key.</p>	<p>Users can create and delete API keys.</p> <p>Users cannot see the authentication string for an API key.</p>
Alerts	Users can view alerts.	Users can view and edit alerts.
Robot error logs	Users can download robot error logs.	<p>Users can create and download robot error logs.</p> <p>On dashboards, users can use the Robot error log widget.</p>
Fleet error logs	Users can generate and download fleet error logs.	<p>Users can generate, delete, and download fleet error logs.</p> <p>On dashboards, users can use the Fleet error log widget.</p>
Evacuations	Users can start and stop evacuations in the fleet.	

### 2.18.4 API Keys

Create API keys to enable devices to send REST API requests via the MiR Fleet Integration API—see ["MiR Fleet Integration API" on page 484](#).

Each request to and from an endpoint must contain an API key.

You can request an API key in the web interface under **Setup > Users > API keys**. When you request a key, it is only shown once after the request. We recommend saving the API key in a secure location for later use. If you lose the API Key or want to update the access rights, you must generate a new key.

In each request you send to MiR Fleet, the request must include an **x-api-key** header where you pass the API key.

If you are using the Swagger page, you can enter the API key in the upper-right corner for access.

## 2.19 Security

The cybersecurity of your MiR system depends on how you commission your setup and assess it. We provide security guidelines for how we recommend to set up your system and list common risk factors you must assess—see ["Evaluate cybersecurity" on page 464](#).

MiR Fleet has the following built-in cybersecurity features:

- **Single sign-on support:** You can connect your existing authentication protocol via OpenID—see ["Single Sign-On \(SSO\)" on page 237](#). This enables security policies such as advanced passwords requirements or Multi-Factor authentication.
- **Audit logging:** MiR Fleet publishes security and access-related notifications to an external log file. For an overview of what each log entry includes, see ["Logging" on page 40](#).
- **External database:** The database with MiR Fleet data is saved outside the MiR Fleet software on a Microsoft SQL database. You can choose where the database is hosted and what security measures you want to apply to it—see ["SQL Database" on page 237](#).
- **Mutual trust:** Robots and MiR Fleet must exchange mutual trust tokens to enable communication between them—see ["Add robots" on page 276](#). You set this up each time you add a robot to MiR Fleet.
- **User permission system:** Set up users in the system and control which features they have permission to view and edit, and set up their session timeout—see ["Roles and users" on page 95](#).

- **System use notification:** You can write a custom message for all users that is displayed on the sign in page—see ["System settings" on page 106](#).
- **Compatibility with malware protection:** MiR Fleet should be compatible with most malware protection softwares you can run on your windows Server. All MiR Fleet software is compatible with Windows Defender.
- **Basic Denial-of-Service detection and protection:** Rate-limiting in the MiR Fleet Integration API protects against Denial-of-Service attacks—see ["Endpoints" on page 525](#).
- **Signed installation file:** The MiR installation file is signed using an Extended Validation Code Signing certificate to ensure authenticity and protect against tampering.
- **Protection against password attacks:** MiR Fleet locks user accounts if too many failed sign-in attempts are detected.

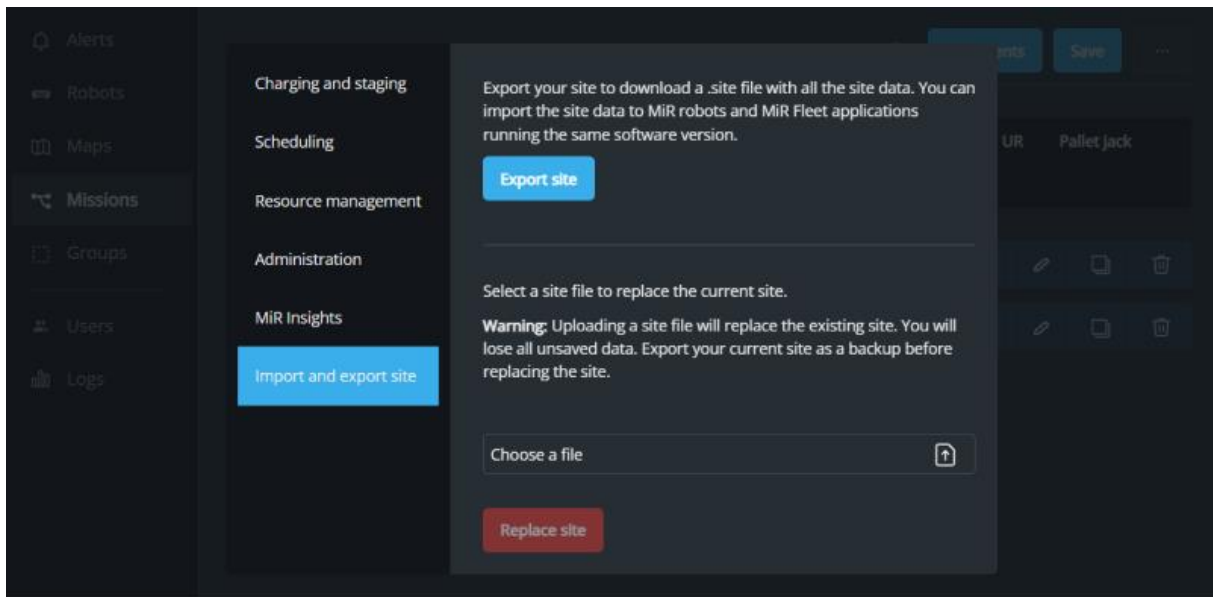
## 2.20 Sites

A site is the whole facility where the robots operate. A site consists of any number of maps and missions including all other linked components, such as Marker types, footprints, and transitions. If two maps are connected using a transition, they must be part of the same site.

Export your site to have a copy of all your site data. We recommend exporting a .site file before you make large changes to your site in case you want to revert to your previous setup.

Import sites to overwrite your current site setup with the data on the saved site file.

You export and import the site from the ["System settings" on page 106](#). Sites are saved as .site files.



A site contains the following elements:

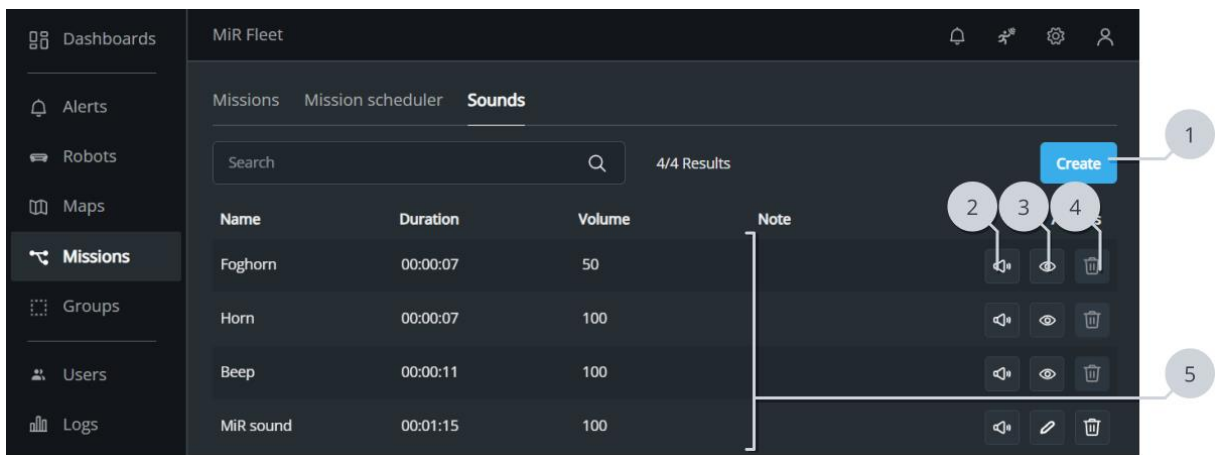
- Cart calibrations
- Cart types
- Carts
- Dashboards
- Docking offsets for markers
- Floor plans
- Footprints
- Groups
- I/O modules
- Maps
- Missions
- Mission groups
- Positions and markers
- Sounds
- Transitions
- Zones

### 2.20.1 Site compatibility

You can only import .site files that were exported from a MiR Fleet running any other software 4.x versions. You cannot import .site files from software 2.x or 3.x.

## 2.21 Sounds

You can make the robot play sounds using Play sound actions in missions or Sound and light zones on maps. Sounds are often used to alert personnel in operating hazard zones—see ["Mitigate risks" on page 439](#).



Description	Description
<p><b>1 Create</b></p> <p>Opens the dialog box to create a new sound.</p>	<p><b>2 Play sound</b></p> <p>Plays the sound on the device you have used to access the MiR Fleet interface.</p>
<p><b>3 View or Edit</b></p> <p>Opens the dialog box to view or edit the selected sound, depending on your user permission.</p>	<p><b>4 Delete</b></p> <p>Deletes the sound if you have permission to edit sounds.</p>
<p><b>5 Sounds</b></p>	



Description	Description
Lists all sounds saved on MiR Fleet. You can sort the sounds based on the name, the duration, or the volume.	

## 2.22 Synchronization

MiR Fleet shares all of its site data with connected MiR robots so they are all running on the same data—see "[Sites](#)" on [page 102](#). Each time a change happens to the site on MiR Fleet, the change is pushed to all connected robots.

The synchronization is only from MiR Fleet to the robots. Changes on robots are not synchronized back to MiR Fleet.

### 2.22.1 Initial synchronization

When you add a new robot to MiR Fleet, there is an initial synchronization. The entire site stored in the Fleet is pushed to the robot, which replaces whatever is stored on the robot.

### 2.22.2 Continuous synchronization

Whenever site data is changed on MiR Fleet, the new change is pushed to the robots. In this way the site is always up to date on all robots.

Synchronization can only occur as long as MiR Fleet and the connected robot are both on the same network and they have a stable connection. If a robot disconnects, it cannot receive any updates until it is reconnected. Once reconnected, its site data is updated to the latest version on MiR Fleet.

### 2.22.3 Collision avoidance

Collision avoidance in an internal functions that synchronizes the current footprints and positions of the robots connected to the fleet. It is intended to prevent collision between loads or top modules on robots that exceed the robot's footprint.

If a MiR robot detects another robot with its laser scanners, MiR Fleet forwards the footprint of the robot in that location. The robot adds the footprint of the detected robot to its obstacle cloud so it can safely navigate around it.

If robots collide even with Collision avoidance enabled, use Directional zones to control traffic flow in areas where collisions happen often.

## 2.23 System settings

### 2.23.1 MiR Fleet

In the MiR Fleet system settings there are the following options.

#### Charging and staging

For more information about Auto charging and Auto staging, see ["Auto Charging and Auto Staging" on page 13](#).

- **Auto charging:** Enable MiR Fleet to initiate the Auto charging sequence for a robot that is idle for longer than the set Idle time.
- **Auto staging:** Enable MiR Fleet to initiate the Auto staging sequence for a robot that is idle for longer than the set Idle time.
- **Idle time:** Enter the minimum number of seconds a robot must be in Ready state (idle) before MiR Fleet initiates the Auto charging or Auto staging sequence.
- **Minimum charging time:** Enter the minimum number of minutes a robot must be assigned to a charger before it can be released for a staging mission. This does not affect missions used for evacuations or user scheduled missions. This is also dependent on Minimum battery percentage for release.
- **Minimum battery percentage for release:** Enter the minimum battery percentage a robot must be charged to before it can be released for a staging or user scheduled mission. This does not affect missions used for evacuations. Staging missions are also dependent on the **Minimum charging time** parameter.
- **Maximum battery percentage for Auto charging:** Enter the maximum battery percentage a robot can have for MiR Fleet to initiate an Auto charging sequence for the robot.

#### Scheduling

Enter the minimum battery percentage a robot must have before the mission scheduler can assign a new user scheduled mission to the robot. This does not affect missions used for Auto charging, Auto staging, and evacuations.

## Resource management

Adjust the resource assignment distance. This is the maximum length in meters between the robot and the next fleet resource it needs for its mission before MiR Fleet assigns the resource to the robot—see ["Resource handling" on page 87](#).

## Administration

- **Fleet name:** Enter the name of MiR Fleet. This is a custom name used to identify this instance of MiR Fleet. It is shown on the sign-in page.
- **System use notification:** Enter any important notification for users. Your note is shown on the sign-in page.

## MiR Insights

If you have a MiR Insights license, enter the client ID and secret to allow MiR Fleet to connect to MiR Insights.

You can get the ID and secret from [insights.mir-robots.com](https://insights.mir-robots.com) under **Fleets > Connect a MiR Fleet**.

For more information about MiR Insights, see *MiR Insights User Guide*. You can find this guide on [MiR Support Portal](#).

## Site settings

Select **Export site data** to download a .site file of the current site—see ["Sites" on page 102](#).

Select **Replace site** to upload a .site file and replace the current site with its data. Always export the current site data before replacing the site with another .site file. This ensures you have a backup of any data that you regretted replacing.

### 2.23.2 MiR robots

In the MiR robot system settings, you only make changes to the selected robot. For an overview of what you can modify on the robot, see ["System settings" on the previous page](#).

## 2.24 Transitions

A transition is a connection between two maps in the site. It defines how the robot should change from one map to another. Each transition consists of:

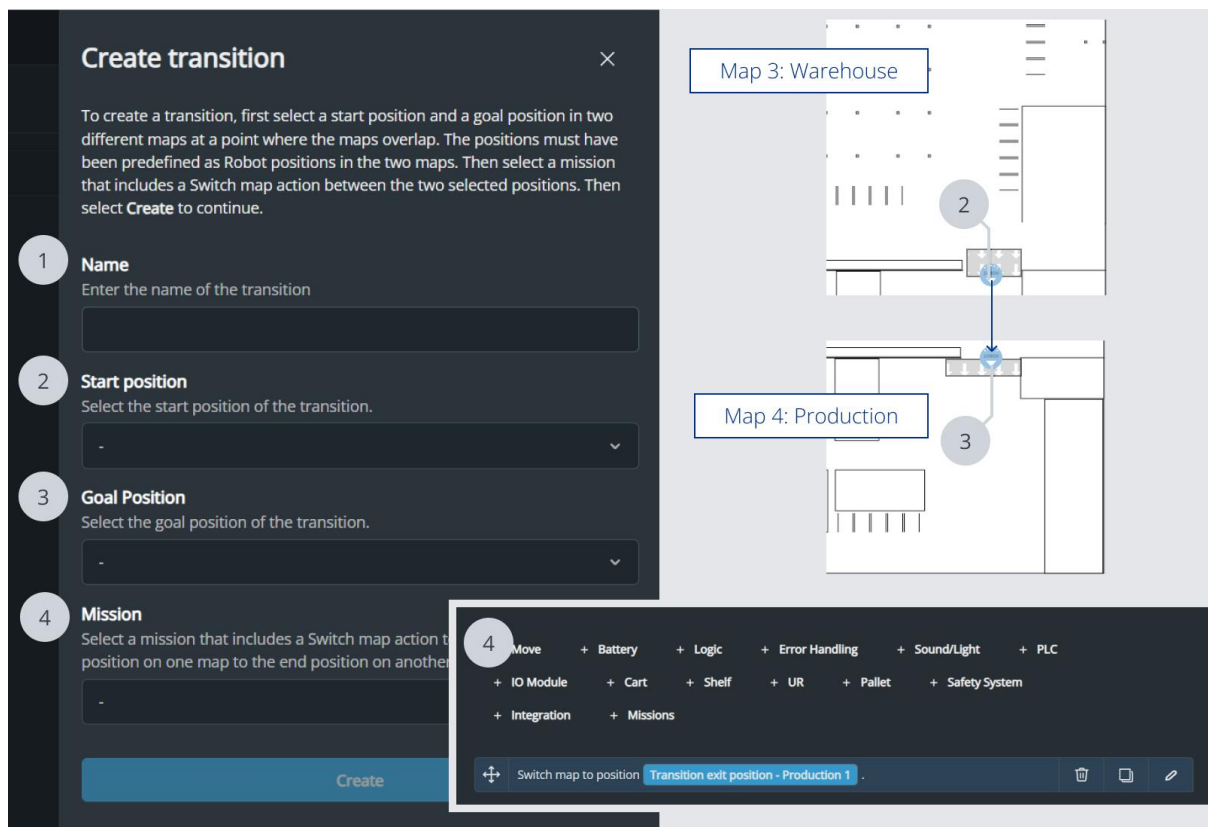
- A position in the map the robot is on
- A position in the same physical location as the first position, but on the map the robot should transition to
- A transition mission that contains a Switch map action

If you define transitions between all maps in the site, the robot will automatically transition between maps when it is running missions that sends it to locations on other maps.

You must define a new transition for each transition direction. Going from map A to map B and from map B to map A requires two different transitions and transition missions.

When the robot transitions between maps, it will stop at the transition position shortly before continuing its mission.

For guidelines on when to use transitions, see ["Create transitions" on page 325](#).



Description		Description	
1	<b>Name</b> Enter the name of the transition.	2	<b>Start position</b> Select the start position of the transition.
3	<b>Goal position</b> Select the goal position of the transition.	4	<b>Mission</b> Select a mission that includes a Switch map action to switch from the start position on one map to the end position on another map.

## 3. Robot interface

The following sections describe how you use the pages available in the robot interface—see ["Interfaces and modes" on page 32](#).

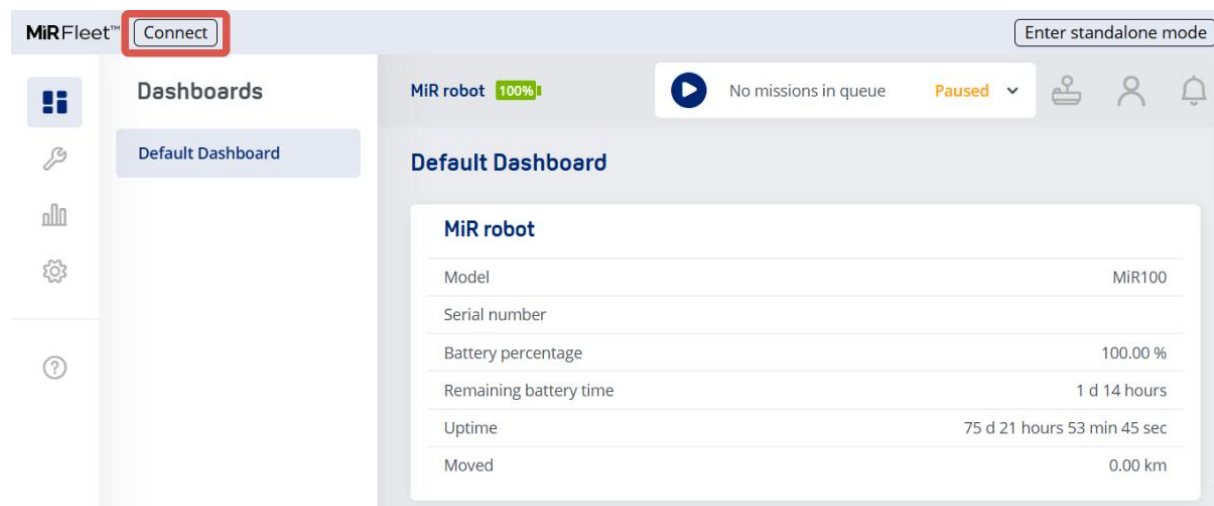
### 3.1 Mutual trust page

Use the mutual trust page to set up mutual trust between the robot and MiR Fleet.

Before adding a robot to MiR Fleet, you must set up mutual trust between that robot and MiR Fleet—see ["Add robots" on page 276](#).

The mutual trust key for each robot is based on the robot's MAC address and the IP address. Once you have added mutual trust for a robot, it is not necessary to do it again, unless you change the IP address of the robot or MiR Fleet.

To access the mutual trust page, go to the robot interface of your robot and select **Connect**.



### 3.2 Setup

The setup menu lets you record floor plans, select the active map, and manage I/O modules.

### 3.2.1 Floor plan recording

Use the floor plan recording page to record new floor plans that are immediately uploaded to MiR Fleet—see ["Record floor plan" on page 300](#).

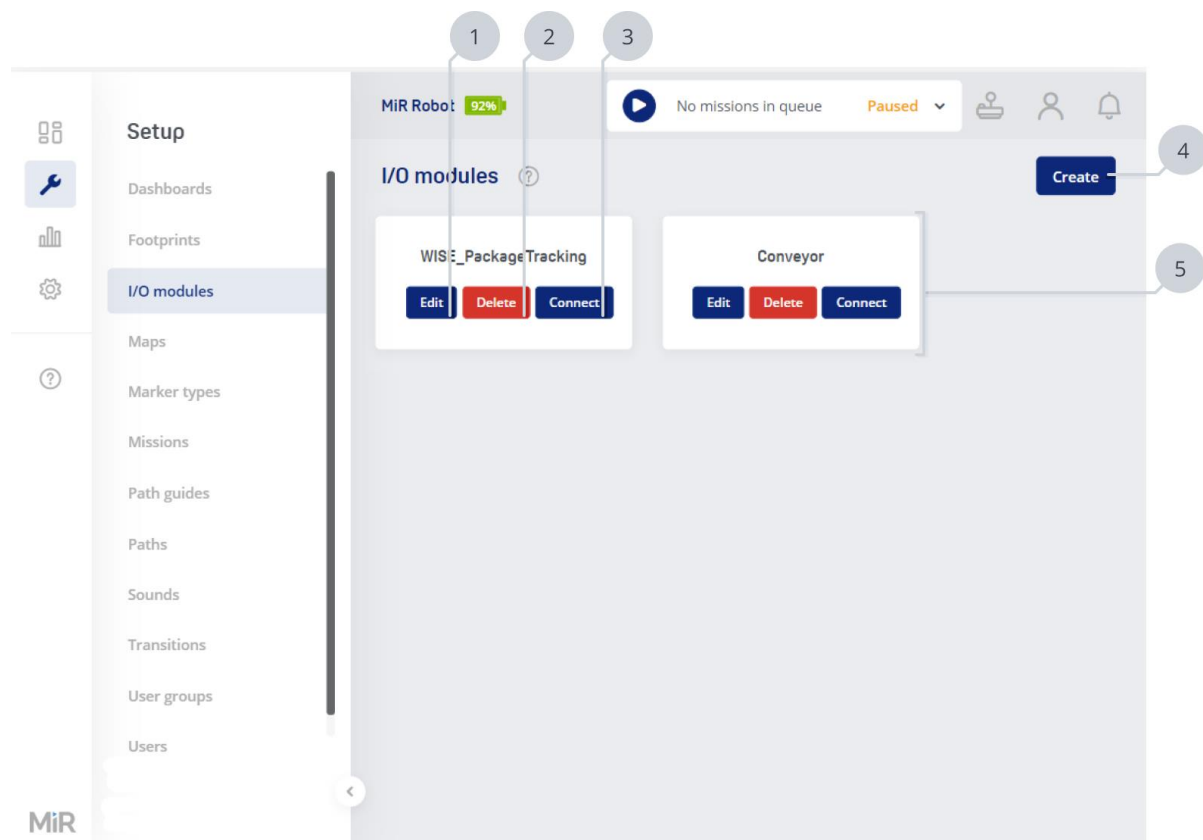
### 3.2.2 I/O modules

An I/O module defines and controls the robot's connection to an I/O device that the robot can communicate with. The device must be connected to the same network as the robot and is identified using its IP address.

I/O modules are used for receiving inputs and sending outputs with external devices, enabling the robot and the connecting device to send signals that can be linked to certain actions or missions.

Control or read I/O registers using I/O actions in missions or Integration zones on maps.

For more information about setting up WISE modules, see the guide *How to use WISE modules*.

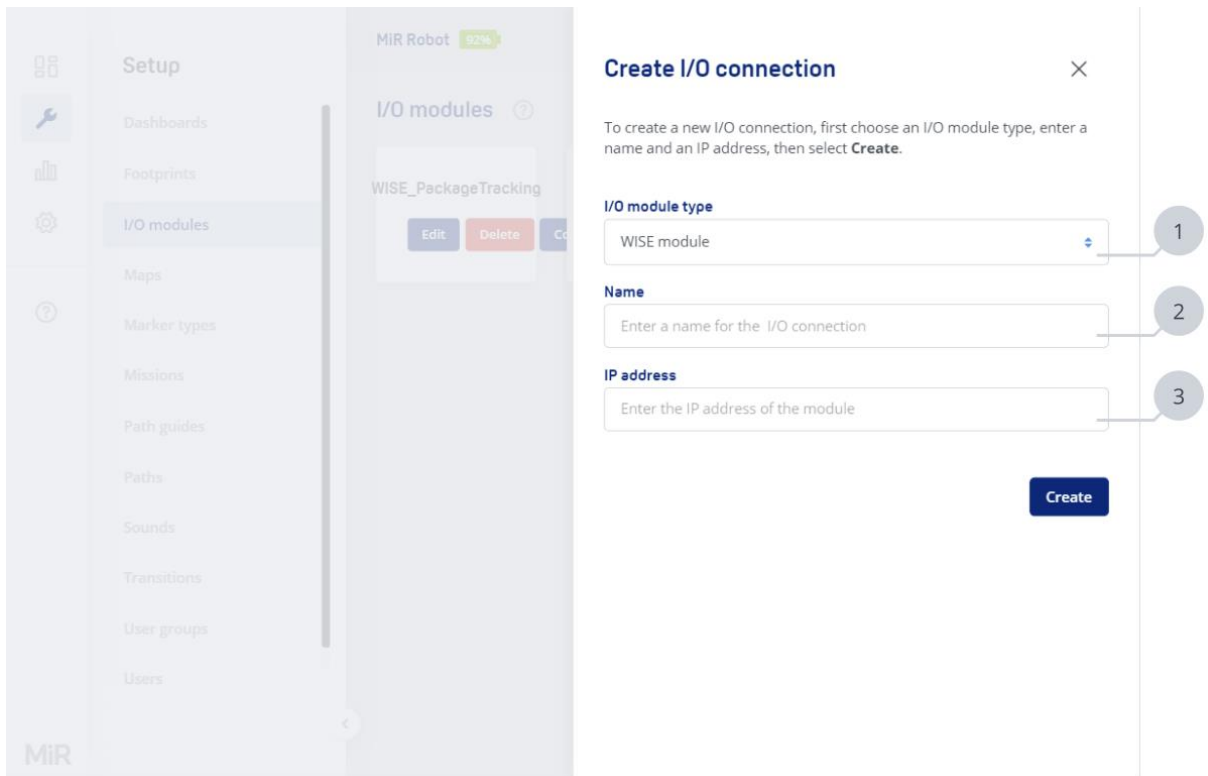


Description	Description
<p>1 <b>Edit</b></p> <p>Opens the dialog box to edit the selected I/O module.</p>	<p>2 <b>Delete</b></p> <p>Removes the selected I/O module.</p>
<p>3 <b>Connect</b></p> <p>Tries to connect the robot to the I/O device. Once connected, you can see the inputs received from the device and control which outputs from the robot are active or inactive. You can use this to test any connected I/O device.</p>	<p>4 <b>Create</b></p> <p>Opens the dialog box to create a new I/O module—see "<a href="#">Create I/O connection</a>" below.</p>
<p>5 <b>I/O modules</b></p> <p>All of the I/O modules you create to define a connection to an I/O device are displayed here.</p>	

### Create I/O connection

To create a new I/O connection, first choose an I/O module type, enter a name and an IP address, then select **Create**.





Description		Description	
1	<b>I/O module type</b> Select the type of I/O device you want to connect to the robot.	2	<b>Name</b> Enter a name for the I/O module.
3	<b>IP address</b> Enter the IP address of the I/O device you want to connect to the robot.		

### 3.2.3 Maps

In the robot interface, you can see a list of all available maps and select which is active for the robot.

You cannot make any modifications to the maps from the robot interface. All modifications are done in the MiR Fleet interface.

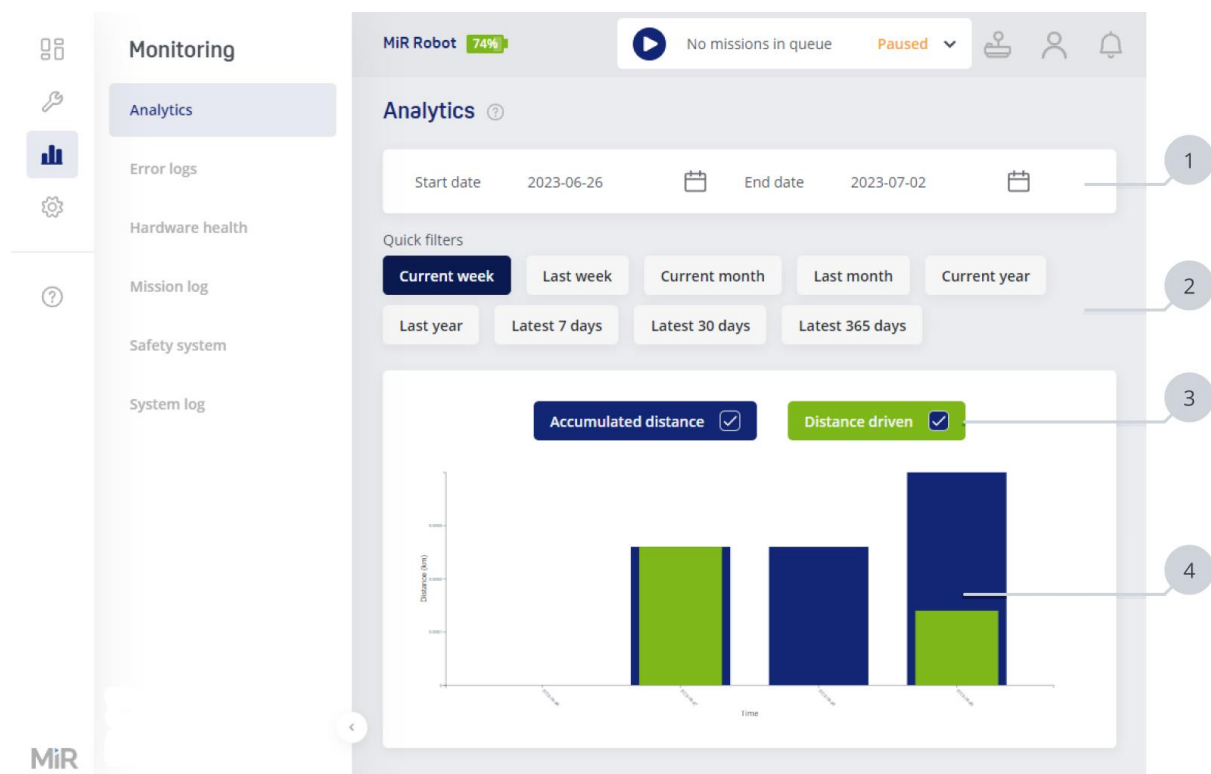
## 3.3 Monitoring

The Monitoring menu contains pages that let you view data collected by the robot.

### 3.3.1 Analytics

Analytics gives a graphic overview of the robot's driven distance over a specified period of time.

You can select a period either by specifying a fixed start and end date or by clicking on one of the buttons spanning from current week to the last 365 days. In addition, you can choose whether to see a chart per day or per month, and you can see a graph showing the accumulated distance for the selected time period in addition to the default bar graph view.



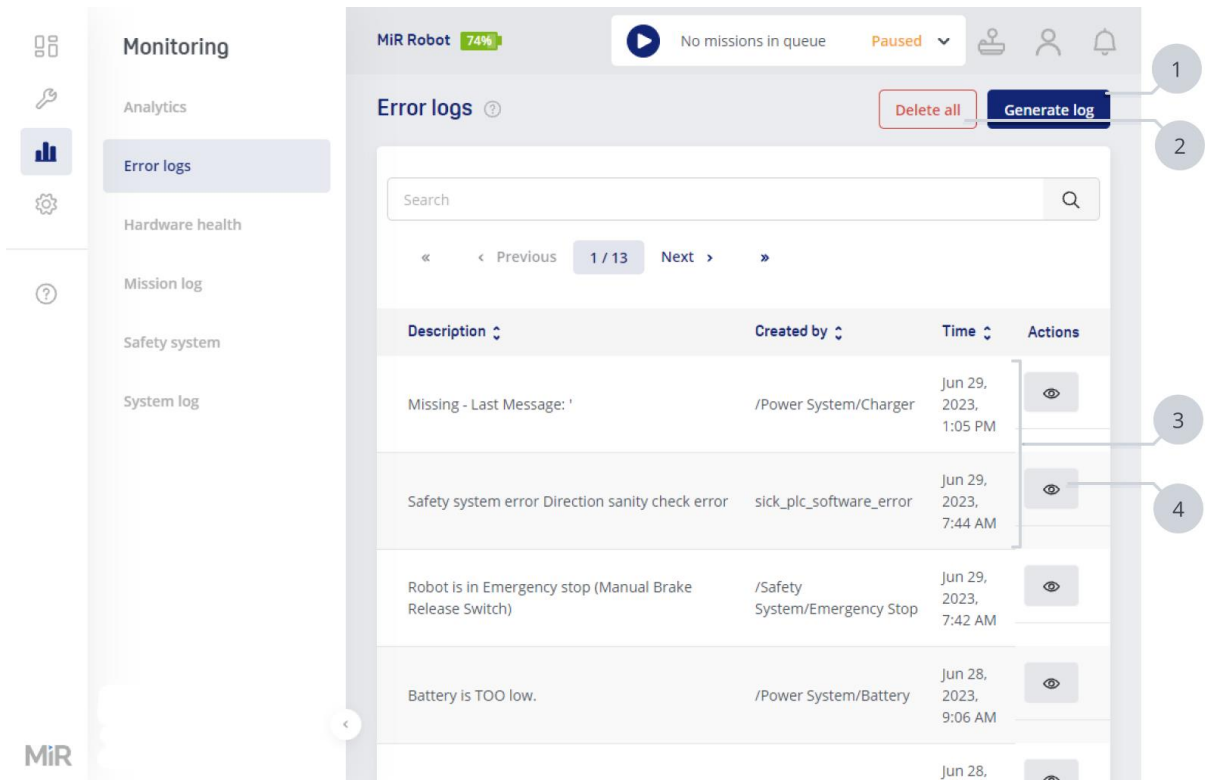
Description	Description
<p>1 <b>Start date and End date</b></p> <p>You can limit the data to a specific time span by selecting a start and end date.</p>	<p>2 <b>Quick filters</b></p> <p>You can select one of the quick filters to limit the shown data to the selected time span.</p>
<p>3 <b>Legend</b></p> <p>You can select whether the graph should show both or only one of the following data sets:</p> <ul style="list-style-type: none"> <li>• <b>Accumulated distance:</b> Is the sum of all the distances traveled on the previous days included in the time span.</li> <li>• <b>Distance driven:</b> Is the distance the robot has driven that day only.</li> </ul>	<p>4 <b>Graph</b></p> <p>The graph displays a combination of the distance the robot has driven each day or the accumulated distance it has driven from the first data point. You can use the legend to filter the data.</p>

### 3.3.2 Error logs

An error log is a file that records everything that has happened in the robot's system in the last 30 seconds before the file is generated.

Error logs are used by MiR Technical Support to diagnose the cause of any issues occurring with your robot. Error logs are generated automatically if the robot detects an error and enters Error state. If an error is not recognized by the robot, you can generate an error log manually within 30 seconds of the error occurring.

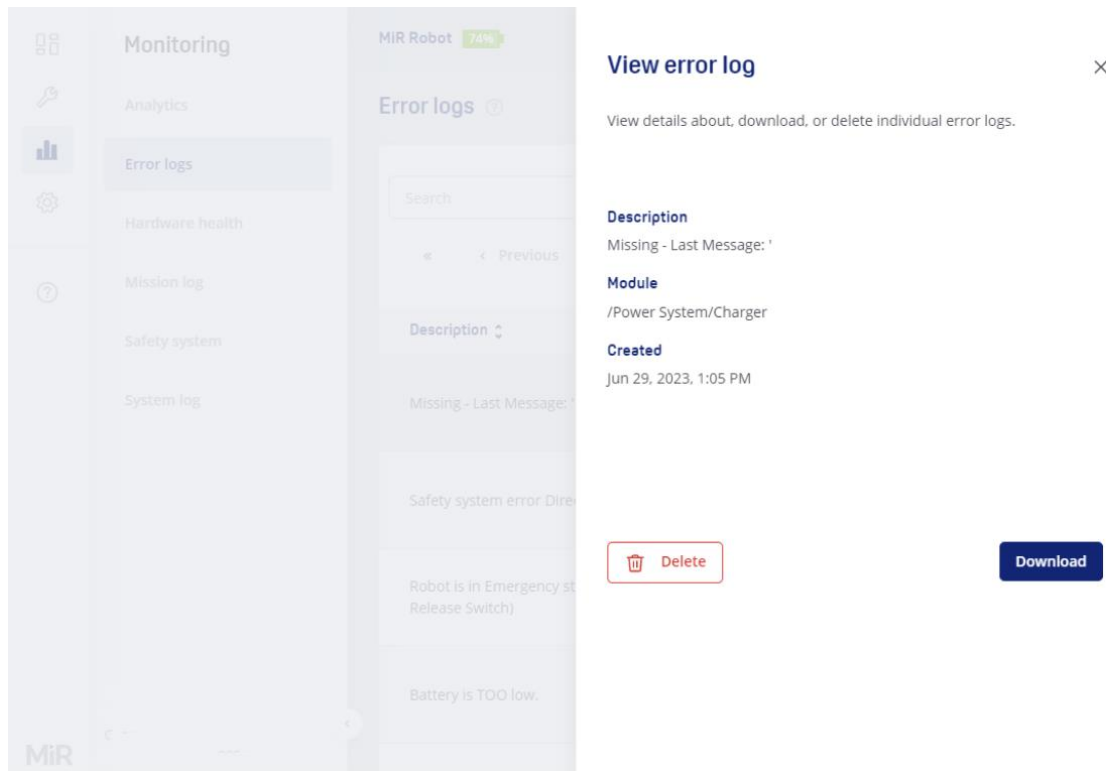
To send an error log to MiR Technical Support, determine which error log captures the error you are experiencing based on the date and description, and download the encrypted file. You can then send this file with your support request.



Description	Description
<p>1 <b>Generate log</b></p> <p>Generates a new error log that contains all of the system data on the robot from the last 30 seconds. For more information about error logs, see the guide <i>How to generate an error log</i>. You can find this guide on <a href="#">MiR Support Portal</a>.</p>	<p>2 <b>Delete all</b></p> <p>Deletes all existing error logs.</p>
<p>3 <b>Error logs</b></p> <p>Lists all error logs generated by . You can sort the error logs based on the description, which user created them, or when they were generated.</p>	<p>4 <b>View error log</b></p> <p>View details about an error log or download it—see "<a href="#">View error log</a>" on the <a href="#">next page</a>.</p>

## View error log

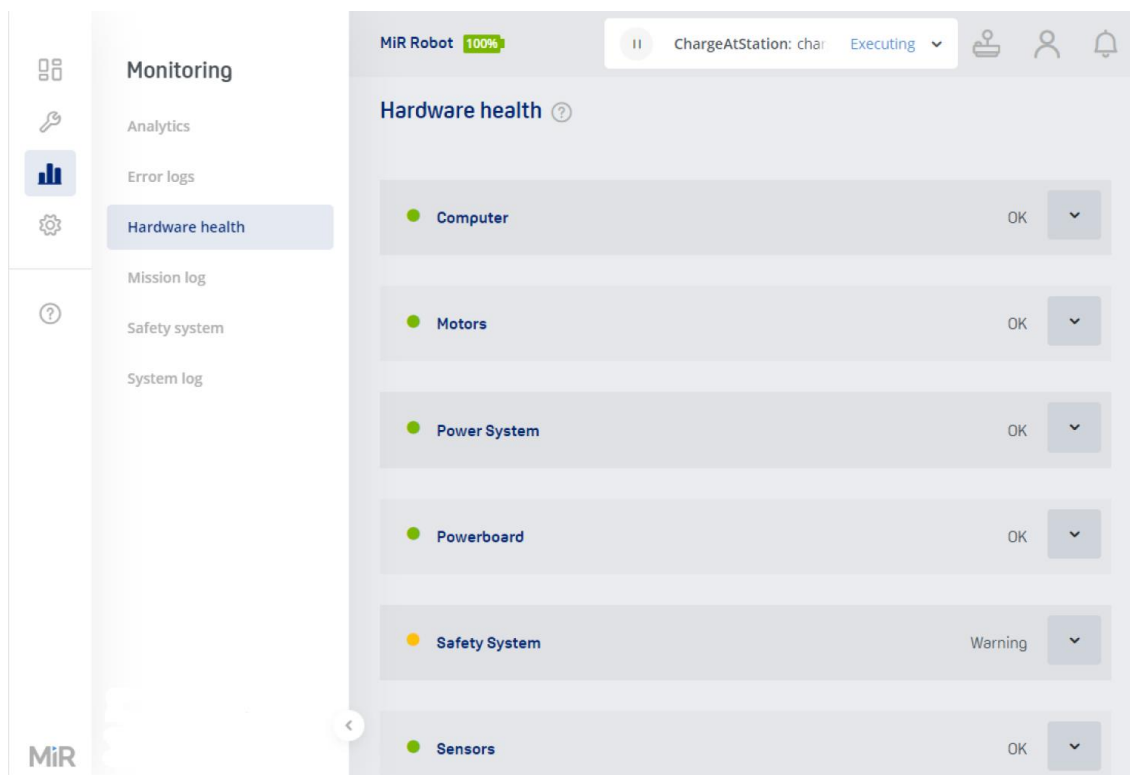
View details about, download, or delete individual error logs.



### 3.3.3 Hardware health

Hardware health allows you to see information on the robot's hardware components and check their condition.

If all sub-components are functioning as expected, the group will be marked with a green dot, whereas if one or more components in a group are not in the expected condition, the group will be marked with a yellow or red dot. To find out more about the condition, you can expand the group by selecting the arrow next to the group name and see which components are not functioning correctly and why. Each sub-component can be further expanded into one or more sub-parts for further information on the condition.



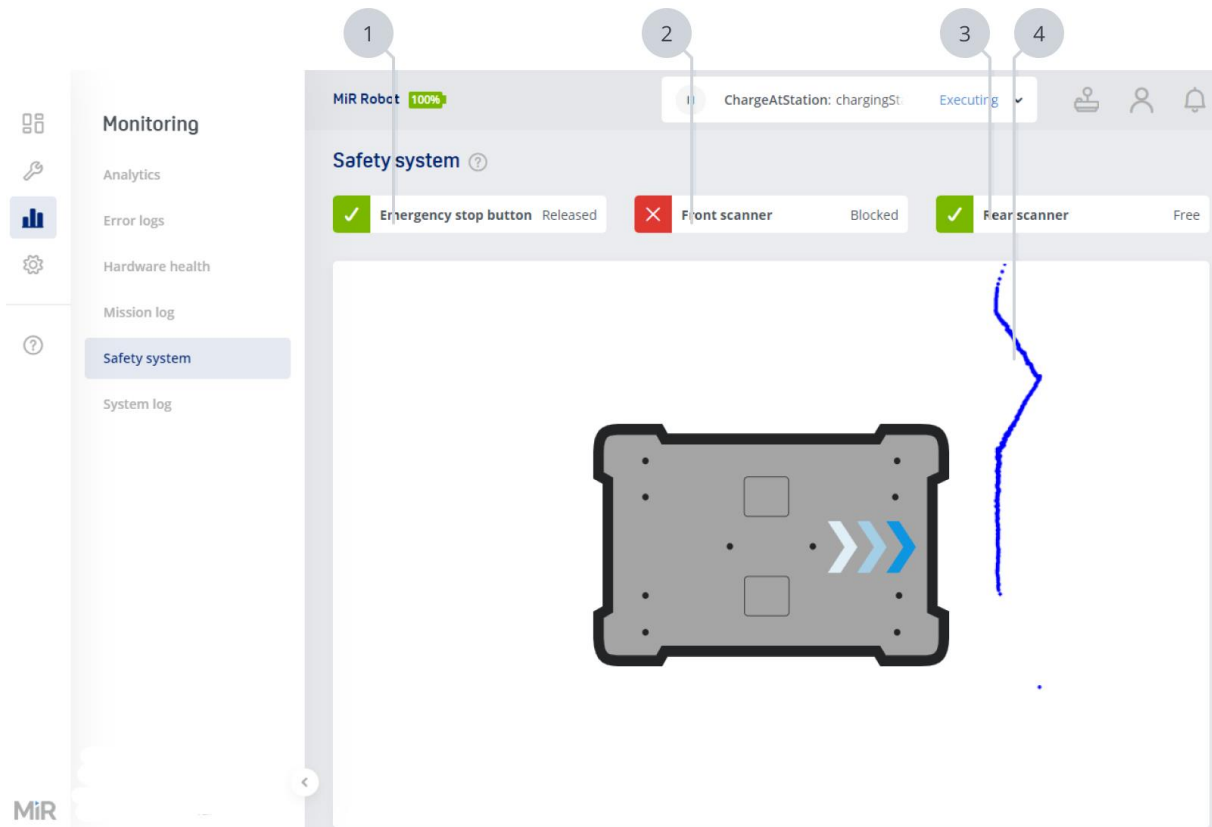
### 3.3.4 Safety system

Safety system provides a live view of the input from the laser scanners and the state of any connected Emergency stop buttons

Use this page to see if the scanners are detecting obstacles as expected. If data is not aligned, is drifting, or there are outlier data points, it may indicate an issue with the laser scanners.

It can often help to clean the safety laser scanners as described in the maintenance section in the user guide or integrator manual for your robot application.

If the surrounding obstacles are transparent or reflective, or there are interferences such as strong light or dense particles in the air, this may also affect the laser scanner data.



Description	Description
<p>1 <b>Emergency stop button</b></p> <p>If any Emergency stop button on the robot is pressed, it is indicated here.</p>	<p>2 <b>Front scanner</b></p> <p>If the front safety laser scanner detects an object within the active Protective field, it is indicated here.</p>
<p>3 <b>Rear scanner</b></p> <p>If the rear safety laser scanner detects an object within the active Protective field, it is indicated here.</p>	<p>4 <b>Scanner data</b></p> <p>A live stream of the scanner data collected by the safety laser scanners is shown here. Each blue dot represents a point where the scanner has detected an obstacle.</p> <p>Single points that are far away from</p>

Description	Description
	<p>clusters of other points are often disregarded as noise. If there are many outliers in the scanner data, consider cleaning the scanners—see the maintenance section in the user guide or integrator manual for your robot application.</p>

### 3.3.5 System log

The system log contains events that are logged by the operating system components. The system log contains information about the system state at a given time, the affected module, a short explanation, and a time stamp. The system log is mainly used by system supporters for troubleshooting.

The screenshot shows the MiR Robot interface. At the top, it displays 'MiR Robot 73%' and 'No missions in queue Paused'. The sidebar on the left has 'Monitoring' selected, with sub-items: Analytics, Error logs, Hardware health, Mission log, Safety system, and System log. The main panel shows the 'System log' with a table of entries:

State	Module	Message	Time
✓	/rosbridge_websocket	[Client 15] Unsubscribed from /diagnostics_agg	1:25:16 PM
✓	/rosbridge_websocket	[Client 15] Subscribed to /mir_log	1:25:16 PM
✓	/rosbridge_websocket	[Client 15] Subscribed to /diagnostics_agg	1:24:23 PM
⚠	/rosterial_server	1688037551.232747: Left encoder reset: -105 now offset with 2415	1:19:11 PM
✓	/rosterial_server	1688037551.136963: Node 4: Going to operational	1:19:11 PM
⚠	/rosterial_server	1688037549.432701: Right encoder reset: 225 now offset with 5190	1:19:09 PM
✓	/rosterial_server	1688037549.348739: Node 3: Going to operational	1:19:09 PM
✓	/rosterial_server	1688037547.468245: Node 20: Going to operational	1:19:07 PM
✗	/StateTF	encoder data is missing, time since last encoder msgs: 0.104786	1:19:07 PM
✓	/supervisor	Robot State Changed: EmergencyStop -> Pause	1:19:05 PM



## 3.4 System

The System menu contains pages that let you modify and view settings specifically for the robot.

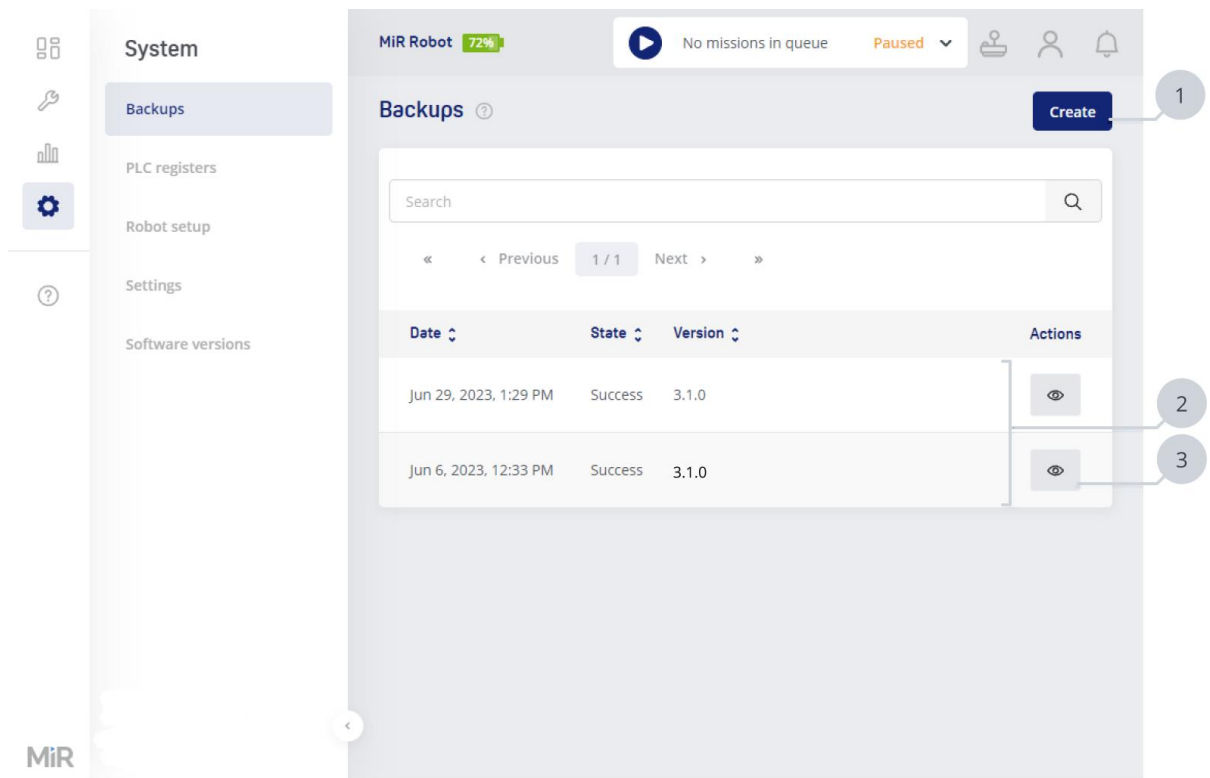
### 3.4.1 Backups

A Backup is a copy of the configuration and system state data of your robot. You can use a backup to revert your robot to a previous state. It contains the site data and system settings at the time the backup was generated. This is most often used if the system fails and you need to recover lost data.

There are two types of backup files: an Application backup and a Platform backup. Application backups contain site data and changes you have applied in the system setting. Platform backups contain hardware related configurations that are set outside of the robot interface.

You can only view and manage Application backups in the robot interface. To manage your Platform backups, you must access the Platform interface—see the guide *How to use backups and recover data on MiR robots*.

Both backup types are automatically created before you update the software version or roll back to a previous backup. You can also generate Application backups manually on this page.

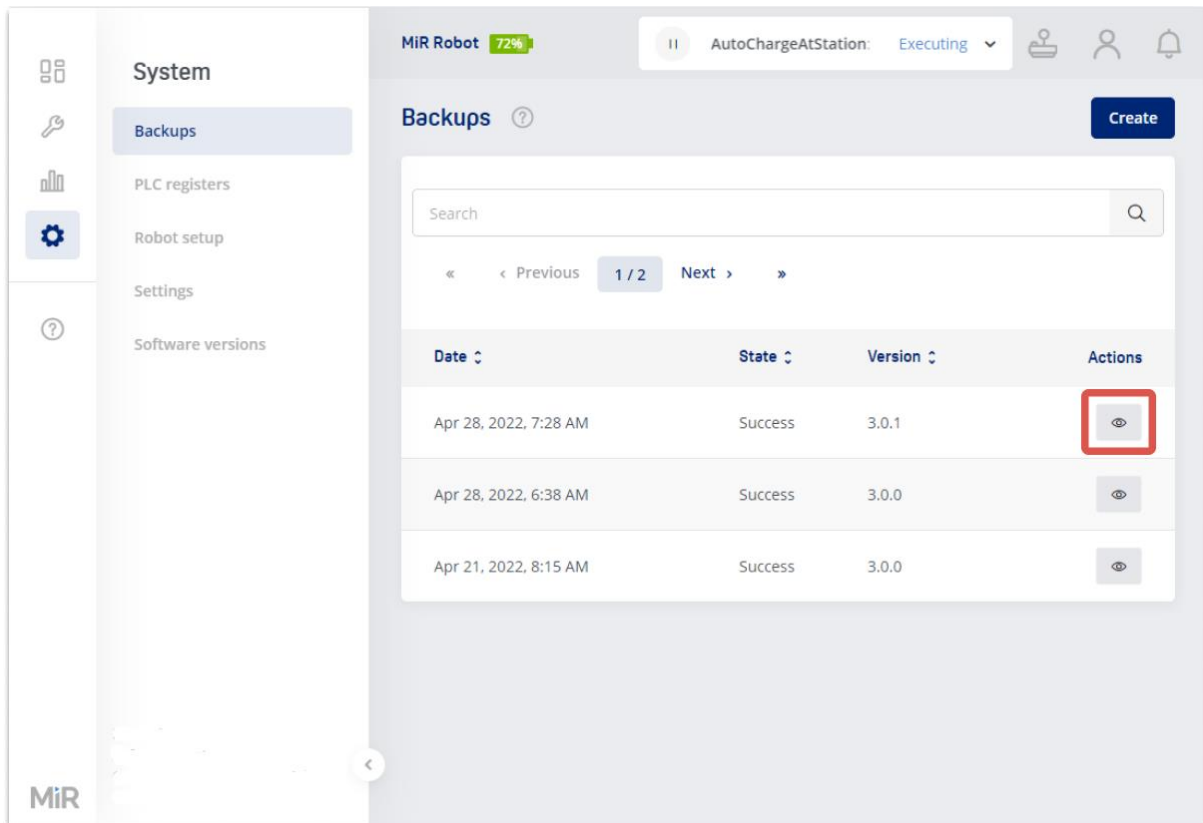


Description	Description
<p>1 <b>Create</b></p> <p>Creates a new backup of your robot's current site and settings. It may take the robot a few seconds to create the backup.</p>	<p>2 <b>Backups</b></p> <p>Lists all backups created for the robot. You can sort the backups based on the date they were created, the state, or the software version.</p>
<p>3 <b>View</b></p> <p>See more information about the selected backup, and choose to either remove or restore the backup—see <a href="#">"View backup" on the next page</a>.</p>	




## View backup

See information about the selected backup, or choose to roll back to this version or remove the backup.

Removing backups you are not going to use frees up more hard drive space on the robot.



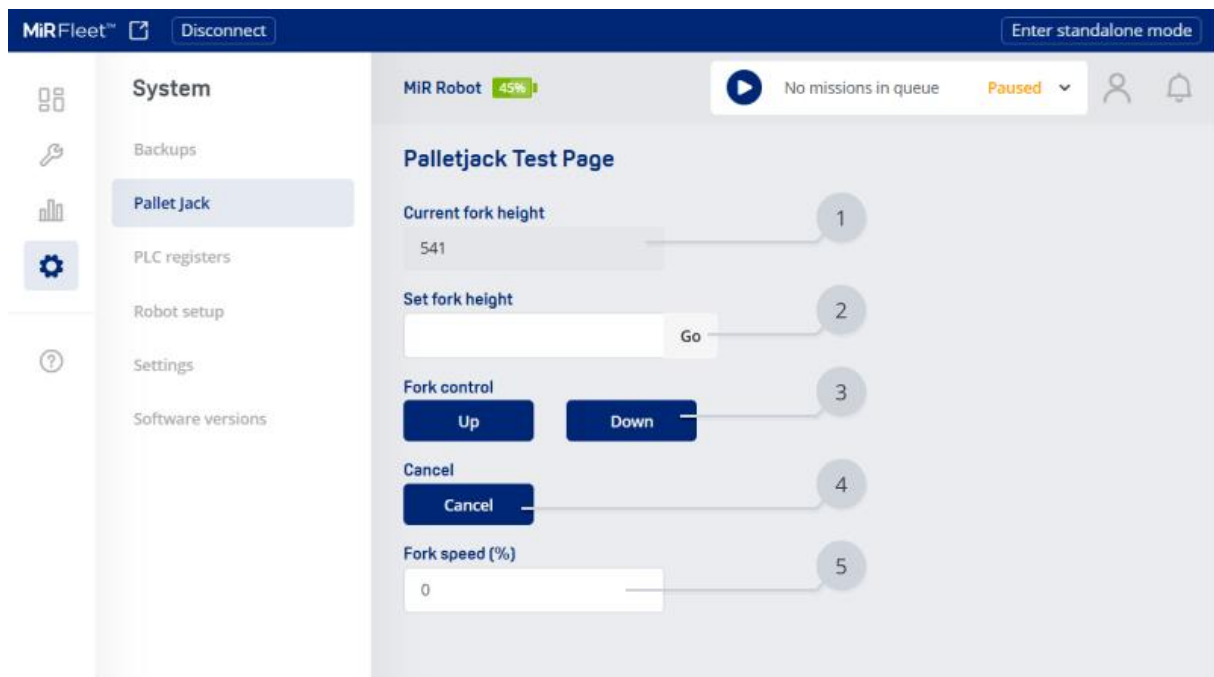
The screenshot displays the MiR Robot interface. On the left, a sidebar shows the 'System' menu with options for 'Backups', 'PLC registers', 'Robot setup', 'Settings', and 'Software versions'. The main panel is titled 'Backups' and features a search bar, a pagination control showing '1 / 2', and a table of backup records. The table has columns for 'Date', 'State', 'Version', and 'Actions'. The first row, representing a backup from 'Apr 28, 2022, 7:28 AM' with 'State: Success' and 'Version: 3.0.1', has its 'Actions' column highlighted with a red box. The other two rows show backups from 'Apr 28, 2022, 6:38 AM' and 'Apr 21, 2022, 8:15 AM', both with 'State: Success' and 'Version: 3.0.0'. A 'Create' button is visible in the top right corner of the main panel.

Date	State	Version	Actions
Apr 28, 2022, 7:28 AM	Success	3.0.1	
Apr 28, 2022, 6:38 AM	Success	3.0.0	
Apr 21, 2022, 8:15 AM	Success	3.0.0	

### 3.4.2 Pallet jack

Only for MiR1200 Pallet Jack

Use this page to control the forks on a pallet jack robot.



Description	Description
<p>1 <b>Current fork height</b> View the current height of the fork from the ground in mm.</p>	<p>2 <b>Set fork height</b> Enter the height you want to move the forks to and select <b>Go</b> to make the robot move the forks to the entered height in mm.</p>
<p>3 <b>Fork control</b> Make the forks move up or down.</p>	<p>4 <b>Cancel</b> Cancel any movement the forks are executing.</p>
<p>5 <b>Fork speed</b> Adjust how fast the forks move up or down.</p>	

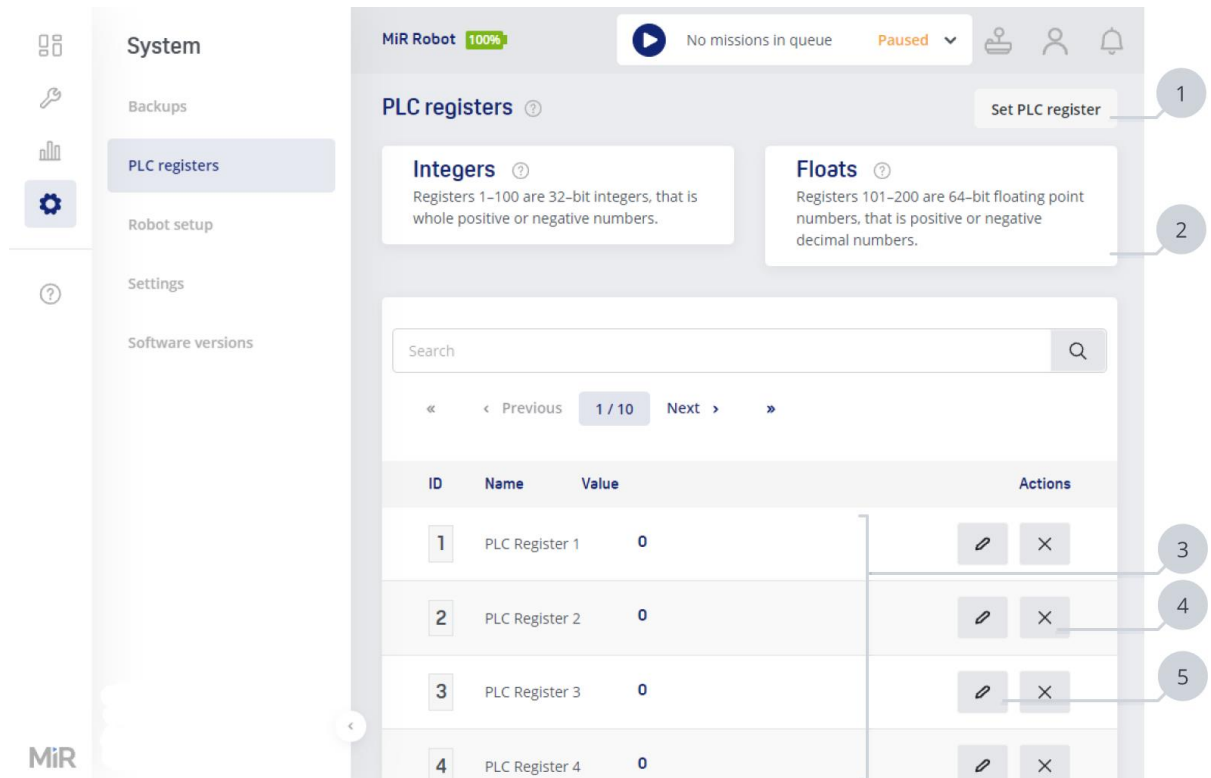
### 3.4.3 PLC registers

You must enable PLC registers under **System > Settings > Features** to see this page.

PLC registers store numbers that you can use or manipulate in missions or maps to control top modules or other connected devices or to trigger an action in a mission.

PLC registers can be accessed through a serial interface using the robot's USB port (via RS232 adapter) or through a REST interface using the robot's Ethernet connection.

You can control or read PLC registers using PLC actions in missions, I/O module zones on maps, or the PLC register widget on a dashboard.

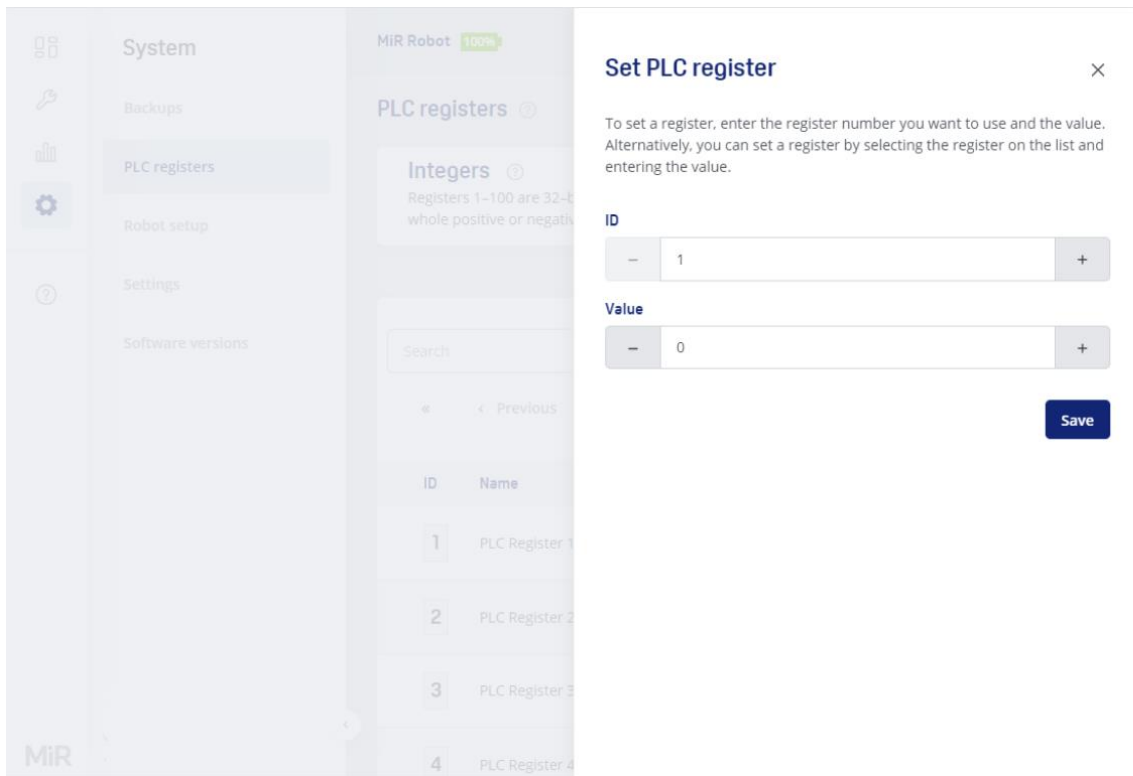


Description	Description
<p>1 <b>Set PLC register</b></p> <p>Opens the dialog box to set a PLC register—see <a href="#">"Set PLC register" on the next page</a>.</p>	<p>2 <b>Integer and Float information</b></p> <p>The PLC registers are divided between floats and integers.</p> <p><b>Integers:</b> Registers 1–100 are 32-bit integers, that is whole positive or negative numbers.</p>

Description	Description
	<p><b>Floats:</b> Registers 101–200 are 64-bit floating point numbers, that is positive or negative decimal numbers.</p>
<p>3 <b>PLC registers</b></p> <p>Lists all PLC registers available on the robot. You can sort the registers based on the ID, name, or values they are set to.</p>	<p>4 <b>Reset</b></p> <p>Reset the PLC register to its default settings.</p>
<p>5 <b>Edit</b></p> <p>Opens the dialog box to edit the selected PLC register—see <a href="#">"Edit PLC register"</a> on the next page.</p>	

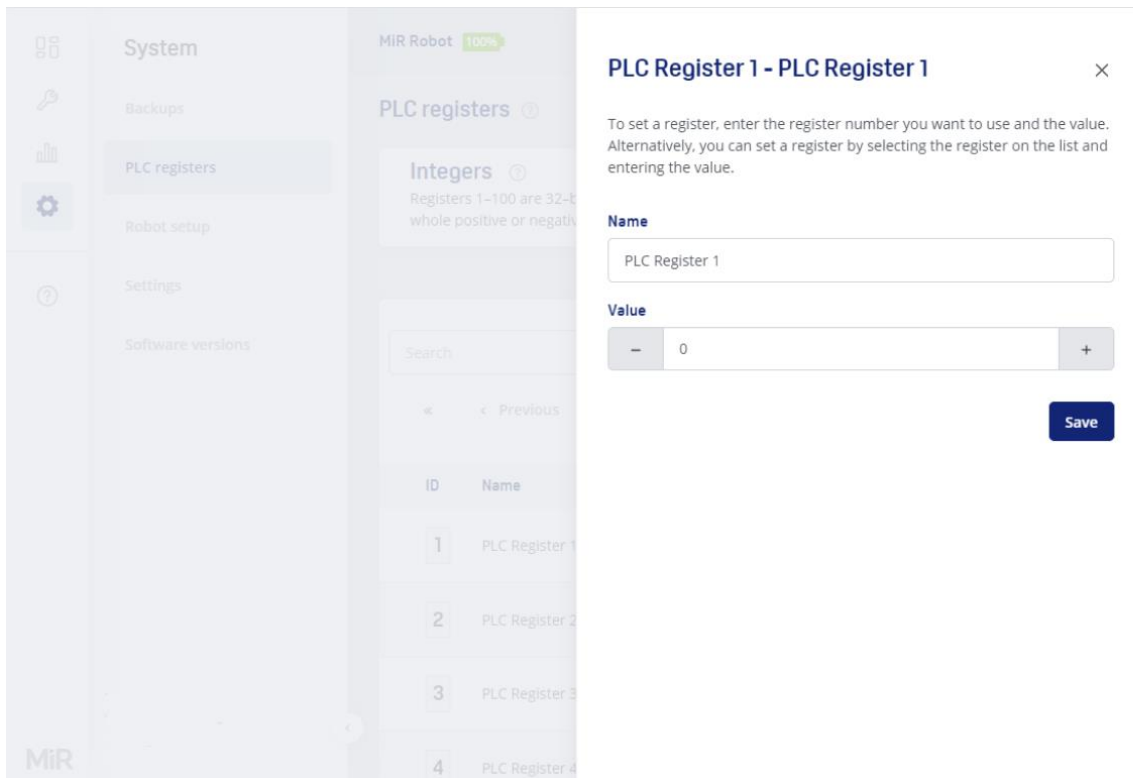
### Set PLC register

To set any of the PLC registers on the robot, first enter the ID for the register you want to set, then enter the value you want to change the register to. Select **Save** to set the register to the new value.



### Edit PLC register

To edit a PLC register, first change the name or value of the PLC register, then select **Save**.



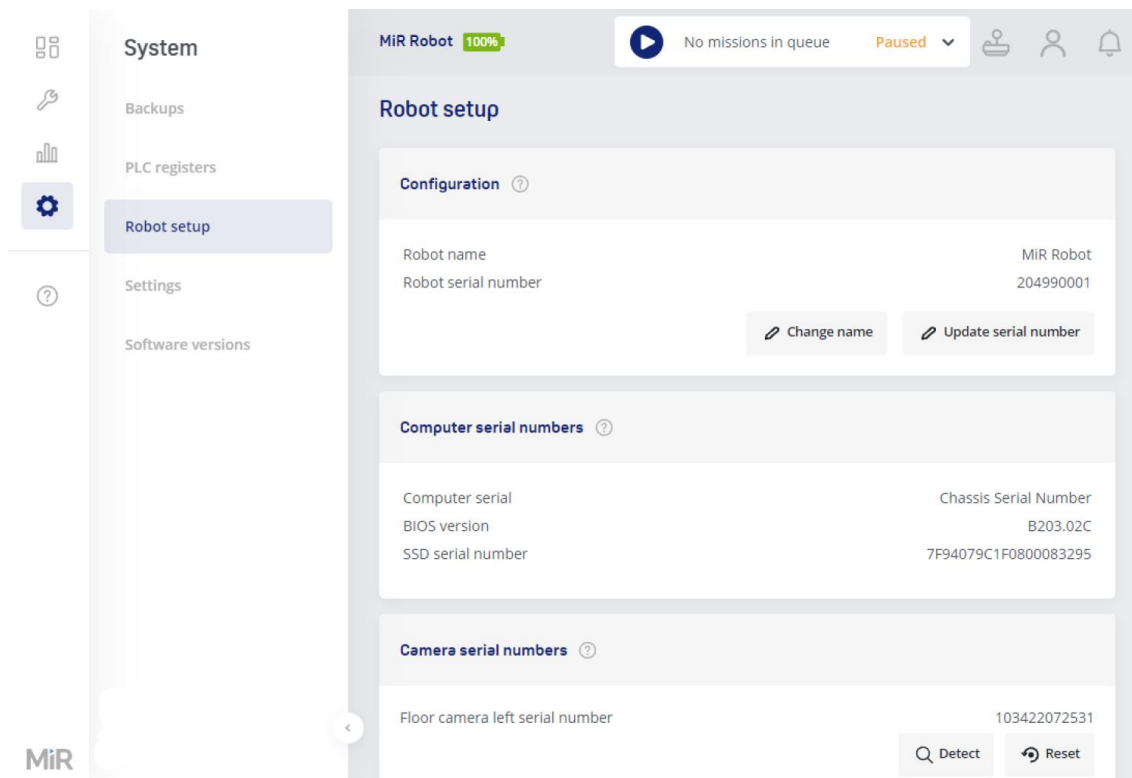
### 3.4.4 Robot setup

In the Robot setup section, you can:

- View and update the serial numbers of the robot and its components
- Run calibrations of IMU and encoders
- Enable the charging relay
- Download SICK configuration files.

Not all options are available for all robot models.



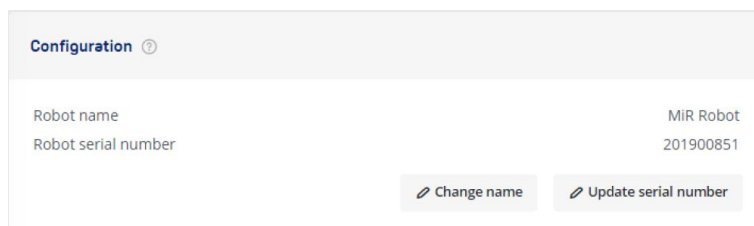


### Updating and detecting serial numbers

The following setup settings are used to view, update, or detect the serial numbers of the robot or its components.

#### Configuration

The **Robot name** section gives you the option of changing the name of the robot and reading or updating the serial number.



### Computer serial numbers

The **Computer serial numbers** section lists the serial numbers of the robot's hardware components.

Computer serial numbers ⓘ	
Computer serial	GEMY84300J4F
BIOS version	MYBDWi30.86A.0044.2017.1130.1251
SSD serial number	D702078610EE05224987

### Laser scanner serial numbers

MiR100 and MiR200 only

The **Laser scanner serial numbers** section contains the serial numbers of the front and rear laser scanners and the functions for detecting and swapping the two scanners. **Detect** identifies the serial numbers of the two scanners connected to the robot. Only use this if a scanner has been disconnected or replaced. **Swap** swaps the front and rear laser scanners. Use this only if you can see that the data from the laser scanners have been inverted so obstacles in front of the robot are shown to be detected behind the robot.

Laser scanner serial numbers ⓘ	
Front laser serial number	AL04PV7V
Back laser serial number	AL04PV7T
<input type="button" value="Q Detect"/> <input type="button" value="↔ Swap"/>	

### Camera serial numbers

The **Camera serial numbers** section contains the serial numbers of the cameras and the functions for detecting the serial numbers automatically. **Detect** identifies the serial numbers of the connected camera. If there are more than one camera listed, you must disconnect any other cameras from the robot computer before you can detect the serial number correctly. **Reset** resets the serial number to the default value.

Camera serial numbers ⓘ			
Floor camera left serial number	944122071423	<input type="button" value="Detect"/>	<input type="button" value="Reset"/>
Floor camera right serial number	944122073686	<input type="button" value="Detect"/>	<input type="button" value="Reset"/>

[Go to camera settings](#)

## Calibrating sensors

The following setup settings are used to calibrate the sensors in the robot.

### Inertial measurement unit

**Inertial measurement unit** (IMU) section lets you calibrate the IMU's 360 degree rotation. You must calibrate the IMU if you have replaced the robot's drive wheels, a motor, an encoder, the MiR board (MiR100 and MiR200 only), the motor controller carrier board or motor controller, the robot computer, or the wiring harness connected to the motor encoders.

To calibrate the IMU, select **Calibrate** and make sure the robot has enough space to rotate around itself. The robot will start spinning on the spot while the progression of the calibration is shown in percentage. After a couple of minutes, the calibration is finished and you get to decide if you want to keep the new calculated value. If the value deviates significantly from the original one, it will show in red color and you can choose to discard the calibration and restore to the default value.

For detailed instructions on how to calibrate the IMU, see the guide *How to calibrate the IMU*.

To perform the calibration, the robot must have an active map.

Inertial measurement unit ⓘ	
Gain	1.00000

### Laser scanner calibration

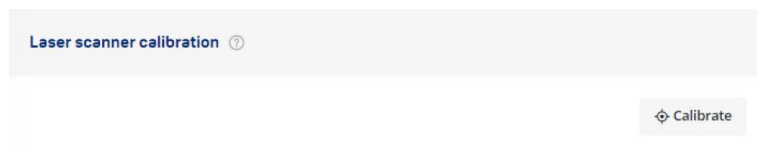
MiR100 and MiR200 only

The **Laser scanner calibration** section lets you calibrate the laser scanners. You must calibrate the laser scanners if you feel like the scanner data is inaccurate, or you have replaced a laser scanner, the robot computer, or the safety PLC.

To calibrate a laser scanner, place the robot approximately two meters in front of a wall, and select **Calibrate**. The robot now moves to a start position. Measure the distance from the front of the laser scanner to the wall and enter the distance in the dialog box in the interface. Follow the instructions in the interface, until the calibration is completed.

For detailed instructions on how to calibrate the scanners, see the guide *How to calibrate the safety laser scanners*.

After a calibration, the robot must be restarted.

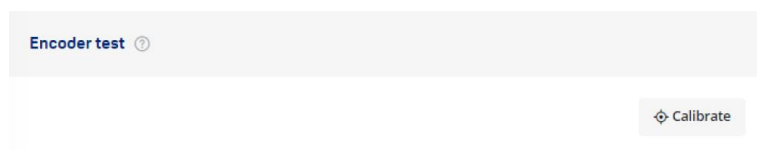


### Encoder test

The **Encoder test** runs the robot through a sequence where it moves forward at both a slow and a faster speed and then moves back again. If you feel like your robot does not localize or dock well, it is a good idea to run this test to see if the robot's encoders are poorly calibrated.

Select **Test** to start the test. While the robot is moving, stand behind it and observe whether the robot is drifting slightly to the left or right. You are prompted to enter your observations in the test dialog box.

When the test is finished, the results from the test are displayed in a table. The results include notes on what you observed and the number of recorded encoder ticks for each drive wheel. If the robot has a tendency to drift to either side, but the number of encoder ticks recorded for each drive wheel is almost the same, it indicates that there is an issue with the encoders. If the encoder ticks represent the robot's driving behavior accurately, there may be another issue with the robot. Contact your distributor for further assistance.



### Laser scanner transform and drive wheels

The **Laser scanner transform and drive wheels** section displays the current calibration of the robot's scanners and wheels. If the scanners or wheels have been replaced, you must calibrate the transforms.

Before starting the calibration, you will need to make a small mission with three positions where the robot loops in a figure eight pattern. Start the mission, and then select **Calibrate** and wait until the calibration is completed (10-15 minutes).

When finished, decide if you want to keep the new calculated values. If any values deviate significantly from the original ones, they will show in red color and you can choose to discard the calibration and restore to the default values.

Laser scanner transform and drive wheels ⓘ	
Front laser scanner transform: X	0.4200
Front laser scanner transform: Y	0.2380
Front laser scanner transform: Yaw	0.7854
Back laser scanner transform: X	-0.3650
Back laser scanner transform: Y	-0.2380
Back laser scanner transform: Yaw	-2.3562
Diameter of the towing wheels	0.1250
Robot wheelbase length	0.4510

**Calibrate**

### Proximity sensors

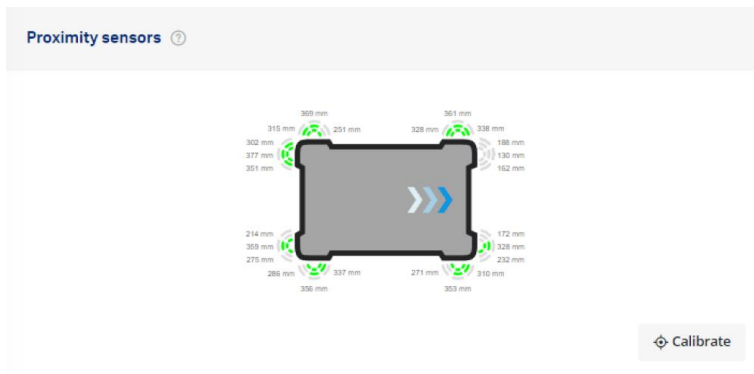
MiR250, MiR500, MiR600, MiR1000, and MiR1350 only

The **Proximity sensors** section lets you calibrate the proximity sensors. To calibrate, place the robot on a level floor and clear all obstacles in the area around the robot.

In the depiction of the robot, the current values shown next to each proximity sensor are the distances in millimeters between the proximity sensors and the nearest detected objects. If the sensors are well-calibrated, the values should be close to 365 mm when the robot is driving on a flat surface with no nearby obstacles. Select **Calibrate** and wait for the robot to determine the calibration offset values.

The offset values are displayed next to the values for the distance between the floor and the sensors. The offset values should result in the distances being around 365 mm. Evaluate whether the calibration corrections seem valid, and select **Apply** or **Cancel** depending on the calibration.

For detailed instructions on how to calibrate the proximity sensors, see the guide *How to calibrate the proximity sensors*.



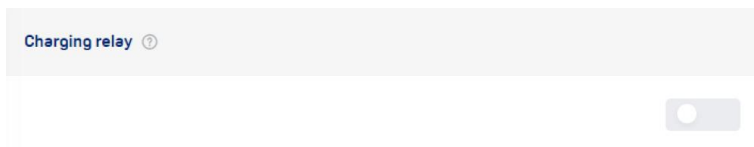
### Other functions

Depending on your robot model, there are also other functions available under Robot setup.

### Charging relay

MiR100 and MiR200 only

The **Charging relay** section contains the option of turning the robot's internal charging relay on or off. The charging relay is used when automatic charging is carried out by connecting the robot to a charging station.



### SICK configuration files

MiR250, MiR500, MiR600, MiR1000, and MiR1350 only

The **SICK configuration files** section lets you download MiR's supported SICK configurations for various robot applications.



### 3.4.5 Settings

Under the robot settings, you can configure the following:

- **3D cameras:** Use the 3D camera settings to modify your camera configuration. It is important that you enter the correct camera setup, type, and serial number to ensure the robot handles and accesses the camera data correctly. The serial numbers in these settings can also be accessed and changed in the **System > Robots setup** section. For more information about troubleshooting the 3D cameras, see the guide *Troubleshoot 3D cameras not working*.
- **Battery usage:** Use the battery settings to modify what kind of battery the robot is using and change at which point the charger begins topping up the battery. When the battery is being topped up, it means the charger cycles between charging and discharging when the battery is almost fully charged. By default, the battery percentage cycles between 90% and 100% when the robot is topping up the battery.
- **Calibration:** Use the calibration settings to change how the robot docks to all markers of a specific type. You should only use these offsets if you notice that certain markers always need the same adjustment for a specific robot. For more information about the calibration values, see the guide *How to change docking parameters*.
- **Date and time:** Edit the date and time.
- **Distributor data:** Enter data about the distributor supplying the product.
- **Docking behavior:** Use the docking settings to modify how the robot docks to and undocks from markers. For more information about how the robot docks and undocks, see the user guide for your robot.
- **Error handling behavior:** Use the error handling settings to change the thresholds that trigger certain errors on the robot. We do not recommend modifying these to values where hardware errors on the robot are ignored, unless you are recommended to do so by MiR Technical Support due to unusual circumstances.
- **Features:** Use the feature settings to enable the features you want to use on your robot. Enabling features provides access to new sections of the interface, access to actions you can add to missions, or automatically maps how the electrical interfaces link to certain functions on the robot.
- **Localization :** Use the localization setting to adjust how the localization algorithm runs on the robot. This can be used to increase how uncertain the robot's localization can be before the robot reports an error, and to make the localization algorithm better at accounting for slopes.
- **Motor controller:** Use the motor controller settings to change the gear ratio of the robot and to modify the thresholds before the motor controller can trigger an error.

- **Planner behavior:** Use the planner settings to modify how the robot plans and follows its paths. This can be used if the robot often does not plan or follow paths as you would expect. Not all issues are best solved by modifying the planner settings. Often you can use zones and positions to optimize the robot's path planning. For more information about how to modify the settings to enable Line-following mode, see the guide *How to enable Line-following mode*. You can find this guide on [MiR Support Portal](#).
- **Proximity sensors:** Use the proximity sensor settings to enable or disable the proximity sensors on the robots (also called ultrasound sensors on MiR100 and MiR200).
- **UR connection:** Use the UR settings to enter the IP address of a connected Universal Robots arm.
- **Wi-Fi connections:** Use the Wi-Fi settings to view and modify the wireless networks on the robot. For more information about connecting the robot to a network, see ["Add robots" on page 276](#).
- **WISE configuration:** Use the WISE configuration setting to enter a password and username the robot uses when it connects to a WISE module. You can only save one set of credentials, so all WISE modules the robot must connect to must share the same credentials.

### 3.4.6 Software versions

A software update file for a MiR robot is a .mir file that contains the software for MiR robots. There are two types of robot software: Application software and Platform software. The Platform software is the operating system software that the Application software runs on. The Application and Platform software control different aspects of how the robot operates.


The software versions page lets you update the robot to different versions of the MiR software. All software versions you have uploaded to the robot are shown in the list and you can choose to update to the newest software on the list.

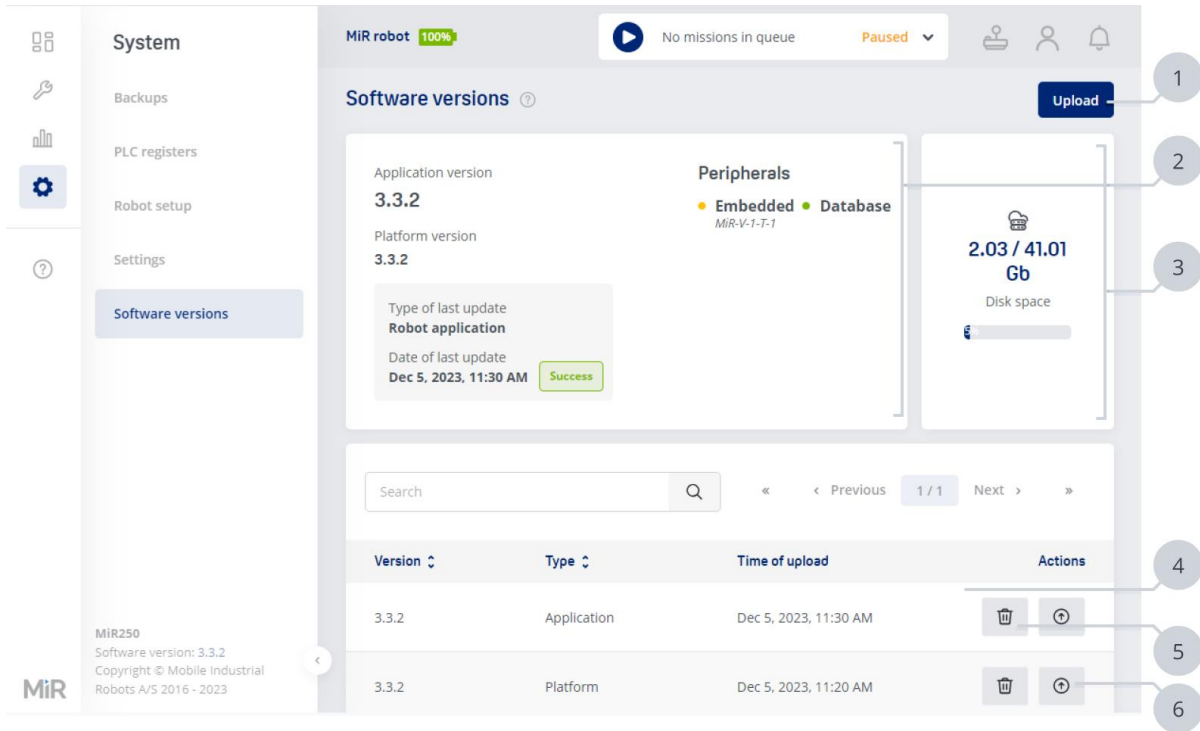
To add a new software version, select **Upload**, and select the software file on your computer.

When you want to switch to another software version, select **Update** next to the file. When the robot finishes updating, restart the robot, and sign in to the interface again. The robot is now ready to operate with the new software version.

If you have any issues during the update process, see the guide *How to upgrade a MiR robot's software*.



 If a hook is mounted on the robot, the hook must be running the same software version. Go to **Hook > Software versions** and update the hook.

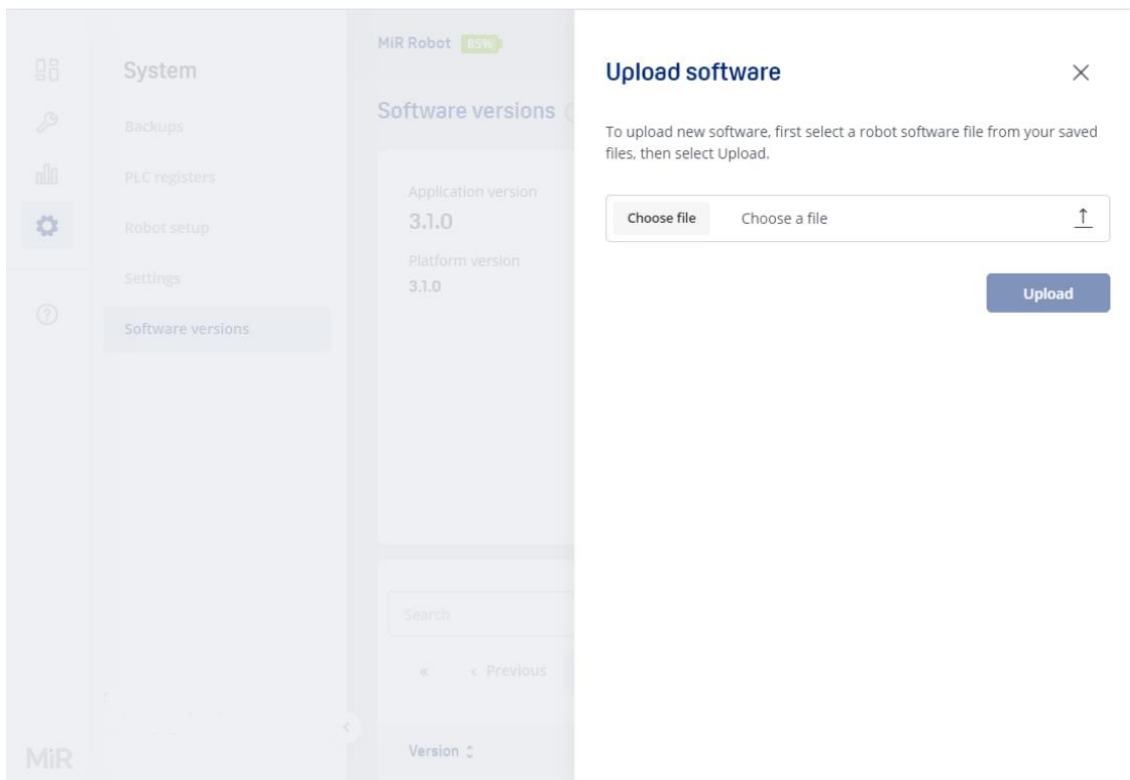


Description	Description
<p>1 <b>Upload</b></p> <p>Opens the dialog box to upload a new software version—see "<a href="#">Upload new software</a>" on the next page.</p>	<p>2 <b>Current version information</b></p> <p>Shows information about the current software version on the robot and the peripherals, and the last time the robot was updated.</p>
<p>3 <b>Disk space</b></p> <p>Shows how much disk space is left on the robot. If you are running out of disk space, consider removing old software versions or unnecessary backups from</p>	<p>4 <b>Software versions</b></p> <p>Lists all software versions that have been applied to . You can sort the software versions based on the version, type, or when they were uploaded.</p>

Description	Description
the robot.	
<p>5 <b>Delete software version</b></p> <p>Deletes the software version from the robot. If you still have the software file on another device, you can always upload it to the robot again.</p>	<p>6 <b>Update software</b></p> <p>Updates the robot to the selected software version.</p>

### Upload new software

To upload new software, first select a robot software file from your saved files, then select Upload.



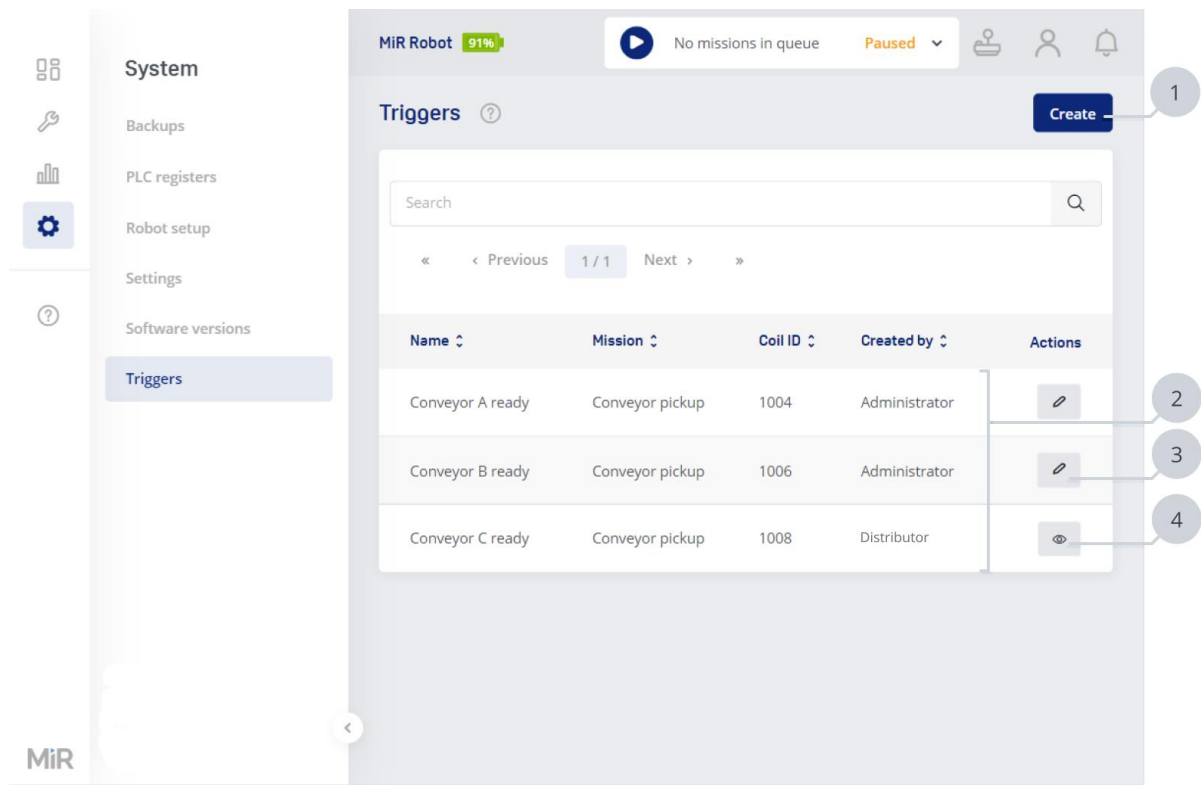
### 3.4.7 Triggers

You must enable Modbus under **System > Settings > Features** to see this page.

A trigger is a link that lets you define what mission should be added to the robot queue when a specific Modbus coil is active. Triggers enable remote devices to add missions to the robot's mission queue.

To setup Modbus communications between the robot and other devices, see the guide *How to use Modbus with MiR robots*.

Robots that are running MiR Fleet Enterprise do not support any of the mission triggers. All missions must be assigned to robots through MiR Fleet.

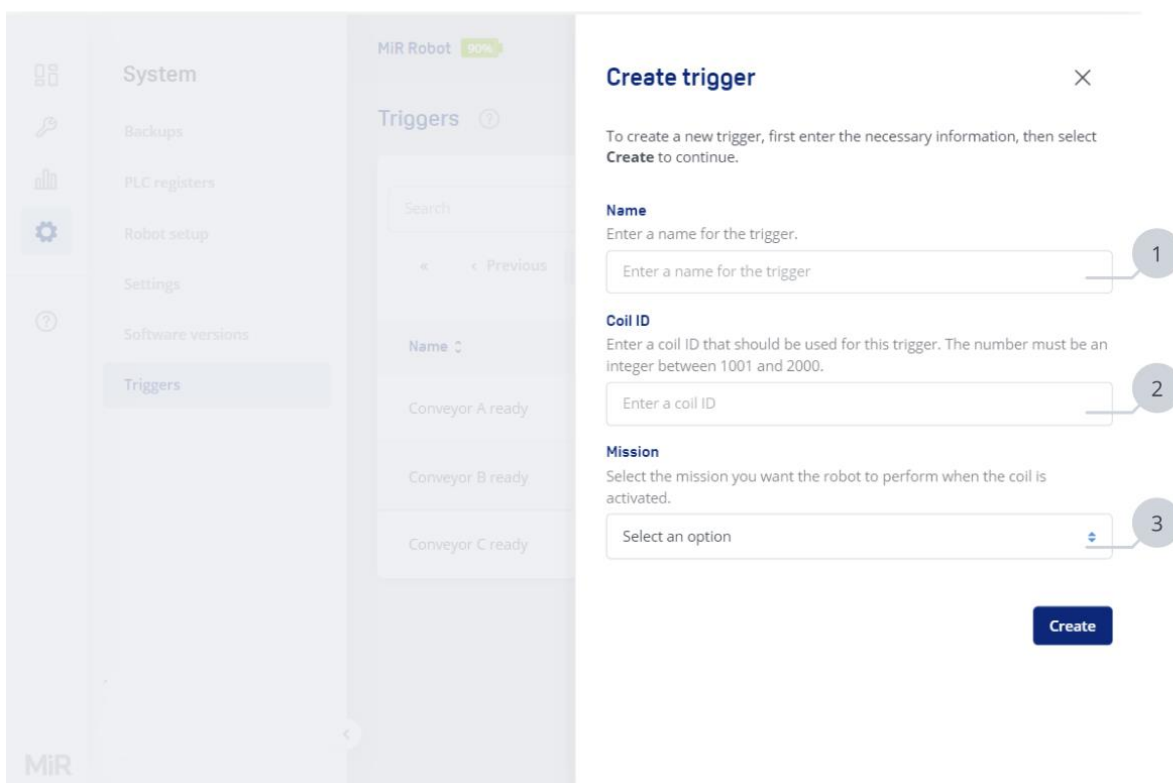


Description	Description
<p>1 <b>Create</b></p> <p>Opens the dialog box to create a new trigger—see "<a href="#">Create trigger</a>" on the next page.</p>	<p>2 <b>Triggers</b></p> <p>Lists all triggers used by the robot. You can sort the triggers based on the name, the mission it triggers, the coil ID it is linked to, or which user created them.</p>

Description	Description
<p>3 <b>Edit</b></p> <p>Opens the dialog box to edit the selected trigger—see <a href="#">"Edit trigger" on the next page.</a></p>	<p>4 <b>View</b></p> <p>Opens the dialog box to view information about the selected trigger. Your user group does not have permission to edit the trigger.</p>

### Create trigger

To create a new trigger, first enter the necessary information, then select **Create** to continue.



Description	Description
<p>1 <b>Name</b></p>	<p>2 <b>Coil ID</b></p>

Description	Description
Enter a name for the trigger.	Enter a coil ID that should be used for this trigger. The number must be an integer between 1001 and 2000.
<p>3 <b>Mission</b></p> <p>Select the mission you want the robot to perform when the coil is activated.</p>	

## Edit trigger

To edit a trigger, first edit the necessary information, then select **Save** to continue. Select **Delete** to delete the selected trigger.

The screenshot shows the MiR Robot interface with the 'Edit trigger' dialog box open. The background shows the 'System' menu with 'Triggers' selected. The dialog box has a title bar with a close button (X). Below the title bar, there is a description: 'To edit a trigger, first edit the necessary information, then select **Save** to continue. Select **Delete** to delete the selected trigger.' The dialog contains four sections:

- Name**: 'Edit the name of the trigger.' with a text input field containing 'Conveyor A ready'.
- Coil ID**: 'Enter a coil ID that should be used for this trigger. The number must be an integer between 1001 and 2000.' with a text input field containing '1004'.
- Mission**: 'Select the mission you want the robot to perform when the coil is activated.' with a dropdown menu showing 'Conveyor pickup'.
- Position**: 'Conveyor A pickup' with a dropdown menu showing 'Conveyor A pickup'.

At the bottom of the dialog, there are two buttons: a red 'Delete' button and a blue 'Save' button.

You can change any of the fields described in ["Create trigger" on the previous page](#).

## 3.5 Help

The Help menu contains pages that let you view information about the robot and its API.

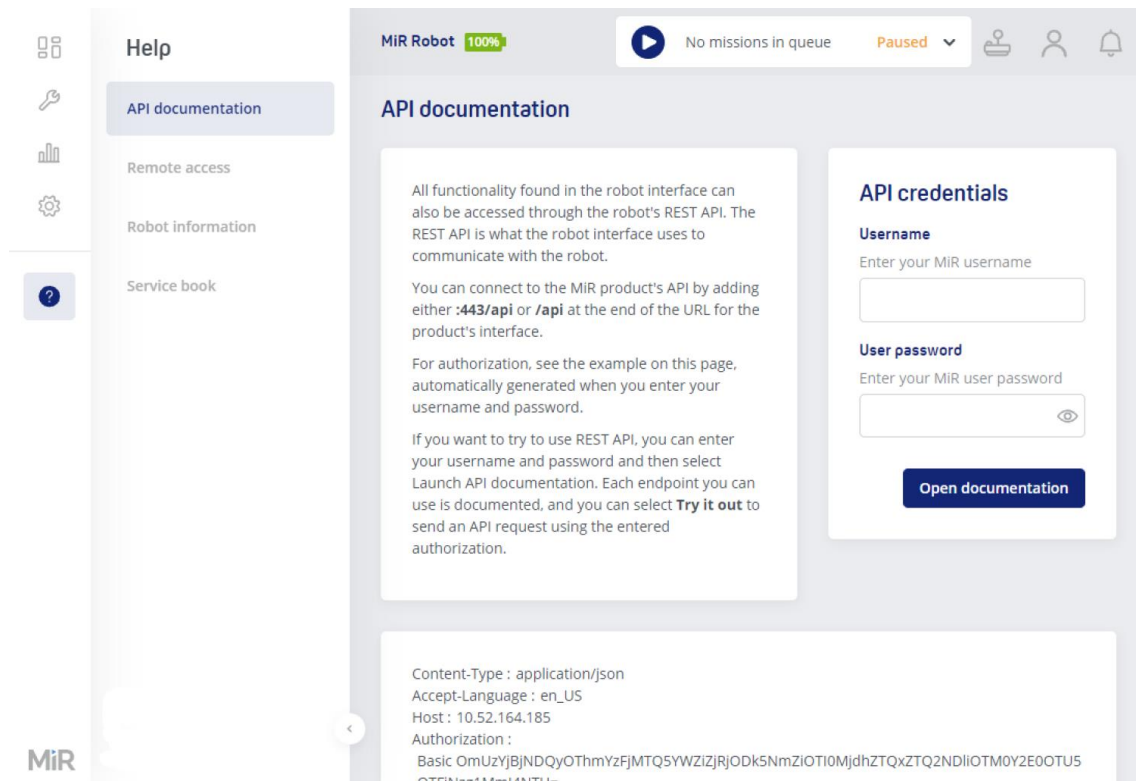
### 3.5.1 API documentation

All functionality found in the interface can also be accessed through the REST API. The REST API is what the interface uses to communicate with the product.

You can connect to the MiR product's API by adding either **:443/api** or **/api** at the end of the URL for the product's interface.

For authorization, see the example on this page, automatically generated when you enter your username and password.

If you want to try to use REST API, you can enter your username and password and then select **Launch API documentation**. Each endpoint you can use is documented, and you can select **Try it out** to send an API request using the entered authorization.



The screenshot displays the MiR Robot interface. At the top, a status bar shows 'MiR Robot 100%' and 'No missions in queue' with a 'Paused' dropdown menu. The left sidebar contains a 'Help' menu with options: 'API documentation' (selected), 'Remote access', 'Robot information', and 'Service book'. The main content area is titled 'API documentation' and contains the following text:

All functionality found in the robot interface can also be accessed through the robot's REST API. The REST API is what the robot interface uses to communicate with the robot.

You can connect to the MiR product's API by adding either **:443/api** or **/api** at the end of the URL for the product's interface.

For authorization, see the example on this page, automatically generated when you enter your username and password.

If you want to try to use REST API, you can enter your username and password and then select **Launch API documentation**. Each endpoint you can use is documented, and you can select **Try it out** to send an API request using the entered authorization.

Below the text is a form for 'API credentials' with fields for 'Username' and 'User password', and an 'Open documentation' button. At the bottom, there is a code block showing headers:

```
Content-Type : application/json
Accept-Language : en_US
Host : 10.52.164.185
Authorization :
Basic OmUzYjBjNDQyOThmYzFjMTQ5YWZlZjRjODk5NmZlOTI0MjdhZTQxZTQ2NDI0OTM0Y2E0OTU5
```

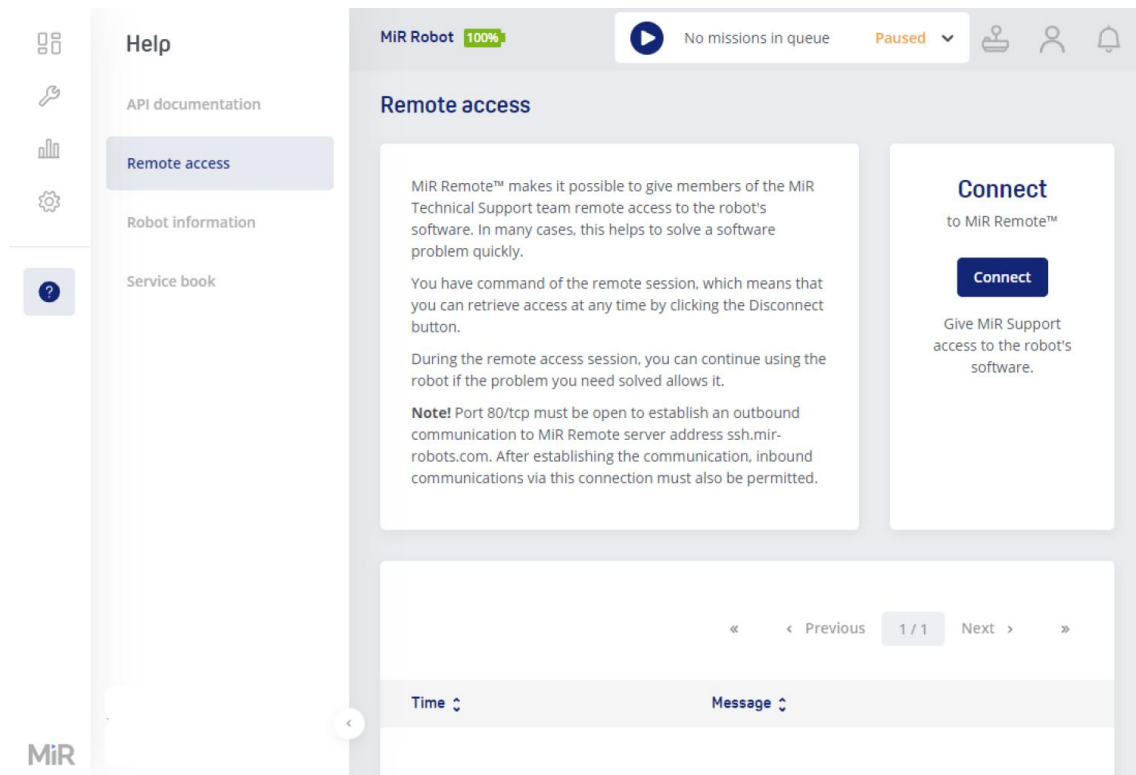
### 3.5.2 Remote access

MiR Remote™ makes it possible to give members of the MiR Technical Support team remote access to the robot's software. In many cases, this helps to solve a software problem quickly.

You have command of the remote session, which means that you can retrieve access at any time by clicking the Disconnect button.

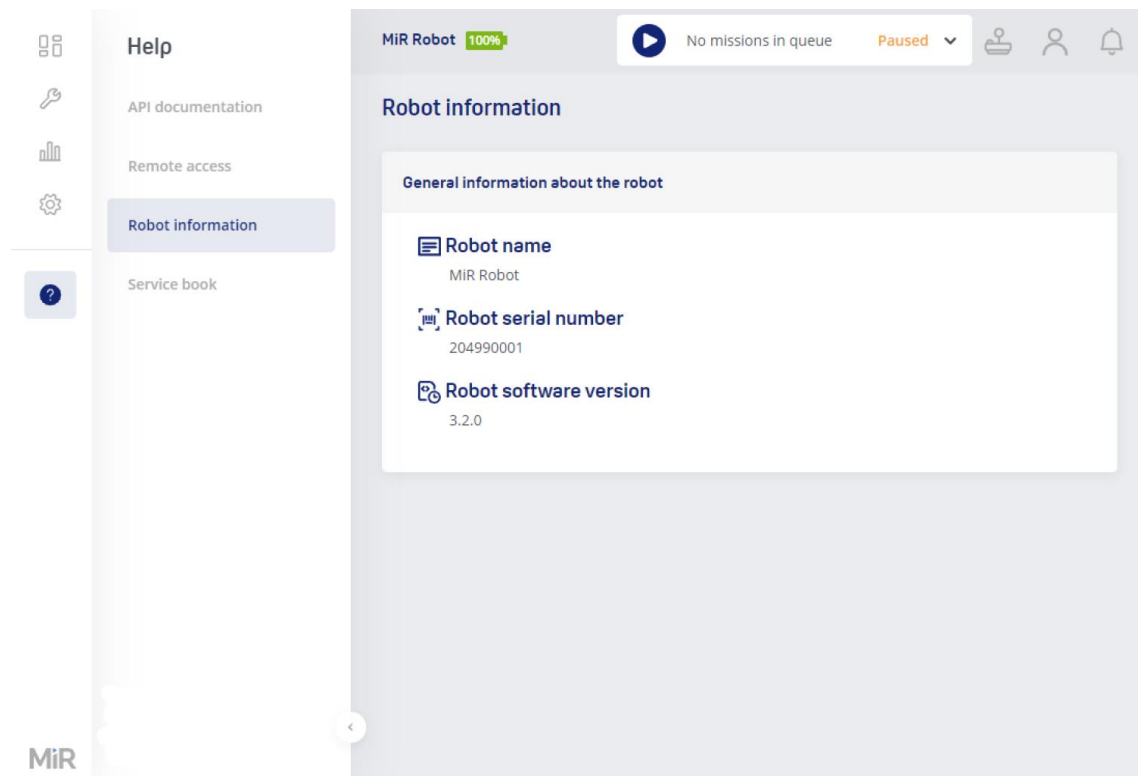
During the remote access session, you can continue using the robot if the problem you need solved allows it.

Port 80/tcp must be open to establish an outbound communication to MiR Remote server address `ssh.mir-robots.com`. After establishing the communication, inbound communications via this connection must also be permitted.



### 3.5.3 Robot information

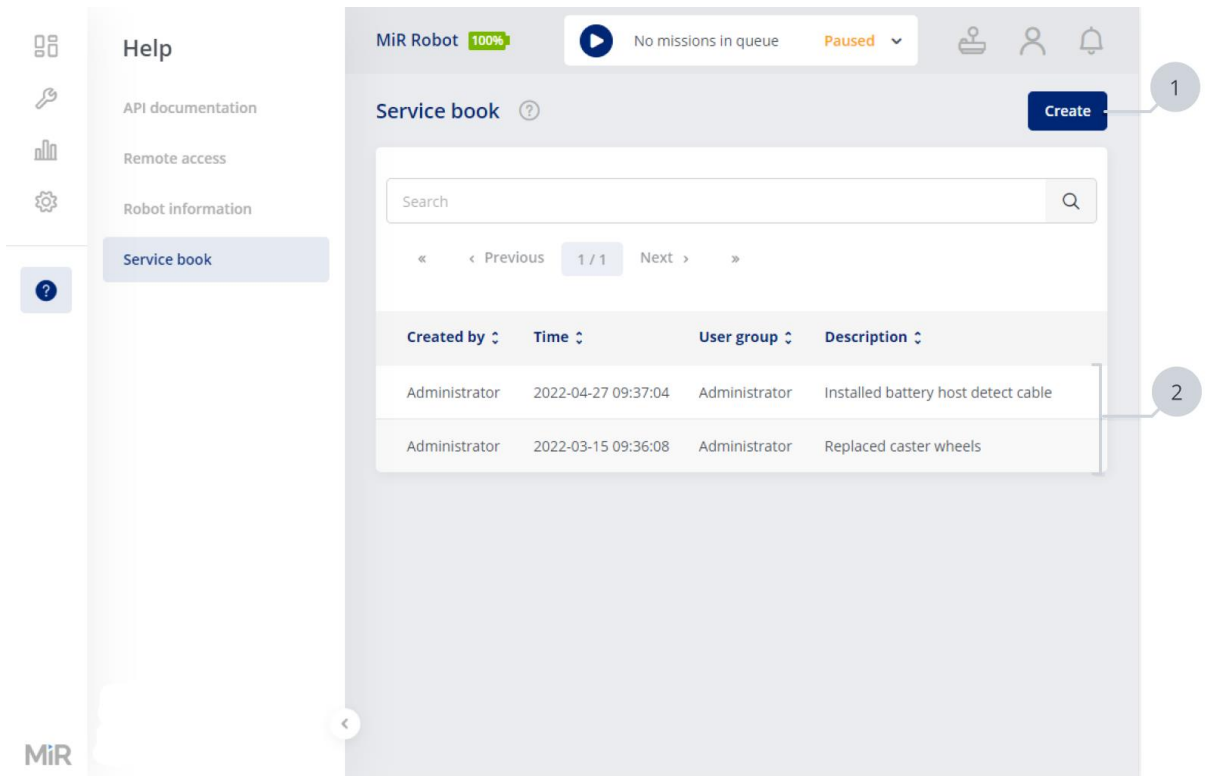
See the name, serial number, and software version of your robot.



### 3.5.4 Service book

In the Service book you can enter notes about the robot, for example, about changes made in the robot. The notes can be read by all user groups and cannot be deleted.

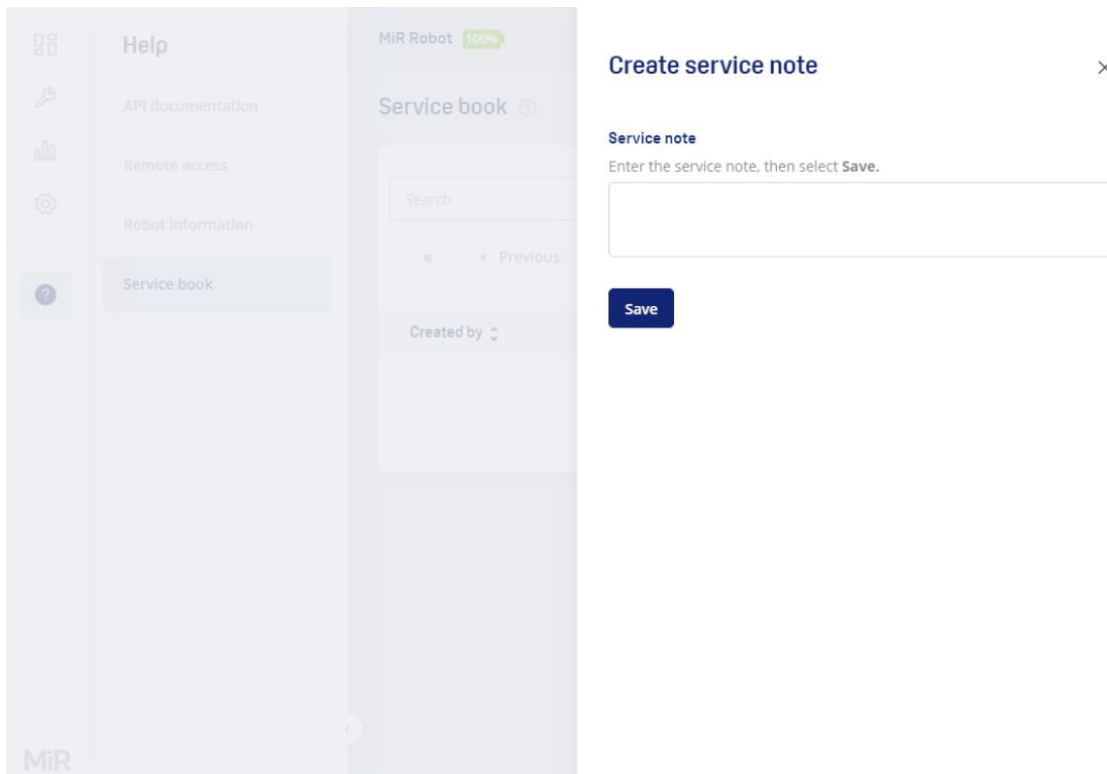




Description	Description
<p>1 <b>Create</b></p> <p>Opens the dialog box to create a new service note entry—see "<a href="#">Create service note</a>" below.</p>	<p>2 <b>Service notes</b></p> <p>Lists all service notes made for . You can sort the service notes based on which user created them, the time they were created, or the content.</p>

**Create service note**

Enter the service note, then select **Save**.



## 3.6 Hook

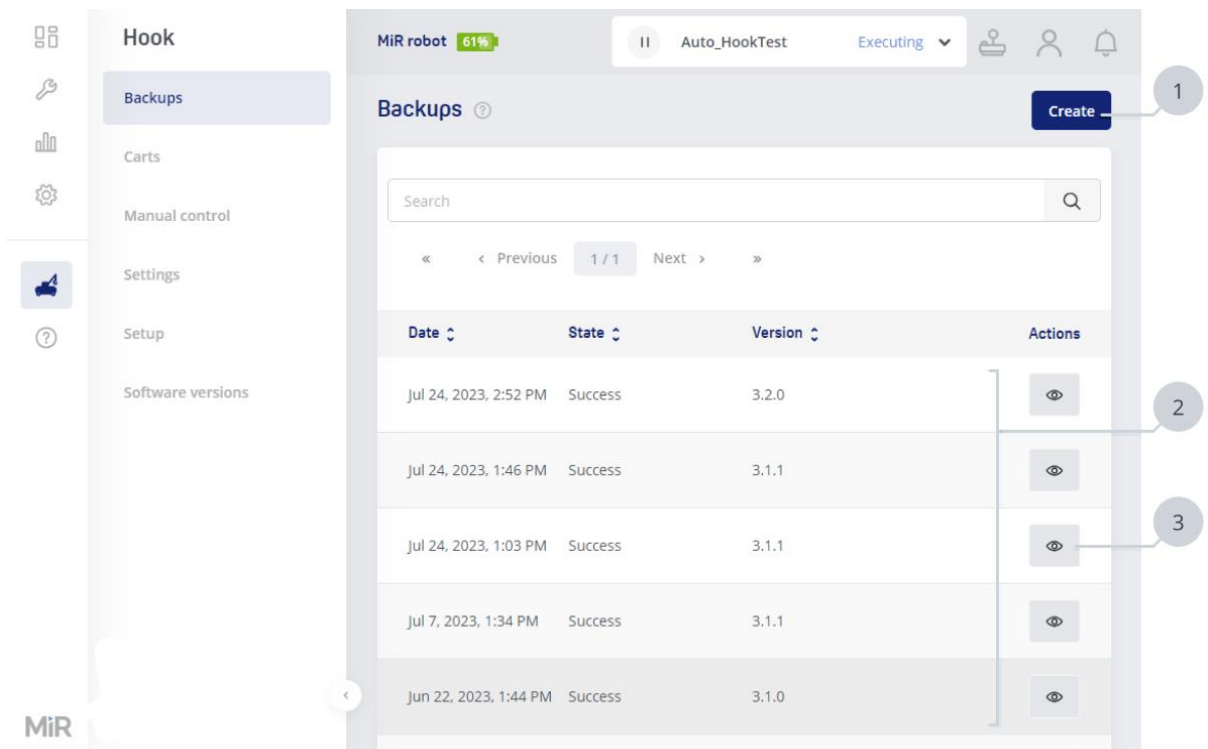
You must enable **Hook** under **System > Settings > Features** to see this menu.

The Hook menu contains the following pages that let you create, modify, and remove components for hook specific mission or lets you modify and view settings specifically for MiR Hook.

### 3.6.1 Backups

A hook backup is a copy of the configuration and system state data of the hook configuration. You can use a backup to revert your hook to a previous state. It contains the hook settings. This is most often used if the system fails and you need to recover lost data.

Hook backups are automatically created before you update the software version of the hook or roll back to a previous backup. You can also generate backups manually on this page.

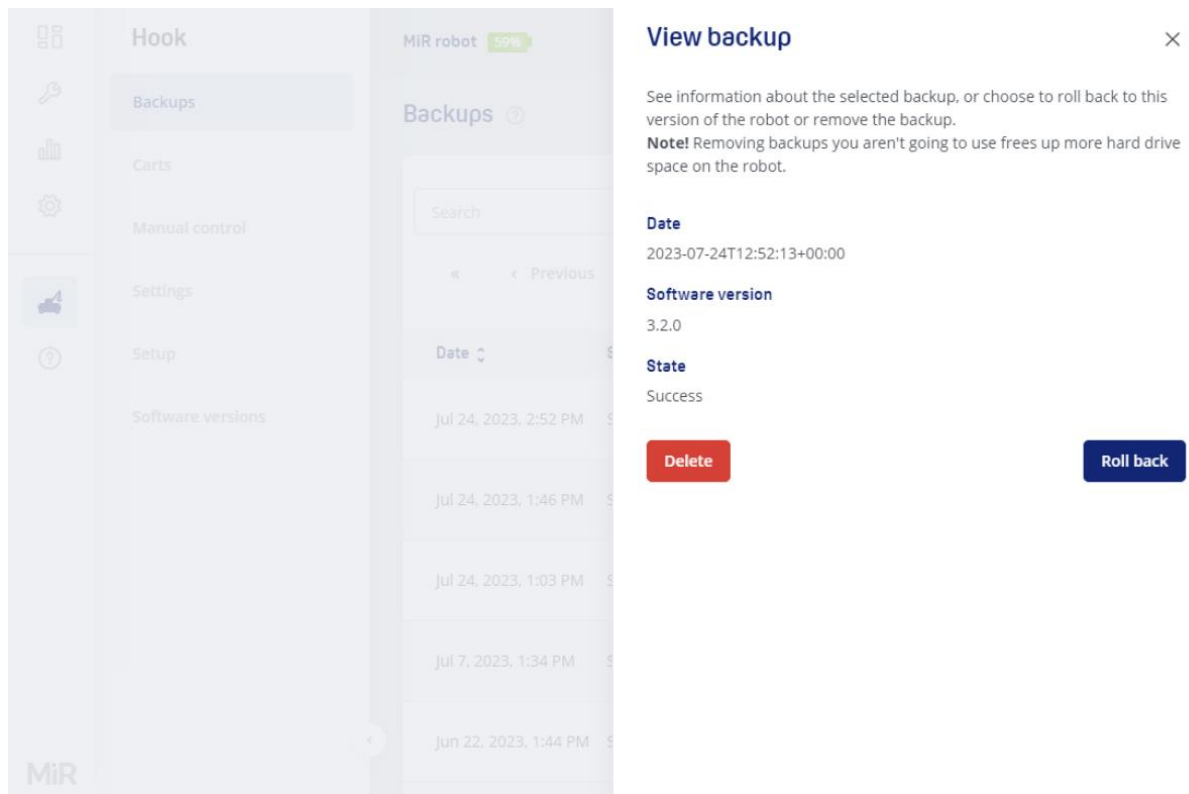


Description	Description
<p>1 <b>Create</b></p> <p>Creates a new backup of your hook's current settings. It may take the robot a few seconds to create the backup.</p>	<p>2 <b>Backups</b></p> <p>Lists all backups created for the hook. You can sort the backups based on the date they were created, the state, or the software version.</p>
<p>3 <b>View</b></p> <p>See more information about the selected backup, and choose to either remove or restore the backup—see <a href="#">"View backup" on the next page.</a></p>	

## View backup

See information about the selected backup, or choose to roll back to this version or remove the backup.

Removing backups you are not going to use frees up more hard drive space on the robot.

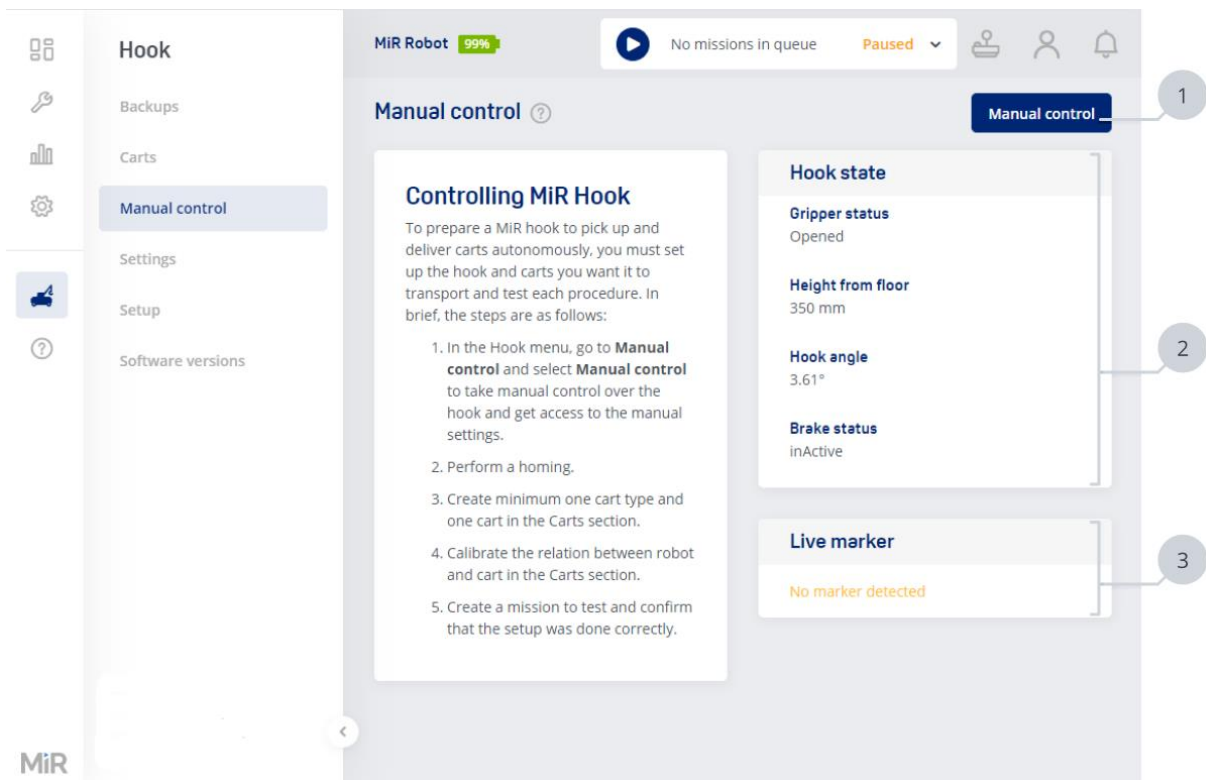


### 3.6.2 Manual control

To prepare a MiR hook to pick up and deliver carts autonomously, you must set up the hook and carts you want it to transport and test each procedure. In brief, the steps are as follows:

- 1 In the Hook menu, go to **Manual control** and select **Manual control** to take manual control over the hook and get access to the manual settings.

- 2 Deactivate the brakes and move the hook arm to the 0 position (so the gripper is directly behind the robot) and activate the brakes. Select **Home** to calibrate this as the home position.
- 3 Create a cart in the interface and calibrate the cart to the robot.
- 4 Create a mission to test and confirm that the setup was done correctly.



Description	Description
<p>1 <b>Manual control</b></p> <p>Opens the dialog box to control the hook manually—see "<a href="#">Manual control</a>" on page 151.</p>	<p>2 <b>Hook state</b></p> <p>Displays the current state of the hook—see "<a href="#">Hook state</a>" on the next page.</p>

Description	Description
3 <b>Live marker</b>  If the hook detects an ID tag, it will display the information about the tag here.	

### Hook state

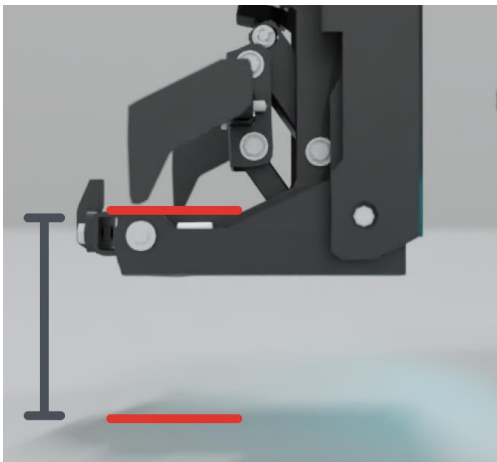
Under Manual control, you can see the following states of the hook:

- **Gripper status**

Indicates whether the gripper is opened or closed.

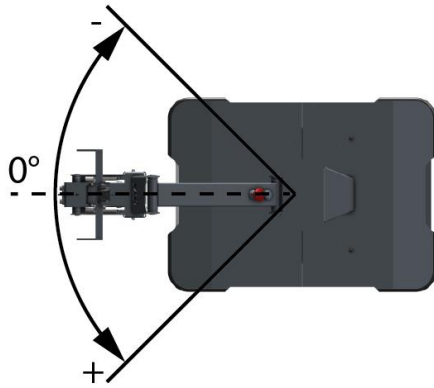
- **Height from floor**

The height of the hook from the floor is measured from the upper plane of the gripper.



- **Hook angle**

The angle in which the hook arm is currently positioned. When the arm is centered along the middle of the robot, the angle is 0. When the hook is oriented towards the left side of the robot, the angle is negative, and on the right side the angle is positive.

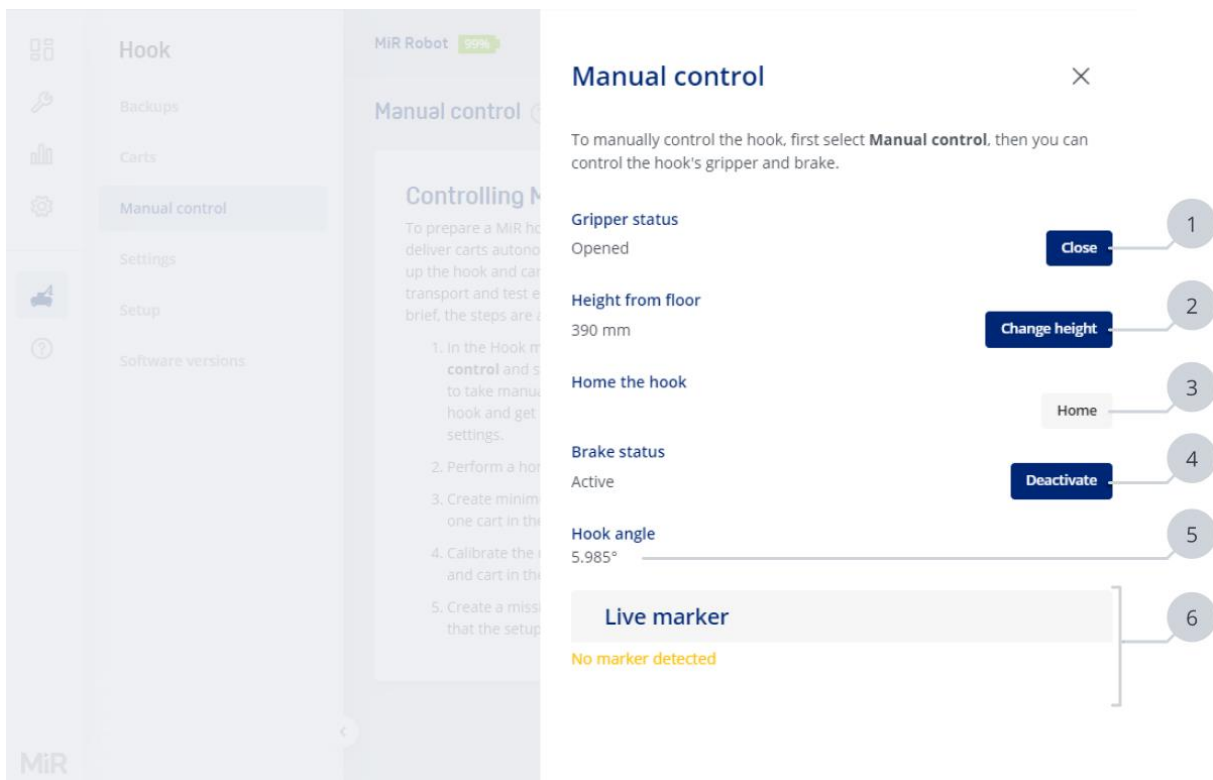


- **Brake status**

When the hook's brake is active, you cannot rotate the hook arm. When deactivated, the hook arm can swing freely.

### Manual control

To manually control the hook, first select **Manual control**, then you can control the hook's gripper and brake.



Description		Description	
1	<b>Gripper status</b> Closes and opens the hook gripper.	2	<b>Height from floor</b> Change the height of the hook. The distance is measured from the upper plane of the gripper—see <a href="#">"Hook state" on page 150</a> .
3	<b>Home the hook</b> Set the 0 point for the hook angle.	4	<b>Brake status</b> Deactivates the brake to let the hook swing freely, or activates the brake to stop the hook from moving to the sides.
5	<b>Hook angle</b>	6	<b>Live marker</b>



Description	Description
<p>The angle that the hook is currently positioned at. When the arm is centered along the middle of the robot, the angle is 0—see <a href="#">"Hook state" on page 150</a>.. When the hook is oriented towards the left side of the robot, the angle is negative, and on the right side the angle is positive.</p>	<p>If the hook detects an ID tag, it will display the information about the tag here.</p>

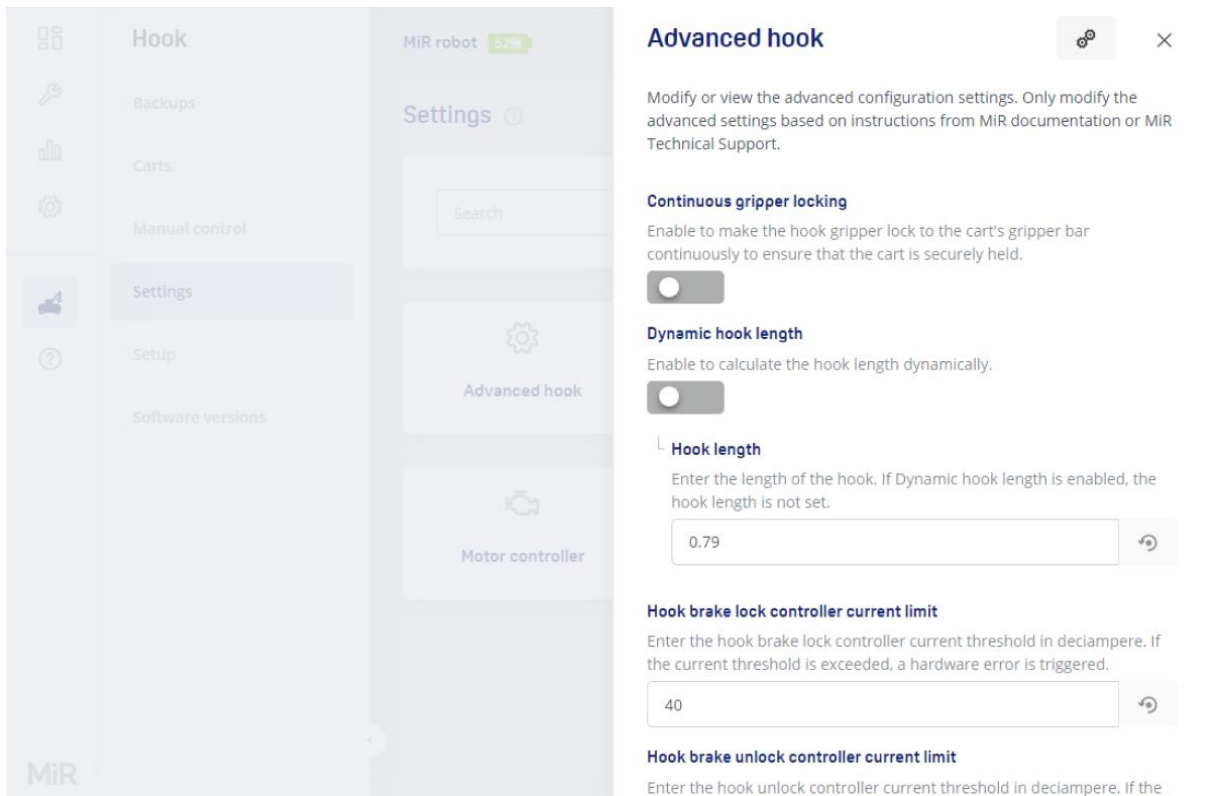
### 3.6.3 Settings

The Settings page contains all parameters you can modify to change the behavior of the hook top module. The settings are divided into the following categories:

#### Advanced settings

Modify or view the advanced configuration settings. Only modify the advanced settings based on instructions from MiR documentation or MiR Technical Support.

Use these settings to enable features that can improve your hook's performance. Only enable these features if recommended to do so by MiR Technical Support. The current limit can also be set in the Motor controller settings—see ["Motor controller" on page 156](#).



**Advanced hook**

Modify or view the advanced configuration settings. Only modify the advanced settings based on instructions from MiR documentation or MiR Technical Support.

**Continuous gripper locking**

Enable to make the hook gripper lock to the cart's gripper bar continuously to ensure that the cart is securely held.

**Dynamic hook length**

Enable to calculate the hook length dynamically.

**Hook length**

Enter the length of the hook. If Dynamic hook length is enabled, the hook length is not set.

**Hook brake lock controller current limit**

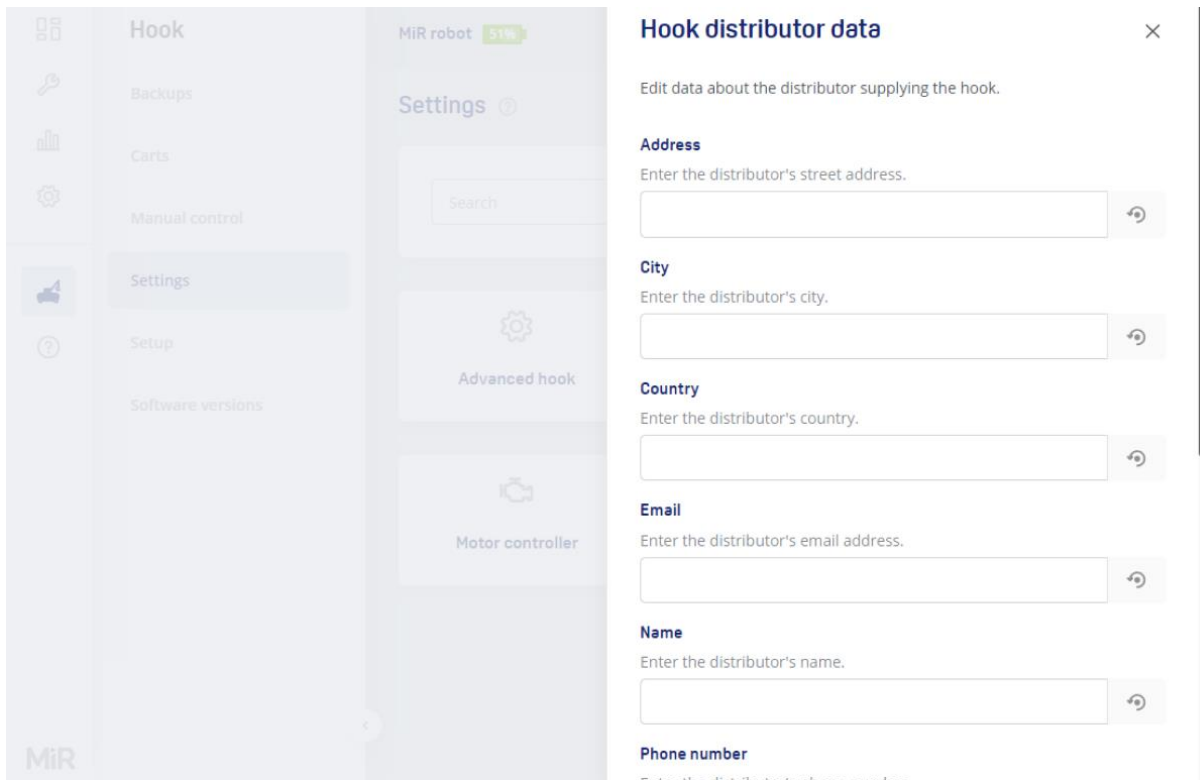
Enter the hook brake lock controller current threshold in deciampere. If the current threshold is exceeded, a hardware error is triggered.

**Hook brake unlock controller current limit**

Enter the hook unlock controller current threshold in deciampere. If the

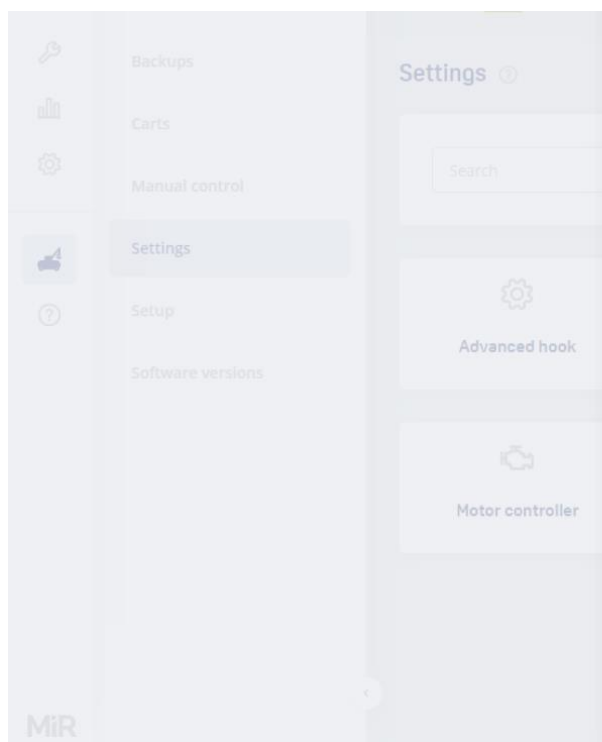
## Distributor data

Edit data about the distributor supplying the hook.



## Modified defaults

Modified defaults provides an overview of which parameters have been changed from the default values. The list shows both the default values and the new values of the changed parameters. All changed parameters from **Settings** are included in the Modified defaults list.



Modified defaults provides an overview of which parameters have been changed from the default values. The list shows both the default values and the new values of the changed parameters. All changed parameters from **System > Settings** are included in the Modified defaults list.

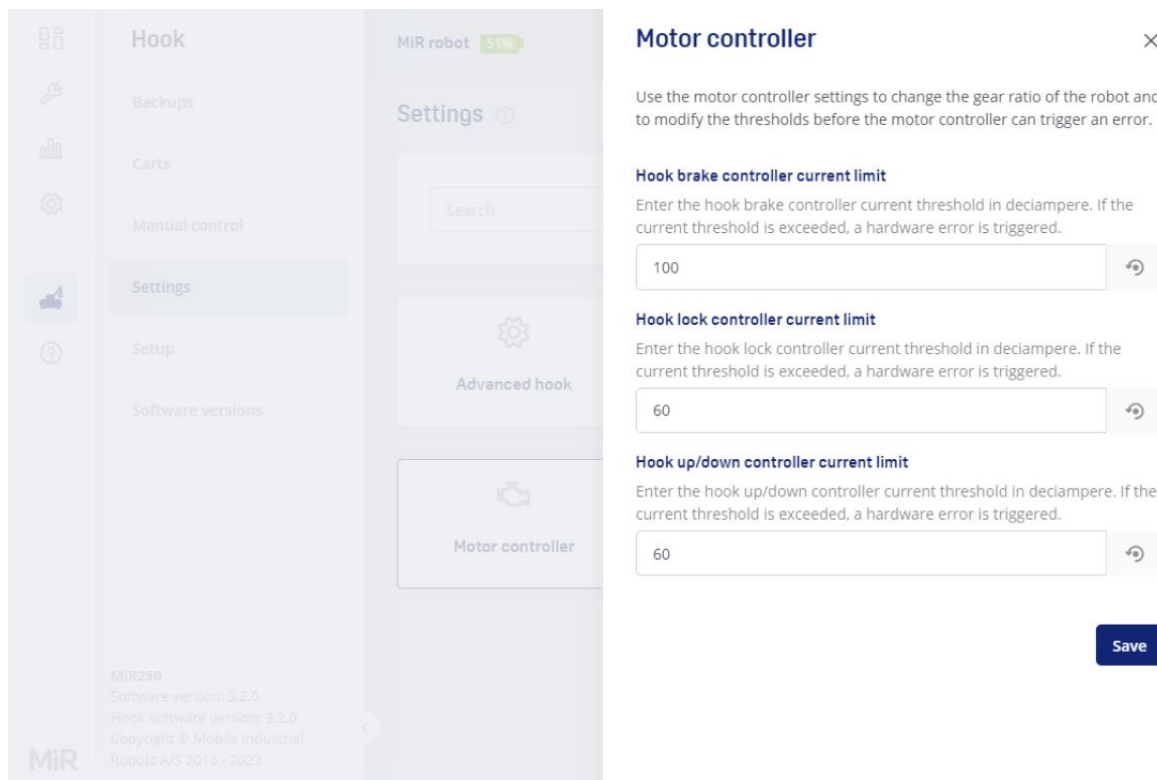
Hook	
	Value
Hook encoder angle calibration offset	0
Hook actuator serial number	AU050WLO
Hook brake serial number	AU051CLR

## Motor controller

Use the motor controller settings to change the gear ratio of the robot and to modify the thresholds before the motor controller can trigger an error.

MiR Hooks have three motors where you can modify the current limit. For more information about how to determine when to modify these values, see the guide *Troubleshoot MiR Hook 100 or MiR Hook 200 issues* or *Troubleshoot MiR250 Hook issues*. You can find these guides on [MiR Support Portal](#).

- The Hook brake motor controls the brake motor that locks the hook arm so it does not swing freely from side to side. It pushes the brake pads together to prevent the hook arm from turning.
- The Hook lock motor controls the gripper that locks and releases the hook gripper bar. It moves the gripper to lock to the bar until the current limit is reached.
- The Hook up/down controller controls the height of the hook. It moves the whole hook arm up and down to the chosen height.



### 3.6.4 Setup

In the **Setup** section you can change the name of the hook, find the serial number and its integrated components, and calibrate the hook encoder if the encoder has been replaced.

#### Configuration

In the Configuration group you may change the name of the hook and the serial number.

The serial number is a 9- or 15-digit number identifying the hook. The serial number is also found in the Help section under Robot information and on the product label on the hook.

#### Motor controller serial numbers

The motor controller serial number group lists the serial numbers of the motor controllers for the actuators and brake.

Select **Detect** to automatically find the serial number of a motor controller. Make sure only one motor controller is physically connected when you select **Detect**.

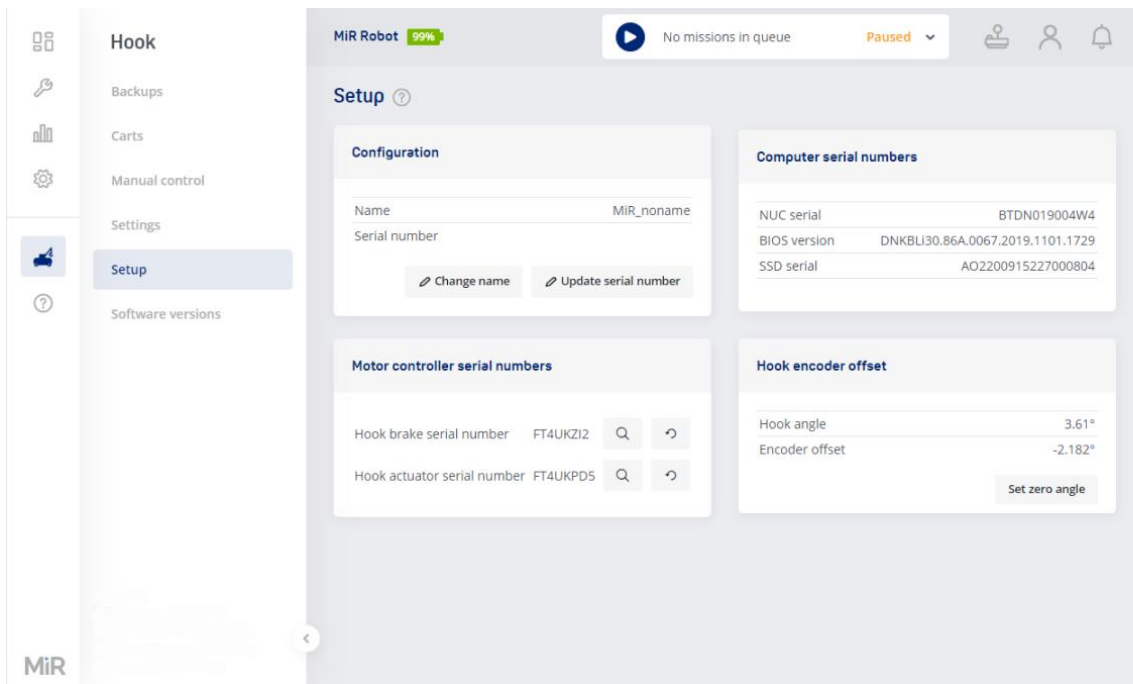
#### Computer serial numbers

The computer serial numbers group lists the information about the hook computer.

### Hook encoder offset

The hook encoder offset group shows the horizontal angle of the hook arm. You can zero calibration the hook by selecting **Set zero angle**.

The hook encoder has already been zero calibrated from the factory, and it should only be performed again if the encoder has been replaced.

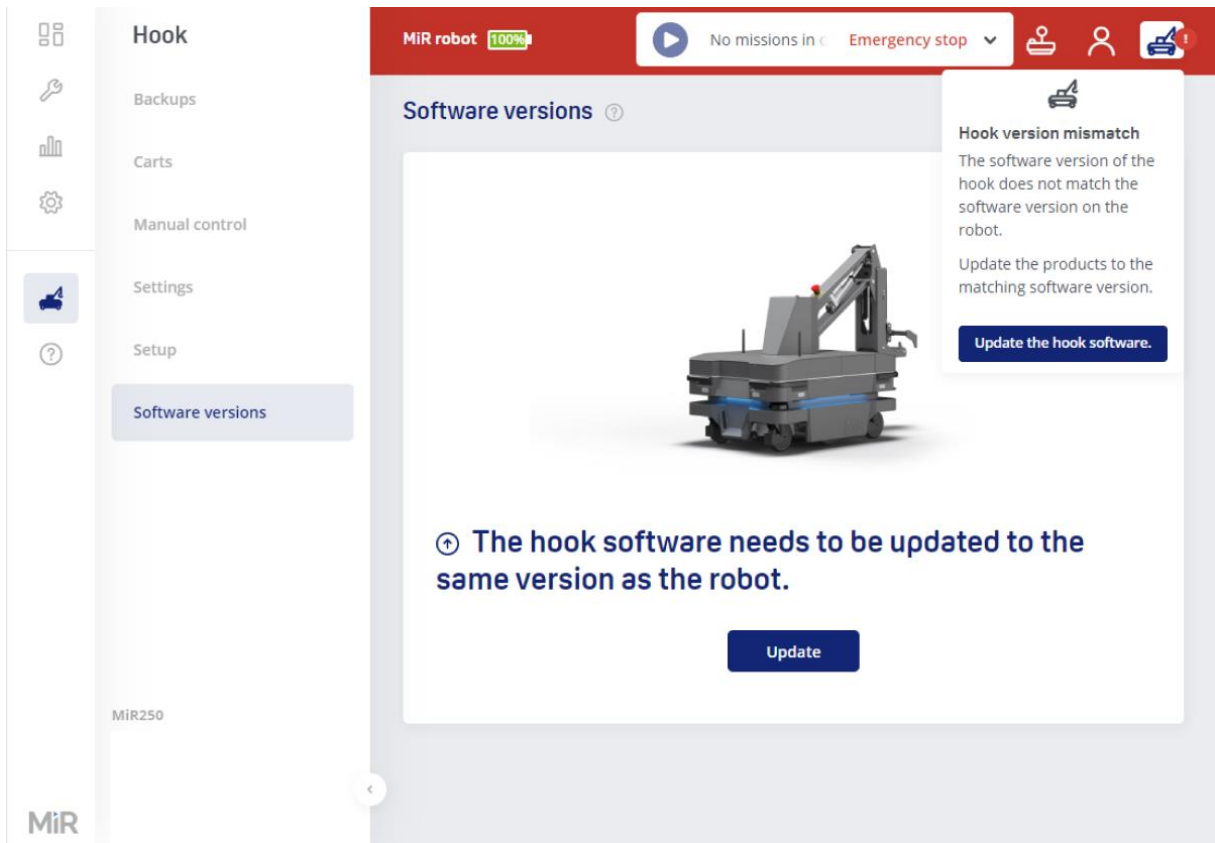


### 3.6.5 Software versions

A software update file is a .mir file that contains MiR robot software. MiR hook robots use the same file that you use to update a MiR robot.

The MiR robot and the mounted hook must be running the same software versions. When you update your robot's software and restart the robot, the hook will search for and transfer the same software. To apply the software to the hook, select **Update**.

Use this page to see the status of the hook software. When you upload the latest software from MiR, you ensure you are running with the latest updates and features.



## 4. Commissioning

Commissioning consists of defining, installing, creating, integrating, testing, documenting, and verifying your site. The following sections give key points and step-by-step guidance to commission your site.

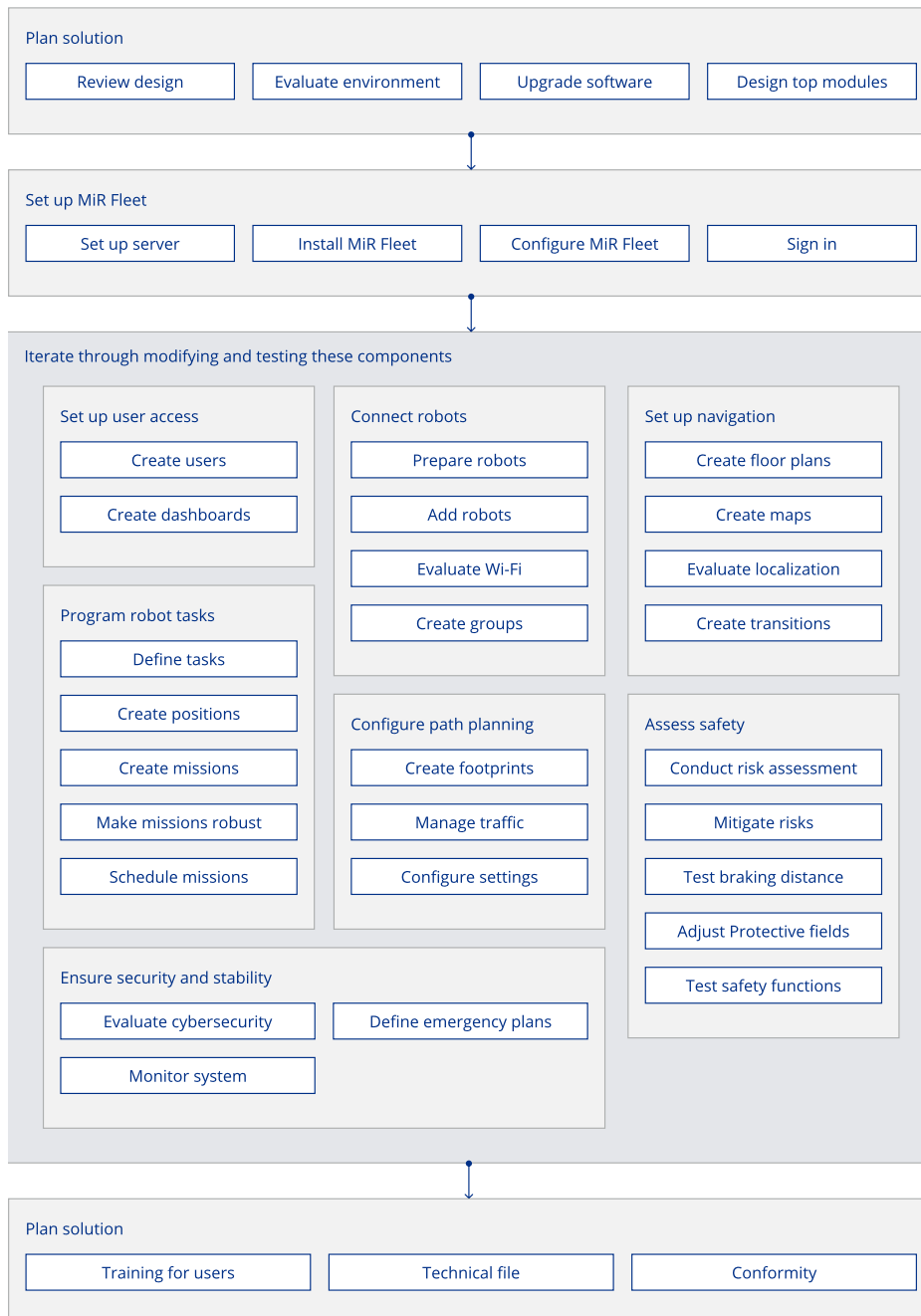
Use the commissioning checklist to keep an overview of your commissioning progress

When commissioning your MiR system, keep the following points in mind:

- Only assigned persons should be present during commissioning. The commissioner has the main responsibility to ensure the robot is commissioned safely.
- Commissioning is an iterative process that requires adjustments and frequent testing.
- Integration-specific commissioning tasks will not be in the guide and depend on the solution and integration design. These will depend on the solution and integration design.
- The commissioner must identify all additional tasks to fully commission the system.
- You can use a robot in Standalone mode to create the site directly on the robot—. This is useful if you do not want to jump between the robot and MiR Fleet interface while creating the site.



**Figure 4.1** Flow diagram of the commissioning process



When considering the development of a MiR system, focus on specific needs, goals, and challenges.

Consider the following points in the process:

- **Requirements for solution:** Start by clearly defining the requirements and objectives of the system. Understand the specific tasks, workflows, and challenges that the MiR robots will address.
- **Initial and final risk assessment:** Create a risk assessment of the expected solution to identify any critical safety issues. Any major changes made while commissioning must be evaluated in a new risks assessment. Always make sure that you evaluate the safety of a solution before implementing it. Conduct a final risk assessment to ensure you have implemented all relevant risk mitigation means.
- **Integration with existing systems:** Evaluate the compatibility of MiR robots with existing systems, such as warehouse management software (WMS), enterprise resource planning (ERP), or other automation systems. Ensure seamless integration to optimize overall operations.
- **Task definition and planning:** Define specific tasks that MiR robots will perform and plan task execution. Consider factors such as task priorities, routes, and any dynamic changes in the environment that the robots need to adapt to.
- **Mapping and navigation:** Utilize MiR's mapping and navigation capabilities by understanding how maps and zones work, creating an accurate map of the operational area, which considers dynamic obstacles and ensures efficient transportation paths.
- **User training:** Provide comprehensive training for operators and maintenance personnel. Ensure that they are familiar with the robots, their behavioral patterns, and the basic principles for navigation and dynamic obstacles.

## 4.1 Commissioning checklist

Use the commissioning checklist as a guidance for the commissioning process. The list does not cover all necessary considerations to declare your installation safe and reliable. Additional items can be added based on the specific application or end user needs.

Plan solution	Comments
<p><b>Define the requirements and expectations for the MiR project.</b> See <a href="#">"Review design"</a> on page 176.</p> <ul style="list-style-type: none"> <li>• Fill out the MiR Project Design template to describe your MiR project.</li> <li>• Determine how you will meet the requirements you have set.</li> <li>• Evaluate if the requirements are realistic.</li> </ul>	

Plan solution	Comments
<p><b>Adjust your site to meet the operating requirements for MiR robots.</b> See <a href="#">"Evaluate environment" on page 178.</a></p> <ul style="list-style-type: none"> <li>• Measure the smallest areas and verify they meet the robots' space requirements.</li> <li>• Test the robots on all driving surfaces and inclines, including gaps and steps, and identify if any areas require special driving behavior.</li> <li>• Modify areas with direct lighting, reflective surfaces, or transparent obstacles that may interfere with the robots' sensor system.</li> <li>• Evaluate if the site meets all environmental specifications for MiR robots.</li> <li>• Check your site for missing, broken, or faulty equipment.</li> <li>• Determine where you can install charging stations, load transfer stations, and a service area if relevant.</li> </ul>	
<p><b>Apply the software version you want to use for the MiR system.</b> See <a href="#">"Upgrade software" on page 181.</a></p> <ul style="list-style-type: none"> <li>• Determine which software version best suits your site.</li> <li>• Upgrade all MiR robots to the same software version as MiR Fleet.</li> </ul>	
<p><b>Purchase or design a top module for the robot tasks.</b> See <a href="#">"Design top modules" on page 204.</a></p> <ul style="list-style-type: none"> <li>• Determine which electrical interfaces on the robot the top module will need.</li> <li>• Choose a top module design that is compatible with your MiR robots.</li> <li>• Evaluate if the combined top module and robot application meets relevant standards.</li> </ul>	

Set up MiR Fleet	Comments
<p><b>Configure your network and server to meet the MiR system requirements.</b> See <a href="#">"Meet system requirements" on page 209</a>.</p> <ul style="list-style-type: none"> <li>• Make sure your server and network structure meet the requirements for MiR systems.</li> <li>• Ensure robots have a fixed IP address using a method that is approved by your IT department.</li> </ul>	
<p><b>Install MiR Fleet on a prepared server.</b> See <a href="#">"Install MiR Fleet" on page 217</a>.</p> <ul style="list-style-type: none"> <li>• Remove previous MiR Fleet software versions.</li> <li>• Install Microsoft .NET Server Hosting and Microsoft SQL server.</li> <li>• Create a user to run MiR Fleet Windows Service.</li> <li>• Install MiR Fleet on the selected host server.</li> </ul>	
<p><b>Configure MiR Fleet according to site cybersecurity policies and preferences.</b> See <a href="#">"Configure MiR Fleet" on page 234</a>.</p> <ul style="list-style-type: none"> <li>• Configure firewalls, TLS, and SSO.</li> <li>• Connect an SQL database.</li> </ul>	
<p><b>Sign in the first time.</b> See <a href="#">"Sign in" on page 254</a>.</p> <ul style="list-style-type: none"> <li>• Sign in to the MiR Fleet interface.</li> <li>• Create an admin user.</li> </ul>	

Set up users	Comments
<p><b>Create roles and users with the necessary permissions to complete user tasks.</b> See <a href="#">"Create users and roles" on page 257.</a></p> <ul style="list-style-type: none"> <li>• List all the users for the MiR system and identify which permissions they need to complete their tasks.</li> <li>• Identify permissions that enable users to make critical changes.</li> <li>• Create roles for each permission group or user type.</li> <li>• Create a user in MiR Fleet for each user and assign the user to relevant roles.</li> <li>• Test that all users have access to the features and tools they need.</li> <li>• Test that users do not have access to anything they are not trained to handle.</li> <li>• Document the overview of users, permissions, and tasks.</li> </ul>	
<p><b>Make dashboards to collect relevant functions on a single page in the interface.</b> See <a href="#">"Create dashboards" on page 264.</a></p> <ul style="list-style-type: none"> <li>• Determine what functions your users need to complete their tasks.</li> <li>• Create dashboards that make these tasks easier to complete.</li> <li>• Test that your dashboards work as expected.</li> <li>• Document which tasks or user each dashboard is intended for.</li> </ul>	

Connect robots	Comments
<p><b>For each robot, check the robot's condition and set up the robots to prevent unauthorized access.</b> See <a href="#">"Prepare robots" on page 269</a>.</p> <ul style="list-style-type: none"> <li>• Go through the received product inspection checklist.</li> <li>• Upgrade the robot software.</li> <li>• Protect against booting up from USB.</li> <li>• Set a password for the robot BIOS.</li> <li>• Disable all unused features.</li> <li>• Disable or password protect robot access points.</li> <li>• Update robot access points.</li> </ul>	
<p><b>Connect robots to MiR Fleet.</b> See <a href="#">"Add robots" on page 276</a>.</p> <ul style="list-style-type: none"> <li>• Connect robots to the same Wi-Fi network as MiR Fleet.</li> <li>• Set up mutual trust between MiR Fleet and the robots.</li> <li>• Add one robot to MiR Fleet to use to setup and test the site with a single robot.</li> <li>• When the site is complete and tested, add all other robots to MiR Fleet.</li> </ul>	
<p><b>Adjust the Wi-Fi network so robots have a stable connection across the whole site.</b> See <a href="#">"Evaluate Wi-Fi" on page 285</a>.</p> <ul style="list-style-type: none"> <li>• Review the network requirements for MiR systems.</li> <li>• Apply necessary adjustments to your site network.</li> <li>• Send two robots across the site to gather Wi-Fi signal data.</li> <li>• Evaluate the Wi-Fi performance and coverage.</li> <li>• Document the Wi-Fi heatmaps made between any changes to the site network.</li> </ul>	

Connect robots	Comments
<p><b>Create groups to define which missions, charging stations, and Staging positions each robot can be assigned to.</b> See <a href="#">"Create groups" on page 288.</a></p> <ul style="list-style-type: none"> <li>• Determine which charging stations, Staging positions, and missions you want each robot to be compatible with.</li> <li>• Create groups that enable each robot to access the necessary resources and missions.</li> <li>• Add robots and resources to groups.</li> <li>• Document the overview of groups, robots, and resources.</li> <li>• Document how to determine which group new elements should be added to.</li> </ul>	
<p><b>Verify that robots are correctly managed by MiR Fleet.</b> See <a href="#">"Test MiR Fleet" on page 291.</a></p> <ul style="list-style-type: none"> <li>• Verify that data synchronizes correctly across all robots.</li> <li>• Test that robots are assigned to missions as expected.</li> <li>• Test that robots are assigned to charging stations and Staging positions as expected.</li> <li>• Test that robots still run correctly after turning them off and on again.</li> </ul>	
Set up navigation	Comments
<p><b>Record or upload accurate floor plans.</b> See <a href="#">"Create floor plans" on page 294.</a></p> <ul style="list-style-type: none"> <li>• Determine your floor plan strategy.</li> <li>• If you are recording any floor plans, prepare the operating area.</li> <li>• Optimize the floor plan quality.</li> <li>• Remove invalid floor plans from MiR Fleet.</li> </ul>	

Set up navigation	Comments
<p><b>Combine floor plans to create maps.</b> See <a href="#">"Create maps"</a> on <a href="#">page 311</a>.</p> <ul style="list-style-type: none"> <li>• Determine the number of maps you need to cover the operating area.</li> <li>• Determine the naming scheme for your maps.</li> <li>• Align floor plans and merge them together to create a straight map.</li> <li>• Add or remove obstacles to make the maps accurate.</li> <li>• Document which areas each map covers.</li> </ul>	
<p><b>Ensure robots can localize themselves across the whole operating area.</b> See <a href="#">"Evaluate localization"</a> on <a href="#">page 321</a>.</p> <ul style="list-style-type: none"> <li>• Localize all robots on the active map.</li> <li>• Drive one robot across the whole map.</li> <li>• Evaluate the localization score.</li> <li>• Apply map changes if necessary.</li> <li>• Document the final localization score and evaluate the localization regularly to ensure the map continues to be accurate.</li> </ul>	
<p><b>Create transitions between maps.</b> See <a href="#">"Create transitions"</a> on <a href="#">page 325</a>.</p> <ul style="list-style-type: none"> <li>• Create transition positions where maps overlap.</li> <li>• Create transition missions for each map transition.</li> <li>• Define transitions.</li> <li>• Test that a robot can travel to and from each map using the appropriate transition.</li> <li>• Document the transitions, including the relevant map locations, positions, and missions.</li> </ul>	



Program robot tasks	Comments
<p><b>Define the robot tasks and break them down into mission building blocks and categories.</b> See <a href="#">"Define tasks" on page 333</a>.</p> <ul style="list-style-type: none"> <li>• List all robot tasks.</li> <li>• Define the actions the robots must perform to complete the tasks.</li> <li>• Identify actions that are repeated across tasks.</li> <li>• Plan how you want to organize missions into mission groups.</li> <li>• Agree on a naming scheme.</li> </ul>	
<p><b>Create markers and positions at all points of interest for the robot tasks.</b> See <a href="#">"Create positions and markers" on page 337</a>.</p> <ul style="list-style-type: none"> <li>• Determine which type of position or marker is best suited for each point.</li> <li>• Agree on a naming scheme.</li> <li>• Add positions and markers.</li> <li>• Test how the robot docks to each marker and apply corrections.</li> </ul>	
<p><b>Create mission groups and missions for the robot tasks.</b> See <a href="#">"Create missions and mission groups" on page 347</a>.</p> <ul style="list-style-type: none"> <li>• Create the mission groups you planned.</li> <li>• Create reusable missions for all actions that are repeated across robot tasks.</li> <li>• Create missions for the robot tasks.</li> <li>• Test the missions and apply corrections.</li> <li>• Document the mission setup.</li> </ul>	

Program robot tasks	Comments
<p><b>Make missions more robust.</b> See <a href="#">"Make missions robust" on page 377.</a></p> <ul style="list-style-type: none"> <li>• Test missions where you expose the robot to unexpected events you want it to handle autonomously.</li> <li>• Use Try/Catch actions to define alternative actions in missions when exceptions or unexpected events occur.</li> <li>• Iterate between testing and modifying missions.</li> <li>• Update the mission documentation if necessary.</li> </ul>	
<p><b>Set up system for scheduling missions.</b> See <a href="#">"Schedule missions" on page 398.</a></p> <ul style="list-style-type: none"> <li>• Determine triggers for each mission.</li> <li>• Create or modify any external system to schedule missions automatically.</li> <li>• Create or modify dashboards to help users schedule missions manually.</li> <li>• Document the chosen triggers for each mission and responsibilities for ensuring each mission runs.</li> </ul>	
Configure path planning	Comments
<p><b>Create footprints that make robots plan efficient paths that avoid collisions.</b> See <a href="#">"Create footprints" on page 404.</a></p> <ul style="list-style-type: none"> <li>• Determine the dimensions of all the possible top modules and loads on the MiR robots.</li> <li>• Agree on a naming scheme that makes it clear which footprint to use.</li> <li>• Create footprints.</li> <li>• Document the size and purpose of each footprint.</li> </ul>	

Configure path planning	Comments
<p><b>Add zones to maps to make MiR Fleet prevent robots from blocking each other and prevent robot from driving into unintended areas.</b> See <a href="#">"Manage traffic"</a> on page 411.</p> <ul style="list-style-type: none"> <li>• Mark points of interest, robot routes, existing traffic conditions, and other relevant factors on a map of the operating area.</li> <li>• Use Preferred, Unpreferred, and Directional zones to coordinate the robot routes.</li> <li>• In all areas the robot must never enter, add Forbidden zones.</li> <li>• At all crossings, docking stations, and narrow corridors, use Limit-robots zones to prevent robots blocking each other.</li> <li>• Document all of the zones and their purposes.</li> </ul>	
<p><b>Adjust the robot settings to change how the robot drives and plans paths.</b> See <a href="#">"Configure settings"</a> on page 428.</p> <ul style="list-style-type: none"> <li>• Define how much robots can deviate in their path planning.</li> <li>• Determine if you need to adjust the camera filtering settings.</li> <li>• Apply settings changes to all other relevant robots.</li> </ul>	
Assess safety	Comments
<p><b>Conduct a risk assessment.</b> See <a href="#">"Conduct risk assessment"</a> on page 438.</p> <ul style="list-style-type: none"> <li>• Use the <i>MiR Risk Assessment Guide</i> for MiR-specific use cases. You can find this guide on <b>MiR Support Portal</b>.</li> </ul>	
<p><b>Implement risk mitigation means.</b> See <a href="#">"Mitigate risks"</a> on page 439.</p> <ul style="list-style-type: none"> <li>• Mark high-risk areas as Operating hazard zones.</li> <li>• Mark all docking and charging areas as Operating hazard zones.</li> <li>• Implement lights and sounds in the robot maps and missions to increase personnel awareness.</li> <li>• Design docking areas to include escape routes for personnel.</li> </ul>	

Assess safety	Comments
<p><b>Ensure the robots' braking distance is safe with all loads.</b> See <a href="#">"Test braking distance" on page 448.</a></p> <ul style="list-style-type: none"> <li>• Conduct a brake test with the lightest, heaviest, and largest loads, and any loads with unusual properties.</li> <li>• Adjust the robot, the environment, or the robot payload until the stopping distance criteria in the brake test are met.</li> </ul>	
Ensure security and stability	Comments
<p><b>Assess the cybersecurity of your MiR system.</b> See <a href="#">"Evaluate cybersecurity" on page 464.</a></p> <ul style="list-style-type: none"> <li>• Conduct a cybersecurity risk assessment.</li> <li>• Protect the robots from unauthorized access.</li> <li>• Manage users</li> <li>• Implement an auditing system</li> </ul>	
<p><b>Document emergency plans for all potential emergency situations.</b> See <a href="#">"Define emergency plans" on page 466.</a></p> <ul style="list-style-type: none"> <li>• Identify possible emergency situations.</li> <li>• Implement preventative actions to reduce the likelihood and severity of the emergency.</li> <li>• Document what must be done and who is responsible for each task if the event should occur.</li> </ul>	

Ensure security and stability	Comments
<p><b>Monitor site events and analyze data for performance improvements and unintended or suspicious behavior.</b> See <a href="#">"Monitor system"</a> on page 470.</p> <ul style="list-style-type: none"> <li>• Determine relevant monitoring tools and information.</li> <li>• Define behavior to be acted on.</li> <li>• Set up filters on Event viewer to detect relevant messages in MiR Fleet.</li> <li>• Set up an application to monitor important messages from MiR Fleet audit log.</li> <li>• Define responsibilities for incoming alerts.</li> <li>• Consider setting up webhook subscriptions to events of interest.</li> <li>• Consider purchasing a MiR Insights license.</li> </ul>	
Prepare handover	Comments
<p><b>Ensure your MiR system conforms to relevant standards to maintain a safe solution.</b> See <a href="#">"Assess conformity"</a> on page 472.</p> <ul style="list-style-type: none"> <li>• Assess the system conformity regularly against the latest relevant standards.</li> <li>• Collect a technical file with all relevant system documentation and information of use for the complete system.</li> <li>• Establish a maintenance schedule.</li> <li>• Assess any custom top module or modification made to approved top modules.</li> </ul>	

Prepare handover	Comments
<p><b>Train users so they can correctly and safely execute tasks they are responsible for.</b> See <a href="#">"Train users" on page 475</a>.</p> <ul style="list-style-type: none"> <li>• Determine the responsibilities of your users and categorize them into user types.</li> <li>• Train users in their relevant tasks and knowledge of the MiR system.</li> <li>• Keep training records and schedule new training sessions if new equipment or processes are introduced.</li> </ul>	

## 4.2 Migrate data from SW 2.x or 3.x

If you already have a MiR system running on software 2.x or 3.x, and you want to migrate to MiR Fleet Enterprise, read this section to understand what is required to migrate your existing data.

### 4.2.1 Migrate from Software 2.x

Before upgrading to MiR Fleet Enterprise software, upgrade your robots to 3.x software. The upgrade to software 3.x takes longer than standard upgrades, and there is a risk of data loss if the process is done incorrectly. Follow the *MiR Software 2.x to 3.x Migration Guide*. You can find this guide on [MiR Support Portal](#).

This upgrade will change the database structure to one that can become compatible with MiR Fleet Enterprise software. You cannot upgrade directly from software 2.x to MiR Fleet Enterprise without first upgrading to 3.x.

### 4.2.2 Migrate from Software 3.x

MiR Technical Support can convert a 3.x error log to a MiR Fleet Enterprise site file with most of your site data. Some features are no longer available where you will have to use an alternative solution—see the MiR Fleet Enterprise Release Note.

Generate an error log on a robot or MiR Fleet running on software 3.x. Create a support request for MiR Technical Support and include the error log to request a MiR Fleet Enterprise version of the site.

Install a new instance of MiR Fleet using the MiR Fleet Enterprise installer—see ["Install MiR Fleet" on page 217](#).

Upload the site file with the migrated data. Check that the data has been migrated correctly.

All robots must run MiR Fleet Enterprise software to be able to communicate with MiR Fleet—see ["Upgrade software" on page 181](#).

## 4.3 Plan solution

### **After reading this chapter, you can:**

Define and verify your solution plan.

**Define the requirements and expectations for the MiR project.** See ["Review design" on the next page](#).

- Fill out the MiR Project Design template to describe your MiR project.
- Determine how you will meet the requirements you have set.
- Evaluate if the requirements are realistic.

**Adjust your site to meet the operating requirements for MiR robots.** See ["Evaluate environment" on page 178](#).

- Measure the smallest areas and verify they meet the robots' space requirements.
- Test the robots on all driving surfaces and inclines, including gaps and steps, and identify if any areas require special driving behavior.
- Modify areas with direct lighting, reflective surfaces, or transparent obstacles that may interfere with the robots' sensor system.
- Evaluate if the site meets all environmental specifications for MiR robots.
- Check your site for missing, broken, or faulty equipment.
- Determine where you can install charging stations, load transfer stations, and a service area if relevant.

**Apply the software version you want to use for the MiR system.** See ["Upgrade software" on page 181](#).

- Determine which software version best suits your site.
- Upgrade all MiR robots to the same software version as MiR Fleet.

**Purchase or design a top module for the robot tasks.** See ["Design top modules" on page 204](#).

- Determine which electrical interfaces on the robot the top module will need.
- Choose a top module design that is compatible with your MiR robots.
- Evaluate if the combined top module and robot application meets relevant standards.

### 4.3.1 Review design

#### Section overview

Define the requirements and expectations for the MiR project.

- Fill out the MiR Project Design template to describe your MiR project.
- Determine how you will meet the requirements you have set.
- Evaluate if the requirements are realistic.

Before implementing any robot setup, ensure that the design plans for the project have been reviewed and verified.

MiR provides a generic design review checklist that includes the most common aspects of a MiR project that be verified before commissioning. The design review checklist can also be downloaded in a separate .pdf file [here](#) or from [MiR Support Portal](#).

Store the documentation of the design review for at least 10 years and make it available upon request.

Plan project	✓	Comments
Make a timeline and project planning document.	<input type="checkbox"/>	
Prepare descriptions of each project task, including: <ul style="list-style-type: none"> <li>• Scope of the task</li> <li>• Goal of the task</li> <li>• Who is responsible for the task</li> <li>• Expected time until the task is completed</li> <li>• Dependencies on other tasks</li> </ul>	<input type="checkbox"/>	



<b>Plan project</b>	✓	<b>Comments</b>
Clarify any final responsibilities or expectations between parties.	<input type="checkbox"/>	
Define risk assessment expectations and discuss with stakeholders.	<input type="checkbox"/>	
Define cybersecurity expectations and discuss with stakeholders.	<input type="checkbox"/>	
Consider risks of potential site changes in the future.	<input type="checkbox"/>	
<b>Plan site</b>	✓	<b>Comments</b>
Verify that MiR robots are suitable for the intended tasks.	<input type="checkbox"/>	
Calculate the required number of robots and chargers.	<input type="checkbox"/>	
Specify throughput and cycle time requirements.	<input type="checkbox"/>	
Consider operating and maintenance windows.	<input type="checkbox"/>	
Prepare robot routes in an accurately dimensioned layout of the site.	<input type="checkbox"/>	
Consider level of integration (ERP control, MiR Fleet mission scheduling, or manual control).  Assess how you want to integrate MiR Fleet with existing management systems—see <a href="#">"MiR Fleet Integration API" on page 484</a> .	<input type="checkbox"/>	
Conduct an initial risk assessment of the intended solution to evaluate if the expected safety is acceptable.	<input type="checkbox"/>	
<b>Check specifications</b>	✓	<b>Comments</b>
Verify that the site environment is suitable for MiR applications.	<input type="checkbox"/>	
Verify that your server meets minimum MiR Fleet requirements and has required applications installed—see <a href="#">"Meet system requirements" on page 209</a> and <a href="#">"Install MiR Fleet" on page 217</a> .	<input type="checkbox"/>	

Check specifications	✓	Comments
Verify that the Wi-Fi network can meet MiR requirements.	<input type="checkbox"/>	
Evaluate that the site environment can meet the space requirements for the intended number of robots.	<input type="checkbox"/>	
Evaluate possible friction coefficient between flooring and robots.	<input type="checkbox"/>	

Check available resources	✓	Comments
Verify that all necessary information for each load transfer station is available for the project.	<input type="checkbox"/>	
Retrieve necessary documentation on relevant external equipment including communication methods and functionality.	<input type="checkbox"/>	
Approve CAD drawings of top module, load carriers, and other assets designed for this project.	<input type="checkbox"/>	

### 4.3.2 Evaluate environment

#### Section overview

Adjust your site to meet the operating requirements for MiR robots.

- Measure the smallest areas and verify they meet the robots' space requirements.
- Test the robots on all driving surfaces and inclines, including gaps and steps, and identify if any areas require special driving behavior.
- Modify areas with direct lighting, reflective surfaces, or transparent obstacles that may interfere with the robots' sensor system.
- Evaluate if the site meets all environmental specifications for MiR robots.
- Check your site for missing, broken, or faulty equipment.
- Determine where you can install charging stations, load transfer stations, and a service area if relevant.



You can find the environment and space requirements under the specifications for your robot on [MiR Support Portal](#).

## Surfaces

- The floor surface must be clean and dry.
- The floor material and condition can affect the performance and safety of the robots. Test the robots on all the floor types they will operate on.
- The floor surface must be even without bumps and holes.
- The floor may not have gaps or sills that exceed the maximum size described in the robots' specifications.
- The friction coefficient between the floor and the drive wheels on the robots should be in the range of 0.60 to 0.80. This parameter is dependent on the results of the brake test—see "[Test braking distance](#)" on page 448
- The surface slope may not exceed the maximum incline described in the robots' specifications.
- If these surface requirements are not met, we cannot guarantee the robots' stability even if you are within the payload specifications.

## Light, reflections, and materials

- Bright sunlight and reflective or transparent objects can affect the performance of the robots' laser scanners and cameras. This can result in robots detecting nonexistent objects or failing to detect real objects.
- Avoid making markers in very high gloss or transparent materials. This can reduce the effectiveness of the robots' scanners, hindering a successful docking.
- If you cannot avoid glass or glossy material, you can apply a line of matte tape on transparent or reflective objects at the robots' scanner height (200 mm from the ground) to make sure the robots detect them correctly. Always test that this solution works from various robot distances.
- Take time of day and season into account when evaluating the natural lighting conditions on site, as these may change enough to affect robots' laser scanner and camera performance.

## Required space

- Operating the robots in areas that do not meet the approved specifications for required space can result in the robots entering Protective stop often and planning unexpected routes. See the space requirement guide for your robot applications. You can find these guides on [MiR Support Portal](#).
- For personnel safety, ensure to have sufficient space and escape routes according to standards and regulations of the country where the robots are installed—see "[Escape routes](#)" on [page 444](#). For more information, see ISO 3691-4 or ANSI B56.5.

## Temperature and humidity

Temperatures and air humidity outside of the approved range can affect the performance and durability of the robots. This is especially relevant for the robots' batteries.

## Dust

Dusty environments can affect the performance, durability, and maintenance frequency of the robots. Make sure the environment the robots operate in is suitable for its IP rating.

## Static landmarks and dynamic obstacles

The robots uses static landmarks to navigate by. If they cannot detect enough distinguishing landmarks, it cannot navigate the map efficiently and will report localization errors.

## Elevators

If you intend for the robots to use elevators to travel between floors, verify that the combined weight of the robot, including payload, battery, and top module does not exceed the limits of the elevator.

## Equipment

Check that any equipment that robots operate with or around are functioning correctly and have all their safeguards.

Check that there is solid, matte material at 200 mm from ground height that MiR robots can use to detect the equipment. If robots cannot reliably detect the equipment, either install panels at 200 mm height, or mark the area with Forbidden zones—see "[Manage traffic](#)" on [page 411](#).

Assess if you are missing any relevant equipment for the intended solution.

## Charging stations and staging areas

Allocate certain areas of your site for charging and staging. Areas where you install charging stations must meet the electrical and environmental requirements for the charging station—see the user guide for the charging station.

### 4.3.3 Upgrade software

#### Section overview

Apply the software version you want to use for the MiR system.

- Determine which software version best suits your site.
- Upgrade all MiR robots to the same software version as MiR Fleet.

Upgrade your system to the latest software version to keep your system up to date. To see which issues are resolved in each release, see the release notes for each released software. For software release notes, see [MiR Support Portal](#).

Selecting which software version is best for your system depends on:

- **Resolved issues and desired features:** Newer software versions often have more features, are more secure, or have fewer issues. We often recommend applying the latest software version.
- **The robot's model:** Robots are only supported from the software version that was released at the same time as the first model of the robot. It is written in the robot's user guide or integrator manual which is the minimum required software if applicable to your robot model.
- **Existing MiR setup:** If you already have MiR Fleet and a number of robots connected to it, you may prefer to use the same software version that your robots are already running. If you want to add a new robot that is not supported on the software that your MiR Fleet is already using, you must update MiR Fleet and all connected robots to a compatible software version.

#### Software version format

The MiR software version system uses semantic versioning:

- Major (**X**.x.x) releases include the most significant changes that affect the entire robot software.
- Minor (x.**X**.x) releases often include new features and smaller changes that only affect parts of the software.
- Patch (x.x.**X**) releases focus on fixing small issues in the software and introducing quality improvements.

### Software compatibility on MiR Fleet

For full compatibility, upgrade MiR robots to the exact same software version as MiR Fleet.

You can add and run robots to MiR Fleet that use different minor and patch MiR Fleet Enterprise software versions—see ["Software version format" on the previous page](#)—but we do not guarantee behavioral compatibility.

To upgrade a robot to a new major software version see ["Migrate data from SW 2.x or 3.x" on page 174](#).

### Upgrade robot software

MiR robots have two different .mir upgrade files you can apply to the robot.

The **Platform software** controls the operating system software and other core applications that the robot software depends on, such as upgrade and backup control and Wi-Fi settings.

The **Application software** runs on the Platform OS. It contains software that controls robot-related tasks like navigation, mission handling, and communication with MiR Fleet.

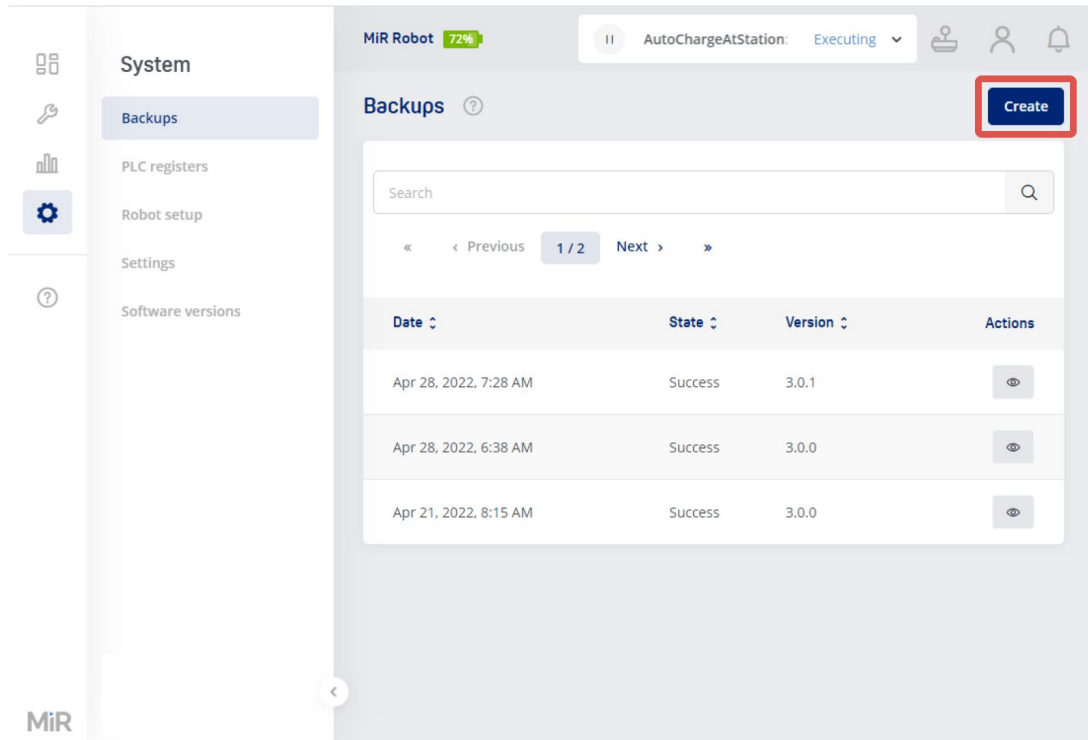
It is recommended that the robot runs the same Platform and Application software versions to ensure the robot performs as expected. The process for updating the Application and Platform software is the same on MiR robots—see ["Upgrade in the robot interface" on page 186](#) or ["Upgrade in the platform interface" on page 189](#).

The Platform software is designed to be backwards compatible with older versions of the Application software. To avoid issues while updating, always upgrade the Platform software before updating the Application software.

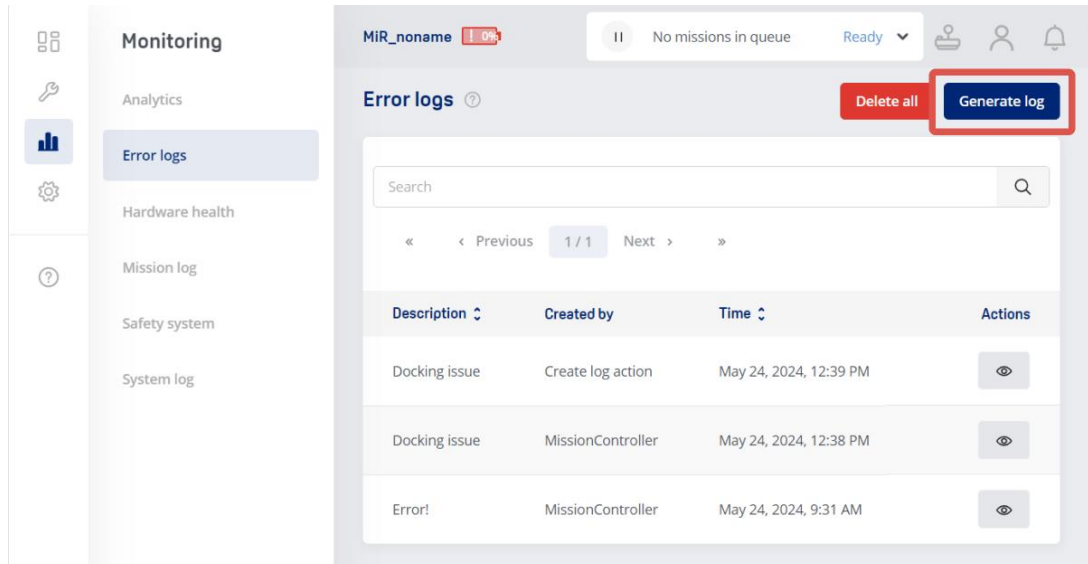
When updating MiR hook, you can upgrade the Application software through the interface, but you must upgrade the Platform software using the platform interface—see ["Upgrade the MiR hook software" on page 192](#).

### Before updating the software

- 1 Create a backup. Go to **System > Backups**, and select **Create**. This is so you can revert the robot in case any data is lost.



- 2 Generate an error log. Go to **Monitoring > Error logs**, select **Generate log**, then download the generated error log. You can send this error log to MiR Technical Support in case the upgrade fails completely.



The screenshot shows the MiR Monitoring interface. On the left is a navigation menu with 'Monitoring' selected. The main content area displays 'Error logs' with a search bar and a table of error entries. The 'Generate log' button is highlighted with a red box.




Description	Created by	Time	Actions
Docking issue	Create log action	May 24, 2024, 12:39 PM	
Docking issue	MissionController	May 24, 2024, 12:38 PM	
Error!	MissionController	May 24, 2024, 9:31 AM	




3 To ensure there is enough disk space on your product, remove any old backups or software files that you are not likely to use.

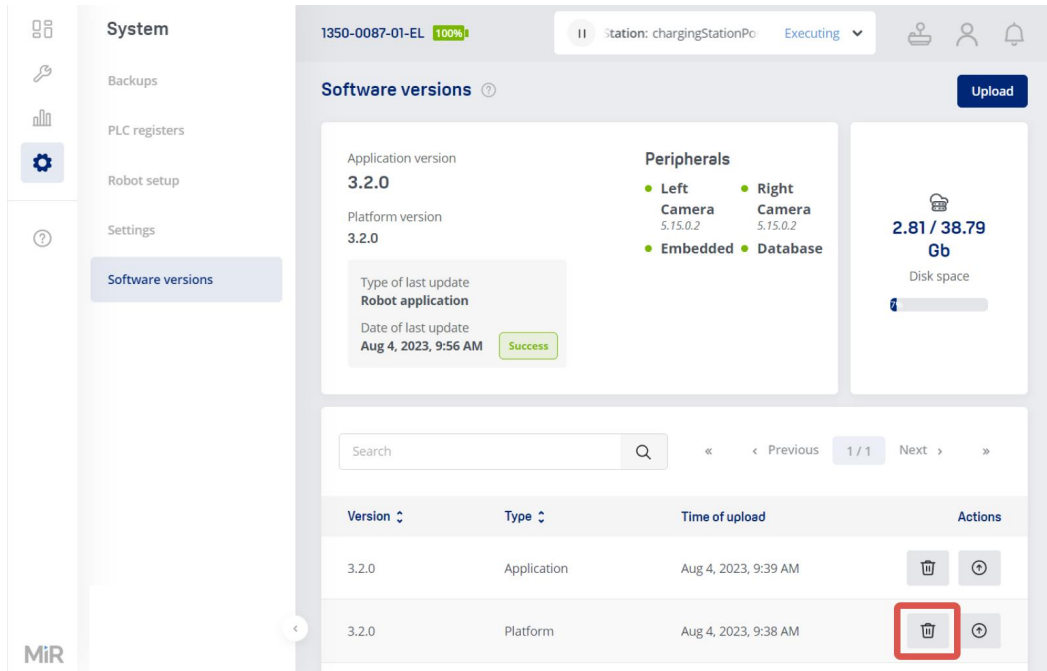
- To remove backups, go to **System > Backups**, select **View**, and select **Delete** for any backups you want to remove.

The screenshot shows the MiR200 System Backups interface. The left sidebar contains the following menu items: System, Backups (selected), PLC registers, Robot setup, Settings, and Software versions. The main content area displays the 'Backups' section with a search bar and a table of backup records. The table has the following columns: Date, State, Version, and Actions. The first row of the table is highlighted, and the 'View' icon in the Actions column is circled in red.

Date	State	Version	Actions
Apr 28, 2022, 7:28 AM	Success	3.0.1	
Apr 28, 2022, 6:38 AM	Success	3.0.0	
Apr 21, 2022, 8:15 AM	Success	3.0.0	

MiR200  
Software version: 3.0.0  
Copyright © Mobile Industrial Robots A/S 2016 - 2022

- To remove old software files, go to **System > Software versions**, and select **Delete**  for any software files you want to remove.

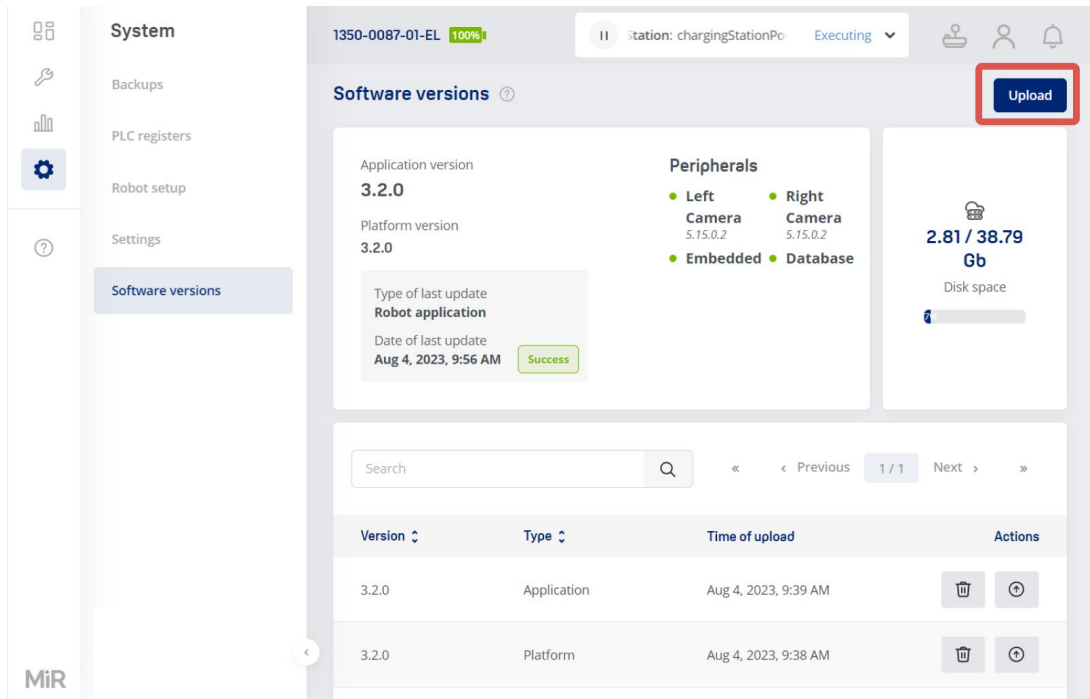


- 4 Make sure the robot is charged to at least 50%, and disconnect it from any charging stations or cable chargers. The robot should be in Ready or Paused state, not executing.

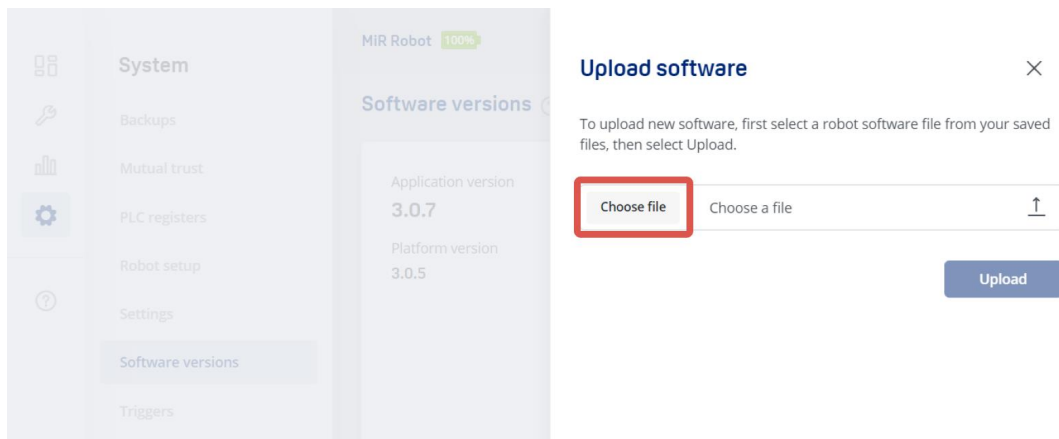
### Upgrade in the robot interface

- 1 Download the MiR Platform and MiR Application software files to a PC that you can use to connect to the robot. The file titles use the formats **MiR\_Platform-x.x.x.mir** and **MiR\_Application-x.x.x.mir**.
- 2 Connect to the robot interface from a browser.
- 3 Sign in to the interface for the robot you want to upgrade, and go to **System > Software versions**.

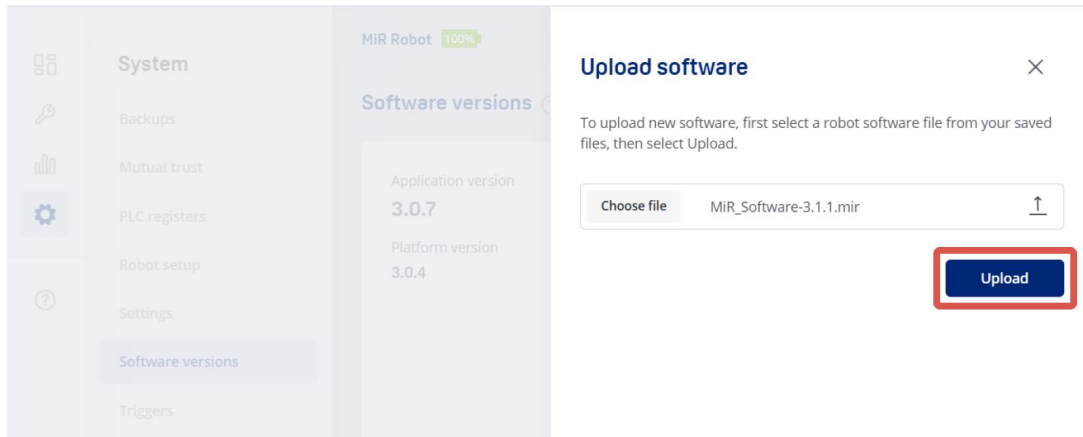
4 Select **Upload**.



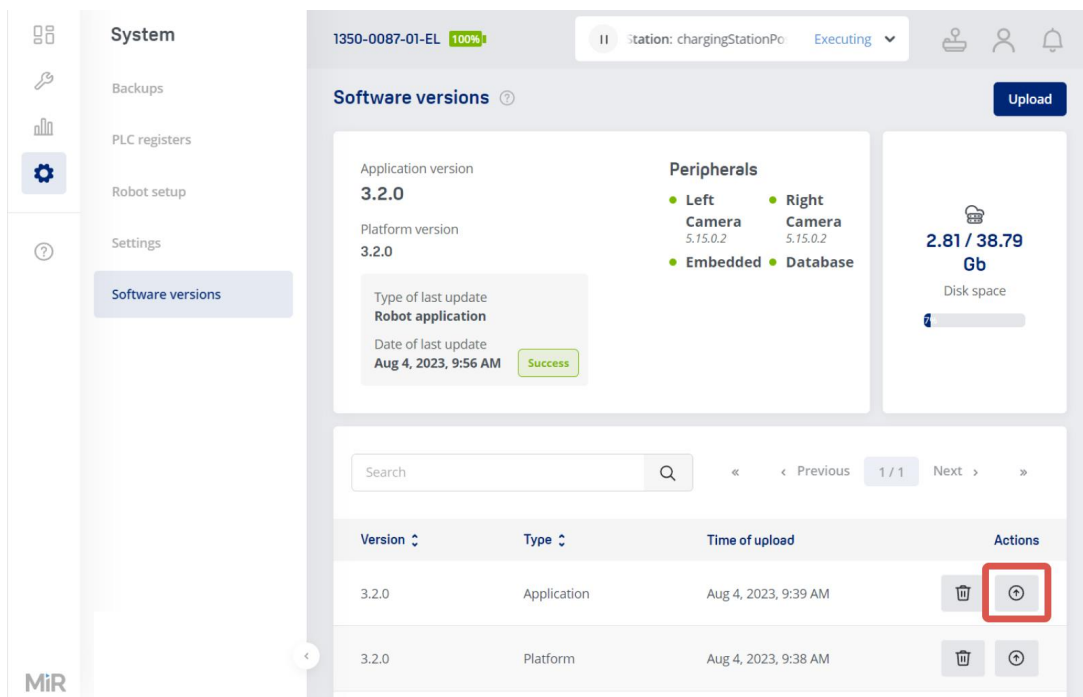
5 Select **Choose file**.



- 6 Select the Platform software file you want to apply, and then select **Upload**. Once the file has finished uploading, it is included in the list of uploaded software versions. If the upload fails, see "The upload failed" on page 193.



- 7 Select **Update** for the software file you want to apply.



- 8 Wait for the robot to finish updating. The interface will notify you once the upgrade is complete. If the upgrade takes longer than 40 minutes, see ["The upgrade is taking longer than 40 min" on page 194](#).
- 9 Repeat from [step 4](#) with the Application software instead. It is recommended that the Application and Platform software are upgraded to the same version.
- 10 Once the interface notifies you that the software has been successfully upgraded, restart your robot. If the upgrade is not applied after restarting the robot, see ["The software version has not been upgraded after restarting the robot" on page 203](#).

**CAUTION**

Do not restart the robot until the software has completely finished updating. Turning off the robot in the middle of an upgrade may result in a malfunction of the robot system.

**Upgrade in the platform interface****NOTICE**

Do not upgrade the robot to an older application software than it is already running through the platform. This can lead to loss of data if you do not have a backup.

In case you cannot connect to the robot interface, you can also upgrade the software using the platform interface instead:

- 1 Download the MiR Platform and MiR Application software files to a PC that you can use to connect to the robot. The file titles use the formats **MiR\_Platform-x.x.x.mir** and **MiR\_Application-x.x.x.mir**.
- 2 Connect your PC to your robot's local network via an access point or Ethernet cable.

- 3 Open a browser window, and go to 192.168.12.20:8888. This connects you to the robot computer.

If you are updating a MiR hook, go to 192.168.12.21:8888 instead.

- 4 Sign in with the **unique credentials** from the documentation that came with your robot. Contact MiR Technical Support if you are unable to sign in, or if the unique credentials have been lost.

If you never received any unique credentials with your robot, use the default credentials:

- **Username:** mir
- **Password:** mirremote

- 5 Under **File upload**, select **Choose software**.

**SYSTEM OVERVIEW**

Hostname	mir-edb-intel-gen1
Rootfs Partition	(B)
Product Config Overlay	ACTIVE
Product Type	HOOK
Product Model	MIR250
Product Revision	MK1
Applied Application	b3501ce2-35c8-11ee-b6d9-ed24c5d8cf6b
Application Status	ACTIVE
Pending Application	Restore
Application Version	3.2.1.0-gb0fa776c92f1-1-rc.1.3.2.x
Platform Version	3.2.0.0-0-
Current Time	08/08/2023, 11:32:58 (Europe/Copenhagen)

- Reboot Platform
- Restart Application
- Stop Application
- Robot Settings

**FILE UPLOAD**

- Choose Software
- Choose Migration

**SOFTWARE**

Show 10 entries Search:

- 6 Select the Platform software file you want to apply. Once the file has finished uploading, it is displayed under **Software**. If the upload fails, see ["The upload failed" on page 193](#).
- 7 Select **Apply** for the software you just uploaded. The upgrade process is shown in the **Software event log**.

**FILE UPLOAD**

Choose Software

Choose Migration

**SOFTWARE**

Show 10 entries Search:

Time	Guid	Version	Type	Action
08/08/2023, 11:38:09	37ca3736-35cf-11ee-b6d9-ed24c5d8cf6b	3.2.0	Robot Platform	<span style="border: 2px solid red; padding: 2px 5px;">Apply</span> <span style="padding: 2px 5px; margin-left: 5px;">Delete</span>
08/08/2023, 10:52:14	b3501ce2-35c8-11ee-b6d9-ed24c5d8cf6b	3.2.0	Robot Application	<span style="padding: 2px 5px;">Apply</span> <span style="padding: 2px 5px; margin-left: 5px;">Delete</span>
04/08/2023, 13:46:41	496f913c-32bb-11ee-be59-434613c6e282	3.2.0.0-0-	Robot Application	<span style="padding: 2px 5px;">Apply</span> <span style="padding: 2px 5px; margin-left: 5px;">Delete</span>
04/08/2023, 13:36:33	ab224cb8-32ba-11ee-be59-434613c6e282	3.2.0.0-0-	Robot Platform	<span style="padding: 2px 5px;">Apply</span> <span style="padding: 2px 5px; margin-left: 5px;">Delete</span>
03/08/2023, 16:27:48	c011373a-3209-11ee-9fbe-37b9bf818cd1	3.2.0.0-g057982183b8f-4-3.2.x_RCORE-000_npth_ghost-clear	Robot Application	<span style="padding: 2px 5px;">Apply</span> <span style="padding: 2px 5px; margin-left: 5px;">Delete</span>
28/07/2023, 13:38:52	1c0501f8-2d3b-11ee-82db-c9854b9baa39	3.2.0.0-0-	Robot Application	<span style="padding: 2px 5px;">Apply</span> <span style="padding: 2px 5px; margin-left: 5px;">Delete</span>
28/07/2023, 13:37:10	017680d2-2d3b-11ee-82db-c9854b9baa39	3.2.0.0-0-	Robot Platform	<span style="padding: 2px 5px;">Apply</span> <span style="padding: 2px 5px; margin-left: 5px;">Delete</span>
28/07/2023, 10:02:09	e0a6fe86-2d1c-11ee-817d-d791a375865f	3.2.0.0-ge8d4022ab31c-11-rc.11.3.2.x	Robot Platform	<span style="padding: 2px 5px;">Apply</span> <span style="padding: 2px 5px; margin-left: 5px;">Delete</span>

- 8 Wait until the upgrade process changes from **In progress** to **Done**. Refresh the page to upgrade the displayed progress. If the upgrade takes longer than 40 minutes, see ["The upgrade is taking longer than 40 min" on page 194](#).
- 9 Repeat from [step 5](#) with the Application software instead. It is recommended that the Application and Platform software are upgraded to the same version.

10 Under **System overview**, select **Reboot Platform**.



**SYSTEM OVERVIEW**

Hostname	mir-edb-intel-gen1
Rootfs Partition	(A)
Product Config Overlay	ACTIVE
Product Type	HOOK
Product Model	MIR250
Product Revision	MK1
Applied Application	b3501ce2-35c8-11ee-b6d9-ed24c5d8cf6b
Application Status	ACTIVE
Pending Application	Restore
Application Version	3.2.1.0
Platform Version	3.2.1.0
Current Time	08/08/2023, 11:45:11 (Europe/Copenhagen)

Reboot Platform

Restart Application

Stop Application

Robot Settings

### Upgrade the MiR hook software

MiR hook top modules also use both an Application software and Platform software. MiR hook must be running the same software versions as the MiR robot it is mounted to. You must upgrade the robot first, and then you apply the same upgrade to the hook.

Use the platform interface to upgrade the hook software. Use the same process as described in ["Upgrade in the platform interface" on page 189](#) but connect to the hook computer using the address 192.168.12.21:8888 instead. If the robot has lost connection to the hook after updating the robot software, restart the robot and wait 20 minutes to see if the connection is reestablished.



## Troubleshooting

### The upload failed

If the upload fails, the product will usually report an error in the interface. The following four sections describe the most common reasons for errors.

### You were not connected directly to the robot

If you connect to the robot by using its IP address through the local network, the connection can be too unstable to correctly upload the upgrade file. When updating a MiR robot, we recommend connecting to the robot's own network, either using an Ethernet cable connected to the robot or by connecting to a Wi-Fi access point connected to the robot.

### The file failed to upload

After having selected which file you want to upload, if you refresh the page, close down the interface, lose connection, or in some other way interrupt the transfer, the file will not be uploaded. No error message is displayed, and the robot does not begin the upload process. In this case, ensure the connection is stable, and try uploading the file again.

### The file is corrupt

If the product reports that the upgrade file is corrupt, the issue may be resolved with one of the following solutions:

- The file was corrupted while downloading. Often, corrupted files will only be a few KB in size. Download the file again, and try to upload the new file to your product.
- If you are trying to apply an upgrade that is lower than the product's current software version, the upgrade will fail. You must either use a backup to revert to the software version you want the robot to run or revert to an older software version and then upgrade to the desired software version.

### The upgrade failed

If the upgrade failed, it may be because the file is corrupt or you are trying to upgrade to a software version that is lower than the robot's current version—see "[The file is corrupt](#)" above.

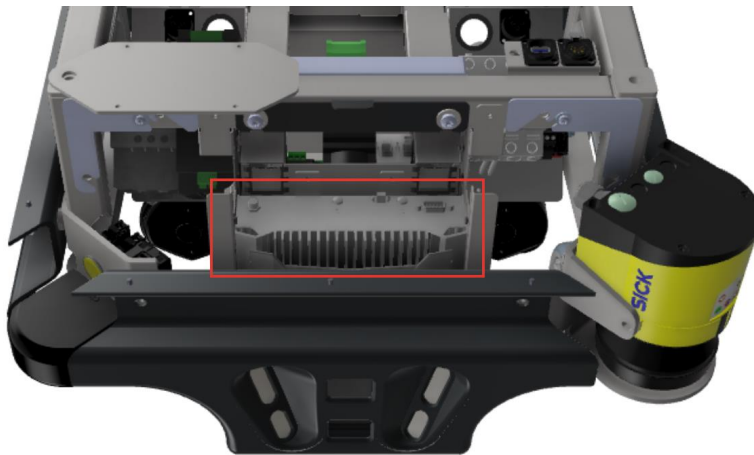
**The upgrade is taking longer than 40 min**

If the upgrade has taken longer than 40 minutes, an issue occurred during the upgrade process. There are several things you can do to determine and resolve the issue:

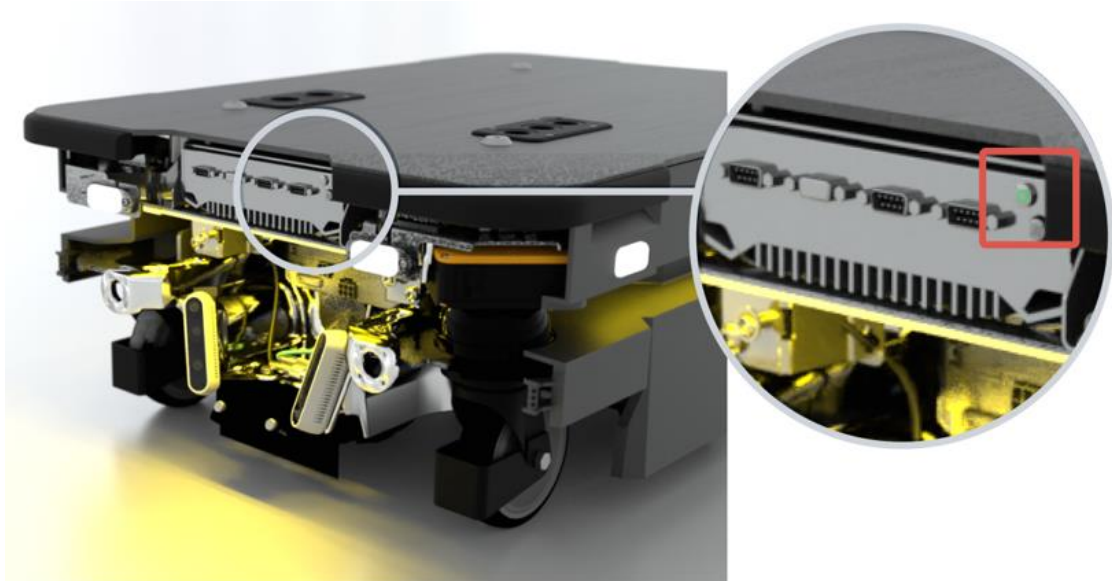
- For MiR250, MiR500, MiR600, MiR1000, and MiR1350, check the buttons on the control panel. If the buttons on the control panel are not lighting up, see ["The buttons in the control panel do not light up" on page 197](#).
- Check if you can connect to the robot interface, and if you can, check if the robot:
  - Is in Protective stop
  - Is reporting errors
  - Has not been upgraded to the desired software version
  - Cannot be operated for some other reason

If the robot behaves as described in any of the points above, see ["You can connect to the robot interface, but the robot is inoperable" on page 197](#).

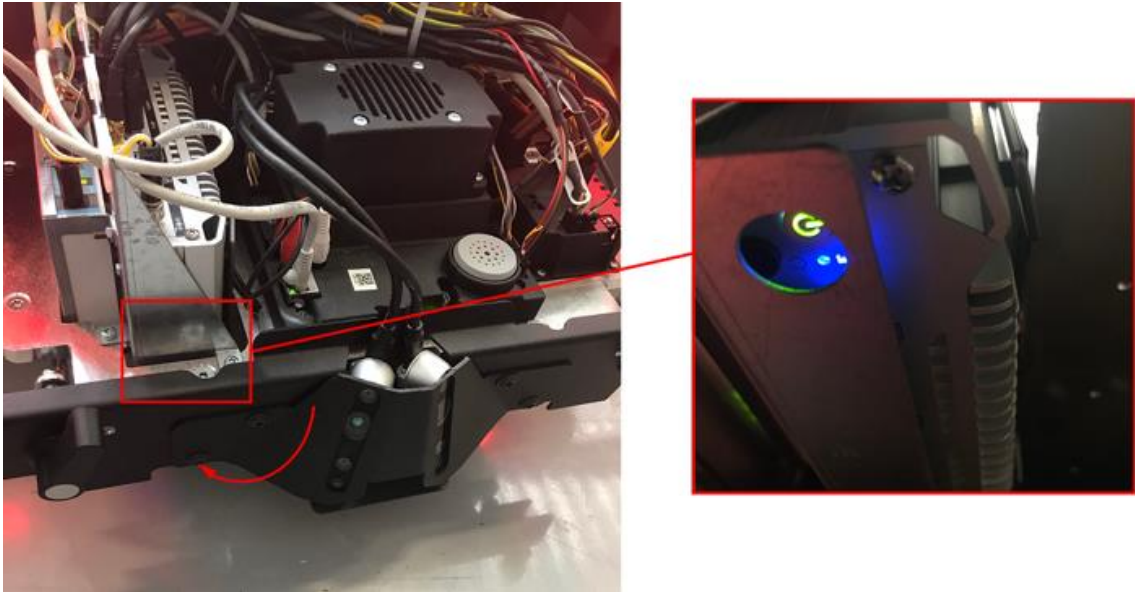
- Check if the robot computer has turned on. If not, see ["The robot computer is off" on page 197](#). To check if the robot computer is on, access the front of the robot, and check if the power button on the computer is lighting up.
- For MiR100 and MiR200, the Power button is facing the bottom of the robot and is difficult to access. If any LEDs are lit on the top of the computer, the computer is on. If not, check the Power button of the robot computer.



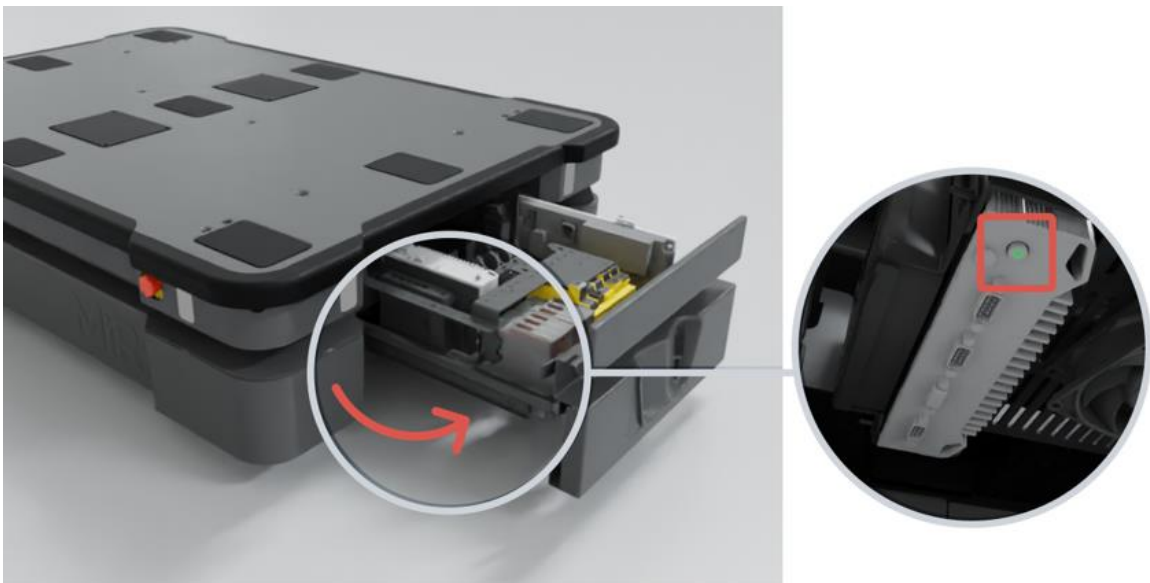
- For MiR250, the Power button is facing the front of the robot.



- For MiR500 and MiR1000, the Power button is facing the bottom of the robot.



- For MiR600 and MiR1350, the Power button is facing the bottom of the robot.



- Check the status lights:
  - If the status lights are wavering yellow, see ["The status lights are wavering yellow" on page 199.](#)
  - If the status light are yellow and the corners flash red 4–5 times, see ["The status lights are yellow with red flashing corners" on page 202.](#)
  - If the status lights are off, see ["The status lights are off" on page 203.](#)

#### **The buttons in the control panel do not light up**

If the buttons on the control panel do not light up, try to disconnect the battery for 30 seconds, then reconnect the battery, wait 30 seconds again, and then turn on the robot. If the robot still does not turn on, try connecting a cable charger to the robot. Use one of the recommended MiR cables chargers: MiR Cable Charger Lite 48V 3A or MiR Battery Charger 48V 12A. Before connecting the charger to the robot, make sure to:

- Connect the charger to a power source.
- Verify that the charger is powered.
- Verify that the battery is connected to the robot.

If the battery does not charge, contact MiR Technical Support.

#### **You can connect to the robot interface, but the robot is inoperable**

If you can access the interface, you can check the current state of the robot's components under **Monitoring > Hardware health**. After a software upgrade, sometimes the robot may show errors where your robot just needs time to finish updating and starting up sub-components. Wait 20 minutes for the software to finish starting up. Do not restart the robot.

If there are still any reported errors, you may be able to use one of the troubleshooting guides or error lists available on MiR Support Portal to resolve the issue. If you are in doubt about whether a warning under Hardware Health is significant, contact MiR Technical Support.

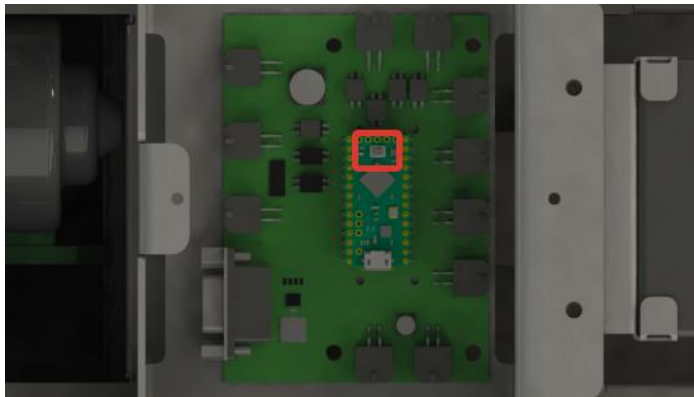
#### **The robot computer is off**

The process for troubleshooting this issues depends on which robot you have. Follow the instructions in the section for your robot.

### **MiR100 and MiR200**

For MiR100 and MiR200 you can check multiple components by following these steps:

- 1 Disconnect the battery for 30 seconds, then reconnect it.
- 2 Turn on the robot, and listen for a clicking noise.
  - If you do not hear a clicking noise and the robot does not turn on, check that the main fuses are connected. If they are connected and you still do not hear a clicking noise when you turn on the robot, contact MiR Technical Support.
  - If you hear a clicking noise, check if the lights on the MiR board in the center of the robot are on.
    - If the lights are off, turn on the robot computer manually.
    - If the lights are on, remove the metal mesh case around the MiR board by unscrewing the four screws first. Then, press the Reset button on the MiR board.



After you pressed the Reset button, the status lights should react. Wait a few minutes for the robot to finish starting up. Once the robot is functional, you can safely turn it off and on. If not, contact MiR Technical Support.

### **MiR250, MiR500, MiR600, MiR1000, and MiR1350**

Turn on the robot computer manually by pressing the Power button on the computer. The robot should start up correctly now.

If the robot computer continues to fail to turn on when you turn on the robot, contact MiR Technical Support.

**The status lights are wavering yellow**

If that status lights are wavering yellow, and the robot computer is turned on, then there may be a problem with the connection from the power board to the robot computer via the router. There are three common causes for this:

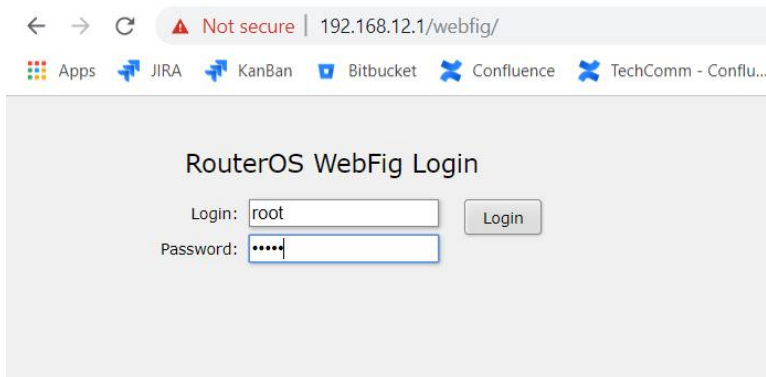
- If you have connected to the robot directly using an Ethernet cable, check that the Ethernet cables for the robot computer and power board are plugged in to the router securely.
- If the robot is MiR250, MiR500, MiR600, MiR1000, or MiR1350, the software upgrade might have included an upgrade for the power board. Turn the robot off, disconnect the battery, and wait one minute for the power board to shut down completely. Reconnect the battery and turn on the robot. The power board should be upgraded now and the robot should run.

- The robot computer may have been assigned another IP address during the upgrade. To check this, follow these steps:

- 1 Connect to the robot's Wi-Fi if you have not already done so. Open a browser window, and go to the address **192.168.12.1/webfig/** that connects you to the router.
- 2 Sign in using the credentials:

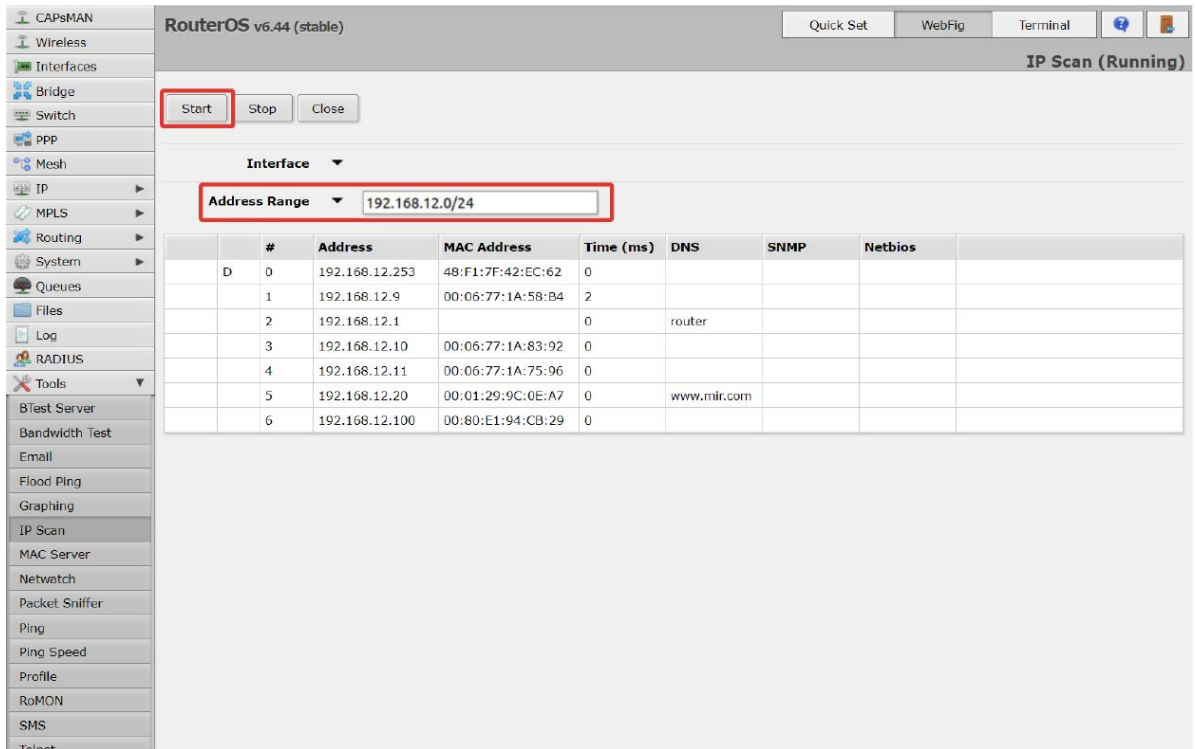
**Login:** root

**Password:** mirex





- 3 Go to **Tools > IP Scan**, set the **Address Range** to **192.168.12.0/24**, and select **Start**.



The screenshot shows the RouterOS v6.44 (stable) web interface. The 'Tools' menu is open, and 'IP Scan' is selected. The 'IP Scan (Running)' window is displayed, showing a 'Start' button highlighted with a red box. The 'Address Range' dropdown is also highlighted with a red box and set to '192.168.12.0/24'. Below the dropdown, a table displays the results of the IP scan.

	#	Address	MAC Address	Time (ms)	DNS	SNMP	Netbios
D	0	192.168.12.253	48:F1:7F:42:EC:62	0			
	1	192.168.12.9	00:06:77:1A:58:B4	2			
	2	192.168.12.1		0	router		
	3	192.168.12.10	00:06:77:1A:83:92	0			
	4	192.168.12.11	00:06:77:1A:75:96	0			
	5	192.168.12.20	00:01:29:9C:0E:A7	0	www.mir.com		
	6	192.168.12.100	00:80:E1:94:CB:29	0			

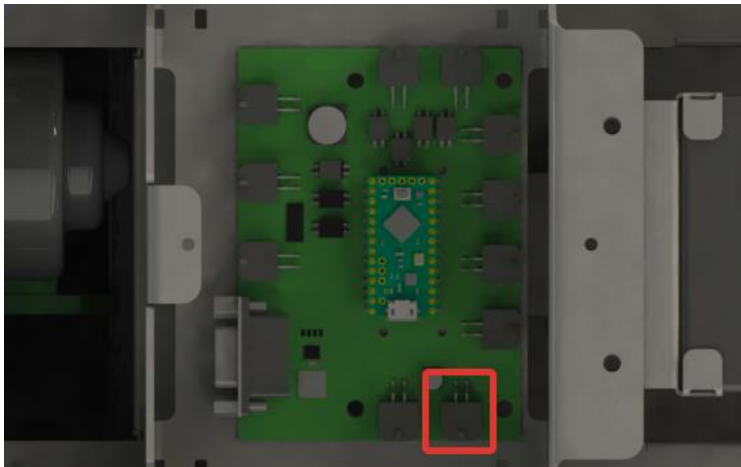
- 4 In the list of connected devices, check if the following IP addresses are listed:
- The IP address of any devices connected to the Ethernet top interface
  - 192.168.12.1—the router
  - 192.168.12.9—the safety PLC
  - 192.168.12.10—the front safety laser scanner (are not found on MiR100 and MiR200 robots)
  - 192.168.12.11—the rear safety laser scanner (are not found on MiR100 and MiR200 robots)
  - 192.168.12.20—the robot computer
  - 192.168.12.21—the MiR hook if one is mounted to your robot
  - 192.168.12.100—the power board

- 5 If 192.168.12.20 is missing and is replaced with another IP address that is not listed, you will have to run the set up script for the router by following the guide *How to set up the router or Ethernet switch on MiR robots*. You can find this guide on [MiR Support Portal](#).  
If 192.168.12.20 is missing and only the other listed IP addresses are shown, the router is not connected to the robot computer. Check the Ethernet cable from the router to the computer—see your robot's wiring diagram if you are uncertain which cable it is. You can find this guide on [MiR Support Portal](#).

### The status lights are yellow with red flashing corners

If the status lights are yellow and the corners are flashing red on MiR100 or MiR200 this indicates that there was an issue with a MiR board firmware upgrade. To resolve this, follow these steps:

- 1 Turn off the robot, and remove the top cover.
- 2 Disconnect the power cable from the MiR board. The connector for the power cable is indicated in the following image.



- 3 Turn on the robot. The robot will not finish starting up completely.
- 4 Turn on the robot computer manually by pressing the power button on the computer.

- 5 Reconnect the power cable to the MiR board.
- 6 Wait 20 minutes for the MiR board to upgrade.
- 7 Restart the robot.

#### **The status lights are off**

If the status lights are off, the robot may still be updating. Wait another 20 minutes, and if the robot does not turn on by itself, press the Power button to turn the robot on manually. If the robot does not turn on, contact MiR Technical Support.

#### **The software version has not been upgraded after restarting the robot**

If the upgraded version of the robot software is corrupted, the robot will not start up with that version. Retry the upgrade.

### **Upgrade MiR Fleet software**

To upgrade MiR Fleet, download and run the installer of the latest version—see "[Install MiR Fleet](#)" on [page 217](#). The installation file will detect if an existing instance of MiR Fleet is on the device and will update the software of the existing application.

### **Downgrade robot software**

Before downgrading the software check the following:

- The software you want to downgrade to is supported on all the robots in your fleet. Some robot hardware versions are only compatible with later software versions. This is indicated in the hardware release note. You can find the release notes on [MiR Support Portal](#).
- You are not downgrading from software 3.x or MiR Fleet Enterprise to software 2.x. If you want to downgrade to software 2.x, you must contact MiR Technical Support.
- There are no critical issues that have been solved in the software versions released after the version you are downgrading to.

If you want to downgrade the robot software, use the Platform interface to apply an older software version—see "[Upgrade in the platform interface](#)" on [page 189](#).

Follow the same steps as you would when upgrading to a newer software. Always start with downgrading the Platform software before the Application software.

When you downgrade, all of the robot settings and calibrations are saved.

If you downgrade from MiR Fleet Enterprise software to 3.x software, none of the site data is migrated since all site data is saved on MiR Fleet.

### 4.3.4 Design top modules

#### Section overview

Purchase or design a top module for the robot tasks.

- Determine which electrical interfaces on the robot the top module will need.
- Choose a top module design that is compatible with your MiR robots.
- Evaluate if the combined top module and robot application meets relevant standards.

A top module is any device you mount on a MiR base robot. The top module defines the function of the robot. There are the following options for acquiring a top module:

- Purchase MiR robot applications where a top module is already mounted to the robot, such as shelf carriers, hooks, and pallet lifts.
- Purchase a top module from many third-party suppliers at [MiR Go](#).
- Design your own top module.

If you design a custom top module, MiR takes no responsibility for the design or compliance of the application.

Your top module must meet all electrical current, voltage, stability, payload, and size requirements. All specifications are described in the user guide or integrator manual for your robot base model. Different robot models have different specifications. A top module designed for a specific MiR robot model is not necessarily compatible with other models.

When you run brake tests with your MiR robot, include the top module and its expected payload—see "[Test braking distance](#)" on page 448.

#### Design considerations

When designing top modules for MiR robots, consider the following sections:

### Size

If the size of the top module exceeds the physical dimensions of the robot, you must update the footprint and the Protective fields accordingly—see ["Create footprints" on page 404](#) and ["Adjust Protective fields" on page 457](#).

### Payload, weight, and stability

Find the maximum payload and the CoM load triangle for your robot application in its user guide or integrator manual. The maximum payload of your robot includes the weight of the top module itself.

### Actuators and power

- Use the Power interface on the robot to power actuators on the top module.
- Use actuators that revert to a safe state (often mechanical brakes or blocks) when they lose power.
- Connect actuators to the 48V Safe Power output on the robot. This ensures that the top module reverts to a safe state when the robot is powered off or when it enters Emergency or Protective stop.
- If you have actuators that need to be powered at all times, regardless of whether the robot is in Protective or Emergency stop, use the 48V power output and enable the feature **Turn off TOP FUSE** in the robot interface. Only do this if you are certain that the top module actuator you have connected it to does not pose any risk to nearby personnel in any unexpected events.
- Verify and limit inrush current and install a pre-charge circuit if necessary.
- Use fuse and diode protection to prevent damage to the robot.
- Ensure compliance with top interface specifications. See the user guide or integrator manual for your robot application for the appropriate specifications. Specifications can also be found on [MiR Support Portal](#) under **Specifications**.

### I/O signals

Use the I/O interface on the robot to send signals between the robot and the top module.

Always document what each input signal activates on the top module and what state the output trigger indicates to the robot.

Make missions for each top module actions so you can easily add top module control to other missions without looking up what functions each I/O port triggers or indicates.

### Wi-Fi and antenna

The design of the top module mounted to your MiR robot also influences the quality of the Wi-Fi connection. The following list provides some of the minimum requirements you should consider. MiR recommends that you contact a Wi-Fi specialist to help determine the optimal equipment and setup for your site.

- To avoid invalidating compliance with radio emission regulations, such as FCC and CE, use the antennas that are already mounted to the robots.
- Do not surround antennas with conductive, dense, or metallic material. They will have a negative effect on the performance of the antennas.
- You can relocate antennas by extending their connection with cables.  
Keep antenna cables as short as possible and with the fewest connection point. The longer the cable and the greater number of connection points, the more the signal degrades.
- Use 50  $\Omega$  coax antenna cables.
- Select equipment that is compliant and suited for the frequency band your site or application uses.
- If your top module manages its own Wi-Fi connection, consider aspects such as:
  - Range
  - Radiation pattern (often you want to use omnidirectional antennas)
  - Frequency band
  - Efficiency
  - Impedence
  - Compliance with standards
  - Interference between the robot and other radio device on the top module
  - Gain
  - Polarization
  - Voltage standing wave ratio (VSWR)

### Safety and compliance

The robot and top module must be treated as one combined application when assessing the safety and compliance.

For all standards that the robot and top module application comply with, draw up a statement of conformity and store the statement of conformity for at least 10 years as part of the technical file and make it available upon request.

The two main standards that a MiR base robot comply with are EMC and Radio legislation. Verify that the robot still complies with these standards after the top module is mounted to it. You must draw up a new declaration or statement of conformity for the combined application.

When designing a top module within the European Union, verify that the application meets the requirements in the Machinery Directive.

### **MiR top module best practices**

The following sections outline best practices to keep in mind when using any of the MiR top modules.

#### **Shelf carrier**

- Use the correct offsets for custom carts.
- Use non-reflective, gray paint for the shelf legs.

#### **Hook**

- Use April tags instead of QR codes.
- Mount April tags close to hook gripper bar.
- Set cart dimensions and calibrations accurately.
- Set the Drive and Lock height so the hook gripper is NOT supporting the weight of the cart.
- Manufacture carts and ID tags in consistent quality.
- Make ID tags non-reflective and mounted so they are perpendicular to the floor.
- Set cart pickup position in locations where the area behind the ID tag is static.
- Create a hook dashboard with missions such as pick-in-place, drop-in-place, reverse with and without release for easier recovery and troubleshooting.
- Ensure sufficient space is available for cart drop-offs and robot movement after drop-off.
- Ensure that the floor is level at pickup and drop-off areas.
- Ensure cartwheels are level and not prone to tilting.
- Ensure a robust preventative maintenance plan for carts is in place.
- Use backstops, grip tape or similar to prevent carts from moving before, during, or after pickup or drop-off.
- Ensure fixed wheels are in the back for carts larger than 0.8 meters in length.

- Ensure proper cart grounding.
- Use drive through drop-off areas instead of reverse drop-off to avoid space and cycle time issues.

#### Pallet lift

- Use non-reflective, gray paint for the racks.
- Do not drive with the lift in raised position to avoid reduced driving speed, pallet tipping, premature actuator failure.
- Ensure pallet racks are within specification to use the Check position status action. The Check position status action only works on standard pallet racks and not Bar markers. See *How to use Check position status* for more information. You can find this guide on [MiR Support Portal](#).
- Use sensors and WISE modules to detect pallets for bar markers if you do not want to use Check position status.
- Consider using sensors and WISE modules even for pallet racks as a fail-safe.
- Consider using template missions for pallet pickups and drop-offs for easier, quicker, and less complicated missions.

#### Shelf lift

- Do not use shelf lifts for picking and moving pallets as safety field sets are set for shelf legs and the payload capacity is different.

## 4.4 Set up MiR Fleet

### **After reading this chapter, you can:**

Set up your MiR Fleet Enterprise server to meet IT requirements.

**Configure your network and server to meet the MiR system requirements.** See "[Meet system requirements](#)" on the next page.

- Make sure your server and network structure meet the requirements for MiR systems.
- Ensure robots have a fixed IP address using a method that is approved by your IT department.



**Install MiR Fleet on a prepared server.** See ["Install MiR Fleet" on page 217.](#)

- Remove previous MiR Fleet software versions.
- Install Microsoft .NET Server Hosting and Microsoft SQL server.
- Create a user to run MiR Fleet Windows Service.
- Install MiR Fleet on the selected host server.

**Configure MiR Fleet according to site cybersecurity policies and preferences.** See ["Configure MiR Fleet" on page 234.](#)

- Configure firewalls, TLS, and SSO.
- Connect an SQL database.

**Sign in the first time.** See ["Sign in" on page 254.](#)

- Sign in to the MiR Fleet interface.
- Create an admin user.

### 4.4.1 Meet system requirements

#### Section overview

Configure your network and server to meet the MiR system requirements.

- Make sure your server and network structure meet the requirements for MiR systems.
- Ensure robots have a fixed IP address using a method that is approved by your IT department.

For the MiR system to run smoothly, the host system must meet the requirements outlined in this section.

The scale and number of connected devices to the MiR system affects the host system requirements. If you expand the MiR system, you must revisit the performance specifications of your system to verify that the system can support the expansion.

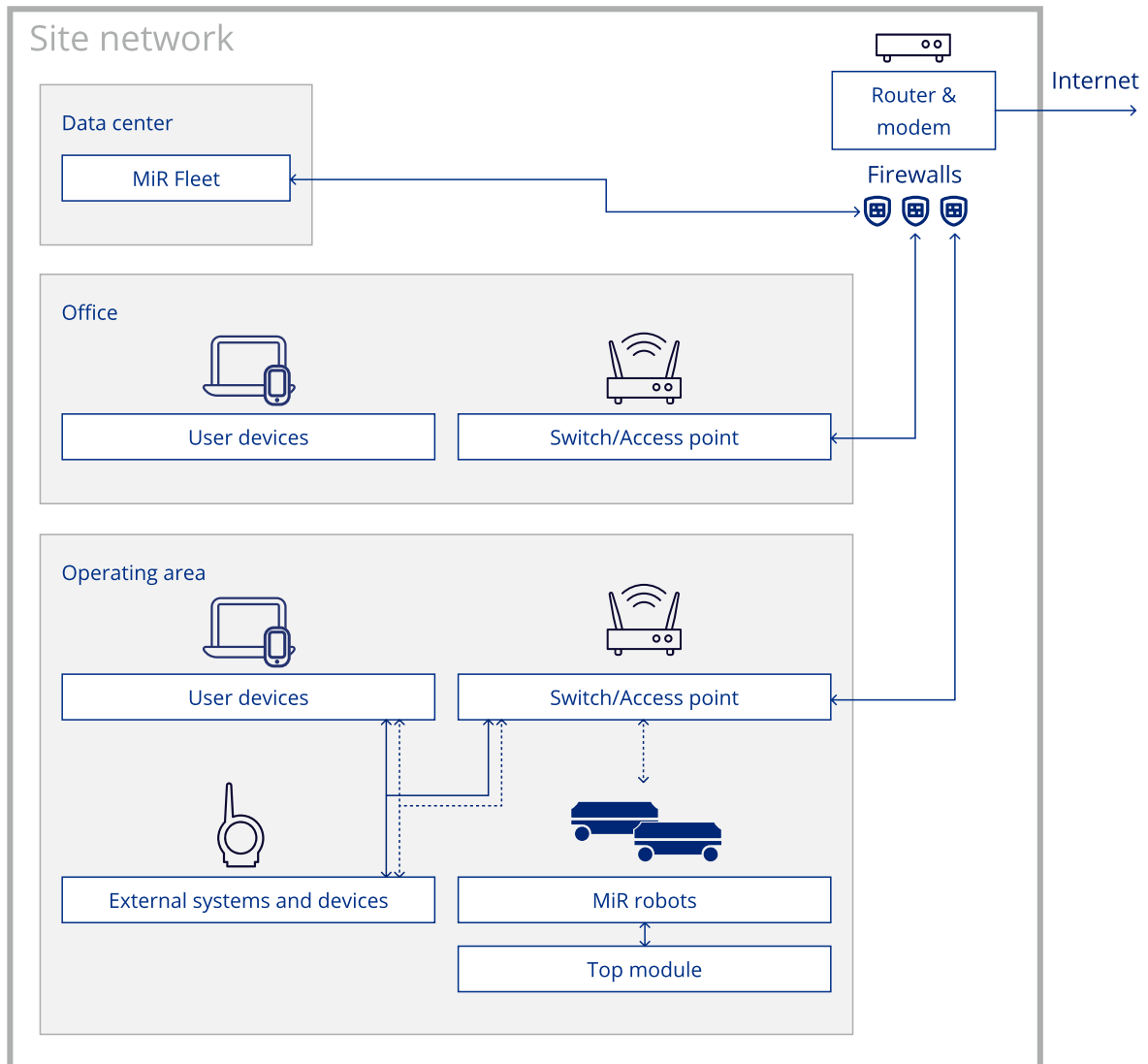
### MiR Fleet server minimum requirements

CPU	Intel Core i7
RAM	8 GB
Storage	100 GB
OS	Windows Server 2022

### Network requirements

All devices on your MiR system must be connected to the same network:

- The MiR Fleet server must use a wired connection to the network.
- All MiR robots must be connected to the network through wireless access points.
- Additional devices can be connected using either option.



These are the minimum requirements for the network in all areas where robot operate. When testing that these requirements are met, take all additional devices you will connect to the MiR system into account as well.

**Table 4.1** Guidelines for Wi-Fi requirements

Parameter	Description	Requirement
Signal strength	The signal strength from the robots' perspectives when connected to the best access point.	Min. -70 dBm
Secondary signal strength	The signal strength from the robots' perspectives when connected to the second best access point.	Min. -75 dBm
Signal to noise ratio	Signal to noise ratio from the robots' perspectives.	Min. 20 dBm
Data rate	Rate of data communicated to and from each robot. This is the transfer speed of communications.	Min. 20 Mbps
Channel interference	Number of access points per channel available when signal strength is lowered to -85 dBm	Max. 2 AP/ch at -85 dBm
Latency (round trip time, ping)	Time taken to send and receive messages from robots by, for example, pinging them.	Max. 200 ms

**Table 4.2** Guidelines for general network requirements

Parameter	Description	Requirement
Throughput	<p>The maximum potential data that can be transferred over a period of time. Must support 20 Mbps for each robot on the network.</p> <p>The data rate and throughput requirements should be interpreted as that the network must support robots sending and receiving at least 3 Mbps of data with a transfer rate of at least 20 Mbps.</p>	Min. 3 Mbps/robot
Packet loss	Percentage of communication packets that can be lost.	Max. 2%

Additionally, your network must fulfill the following requirements:

- There must be full Wi-Fi coverage throughout the traveling path of all robots. You can evaluate this using heatmaps—[Wi-Fi heatmap](#).
- There must be a WLAN controller to secure that roaming happens at the correct time and without any authentication errors. Make sure that the company network access points are controlled by the same controller.
- The access points must be set up to communicate and share roaming information.
- Make sure that load balancing on the access points is disabled. The robots must be able to roam freely. Load balancing controls the balancing of traffic, for example, between the 2.4 and 5 GHz bands of your network. Note that terminology may vary between manufacturers of routers and access points.
- Airtime balancing must be disabled if possible. This setting changes how much time a device gets on the network depending on the signal strength. Low-signal devices get less time than high-signal devices. This means that a robot that is far away will be allowed less data than a robot that is close by. Note that terminology may vary between manufacturers of routers and access points. Airtime balancing may also be called airtime fairness by some manufacturers.

The robots are able to perform some tasks under worse conditions than mentioned above. However, key features in MiR Fleet such as Collision avoidance, Auto charging and staging, Limit-robots zones, data synchronization between robots, and fleet mission execution will not work optimally or at all.

If more than 60% of the Wi-Fi channel is being used, the MiR system will be affected by latency and packet loss.

### IP addresses

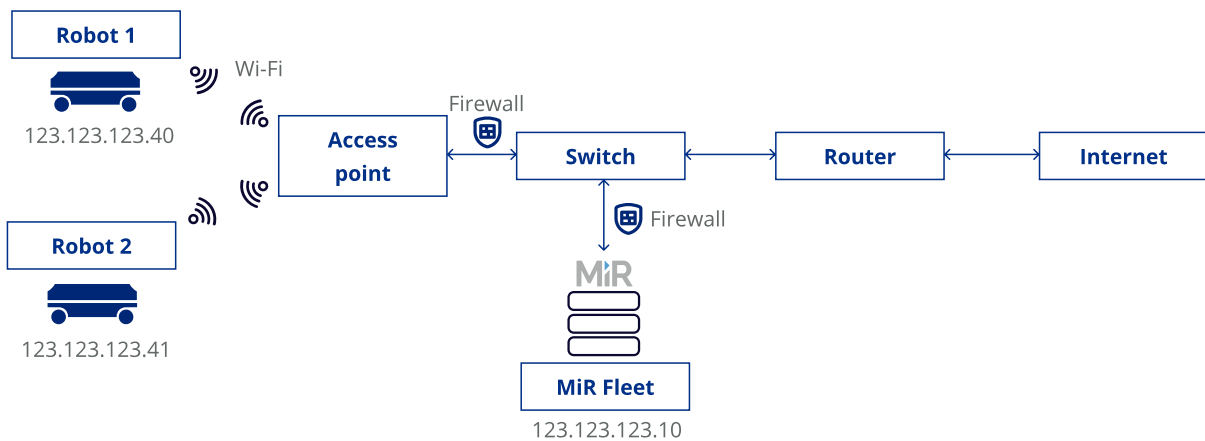
MiR Fleet identifies robots by their IP address. Robots must keep a permanent IP address on the network once connected to MiR Fleet.

Contact your IT department to determine the best solution for your network structure. To avoid conflicts, make sure unique IP addresses are mapped to specific devices.

There are two main ways to ensure IP addresses on devices are set: use static IP addresses or DHCP reservations.

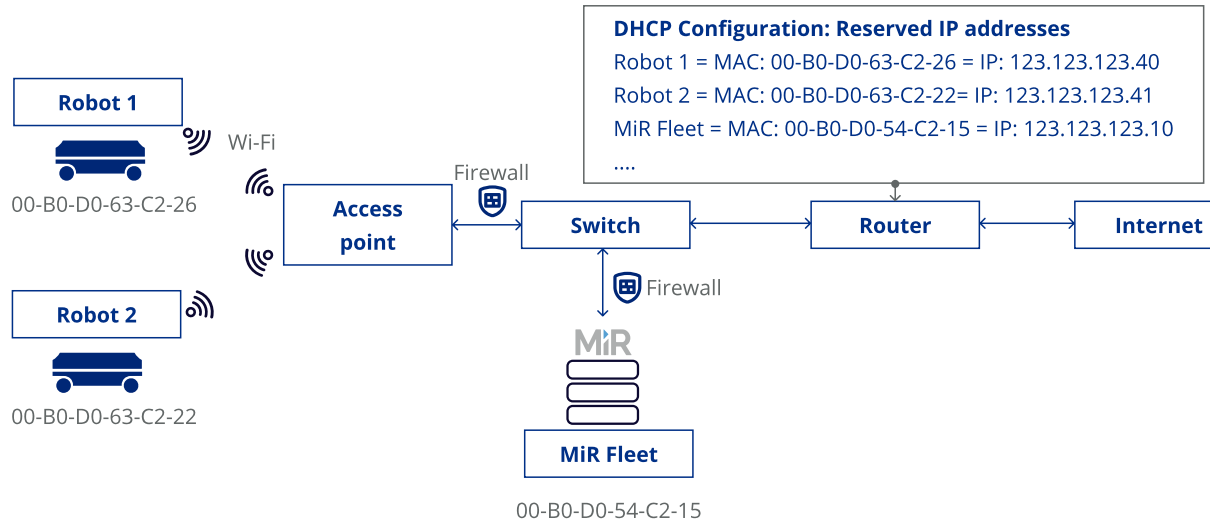
When you use a static IP solution, you set the IP address of the device on the network and configure the system not to change the IP address. You must keep track of which IP addresses have been assigned to avoid assigning the same address to multiple devices.

**Figure 4.2** Diagram of a static IP solution



When you use DHCP reservations, in the DHCP server's configuration, you reserve an IP address by binding it to the MAC addresses of robots. The network router automatically assigns the reserved IP addresses to the relevant devices. This protocol reduces the risk of assigning the same address to multiple devices.

**Figure 4.3** Diagram of a DHCP reserved IP solution



If you experience issues with devices that cannot be reached on your network, contact your IT department to ensure the devices are on the correct network, assigned appropriate IP addresses, and the firewalls are configured to allow communication to MiR products—see "Ports" below.

**Ports**

Table 4.3 identifies all ports used in the communication between MiR devices. Contact your IT department to configure the network firewalls to allow MiR device communications. You can configure MiR Fleet to use another port than 443 if necessary—see "Configure MiR Fleet" on page 234.

**Table 4.3** Ports used by the MiR Fleet server

Service	Target	Direction	Port
Data	SQL database server	Outbound	1433/tcp
Robot communication	Robots	Outbound and inbound	5060/tcp

Service	Target	Direction	Port
Web interface	User device	Inbound	443/tcp
MiR Fleet Integration API	External devices	Inbound	443/tcp
Data monitoring	MiR Insights Online	Outbound	443/tcp, 5671/tcp, 5672/tcp
Shop Floor connector—see <a href="#">"Shop Floor Connector" on page 596</a>	External devices	Inbound	Custom

**Table 4.4** Ports used by the MiR robots

Source	Target	Direction	Port
MiR Fleet communication	MiR Fleet	Outbound and inbound	5060/tcp
Web interface	User device	Inbound	443/tcp
Top module API	Top module	Inbound	HTTP: 8080/tcp, HTTPS: 443/tcp
Modbus	Top module	Inbound	502/tcp
Remote support	Technical support	Outbound	443/tcp



## 4.4.2 Install MiR Fleet

### Section overview

Install MiR Fleet on a prepared server.

- Remove previous MiR Fleet software versions.
- Install Microsoft .NET Server Hosting and Microsoft SQL server.
- Create a user to run MiR Fleet Windows Service.
- Install MiR Fleet on the selected host server.

While installing MiR Fleet, use Event Viewer to see more information about any critical errors or warnings that may occur during the installation.

By default, MiR Fleet is installed under **C:\Program Files\MiRFleet\fleet**.

MiR Fleet runs as a Windows service on Windows Server 2022. This means:

- You can easily install it by running an executable file.
- You can find the software under Apps & Features on your Windows host server, from where you can manage and uninstall it.
- MiR Fleet always starts up automatically when the operating system starts up.
- You can start, stop, and restart MiR Fleet under Services on your Windows host server.

### Before installing

Before installing MiR Fleet, you must complete the following prerequisites.

#### Previous installations

If you have installed a previous version of MiR Fleet based on 2.x or 3.x software on the host, remove it by following the guide *How to uninstall MiR Fleet Server Solution*. You can find this guide on [MiR Support Portal](#).

If you have previous version of MiR Fleet Enterprise installed on the server, running a new installer will update your existing installation.

### Microsoft .NET Server Hosting

The MiR Fleet host server must have .NET 8 Server Hosting installed. You can check if you already have .NET 8 Server Hosting by searching for it under Apps & Features on your Windows host, or run the command:

```
dotnet --info
```

If you do not have .NET installed, download it from Microsoft here: <https://dotnet.microsoft.com/en-us/download/dotnet/thank-you/runtime-aspnetcore-8.0.8-windows-hosting-bundle-installer>.

### Microsoft SQL Server

The MiR Fleet database must be hosted on a server with Microsoft SQL (MSSQL) Server installed. You can either host the database locally on the same server as MiR Fleet, or you can host it on a separate device. You can select the database location during installation or you can configure it after the installation—see "[Configure MiR Fleet](#)" on page 234.

You can check if you already have Microsoft SQL Server by searching for it under Apps & Features on the Windows host where the Microsoft SQL Server instance is supposed to run.

If you do not have MSSQL installed, download it from [Microsoft.com](https://www.microsoft.com).



#### NOTICE

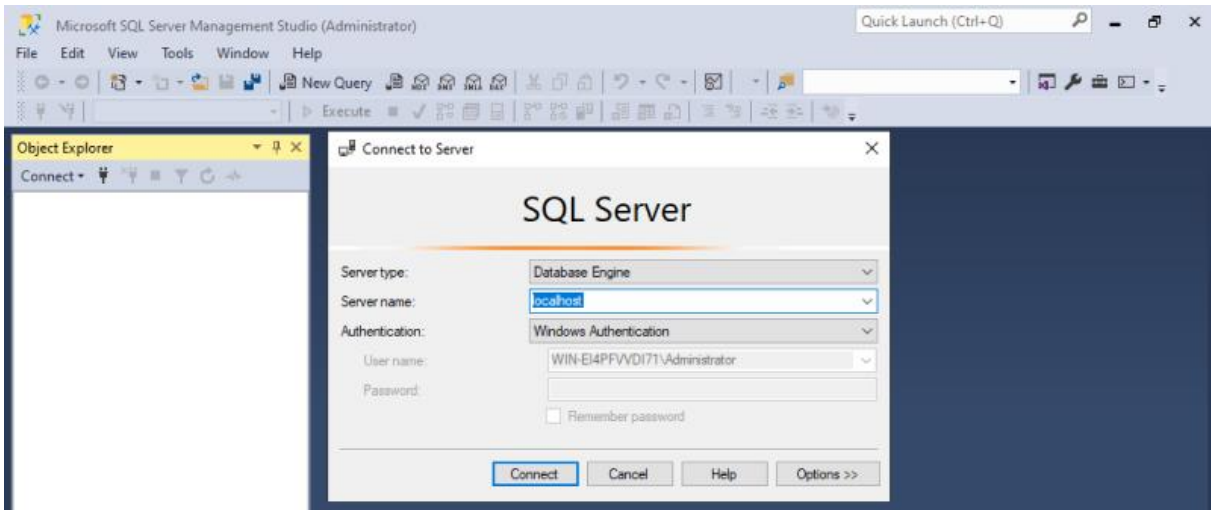
Restart your host server after installing MSSQL. The MiR Fleet installation will fail if you have not restarted the server.

### Server Management Studio

We recommend installing SQL Server Management Studio (SSMS) after the SQL installation is complete. SSMS can be used to view the database behind MiR Fleet and troubleshoot issues.

To access the database with SSMS, use the following connection parameters:

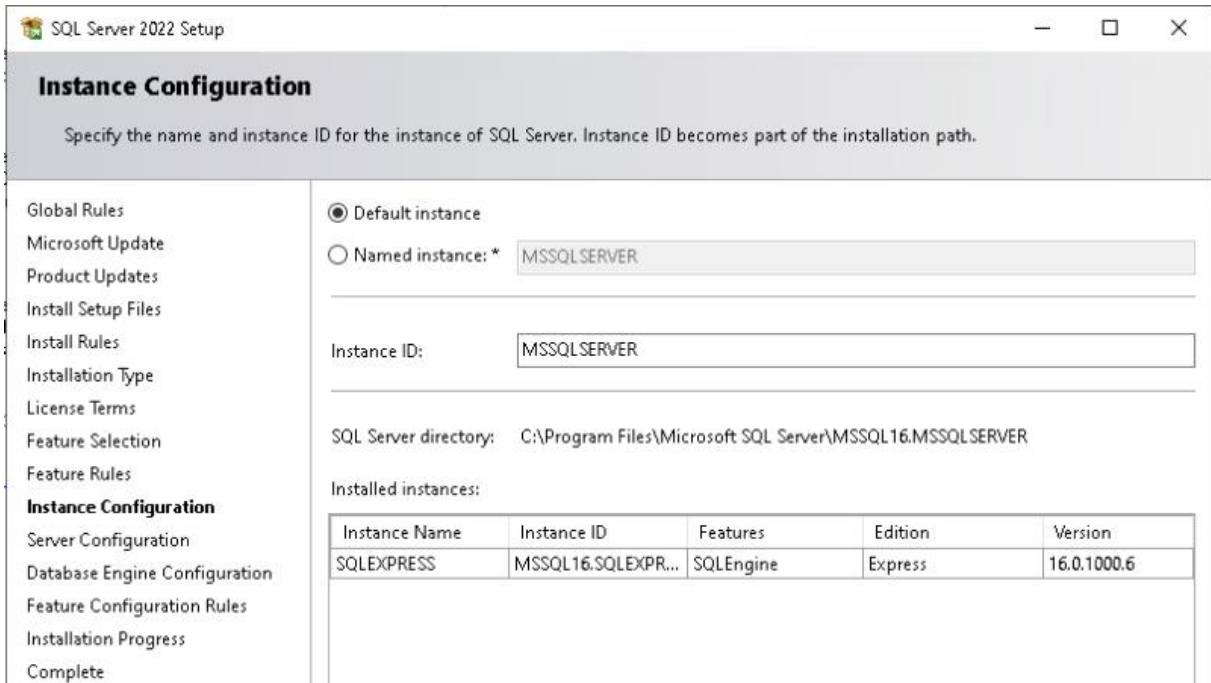
- **Server type:** Database Engine
- **Server name:** localhost
- **Authentication:** Windows Authentication



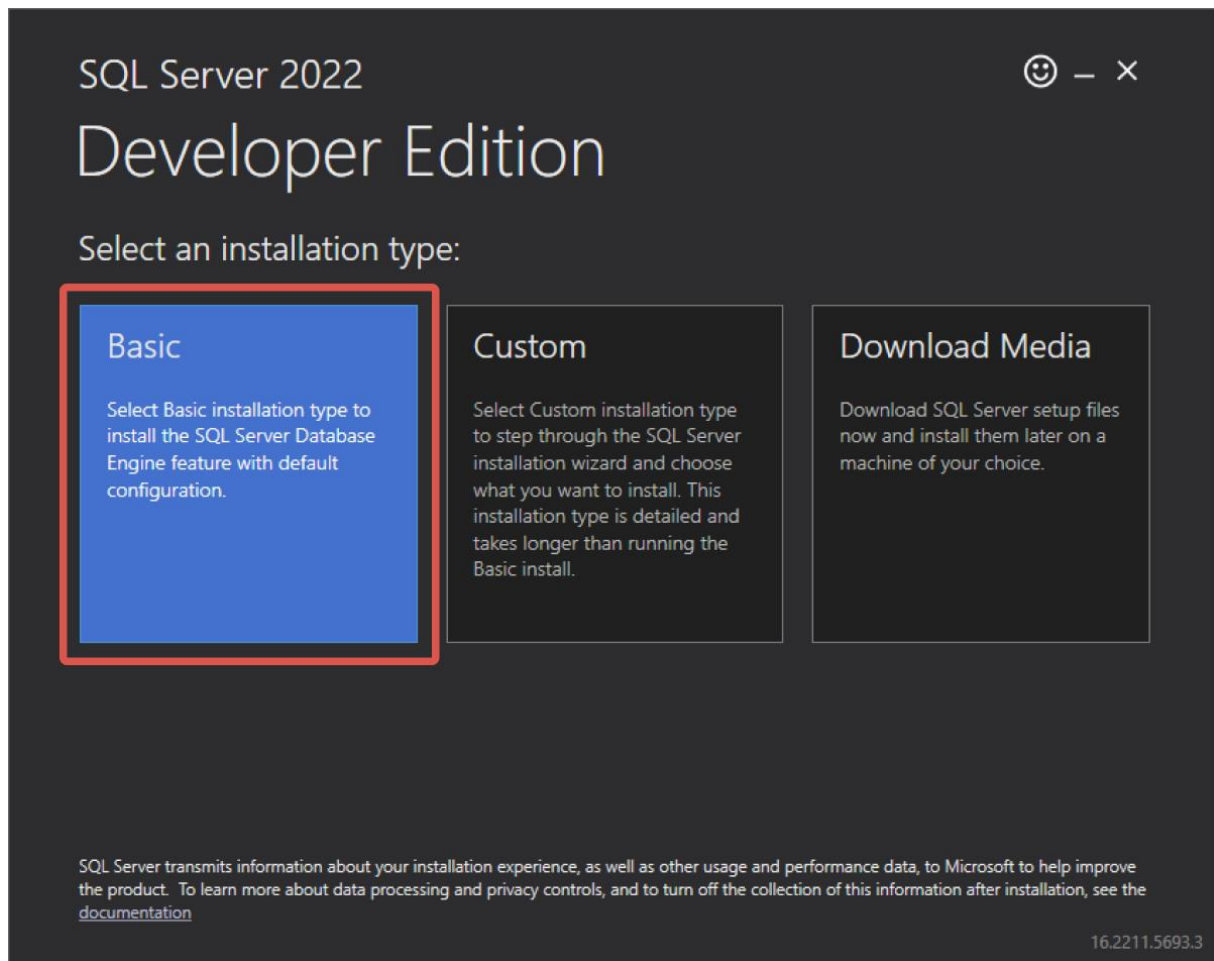
**Free editions**

You can use the free editions of MSSQL, but they are not intended for large-scale production environments.

If you use the Express edition, the default hostname is localhost\SQLEXPRESS. If installed on the same host server as MiR Fleet, choose the **Custom** installation type and select **Default instance** when setting up the instance configuration.



If you use the Developer edition, choose the **Basic** installation type



### User for MiR Fleet Windows Service

When installing MiR Fleet, you select which user to associate with MiR Fleet. This is the service account MiR Fleet will use to run as a Windows server. We highly recommend to use a Managed Service Account (MSA) for running MiR Fleet. MSAs provide automatic password management, secure credential generation, and management delegation capabilities.

The following user types are supported:

- A local service account can be used if both MiR Fleet and Microsoft SQL Server run on the same host machine.
- A username and password based account or an Active Directory (AD) account can be used if

both MiR Fleet and the Microsoft SQL Server run on the same host machine.

- An organization account or Group Managed Service Account (gMSA) must be used if the Microsoft SQL Server runs on a separate machine.

MiR Fleet uses Windows Authentication to authenticate to the MSSQL database. The chosen service account must have access to the Microsoft SQL Database. If your database is on another server, gMSAs are an easy way to use the same secure account to access resources on multiple servers.

If you have not already created a database for MiR Fleet, the user associated with your MiR Fleet installation is used to set up the database automatically when installing MiR Fleet and will be the owner of the MiR Fleet database. Depending on your IT setup, you may need to modify the user permission to ensure:

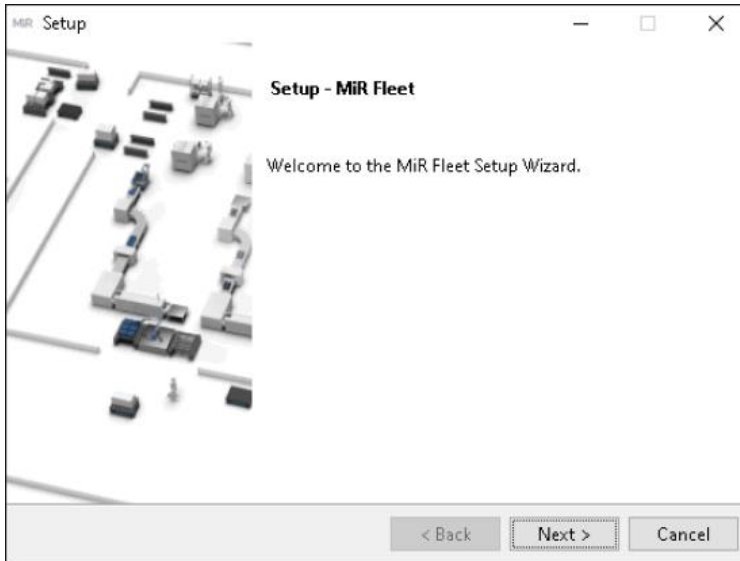
- The user has the necessary permissions to edit the database and to be the owner of it.
- The user has the right **Log on as a service**.

## Install MiR Fleet

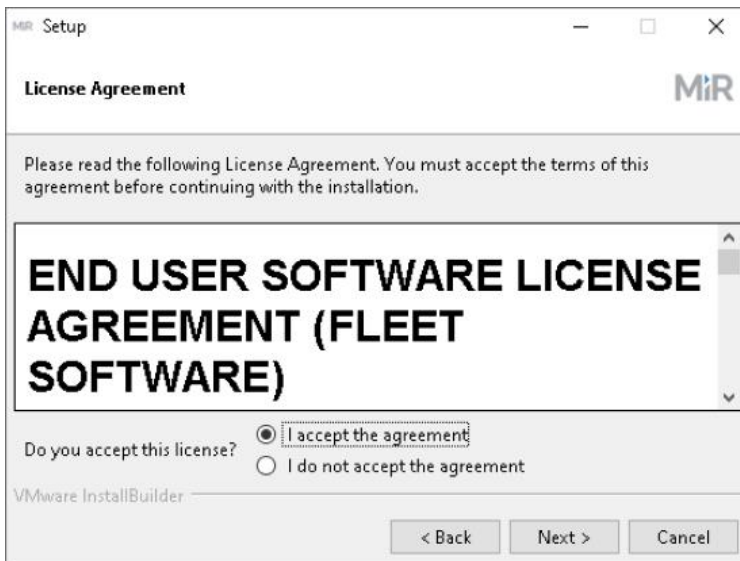
To install MiR Fleet, follow these steps:

- 1 Verify that you have:
  - Checked that network and server host meet the requirements.
  - Removed all previous MiR Fleet installations.
  - Installed .NET server hosting.
  - Installed Microsoft SQL Server.
  - Created a Windows Service user for MiR Fleet.
- 2 If you have just installed any of the application from the previous step, restart the host server before starting the MiR Fleet installation.

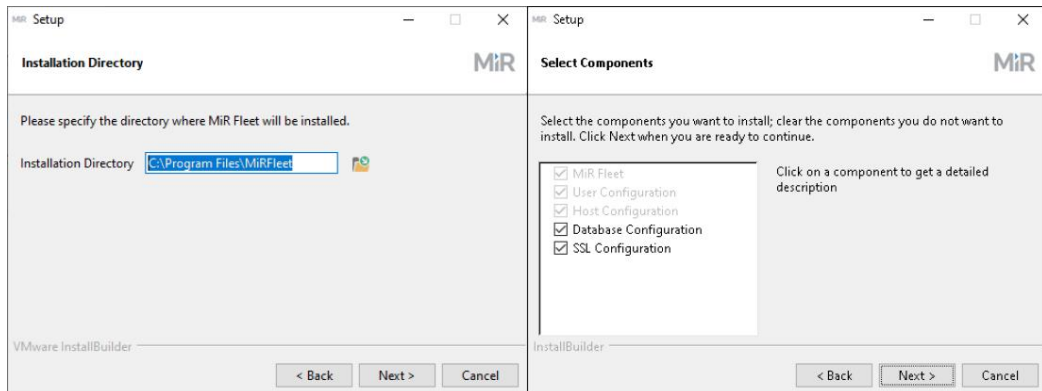
- 3 Run the MiR Fleet installer file.



- 4 Accept the end user license agreement.

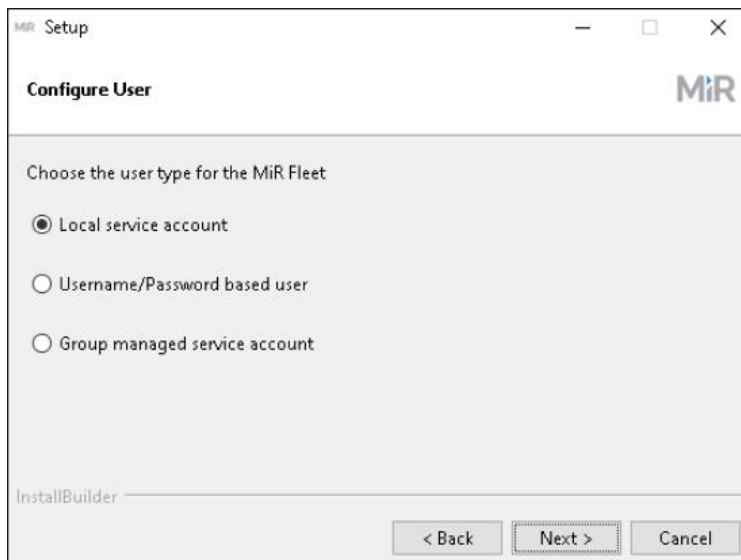


- 5 Select the install destination and components you want to install.

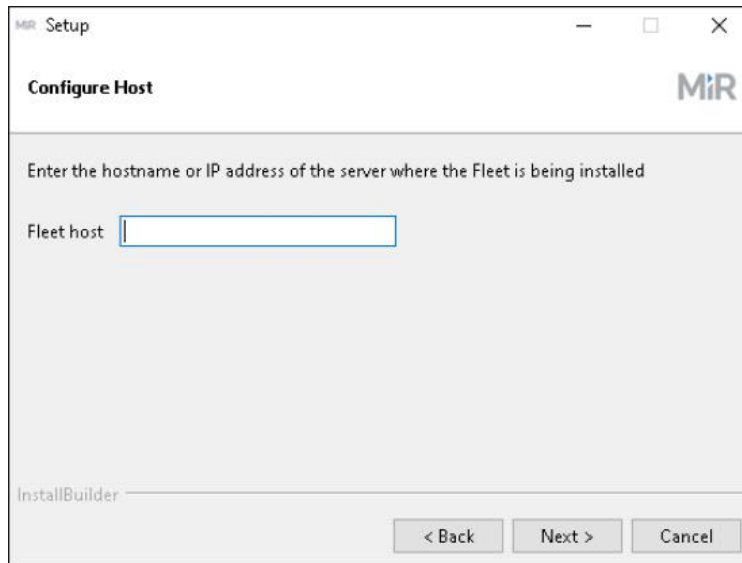


Depending on the components you have selected, the following setup steps will be shown or skipped.

- 6 Select the user configuration you decided to be the user for MiR Fleet Windows Service—see "[User for MiR Fleet Windows Service](#)" on page 220.



- 7 Enter the IP address or hostname of the server hosting MiR Fleet.

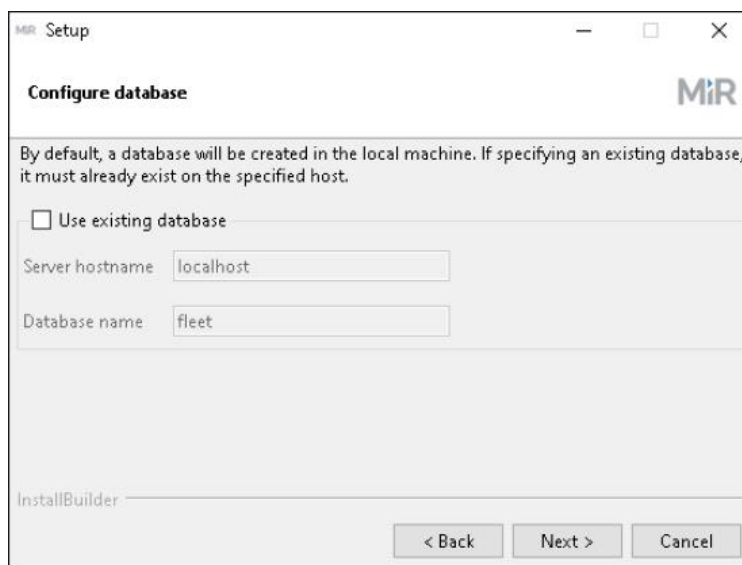


The screenshot shows a window titled "MiR Setup" with a "Configure Host" header and the MiR logo. The main area contains the instruction "Enter the hostname or IP address of the server where the Fleet is being installed" and a text input field labeled "Fleet host". At the bottom, there are three buttons: "< Back", "Next >", and "Cancel". The text "InstallBuilder" is visible in the bottom left corner of the window.



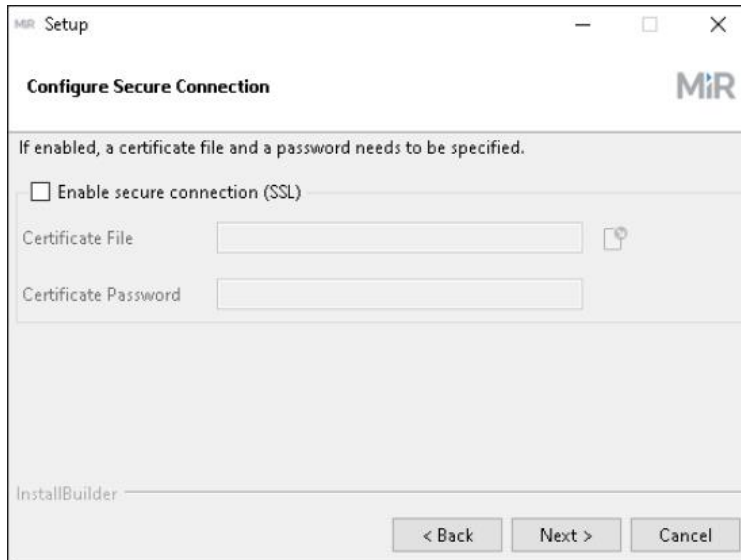
8 Choose the database configuration you want. You can also configure your database settings after the installation—see ["Configure MiR Fleet" on page 234](#).

- To create a new default database on the host server, do not modify any settings and select next. You must have Microsoft SQL installed on the host server.
- To use a database on another device, enable **Use existing database**. The chosen device must have Microsoft SQL Server installed and there must be an existing database instance that is owned by the same account that owns the MiR Fleet Windows service. To connect another server, enter the hostname of the database server and the name of the database instance.



The screenshot shows a window titled "MiR Setup" with a "Configure database" section. The window includes a MiR logo in the top right corner. Below the title bar, there is a note: "By default, a database will be created in the local machine. If specifying an existing database, it must already exist on the specified host." There is a checkbox labeled "Use existing database" which is currently unchecked. Below this checkbox are two text input fields: "Server hostname" with the value "localhost" and "Database name" with the value "fleet". At the bottom of the window, there are three buttons: "< Back", "Next >", and "Cancel". The text "InstallBuilder" is visible in the bottom left corner of the window.

- 9 Configure SSL. You can also configure SSL after the installation—see "[Configure MiR Fleet](#)" on page 234.



- 10 Start the installation and wait for it to complete.

### 4.4.3 Troubleshoot start-up issues

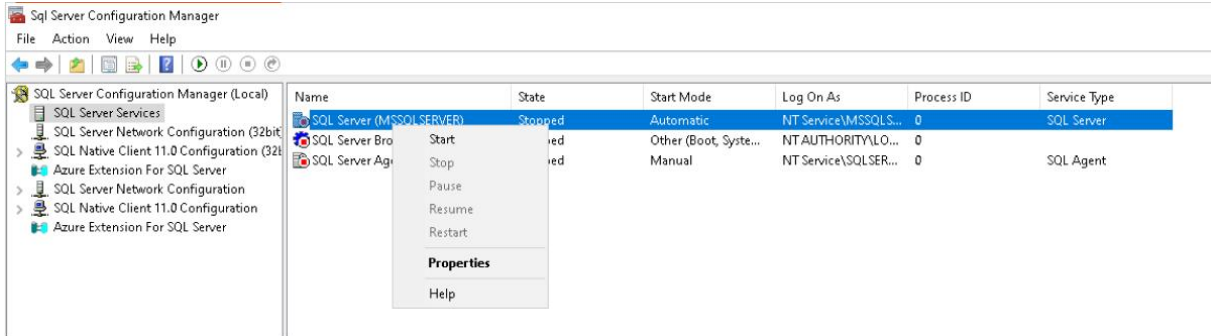
While installing MiR Fleet, use Event Viewer to see more information about any critical errors or warnings that may occur during the installation.

Issue	Solution
<p><b>MSSQL not reachable</b></p> <p>Exception Info: Microsoft.Data.SqlClient.SqlException (0x80131904): A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured to allow remote connections. (provider: Named Pipes Provider, error: 40 - Could not open a connection to SQL Server)</p>	<ul style="list-style-type: none"> <li>• Check that MSSQL is installed and running—see <a href="#">"Microsoft SQL is installed and running"</a> on page 229.</li> <li>• Check that MiR Fleet configuration file points to the correct server hostname—see <a href="#">"Microsoft SQL destination"</a> on page 229.</li> </ul>
<p><b>Unable to create database</b></p> <p>Description: The process was terminated due to an unhandled exception.</p> <p>Exception Info: Microsoft.Data.SqlClient.SqlException (0x80131904): CREATE DATABASE permission denied in database 'master'.</p> <p>or</p> <p>Login failed for user 'NT SERVICE\MiRFleet'. Reason: Could not find a login matching the name provided. [CLIENT: &lt;local machine&gt;]</p>	<p>Ensure the database and MiR Fleet application are owned by the same user and exists in MSSQL—see <a href="#">"Database and MiR Fleet are owned by same user"</a> on page 229.</p>

Issue	Solution
<p><b>MiR Fleet interface unreachable</b></p> <p>When you try to access the MiR Fleet interface, and the browser reports that the site cannot be accessed.</p> <pre>This site can't be reached ERR_CONNECTION_TIMED_OUT</pre>	<ul style="list-style-type: none"> <li>• Ensure your application and network meet firewall requirements and is set to the correct IP address—see <a href="#">"Network settings and changing the IP address" on page 232.</a></li> <li>• Ensure you are using the correct URL—see <a href="#">"Sign in" on page 254.</a></li> </ul>
<p><b>MiR Fleet and robots not communicating</b></p> <p>You cannot get robots connected to MiR Fleet.</p>	<p>Ensure your application and network meet firewall requirements and is set to the correct IP address—see <a href="#">"Network settings and changing the IP address" on page 232.</a></p>
<p><b>MiR Fleet configuration file corrupted</b></p> <pre>Exception Info: System.IO.InvalidDataException: Failed to load configuration from file 'C:\Program Files\MiRFleet\fleet\custom-settings.json'. ---&gt; System.FormatException: Could not parse the JSON file.</pre>	<p>Open the <code>custom-setting.json</code> file and run it through a JSON validation tool to check for syntax errors.</p>
<p><b>DotNet not installed</b></p> <pre>Path: C:\Program Files\MiRFleet\fleet\Mir.Fleet.exe Message: You must install .NET to run this application.</pre>	<p>Download .NET—see <a href="#">"Install MiR Fleet" on page 217.</a></p>

### Microsoft SQL is installed and running

Open the SQL Server Configuration Manager (SSMS can be installed after installing MSSQL—see "Microsoft SQL Server " on page 218), go to SQL Server Services and ensure SQL Server is running. If not, select it, right-click, and select **Start**.



### Microsoft SQL destination

Open the `appsettings.json` under `C:\Program Files\MiRFleet\fleet` (unless another installation folder was selected). The database configuration has the following format:

```

"Database": {
  "Type": "mssql",
  "ConnectionString": "Data Source=<hostname>;
    initial catalog=<db_name>;
    persist security info=True;
    Integrated Security=SSPI;
    TrustServerCertificate=True;"
}
    
```

Ensure `<hostname>` refers to the correct server hostname, and `<db_name>` refers to the database name (`fleet` by default). Restart the application if you make any changes.

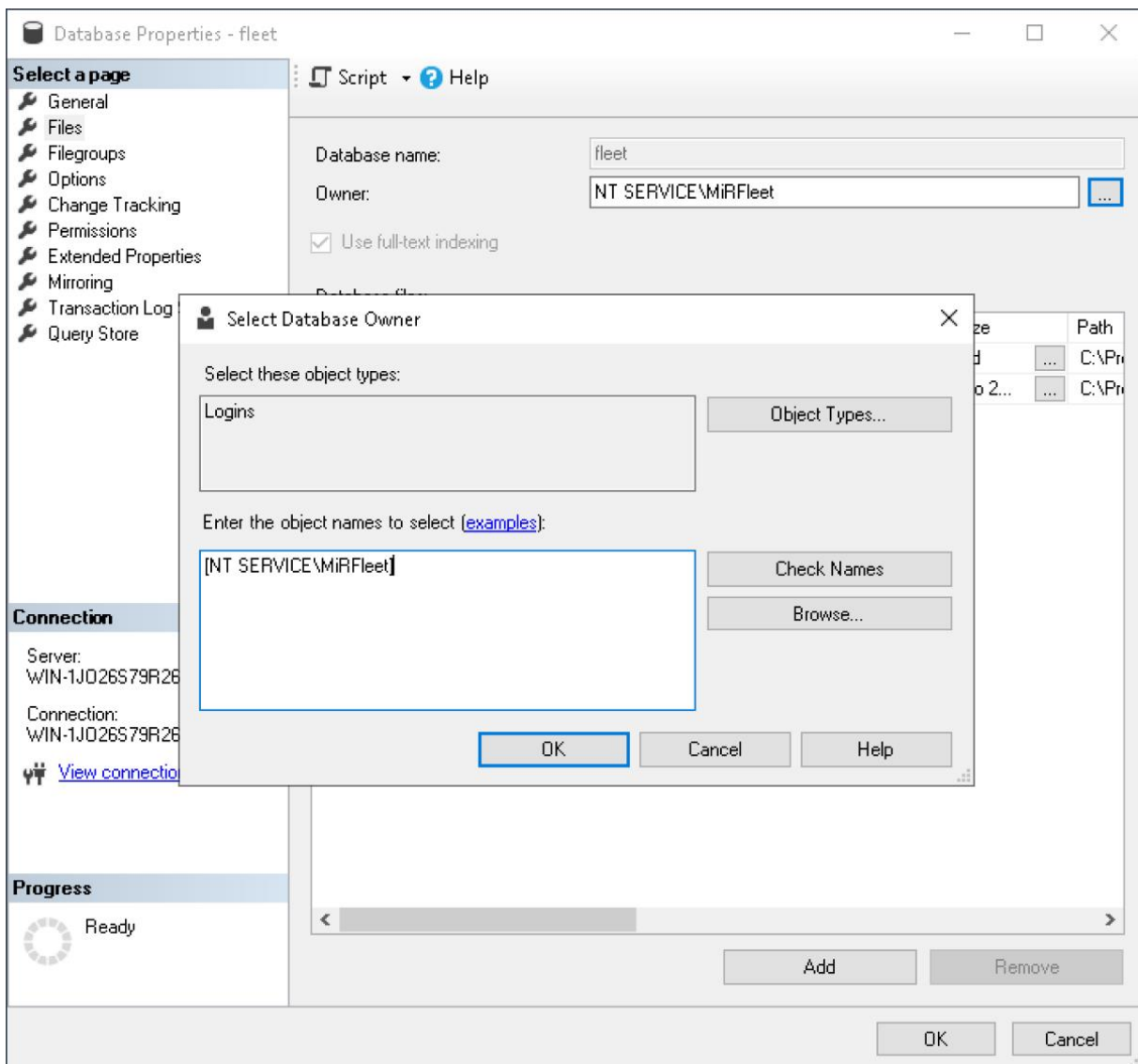
### Database and MiR Fleet are owned by same user

Ensure the fleet database is owned by the same user who owns the MiR Fleet service:

- 1 Open Microsoft SQL Server Management Studio (SSMS can be installed after installing MSSQL—see "Microsoft SQL Server " on page 218).

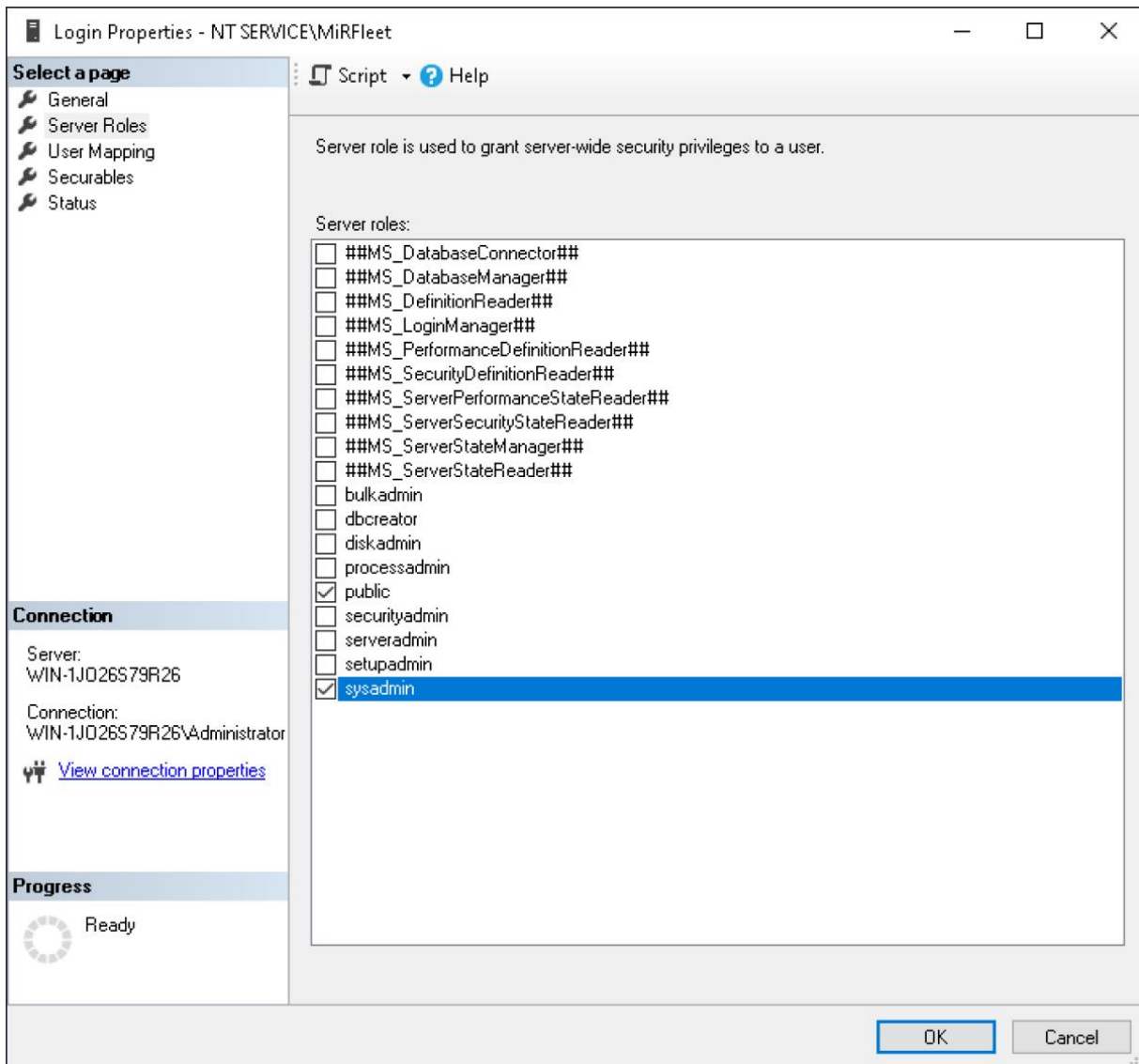
- 2 Connect to the database and open the fleet database.
- 3 Go to **Files**, and ensure **Owner** is set to the right user. If local service account is used, the user is NT SERVICE\MiRFleet.

If the user you want to use does not exist, go to **Security > Logins**, and add a **New login**. Search for the desired user and add it to the system. Go back to the fleet database, and set the user as the owner.



Once the same user owns the database and the MiR Fleet service, give sysadmin privilege to the user and restart the service:

- 1 In Microsoft SQL Server Management Studio, go to **Security > Logins**.
- 2 Double-click on the user. If a local service account is used, the user is NT SERVICE/MiRFleet.
- 3 Go to **Server Roles**, and ensure **sysadmin** is set. This is not needed by default, once the database has been created, the privilege can be revoked.



## Network settings and changing the IP address

Ensure the correct ports are allowed in the firewall—see ["Firewall" on page 234](#)

Ensure you have set up IP addresses in your network with either static IP addresses or DHCP reservations—see ["IP addresses" on page 214](#).

To check which IP address the MiR Fleet service uses, open the file `appsettings.json` under **C:\Program Files\MiRFleet\fleet** (unless another installation folder was selected). The MiR Fleet configuration has the following format:



```
"Mir.Fleet": {  
  "Host": "<hostname>"  
}
```

Where `<hostname>` is the IP address of MiR Fleet.

If you need to change the IP address the MiR Fleet software uses, follow these steps to ensure the MiR system updates to use the correct address to communicate with MiR Fleet:

- 1 Remove all robots from MiR Fleet.
- 2 Stop the MiR Fleet service.
- 3 Change the IP address in the `appsettings.json`.
- 4 Clean up the local certificates:
  - a Open Microsoft SQL Server Management Studio (SSMS can be installed after installing MSSQL—see "[Microsoft SQL Server](#)" on page 218
  - b Connect to the database.
  - c Open the fleet database.
  - d Find the `Store.Certificates.LocalCertificates` table.
  - e Right-click and select **Edit top 200 rows**.
  - f Remove the existing row.
- 5 Start the MiR Fleet service and add the robots to MiR Fleet again.

#### 4.4.4 Configure MiR Fleet

##### Section overview

Configure MiR Fleet according to site cybersecurity policies and preferences.

- Configure firewalls, TLS, and SSO.
- Connect an SQL database.

Many of the MiR Fleet configurations can be customized while installing MiR Fleet—see "[Install MiR Fleet](#)" on page 217—or after installing by changing `custom-settings.json` or `appsetting.json`. These files are located in the directory where MiR Fleet is installed (default: **C:/Program Files/MiR Fleet/fleet**).

##### Firewall

MiR Fleet uses ports 443, 5050, and 5060 by default. You can close off any other unused ports using the host device firewall.

Even if TLS is disabled, MiR Fleet still uses port 443, unless you can change the port by overriding it under `WebHostOptions` in the `appsettings.json` file.

If you have already set up the firewall based on the cybersecurity standards of your company, make sure to open ports 443 (or the user configured port) and 5060 for the MiR Fleet host server.

To do this on the MiR Fleet server, open **Windows Defender Firewall with Advanced Security**, and go to **Inbound Rules > MiR Fleet > Protocol and Ports**.

**Table 4.5** Ports used by the MiR Fleet server

Service	Target	Direction	Port
Data	SQL database server	Outbound	1433/tcp
Robot communication	Robots	Outbound and inbound	5060/tcp

Service	Target	Direction	Port
Web interface	User device	Inbound	443/tcp
MiR Fleet Integration API	External devices	Inbound	443/tcp
Data monitoring	MiR Insights Online	Outbound	443/tcp, 5671/tcp, 5672/tcp
Shop Floor connector—see <a href="#">"Shop Floor Connector" on page 596</a>	External devices	Inbound	Custom

**Table 4.6** Ports used by the MiR robots

Source	Target	Direction	Port
MiR Fleet communication	MiR Fleet	Outbound and inbound	5060/tcp
Web interface	User device	Inbound	443/tcp
Top module API	Top module	Inbound	HTTP: 8080/tcp, HTTPS: 443/tcp
Modbus	Top module	Inbound	502/tcp
Remote support	Technical support	Outbound	443/tcp

## Transport Layer Security (TLS)

To use TLS, ensure your MiR Fleet host device meets the following requirements:

- TLS 1.3 is enabled.
- TLS\_AES\_256\_GCM\_SHA384 and TLS\_AES\_128\_GCM\_SHA256 are enabled as encryption algorithms.
- There is a valid .pfx certificate saved on the device for the MiR Fleet application. The authority that signed the certificate must be in the Microsoft Certificate Store of the host device.
- There is a key file for the certificate on the device, or you have the password for the certificate.

The TLS encryption algorithms are machine settings that can be reset for external reasons. If the MiR Fleet interface cannot be accessed through SSL, check the TLS and encryption settings on the MiR Fleet host device.

If you did not enable TLS during the installation, you can set it up at any time in the `appsettings.json` configuration file. Reference the .pfx certificate and its corresponding password under `WebHostOptions` in the `appsettings.json` configuration file.

```
"WebHostOptions": {
  "WebUi": {
    "Enabled": "true",
    "Port": 8800,
    "Ssl": "false",
    "Certificate": {
      "File": "C:\\<absolute_path>\\<certificate_name>.pfx",
      "Password": "<passphrase>"
    }
  }
},
```

There are the following properties:

- `Port`: The port MiR Fleet uses for communication.
- `Ssl`: Set to True to enable TLS.
- `Certificate`: If you enable TLS, you must reference the certificate and password **or** `KeyFile`:
- `File`: The path to the .pfx certificate file.
- `Password`: Your certificate password.
- `KeyFile`: The path to the key file for the certificate.

## Single Sign-On (SSO)

MiR Fleet supports SSO via OpenID Connect (OIDC).

SSO can be set up under `Mir.Fleet.IntegrationApi.Options.RemoteIdOptions` in the `custom-settings.json` configuration file. To set up SSO, see ["Setting up Single Sign-on \(SSO\)" on the next page](#).

## SQL Database

You can configure your SQL database connection in the `appsettings.json` configuration file. To change which device MiR Fleet looks for a database on, enter the IP address of the host under `Database.ConnectionString.DataSource`. The value can be set to an IP address if you are using a database hosted on another device. By default, the value is `localhost` if the database is saved on the same device as MiR Fleet.

```
"Database": {
  "Type": "sqlite",
  "ConnectionString": "Data Source=sqlite.db"
},
```

## Include site in error logs

You can decide if your site data is included in error logs—see ["Sites" on page 102](#). The site data can help MiR Technical Support understand the context of an issue.

By default, all site data is included. To change this so no site data is included, open the `appsettings.json` configuration file and add the line `"SiteFileOptOut": "true"`:

```
{
  "Mir.Fleet": {
    "Host": "<insert fleet ip here>"
  },
  "SiteFileOptOut": "true"
}
```

To make MiR Fleet include the data in error logs, either remove the line or change it to `"SiteFileOptOut": "false"`.

### 4.4.5 Setting up Single Sign-on (SSO)

Single Sign-on (SSO) lets you connect an existing authentication system to MiR Fleet and lets your users use the same credentials across systems. It is optional if you want to use it.

To use SSO, your setup must meet the following requirements:

- OpenID Connect must be used together with an Identity Provider (IdP).
- The MiR Fleet application must be able to communicate with an IdP.
- The MiR Fleet application must be configured to run HTTPS and SSL—see ["Configure MiR Fleet" on page 234](#).
- The MiR Fleet application must be configured to use a remote IdP.
- Redirect and callback URIs must be configured on an IdP—see ["Redirect URI" on page 242](#).
- MiR Fleet roles and permissions must be mapped on an IdP—see ["Map roles" on page 243](#).

In the `custom-settings.json` file, you must include the following data to set up SSO:

```
"Mir.Fleet.IntegrationApi.Options.RemoteIdpOptions": {
  "OidcProviders": [
    {
      "Authority": "<OpenID-Configuration-Document-URI>",
      "Name": "<Mir-Specific-Identifier>",
      "ClientId": "<Unique-Client-Id>",
      "ClientSecret": "<Unique-Client-Secret>",
      "DisplayName": "<Some-Name [optional]>",
      "ClaimMap": {
        "NameClaim": "<name of user's full name claim [optional] (default is name)>",
        "IdClaim": "<name of user's id claim [optional] (default is oid)>",
        "RolesClaim": "<name of user's roles claim [optional] (default is roles)>",
      },
      "DisableOfflineAccessScope": <true/false [optional] (default is false)>,
      "CustomScopes": <list of strings (e.g. ["scope1", "scope2"]) [optional]>
      (default is empty)
    }
  ]
}
```

- **Authority:** The address of an OpenID configuration document.
- **Name:** Enter the name you want to use to identify the IdP within the MiR system. The name must be URL safe and comply with RFC 3986. To ensure this, only use alphanumeric values, hyphens, and underscores.

- `ClientId`: Your client ID.
- `ClientSecret`: Your client secret.
- `DisplayName`: (Optional) the name shown for the SSO sign-in button in the MiR Fleet interface.
- `ClaimMap`: (Optional) contains the following properties:
  - `NameClaim`: The name of the claim containing the user's full name.
  - `IdClaim`: The name of the claim containing the user's object ID.
  - `RolesClaim`: The name of the claim containing the list of roles.
- `DisableOfflineAccessScope`: Disable requesting the offline access scope. Is `False` by default.
- `CustomScopes`: Request additional scopes to the iDP. Is empty by default.

The following examples show you where to find these values and set up SSO with Microsoft Entra ID (formerly Azure AD) and Auth0. You can use another system, these are just examples to help get you started.

### SSO with Microsoft Entra ID

When setting up Microsoft Entra ID, you must use Microsoft Azure Portal to create the application and find all the configuration values you need and set up users.

Create a new application, and follow the instructions in the following sections to set up the application.

#### Set up `custom-settings.json`

You must enter your SSO configuration values in the `custom-settings.json`. This file is located in the directory where MiR Fleet is installed (default: **C:/Program Files/MiR Fleet/fleet**).

The required values are:

- **Authority:** Enter the URL copied from **OpenID Connect metadata document**. You can find the list of endpoints under **Overview > Endpoint**.

## Endpoints ×

Copy to clipboard

OAuth 2.0 authorization endpoint (v2)	<code>https://login.microsoftonline.com/311a22fd-9576-.../oauth2/v2.0/authorize</code>	
OAuth 2.0 token endpoint (v2)	<code>https://login.microsoftonline.com/311a22fd-9576-.../oauth2/v2.0/token</code>	
OAuth 2.0 authorization endpoint (v1)	<code>https://login.microsoftonline.com/311a22fd-9576-.../oauth2/authorize</code>	
OAuth 2.0 token endpoint (v1)	<code>https://login.microsoftonline.com/311a22fd-9576-.../oauth2/token</code>	
OpenID Connect metadata document	<code>https://login.microsoftonline.com/311a22fd-9576-.../v2.0/.well-known/openid-configuration</code>	
Microsoft Graph API endpoint	<code>https://graph.microsoft.com</code>	
Federation metadata document	<code>https://login.microsoftonline.com/311a22fd-9576-.../federationmetadata/2007-06/federationmetadata.xml</code>	
WS-Federation sign-on endpoint	<code>https://login.microsoftonline.com/311a22fd-9576-.../wsfed</code>	
SAML-P sign-on endpoint	<code>https://login.microsoftonline.com/311a22fd-9576-.../saml2</code>	
SAML-P sign-out endpoint	<code>https://login.microsoftonline.com/311a22fd-9576-.../saml2</code>	

- **Name:** Enter the name you want to use to identify the IdP within the MiR system. The name must be URL safe and comply with RFC 3986. To ensure this, only use alphanumeric values, hyphens, and underscores.



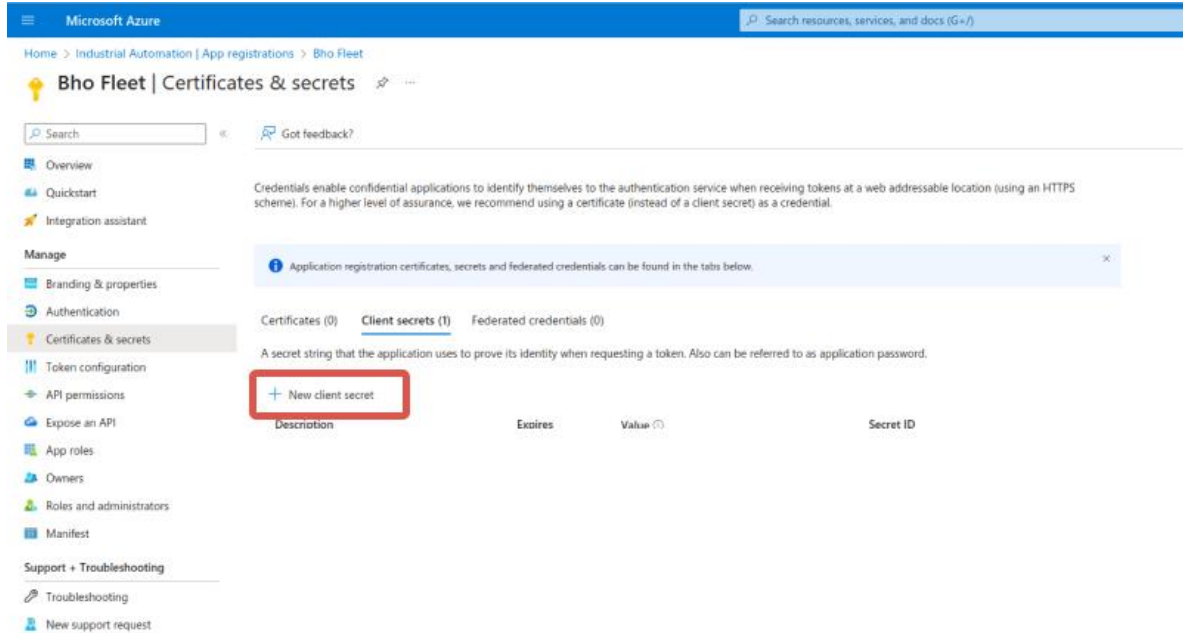
- ClientId: Enter the Client or Application ID found on the **Overview** page.

The screenshot shows the Microsoft Azure portal interface for an application named "Bho Fleet". The page is titled "Microsoft Azure" and "Bho Fleet". The left sidebar contains navigation options: Overview (selected), Quickstart, Integration assistant, and a "Manage" section with options like Branding & properties, Authentication, Certificates & secrets, Token configuration, API permissions, and Expose an API. The main content area shows the "Essentials" section with the following details:

- Display name : [Bho Fleet](#)
- Application (client) ID : 26b1be64-3606-4e30-b704-f266fbf8927b
- Object ID : 3417111c-674f-4894-...
- Directory (tenant) ID : 311a22fd-9576-40ab-...
- Supported account types : [My organization only](#)

A blue banner at the bottom of the Essentials section contains the message: "Starting June 30th, 2020 we will no longer add any new features to Azure Active Directory Au". At the bottom of the page, there are links for "Get Started" and "Documentation".

- **ClientSecret:** To create a client secret, go to **Certificates & secrets** and create a new client secret. Enter the secret in the `custom-settings.json` file and consider also storing it in another safe location. You can only view the client secret once just after generating it in Microsoft Azure Portal.



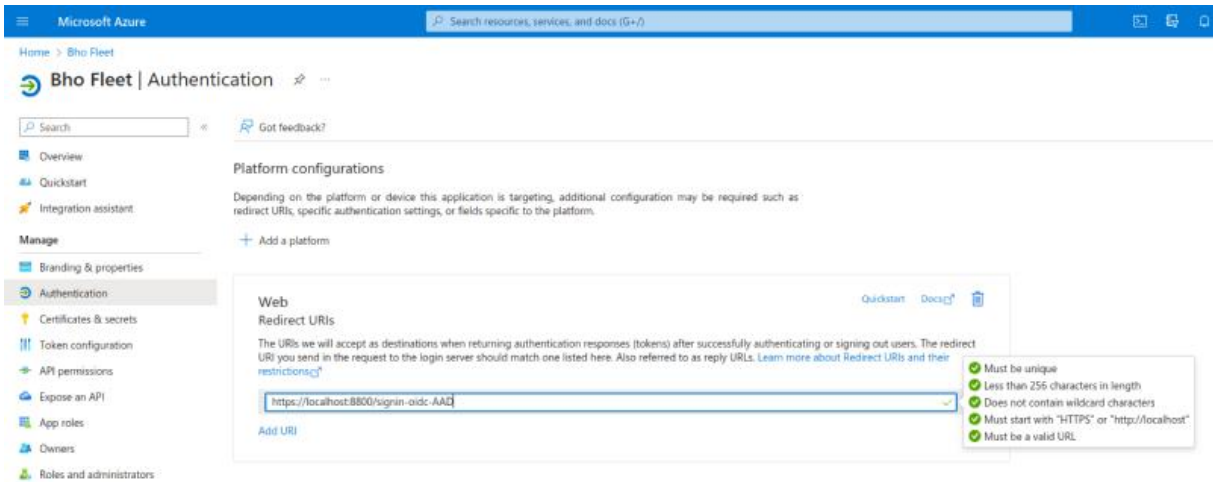
## Redirect URI

The redirect URI or reply URL is the location where the authorization redirects a user once they have been authenticated.

You must set the redirect URI to the IP address and port of MiR Fleet—see ["Sign in"](#) on [page 254](#)—, followed by `/signin-oidc-<oidcProviderName>`, where `<oidcProviderName>` is the name you assigned the Object Identifier (OID)—see ["Set up custom-settings.json"](#) on [page 239](#).

For example: `https://192.68.7.7:443/signin-oidc-fleetapp`

Set the redirect URI under **Authentication > Platform configuration > Web**.

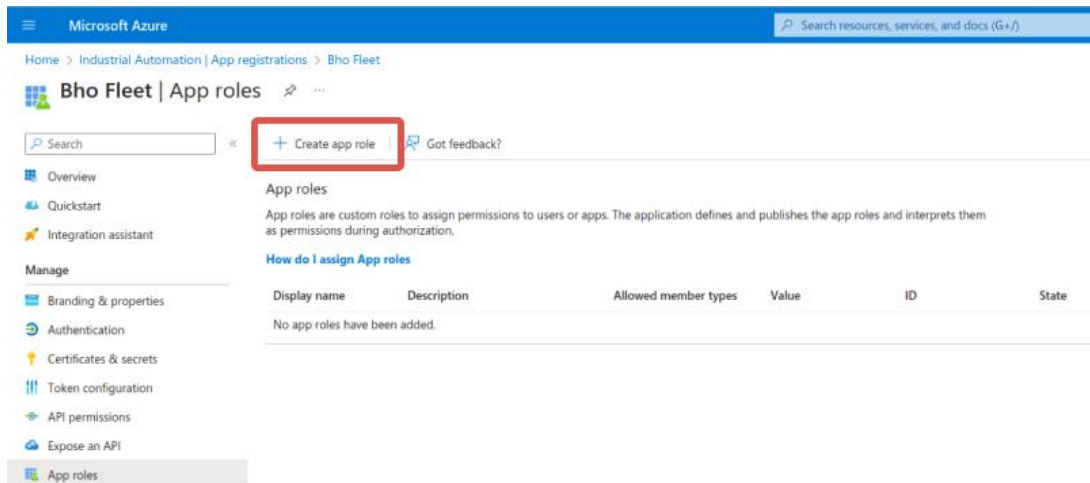


### Map roles

When setting up Microsoft Entra ID, you must create roles in MiR Fleet that match the same roles in your Microsoft Entra ID application. All users set up in Microsoft Entra ID can then sign in to MiR Fleet, and are assigned the corresponding role.

To set up a new application without any existing users or roles, follow these steps:

- 1 Under **App roles**, select **Create app role**.



**2** Enter the role details:

- **Display name:** Enter the name you want to use to identify the role in Microsoft Azure Portal.
- **Allowed member types:** Select whether the role can be assigned to users and groups or applications.
- **Value:** Enter the role name in MiR Fleet that you want this role to correspond to.
- **Description:** Enter a description that is displayed in app assignment and consent experiences.
- **Enable:** Select to activate the role.

### Create app role ×

Display name \* ⓘ

 ✓

Allowed member types \* ⓘ

Users/Groups

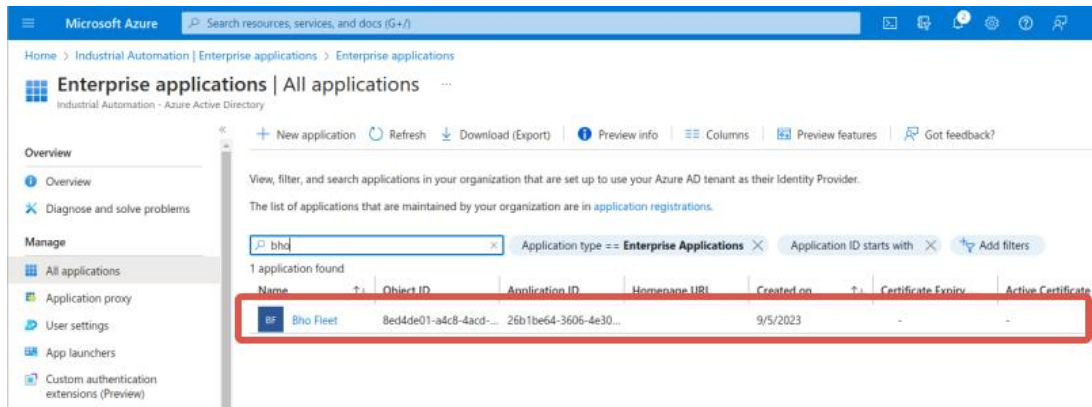
Applications

Both (Users/Groups + Applications)

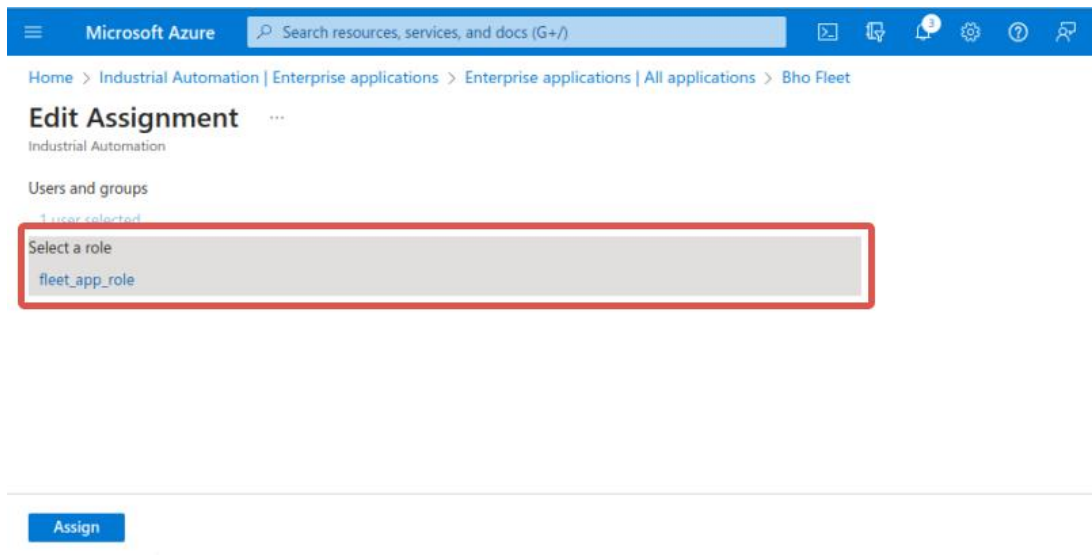
Value \* ⓘ

 ✓

- 3 Go to **All applications** and open your application.



- 4 Assign the role to a user.



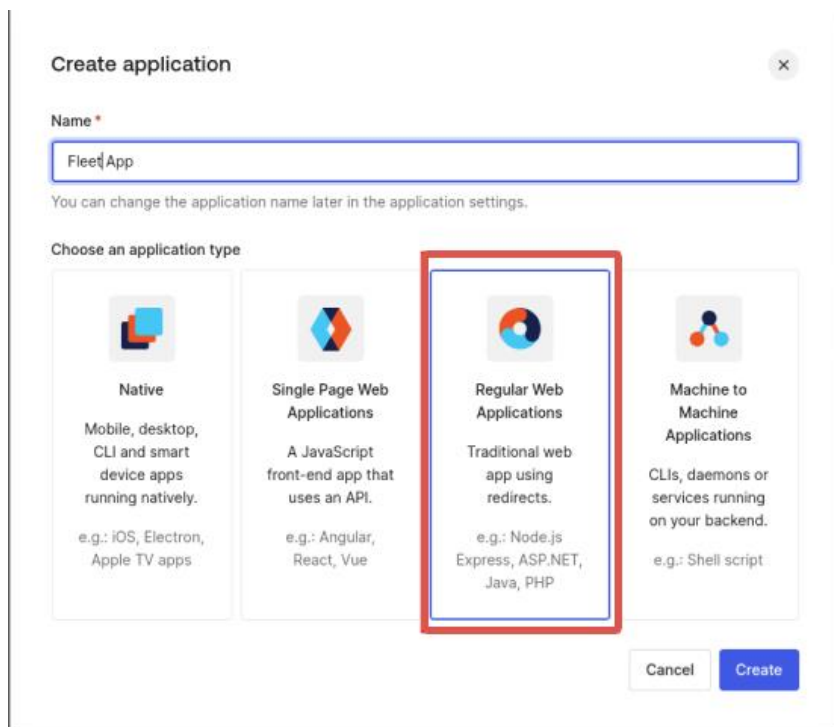
### Sign in to MiR Fleet

When you have updated the `custom-setting.json`, next time you open the MiR Fleet interface, you can choose to sign in using Microsoft Entra ID and use the credentials for the user you assigned the to the role you mapped to MiR Fleet—see ["Map roles" on page 243](#).

## Auth0

When setting up Auth0, use the Auth0 portal to create the application and find all the configuration values you need and set up users.

Create a new application using the Regular Web Application template, and follow the instructions in the following sections to set up the application.



### Set up custom-settings.json

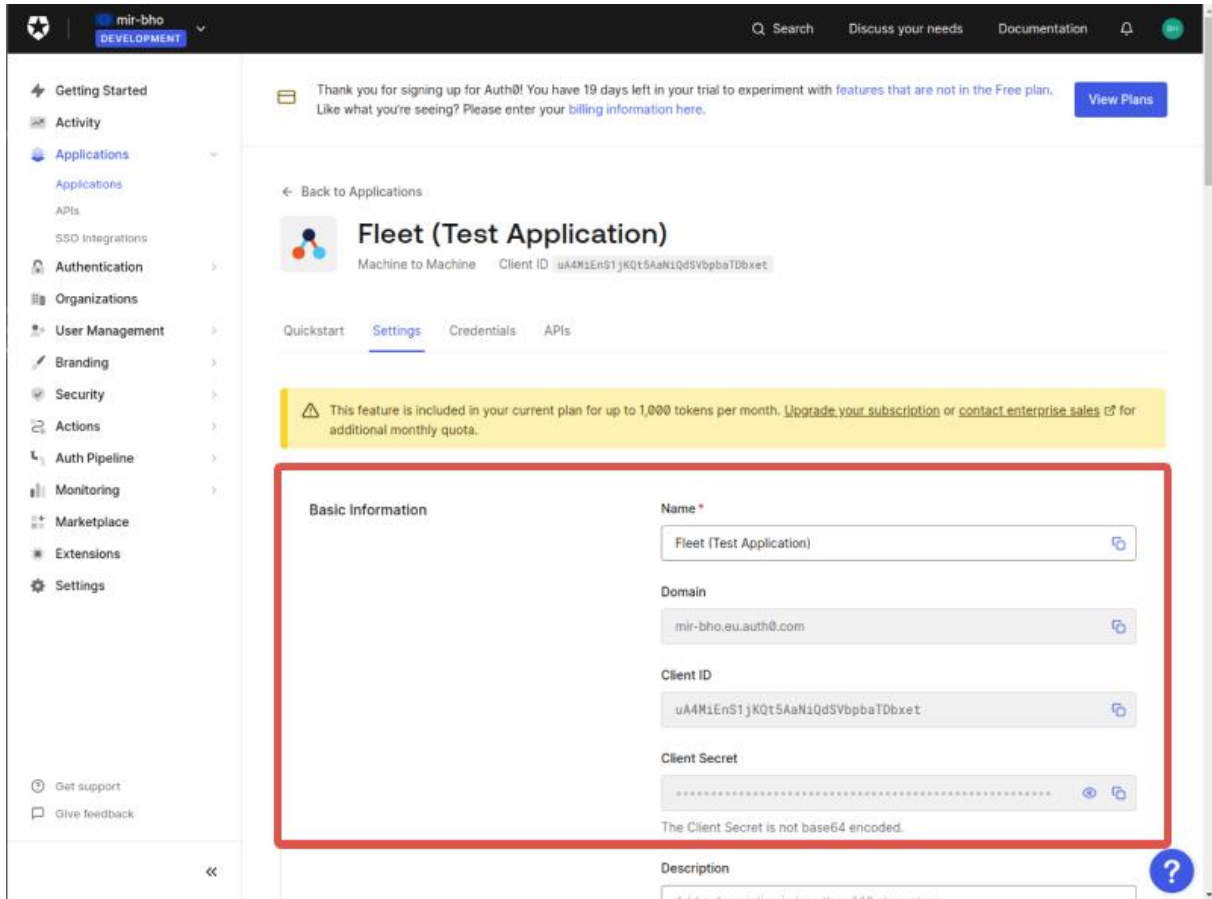
You must enter your SSO configuration values in the `custom-settings.json`. This file is located in the directory where MiR Fleet is installed (default: **C:/Program Files/MiR Fleet/fleet**).

When you create a new application, under **Basic information** you can find the configuration information you need:

- **Authority:** Enter the URL under **Basic information > Domain**.
- **Name:** Enter the name you want to use to identify the IdP within the MiR system. The name must be URL safe and comply with RFC 3986. To ensure this, only use alphanumeric values,

hyphens, and underscores.

- ClientId: Enter the ID under **Basic information > Client ID**.
- ClientSecret: Enter the secret under **Basic information > Client Secret**.



### Redirect URI

The redirect URI or reply URL is the location where the authorization redirects a user once they have been authenticated.

You must set the redirect URI to the IP address and port of MiR Fleet—see ["Sign in" on page 254](#)—, followed by `/signin-oidc-<oidcProviderName>`, where `<oidcProviderName>` is the name you assigned the OID—see ["Set up custom-settings.json" on page 239](#).

For example: `https://192.68.7.7:443/signin-oidc-fleetapp`

Set the redirect URI under **Application URIs**.

**Application URIs****Application Login URI**

In some scenarios, Auth0 will need to redirect to your application's login page. This URI needs to point to a route in your application that should redirect to your tenant's `/authorize` endpoint. [Learn more](#)

**Allowed Callback URLs**

After the user authenticates we will only call back to any of these URLs. You can specify multiple valid URLs by comma-separating them (typically to handle different environments like QA or testing). Make sure to specify the protocol ( `https://` ) otherwise the callback may fail in some cases. With the exception of custom URI schemes for native clients, all callbacks should use protocol `https://`. You can use [Organization URL](#) parameters in these URLs.

**Grant types**

You must set grant types to the following configuration:

The screenshot shows the 'Advanced Settings' page with the 'Grant Types' tab selected. Under the 'Grants' section, the following options are visible:

- Implicit
- Authorization Code
- Refresh Token
- Client Credentials
- Password
- MFA

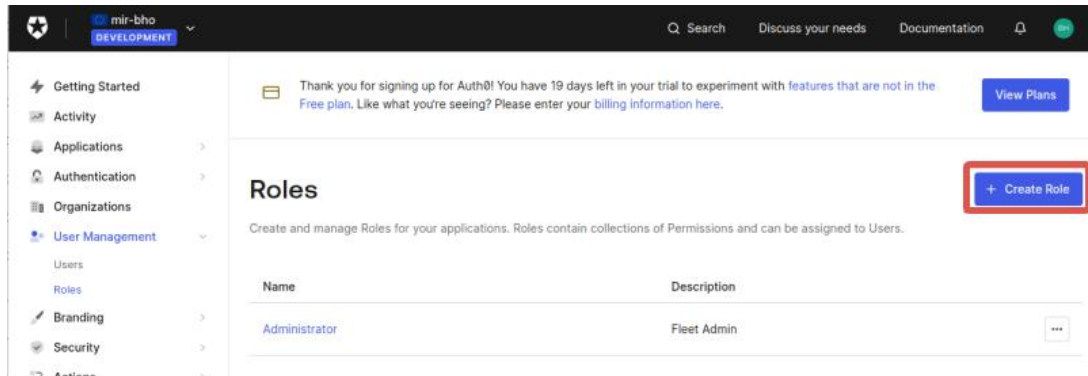
**Set up login flow and map roles**

When setting up Auth0, you must create roles in MiR Fleet that match the same roles in your Auth0 application. All users set up in Auth0 can then sign in to MiR Fleet, and are assigned the corresponding role.

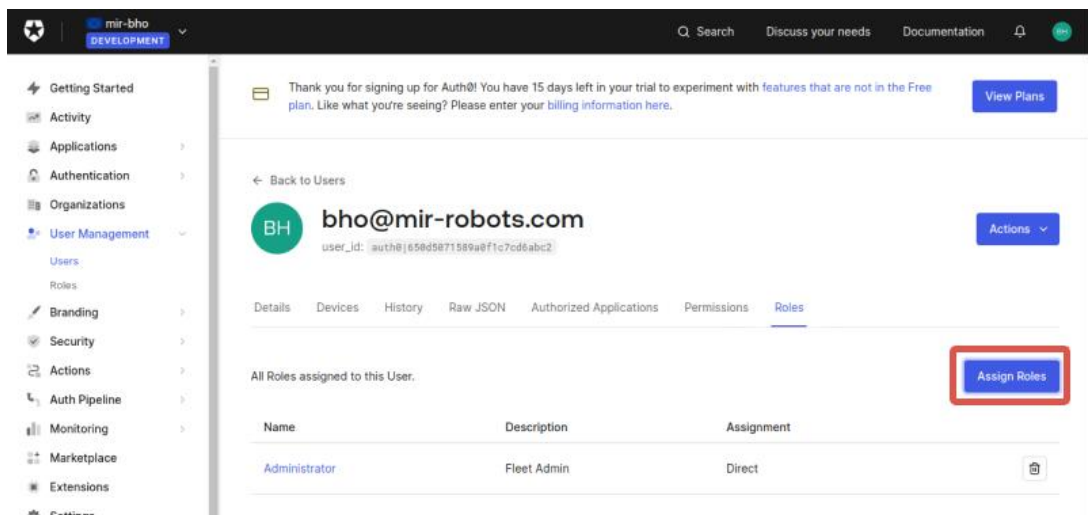
To create a role and set up Auth0 to work with MiR Fleet, follow these steps:

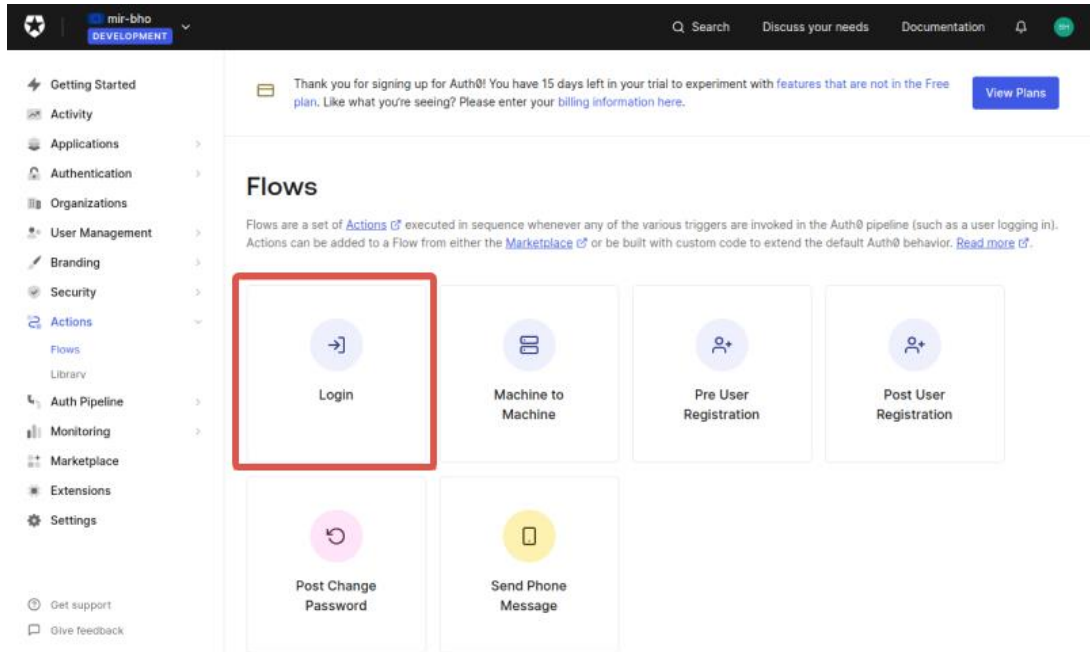


- 1 Under **User management > Roles**, create a new role. The role name must match a role in MiR Fleet.



- 2 Go to **User management > Users**, select a user, and assign the role to the user.



**3** Go to **Actions > Flows > Login**.

4 Create a custom login action with the following details:

- **Name:** fleet/roles
- **Trigger:** Login / Post Login
- **Runtime:** Node 18

### Create Action ×

**Name \***

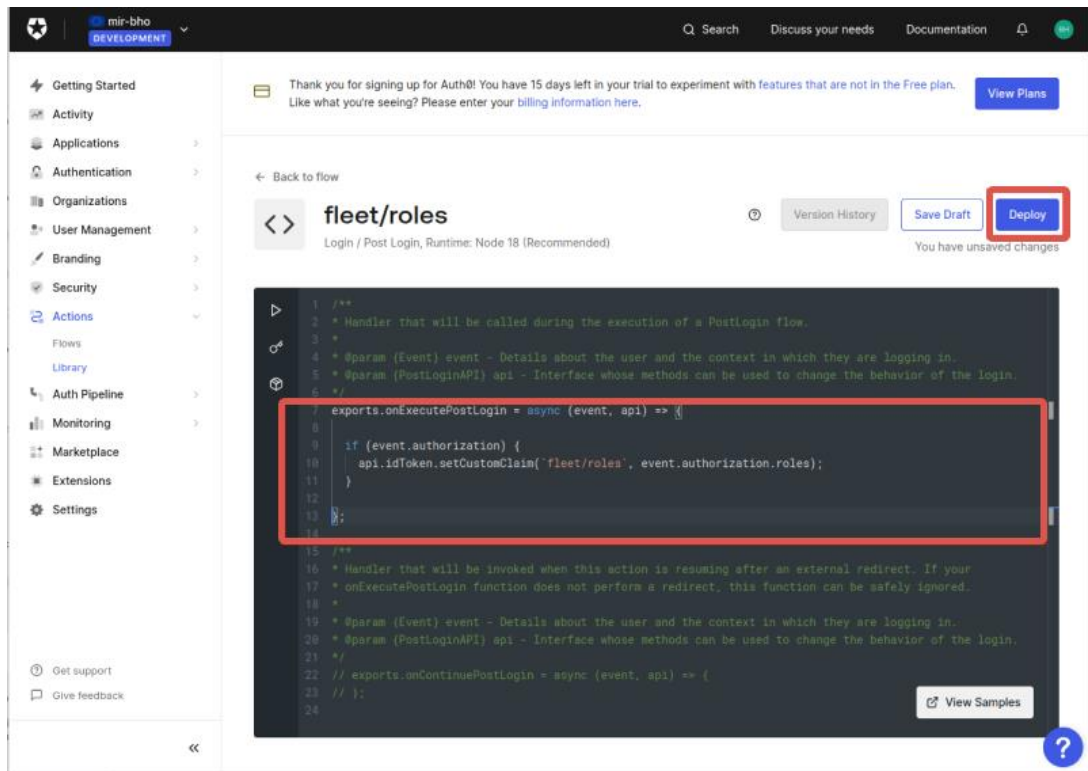
**Trigger \***

**Runtime \***

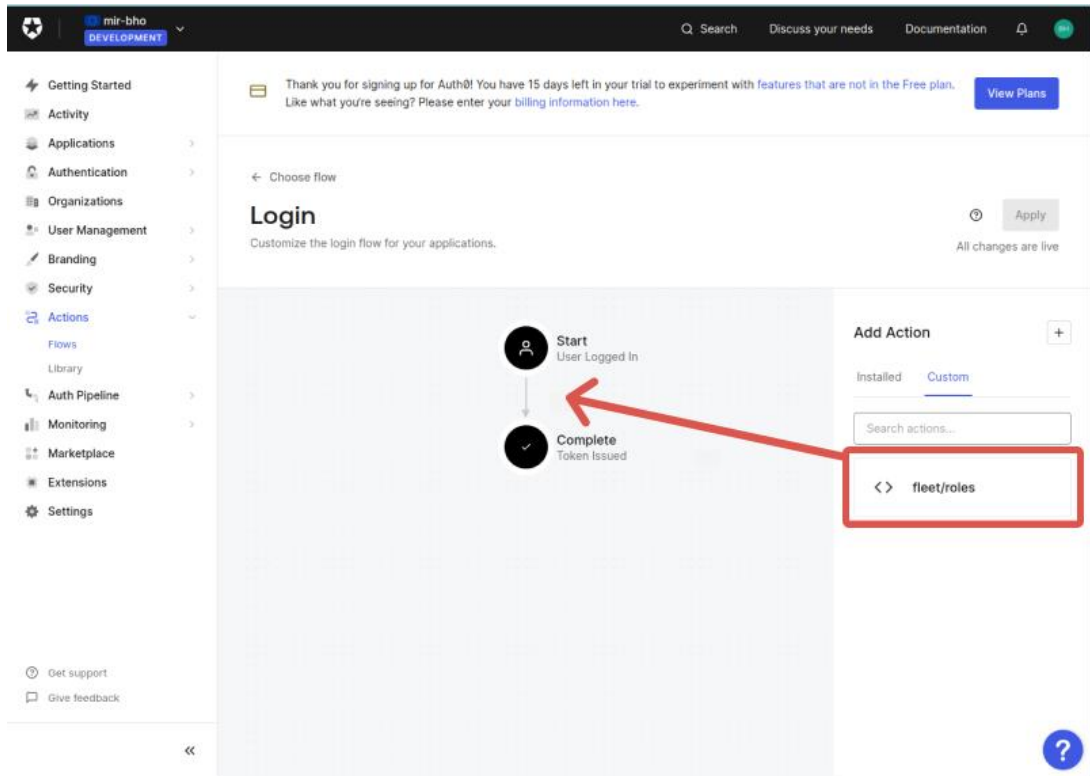
- 5 Create an action script to be executed after login. Use the following template.

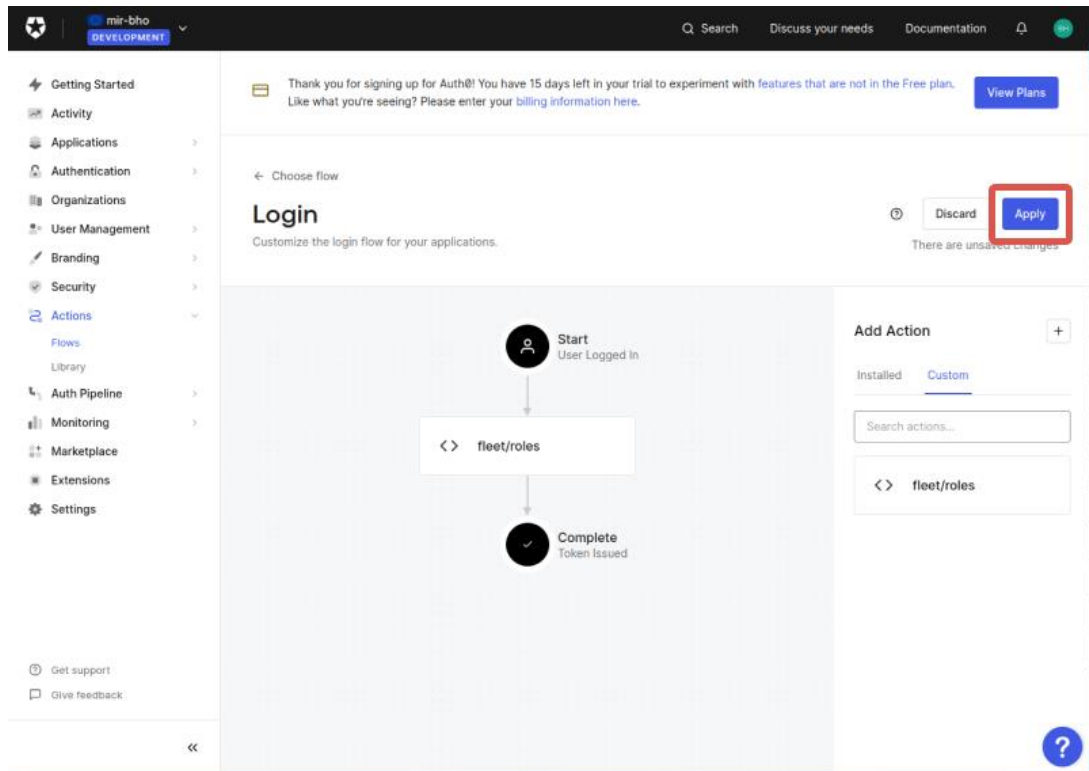
```
exports.onExecutePostLogin = async (event, api) => {  
  if(event.authorization){  
    api.idToken.setCustomClaim(`fleet/roles`, event.authorization.roles)  
  }  
};
```

Select **Deploy** when you have added the script.



6 Drag the action script into the login flow.



7 Select **Apply**.**Sign in to MiR Fleet**

When you have update the `custom-setting.json`, next time you open the MiR Fleet interface, you can choose to sign in with Auth0 and use the credentials for the user you assigned the to the role you mapped to MiR Fleet—see ["Set up login flow and map roles"](#) on page 248.

**4.4.6 Sign in****Section overview**

Sign in the first time.

- Sign in to the MiR Fleet interface.
- Create an admin user.

To access the MiR Fleet interface, enter one of the following addresses into a browser depending on the device you are using and the port settings. Port 443 is the default port, but if you choose not to use TLS and change the port to 80, you can access the MiR Fleet interface without defining the port in the address.

User device	Port	Address
The device hosting MiR Fleet	80	https://
The device hosting MiR Fleet	443	http://:443
A device on the same network as the device hosting MiR Fleet	80	https://fleet_device_ip_address
A device on the same network as the device hosting MiR Fleet	443	http://fleet_device_ip_address:443

MiR Fleet cannot be accessed by a device that is not on the same network.

The first time you sign in to the MiR Fleet interface, use the following user:

- Username: admin
- Password: mir

After the first access, you must change the password for the **admin** user.

The session lasts 12 hours. If you do not create a new password for the **admin** user, you can sign in using the credential above until you assign a new password.

Passwords must contain at least:

- Eight characters
- One special character
- One lower-case letter
- One upper-case letter
- One number

## 4.5 Set up users

**After reading this chapter, you can:**

Limit each user to the access level they need to complete their tasks and collect relevant tools on a single dashboard for ease of use.

**Create roles and users with the necessary permissions to complete user tasks.**

See ["Create users and roles" on the next page](#).

- List all the users for the MiR system and identify which permissions they need to complete their tasks.
- Identify permissions that enable users to make critical changes.
- Create roles for each permission group or user type.
- Create a user in MiR Fleet for each user and assign the user to relevant roles.
- Test that all users have access to the features and tools they need.
- Test that users do not have access to anything they are not trained to handle.
- Document the overview of users, permissions, and tasks.

**Make dashboards to collect relevant functions on a single page in the interface.**

See ["Create dashboards" on page 264](#).

- Determine what functions your users need to complete their tasks.
- Create dashboards that make these tasks easier to complete.
- Test that your dashboards work as expected.
- Document which tasks or user each dashboard is intended for.



## 4.5.1 Create users and roles

### Section overview

Create roles and users with the necessary permissions to complete user tasks.

- List all the users for the MiR system and identify which permissions they need to complete their tasks.
- Identify permissions that enable users to make critical changes.
- Create roles for each permission group or user type.
- Create a user in MiR Fleet for each user and assign the user to relevant roles.
- Test that all users have access to the features and tools they need.
- Test that users do not have access to anything they are not trained to handle.
- Document the overview of users, permissions, and tasks.

Always set up your fundamental users before creating anything else on the site. The user hierarchy determines which level each user is able to apply changes to the site later on—see ["Roles and users" on page 95](#).

All users of the robot must have a user profile in the system. Elements created by users can always be edited by the parent-user that created the child-user.

### Plan

Before creating any users or roles, determine critical permissions that only specific users need access to and map out which minimum permissions users need to complete their tasks.

### Identify critical permissions

Before creating any users or roles, identify which permissions—see ["Roles and users" on page 95](#)—can enable users to make critical changes such as:

- Modifying the system settings.
- Deleting important data that would be difficult to reconstruct.
- Creating misleading site elements that have not been validated.

If you assign a user to a role with permissions to make critical modifications, you must ensure:

- The user is trained to use that part of the interface correctly.
- The user understands the purpose of the setup.
- The user understands the consequences of making changes and the necessity of testing and checking any modifications.
- The user knows how to make a backup and always does this before making any changes.

You can choose to manage users and passwords by setting up MiR Fleet to use Single Sign-on (SSO)—see "[Setting up Single Sign-on \(SSO\)](#)" on page 238.

To ensure that each user you set up in your MiR system has access to the necessary functions, determine which users you want and what each users can do in the MiR system.

### Mapping user permissions

You can take a "task-based" approach and consider the following questions:

- Which tasks are MiR robots used for?
- What maintenance, update, and setup tasks related to MiR robots are there?
- Which robot functions are needed to complete each task?
- Can all of the necessary functions be collected on a single dashboard with widgets?
- Who is responsible for each task?

You can take a "user-based" approach and consider the following questions:

- Which users do you have on the site.
- What tasks does each user have with the MiR robot?
- What features do users need to complete their tasks?
- Can all of the necessary functions be collected on a single dashboard with widgets?

Map the results into a table similar to the following.

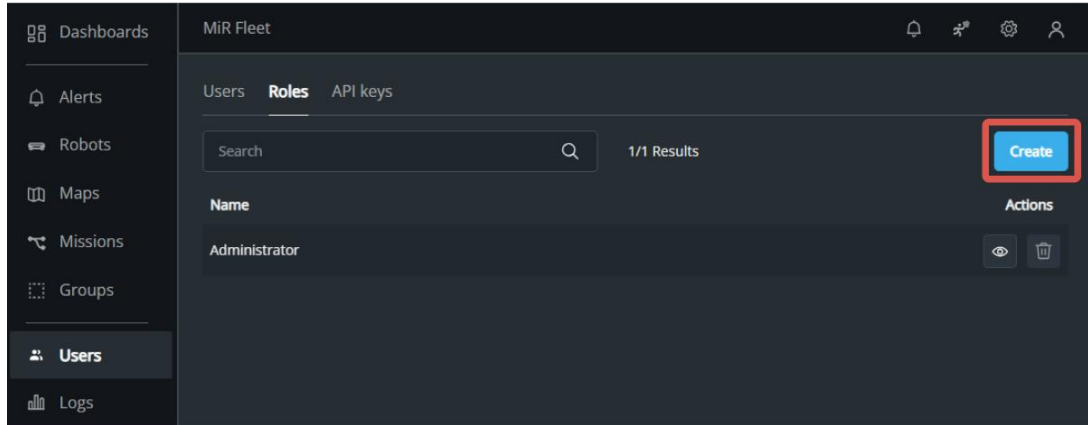
**Table 4.7** Example of how to map users and permissions

User	Tasks	Needs	Permissions
Floor operator 1	Starting missions x, y, and z	Start these three missions	Schedule missions
Floor operator 2	Starting mission y	Start a single mission	Schedule missions
Commissioner	Setting up the MiR system	Modify everything	All
Floor operations manager	Monitor the status of missions and robots	An overview of the MiR system status	View scheduled missions, view fleet error logs, view site
Service engineer	Maintain and test the robot	Drive the robot manually, run maintenance missions, and access to all diagnostics and system logging.	Create fleet error logs, create robot error logs, view site, schedule missions, edit alerts
Network manager	Testing and maintaining the I/O communications	Manage all I/O related settings and mission actions	Edit I/O modules, edit missions, schedule missions

## Create roles and edit permissions

To create a role and edit permissions:

- 1 Go to **Users > Roles**, and select **Create**.

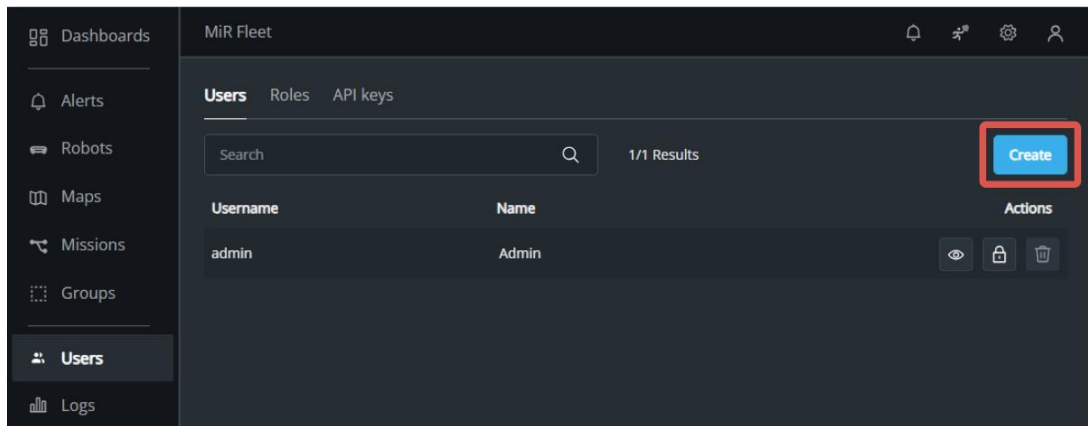


- 2 Enter a name and description for the role.
- 3 Select which permissions users assigned to this role should have.

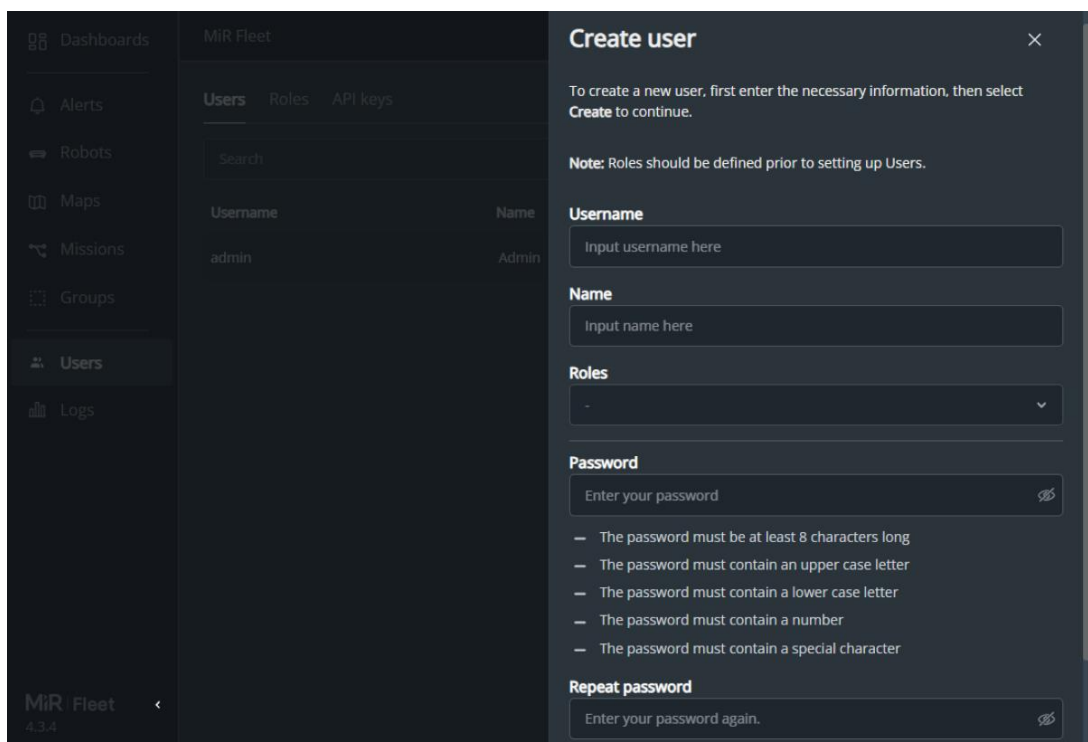
## Create new users

To create a new user, follow these steps:

- 1 Go to **Users**, and select **Create**.



- 2 Fill in the user credentials. The password you enter is only used the first time the user signs in to the interface. The user is immediately prompted to enter a personal password the first time they sign in.



- 3 Save the user.
- 4 Communicate the username and temporary password to the user using a secure method. Encourage the user to sign in as soon as possible to change the password to something private.

## Test

For each user:

- 1 Check that all safety-related settings cannot be changed.
- 2 Check that they can activate all relevant tasks on the robot.
- 3 Check that they can edit the elements they are allowed to edit.

## Maintain

Regularly check the users, permissions, and user activity in the system to ensure unauthorized users cannot access the system.

### Checking behavior

You can check the logs—see ["Logging" on page 40](#)—to see what users have been doing in MiR Fleet. Unexpected behavior can be a sign that:

- The user has inadequate training to understand the consequences of their changes.
- The site is not robust and issues are being troubleshooted as they arise instead of analyzing the root cause of an issue.
- The user access credentials have been compromised and an unauthorized user is tampering with your site.

### Review users and permissions

If the users responsible for certain tasks change over time, you have new employees that have replaced old ones, or you have troubleshooted permission issues as they arise, review the current state of users and roles in the system:

- Check for users who no longer need to access the robot and remove them from the system. This makes it easier to gain an overview of your users and prevents attackers from using old credentials to access the system.
- Check that the role descriptions still match the assigned permissions. Avoid allowing users more access than they need. Revisit the roles you have defined if you need more control of user permissions.

### Compromised password

If you suspect a user password has been compromised, change the password immediately or delete the user. To change the password, you need the current password of the user.

If you do not have the current password and the user is not able to change the password himself, delete the user and recreate it in MiR Fleet.

Once the password is changed or the user is deleted, any existing instances where the user is signed in are automatically signed out of the system.

To change the password for an existing user or delete a user, follow these steps:

- 1 Sign in to the MiR Fleet interface with a user that has permission to manage users.
- 2 Go to **Users**, and select **Edit** for the compromised user.
- 3 If you are changing the password, enter the existing password of the user, followed by a new password.  
  
If you are removing a user, note the user data and roles or take a screenshot of the user setup, then select **Delete**. Recreate the user with the same data.
- 4 The user will be prompted to change his password again when he signs in with the new password.

## Document

Keep and update the user needs map and permissions map.

If a user no longer needs access to the MiR Fleet, remove that user from the user and permissions map. This makes it easier to check if there are unnecessary users in the system.

## 4.5.2 Create dashboards

### Section overview

Make dashboards to collect relevant functions on a single page in the interface.

- Determine what functions your users need to complete their tasks.
- Create dashboards that make these tasks easier to complete.
- Test that your dashboards work as expected.
- Document which tasks or user each dashboard is intended for.

Use dashboards to collect the features a users needs on a single page or the tools you need for a specific process.

## Plan

When creating new dashboards, consider the following:

- Who will be using the dashboards?
- Which features will they need to use?  
For example, if your robot uses many I/O modules, you may want to monitor them from the dashboard, or if there is a mission that the robot often has to execute on demand, you may want to add it to the dashboard.
- Will you base your dashboards on what is needed for a process or what is need for a user?  
For example, you could choose to make a dashboard for each user or role that fits the user needs. Or you could make dashboards for tasks or processes, and the users must choose the dashboard relevant for what they are doing now.
- Do you want the commissioner to control which dashboards are created, or should each user decide how they want to customize their own dashboards?



## Create dashboards

To create a dashboard, follow these steps:

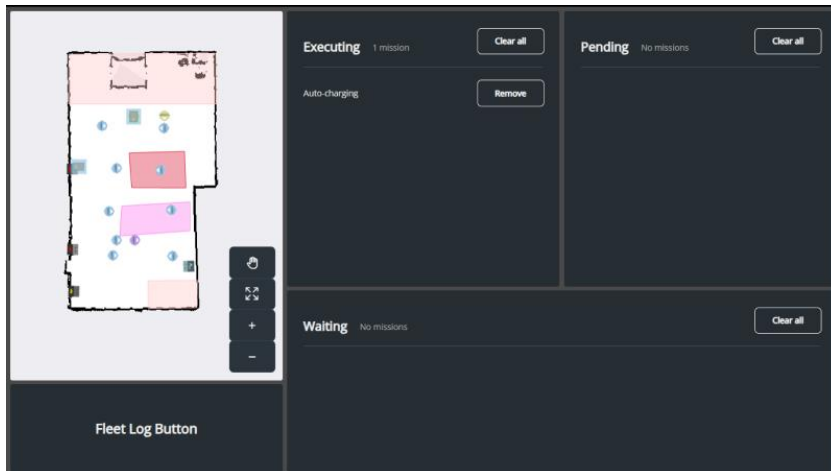
- 1 In the MiR Fleet interface, go to **Dashboards**.
- 2 Select **Create**, and enter a name for the dashboard.
- 3 Use the following tools to create your dashboard:
  - **Size:** Enter the number of rows and columns you want to base the dashboard on.
  - **Ratio:** Under the dashboard settings, enter the aspect ratio of the dashboard.
  - **Widgets:** Add any number of widgets to the dashboard. You can move and resize the widgets across the dashboard.
  - **Widget settings:** Select a widget to open the widget settings. The **Settings** tab contains everything specific to this widget. The **Robot** tab lets you choose if this widget should only apply to a specific robot, and the **Visual** tab lets you customize the appearance of the widget.
  - **Color:** Under the dashboard settings, change the background color of the dashboard.
  - The widget editor supports several standard keyboard shortcuts—see "[Dashboards](#)" on page 21.

## Recommended combinations

The following sections describe some common dashboard combinations you can use.

### Site status and mission overview

If you want a quick overview of ongoing and planned missions, create a dashboard with the different mission overviews and maps of all the relevant areas.

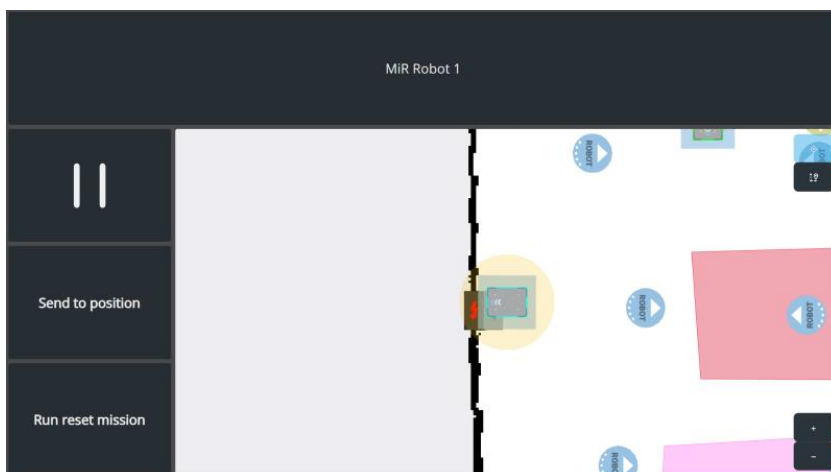


### Monitoring specific robots

You can create generic dashboards that you can use across all robots to see where they are and assign generic missions. Use the Robot selector widget to change which robot the dashboard applies to.

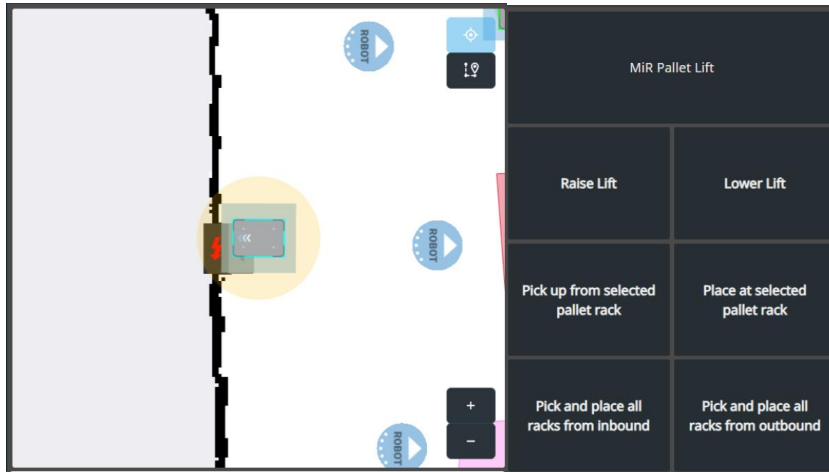
Useful generic missions can be:

- Missions with Relative move actions that you can use to move the robot away from obstacles it has gotten too close to.
- Missions that drive the robot to maintenance locations.
- Missions that reset or control the robot's top module.



### Task-oriented dashboards

You can collect all relevant missions for a specific purpose on a single dashboard. Use the Robot selector widget if you have several robots you want to be able control using the same dashboard.



### Linking between dashboards

Use the Link widget to organize your dashboards on an overview dashboard.

### Test

Once you have completed a dashboard, test that you can complete all the intended tasks using the dashboard.

### Document

- Keep an overview of which dashboard are intended for which users or tasks.
- Document which missions are accessible through dashboards.

## 4.6 Connect robots

### After reading this chapter, you can:

Ensure MiR Fleet controls robots correctly and reliably with missions.

**For each robot, check the robot's condition and set up the robots to prevent unauthorized access.** See ["Prepare robots" on the next page](#).

- Go through the received product inspection checklist.
- Upgrade the robot software.
- Protect against booting up from USB.
- Set a password for the robot BIOS.
- Disable all unused features.
- Disable or password protect robot access points.
- Update robot access points.

**Connect robots to MiR Fleet.** See ["Add robots" on page 276](#).

- Connect robots to the same Wi-Fi network as MiR Fleet.
- Set up mutual trust between MiR Fleet and the robots.
- Add one robot to MiR Fleet to use to setup and test the site with a single robot.
- When the site is complete and tested, add all other robots to MiR Fleet.

**Adjust the Wi-Fi network so robots have a stable connection across the whole site.** See ["Evaluate Wi-Fi" on page 285](#).

- Review the network requirements for MiR systems.
- Apply necessary adjustments to your site network.
- Send two robots across the site to gather Wi-Fi signal data.
- Evaluate the Wi-Fi performance and coverage.
- Document the Wi-Fi heatmaps made between any changes to the site network.

**Create groups to define which missions, charging stations, and Staging positions each robot can be assigned to.** See ["Create groups" on page 288](#).

- Determine which charging stations, Staging positions, and missions you want each robot to be compatible with.
- Create groups that enable each robot to access the necessary resources and missions.
- Add robots and resources to groups.
- Document the overview of groups, robots, and resources.
- Document how to determine which group new elements should be added to.

**Verify that robots are correctly managed by MiR Fleet.** See ["Test MiR Fleet" on page 291](#).

- Verify that data synchronizes correctly across all robots.
- Test that robots are assigned to missions as expected.
- Test that robots are assigned to charging stations and Staging positions as expected.
- Test that robots still run correctly after turning them off and on again.

### 4.6.1 Prepare robots

#### **Section overview**

For each robot, check the robot's condition and set up the robots to prevent unauthorized access.

- Go through the received product inspection checklist.
- Upgrade the robot software.
- Protect against booting up from USB.
- Set a password for the robot BIOS.
- Disable all unused features.
- Disable or password protect robot access points.
- Update robot access points.

## Check your robots

Go through the *Received Product Inspection Checklist* for each robot to check for out-of-box issues.

In the latest versions of the quick starts and integrator manuals, you can find a printed version of the checklist.

Store the checklist for at least 10 years and make it available upon request.

## Upgrade the software

Ensure that all robots are upgraded to the same software version—see ["Upgrade software" on page 181](#).

## Prevent unauthorized access to the robots

The following sections describe how you can protect robots against unauthorized access.

### Protect from USB boot

The robot computer will boot up from any connected USB drive. We recommend protecting your robot by:

- Restricting access to any USB ports on the robot. Use a USB lock to close all exposed ports.
- Disabling the boot from USB function. This can become an issue if you ever need to restore the robot from a corrupt platform configuration.

If you choose to disable boot from USB, contact MiR Technical Support.

### Protect the BIOS

Any user can access the BIOS by default. You can enable password protection to prevent attackers from changing the BIOS.

To do this, connect a keyboard and screen to the robot computer. Turn on the robot and interrupt normal startup by pressing a key before the loading screen appears:

- For most MiR robots, press the Delete key.
- For MiR100 hardware version 4.0 and lower, press the F2 key.

Once you are in the BIOS, set a password.

### Restrict Unused Functionality

To close off unused communication ports, disable any unused MiR robot features.

For each robot, in the robot interface, go to **System > Features**, and disable any of the features the robot does not need.

### Protect or disable Wi-Fi access points

Older robots were created with built-in access points you could use to connect to the robot network wirelessly. Newer models do not support this for cybersecurity reasons, but we offer access point dongles that you can connect to robots instead.

MiR supports connecting to your robot's network wirelessly for initial configuration. Leaving the access point enabled when the robot is operating poses a risk to the robot's cybersecurity.

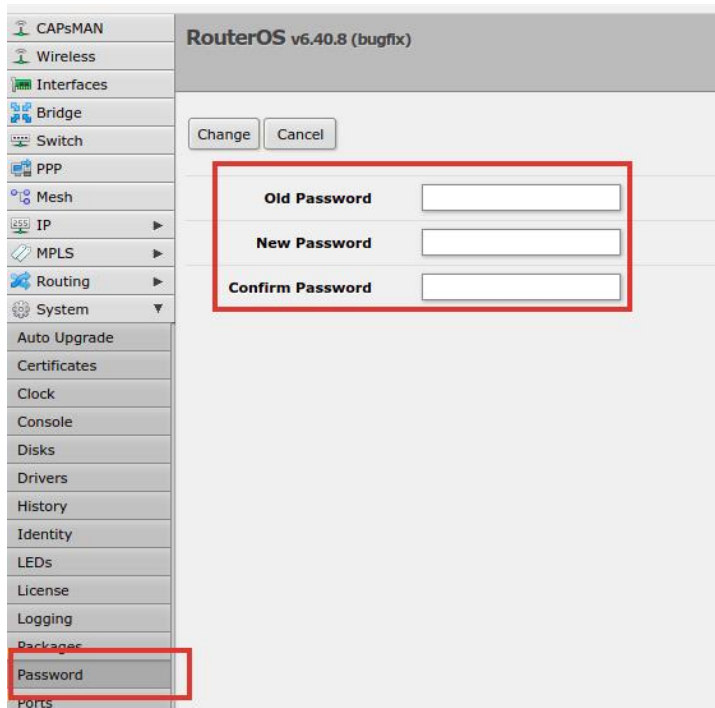
If you choose to leave the access point enabled during operation, change the password of the hotspot and the configuration page to increase the security.

To configure the security or disable an access point (either the inbuilt access point in older models, or MiR Access Point dongle), follow these steps:

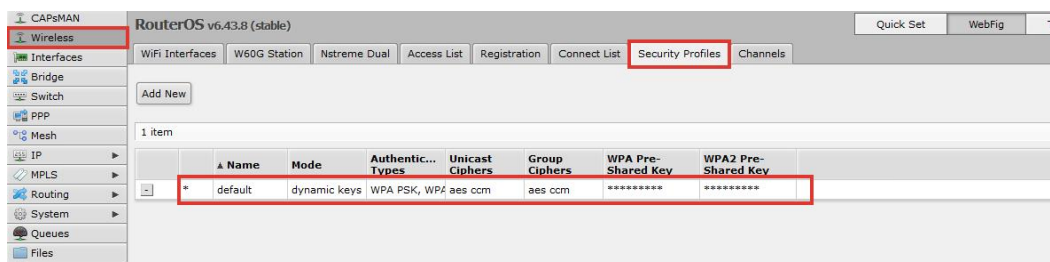
- 1 Connect to the access point of your robot or dongle.
- 2 Open a browser, and go to the address **192.168.12.1/webfig**. This is the access point configuration page.
- 3 Enter the following default credentials to sign in:
  - Username: root
  - Password: mirex

4 Follow the step relevant for the change you want to make:

- To change the password used to sign in to the access point configuration page, go to **System > Password**, and enter a secure password.

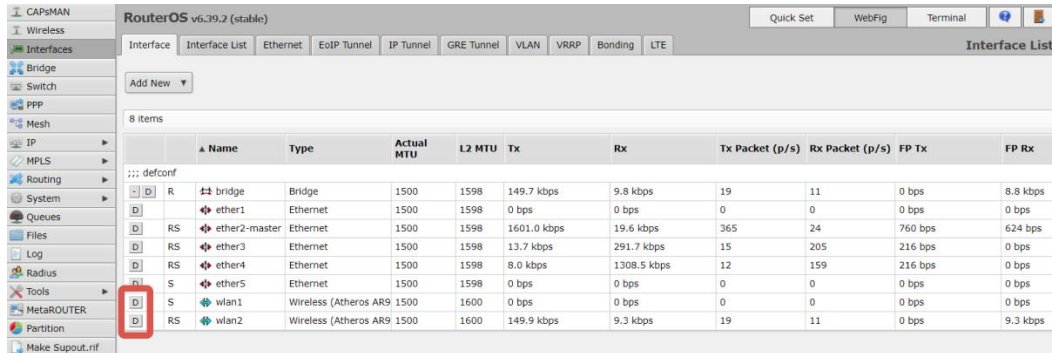


- To change the password used to sign in to the network, go to **Wireless**, select the access point profile (do not select the -), and go to **Security Profiles**. Change the password under the two **Pre-Shared Key** fields.





- To disable the internal access point, go to **Interfaces**, and select **D** for the two **wlan** interfaces to disable them.



### Update access point or switch firmware

MiR robots support up to firmware version 6.49.13 on the Mikrotik switches and routers. Update to this version to ensure you have the latest supported version and the cybersecurity patches it includes.

To update the firmware, follow these steps:

- 1 Download version [6.49.13 for MIPSBE](#) from [Mikrotik's website](#). We do not support any version 7 firmwares and can only guarantee full compatibility with version 6.49.13.
- 2 Connect your device to the same network as the switch, router, or access point you want to upgrade.

3 Open WinBox on your computer and enter the following:

- Connect To:
  - 192.168.12.1 (for a robot switch or router)
  - 192.168.12.2 (for MiR Access Point)
- Login: admin
- Password: (blank)



The screenshot shows the WinBox (64bit) v3.28 (Addresses) interface. The 'Connect To' field is set to 192.168.88.1, the 'Login' field is set to admin, and the 'Password' field is empty. The 'Connect' button is highlighted with a red box.

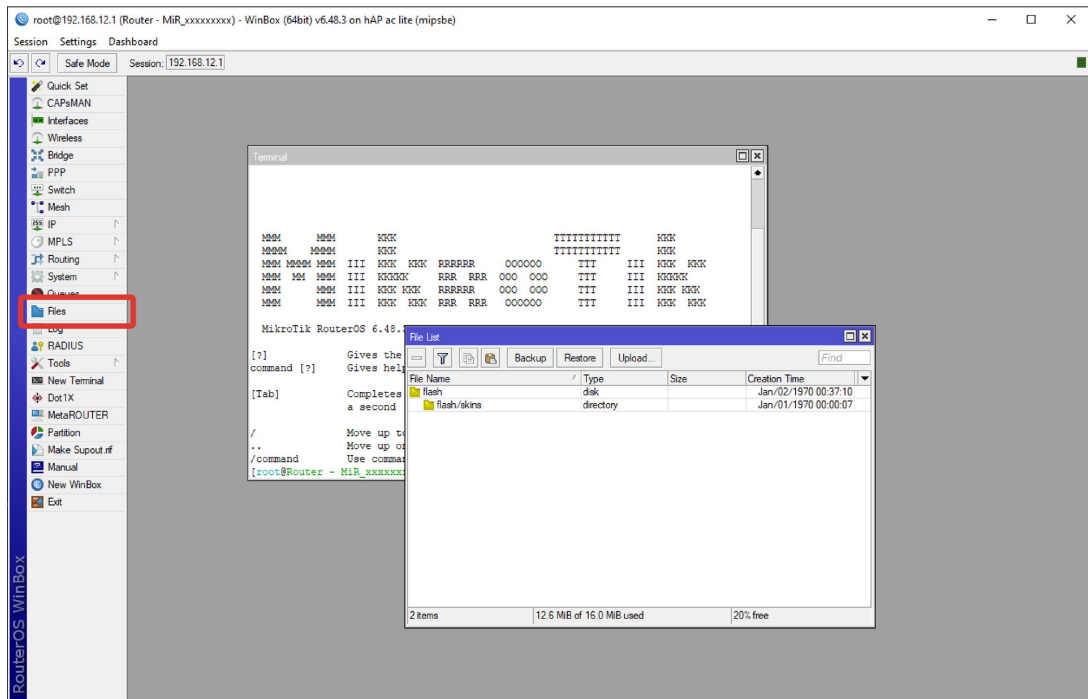
If an error is reported, you may have to enable **Legacy Mode** from the **Tools** tab.

4 Select **Connect** to connect to the router.



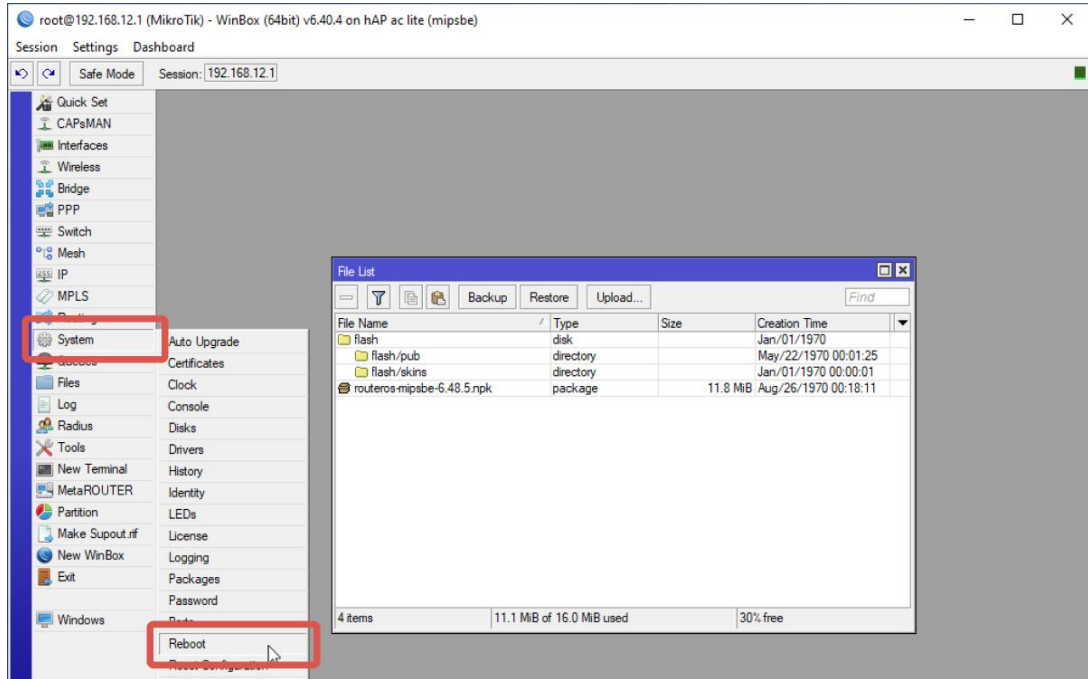
The screenshot shows the WinBox (64bit) v3.28 (Addresses) interface. The 'Connect To' field is set to 192.169.88.1, the 'Login' field is set to admin, and the 'Password' field is empty. The 'Connect' button is highlighted with a red box.

5 Select **Files** to open the file list.



6 Drag and drop the firmware file into the File List in the root folder. Do not upload the file into any existing folders.

- 7 Go to **System > Reboot** to reboot the device. Do not disconnect the device from its power supply, this will remove the firmware file.



- 8 Wait between three to five minutes for the new firmware to be applied. You will be disconnected from WinBox, and are not informed when the update is complete. Once you can reconnect to the router, switch, or access point, the update is complete.

## 4.6.2 Add robots

### Section overview

Connect robots to MiR Fleet.

- Connect robots to the same Wi-Fi network as MiR Fleet.
- Set up mutual trust between MiR Fleet and the robots.
- Add one robot to MiR Fleet to use to setup and test the site with a single robot.
- When the site is complete and tested, add all other robots to MiR Fleet.

When you prepare robots, you have to access the robot interface. To do this, see the user guide or integrator manual for your robot application. You can find these guides on [MiR Support Portal](#).

### Connect robots to Wi-Fi

Connect robots to a Wi-Fi network in the robot interface under **System > Settings > Wi-Fi**. To do this, see the user guide or integrator manual for your robot application. You can find these guides on [MiR Support Portal](#).

When you create a Wi-Fi connection, contact your IT department to agree on the correct configuration for the Wi-Fi connection:

- **Network:** Select the network you want to connect to from the list of available networks. If you cannot see the network you are looking for, select **Reload networks**.
- **Description:** Describe the network you want to connect to.
- **Security type:** Select a security protocol.
- **Background scan frequency:** Select whether the robot must scan for a better Wi-Fi signal more often. This means the robot is more likely to be using the strongest signal more often, but may also make the robot jump more frequently between access points.
- **Channel selection:** By default, the robot scans all frequency channels across the 2.4 GHz and 5 GHz bands. To reduce scanning time, if you know which channels your network is using, you can limit the robot to only scan on the necessary channels. If you select Custom channel set, you have the following two options:
  - **2.4 GHz channels:** Enter the channels the robot should scan on the 2.4 GHz band.
  - **5 GHz channels:** Enter the channels the robot should scan on the 5 GHz band.
- **Password:** Enter the password used to sign in to the network. The network must be password protected.
- **DNS:** If relevant, enter DNS servers using the format: xxx.xxx.xxx.xxx. Use semicolon (;) as the delimiter.
- **Use static IP:** To use a static IP address, select this check box and fill in: **IP address**, **Netmask**, and **Gateway**. Contact your IT department to determine if the network is configured to support static IP and to receive available addresses on the network—see "[IP addresses](#)" on page 214.

When you are connected to the same network, you can access the robot's interface by entering the IP address displayed under the connection description into your internet browser.

### Default gateway IP address

Instead of using a Wi-Fi connection, you can set the robot's default gateway IP address without a Wi-Fi connection being activated. Use this if you utilize an external router or 5G CPE that is connected to the robot via Ethernet to extend the robot's wireless capabilities. Do **not** use this feature along with the normal Wi-Fi connections. For more information, see the guide *How to connect a robot using a 4G or 5G cellular modem*. You can find this guide on [MiR Support Portal](#).

Note that the communication ports depends on which software version robots are running—see "[Meet system requirements](#)" on page 209.

To use this setting, set the **Default gateway** to **Static** under **System > Settings > Wi-Fi > Advanced**. Enter a valid IP address under **Gateway IP**. The IP address must be in the 192.168.12.1–192.168.12.99 range and not reserved for internal use. If the entered address cannot be reached or is invalid, the robot will report an error.

### Add your first robot

Before setting up your site, add a single robot to MiR Fleet to act as your model robot. Use this robot to create and test your site.

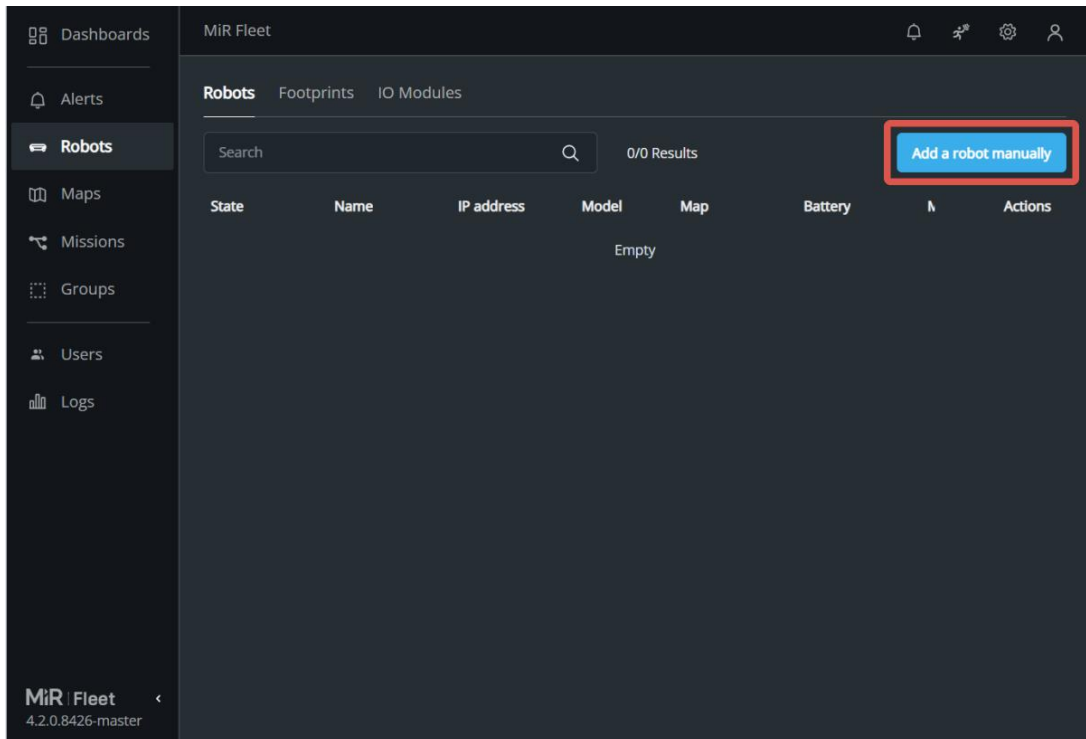
Using a single robot for set up makes it easier to avoid synchronization issues, and verify that your set up works. This means issues that arise later are likely only due to resource assignment, deadlocks, groups or other settings relating to multiple robots.

For each robot you add to MiR Fleet, you must set up mutual trust—see "[Mutual trust page](#)" on page 110. If the robot or MiR Fleet IP address ever changes, you must generate a new mutual trust agreement between them.

To add your first robot to MiR Fleet, follow these steps:

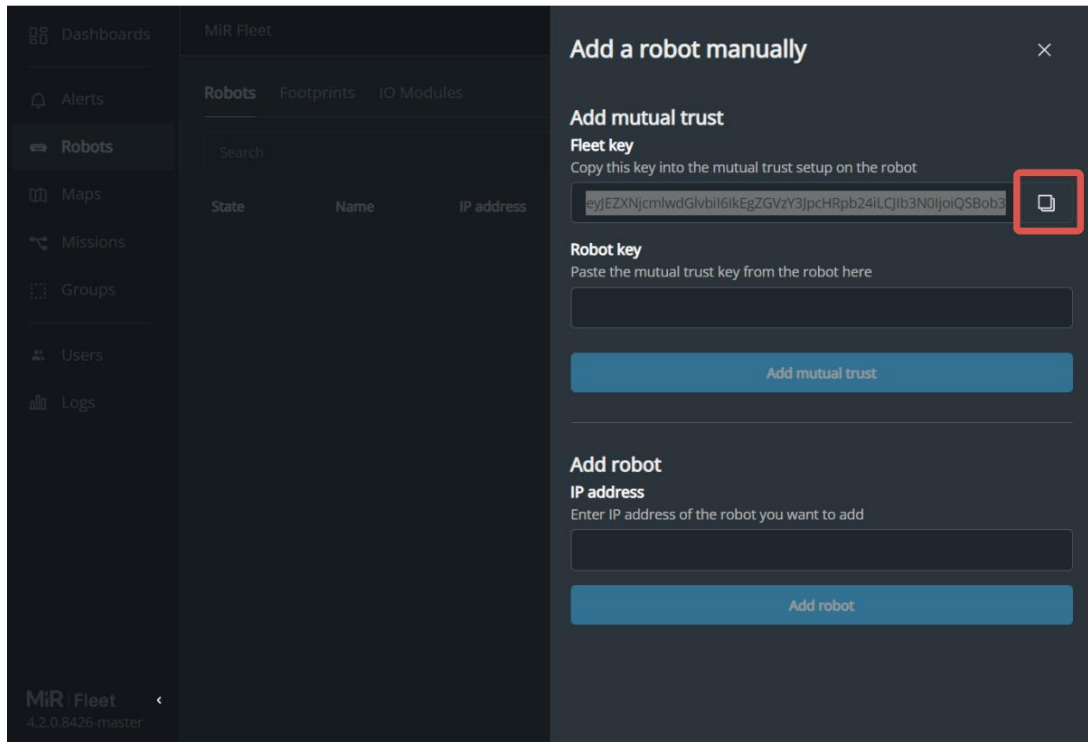
- 1 Connect your robot to the same Wi-Fi network as MiR Fleet—see "[Connect robots to Wi-Fi](#)" on the previous page.
- 2 In the MiR Fleet web-interface, go to **Robots**.

3 Select **Add a robot manually**.



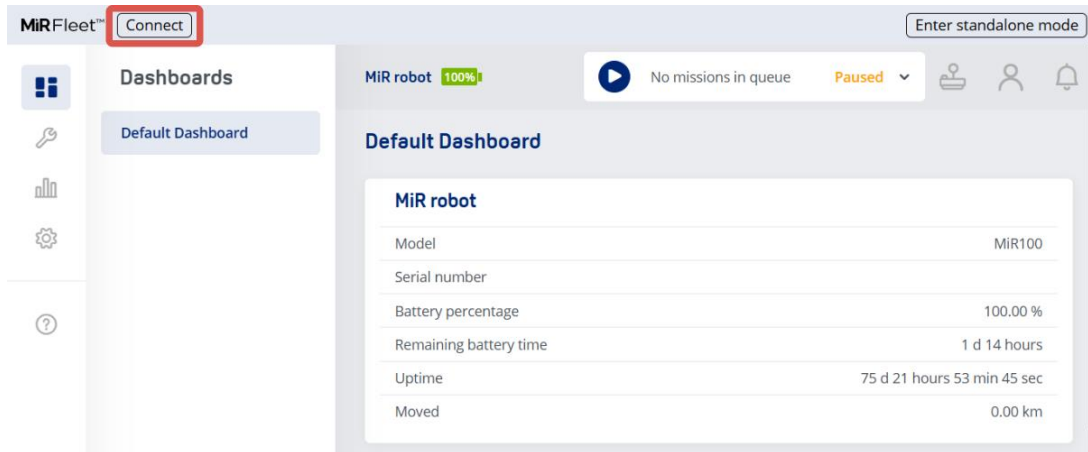
- 4 Under **Fleet key** copy the key.

If you are using Google Chrome, the Copy button is only displayed if you allow Clipboard permissions in your browser settings.

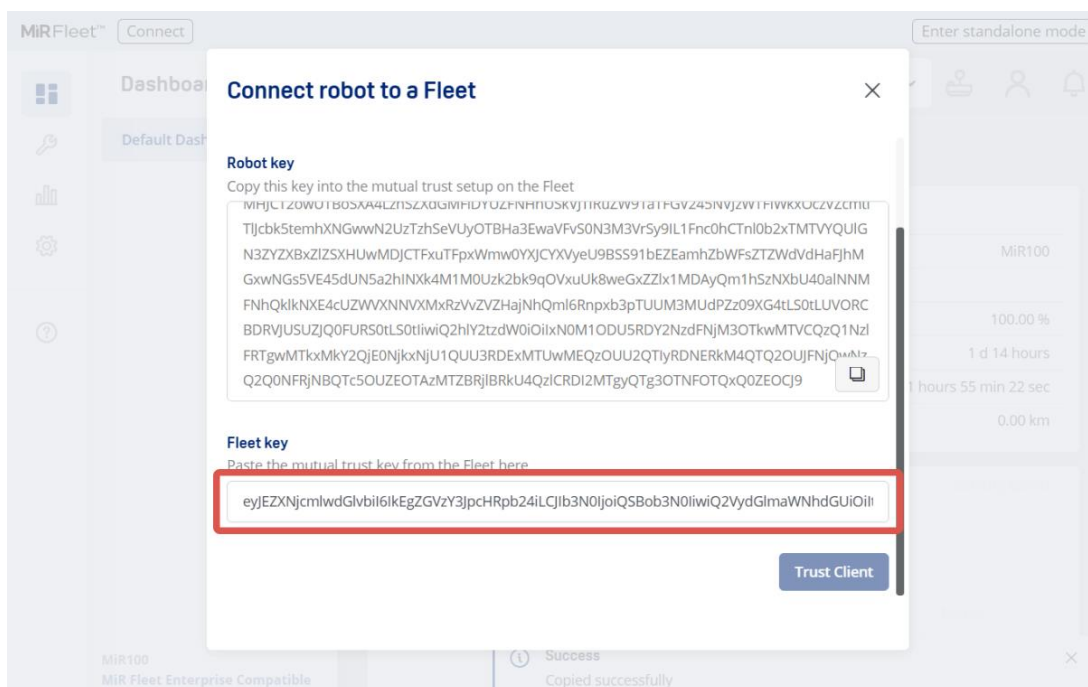




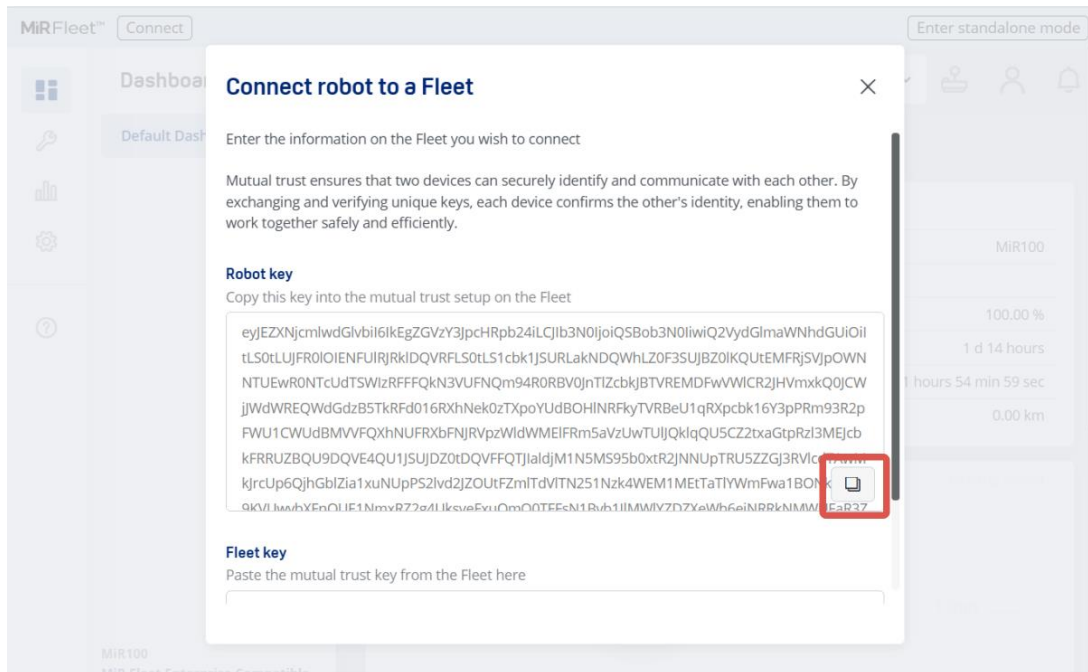
- 5 Open the robot interface on your robot—see "Interfaces and modes" on page 32—and select **Connect**.



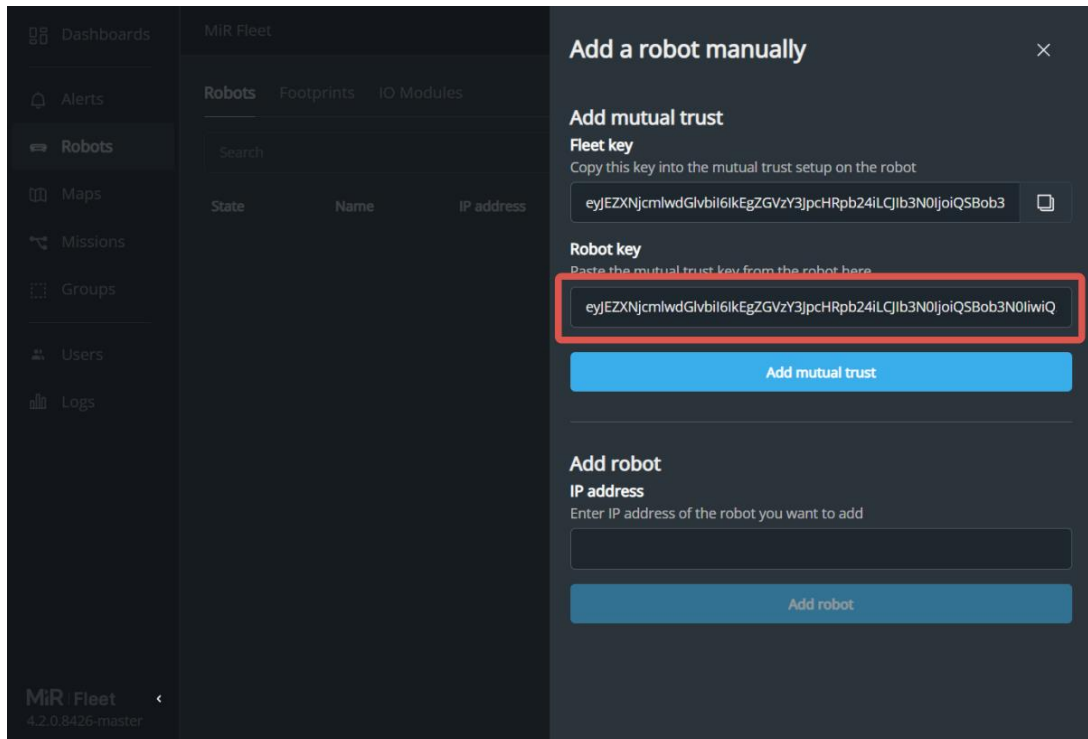
- 6 Paste the copied fleet key under **Fleet key** and select **Trust client**.



7 Under **Robot key** copy the key.

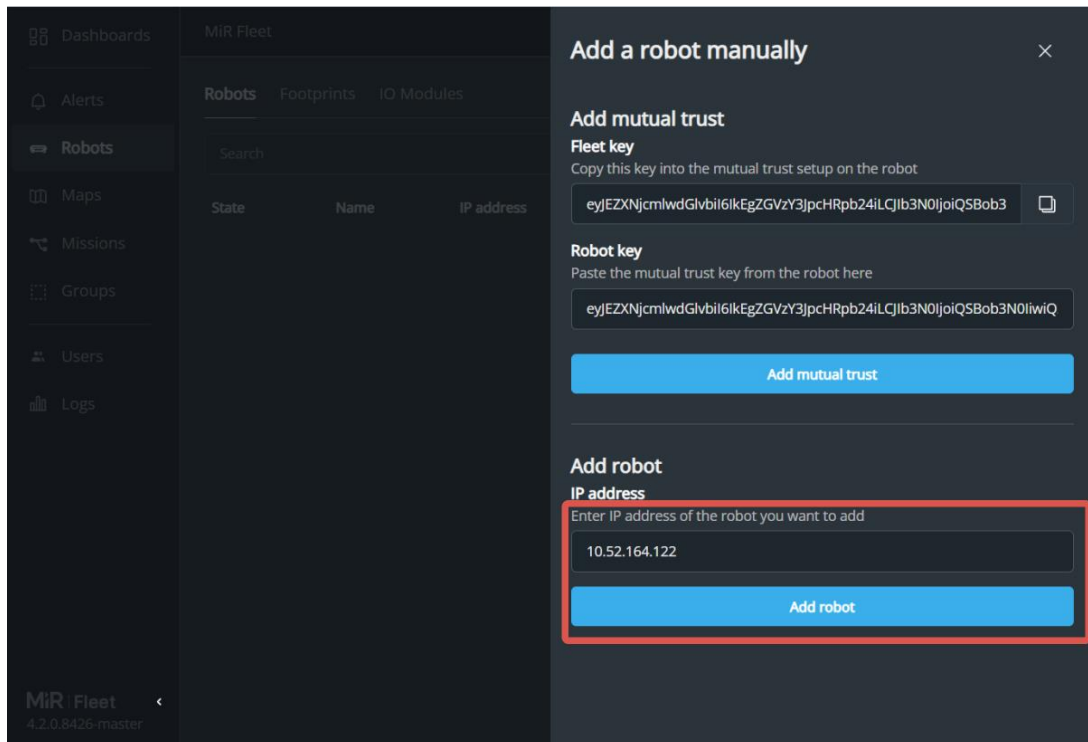


- 8 In the MiR Fleet interface, paste the robot key under **Robot key**.



- 9 Select **Add mutual trust**.

- 10 Enter the IP address of the robot under **IP address**.



- 11 Select **Add robot**. If MiR Fleet does not register that the robot exists, verify that the robot and MiR Fleet are on the same network and sub-net.

### Add remaining robots

Do not add the remaining robots until you have finished creating and testing the site with the model robot. It is easiest to finish setting up the site and validating it with a single robot first.

When you add robots to MiR Fleet, all of their site data from MiR Fleet is pushed to the robot.

By following the same steps in "[Add your first robot](#)" on page 278, add all of the other robots you want to connect to MiR Fleet. For each added robot, go to the robot interface, and on the active map, localize the robot correctly—see "[Localize the robot](#)" on page 321.

For each robot, verify the following:

- The site data from MiR Fleet is correctly synchronized.
- The robot is displayed on the map in the MiR Fleet interface.
- MiR Fleet sends the robot to charge or stage automatically when the robot is idle.
- MiR Fleet can assign missions to the robot.

After testing the robots, assign them to groups—see ["Create groups" on page 288](#)—and verify that:

- MiR Fleet sends the robot to charge or stage automatically when the robot is idle.
- MiR Fleet can assign missions to the robot.

Continue to test the functionality of each of your missions—see ["Create missions and mission groups" on page 347](#)—while several robots are operating, and look for signs of:

- Poor traffic management
- Deadlocks
- Poor Wi-Fi connection

### 4.6.3 Evaluate Wi-Fi

#### Section overview

Adjust the Wi-Fi network so robots have a stable connection across the whole site.

- Review the network requirements for MiR systems.
- Apply necessary adjustments to your site network.
- Send two robots across the site to gather Wi-Fi signal data.
- Evaluate the Wi-Fi performance and coverage.
- Document the Wi-Fi heatmaps made between any changes to the site network.

#### Plan

Employ a Wi-Fi specialist to help set up your site.

Verify that your site meets the "[Meet system requirements](#)" on page 209 and review the *MiR Network and Wi-Fi Guide* for advice and general Wi-Fi principles. You can find this guide on [MiR Support Portal](#).

## Execute

MiR does not provide instructions for setting up your Wi-Fi network.

## Test

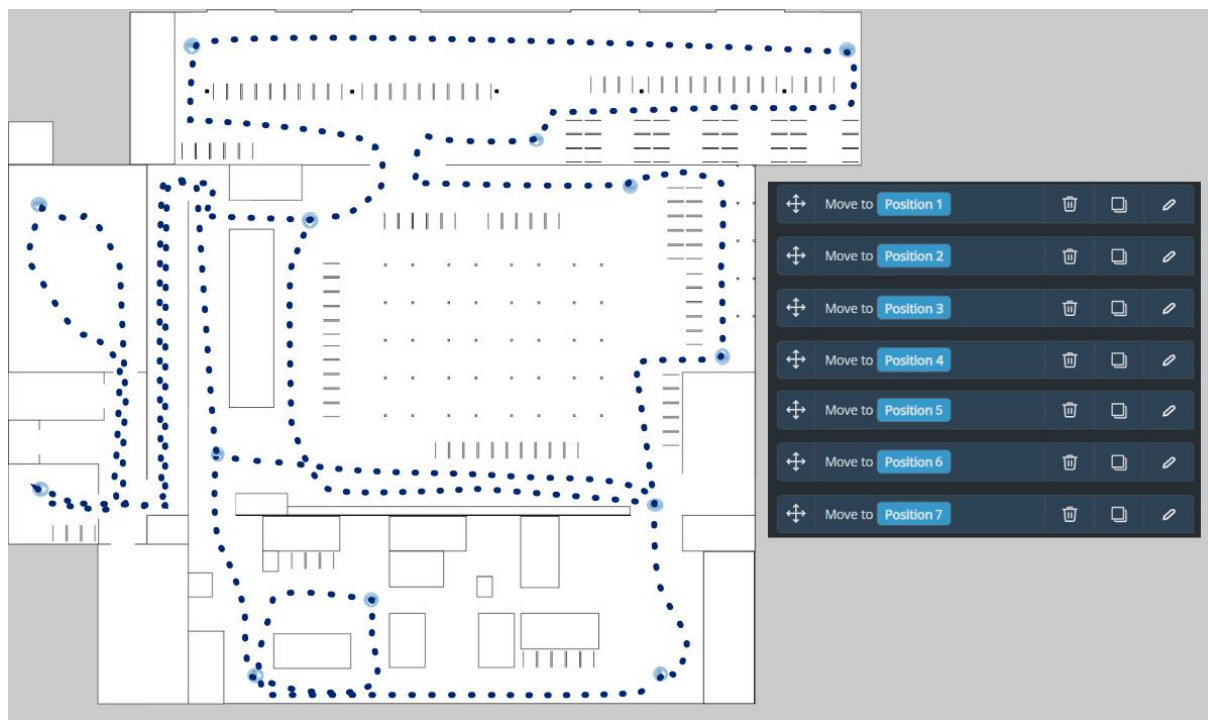
To test the Wi-Fi performance and coverage across the site, have at least two robots perform missions that make them travel across the entire site. While driving, the robots gather Wi-Fi connection data automatically that you can view using MiR Insights. If you do not use MiR Insights, use a third-party service to assess your Wi-Fi.

MiR Insights displays the data in a heatmap that enables you to immediately identify any area with poor Wi-Fi coverage. You can filter which Wi-Fi metric the heatmap is based on.

Using two different robots lets you evaluate if there is a difference between the robot performances. *MiR Network and Wi-Fi Guide* includes a section regarding modifications on robots that may affect their wireless performance. You can find this guide on [MiR Support Portal](#).

To test your site, follow these steps:

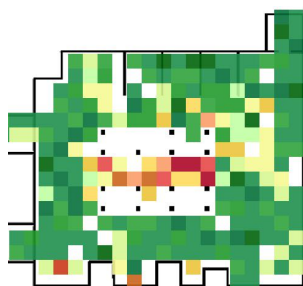
- 1 Place Robot positions on the map so that the robots will drive around the entire site when driving between them.
- 2 Make two missions that use Move actions to make the robots drive between the positions. Make the order that the robot moves between positions different in both missions.
- 3 Using two different robots, run the two missions you created. You can have them run at the same time, but make sure to monitor the robots during the process to make sure they do not block each other. Make the robots execute the mission twice in case there is any difference in the environment depending on the time of day.



Red areas in the **signal strength** heatmaps indicate that the signal strength is not sufficient for MiR robots to operate reliably. This identifies areas where you are missing access points or need to increase the signal strength of nearby access points.

Red areas in the **access point** heatmaps indicate that the robot spends time roaming between access points because there are too many overlapping signals, without any that are significantly stronger than others.

**Figure 4.4** Example of a heatmap with poor connection in the center



Review the heatmaps, and use the guidelines in *MiR Network and Wi-Fi Guide* to improve the Wi-Fi coverage in any areas where the heatmaps are red. MiR offers conducting a thorough Wi-Fi analysis of your site as a service.

## Document

- Document the heatmaps and any changes made since the last heatmap. Check the updated heatmaps regularly to evaluate the Wi-Fi network and react to any issues.
- Ensure the your IT department or Wi-Fi specialist fully documents the Wi-Fi network setup and provides a maintenance plan. If you expand or modify your site, you will need to re-evaluate the Wi-Fi network setup.

### 4.6.4 Create groups

#### Section overview

Create groups to define which missions, charging stations, and Staging positions each robot can be assigned to.

- Determine which charging stations, Staging positions, and missions you want each robot to be compatible with.
- Create groups that enable each robot to access the necessary resources and missions.
- Add robots and resources to groups.
- Document the overview of groups, robots, and resources.
- Document how to determine which group new elements should be added to.

Use groups to define which missions specific robots can be assigned to, and define which charging stations and Staging positions they can be sent to for Auto charging and Auto staging.

For more information about how groups works, see ["Groups" on page 30](#).

Groups can solve issues like:

- Robots not being assigned to appropriate missions.
- Robot traveling unnecessary distances.



## Plan

Each group you define must include a MiR robot to have an effect. You can choose whether to:

- Create mixed groups with several robots and several elements. This option often results in the fewest number of groups, but can be difficult to follow. If you choose this option, we recommend choosing robots of a specific category, such as:
  - The top module on your robot
  - The robot model
  - Location
  - Priority
- Create a group for each robot to define which missions and elements each robot can use. This makes it easy to see what each robot has access to, but if you have many robots, it takes time to set up and keep up to date.
- Create separate groups for missions, Staging positions, and charging stations. This option keeps a similar structure to MiR Fleet running software 3.x.

MiR Fleet only sends robots to charging stations that they are compatible with. You do not need to make groups to split between MiR Charge 24V and the MiR 48V charging stations.

If you have different MiR 48V charging station, and you want MiR1200 Pallet Jack to only use Charger 48V 105A, you must use groups to define this behavior.

As you create elements in your site, add them to appropriate groups. Doing this gradually can help keep track of your elements and intentions.

You can always refine your groups later. Start with the largest groups you want to define and then create more groups as you find cases where it is necessary.

## Create groups

To set up a group:

- 1 In the MiR Fleet interface, go to **Groups**.
- 2 Select **Create** and enter a name for the group.

- 3 Select the robots, mission groups, Staging positions, and charging stations you want to be in the group. All entities can be added to any number of groups.
- 4 When you have added all the entities you want to share a group, select **Create**.
- 5 As you add more robots, mission groups, and components to your site, make sure to update groups by selecting **Edit** for each relevant group.

## Test

- **Charging stations:** Wait for a robot to be assigned a charging station. Once you have verified the robot is assigned an appropriate charging station, assign the robot to another mission to keep it occupied while you test the other robots. Repeat until you have verified that all robots are sent to intended charging stations.
- **Staging positions:** Wait for a robot to be assigned a Staging position. Once you have verified the robot is assigned an appropriate Staging position, assign the robot to another mission or send it to a charging station to keep it occupied while you test the other robots. Repeat until you have verified that all robots are sent to intended Staging positions.
- **Mission groups:** For each mission you create, schedule the mission and verify it is assigned to a valid robot.

## Documentation

Create an overview like the example in ["Groups" on page 30](#) to make it easy to see which resources each robot has access to.

## Maintain

Update groups whenever you:

- Create a new charging station or Staging position.
- Add or remove a mission to a mission group.
- Add a new robot to MiR Fleet.

Always update the documentation when you make any changes.

### 4.6.5 Test MiR Fleet

**After reading this chapter, you can:**

**Verify that robots are correctly managed by MiR Fleet.** See ["Test MiR Fleet" above](#).

- Verify that data synchronizes correctly across all robots.
- Test that robots are assigned to missions as expected.
- Test that robots are assigned to charging stations and Staging positions as expected.
- Test that robots still run correctly after turning them off and on again.

To ensure that MiR Fleet is running as intended, always test the MiR Fleet features after setting up or updating MiR Fleet. These tests should always be done after you have added all robots to MiR Fleet.

#### Auto charging and Auto staging

To check that robots Auto charge and Auto stage correctly, check the following:

- Check that you have a sufficient number of charging stations and Staging positions—see ["Auto Charging and Auto Staging" on page 13](#).
- Verify that robots are sent to charging stations or Staging positions as expected. If there is any unexpected behavior, review the Auto charging and Auto staging settings.
- If MiR Fleet has successfully sent charging missions and staging missions to robots, restart MiR Fleet. Once MiR Fleet starts up again, verify that MiR Fleet sends the robots to a valid charging station or Staging position.

#### Synchronization

To check that all the site information that is modified on MiR Fleet is correctly synchronized across the fleet, follow these steps:

- 1 Apply the following changes in MiR Fleet:
  - Add a new zone to one of the maps.
  - Modify an existing mission.
  - Modify a footprint.
- 2 Verify that the changes applied in MiR Fleet apply when robots execute the mission or when viewing the map in the robot's interface. If you modify the robot's current footprint, the robot must execute a Set footprint action to update the footprint with modifications from MiR Fleet.
- 3 Remove the three modifications in the MiR Fleet interface and verify that the changes are synchronized to all connected robots.

### Mission scheduling

To check that MiR Fleet correctly assigns robots to missions, follow these steps:

- 1 Schedule a mission with an argument, and verify that it is assigned to a valid robot and that the mission status updates correctly through Pending, Executing, and Done in the scheduler.
- 2 Schedule another mission through MiR Fleet, and after a valid robot begins to execute it, abort the mission in the MiR Fleet scheduler, and verify that the robot also aborts the mission.

### Reconnect robots

To check that robots can reconnect to MiR Fleet after being turned off without issues, follow these steps for each robot:

- 1 Turn off the robot.
- 2 Turn on the robot and wait for it to finish starting up.

- 3 Sign in to the robot's interface and check **Monitoring > Hardware health**. This will indicate if there are any hardware issues on the robot.
- 4 Do not assign any mission to the robot and verify it is sent to auto charge or stage after the Idle time.
- 5 Schedule a mission for the robot and verify it is assigned the mission and starts executing it.

## 4.7 Set up navigation

### **After reading this chapter, you can:**

Create maps for MiR robots to autonomously navigate their operating area.

**Record or upload accurate floor plans.** See ["Create floor plans" on the next page](#).

- Determine your floor plan strategy.
- If you are recording any floor plans, prepare the operating area.
- Optimize the floor plan quality.
- Remove invalid floor plans from MiR Fleet.

**Combine floor plans to create maps.** See ["Create maps" on page 311](#).

- Determine the number of maps you need to cover the operating area.
- Determine the naming scheme for your maps.
- Align floor plans and merge them together to create a straight map.
- Add or remove obstacles to make the maps accurate.
- Document which areas each map covers.

**Ensure robots can localize themselves across the whole operating area.** See ["Evaluate localization" on page 321](#).

- Localize all robots on the active map.
- Drive one robot across the whole map.
- Evaluate the localization score.
- Apply map changes if necessary.
- Document the final localization score and evaluate the localization regularly to ensure the map continues to be accurate.

**Create transitions between maps.** See ["Create transitions" on page 325](#).

- Create transition positions where maps overlap.
- Create transition missions for each map transition.
- Define transitions.
- Test that a robot can travel to and from each map using the appropriate transition.
- Document the transitions, including the relevant map locations, positions, and missions.

## 4.7.1 Create floor plans

### Section overview

Record or upload accurate floor plans.

- Determine your floor plan strategy.
- If you are recording any floor plans, prepare the operating area.
- Optimize the floor plan quality.
- Remove invalid floor plans from MiR Fleet.

This section prepares you to create floor plan suited for your site and explains how to create or upload floor plans.

For more information about creating maps with your floor plans, see ["Create maps" on page 311](#).

The following aspects affect the quality of the floor plan:

- **Noise:** Mapped walls in places without actual obstacles block the robot from making paths and reduce the localization performance.
- **Wall quality:** Grainy, uneven, and thick lines reduce the localization and planning performance.
- **Floor quality:** Incomplete floor areas where parts of the floor is missing makes the floor plan difficult to plan efficient paths on. Unrecorded floor is displayed as gray patches.
- **Unused areas:** Areas that the robot will not operate in should not be mapped to minimize the data the robot uses for planning.
- **Accuracy:** The scale and alignment of the floor plan should match the physical environment.
- **Obstacle consistency:** Obstacles included in the floor plan must be stationary. If an obstacle that is drawn on the map is moved, the floor plan must be updated.
- **Alignment:** Floor plans should align with the X-Y coordinate system they are based on and long stretches of floor plan must remain straight.

## Plan

Before creating floor plans, you must determine the method best suited for your site and resources and prepare accordingly.

### Recorded versus uploaded

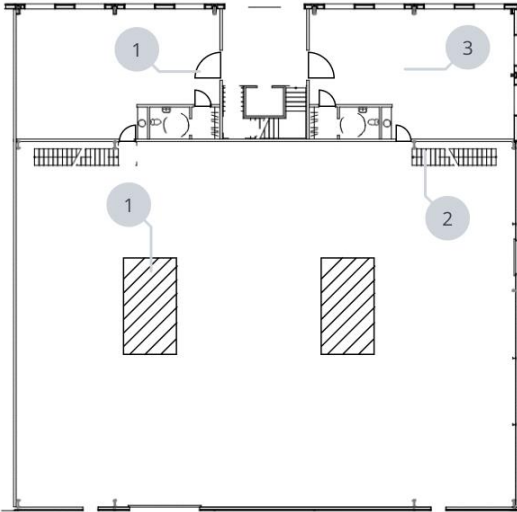
There are two ways to create a floor plan: You can upload a black and white image file to the robot with the floor plan based on the site blueprints, or you can use the robot to manually record a floor plan of the area. Determine which approach you want to apply based on the following guidelines.

	Upload	Record
Intended for	Best suited for sites with where the operating environment matches the blueprint file of the site.	Best suited for sites where many static obstacles have been added (such as furniture, equipment, machinery, shelving).

	Upload	Record
Requirements	Must have an existing image file that is to scale with the real operating environment.	Must prepare the area and clear away dynamic obstacles.
Line quality	Clean and straight lines from the start.	Grainy lines that should be corrected after mapping.
Scale	Can be difficult to scale and predict what is relevant data for the robot.	Will detect everything exactly as it will be seen when operating.
Post-processing	May have to record and append data for obstacles that were not in the original file.	May have to remove obstacles that should not be in the floor plan.
Edit ease	Easy to make corrections while in a CAD program.	Start over if the floor plan is inaccurate.
Accuracy	As accurate as the blueprint to physical life scale.	Data can drift during the recording, so it can take several attempts to get an accurate map.

The following are examples of post-processing you need to apply to floor plans of the same site where one is recorded and the other is an uploaded CAD file. For instructions on how to fix any of the mentioned issues, see ["Create floor plans" on page 299](#).





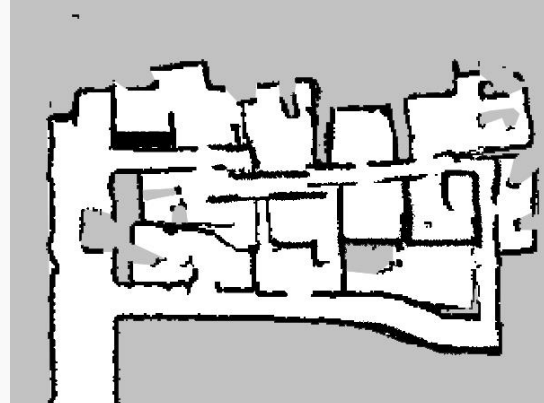
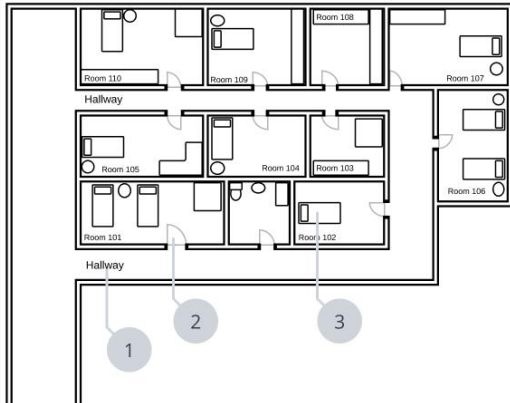
### Uploaded

- 1 Remove annotations and meta-information that are not relevant for the robot. Anything that cannot be detected by the laser scanners is irrelevant.
- 2 Remove objects that are not correctly represented as the robot would detect them at laser scanner height.
- 3 Remove areas where the robot will not operate or cover them with Forbidden zones. This reduces the area the planning algorithm had to compute and reduces planning time.
- 4 Missing objects must be added afterward—see ["Create floor plans" on page 294](#).



### Recorded

- 1 Make grainy lines straight and thin (between 1–5 px).
- 2 Remove dynamic objects.
- 3 Add floor or walls for features that were not completely mapped. Do not include incomplete obstacles.



### Uploaded

- 1 Remove text in areas the robots operate in. You can keep texts in areas robots do not operate in to annotate the floor plan.
- 2 Replace gray lines with black if they represent walls or other static obstacles. Erase them if they are irrelevant annotations.
- 3 Remove furniture and other items that are likely not placed exactly as indicated on the floor plan and are not represented as the robot would detect them at laser scanner height.

### Recorded

Remap recorded floor plans that become askew. This happens if the data drifts while recording. The causes of this are often the same as causes of poor localization—see ["Evaluate localization" on page 321](#).

### Segment the mapping process

If you are mapping a large area, map in segments to "save" your progress. If you do not segment the mapping process you risk:

- Having to restart the entire mapping process if you are unhappy with parts of the floor plan or you were interrupted.
- The floor plan losing accuracy and becoming askew.
- Accidentally remapping an area you were happy with to something worse.
- The robot may fail if you continue recording the same floor plan for over two hours.

After mapping smaller sections of the floor plan, you can combine them into a single floor plan using an image editor or using the append function in the MiR Fleet interface.

Decide on where you want to start and stop each of your floor plans:

- If you have a site large enough to justify making several maps—see ["Create floor plans" on page 294](#)—, segment the mapping process based on which areas you want to be separate maps.
- Make sure there is some overlap between mapped segments. This makes the floor plans easier to align afterward.

### Prepare the area

Before you map a new location, do the following preparations:

- Clear the area of dynamic obstacles, such as pallets and carts. Dynamic obstacles can also be deleted from the floor plan later.
- Ensure that all doors and gates that the robot should be able to go through are opened before mapping. These are ideal places to segment the mapping process, but to ensure that you have an overlapping area to align floor plans with, keep the doors open for comparison data.

### Create floor plans

When you create a new map, you select a saved floor plan you want to base the map on. You can append any number of saved floor plans to the selected floor plan.

You can save any number of floor plans in the system by either recording them on a robot, or uploading .png files to MiR Fleet.

**Record floor plan**

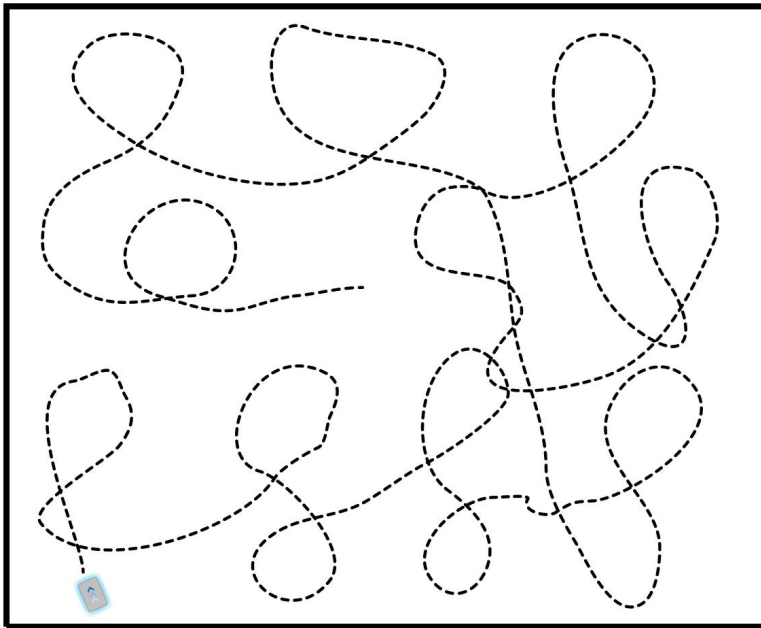
You record floor plans from the robot interface. When you record a floor plan, it is immediately saved to the connected MiR Fleet, and you can either use it to create a new map, or append it to an existing map.

To record a floor plan, drive the robot around its intended operating environment while its sensors gather data to generate a floor plan from. This process is known as mapping.

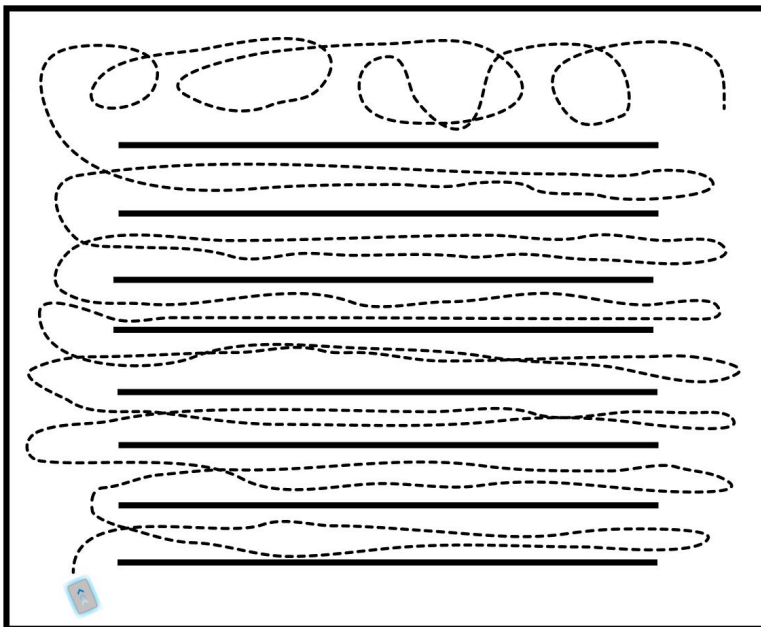
As the robot moves during mapping, the laser scanners detect physical obstacles, which are recorded on the floor plan as walls.

When mapping:

- Focus on mapping in a figure-eight pattern.



- When reaching long corridors with few obstacles, let the robot stay in position for approximately five seconds before moving down the corridor. Drive up and down the corridor until the corridor is straight.



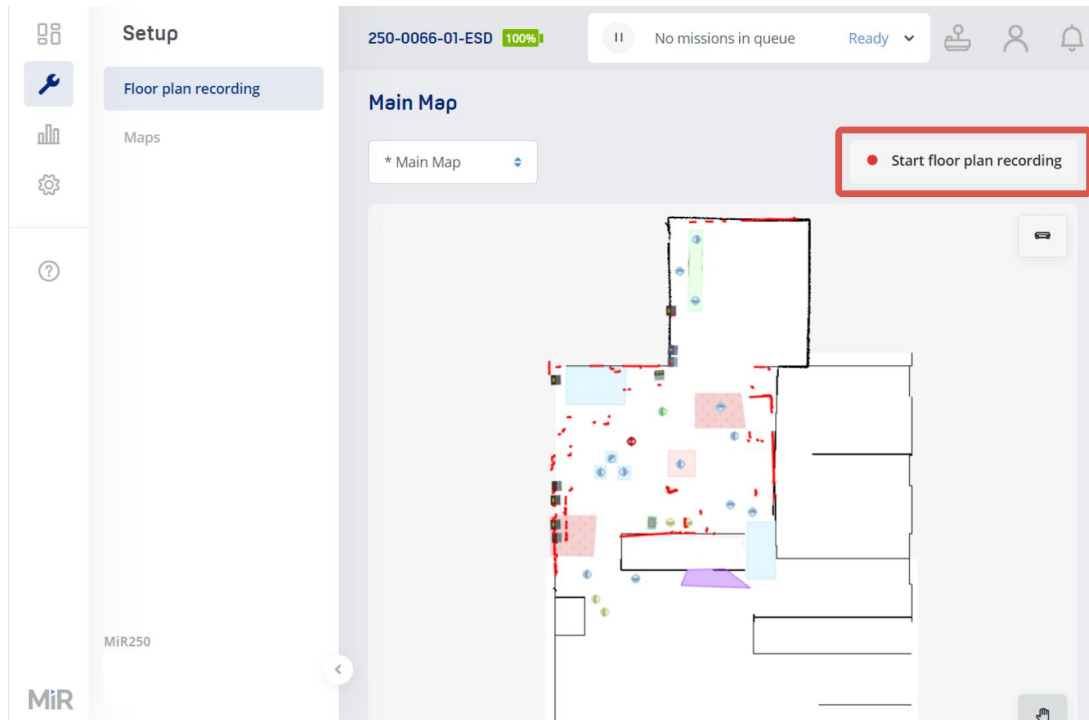
- Walk behind the robot as you map, or find a position where you can oversee the robot without being included in the floor plan.  
Start the mapping in a corner, and position the robot parallel to the wall.
- End the mapping in the same place you started it.
- Stop mapping when you have covered a large area and are happy with the result. You can always append recordings of other areas later.
- Use the smallest MiR robot you have purchased for greater maneuverability.

When mapping avoid:

- Starting the mapping with the robot in a very open space.
- Getting the robot stuck close to walls or objects as you will have to push it away manually.

To record the floor plan, follow these steps:

- 1 Open the robot interface on the robot you want to record the map with.
- 2 Go to **Setup > Floor plan recording** and select **Start recording**.



- 3 Move the robot around the site as you planned. Check that the floor plan looks correct and accurate.
- 4 Select **Stop recording** when you are satisfied with your floor plan recording. The recording is uploaded to MiR Fleet.



### Upload floor plan

If you choose to upload a floor plan, modify your floor plan file to meet the following requirements:

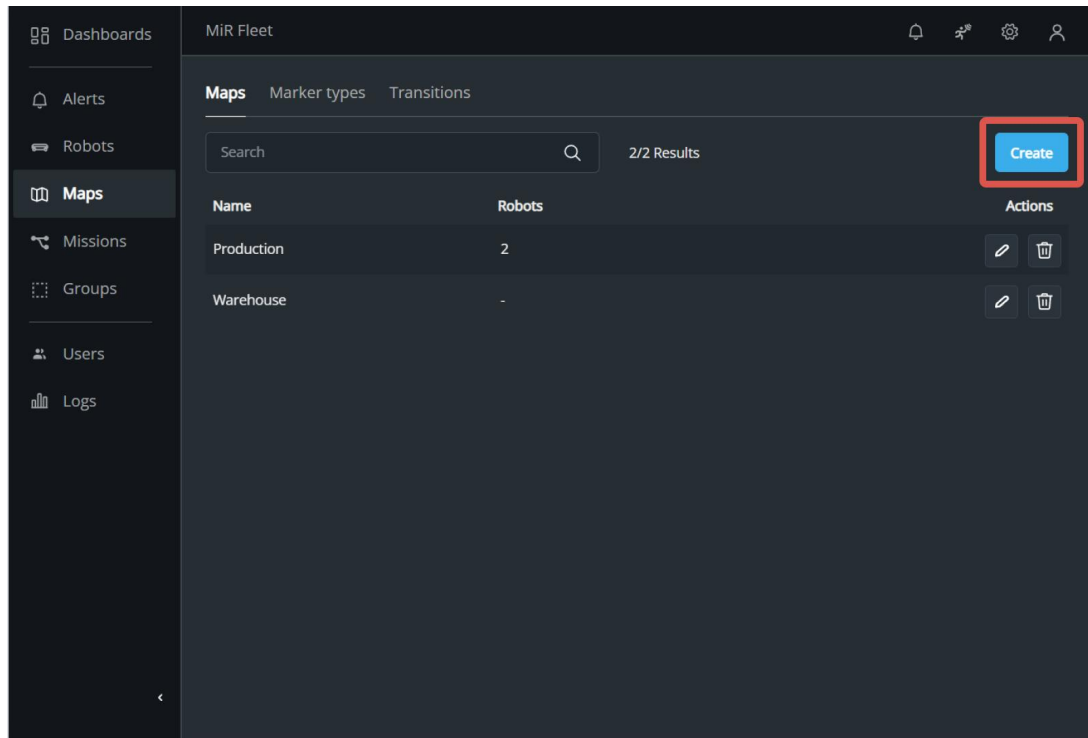
- Make sure all pixels are black for walls, white for floor, and transparent for empty areas without data.
- Remove noise from the floor plan. This includes annotations and anything that the robot cannot detect with the laser scanner (200 mm from the ground).
- Adjust the scale to 20 px to 1 m.
- Format the image as a .png file. We recommend ensuring the scale is correct and everything except the walls and fixed structures are removed before changing the format.

If you make any modifications to the floor plan after converting it to .png, when you save the file, check how the quality and size settings affect the pixel ratio.

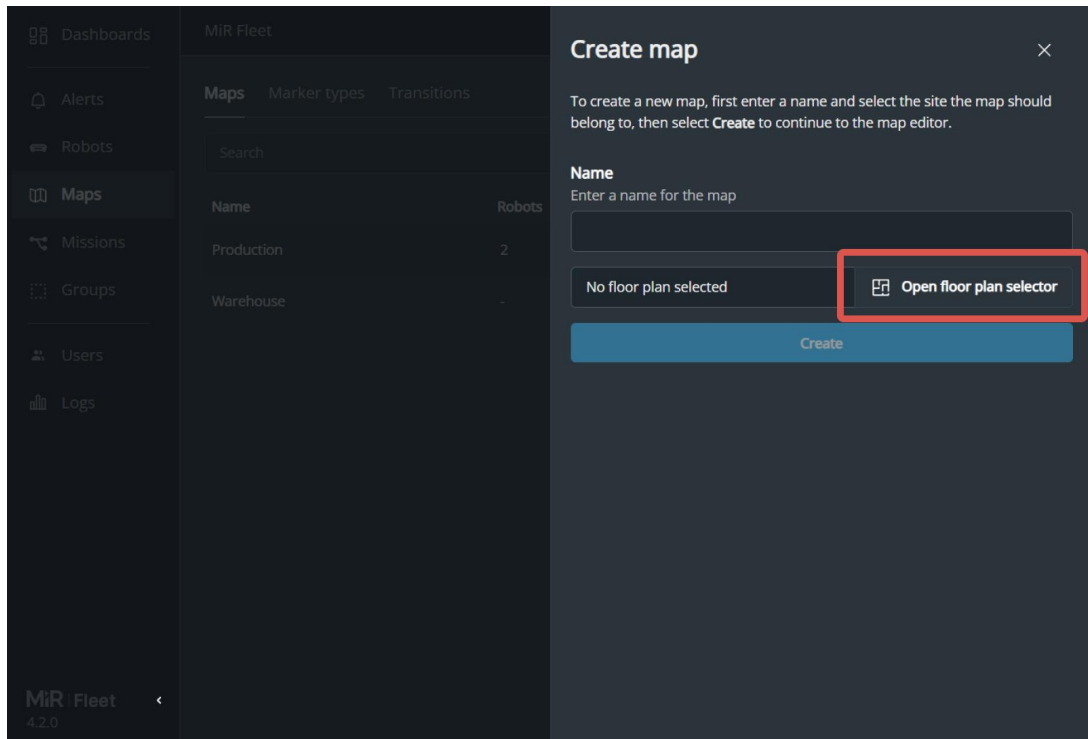
Later, you can update the floor plan with any information that is not already included. This can be stationary equipment and layout changes. Always do this directly on the robot as described in ["Add unmapped objects" on page 315](#).

To upload a floor plan, follow these steps:

- 1 Open the MiR Fleet interface.
- 2 Go to **Setup > Maps** and select **Create map**.



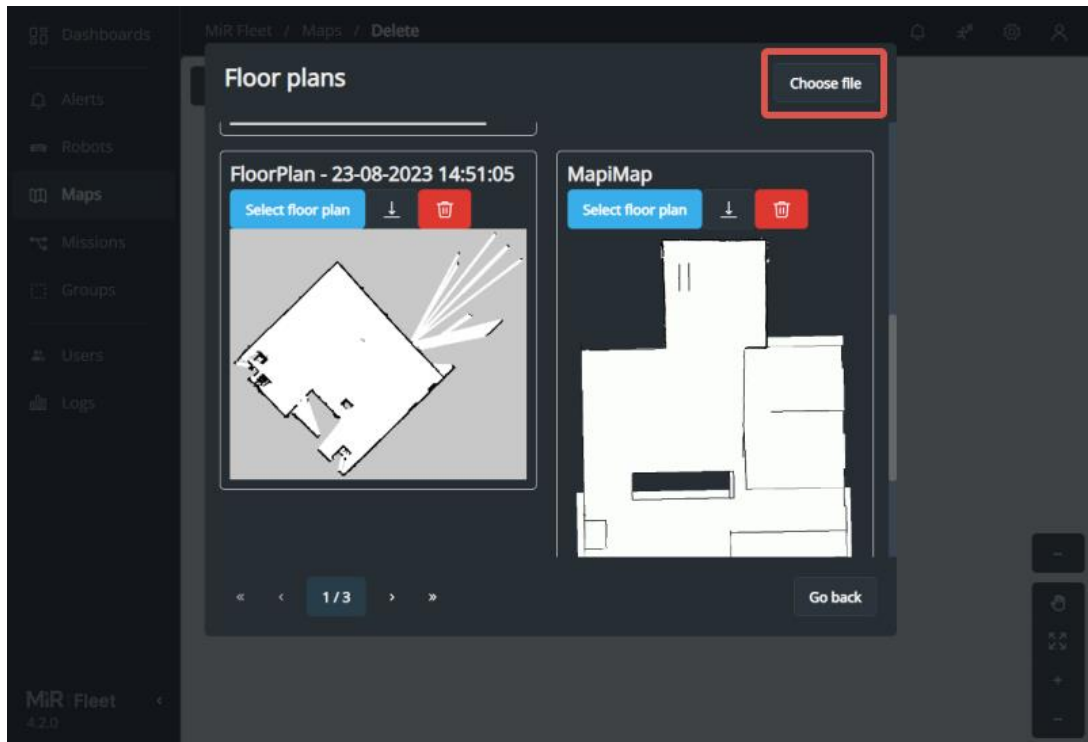
3 Select **Open floor plan selector**.



The screenshot displays the MiR Fleet software interface. On the left is a dark sidebar with navigation options: Dashboards, Alerts, Robots, Maps, Missions, Groups, Users, and Logs. The main area is titled 'MiR Fleet' and contains tabs for 'Maps', 'Marker types', and 'Transitions'. Below these tabs is a search bar and a table with columns 'Name' and 'Robots'. The table lists 'Production' with 2 robots and 'Warehouse' with 0 robots. A 'Create map' dialog box is open on the right, containing instructions, a name input field, a 'No floor plan selected' button, and a highlighted 'Open floor plan selector' button. A blue 'Create' button is at the bottom of the dialog.

Name	Robots
Production	2
Warehouse	-

4 Select **Choose file**.



5 Browse for the .png file you want to upload and select it.

The selected floor plan is now saved with the other floor plans.

### Optimize floor plan quality

Optimize the floor plan with the following:

- Remove all dynamic obstacles from the floor plan.
- Add lines for all walls and stationary structures.
- Adjust all black lines in the floor plan to be 1 px thick.
- Replace all gray spots with floor or walls, unless they are unused areas.

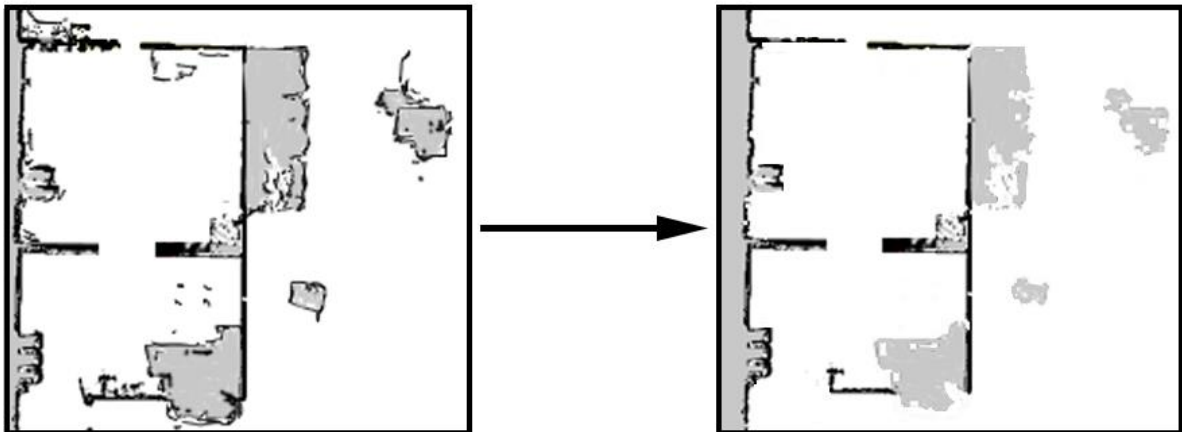
The robot navigates best when using a clean floor plan with as little noise as possible. [Figure 4.5](#) is an example of what a floor plan on a map can look like after the mapping process but where it still needs further editing.

**Figure 4.5** Example of a floor plan that includes too much noise and too many dynamic obstacles



After you add a floor plan to a map, you can use the following map tools to optimize the map. You can also make these changes in your preferred image editor if you export the floor plan:

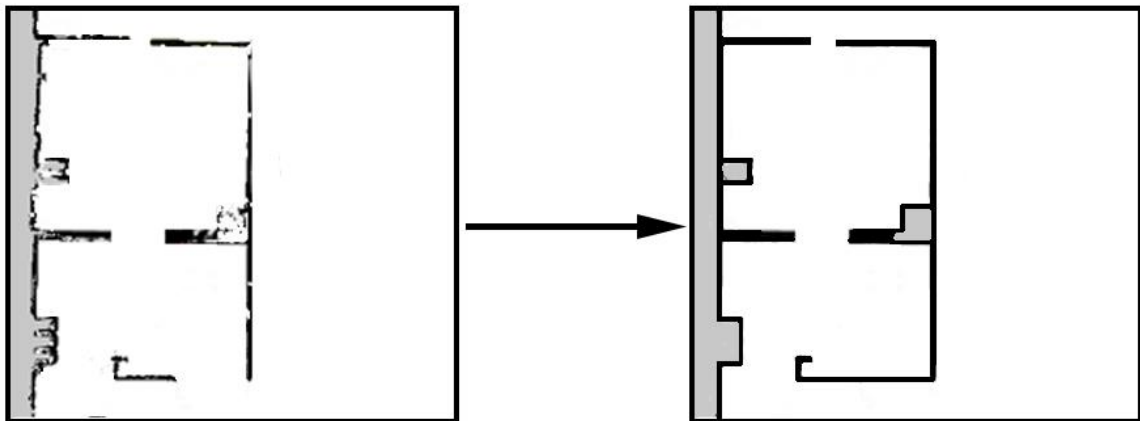
- Use **Tool type > Erase uploaded or recorded data** when editing walls to remove walls that were created around dynamic obstacles and noise on the map.



- Use **Tool type > Floor** when editing floors to fill out the gray areas where there should be floor. When using this tool, you do not affect the walls on the map.



- Use **Tool type > Wall** when editing walls to create solid and even walls.



## Test

The floor plan quality is tested with the localization score—see ["Evaluate localization" on page 321](#).

## Clean up and documentation

Delete floor plans from the floor plan selector that are not to be used. This prevents maps being made in the future based on poor floor plans.

Download the final floor plans so you have a backup and save these on a separate server.

## 4.7.2 Create maps

### Section overview

Combine floor plans to create maps.

- Determine the number of maps you need to cover the operating area.
- Determine the naming scheme for your maps.
- Align floor plans and merge them together to create a straight map.
- Add or remove obstacles to make the maps accurate.
- Document which areas each map covers.

This section explains how to create a new map based on saved floor plans and how to make several maps to cover the operating area to achieve the best planning performance.

For more information about creating floor plans, see ["Create floor plans" on page 294](#).

For more information about adding markers and positions, see ["Create missions and mission groups" on page 347](#).

For more information about using zones, see ["Manage traffic" on page 411](#).

### Plan

Before creating any maps, consider if you need to split the operating area into several smaller maps and connect them with transitions:

- **If the operating area is very large:** If robots take a long time to plan its routes or often reports CPU errors when planning, your map is too large or complex. The exact size where the planning speed is significantly affected depends on the complexity of the map and the number of zones, but as a general rule, avoid making maps larger than 300 × 300 meters (6 000 × 6 000 pixels).

MiR Fleet prevents you from recording or uploading floor plans that exceed 750 m in one direction or have an area above 250 000 m<sup>2</sup>.

- **If the robot must operate on different floors:** You must have a map for each floor. For more information on setting up transitions for ramps, see ["Ramps" on page 329](#). For elevators, you must make a custom elevator control system. You can create transition positions inside the elevator on the map for each floor and set up transitions for each floor shift.
- **If you want to control how robots transition between areas:** Split the areas into smaller maps. When you set up transitions, you choose the exact positions where robots transition to the other map. You can also use map zones to control robot planning and driving behavior— see ["Manage traffic" on page 411](#).

For more information about how to connect maps with transitions, see ["Create maps" on the previous page](#).

When you name the maps, consider the following:

- What different areas are called.
- When to use numeric or alphanumeric numbering.
- Capitalization structure.
- Allowed separators.
- Do not use special characters.

## Create maps

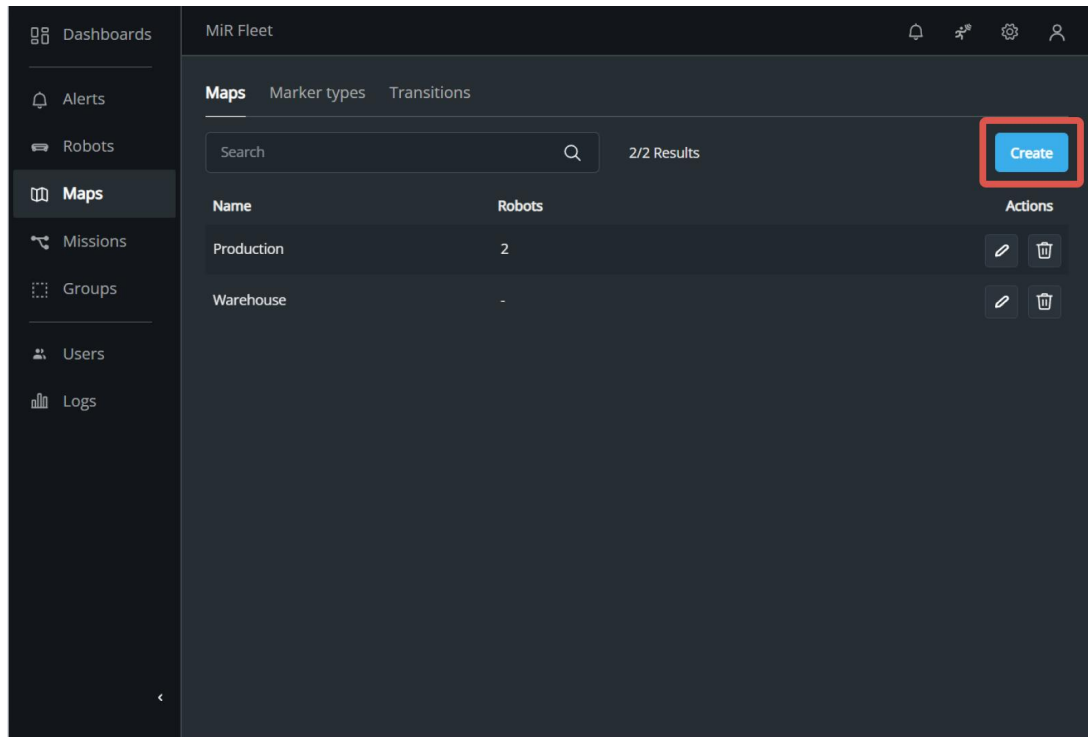
### Combine floor plans to make a map

Once you have recorded or uploaded all of the floor plans of the site create the maps you want your site to consist of.

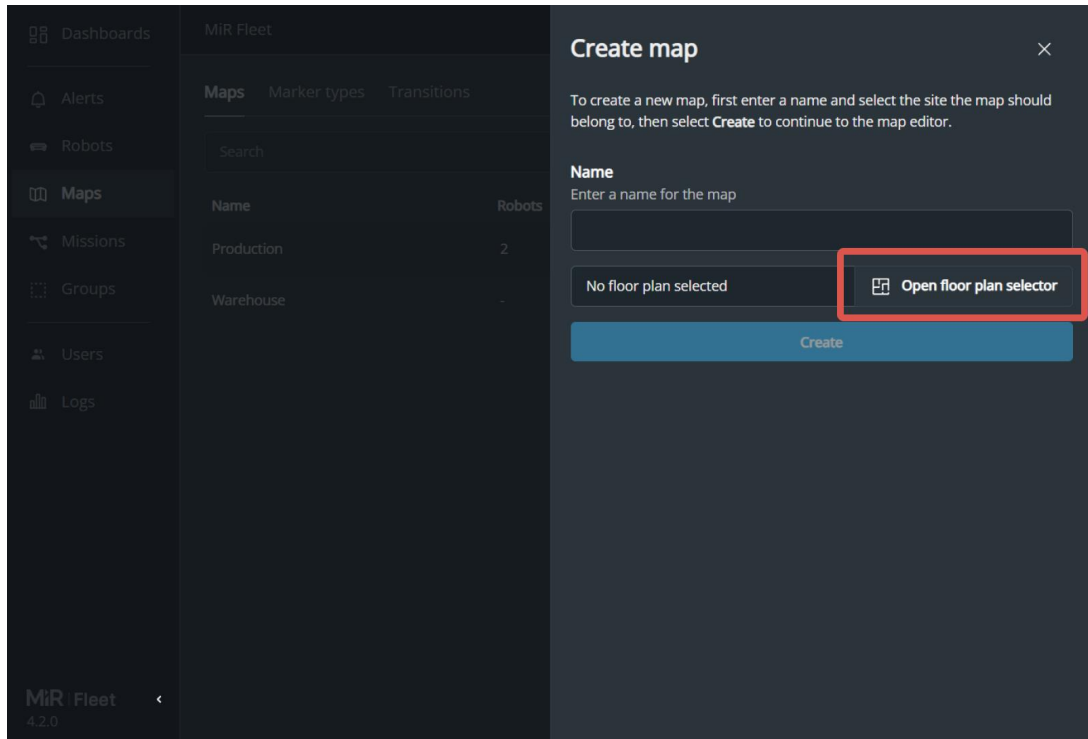
To create a map, follow these steps:



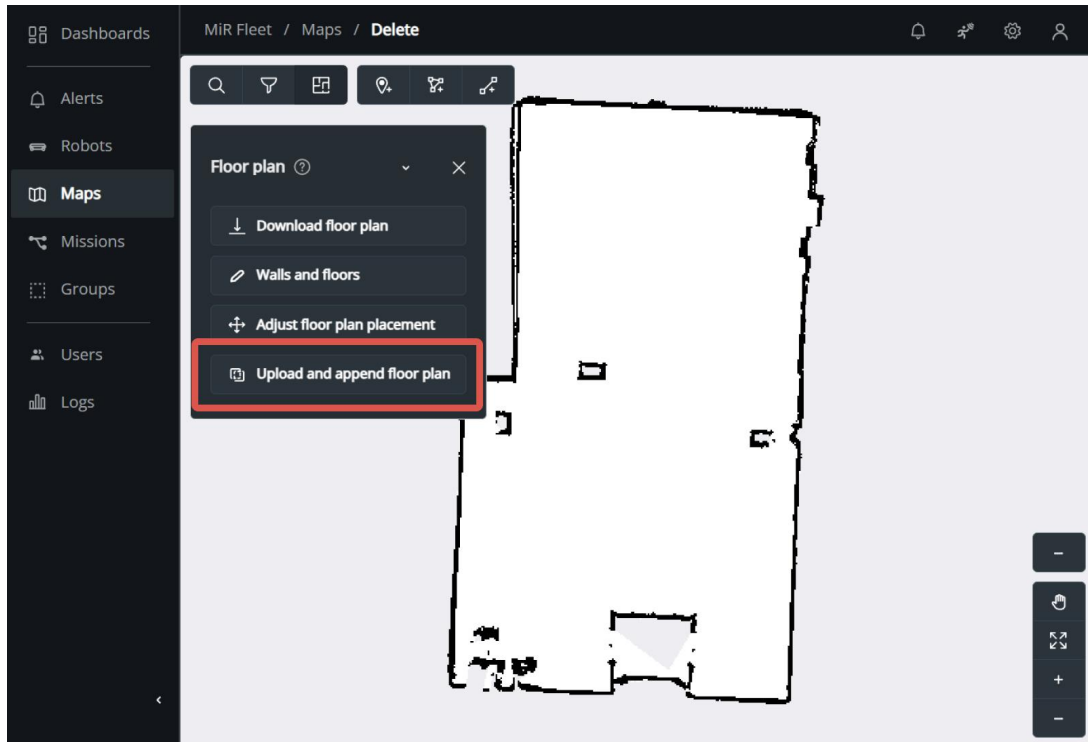
- 1 In the MiR Fleet interface, go to **Maps**, and select **Create**.



- 2 Name the map, and select **Open floor plan selector**, and select the floor plan you want to base your map on.



- 3 If you want to expand the map with more of the saved floor plans, select **Floor plan > Upload and append floor plan**. Select the floor plan you want to add and adjust the floor plan to make it align with the existing floor plan.



- 4 Make any necessary floor and wall corrections, and clean up the floor plan—see ["Optimize floor plan quality" on page 308](#).

### Add unmapped objects

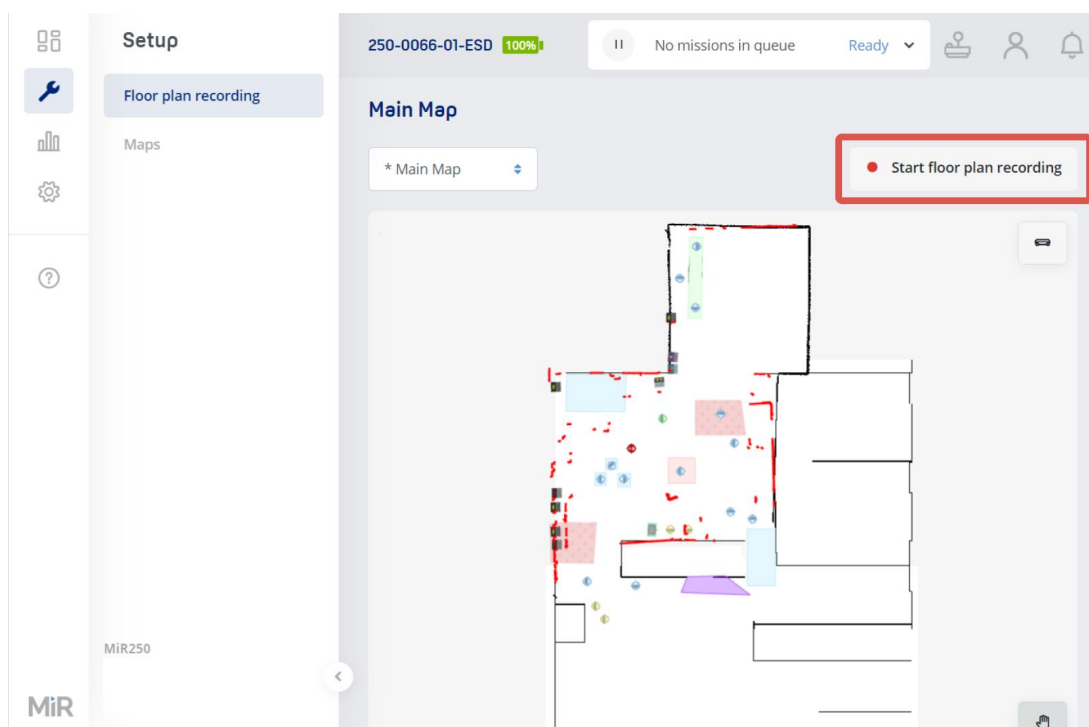
You can add unmapped objects by either recording and appending them, or drawing them based on the robot's current scanner data. Recording and appending lets you add many things at once, whereas drawing them manually can be more time consuming but gives you complete control of what is included.

Only record within a contained area. If you record a large area that overlaps with existing parts of your map, it can be difficult to align them accurately.

### Append recording

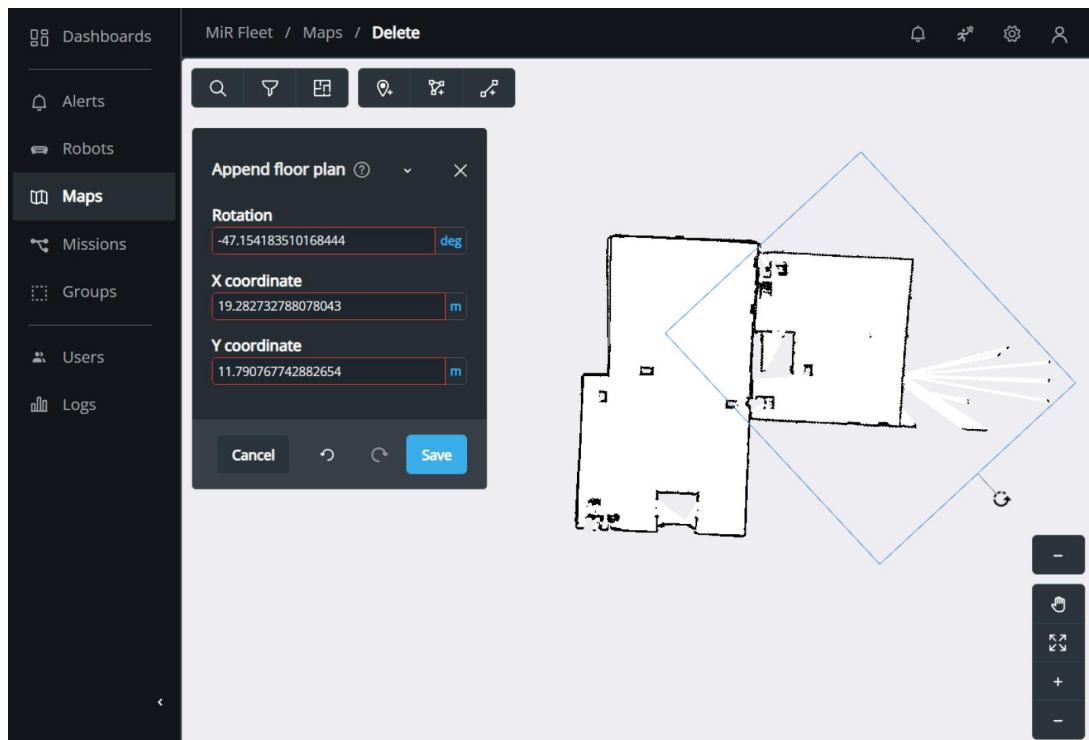
To record and append an unmapped feature, follow these steps:

- 1 Move the selected robot to the features you want to map.
- 2 Open the robot interface on the robot you want to record with.
- 3 Go to **Setup > Floor plan recording** and select **Start recording**.

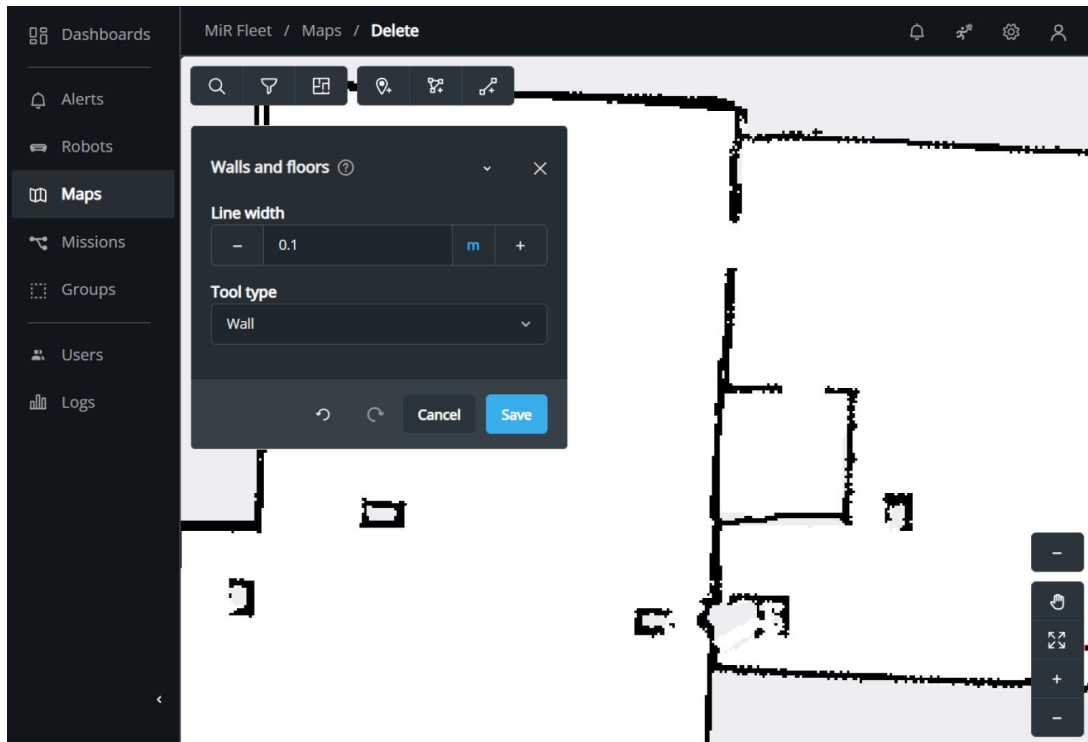


- 4 When you have finished the recording, select **Stop recording**.
- 5 In the MiR Fleet interface, open the map editor for the map you want to add the feature to.

- 6 Select **Floor plan** > **Upload and append floor plan**.
- 7 Select the floor plan recording you just made.
- 8 Position the new floor plan on the map.



- 9 Clean up the floor plan—see "[Optimize floor plan quality](#)" on page 308.

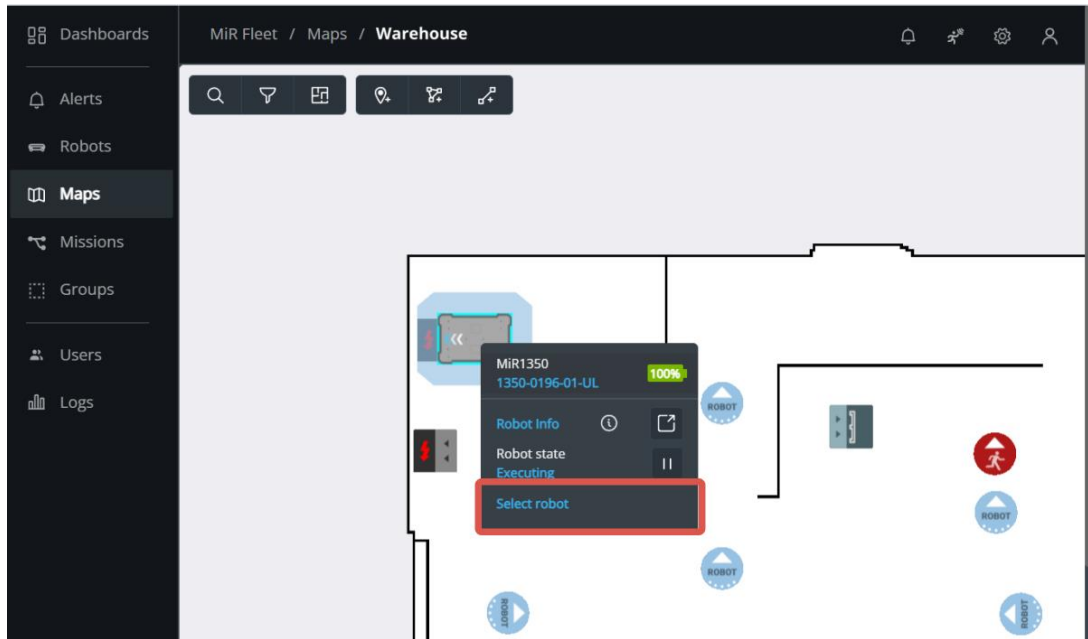


### Draw laser scanner data

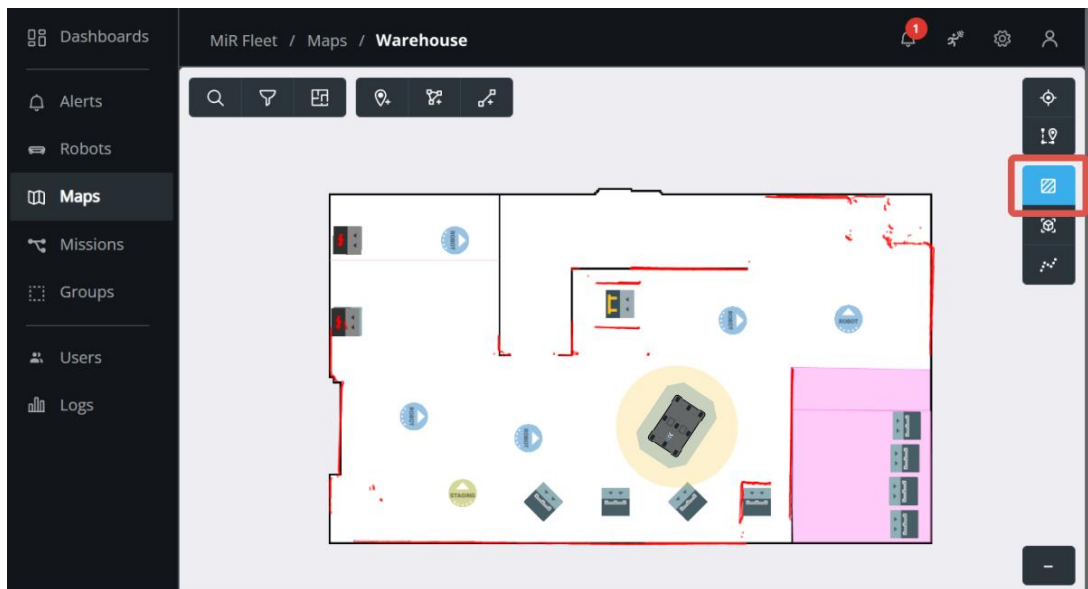
To draw on the floor plan based on scanner data, follow these steps:

- 1 Move the selected robot to the features you want to map.
- 2 In the MiR Fleet interface, go to **Setup > Maps** and select the map you want to adjust.

- 3 Select the robot you moved to the unmapped feature.

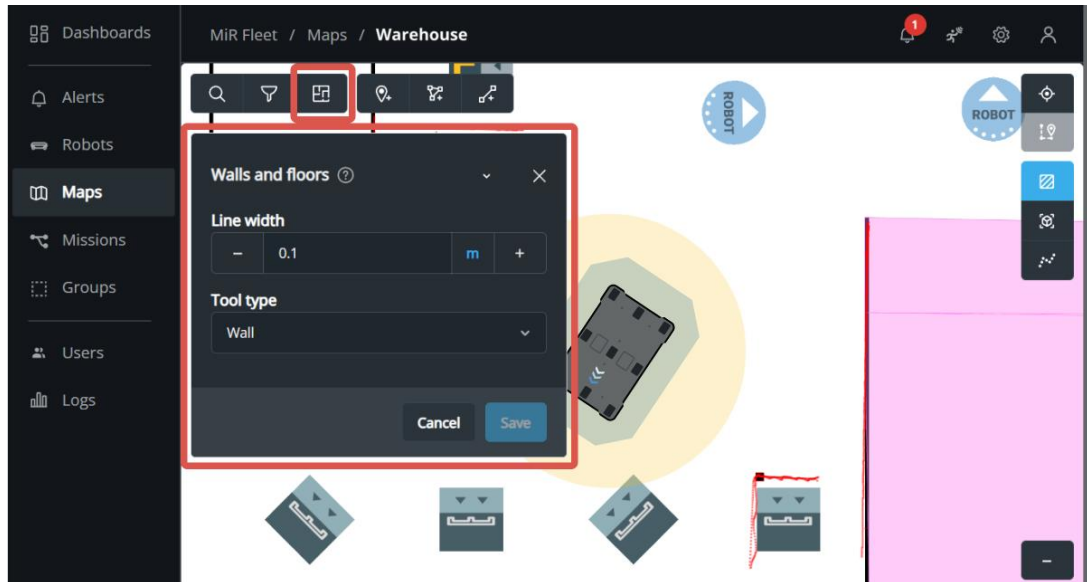


- 4 Enable laser scanner data.



The red dots represent the laser scanner data detected by the selected robot.

- 5 Under **Floor plan > Walls and floors**, select the desired wall drawing settings and trace the laser scanner data with the draw tool.



## Test

The map quality is tested with the localization score—see ["Evaluate localization" on the next page](#).

## Document

Document what each map you have created represents.



### 4.7.3 Evaluate localization

#### Section overview

Ensure robots can localize themselves across the whole operating area.

- Localize all robots on the active map.
- Drive one robot across the whole map.
- Evaluate the localization score.
- Apply map changes if necessary.
- Document the final localization score and evaluate the localization regularly to ensure the map continues to be accurate.

Localization is the process robots use to identify where they are on the map with respect to their surrounding environment. If the robot often places itself inaccurately on the map, it indicates that it has not localized itself well.

#### Localize the robot

When you localize a robot, you provide the robot with its current position on the map. Once it has this input, it can adjust its map position based on the input from the laser scanners, the Inertial measurement unit (IMU) and the wheel rotations from the motor encoders.

To set the initial position and adjust the robot's localization, follow these steps:

- 1 In the robot interface, select **Active** for the map of the area the robot is currently located in. The correct map may already be active, depending on your setup.
- 2 Move the robot manually to a location on the map where it can detect several static obstacles that are also drawn on the map.
- 3 Open the active map.
- 4 Select **Set robot start position**.

- 5 Select the point on the map where the robot is approximately located, then adjust the orientation. The red lines that indicate the laser scanner input should line up approximately with the walls on the map. It does not have to be a perfect match.
- 6 Select **Adjust localization**. The robot makes a small adjustment to its current location so the scanner data lines up better with the walls on the map. You may have to adjust the localization several times. If the localization continues to be inaccurate, move the robot to another location on the map and repeat the process.

### Evaluate localization

MiR robots record localization scores across the map when they are operating. The score indicates how confident the localization algorithm has placed the robot correctly on the map.

To test the localization across the site, have at least two robots perform missions that make them travel across the entire site. While driving, the robots gather localization scores automatically that you can view using MiR Insights. If you do not use MiR Insights, you can assess the localization manually by watching how well the red scanner data aligns with walls on the map.

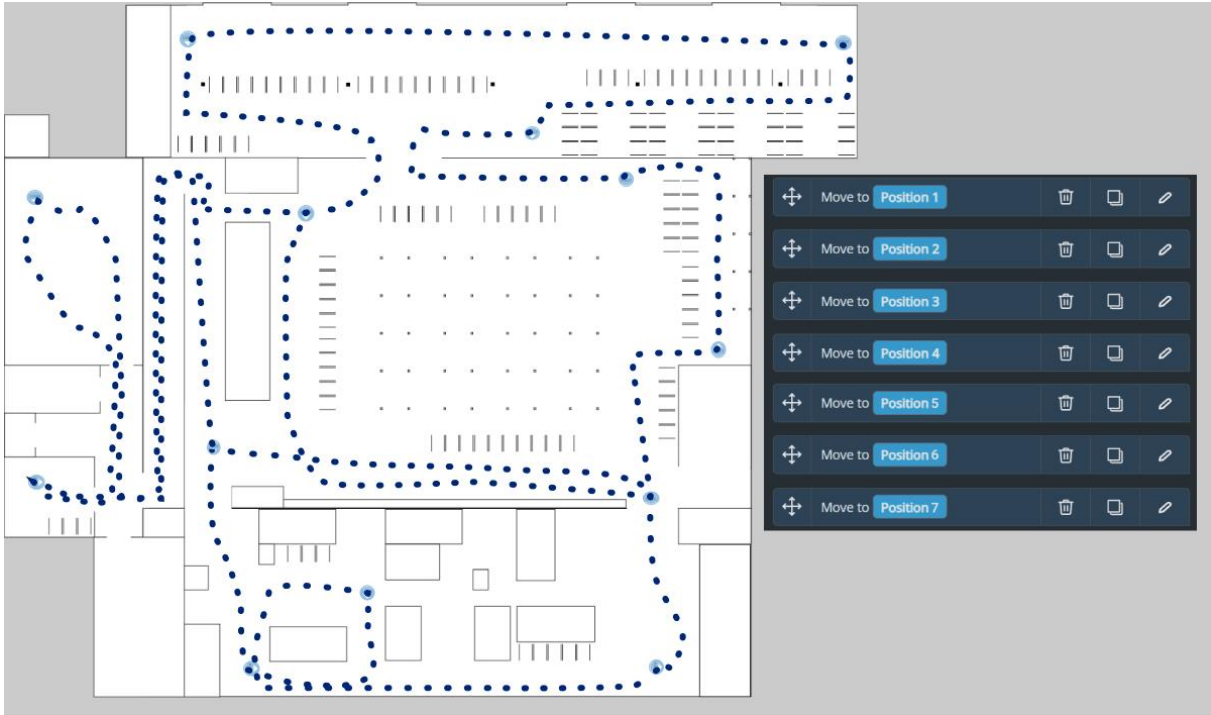
MiR Insights displays the data in a heatmap that enables you to immediately identify any area with poor localization scores.

Using two different robots lets you evaluate if there is a difference between the robots' performance. Differences in performance can be due to calibration, how well-maintained the robot is, the top module, or the robot model.

To test your site, follow these steps:

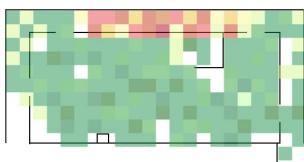
- 1 Place Robot positions on the map so that the robots will drive around the entire site when driving between them.
- 2 Make two missions that use Move actions to make the robots drive between the positions. Make the order that the robot moves between positions different in both missions.

- Using two different robots, run the two missions you created. You can have them run at the same time, but make sure to monitor the robots during the process to make sure they do not block each other. Make the robots execute the mission twice in case there is any difference in the environment depending on the time of day.



In MiR Insights you can view the results across the map immediately under **Heatmaps**.

**Figure 4.6** Heatmap example with poor localization in the hallway



In the heatmap, any areas where the robot does not localize well are marked in red. Poor localization scores are often due to:

- **Not enough unique static landmarks:** This is common in maps with many repetitive rooms or hallways structures, or in maps with very few defining features.
- **Too many dynamic obstacles:** If unmapped obstacles are blocking the robot from detecting any static landmarks on the map, there is not enough data the robot can compare with the map to estimate its current position.
- **Inaccurately mapped areas:** If the scanner data does not match the map, the robot cannot make an accurate comparison.

### Improve localization

For more information about how localization works, see *MiR250, MiR600, and MiR1350 Technical Guide*.

To solve localization issues, consider the following:

- Add landmarks.
- Review map quality.
- Use Planner zones with No-localization zone actions where necessary.
- Use Adjust localization. Only use this action for minor corrections in the localization accuracy and always test the effect of the action.

### Document

Document the localization heatmaps. Check the localization scores regularly to verify the quality of the map.

## 4.7.4 Create transitions

### Section overview

Create transitions between maps.

- Create transition positions where maps overlap.
- Create transition missions for each map transition.
- Define transitions.
- Test that a robot can travel to and from each map using the appropriate transition.
- Document the transitions, including the relevant map locations, positions, and missions.

This section explains how to create, organize and test transitions.

### Plan

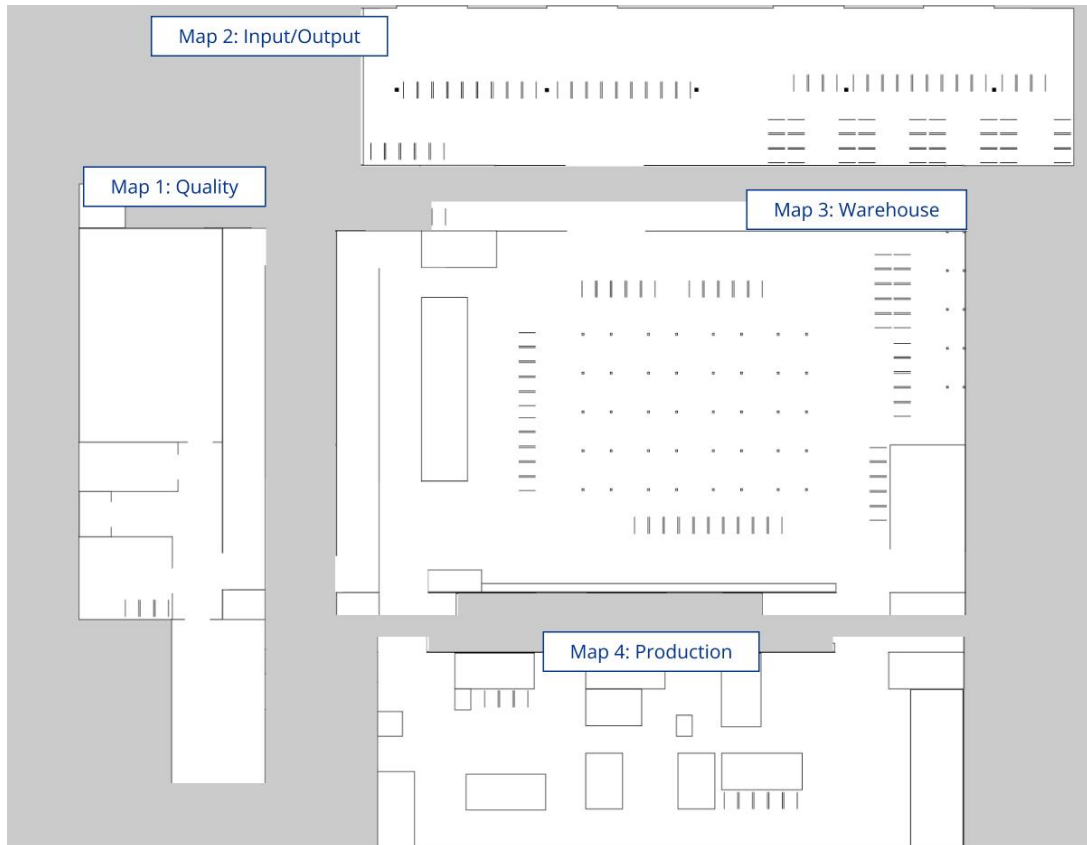
For each map you want to connect, consider the following:

- Robots must drive to the transition position to change maps, so often you want to split maps where there is a single entry location.
- You can create multiple transitions between the same map if necessary, but each transition takes time to set up and maintain.
- You must create transitions for both directions. These can be at the same location or different locations.
- Transitions are often also used when changing floors with ramps or elevators.

### Connect maps with transitions

Make transitions to connect the map by following these steps:

- 1 For all maps, make sure there are overlapping areas where you want robots to transition between maps.

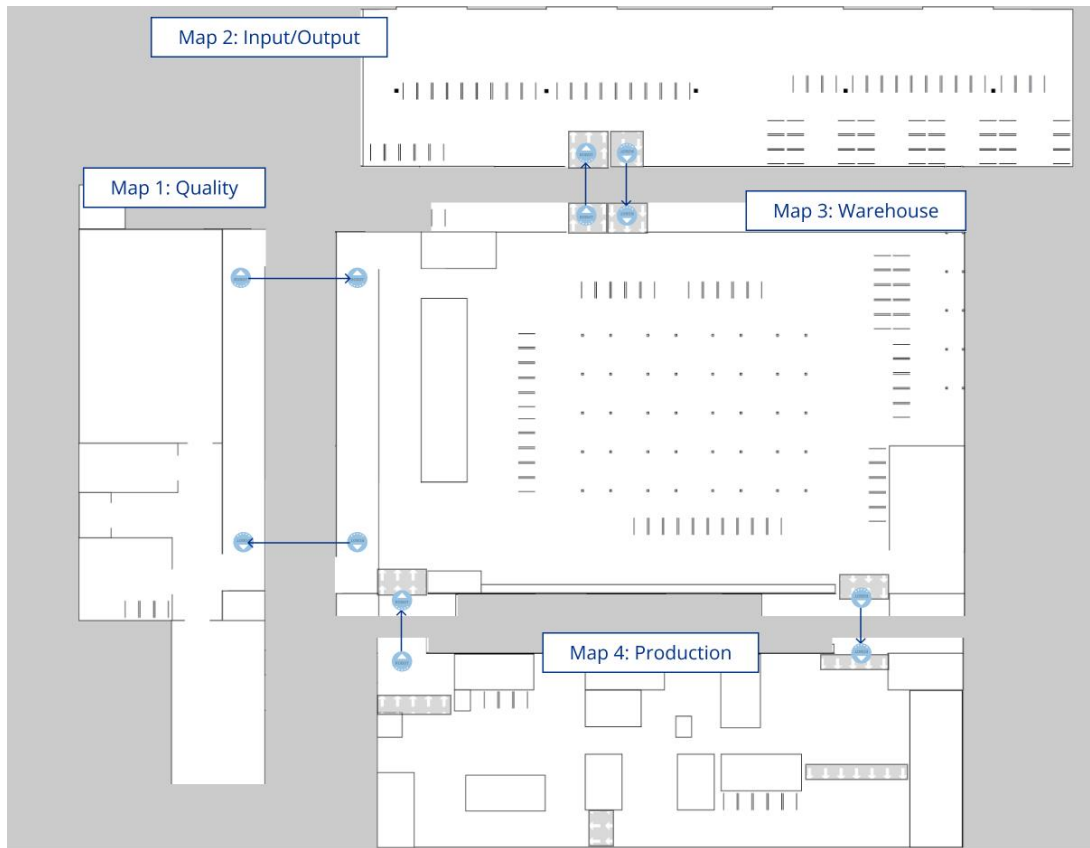


- 2 Create pairs of positions for each map transition. Make separate position pairs for each transition direction.

To make transition positions, follow these steps for each transition:

- a Localize the robot—see ["Evaluate localization" on page 321](#).
- b Move your robot to an overlapping point between two maps where you want to create a transition.
- c Position the robot so that it faces the entry direction of the new area.  
Once satisfied, do not move the robot again until you have created transition positions on both maps.
- d In the robot interface, on the active map, create a Robot position using the robot's current position—see ["Create positions and markers" on page 337](#).  
Use a position name that clarifies which transition the position is part of.
- e Activate the second map for the transition.
- f Correctly localize the robot without moving it.
- g On the active map, create a Robot position using the robot's current position.  
Use a position name that clarifies which transition the position is part of.

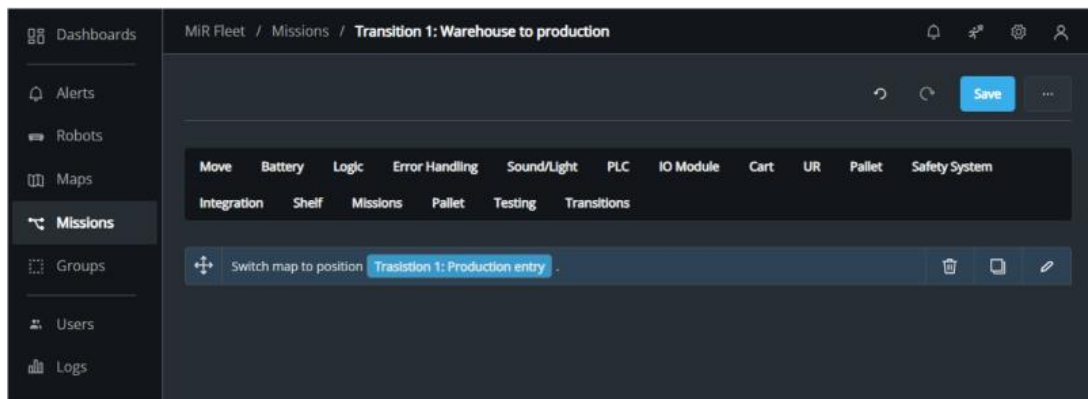
The transitions must point in the same direction as the traffic flow.





- 3 Create transition missions for each transition. Create a mission group for transitions to categorize the missions and name each mission to indicate which transition it is used for.

A transition mission must consist of a Switch map action that specifies which map the robot must switch to and which position it starts from on the map after switching.



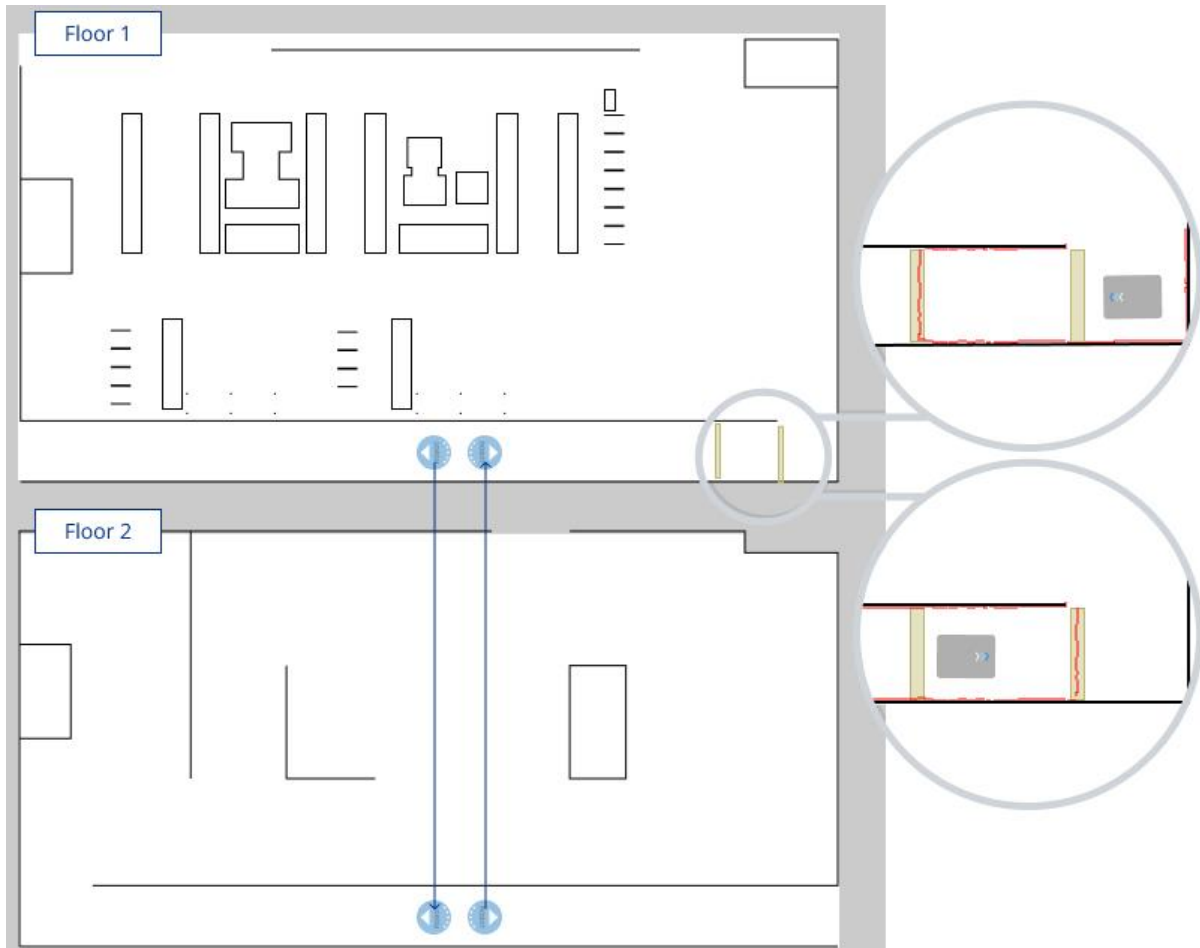
- 4 Under **Transitions**, create a transition for each of the entry and exit transitions for all maps—see "[Transitions](#)" on page 107.

### Ramps

If you have a ramp connecting two floors, you may have to add Access zones to make the robot plan through the ramp:

- 1 Drive a robot to the bottom of the ramp just before driving onto the ramp.
- 2 If the robot detects the ramp (red dots are shown in the map interface further up the ramp), add an Access zone on top of the red line.
- 3 Drive the robot up the ramp.
- 4 Stop the robot once all wheels are on the ramp.

- 5 If the robot detects the floor at the bottom of the ramp (red dots are shown in the map interface at the bottom of the ramp), add an Access zone on top of the red line.



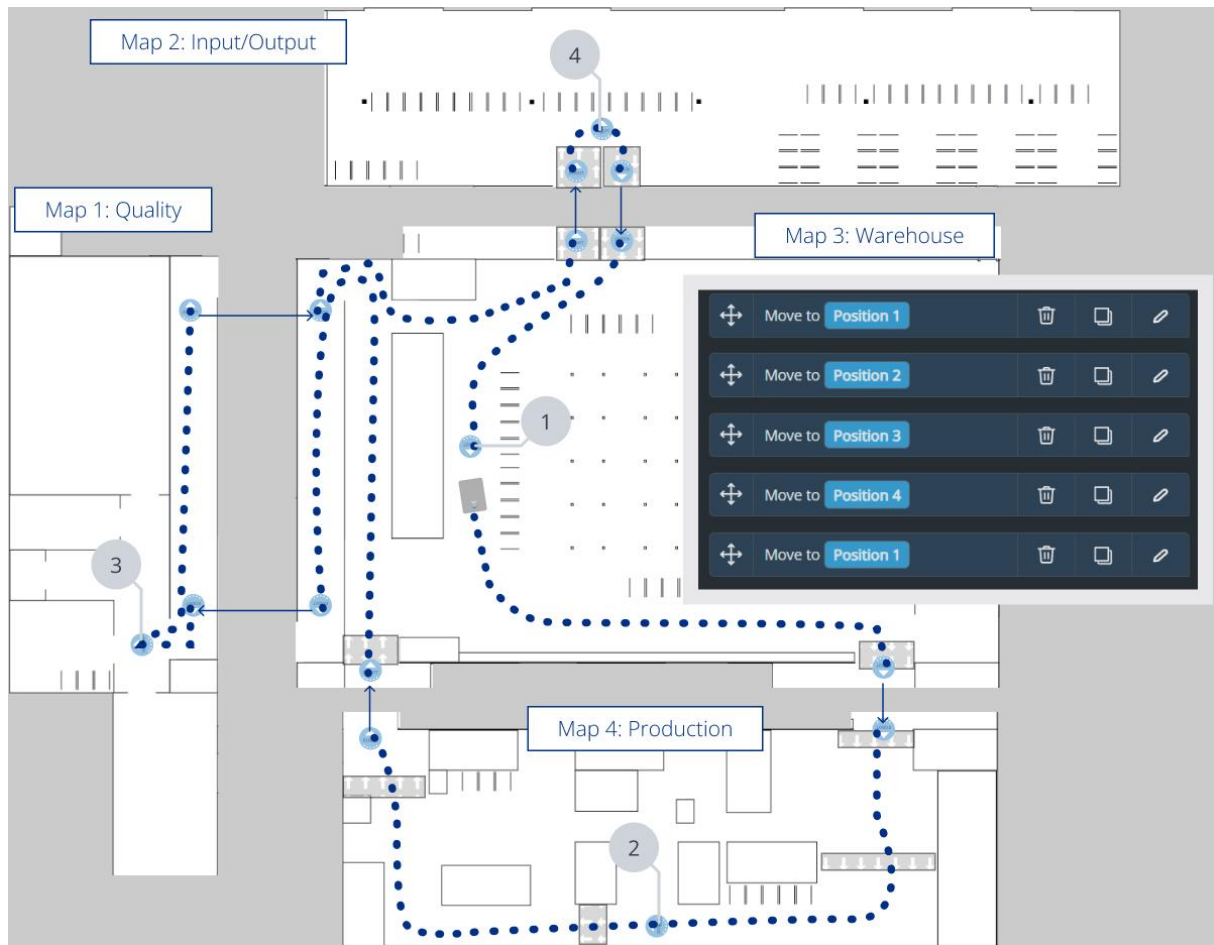
### Elevators

MiR Fleet Enterprise does not support elevators. Use MiR Fleet Integration API to define your own elevator control—see "[MiR Fleet Integration API](#)" on page 484.

### Test

Create a position on all maps that are connected with transitions.

Create a mission that sends the robot to each of the positions and then return to its start position. Running this mission tests that robots can enter and exit each map automatically.



## Document

Document all transitions you created making it clear which two Robot positions and transition mission are connected. Identify where the positions are on each map and which positions are connected with a transition.

## 4.8 Program robot tasks

### After reading this chapter, you can:

Create missions that make robots perform the identified robot tasks.

**Define the robot tasks and break them down into mission building blocks and categories.** See ["Define tasks" on the next page](#).

- List all robot tasks.
- Define the actions the robots must perform to complete the tasks.
- Identify actions that are repeated across tasks.
- Plan how you want to organize missions into mission groups.
- Agree on a naming scheme.

**Create markers and positions at all points of interest for the robot tasks.** See ["Create positions and markers" on page 337](#).

- Determine which type of position or marker is best suited for each point.
- Agree on a naming scheme.
- Add positions and markers.
- Test how the robot docks to each marker and apply corrections.

**Create mission groups and missions for the robot tasks.** See ["Create missions and mission groups" on page 347](#).

- Create the mission groups you planned.
- Create reusable missions for all actions that are repeated across robot tasks.
- Create missions for the robot tasks.
- Test the missions and apply corrections.
- Document the mission setup.

**Make missions more robust.** See ["Make missions robust" on page 377](#).

- Test missions where you expose the robot to unexpected events you want it to handle autonomously.
- Use Try/Catch actions to define alternative actions in missions when exceptions or

unexpected events occur.

- Iterate between testing and modifying missions.
- Update the mission documentation if necessary.

**Set up system for scheduling missions.** See "[Schedule missions](#)" on page 398.

- Determine triggers for each mission.
- Create or modify any external system to schedule missions automatically.
- Create or modify dashboards to help users schedule missions manually.
- Document the chosen triggers for each mission and responsibilities for ensuring each mission runs.

## 4.8.1 Define tasks

### Section overview

Define the robot tasks and break them down into mission building blocks and categories.

- List all robot tasks.
- Define the actions the robots must perform to complete the tasks.
- Identify actions that are repeated across tasks.
- Plan how you want to organize missions into mission groups.
- Agree on a naming scheme.

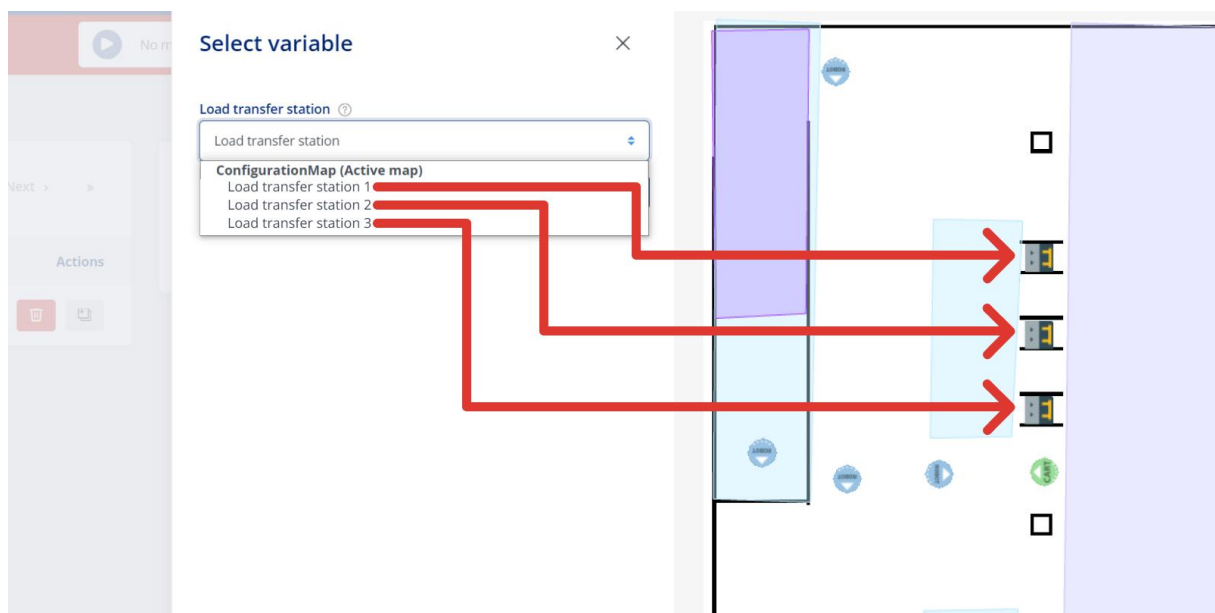
### Identify reusable missions

List all of the robot tasks you have identified, and break them down into smaller tasks based on the actions in the MiR interface.

After creating the overview, identify actions that are repeated across tasks. When creating missions, start by creating generic and reusable missions for all repeated actions. Creating smaller mission for sub-tasks you can nest into larger missions have the following benefits:

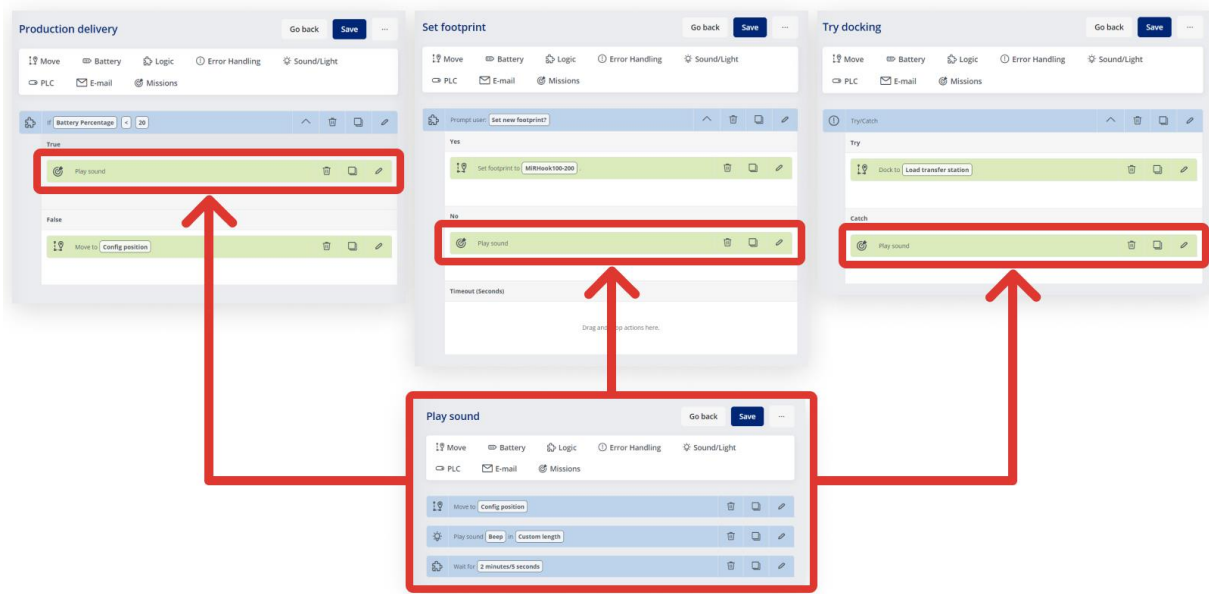
- You spend less time defining the same sequence of actions for the same sub-task in all missions that use it.
- If the sub-task changes and you need to update all mission with the sub-task, you only need to update a single mission for the sub-task.
- Makes missions easier to read, provided the missions for the sub-task is well named.

Use arguments—["Arguments" on page 81](#)—in the missions for sub-tasks to let you select specific map elements each time the missions is used in other missions.



Examples of reusable missions are:

- Docking to load transfer stations
- Undocking from load transfer stations
- Travel behavior through certain areas
- Top module or connected I/O module activation
- Troubleshooting sequence



You can nest reusable missions inside other reusable missions. If you do so, keep the following in mind:

- Keep documentation of what each mission does, which mission it is nested in, and which missions are nested in it. If you make change to the missions, use your documentation to verify that the change is applicable for all connected missions.
- Nesting beyond three levels is highly unrecommended.
- Avoid using missions recursively. It can be difficult to predict how many times the mission will be replayed.
- Scope the purpose of a mission and keep each mission focused on one purpose.
- Do not let a mission run for too long without ending. This can make the robot interface respond very slowly or MiR Fleet fail to send MiR robots to Auto charge correctly.

## Naming scheme

Determine a suitable and consistent naming scheme to make site elements easy to identify.

Names for missions, positions, arguments, and other elements are often based on the following three aspects:

- **Robot and top module applications:** Identifies which robot the element is intended for
- **Locations:** Identifies which location the element is relevant for.
- **Tasks:** Identifies which tasks the element is intended for.

To avoid variations, agree on:

- Terms to use within the categories
- The order of appearance in names
- When to use numeric or alphanumeric numbering.
- Capitalization structure
- Allowed separators
- Do not use special characters

### Categorize missions

Use mission groups to categorize missions.

Mission groups are also used in groups—see ["Create groups" on page 288](#)—to specify which robot can be assigned certain missions.

Use specific mission group categories to keep the mission groups small and modular when adding them to groups. Mission groups are often based on the top module or purpose of missions.

### Inputs and triggers

For each mission, determine what information the mission is dependent on and what should trigger the mission. Use this to set up a scheduling system—see ["Schedule missions" on page 398](#).

### Document

Document all schemas and naming schemes you use to make it easy to add new elements according to the same logic.

After creating an overview of all the expected missions, keep the overview up to date with any changes that are applied when the missions are made and the site is tested.



## 4.8.2 Create positions and markers

### Section overview

Create markers and positions at all points of interest for the robot tasks.

- Determine which type of position or marker is best suited for each point.
- Agree on a naming scheme.
- Add positions and markers.
- Test how the robot docks to each marker and apply corrections.

### Plan

For each robot task, identify which locations the robot needs to stop to complete the task and determine which position or marker type to use:

- For shelves, carts, and pallets, there are specific marker and position types you must use—see ["Markers" on page 52](#) and ["Positions" on page 50](#).
- For custom load transfer station or other purposes, you must evaluate which of the standard markers and positions you want to use:
  - Use markers when you prefer the robot spends time positioning itself accurately relative to a physical entity.
  - Use positions when you prefer the robot quickly positions itself approximately at the correct position.

For the exact positional accuracy the different marker types and positions provide, see the latest specifications of your product on [MiR Support Portal](#).

Follow these best practices:

- When you create the physical marker, meet the marker design requirements under ["Markers" on page 52](#).
- When you create markers and positions, follow the naming scheme—see ["Naming scheme" on page 335](#).
- Use specific pickup locations on the map instead of using pickup from current location.

- Ensure that there is enough space next to pick up and drop off location for the robot to maneuver.
- For robot with MiR hook top modules, Use drive through drop-off areas instead of reverse drop-off to avoid space and cycle time issues.

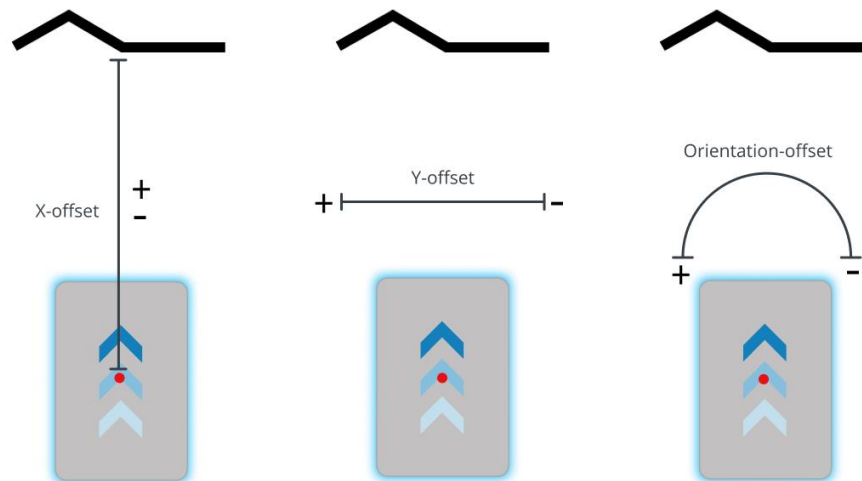
### Create markers and positions

To create a position or marker, follow these steps:

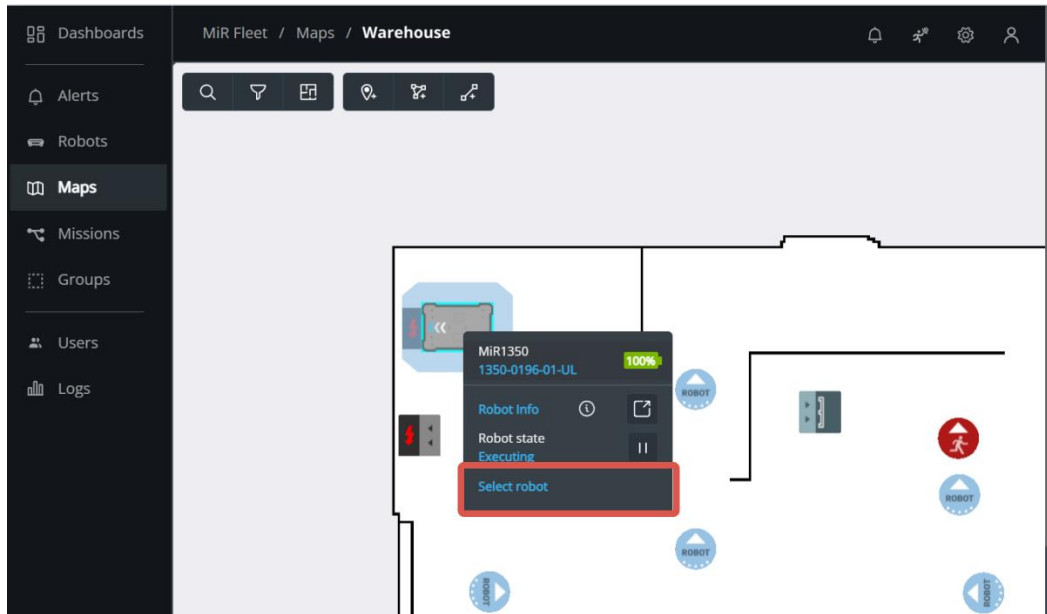
- 1 In the MiR Fleet interface, go to **Maps** and open the map you want to add a marker or position to.
- 2 Select **Add position**, and select the position or marker type you want to create.

3 You can use the following ways to position the element on the map:

- Drag it across the map to position it approximately where you want it.
- Use the rotate tool to orient it.
- Use the **X**, **Y**, and **Orientation** fields to enter an exact value.



- If you select a robot on the map before creating the marker or position, you can use the robot to place positions or detect markers more accurately:



- Select **Use current position** to make the robot create a position where the robot currently is
- Select **Detect marker** to make the robot create a marker based on its current scanner data. The robot automatically sets the offset values of the marker so they correspond to the robot's current position.
  - If the robot cannot detect the marker, verify that the robot is correctly positioned—see "[Markers](#)" on page 52—and that the laser scanners can detect the marker in the active map by checking that red lines are displayed on the map where the marker is.
  - If you are trying to make the robot detect an L-marker but it keeps detecting other objects with a 90° angle instead, shield the objects it is not supposed to detect with a flat plate.
  - If you want to change how the robot positions itself relative to the marker, change the docking offsets. To make the robot dock straight to the marker, set the orientation offset to 0. If you want the robot to reverse into the marker, set the orientation offset to 180°. Robots are more accurate when docking straight to a marker than at an angle.

### Marker detection issues

If a robot fails to detect or dock to a marker, check if the robot's laser scanners are able to detect the marker from the marker Entry position. To do this, send the robot to the marker's Entry position, and view the active map in the robot's interface. The red dots in the map represent the laser scanner data.

If the scanner data does not match with the marker:

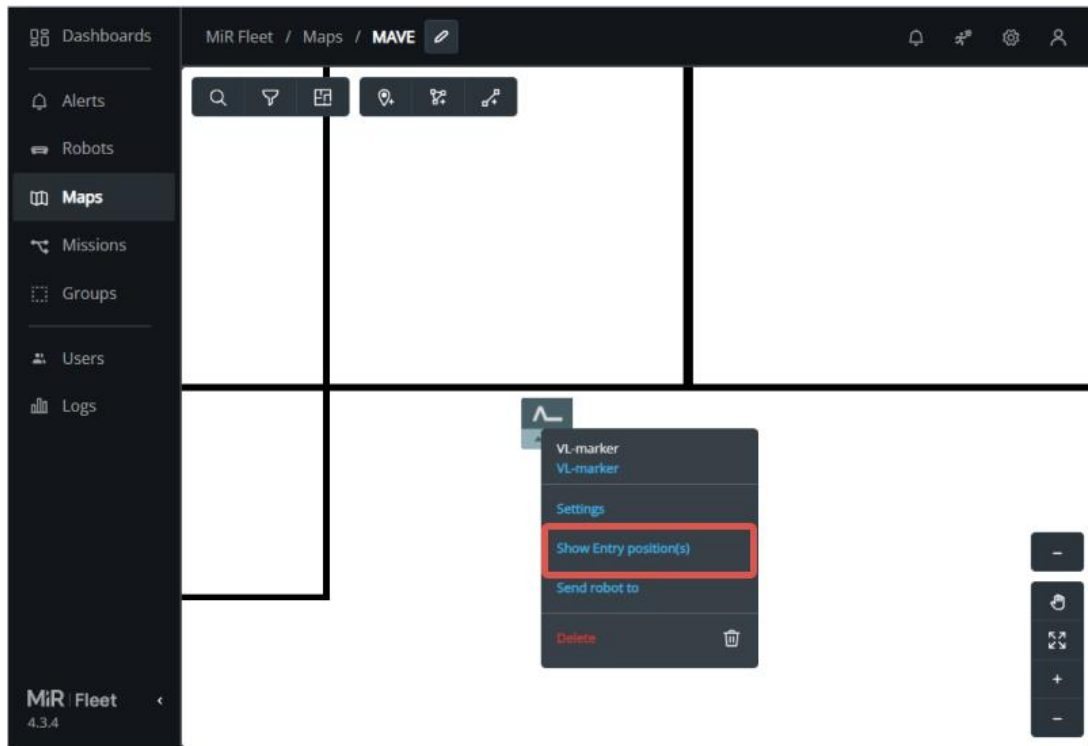
- Verify that there are no obstacles or light interference between the marker and the robot. See ["Adjust Entry position" below](#) to modify the Entry position.
- Verify that the marker is designed according to the specifications in ["Markers" on page 52](#).

### Adjust Entry position

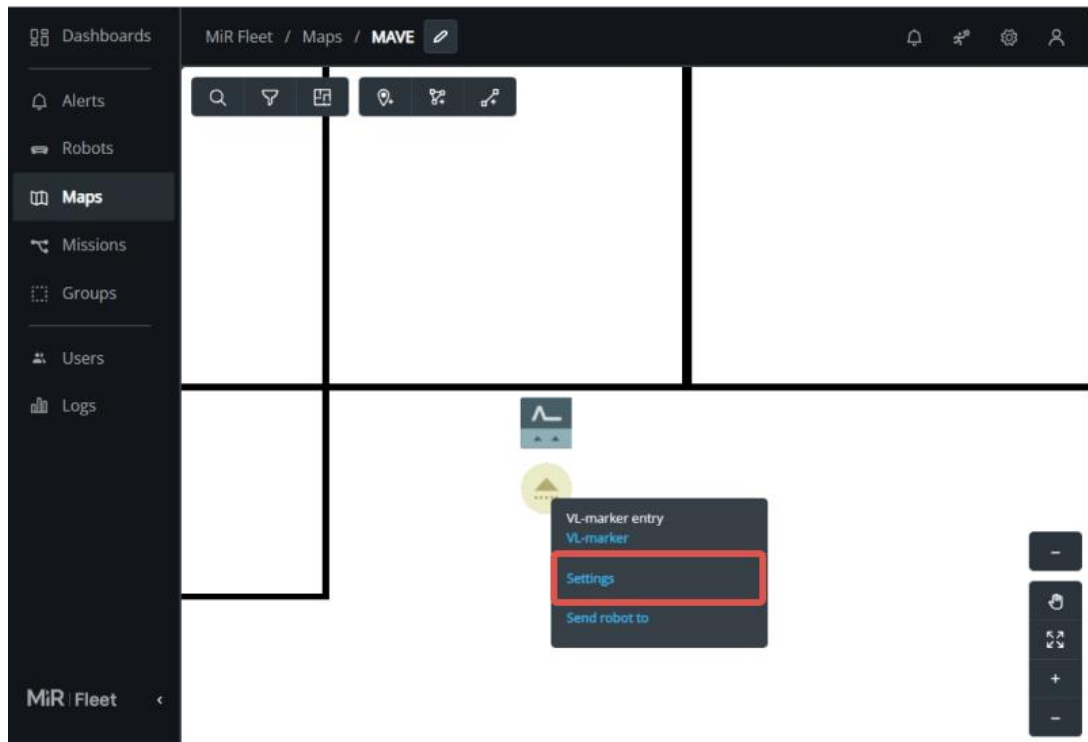
To adjust the Entry position of a marker or position, follow these steps:

- 1 Open the map editor for the map with the position or marker you want to modify.
- 2 Select the position or marker on the map.

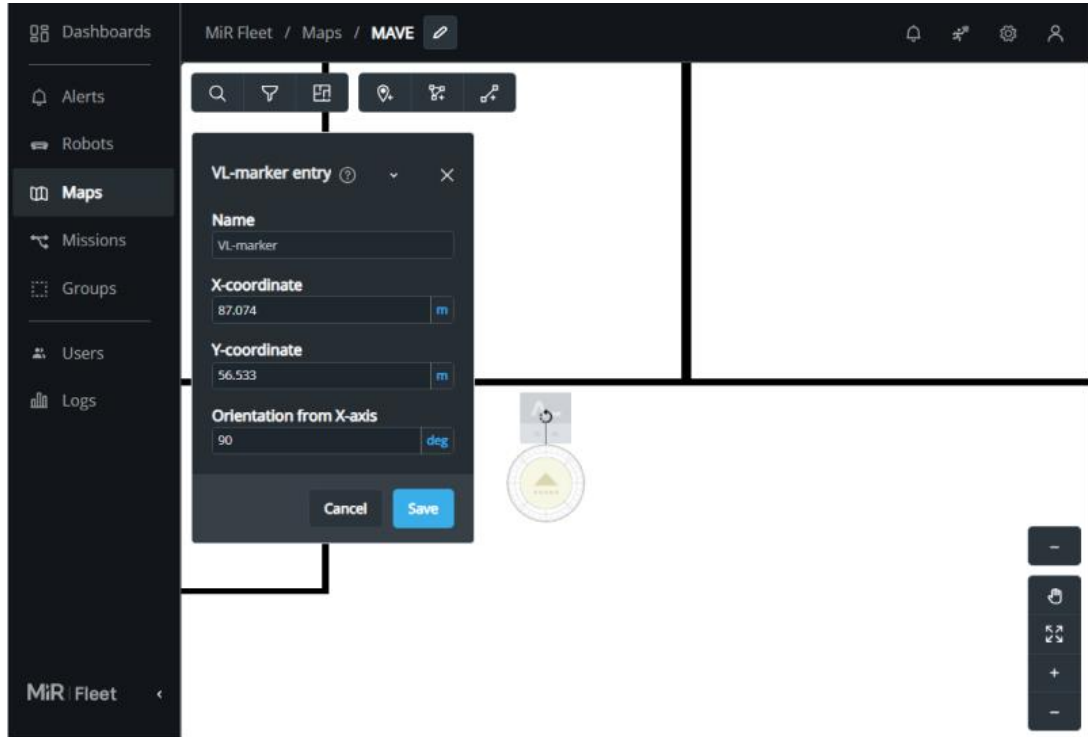
- 3 Select **Show Entry position(s)**. The Entry positions for the selected element are now displayed on the map.



- 4 Select the Entry position you want to modify, and select **Settings**.



- 5 Modify the location of the Entry position. The X and Y coordinates are relative to the map coordinates.

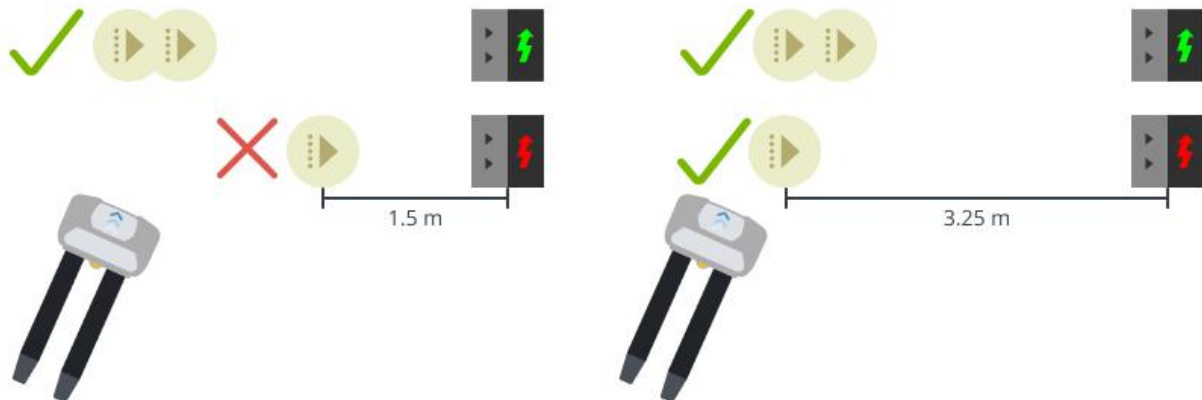




### Update Entry positions for MiR1200 Pallet Jack

For all markers except Charger 48V 105A, you must adjust the Entry position for MiR1200 Pallet Jack to be able to use the marker.

Entry positions must be moved so they are 3.25 m offset from the marker.



You may have to adjust the docking offsets for pallet jack robots to ensure they dock as intended—see ["Marker accuracy across all robots"](#) below.

### Marker accuracy across all robots

If you have more than one robot, you may experience that robots do not all dock to markers with the same accuracy.

Robots will dock with a high precision, meaning that the same robot will dock to almost the same position to the same marker every time. If you want to increase precision, ensure that you are using Slow docking—see ["Markers"](#) on page 52.

If you want all robots to dock to all markers with the same accuracy, use the docking offsets to adjust the robot's final docking positions. There are two sets of docking offsets you can modify:

- The marker's docking offsets. Adjust these to modify how all robots dock to a specific marker.
- The robot's global docking offsets. Adjust these to modify how a specific robot docks to all markers of a specific type.

If different groups of robots should dock to the same marker differently, create multiple markers on the map for the same physical entity. Adjust the marker offsets of each marker to define how you want each group to dock to the marker. Make sure to use the correct marker for each robot when you send it to dock.

### Increase shelf accuracy

MiR recommends that you use markers to increase the accuracy when the robot places shelves on Shelf positions.

For all shelf lifting robots, MiR currently only recommends using V and VL-markers with Shelf positions. The robot will not be able to detect Bar-markers and L-markers reliably while carrying a shelf because the shelf legs reduce the laser scanners' field of view.

To set up a Shelf position with a marker, you must:

- Create a physical marker and define it on a map.
- Create a Shelf position in front of the physical marker.
- Make the X and Y coordinates of the Entry position to the Shelf position the same as the Entry position to the marker.
- In missions, make the robot dock to the marker when placing the shelf, and make the robot dock to the Shelf position when picking up the shelf.

### Test

Test that robots can dock to all markers:

- If you have a line of positions or markers, test these while other robots are in the adjacent positions.
- If you have heavily trafficked areas close to markers, test these under the worst traffic conditions.

### Document

Keep an overview of where all markers and positions are on the map, including their name and purpose.

### 4.8.3 Create missions and mission groups

#### Section overview

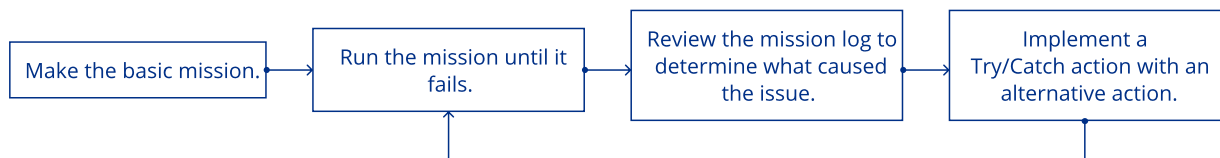
Create mission groups and missions for the robot tasks.

- Create the mission groups you planned.
- Create reusable missions for all actions that are repeated across robot tasks.
- Create missions for the robot tasks.
- Test the missions and apply corrections.
- Document the mission setup.

All tasks you want robots to execute are defined in missions. When you make missions, we recommend this basic process:

- 1 Create building block missions that define basic actions.
- 2 Create missions that just execute the necessary task.
- 3 Test the mission. Also test for relevant unexpected events.
- 4 Apply corrections and make the mission robust against unexpected events.
- 5 Repeat steps 4 and 5 until you are satisfied.

The following sections go more into depth.

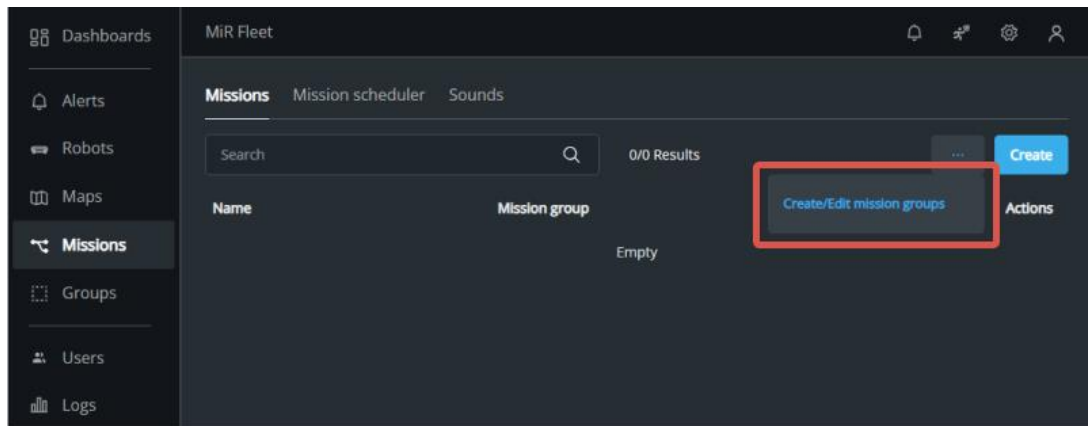


When you create missions and mission groups, follow the "[Naming scheme](#)" on page 335.

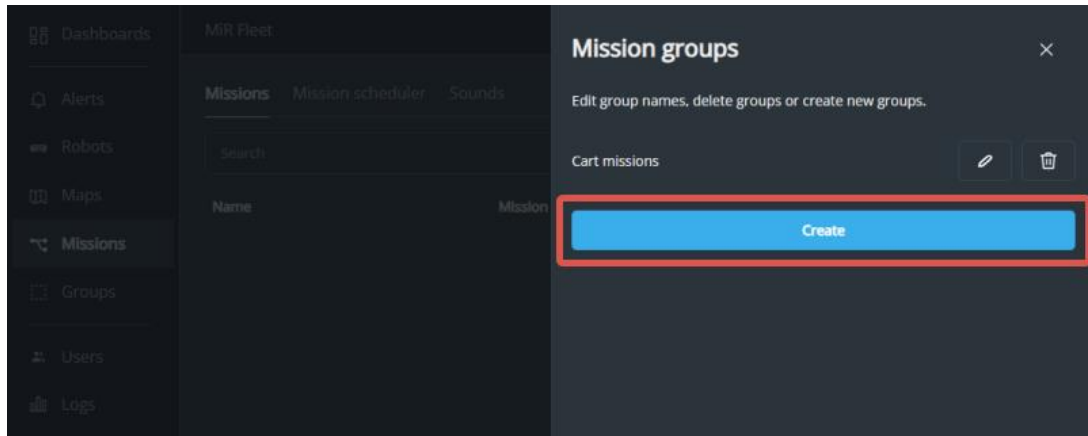
## Create a mission group

To create a mission group, follow these steps:

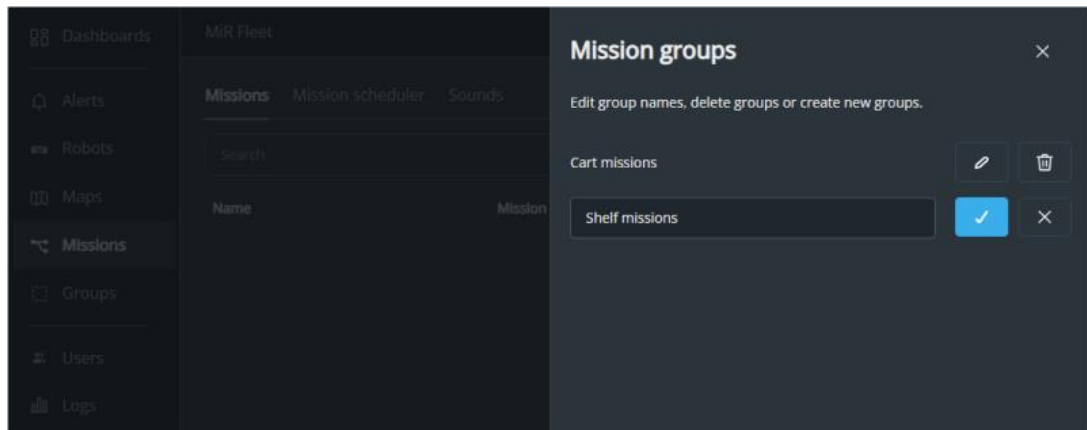
- 1 In the MiR Fleet interface, under **Missions**, select the three dots, and select **Create/Edit mission groups**.



- 2 Select **Create**.



- 3 Enter a name for the mission group, and confirm your choice.

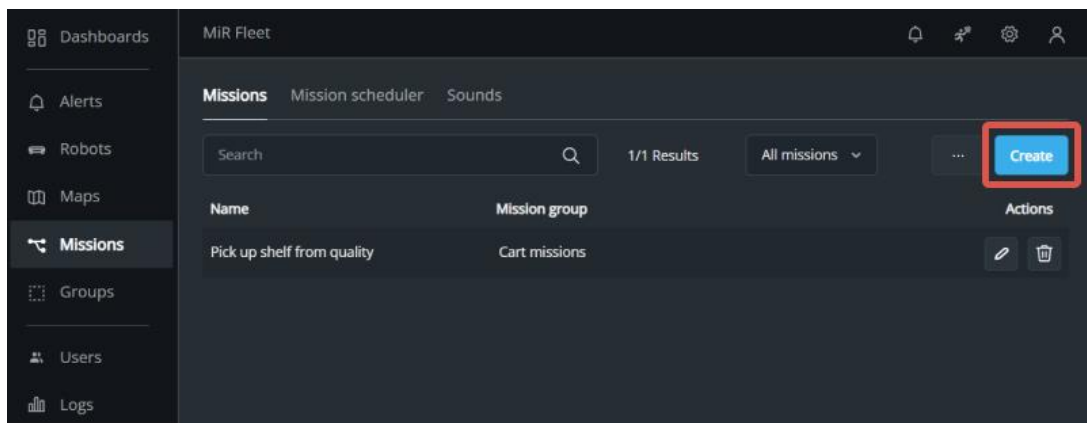


When you create or edit missions you can now assign them to the mission group.

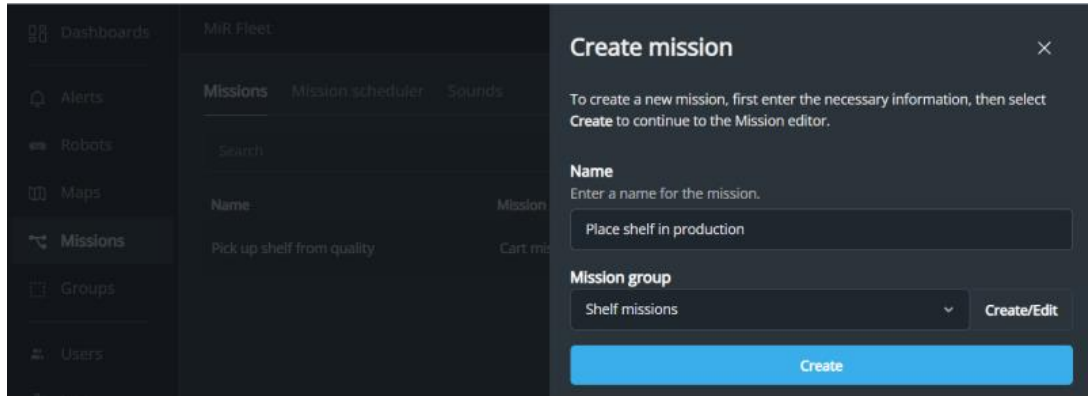
## Create a mission

To create a mission, follow these steps:

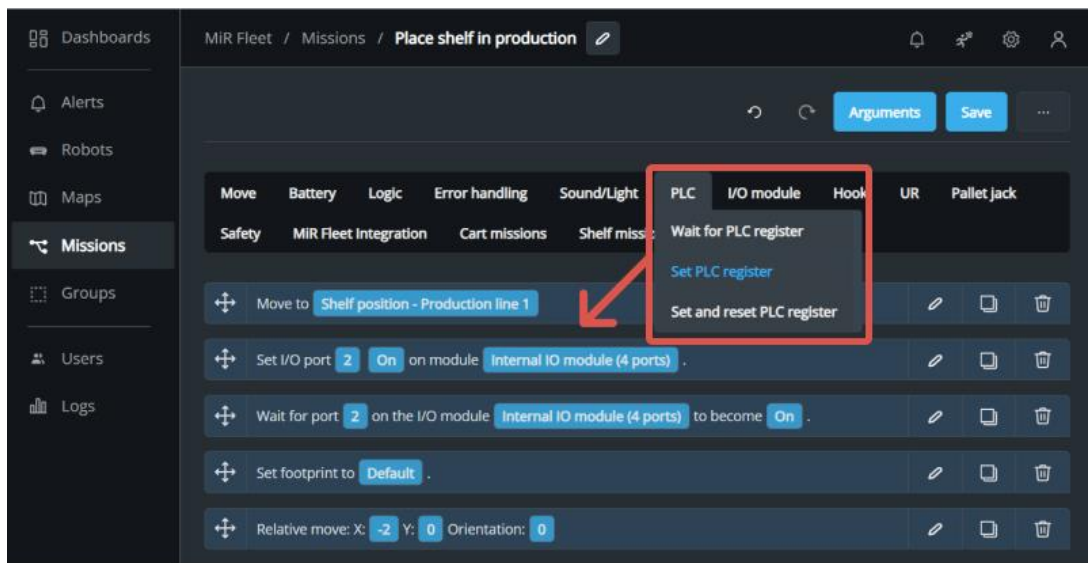
- 1 Plan your mission, and ensure you have an overview of any existing missions you want to nest into the mission—see ["Define tasks" on page 333](#).
- 2 In the MiR Fleet interface under **Missions**, select **Create**.




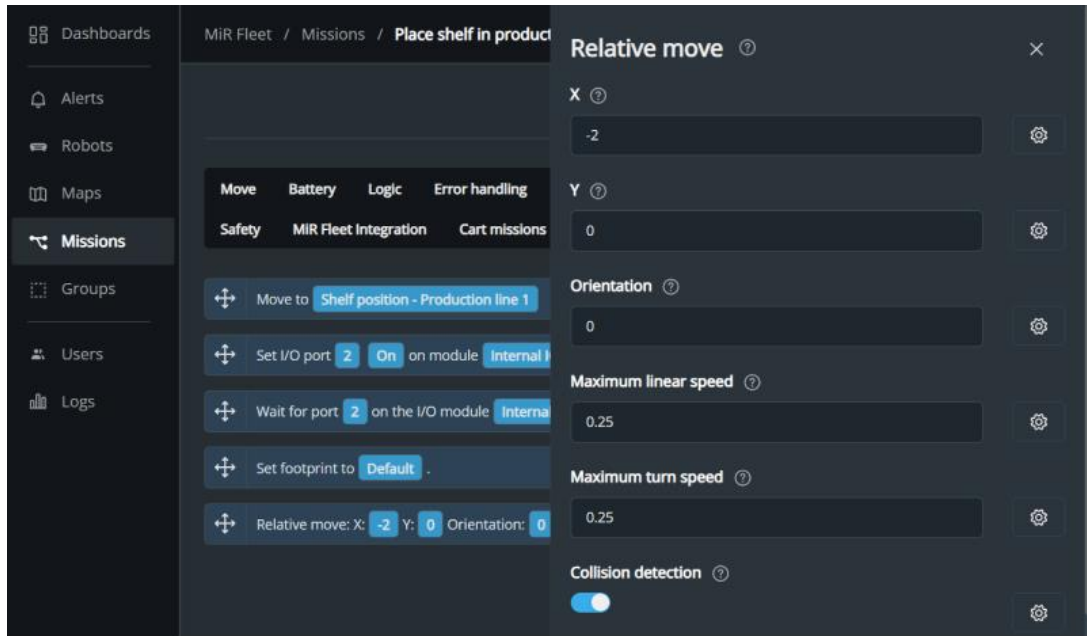
- 3 Name your mission, and assign it to a mission group. Make sure you apply any agreed naming schemes or organizational structures.



- 4 Add actions to your mission by selecting them from the action menus, or nest missions by selecting existing mission from the mission group menus. You can drag and drop actions and missions to reorder them and nest them into sections of nestable actions.



- 5 Edit action parameters and mission arguments by selecting **Edit**  to the right of the action line.



- 6 Save your mission, and make sure to test the mission immediately—see ["Test" on the next page](#).

### Best practices

- Missions where you loop an If or While action without any other actions can slow down the system. Always add a Wait action to avoid robot continuously evaluating a conditional statement.
- Avoid nesting more than three layers of missions.
- Save regularly.
- Make copies of the template missions so you can apply modifications.
- Use PLC registers to track the mission progress—see ["Make missions robust" on page 377](#).
- Add Try/Catch to all move actions—see ["Make missions robust" on page 377](#).
- Avoid creating recursive missions that risk continuing indefinitely.

## Test

After you create a mission, always run the mission to test that the robot executes it correctly.

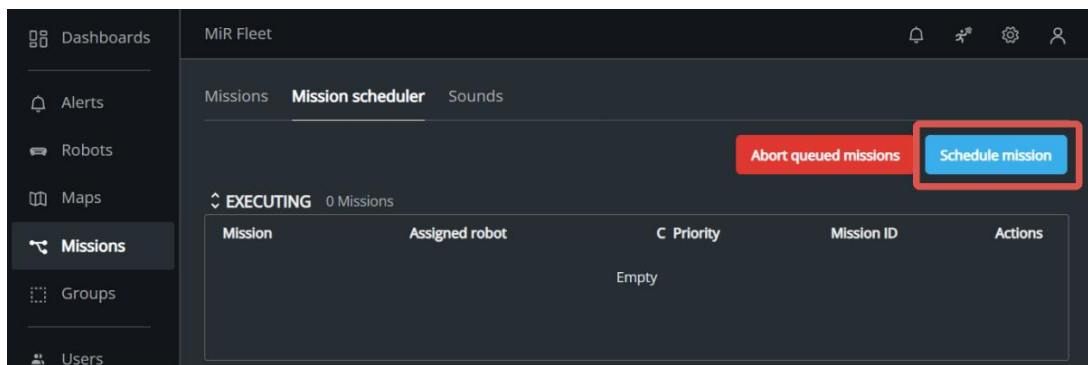


### NOTICE

Always test missions without load on the robot to minimize potential hazards.

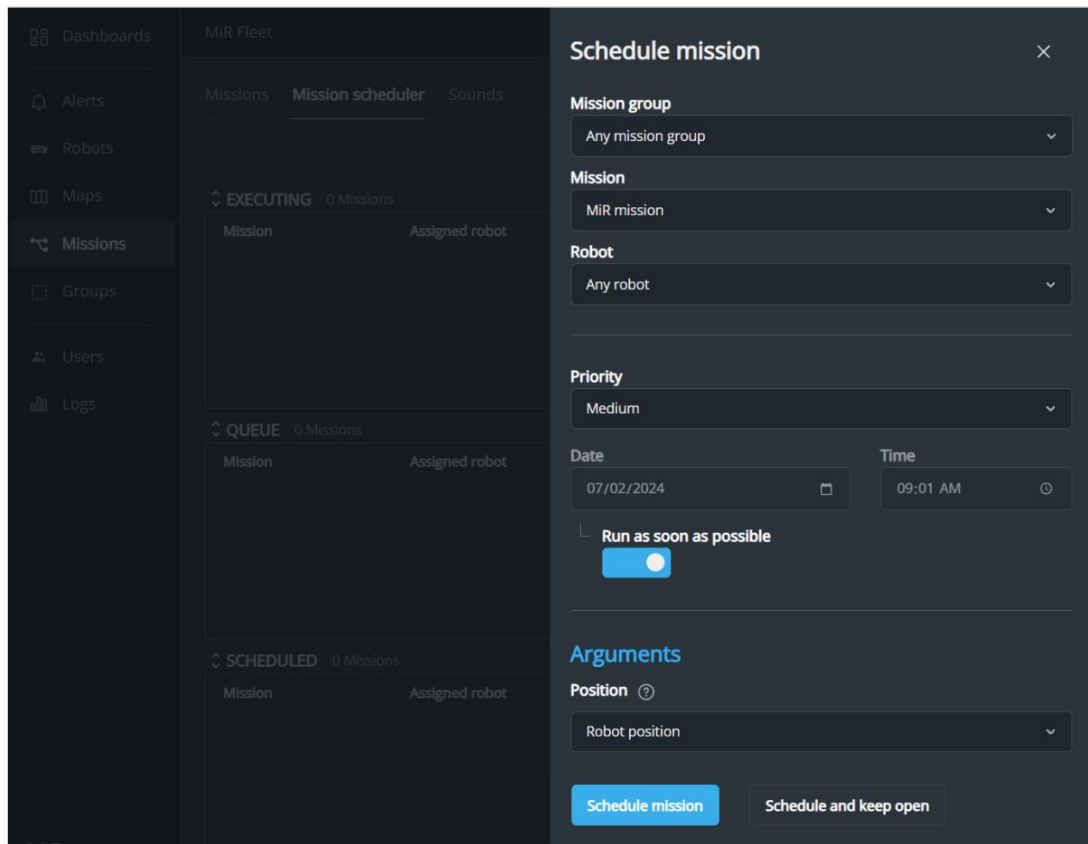
To run a mission, follow these steps:

- 1 Go to **Missions > Mission scheduler**, and select **Schedule mission**.



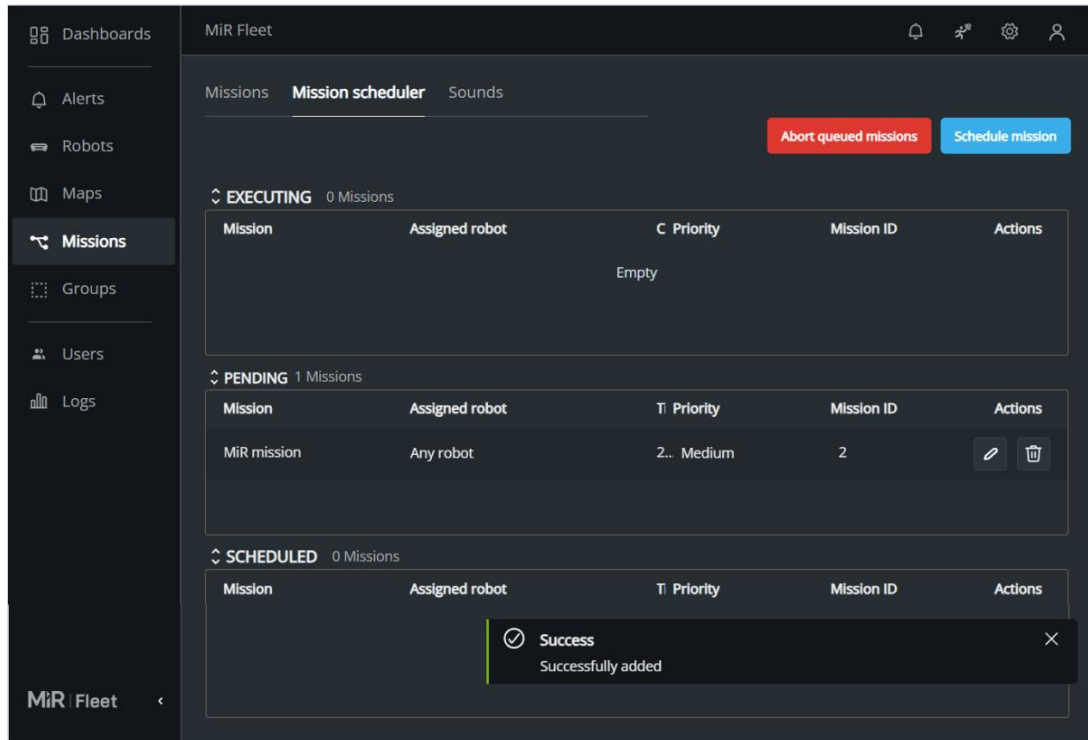


- 2 Fill out the scheduling parameters—see "Schedule mission" on page 85.



- 3 Select **Schedule mission** to schedule the mission and close the dialogue box. Select **Schedule and keep open** to schedule the mission and keep the dialogue box open to schedule another mission.

- 4 Verify that the mission has been added to the mission queue.



Run the mission 5–10 times to ensure that it runs smoothly.

Run some tests where you introduce possible failure triggering events, such as blocking the robot's path, disrupting the robot's field of view when docking, or canceling missions and restarting them.

If something interrupts the mission, use a Try/Catch action in that step of the mission, and decide what the robot does if a mission action fails—see ["Make missions robust" on page 377](#).

If the robot enters Protective stop in areas where the robot does not have enough space to maneuver, consider the following solution:

- Determine if you can move the structure or equipment to make more space.
- Use a Change footprint action to make the robot use a smaller footprint in these areas—see ["Create footprints" on page 404](#). Change the footprint back to the default after it has passed the difficult area.

- Use a Mute Protective fields action, and add the failing action to the scope. Mark the area where the robot has muted Protective fields as an operating hazard zone if you do this—see "[Operating hazard zones](#)" on page 440.

## Document

Create a document with descriptions of each mission:

- Identify which missions are building blocks for other missions.
- Identify what arguments are used in each mission and their purpose.
- Describe what the intended use of each mission is.

Keep the document up to date with a log of any changes that are applied to each mission explaining why the change was made. This ensures you are aware of the change and the issue that caused the change next time you update the mission.

Consider creating graphical representations of your mission structure and logic such as flow charts, logic charts, or Nassi Shneiderman diagrams.

### 4.8.4 Mission examples

The main way to use MiR robots is through missions that you create.

This section of the guide provides examples of small missions that focus on describing common uses of MiR robots.



You can practice making missions in the *Missions* courses. You can find these courses on [MiR Academy](#).

#### Example mission *Narrow doorway*

*Narrow doorway* is a mission example that modifies the robot's navigation and safety settings temporarily to decrease the amount of space it needs. This enables the robot to drive through narrow doorways or similar.

**CAUTION**


When the robot operates with modified safety settings, the robot may not stop for detected personnel or obstacles in time to avoid collision. When the robot operates with a reduced footprint it can reduce the navigation performance of the robot.

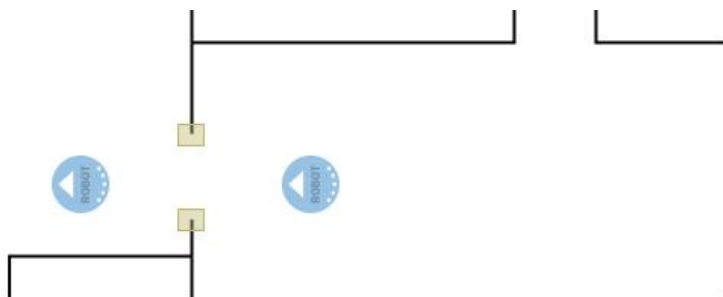
- Ensure that these setting changes are only applied in a limited area.
- Mark the area as an operating hazard zone.

This mission only drives the robot through the doorway in one direction. If you want the robot to go both ways, you need to make a new set of positions facing the opposite direction and a new mission using these positions.

**Environment**

The mission *Narrow doorway* is based on an environment with the following characteristics:

- There is a doorway that the robot can physically fit through but will not plan its path through or enters Protective stop when it gets too close.
- There are Robot positions  on either side of the doorway. The positions must be oriented in the direction of the robot's driving direction.
- In cases where the robot does not drive through the doorway even with the described setup and mission, you can add an Access zone on top of the edges of the doorway—see the guide *How to use zones on a map*. You can find this guide on [MiR Support Portal](#).
- The area around the doorway is marked as an operating hazard zone as the robot will be driving through the doorway with muted Protective fields—see "[Operating hazard zones](#)" on [page 440](#).
- You have a footprint named **Narrow footprint** that is as narrow as possible for your robot. The robot must be centered in the middle of the footprint.



### Mission breakdown

The mission in [Figure 4.7](#) can be broken down into the following steps:

- 1 Use a Move action to send the robot to the start position in front of the narrow doorway.
- 2 Use an Adjust localization action to make the robot adjust its position on the map slightly according the current sensor input. This can make the robot position itself more accurately.
- 3 Use a Set footprint action to change the robot's footprint to a more narrow version.
- 4 Use a Mute protective fields action with a Move action in its scope.

This sends the robot to the position on the other side of the narrow doorway while muting the Protective fields. The robot plays warning sounds while executing actions in the Mute Protective fields action scope.



#### CAUTION

When the robot drives with muted Protective fields, the robot may not stop in time to avoid a collision with personnel.

- Only use Mute protective field actions for actions where the robot is inside a marked operating hazard zone.
- Ensure personnel are informed to stay out of the operating hazard zone when the robot is in it.

Consider nesting this action in a Try/Catch action to provide the robot with an alternative if it fails to drive through the doorway.

- 5 Use a Set footprint action to restore the robot's default footprint. If you do not add this action, the robot will continue to drive with the narrow footprint and may enter Protective stop more often than necessary.

**Figure 4.7** A mission to send the robot through narrow gaps



### Example mission *Variable docking*

*Variable docking* is a mission example where you use arguments so when the mission is nested in another mission or scheduled, you must select which marker the robot should dock to and which I/O port it should wait to be activated before it can undock.

The principles of this mission are useful if you have a site with several almost identical stations where you want to be able to send the robot to any of the stations and execute the same actions. By using arguments, you can change the necessary action parameters each time you run the mission—see "[Define tasks](#)" on page 333.

### Environment

The mission *Variable docking* is based on an environment with the following characteristics:

- There are several markers that the robot can dock to—see the guide *How to create and dock to V-markers, VL-markers, L-markers, and Bar-markers*. You can find this guide on [MiR Support Portal](#).
- There is a loading mechanism at each marker that uses I/O ports to communicate with the robot.
  - The robot must activate one of the input ports to activate the loading mechanism.
  - The loading mechanism activates one of the outputs to indicate it has finished loading.
- One of the robot's PLC registers is reserved to be a mission progress counter. In this example, register 1 is used.

### Mission breakdown

The mission in [Figure 4.8](#) can be broken down into the following steps:

- 1 Use an endless loop to make the robot repeat the mission until it succeeds.
- 2 Use a Try/Catch action to make the robot try to execute the Try actions (actions 3-6), but apply the Catch action (action 8) in case any of the Try actions fail. Use this setup if you want to apply the same Catch action if any of the actions fail. If you want the robot to react differently depending on which action failed, you should add Try/Catch actions for each action instead.

The next four actions use If actions and a PLC counter to make sure the next action is only executed after the previous one was completed successfully. The counter also prevents the robot from executing a completed action a second time. The PLC counter only increments when an action is completed successfully. If an action fails, the robot executes the action under Catch (action 8) and then starts from the beginning of the loop.

- 3 Use a Move action to send the robot to the marker's Entry position. This is optional. It ensures the robot docks more accurately to the marker—see "[Fast docking and Slow docking](#)" on page 57.

Use an argument that you can reuse in the next Dock action to make the robot drive to the same marker that it docks to.

- 4 Use a Dock action to make the robot dock to the marker.  
Use the same argument from the previous action.

- 5 Use I/O ports, or another method of external communication, to make the robot indicate to the loading device that the robot is now in position, and the device can begin loading.

Once loading begins, the robot waits for a chosen I/O port to activate, which indicates that the loading has finished.

- 6 Use a Relative move action to make the robot undock from the marker by reversing two meters.
- 7 Use a Break action at the end of the main actions to make the robot break out of the endless loop as soon as it has completed all the main tasks.
- 8 If the robot fails any of the actions under **Try**, the robot emits a beeping noise to alert nearby personnel that there is an issue, waits one minute, and then restarts the actions under **Try**. The progress counter ensures that the robot does not repeat actions it has already completed.



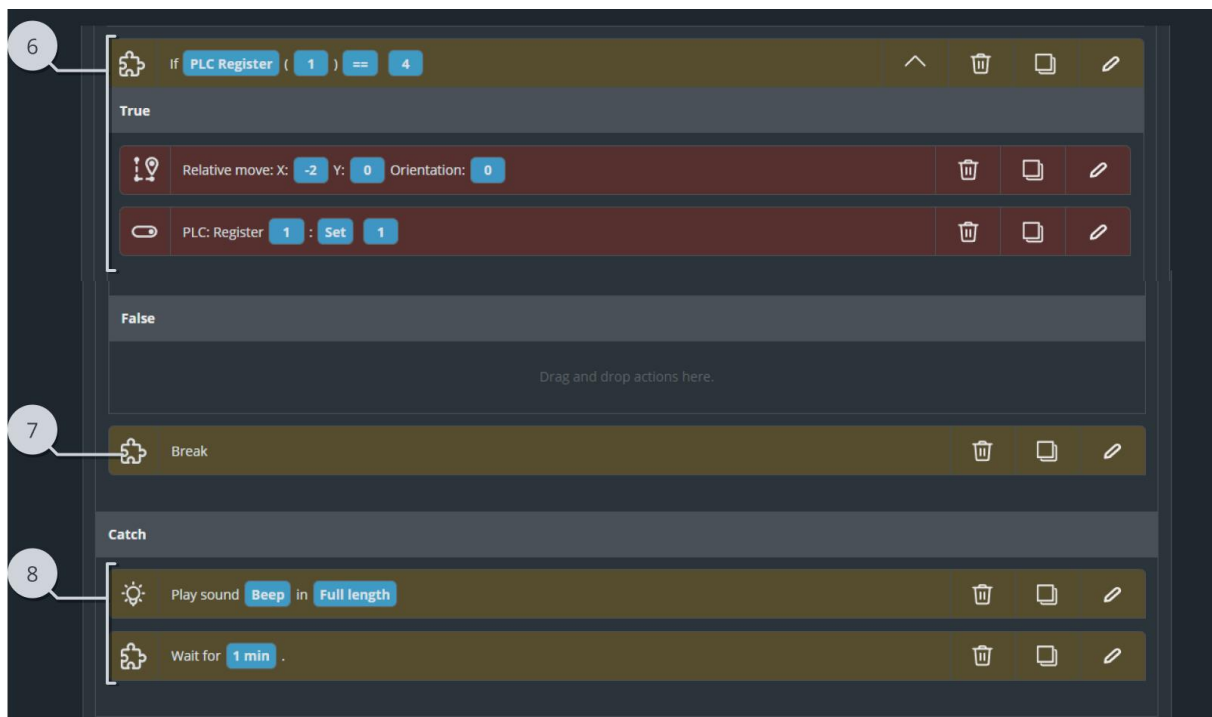
**Figure 4.8** A mission where the robot docks to a selected marker and keeps track of its progress using a PLC register

The screenshot displays a PLC programming environment with a loop structure. The loop is configured for 'endless' iterations. It contains a 'Try/Catch' block and five conditional steps, each triggered when 'PLC Register ( 1 )' equals a specific value. The actions performed in each step are as follows:

- Step 1:** If PLC Register ( 1 ) == 1, the robot performs 'Move to Marker' and sets 'PLC: Register 1 : Set 2'.
- Step 2:** If PLC Register ( 1 ) == 2, the robot performs 'Dock to Marker' and sets 'PLC: Register 1 : Set 3'.
- Step 3:** If PLC Register ( 1 ) == 3, the robot performs 'Set I/O port Input port On on module Conveyor', 'Wait for port Output port on the I/O module Conveyor to become On', and sets 'PLC: Register 1 : Set 4'.
- Step 4:** (No actions are visible for this step in the image).
- Step 5:** (No actions are visible for this step in the image).

Numbered callouts (1-5) on the left side of the image identify the following elements:

- Loop header: 'Loop for endless iterations.'
- Try/Catch block.
- First conditional step: 'If PLC Register ( 1 ) == 1'.
- Second conditional step: 'If PLC Register ( 1 ) == 2'.
- Third conditional step: 'If PLC Register ( 1 ) == 3'.



### Variations

These are some suggestions for how to expand or modify this mission for your site:

- If you use L-markers in your site, you can add a Mute protective fields action if the robot has trouble docking without entering Protective stop.
- You can modify what actions the robot does when it docks to the marker. It can also be pallet racks, shelves, or other kinds of docking stations.
- You can add a PLC counter to count the number of times the robot failed to limit the number of times the robot should try to repeat the mission.
- You can use multiple Try/Catch actions to provide unique Catch actions for each Try action when it fails.

### Example mission *Place shelf at VL-marker*

*Place shelf at VL-marker* is an example mission that makes the robot place a shelf accurately on a shelf position by docking to a VL-marker.

If you need to place shelves with greater precision or have shelves placed close to obstacles or other shelves, you can do so by creating shelf positions in front of markers. If you do not need the robot to place shelves accurately or close together, you can also use the Shelf position alone for both placing and picking up shelves.

To use a marker with a shelf position correctly, your missions must be designed so:

- The robot docks to the marker when placing a shelf. By docking to a marker when the robot places the shelf, it allows the robot to place the shelf accurately to a physical entity while muting its Protective fields.
- The robot docks to the shelf when picking it up. The robot has to dock to the shelf to pick it up to make sure the robot positions itself correctly relative to the shelf type.

### Environment

The mission *Place shelf at VL-marker* is based on an environment with the following characteristics:

- This mission is designed for MiR250 Shelf Carrier. The selected actions for raising and lowering the shelf depend on the robot type.
- At each place where the robot can place a shelf there is a VL-marker the robot can dock to.
- The docked position of each VL-marker is the same as the center of the paired Shelf position. To do this, before adding the VL-markers to the map, send the robot to the Shelf position, and make it detect the VL-marker from there. This will automatically make the docked position for the VL-marker the same as the Shelf position.
- The Entry position of each of the VL-markers has been modified to the same Entry position as the paired Shelf position. This ensures that the robot starts docking far enough from the VL-marker that neighboring shelves do not block the robot. This is mostly relevant if you have a line of marker and Shelf positions.

To make the Entry positions the same, follow these steps:

- 1 Open the Setting for the Shelf Entry position—see "[Adjust Entry position](#)" on page 341.
- 2 Copy the X and Y-coordinates.
- 3

Open the settings for the VL-marker Entry position.

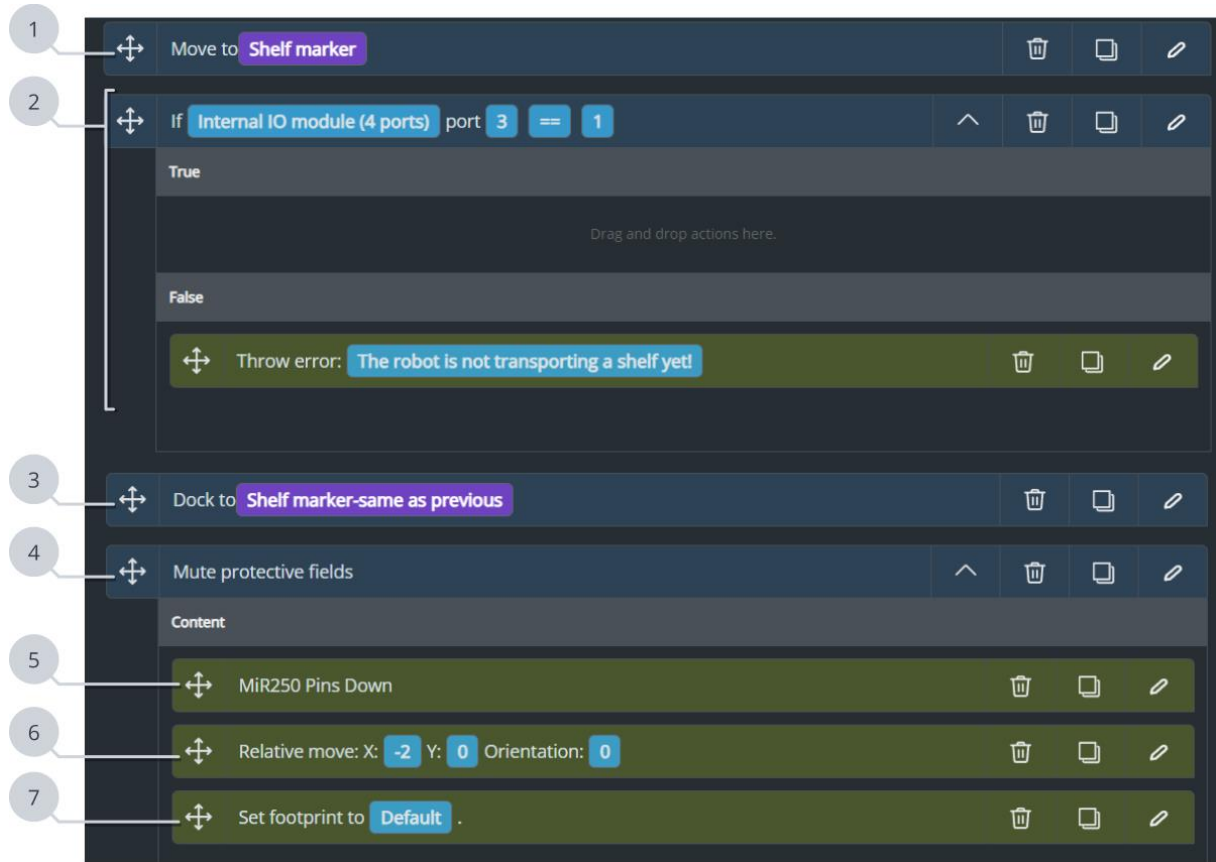
- 4 Paste the X and Y-coordinates copied from the Shelf entry position and select **Save**.

### Mission breakdown

The mission in [Figure 4.9](#) can be broken down into the following steps:

- 1 Use a Move action to make the robot move to the Entry position of the VL-marker.
- 2 Use an If statement to make the robot check if its pins are up. If the pins are down, the robot cannot be transporting a shelf, and it reports an error to inform the user that it should receive another mission.
- 3 Use a Dock action to make the robot dock to the VL-marker. This will position the robot in the center of the Shelf position.
- 4 Use a Mute protective fields action to prevent the robot from entering Protective stop after it detaches and undocks from the shelf.
- 5 Use a generic mission to lower the pins on the top module. In this case, the site has a *MiR250 Pins down* mission to make the robot lower the shelf carrier pins and detach itself from the shelf.
- 6 Use a Relative move action to make the robot undock from the shelf by moving two meters back.
- 7 Use a Set footprint action to make the robot apply its default footprint now that it is no longer transporting a shelf.

**Figure 4.9** A mission that makes a robot dock to a VL-marker, place a shelf, and undock



### Variations and improvements

These are some suggestions for how to expand or modify this mission for your site:

- If you choose to use L-markers and you have issues with the robot entering Protective stop when it docks to the marker, make sure to nest both the docking and undocking actions inside a Mute protective fields action.
- Use a copy of the *MiR250 Pins down* mission that includes a Try/Catch action for communicating with the top module so you can include an alternative action in case there is a communication failure.

### Example mission *Pick up and collect with cart*

*Pick up and collect with cart* is a mission example that demonstrates how you use cart actions to make a robot pick up a cart, drive it to several positions where the cart can be loaded, and then return the cart to its original position.

### Environment and setup

The mission *Pick up and collect with cart* is based on an environment with the following characteristics:

- The mission is designed for a robot using a MiR hook top module.
- There is already a cart ready for the robot at the Cart position.
- The robot just has to stop and wait at its two pick up locations. It does not need to signal or communicate with any devices.
- It is not vital for the hook to make it to every stop. Instead of waiting and trying to get to the position, it is preferable for the robot to just drive to the next stop.
- One of the robot's PLC registers is reserved to be a mission progress counter. In this example, register 1 is used.

### Mission breakdown

The mission in [Figure 4.10](#) can be broken down into the following steps:

- 1 Use a Set PLC register to make sure the PLC register you want to use for counting starts from 1.
- 2 Use a While loop to make the robot repeat the next actions until it succeeds.
- 3 Inside the Try section of a Try/Catch action, use a Pick up cart action to make the robot pick up a cart from the selected Cart position. Use the PLC register to indicate when it has succeeded the Pick up action to exit the While loop.
- 4 Inside the Catch section of the Try/Catch action, use a Relative move action to make the robot move forward a bit before trying to pick up the cart again. For every time it fails to pick up the cart, the PLC counter increments by one.
- 5 Use an If action to make the robot report an error if the PLC counter exceeds 3.

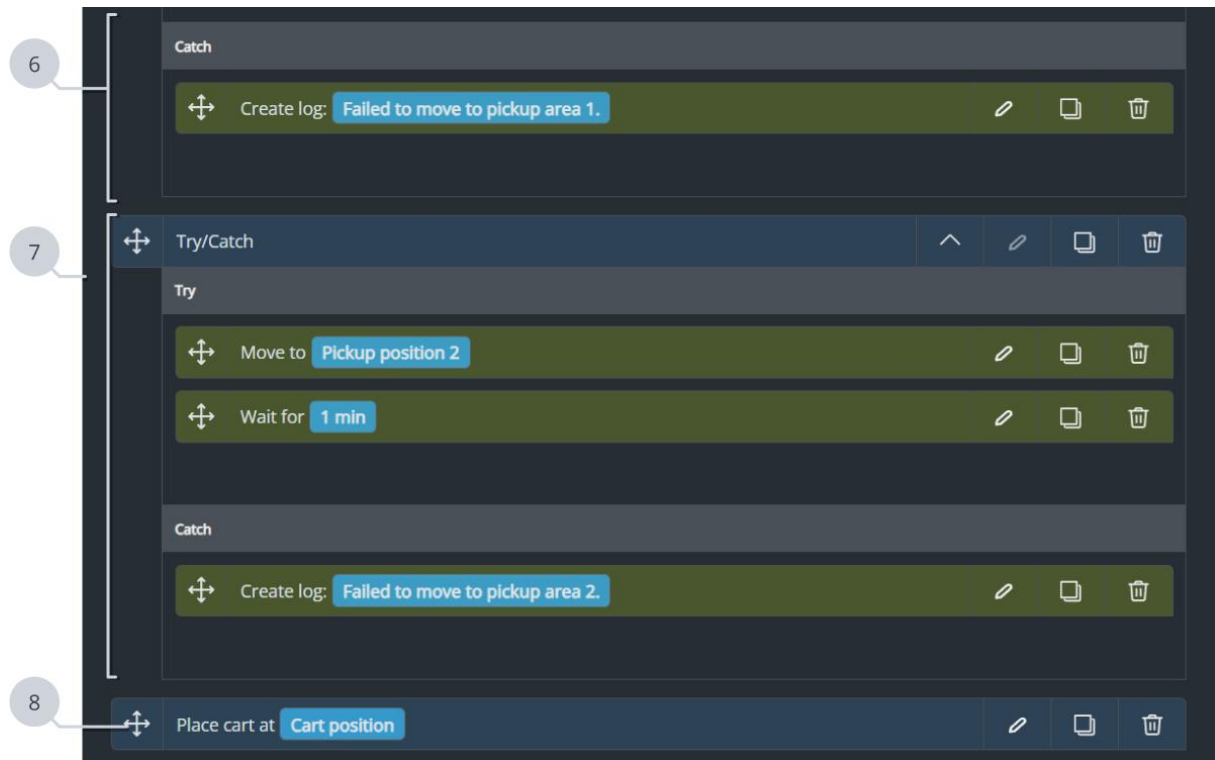
- 6 Use a Move action in a Try/Catch to make the robot move to a pick-up location. The robot waits one minute for the cart to be loaded. If it fails to reach the position, it generates an error log, and then continues to the next action.
- 7 Repeat the same Try/Catch action but with a new pick-up location.
- 8 Use a Place cart action to make the robot place the loaded cart back where it started.

**Figure 4.10** A mission that makes a robot with a hook pick up a cart and stop by two pick up locations before returning the cart to the start position

The screenshot shows a mission programming interface with the following components and callouts:

- 1:** A PLC Register block: `PLC: Register 1 : Set 2`.
- 2:** A While loop block: `While PLC Register ( 1 ) != 1`.
- 3:** A Try/Catch block containing:
  - Try:**
    - Pick up cart at `Cart position`
    - PLC: Register `1 : Set 1`
  - Catch:**
    - Relative move: X: `0.3` Y: `0` Orientation: `0`
    - PLC: Register `1 : Add 1`
    - If `PLC Register ( 1 ) > 4`:
      - True:** Throw error: `Failed to pick up cart after 3 attempts!`
      - False:** Drag and drop actions here.
- 4:** Points to the Relative move block in the catch section.
- 5:** Points to the If block in the catch section.
- 6:** A Try/Catch block containing:
  - Try:**
    - Move to `Pickup position 1`
    - Wait for `1 min`





### Variations and improvements

These are some suggestions for how to expand or modify this mission for your site:

- The Place cart action at the end of the mission could be nested in a loop and Try/Catch action to implement an alternative action in case the action fails.
- You can add more Robot positions where the robot can stop while towing the cart. You might also want to include some communication actions to indicate to another system or nearby personnel that the robot has arrived.
- If it is important for the robot to reach every stop, you can include more actions under Catch to make the robot try to reach the position again instead of continuing to the next stop.

### Example mission *Pick up and place pallet from rack*

*Pick up and place pallet from rack* is a mission example where a robot picks up a pallet from one pallet rack, drives to another pallet rack, and places the pallet.

### Environment

The mission *Pick up and place pallet* is based on an environment with the following characteristics:

- Your site has MiR pallet racks. You can also use many parts of this mission with custom pallet racks, but you will likely have to define them using Bar-markers, and the Check position status action can only be used with Pallet rack markers that match MiR pallet rack dimensions.
- Your robot has a MiR pallet lift top module.
- There is a pallet on *Pallet rack 1*.
- There is a nearby user who can verify that the robot does not have a pallet on it before starting the mission.

### Mission breakdown

The mission in [Figure 4.11](#) can be broken down into the following steps:

- 1 Use an If action to check if the robot is already carrying a pallet.
- 2 Use a generic mission to pick up a pallet from a selected rack. In this case, the site has a *Pick pallet from rack* mission to make the robot pick up a pallet from a pick up location. This mission makes the robot execute the following actions:
  - a Moves the robot to the Entry position.
  - b Checks if there is a pallet. If the robot cannot detect a pallet within 10 seconds, the robot reports an error.
  - c The robot lowers the lift or forks using the template mission *Lift down*. This template mission can also be made more robust. The mission makes sure that the lift is not being raised (input port 3 is off ) and then lowers the lift (activates input port 2) until the lift is lowered (output port 2 is on).
  - d The robot docks to the pallet rack.
  - e The robot raises the lift to pick up the pallet using the template mission *Lift up*. This template mission can also be made more robust if you have issues with raising the lift, for example, make it try to raise the lift a second time if it fails once. The mission makes sure that the lift is not being lowered (input port 2 is off ) and then raises the lift (activates input port 3) until the lift is raised (output port 3 is on).
  - f The robot reverses two meters to undock from the pallet rack.
  - g The robot lowers the lift using the template mission *Lift down*. This positions the lift securely on the robot.

- 3 Use a generic mission to place a pallet on a selected rack. In this case, the site has a *Place pallet from rack* mission to make the robot place the pallet it is carrying onto a pallet rack. This template mission makes the robot execute the following actions:
  - a Moves the robot to the Entry position of the pallet rack.
  - b Checks if the pallet rack is empty. If the robot cannot determine if the rack is empty in 10 seconds, the robot reports an error.  
You can only use this action if your pallet racks are defined as Pallet rack markers.
  - c The robot raises the lift using the template mission *Lift up*.
  - d The robot docks to the pallet rack.
  - e The robot lowers the lift to place the pallet using the template mission *Lift down*.
  - f The robot reverses two meters to undock from the pallet rack.
  
- 4 Use a Throw error action to make the robot report an error if no one responds to the prompt.

**Figure 4.11** A mission to make the robot pick up and place a pallet using copies of the template missions

The figure displays a mission editor interface with the following structure:

- 1**: Conditional mission: `If Internal IO module (4 ports) port 3 == 1`
  - True** branch:
    - 2**: `Pick pallet from rack`
      - 2.a**: `Move to PalletRackMarker`
      - 2.b**: `Check whether PalletRackMarker is Occupied with timeout 10 sec`
      - 2.c**: `Lift Down`
      - 2.d**: `Dock to PalletRackMarker`
      - 2.e**: `Lift Up`
      - 2.f**: `Relative move: X: -2 Y: 0 Orientation: 0`
      - 2.g**: `Lift Down`
    - 3**: `Place pallet on rack`
      - 3.a**: `Move to PalletRackMarker`
      - 3.b**: `Check whether PalletRackMarker is Free with timeout 10 sec`
      - 3.c**: `Lift Up`
      - 3.d**: `Dock to PalletRackMarker`
      - 3.e**: `Lift Down`
      - 3.f**: `Relative move: X: -2 Y: 0 Orientation: 0`
  - False** branch:
    - 4**: `Throw error: Robot is already carrying a pallet!`

### Variations and improvements

These are some suggestions for how to expand or modify this mission for your site:

- Add Try/Catch actions around each actions in the template mission to provide the robot with an alternative action or solution whenever one of the actions fail.
- Make the pallet rack parameters into arguments so you can reuse the same mission for any set of pallet racks.
- If you cannot define your pallet rack as Pallet rack markers, you can use Bar-markers instead. In this case, you must remove the Check position status actions from the template mission, as this action is only compatible with MiR pallet racks.
- Implement a mission progress counter so the robot will continue the mission even if it is canceled or fails in the middle of the mission. This is especially relevant if the robot has already picked up the pallet and is driving toward the goal pallet rack but is interrupted. If the mission is restarted, it will go back to the first pallet rack and fail the mission since there is no longer a pallet for it to pick up.
- Implement another method or system to guarantee that the robot is not already carrying a pallet before starting the mission.

### Example mission *MiR1200 Pallet Jack pick up and place pallet*

*MiR1200 Pallet Jack pick up and place pallet* is a mission example where MiR1200 Pallet Jack picks up a pallet from one Pallet position, drives to another Pallet position, and places the pallet there.

### Environment

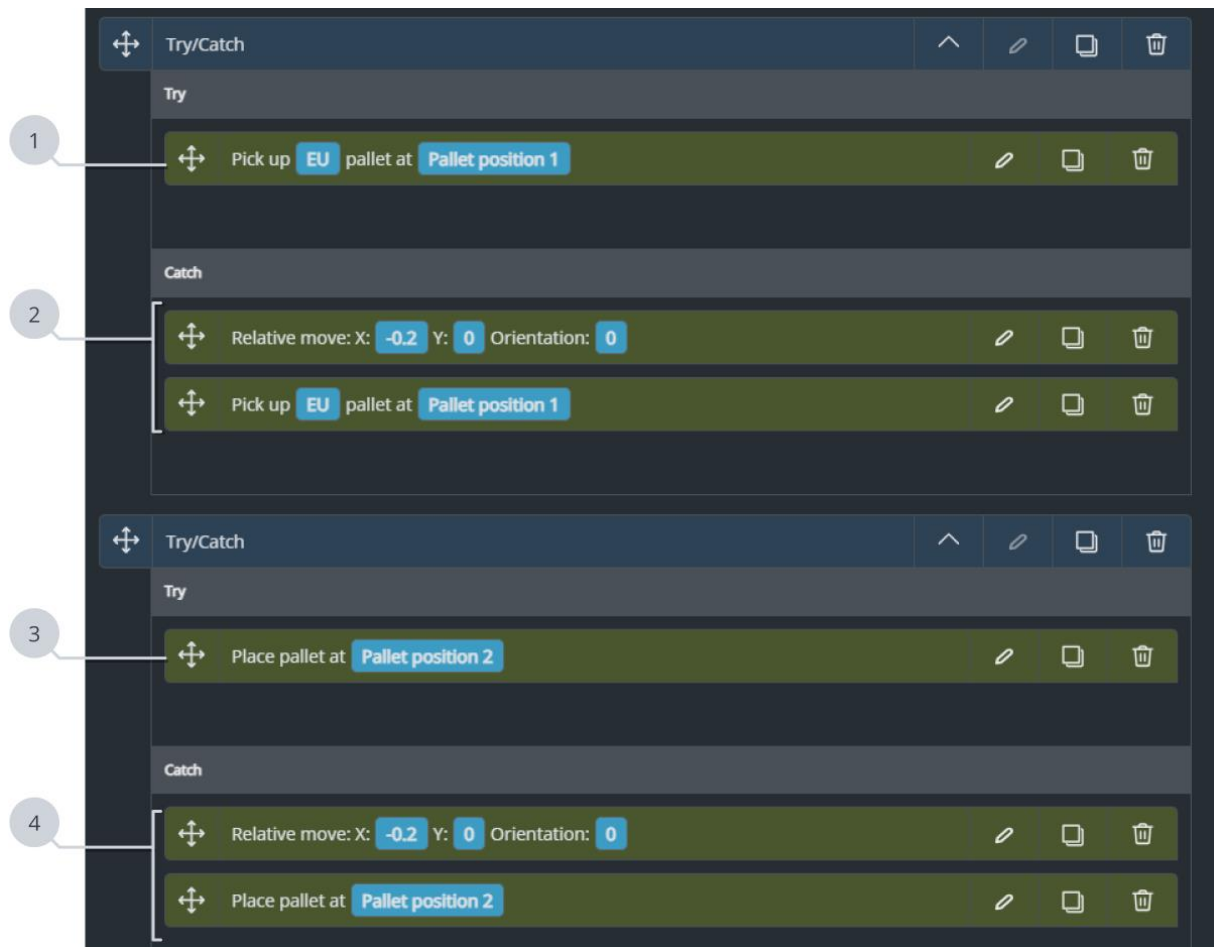
The mission *MiR1200 Pallet Jack pick up and place pallet* is based on an environment with the following characteristics:

- Your site has a MiR1200 Pallet Jack.
- Your site has two Pallet positions with an EU pallet on one of them.

### Mission breakdown

The mission in [Figure 4.12](#) can be broken down into the following steps:

- 1 Use the Pick up pallet action to make the robot pick up a pallet. This action makes the robot execute the following actions:
  - a Moves the robot to the most suitable of the three Entry positions of the Pallet position.
  - b Checks if there is a pallet on the Pallet position. If the robot cannot detect a pallet within 10 seconds, the robot reports an error.
  - c The robot lowers the forks.
  - d The robot docks to the pallet rack.
  - e The robot raises the forks to pick up the pallet
  - f The robot reverses two meters to undock from the Pallet position.
  
- 2 If the robot fails to pick up the pallet, it reverse 20 cm and tries again once. If it fails again, the robot reports an error and fails the mission.
  
- 3 Use the Place pallet action to make the robot place a pallet it is carrying onto the designated Pallet position. This action makes the robot execute the following actions:
  - a Moves the robot to the most suitable of the three Entry positions of the Pallet position.
  - b Checks if the Pallet position is empty. If the robot cannot determine if the rack is empty in 10 seconds, the robot reports an error.
  - c The robot docks to the Pallet position.
  - d The robot lowers the forks to place the pallet.
  - e The robot reverses two meters to undock from the Pallet position.
  
- 4 If the robot fails to place the pallet, it reverse 20 cm and tries again once. If it fails again, the robot reports an error and fails the mission.

**Figure 4.12** A mission to make the robot pick up and place a pallet

### Variations and improvements

These are some suggestions for how to expand or modify this mission for your site:

- Implement a mission progress counter so the robot will continue the mission even if it is canceled or fails in the middle of the mission. This is especially relevant if the robot has already picked up the pallet and is driving toward the goal Pallet position but is interrupted. If the mission is restarted, it will go back to the first Pallet position and fail the mission since there is no longer a pallet for it to pick up.
- Make the robot check if there is an available pallet on the Pallet position and provide an alternative location or error stating that there is no pallet.
- Make the robot check if it is already carrying a pallet before trying to pick up another.



## 4.8.5 Make missions robust

### Section overview

Make missions more robust.

- Test missions where you expose the robot to unexpected events you want it to handle autonomously.
- Use Try/Catch actions to define alternative actions in missions when exceptions or unexpected events occur.
- Iterate between testing and modifying missions.
- Update the mission documentation if necessary.

A robust mission is designed to register when something in the mission has gone wrong and applies a solution to handle the unexpected event.

The following sections describe the main kinds of actions you can use to make missions robust.

### Actions to make missions robust

#### Try/Catch

Use a Try/Catch action to define exactly what the robot should do if it fails an action. We recommend always using a Try/Catch action around any action that can fail where you want the robot to do more than just abort the mission and report an error.

#### Create log

Use a Create log action if you have an issue that you need help from MiR Technical support to resolve. Make sure to implement the Create log action right after the issue occurs. Error logs only capture the last 30 seconds of a robot's activity, so it is important that you send an error log that was generated briefly after the failure occurred.

If the robot generates many error logs, we recommend removing old logs to avoid them taking up space on the robot.

### Report error

Use a Report error action to make the robot stop its mission and go into error mode. This is useful in situations where you have no alternative action for the robot, and a user must intervene to assign it a new task. Combine this with a Send email action to make sure an operator is notified that the robot failed its mission.

### Loops

Use loops when you want a robot to try to execute a failed action again. Often, you will want the robot to try another action after it has failed the first time to ensure that the circumstances in the second attempt have changed. You can use the following actions to create a loop:

#### Loop

Use when you have an action you want to repeat. This can be useful to make the robot repeat an action a specified number of times—see ["Make missions robust" on the previous page](#)

#### While

Use when you have an action you want to repeat provided a specified condition is met. This can be used in much the same way as a Loop action, but you can specify under what condition the robot should repeat the action.

### Communication

You can alert relevant personnel that the robot has run into an exception and may need assistance to determine the correct action. You can use the following actions to notify personnel or external devices:

#### Play sound and Show light

Use when you want personnel that are physically close to the robot to be alerted that the robot is nearby or needs intervention. Use in areas where obstacles or personnel can unintentionally be in the way of the robot and someone needs to clear the robot's path so it can continue its mission.

For more advanced use, you can map specific sounds or lights to specific issues to make it easier to identify the issue.

### Set output and Set PLC register

Use if you have an external system connected to the robots where you can use PLC registers or I/O modules to signal different states or stages in the mission. The example Resumable missions with progress tracking could be used with an external system that reads the same register to see how far the robot has progressed in the mission. You could expand on it to a specific register value triggers an external device at a certain point in the mission.

### Triggers and checks

You can use PLC registers, I/O modules, and Position status checks to define what the robot should do depending on the current state of a chosen register, input, or position. You can use the following actions to check for external input for the mission:

#### Position status check

Use whenever the robot should check:

- If there is room for the robot at a position.
- If there is a pallet on a MiR pallet rack.

You can use this before the robot moves to any position or docks to a MiR pallet rack to immediately provide an alternative action if the position or pallet rack is not as expected. For a long row of markers, it can take a long time for the robot to check the status of each marker. Consider using I/O modules with an external system to indicate which marker to pick up or place a load at instead.

For more information about how this action works and examples of usage, see the guide *How to use Check position status*. You can find this guide on [MiR Support Portal](#).

#### Wait for PLC register / Wait for input

Use whenever you have an external device or system that signals a state via an input.

#### If

Use to specify under what condition an action or group of actions should be executed.

## Corrections

You can use some actions to modify the situation in case of an exception, and then make the robot try again after trying to automatically resolve the issue. These should only be implemented after testing and assessing that it is a safe solution and it resolves the issue. You can use the following actions to modify the robot's situation before retrying an action:

### Adjust localization

Use when the robot's localization tends to drift in certain parts of the mission and adding an Adjust localization action often corrects the robot's localization. Only use this for actions that span over a very limited area so you are certain that the robot adjusts its localization in the expected and tested area.

You can consider using this action in combination with moving the robot to a configuration position first. The configuration position should be somewhere with many static landmarks the robot can use to more accurately correct its localization. If you do so, keep in mind that depending on how poorly the robot is localized, it may not be able to reach the position accurately, so the configuration position should not be too far from the robot and the position must have a large distance threshold.

You should never use two Adjust localization actions consecutively. This can cause unexpected localization issues.

### Change footprint

Use when the robot sometimes fails to plan a path due to limited space.

Only use this for actions that span over a very limited area so you are certain that the robot is running with another footprint in the expected and tested area. Remember to reset the footprint after changing it.

### Mute protective fields

Use when the robot sometimes enters Protective stop due to limited space but can navigate the area when the Protective fields are muted. This can be used in combination with changing the robot's footprint to a smaller one.

Only use this for actions that span over a very limited area so you can mark the area as an operating hazard zone—see "[Operating hazard zones](#)" on page 440. Make sure to also consider when the robot should reactivate the Protective fields.

### Relative move

Use when it helps to set a static movement for the robot through a difficult area instead of using the planner. Make sure you are fully controlling the direction and placement of the robot before using the Relative move action. It is only intended to be used for minor corrections. Avoid having several consecutive Relative move action as the robot's localization will become increasingly inaccurate.

### Planner settings

Use when changing one of the planner settings helps the robot resolve an issue that often occurs at this point in the mission. For more information about the planner settings, see *MiR Robot Interface Guide*. You can find this guide on [MiR Support Portal](#).

### Test mission robustness

Test missions where you prompt actions to fail. Verify that the actions in the Catch scope make the robot apply a suitable alternative.

If you have implemented any looping functions, make sure to test that the robot loops the expected number of times.

### Documentation

- Update the mission documentation if relevant.
- Include an overview of the basic troubleshooting or solution that are applied when specific actions fail.

### Robust mission examples

The following sections describe common action combinations you can use to make your missions more robust:

- "[Repeat actions](#)" on the next page: Make loops where the robot repeats an action until it succeeds. You can expand the examples to integrate different solutions to resolve the issue before the robot tries to execute the failing action again.
- "[Wait or check for a response](#)" on page 386: Make the robot check an I/O module input or check for obstacles to determine the correct course of action.
- "[Location specific solutions](#)" on page 388: Make the robot use different solutions to resolve a failed action depending on where the robot is on the map.

- ["Resumable missions with progress tracking" on page 392](#): Make robots keep track of how far it has progressed in the mission in case the mission is aborted and rescheduled with the expectation that the robot continue from where it left off.
- ["Use a redundant system" on page 394](#): Make the robot use a second communication line in case the first one fails.
- ["Top module cycle counting and resetting" on page 394](#): Make the robot count how many times its top module is activated, and reset it regularly after being used a set number of times.

### Repeat actions

After a robot fails an action, you can make it repeat the action for a specified number of attempts or until it succeeds. When you implement this solution we recommend implementing:

- a solution before repeating the action.
- a counter so the robot eventually reports an error instead of continuously repeating the same action.

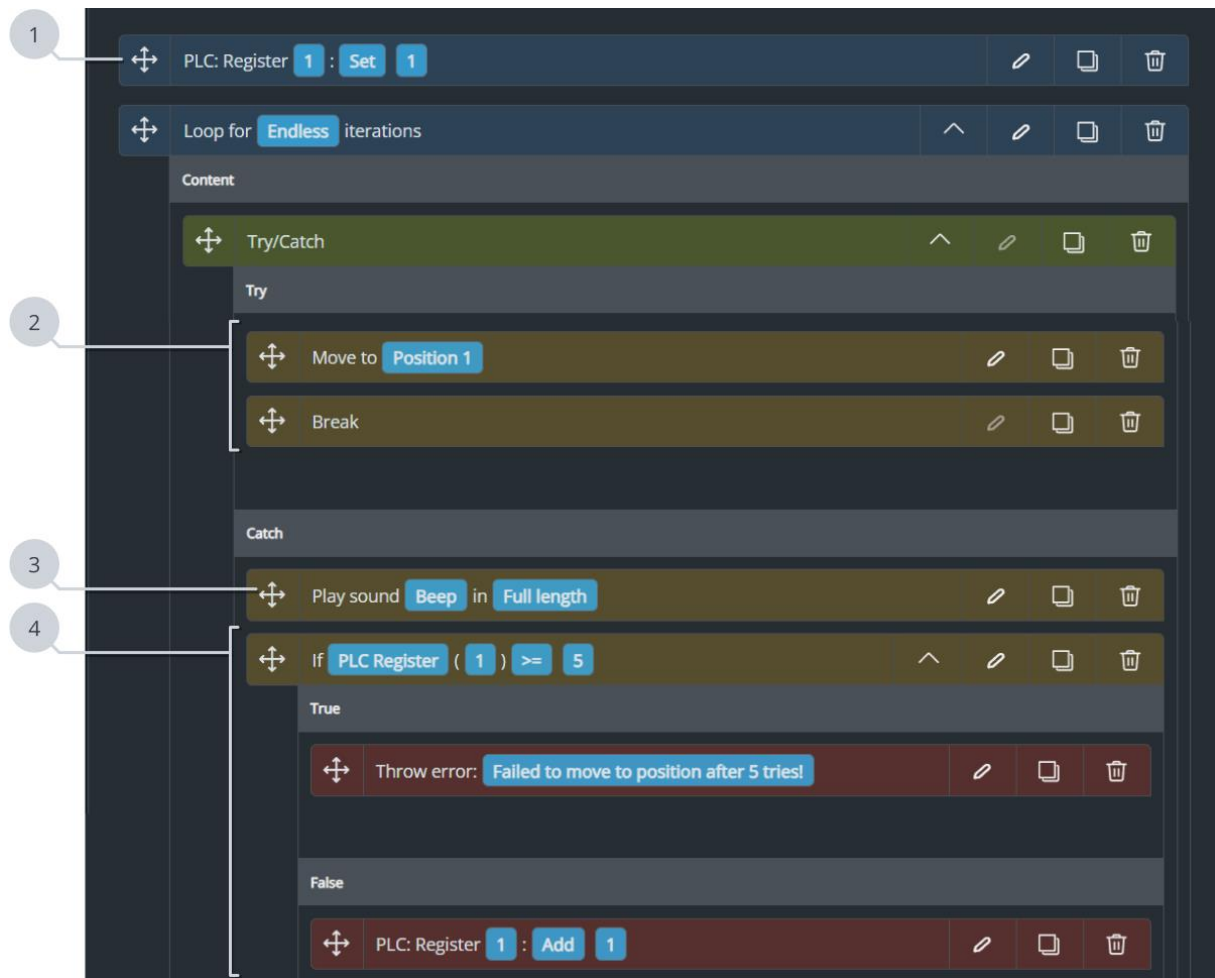
### Single solution

The example mission in [Figure 4.13](#) executes as follows:

- 1 The robot sets PLC register 1 to 1. This register counts how many times the robot has repeated the loop below.
- 2 The robot tries to move to Position 1. If it succeeds, the loop breaks and the robot continues the mission.
- 3 If the robot fails to move to the position, it plays a sound to alert nearby personnel that it cannot reach its goal destination. This solution only works if the robot operates in areas with personnel who have been told what to do when the robot beeps.
- 4 The robot checks if the loop has been repeated five times. If it has, the robot reports an error.

If not, the robot adds 1 to the PLC register counter to keep track of how many times the loop has been repeated. The mission continues to the top of the loop, and the robot tries to move to Position 1 again.

**Figure 4.13** A mission where the robot tries to move to Position 1 up to five times where it tries to notify nearby personnel each time it fails



### Multiple solutions

You can expand the previous example to change which solution is applied after each iteration.

The example mission in [Figure 4.14](#) executes as follows:

- 1** The robot tries to move to Position 1. If it succeeds, the loop breaks, and the robot continues the mission.

2 If this is the first time the robot fails to move, it tries to beep to alert nearby personnel that its path is blocked. The mission continues to the top of the loop and the robot tries to move to Position 1 again.

3 If this is the second time the robot failed to move, it tries to move to a waiting position where it is not in the way of other vehicles. The mission continues to the top of the loop, and the robot tries to move to Position 1 again.

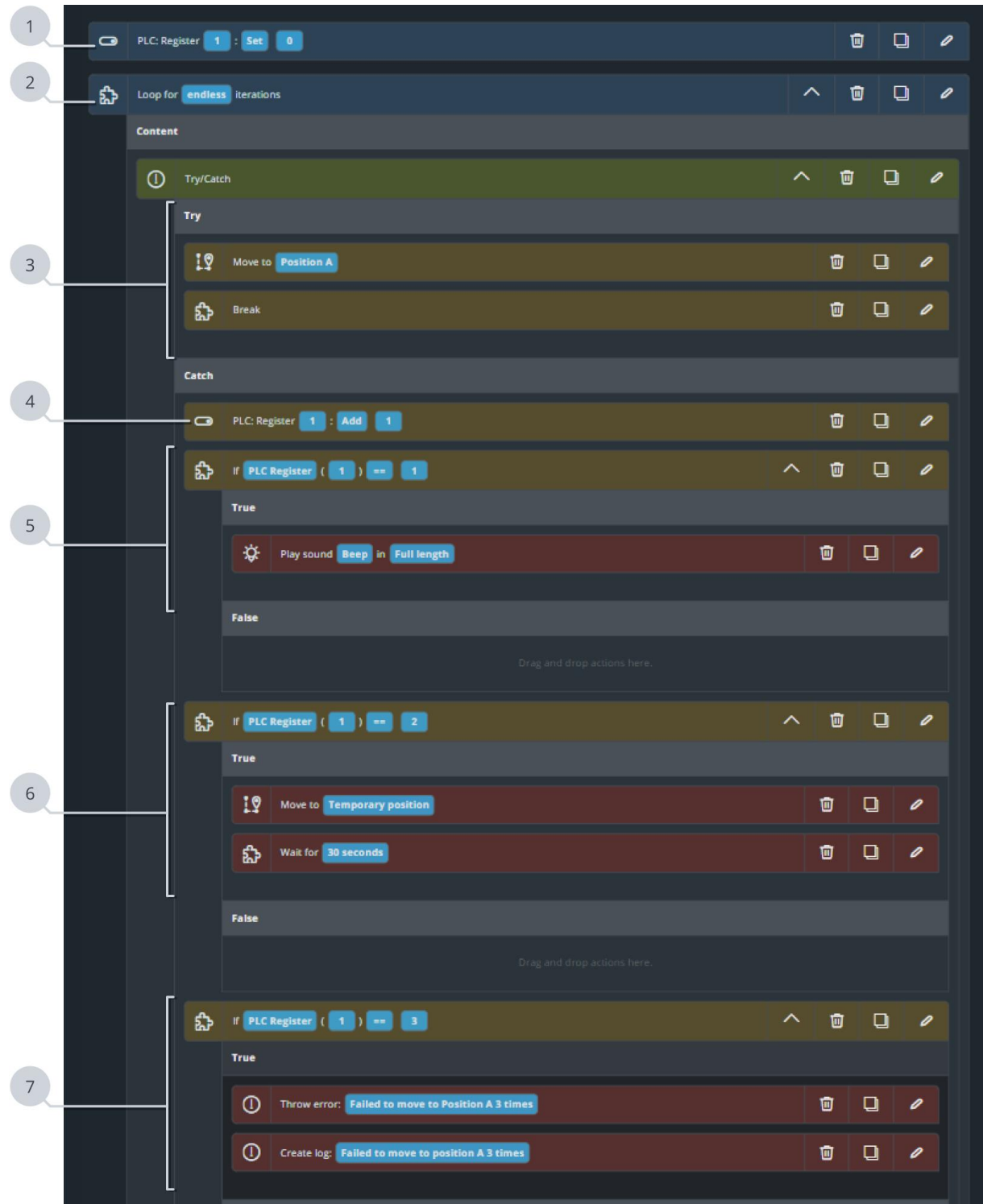
This can be a useful solution if the robot has to go through a narrow passage and should allow other vehicles from the opposite direction to pass first.

4 If this is the third time the robot fails to move, it reports an error.

5 After applying the solution for this iteration, the robot increases the counter by 1, so next time it fails to move to Position 1, it tries another solution.



**Figure 4.14** A mission where the robot tries to move to Position 1 and tries different solutions if it fails



You can expand this mission with any number of solutions. Only implement solutions that you have tested and verified can resolve the issue.

### Wait or check for a response

Implement Wait for output actions and Check position status actions to make the robot wait for a specific result before continuing.

If you implement these actions, always add a timeout to the action to avoid the robot waiting for an undefined length of time.

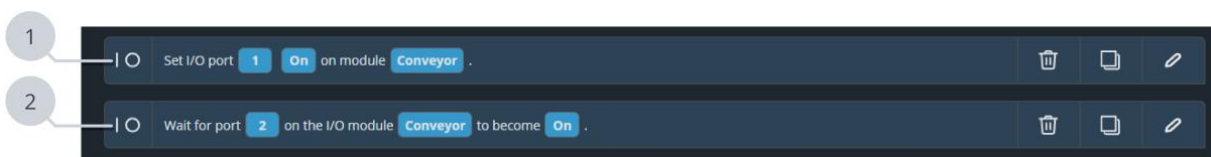
### Wait for I/O output

If you have one port that triggers an event when activated and another port that activates as soon as the triggered event is complete, you should always implement a Wait for output action. In this way, you can confirm that the event the robot was supposed to trigger has been completed.

The example mission in [Figure 4.15](#) executes as follows:

- 1 The robot tries to activate a conveyor by enabling port 1 in the conveyor's I/O controller.
- 2 The robot waits for the conveyor controller to signal it has finished loading an item by activating port 2. The robot can then continue to the next part of its mission and be sure that the conveyor has finished loading the item.

**Figure 4.15** A mission that makes the robot wait for a response before continuing



In both actions, you should set a timeout to ensure the robot reports an error if it does not receive a response.

### Wait for position status

If the robot triggers a physical change that the robot can detect using the cameras or laser scanners, you should implement a Check position status. In this way, you can confirm that the event the robot was supposed to trigger has been completed.

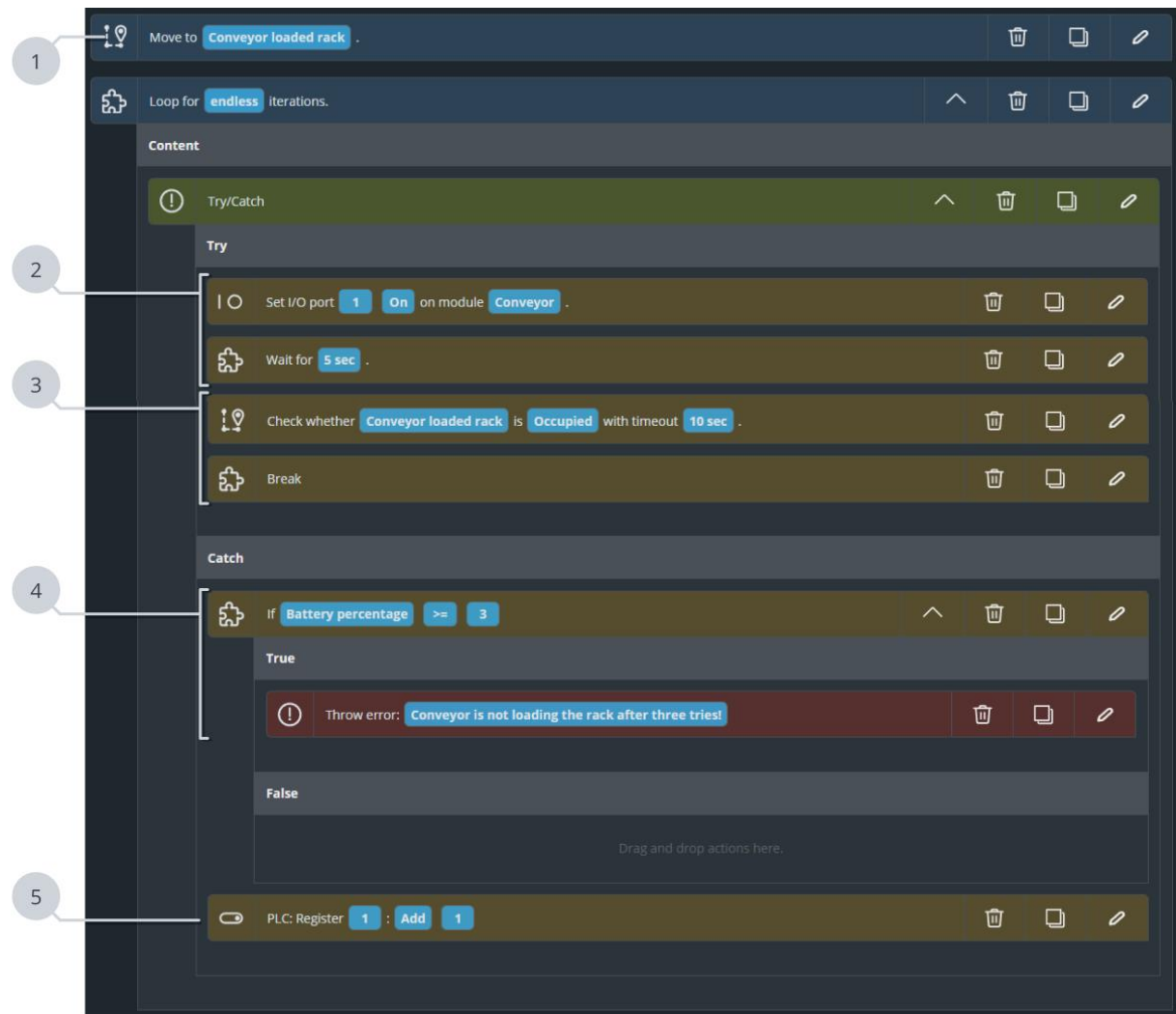


For more information about how the Check position status action works and examples of usage, see the guide *How to use Check position status*. You can find this guide on [MiR Support Portal](#).

The example mission in [Figure 4.15](#) executes as follows:

- 1 The robot moves to the Entry position of a pallet rack.
- 2 The robot tries to activate a conveyor by enabling port 1 in the conveyor's I/O controller. When the conveyor activates, it loads a pallet onto the pallet rack for the robot.
- 3 The robot checks if a pallet has been placed on the pallet rack. If it has, it breaks out of the loop and continues the mission.
- 4 If the robot does not detect a pallet on the pallet rack, it checks how many times it has repeated the loop. If it is the third time it has tried to activate the conveyor, it reports an error.
- 5 If the robot has not repeated the loop three times, it increases the counter, starts from the top of the loop, and tries to activate the conveyor again.

**Figure 4.16** A mission where the robot checks if a rack is loaded, and if not, the robot activates a conveyor to load the rack



**Location specific solutions**

If you have a case where you can implement different solutions to remedy common issues that occur in a Move action, but each solution is only useful in specific areas of the map, you can use I/O module zones to separate what the robot should do in each area.

Figure 4.17 is a map example where I/O modules zones have been implemented to support the mission example in Figure 4.18. In the map setup, it is important to:

- Assign different register values to each zone that you can use to trigger different actions in the corresponding mission.
- Choose to reset the chosen register to the default value when the robot exits the zone. If you do this, you can include an alternative action in case the robot is not in any of the zones. You can also choose not to reset the value so the robot will always apply one of the chosen solutions.

**Figure 4.17** A map where I/O module zones are used to affect which solution is applied in each area

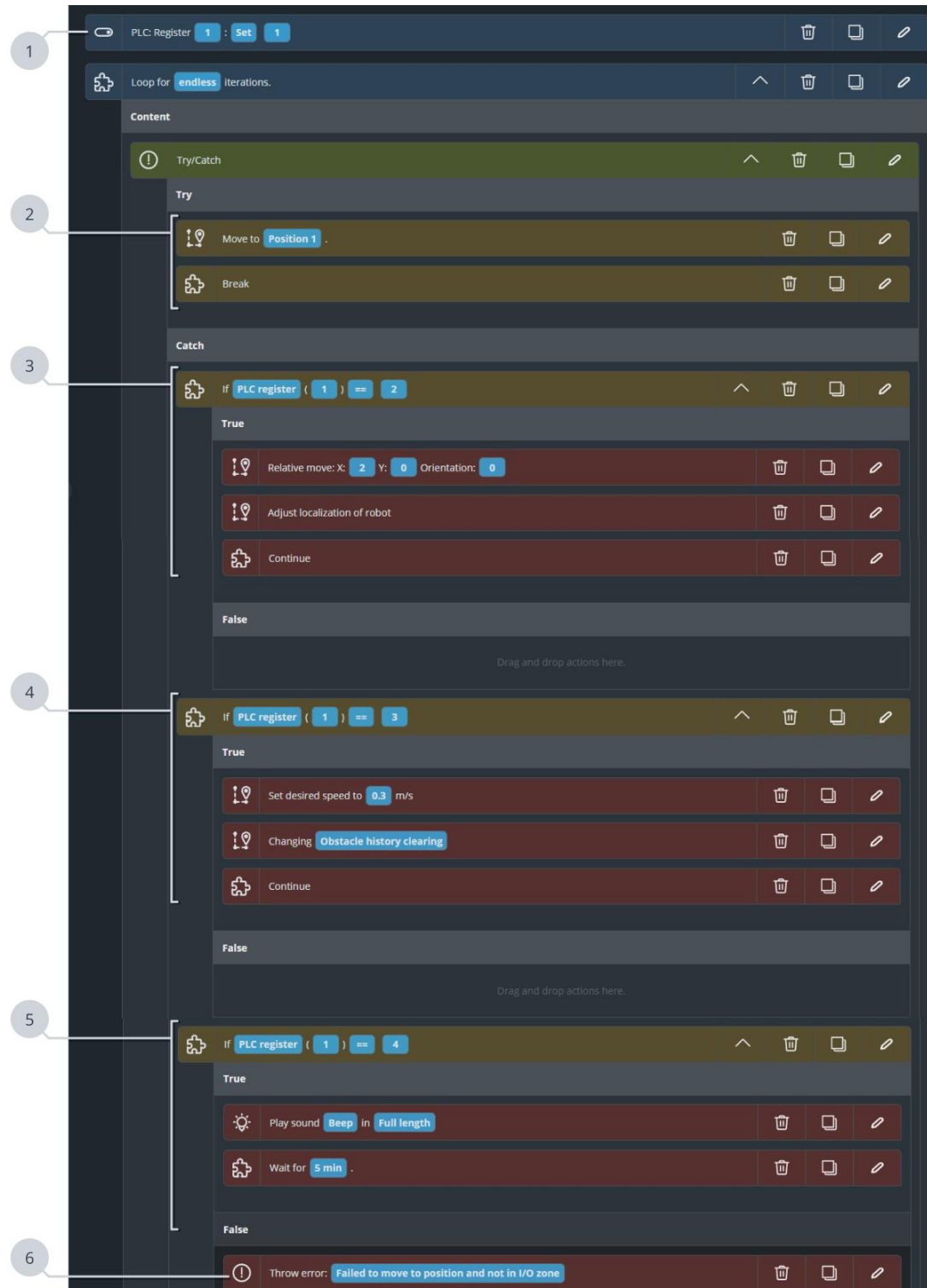


The example mission in [Figure 4.18](#) executes as follows:

- 1 The robot sets PLC register 1 to 1. This register indicates which I/O module zone the robot is in. When it is 1, the robot is not in any zone.
- 2 The robot tries to move to Position 1. If it succeeds, the loop breaks, and the robot continues the mission.

- 3 If the robot fails to move to Position 1 and is inside the first I/O module zone, the robot moves two meters straight forward and adjusts its localization. The mission continues to the top of the loop, and the robot tries to move to Position 1 again.
- 4 If the robot fails to move to Position 1 and is inside the second I/O module zone, the robot slows down to 0.3 m/s. This can make it more reliable at navigating dynamic obstacles. The mission continues to the top of the loop, and the robot tries to move to Position 1 again.
- 5 If the robot fails to move to Position 1 and is inside the third I/O module zone, the robot emits a beep sound to alert nearby personnel to clear its path. The mission continues to the top of the loop, and the robot tries to move to Position 1 again.
- 6 If the robot fails to move to Position 1 and is not inside any of the I/O module zones, it reports an error.

**Figure 4.18** A mission where the robot applies different solutions for each I/O module zone



### Resumable missions with progress tracking

You can use PLC registers or I/O modules to register how far in a mission a robot has progressed. This can be beneficial for the following reasons:

- If you or an external system cancels a mission to resolve an issue or to quickly use the robot for another purpose, when you reschedule the same mission, the robot will continue where it stopped instead of starting over again.
- If you have the robot connected to an external system, the system can follow along in the mission progress and knows at exactly what point the robot failed or was stopped.

The example mission in [Figure 4.19](#) executes as follows:

- 1 PLC register 1 is used to track the progress in this mission. If the PLC register is above the values used in this mission, an operator is asked if it is intended for the robot to start from the beginning of the mission.
- 2 If PLC register 1 is set to 1, the robot picks up a cart and sets PLC register 1 to 2 to indicate it has finished the first action and is now transporting a cart.
- 3 If PLC register 1 is set to 2, the robot places the cart it picked up in the first action and sets PLC register 1 to 3 to indicate it has finished the second action and has placed the cart.
- 4 If PLC register 1 is set to 3, the robot has finished transporting the cart and moves to Position 1 and sets PLC register 1 to 1 to indicate it has finished the mission and can start from the beginning next time the mission is scheduled.



**Figure 4.19** A mission where the robot keeps track of how far it has progressed

The image displays a mission editor interface with four numbered steps, each representing a conditional action based on the state of PLC Register 1.

- Step 1:** An 'If' condition is set to 'PLC register ( 1 ) > 3'. The 'True' branch contains:
  - A 'Prompt user' action with the text: 'Invalid starting register for PLC register 1. Yes: Resets the register to 1. No: ends the mission.'
  - A 'Yes' branch containing a 'PLC: Register 1 : Set 1' action.
  - A 'No' branch containing a 'Return' action.
  - A 'Timeout' branch containing a 'Throw error: ' action.
- Step 2:** An 'If' condition is set to 'PLC register ( 1 ) == 1'. The 'True' branch contains:
  - 'Pick up cart at Cart position 1' action.
  - 'PLC: Register 1 : Add 1' action.
- Step 3:** An 'If' condition is set to 'PLC register ( 1 ) == 2'. The 'True' branch contains:
  - 'Move to Cart loading .' action.
  - 'PLC: Register 1 : Add 1' action.
- Step 4:** An 'If' condition is set to 'PLC register ( 1 ) == 3'. The 'True' branch contains:
  - 'Place cart at Cart position 2' action.
  - 'PLC: Register 1 : Set 1' action.

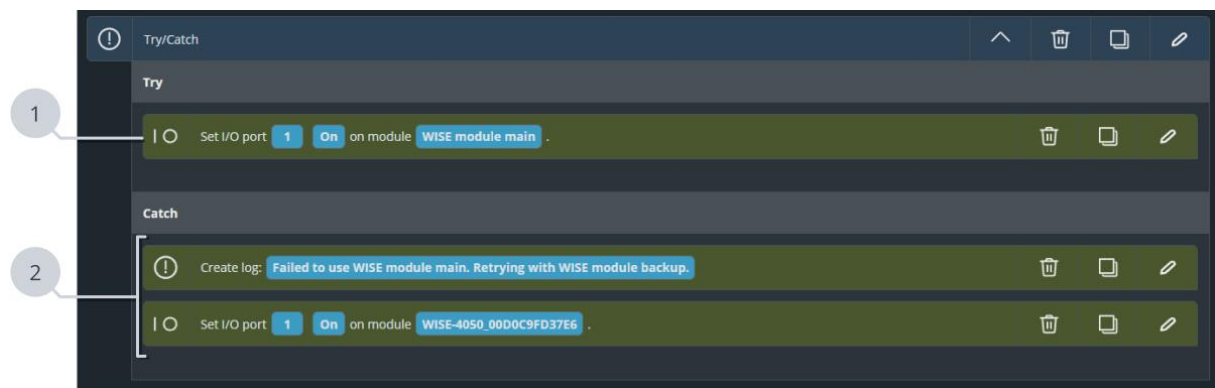
### Use a redundant system

You can make communication more secure by ensuring there are two lines of communication instead of one. This means that you have a redundant second line of communication that is never needed until you have a case where the first line fails.

The example mission in [Figure 4.20](#) executes as follows:

- 1 The robot tries to activate port 1 on a WISE module. This action should have a timeout to avoid the robot trying to activate the WISE module for an undefined length of time.
- 2 If the robot fails to get a response from WISE module main, the robot tries to connect to a backup WISE module and creates an error log to inform that the first WISE module could not be used.

**Figure 4.20** A mission where the robot uses an alternative WISE module if it can not connect to the first one



### Top module cycle counting and resetting

If a top module is not responding as expected, sometimes you can resolve the issue by resetting or recalibrating the top module automatically.

If you have a top module that meets the following requirements, you can make a reset calibration mission that can be run automatically in missions:

- It is possible to trigger the calibration using I/O modules or PLC registers.
- The calibration can be done without user intervention.
- The calibration can be done safely without being monitored.

You can also preemptively try to avoid these issues by making the top module reset regularly after a certain number of uses. This is highly recommended with the MiR pallet and shelf lift top modules.

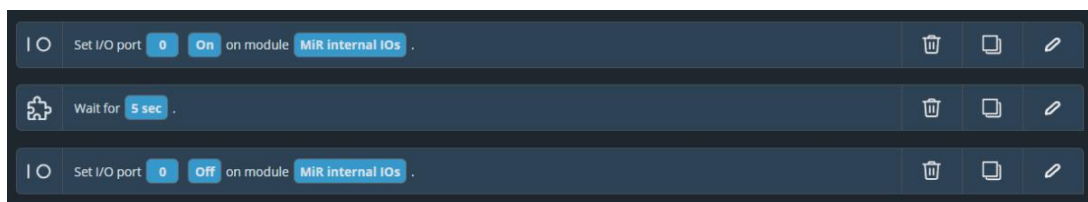
To integrate this into your setup, you must include a counter increment and a counter check in all missions where your top module is activated:

- The counter increment must be included after every action that activates the top module. You can use a Set PLC register action for this.
- The counter check must be implemented before every action that activates the top module. This ensures that the top module is reset right before it needs to be used. You can use an If statement to check if the PLC register used as a counter has exceeded the number of activations before the top module should be reset.

An example of how to implement this in part of a mission is shown in [Figure 4.21](#) and executes as follows:

- 1 PLC register 10 in this example is used as the top module activation counter. The robot checks if the top module has been activated more than 50 times.
- 2 If the top module has been used more than 50 times, the robot makes the top module reset itself and reset the activation counter to 0.

The process to reset your top module can be made into a mission to make it easier to reuse in other missions. For a MiR lift top module, you can make it automatically reset by activating output 0 for at least 5 s.

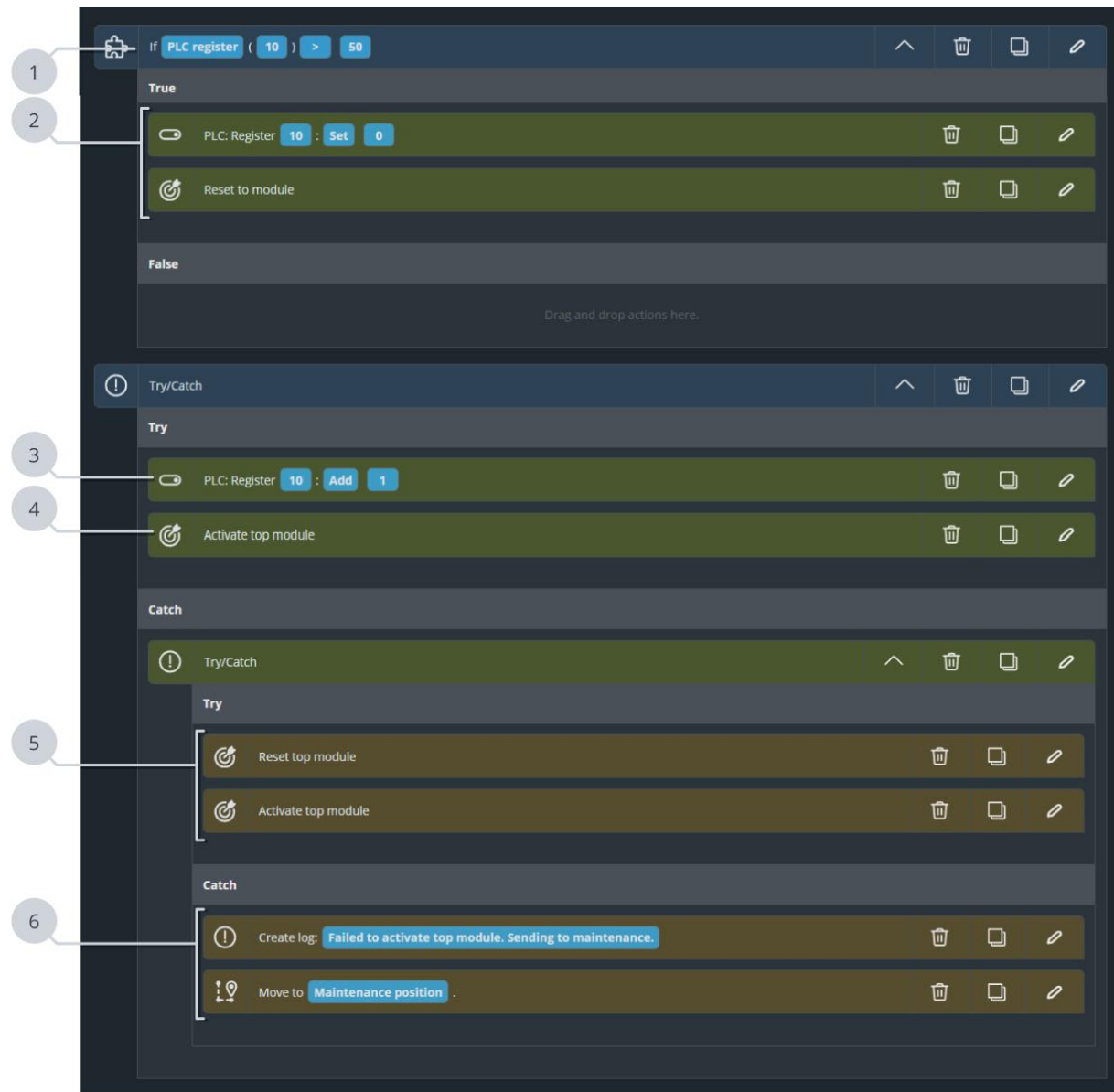


- 3 Before activating the top module, the activation counter is increased by 1. This must also be implemented in all other missions where the top module is activated if you want to count the total number of activations.

- 4 The robot activates the top module. The process to activate the top module can be made into a mission to make it easier to reuse in other missions.
- 5 If the top module does not activate successfully, the robot tries to reset the top module and then activate it again.
- 6 If the top module fails to either reset or activate a second time, an error log is created in case you need assistance from MiR Technical Support, and the robot is sent to maintenance.

You should only send the robot to another position with a faulty top module if it is safe to drive the robot regardless of the top module state.

**Figure 4.21** A mission that handles automatically resetting the top module after 50 uses and sending the robot to maintenance if it continues to fail



## 4.8.6 Schedule missions

### Section overview

Set up system for scheduling missions.

- Determine triggers for each mission.
- Create or modify any external system to schedule missions automatically.
- Create or modify dashboards to help users schedule missions manually.
- Document the chosen triggers for each mission and responsibilities for ensuring each mission runs.

### Plan

For each mission, define:

- What should trigger the mission
- What input the robot needs to complete the mission
- Under which conditions the mission can be executed

Depending on the complexity of these, choose which of the following options best suite your site. You can use a combination of these solutions to trigger different missions:

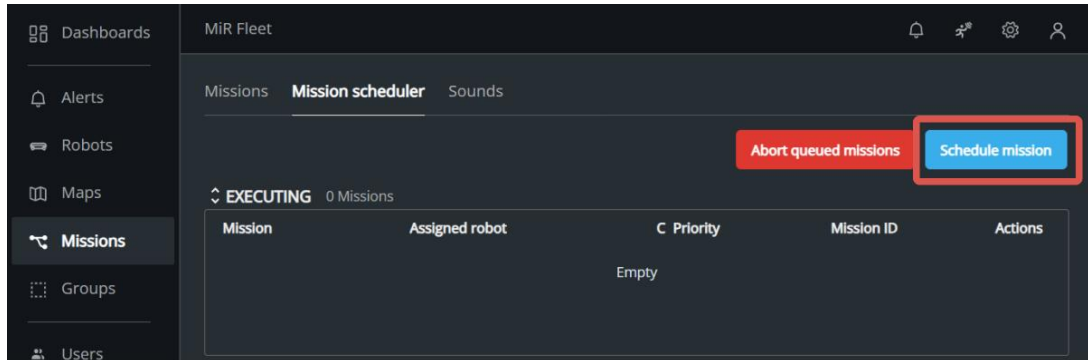
- **Schedule missions directly from the mission scheduler.** This requires no additional setup, but the users that are scheduling missions need to know which mission are relevant for them and easily identify them in the mission scheduler. This solution allows great flexibility, but requires that users are very well trained and informed of the MiR setup. Missions can only be triggered by a user, making the user responsible for ensuring it is appropriate to execute the mission.
- **Create dashboards with buttons to schedule specific missions.** This approach lets you tailor a dashboard for certain users or tasks with the relevant missions and group missions for easier usage. Missions can only be triggered by a user, making the user responsible for ensuring it is appropriate to execute the mission.
- **Create an API client to schedule missions automatically.** This is the most automated option, but requires that you can clearly define when the mission should be scheduled. The client must be able to determine the correct inputs for the mission and evaluate the factors to trigger the mission. This solution requires that you have a well-defined workflow.

## Schedule missions

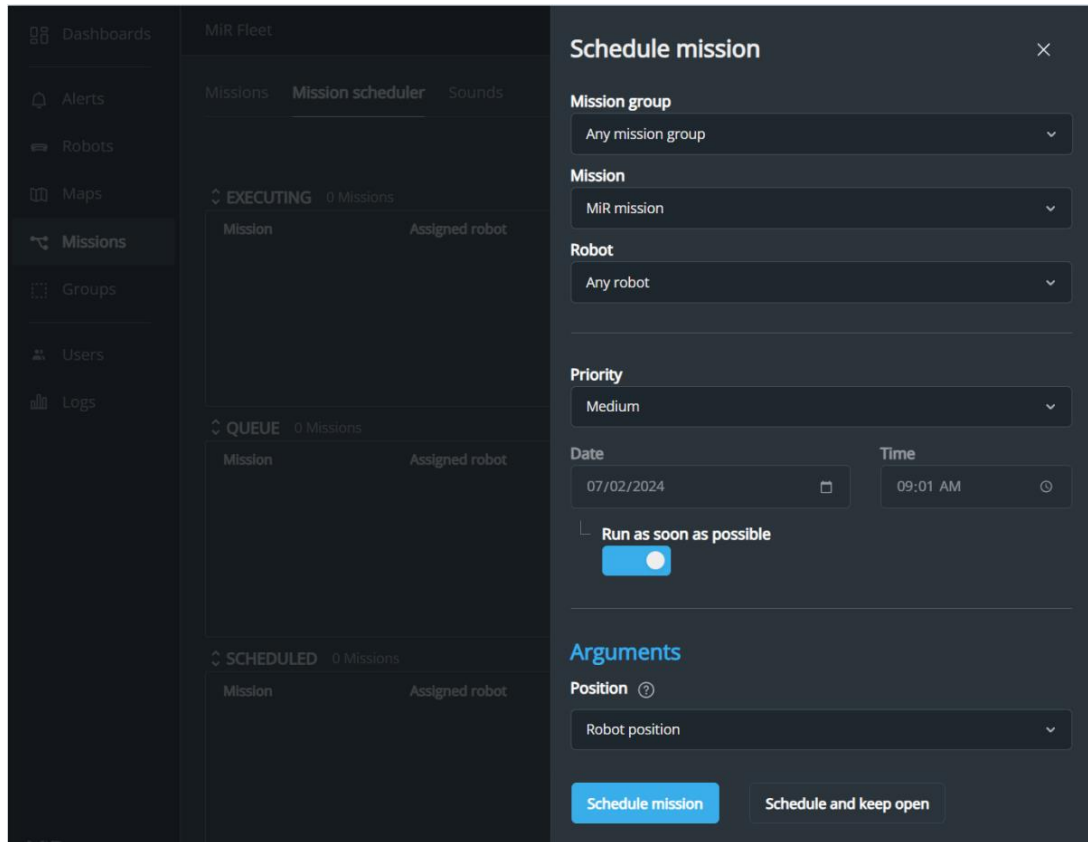
The following sections describe how to schedule missions using three different methods.

### Manually from the mission scheduler

- 1 Go to **Missions > Mission scheduler**, and select **Schedule mission**.



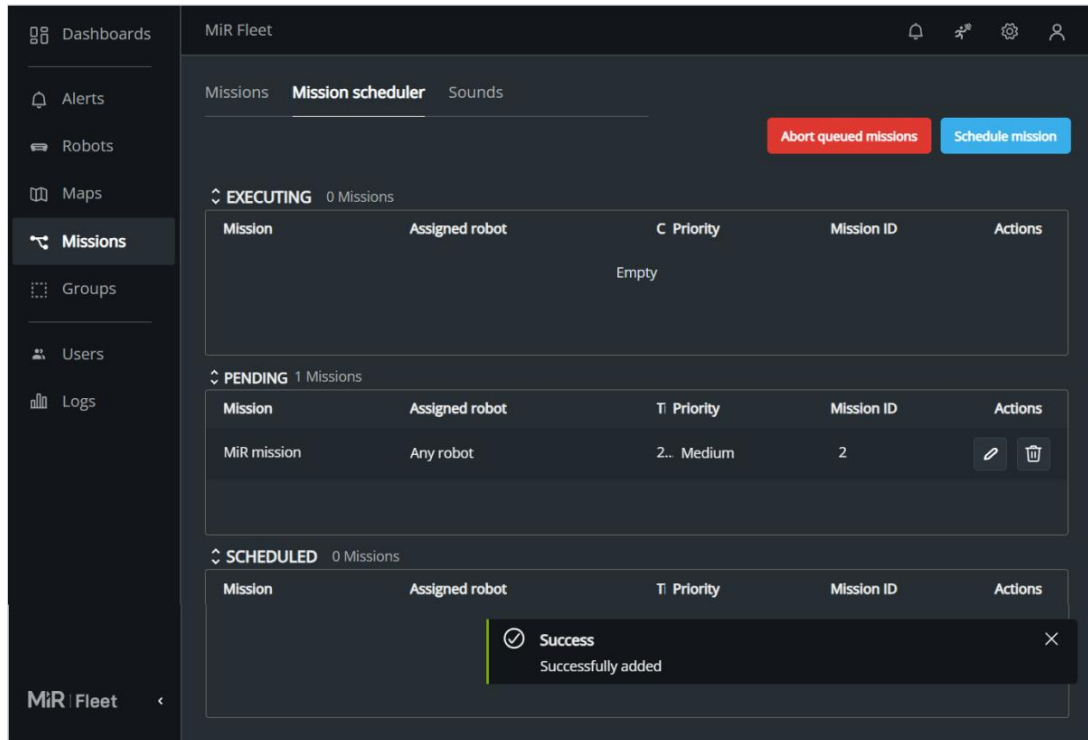
- 2 Fill out the scheduling parameters—see "Schedule mission" on page 85.



- 3 Select **Schedule mission** to schedule the mission and close the dialogue box. Select **Schedule and keep open** to schedule the mission and keep the dialogue box open to schedule another mission.



- 4 Verify that the mission has been added to the mission queue.



### Manually from a dashboard

- 1 Create a dashboard with a Mission widget—see "[Create dashboards](#)" on page 264.
- 2 Configure the widget to schedule a specific mission. Choose if the arguments for the mission are fixed or if the user select the arguments when they schedule the mission.
- 3 Save the dashboard.
- 4 When viewing the dashboard, select the created widget. The mission is added to the appropriate list in the mission scheduler and will be assigned a robot when appropriate.

### Automatically with the API

Use Serial orders in the MiR Fleet Integration API to schedule missions through an external device—see ["Serial-orders" on page 521](#) and ["MiR Fleet Integration API" on page 484](#).

### Test

Set up the scenario where you need to run robot missions. Run the mission using the chosen method and assess:

- Are there situations where you may need to make an exception and use an alternative mission?
- If the mission is manually triggered, is it intuitive to determine the correct mission for the situation?
- If you use serial orders through the API, thoroughly test that the mission has the correct arguments and is triggered at the expected times.
- Are there certain missions that are always executed right after one another? If so, consider nesting these missions into a single mission.
- Is it easy to check if any preconditions for the missions are fulfilled? Is possible, enforce missions to check preconditions such as the top module state or if a load transfer station is occupied.

### Document

Keep the overview of required inputs, triggering events, and mission preconditions. Update the overview with relevant changes.

If you automate any of the inputs or checks, describe how the trigger or data is determined. If you ever make system changes, check that they do not affect

## 4.9 Configure path planning

### **After reading this chapter, you can:**

Adjust the map and robot settings to make robots plan paths and drive between points as intended.

**Create footprints that make robots plan efficient paths that avoid collisions.** See ["Create footprints" on the next page](#).

- Determine the dimensions of all the possible top modules and loads on the MiR robots.
- Agree on a naming scheme that makes it clear which footprint to use.
- Create footprints.
- Document the size and purpose of each footprint.

**Add zones to maps to make MiR Fleet prevent robots from blocking each other and prevent robot from driving into unintended areas.** See ["Manage traffic" on page 411](#).

- Mark points of interest, robot routes, existing traffic conditions, and other relevant factors on a map of the operating area.
- Use Preferred, Unpreferred, and Directional zones to coordinate the robot routes.
- In all areas the robot must never enter, add Forbidden zones.
- At all crossings, docking stations, and narrow corridors, use Limit-robots zones to prevent robots blocking each other.
- Document all of the zones and their purposes.

**Adjust the robot settings to change how the robot drives and plans paths.** See ["Configure settings" on page 428](#).

- Define how much robots can deviate in their path planning.
- Determine if you need to adjust the camera filtering settings.
- Apply settings changes to all other relevant robots.

## 4.9.1 Create footprints

### Section overview

Create footprints that make robots plan efficient paths that avoid collisions.

- Determine the dimensions of all the possible top modules and loads on the MiR robots.
- Agree on a naming scheme that makes it clear which footprint to use.
- Create footprints.
- Document the size and purpose of each footprint.

Use footprints to make the robot plan paths with a safe distance to obstacles to prevent collisions and Protective stops.

### Plan

When you design footprints keep the following points in mind:

- Footprints should be 10–15% larger than the area the robot and its load or top module occupy
- If the robot picks up a load that extends the robot's footprint, the robot must use a footprint suitable for that load.
- Footprints should be larger than the Protective field at standstill and wider than the widest Protective field—see ["How Protective fields affect the footprint size" on page 406](#). Making the footprint smaller than this will result in the robot planning paths that bring the robot close enough to obstacles to trigger a Protective stop.
- If you have a robot with a MiR hook, the footprint does not need to be increased when the robot is towing a cart. The robot will navigate with the cart using the dimensions provided for the cart—see ["Carts" on page 15](#).
- The interface comes with default footprints for MiR Shelf Lift and MiR250 Shelf Carrier. If you are using a shelf with dimensions not supported by MiR, you must create your own footprint that fits the dimensions of the shelf.
- Increase the footprint size if the robot often plans paths where the robot enters Protective stop because it drives too close to an obstacle, and it is not the obstacle that moves toward the robot. This will make the robot require more space around it to operate, and it will plan its paths with a greater distance to obstacles.

- Decrease the footprint to make the robot fit through more narrow spaces. If the robot plan paths around areas that you want it to drive through, make the footprint smaller until it fits. Avoid making the footprint smaller than the default footprint. This can result in the robot entering Protective stop while driving because it plans too close to obstacles.

The footprint only affects how the robot plans its paths and navigates around obstacles. If you have issues with the robot entering Protective stop too late or too soon, adjust the Protective fields—see ["Adjust Protective fields" on page 457](#).

**WARNING**

Driving with a tall payload under low hanging fixtures can cause a collision that tips over the load, causing serious injury to personnel and damage to the robot and to equipment.

- Define new footprints for the various heights that the robot and its load can have to minimize the risk of the load colliding with low hanging fixtures.
- Mark areas with known low hanging fixtures as Forbidden zones.

**WARNING**

The default Protective fields are not configured for loads or top module that extend over the front or rear of the robot. If personnel walk in front of a robot with a top module or load the extends over the front of it, the robot may not stop in time to prevent a collision.

- Avoid extending the physical dimensions in front of or behind the robot.
- Mark all areas where the robot drives with an unsafe load as operating hazard zones.
- Modify the Protective field sets if necessary. Extending the footprint only affects the planner algorithm, not the Personnel detection safety function that stops robots to avoid collisions.

**Footprint variations**

Create a footprint for each:

- Top module or load that exceeds the width or length of the robot.
- Top modules with moving parts that can extend beyond the physical dimensions of the robot.
- Top module or loads that are high enough to prevent the robot from operating in specific areas due to low hanging fixtures.

If robots drive with many different sized loads, consider making a footprint for the largest possible case, and always use this footprint for all the load sizes.

#### **How Protective fields affect the footprint size**

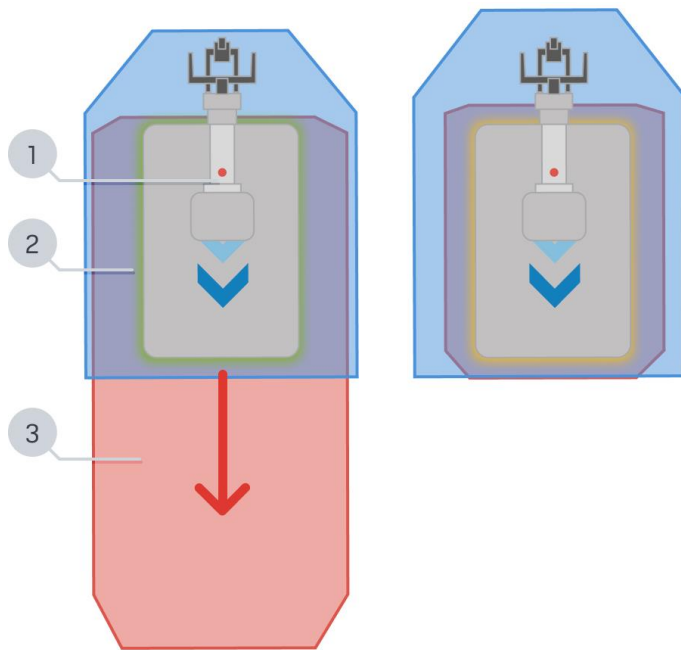
Protective fields and footprints are both areas relative to the robot that affect the robot's driving behavior. The footprint is used for navigation and path planning, whereas the Protective fields are a safety feature that ensures the robot stops before colliding with an obstacle.

During commissioning, you must verify the Protective fields via brake testing—see ["Test braking distance" on page 448](#). Do this before defining any footprints.

Once the Protective fields are approved, create footprints based on the verified Protective fields.

If you update the Protective fields, you must reevaluate the footprints based on the new Protective fields.

The Protective fields change with the robot's speed, and the footprint is constant for all robot speeds and corresponding Protective fields. This means that the footprint must be sized appropriately for the range of Protective fields.

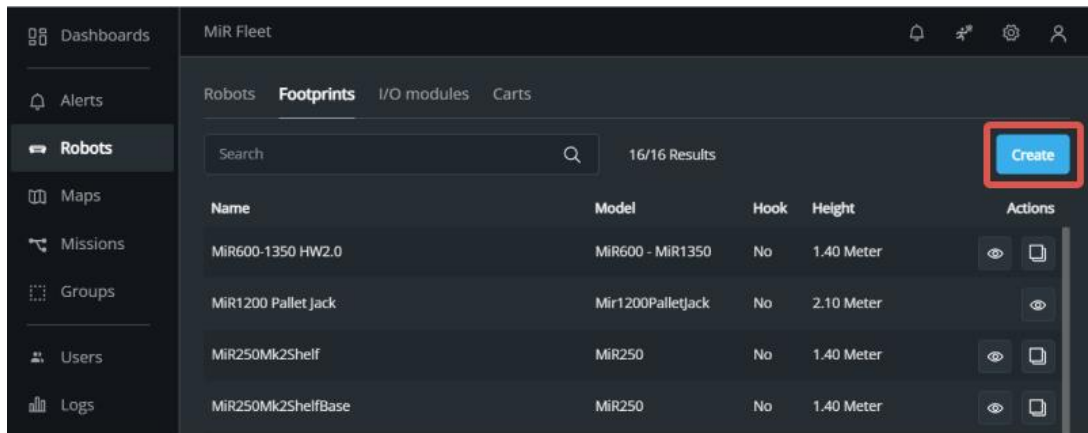


Pos.	Description	Pos.	Description
1	Physical dimensions: As a minimum, adjust the length, width, and height of the footprint to be 10–15% larger than the physical footprint of the top module and its expected payload.	2	Navigation: Adjust the length and width of the footprint to be as small as possible for both safe and agile autonomous driving. If the footprint is too small, the robot will navigate too close to a person or an object and go into Protective stop.
3	Protective field: Make the footprint wider than the widest Protective field, and longer than the Protective field at standstill.		

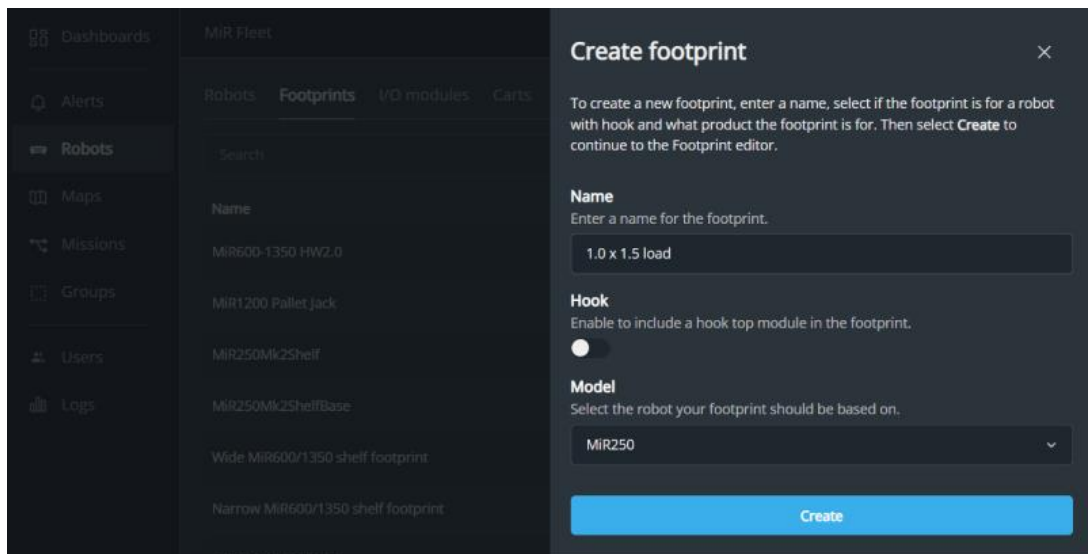
### Create a footprint

To create a custom footprint, follow these steps:

- 1 In the MiR Fleet interface, go to **Robots > Footprints**, and select **Create**.



- 2 Enter a name for the footprint, and select the relevant options for the robot application the footprint will be used for. Use a naming standard that makes it clear for which robot the footprint is used for and under which conditions.

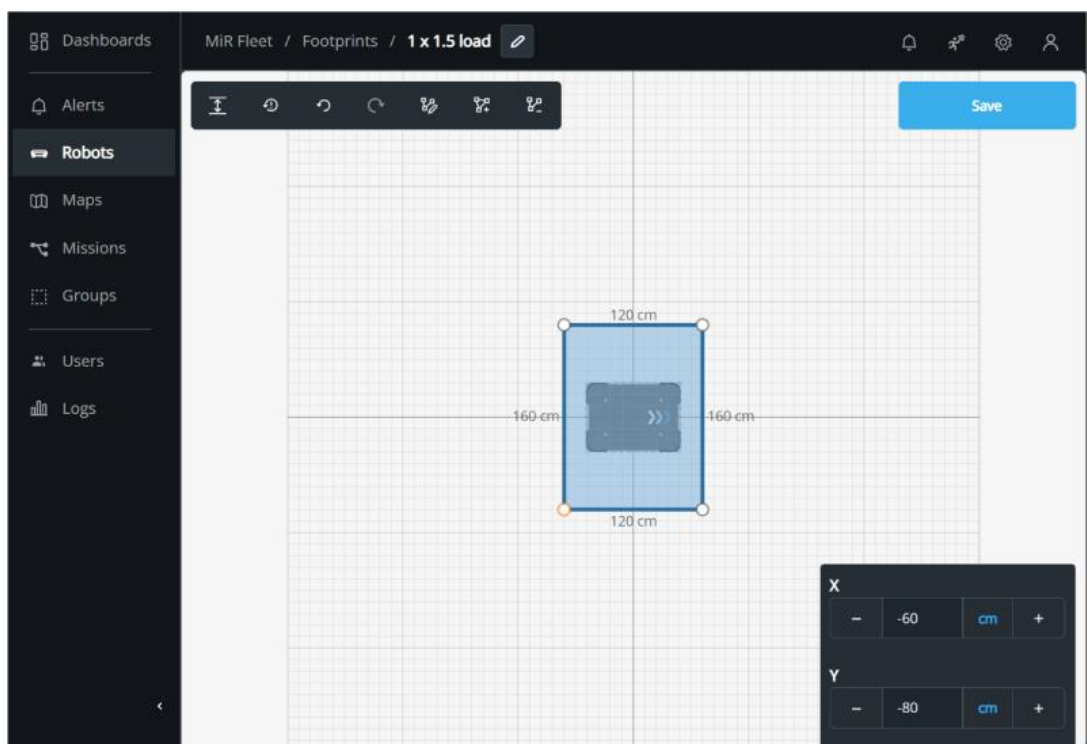




3 Use the following tools to create the footprint:

- **Edit height:** Set the height of the footprint. This is relevant if there are low hanging obstacles the robot must drive under
- **Select:** Drag existing points to change the shape of the footprint.
- **Add point:** Add new points to the existing edges.
- **Remove point:** Remove existing points.

The footprint must be convex and larger than the base robot's physical dimensions. If at anytime the footprint becomes invalid, the edge that needs to be changed turns red.

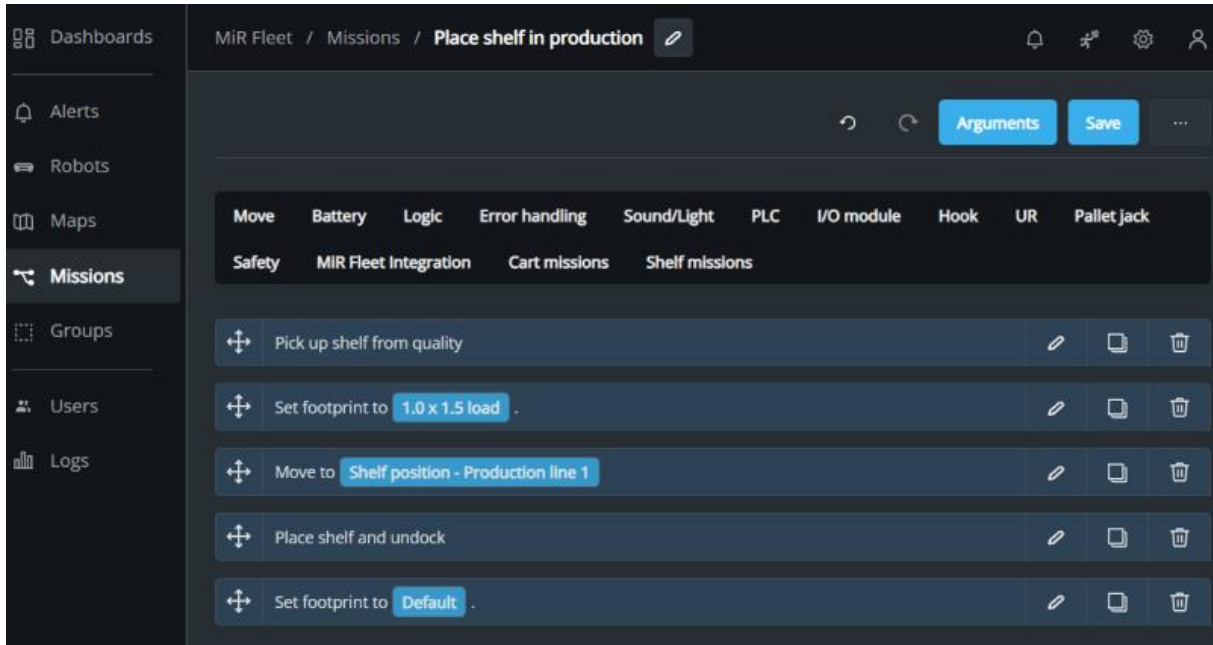


4 Select **Save** when you are finished editing the footprint.

### Set the footprint

If you have installed a top module on the robot, change the default footprint to a one that matches the robot's dimensions with the top module—see ["Active and default footprint"](#) on page 29.

In missions where the robot picks up loads that affect the robot's physical dimensions, use a Set footprint action to set the correct footprint that matches the load size. When the robot removes the load, use a Set footprint action to restore the default footprint.



The robot does not reset the footprint to the default when it ends, aborts, or fails missions. If a robot fails or abort a mission and its load is removed manually, you must run a mission to set the correct footprint for the robot.

Consider starting all missions with the footprint you expect the robots to have to avoid the robot operating with unintended footprints.

## Test

Set the robot to use the footprint in the relevant missions, and run the missions. Verify the following:

- Check that the robot can plan and execute suitable paths with the footprint.
- Test that the footprint is not so large that it compromises the robot's maneuverability when driving slowly, for example, when docking. Consider changing footprints if necessary
- Verify that the robot changes to the correct footprint when executing relevant missions. You can see the outline of the robot's footprint when viewing the robot in a map in the interface.

## Document

- For each footprint, describe the footprints purpose, size, and when to use it.
- Document any changes made to footprints. Asses footprint sizes regularly to make sure they are still optimal for the robot loads and environment.

### 4.9.2 Manage traffic

#### Section overview

Add zones to maps to make MiR Fleet prevent robots from blocking each other and prevent robot from driving into unintended areas.

- Mark points of interest, robot routes, existing traffic conditions, and other relevant factors on a map of the operating area.
- Use Preferred, Unpreferred, and Directional zones to coordinate the robot routes.
- In all areas the robot must never enter, add Forbidden zones.
- At all crossings, docking stations, and narrow corridors, use Limit-robots zones to prevent robots blocking each other.
- Document all of the zones and their purposes.


Implement traffic management to avoid congestion between MiR robots. As a best practice, aim to always have robots moving, even if it means robots cannot always take the most direct route.

#### Plan

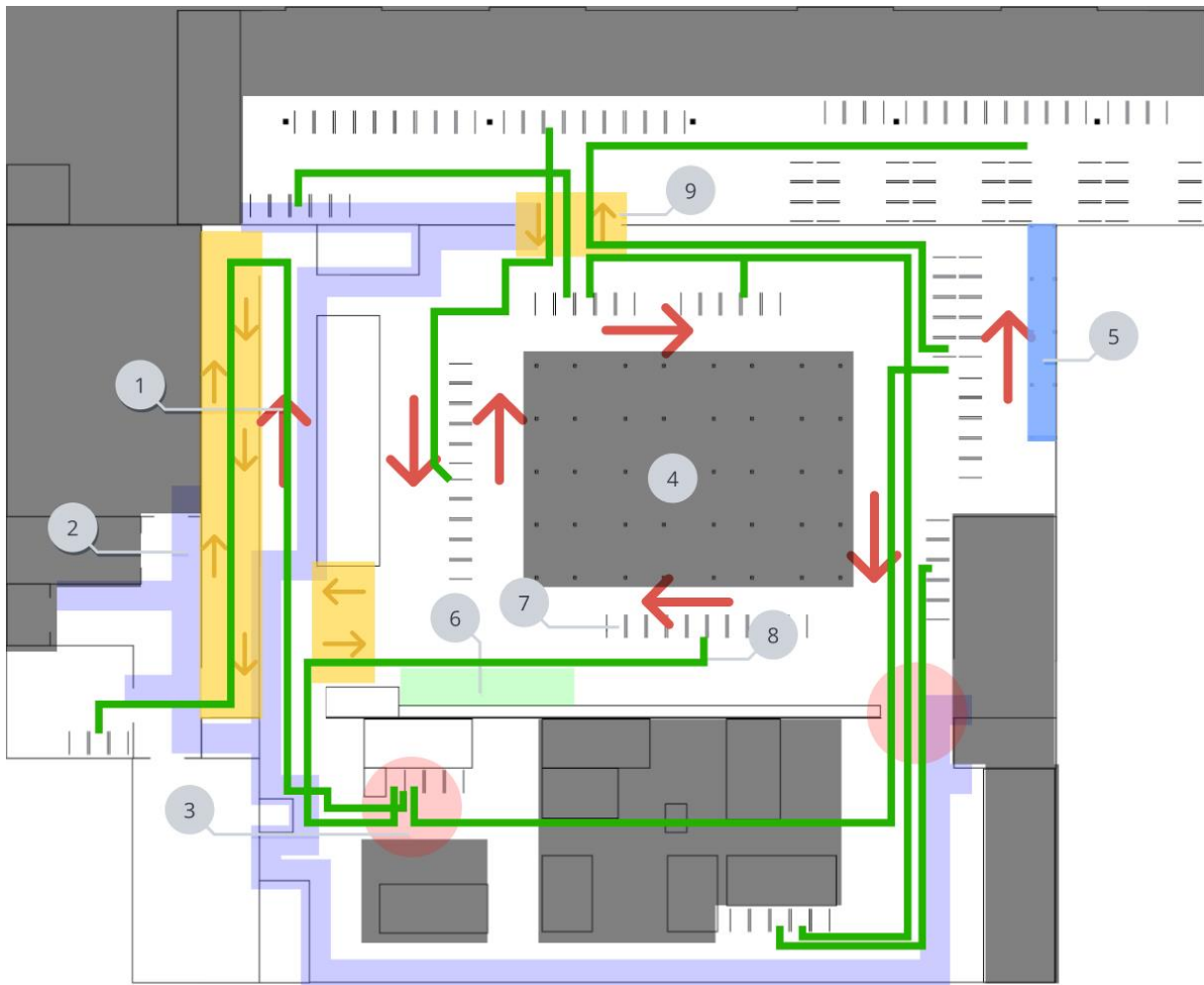
Before implementing zones to manage traffic, mark the following on a floor plan or blueprint of your site. You can export the floor plan of your MiR map from the interface.

**Table 4.8** Descriptions of points of interest in your site floor plan

Pos.	Description	Pos.	Description
1	<p>↑ Existing traffic rules: When you implement directional zones, use existing traffic rules—see <a href="#">"Making circular traffic flow"</a> on page 422.</p>	2	<p>▮ Pedestrian routes: This identifies areas that should prioritize personnel—see <a href="#">"Marking pedestrian routes"</a> on page 421.</p>
3	<p>● Areas with crossing traffic: This includes existing traffic conditions, and any you predict based on the direct routes robots will take without any map zones—see <a href="#">"Making circular traffic flow"</a> on page 422 and <a href="#">"Crossings and lanes"</a> on page 424.</p>	4	<p>■ Areas robot never need to go: This identifies areas to be marked with Forbidden zones—see <a href="#">"Marking forbidden areas"</a> on page 420.</p>
5	<p>■ Difficult to detect obstacles: This includes:</p> <ul style="list-style-type: none"> <li>• Glass</li> <li>• Holes</li> <li>• Low hanging obstacles</li> <li>• Reflective material</li> </ul> <p>These areas should be marked with Forbidden zones—see <a href="#">"Marking forbidden areas"</a> on page 420.</p>	6	<p>■ Areas you expect to install charging stations: This identifies docking areas—see <a href="#">"Docking areas "</a> on page 425.</p>

Pos.	Description	Pos.	Description
7	Load transfer stations for MiR robots: This identifies docking areas—see <a href="#">"Docking areas "</a> on <a href="#">page 425</a> .	8	Most direct routes for robot to take between positions: This illustrates how the robot will travel without any map zones. To change how the robot plans paths, see <a href="#">"Making circular traffic flow"</a> on <a href="#">page 422</a> and <a href="#">"Creating preferred paths"</a> on <a href="#">page 423</a> .
9	 Areas with multi-directional traffic: Measure if there is only space for one robot, or if two robots can cross each other—see <a href="#">"Making circular traffic flow"</a> on <a href="#">page 422</a> and <a href="#">"Crossings and lanes"</a> on <a href="#">page 424</a> . For your robot's space requirements, see the <i>Space Requirements</i> guide for your robot. You can find this guide on <a href="#">MiR Support Portal</a> .		

**Figure 4.22** Identification of points of interest in an example site



### Naming scheme

Before creating any zones or positions, ensure you have planned a naming scheme. Naming schemes for zones are often based on location or purpose of the zone.

### Traffic rules

Define and implement a behavioral hierarchy between traffic elements and description of correct behavior in certain situations and areas.

The following is an example you can modify to your site. Use map zones, floor markings, signs, and API integrations to enforce, clarify, and ease the implementation of these rules.

Traffic element	Rules
Pedestrians	<ul style="list-style-type: none"><li>• Highest priority in all scenarios.</li><li>• Always have the right of way at designated routes.</li><li>• Must follow standard pedestrian rules at intersections, traffic signals, etc.</li></ul>
Automated guided vehicles (AGV)	<ul style="list-style-type: none"><li>• Second priority after pedestrians.</li><li>• Must adhere to traffic signals.</li><li>• Must slow down or stop if a pedestrian is in the path.</li><li>• Must give way to pedestrians at intersections</li></ul>
Autonomous mobile robots (AMR)	<ul style="list-style-type: none"><li>• Similar priority to AGVs.</li><li>• Must follow designated paths or lanes.</li><li>• Must adhere to traffic signals.</li><li>• Must yield to both pedestrians and AGVs.</li></ul>
Forklifts/trains	<ul style="list-style-type: none"><li>• Lower priority compared to AGVs and AMRs.</li><li>• Must operate in areas separate from AGVs and AMRs whenever possible.</li><li>• Must follow standard forklift safety guidelines.</li><li>• Must yield to pedestrians, AGVs, and AMRs.</li><li>• Must slow down in shared spaces and intersections.</li><li>• Must never overtake AGVs or AMRs.</li><li>• Must utilize dedicated parking areas.</li></ul>

Traffic element	Rules
Intersection rules	<ul style="list-style-type: none"><li>• Pedestrians always have the right of way.</li><li>• Forklifts must yield to AGVs, AMRs, and pedestrians.</li><li>• Vehicles and robots must come to a complete stop at designated stop signals or signs.</li><li>• Controlled intersections may have traffic lights or other signaling devices.</li></ul>
Straight corridor rules	<ul style="list-style-type: none"><li>• AGVs and AMRs must maintain a consistent speed within the corridor.</li><li>• Forklifts must use designated lanes or pathways.</li><li>• Pedestrians have the right of way, and AGVs and AMRs must slow down or stop when encountering pedestrians.</li><li>• Vehicles are not allowed to overtake each other if driving in the same direction.</li></ul>
Emergency situations	<ul style="list-style-type: none"><li>• All automated vehicles and robots should be set up to respond to emergency signals or alarms.</li><li>• Emergency exits and evacuation routes should be cleared for pedestrians.</li><li>• Vehicles and robots should have mechanisms for immediate stop in emergency situations.</li><li>• The evacuation feature in MiR Fleet could be set up in accordance with the site-specific emergency procedures.</li></ul>

## Create zones

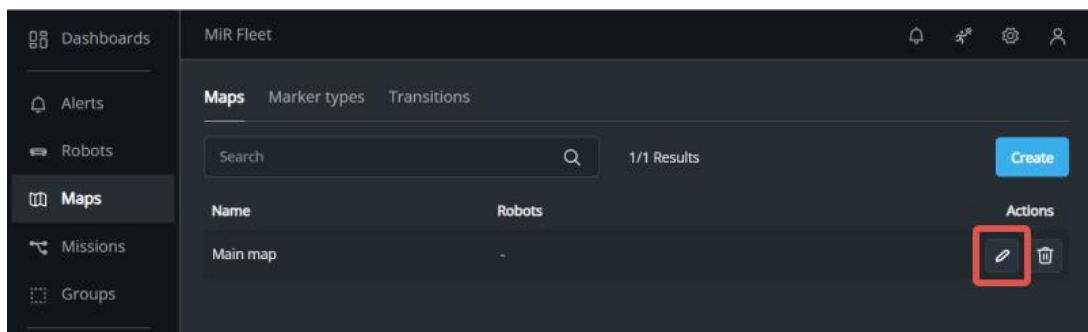
When you create zones, follow these guidelines:



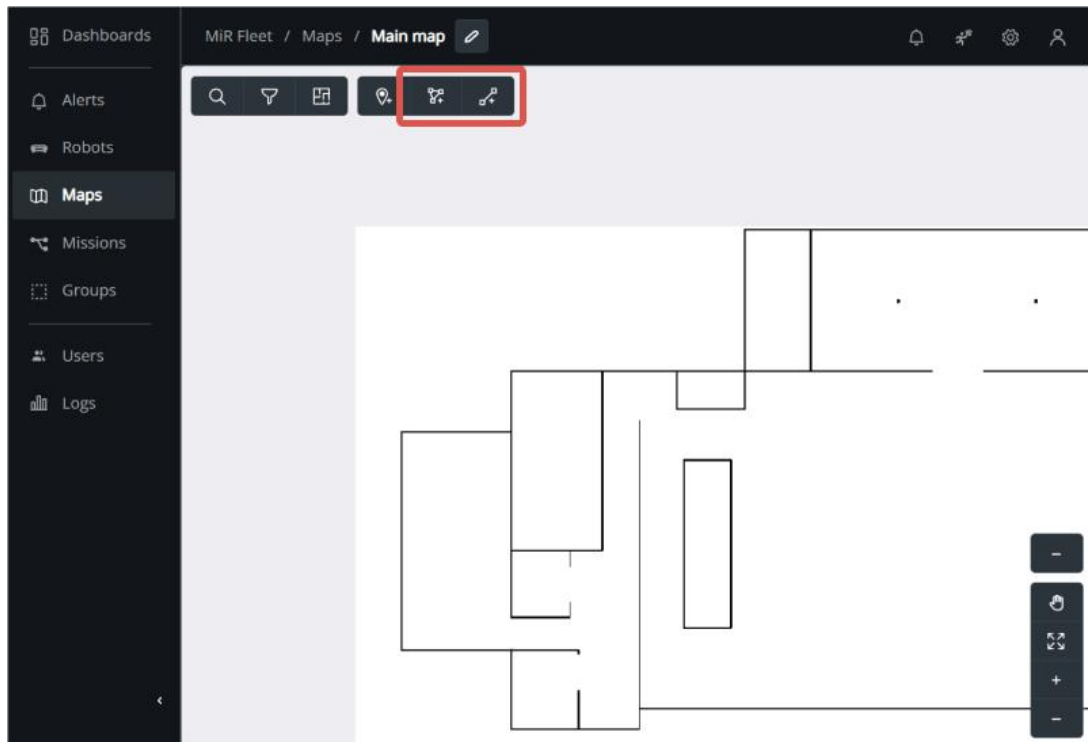
- Keep zone shapes straight and rectangular.
- Name zones according to your defined naming scheme.
- Avoid overlapping zones, especially if the zones are conflicting. For example, overlapping Speed zones with two different speed settings.

To create a zone, follow these steps:

- 1 Go to **Setup > Maps**, and select **Edit**  for the map you want to add zones to.

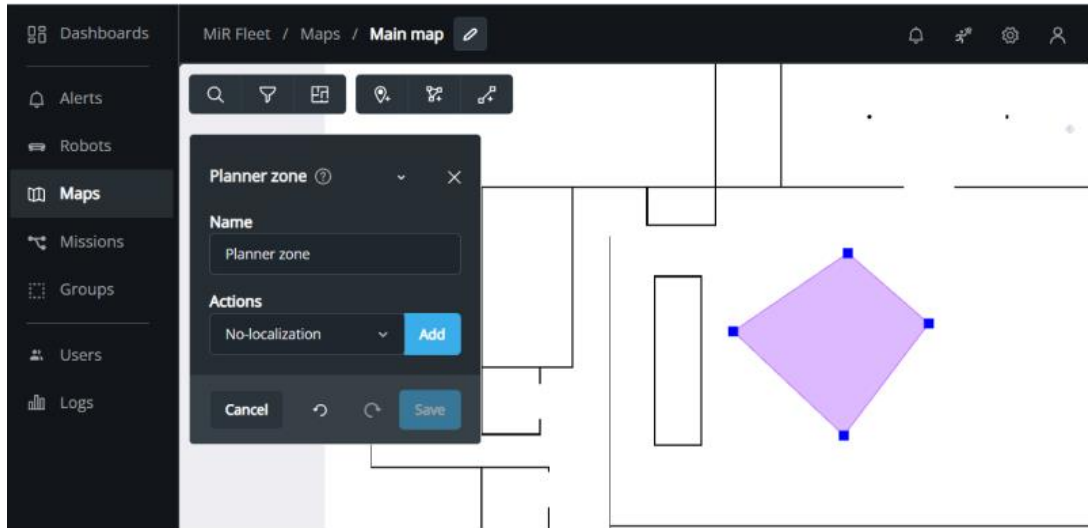


- 2 In the toolbar, select **Draw a new line** or **Draw a new shape**.

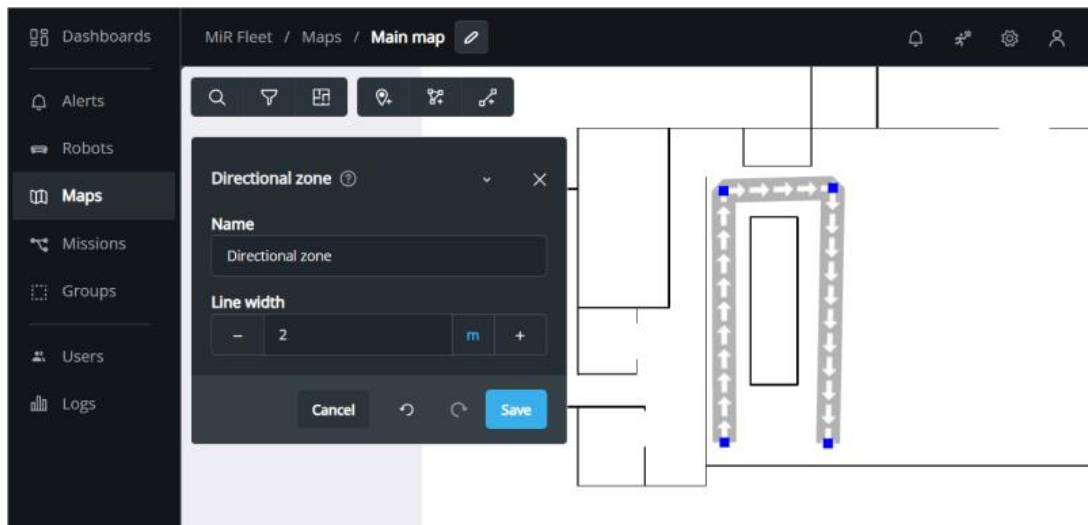


- 3 Select the type of zone you want to create.

- 4 If you selected **Draw a new shape**, select points on the map to create a polygon shape for the zone, and modify the zone settings.



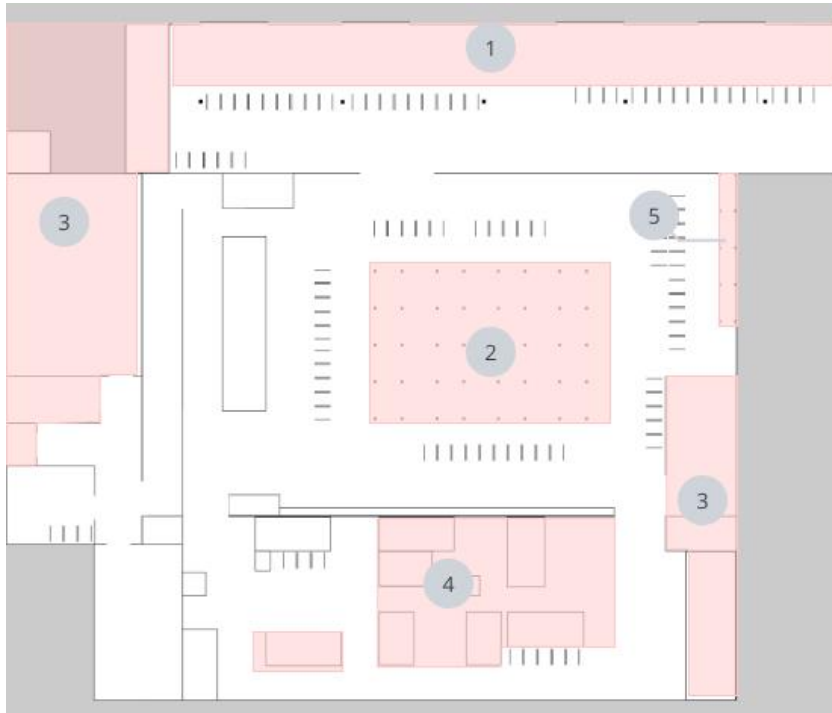
If you selected **Draw a new line**, select points on the map to create a line for the zone, and modify the line width.



### Marking forbidden areas

Mark any areas that MiR robots should never enter as Forbidden zones on the map. [Figure 4.23](#) is an example of common areas to mark as Forbidden zones. Always use Forbidden zones in areas that are described under *Foreseeable misuse* in the user guide or integrator manual for your robot application.

**Figure 4.23** Example of where you may apply Forbidden zones



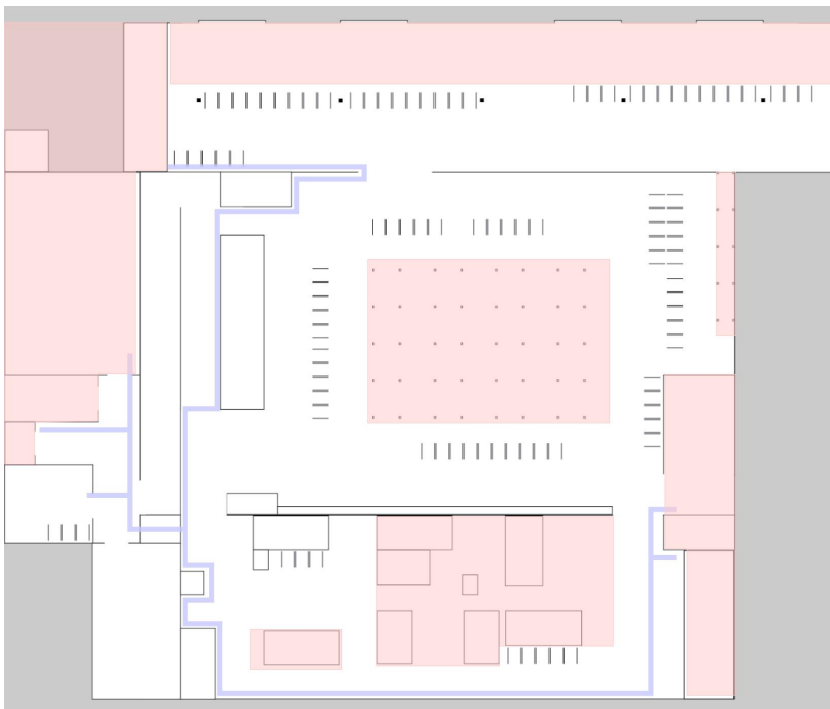
Pos.	Description	Pos.	Description
1	The inbound and outbound delivery area is busy when a truck arrives. Robots do not help loading the trucks and should not enter the area.	2	The long term storage area has narrow isles that are only intended for forklift drivers. Robots in this area would only block forklift drivers.

Pos.	Description	Pos.	Description
3	There are no robot-related tasks in these areas. You can reduce the robot's planning time by marking unused areas with Forbidden zones.	4	There is no reason for the robot to enter the production area, and there are often many dynamic obstacles that will block the robot's path and force the robot to plan around the area anyway.
5	Mark items robots cannot detect, such as glass, overhangs, or holes, as Forbidden zones to avoid robots driving into them.		

### Marking pedestrian routes

To keep a site's pedestrian pathways clear of robots, you can either mark them as Unpreferred zones or Forbidden zones. With Unpreferred zones, robots can still drive over the pedestrian route if an obstacle is blocking its path.

**Figure 4.24** Example of an Unpreferred zone that marks the pedestrian route at the site



### Making circular traffic flow

Use a circular flow to reduce the number of areas where robots may cross paths and risk blocking each other.

Use Directional zones to force the robot traffic in certain directions. [Figure 4.25](#) is an example site with many circular routes to prevent robots meeting head on in aisles where they cannot pass each other.

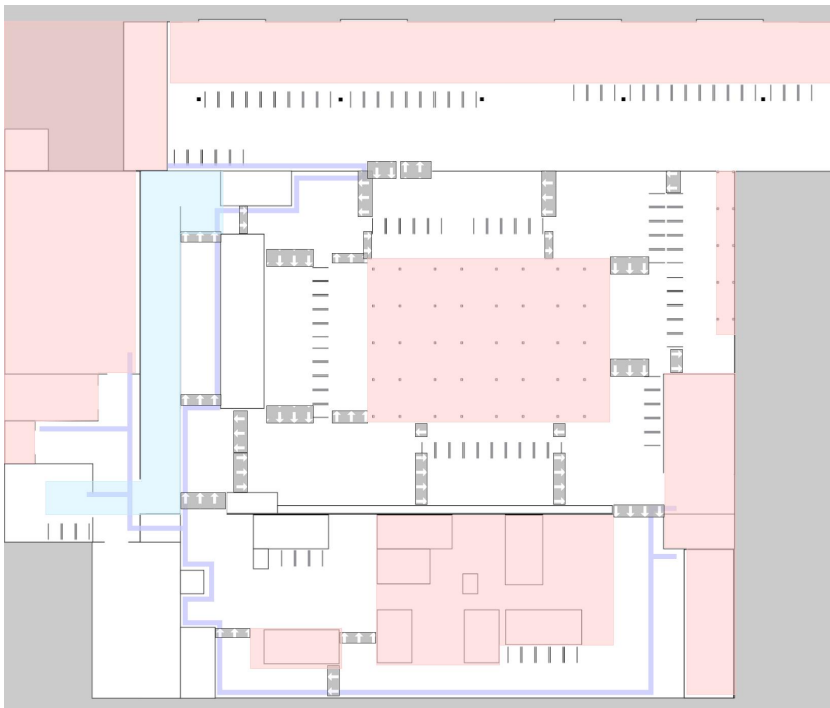
You can use two opposing Directional zones to make two lanes in opposite directions in wide aisles to make it easier for robots to pass each other—see ["Crossings and lanes" on page 424](#).

To make sure robots can still path plan efficiently and maneuver well in docking areas, only place Directional zones in the intersections. This also keeps the map more readable and easier to update, while still having the same effect on the robot's global path.

For all areas where robots cannot pass each other, and you cannot set up a circular route, use Limit-robots zones to prevent robots from entering the area from either end at the same time. Make enough room at either end that a waiting robot does not block a robot exiting the zone.

Create Limit-robots zones so the space between each is greater than the resource assignment distance—see ["Resource handling" on page 87](#). This helps prevent deadlocks.

**Figure 4.25** Example of a site where Directional zones are used to make a circular traffic flow



## Creating preferred paths

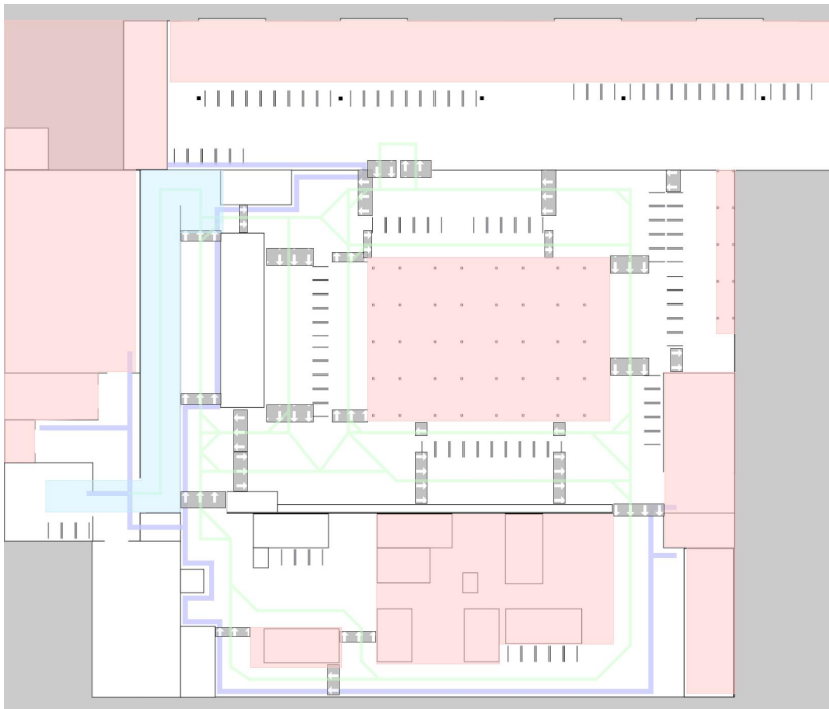
Use Preferred zones to mark the paths you want robots to travel on, and encourage robots to use the center of the lane.

This can be especially useful in areas where you have two Directional zones in opposite directions and you want to ensure that if two robots are passing each other, they are both already positioned to provide enough room for the other robot to pass.

If you want robots to take turns smoothly and save a little time, we recommend adding angled lines at the corners, so robots have a faster option if there is space.

To modify how much robot are allowed the deviate from the Preferred path if there is an obstacle in the way, see ["Configure settings" on page 428](#).

**Figure 4.26** Example of a site where Preferred zones are used to center robots in the lanes



## Forklifts

If MiR robots and forklift drivers operate in the same area, forklift drivers must be trained to behave safely around robots. Apply the following guidelines:

- Robots always have the right of way.
- Never expect robots to avoid raised forks or empty pallets.
- Park forklifts outside robot operation areas. Place Forbidden zones where forklifts are parked.
- Make forklifts and MiR robots apply the same directional traffic rules.
- Make MiR robots play sounds at intersections or other places with reduced visibility.

## Crossings and lanes

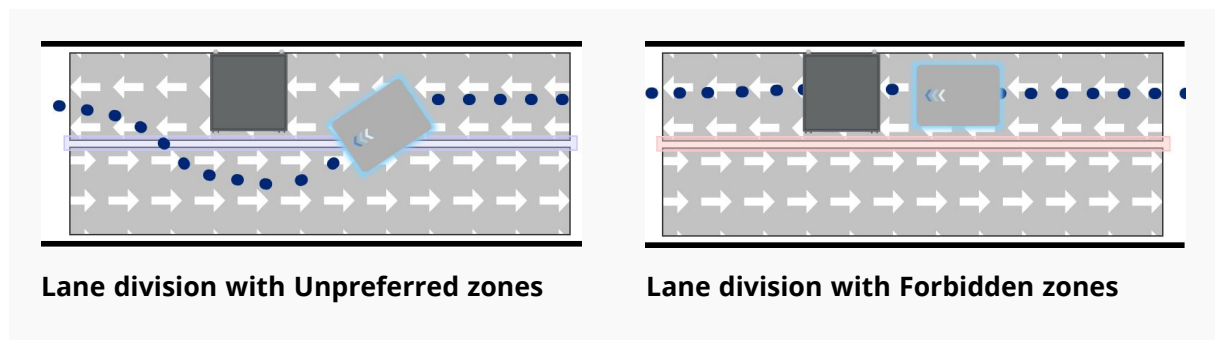
If you have a site where you use two lanes in opposite directions, consider whether or not you want to enable robots to cross over into the other lane:

- If you want to discourage robots from going into the opposite lane so it only does so when an obstacle is blocking its main path, use an Unpreferred zone to split the two lanes.



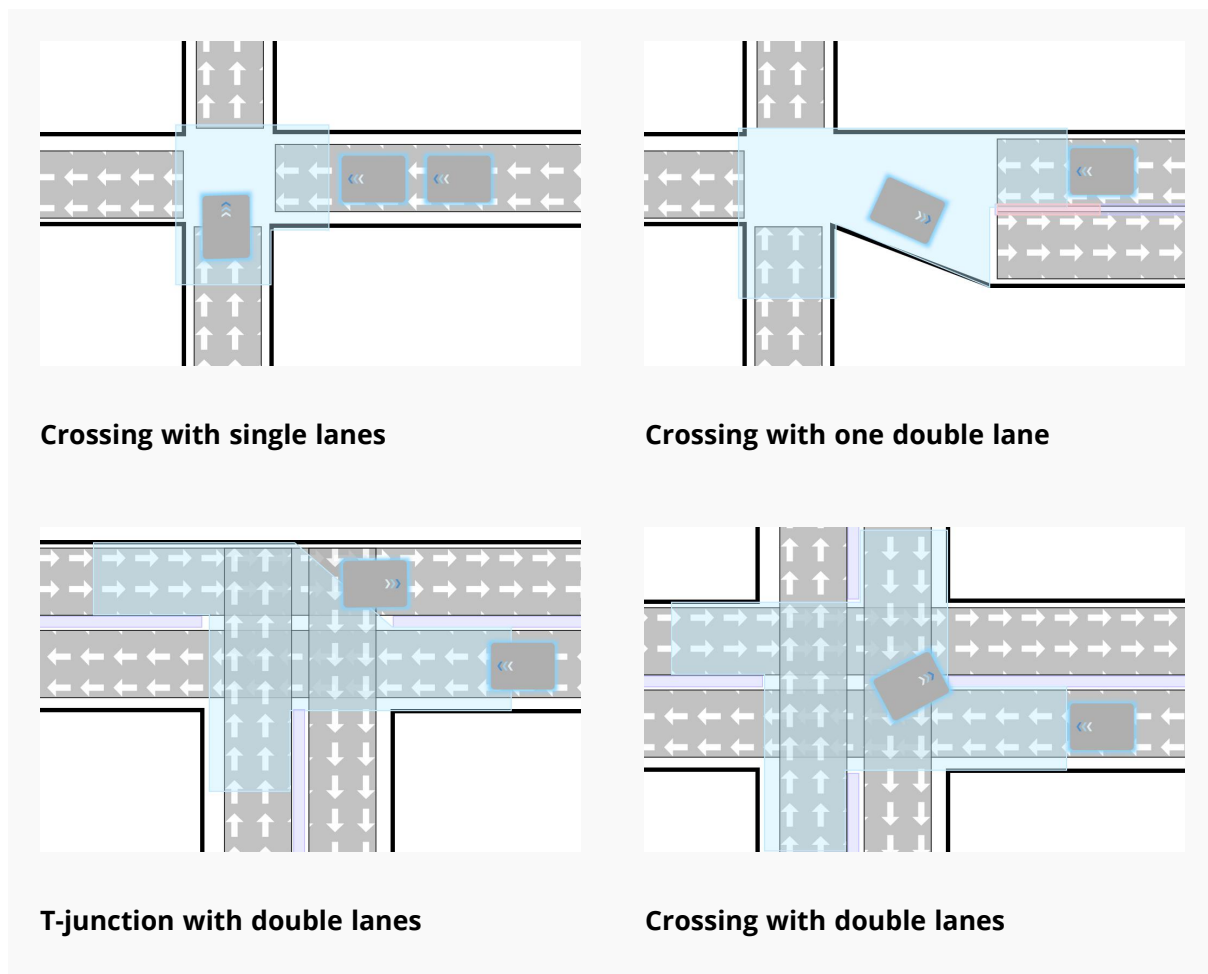
Robots can sometimes drive against Directional zones if an obstacle is blocking their path. For more information, see *MiR250, MiR600, and MiR1350 Technical Guide*. You can find this guide on [MiR Support Portal](#).

- If you never want robots to drive onto the opposite lane, use a Forbidden zone. If an obstacle gets in its path, robots will either report an error or wait until the obstacle is moved, depending on the robot's system settings.



For all of the crossings on your site, you want to ensure that only one robot at a time is in the middle of the crossing to prevent deadlocks. You can do this using a Limit-robots zone that expands slightly into the entry lanes.





The principle of using a Limit-robots zone can be applied to any kind of crossing, it just needs to be expanded to fit the number of entries or exits in the crossing. Add Unpreferred zones or Forbidden zones to better control the robots entering and exiting the crossing.

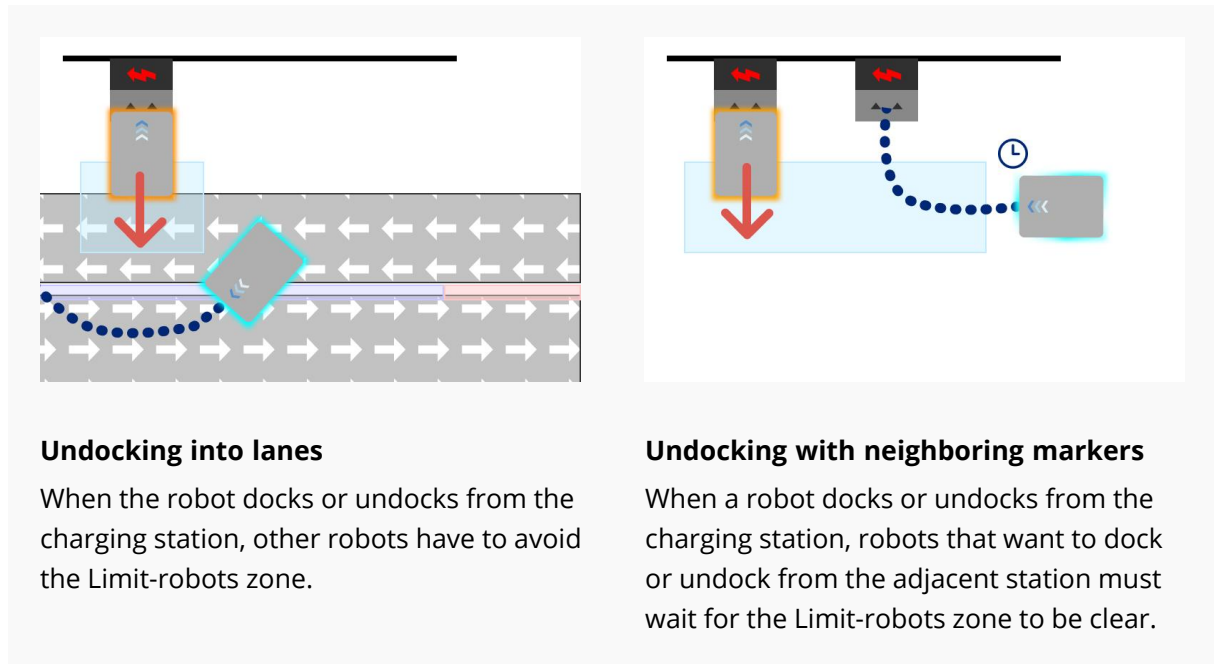
### Docking areas

Robot mute their Protective fields when docking to markers. Physically mark all docking areas as Operating hazard zones—see "[Operating hazard zones](#)" on page 440.

When robots dock and undock from markers, it is a good idea to make sure other robots do not get too close and interrupt the docking process.

This is especially relevant for undocking robots that have muted Protective fields until they finish the undocking process. If another robot is too close, it might be inside the undocking robot's Protective field when the fields are activated again, and then neither robot will be able to move without intervention.

To avoid this, add a Limit-robots zone just in front of the marker in the undocking area. Make sure that when robots are docked at the marker, it is not occupying the zone.

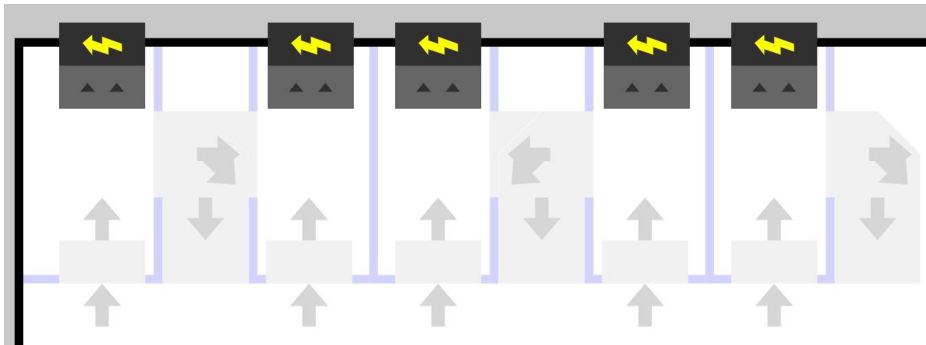


If you have two or more markers next to each other, use a Limit-robots zone in front of the markers to prevent robots from going near other undocking robots.

Instead of using Limit-robots zones to prevent deadlocks around markers, you can also route the traffic in and out of each marker using Directional zones so robots leaving the marker do not block robots driving toward the marker.

You can do this using the setup illustrated in [Figure 4.27](#). If you experience that exiting robots block each other, you can use Limit-robots zones on the exit routes from the markers to prevent this.

**Figure 4.27** At each charging station, the robot leaving the charger will leave from another direction than any incoming robots



### Service area

When robots are not connected to MiR Fleet, they are not recognized as a robot by other robots. This means the robot will not coordinate footprints when navigating around each other or respect Limit-robots zones.

To avoid issues with robot that are disabled from MiR Fleet, mark an area of the map as Forbidden zone or Unpreferred zone where only disconnected robots can be. This is mostly relevant for robot that have been removed for service or maintenance.

### Test

Test all of the missions you have created.

- Focus on testing so multiple robots have to coordinate driving around crossings and docking stations.
- Imagine the most likely scenarios where dynamic obstacles or other robots may interfere with a mission and test the robot behavior.
- Test how robots replan their path when the expected route is blocked.
- Run continuous tests on the entire production flow and observe how robots handle traffic, dynamic obstacles, such as pedestrians, or other foreseen challenges. Make corrective actions such as adjusting zones, traffic rules, timing of mission execution, or similar as necessary.

### Cycle time

Test each robot task, and verify that the setup meets cycle time requirements. If the cycle time does not meet requirements, consider the following:

- Use Speed zones to increase the robot's goal speed in areas where the robot drives straight and there is no risk of harm to personnel—see ["Assess safety" on page 437](#).
- Optimizing the traffic direction for shorter routes.
- Making the robot play sounds to inform people to clear the way.

### Crossings and docking areas

Test that all docking areas and crossings are able to handle multiple robots planning paths through them at the same time. Verify that:

- Robots are not waiting too long at Limit-robots zones. Adjust the size of the zones and the Resource assignment distance if necessary—see ["Resource handling" on page 87](#).
- Robots do not get stuck in deadlocks between resources.
- Robots do not meet head on.

### Document

- Keep your planning overview for future reference
- Create a new overview including the zones and the effect the zones have on the robot's path planning and why they are implemented.
- Document any changes made to zones. Assess zones regularly to make sure they are still optimal for the robot loads, environment, and desired traffic flow.

## 4.9.3 Configure settings

### Section overview

Adjust the robot settings to change how the robot drives and plans paths.

- Define how much robots can deviate in their path planning.
- Determine if you need to adjust the camera filtering settings.
- Apply settings changes to all other relevant robots.

Use system settings to change the robots general driving and planning behavior.

## Line-following

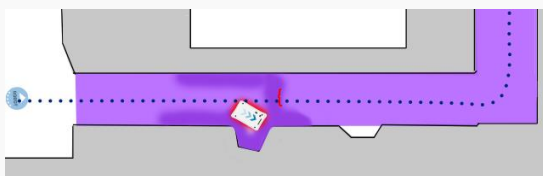
Optimizing the timeout and deviation of paths is useful in situations where you want to configure how strictly the robot should follow the path it has planned. Making the robot follow the exact path it has planned with little or no deviation is known as Line-following mode. This can be useful, for example, in narrow corridors where there is not enough space for the robot to go around dynamic obstacles or in areas where it is very important that personnel can predict the robot's movement.



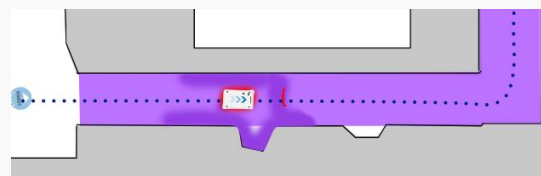
For more information on Line-following, see the guide *How to set up Line-following mode*. You can find this guide on [MiR Support Portal](#).

You can modify the robot's Line-following behavior using the following parameters under **System > Settings > Planner**:

- **Maximum path deviation for new global path** defines the maximum distance in meters that a re-planned global path is allowed to deviate from the original global path. By default, this parameter is disabled, meaning the robot can create a new global path using all possible areas of the map. We do not recommend changing this value unless you want to prevent the robot from finding an alternative route if the main route is blocked.
- **Maximum path deviation for local path** defines the maximum distance in meters that the local path is allowed to deviate from the global path before the robot makes a new global path. By default, this parameter is disabled, meaning the robot can deviate from the global path using the local planner to go around an obstacle as far as possible in the map.
- **Timeout before triggering a new global path** defines the maximum time the robot's path can be blocked before the robot generates a new global path. By default, this value is 0, meaning the robot will not wait if its current global path is blocked by an obstacle it cannot navigate around using the local planner. If you want the robot to wait and see if the obstacle moves before planning a new path, enter the maximum waiting time.



Line-following disabled



Line-following enabled

## Speed settings

You can modify the robot's speed using the following parameters under **System > Settings > Planner**:

- **Maximum allowed speed** defines the overall speed limit of the robot. The maximum allowed speed will never be exceeded no matter what is stated in a mission or Speed zone. This setting can be useful if, for example, the robot transports motion sensitive objects or if the work environment in other ways requires the robot to always stay below a certain speed threshold.
- **Desired speed** sets the desired speed of the robot. This setting can be useful in the same way as maximum allowed speed, but with this setting, the robot will drive faster than the set desired speed in a Speed zone that requires it.
- **Centripetal force limitation** (*MiR250, MiR600, and MiR1350 only*) controls how fast the robot is allowed to drive around corners. This is useful if you have a very heavy load on the robot and want to ensure the robot maintains a steady driving behavior.
- **Linear acceleration** (*MiR100, MiR200, MiR600, and MiR1350 only*) limits linear acceleration rate in percent with relative to the default value. The value is used for both acceleration and deceleration.

## Footprints

Under **System > Settings > Planner**, you can use the **Default robot footprint** setting to set the default footprint of the robot. Make sure to select a footprint that defines the robot's dimensions with its top module.

You can choose whether or not the default footprint is while the robot is carrying a load or when it is unloaded. Make sure to document what footprint is set as the default and when the footprint is intended to be used. Note that the robot never automatically changes to the default footprint during or between missions. You must use a **Set footprint** action to change the footprint back to the default footprint.

For more information about footprints, see ["Create footprints" on page 404](#).

## Obstacle history clearing



### CAUTION

Modifying this setting may result in the robot colliding with low-hanging obstacles when it reverses or pivots.

- Check the robot's operating environment for any obstacles that can only be detected by the robot's 3D cameras and are low enough to collide with the robot or its load. Either remove the obstacles, ensure that the robot is facing the obstacle whenever it is close to it, or mark the obstacle on the map with a Forbidden zone.

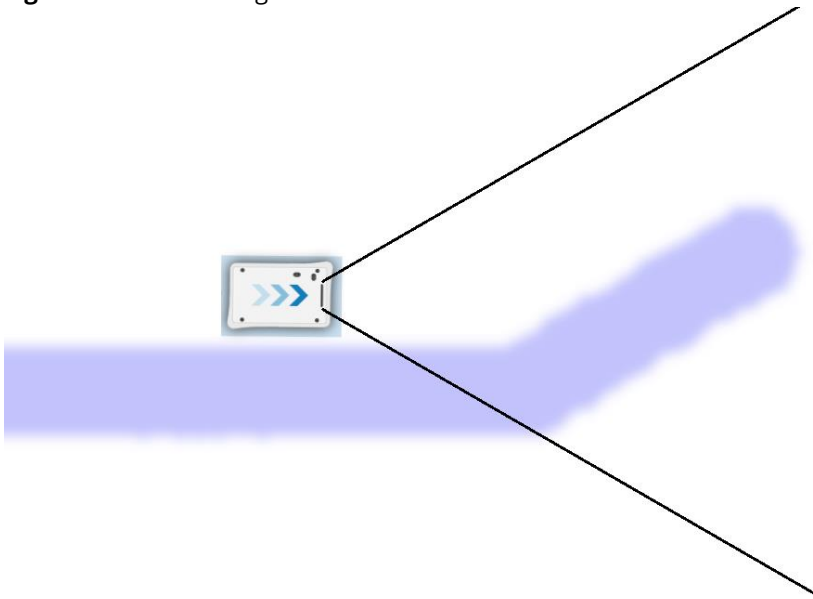
If you have the robot operating in an area with many dynamic obstacles that the robot often has to drive around, it may be beneficial to change the robot's obstacle history clearing settings.

**No clearing** is the default setting, in which the robot remembers all obstacles it has detected and only clears the obstacle history within the field of view of the sensors.

This setting produces the normal behavior of the robot and is intended for all environments, especially where the dynamic obstacles in the robot's path rarely change.

The blue cloud indicates a static object only seen by the cameras and laser scanners. The black lines indicate the field of view of the cameras. In this mode, the robot will remember the obstacle history when an obstacle detected by the cameras and laser scanners is out of view of the cameras, but will assume it is still there and make a new local plan to avoid it, if possible, or stop and wait for it to clear.

**Figure 4.28** No clearing



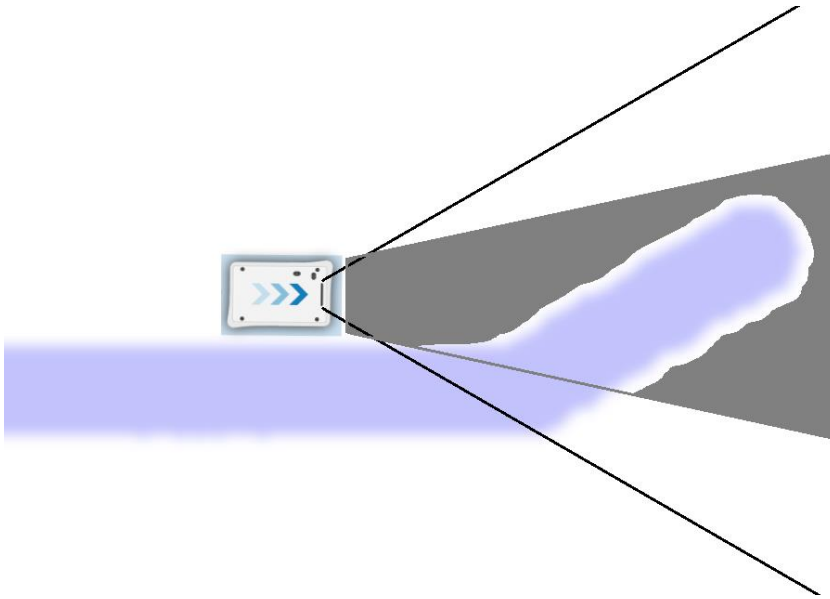
**Clear in front of robot** disables the obstacle history in a cone in front of the robot. The cone starts in the width of the robot's footprint and increases by 30 cm per meter until the end of the cameras' range.

This setting is intended for all driving configurations in environments where the robot's path is frequently interrupted by dynamic obstacles in motion, such as people walking by in front of the robot.

The blue cloud indicates a static object only seen by the cameras and scanners. The black lines indicate the field of view of the cameras. The dark gray cone indicates where the robot will clear the obstacle history as soon as the obstacle is out of view of the cameras and scanners.



**Figure 4.29** Clear in front of robot

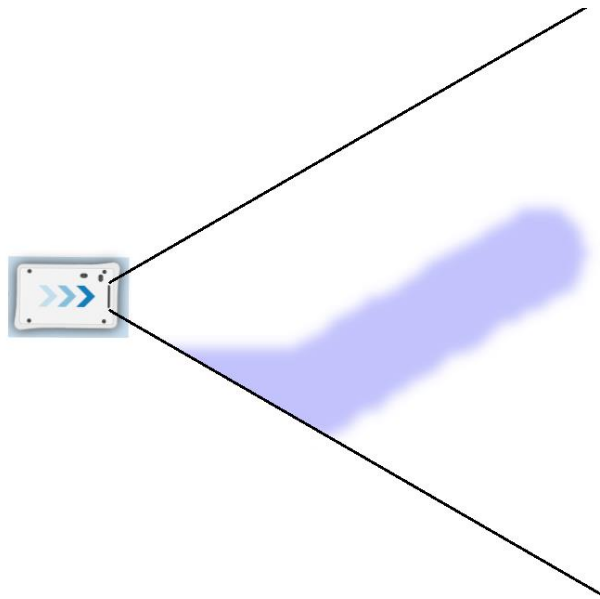


**Clear all** disables the obstacle history entirely, so that the robot only avoids obstacles it can see with the cameras.

This setting is intended for driving in Line following mode. That is, with little or no path deviation and a path timeout set to infinite waiting, and in environments where the robot's path is frequently interrupted by dynamic obstacles in motion, such as people walking in front of the robot.

When this option is used in the intended way, the robot will stop for all obstacles in the field of view and wait for them to move, thus limiting the risk of hitting unseen obstacles.

The blue cloud indicates a static object only seen by the cameras and laser scanners. The black lines indicate the field of view of the cameras. In this mode, the robot will clear the obstacle history detected by the cameras and laser scanners as soon as the obstacle is out of view of the cameras.

**Figure 4.30** Clear all

## Camera filters



### CAUTION

Modifying this setting may result in the robot colliding with low-hanging obstacles if the cameras can no longer detect the obstacles.

- Check the robot's operating environment for any obstacles that can only be detected by the robot's 3D cameras and make sure that the camera still detects the obstacles after modifying the filter settings or mark the obstacle on the map with a Forbidden zone.

If there is no apparent reason why the cameras are detecting obstacles where there are none, you can modify the camera filter settings under **System > Settings > 3D Cameras**. A higher filter configuration will allow the robot to see fewer obstacles. A lower filter configuration will allow the robot to see more obstacles.

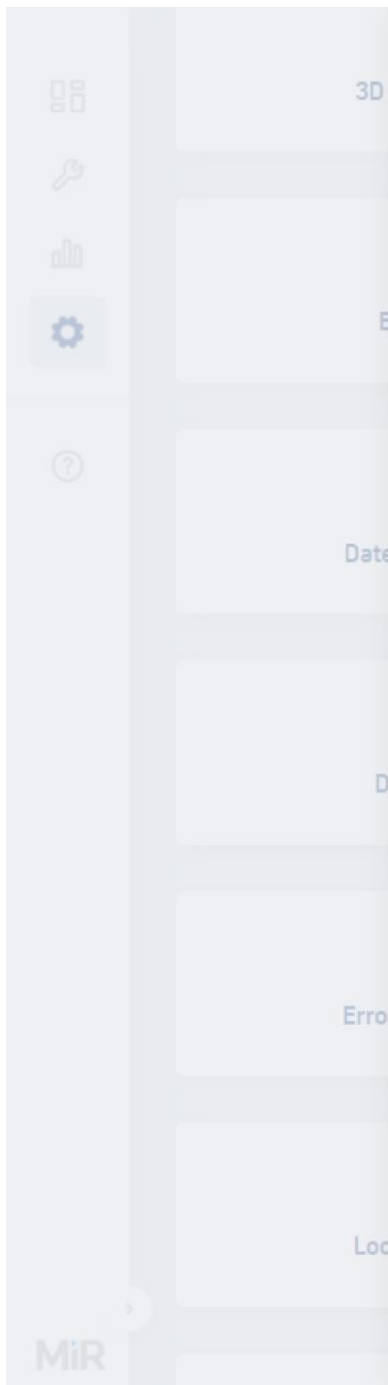
To correctly modify the settings, follow these steps:

- 1 Position the robot in the place where it is detecting a non-existent obstacle. You can see on the robot's active map that there are obstacle clouds in an area where there are no obstacles.
- 2 Go to **System > Settings > 3D Cameras**, and increase the camera filter strength, by selecting **Right camera filter configuration** or **Left camera filter configuration** depending on which side of the robot the obstacle clouds appear.
- 3 Select **Save changes**.
- 4 Check on the map if the obstacle clouds have disappeared. If not, continue to increase the filter strength, and check if the obstacle clouds are gone between each time you apply the filter changes.

If the cameras still detect obstacle clouds, even at the strongest filter settings, check the cameras for contamination, scratches, or other damage that may be affecting their performance.

### Settings across robots

If you make any changes to the robot settings, apply the same changes to all other robots. This ensures consistent behavior across all robots. Use the Modified defaults page under **System > Settings** to see all settings you have modified. The modified defaults should be the same for all robots.



## Modified defaults ×

Modified defaults provides an overview of which parameters have been changed from the default values. The list shows both the default values and the new values of the changed parameters. All changed parameters from **System > Settings** are included in the Modified defaults list.

### Robot

3D camera model	intel_d435	none
Default robot footprint		mirconst-guid-0000-0001-footprint009
Desired speed	0.8	1.2
Distance to marker for disabling collision checks	0.2	0.9
Distance to VL-marker before muting front protective fields.	0.4	0.9
Elevators	false	true
Left camera serial number	1234567890	103422072531
Maximum allowed speed	1.2	2
Modbus	false	true
Mute protective fields.	false	true
Obstacle history clearing	no_clearing	clear_all
Parameter for driving more straight during docking	1.0	.5
Right camera serial number	1234567890	051422070790
User disconnected Wi-Fi	true	false

## 4.10 Assess safety

**After reading this chapter, you can:**

Assess the safety risks that personnel working with robot are exposed to and implement or adjust safeguards accordingly.

**Conduct a risk assessment.** See ["Conduct risk assessment" on the next page](#).

- Use the *MiR Risk Assessment Guide* for MiR-specific use cases. You can find this guide on **MiR Support Portal**.

**Implement risk mitigation means.** See ["Mitigate risks" on page 439](#).

- Mark high-risk areas as Operating hazard zones.
- Mark all docking and charging areas as Operating hazard zones.
- Implement lights and sounds in the robot maps and missions to increase personnel awareness.
- Design docking areas to include escape routes for personnel.

**Ensure the robots' braking distance is safe with all loads.** See ["Test braking distance" on page 448](#).

- Conduct a brake test with the lightest, heaviest, and largest loads, and any loads with unusual properties.
- Adjust the robot, the environment, or the robot payload until the stopping distance criteria in the brake test are met.

**If necessary, adjust the Protective fields according to the brake test results** See ["Adjust Protective fields" on page 457](#)..

- Modify all relevant field sets and cases in SICK software.
- Save copies of each modified version of the SICK configuration you create.
- Conduct brake tests and missions with docking or narrow areas with the new SICK configuration file.
- Assess if the robot meets all required standards and safety requirements with the new SICK configuration file.

- Document which robots operate with modified SICK configuration, how the configuration differs from the standards, and verification that the configuration has been safely tested and validated.

**Test the robots safety functions** See "[Test safety functions](#)" on page 461..

- Inspect the safety laser scanners.
- Verify that Emergency stop buttons are accessible when the robot is operating.
- Verify that the safety lasers scanners can detect obstacles.
- Verify that the Emergency stop buttons work.

### 4.10.1 Conduct risk assessment

#### Section overview

Conduct a risk assessment.

- Use the *MiR Risk Assessment Guide* for MiR-specific use cases. You can find this guide on MiR Support Portal.

To achieve a safe installation, it is necessary to conduct a risk assessment of the robot in the environment it will be used in. This is the responsibility of the commissioner.

The risk assessment must cover both the robot itself and also take into account carts, potential load transfer stations, work cells, and the work environment.



#### NOTICE

Mobile Industrial Robots takes no responsibility for the creation and performance of the risk assessment, but we provide information and guidelines that may be used in this section. For more guidelines, see *MiR Risk Assessment Guide*. You can find this guide on [MiR Support Portal](#).

It is recommended that the integrator follows the guidelines in ISO 12100, EN ISO 3691-4, EN 1525, ANSI B56.5, or other relevant standards to conduct the risk assessment.

In ISO 3691-4 Annex B there is a list of possible significant hazards and hazardous situations that the integrator should consider.

A risk assessment of the application must be used to determine the adequate information for users. Special attention to at least the following Essential Health and Safety Requirements (EHSR) from Directive 2006/42/EC must be taken:

- 1.2.2 Control devices
- 1.3.7 Risk related to moving parts
- 1.7.1 Information and warning on the machinery
- 1.7.2 Warning of residual risks
- 1.7.3 Marking of the machinery
- 1.7.4 Instructions

The risk assessment will lead to new instructions that must be written by the party who draw up the CE marking. The instructions must at least include:

- Intended use and foreseeable misuse.
- A list of residual risks.
- Training required for personnel.

### 4.10.2 Mitigate risks

#### **Section overview**

Implement risk mitigation means.

- Mark high-risk areas as Operating hazard zones.
- Mark all docking and charging areas as Operating hazard zones.
- Implement lights and sounds in the robot maps and missions to increase personnel awareness.
- Design docking areas to include escape routes for personnel.

After creating a risk assessment, you must mitigate the identified residual risks if possible.

The robot provides some additional tools you can use to mitigate some risks. These are described in the following sections.

## Operating hazard zones

Operating hazard zones are areas that must be visibly marked to comply with safety standards in EN 1525 and ISO 3691-4. They must adhere to the following:

- The operating hazard zone borders must be at least one meter from the identified hazard in all directions.
- Personnel must be instructed to stay clear of operating hazard zones when a robot is approaching.
- It is not allowed to have work stations in operating hazard zones.

Areas where you must apply operating hazard zones for MiR robots are:

- Areas where the robot drives with muted Protective fields—see the user guide or integrator manual for your robot application for all instances where the robot automatically mutes the Protective fields.
- Areas with inadequate clearance for escape routes.



### WARNING

When the robot is in an operating hazard zone, there is a risk of injury to any personnel within the zone.

- Ensure that all personnel are instructed to stay clear of operating hazard zones when the robot is in the zone.

You can add zones to the map in the robot interface to mitigate the risks to personnel in operating hazard zones. We recommend considering whether adding the following zones can reduce the risks in an operating hazard zone:

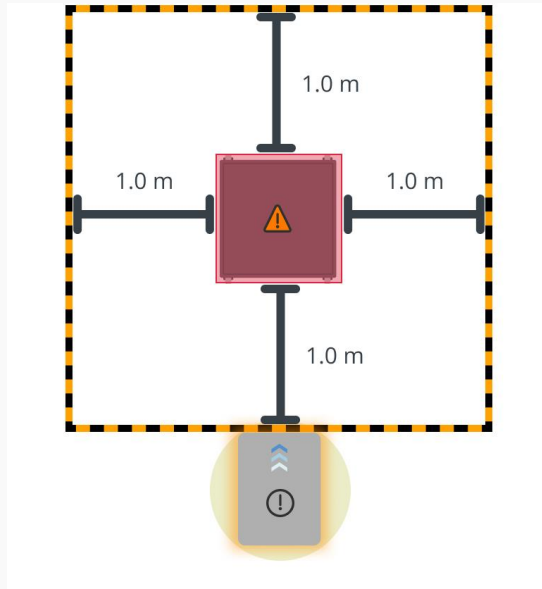
- Speed zones can be used to reduce the speed of the robot to the minimum robot speed.
- Sound and light zones can be used to add acoustic and visual warnings when the robot drives into the zones.



For more information about zones, see *MiR Robot Interface Guide*. For examples of operating hazard zones, see *MiR Commissioning Guide*. You can find these guides on [MiR Support Portal](#).



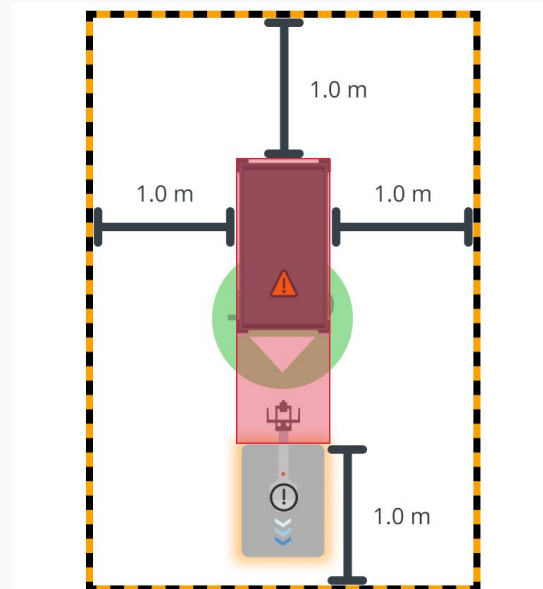
The following illustrations are examples of common uses of MiR robots where you must mark an operating hazard zone.



#### Robot docking to a shelf with active Protective fields

When MiR100 or MiR200 docks to a marker or shelf, there is a risk of injury if personnel are too close to the marker or shelf.

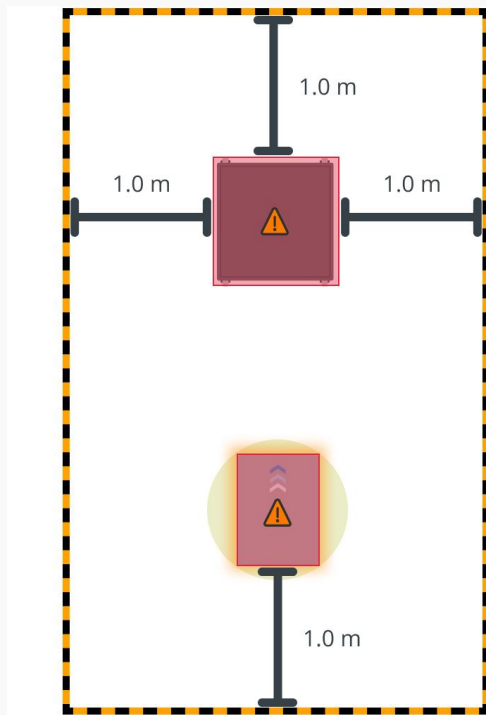
To determine the size of the operating hazard zone, mark the area one meter around the marker or shelf.



#### Robot placing a cart with active Protective fields

When a robot with a hook reverses to place a cart, there is a risk of injury if personnel are too close to the cart or are in the area behind the robot when it starts reversing.

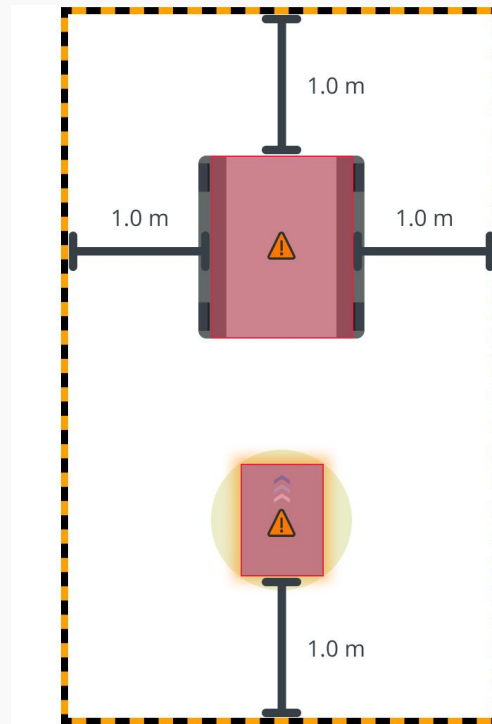
To determine the size of the operating hazard zone, mark the area one meter around the cart when it is on the Cart position.



#### Robot docking to a shelf with muted Protective fields

When MiR250, MiR500, MiR600, MiR1000, or MiR1350 docks to a marker or shelf, there is a risk of injury if personnel are too close to the marker or shelf or to the robot when it mutes its Protective fields while docking.

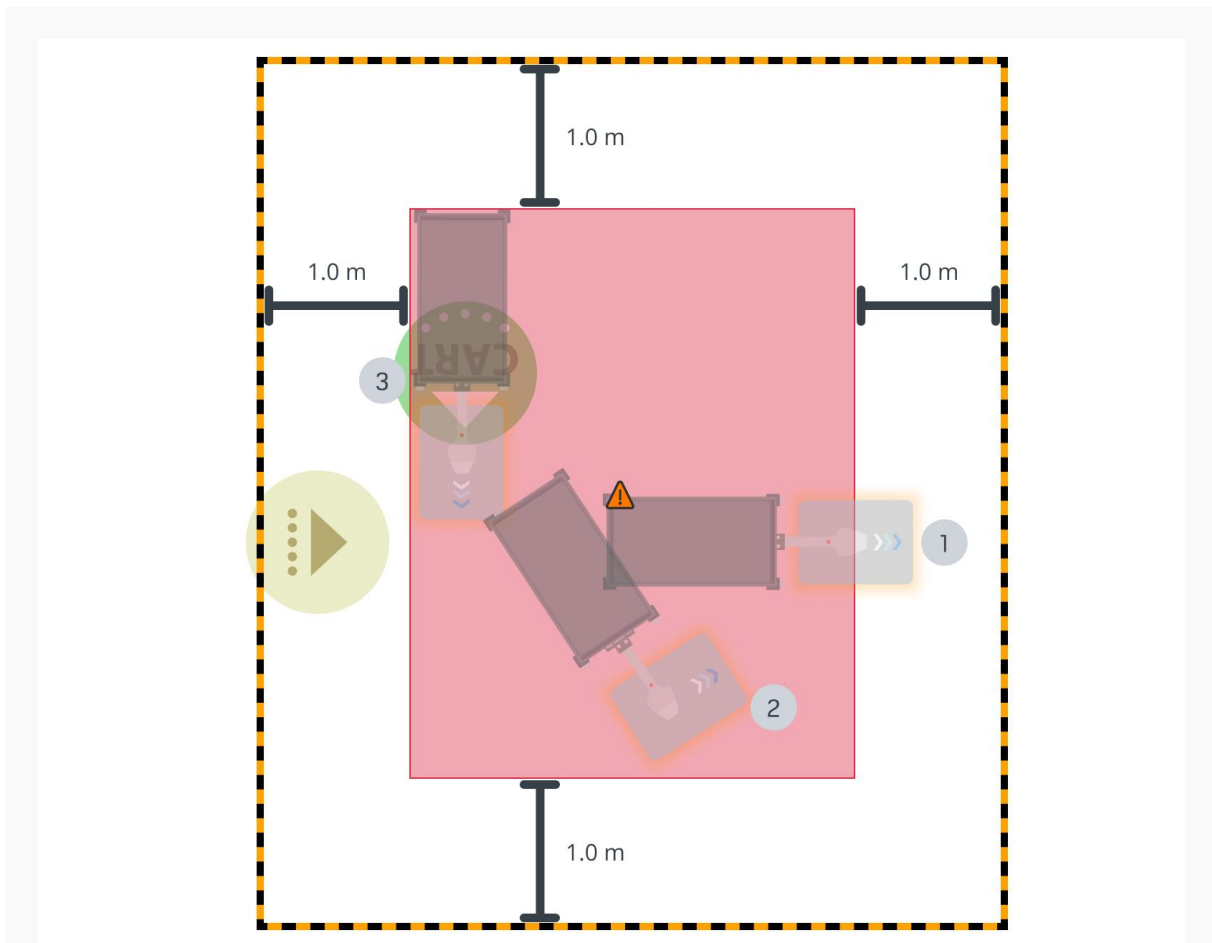
To determine the size of the operating hazard zone, mark the area one meter around the shelf and the robot when it is at the Entry position for the shelf.



#### Robot docking to a pallet rack with muted Protective fields

When MiR500, MiR600, MiR1000, or MiR1350 docks to a pallet rack, there is a risk of injury if personnel are too close to the area of the rack where the pallets are placed or to the robot when it mutes its Protective fields while docking.

To determine the size of the operating hazard zone, mark the area one meter around the pallet rack and the robot when it is at the Entry position for the pallet rack.



### Robot places cart in reverse

When a robot with a hook reverses to place a cart, there is a risk of injury if personnel are behind the robot or cart at any point during the reverse process.

To determine the size of the operating hazard zone, mark the furthest distances from the Cart position where the robot reverses:

- 1 The furthest the robot drives beyond the Cart position before reversing. Mark the area up to the center of the hook arm. Only the area behind the cart and around the hook are considered hazardous, since the laser scanners are not muted.
- 2 The furthest the robot drives while turning 90° while reversing with the cart.
- 3 The final position the robot places the cart at.

## Escape routes

When you have multiple markers, shelves, pallet racks, or other loading stations placed next to each other, you must ensure that there are sufficient pedestrian escape routes around the stations.



### NOTICE

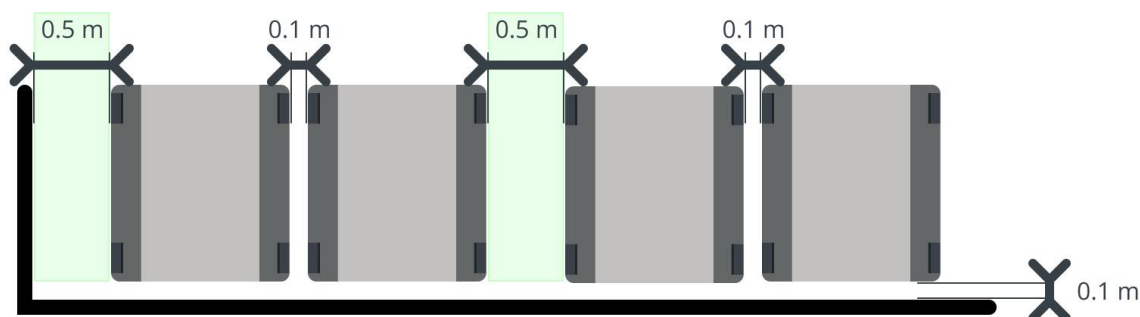
Pedestrian escape routes are cleared paths that are at least 0.5 m wide and 2.1 m high. There must always be an escape route on either side of the robot, also in operating hazard zones.

If there is 0.5 m or more between each station, there are sufficient escape routes for personnel. If not, apply one of the following configurations.

### Stations against a wall or structure

When setting up stations against a wall or other fixed structures, place the stations so there is 0.5 m of space between every second station. This ensures there is always an escape route on at least one side of each station.

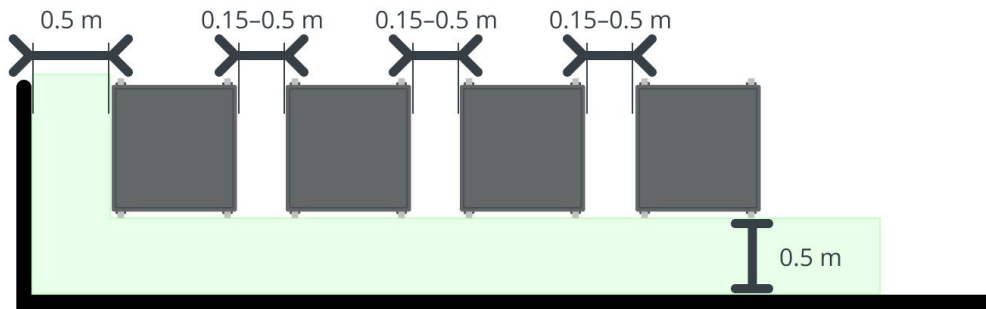
**Figure 4.31** Work station placement so there are escape routes between stations



### Stations close together

When setting up stations so they are placed as closely together as possible, there must be at least 0.5 m of free space behind the stations and 0.5 m of free space on the side of one of the stations leading out of the station line.

**Figure 4.32** Work station placement so there are escape routes behind the stations



## Sounds and light

Use sounds and lights on the robot to alert personnel of the robot, which can be used to draw attention to certain risks.



### CAUTION

Unaware personnel may not see the robot in certain situations and risk colliding with the robot. This may result in injury to personnel or damage to equipment.

- Make sure to adjust the volume of the robot's warning sounds so they are audible in the robot's work environment.
- Implement warning sounds from the robot in missions and areas where it can reduce the risk of hazardous situations.

## Using Sound and light zones

On maps, you can add Sound and light zones to trigger the robot to play selected sounds and flash its indicator lights.

Use this zone in the following areas:

- On operating hazard zones to mitigate the risk identified in the zone.
- Around corners and intersections.
- In areas where personnel may not hear or see the robot easily when it is nearby.

### Using Sound and light actions

In missions, you can add Sound and light actions to make the robot play selected sounds and flash its indicator lights at certain points in the mission.

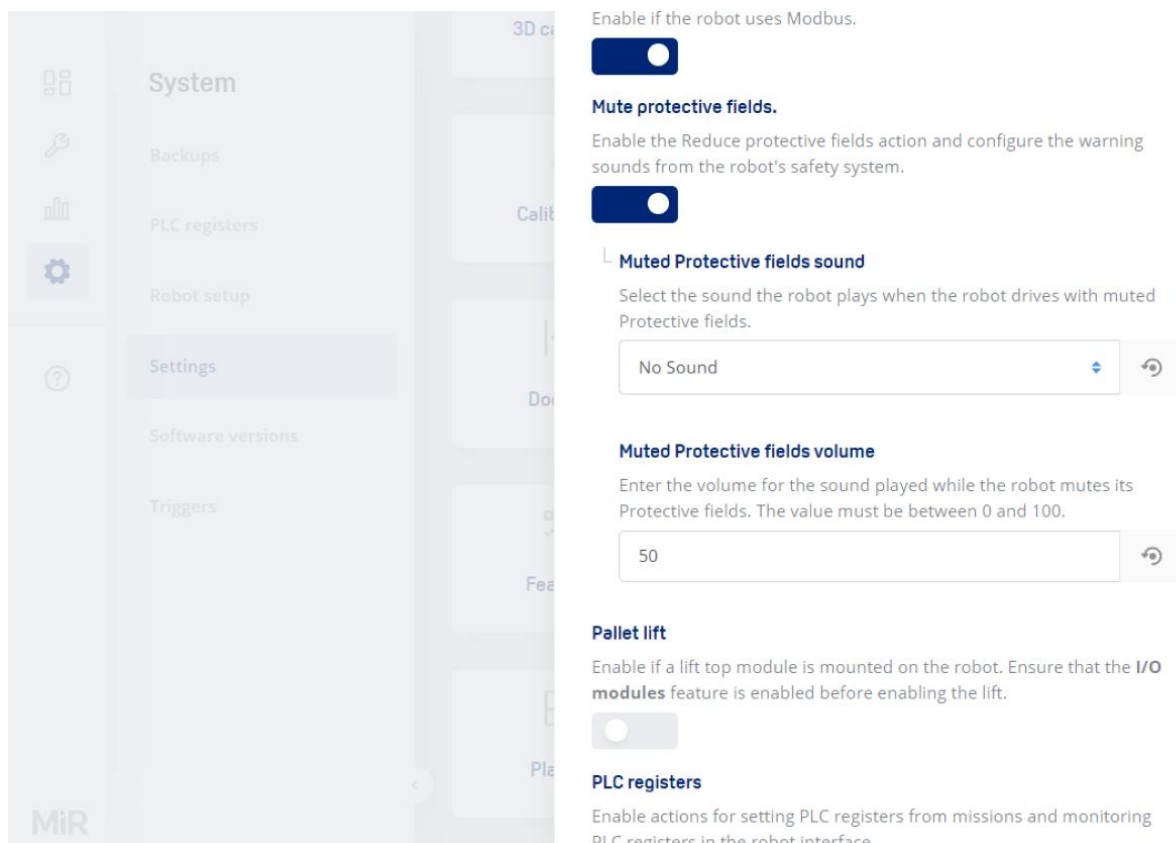
Use these actions in the following situations:

- When the mission also has a Muted protective fields action.
- When the robot is executing an action that could potentially harm personnel that are positioned close to a potential risk.
- When the robot is executing an action that needs supervision.
- Just before the robot executes an action that will make it move.

## System settings

When the robot drives with muted Protective fields it emits a warning sound. In the robot interface under **System > Settings > Features**, you can choose which sound the robot makes and the volume of the sound when the robot is executing actions within a Mute Protective fields action.

**Figure 4.33** In the Features settings, you can modify the sounds the robot plays when the robot mutes its Protective fields



### Mute protective fields.

Enable the Reduce protective fields action and configure the warning sounds from the robot's safety system.



### Muted Protective fields sound

Select the sound the robot plays when the robot drives with muted Protective fields.

No Sound

### Muted Protective fields volume

Enter the volume for the sound played while the robot mutes its Protective fields. The value must be between 0 and 100.

50

### Pallet lift

Enable if a lift top module is mounted on the robot. Ensure that the I/O modules feature is enabled before enabling the lift.



### PLC registers

Enable actions for setting PLC registers from missions and monitoring PLC registers in the robot interface.

### 4.10.3 Test braking distance

#### Section overview

Ensure the robots' braking distance is safe with all loads.

- Conduct a brake test with the lightest, heaviest, and largest loads, and any loads with unusual properties.
- Adjust the robot, the environment, or the robot payload until the stopping distance criteria in the brake test are met.

It is the responsibility of the commissioner to perform an adequate test of the robot's braking capability under the driving conditions the robot will be operating in. The brake test must be executed:

- With maximum payload.
- With the lightest production payload that the robot is going to operate with.
- Within each speed interval for each Protective field.
- At the steepest supported decline in the robot's work area.

Any hardware, footprint, or safety configuration changes must be production-ready before performing brake tests.

The commissioner must iteratively adjust the modifiable parameters and then run a brake test until the robot successfully stops in time in every test.



See a [video](#) of the process on the MiR TechComm videos channel on vimeo.com.

#### Dependent factors

The braking distance of the robot is particularly dependent upon the parameter in [Table 4.9](#). If the robot fails a brake test, you must either modify a chosen parameter to reduce the distance, or you can increase the size of the robot's Protective fields to make the robot stop sooner. To change the Protective fields, see "[Adjust Protective fields](#)" on page 457.



**Table 4.9** Parameters affecting the robot's braking distance

Parameter	Design consideration for Protective fields
Top module	<p>Adjust the length, width, and shape of the Protective fields accordingly to match the footprint of the top module if it extends beyond the top cover of the robot.</p> <p>Adjust the shape of the Protective fields accordingly to create cutouts for cart or shelf legs in the view of the scanners.</p>
Robot speed	Speed affects the braking distance according to a quadratic relationship. For example, 2× higher speed means 4× longer braking distance. The higher the maximum speed in a monitoring case, the longer the Protective fields must be.
Mass moment of inertia	A higher mass moment of inertia increases the braking distance when the robot pivots. The higher the mass moment of inertia, the wider the Protective fields must be.
Payload weight	A higher payload increases the braking distance when the robot drives straight. The higher the payload, the longer the Protective fields must be.
Payload position	When the payload's center of gravity is behind the drive wheels, it increases the braking distance when the robot drives straight. The further the center of gravity is behind the drive wheels, the longer the Protective fields must be.
Floor surface friction	A lower friction between the wheels and floor increases the braking distance. The lower the frictional coefficient, the longer the Protective fields must be.

Parameter	Design consideration for Protective fields
Floor surface grade	When the robot is driving down a decline, it increases the braking distance. The steeper the maximum slope the robot drives down, the longer the Protective fields must be.

### Brake test method

This section presents a recommended approach for conducting linear and pivot brake tests for this purpose.

#### Test setup

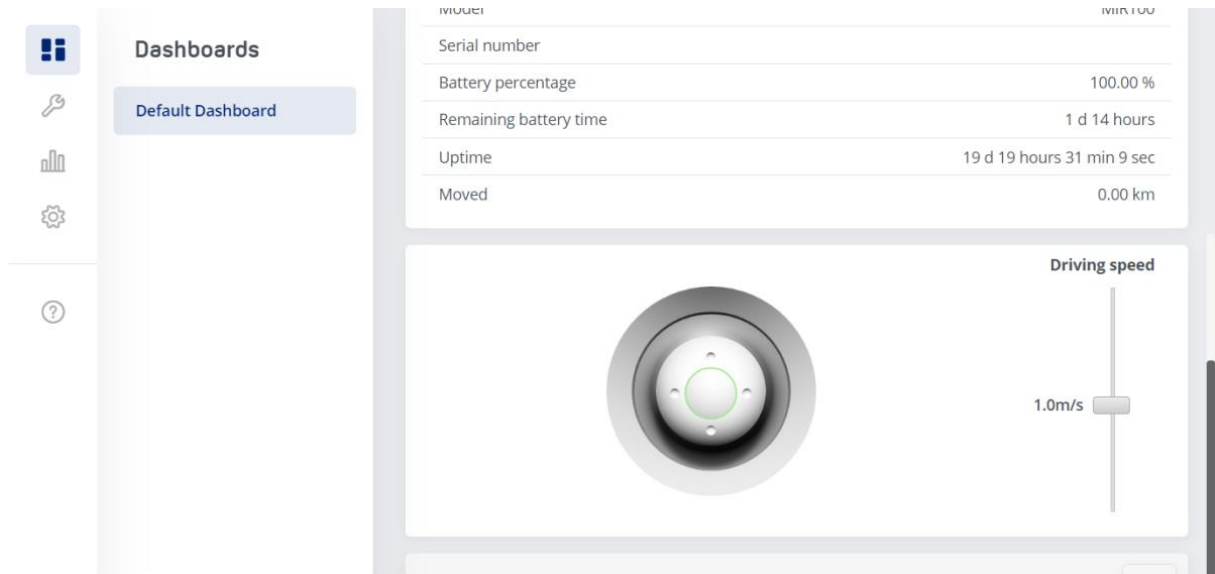
In order to conduct brake tests for verifying robot Protective fields, the following items are needed:

- **Robot:** the robot including its top module and maximum payload.
- **Brake test area:** approximately 3 m × 10 m; or the distance it takes for the robot to reach maximum speed and stop safely.
- **Tape measure:** minimum 1 m of measuring length.
- **Test object:** An object such as a cardboard box or similar that the robot can safely collide with. Recommended size: 200 mm high and 600 mm wide (approximating a cross-section of a human body lying on the ground).

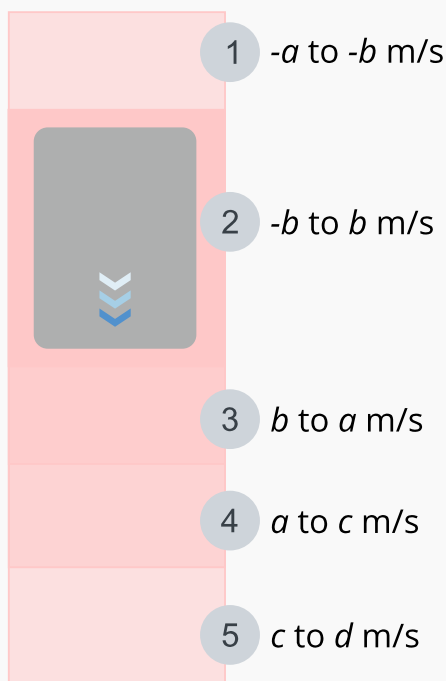
### Changing the speed

You must set up your robot to drive at the different monitoring case speeds. In the robot interface, use the joystick in Default Dashboard to drive the robot at different speeds. Adjust the slider to change the robot's driving speed.

**Figure 4.34** The Default Dashboard in the robot interface



The following is a generic example where the monitoring cases use speeds  $a$ ,  $b$ ,  $c$ , and  $d$  for both reverse driving and forward driving. See the user guide or integrator manual for your robot application for the specific Protective field sizes.



Case	Joystick speed	Drive direction
1	$a - 0.05$ m/s	Reverse
2	$b - 0.05$ m/s	Reverse and forward
3	$a - 0.05$ m/s	Forward
4	$c - 0.05$ m/s	Forward
5	$d - 0.05$ m/s	Forward

## Method

The following two sections describe how to perform a linear brake test and pivot brake test.

### Linear brake test

A linear brake test must be conducted to ensure that the length of the Protective fields of the robot are correctly configured for each monitoring case and the planned payload.

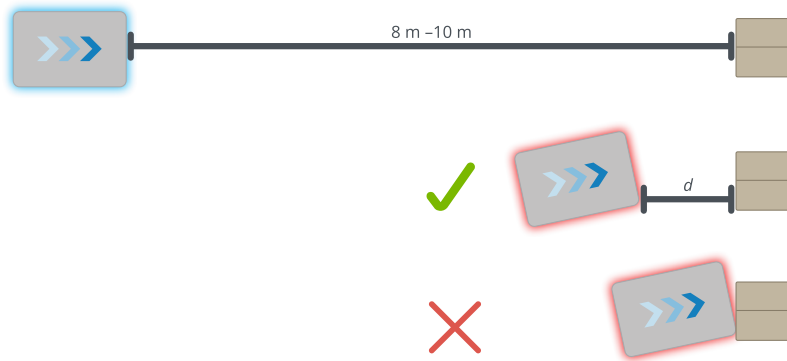
After driving the robot toward a test object and braking occurs, the distance  $d$  between the stopped robot and the test object should be measured and recorded for each brake test. Before conducting a linear brake test, a pass criterion for the test must be defined.

*Pass criterion = acceptable distance  $d$  for the site (we recommend 215 mm as a starting point, or 65 mm if personnel wear safety shoes).*

To conduct a linear brake test for forward and reverse driving, follow these steps:

- 1 Place a static test object front and center relative to the driving path of the robot—see [Figure 4.35](#).
- 2 Set the joystick to the lowest speed, and push the joystick directly forward for forward driving (or directly backward for reverse driving) to drive the robot straight towards the test object.
- 3 Once the robot has completed braking and come to a complete stop near the test object, measure and record the distance  $d$  from the part of the robot that is closest to the test object.
- 4 Repeat steps 2–3 a total of five times for the selected joystick speed.
- 5 Repeat steps 2–4 for each joystick speed.

**Figure 4.35** Linear brake test diagram



The test is considered passed when the distance  $d$  is equal to or greater than the pass criterion value. Likewise, the test is considered failed when the distance  $d$  is less than the pass criterion value. "P" for Pass or "F" for Fail should be recorded for each test in the test matrix.

**Table 4.10** Linear brake test example matrix

	Forward driving speed (m/s)						Reverse driving speed (m/s)					
	0.1	0.3	...				-0.1	-0.3	...			
<b>Test 1</b>												
Result												
$d$ (mm)												
<b>Test 2</b>												
Result												
$d$ (mm)												
<b>Test 3</b>												
Result												
$d$ (mm)												
<b>Test 4</b>												
Result												
$d$ (mm)												
<b>Test 5</b>												

	Forward driving speed (m/s)						Reverse driving speed (m/s)					
	0.1	0.3	...				-0.1	-0.3	...			
Result												
$d$ (mm)												

**Pivot brake test**

The pivot brake test must be conducted to ensure that the width of the standstill Protective field for the robot is sufficient to prevent a collision when the robot is pivoting on its axis with maximum payload.

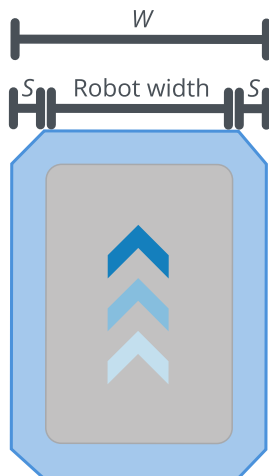
If you are using the default SICK configuration, the standstill Protective field corresponds to monitoring case 1 (0 to 0.1 m/s) for forward and reverse driving.

After pivoting the robot toward a test object and braking occurs, the distance  $p$  between the stopped robot and the test object should be measured and recorded for each brake test. Before conducting a pivot brake test, a pass criterion for the test must be defined.

*Pass criterion = acceptable distance  $p$  for the site (we recommend 215 mm as a starting point, or 65 mm if personnel wear safety shoes).*

Determine the distance  $S$  for object placement relative to the side of the robot.

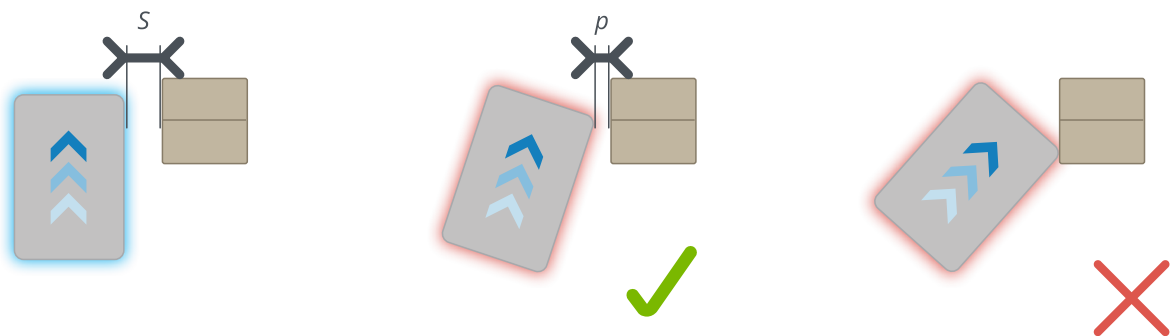
**Figure 4.36** Illustration of  $W$  and  $S$



To conduct a pivot brake test to the left and right of the robot, follow these steps:

- 1 Place a static test object a distance  $S$  to the left of the front-left corner of the robot (or to the right of the front-right corner of the robot). Make sure the object is placed just outside of the standstill Protective field so that the robot is not in Protective stop—see [Figure 4.37](#).
- 2 Positioned in front of the robot and using the joystick, push the joystick directly to the left for pivoting the robot in a clockwise direction (or directly to the right for pivoting the robot in a counterclockwise direction) toward the test object.
- 3 Once the robot has completed braking and come to a complete stop near the test object, measure and record the distance  $p$  from the part of the robot that is closest to the test object.
- 4 Repeat steps 2–3 a total of 5 times.
- 5 Repeat steps 1–4 for pivoting the robot to the right toward a test object.

**Figure 4.37** Pivot brake test diagram



The test is considered passed when the distance  $p$  is equal to or greater than the pass criterion value. Likewise, the test is considered failed when the distance  $p$  is less than the pass criterion value. "P" for Pass or "F" for Fail should be recorded for each test in the test matrix.

**Table 4.11** Pivot brake test example matrix

	Left	Right
<b>Test 1</b>		
Result		
$\rho$ (mm)		
<b>Test 2</b>		
Result		
$\rho$ (mm)		
<b>Test 3</b>		
Result		
$\rho$ (mm)		
<b>Test 4</b>		
Result		
$\rho$ (mm)		
<b>Test 5</b>		
Result		
$\rho$ (mm)		



### 4.10.4 Adjust Protective fields

#### Section overview

If necessary, adjust the Protective fields according to the brake test results.

- Modify all relevant field sets and cases in SICK software.
- Save copies of each modified version of the SICK configuration you create.
- Conduct brake tests and missions with docking or narrow areas with the new SICK configuration file.
- Assess if the robot meets all required standards and safety requirements with the new SICK configuration file.
- Document which robots operate with modified SICK configuration, how the configuration differs from the standards, and verification that the configuration has been safely tested and validated.

MiR delivers robots with a default SICK safety configuration file that contains factory set Protective fields. As part of commissioning, the default configuration can be modified according to the needs and parameters of the work environment. The resulting modified configuration must be verified to ensure the robot can function safely. This commissioning approach is consistent with the EU Machinery Directive (MD) that promotes an "on site configuration and CE marking of the robot in its intended application".

If you have a CE marked MiR robot, the Protective field sets are configured to comply with the safety standards of the robot. Modifications may prevent the robot from stopping in time to avoid collision with personnel and equipment. Any modifications of the SICK configuration requires a new CE certification of the robot and compliance to all safety standards listed in the specification of the application and in other way declared.

You must either modify the fields or redesign the application if the robot fails during the brake test—see ["Test braking distance" on page 448](#)—or if the robot is driving with a top module or load that interferes with the robot's Protective fields. MiR provides default SICK configuration files that can be used with MiR top modules and in environments that meet MiR requirements—see ["Evaluate environment" on page 178](#).

Do not modify the safety system without a competent third party to evaluate the safety of the design and performance of the robot after the modifications are applied.

If you modify a SICK configuration file, always save a copy before making more modifications.

To modify the Protective fields, see the guide that applies to your robot:

- *How to adjust the Protective field sets on MiR100 and MiR200*
- *How to adjust the Protective field sets on MiR250*
- *How to adjust the Protective field sets on MiR500, MiR600, MiR1000, and MiR1350*

You can find these guides on [MiR Support Portal](#).

After modifying the Protective fields, you may need to modify the robot's footprints to reflect your changes—see "[Create footprints](#)" on page 404. You must also test the safety of the Protective fields as described in "[Test braking distance](#)" on page 448.

### The default SICK configuration file

The default SICK safety configuration file can be used as a starting point for modification. To modify the safety configuration, you must open the configuration file using SICK's configuration software:

- For MiR100 and MiR200, use SICK Flexi Soft Designer.
- For MiR250, MiR500, MiR600, MiR1000, and MiR1350, use SICK Safety Designer.

You can download the MiR standard safety configurations in the robot interface under **System > Robot setup > SICK configuration files** or under **Software** on [MiR Support Portal](#).

The following table describes the relevant settings in the default SICK configuration file for the front and rear laser scanners for MiR250, MiR500, MiR600, MiR1000, and MiR1350.

Section	Default setting	Description
Application	Application type: Mobile	The Mobile setting is intended to protect people when vehicles are moving or docking.
Application	Display alignment: Upside down	The mounting orientation of the optics cover on the safety laser scanners.

Section	Default setting	Description
Monitoring plane and Fields	Safety task: Hazardous area protection (horizontal)	People approach the robot in a plane that is horizontal to the safety laser scanners.
Monitoring plane and Fields	Object resolution: Leg (70 mm) or Hand (20 mm)	The object resolution defines the size that an object must be for the laser scanner to detect it reliably. Depending on the field, this is either set to leg or hand. You can also change this setting for individual fields.
Monitoring plane and Fields	Multiple Sampling: 2×	Multiple sampling indicates how often an object has to be scanned before the safety laser scanner reacts. 2× is the minimum setting and keeps response time quick.
Monitoring Cases	Speed intervals are robot dependent and can be seen in the manual for your robot application.	The table configures the monitoring cases for the speed interval case sequence and the associated Protective fields.

The following table describes the relevant settings in the default SICK configuration file for the front and rear laser scanners for MiR100 and MiR200.

Section	Default setting	Description
System parameters	Rotation of the 7-segment display: 180°	The mounting orientation of the safety laser scanners is upside down.

Section	Default setting	Description
Resolution/scanning range	Application type: Mobile	The Mobile setting is intended to protect people when vehicles are moving or docking.
Resolution/scanning range	Object resolution: Leg (50 mm)	The object resolution defines the size that an object must be for the laser scanner to detect it reliably. In most cases, the scanners are intended for detecting legs.
Restart	Restart of the local OSSDs: 2 seconds	The amount of time the Protective fields must be clear before the robot can drive again.
Cases	Speed intervals are robot dependent and can be seen in the manual for your robot application.	The table configures the monitoring cases for the speed interval case sequence and the associated Protective fields.

### Performance priorities

Modifications you make in the safety configuration may affect the safety or agility of the robot. Often, by increasing the robot's agility (its ability to maneuver in dynamic and tight spaces) you reduce the robot's safety capabilities. While commissioning, you must find the suitable balance of safety and agility for your setup. The main method of determining if the Protective fields are adequately safe is by running brake tests—see ["Test braking distance" on page 448](#).

Ensure that the robot's Protective fields are large enough to ensure the robot will not collide with personnel or any objects that may cause a hazardous situation. If the robot is unable to operate in the work environment without reducing the size of the Protective fields, you must ensure that necessary safeguards are installed and risk mitigating actions are taken.

### 4.10.5 Test safety functions

#### Section overview

Test the robots safety functions.

- Inspect the safety laser scanners.
- Verify that Emergency stop buttons are accessible when the robot is operating.
- Verify that the safety lasers scanners can detect obstacles.
- Verify that the Emergency stop buttons work.

Prior to modifying the SICK safety configuration, check that the safety system components are in good condition and operating as intended.

To access the laser scanners:

- For MiR100 and MiR200, remove the top cover.
- For MiR250, remove the front, rear, and side covers.
- For MiR500, MiR600, MiR1000, or MiR1350, remove the front, rear, and side hatches followed by the front-left and rear-right corner covers.

See the user guide or integrator manual for your robot application for instructions on how to remove the covers and hatches.

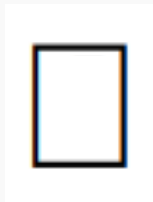
#### Visual check

- 1 Verify that the front laser scanner is not damaged. Look for signs such as dents, open cable ends, and poor alignment.
- 2 Verify that the optical cover on the laser scanner is not contaminated or scratched. In the case of an unclean optical cover, gently wipe the cover with a cloth—see the section *Maintenance* in the user guide or integrator manual for your robot application. In the case of a scratched optical cover, replace the optical cover. See SICK's documentation for the laser scanner type for instructions.

- 3 Verify there are no objects, such as cables, in the scanning view of the laser scanner.
- 4 Repeat steps 1–3 for the rear laser scanner.
- 5 Check that all Emergency stop buttons are accessible.

### Functional check

- 1 Turn on the robot, and wait for it to finish starting up.
- 2 Observe the digital display on the front laser scanner. Verify the main display indicates that the scanner is ready for operation.



**MiR100 and MiR200**



**MiR250, MiR500, MiR600, MiR1000, and MiR1350**

- 3 Place a test object in the Protective field, and verify that the laser scanner displays an interruption of the Protective field on the digital display.



**MiR100 and MiR200**



**MiR250, MiR500, MiR600, MiR1000, and MiR1350**

- 4 Repeat steps 1–2 for the rear laser scanner.
- 5 For each Emergency stop button, press the button, verify the robot enters Emergency stop, release the button, and press the Resume button.

If one of the above items is not met or a fault is revealed, generate a SICK report from the Safety Designer software—see *How to generate a SICK report*—and send it to a MiR representative for consultation before proceeding with modification of the SICK safety configuration. You can find this guide on [MiR Support Portal](#).

## 4.11 Ensure security and stability

### **After reading this chapter, you can:**

Improve the cybersecurity and stability of your MiR system and monitor performance.

**Assess the cybersecurity of your MiR system.** See "[Evaluate cybersecurity](#)" on the [next page](#).

- Conduct a cybersecurity risk assessment.
- Protect the robots from unauthorized access.
- Manage users
- Implement an auditing system

**Document emergency plans for all potential emergency situations.** See "[Define emergency plans](#)" on [page 466](#).

- Identify possible emergency situations.
- Implement preventative actions to reduce the likelihood and severity of the emergency.
- Document what must be done and who is responsible for each task if the event should occur.

**Monitor site events and analyze data for performance improvements and unintended or suspicious behavior.** See "[Monitor system](#)" on [page 470](#).

- Determine relevant monitoring tools and information.
- Define behavior to be acted on.
- Set up filters on Event viewer to detect relevant messages in MiR Fleet.
- Set up an application to monitor important messages from MiR Fleet audit log.
- Define responsibilities for incoming alerts.
- Consider setting up webhook subscriptions to events of interest.
- Consider purchasing a MiR Insights license.

### 4.11.1 Evaluate cybersecurity

#### Section overview

Assess the cybersecurity of your MiR system.

- Conduct a cybersecurity risk assessment.
- Protect the robots from unauthorized access.
- Manage users
- Implement an auditing system

Cybersecurity in the context of MiR products means protecting IT (Information Technology) and OT (Operational Technology) assets from unauthorized access, use, disruption, modification, or destruction.

The following lists all the actions we recommend you take to increase the cybersecurity of your site. For an overview of all the cybersecurity features that are automatically applied or are features you choose to use, see ["Security" on page 101](#).

Actions you should complete on **MiR Fleet** and the whole system:

- Conduct a cybersecurity risk assessment and define regular security maintenance to ensure that your security is up-to-date—see ["Risk assessment guiding questions" on the next page](#).
- Check that your MiR Fleet installer is signed and verified before installing MiR Fleet—see ["Install MiR Fleet" on page 217](#).



- Use a Managed Service Account to run the MiR Fleet service—see ["User for MiR Fleet Windows Service" on page 220](#).
- Manage your users and passwords so all users have a unique password and only have the necessary permissions to complete their tasks—see ["Create users and roles" on page 257](#).
- Automated, continuous monitoring of the Audit logs can help detect anomalies during the operation of the MiR Fleet—see ["Monitor system" on page 470](#).
- When decommissioning, follow data disposal procedures for the disk of the robot computer and for the external MiR Fleet database—see ["Configure MiR Fleet" on page 234](#).

Actions you should complete on each **MiR robot**—see ["Add robots" on page 276](#) for step-by-step instructions:

- Disable the internal Wi-Fi access point in older robots so they do not broadcast a network connection that can be accessed by attackers.
- Change the password for the access point device if you continue to use these while the robot is operating.
- Update the access point device if you continue to use these while the robot is operating.
- Restrict the unused functionalities on robots to close off potential communication interfaces attackers can access.
- Protect the BIOS and open ports of your robot and server.

### Risk assessment guiding questions

Before operating a MiR system, it is essential to conduct a cybersecurity risk assessment. Consider the following topics to ensure a secure system.

MiR system should not be operated on open and unsecured networks. Limit system access to authorized personnel only. Consider the following options:

- Use segmented networks to separate the MiR system from other assets within the company.
- Implement strict firewall rules to allow only authorized traffic between different network segments.
- Protect against physical access to the facility where robots are operated and on the robot itself using port locks. Consider who has access to the robots and where they are stored.
- Disable any Wi-Fi access points on older robots after setup as part of the commissioning

process.

- Review users and roles in MiR Fleet to ensure that only authorized personnel can access the MiR system.

Unauthorized access to the MiR system can cause significant damage. Consider the following potential risks:

- Interruption of mission scheduling and execution.
- Disruption of the robot's connection.
- Infection of robots with malware.
- Remote operation of robots by unauthorized parties.

To reduce the impact if your system is attacked, create frequent backups of the system in a known good state to avoid any potential data loss.

To reduce the risk of your system becoming outdated, create a regular maintenance schedule to check all security related configurations and settings. Part of the regular maintenance should also be to review if there are new standards, site modifications, or other changes that also needs to be added to the maintenance schedule.

## 4.11.2 Define emergency plans

### Section overview

Document emergency plans for all potential emergency situations.

- Identify possible emergency situations.
- Implement preventative actions to reduce the likelihood and severity of the emergency.
- Document what must be done and who is responsible for each task if the event should occur.

The following table identifies common scenarios you should create and document emergency plans for.

<b>Emergency</b>	<b>Risks</b>	<b>Preventative actions</b>	<b>On the spot actions</b>
MiR Fleet loses power	Robots are no longer assigned missions. Robots are not blocked from occupied resources and can create a deadlock.	Use an uninterruptible power supply (UPS).	Once MiR Fleet is powered, check for deadlock alerts and the mission scheduler.
One robot loses power	The robot blocks other robots, vehicles, or personnel from completing their tasks. The robot cannot complete tasks it was assigned leading to production downtime	Configure the Auto charging system to ensure robots are not low on power. Ensure good Wi-Fi connection across the site.	Turn on the robot and either send it to the nearest charging station, or push it to a power supply and connect a cable charger.
Wi-Fi network loses power	Robots are no longer assigned missions. Robots are not blocked from occupied resources and can create a deadlock.	Ensure good Wi-Fi connection across the site. Ensure the Wi-Fi system is stable and is hooked up to an uninterruptible power supply (UPS).	Restore the network as soon as possible. Once the MiR system is connected, check for deadlock alerts and the mission scheduler.

Emergency	Risks	Preventative actions	On the spot actions
<p>Building emergency</p>	<p>Depends on the extent of the emergency. Can range between production downtime, loss or damage to equipment, or any level of injury, including fatality of personnel.</p>	<p>Define Try/Catch actions to handle minor emergencies where robots need to navigate around an occupied area.</p> <p>Define Emergency zones and positions to handle emergencies where all robots must evacuate an area. Include robot management in emergency protocol.</p>	<p>Execute the standard emergency protocol and any robot-specific tasks added to this.</p>
<p>Someone deletes an important site component or overwrites the whole site</p>	<p>Robots may fail certain tasks or complete them incorrectly.</p>	<p>Take regular data backups of the .site file or the SQL database.</p>	<p>Restore the data. Determine who caused the data loss, and revise user access levels if necessary.</p>
<p>MiR Fleet is uninstalled</p>	<p>Robots are not given orders until MiR Fleet is reinstalled. All robots must be reconnected, and all custom configuration outside the database must be set up again.</p>	<p>Take regular data backups of the .site file or the SQL database.</p>	<p>Restore the data. Determine who uninstalled MiR Fleet, and revise user access levels if necessary.</p>

Emergency	Risks	Preventative actions	On the spot actions
Database becomes corrupt or is lost	All data since your last database back up is lost.	Take regular data backups of the .site file or the SQL database.	Restore the data. Determine who caused the data loss, and revise user access levels if necessary.

In many of these cases, MiR Fleet is either restarted or loses connection to MiR robots.

### Evacuations

You can set up zones that are cleared of robots in case of an emergency. Evacuation zones should be created and Evacuation positions should be added to the map—["Evacuations" on page 20](#).

### Alerts

Use the Alerts page to see when robots report errors, enter Protective stop or Emergency stop, and when robots trigger a deadlock—see ["Alerts" on page 9](#).

### Event viewer

Go through the Event viewer record after an emergency situation to see if anyone made modifications to the site and if there are any additional issues in the MiR system that must be resolved.

### Backups

Create backups regularly to avoid data loss from unexpected events.

You can manually create backups of your data by:

- Creating a back up of the MiR Fleet Microsoft SQL database.
- Exporting a .site file from the Settings page.

The following data backups are also generated automatically:

- A .site file of the current site is always generated when you import and overwrite with a new site.
- An application backup of MiR Fleet is generated when you update MiR Fleet in case the update fails. The backup is removed once the update finished successfully.

### Redundancy and hot spare

MiR Fleet cannot run in a redundant system where two symmetrical instances of MiR Fleet are running.

MiR Fleet cannot run with a hot spare where you have a MiR Fleet on standby that activates if your main instance fails.

You can have a cold spare, an inactive copy of MiR Fleet, ready to be activated in case your main instance fails. Make sure to keep your cold spare up-to-date every time you make a change, and ensure that it's IP address is the same as your main instance. Never have both instances running at the same time.

### 4.11.3 Monitor system

#### Section overview

Monitor site events and analyze data for performance improvements and unintended or suspicious behavior.

- Determine relevant monitoring tools and information.
- Define behavior to be acted on.
- Set up filters on Event viewer to detect relevant messages in MiR Fleet.
- Set up an application to monitor important messages from MiR Fleet audit log.
- Define responsibilities for incoming alerts.
- Consider setting up webhook subscriptions to events of interest.
- Consider purchasing a MiR Insights license.

#### Alerts

Assign a user or group of users responsible for monitoring the Alerts page—see ["Alerts" on page 9](#)—and for resolving any reported issues.

### Event Viewer and audit logs

Use notifications in the Event Viewer to monitor functional events—see ["Logging" on page 40](#).

Create an application to follow the Audit log and trigger a reaction if any suspicious behavior is reported—see ["Logging" on page 40](#).

The audit log is never rotated automatically. You must implement a protocol to clear the log after a given size or date to avoid running out of disk space. If you delete, rename, or move the audit log file, a new one is automatically generated.

### MiR Insights

MiR Insights runs continuously alongside MiR Fleet to give real-time data on several metrics. For more information about MiR Insights, see *MiR Insight User Guide*. You can find this guide on [MiR Support Portal](#).

### MiR Fleet Integration API

Set up subscriptions in the integration APIs to receive event data when certain events occur—see ["MiR Fleet Integration API" on page 484](#).

## 4.12 Prepare handover

### **After reading this chapter, you can:**

Handover the MiR system and ensure that the users of the system know how to use and maintain the system and ensure continued safety.

**Ensure your MiR system conforms to relevant standards to maintain a safe solution.** See ["Assess conformity" on the next page](#).

- Assess the system conformity regularly against the latest relevant standards.
- Collect a technical file with all relevant system documentation and information of use for the complete system.
- Establish a maintenance schedule.
- Assess any custom top module or modification made to approved top modules.

**Train users so they can correctly and safely execute tasks they are responsible for.** See ["Train users"](#) on page 475.

- Determine the responsibilities of your users and categorize them into user types.
- Train users in their relevant tasks and knowledge of the MiR system.
- Keep training records and schedule new training sessions if new equipment or processes are introduced.

### 4.12.1 Assess conformity

#### Section overview

Ensure your MiR system conforms to relevant standards to maintain a safe solution.

- Assess the system conformity regularly against the latest relevant standards.
- Collect a technical file with all relevant system documentation and information of use for the complete system.
- Establish a maintenance schedule.
- Assess any custom top module or modification made to approved top modules.

You must ensure that when you handover the MiR system, it complies to all relevant standards, directives, laws and regulations. All documentation certifying the system must be included in a Technical file that fully describes the MiR system.

After handing over the system, you must also establish how the users keep the MiR system compliant at all times and ensure continued compliancy with new standards.

#### Conformity assessment

It is your responsibility to ensure that your MiR system complies to all relevant standards, directives, laws and regulations at all times. You must ensure any of the following:

- You have a competent team that ensures current standards are met and that they maintain their competency. You must always be able to ensure the MiR system stays up-to-date with all relevant standards.



- You have a competent third party company to assess the MiR system and have scheduled regular checks to ensure that the MiR system stays up-to-date with all relevant standards.

You must assess your MiR system when it is installed and in regular intervals afterwards to ensure continued conformity to standards. It is the commissioner's responsibility to determine a suitable schedule for re-assessing the MiR system. The aspects you must ensure include:

- Risk assessments
- Conformity assessments
- Relevant certifications and markings
- Required safeguarding

Upon request, MiR or a designated agent can audit the skills and capabilities of the team you have made responsible for assessing the conformity of your MiR system.

Within EU, your MiR system must comply to the following standards:

- Machinery Directive
- Radio Equipment Directive
- Other standards depending on your top module design.

### **Robots used as partly completed machinery**

If you have purchased a MiR robot as partly completed machinery (PCM), it is not known or determined by Mobile Industrial Robots how the application will be used. Adequate risk reduction measures and a risk assessment must be implemented prior to handover to the end user.

It must be agreed between the commissioner and the end user who shall make the CE marking for the application and draw up a Declaration of Conformity.

The following is meant as advice to the CE marking process

- Access to moving parts must be prevented by means of safeguards or protective devices. The Essential Health and Safety Requirement EHSR 1.3.7 in the Machinery Directive (MD) addresses risks related to moving parts.
- It is required to prevent persons from coming so close to the robot, if it can not reliably stop before collision. This assessment is in line with current safety standards for Automated Mobile Robots, for example ISO 3691-4:2020.

- With reference to MD Clause 1.1.2 (b), because PCM robots operate without an efficient speed and separation monitoring system, the best method of minimizing the risks of using a PCM robot is to inform users of risks. The following examples of information can be employed where relevant:
  - Information and signs on the machine
  - Acoustic or light signals
  - Information and signs in the environment
  - Only allow people with special training access to the areas where the robots operate
  - Specify PPE (personnel protection equipment), and make sure it is used

### Technical file

When you hand over the MiR system within EU, you must include a technical file in your handover. The technical file must adhere to Annex VII in the Machinery Directive. The technical file must be stored for at least 10 years from the day of handing over the finished MiR system to the customer.

You may include documentation created by MiR in your technical file, but MiR documentation alone cannot sufficiently describe the integrated MiR system and your specific use of MiR products.

### Maintenance schedule

As part of the technical file, you must include a maintenance schedule.

You can find information about how to maintain all MiR robot application in the user guide or integrator manual for the application. The frequency of each maintenance task depends on the robot's working environment and tasks. This should be reflected in the maintenance schedule. We recommend revising the maintenance schedule regularly to update the intervals based on the robots state with the current intervals.

You must also include the required maintenance of all other equipment in your MiR system, including any custom top modules.

For more information about MiR robot maintenance, see:

- For MiR100 and MiR200, see the Maintenance section in the user guide.
- For MiR250, see the [MiR250 Maintenance Guide](#).
- For MiR500 and MiR1000, see the Maintenance section in the user guide.
- For MiR600 and MiR1350, see the [MiR600 and MiR1350 Maintenance Guide](#).

You can find these guides on [MiR Support Portal](#).

### Custom top modules

When you mount a custom top module to a robot, you must ensure that the top module and robot combination is compliant with all relevant standards.

MiR robots are compliant with the EU Radio Equipment Directive. If you do not modify any existing radio signal devices on the robot or add any additional devices, the combined application remains compliant with the EU Radio Equipment Directive.

Any electronic top modules must be compliant with the local EMC regulations.

Other relevant regulations depend on design and function of the top module. Investigate which regulations would be relevant before completing the top module design.

### 4.12.2 Train users

#### Section overview

Train users so they can correctly and safely execute tasks they are responsible for.

- Determine the responsibilities of your users and categorize them into user types.
- Train users in their relevant tasks and knowledge of the MiR system.
- Keep training records and schedule new training sessions if new equipment or processes are introduced.

When you hand over the robot application, you must ensure that all relevant users have received the necessary training and that there are training plans for future users and updates in the application.

The robot is only intended to be used by personnel that have received training in their required tasks.



#### NOTICE

The commissioner is obligated to provide information to limit hazards, and the training must address hazards from the risk assessment created during the commissioning.

## User responsibilities

There are three main user types that interact with MiR Fleet Enterprise. All other persons in the vicinity of the robot application are considered indirect users and must know how to act when they are close to the robot. For example, they must be aware that visibly-marked operating hazard zones must be respected.

### Direct users

Direct users are familiar with the safety precautions in this manual and have the following main tasks:

- Assigning missions to the robot.
- Fastening loads to the securely.
- Loading and unloading from a paused robot.

### Operators

Operators of MiR Fleet Enterprise must comply with their respective countries' laws and regulations regarding operating forklifts.

Operators have thorough knowledge of the robot application and of the relevant safety precautions. Operators have the following main tasks:

- Servicing and maintaining the robot application.
- Creating and changing missions and map features in the robot interface.
- Ensuring that visitors and personnel are properly equipped and briefed regarding robot safety before entering the robot's work environment.

### Commissioners

Commissioners have thorough knowledge of all aspects of commissioning, safety, use, and maintenance of the robot application and have the following main tasks:

- Commissioning of the product. This includes creating maps and restricting the user interface for other users.
- Conducting the risk assessment.

- Determining the payload limit, weight distribution, safe fastening methods, safe loading and unloading of loads on the robot application, and ergonomic loading and unloading methods if relevant.
- Marking operating hazard zones.
- Ensuring that operators and direct users receive the necessary training for their responsibilities—see "[User training](#)" below.

## User training

When training direct users and operators, the training must adhere to the following:

- Be conducted in an area away from other workplace activities.
- Be conducted under the supervision of a trainer.
- Include all operating tasks.

Appropriate training records must be kept, and retraining must be supplied when new equipment is introduced, existing equipment is modified, or operating conditions are changed.



[MiR Academy](#) offers training material that covers most of the topics required for operators and direct users. We recommend operators complete Bronze and Silver level training and direct users complete Bronze level training at least. You can also apply for Specialist Classroom Training, which provides hands-on practice that builds on Bronze and Silver level training. Online training only offers training for how to use the MiR robot. You must supply additional training specific to the application the MiR robot is used for.

### Training of direct users

**Table 4.12** Training for operating the robot

Knowledge of	Trained to
Intended use	<ul style="list-style-type: none"> <li>• Apply the robot for tasks it is designed and well-suited for.</li> <li>• Identify what environments the robot can safely operate in.</li> </ul>
Light indicators and sounds	<ul style="list-style-type: none"> <li>• Understand the robot's status and driving intentions.</li> </ul>
Power, Restart, Stop, Emergency stop buttons, and Operating mode key	<ul style="list-style-type: none"> <li>• Turn the robot on and off.</li> <li>• Stop the robot manually.</li> <li>• Use the relevant functions on the display.</li> <li>• Resume operation after the robot has stopped.</li> <li>• Push or pull the robot by hand.</li> <li>• Change the robot's mode.</li> </ul>
Charging interfaces and charging options	<ul style="list-style-type: none"> <li>• Charge the robot.</li> </ul>
Batteries and battery connection	<ul style="list-style-type: none"> <li>• Disconnect and connect batteries.</li> <li>• Remove batteries.</li> <li>• Handle batteries outside of the robot safely.</li> </ul>
Operating instructions	<ul style="list-style-type: none"> <li>• Send the robot to positions on the map and queue missions.</li> </ul>

**Table 4.13** Training for Emergency situations or event of failure

Knowledge of	Trained to
Emergency stop buttons	<ul style="list-style-type: none"><li>• Use Emergency stop buttons in case of an Emergency where the robot must be stopped immediately.</li></ul>
Errors	<ul style="list-style-type: none"><li>• Resolve common errors, and make the robot continue its mission or send it on a new one.</li></ul>
Escape routes and emergency procedure	<ul style="list-style-type: none"><li>• React correctly and quickly in an Emergency situation in case of any dangerous scenarios described in this manual or other unexpected scenarios.</li></ul>
Operating the tiller for MiR1200 Pallet Jack	<ul style="list-style-type: none"><li>• Control the robot manually using the tiller.</li></ul>

**Table 4.14** Training for safety

Knowledge of	Trained to
Floor markings	<ul style="list-style-type: none"><li>• Behave correctly and safely within pedestrian areas, operating hazard zones, and other marked areas on the site.</li></ul>
Stability characteristics with and without load	<ul style="list-style-type: none"><li>• Operate the robot correctly according to the load.</li><li>• Load the robot correctly and below the maximum payload.</li></ul>
Residual risks	<ul style="list-style-type: none"><li>• Avoid or behave safely in any scenarios where the robot can pose a risk to personnel or equipment.</li><li>• Maintain the means that have been applied to mitigate the risks.</li></ul>
The robot's safety functions	<ul style="list-style-type: none"><li>• Identify in which scenarios the robot's safety functions are not sufficient for maintaining safety, for example, when the robot's Protective fields are muted, and how to behave to ensure safety.</li><li>• Interact confidently with the robot in scenarios where the safety functions ensure safety.</li></ul>



**Table 4.15** Training for understanding how the robot works

Knowledge of	Trained to
Robot's sensors	<ul style="list-style-type: none"> <li>Identify cases where the sensors may not be able to detect obstacles.</li> <li>Troubleshoot simple issues where the sensors are blocked, dirty, or are unable to detect the object type.</li> </ul>
Navigation and localization	<ul style="list-style-type: none"> <li>Troubleshoot localization issues.</li> </ul>
Global and local path planning	<ul style="list-style-type: none"> <li>Troubleshoot simple path planning issues where the robot's path has been blocked.</li> </ul>
Wi-Fi	<ul style="list-style-type: none"> <li>Connect to the robot over Wi-Fi.</li> </ul>
Ideal operating environments	<ul style="list-style-type: none"> <li>Ensure that the robot's work area supports the robot's requirements and limitations.</li> </ul>

**Table 4.16** Training for maintenance

Knowledge of	Trained to
Regular cleaning and checks	<ul style="list-style-type: none"> <li>Clean and check the robot regularly—see the section <i>Maintenance</i> in the user guide or integrator manual for your robot application.</li> </ul>

### Training of operators

Operators must be trained in all of the same content as direct users and in the additional content in the following tables.

**Table 4.17** Training for site configuration

Knowledge of	Trained to
Maps	<ul style="list-style-type: none"><li>• Update the map with new markers and positions.</li></ul>
Missions	<ul style="list-style-type: none"><li>• Create application specific missions.</li><li>• Improve mission robustness.</li><li>• Nest missions and using arguments to reuse mission material efficiently.</li></ul>
Calibrations	<ul style="list-style-type: none"><li>• Calibrate markers, shelves, and carts.</li></ul>
Wi-Fi	<ul style="list-style-type: none"><li>• Identify symptoms of poor Wi-Fi coverage.</li><li>• Identify ways to improve Wi-Fi coverage.</li></ul>

**Table 4.18** Training for maintenance

Knowledge of	Trained to
Regular maintenance and replacement of parts	<ul style="list-style-type: none"><li>• Clean and check the robot regularly—see the section <i>Maintenance</i> in the user guide or integrator manual for your robot application.</li></ul>
Internal parts	<ul style="list-style-type: none"><li>• Identify the robot's internal parts and know their purpose.</li></ul>

**Table 4.19** Training for MiR Fleet

Knowledge of	Trained to
Robots on MiR Fleet	<ul style="list-style-type: none"><li>• Add and remove robots to and from MiR Fleet.</li><li>• Troubleshoot synchronization issues.</li><li>• Update the software version of robots.</li></ul>
Missions on MiR Fleet	<ul style="list-style-type: none"><li>• Schedule missions.</li><li>• Review mission statuses.</li></ul>

### Visitor and indirect user training

For all personnel or visitors who access the operating environment of the robot but do not interact directly with the robot, you must provide a brief training including, but not limited to, the following topics:

- MiR robots navigate around obstacles automatically.
- If a person enters the active Protective field around the robot, the robot will immediately engage the brakes and the status light turn red.
- After the robot has stopped for a person, it will begin operating again when the field is clear.
- Which areas on the site the robot operates in and where it is not allowed.
- What they must do in case an event from any defined Emergency plan occurs—see "[Define emergency plans](#)" on page 466.

## 5. APIs and integration

MiR Fleet Enterprise includes two application platform interfaces (API) to integrate MiR Fleet and MiR robots with other devices. MiR also offers an additional application to support select 3.x API endpoint requests.

The interfaces are intended for the following purposes:

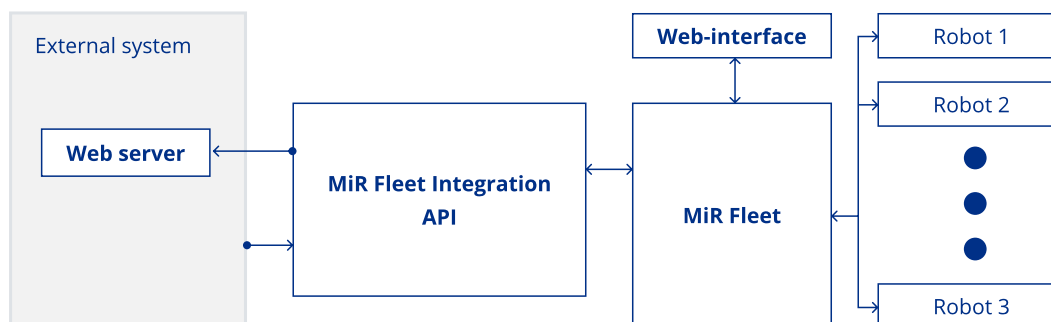
- Use MiR Fleet Integration API to set up automated communication between MiR Fleet and external devices—see "[MiR Fleet Integration API](#)" below.
- Use Top Module API to set up automated communication between a custom top module on a MiR robot and external devices—see "[Top Module API](#)" on page 582.
- Use the Shopfloor connector application to make your software 3.x application compatible with MiR Fleet Integration API—see "[Shop Floor Connector](#)" on page 596.
- Use the robot API for direct communication and integration with robots. The robot API can only view and change the same parameters that are available in the robot interface—see "[Robot APIs](#)" on page 620 for the full list of available endpoints.

### 5.1 MiR Fleet Integration API

The MiR Fleet Integration API is intended to integrate your MiR system into an external system through MiR Fleet. Use it after you have set up your MiR system using the web-interface.

The API does not interact directly with MiR robots. The API connects the external system to MiR Fleet, and MiR Fleet manages robots based on these inputs.

MiR Fleet Integration API is based on the RESTful architectural style.



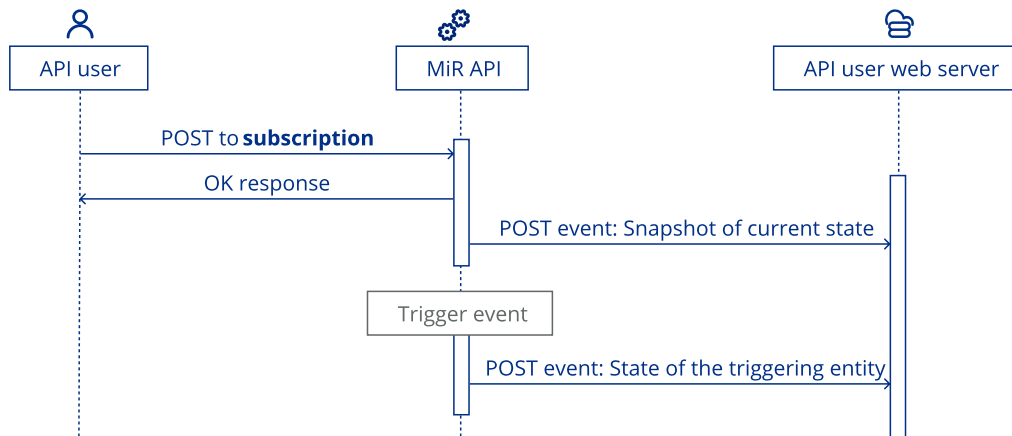
## Webhooks

The API is highly event-based, and most use cases involve webhooks to receive data—see ["Webhooks" on page 490](#).

You can set up webhooks that you subscribe to events in MiR Fleet. When a triggering event occurs, the webhook server you have set up a subscription for receives a POST request from the API with information about the event—see ["Event subscriptions" on page 496](#).

Use webhooks for the following:

- Receive updates whenever anything changes in the MiR Fleet system.
- Program system responses when certain events occur.
- Monitor mission and robot statuses.



## Endpoints

Use the API endpoints to send data to or request data from MiR Fleet—see ["Endpoints" on page 525](#).

Most data is retrieved from event subscriptions through webhooks—see ["Webhooks" above](#).

There are examples of what you use endpoints for:

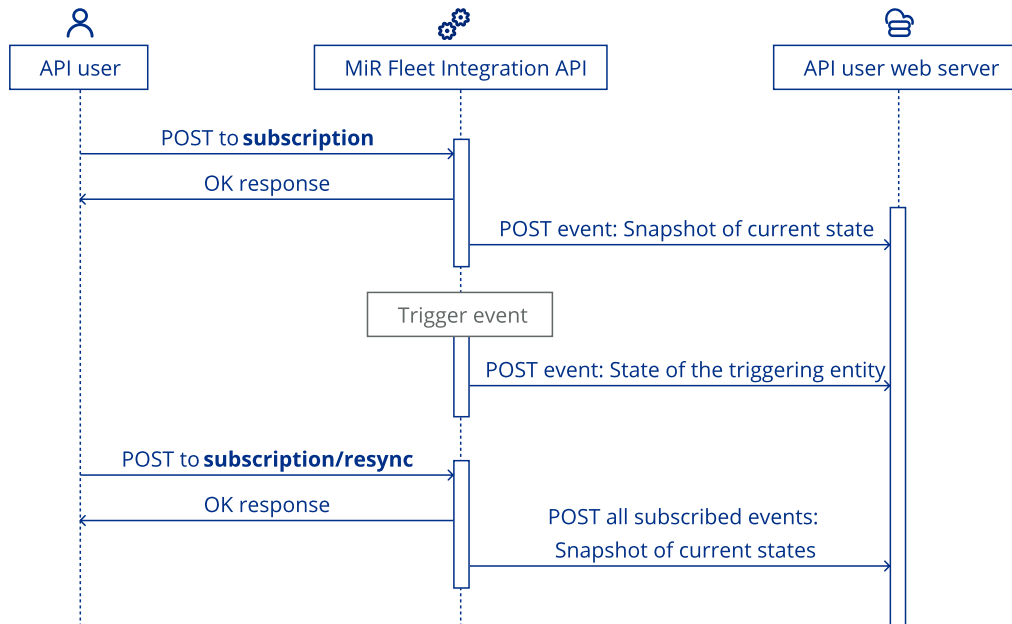
- Create, delete, or modify components in MiR Fleet
- Schedule serial-orders
- Set up webhooks by subscribing to events
- Request information or updates from MiR Fleet

### Snapshots

When you subscribe to certain events, the first event your webhook client receives is a snapshot of the current state of the selected entity on MiR Fleet Integration API. This applies for the following events you can subscribe to:

- Site
- RobotIdentity
- System
- SerialOrderStatus

At any time, you can request a resync using the `subscription/resync` endpoint to make all active subscriptions send a new snapshot.



### Serial-orders

Serial-orders are unique to MiR Fleet Integration API—see "[Serial-orders](#)" on page 521.

A serial-order is a defined sequence of missions that you want a single robot to execute consecutively.

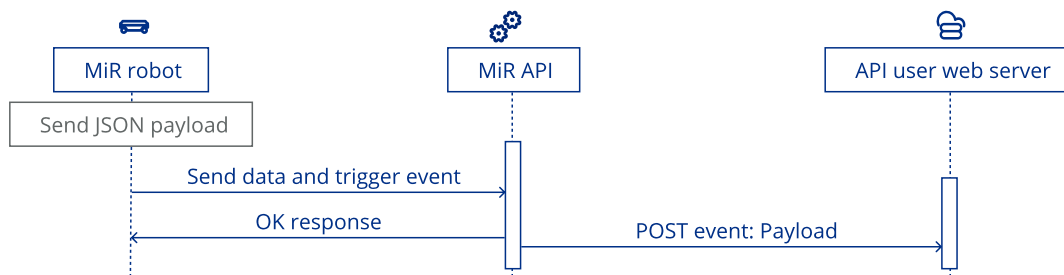
Serial-orders consist of any number of phases. Each phase specifies a mission the robot must execute. When a robot is assigned a serial-order, it executes all of the phases in the specified order. If a mission in a phase uses arguments, you can pass values to the arguments.

### Integration actions

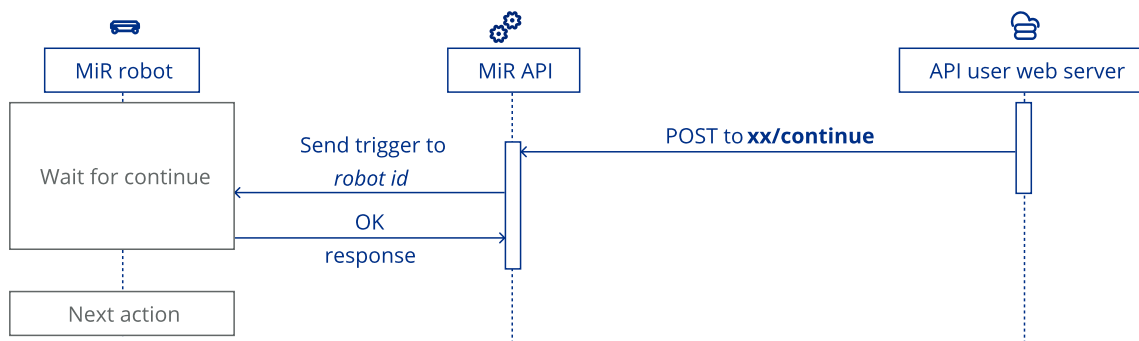
There are two actions you can use in missions to enable direct communication to MiR Fleet Integration API during a mission.

**Send JSON payload** triggers the robot to send a JSON payload via the chosen API. Subscribe to the Payload event for the chosen API to make the API push the payload event to the subscribed endpoint. The event also includes the ID of the robot that executed the action to trigger the event.

There are no restrictions to the format of the payload. This action is designed to enable any custom communication to be sent to your own client when a robot reaches a specific part of its mission.



**Wait for continue** makes the robot wait until you POST the robots ID to `api/v1/system/robot/continue` or `api/v1/continue`.



Use these two actions in combination to make the robot communicate its status and wait to continue its order until it receives a confirmation. You can use this in combination with undefined phases or arguments, so the robot waits before starting the next phase until everything is defined.

## Zones

There are two zones you can use in maps to enable direct communication to MiR Fleet Integration API in certain areas. You can control either zone using the `api/v1/site/zone` endpoint—see ["Endpoints" on page 525](#).

**Lock zones** prevent all robots from entering an area when the zone is active—see ["Zones" on page 62](#). You can lock and unlock the zone with `POST /site/zone/lock`.

**Integration zones** can be set to trigger the robot to send a JSON payload via MiR Fleet Integration API or MiR Top Module API—see ["Zones" on page 62](#). For MiR Fleet Integration API to receive the payload, subscribe to Payload—see ["Payload" on page 509](#)—and define which endpoint you want the API to POST the payload to.

### 5.1.1 Getting started

#### Section overview

- Determine the address you use to send requests to endpoints.
- View the Swagger page with all available endpoints.
- Create an API key to authenticate your requests.
- Get started creating webhooks.

The base URL address of the Fleet Integration API is the IP address of MiR Fleet followed by `api/v1`: `<MiR_Fleet_IP_address>/api/v1`. To access any API endpoint, add it to base URL.

You can see an overview of all the endpoints under ["Endpoints" on page 525](#).

You can try out any of the endpoints using the Swagger interface. Access the interface by adding `/swagger` to the end of the base URL: `https://MiR_Fleet_IP_address/swagger`.

None of the output messages from MiR Fleet Integration API are encoded. This enables you to integrate the level and type of protection that is suitable for your integration.

#### API keys

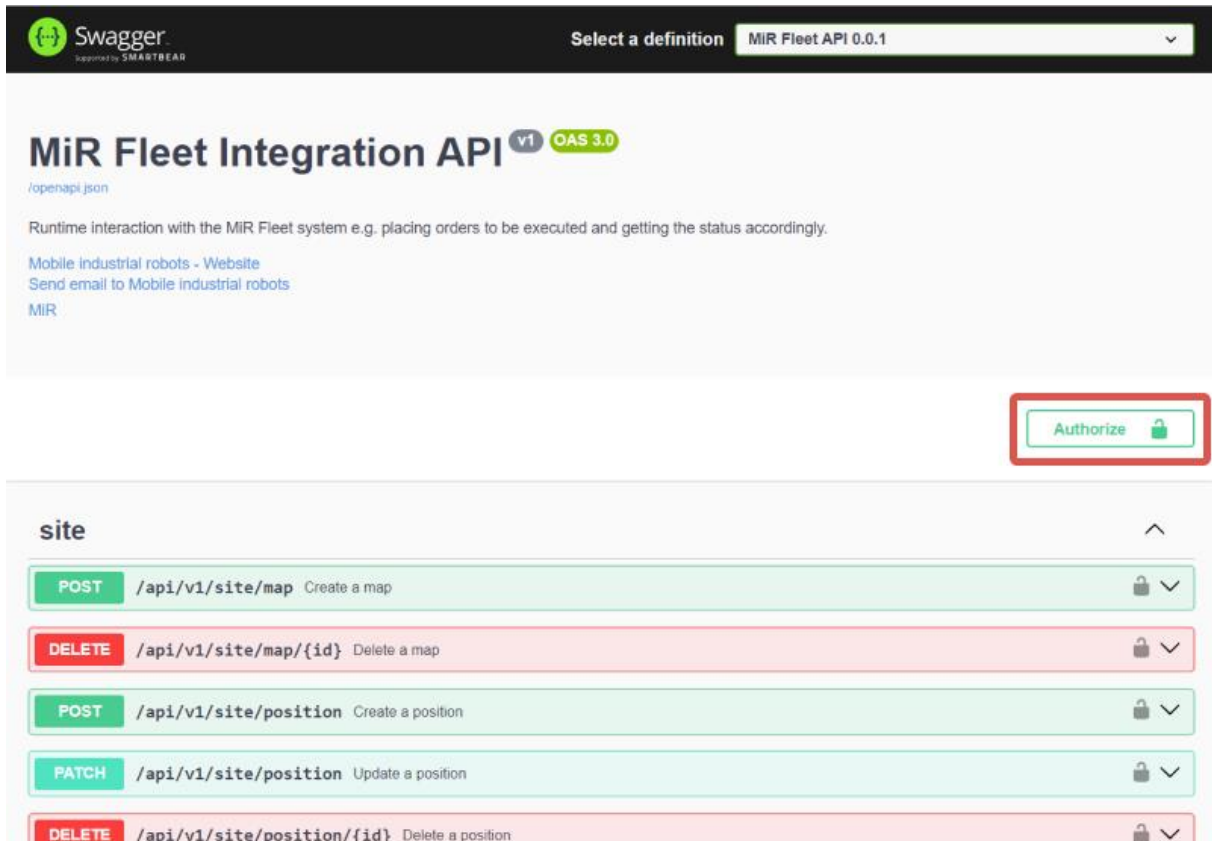
Each request to an endpoint must contain an API key.



You can request an API key in the web interface under **Setup > Users > API keys**—see ["Roles and users" on page 95](#). When you request a key, it is only shown once after the request. Save the API key in a secure location for later use. If you lose the API Key, you must generate a new key.

In each request you send to MiR Fleet, the request must include an **x-api-key** header where you pass the API key.

If you are using the Swagger page, you can enter the API key in the upper-right corner for access.



The screenshot displays the Swagger UI for the MiR Fleet Integration API. At the top, the Swagger logo is visible, along with the text "Select a definition" and a dropdown menu showing "MiR Fleet API 0.0.1". The main heading is "MiR Fleet Integration API" with "v1" and "OAS 3.0" badges. Below the heading, there is a link to "/openapi.json" and a description: "Runtime interaction with the MiR Fleet system e.g. placing orders to be executed and getting the status accordingly." There are also links for "Mobile industrial robots - Website", "Send email to Mobile industrial robots", and "MiR". A red box highlights the "Authorize" button in the top right corner. Below the header, there is a list of API endpoints under the "site" group:

Method	Endpoint	Description	Lock Icon	Dropdown
POST	/api/v1/site/map	Create a map	🔒	▼
DELETE	/api/v1/site/map/{id}	Delete a map	🔒	▼
POST	/api/v1/site/position	Create a position	🔒	▼
PATCH	/api/v1/site/position	Update a position	🔒	▼
DELETE	/api/v1/site/position/{id}	Delete a position	🔒	▼

## Webhooks

The MiR Fleet Integration API is intended to use webhooks that receive event information when triggering events occur. To set up a webhook to listen to events:

- Create a webhook client—see ["Webhooks" on the next page](#).
- Use the subscription endpoint to subscribe your webhook server to a specific events type—see ["Subscription" on page 538](#) and ["Event subscriptions" on page 496](#).

- Program your client to parse the event data it receives when an event occurs. Use the event-contracts to see the data format—see ["Schemas and types" on page 552](#).
- Program the client to send appropriate responses to other devices, interfaces, or MiR Fleet endpoints based on the events that occur—see ["Endpoints" on page 525](#).

We recommend starting with subscribing to Error events and displaying all error events in an interface. Error events report any errors that occur while you are using the API—see ["Error" on page 502](#).

## 5.1.2 Webhooks

### Section overview

- How to create a webhook client
- How to set up a subscription for the client
- How MiR Fleet communicates with the client

Set up webhook clients to listen for events that you can subscribe to—see ["Event subscriptions" on page 496](#).

### Client Setup

MiR Fleet Integration API is event-based by design and sends the events as JSON objects in POST requests to the endpoints implemented in your webhook client. The endpoints should follow the structure specified in the event-contracts—see ["Event subscriptions" on page 496](#).

Your webhook client application must:

- Be an API based on REST principles.
- Be able to receive an event. For this it must have:
  - A controller with an implemented POST method.
  - A data structure corresponding to the expected event-contracts—see ["Event subscriptions" on page 496](#). You can also use the `GET api/v1/subscription/event-contracts` method to receive a list of all the event-contracts.
- Have a controller for every event type that the user is expecting to receive since all events have different data structures.

- Have an endpoint addresses that specifies a full URL (not just the base URL) for specific event publishing. For example, the address for Serial-order status events might look like:

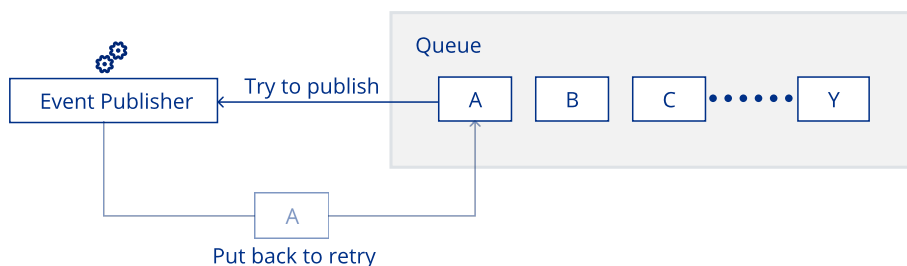
`http://some-address/events/serial-order-status.`

The following example shows the contract for the subscription event that is published when a subscription is added.

```
{
  "base-url": "https://test1.com",
  "endpoints": [
    {
      "event-type": "Subscription",
      "endpoint-paths": [
        "p1",
        "p2"
      ]
    },
    {
      "event-type": "Site",
      "endpoint-paths": [
        "p3"
      ]
    }
  ],
  "state": "Subscribed",
  "timestamp": "2022-09-19T09:08:12.8651699Z"
}
```

The API does not store any data. It only sends the data from MiR Fleet to the user when the data related to an event changes. Your webhook or external system must handle the data sent by the API as you intend. The API has the following behavior:

- All events are sent to the user in the sequence that they have been received from the MiR Fleet system.
- If your receiving server is down, the API retries sending the event using an Exponential Back Off Strategy. It tries sending the event the number of times you have specified the `max-retries` parameter to when you set up the subscription. The wait time between retries is first with 100 ms, then 500 ms, and then 2 500 ms intervals for all attempts after the second retry. The default number of retries is three.



## Access tokens

If an access token is provided in the request upon subscription, then MiR Fleet will send this token with every request to the customer's webhook client.

The access token is passed in the header using the Bearer authentication format:

```
-H "Authorization: Bearer <token>"
```

The token must meet the following requirements:

- Be a valid Microsoft access token
- Contain 256–2048 characters
- Contain numbers
- Contain upper and lower case letters
- Contain underscore
- Cannot contain special characters

## Server certificate

If an HTTPS URL is used, then the machine where MiR Fleet is running will validate the certificate as part of establishing a connection to the server. This means that the MiR Fleet machine will read the web server's certificate and only establish a connection if it matches a trusted certificate or if there is a trusted certificate chain to that certificate in the request.

The certificate should be installed in the appropriate certificate store on the machine running MiR Fleet.

A flag can be set when subscribing to enable the limited TLS feature. This means that even though an HTTPS endpoint is used, the TLS errors will be ignored. As such, the connection will be insecure.

## Encoding

The output events from the API are not encoded.

## Code examples for listening for POST requests

The following code examples show how to get started with making a simple client to listen for events on a specific endpoint and print the received data to a terminal.

All of the code examples are based on a subscription where "base-url":  
"https://1.2.3.4:5000/api" and "endpoint-paths":["/data/path"]].



#### NOTICE

The following code examples are autogenerated. They are not tested or verified. MiR takes no responsibility in the design or functionality of your Webhook client. The code examples are provided for inspiration and to help you get started.

### Python

This example uses Flask. You can install Flask with the command:

```
pip install Flask
```

```
from flask import Flask, request, jsonify

app = Flask(__name__)

@app.route('/data/path', methods=['POST'])
def data_path():
    # Get the data from the POST request
    data = request.get_json()
    if not data:
        response = {"error": "No data provided"}
        print(response) # Print the response to the terminal
        return jsonify(response), 400

    # Process the data (this is just an example)
    processed_data = {"received_data": data}

    # Print the received data to the terminal
    print(f"Received data: {data}")

    # Return a response
    return jsonify(processed_data), 200

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

For more information, see <https://flask.palletsprojects.com/en/stable/quickstart/>.

### C++

This example uses Crow and Boost. In Ubuntu, you can install these dependencies with:

```
sudo apt-get update
```

```
sudo apt-get install libboost-all-dev
```

```

#include "crow_all.h"
#include <iostream>

int main()
{
    crow::SimpleApp app;

    CROW_ROUTE(app, "/data/path").methods(crow::HTTPMethod::POST)([](const
crow::request& req) {
        auto body = crow::json::load(req.body);
        if (!body)
        {
            crow::json::wvalue response;
            response["error"] = "No data provided";
            std::cout << response << std::endl; // Print the response to the terminal
            return crow::response(400, response);
        }

        // Process the data (this is just an example)
        crow::json::wvalue processed_data;
        processed_data["received_data"] = body;

        // Print the received data to the terminal
        std::cout << "Received data: " << body << std::endl;

        return crow::response(200, processed_data);
    });

    app.port(5000).multithreaded().run();
}

```

For more information, see <https://crowcpp.org/master/>.

## Java

Create a Spring Boot project using the following commands:

```

curl https://start.spring.io/starter.zip -d dependencies=web -d name=MySpringBootApplication -d
packageName=com.example.myspringbootapp -d groupId=com.example -d arti-
factId=MySpringBootApplication -d version=0.0.1-SNAPSHOT -o MySpringBootApplication.zip

unzip MySpringBootApplication.zip -d MySpringBootApplication

cd MySpringBootApplication

```

Spring Boot projects use port 8080 by default. You can change the port under **application.properties**.

Edit **MySpringBootApplicationApplication.java** to the following:

```
package com.example.myspringbootapp;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.*;

import java.util.Map;

@SpringBootApplication
@RestController
public class MySpringBootApplication {

    public static void main(String[] args) {
        SpringApplication.run(MySpringBootApplication.class, args);
    }

    @PostMapping("/data/path")
    public Map<String, Object> dataPath(@RequestBody Map<String, Object> data) {
        if (data == null || data.isEmpty()) {
            System.out.println("{\"error\": \"No data provided\"}");
            return Map.of("error", "No data provided");
        }

        // Print the received data to the terminal
        System.out.println("Received data: " + data);

        // Process the data (this is just an example)
        return Map.of("received_data", data);
    }
}
```

For more information, see <https://spring.io/guides/tutorials/rest>.

## JavaScript

This example uses Node.js.

In your Node.js project, install express:

```
npm install express
```

Edit index.js with the following:

```
const express = require('express');
const app = express();
const port = 8000;

// Middleware to parse JSON request bodies
app.use(express.json());

app.post('/data/path', (req, res) => {
```

```
const data = req.body;
if (!data || Object.keys(data).length === 0) {
  const errorResponse = { error: "No data provided" };
  console.log(errorResponse); // Print the response to the terminal
  return res.status(400).json(errorResponse);
}

// Print the received data to the terminal
console.log(`Received data: ${JSON.stringify(data)}`);

// Process the data (this is just an example)
const processedData = { received_data: data };

res.status(200).json(processedData);
});

app.listen(port, () => {
  console.log(`Server is running on http://localhost:${port}`);
});
```

For more information, see <https://expressjs.com/>.

### 5.1.3 Event subscriptions

#### Section overview

- Get an overview of what data you can subscribe webhook clients to.
- See what triggers the different event-types.

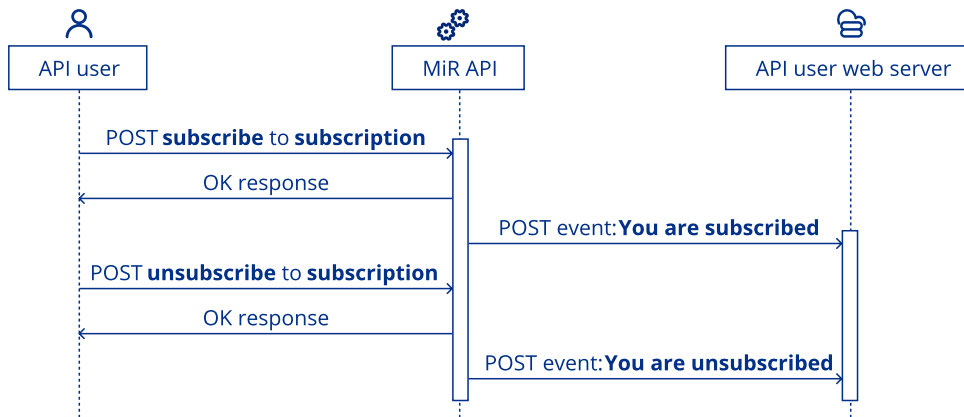
This page lists all of the events you can subscribe a custom [webhook](#) client to and what triggers each event-type. MiR Fleet Integration API sends POST requests to the subscribed webhook client when an event is triggered.

You can set up multiple subscriptions to the same event-type to make MiR Fleet push event-data to several endpoints.

For a full list of events, you can also use the `api/v1/subscription/event-types` endpoints.

When you subscribe or unsubscribe a subscription, you receive a response to confirm that the subscription was set up correctly, and any webhooks that subscribe to [Subscription](#) receives an event.





## Subscribe to an event

To subscribe to an event, follow these steps:

- 1 Determine which events contain the data you are interested in. To see event-types you can:
  - Send the request `GET https://<MiR_Fleet_IP_address>/api/v1/subscription/event-types`. This will return all the valid event-types.
  - Access the Swagger page—see "[Getting started](#)" on page 488— and execute the `GET api/v1/subscription/event-types` action. This will return all the valid event-types.
  - The sections following these instructions describe in detail what each event-type includes.
- 2 Create a webhook client that listens for POST requests on an endpoint—see "[Webhooks](#)" on page 490. You can customize the endpoints in anyway you want.

- 3 Send a the request POST `https://<MiR_Fleet_IP_address>/api/v1/subscription` with the following content:

```
{
  "base-url": "https://<Webhook_client_IP>",
  "endpoints": [
    {
      "event-type": <Event_type>,
      "endpoint-paths": [
        "<custom/endpoint/path>"
      ]
    }
  ],
  "ignore-certificate-errors":true
}
```

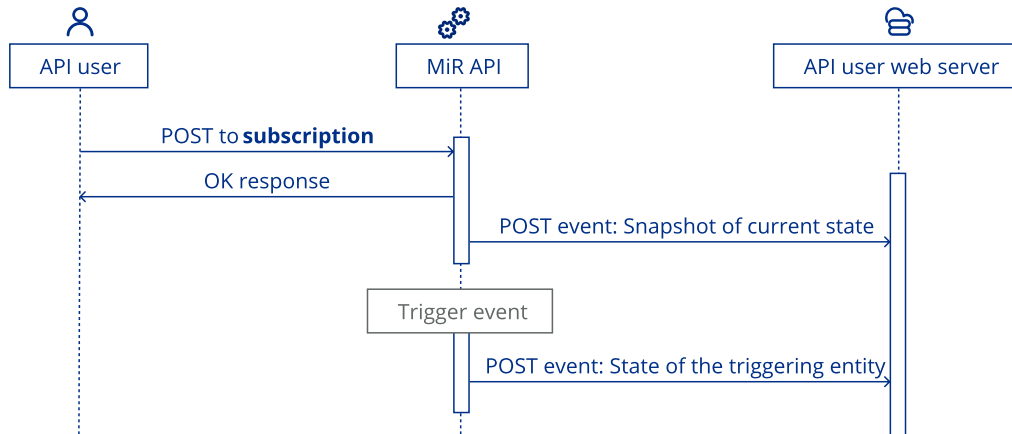
- `<Webhook_client_IP>`: The IP address of the device where the webhook client is running.
- `<Event_type>`: Enter the event-type you want the webhook client to subscribe to.
- `<custom/endpoint/path>`: Enter the endpoint you want the POST request with the event to be sent to.
- `<optional_token>`: Use this field if your webhook client requires a security token to validate the POST request—see ["Webhooks" on page 490](#).

For example, the following request would make MiR Fleet Integration API POST an Alert event to `https://123.4.56.7/api/alerts_events` whenever an alert occurs:

```
{
  "base-url": "https://123.4.56.7/api",
  "endpoints": [
    {
      "event-type": "Alert",
      "endpoint-paths": [
        "alerts_events"
      ]
    }
  ],
  "ignore-certificate-errors":true
}
```

- 4 Verify that your webhook client receives the expected event when it is triggered. Many events send a snapshot of the current state when you set up the subscription.

- 5 Modify your webhook client to extract the information from the event data that you are interested in. Events always follow the corresponding event-contract.



## Subscribe to an event

To subscribe to an event, follow these steps:

- 1 Determine which events contain the data you are interested in. To see event-types you can:
  - Send the request `GET https://<MiR_Fleet_IP_address>/api/v1/subscription/event-types`. This will return all the valid event-types.
  - Access the Swagger page—see ["Getting started" on page 488](#)— and execute the `GET api/v1/subscription/event-types` action. This will return all the valid event-types.
  - The sections following these instructions describe in detail what each event-type includes.
- 2 Create a webhook client that listens for POST requests on an endpoint—see ["Webhooks" on page 490](#). You can customize the endpoints in anyway you want.

- 3 Send a the request POST `https://<MiR_Fleet_IP_address>/api/v1/subscription` with the following content:

```
{
  "base-url": "https://<Webhook_client_IP>",
  "endpoints": [
    {
      "event-type": <Event_type>,
      "endpoint-paths": [
        "<custom/endpoint/path>"
      ]
    }
  ],
  "ignore-certificate-errors":true
}
```

- `<Webhook_client_IP>`: The IP address of the device where the webhook client is running.
- `<Event_type>`: Enter the event-type you want the webhook client to subscribe to.
- `<custom/endpoint/path>`: Enter the endpoint you want the POST request with the event to be sent to.
- `<optional_token>`: Use this field if your webhook client requires a security token to validate the POST request—see ["Webhooks" on page 490](#).

For example, the following request would make MiR Fleet Integration API POST an Alert event to `https://123.4.56.7/api/alerts_events` whenever an alert occurs:

```
{
  "base-url": "https://123.4.56.7/api",
  "endpoints": [
    {
      "event-type": "Alert",
      "endpoint-paths": [
        "alerts_events"
      ]
    }
  ],
  "ignore-certificate-errors":true
}
```

- 4 Verify that your webhook client receives the expected event when it is triggered. Many events send a snapshot of the current state when you set up the subscription.

- 5 Modify your webhook client to extract the information from the event data that you are interested in. Events always follow the corresponding event-contract.

## Alert

Subscribe to `Alert` to receive an event when an alert is triggered in the system.

Parameter	Type	Description
<code>id</code>	string	Identification string of the alert.
<code>name</code>	string	<ul style="list-style-type: none"><li>"<code>DeadlockDetected</code>": Occurs when MiR Fleet detects a deadlock in the system.</li><li>"<code>RobotError</code>": Occurs when a connected robot reports an error.</li><li>"<code>RobotEStop</code>": Occurs when a connected robot goes in Protective or Emergency stop.</li></ul>
<code>message</code>	string	More information about the alert. This is the same as the message shown in the interface.
<code>timestamp</code>	string	The time the alert was recorded.

### Alert event-contract

```
{
  "Alert Event": {
    "id": "d1943b99-45f1-4571-9ced-e49305347b25",
    "name": "CustomAlert",
    "message": "Alert message",
    "timestamp": "2022-09-19T09:08:12.8651699Z"
  }
}
```

## Error

Subscribe to `Error` to receive an event when an error occurs in MiR Fleet Integration API. This is only related to system errors that can occur when you send API requests. It does not catch any robot errors or errors in the request you sent in the API.

Parameter	Type	Description
action-type	string	The error occurred while trying to: <ul style="list-style-type: none"><li>"Create": create an entity.</li><li>"Update": update or modify an entity.</li><li>"Delete": delete an entity.</li><li>"LockStateChange": change state of a Lock zone.</li><li>"UpdatePhase": update a phase in a serial-order.</li><li>"ExecutePhase": execute a phase in a serial-order.</li><li>"TopModuleEvent": send a top module event using the Top Module Integration API.</li><li>"RobotIdentityEvent": send a robot identity event.</li><li>"RobotRuntimeEvent": send a robot runtime event.</li><li>"Abort": abort an ongoing process.</li><li>"ExecuteFallback": execute a fallback order.</li><li>"AbortFallback": abort a fallback order.</li></ul>
entity-type	string	<ul style="list-style-type: none"><li>"Map"</li><li>"Position"</li><li>"Zone"</li><li>"SerialOrder"</li></ul>

Parameter	Type	Description
		<ul style="list-style-type: none"><li>"Robot"</li></ul>
entity-id	string	The ID of the entity related to the error.
error-type	string	<ul style="list-style-type: none"><li>"NotFound": The entity in a request does not exist in the system</li><li>"Conflict": The requested change conflicts with existing data in the system.</li><li>"Forbidden": The requested change affects an entity that is being used in another process.</li><li>"Internal": The operation failed</li><li>"InvalidInput": A request has invalid input that cannot be used for this field.</li></ul>
message	string	More information about why the error occurred.
timestamp	string	The time the error was recorded.

Error events can be triggered by using any of the following endpoints:

- [/site/map](#)
- [/site/position](#)
- [/site/zone](#)
- [/serial-order](#)

#### Error event-contract

```
{
  "Error Event": {
    "action-type": "Create",
    "entity-type": "Map",
```

```

    "entity-id": "85d80812-0a58-4261-916a-f438fa519051",
    "error-type": "Internal",
    "message": "Encountered an internal error.",
    "timestamp": "2022-09-19T09:08:12.8651699Z"
  }
}

```

### Map error events

Action	Error trigger	Error type	Message
CreateMap	A map with the same ID already exists	Conflict	Map with id {mapId} already exists
CreateMap	A map with the same name already exists	Conflict	Map with name {mapName} already exists
CreateMap	The map was not successfully created or an unknown error has occurred	Conflict, Internal, NotFound	
DeleteMap	Could not find a map with the specified ID or the map was not successfully deleted	Internal, NotFound	
DeleteMap	Could not delete map due to occupied positions	Forbidden	Unable to delete map {mapExternalId} because of block positions: {positionExternalId}



## Position error events

Action	Error trigger	Error type	Message
CreatePosition	A position with the same ID already exists	Conflict	Position with id {positionExternalId} already exists
CreatePosition	A position with the same name already exists	Conflict	Position with name {positionName} already exists
CreatePosition	The position was not successfully created	Internal	
CreatePosition	Have not defined the bar property for a Bar-marker	InvalidInput	Bar Marker requires Bar property
CreatePosition	Could not find a map with the specified ID	NotFound/Internal	Error getting id information for map {mapExternalId}
UpdatePosition	The position was not successfully updated	Internal	
UpdatePosition	The received	InvalidInput	The information

Action	Error trigger	Error type	Message
	input is invalid for the selected position		provided for position {positionExternalId} includes properties not expected for update of position of type {positionType}
UpdatePosition	Could not find a position with the specified ID	NotFound, Internal	
DeletePosition	The position was not successfully deleted or a position with that ID was not found	NotFound, Internal	

#### Zone error events

Action	Error trigger	Error type	Message
CreateZone	A zone with the same ID already exists	Conflict	Zone with id {zoneExternalId} already exists
CreateZone	Could not find a map with the specified ID	NotFound, Internal	Error getting id information for map {mapExternalId}

Action	Error trigger	Error type	Message
CreateZone	A zone with the same name already exists	Conflict	Zone object with name {zoneName} already exists.
CreateZone	The zone was not successfully created	Conflict, NotFound, Internal	
DeleteZone	Could not find a zone with the specified ID or the zone was not successfully deleted	NotFound, Internal	
ActivateLockZone	Could not find a zone with the specified ID or zone was not successfully locked or unlocked on MiR Fleet	NotFound, Internal, Conflict	

#### Serial-order error events

Action	Error trigger	Error type	Message
Create	The serial-order, phase or fallback part of the serial-order was not successfully created	Conflict, Internal	

Action	Error trigger	Error type	Message
Execute phase	<ul style="list-style-type: none"> <li>• Could not set the WaitingForInput status</li> <li>• Could not retrieve serial-order data</li> <li>• The request does not have appropriate user authority</li> <li>• Did not respond with a success status</li> <li>• Could not assign the phase to the reserved robot</li> </ul>	Internal	Abort required
Execute phase	Could not find a mission with the specified ID for the phase mission	InvalidData	Mission Id {id} provided is unknown.
Execute phase	Could not find a mission with the specified ID for this fallback mission	InvalidData	Fallback Mission Id {id} provided is unknown.
Update order	Could not find a serial-order or phase with the specified ID	NotFound	
Update order	The priority is not an accepted value	InvalidInput	

Action	Error trigger	Error type	Message
Update order	The serial-order was not successfully updated	Internal	
Update phase	The phase already has a state.	Forbidden	Attempted to update Phase {i} of Serial Order {id}. Updating a Phase that has been or is in progress of being executed is not allowed.

## Payload

Subscribe to `Payload` to receive events when a Send JSON payload action is executed by a robot or the robot in a mission or Integration zone—see ["Integration actions" on page 487](#) and ["Zones" on page 488](#). The robot can pass a customized JSON body in the event.

Parameter	Type	Description
name	string	The name of the event you defined in the mission action.
payload	string	The custom JSON payload you defined in the action.
robot-id	string	The ID of the robot that executed the action.
serial-order-id	string	The ID of the serial-order where the action was executed.
phase-index	string	The phase in the serial-order the action was executed

Parameter	Type	Description
		in.
is-fallback	bool	Is true if the executed phase is a fallback phase.
timestamp	string	The time the Send JSON payload action was executed.

### Payload event-contract

```

{
  "Payload Event": {
    "name": "CustomEvent",
    "payload": {
      //The JSON payload you defined in the action is pasted here. The following
      is an example."Speed": "Low",
      "Height": 5,
      "IsPaused": true,
      "Orientation": 0.5
    },
    "robot-id": "d7b06bff-e56b-41ab-bea1-a30b32639661",
    "serial-order-id": "d1943b99-45f1-4571-9ced-e49305347b25",
    "phase-index": 1,
    "is-fallback": true,
    "timestamp": "2022-09-19T09:08:12.8651699Z"
  }
}

```

### SerialOrderStatus

Subscribe to `SerialOrderStatus` to receive events whenever a phase or serial-order changes status.

Parameter	Type	Description
is-snapshot	bool	Is true if the event is a snapshot of all current serial-orders phases in MiR Fleet.
serial-order-	<a href="#">serial-</a>	Either a list of objects for all current serial-orders (if

Parameter	Type	Description
status-events	<a href="#">order-status-event</a> object	"is-snapshot": True), or a single object with the order that changed status and triggered the event.

### SerialOrderStatus event-contract

```
{
  "Serial Order Status Event": {
    "is-snapshot": false,
    "serial-order-status-events": [
      {
        "serial-order-id": "2ed45aec-3e6c-4969-bd0d-d6746ab4fbc0",
        "phase-index": 1,
        "state": "Executing",
        "state-change-timestamp": "2022-09-19T09:08:12.8651699Z",
        "robot-id": "80f94c98-44db-404e-8f65-8a74b89938c7",
        "message": "Test string!",
        "is-fallback": false,
        "order-type": "UI"
      }
    ]
  }
}
```

### RobotIdentity

Subscribe to `RobotIdentity` to receive events when a robot's software version or name changes.

Parameter	Type	Description
is-snapshot	bool	Is true if the event is a snapshot of all current robots in MiR Fleet.
robot-identity-events	<a href="#">robot-identity-event</a> object	Either a list of objects for all robots (if "is-snapshot": True), or a single object with the robot that changed software version or name and triggered the event.

### RobotIdentity event-contract

```

{
  "Robot Identity Event": {
    "is-snapshot": false,
    "robot-identity-events": [
      {
        "robot-id": "822db143-c9ab-4023-973b-a431e38d6a86",
        "serial-number": "0001",
        "name": "Robot",
        "model": "MiR100",
        "software-version": "4.0.0",
        "timestamp": "2022-09-19T09:08:12.8651699Z"
      }
    ]
  }
}

```

### RobotRuntime

Subscribe to `RobotRuntime` to receive events with robot coordinates and when the robot enters Emergency or Protective stop. This subscription is a data stream. An event is posted to subscribed endpoints every second.

Parameter	Type	Description
robot-id	string	The ID of the robot.
pose-x	double	The X coordinate on the map the robot is positioned.
pose-y	double	The Y coordinate on the map the robot is positioned.
pose-orientation	double	The orientation of the robot relative to the map.
map-id	string	The ID of the map the robot is operating on.
in-emergency-stop	bool	Is <code>True</code> when the robot is in Protective or Emergency stop.



Parameter	Type	Description
timestamp	string	The time the robot changed position or stopped status.

### RobotRuntime event-contract

```
{
  "Robot Runtime Event": {
    "robot-id": "0ada8a13-f035-4564-8b44-b148bec9738c",
    "pose-x": 1.5,
    "pose-y": 1.5,
    "pose-orientation": 1.5,
    "map-id": "80f94c98-44db-404e-8f65-8a74b89938c7",
    "in-emergency-stop": false,
    "timestamp": "2022-09-19T09:08:12.8651699Z"
  }
}
```

### RobotState

Subscribe to `RobotState` to receive events when a robot changes state. An event is only triggered when the robot changes states, not battery percentage.

Parameter	Type	Description
robot-id	string	The ID of the robot.
robot-state	state objects	The path of state categories—see <a href="#">"Robot states and fleet behavior" on page 91</a> .
robot-end-state	string	The robot's current state—see <a href="#">"Robot states and fleet behavior" on page 91</a> .
battery-percentage	integer	The robot's current battery percentage.

Parameter	Type	Description
uptime	string	The total time the robot has been operating.
battery-time-remaining-in-minutes	integer	The number of minutes the robot can operate until it needs to be charged.
total-distance-moved-in-meters	integer	The total distance the robot has driven.
pose-x	double	The X-coordinate of the robot's current position on the map.
pose-y	double	The X-coordinate of the robot's current position on the map.
pose-orientation	double	The orientation of the robot.
map-id	string	The ID of the map the robot is currently on.
timestamp	string	The time the robot state changed

## Site

Subscribe to `Site` to receive events when a map or map element is changed.

Parameter	Type	Description
is-snapshot	bool	Is true if the event is a snapshot of all current site elements in MiR Fleet.
site-events	<a href="#">site-event</a>	Either a list of objects for all site elements (if " <code>is-</code>

Parameter	Type	Description
	object	snapshot": True), or a single object with the site element that changed and triggered the event.

### Site event-contract

```

{
  "Site Event": {
    "is-snapshot": true,
    "site-events": [
      {
        "entity-id": "822db143-c9ab-4023-973b-a431e38d6a86",
        "entity-type": "Map",
        "action-type": null,
        "status-change-timestamp": "2022-09-19T09:08:12.8651699Z",
        "map": {
          "name": "Map",
          "image": "iVBORw0KG...==",
          "origin": {
            "x": 1,
            "y": 5
          }
        },
        "position": null,
        "zone": null,
        "mission": null
      },
      {
        "entity-id": "822db143-c9ab-4023-973b-a431e38d6a86",
        "entity-type": "Position",
        "action-type": null,
        "status-change-timestamp": "2022-09-19T09:08:12.8651699Z",
        "map": null,
        "position": {
          "base-position": {
            "type": "Robot",
            "map-id": "80f94c98-44db-404e-8f65-8a74b89938c7",
            "name": "Position3",
            "map-pose": {
              "x": 1.3,
              "y": 2.7,
              "orientation": 0.8
            }
          }
        },
        "marker": {
          "type": "VL",
          "map-id": "7b09f46e-4c90-46bf-8230-f0128ed0b0fb",
          "name": "DockingStation1",
          "map-pose": {
            "x": 1.3,

```

```

        "y": 2.7,
        "orientation": 0.8
    },
    "entry-position-offset-pose": {
        "x": 0.6,
        "y": 0.2,
        "orientation": 0.3
    },
    "offset-pose": {
        "x": 0.1,
        "y": 0.1,
        "orientation": 0.2
    }
},
"charger": {
    "type": "48V",
    "map-id": "24f6c058-a582-46ab-8b73-e3fbae58c6fc",
    "name": "MainEntryCharger",
    "map-pose": {
        "x": 1.3,
        "y": 2.7,
        "orientation": 0.8
    },
    "entry-position-offset-pose": {
        "x": 0.6,
        "y": 0.2,
        "orientation": 0.3
    }
},
"utility-position": {
    "type": "SHELF",
    "map-id": "85d80812-0a58-4261-916a-f438fa519051",
    "name": "Pickup10",
    "map-pose": {
        "x": 1.3,
        "y": 2.7,
        "orientation": 0.8
    },
    "entry-positions": [
        {
            "entry-position-type": "SHELF LONG",
            "offset-pose": {
                "x": 0.6,
                "y": 0.9,
                "orientation": 0.5
            }
        },
        {
            "entry-position-type": "SHELF SHORT",
            "offset-pose": {
                "x": 0.1,
                "y": 0.1,
                "orientation": 0.2
            }
        }
    ]
}

```

```

    ]
  },
  "zone": null,
  "mission": null
},
{
  "entity-id": "822db143-c9ab-4023-973b-a431e38d6a86",
  "entity-type": "Mission",
  "action-type": null,
  "status-change-timestamp": "2024-07-16T08:18:15.8651699Z",
  "map": null,
  "position": null,
  "zone": null,
  "mission": {
    "name": "Drive to Charger"
  }
}
]
}

```

## Subscription

Subscribe to `Subscription` to receive an event when a subscription in the system changes status.

Parameter	Type	Description
base-url	string	The base URL you subscribe to the selected events.
endpoints	<a href="#">endpoint</a> objects	List of event-types the URL is subscribed to and the endpoint path the event is posted to.
state	string	<ul style="list-style-type: none"> <li>"Subscribed"</li> <li>"Unsubscribed"</li> </ul>
timestamp	string	The time the subscription was changed.

### Subscription event-contract

```
{
  "base-url": "http://base-url.com/",
  "endpoints": [
    {
      "event-type": "Subscription",
      "endpoint-paths": [
        "p1",
        "p2"
      ]
    },
    {
      "event-type": "Site",
      "endpoint-paths": [
        "p1"
      ]
    },
    {
      "event-type": "RobotIdentity",
      "endpoint-paths": [
        "p15"
      ]
    }
  ],
  "state": "Subscribed",
  "timestamp": "2022-09-19T09:08:12.8651699Z"
}
```

### System

Subscribe to `System` to receive an event when a robot changes connection status or an evacuation changes status.

Parameter	Type	Description
is-snapshot	bool	Is true if the event is a snapshot of all current system statuses in MiR Fleet.
system-events	<a href="#">system-event</a> object	Either a list of objects for all robots and evacuation statuses (if "is-snapshot": True), or a single object with the system element that changed and triggered the event.

### System event-contract

```

{
  "System Event": {
    "is-snapshot": true,
    "system-events": [
      {
        "evacuation": {
          "evacuation-state": "InEvacuationOngoing"
        },
        "robot": null,
        "timestamp": "2022-09-19T09:08:12.8651699Z"
      },
      {
        "evacuation": null,
        "robot": {
          "robot-id": "80e94c98-44db-404e-8f65-8a74b89938c7",
          "robot-state": "Added"
        },
        "timestamp": "2022-09-19T09:08:12.8651699Z"
      }
    ]
  }
}

```

### TopModule

Subscribe to `TopModule` to receive custom events defined using the Top Module Integration API.

Parameter	Type	Description
event-type	string	<ul style="list-style-type: none"> <li>"Event"</li> <li>"Error"</li> </ul>
event	<a href="#">event</a> object	Custom event from the Top Module Integration API
error	<a href="#">error</a> object	Custom error from the Top Module Integration API

### TopModule event-contract

```

{
  "Top Module Event": {

```

```

    "event-type": "Event",
    "event": {
      "name": "CustomEvent",
      "payload": {
        "Speed": "Low",
        "Height": 5,
        "IsPaused": true,
        "Orientation": 0.5
      },
      "robot-id": "85d80812-0a58-4261-916a-f438fa519051",
      "serial-order-id": "208b6817-34d5-4077-9156-919040abe5fc",
      "phase-index": 1,
      "is-fallback": false,
      "timestamp": "2022-09-19T09:08:12.8651699Z"
    },
    "error": {
      "name": "CustomError",
      "message": "Error from Top Module",
      "robot-id": "85d80812-0a58-4261-916a-f438fa519051",
      "serial-order-id": "208b6817-34d5-4077-9156-919040abe5fc",
      "phase-index": 0,
      "is-fallback": true,
      "timestamp": "2022-09-19T09:08:12.8651699Z"
    }
  }
}

```

## Group

Subscribe to `Group` to receive an event when a group is created, changed, or deleted.

Parameter	Type	Description
<code>is-snapshot</code>	<code>bool</code>	Is true if the event is a snapshot of all current groups in MiR Fleet.
<code>group-events</code>	<code>group-event</code> object	List of <code>group-events</code> .

## Group event-contract

```

"is-snapshot": true,
"group-events": [
  {

```



```

"entity-id": "102fd74a-bbb0-4b25-b42c-8b8e35225c0d",
"entity-type": "MissionGroup",
"action-type": "Create",
"status-change-timestamp": "2024-07-04T17:10:14.48348Z",
"group": null,
"mission-group": {
  "name": "MissionGroup1",
  "mission-ids": [
    "224ee3e8-3101-45dc-b346-3b3282101598",
    "65fc3f77-5401-4168-9841-36ea836b9177"
  ]
},
},
{
"entity-id": "a15fc10e-b106-4411-9587-3d7f0f48de67",
"entity-type": "Group",
"action-type": "Create",
"status-change-timestamp": "2024-07-04T17:10:14.48348Z",
"group": {
  "name": "Group1",
  "robot-ids": [
    "4d95cdb0-e314-4338-b594-4c7803fecc8b"
  ],
  "position-ids": [
    "6e7c2e57-dfb1-4ba5-a2e7-75692c4d830f",
    "ca245603-59ea-4cc3-863e-76185763f73c"
  ],
  "mission-group-ids": [
    "102fd74a-bbb0-4b25-b42c-8b8e35225c0d"
  ]
},
"mission-group": null
}
]

```

### 5.1.4 Serial-orders

#### Section overview

- What serial-orders consist of
- How to schedule serial-order through MiR Fleet Integration API
- How to update the phases in serial-orders

A serial-order is a defined sequence of missions that you want a single robot to execute consecutively.

Serial-orders consist of any number of phases. Each phase specifies a mission the robot must execute. When a robot is assigned a serial-order, it executes all of the phases in the specified order. If a mission in a phase uses arguments, you can pass values to the arguments.

The following points describe serial-order behavior:

- A serial-order can not be split between multiple robots. If you want two tasks completed by different robots, schedule a separate serial-order for each.
- You can change, delete, and update phases that are not being executed.
- You can update the arguments in phases that are not being executed.
- You cannot use any default site elements as arguments in a serial-order, for example, the three default sounds: Foghorn, Horn, and Beep. If you want to use a default element, you must make your own copy of it.
- You can leave any number of phases empty or incomplete. The integration API will prompt you with a serial-order event when it reaches an incomplete phase and waits until you update the phase.
- Serial-orders are only assigned and reserved to a robot when the first phase contains a mission.
- The phase count always starts from 1 and increases by 1 for each phase.
- For each phase, you can specify a fallback mission that is run instead if the main mission is aborted.
- When you schedule a mission through the interface, it is converted to a serial-order in the API and triggers a serial-order-status-event.

### Receive serial-order statuses

If you are going to use serial-orders, we recommend setting up a subscription to `SerialOrderStatus` with an application that displays all posted serial-order-status-events—see "[Event subscriptions](#)" on page 496.

To set up an application that receives serial-order statuses, follow these steps:

- 1 Create a [webhook client](#) that can parse POST requests where the content follows the [serial-order-status-events](#) schema.

- 2

Subscribe to [OrderStatus](#) by sending a POST request to the subscription endpoint.

- 3 Design your webhook client to display all events received from MiR Fleet Integration API.

### Schedule serial-order with arguments

You schedule serial-orders by sending a POST request with a serial-order object to the [serial-order](#) endpoint.

The serial-order object must contain at least one phase object. The phase must refer to a mission using the mission-id. You can see the mission-id by:

- Opening the mission editor for the relevant mission in the MiR Fleet interface. The mission-id is the UUID at the end of the URL when you open the mission editor.
- Sending a GET request to the [site/snap-shot](#) endpoint.

If the selected mission has any arguments, you must assign values to the arguments. If you do not assign values or the values are invalid, all clients subscribed to OrderStatus receive an event with the status: "WaitingForInput". The serial-order is paused until it receives an update to make the arguments valid.

You cannot use any default site elements as arguments in a serial-order, for example, the three default sounds: Foghorn, Horn, and Beep. If you want to use a default element, you must make your own copy of it.

If an argument uses site elements like footprints, sounds, and carts, you can find the ID number to refer to these elements directly in your site SQL database—see ["SQL Database" on page 237](#).

The following is an example of a complete serial-order request you can POST to [serial-order](#) with one mission, one argument, and a fallback mission in case the serial-order is aborted.

```
{
  "serial-order": {
    "id": "746bae3c-296e-48bf-b8c4-e7cda7404626",
    "robot-id": "2131af7f-d24e-4b26-9c33-c6fb276cefbc",
    "phases": [
      {
        "mission-id": "efd74342-9f8a-46d2-b711-b38e1225abae",
        "arguments": [
          {
            "name": "var_1",
            "value": "1"
          }
        ]
      }
    ]
  }
}
```

```

    ],
    "fallback-mission-id": 2
  }
]
}

```

You can schedule an empty serial-order to reserve a robot for missions that will be added in a later update.

```

"serial-order": {
  "id": "1fd9ed2b-5822-480d-b52e-da7911071936",
  "phases": [
    {
      },
    {
      },
    {
      }
  ]
}

```

### Update serial-order phase

If the webhook client subscribed to OrderStatus receives an event with the status "WaitingForInput", the serial-order has been paused and you must send an update to the serial-order phase.

The update must specify the ID of the serial-order to be updated and the phase you want to update. You can get the ID and phase-index from the status events you received regarding the serial-order.

Send a PUT request to [serial-order/phase](#) to update the serial-order. You can only update one phase at a time.

```

{
  "id": "1fd9ed2b-5822-480d-b52e-da7911071936",
  "index": 1,
  "phase": {
    "mission-id": "efd74342-9f8a-46d2-b711-b38e1225abae",
    "arguments": [
      {
        "name": "var_timeout_1",
        "value": "1"
      }
    ]
  }
}

```

```
} } ]
```

## 5.1.5 Endpoints

### Section overview

- Endpoints you can use in the MiR Fleet Integration API
- The purpose of each field in the endpoints
- The format of many common requests

Use endpoints to send orders or map updates to MiR Fleet and request information on demand.

There is a rate limit to the number of requests you can send based on the Sliding Window strategy:

- Maximum number of requests per window: 200
- Window duration: 1 minute

If the number of requests surpasses the maximum amount allowed, a 503 Service Unavailable HTTP Response will be returned. This provides security against Denial of Service (DoS) attacks.

### Site

Use site endpoints to retrieve or update maps, positions, and missions.

#### GET `api/v1/site/snapshot`

Retrieve all site data.

#### Responses:

- 200 OK: Returns a snapshot of all site data and missions.

```
{
  "site-events": [
    {
      "entity-id": "string",
      "entity-type": "string",
      "action-type": "string",
      "status-change-timestamp": "2024-07-31T14:28:57.879Z",
      "map": {
        "name": "string",
        "image": "string",
        "origin": {
          "x": 0,
          "y": 0
        }
      }
    },
    "position": {
      "base-position": {
        "type": "string",
        "map-id": "string",
        "name": "string",
        "map-pose": {
          "x": 0,
          "y": 0,
          "orientation": 0
        }
      }
    },
    "marker": {
      "type": "string",
      "map-id": "string",
      "name": "string",
      "map-pose": {
        "x": 0,
        "y": 0,
        "orientation": 0
      }
    },
    "entry-position-offset-pose": {
      "x": 0,
      "y": 0,
      "orientation": 0
    },
    "offset-pose": "string",
    "bar": {
      "length": 0,
      "distance": 0
    }
  },
  "charger": {
    "type": "string",
    "map-id": "string",
    "name": "string",
    "map-pose": {
      "x": 0,
```

```
        "y": 0,
        "orientation": 0
    },
    "entry-position-offset-pose": {
        "x": 0,
        "y": 0,
        "orientation": 0
    }
},
"utility-position": {
    "type": "string",
    "map-id": "string",
    "name": "string",
    "map-pose": {
        "x": 0,
        "y": 0,
        "orientation": 0
    },
    "entry-positions": [
        {
            "entry-position-type": "string",
            "offset-pose": {
                "x": 0,
                "y": 0,
                "orientation": 0
            }
        }
    ],
    "offset-pose": {
        "x": 0,
        "y": 0,
        "orientation": 0
    }
},
"zone": {
    "name": "string",
    "map-id": "string",
    "polygon": [
        {
            "x": 0,
            "y": 0
        }
    ],
    "type": "string",
    "events": [
        "string"
    ],
    "lock-zone-state": "string"
},
"mission": {
    "name": "string"
}
}
```

```
} ]
```

- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### GET [api/v1/site/export](#)

Download the current site as a .site file.

#### Responses:

- 200 OK: Returns the site file.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### POST [api/v1/site/import](#)

Import a site file to MiR Fleet.

You must specify these headers in request body:

- Content-Type: multipart/form-data
- Content-Disposition: form-data; name="site-file"

#### Request body:

Parameter	Type	Required	Constraints
site-file	form-data	Yes	Must be a MiR .site file.

#### Responses:

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 409 Conflict: A site is already being imported



**POST `api/v1/site/map`**

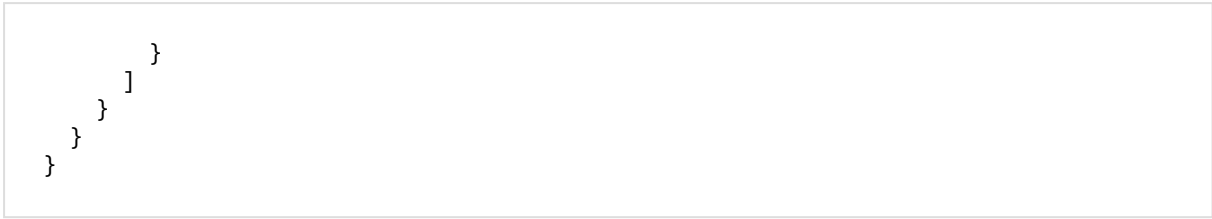
Create a new map using geoJSON data.

This endpoint can trigger an error event—see ["Map error events" on page 504](#).

**Request body:**

Parameter	Type	Required	Constraints
map	map object	Yes	Defined <a href="#">map</a> schema

```
{
  "map": {
    "name": "Map_1",
    "geo-json": {
      "type": "FeatureCollection",
      "features": [
        {
          "type": "Feature",
          "properties": {},
          "geometry": {
            "type": "Polygon",
            "coordinates":
              [
                [
                  [
                    0,
                    0
                  ],
                  [
                    0,
                    20
                  ],
                  [
                    20,
                    20
                  ],
                  [
                    20,
                    0
                  ],
                  [
                    0,
                    0
                  ]
                ]
              ]
          }
        }
      ]
    }
  }
}
```

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 409 Conflict: A map with the specified values already exists in the database.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**DELETE** `api/v1/site/map/map_id`

Delete a map based on the map `id`.

This endpoint can trigger an error event—see ["Map error events" on page 504](#).

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**POST** `api/v1/site/position`

Create a position or marker.

This endpoint can trigger an error event—see ["Position error events" on page 505](#).

**Request body:**

Parameter	Type	Required	Con-straints	Example
type	string	Yes	Defined <a href="#">position</a> types	"Util-ityPosition"
id	string	Yes	.NET format Specifier D	"7c9e6679-7425-40de-944b-e07fc1f90ae7"
map-id	string	Yes	.NET format Specifier D	"056c3989-d316-4fca-9e0f-a68cb-c9d74b3"
name	string	Yes	Unique position name	"Position 1"
map-pose	<a href="#">pose</a> object	Yes	Defined <a href="#">pose</a> schema	{"x": 0, "y": 0, "orientation": 0}
base-position-type	string	If "type": "BasePosition"	Defined base position types	"Staging"
offset-pose	<a href="#">pose</a> object	If "type": "Marker"	Defined <a href="#">pose</a> schema	{"x": 0, "y": 0, "orientation": 0}

Parameter	Type	Required	Con-straints	Example
marker-type	string	If "type": "Marker"	Defined <a href="#">marker types</a>	"bar"
bar	<a href="#">bar</a> object	If "marker-type": "Bar"	Defined <a href="#">bar</a> schema	{ "distance": 0, "length": 0}
entry-position-offset-pose	<a href="#">pose</a> object	If "type": "Marker" or "type": "Charger"	Defined <a href="#">pose</a> schema	{ "distance": 0, "length": 0}
entry-positions	<a href="#">entry-position</a> object	If "type": "UtilityPosition"	Defined <a href="#">entry-position</a> schema	[ { "entry-position-type": "CartLeft", "offset-pose": { "x": 10, "y": 16, "orientation": 0 } }, { "entry-position-type": "CartRight", "offset-pose": { "x": 20, "y": 26, "orientation": 0 } } ]
entry-position-type	string	If "type": "UtilityPosition"	Defined <a href="#">entry-position</a> type	"ShelfLong"
charger-	string	If "type":	Defined	"MiRCharge24-

Parameter	Type	Required	Con-straints	Example
type		"Charger"	charger type	V"
utility- position- type	string	If "type": "Util- ityPosition"	Defined utility- position type	"Cart"

```
{
  "type": "BasePosition",
  "id": "7c9e6679-7425-40de-944b-e07fc1f90ae7",
  "map-id": "056c3989-d316-4fca-9e0f-a68cbc9d74b3",
  "name": "Position",
  "map-pose": {
    "x": 0,
    "y": 0,
    "orientation": 0
  },
  "base-position-type": "Staging"
}
```

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 409 Conflict: Record with the specified values already exists in the database.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**PATCH [api/v1/site/position](#)**

Update or modify a position or marker.

This endpoint can trigger an error event—see ["Position error events" on page 505](#).

**Request body:**

Parameter	Type	Required	Constraints	Example
id	string	Yes	.NET format Specifier D	"7c9e6679-7425-40de-944b-e07fc1f90ae7"
map-id	string	No	.NET format Specifier D	"056c3989-d316-4fca-9e0f-a68cbc9d74b3"
name	string	No	Unique position name	"Position 1"
map-pose	<a href="#">pose</a> object	No	Defined <a href="#">pose</a> schema	{ "x": 0, "y": 0, "orientation": 0 }
offset-pose	<a href="#">pose</a> object	No	Defined <a href="#">pose</a> schema	{ "x": 0, "y": 0, "orientation": 0 }
bar	<a href="#">bar</a> object	No	Defined <a href="#">bar</a> schema	{ "distance": 0, "length": 0 }
entry-position-offset-pose	<a href="#">pose</a> object	No	Defined <a href="#">pose</a> schema	{ "distance": 0, "length": 0 }
entry-positions	<a href="#">entry-position</a> object	No	Defined <a href="#">entry-position</a> schema	[ { "entry-position-type": "CartLeft", "offset-pose": { "x": 10, "y": 16, "orientation": 0 } }, { "entry-

Parameter	Type	Required	Constraints	Example
				<pre>position-type": "CartRight", "offset-pose": { "x": 20, "y": 26, "orientation": 0 } } ]</pre>

```
{
  "id": "B62DA01F-9ABD-4d9d-80C7-02AF85C8228C",
  "name": "NewName",
  "map-pose": {
    "x": 1,
    "y": 2,
    "orientation": 1.5
  },
  "offset-pose": {
    "x": 1,
    "y": 2,
    "orientation": 1.5
  },
  "bar": {
    "length": 10,
    "distance": 10
  },
  "entry-position-offset-pose": {
    "x": 1,
    "y": 2,
    "orientation": 1.5
  },
  "entry-positions": [
    {
      "entry-position-type": "Type",
      "offset-pose": {
        "x": 1,
        "y": 2,
        "orientation": 1.5
      }
    },
    {
      "entry-position-type": "Type",
      "offset-pose": {
        "x": 1,
        "y": 2,
        "orientation": 1.5
      }
    }
  ]
}
```

**Responses:**

- 202 Accepted: Response body:

```
{
  "id": "B62DA01F-9ABD-4d9d-80C7-02AF85C8228C"
}
```

- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**DELETE `api/v1/site/position/position-id`**

Delete a position based on its position `id`.

This endpoint can trigger an error event—see ["Position error events" on page 505](#).

**Responses:**

- 202 Accepted: Response body:

```
{
  "id": "B62DA01F-9ABD-4d9d-80C7-02AF85C8228C"
}
```

- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**POST `api/v1/site/zone`**

Create a zone.

This endpoint can trigger an error event—see ["Zone error events" on page 506](#).

**Request body:**



Parameter	Type	Required	Constraints
zone	zone object	Yes	Defined <a href="#">zone</a> schema

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**DELETE [api/v1/site/zone/id](#)**

Delete a zone.

This endpoint can trigger an error event—see ["Zone error events" on page 506](#).

**Responses:**

- 202 Accepted: Response body:

```
{
  "id": "B62DA01F-9ABD-4d9d-80C7-02AF85C8228C"
}
```

- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**POST [api/v1/site/zone/lock](#)**

Lock or unlock a Lock zone—see ["Zones" on page 62](#).

This endpoint can trigger an error event—see ["Zone error events" on page 506](#).

**Request body:**

Parameter	Type	Required	Constraints	Example
zone-id	string	Yes	.NET format Specifier D	"7c9e6679-7425-40de-944b-e07fc1f90ae7"
is-locked	bool	Yes		"7c9e6679-7425-40de-944b-e07fc1f90ae7"

```
{
  "zone-id": "7c9e6679-7425-40de-944b-e07fc1f90ae7",
  "is-locked": true
}
```

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

## Subscription

Use subscription endpoints to set up [subscriptions to events](#). Set up a [webhook client](#) to monitor the events you subscribe and unsubscribe to.

### POST [api/v1/subscription](#)

Create a subscription to an event. Specify the address of the endpoint you want MiR Fleet to publish the event to when it is triggered.

Each subscription must have a unique combination of base URL, event-type, and endpoint path.

**Request body:**

Parameter	Type	Required	Description
base-url	string	Yes	The web-server endpoint address the API sends a POST request each time an event of the chosen event-type is triggered. Defined in Uri format.
endpoints	list of <a href="#">endpoint</a> objects	Yes	The endpoints you want to subscribe to certain event-types.
access-token	string	No	The token used to authenticate back to your web-server—see <a href="#">"Access tokens" on page 492</a> .
limited-tls	bool	No	If <code>"True"</code> , the API ignores TLS errors and continues with an insecure connection.
max-retries	int	No	The maximum number of times the API should try to send the POST request without receiving a confirmation response. By default, the value is three, except for RobotRunTime events, which is two by default.
ignore-certificate-errors	bool	No	If <code>"True"</code> , the API disables validation of certificates for the related HTTPS base-url. By default, this is set to <code>"False"</code> .

```
{
  "base-url": "https://test.com/api",
  "endpoints": [
    {
      "event-type": "Subscription",
      "endpoint-paths": [
        "integration/subscription",
        "monitoring/subscription"
      ]
    },
    {
      "event-type": "Payload",
      "endpoint-paths": [
        "integration/payload"
      ]
    }
  ],
  "access-token": "token",
  "max-retries": 10,
  "ignore-certificate-errors": true
}
```

**Responses:**

- 201 Created
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 404 Not Found: The specified values were not found in the database.
- 409 Conflict: The specified values already exists in the database.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**GET [api/v1/subscription](#)**

Get a list of all active subscriptions.

**Responses:**

- 200 OK: Returns a list of subscriptions in the same format as the subscription request bodies— see "[POST `api/v1/subscription`](#)" on page 538.
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**DELETE api/v1/subscription**

Removes all active subscriptions

- 204 No content: All subscriptions have been removed.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 404 Not Found: The specified values were not found in the database.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**POST api/v1/subscription/unsubscribe**

Unsubscribe all subscriptions to a selected base-URL.

Parameter	Type	Required	Constraints
base-url	string	Yes	Existing base-URL that is subscribed to any number of events. Defined in URI format.

```
{  
  "base-url": "https://test.com/api/"  
}
```

**GET api/v1/subscription/event-types**

Retrieve a list of all event-types in string format.

**Responses:**

- 200 OK: Returns an array of strings of all the Event types you can subscribe to. Response body:

```
{
  "event_types": [
    "event1",
    "event2"
  ]
}
```

- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### GET [api/v1/subscription/event-contracts](#)

Retrieve a list of all event-contracts in string format. The event-contracts describe the schemas of the [event data](#) that is sent to the subscribed endpoint.

#### Responses:

- 200 OK: Returns an array of strings of all the subscription contracts you have set up. Response body:

```
{
  "contracts": [
    "contract1",
    "contract2"
  ]
}
```

- 400 Bad Request: The request is invalid.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### POST [api/v1/subscription/resync](#)

All subscriptions you have set up send a snapshot event of their event type—see ["Snapshots" on page 486](#).

#### Responses:

- 202 Accepted
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

## System

Use system endpoints to control evacuations, retrieve robot information, and control the Continue/Pause state of robots.

### POST `api/v1/system/evacuation`

Start an evacuation.

**Request body:** None

**Responses:**

- 201 Created
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### DELETE `api/v1/system/evacuation`

Stop the current evacuation.

**Request body:** None

**Responses:**

- 200 OK
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### GET `api/v1/system/version`

Retrieve the software version of MiR Fleet.

**Responses:**

- 200 OK: Returns the current MiR Fleet software version in strong format. Response body:

```
{
  "version": "4.0.0"
}
```

- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### POST [api/v1/system/robot/continue](#)

Make a robot that is paused in a Wait for Continue action to continue to the next action in the mission.

#### Request body:

Parameter	Type	Required	Constraints	Example
robot-id	string	Yes	.NET format Specifier D	"7c9e6679-7425-40de-944b-e07fc1f90ae7"

```
{
  "robot-id": "7c9e6679-7425-40de-944b-e07fc1f90ae7"
}
```

#### Responses:

- 200 OK
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 404 Not Found: The specified values were not found in the database.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### Serial-order

Use serial-order endpoints to start or update serial-orders on MiR Fleet.



**POST `api/v1/serial-order`**

Create a new serial-order.

This endpoint can trigger an error event—see ["Serial-order error events" on page 507](#).

**Request body:**

Parameter	Type	Required	Constraints	Example
serial-order	serial-order object	Yes	Defined serial-order schema	{ "serial-order": { "id": "B62DA01F-9ABD-4d9d-80C7-02AF85C8228C", "phases": [] } }

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid. Happens when:
  - No Phases are provided in the request.
  - MissionId provided for any Phase is not in valid UUID format or does not exist.
  - ID is not in UUID format.
  - Any required parameter is missing according to the OpenApi spec.
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**PATCH `api/v1/serial-order`**

Update the priority, start-time, or assigned robot of a serial-order.

This endpoint can trigger an error event—see ["Serial-order error events" on page 507](#).

**Request body:**

Parameter	Type	Required	Constraints	Example
serial-order-id	string	Yes	.NET format Specifier D	"7c9e6679-7425-40de-944b-e07fc1f90ae7"
priority	string	Yes	Defined Priority type	"High"
earliest-start-time	string	No	Defined DateTime.ToUniversalTime in format: YYYY-MM-ddThh:mm:ssZ.	"2024-03-14T08:45:32Z"
robot-id	string	No	The robot-id of the robot that must execute the serial-order	"7c9e6679-7425-40de-944b-e07fc1f90ae7"

```
{
  "serial-order-id": "6c8e6679-7425-40de-944b-e07fc1f90ae5",
  "priority": "Low",
  "earliest-start-time": "2024-03-14T08:45:32Z",
  "robot-id": "7c9e6679-7425-40de-944b-e07fc1f90ae7"
}
```

### Responses:

- 202 Accepted
- 400 Bad Request: The request is invalid. Happens when:
  - ID is not in UUID format
  - Any required parameter is missing according to the OpenApi specifications
- 401 Unauthorized: Client is not authorized to make the specified request.
- 404 Not Found: The specified values were not found in the database.

**PUT `api/v1/serial-order/phase`**

Update the phase or phase arguments of a serial-order.

This endpoint can trigger an error event—see ["Serial-order error events" on page 507](#).

**Request body:**

Parameter	Type	Required	Constraints	Example
serial-order-id	string	Yes	.NET format Specifier D	"7c9e6679-7425-40de-944b-e07fc1f90ae7"
index	int	Yes	The phase index start from 0	1
phase	<a href="#">phase</a> object	Yes	Defined <a href="#">phase</a> schema	{ { "mission-id": "efd74342-9f8a-46d2-b711-b38e1225abae", "arguments": [ { "name": "arg-1", "value": "1" } ] } }

```
{
  "serial-order-id": "6c8e6679-7425-40de-944b-e07fc1f90ae5",
  "index": 1,
  "phase": {
    {
      "mission-id": "efd74342-9f8a-46d2-b711-b38e1225abae",
      "arguments": [
        {
          "name": "arg-1",
          "value": "1"
        }
      ]
    }
  }
}
```

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid. Happens when:
  - Phase Index is a negative number
  - MissionId provided for any Phase is not in valid UUID format or does not exist.
  - ID is not in UUID format
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

#### **DELETE** `api/v1/serial-order/id`

Abort a specific serial-order by specifying its `id` in the endpoint.

##### **Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid. Happens when ID is not in UUID format
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

#### **DELETE** `api/v1/serial-order/fallback/id`

Abort a specific serial-order's fallback mission by specifying its `id` in the endpoint.

##### **Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid. Happens when the ID is not in UUID format
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### **Top module**

Use the top module endpoint to trigger top module events and communicate with .

#### **POST** `api/v1/top-module`

##### **Request body:**

Parameter	Type	Required	Constraints	Example
robot-id	string	Yes	.NET format Specifier D	"7c9e6679-7425-40de-944b-e07fc1f90ae7"
event	Top Module event object	Yes	Defined event schema	{ "name": "MyTextEvent", "payload": { "MyName": "MyValue" } }

### Responses:

- 202 Accepted
- 400 Bad Request: The request is invalid. Happens when:
  - No Phases are provided in the request
  - MissionId provided for any Phase is not in valid UUID format or does not exist.
  - Id is not in UUID format
  - Any required parameter is missing according to the OpenApi spec;
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

## Group

Use the group endpoints to manage groups in MiR Fleet.

### GET /api/v1/group/snapshot

Get a list of all groups and the content of each group.

### Responses:

- 200 OK: Return a list of group events:

```

{
  "group-events": [
    {
      "entity-id": "string",
      "entity-type": "string",
      "action-type": "string",
      "status-change-timestamp": "2024-07-31T14:28:57.879Z",
      "group": {
        "name": "string",
        "robot-ids": [
          "string"
        ],
        "position-ids": [
          "string"
        ],
        "mission-group-ids": [
          "string"
        ]
      },
      "mission-group": {
        "name": "string",
        "mission-ids": [
          "string"
        ]
      }
    }
  ]
}

```

- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

### POST /api/vi/group

Create a new group.

#### Request body:

Parameter	Type	Required	Constraints	Example
id	string	Yes	.NET format Specifier D	"7c9e6679-7425-40de-944b-e07fc1f90ae7"

Parameter	Type	Required	Constraints	Example
name	String	Yes	Uniquq groups name	"Group name"
robot-ids	List of strings	No	.NET format Specifier D	[ "7c9e6679-7425-40de-944b-e07fc1f90ae7", "a90f0e9d-5091-4e97-bf85-819da301a451" ]
position-ids	List of strings	No	.NET format Specifier D	
mission-group-ids	List of strings	No	.NET format Specifier D	

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid. Happens when:
  - Id is not in UUID format
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**DELETE /api/v1/group/id**

Delete the group with the entered ID.

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid. Happens when:
  - Id is not in UUID format
- 401 Unauthorized: Client is not authorized to make the specified request.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

## Robot

[GET /api/v1/robot/robot-identity/snapshot](#)

### 5.1.6 Schemas and types

#### Section overview

- The format of all the objects used in MiR Fleet Integration API.
- All valid types you can use in certain fields.

Use the following sections as reference when you are sending requests to the API and interpreting the responses.

## Site

### map

A map where geo-JSON objects are used to define the walls in the floor plan.

Parameter	Type	Required	Description
name	string	Yes	The name of the map.
id	string	Yes	Unique identification code for the map in .NET format



Parameter	Type	Required	Description
			Specifier D.
geo-json	GeoJSON	Yes, if POSTing to map endpoint	Polygon coordinates to define the walls in the map.
image	string	Yes, if part of <a href="#">site-event</a> .	The name of the floor plan image in the MiR software.
origin	origin object	Yes, if part of <a href="#">site-event</a> .	The image coordinates where the map's origin is.

#### origin

The (0, 0) point of a map relative to the floor plan image.

Parameter	Type	Required	Description
x	double	Yes	The X coordinate of the floor plan image where the X coordinate of the map is 0.
y	double	Yes	The Y coordinate of the floor plan image where the Y coordinate of the map is 0.

#### position

Any type of position or marker you can put on a map and all of the settings related to that position.

Parameter	Type	Required	Description
type	string	Yes	<ul style="list-style-type: none"> <li>"BasePosition": A position without additional properties.</li> <li>"Marker": A standard marker or pallet rack.</li> <li>"Charger": A charging station.</li> <li>"UtilityPosition": A Cart or Shelf position.</li> </ul>
id	string	Yes	Unique identification code for the position in .NET format Specifier D.
map-id	string	Yes	The "id" of the <a href="#">map</a> object the position is in.
name	string	Yes	The unique name of the position.
map-pose	<a href="#">pose</a> object	Yes	The coordinate point on the map the position is located.

#### If "type": "BasePosition"

Parameter	Type	Required	Description
base-position-type	string	Yes	<ul style="list-style-type: none"> <li>"Robot": Robot position</li> <li>"Emergency": Evacuation position</li> <li>"Staging": Staging position</li> </ul>

## If "type": "Marker"

Parameter	Type	Required	Description
marker-type	string	Yes	<ul style="list-style-type: none"> <li>"V": V-marker</li> <li>"VL": VL-marker</li> <li>"L": L-marker</li> <li>"Bar": Bar-marker. This type requires that you define the "bar" property.</li> <li>"PalletRack": Pallet rack marker</li> </ul>
offset-pose	pose object	No	The docking offset.
entry-position-offset-pose	pose object	Yes	The location of the Entry position relative to the marker's "map-pose".
bar	bar object	Yes	If "marker-type": "Bar", the size of the Bar marker.

## If "type": "Charger"

Parameter	Type	Required	Description
charger-type	string	Yes	<ul style="list-style-type: none"> <li>"MiRCharge24V": MiR Charge 24V marker.</li> <li>"MiRCharge48V": MiR Charge 48V marker.</li> </ul>

Parameter	Type	Required	Description
entry-position-offset-pose	<a href="#">pose</a> object	Yes	The location of the Entry position relative to the marker's "map-pose".

#### If "type": "UtilityPosition"

Parameter	Type	Required	Description
utility-position-type	string	Yes	<ul style="list-style-type: none"> <li>"Shelf": Shelf position.</li> <li>"Cart": Cart position.</li> </ul>
entry-position-offset-pose	<a href="#">pose</a> object	Yes	The location of the Entry position relative to the marker's "map-pose".
entry-positions	<a href="#">entry-position</a> object	Yes	Define the Entry positions related to the position.
offset-pose	<a href="#">pose</a> object	Yes	The docking offset.

#### [entry-position](#)

The Entry positions for a Cart or Shelf position.

Parameter	Type	Required	Description
entry-position-type	string	Yes	<ul style="list-style-type: none"> <li>CartLeft</li> <li>CartRight</li> </ul>

Parameter	Type	Required	Description
			<ul style="list-style-type: none"> <li>• CartPickup</li> <li>• ShelfShort</li> <li>• ShelfLong</li> </ul>
offset-pose	<a href="#">pose</a> object	Yes	The location of the Entry position relative to the marker's "map-pose".

### [pose](#)

The base object to describe the position of any element on the map. If your object is af "offset", the pose is relative to position of the parent object.

Parameter	Type	Required	Description
x	double	Yes	The X translation relative to the parent coordinate in meters. This is often translation forward or backward. The value you enter is rounded to 3 decimals.
y	double	Yes	The Y translation relative to the parent coordinate in meters. This is often translation to the left or right. The value you enter is rounded to 3 decimals.
orientation	double	Yes	The orientation relative to the parent coordinate in degrees. The orientation rotates counter-clockwise. The value you enter is rounded to 0 decimals.

**bar**

The length of a bar and distance between bars for a Bar-marker.

Parameter	Type	Required	Description
length	double	Yes	The length of the bars of the Bar-marker.
distance	double	Yes	The distance between the bars of the Bar-marker

**zone**

MiR Fleet management zones you can put on a map and all of the settings related to the zones.

Parameter	Type	Required	Description
id	string	Yes	Unique identification code for the zone in .NET format Specifier D.
name	string	Yes	Unique name for the zone.
map-id	string	Yes	The "id" of the <a href="#">map</a> object the zone is in.
geo-json	GeoJSON	Yes	Polygon coordinates to define the position and shape of the zone.

Parameter	Type	Required	Description
type	string	Yes	<ul style="list-style-type: none"> <li>"LockZone": A Lock zone with the "is-locked" field</li> <li>"IntegrationZone": An Integration zone with the "events" field.</li> </ul>
is-locked	bool	If "type": "LockZone"	Stops robots from entering the zone when "True".
events	list of zone-event objects	If "type": "IntegrationZone"	For each event, you can define any custom JSON payload event.

#### zone-event

The payload event that occurs when a robot enters or exits an Integration zone.

Parameter	Type	Required	Description
name	string	Yes	Unique identification code for the zone in .NET format Specifier D.
event	JSON payload	Yes	The custom content that any clients subscribed to Payload

Parameter	Type	Required	Description
type	string	Yes	<ul style="list-style-type: none"> <li>"Entry": The event is triggered when the robot enters the zone.</li> <li>"Exit": The event is triggered when the robot exits the zone.</li> </ul>
address	string	Yes	<ul style="list-style-type: none"> <li>"Integration": The event is triggered in MiR Fleet Integration API.</li> <li>"TopModule": The event is triggered in Top Module API.</li> </ul>

### mission

A mission that can be scheduled as a serial-order

Parameter	Type	Required	Description
name	string	Yes	The name of the mission.

## Subscription

### endpoints

List of endpoints you want to subscribe to a selected event-type.



Parameter	Type	Required	Description
event-type	string	Yes	<ul style="list-style-type: none"><li>"Alert": events concerning Fleet UI alerts.</li><li>"Error": events concerning errors that have occurred in the system.</li><li>"Payload": events concerning JSON payloads triggered by Integration actions or Integration zones.</li><li>"SerialOrderStatus": events concerning updates in serial-orders and scheduled missions.</li><li>"RobotIdentity": events concerning changes to robot identities.</li><li>"RobotRuntime": events sent every second for every robot describing its <a href="#">pose</a>.</li><li>"RobotState": events concerning changes in the robot states.</li><li>"Site": events concerning changes to site data. This includes maps and positions.</li><li>"Subscription": events concerning a user subscribing or unsubscribing to an event type.</li></ul>

Parameter	Type	Required	Description
			<ul style="list-style-type: none"> <li>"System": events concerning evacuation state changes and robots.</li> <li>"TopModule": events concerning data sent from top modules (always associated with a robot).</li> </ul>
endpoint-paths	list of strings	Yes	List of all endpoints from the base-url you want to subscribe to the selected event-type. Are listed without the base-url and without starting slash.

## SerialOrder

### serial-order

A series of missions with arguments. MiR Fleet assigns a valid robot to execute the series of mission in the specified order. Each mission and its arguments are defined in a phase of the serial-order.

Parameter	Type	Required	Description
id	string	Yes	Unique identification code for the serial-order in .NET format Specifier D.
priority	string	No	<ul style="list-style-type: none"> <li>"High": MiR Fleet assigns all valid pending high priority missions first.</li> </ul>

Parameter	Type	Required	Description
			<ul style="list-style-type: none"> <li>"Medium": MiR Fleet assigns medium priority missions when there are no pending high priority missions.</li> <li>"Low": MiR Fleet assigns low priority missions when there are no other valid pending missions.</li> </ul>
earliest-start-time	string	No	The earliest time, in Coordinated Universal Time (UTC) format, that the first mission can be assigned to a robot.
robot-id	string	No	The robot-id of the robot that must execute the serial-order
phases	list of <a href="#">phase</a> objects	Yes	Defined <a href="#">Phase</a> schema

### [phase](#)

Defines a mission you want to add to a serial-order and the values for the mission arguments.

Parameter	Type	Required	Description
mission-id	string	No	Valid UUID and refers to an existing mission
arguments	list of <a href="#">argument</a> objects	No	Defined <a href="#">Argument</a> schema

Parameter	Type	Required	Description
fallback-mission-id	int	No	Positive number and refers to an existing mission

### argument

Values you can pass to arguments in the mission of the phase.

Parameter	Type	Required	Description
name	string	Yes	Name of the argument.
value	string	Yes	The value you set the argument to.

## System

### evacuation

Information about the current evacuation state.

Parameter	Type	Required	Description
evacuation-state	string	Yes	<ul style="list-style-type: none"> <li>"NoEvacuation": no evacuation is ongoing.</li> <li>"InEvacuationOngoing": the system is in evacuation mode and the relevant actions are being executed.</li> <li>"InEvacuationComplete": the system is in evacuation mode and the relevant actions have been completed.</li> </ul>

### robot

Information about a robot connected to MiR Fleet

Parameter	Type	Required	Description
robot-id	string	Yes	Unique identification code for the map in .NET format Specifier D.
robot-state	string	Yes	<ul style="list-style-type: none"><li>"Added": the robot has been added to the fleet.</li><li>"Removed": the robot has been removed from the fleet.</li><li>"ConnectionAvailable": the robot has reconnected to MiR Fleet.</li><li>"ConnectionUnavailable": the robot has lost connection to MiR Fleet.</li></ul>

## Top Module

### event

The information received when you define a custom event using the Top Module Integration API.

Parameter	Type	Required	Description
name	string	Yes	The name of the event defined in the Top Module API.
payload	JSON payload	Yes	The custom content defined in the Top Module API.

Parameter	Type	Required	Description
robot-id	string	Yes	The ID of the robot that triggered the event.
serial-order-id	string	Yes	The ID of the serial-order the robot was executing when the event was triggered.
phase-index	int	Yes	The phase in the serial-order the robot was executing when the event was triggered.
is-fallback	bool	Yes	Is "True" if the phase is a fallback phase.
timestamp	string	Yes	The time the event occurred.

### error

The information received when you define a custom error using the Top Module Integration API.

Parameter	Type	Required	Description
name	string	Yes	The name of the error.
message	string	Yes	More information about the error.
robot-id	string	Yes	The ID of the robot the error occurred on.
serial-order-id	string	Yes	The ID of the serial-order the

Parameter	Type	Required	Description
			robot was executing when the error occurred.
phase-index	int	Yes	The phase in the serial-order the robot was executing when the error occurred.
is-fallback	bool	Yes	Is "True" if the phase is a fallback phase.
timestamp	string	Yes	The time the error occurred.

## Groups

### group

Parameter	Type	Description
name	string	The name of the group.
robot-ids	List of strings	List of IDs (in .NET format Specifier D) for all robot in the group.
position-ids	List of strings	List of IDs (in .NET format Specifier D) for all positions in the group.
mission-group-ids	List of strings	List of IDs (in .NET format Specifier D) for all mission groups in the group.

### mission-group

Parameter	Type	Description
name	string	The name of the mission group.
mission-ids	List of strings	List of IDs (in .NET format Specifier D) for all missions in the group.

## Events

### system-events

Information about the evacuation state and robots connected to MiR Fleet.

Parameter	Type	Description
evacuation	<a href="#">evacuation</a> object	The evacuation status of MiR Fleet.
robot	<a href="#">robot</a> object	The status of robots connected to MiR Fleet.
timestamp	string	The time the event occurred.

### serial-order-status-events

The status of any phases within an order from when they are scheduled until they are completed or aborted.

Parameter	Type	Description
serial-order-id	string	The ID of the serial-order the phase is part of.



Parameter	Type	Description
phase-index	int	specifies which phase in the serial-order is relevant to the event.
state	string	<ul style="list-style-type: none"><li>"Waiting": The phase is scheduled and is waiting for the earliest start time.</li><li>"Pending": The phase is scheduled and is ready to be assigned to a robot.</li><li>"Outbound": The phase has been assigned to a robot, but the communication is still in transit to the robot.</li><li>"Executing": The phase is being executed by a robot.</li><li>"Finished": The phase has finished.</li><li>"Aborting": The phase has been aborted, but the communication is still in transit to the robot.</li><li>"Aborted": The phase has been aborted, and the robot has confirmed the abort phase.</li><li>"Paused": The phase is paused because the robot is in Paused state.</li><li>"WaitingForInput": The phase is missing arguments or a defined mission.</li></ul>
state-change-timestamp	string	The time the phase changed status.
robot-id	string	The ID Of the robot executing the phase.
message	string	If "state": "WaitingForInput" the message is one of the following:

Parameter	Type	Description
		<ul style="list-style-type: none"><li>"MissingMissionId"</li><li>"MissingArguments"</li><li>"UnknownMissionId"</li><li>"IncompleteRequest"</li></ul>
is-fallback	bool	Is "True" if the phase status event is for a fallback phase.
order-type	string	<ul style="list-style-type: none"><li>"UI": an order scheduled in the MiR Fleet interface.</li><li>"Api": an order scheduled through MiR Fleet Integration API</li><li>"Evacuation": an order related to an evacuation triggered in MiR Fleet.</li><li>"Autocharging": an order to send the robot to charge at a station sent though the Auto charging sequence.</li><li>"Autostaging": an order to send the robot to a Staging position sent though the Auto staging sequence.</li><li>"GoTo": an order sent to a robots using the <b>Go to</b> command in a map viewer in the MiR Fleet interface.</li></ul>

### robot-identity-events

Information about individual robots connected to MiR Fleet.

Parameter	Type	Description
robot-id	string	The ID of the robot.
serial-number	string	The serial number of the robot.
name	string	The name of the robot.
model	string	The robot model.
software-version	string	The software version the robot is running.
timestamp	string	The time the robot changed status.

#### site-events

Defines maps or positions and zones on a map in MiR Fleet.

Parameter	Type	Description
entity-id	string	The ID of the site element.
entity-type	string	<ul style="list-style-type: none"><li>"Map"</li><li>"Position"</li><li>"Zone"</li><li>"Mission"</li></ul>
action-type	string	<ul style="list-style-type: none"><li>"Create": created a entity.</li><li>"Update": updated or modified an entity.</li></ul>

Parameter	Type	Description
		<ul style="list-style-type: none"> <li>"Delete": deleted an entity.</li> </ul>
status-change-timestamp	string	The time the site element changed status.
map	map object	The map that changed.
position	position object	The position that changed.
zone	zone object	The zone that changed.
mission	mission object	The mission that changed.

### group-events

Defines the groups and mission groups in MiR Fleet and which robots, missions, mission groups, and positions are in each group.

Parameter	Type	Description
entity-id	string	The ID of the site element.
entity-type	string	<ul style="list-style-type: none"> <li>"Group"</li> <li>"MissionGroup"</li> </ul>
action-type	string	<ul style="list-style-type: none"> <li>"Create": created an entity.</li> <li>"Update": updated or modified an entity.</li> <li>"Delete": deleted an entity.</li> </ul>

Parameter	Type	Description
status-change-timestamp	string	The time the site element changed status.
group	<a href="#">group</a> object	The group that changed.
mission-group	<a href="#">mission-group</a> object	The mission-group that changed.

### 5.1.7 Use case examples

The following sections provide suggested processes for how to get started with the various API features.

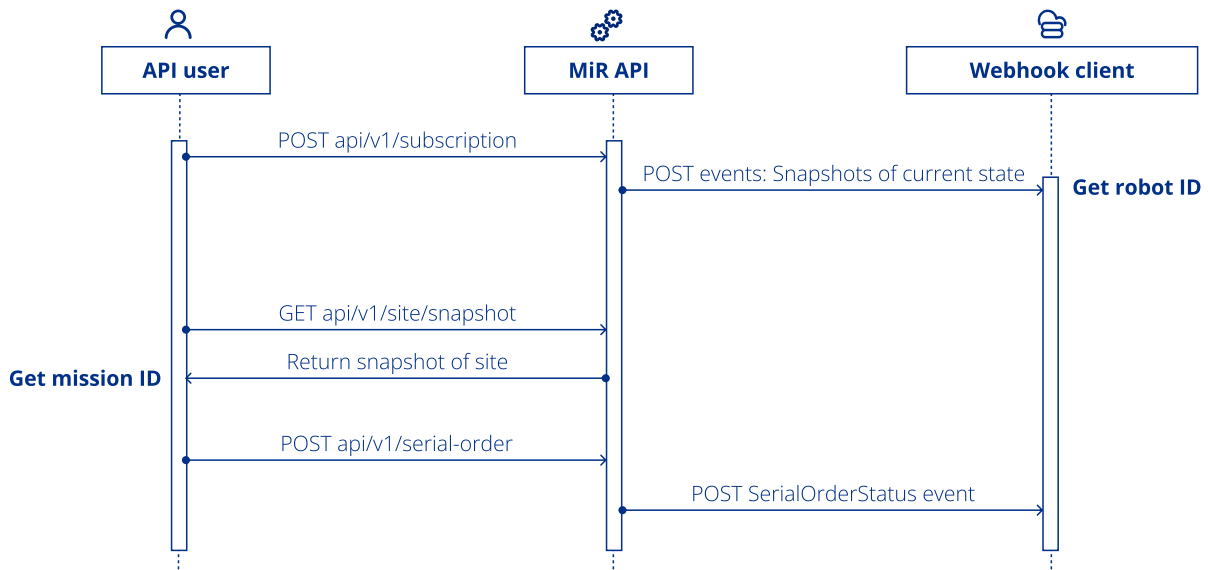
#### Schedule missions

User serial-orders to schedule missions—see ["Serial-orders" on page 521](#).

To schedule missions, you must POST a request to `/api/v1/serial-order` that includes the mission ID and assigns values to any mission arguments.

To receive status information and robot identities from the API, create a webhook client and subscribe it to:

- `SerialOrderStatus` to receive updates on the status of the mission after you schedule it.
- `RobotIdentity` to get the ID of all connected robots. You need this if you want to specify which robot the mission must be assigned to.
- `Error` to receive notifications if MiR Fleet failed to complete the action you requested.



Follow these steps to schedule a mission through the API:

- 1 (optional) Set up a webhook client to receive events. The client must be configured to receive data that follows the structure of the following event-contracts:
  - **SerialOrderStatus event-contract:** These events update the client on any change in the status of scheduled missions.
  - **RobotIdentity event-contracts:** The initial snapshot from this event provides the robot IDs if you want to assign the mission to a specific robot.
  - **Error event-contracts:** These events indicate if you made a valid API request but the action itself failed.

- 2 If you set up a webhook client, send POST `<MiR_Fleet_IP_address>/api/v1/subscription` with the following content:

```
{
  "base-url": "https://<Webhook_client_IP>/api",
  "endpoints": [
    {
      "event-type": "SerialOrderStatus",
      "endpoint-paths": [
        "scheduled_missions/status"
      ]
    },
    {
      "event-type": "RobotIdentity",
      "endpoint-paths": [
        "robots"
      ]
    },
    {
      "event-type": "Error",
      "endpoint-paths": [
        "errors"
      ]
    }
  ],
  "access-token": "token",
  "max-retries": 10,
  "ignore-certificate-errors": true
}
```

This makes the MiR Fleet Integration API POST:

- SerialOrderStatus events to `<Webhook_client_IP>/api/scheduled_missions/status`.
- RobotIdentity events to `<Webhook_client_IP>/api/robots`. A snapshot of the current state is sent immediately after subscribing. Use this to see the ID of each robot connected to MiR Fleet.
- Error events to `<Webhook_client_IP>/api/error`.

- 3 Send GET `<MiR_Fleet_IP_address>/api/v1/site/snapshot` to MiR Fleet.

The return message describes all the maps, map components, and missions in the site.

- 4 Find the mission you want to schedule in the site return message and determine the mission ID.

```
{
  "site-events": [
    {
      "entity-id": "mission_id",
      "entity-type": "mission",
      "mission": {
        "name": "mission_name"
      }
    }
  ]
}
```

- 5 Send POST <MiR\_Fleet\_IP\_address>/api/v1/serial-order with the following content:

```
{
  "serial-order": {
    "id": "746bae3c-296e-48bf-b8c4-e7cda7404626",
    "robot-id": "optional_robot_id",
    "phases": [
      {
        "mission-id": "mission_id"
      }
    ]
  }
}
```

Replace `mission_id` with the ID of the mission and if you want to schedule to a specific robot, replace `optional_robot_id` with the ID of the robot.

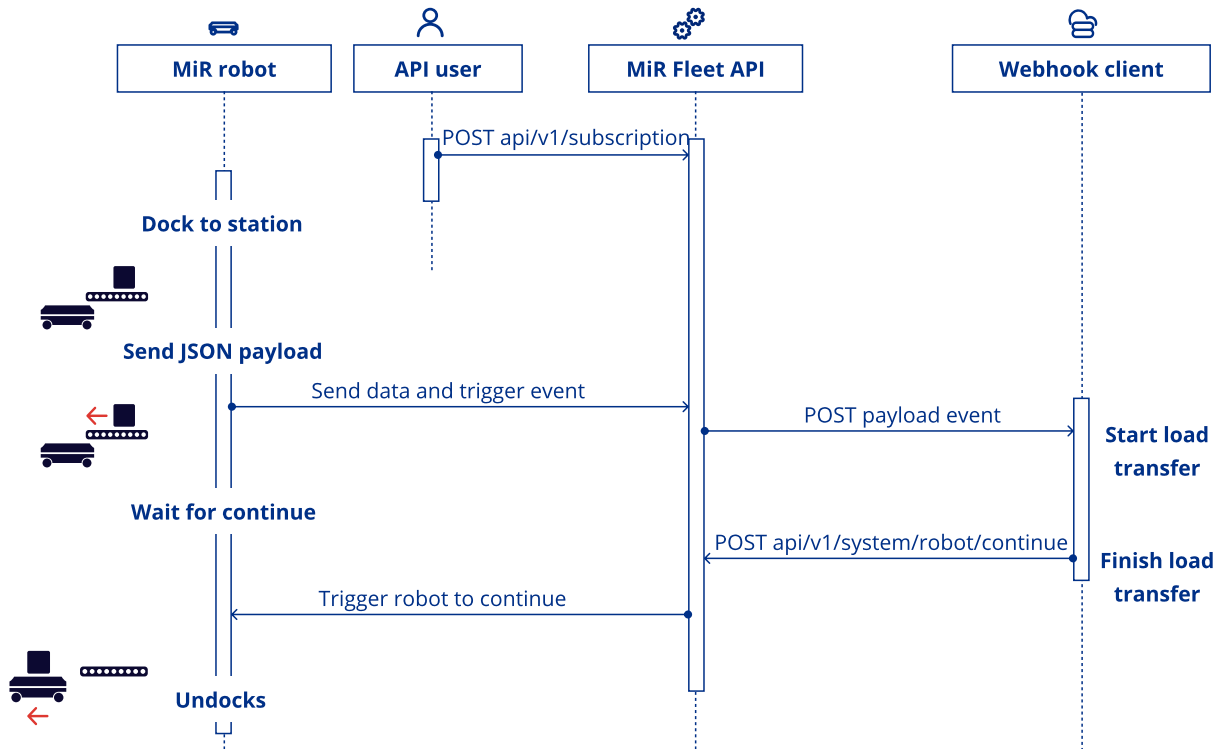
If your mission uses arguments, pass arguments to the mission as described in ["Serial-orders" on page 521](#).

- 6 (Optional) Follow the status of the mission if you subscribe to `SerialOrderStatus`, or see the mission status in the MiR Fleet interface.

## Use Integration actions

Use Integration actions when there is a point in a mission sequence where robots need to trigger an external device or wait for a signal.





For example, you can install a load transfer station that:

- Begins loading when its controller receives an API call from MiR Fleet Integration API.
- Posts an API call to `api/v1/system/robot/continue` when it finishes loading to let the robot continue its mission.

Follow these steps to create a mission to trigger a REST API controlled load transfer station and configure your client to receive payload events:

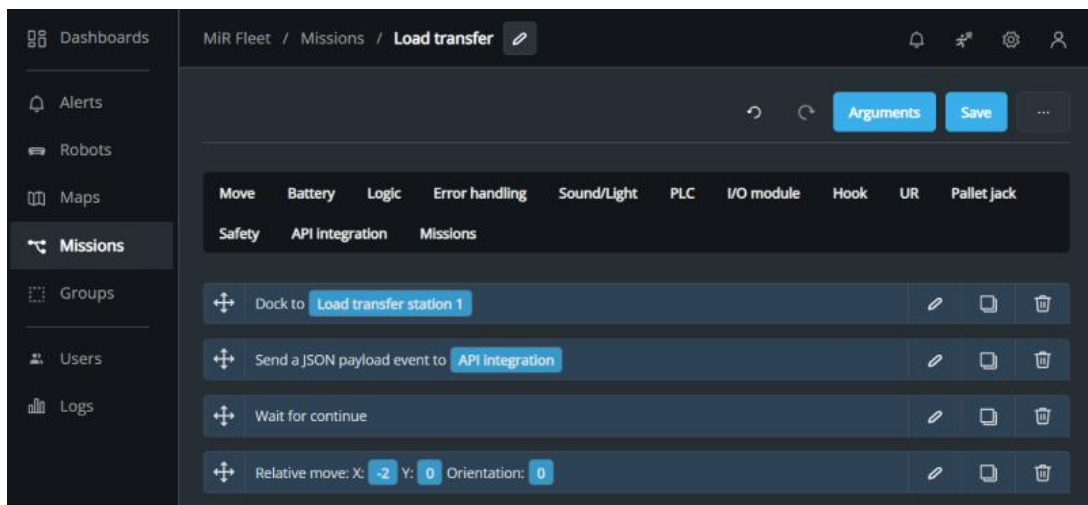
- 1 Set up a webhook client to receive payload events. The client must be configured to receive data that follows the structure of the payload event including your custom JSON payload.

- 2 Send POST <MiR\_Fleet\_IP\_address>/api/v1/subscription with the following content:

```
{
  "base-url": "https://<Webhook_client_IP>/api",
  "endpoints": [
    {
      "event-type": "Payload",
      "endpoint-paths": [
        "payload_message"
      ]
    }
  ],
  "max-retries":10,
  "ignore-certificate-errors":true
}
```

This makes the MiR Fleet Integration API POST payload events to <Webhook\_client\_IP>/api/payload\_message

- 3 Create a mission that makes the robot dock to the transfer station, send a JSON payload, and wait for a continue response.



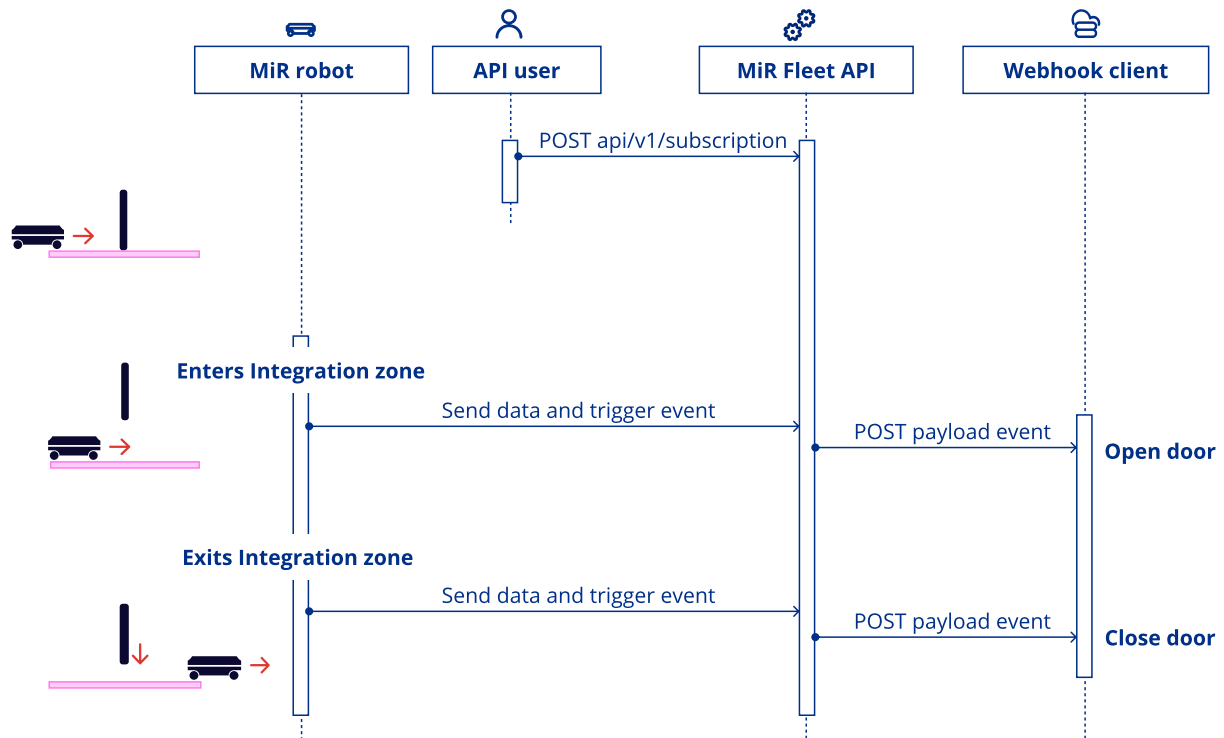
You can customise the JSON payload to include information specific to this mission, for example, which load transfer station to trigger, whether to load or unload the robot, the load type.

- 4 Configure your transfer station controller to:
  - Activate the transfer when the webhook client receives the payload event triggered by the mission action.
  - Send a POST `<MiR_Fleet_IP_address>/api/v1/robot/continue` when the transfer station is done loading.
  
- 5 Run the mission. The load transfer station should activate once the robot finishes docking, and the robot should undock one the load transfer station is finished loading.

### Use Integration zones

Integration zones are useful when there are areas where robots need to communicate with the API to trigger events.

For example, you can use this to control automatic doors so they activate when the robot enters a specific area around the door.



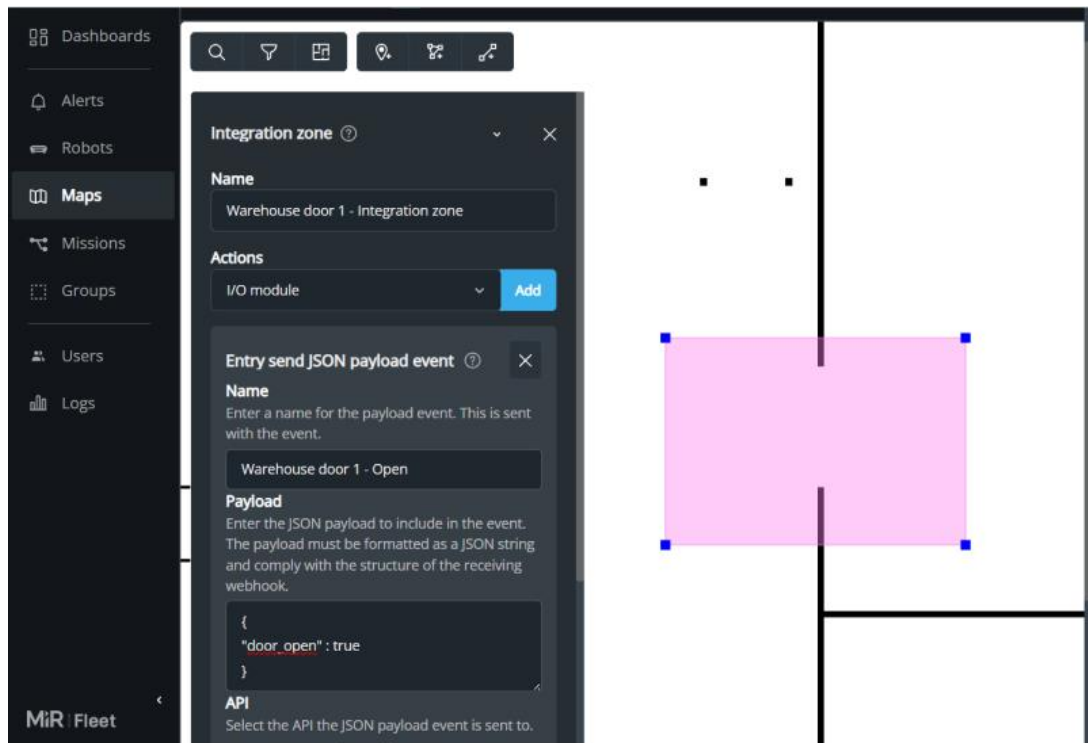
Follow these steps to create a zone to trigger a REST API controlled door and configure your client to receive payload events:

- 1 Set up a webhook client to receive payload events. The client must be configured to receive data that follows the structure of the payload event including your custom JSON payload.
- 2 Send POST <MiR\_Fleet\_IP\_address>/api/v1/subscription with the following content:

```
{
  "base-url": "https://<Webhook_client_IP>/api",
  "endpoints": [
    {
      "event-type": "Payload",
      "endpoint-paths": [
        "payload_message"
      ]
    }
  ],
  "max-retries":10,
  "ignore-certificate-errors":true
}
```

This makes the MiR Fleet Integration API POST payload events to <Webhook\_client\_IP>/api/payload\_message

- 3 Add an Integration zone around the door in the map editor. Configure the zone with:
  - An Entry send JSON payload event action that sends a payload event to open the door.
  - An Exit send JSON payload event action that sends a payload event to close the door.



- 4 Configure your door controller to:
  - Open the door when the webhook client receives the Entry JSON payload event.
  - Close the door when the webhook client receives the Exit JSON payload event.
- 5 Send the robot on a mission through the door. If the robot does not get close enough to the door to enter the Integration zone, add an Access zone on top of the door to make the robot drive closer.

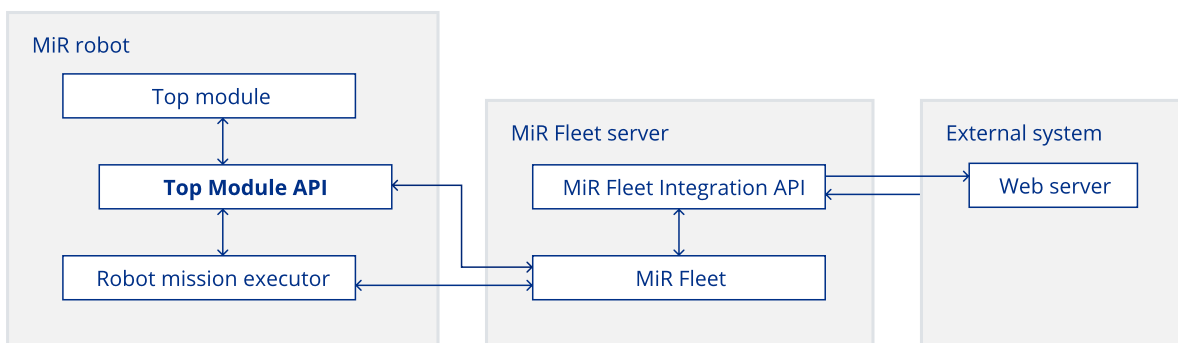
## 5.2 Top Module API

The MiR Top Module API is intended to enable communication between custom top modules and the MiR system. Your top module must have a device that is capable of sending POST requests to the API.

The API runs directly on each robot. Any information that needs to be shared across multiple robots must be forwarded to the MiR Fleet Integration API.

It enables you to define custom events, errors, and commands for your top module that can be forwarded to the MiR system. The API is able to influence the robot mission executor using the integration actions in the mission editor.

MiR Top Module API is based on the RESTful architectural style.



### 5.2.1 Errors and events

The main purpose of the Top Module API is to enable the connected top module to trigger errors or events that are used by the robot or MiR Fleet. You can customize the data that is shared when the top module triggers an event or error.

### 5.2.2 Webhooks

The API is highly event-based, and most use cases involve webhooks to receive data—see ["Webhooks" on page 490](#).

You can set up webhooks that you subscribe to events in MiR Fleet. When a triggering event occurs, the webhook server you have set up a subscription for receives a POST request from the API with information about the event—see ["Event subscriptions" on page 496](#).

Use webhooks for the following:

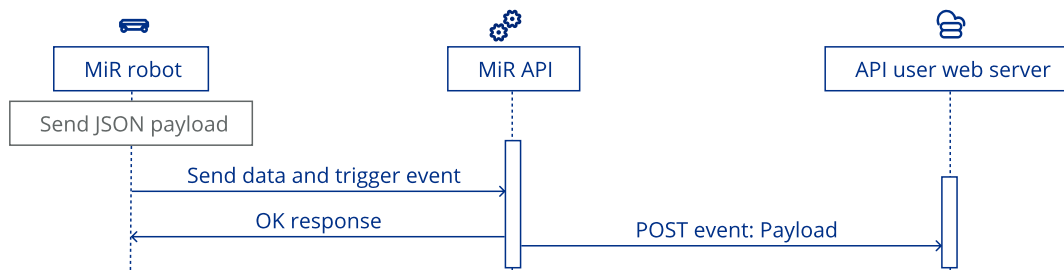
- Receive updates whenever anything changes in the MiR Fleet system.
- Program system responses when certain events occur.
- Monitor mission and robot statuses.

### 5.2.3 Mission actions

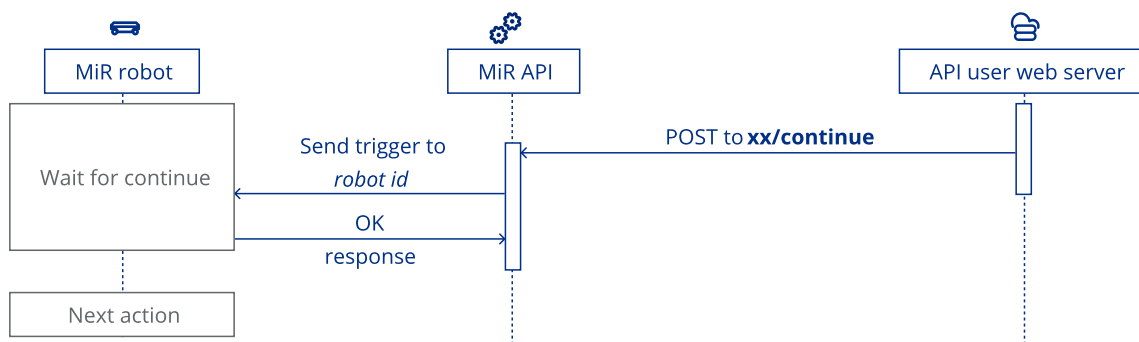
There are two actions you can use in missions to enable direct communication to MiR Top Module API during a mission.

**Send JSON payload** triggers the robot to send a JSON payload via the chosen API. Subscribe to the Payload event for the chosen API to make the API push the payload event to the subscribed endpoint. The event also includes the ID of the robot that executed the action to trigger the event.

There are no restrictions to the format of the payload. This action is designed to enable any custom communication to be sent to your own client when a robot reaches a specific part of its mission.



**Wait for continue** makes the robot wait until you POST the robots ID to `api/v1/system/robot/continue` OR `api/v1/continue`.



Use these two actions in combination to make the robot communicate its status and wait to continue its order until it receives a confirmation.

### 5.2.4 Top module analytics data

Use the API to collect metric data from the top module. This is intended for counts and state values. For each data set you define a name and a float value.

### 5.2.5 Robot states

The API can retrieve the current state of the robot. Use this to manage how the top module reacts to certain robot states.

### 5.2.6 Getting started

#### Section overview

- Determine the address you use to send requests to endpoints.
- View the Swagger page with all available endpoints.
- Create an API key to authenticate your requests.
- Get started creating webhooks.

The base URL address is: `<robot_ip>:5050/api/v1` where `<robot_ip>` is the IP address of the robot. To access any API endpoint, add it to the base URL.

If you are on the robot's network, use `192.168.12.20:5050/api/v1` as the base URL.

You can see an overview of all the endpoints you can use in ["Endpoints" on page 588](#).

None of the output messages from the API are encoded. This enables you to integrate the level and type of protection that it suitable for your integration.

#### Webhooks

Use webhooks with MiR Top Module API to receive event information when triggering events occur. To set up a webhook to listen to events:

- Create a webhook client—see ["Webhooks" on the next page](#).
- Use the subscription endpoint to subscribe your webhook server to a specific events type—see ["Endpoints" on page 588](#) and ["Event subscriptions" on page 586](#).



- Program your client to parse the event data it receives when an event occurs. Use the event-contracts to see the data format—see ["Schemas and types" on page 594](#).
- Program the client to send appropriate responses to other devices, interfaces, or endpoints based on the events that occur—see ["Endpoints" on page 588](#).

We recommend starting with subscribing Error events and displaying all error events in an interface. Error events report any errors that occur while you are using the API.

## 5.2.7 Webhooks

### Section overview

- How to create a webhook client
- How to set up a subscription for the client
- How MiR robots communicate with the client

Set up webhook clients to listen for events that you can subscribe to—see ["Event subscriptions" on the next page](#).

### Client Setup

Your webhook client application must:

- Be an API based on REST principles.
- Be able to receive an event. Specifically:
  - A controller with an implemented POST method.
  - A data structure corresponding to the expected event-contracts—see ["Event subscriptions" on the next page](#). You can also use the `GET api/v1/subscription/event-contracts` method to receive a list of all the event-contracts.
- Have a controller for every event type that the user is expecting to receive since all events have different data structures.
- Have an endpoint addresses that specifies a full URL (not just the base URL) for specific event publishing. For example, the address for Order Status events might look like: `http://some-address/events/order-status`.

The following example shows the contract for the subscription event that is published when a subscription is added.

```

{
  subscription-id": "0f8fad5b-d9cb-469f-a165-70867728950e"
  "event-name": "Subscription"
  "state": "Subscribed"
  "user-friendly-message": "You are now subscribed"
  "timestamp": "2022-10-05T12:35:09.929Z"
}

```

MiR Top Module API is event-based by design and sends the events as JSON objects in POST requests to the endpoints implemented in the your webhook client. The endpoints should follow the structure specified in the event-contracts—see ["Event subscriptions" below](#).

The output events from the API are not encoded.

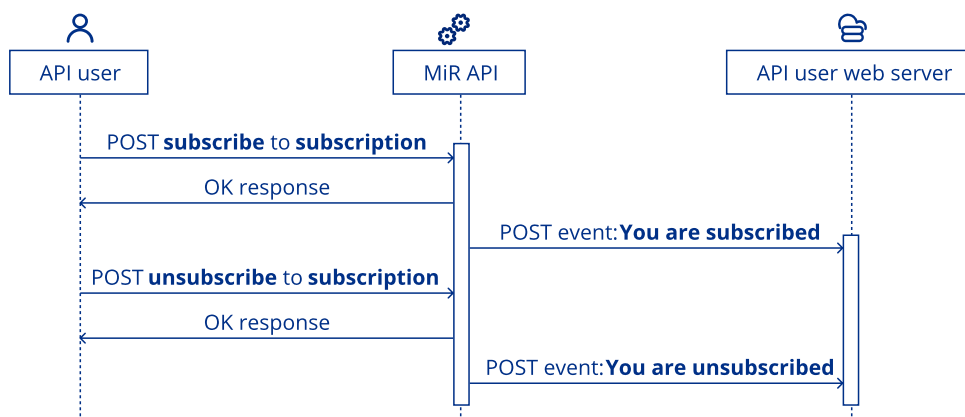
## 5.2.8 Event subscriptions

### Section overview

- Get an overview of what data you can subscribe webhook clients to.
- See what triggers the different event-types.

This page lists all of the events you can subscribe a custom [webhook](#) client to and what triggers each event-type. MiR Top module API sends POST requests to the subscribed webhook client when an event is triggered.

When you subscribe or unsubscribe a subscription, you receive a response to confirm that the subscription was set up correctly, and any webhooks that subscribe to [Subscription](#) receive an event.



## Abort

Subscribe to `Abort` to receive a timestamp when the robot aborts a mission.

Parameter	Type	Description
timestamp	string	Timestamp for aborted order from robot in UTC format.

### Abort event-contract

```
{
  "timestamp": "2022-12-20T15:05:57.1417291Z"
}
```

## Event

Subscribe to `Event` to receive custom events when they are posted to the API through the `api/v1/event` endpoint or mission action.

Parameter	Type	Description
name	string	Name of the custom event.
payload	event object	Custom JSON payload with event information.
timestamp	string	Timestamp for event in UTC format.

### Event event-contract

```
{
  "name": "FirstEvent",
  "payload": { "Speed": "High", "Height": 100, "Simple": false },
  "timestamp": "2022-09-19T09:08:12.8651699Z"
}
```

## Status

Subscribe to `Status` to receive an event when the robot changes status.

Parameter	Type	Description
<code>status</code>	string	Name of the status the robot changed to.
<code>timestamp</code>	string	Timestamp for status change in UTC format.

### Status event-contract

```
{
  "status": "Pause",
  "timestamp": "2022-09-19T09:08:12.8651699Z"
}
```

## 5.2.9 Endpoints

### Section overview

- Endpoints you can use in the MiR Top Module API
- The purpose of each field in the endpoints
- The format of many common requests

Use endpoints to make the top module send events and errors, record metrics, or to subscribe webhook clients to events.

There is a rate limit to the number of requests you can send based on the Sliding Window strategy:

- Maximum number of requests per window: 200
- Window duration: 1 minute

If the number of requests surpasses the maximum amount allowed, a 503 Service Unavailable HTTP Response will be returned. This provides security against Denial of Service (DoS) attacks.

## GET `api/v1/event`

Receive a list of all recorded events in MiR Top Module API.

### Responses:

- 200 OK: returns a list of [event-entries](#). Response body:

```
{
  "event-entries": [
    {
      "name": "FirstEvent",
      "payload": { "Speed": "Low", "Height": 70, "Simple": true },
      "timestamp": "2022-09-19T09:08:12.8651699Z"
    },
    {
      "name": "SecondEvent",
      "payload": { "Speed": "High", "Height": 100, "Simple": false },
      "timestamp": "2022-09-19T09:08:12.8651699Z"
    }
  ]
}
```

## POST `api/v1/event`

Post an event in the Top Module API. The API forwards this event to any client that is subscribed to the `TopModule` event-type in either the Top Module API or the MiR Fleet Integration API.

### Request body:

Parameter	Type	Required	Constraints
name	string	Yes	
payload	JSON payload	Yes	Content uses number, string, and boolean types only.

```
{
```

```
"name": "MyTextEvent",  
"payload": { "MyName": "MyValue" }  
}
```

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 500 Internal Server Error: The robot was unable to complete the request.

**POST `api/v1/error`**

Post an error in the Top Module API. The API forwards this error to any client that is subscribed to the `TopModule` event-type on MiR Fleet Integration API.

**Request body:**

Parameter	Type	Required	Constraints
name	string	Yes	
message	string	Yes	

```
{  
  "name": "WarningName",  
  "message": "The human readable warning"  
}
```

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 500 Internal Server Error: The robot was unable to complete the request.

**POST `api/v1/continue`**

Make the robot continue to the next action in the mission if it is executing a Wait for continue action.

**Request body:** None

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 500 Internal Server Error: The robot was unable to complete the request.

### POST `api/v1/analytics`

Post an entry to the top module analytics endpoint. This endpoint is intended for collecting counting information for MiR Insights based on resending a `name` with increasing value. More than one `name` and `value` pairs can be sent at a time.

**Request headers:**

Parameter	Type	Required	Constraints
TopModuleId	string	yes	Must be the unique identifier of the connected top module.

**Request body:**

Parameter	Type	Required	Constraints
name	string	Yes	Unique for the specific count.
value	number	Yes	The current count value

```
{
  "name": "MetricName",
  "value": 1
}
```

**Responses:**

- 202 Accepted
- 400 Bad Request: The request is invalid.
- 500 Internal Server Error: The robot was unable to complete the request.

### GET `api/v1/status`

Retrieve a list of all recorded statuses the robot has been in.

#### Responses:

- 200 OK: Returns a list of robot [status-entries](#).

```
{
  "status-entries": [
    {
      "status": "Pause",
      "timestamp": "2022-09-19T09:08:12.8651699Z"
    }
  ]
}
```

- 500 Internal Server Error: The robot was unable to complete the request.

### GET `api/v1/abort`

Retrieve a list of all missions the robot has aborted since the last call time the endpoint was called. Charging and staging missions will also be shown if aborted

#### Responses:

- 200 OK: Returns a list of [abort-entries](#).

```
{
  "abort-entries": [
    {
      "timestamp": "2022-09-19T09:08:12.8651699Z"
    },
    {
      "timestamp": "2022-09-19T09:09:10.8651699Z"
    }
  ]
}
```

- 500 Internal Server Error: The robot was unable to complete the request.



## POST `api/v1/subscription`

Subscribe a webhook client to an event-type. Only one client can be subscribed to each event-type.

### Request body:

Parameter	Type	Required	Constraints
target-address	string	Yes	Uri format
event-type	string	Yes	Is one of the defined <a href="#">event-types</a>

### Responses:

- 201 Created
- 400 Bad Request: The request is invalid.
- 409 Conflict: The subscription already exists.
- 500 Internal Server Error: The robot was unable to complete the request.

## DELETE `api/v1/subscription/{event-type}`

Unsubscribe the webhook client subscribed to the selected event-type.

### Responses:

- 204 No content: No clients are subscribed to the event-type.
- 400 Bad Request: The request is invalid.
- 404 Not Found: The specified values were not found in the database.
- 500 Internal Server Error: The robot was unable to complete the request.

## 5.2.10 Schemas and types

### subscription

Parameter	Type	Required	Description
target-address	string	Yes	The web-server endpoint address the API sends a POST request each time an event of the chosen event-type is triggered.
event-type	string	Yes	<ul style="list-style-type: none"><li>"Abort": events concerning aborted missions.</li><li>"Event": events or errors from the Top Module API or the topmodule endpoint in the MiR Fleet Integration API .</li><li>"Status": events concerning order status changes.</li></ul>

### event-entries

Parameter	Type	Required	Description
name	string	Yes	The name of the event.
payload	JSON payload	Yes	Any custom JSON payload with fields that are assigned to bool, string, or number values.
timestamp	string	Yes	The time the event occurred.

Parameter	Type	Required	Description
			Written in Coordinated Universal Time (UTC) format.

### status-entries

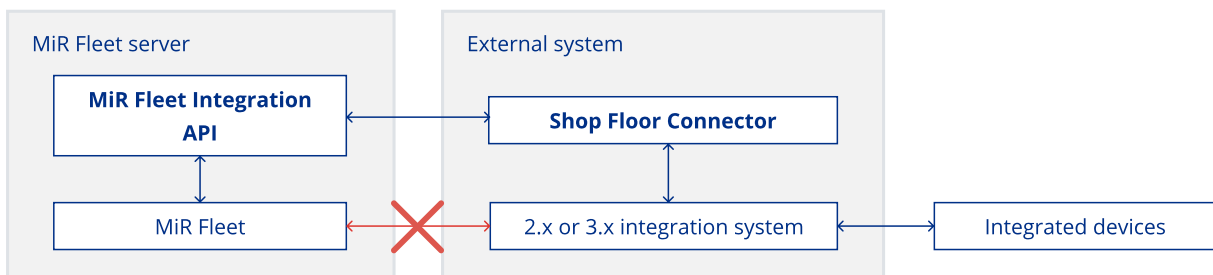
Parameter	Type	Required	Description
status	string	Yes	<ul style="list-style-type: none"><li>"EmergencyStop": An Emergency stop button on the robot has been pressed.</li><li>"ProtectiveStop": A safety function on the robot has triggered a Protective stop.</li><li>"Manual": The robot is in Manual mode and can be driven with the virtual joystick.</li><li>"Error": An error has occurred on the robot that must be resolved and cleared.</li><li>"Operate": The robot is executing or ready to execute a mission.</li><li>"Pause": The robot is Paused and will not continue a mission until it is continued.</li></ul>
timestamp	string	Yes	The time the status changed. Written in Coordinated Universal Time (UTC) format.

**abort-entries**

Parameter	Type	Required	Description
timestamp	string	Yes	The time the abort occurred. Written in Coordinated Universal Time (UTC) format.

### 5.3 Shop Floor Connector

The Shop Floor Connector is a windows application that translates software 3.x API requests to the corresponding requests in MiR Fleet Enterprise. It is intended to support sites that have an existing integration system that is compatible with the 3.x MiR Fleet API system that will no longer work directly with MiR Fleet Enterprise.



The application only supports selected endpoints. If you have an application that uses other endpoints, you must configure these parts of you integration application to use the MiR Fleet Integration API instead—see ["MiR Fleet Integration API" on page 484](#).

Contact MiR Technical Support to request the Shop Floor Connector application.

#### 5.3.1 Getting started

You must make small changes to any existing 3.x integrations to use the Shop Floor Connector. To modify your existing integration, follow these steps:

- 1 Contact MiR Technical Support to request the Shop Floor Connector application.
- 2 Install and configure the Shop Floor Connector application—see ["Installation and set up" on the next page](#).
- 3 Change the following in your API integration application:
  - The base URL for the endpoints must be updated to `<Shop_floor_connector_IP>/api/v1/legacy/`, where `<Shop_floor_connector_IP>` is the IP address of the device hosting the Shop Floor Connector application.
  - Any endpoints that are not supported—see ["Supported endpoints and data" on page 600](#)—must be removed or modified to use the corresponding MiR Fleet Integration API endpoint or subscription—see ["MiR Fleet Integration API" on page 484](#).
  - Any response values that are not supported—see ["Supported endpoints and data" on page 600](#)—must be removed or replaced with values from MiR Fleet Integration API endpoints or events—see ["MiR Fleet Integration API" on page 484](#).

- 4 Start the Shop Floor Connector application as a service.

Now when any application makes a REST call to the connector, the application will use the MiR Fleet Integration API to retrieve the supported information from MiR Fleet or schedule a mission.

- 5 Test that you can access the Swagger page for the Shop Floor Connector application from the device you are hosting the application on. The Swagger site is accessed at:  
`http://localhost:<port>/`

Where `<port>` is the port you entered in the `appsettings.json` file—see ["Installation and set up" on the next page](#).

If you want to test the connection from another device, replace `localhost` with the IP address of the device hosting the Shop Floor Connector application.

- 6 Test all of the integration applications and apply corrections until they successfully achieve the desired result.

### 5.3.2 Installation and set up

The Shop Floor Connector must be installed on a Windows device. It does not have to be the same device where MiR Fleet is installed.

To install the Shop Floor Connector and connect it to MiR Fleet, follow these steps:

- 1 Download the application—see ["Getting started" on page 596](#)—and extract the files into the directory you want the application to run in.

- 2 Open Windows Powershell.

- 3 Create a Windows service for the Shop Floor Connector using the command:

```
sc.exe create "MiR Shop Floor Connector" start= delayed-auto binpath=  
"<path_to_executable>"
```

Where `<path_to_executable>` is the path to the `.exe` file in the directory you extracted the Shop Floor Connector files to.

- 4 Create a dependency to MiR Fleet using the command:

```
sc.exe config "MiR Shop Floor Connector" depend="MiRFleet"
```

- 5 Open the `appsettings.json` file in the Shop Floor Connector directory.

**6** Configure the following lines:

- `FleetIP`: The IP address or host name of the device running MiR Fleet.
- `FleetPort`: The port MiR Fleet uses for user communication. You can see which port is used in the MiR Fleet `appsetting.json` file—see ["Transport Layer Security \(TLS\)" on page 236](#).
- `FleetApiKey`: An API key generated from MiR Fleet.
- `ListenIp`: The IP address or host name of the device running the Shop Floor Connector application. If the Shop Floor Connector is installed on the same device as MiR Fleet, this is the same value entered under `FleetIP`.
- `Port`: The port you want the Shop Floor Connector application to use.

```
{
  "Serilog": {
    "Using": [
      "Serilog.Sinks.Console"
    ],
    "MinimumLevel": {
      "Default": "Warning"
    },
    "WriteTo": [
      {
        "Name": "Console",
        "Args": {
          "outputTemplate": "{Timestamp:G} [{Level:u3}] [{SourceContext}] {Message}{NewLine:1}{Exception:1}"
        }
      }
    ]
  },
  "AllowedHosts": "*",
  "ShopFloorConnectorOptions": {
    "FleetIp": "http://127.0.0.1",
    "FleetPort": 5100,
    "FleetApiKey": "xxx",
    "ListenIp": "http://127.0.0.1",
    "Port": 5052
  },
  "Database": {
    "ConnectionString": "Data Source=sqlite.db"
  }
}
```

**7** Open the Services application in Windows, and start the MiR Shop Floor Connector service.

### 5.3.3 Supported endpoints and data

For most legacy API endpoints, the responses mimic the 3.x parameters and format. In many cases, keys are given the values `N/A` to indicate that the data is not available in MiR Fleet Enterprise software.

#### area\_events

##### PUT /api/v1/legacy/area\_events/{guid}/blocked

###### Request body:

Parameter	Type	Required	Constraints
block	bool	Yes	Set to True to lock the specific Lock zone. Use False to unlock the zone.

```
{
  "block": true
}
```

###### Response:

- 200 OK: Return a confirmation of the zone status.

```
{
  "block": true
}
```

- 400 Bad Request: The request is invalid.
- 404 Not Found: The specified values were not found in the database.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

#### maps

##### GET /api/v1/legacy/maps

Get a list of all the maps in MiR Fleet.



**Response:**

- 200 OK: Returns an array with the following information about each map on MiR Fleet:
  - `id`: The ID of the map.
  - `name`: The name of the map.

```
[
  {
    "id": "bb2d766e-caef-45a5-9c1c-0c09efb374b1",
    "name": "test",
    "url": "N/A"
  }
]
```

- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**GET /api/v1/legacy/maps/id/positions**

Get a list of all positions on the selected map.

**Table 5.1** The corresponding position type to the `type_id`.

ID	Position type	ID	Position type	ID	Position type	ID	Position type
1	Robot	2	Cart	3	Cart pickup	4	Cart left entry
5	Cart right entry	6	Shelf	7	Shelf short entry	8	MiR Charge 24V
9	MiR Charge 24V entry	10	V-marker	11	V-marker entry	12	VL-marker
13	VL-marker entry	14	L-marker	15	L-marker entry	16	Evacuation
17	Precision	18	Precision entry	19	Pallet rack	20	Pallet rack entry
21	MiR Charge 48V	22	MiR Charge 48V entry	23	Bar-marker	24	Bar-marker entry
25	Elevator	26	Elevator entry	27	Staging	28	Shelf long entry

**Response:**

- 200 OK: Returns an array of positions on the specified map with the following data for each position:
  - `guid`: The ID of the position.
  - `name`: The name of the position.
  - `type_id`: The type of position—see [Table 5.1](#).

```
[
  {
    "url": "N/A",
    "map": "N/A",
    "guid": "4a417f8f-36f5-4ce5-a8e5-74cb52876a9c",
    "name": "VL-marker",
    "type_id": 11
  },
  {
    "url": "N/A",
    "map": "N/A",
    "guid": "78502edf-b65c-46be-9fdd-0435d3684238",
    "name": "Shelf position",
    "type_id": 5
  }
]
```

- 400 Bad Request: The request is invalid.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

## missions

### GET /api/v1/legacy/missions

Get a list of defined missions and their ID.

**Response:**

- 200 OK: Returns an array of the ID and name of all missions:
  - `id`: The ID of the mission.
  - `name`: The name of the mission.

```
[
  {
    "id": "83e339a1-6121-4f5e-a1fb-5cd61d90acc5",
    "name": "simple",
    "url": "N/A"
  },
  {
    "id": "f92db91d-5100-43cc-ad8b-39a31974983e",
    "name": "test",
    "url": "N/A"
  }
]
```

- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**mission\_scheduler****GET /api/v1/legacy/mission\_scheduler**

Get a list of all missions that are scheduled, pending, or being executed in the mission scheduler.

**Response:**

- 200 OK: Returns an array of all scheduled, pending, or executing missions:
  - `url`: The endpoint of the scheduled mission.
  - `id`: The ID of the scheduled mission.
  - `state`: The state of the scheduled mission.

```
[
  {
    "state": "Waiting",
    "mission_name": "conveyor1",
    "url": "",
    "id": 4
  },
  {
    "state": "Done",
    "mission_name": "conveyor1",
    "url": "",
    "id": 3
  }
]
```

- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**POST /api/v1/legacy/mission\_scheduler**

Add a mission to the [pending missions list](#). The mission is given a medium priority.

**Request body:**

Parameter	Type	Required	Constraints
<code>mission_id</code>	string	Yes	The unique ID of the mission
<code>high_priority</code>	boolean	No	This parameter is deprecated and default to False
<code>robot_id</code>	string	No	The ID of the robot you want the mission specifically assigned to.

Parameter	Type	Required	Constraints
earliest_start_time	string	No	Defined DateTime.ToUniversalTime in format: YYYY-MM-ddThh:mm:ssZ.

```
{
  "mission_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "high_priority": true,
  "robot_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "earliest_start_time": "2024-09-20T13:23:40.412Z"
}
```

**Response:**

- 201 Created: Returns information about the scheduled mission:
  - `url`: The endpoint of the scheduled mission instance.
  - `id`: The ID of the scheduled mission instance.
  - `state`: The state of the scheduled mission.

```
[
  {
    "url": "/api/v1/legacy/mission_scheduler/3c60d68e-0727-43cb-91ba-f0d8376ede1d",
    "id": "3c60d68e-0727-43cb-91ba-f0d8376ede1d",
    "state": "Pending"
  },
  {
    "url": "/api/v1/legacy/mission_scheduler/12b1f2f0-71cb-47af-bdd7-8ec6ef541292",
    "id": "12b1f2f0-71cb-47af-bdd7-8ec6ef541292",
    "state": "Pending"
  }
]
```

- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**GET /api/v1/legacy/mission\_scheduler/id**

Get the status of the selected mission.

**Response:**

- 200 OK: Returns the status and name of the mission:
  - `id`: The ID of this exact instance of the scheduled mission.
  - `state`: The state of the mission.
  - `url`: the endpoint of the scheduled mission.
  - `mission_id`: The ID of the mission that this exact instance of the scheduled mission is based on.
  - `mission_name`: The name of the mission that was scheduled.
  - `finish_time`: The estimated time the mission is complete.
  - `order_time`: The minimum start time for the mission.
  - `robot_id`: The ID of the robot assigned to the mission.

```
{
  "id": "3c60d68e-0727-43cb-91ba-f0d8376ede1d",
  "state": "Pending",
  "url": "/api/v1/legacy/mission_scheduler/3c60d68e-0727-43cb-91ba-f0d8376ede1d",
  "mission_id": "00000000-0000-0000-0000-000000000000",
  "mission_name": "",
  "finish_time": "1970-01-01T00:00:00",
  "order_time": "1970-01-01T00:00:00",
  "robot_id": "00000000-0000-0000-0000-000000000000",
  "earliest_start_time": "N/A",
  "high_priority": "N/A",
  "priority": "N/A",
  "allowed_methods": "N/A",
  "description": "N/A",
  "mission": "N/A",
  "created_by": "N/A",
  "created_by_name": "N/A",
  "created_by_id": "N/A",
  "fleet_scheduled_guid": "N/A",
  "parameters": "N/A"
}
```

- 404 Not Found: The specified values were not found in the database.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**Delete [/api/v1/legacy/mission\\_scheduler/id](#)**

Abort or cancel a selected mission.

**Response:**

- 204 No Content: The scheduled mission was not found.
- 500 Internal Server Error: MiR Fleet was unable to complete the request.

**positions****GET /api/v1/legacy/positions**

Get the list of all positions across all maps on MiR Fleet.

**Response:**

200 OK: Returns the following data for all positions in MiR Fleet:

- `guid`: The ID of the position.
- `name`: The name of the position.
- `type_id`: The type of position—see [Table 5.1](#).

```
[
  {
    "url": "N/A",
    "map": "N/A",
    "guid": "4a417f8f-36f5-4ce5-a8e5-74cb52876a9c",
    "name": "VL-marker",
    "type_id": 11
  },
  {
    "url": "N/A",
    "map": "N/A",
    "guid": "78502edf-b65c-46be-9fdd-0435d3684238",
    "name": "Shelf position",
    "type_id": 5
  }
]
```

**GET /api/v1/legacy/positions/id**

Get the position data of the selected position.

**Response:**

200 OK: Returns the following position data:



- `map`: The endpoint of the map the position is on.
- `pos_x`: The X-coordinate of the position.
- `pos_y`: The Y-coordinate of the position.
- `map_id`: The ID of the map the position is on
- `orientation`: The orientation of the position.
- `id`: The ID of the position.
- `name`: The name of the position.
- `type_id`: The type of position—see [Table 5.1](#).

```
{
  "map": "/api/v1/legacy/maps/bb2d766e-caef-45a5-9c1c-0c09efb374b1",
  "pos_x": 11.162,
  "pos_y": 22.693,
  "map_id": "bb2d766e-caef-45a5-9c1c-0c09efb374b1",
  "orientation": 0,
  "guid": "78502edf-b65c-46be-9fdd-0435d3684238",
  "name": "Shelf position",
  "parent": "N/A",
  "type_id": 5,
  "docking_offsets": "N/A",
  "help_positions": "N/A",
  "created_by_id": "N/A",
  "created_by": "N/A",
  "parent_id": "N/A",
  "created_by_name": "N/A",
  "allowed_methods": "N/A",
  "type": "N/A"
}
```

## resources

### GET /resources/areas

Get a list of resource queue for Limit-robots zones and Lock zones.

**Response:**

200 OK: Returns an array with the following data for each Limit-robots zone and Lock zone in MiR Fleet:

- `queue`: List of robot IDs for the robot that are currently queued for the resource.
- `map_id`: The ID of the map the zone is on.
- `name`: The name of the zone.
- `blocked_for_entry`: Is true when the zone has been locked.
- `assigned`: The ID of the robot that the zone is currently assigned to.
- `url`: The endpoint path for the zone.
- `guid`: The ID of the zone.

```
[
  {
    "queue": ["3f283e8a-5f3a-4d61-969a-ca6801a46479"],
    "map_id": 0,
    "name": "Limit-robots zone (5)",
    "blocked_for_entry": false,
    "assigned": ["3f283e8a-5f3a-4d61-969a-ca6801a46479"],
    "url": "/v2.0.0/resources/areas/07604ca4-0c3b-11ef-9224-000e8e98e3b1",
    "guid": "07604ca4-0c3b-11ef-9224-000e8e98e3b1"
  },
  {
    "queue": [],
    "map_id": 0,
    "name": "Limit-robots zone (6)",
    "blocked_for_entry": false,
    "assigned": [],
    "url": "/v2.0.0/resources/areas/1db02bef-0c3b-11ef-9224-000e8e98e3b1",
    "guid": "1db02bef-0c3b-11ef-9224-000e8e98e3b1"
  }
]
```

**GET /resources/areas/id**

Get the current resource queue data for the selected zone.

**Response:**

200 OK: Returns the following data:

- `assigned`: The ID of the robot that the zone is currently assigned to.
- `map_id`: The ID of the map the zone is on.
- `name`: The name of the zone.
- `blocked_for_entry`: Is `true` when the zone has been locked in the API.
- `queue`: List of robot IDs for the robots that are currently in queue for the resource.
- `guid`: The ID of the zone.

```
{
  "max_robots": "N/A",
  "assigned": [],
  "map_id": 0,
  "name": "Limit-robots zone (6)",
  "blocked_for_entry": false,
  "queue": [],
  "allowed_methods": [
    "GET"
  ],
  "guid": "1db02bef-0c3b-11ef-9224-000e8e98e3b1"
}
```

### GET /resources/positions

Get a list of resource queue for all positions and markers.

#### Response:

200 OK: Returns an array with the following data for each position and marker in MiR Fleet:

- `queue`: List of robot IDs for the robot that are currently queued for the resource.
- `map_id`: The ID of the map the position is on.
- `name`: The name of the position.
- `url`: The endpoint path for the position.
- `assigned`: The ID of the robot that the position is currently assigned to.
- `guid`: The ID of the position.

```
[
  {
    "queue": ["137cf049-d083-4e02-bc1f-776cd04e45c9", "5ea872ba-de68-442b-973d-
```

```

01da407ddeb8"],
  "map_id": 0,
  "name": "ROEQ_Docking 12 rack",
  "url": "/v2.0.0/resources/positions/04abaf56-2f55-4ea0-be20-f2f4e4d33e9b",
  "assigned": ["137cf049-d083-4e02-bc1f-776cd04e45c9"],
  "guid": "04abaf56-2f55-4ea0-be20-f2f4e4d33e9b"
},
{
  "queue": [],
  "map_id": 0,
  "name": "S4",
  "url": ""/v2.0.0/resources/positions/0668d6e6-06f4-11ef-ada3-000e8e98dddb"",
  "assigned": [],
  "guid": "0668d6e6-06f4-11ef-ada3-000e8e98dddb"
}
]

```

### GET /resources/positions/id

Get the current resource queue data for the selected position.

#### Response:

200 OK: Returns the following data:

- `queue`: List of robot IDs for the robots that are currently in queue for the resource.
- `name`: The name of the position.
- `assigned`: The ID of the robot that the position is currently assigned to.
- `map_id`: The ID of the map the position is on.
- `guid`: The ID of the position.

```

{
  "max_robots": 1,
  "queue": [
    73
  ],
  "allowed_methods": [
    "GET"
  ],
  "name": "24V #1",
  "assigned": [
    73
  ],
  "map_id": 0,
  "guid": "5430de74-0919-11ef-9ace-000e8ea1a127"
}

```

## robots

### GET /api/v1/legacy/robots

Get a list of all the robots connected to MiR Fleet.

#### Response:

200 OK: Returns an array with the following data for each robot connected to MiR Fleet:

- `url`: The URL to access the robot interface.
- `id`: The ID of the robot.

```
[
  {
    "url": "",
    "id": "12747ecc-62aa-4689-8a59-05d8a96fea10"
  },
  {
    "url": "",
    "id": "34547cef-32aw-4523-2a12-35d8a96fqw12"
  }
]
```

### GET /api/v1/legacy/robots/id

Get runtime data about the selected robot. The `state_text` key uses the states supported in MiR Fleet Enterprise software. The following table maps the MiR Fleet Enterprise states to 3.x states.

**Table 5.2** States mapped from MiR Fleet Enterprise to the 3.x equivalent

MiR Fleet Enterprise state	3.x state
Idle	Ready
Operatinoal	Executing
ManualControl	ManualControl

MiR Fleet Enterprise state	3.x state
Paused	Pause
PStop	EmergencyStop
EStop	EmergencyStop
BrakeRelease	EmergencyStop
RestartRequired	EmergencyStop
Starting	Starting
ShuttingDown	ShuttingDown
KeyManual	ManualControl
KeyIdle	EmergencyStop
ExternalCableChargerConnected	None
Error	Error

**Response:**

200 OK: Returns the latest robot runtime data:

- `robot_name`: The name of the robot.
- `robot_model`: The base robot model.
- `software_version`: The software version of the robot.

- `serial_number`: The serial number of the robot.
- `ip`: The IP address of the robot.
- `id`: The ID number of the robot.
- `status`: a `GetStatus` object with the following data:
  - `name`: The name of the robot
  - `moved`: The total distance in meters that the robot moved.
  - `uptime`: The total time the robot has been operating.
  - `battery_percentage`: The robot's current battery percentage.
  - `state_text`: The robot's state—see [Table 5.2](#).
  - `battery_time_remaining`: The estimated length of time to robot can operate before running out of power.
  - `x`: The X-coordinate of the robot's current position on its active map.
  - `y`: The Y-coordinate of the robot's current position on its active map.
  - `orientation`: The robot's current orientation.

```
{
  "status": {
    "joystick_low_speed_mode_enabled": "N/A",
    "mode_id": "N/A",
    "moved": 3000,
    "mission_queue_id": "N/A",
    "header": {
      "stamp": {
        "secs": "N/A",
        "nsecs": "N/A"
      },
      "frame_id": "N/A",
      "seq": "N/A"
    },
    "robot_name": "Ronix2",
    "uptime": 2000,
    "errors": "N/A",
    "unloaded_map_changes": "N/A",
    "distance_to_next_target": "N/A",
    "battery_voltage": "N/A",
    "mode_key_state": "N/A",
    "battery_percentage": 70,
    "map_id": "N/A",
    "software_version": "4.8",
    "safety_system_muted": "N/A",
    "mission_text": "N/A",
    "velocity": {
      "linear": "N/A",
      "angular": "N/A"
    },
    "state_text": "Paused",
  }
}
```

```
"position": {
  "y": 0,
  "x": 0,
  "orientation": 0
},
"footprint": "N/A",
"user_prompt": {
  "question": "N/A",
  "has_request": "N/A",
  "timeout": {
    "secs": "N/A",
    "nsecs": "N/A"
  },
  "user_group": "N/A",
  "guid": "N/A",
  "options": "N/A"
},
"mode_text": "N/A",
"hook_status": {
  "trolley": {
    "width": "N/A",
    "length": "N/A",
    "offset_locked_wheels": "N/A",
    "id": "N/A",
    "height": "N/A"
  },
  "available": "N/A",
  "trolley_attached": "N/A"
},
"session_id": "N/A",
"joystick_web_session_id": "N/A",
"hook_data": {
  "angle": {
    "timestamp": {
      "secs": "N/A",
      "nsecs": "N/A"
    },
    "angle": "N/A"
  },
  "height_state": "N/A",
  "brake_state": "N/A",
  "height": "N/A",
  "length": "N/A",
  "gripper_state": "N/A"
},
"battery_time_remaining": 2000,
"state_id": "N/A"
},
"fleet_state_text": "N/A",
"robot_model": "MiR250",
"allowed_methods": "N/A",
"description": "N/A",
"ip": "ip",
"factory_reset_needed": "N/A",
"created_by_id": "N/A",
```



```

"fleet_state": "N/A",
"created_by": "N/A",
"operation_mode_info ": {
  "synchronization_enabled": "N/A",
  "mission_scheduling_enabled": "N/A",
  "resource_management_enabled": "N/A",
  "initial_synchronization_enabled": "N/A",
  "collision_avoidance_enabled": "N/A",
  "mode_name": "N/A",
  "charging_staging_enabled": "N/A",
  "communicator_enabled": "N/A"
},
"created_by_name": "N/A",
"state_id": "N/A",
"active": "N/A",
"serial_number": "serial",
"robot_group_id": "N/A",
"id": "7c9e6679-7425-40de-944b-e07fc1f90aa4",
"name": "Ronix2"
}

```

## zones

### GET /api/v1/legacy/zones

Get a list of all the zones on all maps. Only Integration zones and Lock zone include type ID to identify the zone type.

#### Response:

200 OK: Returns an array of all the zones in MiR Fleet with the following data for each:

- `guid`: The ID of the zone.
- `name`: The name of the zone.
- `map_id`: The ID of the map the zone is on.
- `type_id`: The type of zone—see [Table 5.3](#).

**Table 5.3** The corresponding zone type to the `type_id`

ID	Zone type	ID	Zone type
101	Planner zone	102	Sound and light zone

ID	Zone type	ID	Zone type
103	Limit-robots zone	104	Speed zone
106	Evacuation zone	201	Floor
202	Wall	203	Directional zone
204	Preferred zone	205	Unpreferred zone
206	Forbidden zone	208	Access zone
301	Integration zone	302	Lock zone

```
[
  {
    "url": "N/A",
    "map": "N/A",
    "guid": "5fb35025-c6f1-413b-9dfd-c62983bfea3b",
    "name": "Lock zone",
    "type_id": 302
  },
  {
    "url": "N/A",
    "map": "N/A",
    "guid": "81fa1650-a030-45c1-ad63-4c7ed588b5ba",
    "name": "Integration zone",
    "type_id": 301
  },
  {
    "url": "N/A",
    "map": "N/A",
    "guid": "b255d592-4b72-44e3-a7e4-37932cc4092b",
    "name": "Forbidden zone",
    "type_id": 0
  }
]
```

### 5.3.4 Logging

You can customize how Shop Floor Connector data is logged and where the data is logged to.

You can change the logging configuration in the file `appsetting.json`—see ["Installation and set up" on page 598](#). This file is located in the directory where the Shop Floor Connector application is installed.

The standard configuration to make the Shop Floor Connector application log files to the console and to a `log.txt` file is:

```
{
  "Serilog": {
    "Using": [
      "Serilog.Sinks.Console",
      "Serilog.Sinks.File"
    ],
    "MinimumLevel": {
      "Default": "Information"
    },
    "WriteTo": [
      {
        "Name": "Console",
        "Args": {
          "outputTemplate": "{Timestamp:G} [{Level:u3}] [{SourceContext}] {Message}"
        }
      },
      {
        "Name": "File",
        "Args": {
          "outputTemplate": "{Timestamp:G} [{Level:u3}] [{SourceContext}] {Message}"
        },
        "path": "Logs/log.txt"
      }
    ]
  }
}
```

The shop floor connector logs are never rotated automatically. The logs are intended for troubleshooting and commissioning purposes, and it is expected that you disable logging after the system is running as intended.

If you chose to continue logging data during operations, you must implement a protocol to clear the log after a given size or date to avoid running out of disk space. If you delete, rename, or move the audit log file, a new one is automatically generated.

### 5.3.5 Version overview

Release notes and known issues are found on [MiR Support Portal](#) for users with access.

The following table provides an overview of the available Shop Floor Connector versions and which MiR Fleet Enterprise versions they are compatible with.

	SFC 1.1.0	SFC 1.1.1	SFC 1.1.2
MFE 1.0.0	×		
MFE 1.1.0		×	×

## 5.4 Robot APIs

You can use endpoints in the robot software to request site information and configure robot settings. You are limited to the same functionality that is available in the robot interface.

The robot API is a poll-based system. the

You can find documentation for the supported API requests for your robot at: [https://supportportal.mobile-industrial-robots.com/support-files/manuals/HTML/en/sw4\\_documentation/Content/api/robot\\_api.htm](https://supportportal.mobile-industrial-robots.com/support-files/manuals/HTML/en/sw4_documentation/Content/api/robot_api.htm)

## 6. Maintenance

To ensure that MiR Fleet is running optimally, it is important to keep your MiR products updated.

### 6.1 Upgrade

To keep your MiR product software up-to-date, see ["Upgrade software" on page 181](#).

### 6.2 Restart

In general, it is not necessary to restart MiR robot or MiR Fleet. If robots or MiR Fleet often report memory leak issues, you can reduce the risk of these issues by restarting the affected product regularly.

For information about how the robots and MiR Fleet behave after and during a restart, see ["Robot states and fleet behavior" on page 91](#).

To restart MiR Fleet, you must restart the MiR Fleet service running on your host server.

To restart a MiR robot, turn off the robot by pressing the Power button for three seconds, wait for the indicator lights to turn off, then press the Power button for three seconds to turn on the robot.

### 6.3 Data backups and site files

You can manually create backups of your data by:

- Creating a backup of the MiR Fleet Microsoft SQL database.
- Exporting a site file from the ["System settings" on page 106](#).

The following data backups are also generated automatically:

- A site file of the current site is always generated when you import and overwrite with a new site.
- An application backup of MiR Fleet is generated when you update MiR Fleet in case the update fails. The backup is removed once the update finished successfully.

## 6.4 Error logs

Error logs are useful for MiR Technical Support to help debug any issues you are experiencing with your MiR system—see ["Logging" on page 40](#).

When you generate an error log, MiR Fleet saves a copy of the current site and a record of recent events.

Generate error logs regularly. MiR Technical Support can use error logs to restore some of you site in case you need to recover your site.

Error logs are generated:

- When the system reports an error.
- When a robot executes a Create log action.
- When you select **Generate error log** under **Monitoring > Error log**.

## 6.5 Robot hardware

For more information about MiR robot maintenance, see:

- For MiR100 and MiR200, see the Maintenance section in the user guide.
- For MiR250, see the [MiR250 Maintenance Guide](#).
- For MiR500 and MiR1000, see the Maintenance section in the user guide.
- For MiR600 and MiR1350, see the [MiR600 and MiR1350 Maintenance Guide](#).

You can find these guides on [MiR Support Portal](#).

Set up a maintenance plan for preventative maintenance and scheduled maintenance. Evaluate the plan during each maintenance interval and adjust the time intervals if necessary.

Keep items you need for maintenance collected and ready for use, including stock of critical parts and wear and tear parts.

## 7. Where to find more information

For online courses to strengthen your understanding of MiR products, go to [Online training](#).

If you are looking for more documentation about all MiR products, go to [MiR Support Portal](#) where we have the following resources:

### 7.1 Documentation

- **Integrator Manuals** provide all the information you need to set up and prepare MiR robots for the commissioning process. It comes in print in the box with the robots. Integrator Manuals are available in multiple languages. These guides are for PCM (partly completed machinery) robots.
- **Quick starts** describe how you start operating MiR robots quickly. It comes in print in the box with the robots. Quick starts are available in multiple languages. These guides are for CE robots.
- **User guides** provide all the information you need to operate and maintain MiR products and how to set up and use top modules and accessories, such as charging stations, hooks, shelf lifts, and pallet lifts. User guides are available in multiple languages. These guides are for CE robots.
- **Risk assessment guide** describes how to conduct a risk assessment and provides some risk assessed use cases.
- **MiR Fleet Enterprise Documentation** provides examples and guidelines to commission your robot system successfully and contains descriptions of all the elements of the robot interface and MiR Fleet interface. The documentation also provides important information and instructions to increase the cybersecurity of your MiR product and REST API references for MiR robots, MiR Hooks, and MiR Fleet.
- **Space requirements** provide helpful information you can use when commissioning or operating your robot.
- **MiR Network and Wi-Fi guide** specifies the performance requirements of your network and how you must configure it for MiR robots and MiR Fleet to operate successfully.
- **Cybersecurity guide** provides important information and instructions to increase the cybersecurity of your MiR product.
- **How-to guides** are short guides providing instruction for maintenance, replacement, commissioning, and other tasks related to MiR products.

- **Troubleshooting guides** can help you determine the cause of an issue you are experiencing with your MiR product and how to resolve it.
- **Release notes** of new products and hardware updates that describe what has been changed and why.
- **Service notes** notify of issues identified in MiR products and changes that are applied.
- **Spare parts overview** list all spare parts you can order for robots.
- **Additional products** list all accessories you can order for robots.
- **Warranty** describes the MiR standard warranty agreement.
- **Certificates and declarations** for MiR products that prove compliance with standards.
- **Technical guides** provide in-depth information about how MiR products work.

## 7.2 Models and drawings

- **Wiring diagrams** are graphic representations of how the components in MiR robots are wired.
- **CAD files** of the robots that are made to scale can be used to help determine the dimensions of the robot or for illustrative purposes.

## 7.3 Resources

- **MiR Insights** is a tool you can use to analyze how well your robots or fleet are performing. MiR Insights requires a paid license. MiR Insights runs continuously alongside MiR Fleet to give real-time data on several metrics, but can also be used in Logs mode. Logs mode can analyze error logs and provide heatmaps for Wi-Fi signal strength and localization.
- **AprilTag** collection can be used instead of generating your own AprilTags.
- **Space calculator** determines the approximate amount of space your MiR robot will need to operate depending on the size of its footprint.
- **Community** is a forum of MiR users with a collection of questions, recommendations, webinars and other community driven material.
- **Marketing and brand portal** is a collection of our graphical elements where you can download color schemes, rendered images of the robots, and icons.