

---

# **Using Avantec HVAC devices with ThingsBoard**

***Release 2.3.4***

**Avantec Manufacturing Limited**

**Nov 08, 2023**



# FIRST STEPS

<b>1</b>	<b>First steps</b>	<b>3</b>
1.1	Why Avantec + ThingsBoard? . . . . .	3
1.1.1	Why Avantec? . . . . .	3
1.1.2	Why ThingsBoard? . . . . .	4
1.1.3	What is Avantec + ThingsBoard? . . . . .	5
1.2	Get Started . . . . .	7
1.2.1	Introduction . . . . .	7
1.2.2	Prerequisites . . . . .	7
1.2.3	Step 1. Tenant Login . . . . .	8
1.2.4	Step 2. Import Avantec Widgets . . . . .	9
1.2.5	Step 3. Import device profile . . . . .	9
1.2.6	Step 4. Import Dashboards . . . . .	9
1.2.7	Step 5. Provision device . . . . .	9
1.2.8	Step 6. Connect device . . . . .	10
1.2.9	Step 7. Assign Device and Dashboards to Customer . . . . .	10
1.2.10	Step 8. Open Dashboards . . . . .	17
1.2.11	Next Steps . . . . .	17
1.2.12	See also . . . . .	17
1.2.13	Your feedback . . . . .	17
<b>2</b>	<b>ThingsBoard</b>	<b>19</b>
2.1	Overview . . . . .	19
2.1.1	Introduction . . . . .	19
2.1.2	Features . . . . .	19
2.1.3	Getting Started Guides . . . . .	20
2.2	Over-the-air firmware updates . . . . .	21
2.2.1	Overview . . . . .	21
2.2.2	Firmware update monitoring dashboard . . . . .	22
2.2.3	Provision OTA package to ThingsBoard repository . . . . .	24
2.2.4	Assign OTA package to device profile . . . . .	28
2.2.5	Assign OTA package to device . . . . .	30
2.2.6	Update process . . . . .	31
2.2.7	Configuration . . . . .	33
2.3	Working with IoT Dashboards . . . . .	33
2.3.1	How to customize . . . . .	34
2.3.2	Next steps . . . . .	34
2.4	Getting Started with Rule Engine . . . . .	35
2.4.1	What is ThingsBoard Rule Engine? . . . . .	35
2.4.2	Typical Use Cases . . . . .	35
2.5	White-labeling . . . . .	35

2.5.1	Feature	36
2.5.2	Next steps	36
2.6	Installation options	36
2.7	Mobile application	36
2.7.1	ThingsBoard mobile application	36
2.7.2	ThingsBoard PE mobile application	37
2.8	ThingsBoard MQTT Device API	37
2.8.1	Introduction	37
2.8.2	Getting started	38
2.8.3	Key-value format	38
2.8.4	Telemetry upload API	39
2.8.5	Attributes API	41
2.8.6	PRC API	44
2.8.7	Claiming API	46
2.8.8	Firmware API	47
2.8.9	Device MQTT Topic	47
<b>3</b>	<b>Avantec Extension</b>	<b>49</b>
3.1	Avantec widgets	49
3.1.1	Widget list	49
3.1.2	Import Avantec Widgets	55
3.1.3	Update Avantec Widgets	57
3.2	Demo Dashboards	57
3.2.1	TA652FC-W	57
3.2.2	TA652FH-W	58
3.2.3	TA692FC-L-5	58
3.2.4	Other Dashboards	58
<b>4</b>	<b>TA652FC-W Wi-Fi Thermostat</b>	<b>59</b>
4.1	TA652FC-W – 2 pipe Fan Coil Wi-Fi Thermostat	59
4.1.1	Introduction	59
4.1.2	Feature List	60
4.1.3	Wiring	60
4.1.4	Mounting	61
4.1.5	Dimension in mm:	62
4.1.6	LCD Interface	63
4.2	Add TA652FC-W to ThingsBoard	68
4.2.1	Step 1. Tenant Login	68
4.2.2	Step 2. Import Detail Dashboard of TA652FC-W	68
4.2.3	Step 3. Import List Dashboard of TA652FC-W	69
4.2.4	Step 4. Provision TA652FC-W device	69
4.2.5	Step 5. Connect TA652FC-W device	72
4.2.6	Step 6. Assign Device and Dashboards to Customer	73
4.2.7	Step 7. Open Dashboards of TA652FC-W	76
4.2.8	Your feedback	77
4.3	Connect TA652FC-W to ThingsBoard	77
4.3.1	Prerequisites. Clear Wi-Fi Configuration	78
4.3.2	Step 1. Get Access-Token	79
4.3.3	Step 2. Power On	79
4.3.4	Step 3. Configure	79
4.3.5	Step 4. Check	82
4.3.6	Troubleshooting	83
4.4	TA652FC-W Thermostat – Demo device profile usage	83
4.4.1	Import device profile	83



4.4.2	Modify device profile's mobile dashboard	84
4.4.3	Clear device profile's mobile dashboard	86
4.5	TA652FC-W Demo Dashboards Usage	87
4.5.1	Overview	87
4.5.2	TA652FC-W Thermostat List	87
4.5.3	TA652FC-W Thermostat (For Mobile App)	93
4.6	TA652FC-W MQTT API	101
4.6.1	Overview	101
4.6.2	Features	101
4.6.3	MQTT Special	102
4.6.4	Flow Chart	102
4.6.5	Telemetry (Time-series data)	114
4.6.6	Shared attributes	115
4.6.7	Client-side attributes	116
4.6.8	Server-side RPC	125
<b>5</b>	<b>TA652FH-W Wi-Fi Thermostat</b>	<b>131</b>
5.1	TA652FH-W — Floor Heating Wi-Fi Thermostat	131
5.1.1	Introduction	131
5.1.2	Feature List	132
5.1.3	Wiring	132
5.1.4	Mounting	133
5.1.5	Dimension in mm:	134
5.1.6	LCD Interface	135
5.2	Add TA652FH-W to ThingsBoard	140
5.2.1	Step 1. Tenant Login	140
5.2.2	Step 2. Import Detail Dashboard of TA652FH-W	141
5.2.3	Step 3. Import List Dashboard of TA652FH-W	141
5.2.4	Step 4. Provision TA652FH-W device	141
5.2.5	Step 5. Connect TA652FH-W device	144
5.2.6	Step 6. Assign Device and Dashboards to Customer	146
5.2.7	Step 7. Open Dashboards of TA652FH-W	148
5.2.8	Your feedback	150
5.3	Connect TA652FH-W to ThingsBoard	150
5.4	TA652FH-W Thermostat – Demo Device Profile Usage	150
5.4.1	Import device profile	150
5.4.2	Modify device profile's mobile dashboard	151
5.4.3	Clear device profile's mobile dashboard	152
5.5	TA652FH-W Demo Dashboards Usage	153
5.5.1	Overview	153
5.5.2	TA652FH-W Thermostat List	154
5.5.3	TA652FH-W Thermostat (For Mobile App)	159
5.5.4	Office center - TA652FH-W Thermostats	167
5.6	TA652FH-W MQTT API	171
<b>6</b>	<b>TA692FC-L-5 LoRaWAN Thermostat</b>	<b>173</b>
6.1	TA692FC-L – FCU Thermostat Series	173
6.1.1	Features	173
6.1.2	Technical Specification	174
6.1.3	Order Code	174
6.1.4	Dimensions / Outline	174
6.1.5	Product pictures	176
6.1.6	Wiring Example for TA692FC-L-1	177
6.1.7	Terminal Labels on TA692FC-L-1	178

6.1.8	Wiring Example for TA692FC-L-2	178
6.1.9	Terminal Labels on TA692FC-L-2	179
6.1.10	Wiring Example for TA692FC-L-3	180
6.1.11	Terminal Labels on TA692FC-L-3	181
6.1.12	Wiring Example for TA692FC-L-4	181
6.1.13	Terminal Labels on TA692FC-L-4	182
6.1.14	Wiring Example for TA692FC-L-5	183
6.1.15	Terminal Labels on TA692FC-L-5	184
6.1.16	Output diagrams	184
6.1.17	LCD Display Content	186
6.1.18	Internal Parameter Menu in TA692FC-L-5	188
6.1.19	Advanced Parameter Menu in TA692FC-L-5	188
6.2	Add TA692FC-L-5 to ThingsBoard	189
6.2.1	Introduction	189
6.2.2	Prerequisites	189
6.2.3	Step 1. MTCAP configuration	190
6.2.4	Step 2. ChirpStack configuration	190
6.2.5	Step 3. Integrating ChirpStack with ThingsBoard PE	190
6.3	MultiTech Conduit® MTCAP-868-041A	211
6.3.1	MTCAP Series	211
6.3.2	MTCAP-868-041A	211
6.4	ChirpStack v3	220
6.4.1	Quick start Amazon AWS	220
6.4.2	Connecting a gateway	227
6.4.3	Connecting a device	235
6.5	TA692FC-L-5-868 Thermostat – Demo device profile usage	246
6.5.1	Import device profile	246
6.5.2	Modify device profile's mobile dashboard	247
6.5.3	Clear device profile's mobile dashboard	248
6.6	TA692FC-L-5 Demo Dashboards Usage	249
6.6.1	Overview	249
6.6.2	TA692FC-L-5 Thermostat List	250
6.6.3	TA692FC-L-5 Thermostat (For Mobile App)	255
6.7	TA692FC-L-5 LoRaWAN API	259
6.7.1	Overview	259
6.7.2	Payload format in LoRa packet used by TA692FC-L-5	259
<b>7</b>	<b>Release Notes</b>	<b>263</b>
7.1	Release Notes	263
7.1.1	v2.3.4 (Nov 8, 2023)	263
7.1.2	v2.3.3 (Sep 25, 2023)	263
7.1.3	v2.3.2 (July 5, 2023)	263
7.1.4	v2.3.1 (July 5, 2023)	264
7.1.5	v2.3 (June 20, 2023)	264
7.1.6	v2.2 (June 1, 2023)	264
7.1.7	v2.1 (Apr 18, 2023)	265
7.1.8	v1.0 (Jul 24, 2020 / Dec 20, 2022)	265
7.2	Upgrade instructions	265
<b>8</b>	<b>Avantec and the project</b>	<b>267</b>
8.1	About us	267
8.2	Copyrights and Licenses	267
8.2.1	Copyrights	267
8.2.2	ThingsBoard License	268

Avantec provides some networking HVAC solutions. A series of networked HVAC devices in these solutions are connected to ThingsBoard IoT platform through MQTT protocol.



## FIRST STEPS

Are you new to Avantec Thermostats or ThingsBoard IoT platform? Learn about them to help you create fantastic project.

- *Why Avantec + ThingsBoard?*
- *Get Started*

### 1.1 Why Avantec + ThingsBoard?

Avantec provides some networking HVAC solutions. A series of networked HVAC devices in these solutions are connected to ThingsBoard IoT platform through MQTT protocol.

#### 1.1.1 Why Avantec?

**Avantec Manufacturing Limited** was founded in 1983. Avantec has been a trusted name in the original equipment and design manufacturing (OEM/ODM) industry. In early years, products manufactured and exported by Avantec were highly diversified, ranging from gaming, electronics dictionary, handheld timers, to corded and cordless telephones.

Over the past decade, we have evolved into product design and serve customers around the world with our branded products. Three major product lines are (1) Telecommunication, (2) Education and (3) HVAC (Heating, Ventilating, Air Conditioning and refrigeration).

Avantec's strength lies in delivering high-quality engineering solutions. We provide one-stop-shop service from product design, structural/mechanical engineering design, tooling maintenance, electronics circuit design, firmware and software coding, to quality control and optimization.

#### HVAC

- **Device Type:**
  - Thermostat
  - Humidistat
  - Sensor for smart city
  - DDC (Direct Digital Controller)
  - VAV (Variable Air Volume) Controller
  - ...
- **Networking:**

- Classic
- Modbus
- BACnet
- RF 433/868/915
- Wi-Fi
- LoRaWAN
- ...

### 1.1.2 Why ThingsBoard?

#### ThingsBoard IoT platform

ThingsBoard is an open-source IoT platform that enables rapid development, management and scaling of IoT projects. Their goal is to provide the out-of-the-box IoT cloud or on-premises solution that will enable server-side infrastructure for your IoT applications.

- **100% Open-source**  
ThingsBoard is licensed under Apache License 2.0, so **you can use any it in your commercial products for free**. You can even host it as a SaaS or PaaS solution.
- **Telemetry Data Collection**  
Collect and store telemetry data in reliable way, surviving network and hardware failures. Access collected data using customizable web dashboards or server-side APIs.
- **IoT Rule Engine**  
ThingsBoard allows you to create complex Rule Chains to process data from your devices and match your application specific use cases.
- **Data Visualization**  
Provides 30+ configurable widgets out-of-the-box and ability to create your own widgets using built-in editor. Built-in line-charts, digital and analog gauges, maps and much more.
- **Multi-tenancy**  
Support multi-tenant installations out-of-the-box. Single tenant may have multiple tenant administrators and millions of devices and customers.
- **ThingsBoard Community Edition and ThingsBoard Professional Edition**  
ThingsBoard includes ThingsBoard CE (Community Edition) and ThingsBoard PE (Professional Edition).

---

**Note:** When we developed TA652FC-W and TA652FH-W Thermostats, we used ThingsBoard CE.

---

## ThingsBoard IoT Gateway

The [ThingsBoard IoT Gateway](#) is an open-source solution that allows you to integrate devices connected to legacy and third-party systems with ThingsBoard.

## ThingsBoard Mobile Application

The [ThingsBoard Mobile Application](#) is an open-source project based on Flutter. It allows you to build your own IoT mobile application with minimum coding efforts. It is powered by ThingsBoard CE IoT platform.

The [ThingsBoard PE Mobile Application](#) is an open-source project based on Flutter. It allows you to build your own advanced IoT mobile application with minimum coding efforts. It is powered by ThingsBoard PE IoT platform.

### 1.1.3 What is Avantec + ThingsBoard?

#### Avantec Extension

We provide *Avantec widgets* and *Demo Dashboards* based on ThingsBoard IoT platform for demonstration.

Of course, you can also customize your own Web UI and Mobile Application based on them.

#### Avantec Devices

##### TA652FC-W

- 2 pipe Fan Coil Wi-Fi Thermostat.
- Firmware ID: TA652FC-W-TB.



##### TA652FH-W

- Floor Heating Wi-Fi Thermostat.
- Firmware ID: TA652FH-W-TB.



#### TA692FC-L-5

- LoRaWAN thermostat for fan coil units.



#### HA652-W

*Coming soon...*

#### TA640FC-W

*Coming soon...*

#### TA670-W

*Coming soon...*

#### DL10-W

*Coming soon...*



## CDT022-W

*Coming soon...*

---

**Tip:** Firmware ID - a hardware device may have several firmwares, which are respectively connected to different software platforms. Firmware ID are used to distinguish these firmwares.

---

## 1.2 Get Started

---

### Tip:

- This section applies when no Avantec HVAC device is added to the ThingBoard server.
  - **If you add some Avantec HVAC devices to ThingsBoard Server again, please refer to the instructions of each device.**
    - *Add TA652FC-W to ThingsBoard*
    - *Add TA652FH-W to ThingsBoard*
- 

Reprinted this article from [Getting Started with ThingsBoard](#), slightly modified.

### 1.2.1 Introduction

The goal of this tutorial is to demonstrate the basic usage of the most popular Avantec HVAC device and ThingsBoard features. You will learn how to:

- Connect devices to ThingsBoard;
- Import real-time end-user dashboards.

We will connect and visualize data from a Avantec HVAC device to keep it simple.

Refer to [Getting Started with ThingsBoard](#) to get support for the following features:

- Define thresholds and trigger alarms;
- Push notifications about new alarms over email, SMS or other systems.

### 1.2.2 Prerequisites

You will need to have ThingsBoard server up and running.

- The easiest way is to use [Live Demo server](#).
- Or [ThingsBoard Cloud](#).
- **The alternative option is to install ThingsBoard using *Installation options*.**
  - **Windows** users should follow this [guide](#).
  - **Linux** users that have docker installed should execute the following commands:

```
mkdir -p ~/.mytb-data && sudo chown -R 799:799 ~/.mytb-data
mkdir -p ~/.mytb-logs && sudo chown -R 799:799 ~/.mytb-logs
docker run -it -p 9090:9090 -p 7070:7070 -p 1883:1883 -p 5683-5688:5683-5688/
  ↪udp -v ~/.mytb-data:/data \
-v ~/.mytb-logs:/var/log/thingsboard --name mytb --restart always thingsboard/
  ↪tb-postgres
```

These commands install ThingsBoard and load demo data and accounts. ThingsBoard UI will be available using the URL: <http://localhost:8080> . You may use username **tenant@thingsboard.org** and password **tenant**. More info about [demo accounts](#) is available.

### Some important parameters

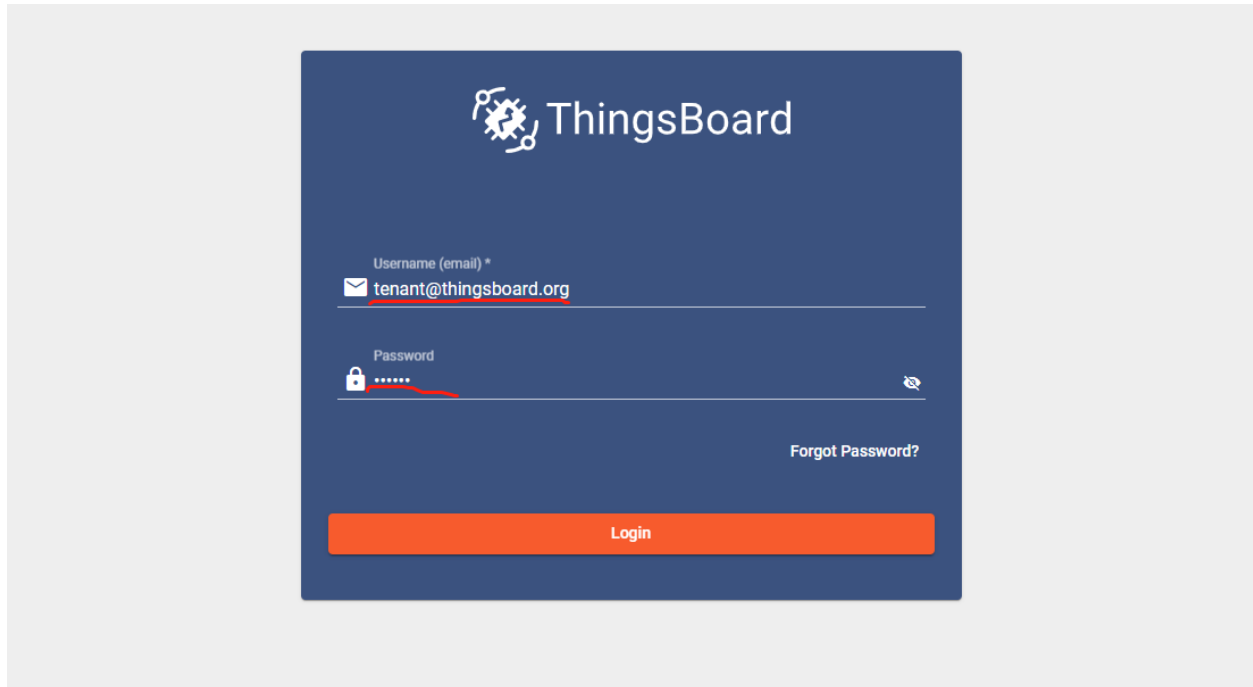
Please remember the following important parameters, which will be used frequently in the following work:

Table 1: Some important parameters

ThingsBoard server	Web URI	Default Tenant Account	MQTT URI / Cloud Host
Live Demo server	<a href="https://demo.thingsboard.io">https://demo.thingsboard.io</a>		mqtt://demo.thingsboard.io
ThingsBoard Cloud (Subscription plans)	<a href="https://thingsboard.cloud">https://thingsboard.cloud</a>		mqtt://mqtt.thingsboard.cloud
Installation	local: <a href="http://localhost:8080">http://localhost:8080</a> remote: <a href="http://your_server_ip:8080">http://your_server_ip:8080</a>	username: <a href="mailto:tenant@thingsboard.org">tenant@thingsboard.org</a> password: tenant See <a href="#">demo accounts</a>	mqtt://your_server_ip

### 1.2.3 Step 1. Tenant Login

- Open ThingsBoard Web UI in browser, e.g. <http://localhost:8080>
- Tenant Administrator login ThingsBoard.



Tenant default username and password, refer to *Some important parameters*.

## 1.2.4 Step 2. Import Avantec Widgets

- See *Import Avantec Widgets*.

## 1.2.5 Step 3. Import device profile

- See *Import Device Profile of TA652FC-W Thermostat*, or
- See *Import Device Profile of TA652FH-W Thermostat*.

## 1.2.6 Step 4. Import Dashboards

- See *Import TA652FC-W Detail Dashboard* and *Import TA652FC-W List Dashboard*, or
- See *Import TA652FH-W Detail Dashboard* and *Import TA652FH-W List Dashboard*.

## 1.2.7 Step 5. Provision device

- See *Step 4. Provision TA652FC-W device*, or
- See *Step 4. Provision TA652FH-W device*.

### 1.2.8 Step 6. Connect device

- See *Step 5. Connect TA652FC-W device*, or
- See *Step 5. Connect TA652FH-W device*.

### 1.2.9 Step 7. Assign Device and Dashboards to Customer

One of the most important ThingsBoard features is the ability to assign Dashboards to Customers. You may assign different devices to different customers. Then, you may create a Dashboard(s) and assign it to multiple customers. Each customer user will see his own devices and will not be able to see devices or any other data that belongs to a different customer.

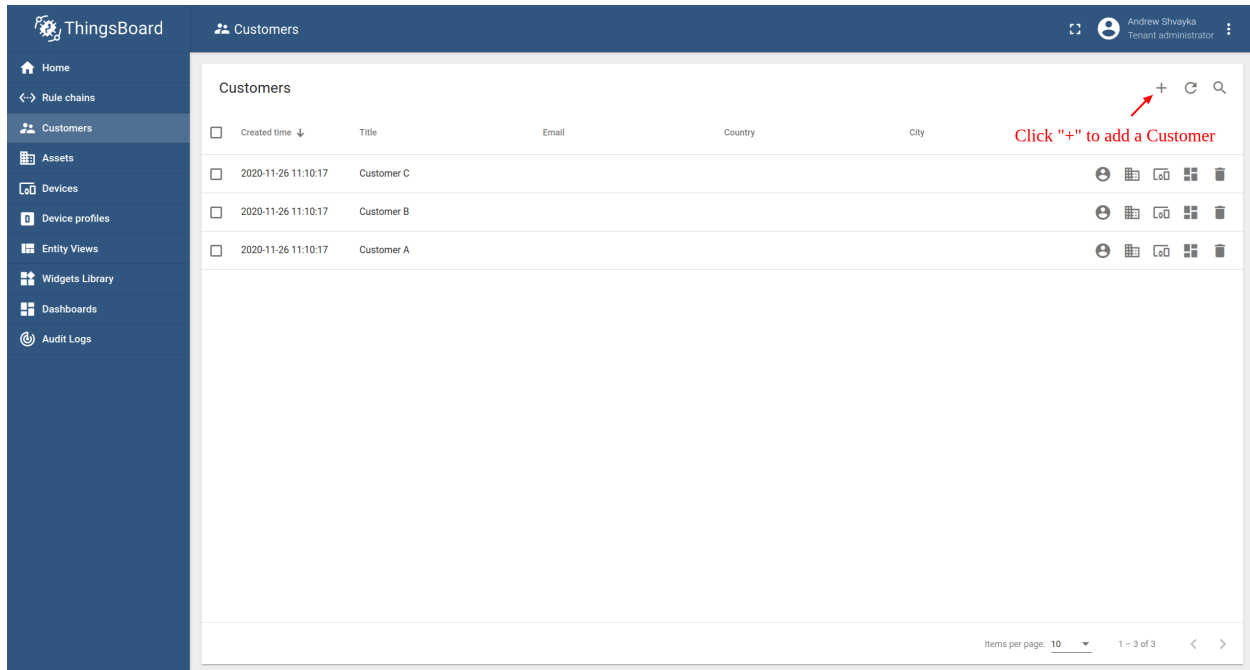
#### Step 7.1 Create customers

Let's create a customer with title "My New Customer". Please see instruction below:

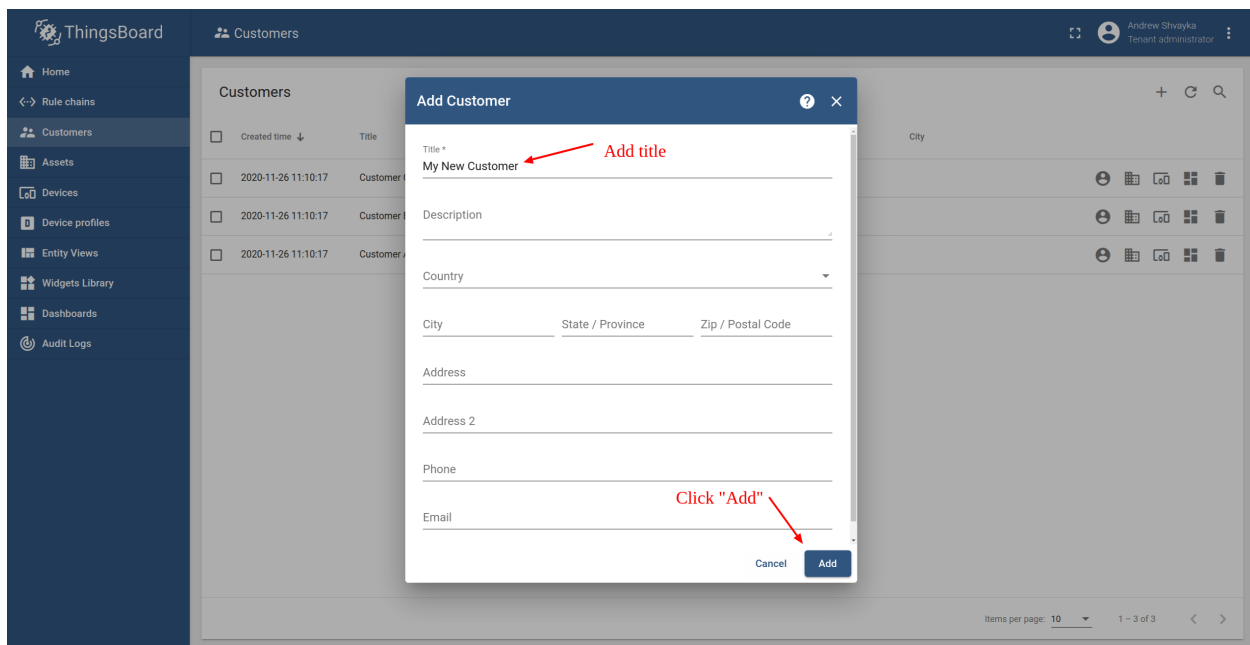
- Navigate to the Customers page.

The screenshot displays the ThingsBoard web interface. On the left, a dark blue sidebar contains a menu with icons and labels: Home, Rule chains, Customers, Assets, Devices, Device profiles, Entity Views, Widgets Library, Dashboards, and Audit Logs. A red arrow points from the 'Customers' menu item to the main content area, accompanied by the text 'Navigate to "Customers" page'. The main content area is titled 'My New Dashboard' and features a table of entities. The table has columns for 'Entity name', 'Entity type', and 'temperature'. It lists one entity: 'My New Device' of type 'Device' with a temperature of 30. Below the table is a 'New Timeseries - Flot' chart showing a single data point for 'temperature' at 30. On the right side of the dashboard, there is an 'Alarms' section with a table of alarms. The bottom right corner of the interface shows a red circular button with a white plus sign, and the text 'Powered by Thingsboard v3.2.0'.

- Click the "+" sign to add a customer.



- Add customer title and click “Add”.



### Step 7.2 Assign dashboards to Customer

Let's share our dashboard with the Customer. The Customer users will have read-only access to the Dashboard.

- See *Step 6.1 Assign dashboards of TA652FC-W to Customer*, or
- See *Step 6.1 Assign dashboards of TA652FH-W to Customer*.

### Step 7.3 Assign device to Customer

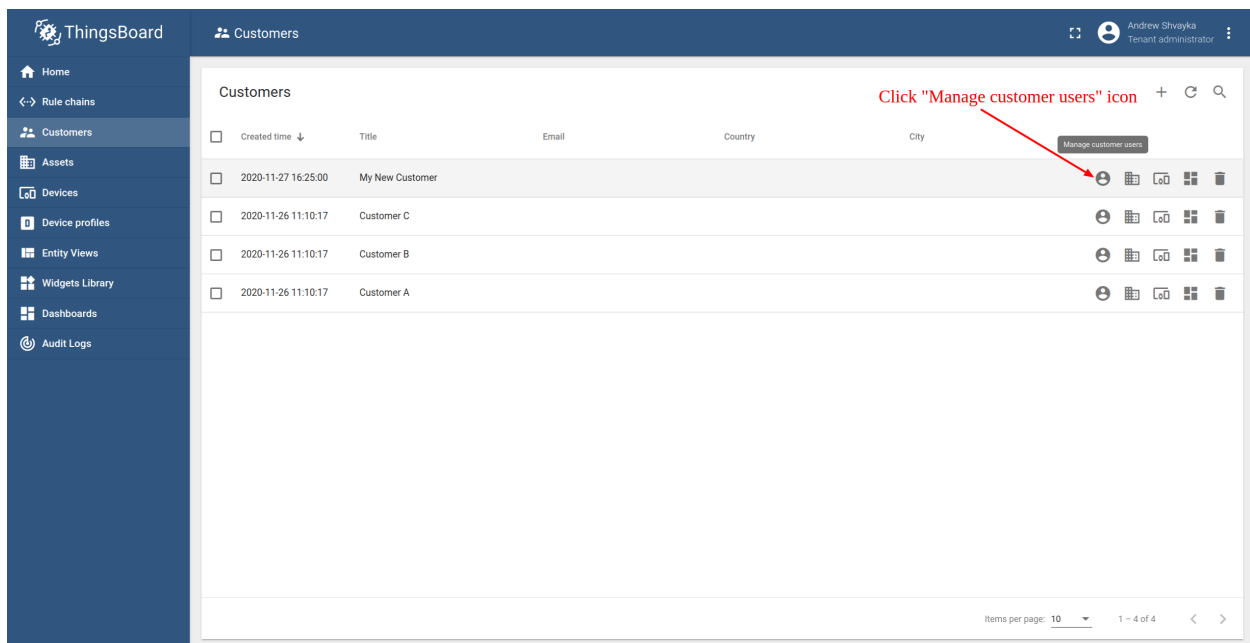
Let's assign device to the Customer. The Customer users will have ability to read and write telemetry and send commands to devices.

- See *Step 6.2 Assign TA652FC-W device to Customer*, or
- See *Step 6.2 Assign TA652FH-W device to Customer*

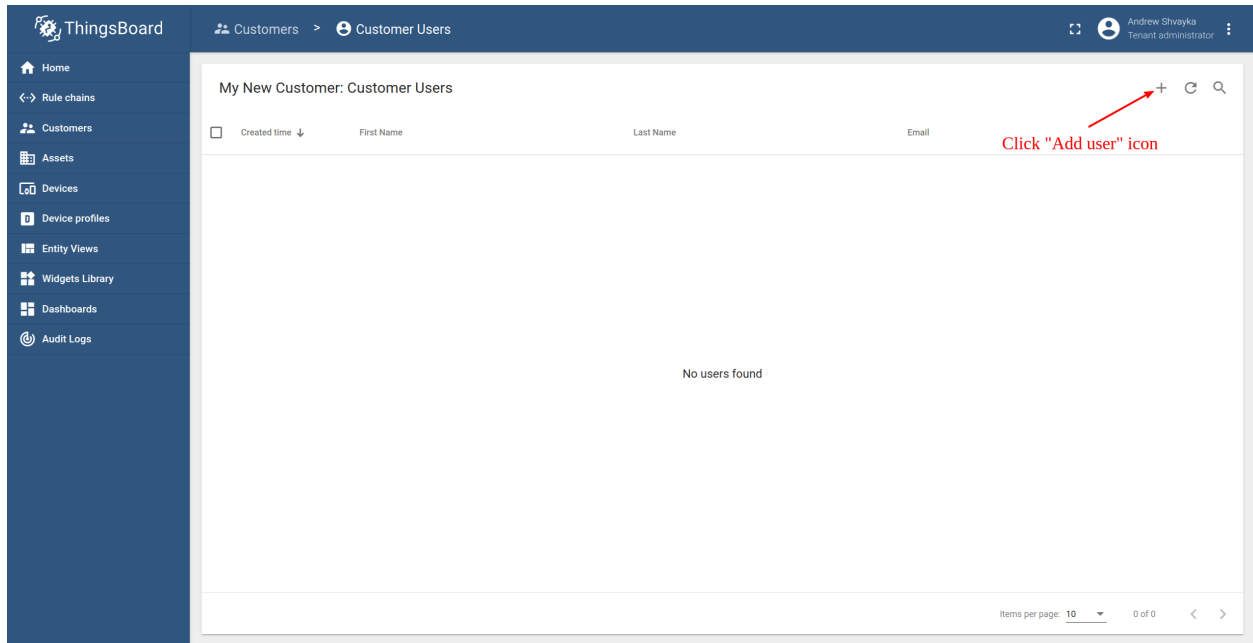
### Step 7.4 Create customer user

Finally, let's create a user that will belong to the customer and will have read-only access to the dashboard and the device. You may optionally configure the dashboard to appear just after user login to the platform web UI.

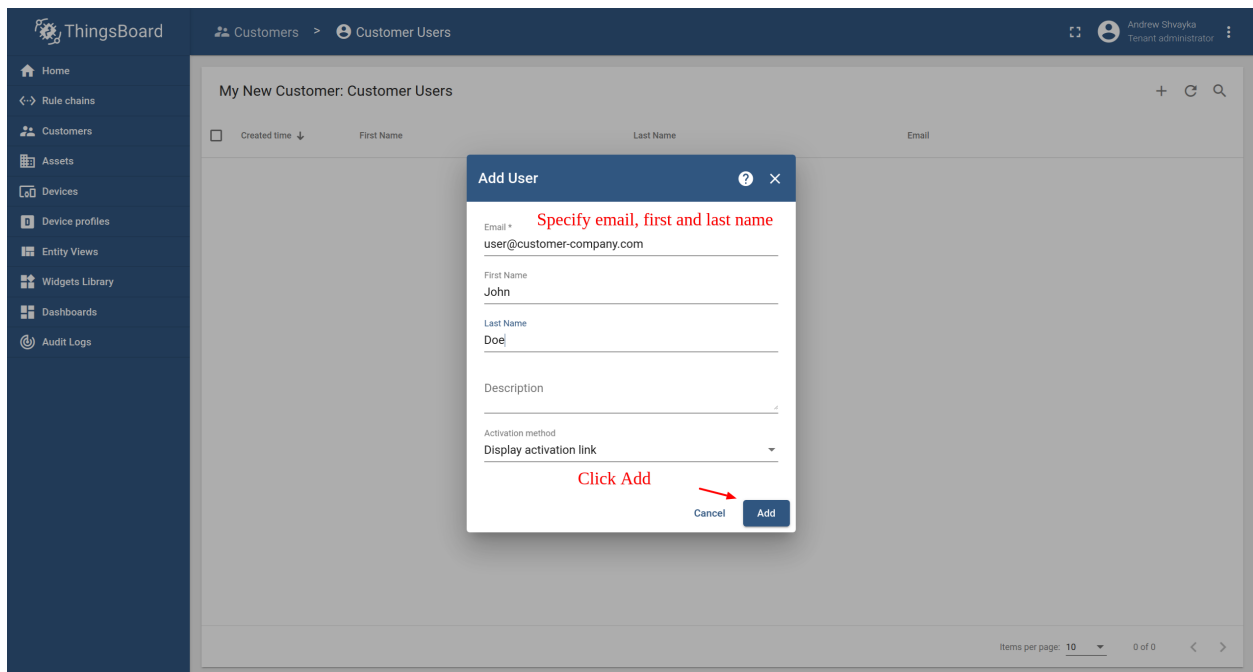
- Navigate back to the “Customers” page and click the “manage customer users” icon.



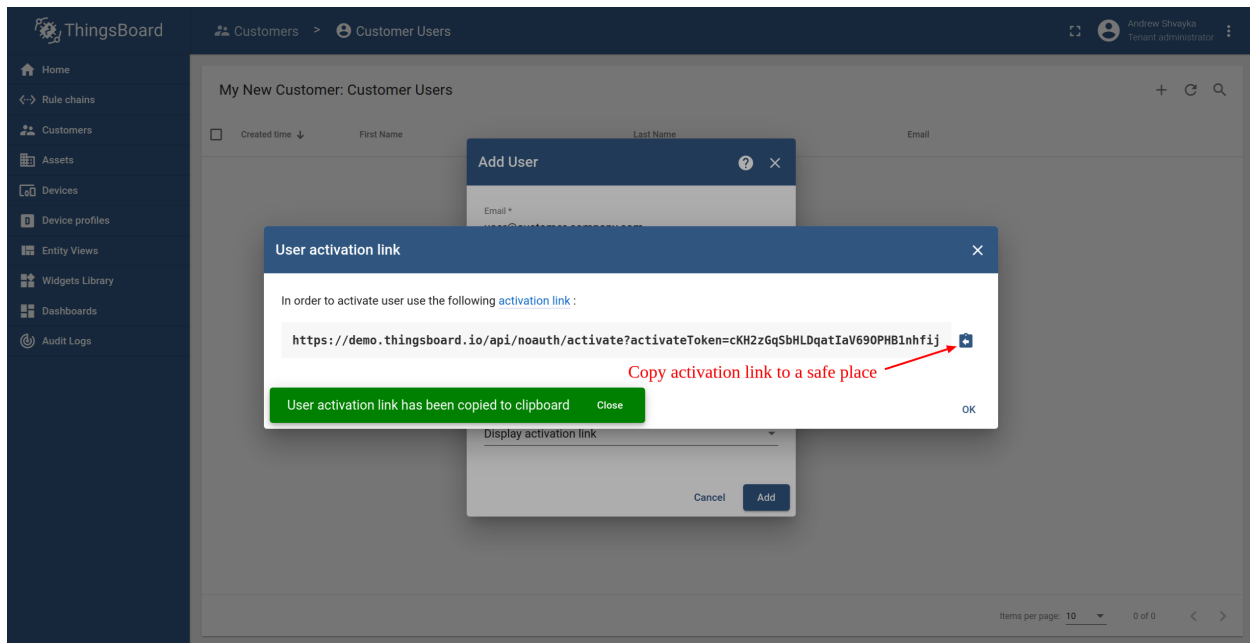
- Click the “Add user” icon.



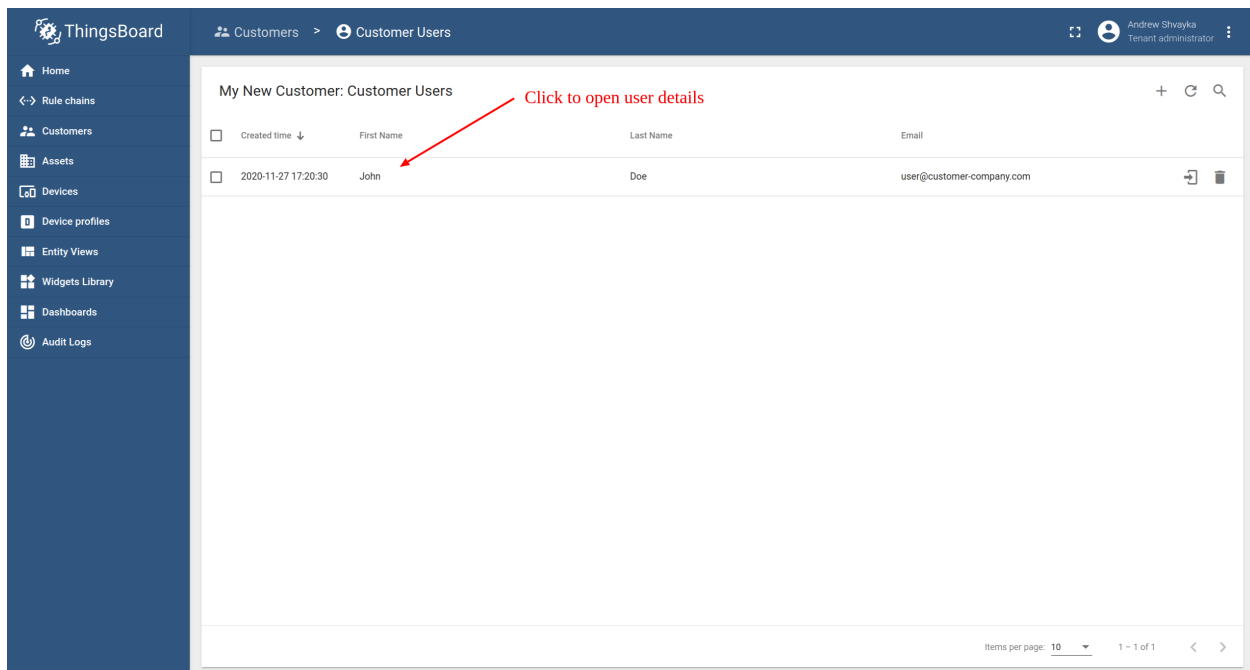
- Specify email that you will use to login as a customer user and click “Add”.



- Copy the activation link and save it to a safe place. You will use it later to set the password.

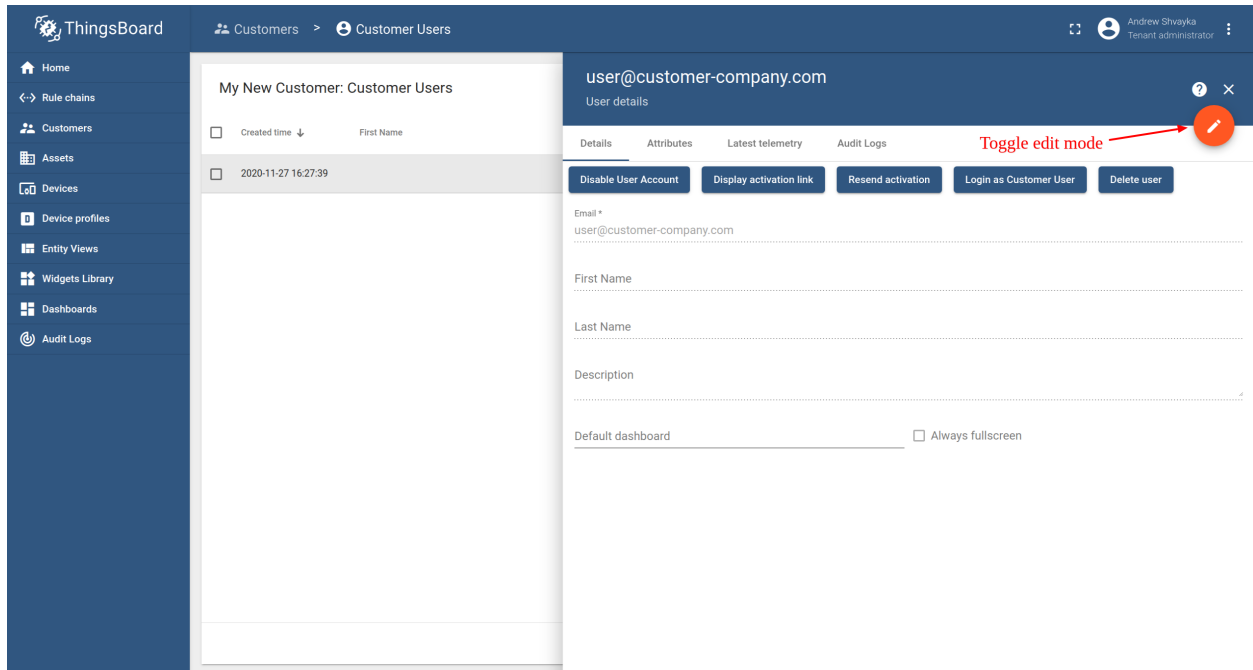


- Open user details.

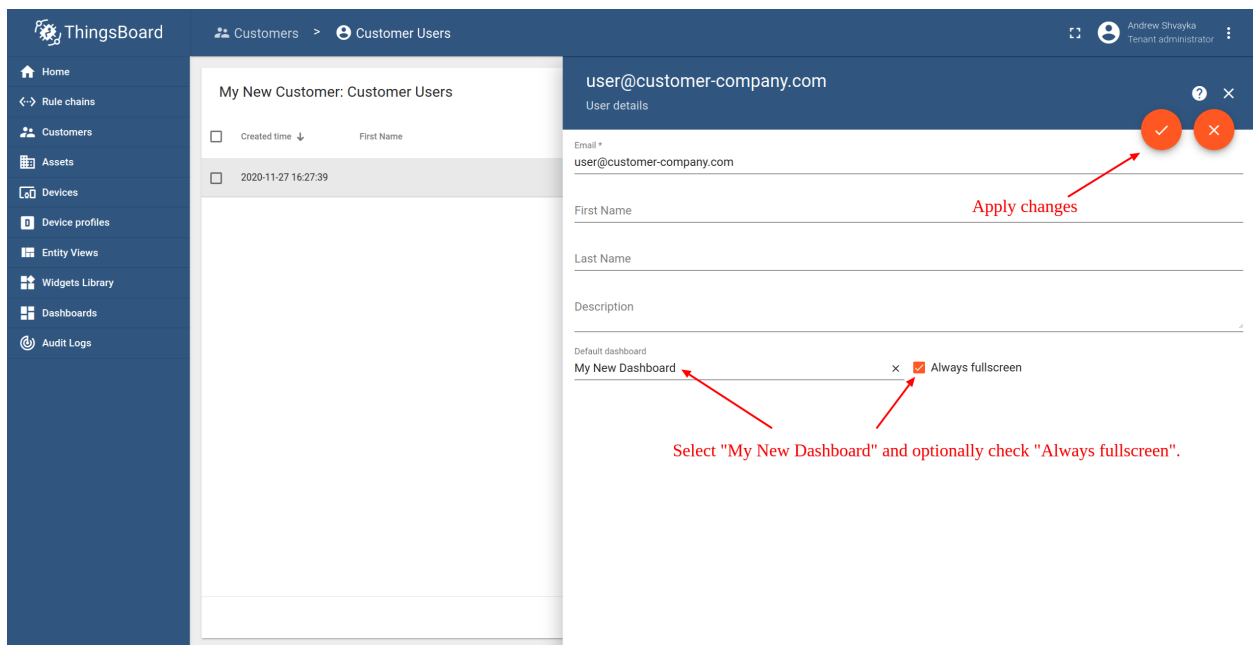


- (Option) Toggle edit mode.





- (Option) Select default dashboard and check “Always fullscreen”. Apply changes.



### Step 7.5 Activate customer user

- Use the activation link to set the password. Click “Create Password”. You will automatically login as a customer user.

The screenshot shows a web browser window with the URL `demo.thingsboard.io/api/noauth/activate?activateToken=ckH2zGqSbHLDqatlaV690PHB1nhfij`. A red arrow points to the URL bar with the text "Insert the saved activation link and press 'Enter'". Below the browser window is a dark blue modal titled "Create Password". It contains two password input fields: "Password" and "Password again", both masked with dots. Below the fields are two buttons: "Create Password" (orange) and "Cancel" (blue). A red arrow points to the "Create Password" button with the text "Input password and click 'Create Password' button.".

- You have logged in as a Customer User. You may browse the data and acknowledge/clear alarms.

The screenshot shows the ThingsBoard dashboard interface. The top navigation bar includes "My New Dashboard", "My New Device", "Realtime - last hour", and a user profile for "John Doe Customer". The main content area is divided into three sections: "Entities", "Alarms", and "New Timeseries - Flot". The "Entities" section shows a table with one row: "My New Device" (Device) with a temperature of 30. The "Alarms" section shows a table with one row: "2020-11-27 15:27:10" (My New Device) with a High Temperature (Critical) status. The "New Timeseries - Flot" section shows a line chart for "temperature" data. A red arrow points to the "Alarms" section with the text "You can hide this items in the dashboard settings. using tenant admin account".

### 1.2.10 Step 8. Open Dashboards

- See *Step 7. Open Dashboards of TA652FC-W*, or
- See *Step 7. Open Dashboards of TA652FH-W*.

### 1.2.11 Next Steps

- *Working with IoT Dashboards* - Customize your Dashboard & Widget.
- *Getting Started with Rule Engine* - Customize your event processing with Rule engine.
- *White-labeling* - Customize your company or product logo with ThingsBoard PE.
- *Platform Integrations* - Connect existing NB IoT, LoRaWAN, SigFox and other devices with specific payload formats directly to ThingsBoard platform.
- *Trendz Analytics* - Converts the IoT dataset into insights and simplifies the decision-making process.
- *Mobile application* - learn how to customize the mobile application.
- *ThingsBoard MQTT Device API | TA652FC-W MQTT API | TA652FH-W MQTT API* - Connect Avantec HVAC device to your existing IoT platform.

### 1.2.12 See also

- *Installation guides* - Learn how to setup ThingsBoard on various available operating systems.
- *Connect your device* - Learn how to connect devices based on your connectivity technology or solution.
- *Data visualization* - These guides contain instructions how to configure complex ThingsBoard dashboards.
- *Data processing & actions* - Learn how to use ThingsBoard Rule Engine.
- *IoT Data analytics* - Learn how to use rule engine to perform basic analytics tasks.
- *Hardware samples* - Learn how to connect various hardware platforms to ThingsBoard.
- *Advanced features* - Learn about advanced ThingsBoard features.

### 1.2.13 Your feedback

Don't hesitate to star Avantec on [github](#) to help us spread the word.



## THINGSBOARD

Here is an overview about ThingsBoard.

- *Key concepts*
- *Installation options*
- *Dashboards | Rule engine | White labeling*
- *MQTT Device API*

## 2.1 Overview

Reprinted articles: <https://thingsboard.io/docs/>

### 2.1.1 Introduction

See [What is ThingsBoard?](#)

### 2.1.2 Features

#### Entities and relations

See [Entities and relations](#) .

#### Working with telemetry data

See [Working with telemetry data](#).

#### Working with IoT device attributes

See [Working with IoT device attributes](#).

Attributes are treated key-value pairs. Flexibility and simplicity of the key-value format allow easy and seamless integration with almost any IoT device on the market.

Device specific attributes are separated into two main groups:

- **client-side attributes** - attributes are reported and managed by the device application. For example current software/firmware version, hardware specification, etc.

- **shared attributes** - attributes are reported and managed by the server-side application. Visible to the device application. For example customer subscription plan, target software/firmware version.

### Using RPC capabilities

See [Using RPC capabilities](#).

ThingsBoard RPC feature can be divided into two types based on originator: device-originated and server-originated RPC calls. In order to use more familiar names, we will name device-originated RPC calls as a **client-side RPC** calls and server-originated RPC calls as **server-side RPC** calls.

#### Client-side RPC

#### Server-side RPC

Server-side RPC calls can be divided into one-way and two-way:

- **One-way server-side RPC** request is sent to the device without delivery confirmation and obviously, does not provide any response from the device. RPC call may fail only if there is no active connection with the target device within a configurable timeout period.
- **Two-way server-side RPC** request is sent to the device and expects to receive a response from the device within the certain timeout. The Server-side request is blocked until the target device replies to the request.

### Claiming devices

Refer to [Claiming devices](#).

*TODO...*

### Over-the-air firmware and software updates

See [OTA Updates](#).

## 2.1.3 Getting Started Guides

These guides provide quick overview of main ThingsBoard features. Designed to be completed in 15-30 minutes.

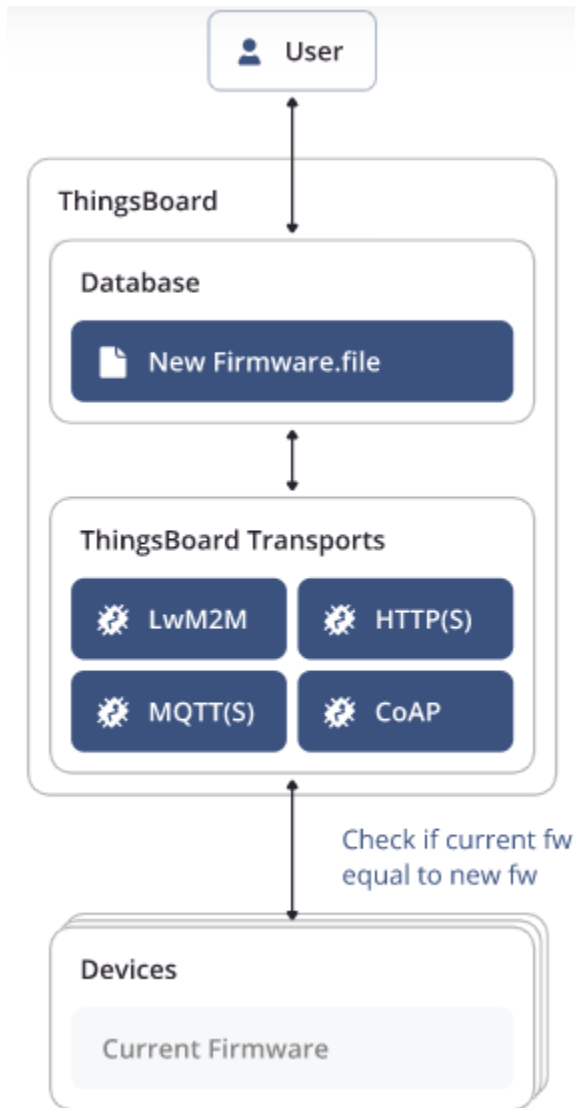
- [Hello world](#) : Learn how to collect IoT device data using MQTT, HTTP or CoAP and visualize it on a simple dashboard. Provides variety of sample scripts that you can run on your PC or laptop to simulate the device.
- [End user IoT dashboards](#) : Learn how to perform basic operations over Devices, Customers, and Dashboards.
- [Device data management](#) : Learn how to perform basic operations over device attributes to implement practical device management use cases.

## 2.2 Over-the-air firmware updates

Refer to [here](#).

### 2.2.1 Overview

Since ThingsBoard 3.3, ThingsBoard allows you to upload and distribute over-the-air(OTA) updates to devices. As a tenant administrator, you may upload firmware packages to the OTA repository. Once uploaded, you may assign them to [Device Profile](#) or [Device](#). ThingsBoard will notify devices about the available update and provide a protocol-specific API to download the firmware. The platform tracks status of the update and stores history of the updates. As a platform user, you may monitor the update process using the dashboard.

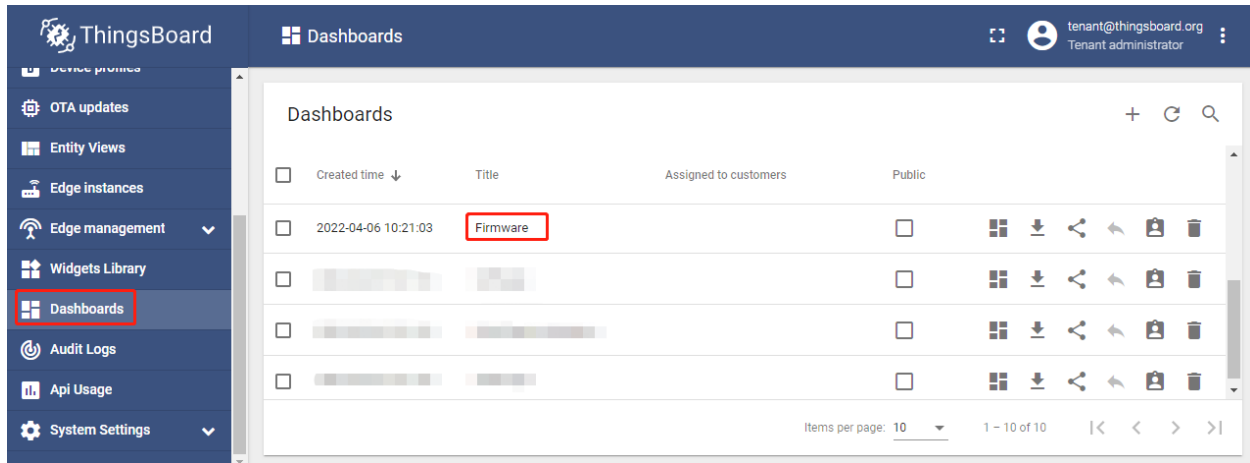


### 2.2.2 Firmware update monitoring dashboard

ThingsBoard provides the summary of the firmware update to monitor and track the firmware update status of your device, such as which devices are updating right now, any boot issues, and which ones have already been updated.

#### Check firmware dashboard

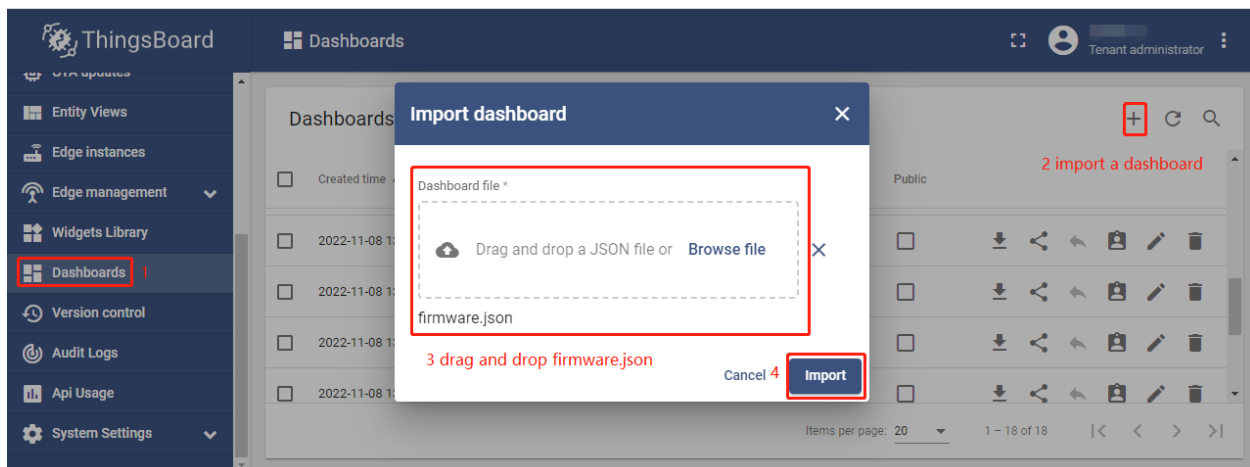
The dashboard is created automatically for each new tenant that you add to ThingsBoard in ThingsBoard CE. But this is not the case in ThingsBoard PE and [ThingsBoard Demo](#). Check if you have a firmware dashboard in your dashboard list:



#### Import firmware dashboard

If there is not firmware dashboard, you can also download the dashboard JSON `firmware.json` and import it for existing tenants.

- **Dashboards** → **+** → **Import a dashboard** → Drag and drop `firmware.json` → **Import**.





## Firmware dashboard details

There you can see a list of all devices with full information about their firmware.

**Monitor the status of devices.**  
Clicking on these tabs open the details.

**1. Search by firmware title;**  
**2. Expand to the full screen;**  
**3. Device firmware history;**  
**4. Change firmware;**  
**5. Download firmware;**  
**6. Copy checksum.**

Click the “History of the firmware updates” button next to the device name to learn about the firmware update status of specific device.

ThingsBoard

Dashboards > Firmware

Device list

Devices

Device	Current FW title	Current FW version	Target FW title	Target FW version	Target FW set time	Progress	Status
Headquarters Fan Coil	TA652FC-W-TB	1.6.12	TA652FC-W-TB	1.6.12	2023-01-10 15:11:12	100%	Updated
Plant Floor Heating	TA652FC-W-TB	1.6.8	TA652FH-W-TB	1.6.8	2022-12-15 18:40:10	100%	Updated

Items per page: 10 1 - 10 of 14

Summary:

- 0 Device Waiting
- 1 Device Updating
- 0 Device Failed
- 2 Device Updated

ThingsBoard

Dashboards > Firmware

Device list > Firmware history: Headquarters Fan Coil

Realtime - last 30 days

Timestamp	Current firmware title	Current firmware version	Target firmware title	Target firmware version	Status
2023-01-10 15:42:14	TA652FC-W-TB	1.6.12			Updated
2023-01-10 15:42:06	TA652FC-W-TB	1.6.12			Updating
2023-01-10 15:42:06	TA652FC-W-TB	1.6.10			Verified
2023-01-10 15:42:06	TA652FC-W-TB	1.6.10			Downloaded
2023-01-10 15:41:34	TA652FC-W-TB	1.6.10			Downloading
2023-01-10 15:41:33	TA652FC-W-TB	1.6.10			

Items per page: 10 1 - 10 of 21

### 2.2.3 Provision OTA package to ThingsBoard repository

- **OTA updates** → + → Input *title & version* → Select **device profile** → select **Firmware** → Enable **Upload binary file** → Drag and drop a package file → Disable **Auto-generate checksum** → Select **MD5** checksum algorithm, Checksum is blank → add.

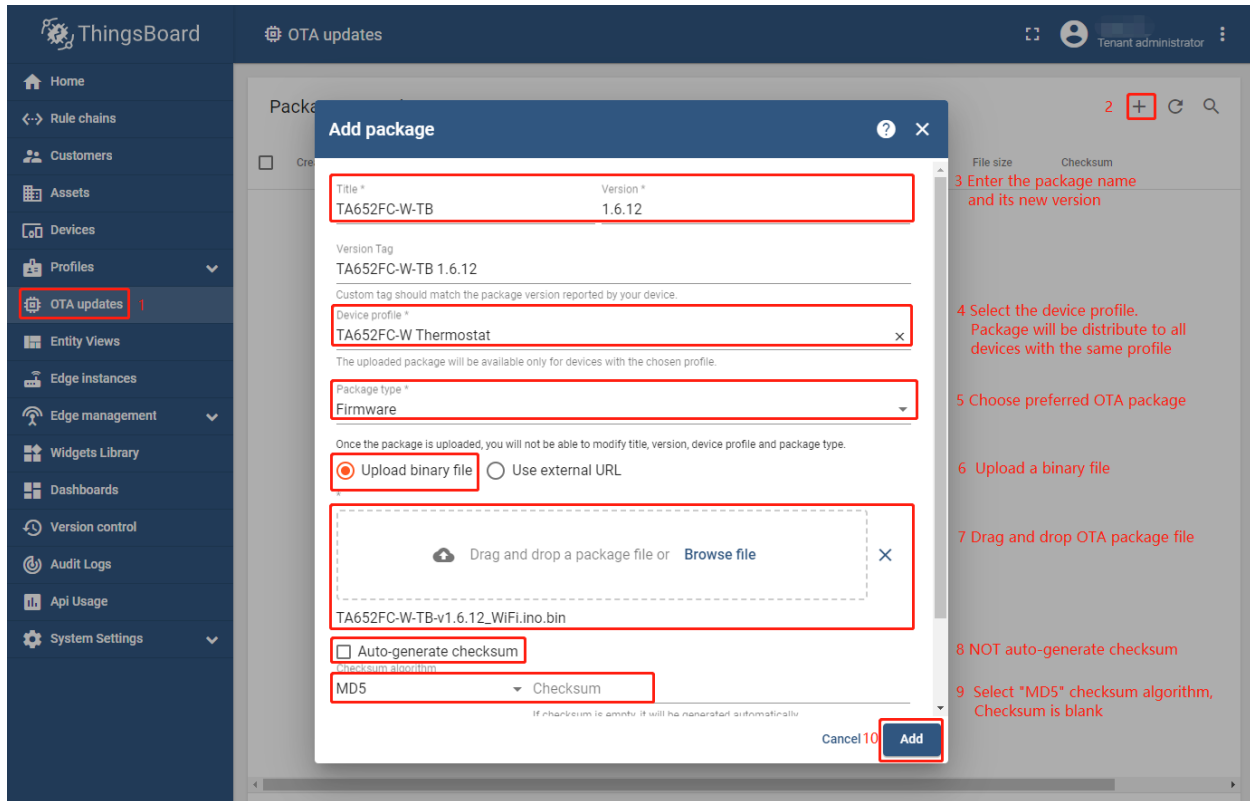


Table 1: OTA package parameters

Model	Title	Version	Device profile	Type	Checksum algorithm
TA652FC-W	TA652FC-W-TB	e.g. 1.6.10	TA652FC-W Thermostat	Firmware	MD5
TA652FH-W	TA652FH-W-TB	e.g. 1.6.10	TA652FH-W Thermostat	Firmware	MD5

- Navigate to the “OTA Updates” menu item to list and upload OTA update packages. Each package consists of:
  - Title - the name of your package. You can use different names for production and debug firmware.
  - Version - the version of your package. Combination of the title and version must be unique in scope of a tenant.
  - Device Profile - each package is compatible with one device profile. We track compatibility to prevent accidental updates of devices with incompatible firmware. Link to a device profile means that device that use this profile may be updated to the current package. However, the update is not triggered, until the user or script *assigns* the package to the device profile or device.
  - Type - can be either Firmware or Software.
  - Checksum algorithm - optional parameter, it is a short name of the checksum algorithm to use. Please select **MD5** checksum algorithm.
  - Checksum - optional parameter, it's a value of the file checksum. If no checksum provided by the user, server will use SHA-256 algorithm automatically.
  - Description - optional text description of the firmware.
- You can browse the provisioned packages as well as search them by title. Also, you are able to download and delete packages.

## Using Avantec HVAC devices with ThingsBoard, Release 2.3.4

ThingsBoard OTA updates

Packages repository

Created time ↓	Title	Version	Version Tag	Package type	Direct URL	File name	File size	Checksum
2023-01-09 15:09:05	TA652FC-W-TB	1.6.12	TA652FC-W-TB 1.6.12	Firmware		TA652FC-W-TB-v1.6.12_WiFi.ino.bin	1.3 MB	MD5: e10696e814e1c02...

1. Add OTA package;  
2. Refresh page;  
3. Search OTA package by title;  
4. Copy the checksum;  
5. Download OTA package;  
6. Delete OTA package.

- To open package details, click the table row. Package details allow you to copy package ID and checksum.

ThingsBoard OTA updates

TA652FC-W-TB  
OTA update details

Details

Open details page Download package Delete package

Copy package ID Copy checksum

Title TA652FC-W-TB Version 1.6.12

Version Tag TA652FC-W-TB 1.6.12

Device profile TA652FC-W Thermostat

Package type Firmware

Checksum algorithm MD5 Checksum e10696e814e1c02ea62464114196c81a

File name TA652FC-W-TB-v1.6.12\_WiFi.ino File size in bytes 1314480 Content type application/octet-stream

Description

In edit mode, you can modify the description

ThingsBoard OTA updates

Tenant administrator

Home Rule chains Customers Assets Devices Profiles OTA updates Entity Views Edge instances Edge management Widgets Library Dashboards Version control Audit Logs Api Usage

Packages repository

Created time ↓	Title	Version
2023-01-09 15:09:05	TA652FC-W-TB	1.6.12

TA652FC-W-TB  
OTA update details

Title \* TA652FC-W-TB Version \* 1.6.12

Version Tag TA652FC-W-TB 1.6.12

Device profile TA652FC-W Thermostat

Package type Firmware

Checksum algorithm MD5 Checksum e10696e814e1c02ea62464114196c81a

File name TA652FC-W-TB-v1.6.12\_WiFi.ino File size in bytes 1314480 Content type application/octet-stream

Description Fw update 1 edit description

2 save changes

- Also, **Audit logs** track information about users who provisioned the firmware.

ThingsBoard Audit Logs

Tenant administrator

Home Rule chains Customers Assets Devices Profiles OTA updates Entity Views Edge instances Edge management Widgets Library Dashboards Version control Audit Logs Api Usage

last day

Observe OTA package status

Timestamp ↓	Entity Type	Entity Name	User	Type	Status	Details
2023-01-09 15:09:05	OTA package	TA652FC-W-TB	lian...@h.hk	Added	Success	...

Items per page: 10 1 - 2 of 2

All actions listed are also available via **REST API**.

### 2.2.4 Assign OTA package to device profile

You may assign firmware to the device profile to automatically distribute the package to all devices that share the same profile. See screenshots below.

The first screenshot shows the 'Device profiles' page in ThingsBoard. The left sidebar has 'Profiles' and 'Device profiles' highlighted. The main panel shows a table of device profiles. The first row, 'TA652FC-W Thermostat', is highlighted. A red box around the 'Created time' column is labeled '2 click'. A red circle around the edit icon in the top right corner is labeled '3'.

The second screenshot shows the 'TA652FC-W Thermostat' profile details page. The 'Details' tab is selected. The 'Assigned firmware' field is highlighted with a red box and labeled '4 Select compatible OTA Update package'. The 'Save changes' button is highlighted with a red circle and labeled '5 Save changes'.

Created time	Name
2023-01-03 17:30:02	TA652FC-W Thermostat
2022-11-08 13:37:20	Charging port
2022-11-08 13:37:20	Air Quality Sensor
2022-11-08 13:37:20	Temperature Sensor
2022-10-13 13:39:06	default

Assigned firmware: TA652FC-W-TB (1.6.12)

The first screenshot shows the 'TA652FC-W Thermostat' device profile details. A modal dialog box is displayed in the center with the message: 'Change of the firmware may cause update of 1 device.' The dialog has 'Cancel' and 'Proceed' buttons. The 'Proceed' button is highlighted with a red box. In the background, the 'Assigned firmware' field shows 'TA652FC-W-TB (1.6.12)'.

The second screenshot shows the same device profile details page. The 'Assigned firmware' field is highlighted with a red box, and a red text annotation next to it reads: 'Firmware has been assigned to the device profile'. The 'Device provisioning' tab is active, and a red pencil icon is visible in the top right corner of the details panel.

The device profile details will let you choose only compatible OTA update packages (see [provisioning](#) for more info). Device profile may be used by thousands of devices. Assignment of the firmware triggers the *Update process*.

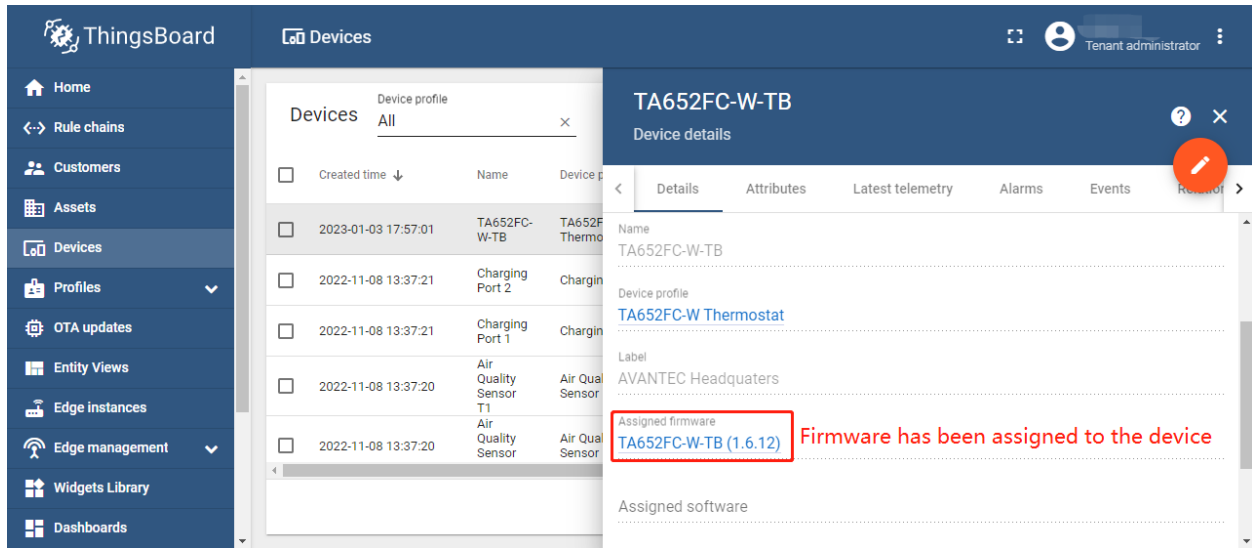
### 2.2.5 Assign OTA package to device

You may also assign firmware to specific device. See screenshots below.

The first screenshot shows the ThingsBoard interface with the 'Devices' menu highlighted. A red box highlights the 'TA652FC-W-TB' device in the list, with a red arrow pointing to it and the text '2 click'. The right panel shows the 'TA652FC-W-TB' device details, with a red circle around the 'Edit' icon (pencil) and the number '3'.

The second screenshot shows the 'TA652FC-W-TB' device details page. A red box highlights the 'Assigned firmware' field, which contains 'TA652FC-W-TB (1.6.12)'. A red arrow points to this field with the text '4 from the drop-down menu, select compatible firmware'. A red circle around the 'Save' icon (checkmark) is labeled with the number '5'.





The firmware version assigned to the device will automatically overwrite firmware version that is assigned to the device profile.

For example, let's assume you have Devices D1 and D2 that has Profile P1:

- If you assign package F1 to Profile P1 (via [profile details UI](#) or REST API), Devices D1 and D2 will be updated to F1.
- If you assign package F2 to Device D1 (via [device details UI](#) or REST API), Device D1 will be updated to F2.
- Subsequent assignment of the package F3 to the Profile P1 will affect only D2, since it has no specific firmware version assigned on the device level. So, D2 will be updated to F3, while D1 will continue to use F2.

Customers may choose available firmware and assign it to the devices that belong to them. However, customers can't provision or manage firmware packages.

## 2.2.6 Update process

Assignment of the firmware to the device or device profile triggers the update process. ThingsBoard tracks the progress of the update and persists it to the device attributes.

Update progress may have one of the following states. The state of the update is stored as an attribute of the device and is used to visualize the update process on the [dashboard](#).

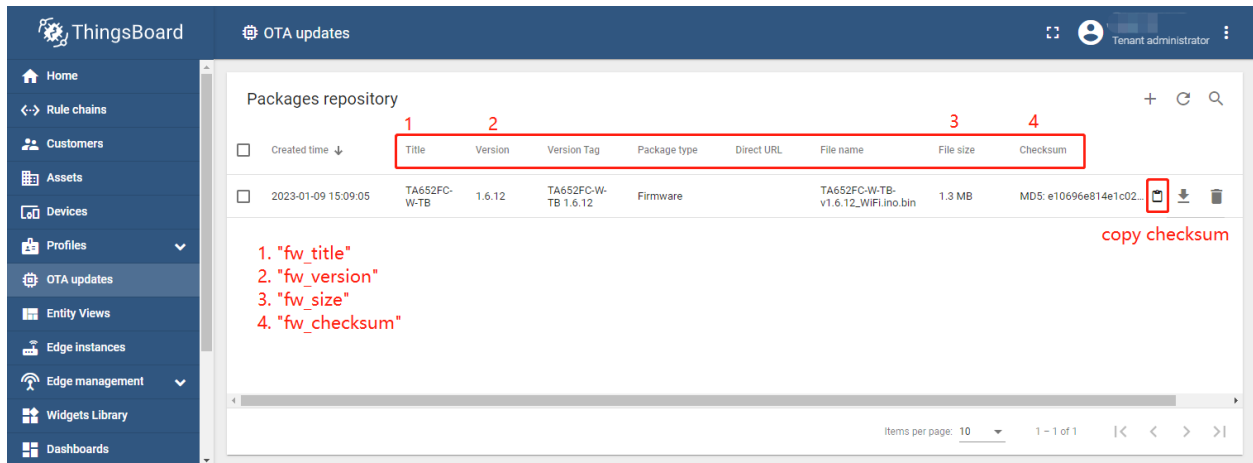
### QUEUED state

The very first state of the firmware update. Means that the notification about new firmware is queued but not yet pushed to the device. ThingsBoard queues the update notifications to avoid peak loads. The queue is processed with the constant pace. By default, it is configured to notify up to 100 device per minute. See [configuration properties](#) for more details.

### INITIATED state

Means that the notification about firmware is fetched from queue and pushed to device. Under the hood, ThingsBoard converts notification to the update of the following *shared attributes*:

- fw\_title - name of the firmware.
- fw\_version - version of the firmware.
- fw\_size - size of the firmware file in bytes.
- fw\_checksum - attribute that is used to verify integrity of the received file.
- fw\_checksum\_algorithm - the algorithm used to calculate file checksum.



Device is able to subscribe to shared attribute update using *MQTT API*.

### Update states reported by the device

The remaining states are reported by the device firmware that is currently processing the update. We have prepared description of those states and sample applications for the most popular protocols written in python. Sample applications simulate behavior of the device firmware and may be used as a reference for the implementation.

- DOWNLOADING - notification about new firmware update was received and device started downloading the update package.
- DOWNLOADED - device completed downloading of the update package.
- VERIFIED - device verified the checksum of the downloaded package.
- UPDATING - device started the firmware update. Typically is sent before reboot of the device or restart of the service.
- UPDATED - the firmware was successfully updated to the next version.
- FAILED - checksum wasn't verified, or the device failed to update. See "Device failed" tab on the Firmware dashboard for more details.
- Once the firmware is updated, ThingsBoard expects the device to send the following telemetry:

for firmware:

```
{"current_fw_title": "TA652FC-W-TB", "current_fw_version": "1.6.3", "fw_state": "UPDATED"}
```

If the firmware update failed, ThingsBoard expect the device to send the following telemetry:  
for firmware:

```
{"fw_state": "FAILED", "fw_error": "the human readable message about the cause of the_↵↵error"}
```

Firmware of the device is updated. To see its status, you should go to the firmware dashboard as it shows in the following paragraph.

To find out about the firmware update, you need to *make a request and subscribe to attributes*.

## 2.2.7 Configuration

### Queue processing pace

To set the max number of devices that will be notified in the chosen time period using the following [configuration](#) properties:

```
export TB_QUEUE_CORE_FW_PACK_INTERVAL_MS=600000
export TB_QUEUE_CORE_FW_PACK_SIZE=100
```

### Max size setting

By default, the maximum size of firmware that we can save in database is 2 gb. It can not be configured.

## 2.3 Working with IoT Dashboards

Reprinted articles: <https://thingsboard.io/docs/user-guide/dashboards/>

ThingsBoard allows you to configure customizable IoT dashboards. Each IoT Dashboard may contain multiple dashboard widgets that visualize data from multiple IoT devices. Once IoT Dashboard is created, you may assign it to one of the customers of you IoT project.

IoT Dashboards are light-weight and you may have millions of dashboards. For example, you may automatically create a dashboard for each new customer based on data from registered customer IoT devices. Or you may modify dashboard via script when a new device is assigned to a customer. All these actions may be done manually or automated via REST API.

You can find useful links to get started below:

- **Dashboards**

ThingsBoard provides the ability to create and manage dashboards. Each dashboard can contain plenty of widgets. Dashboards display data from many entities: devices, assets, etc. Dashboards can be assigned to Customers.

- **Widgets Library**

All IoT Dashboards are constructed using ThingsBoard widgets defined in the Widget Library. Each widget provides end-user functionality such as data visualization, remote device control, alarm management and display of static custom HTML content.

- **Digital** and **analog** gauges for latest real-time values visualization
- Highly customizable Bar and Line **charts** for visualization of historical and sliding-window data points

- **Map** widgets for tracking movement and latest positions of IoT devices on Google or OpenStreet maps.
- **GPIO** control widgets that allow sending GPIO toggle commands to devices.
- **Card** widgets to enhance your dashboards with flexible HTML labels based on static content or latest telemetry values from IoT devices.

### 2.3.1 How to customize

- **In Avantec Widgets & Avantec Dashboards:**
  - Change widget title, background, colors, fonts, shadows, etc
  - Add or remove widget
  - Modify dashboard states, aliases and widget actions
  - Visualizing assets data using Maps and Tables
- Create new Widgets
- Create your Dashboards

### 2.3.2 Next steps

- **ThingsBoard Data visualization**
  - [Visualizing assets data using Maps and Tables](#) : Learn how to: create assets and devices and define their relationships; add the server attributes and create a new dashboard; visualize data from the asset attributes using “Entities Table” and “Map” widgets.
  - [Dashboard states, aliases and widget actions](#) : Learn how to: add and configure new dashboard states; create various aliases; visualize the attributes data using the Image Map widget; create actions in different widgets in order to navigate between states; visualize the telemetry data using Analogue and Digital gauges and the Time-series widget.
  - [Remote device control and alarm management](#) : Learn how to: add and use the Knob Control widget; create alarm rules; handle alarms using the Alarms widget; make a dashboard public.
  - [Basic widget settings](#) : Learn how to: change widget title, background, colors, fonts, shadows, etc.
  - [Latest Values Map widget](#) : Learn how to create Map widget, based on latitude and longitude, and customize the Map widget layout and properties.
  - [Time Series Map widget](#) : Learn how to display your devices with the latest telemetry data on the Time Series Map widget and modify the widget properties.
  - [Trip Animation widget](#) : Learn how to display your devices with the latest telemetry data on the Trip Animation widget and modify the widget properties.
- **ThingsBoard Contribution and Development**
  - [Widgets Development Guide](#) : Learn how to customize and create custom widgets.

## 2.4 Getting Started with Rule Engine

### 2.4.1 What is ThingsBoard Rule Engine?

Rule Engine is an easy to use framework for building event-based workflows. There are 3 main components:

- **Message** - any incoming event. It can be an incoming data from devices, device life-cycle event, REST API event, RPC request, etc.
- **Rule Node** - a function that is executed on an incoming message. There are many different Node types that can filter, transform or execute some action on incoming Message.
- **Rule Chain** - nodes are connected with each other with relations, so the outbound message from rule node is sent to next connected rule nodes.

### 2.4.2 Typical Use Cases

ThingsBoard Rule Engine is a highly customizable framework for complex event processing. Here are some common use cases that one can configure via ThingsBoard Rule Chains:

- Data validation and modification for incoming telemetry or attributes before saving to the database.
- Copy telemetry or attributes from devices to related assets so you can aggregate telemetry. For example data from multiple devices can be aggregated in related Asset.
- Create/Update/Clear alarms based on defined conditions.
- Trigger actions based on device life-cycle events. For example, create alerts if Device is Online/Offline.
- Load additional data required for processing. For example, load temperature threshold value for a device that is defined in Device's Customer or Tenant attribute.
- Trigger REST API calls to external systems.
- Send emails when complex event occurs and use attributes of other entities inside Email Template.
- Take into account User preferences during event processing.
- Make RPC calls based on defined condition.
- Integrate with external pipelines like Kafka, Spark, AWS services, etc.

See [Getting Started with Rule Engine](#).

## 2.5 White-labeling

---

**Tip:** ThingsBoard PE Feature

Only [Professional Edition](#) supports White-labeling feature. Use [ThingsBoard Cloud](#) or [install](#) your own platform instance.

---

### 2.5.1 Feature

ThingsBoard web interface allows you to configure your company or product logo and color scheme in 2 minutes with zero coding efforts and no service restart required. The following configuration options are available:

- Configure color scheme, icon and favicon on System Administrator level;
- Tenant and Customer Administrator UI inherits configuration changes by default;
- Tenant and Customer Administrators are able to set up their own white-labeling configuration;
- System and Tenant Administrator are able to set up custom email server settings and customer email templates to interact with the users;
- Allow System administrator to enable/disable white-labeling for each tenant;
- Allow Tenant administrator to enable/disable white-labeling for each customer;
- Allow Tenant administrator to configure custom translation of system components and end-user dashboard elements.

See [White-labeling](#).

### 2.5.2 Next steps

- [Self-registration](#) - allows tenant to configure sign-up page for its customers to be able to simply sign-up and login to the ThingsBoard with predefined permission configurations.
- [Custom Translation](#) - allows you to upload alternative to existing language translations and extend the translation to specific GUI elements on the dashboards.
- [Custom Menu](#) - allows you to extend ThingsBoard UI with custom menu items and show/hide existing menu items.

## 2.6 Installation options

ThingsBoard is designed to run and utilize on majority of hardware, from local Raspberry PI to powerful servers in the cloud

- The alternative option is to install ThingsBoard using [Installation Guide](#).
- **Windows** users should follow this [guide](#).

## 2.7 Mobile application

### 2.7.1 ThingsBoard mobile application

The ThingsBoard Mobile Application is an open-source [mobile application project](#) based on [Flutter](#). It allows you to build your own IoT mobile application with minimum coding efforts.

This documentation can help you set up and run your first IoT mobile app, learn how to customize the app and publish it to Google Play or App Store.

- [Getting started with mobile application](#) - Learn how to setup and run your first IoT mobile app.
- [Customize your app with mobile application](#) - Learn how to customize your app without code changes.

- [Publish your app with mobile application](#) - Learn how to publish your app to Google Play or App Store.

## 2.7.2 ThingsBoard PE mobile application

The ThingsBoard **PE** Mobile Application is an open-source [PE mobile application project](#) based on [Flutter](#). It allows you to build your own IoT mobile application with minimum coding efforts.

This documentation can help you set up and run your first IoT mobile app, learn how to customize the app and publish it to Google Play or App Store.

- [Getting started with PE mobile application](#) - Learn how to setup and run your first IoT mobile app.
- [Customize your app with PE mobile application](#) - Learn how to customize your app without code changes.
- [Publish your app with PE mobile application](#) - Learn how to publish your app to Google Play or App Store.

## 2.8 ThingsBoard MQTT Device API

### 2.8.1 Introduction

See [ThingsBoard API reference](#).

ThingsBoard API consists of two main parts: **Device API** and **Server-side API**.

- **Server-side API is available as *REST API* and *Websocket API*:**
  - ***REST API*:**
    - \* [Administration REST API](#) - The server-side core APIs.
    - \* [Attributes query API](#) - The server-side APIs provided by [Telemetry Service](#).
    - \* [Time-series query API](#) - The server-side APIs provided by [Telemetry Service](#).
    - \* [RPC API](#) - The server-side APIs provided by [RPC Service](#).
    - \* [REST Client](#)
  - ***Websocket API*:**
    - \* [Websocket API](#) duplicates REST API functionality and provides the ability to subscribe to device data changes.
- **Device API is grouped by supported communication protocols:**
  - [MQTT Device API](#)
  - [CoAP Device API](#) ( Not supported! )
  - [HTTP Device API](#) ( Not supported! )

### 2.8.2 Getting started

See [MQTT Device API Reference](#).

#### MQTT basics

**MQTT** is a lightweight publish-subscribe messaging protocol which probably makes it the most suitable for various IoT devices. You can find more information about MQTT [here](#).

ThingsBoard server nodes act as an MQTT Broker that supports QoS levels 0 (at most once) and 1 (at least once) and a set of predefined topics.

#### Client libraries setup

You can find a large number of MQTT client libraries on the web. Examples in this article will be based on Mosquitto and MQTT.js. In order to setup one of those tools, you can use instructions in our [Hello World](#) guide.

#### MQTT Connect

We will use access token device credentials in this article and they will be referred to later as **\$ACCESS\_TOKEN**. The application needs to send MQTT CONNECT message with username that contains **\$ACCESS\_TOKEN**. Possible return codes and their reasons during connect sequence:

- **0x00 Connected** - Successfully connected to ThingsBoard MQTT server.
- **0x04 Connection Refused, bad user name or password** - Username is empty.
- **0x05 Connection Refused, not authorized** - Username contains invalid **\$ACCESS\_TOKEN**.

### 2.8.3 Key-value format

By default, ThingsBoard supports key-value content in **JSON**. Key is always a string, while value can be either string, boolean, double, long or JSON. For example:

```
{
  "stringKey": "value1",
  "booleanKey": true,
  "doubleKey": 42.0,
  "longKey": 73,
  "jsonKey": {
    "someNumber": 42,
    "someArray": [1, 2, 3],
    "someNestedObject": {"key": "value"}
  }
}
```



## 2.8.4 Telemetry upload API

In order to publish telemetry data to ThingsBoard server node, send PUBLISH message to the following topic:

```
v1/devices/me/telemetry
```

The simplest supported data formats are:

```
{"key1": "value1", "key2": "value2"}
```

or

```
[{"key1": "value1"}, {"key2": "value2"}]
```

**Please note** that in this case, the server-side timestamp will be assigned to uploaded data!

In case your device is able to get the client-side timestamp, you can use following format:

```
{"ts": 1451649600512, "values": {"key1": "value1", "key2": "value2"}}
```

In the example above, we assume that “1451649600512” is a [unix timestamp](#) with milliseconds precision. For example, the value “1451649600512” corresponds to “Fri, 01 Jan 2016 12:00:00.512 GMT”

### Example

Client library	Shell file	JSON file
<b>Mosquitto</b>	<i>mosquitto-telemetry.sh</i>	<ul style="list-style-type: none"> <li><i>telemetry-data-as-object.json</i></li> <li><i>telemetry-data-as-array.json</i></li> </ul>
<b>MQTT.js</b>	<i>mqtt-js-telemetry.sh</i>	<ul style="list-style-type: none"> <li><i>telemetry-data-with-ts.json</i></li> </ul>

### mosquitto-telemetry.sh

```
# Publish data as an object without timestamp (server-side timestamp will be used)
mosquitto_pub -d -h "127.0.0.1" -t "v1/devices/me/telemetry" -u "$ACCESS_TOKEN" -f
↳ "telemetry-data-as-object.json"
# Publish data as an array of objects without timestamp (server-side timestamp will be
↳ used)
mosquitto_pub -d -h "127.0.0.1" -t "v1/devices/me/telemetry" -u "$ACCESS_TOKEN" -f
↳ "telemetry-data-as-array.json"
# Publish data as an object with timestamp (server-side timestamp will be used)
mosquitto_pub -d -h "127.0.0.1" -t "v1/devices/me/telemetry" -u "$ACCESS_TOKEN" -f
↳ "telemetry-data-with-ts.json"
```

### mqtt-js-telemetry.sh

```
# Publish data as an object without timestamp (server-side timestamp will be used)
cat telemetry-data-as-object.json | mqtt pub -v -h "127.0.0.1" -t "v1/devices/me/
↳telemetry" -u '$ACCESS_TOKEN' -s
# Publish data as an array of objects without timestamp (server-side timestamp will be
↳used)
cat telemetry-data-as-array.json | mqtt pub -v -h "127.0.0.1" -t "v1/devices/me/telemetry
↳" -u '$ACCESS_TOKEN' -s
# Publish data as an object with timestamp (server-side timestamp will be used)
cat telemetry-data-with-ts.json | mqtt pub -v -h "127.0.0.1" -t "v1/devices/me/telemetry
↳" -u '$ACCESS_TOKEN' -s
```

### telemetry-data-as-object.json

```
{
  "stringKey": "value1",
  "booleanKey": true,
  "doubleKey": 42.0,
  "longKey": 73,
  "jsonKey": {
    "someNumber": 42,
    "someArray": [1,2,3],
    "someNestedObject": {"key": "value"}
  }
}
```

### telemetry-data-as-array.json

```
[{"key1": "value1"}, {"key2": true}]
```

### telemetry-data-with-ts.json

```
{
  "ts": 1451649600512,
  "values": {
    "stringKey": "value1",
    "booleanKey": true,
    "doubleKey": 42.0,
    "longKey": 73,
    "jsonKey": {
      "someNumber": 42,
      "someArray": [1, 2, 3],
      "someNestedObject": {
        "key": "value"
      }
    }
  }
}
```

(continues on next page)

(continued from previous page)

```
}
}
```

## 2.8.5 Attributes API

ThingsBoard attributes API allows devices to

- Request **client-side** and **shared** device attributes from the server.
- Upload **client-side** device attributes to the server.
- Subscribe to **shared** device attributes from the server.

### Request attribute values from the server

Before sending PUBLISH message with the attributes request, client need to **subscribe** to:

```
v1/devices/me/attributes/response/+
```

Once subscribed, the client may request client-side or shared device attributes to ThingsBoard server node, send **PUBLISH** message to the following topic:

```
v1/devices/me/attributes/request/$request_id
```

where **\$request\_id** is your integer request identifier.

The client should receive the response to the following topic:

```
v1/devices/me/attributes/response/$request_id
```

### Example

The following example is written in javascript and is based on mqtt.js. Pure command-line examples are not available because subscribe and publish need to happen in the same mqtt session.

Client library	Shell file	JavaScript file	Result (JSON file)
MQTT.js	<i>mqtt-js-attributes-request.sh</i>	<i>mqtt-js-attributes-request.js</i>	<i>attributes-response.json</i>

### mqtt-js-attributes-request.sh

```
export TOKEN=$ACCESS_TOKEN
node mqtt-js-attributes-request.js
```

**mqtt-js-attributes-request.js**

```

var mqtt = require('mqtt')
var client = mqtt.connect('mqtt://127.0.0.1',{
  username: process.env.TOKEN
})

client.on('connect', function () {
  console.log('connected')
  client.subscribe('v1/devices/me/attributes/response+')
  client.publish('v1/devices/me/attributes/request/1', '{"clientKeys":"attribute1,
↪attribute2", "sharedKeys":"shared1,shared2"}')
})

client.on('message', function (topic, message) {
  console.log('response.topic: ' + topic)
  console.log('response.body: ' + message.toString())
  client.end()
})

```

**attributes-response.json**

```

{"key1":"value1"}

```

**Please note**, the intersection of client-side and shared device attribute keys is a **bad** practice! However, it is still possible to have same keys for client, shared or even server-side attributes.

**Publish attribute update to the server**

In order to publish client-side device attributes to ThingsBoard server node, send **PUBLISH** message to the following topic:

```

v1/devices/me/attributes

```

**Example**

Client library	Shell file	JSON file
<b>Mosquitto</b>	<i>mosquitto-attributes-publish.sh</i>	<i>new-attributes-values.json</i>
<b>MQTT.js</b>	<i>mqtt-js-attributes-publish.sh</i>	

**mosquitto-attributes-publish.sh**

```
# Publish client-side attributes update
mosquitto_pub -d -h "127.0.0.1" -t "v1/devices/me/attributes" -u "$ACCESS_TOKEN" -f "new-attributes-values.json"
```

**mqtt-js-attributes-publish.sh**

```
# Publish client-side attributes update
cat new-attributes-values.json | mqtt pub -d -h "127.0.0.1" -t "v1/devices/me/attributes" -u '$ACCESS_TOKEN' -s
```

**new-attributes-values.json**

```
{
  "stringKey": "value1",
  "booleanKey": true,
  "doubleKey": 42.0,
  "longKey": 73,
  "jsonKey": {
    "someNumber": 42,
    "someArray": [1,2,3],
    "someNestedObject": {"key": "value"}
  }
}
```

**Subscribe to attribute updates from the server**

In order to subscribe to shared device attribute changes, send **SUBSCRIBE** message to the following topic:

```
v1/devices/me/attributes
```

When a shared attribute is changed by one of the server-side components (such as the REST API or the Rule Chain), the client will **receive** the following update:

```
{"key1": "value1"}
```

**Example**

Client library	Shell file
<b>Mosquitto</b>	<i>mosquitto-attributes-subscribe.sh</i>
<b>MQTT.js</b>	<i>mqtt-js-attributes-subscribe.sh</i>

### mosquitto-attributes-subscribe.sh

```
# Subscribes to attribute updates
mosquitto_sub -d -h "127.0.0.1" -t "v1/devices/me/attributes" -u "$ACCESS_TOKEN"
```

### mqtt-js-attributes-subscribe.sh

```
# Subscribes to attribute updates
mqtt sub -v "127.0.0.1" -t "v1/devices/me/attributes" -u '$ACCESS_TOKEN'
```

## 2.8.6 PRC API

### Server-side RPC

In order to subscribe to RPC commands from the server, send **SUBSCRIBE** message to the following topic:

```
v1/devices/me/rpc/request/+
```

Once subscribed, the client will receive individual commands as a **PUBLISH** message to the corresponding topic:

```
v1/devices/me/rpc/request/$request_id
```

where **\$request\_id** is an integer request identifier.

The client should publish the response to the following topic:

```
v1/devices/me/rpc/response/$request_id
```

### Example

The following example is written in javascript and is based on mqtt.js. Pure command-line examples are not available because subscribe and publish need to happen in the same mqtt session.

Client library	Shell file	JavaScript file
<b>MQTT.js</b>	<i>mqtt-js-rpc-from-server.sh</i>	<i>mqtt-js-rpc-from-server.js</i>

### mqtt-js-rpc-from-server.sh

```
export TOKEN=$ACCESS_TOKEN
node mqtt-js-rpc-from-server.js
```

### mqtt-js-rpc-from-server.js

```

var mqtt = require('mqtt');
var client = mqtt.connect('mqtt://127.0.0.1',{
  username: process.env.TOKEN
});

client.on('connect', function () {
  console.log('connected');
  client.subscribe('v1/devices/me/rpc/request/+')
});

client.on('message', function (topic, message) {
  console.log('request.topic: ' + topic);
  console.log('request.body: ' + message.toString());
  var requestId = topic.slice('v1/devices/me/rpc/request/'.length);
  //client acts as an echo service
  client.publish('v1/devices/me/rpc/response/' + requestId, message);
});

```

### Client-side RPC

In order to subscribe to client-side RPC response from the server, send **SUBSCRIBE** message to the following topic:

```
v1/devices/me/rpc/response/+
```

Once subscribed, the client may send **PUBLISH** message to the following topic:

```
v1/devices/me/rpc/request/$request_id
```

where **\$request\_id** is an integer request identifier. The response from server will be published to the following topic:

```
v1/devices/me/rpc/response/$request_id
```

### Example

The following example is written in javascript and is based on mqtt.js. Pure command-line examples are not available because subscribe and publish need to happen in the same mqtt session.

Client library	Shell file	JavaScript file
MQTT.js	<i>mqtt-js-rpc-from-client.sh</i>	<i>mqtt-js-rpc-from-client.js</i>

### mqtt-js-rpc-from-client.sh

```
export TOKEN=$ACCESS_TOKEN
node mqtt-js-rpc-from-client.js
```

### mqtt-js-rpc-from-client.js

```
var mqtt = require('mqtt');
var client = mqtt.connect('mqtt://127.0.0.1', {
  username: process.env.TOKEN
});

client.on('connect', function () {
  console.log('connected');
  client.subscribe('v1/devices/me/rpc/response/+');
  var requestId = 1;
  var request = {
    "method": "getTime",
    "params": {}
  };
  client.publish('v1/devices/me/rpc/request/' + requestId, JSON.stringify(request));
});

client.on('message', function (topic, message) {
  console.log('response.topic: ' + topic);
  console.log('response.body: ' + message.toString());
});
```

## 2.8.7 Claiming API

Please see the corresponding article to get more information about the [Claiming devices](#) feature.

In order to initiate claiming device, send PUBLISH message to the following topic:

```
v1/devices/me/claim
```

The supported data format is:

```
{"secretKey":"value", "durationMs":60000}
```

**Please note** that the above fields are optional. In case the **secretKey** is not specified, the empty string as a default value is used. In case the **durationMs** is not specified, the system parameter **device.claim.duration** is used (in the file `/etc/thingsboard/conf/thingsboard.yml` ).



## 2.8.8 Firmware API

Replace 8192 with your chunk size.

When ThingsBoard initiates an MQTT device firmware update, it sets the `fw_title`, `fw_version`, `fw_checksum`, `fw_checksum_algorithm` shared attributes. To receive the shared attribute updates, the device has to subscribe to:

```
v1/devices/me/attributes/response/+
```

Where

`+` is the Wildcard character.

When the MQTT device receives updates for `fw_title` and `fw_version` shared attributes, it has to send PUBLISH message to:

```
v2/fw/request/${requestId}/chunk/${chunkIndex}
```

Where

**`${requestId}`** - number corresponding to the number of firmware updates. The `${requestId}` has to be different for each firmware update.

**`${chunkIndex}`** - number corresponding to the index of firmware chunks. The `${chunkID}` are counted from 0. The device must increment the chunk index for each request until the received chunk size is zero. And the MQTT payload should be the size of the firmware chunk in bytes.

For each new firmware update, you need to change the request ID and subscribe to:

```
v2/fw/response/+chunk/+
```

Where

`+` is the Wildcard character.

## 2.8.9 Device MQTT Topic

Table 2: Device MQTT Topic

Function	Topic	Subscribe	Tx	Rx
Telemetry			v1/devices/me/telemetry	
Request tributes	at-	v1/devices/me/attributes	v1/devices/me/attributes/request/	v1/devices/me/attributes/response/\${requestId}
Publish tributes	at-		v1/devices/me/attributes	
Subscribe tributes update	at-	v1/devices/me/attributes		v1/devices/me/attributes
Server-Side RPC		v1/devices/me/rpc/request	v1/devices/me/rpc/response/\${requestId}	v1/devices/me/rpc/request/\${requestId}
Client-Side RPC		v1/devices/me/rpc/response	v1/devices/me/rpc/request/\${requestId}	v1/devices/me/rpc/response/\${requestId}
Claiming device			v1/devices/me/claim	
Firmware updates*	up-	v2/fw/response/+chunk	v2/fw/request/\${requestId}/chunk/	v2/fw/response/\${requestId}/chunk/\${chunkIndex}

---

**Note:**

- The order in which topics are performed.
  - **Firmware updates** needs the support of *Telemetry*, *Request attributes* and *Subscribe attributes update*.
-

## AVANTEC EXTENSION

Here is an overview about Avantec Customization.

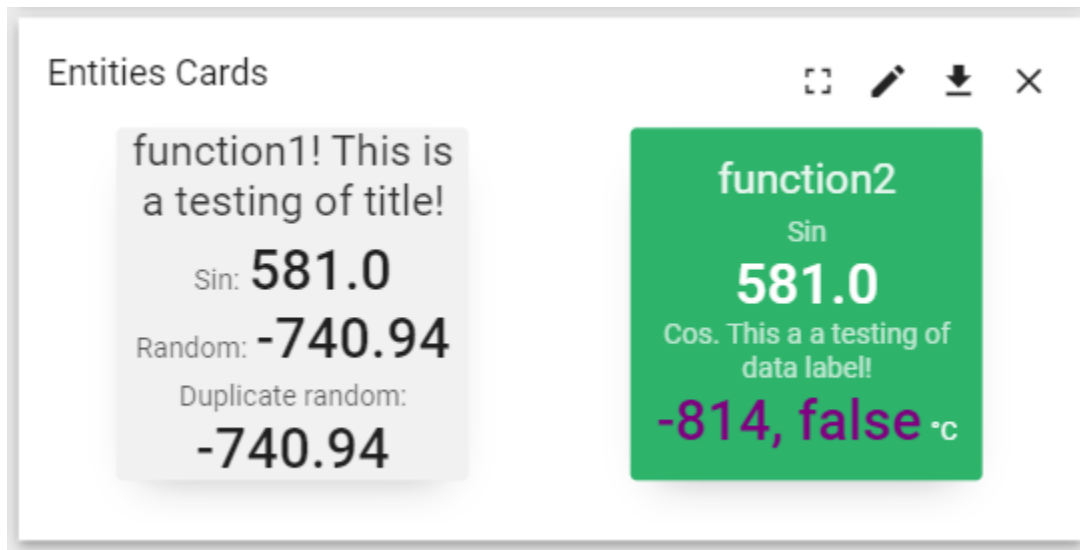
- *Demo Dashboards*
- *Avantec Widgets*

### 3.1 Avantec widgets

#### 3.1.1 Widget list

##### Entities cards

Latest values widget.



Configuration:

1. Add all **datasources** and **data keys** in Data page.
2. Add some **card templates** in Advanced page. A card template per entity alias used in the datasources.
  - **Alias Name:** Entity alias name.
  - **Card HTML pattern:** HTML template. You can use `${dsName}`, `${entityName}`, `${deviceName}`, `${entityLabel}`, `${aliasName}`, `${entityDescription}`, or `${your_datakey_label}` in it.

- **Card style function:** `f(datasource, ctx)`: CSS.
3. Optionally, configure some data keys used in **Card HTML pattern**: Data page → Pen icon → Advanced page.
    - **Cell content function:** `f(value, datasource, ctx)`: text or HTML
    - **Cell style function:** `f(value, datasource, ctx)`: CSS.
  4. Add some action in Action page. A action of element click per entity alias used in the datasources.
    - **Action source:** On HTML element click
    - **Name:** Entity alias name.
  5. Optionally, if you want to hide the border of the widget, you should do this in Settings page:
    - Disable Display widget title.
    - Background color of the widget should be the same as background color of Dashboard.

### RPC button with params & response

Control widget.

For Server-side RPC.

The screenshot shows a configuration window titled "RTC button with params & response". It contains a text input field labeled "RPC param label \*". Below this is a grey button labeled "Send RPC Command". At the bottom is a large text area labeled "RPC command response".

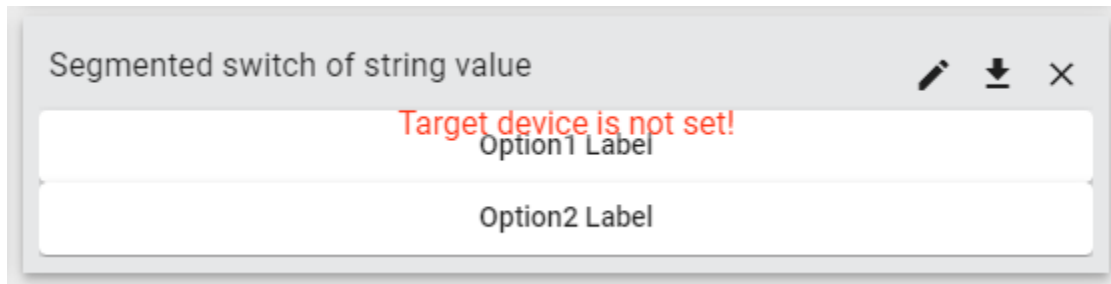
### Segmented switch of boolean value

Control widget.

The screenshot shows a configuration window titled "Segment switch of boolean value". It features a segmented switch control. The top segment is highlighted in red and labeled "True label", with the text "Target device is not set!" displayed above it. The bottom segment is labeled "False label".

### Segmented switch of string value

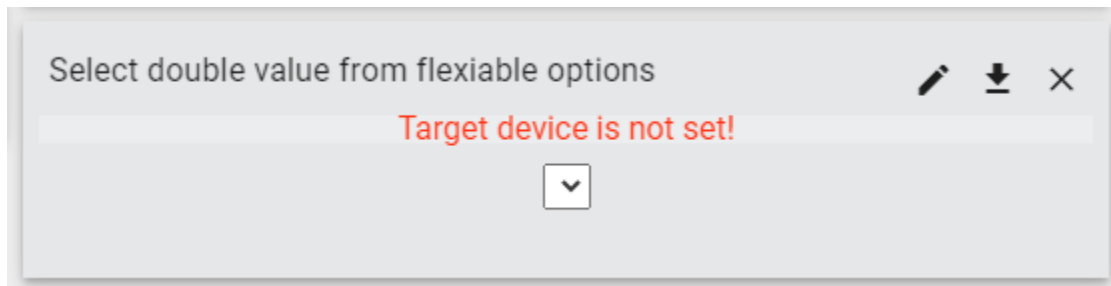
Control widget.



### Select double value from flexible options

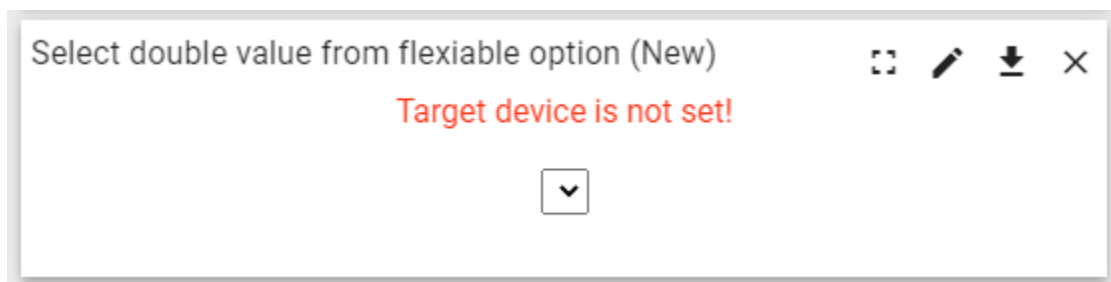
Control widget.

*Deprecated!*



### Select double value from flexible options (New)

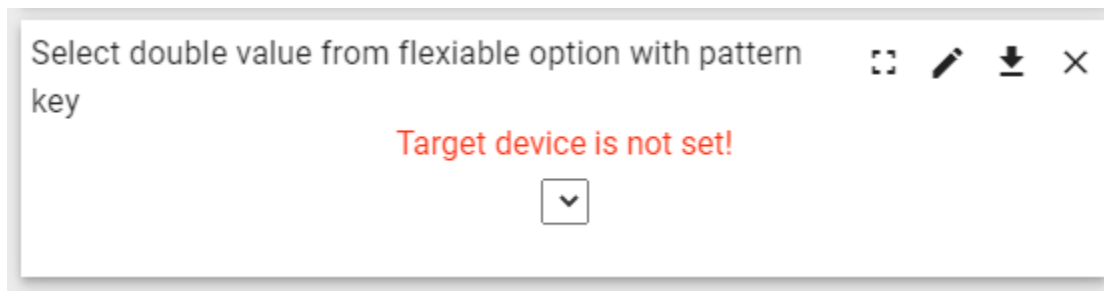
Control widget.



### Select double value from flexible options with pattern key

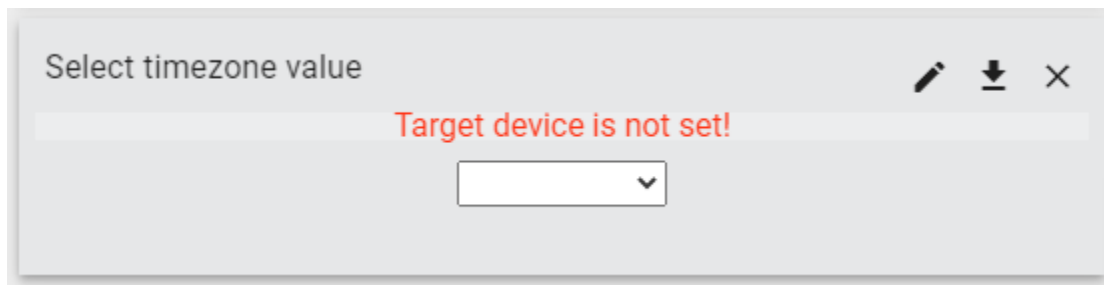
Control widget.

Some parameters of this widget can be appended with a suffix for programming time.



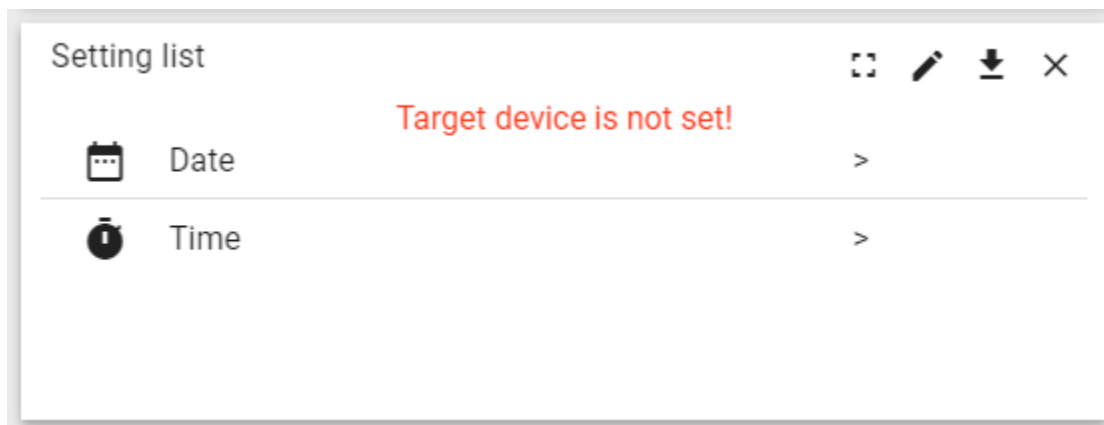
### Select time zone value

Control widget.



### Setting list

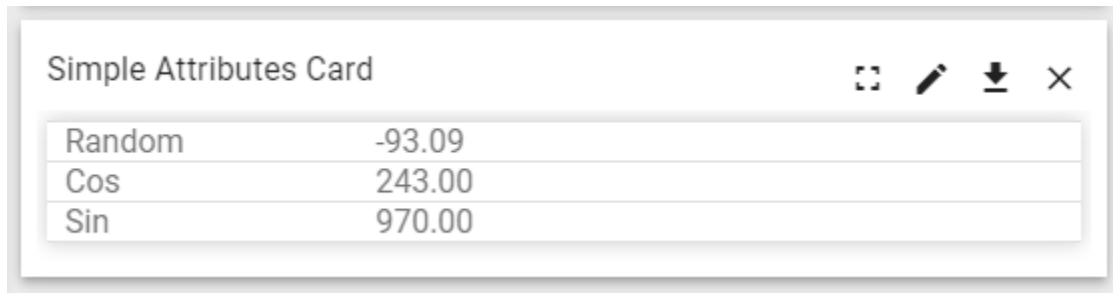
Control widget.



### Simple Attributes Card

Latest values widget.

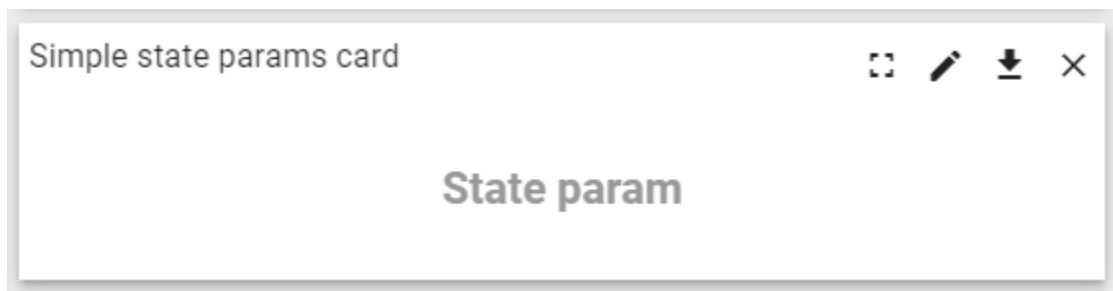
This widget can display the attributes of devices.



### Simple state params card

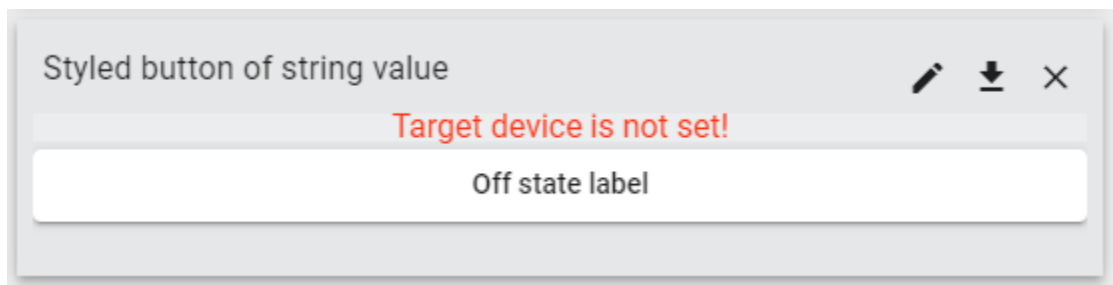
Control widget.

This widget can display the state parameters of the Dashboard.



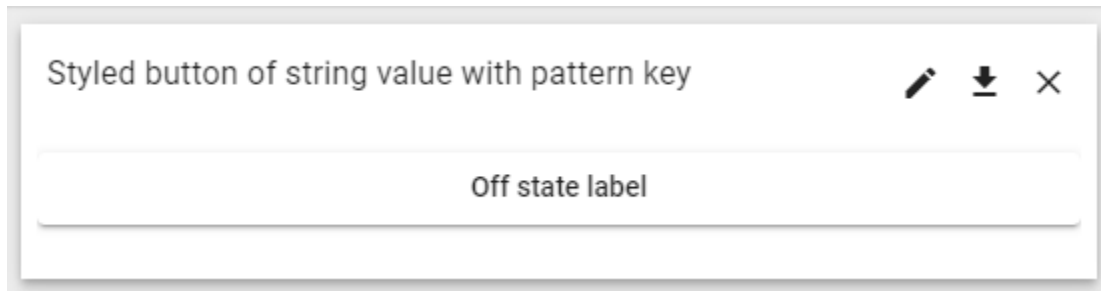
### Styled button of string value

Control widget.



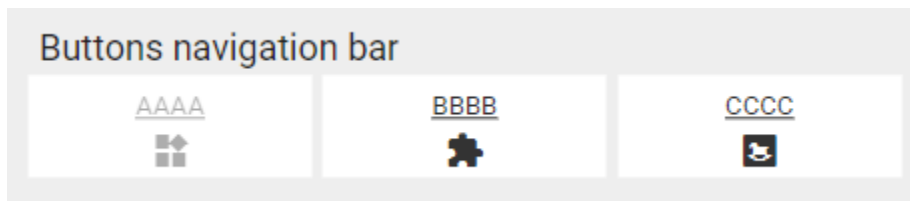
### Styled button of string value with pattern key

Control widget.



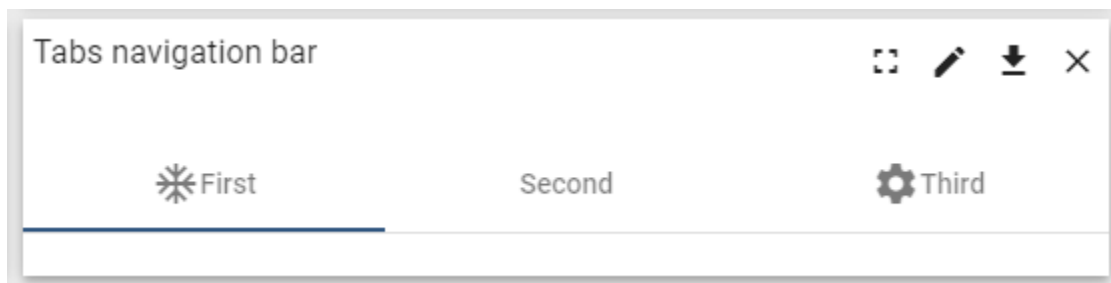
### Buttons navigation bar

Control widget.



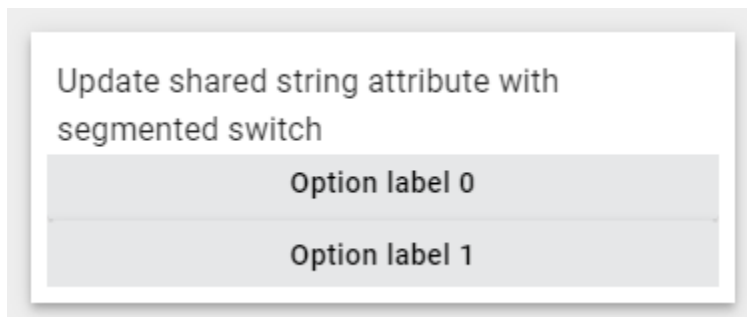
### Tabs navigation bar

Control widget.



### Update shared string attribute with segmented switch

Latest values widget.





### Update time value

Control widget.

### Update time value with pattern key

Control widget.

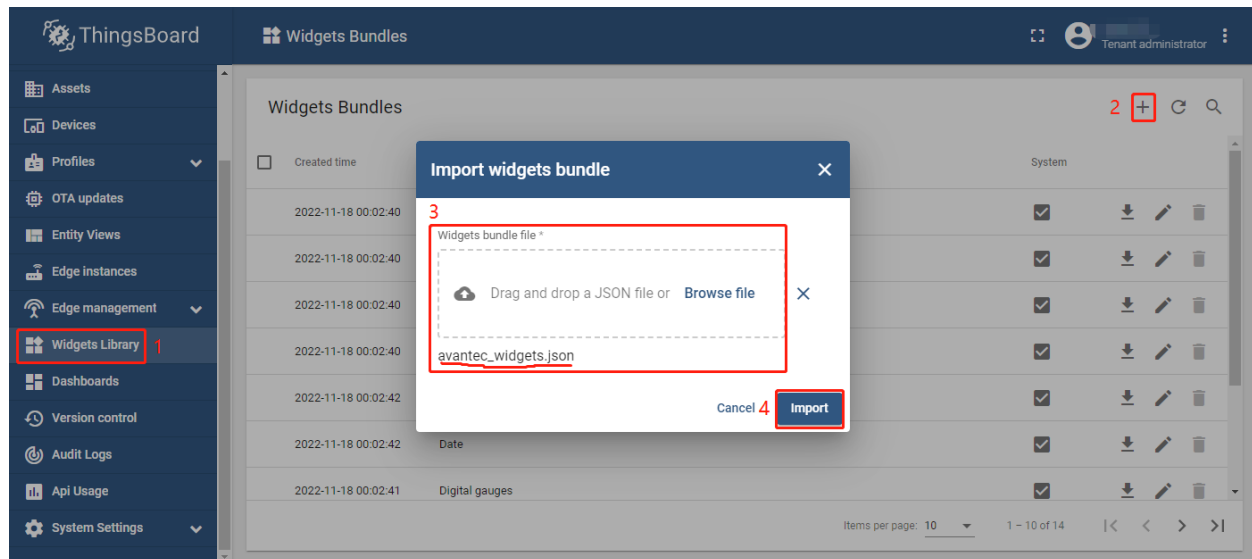
Some parameters of this widget can be appended with a suffix for programming time.

## 3.1.2 Import Avantec Widgets

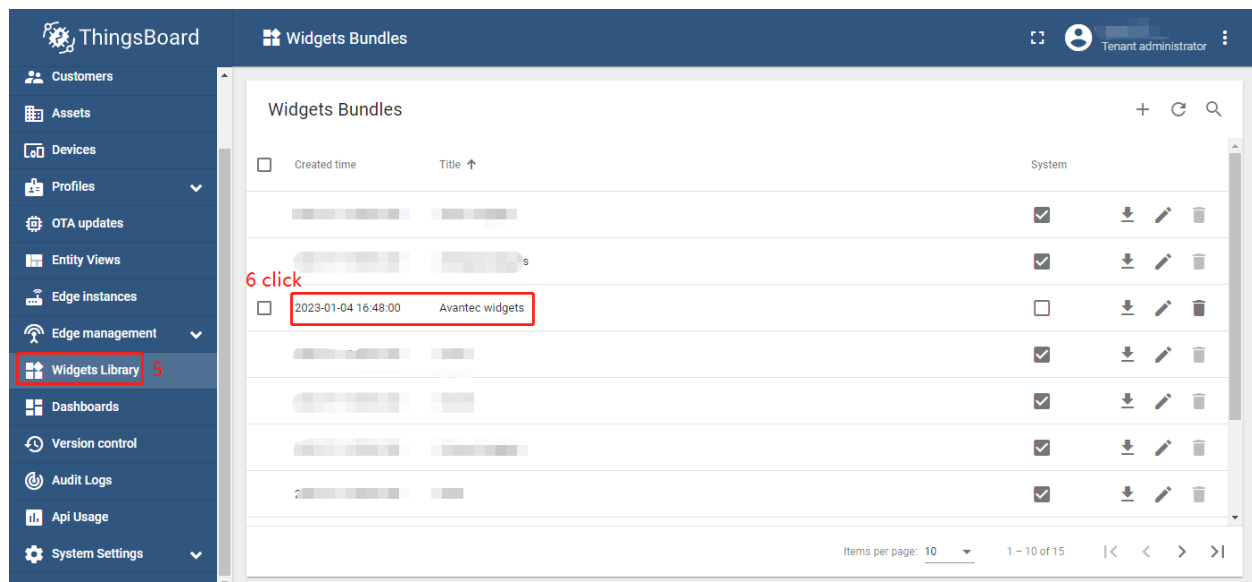
**Tip:** Avantec\_widgets.json can only be imported once. If you have already imported it, you do not need and cannot repeat the import.

If you have already imported it, you can skip this step or [Update Avantec Widgets](#).

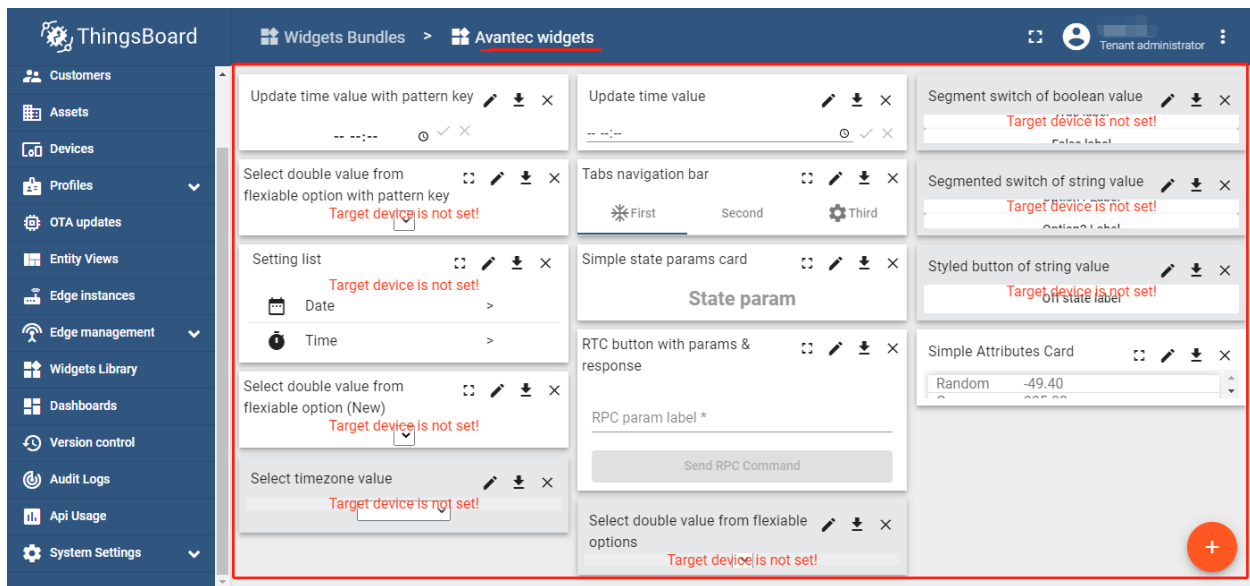
- Download avantec\_widgets.json.
- **Widgets Library** -> + -> **Import widgets bundle** -> **Popup dialog: Import widgets bundle** -> Drag and drop avantec\_widgets.json -> **Import**.




- Widgets Library → click Avantec widgets

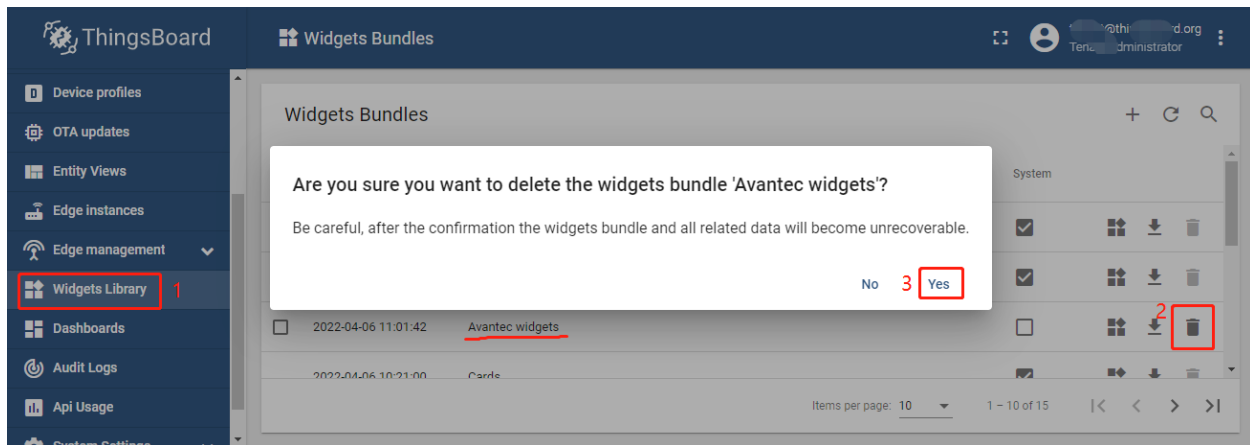


- All Avantec widgets



### 3.1.3 Update Avantec Widgets

- First, Delete Avantec Widgets: **Widgets Library** → Click  in the row of *Avantec widgets* → **Popup dialog: Are you sure you want to delete ...? → Yes.**



- Next, *Import Avantec Widgets*.

## 3.2 Demo Dashboards

### 3.2.1 TA652FC-W

- TA652FC-W Thermostat List* - device list.
- TA652FC-W Thermostat (For Mobile App)* - device detail.

### 3.2.2 TA652FH-W

- *TA652FH-W Thermostat List* - device list.
- *TA652FH-W Thermostat (For Mobile App)* - device detail.
- *Office center - TA652FH-W Thermostats* - multiple TA652FH-W thermostats in an office center.

### 3.2.3 TA692FC-L-5

- *TA692FC-L-5 Thermostat List* - device list.
- *TA692FC-L-5 Thermostat (For Mobile App)* - device detail.

### 3.2.4 Other Dashboards

- Thermostats - All Avantec Thermostas.
  - After import it, Entit aliases must be updated!
  - After import it, some action's targe dashboard must be update in Thermostats Cards - a widget of Entities Cards!

*TODO:...*

- Firmware - firmware update.

*Some versions of ThingsBoard have already installed this Dashboard, which can be used without importing. See [here](#).*

- *Manage device claiming* - ...

*TODO:...*

- *Claiming* - ...

*TODO:...*

## TA652FC-W WI-FI THERMOSTAT

These references will help you learn more about TA652FC-W Wi-Fi Thermostat, operate it, and even realize your personalized Dashboard.

- [\*Specification\*](#)
- [\*Add to ThingsBoard | Connect to ThingsBoard | Demo Dashboards\*](#)
- [\*MQTT Device API\*](#)

### 4.1 TA652FC-W – 2 pipe Fan Coil Wi-Fi Thermostat

**Caution:**

1. Turn off all electrical devices (e.g. heater, cooler) that are connected to the unit before installation and maintenance.
2. The installer must be a trained service personnel.
3. Disconnect the power supply before maintenance.
4. It must be mounted on a dry clean indoor place.
5. Do not expose this unit to moisture.
6. Do not expose this unit to dipping or splashing.

#### 4.1.1 Introduction

TA652FC-W is a controller that controls fan coil system to maintain room temperature at a desired level.

Changeover sensor is required to install when auto changeover is used.

### 4.1.2 Feature List

- Voltage supply: 230Vac
- Temperature display in °C or °F
- Temperature measurable range : 0 - 50 °C
- 2-pipe system
- Manual changeover or Auto changeover can be selected
- Selection of Heat/Cool
- 7days/5+1+1days, 1day program, no program.
- EEPROM stores all settings
- Adjustable control span

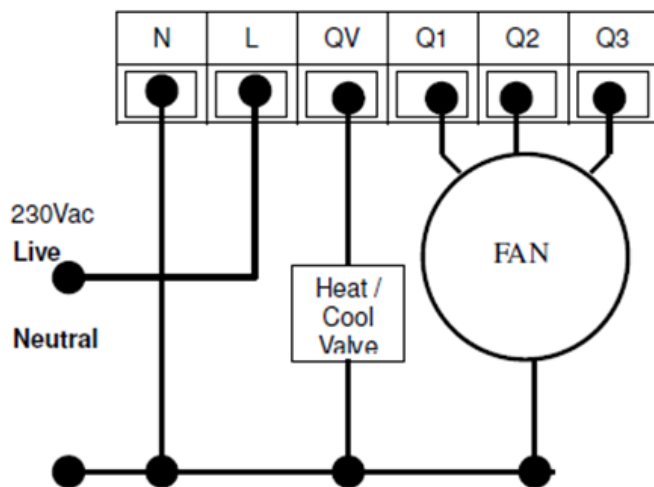
### 4.1.3 Wiring

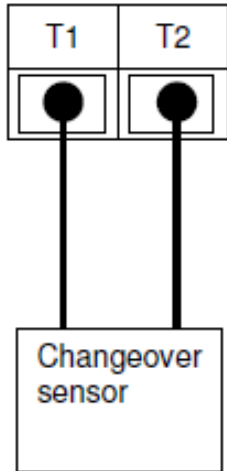
**NOTE:** Power supply of TA652FC-W is 230Vac.

Terminals	Device
L	230Vac Live
N	230Vac Neutral
Qv	Changeover valve
Q1	Fan Low
Q2	Fan Med
Q3	Fan High
T1	Floor Sensor
T2	Floor Sensor

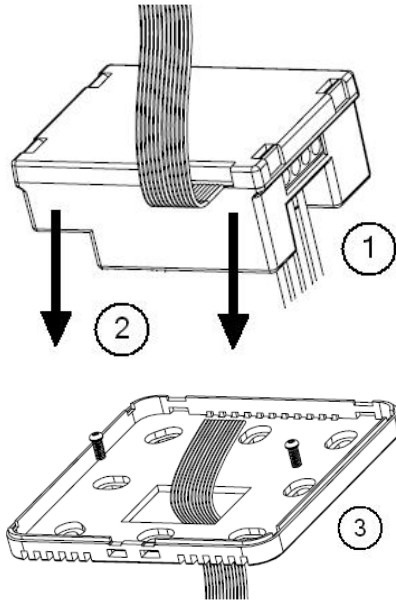
Pull all cables back into the wall beforehand to avoid trapping of wires. Do not use any metal conduits or cables provided with metal sheaths.

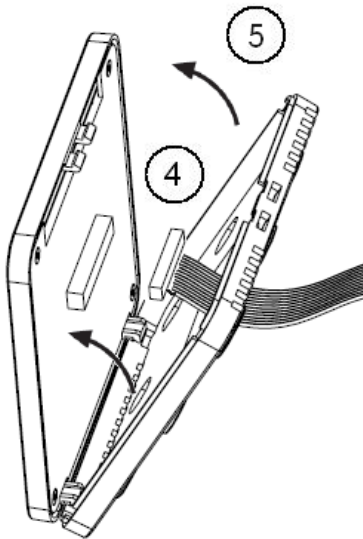
Recommend adding fuse or protective device in the live circuit.





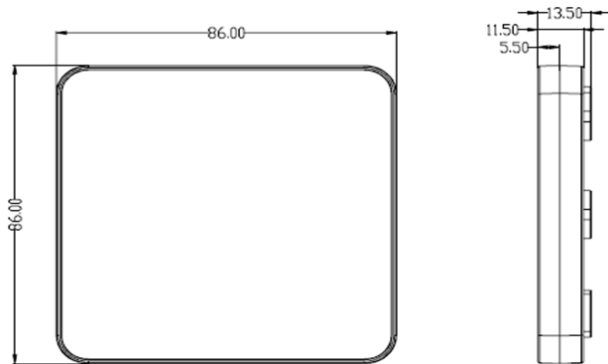
#### 4.1.4 Mounting



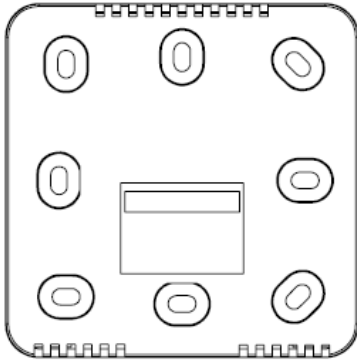


1. Wiring the terminals.
2. Put into junction box.
3. Mount the bottom plate of LCD board into junction box.
4. Connect the wire to the LCD board.
5. Assemble the Top and bottom plate of LCD Board.

#### 4.1.5 Dimension in mm:








### 4.1.6 LCD Interface

#### LCD Indication













sn	item
1	Time
2	Room Temperature
3	Current Set Point
4	Temperature Unit
5	Current Program
6	Heat / Cool Mode
7	Auto Changeover
8	Output is On (when appear)
9	Fan Low/Med/High/Auto
10	Wi-Fi (appear when connected to router)
11	Mode Key: Press to <b>internal setting 1</b> . Long hold to <b>internal setting 2</b> .
12	Clock Key: Press to <b>set clock</b> . Hold to <b>Program</b> the Schedule
13	Short Press: Fan Key, Long Hold: On/Off Key
14	Up/Down key: Adjust Set point or Value of setting.
15	Blank: the area outside of the previous five keys

### Turn On/Off the thermostat

Hold  to turn On / Off the thermostat. When the thermostat is Off. No Output will be activated.

### Clock setting

Normally the clock is automatically set once Wi-Fi is connected and synchronize for each day. So manual set is not necessary when it is online.











- Press  to start the setting
- Press  /  to change the day of week
- Press  again to confirm day of week setting and start to adjust hour
- Press  /  to change the hour
- Press  again to confirm hour setting and start to adjust minutes
- Press  /  to change the minutes
- Press  again to confirm minutes setting and start to adjust day of week
- Press [blank] to confirm or leave the clock setting. Or return after no key pressed for 20 seconds.

### Clock synchronization

When Wi-Fi is connected and time synchronization is need. Please use the App for time synchronization.



### Schedule Programming

When **1 day / 5+1+1 day / 7day program** is selected in internal setting.

- Hold  to start the setting.
- Press  /  to adjust the day of week
- Press  to confirm.
- Press  /  to adjust the time of schedule
- Press  to confirm.
- Press  /  to adjust the setpoint
- Press  to confirm.

- Press **[blank]** to confirm return.

## Override Temperature

The Set point can be adjusted by  / .




When it is in program mode, the set point will be overwritten until the next time slot.



can be pressed to release the override status.

## Internal parameter setting 1

### • Operation:

- Press  key to start the setting
- Press  /  to adjust the value
- Press **[blank]** to confirm and move to next setting

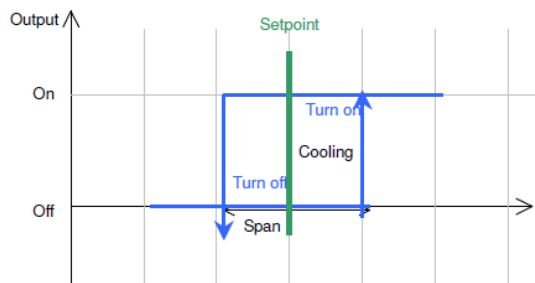
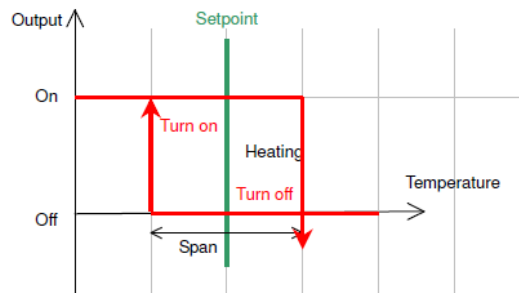
ID	Items	Value	Default Value
P00	User Interface Screen Saver	0-3	0
P01	Screen Saver Count down	0-120	20
P02	Display unit	°C / °F	°C
P03	Time Display unit	12/24	12
P04	Temperature Offset	-5°C - 5°C, -10°F-10°F	0°C
P05	Switching Differential Heat	2 - 4°C, 4 - 8°F	2°C
P06	Switching Differential Cool	2 - 4°C, 4 - 8°F	2°C
P07	Program mode	No program (0) / 1day program (1) / 5+1+1 program (2) / 7day program (3)	3
P12	Force Ventilation	Disable, Enable	Disable
P13	Changeover Mode	Heat, Cool, Auto	Heat
P14	Changeover temperature Heat	27 - 40°C	27°C
P15	Changeover temperature Cool	10-25°C	10°C

### • User Interface Screen Saver:

The thermostat will go to screen saver mode after no key for certain period.

- **Mode 0:** Nothing will be displayed in screen saver mode.
- **Mode 1:** Only room temperature will be displayed in screen saver mode.




- **Mode 2:** Room temperature and Time will be displayed in screen saver mode.
- **Mode 3:** Display all in screen saver mode.
- **Screen Saver Count Down:**  
The count down time (in seconds) to screen saver mode.
- **Display Unit:**  
Temperature unit in Celsius or Fahrenheit.
- **Time Display Unit:**  
12/24.
- **Temperature offset:**  
The temperature of internal sensor can be calibrated from  $-5^{\circ}\text{C}$  -  $+5^{\circ}\text{C}$  in case there is temperature difference between actual value and thermostat.
- **Switching Differential:**  
The difference between switching the heating or controller on and off



- **Program Mode:**
  - 0: **No Program** Mode, the thermostat control the temperature simply according to single setpoint.
  - 1: **1 day** program, the thermostat control the temperature according to single schedule.
  - 2: **5+1+1 day** program, the thermostat control the temperature according to 5 +1+1 schedule (Mon to Fri, Sat, Sun).
  - 3: **7 days** program, the thermostat control the temperature according to 7day program (individual program for each day).
- **Forced Ventilation:**
  - **Disable:** Fan will turn on only when heat/cool is on.
  - **Enable:** Fan keeps on (low) even heat / cool is off.
- **Changeover mode:**
  - 0: **Heat mode**

- 1: **Cool mode**
- 2: **Auto Changeover: When changeover sensor detect the temperature above changeover heat set point. Heat mode will be activated.**  
When changeover sensor detect the temperature below changeover cool set point. Cool mode will be activated.
- **Changeover heating setpoint:**  
Parameter for Auto Changeover mode.
- **Changeover cooling setpoint:**  
Parameter for Auto Changeover mode.

## Internal parameter setting 2

- **Operation:**
  - Hold  key to start the setting
  - Press  /  to adjust the value
  - Press [**blank**] to confirm and move to next setting

ID	Items	Value	Default Value
P19	Clear Wi-Fi Configuration	Yes or No	No
P20	Clear Parameter setting (restore default)	Yes or No	No

- **Clear Wi-Fi Configuration:**  
When set to yes, the SSID and Password stored in the thermostat will be cleared so another SSID and Password can be set again.
- **Clear Parameter setting:**  
When set to yes, all internal parameter setting will be restored to default value in next power on (reset)

## Technical Data

Power supply:	195-250 Vac
Relay Contact Voltage:	230Vac Max. 50/60 Hz
Relay Contact Current:	2A Max. for each
Sensing Element:	103AT
Terminals:	2 sq. mm Cable
Operating Temperature:	32 - 122 °F / 0 - 50 °C
Storage Temperature:	23 - 122 °F / -5 - 50 °C
Operating Humidity:	5-95%RHnon-condensing

## 4.2 Add TA652FC-W to ThingsBoard

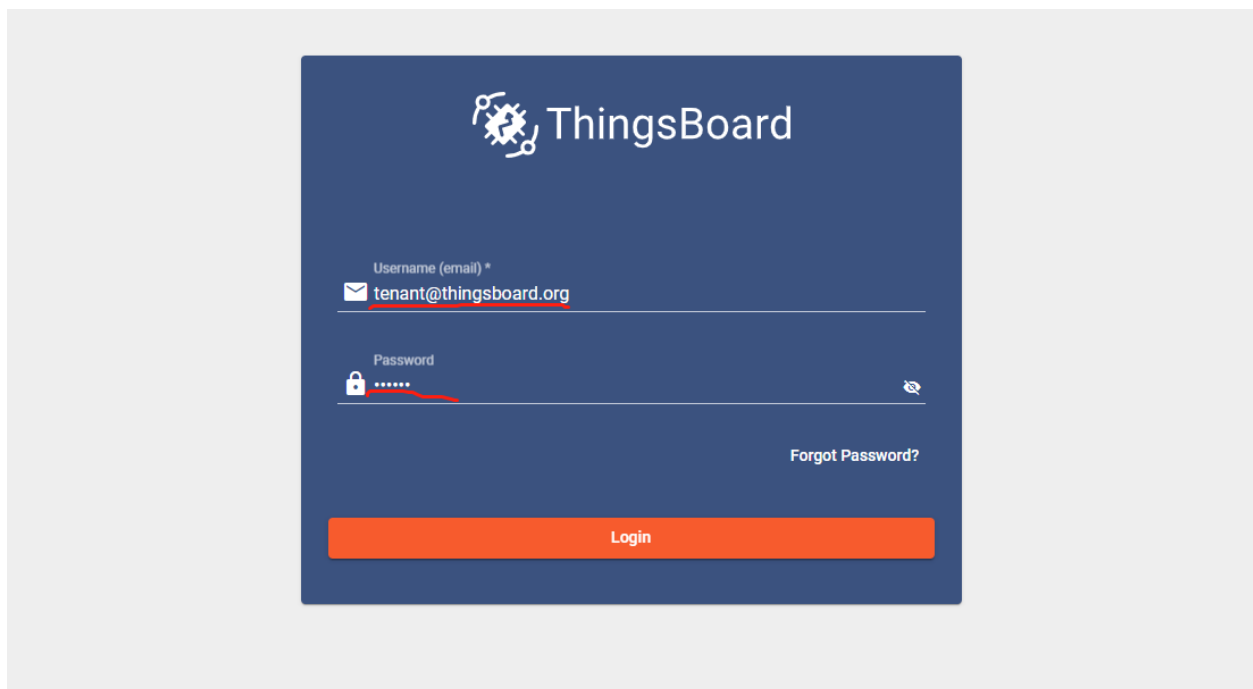
---

### Tip:

- This section applies to the situation where you add TA652FC-W to ThingsBoard Server.
  - If you are adding the first Avantec HVAC device to ThingsBoard Server, please refer to *Get Started*.
- 

### 4.2.1 Step 1. Tenant Login

- Open ThingsBoard Web UI in browser, e.g. <http://localhost:8080>
- Tenant Administrator login ThingsBoard.



Tenant default username and password, refer to *Some important parameters*.

### 4.2.2 Step 2. Import Detail Dashboard of TA652FC-W

See *Import TA652FC-W Detail Dashboard*.

### 4.2.3 Step 3. Import List Dashboard of TA652FC-W

See *Import TA652FC-W List Dashboard*.

### 4.2.4 Step 4. Provision TA652FC-W device

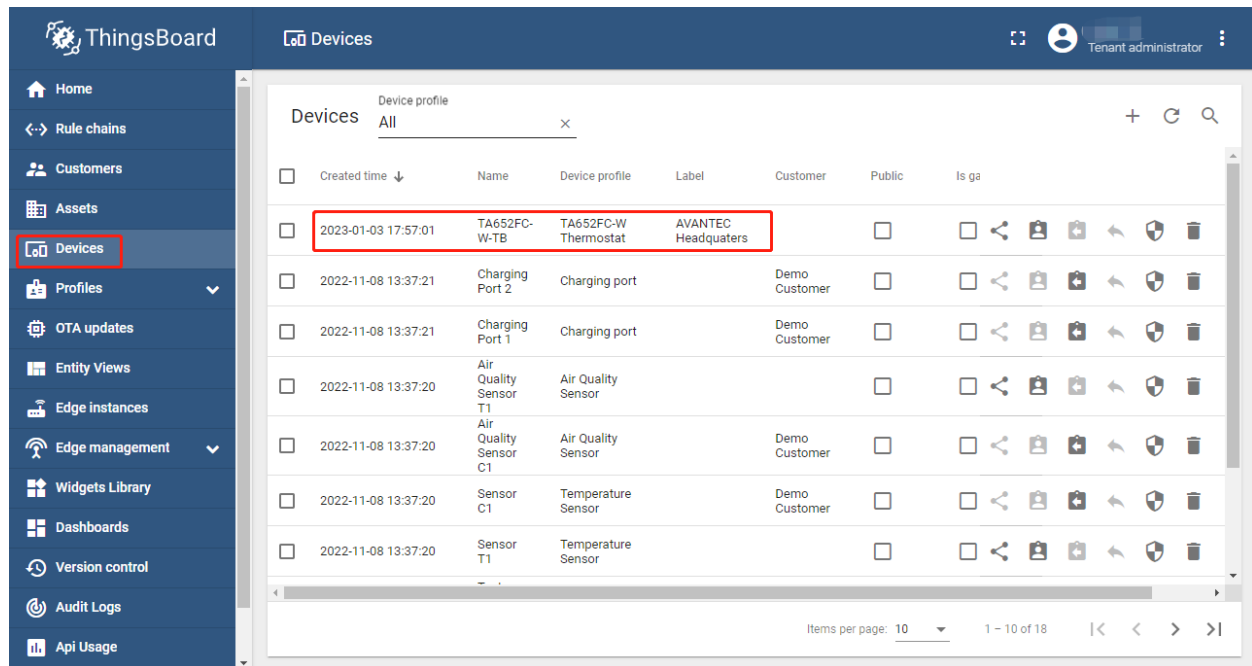
#### Step 4.1 Add device

- **Devices** -> **+** -> **Add new device** -> **Popup Dialog** -> Input **Name, Label & Description**, select device profile -> **Add**.

Field	Value
Name*	My device name, e.g. TA652FC-W-TB, A8:48:FA:57:60:A4
Device profile*	<b>TA652FC-W Thermostat</b>
Label	My device label, e.g. AVANTEC Headquarters
Description	My device description, e.g. A Thermostat for fan-coil

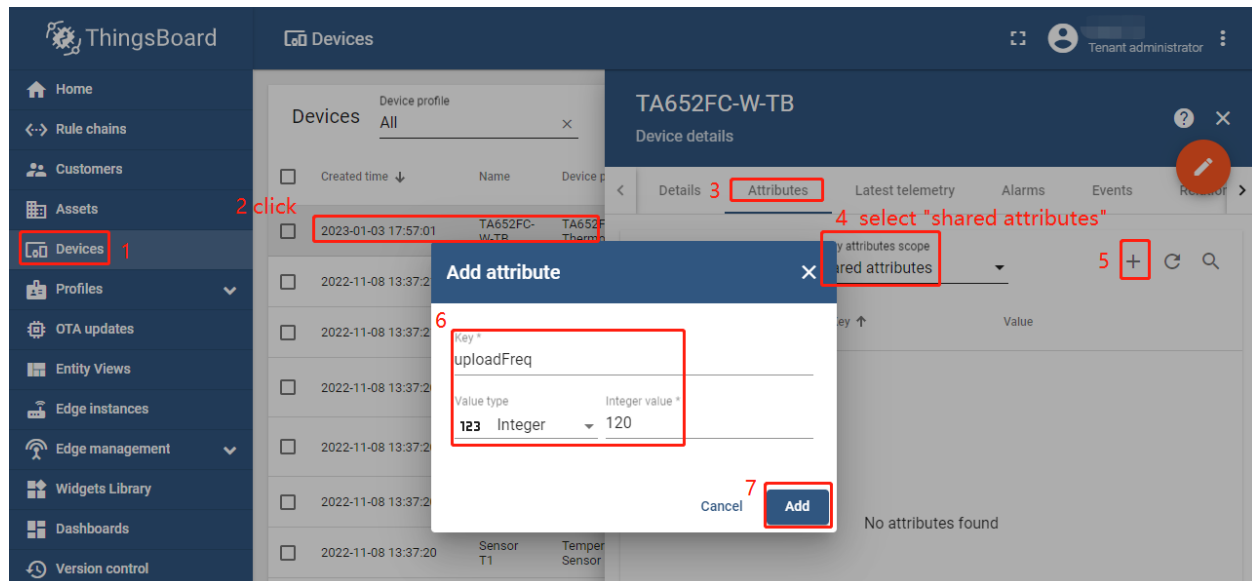
**Note:** The field with \* must be filled in.

- Now my device should be listed first, since the table sort devices using the time of the creation by default.



### Step 4.2 Add shared attributes of new device

- **Devices** → Click *my device* → **Attributes** → **Shared attributes** → + → **Popup Dialog** → Input Key, Value type & value → **Add**



Please add the following Shared attributes of **TA652FC-W**:



Table 1: Add shared attributes of TA652FC-W

Key*	Value Type*	Value*	Memo
<i>uploadFreq</i>	Integer	300	5*60. Telemetry per uploadFreq seconds
<i>uploadThresh-old</i>	Double	1.5	1.5°C. If the temprature (Telemetry data) change exceeds it, up-load immediately!
<i>syncTimeFreq</i>	Integer	86400	24*3600. Sync time per syncTimeFreq seconds
<i>timezone</i>	Integer	480	<b>Please replace with your value.</b> The time offset from UTC, minutes. For example Hongkong is UTC+8:00 time zone, this offset is 480 minutes (8*60)
<i>timeNTPServer</i>	String	pool.ntp.org	SNTP Server URL, e.g. pool.ntp.org, 0.pool.ntp.org, 1.pool.ntp.org, uk.pool.ntp.org, hk.pool.ntp.org, time.nist.gov, ...

**Note:** The field with \* must be filled in.

- Now the shared attributes of my device is like:

The screenshot shows the ThingsBoard interface with the 'TA652FC-W-TB' device selected. The 'Shared attributes' tab is active, showing a list of attributes with their keys and values. A red box highlights the 'Shared attributes' section.

Entity attributes scope	Key	Value
Shared attributes	Last update time	
Shared attributes	cloudHost	mqtt://demo.thingsboard.io
Shared attributes	syncTimeFreq	1800
Shared attributes	timeNTPServer	pool.ntp.org
Shared attributes	timezone	480
Shared attributes	uploadFreq	120

**You may also use:**

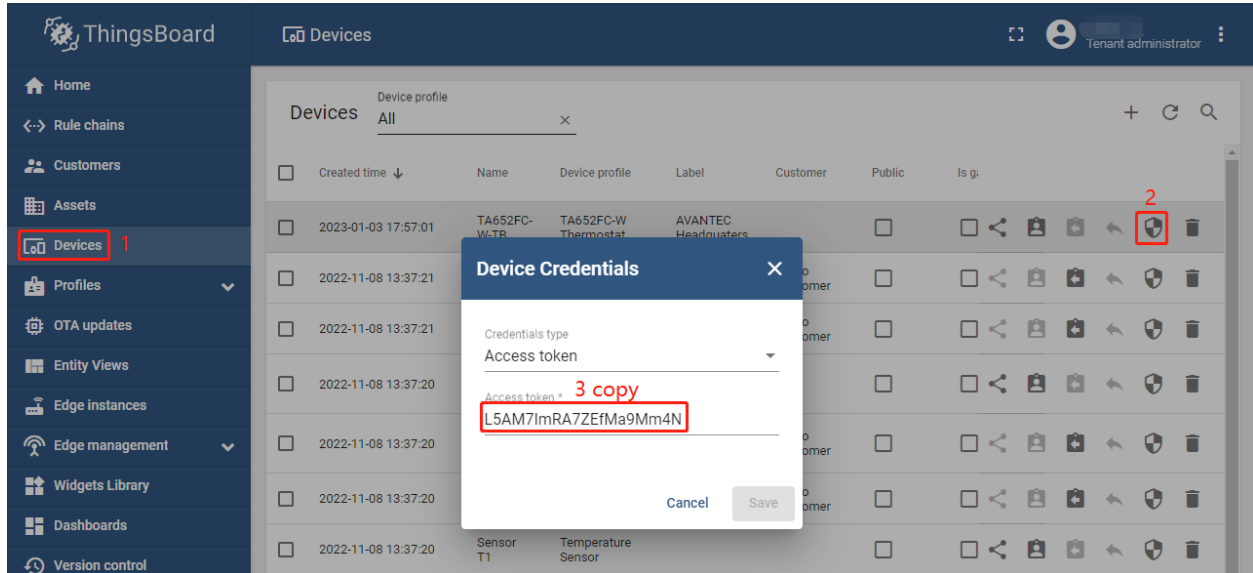
- **Bulk provisioning** to provision multiple devices from a CSV file using UI.
- **Device provisioning** to allow device firmware to automatically provision the device, so you don't need to configure each device manually.
- **REST API** to provision devices and other entities programmatically.

## 4.2.5 Step 5. Connect TA652FC-W device

### Step 5.1 Copy credentials of new device

To connect the device you need to get the device credentials first. ThingsBoard supports various device credentials. We recommend using default auto-generated credentials which is access token for this guide.

- **Devices** → **Manage credentials (icon)** → **Popup Dialog** → **Select Access Token, Ctrl + C**.



**Tip:** The Credentials (Access Token), which you need to use when you're configuring your hardware, for example, *j9JiCkID9E7uE1WhKxnc, lMTQLZ7VSRQSD7ls*.

### Step 5.2 Connect device to ThingsBoard

See *Connect TA652FC-W to ThingsBoard*.

### Step 5.3 Publish data to ThingsBoard

Now your device has already published telemetry data to ThingsBoard. You should immediately see them in the Device Telemetry Tab:

The screenshot shows the ThingsBoard interface with the 'Devices' tab selected in the left sidebar (1). In the 'All' device list, the device 'TA652FC-W-TB' is highlighted (2). The 'Latest telemetry' tab is selected in the top navigation bar (3). The 'Latest telemetry' table shows the following data (4):

Last update time	Key	Value
2023-01-03 18:21:04	current_fw_title	TA652FC-W-TB
2023-01-03 18:21:04	current_fw_version	1.6.8
2023-01-03 18:23:06	iram	148616
2023-01-03 18:23:06	roomTemp	18.3
2023-01-03 18:23:06	spiram	4192139
2023-01-03 18:23:06	wifiRssi	-39

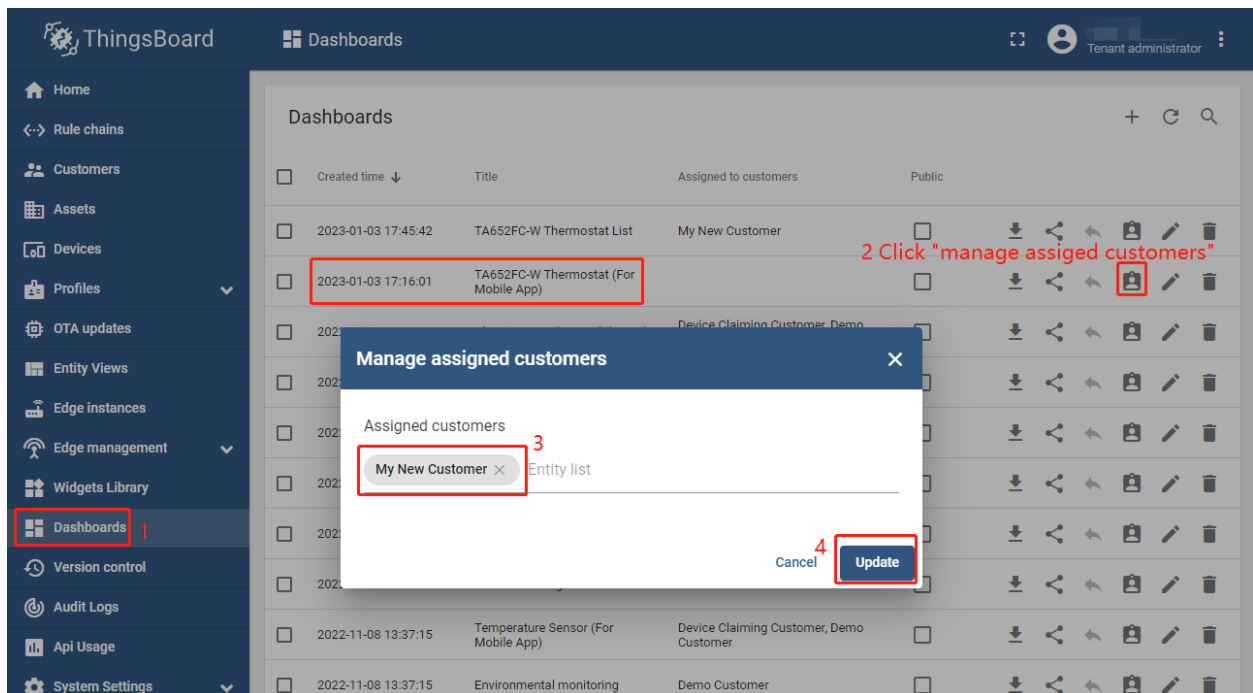
## 4.2.6 Step 6. Assign Device and Dashboards to Customer

One of the most important ThingsBoard features is the ability to assign Dashboards to Customers. You may assign different devices to different customers. Then, you may create a Dashboard(s) and assign it to multiple customers. Each customer user will see his own devices and will not be able to see devices or any other data that belongs to a different customer.

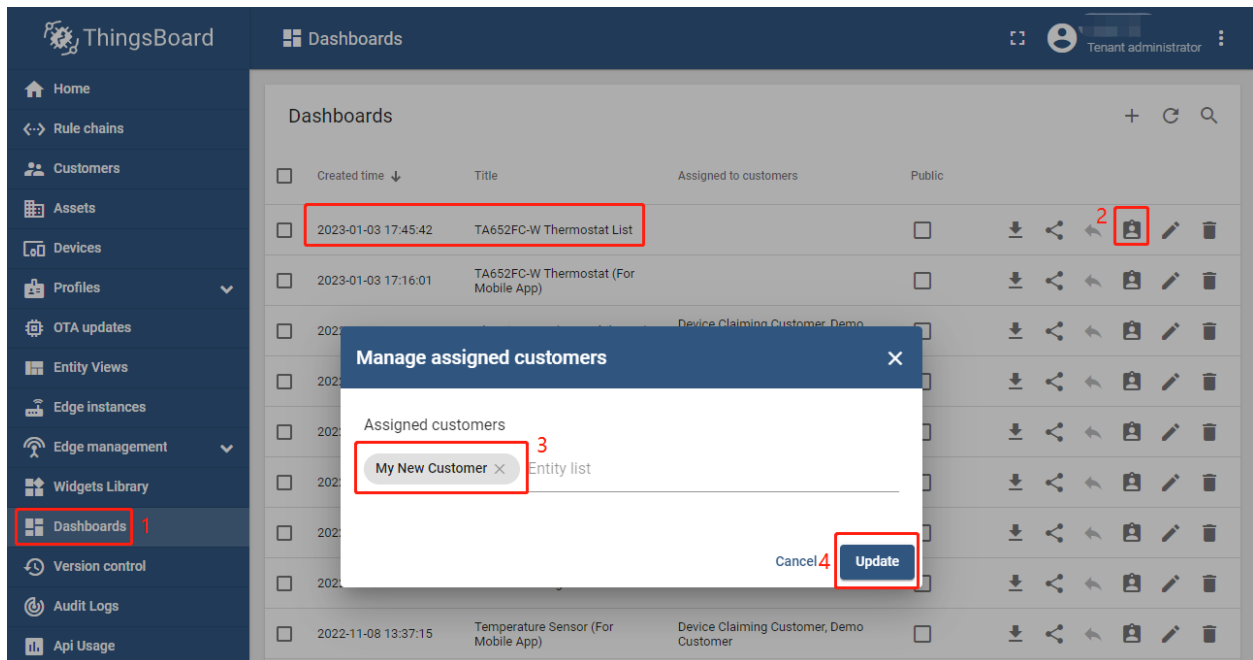
Refer to *Step 7.1 Create customers*, *Step 7.4 Create customer user* & *Step 7.5 Activate customer user*.

### Step 6.1 Assign dashboards of TA652FC-W to Customer

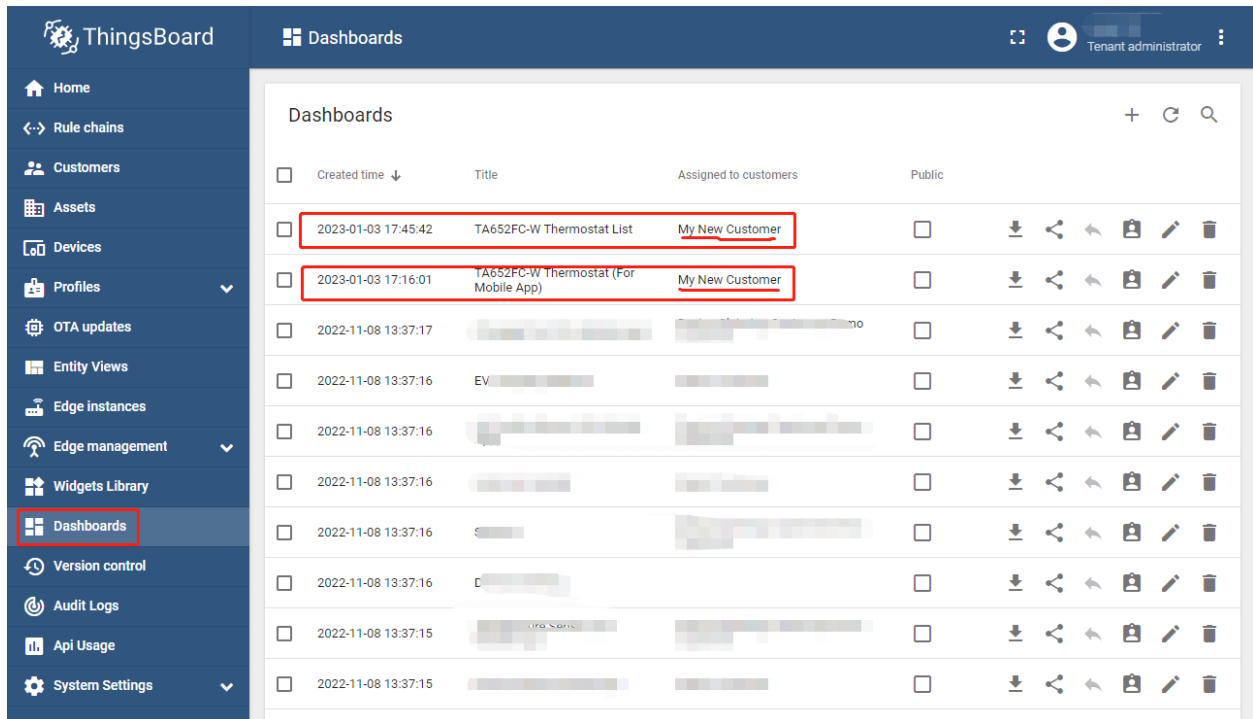
- Assign *Detail dashboard* to Customer: **Dashboards** → Click **Manage assigned customers** (icon) in *Detail dashboard* line → **Pop-up Dialog** → Select *My New Customer* → **Update**.



- Assign *List dashboard* to Customer: **Dashboards** → Click **Manage assigned customers** (icon) in *List dashboard* line → **Popup Dialog** → Select *My New Customer* → **Update**.

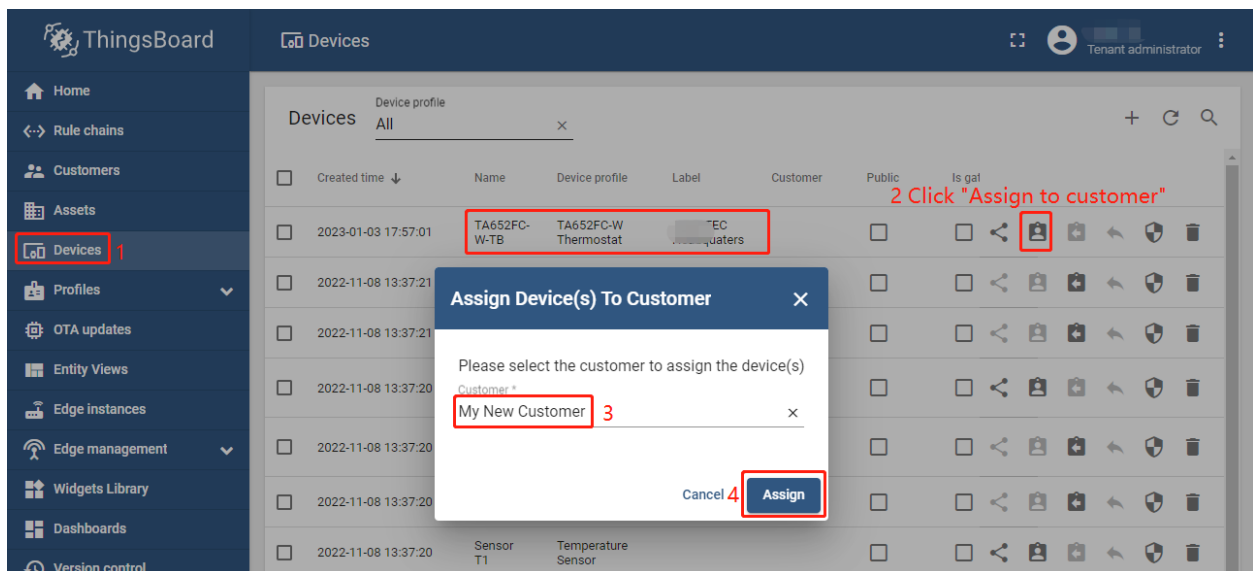


- It's like this now.

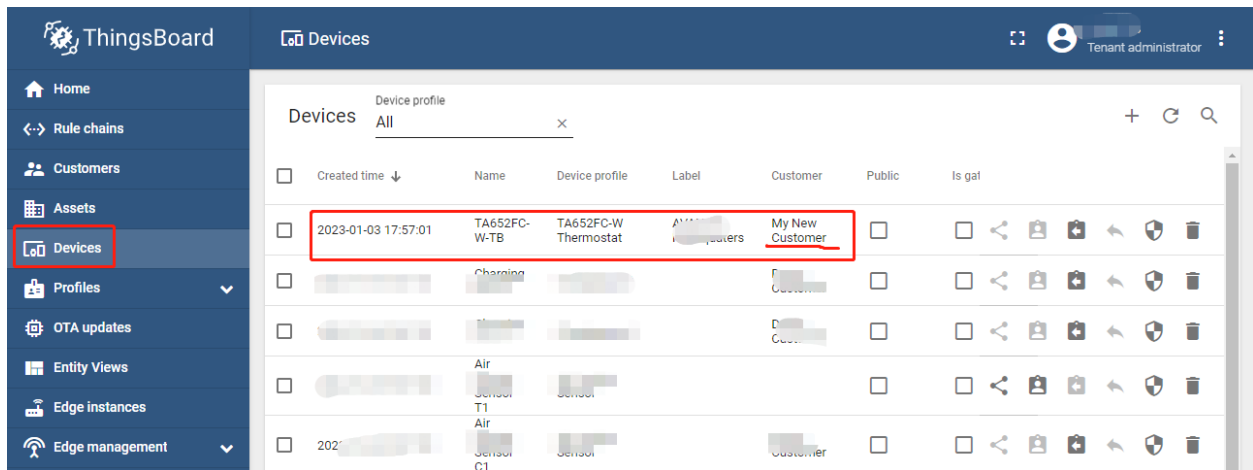


### Step 6.2 Assign TA652FC-W device to Customer

- **Devices** → Click **Assign to customers** (icon) in *My New Device* line → **Popup Dialog** → Select *My New Customer* → **Assign**.

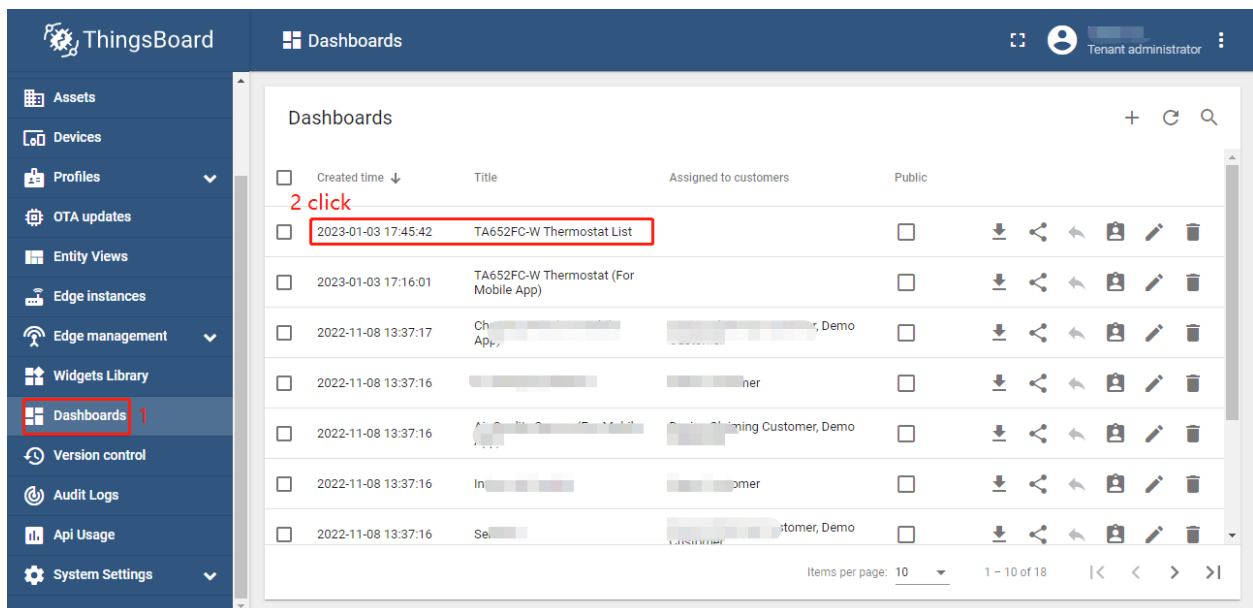


- It's like this now.

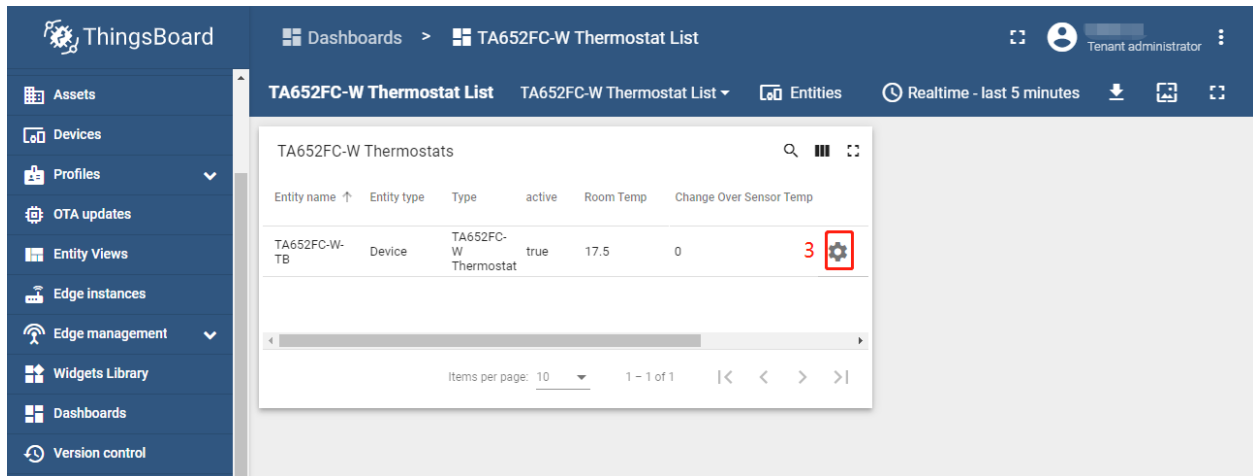


### 4.2.7 Step 7. Open Dashboards of TA652FC-W

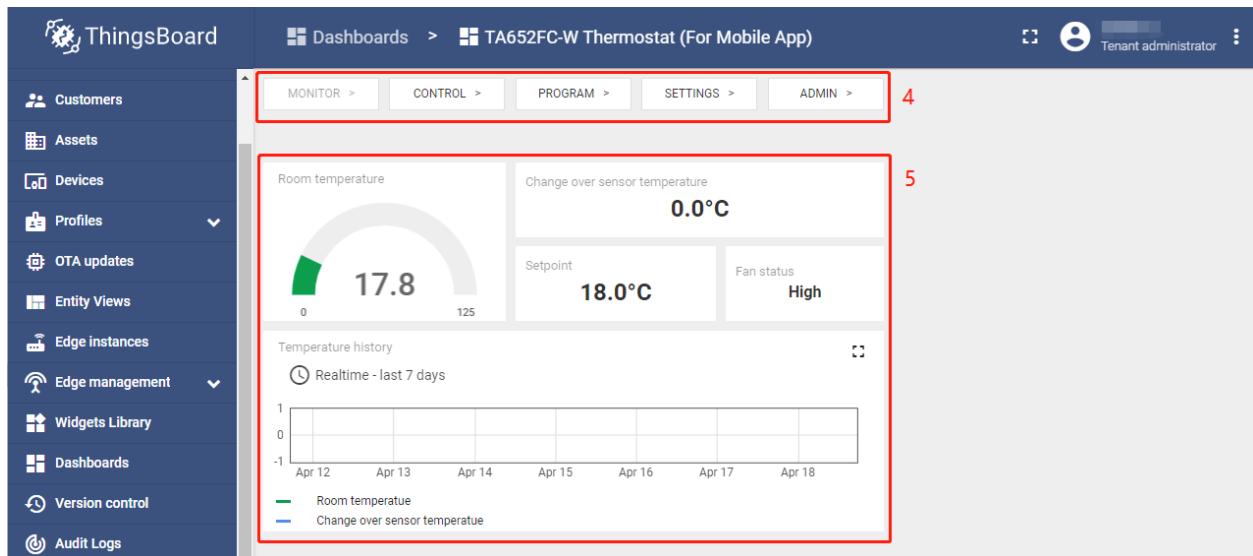
- You are logged in as a Customer User or a Tenant user.
- **Dashboards** → click *my list dashboard*



- Select my device → **Settings** (icon)



- Switch page → Operation



See *TA652FC-W Demo Dashboards Usage*.

## 4.2.8 Your feedback

Don't hesitate to star Avantec on [github](#) to help us spread the word.

## 4.3 Connect TA652FC-W to ThingsBoard

### Tip:

- This section applies to both TA652FC-W and TA652FH-W.
- Unless otherwise specified, all specifications applicable to TA652FC-W are also applicable to TA652FH-W.



This section describes the process of connecting TA652FC-W to ThingsBoard via Wi-Fi. This process applies to all models of thermostats in this series.

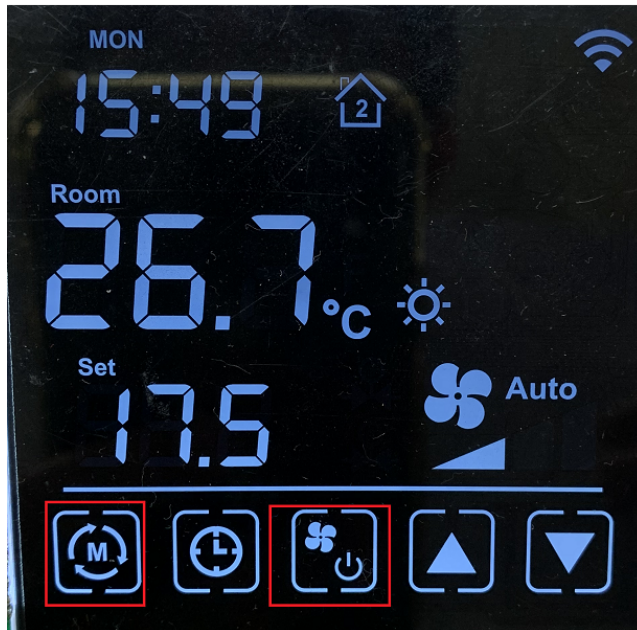
### 4.3. Connect TA652FC-W to ThingsBoard

### 4.3.1 Prerequisites. Clear Wi-Fi Configuration

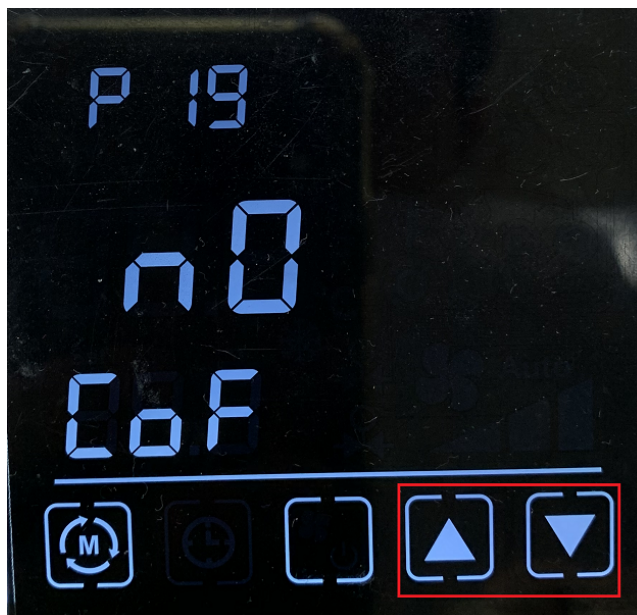
**Tip:** If your TA652FC-W is used for the first time, or your TA652FC-W has never been connected to any Wi-Fi router, you can skip this step.

If your TA652FC-W has been connected to a Wi-Fi router before, when you need to connect to a new Wi-Fi router, you need to clear the Wi-Fi configuration of the TA652FC-W first.

- Press and hold  and  simultaneously for 10 seconds on the TA652FC-W.




- Enter Wi-Fi parameter clearing mode P19.





- Press  or  to select *YES*.



- Press  to return to the normal interface, and the Wi-Fi parameters are cleared.

#### 4.3.2 Step 1. Get Access-Token

Get a access-token of TA652FC-W from ThingsBoard. See [Step 5.1 Copy credentials of new device](#).

#### 4.3.3 Step 2. Power On

When you first power up, TA652FC-W will enter Wi-Fi AP mode without any Wi-Fi parameters. At this point, you can configure the parameters through the web page.

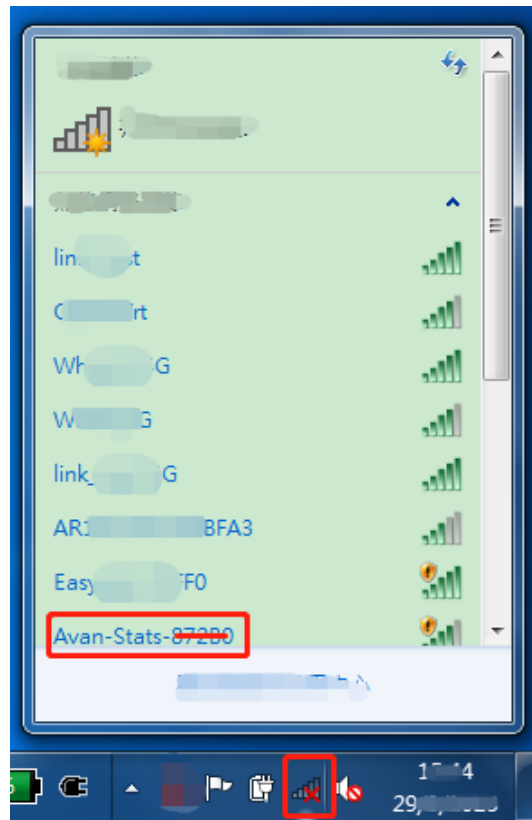
---

**Tip:** TA652FC-W has a different Wi-Fi Hotspot name every time it's powered on.

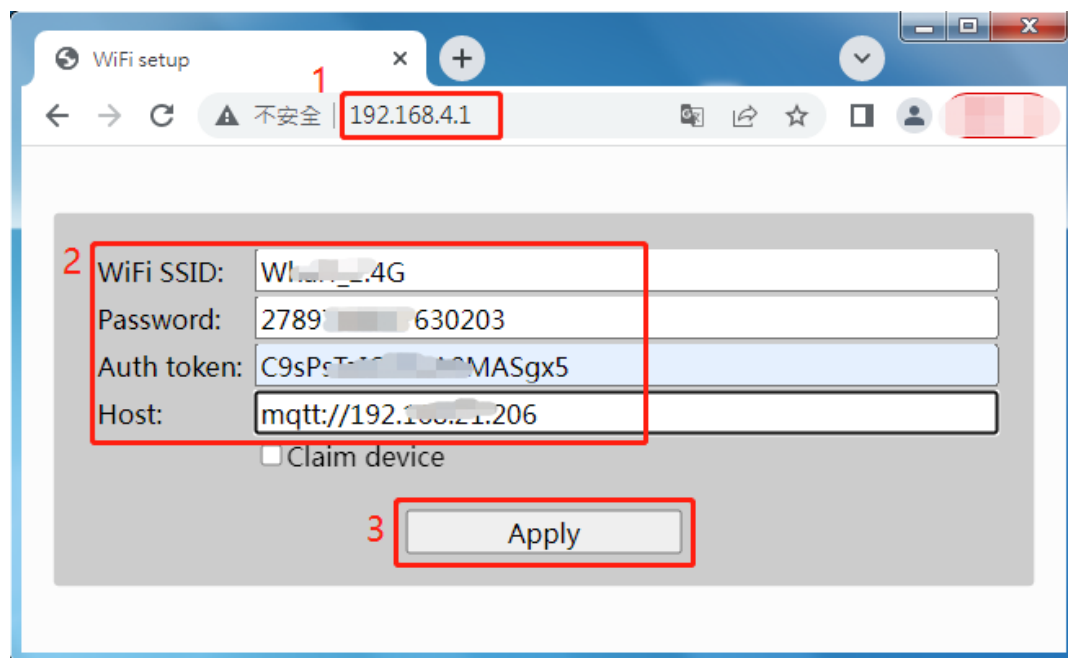
---

#### 4.3.4 Step 3. Configure

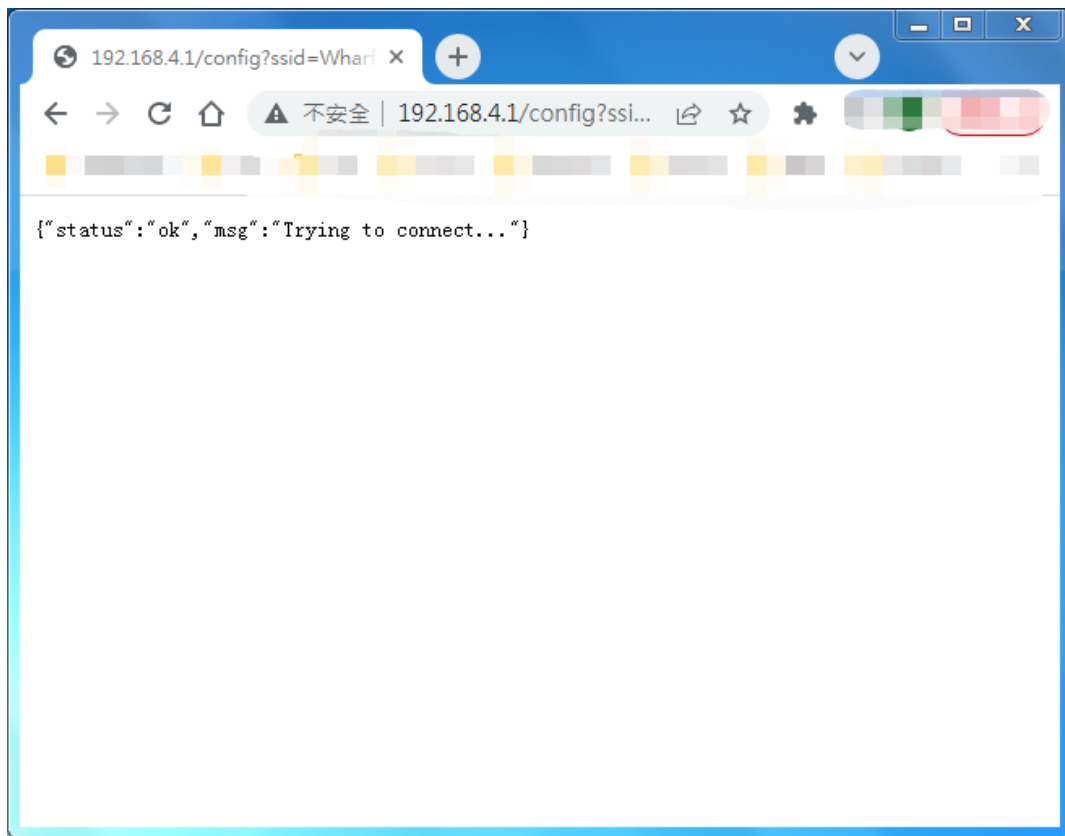
- Connect to TA652FC-W's Wi-Fi hotspot on your computer or smart phone. It's like Avan-Stats-CEBD8.



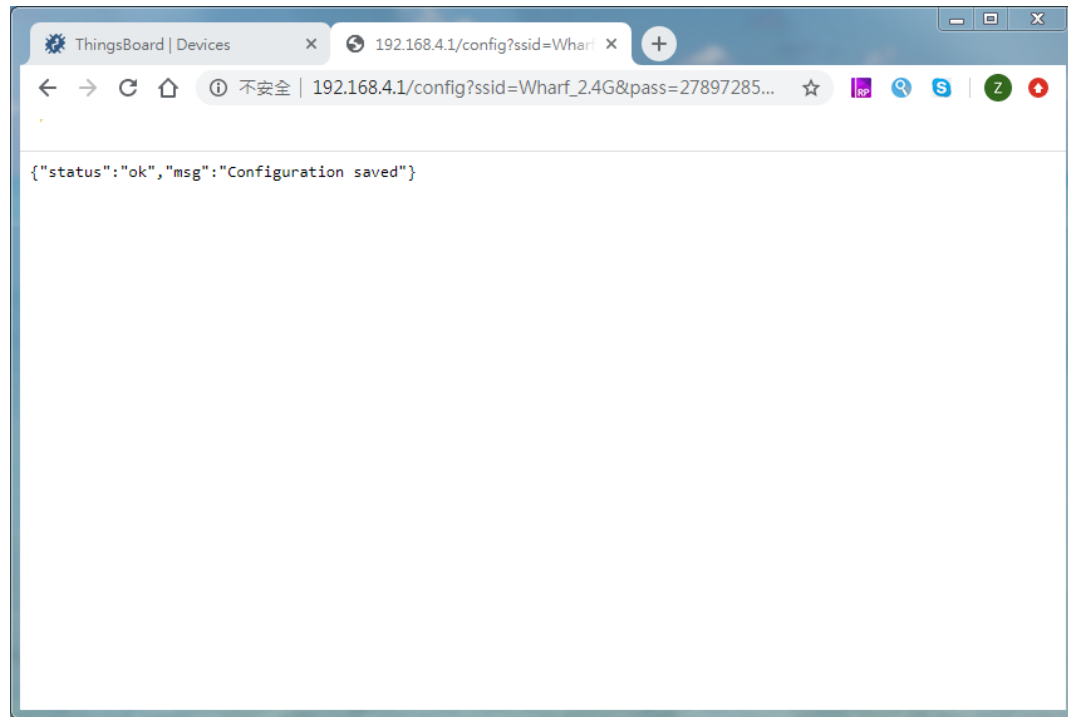
- Open your browser, type `http://192.168.4.1`.
- Input your configuration, then Apply.



Field	Description
Wi-Fi SSID	SSID of your Wi-Fi router
Password	password of your Wi-Fi router
Auth Token	Access Token of your TA652FC-W. See <a href="#">Step 1. Get Access-Token</a>
Host	<p>This ThingsBoard Server's MQTT URL. It must begin with "mqtt://", such as "mqtt://192.168.21.206"</p> <p><b>Please replace 192.168.21.206 with your ThingsBoard IP Address.</b></p> <p>See <a href="#">Step 4.2 Add shared attributes of new device</a></p>



- If the configuration is successful, the following screen *may be* displayed.



#### 4.3.5 Step 4. Check

Check if TA652FC-W is connected to ThingsBoard correctly. If connected correctly, there will be a Wi-Fi icon in the upper right corner of the Thermostat, and the time will no longer be **12:00**. If you do not set the tone zone relationship on ThingsBoard correctly, the time displayed by TA652FC-W may be slightly off.



### 4.3.6 Troubleshooting

Thermostat TA652FC-W can't connect to Wi-Fi:

- If the Thermostat has never been connected to any Wi-Fi router since leaving the factory, it will enter Soft-AP mode. You can search for Wi-Fi SSID similar to “Avan-Stats-CEBD8” through your mobile phone or computer.
- Make sure the Wi-Fi router supports and turns on the 2.4G signal. Currently, some dual-band (2.4G & 5G) Wi-Fi routers can turn off the 2.4G signal. Please turn it on in your router settings.
- Make sure your Wi-Fi SSID and Password are correct, and they are related parameters of 2.4G Wi-Fi signal.
- Confirm that the Token is normal.
  - Confirm that the Token corresponds to the actual model (the Token of TA652FH-W-TB can only be connected to the Thermostat of TA652FH-W-TB. The same is true for TA652FC-W-TB).
  - Confirm that the Token did not fail during the copying process.
  - Confirm that the Token has no special characters. Token can only contain A-Z, a~z, 0~9. Illegal characters such as “-” will appear in the case of product end. You can edit and get a new Token in [Step 1. Get Access-Token](#).
- Confirm that the *Host* parameter is correct. Host must start with “mqtt://”, followed by IP address or domain name of ThingsBoard.
- If the above parameters are confirmed to be correct, you can start from [Step 2. Power On](#) and try several times.

## 4.4 TA652FC-W Thermostat – Demo device profile usage

### 4.4.1 Import device profile

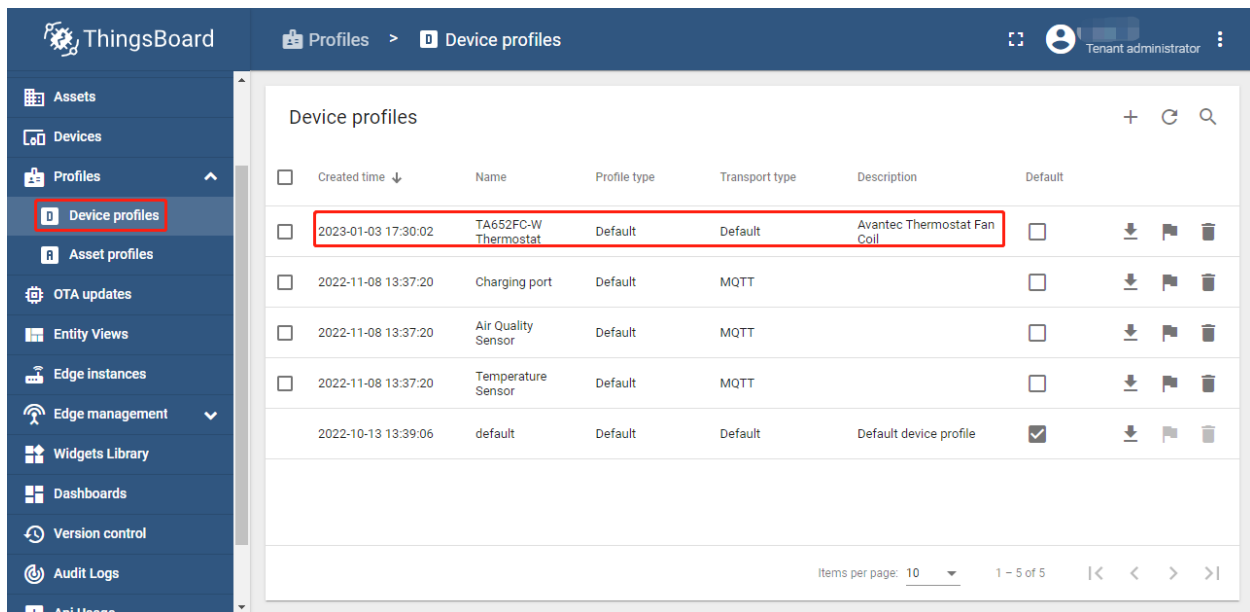
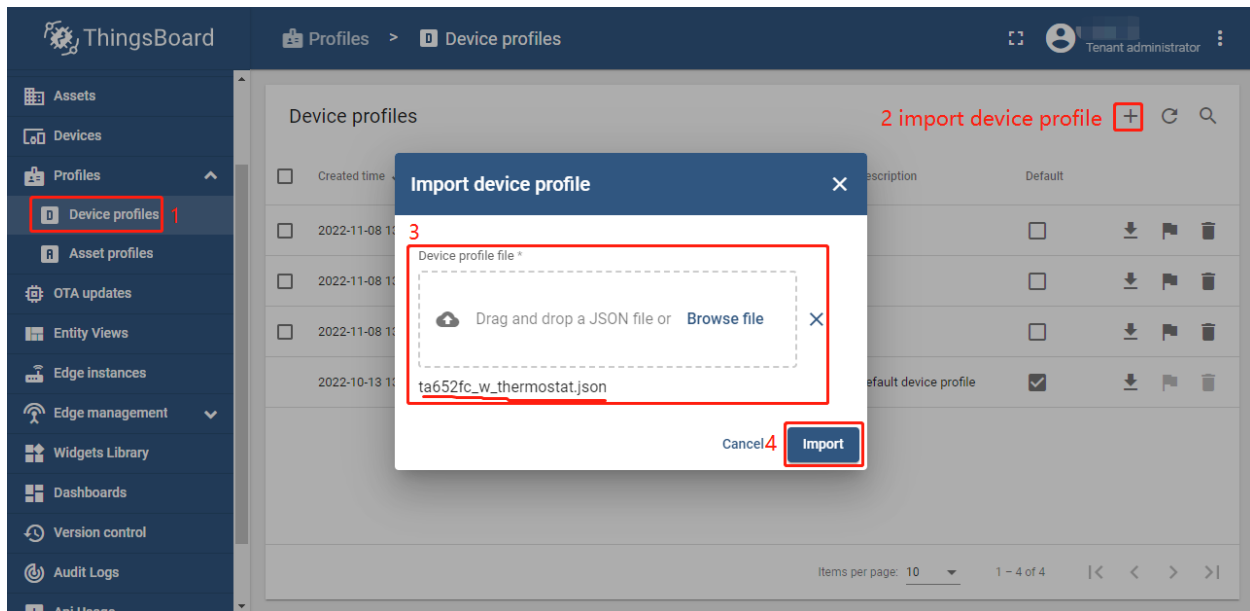
---

**Tip:** A *Device Profile file* can only be imported once. If you have already imported it, you do not need and cannot repeat the import.

If you have already imported it, you can skip this step.

---

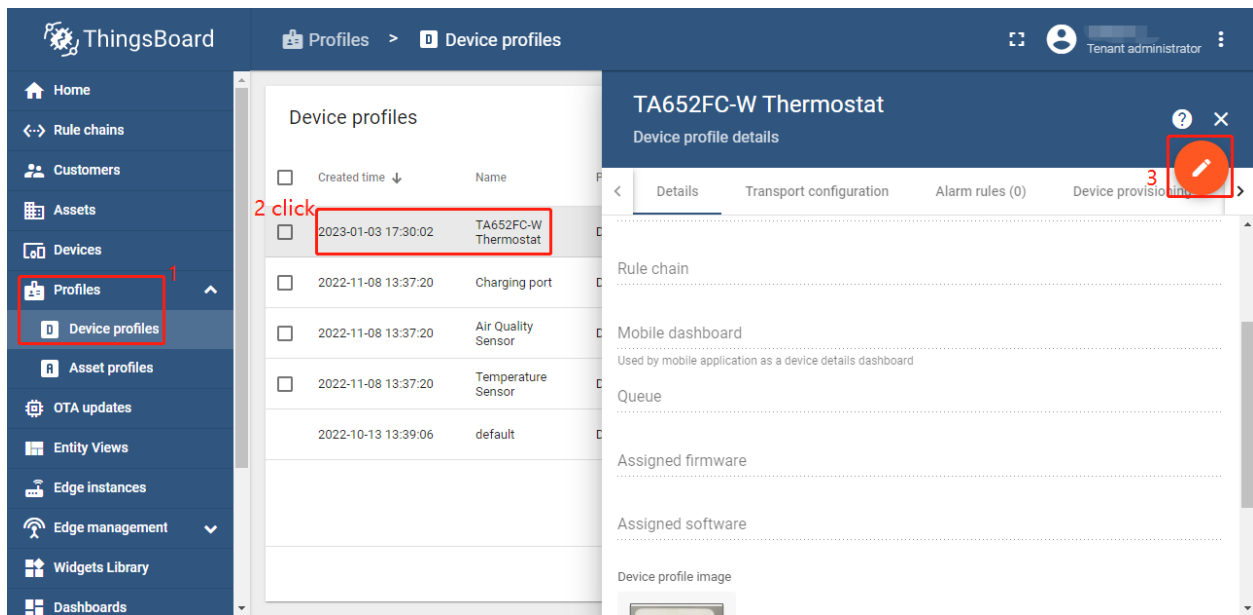
- Download `ta652fc_w_thermostat.json`.
- **Profiles** → **Device profiles** → + → **Popup dialog: Import device profile** → Drag and drop *my device profile File* → **Import**.



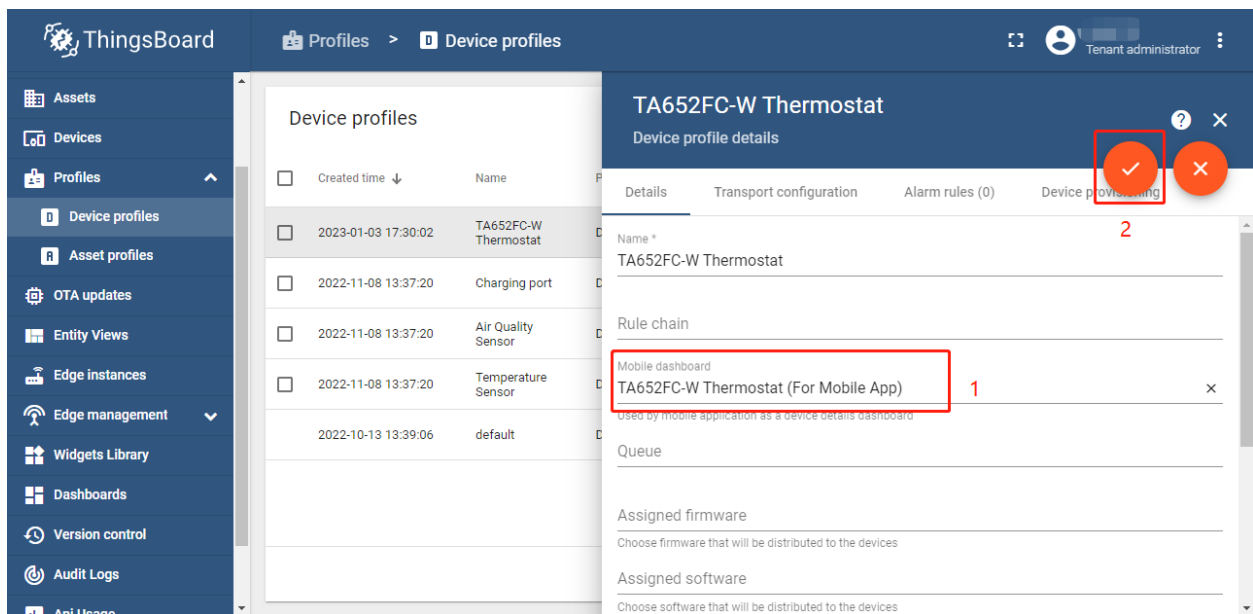
### 4.4.2 Modify device profile's mobile dashboard

Device profile's mobile dashboard is for ThingsBoard Mobile Application or ThingsBoard PE Mobile Application.

- Profiles → Device profiles → click my device profile → Toggle edit mode (red icon)



- Modify *Mobile dashboard* → **Apply changes** (red icon)



These values are shown in the following table:

Field	Value
Mobile dashboard	TA652FC-W Thermostat (For Mobile App)

### 4.4.3 Clear device profile's mobile dashboard

Sometimes if TA652FC-W Thermostat device profile's mobile dashboard is cleared, TA652FC-W Thermostat (For Mobile App) can only be deleted.

- **Profiles** → **Device profiles** → click *my device profile* → **Toggle edit mode** (red icon)

The screenshot shows the ThingsBoard web interface. On the left sidebar, the 'Profiles' menu is expanded, and 'Device profiles' is highlighted with a red box and a red '1'. In the main panel, the 'Device profiles' table is visible. The row for 'TA652FC-W Thermostat' is highlighted with a red box and a red '2'. To the right, the 'TA652FC-W Thermostat' details panel is open. In the top right corner of this panel, there is a red circle with a white pencil icon, labeled with a red '3', which is the 'Toggle edit mode' button.

- Clear *Mobile dashboard* → **Apply changes** (red icon)

The screenshot shows the ThingsBoard web interface with the 'TA652FC-W Thermostat' details panel in edit mode. In the 'Mobile dashboard' section, the text 'TA652FC-W Thermostat (For Mobile App)' is highlighted with a red box and a red '1'. To the right of this text is a red 'X' icon. Above this section, there is a red circle with a white checkmark icon, labeled with a red '2', which is the 'Apply changes' button.



## 4.5 TA652FC-W Demo Dashboards Usage

### 4.5.1 Overview

There are two dashboards related to TA652FC-W, namely TA652FC-W Thermostat List and TA652FC-W Thermostat (For Mobile App). We open the former to start operating TA652FC-W.

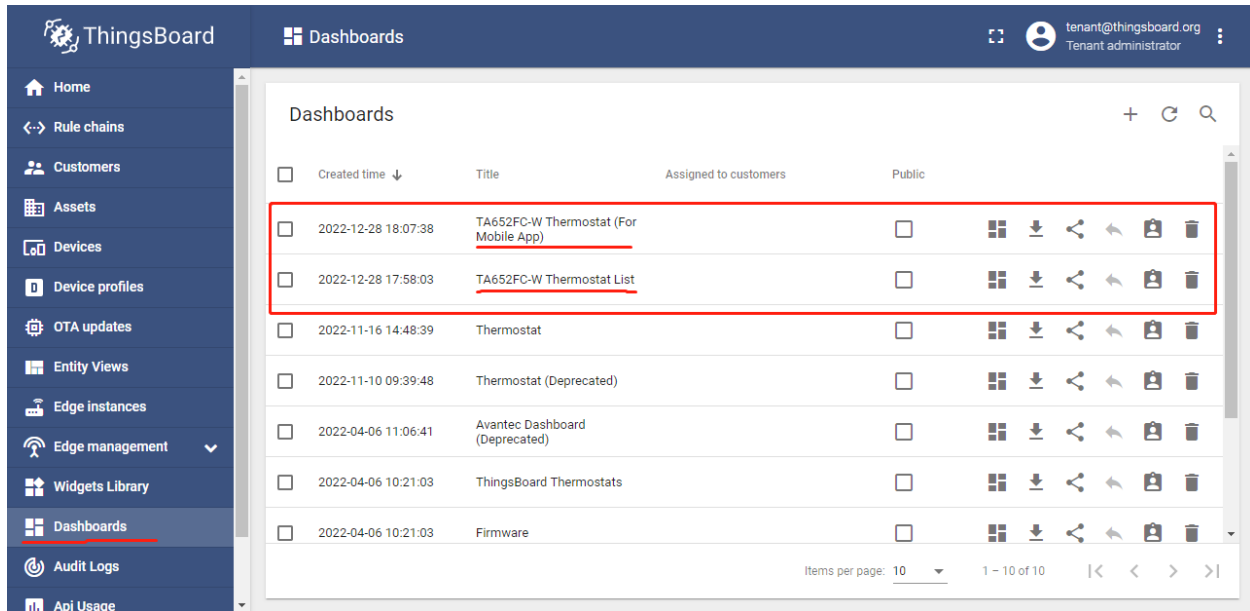


Table 2: TA652FC-W Demo Dashboards

Dashboard	Description	For Web UI	For Mobile App	Entry*
TA652FC-W Thermostat List	list	Yes	No	Yes
TA652FC-W Thermostat (For Mobile App)	details	Yes	Yes	No

#### Hint:

- If *Entry* is *Yes*, then directly enter the Dashboard and there will be data displayed.
- If *Entry* is *No*, there will be no data display when entering this Dashboard directly, and you need to jump to this Dashboard from other Dashboards.

### 4.5.2 TA652FC-W Thermostat List

#### Dashboard states

#### Default state

Default state is root state.

ThingsBoard

Dashboards > TA652FC-W Thermostat List

TA652FC-W Thermostat List

Entities

Realtime - last 5 minutes

TA652FC-W Thermostats

Device name ↑	Label	Type	Active	Room Temp	Change Over Temp	Setpoint	Fan status	Unit
24:0A:C4:2C:EB:D4	old_boardxxx	TA652FC-W Thermostat	true	19	0	21.5	Low	°C
9C:9C:1F:18:72:40	87240	TA652FC-W Thermostat	false	24.2	0	16	High	°C
9C:9C:1F:18:72:80	right	TA652FC-W Thermostat	false					
9C:9C:1F:18:72:84	87284	TA652FC-W Thermostat	false	16.9	0	21	High	°C
9C:9C:1F:19:4D:98	left	TA652FC-W Thermostat	false					
F0:08:D1:43:1A:E4	Hilary	TA652FC-W Thermostat	false	21	0	8.5	Off	°C

Items per page: 10

1 - 7 of 7

Powered by Thingsboard v3.5.1

### • Dashboard bar:

- **TA652FC-W Thermostat List** : Click here to skip to **root state**. Since **default state** is *root state*, click here and there is no response.
- : Click the two ICONS in the upper left corner to display the page in full screen.
- **Realtime - last 5 minutes** : Edit time window.

### • Thermostats widgets:

#### – Fields:

- \* Device name, Label, Type, active.
- \* Room temperature, Change Over Sensor Temperature, Setpoint, Fan status, Unit: Refer to *Monitor state*.

#### – Actions:

- \* : skip to *TA652FC-W Thermostat (For Mobile App)*.
- \* : Popup dialog to editing a device's label.

## Import List Dashboard

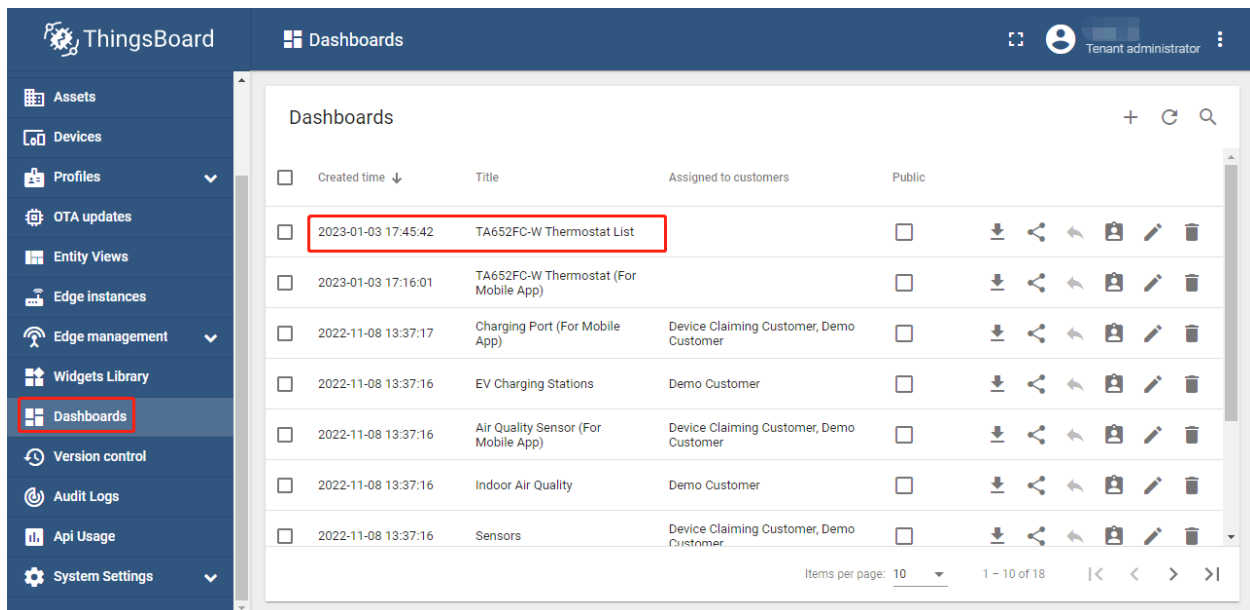
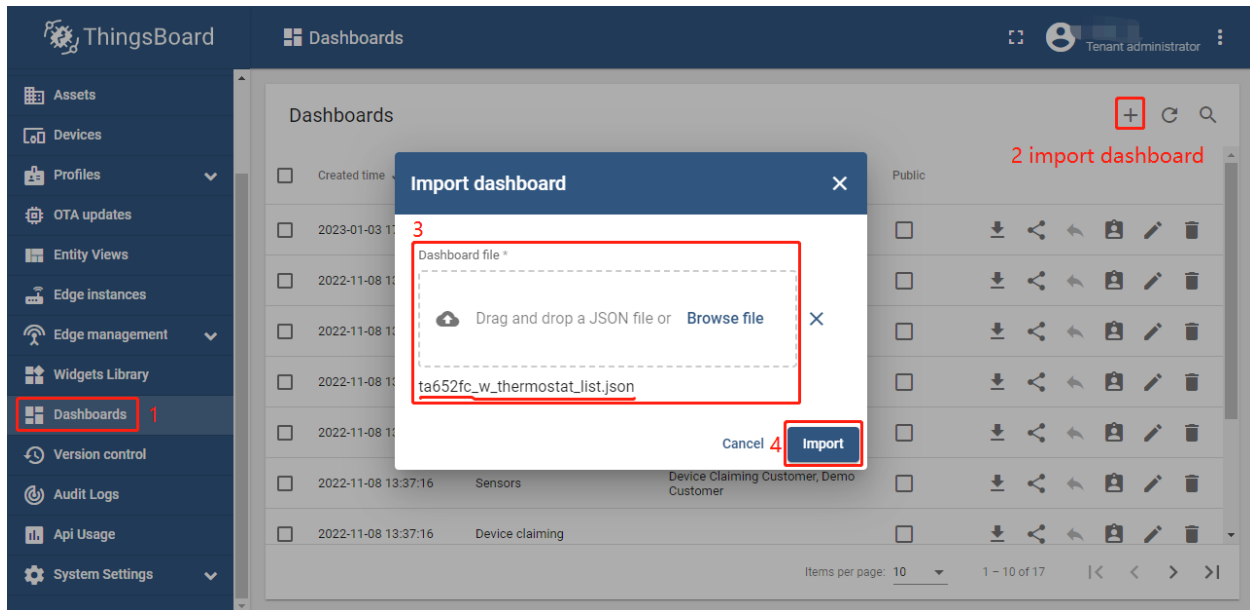
**Tip:** A *Dashboard file* can only be imported once. If you have already imported it, you do not need and cannot repeat the import.

If you have already imported it, you can skip this step.

In order to use this dashboard, you must to create *TA652FC-W Thermostat Device Profile* and *TA652FC-W Thermostat (For Mobile App)*. If they don't exist, you can import them. See *Import Device Profile of TA652FC-W Thermostat* or *Import TA652FC-W Detail Dashboard*.

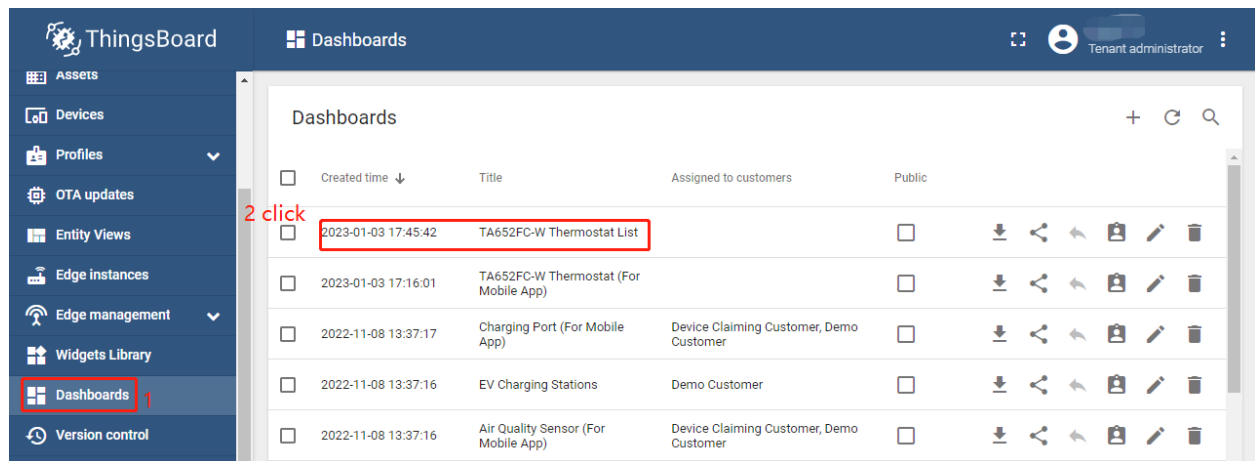
First, you can import this dashboard.

- Download `ta652fc_w_thermostat_list.json`.
- **Dashboards** → + → **Popup dialog: Import dashboard** → Drag and drop *list dashboard File* → **Import**.

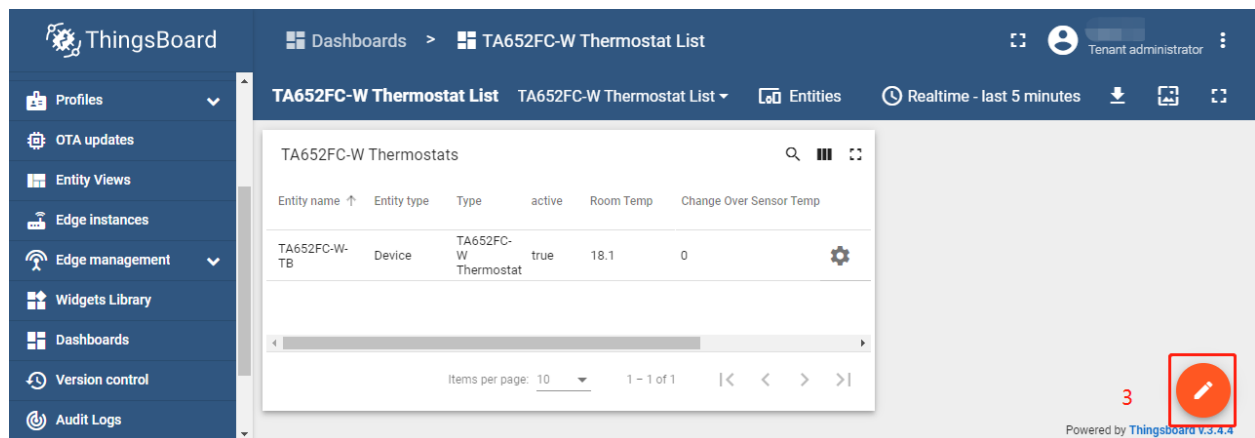


Next, modify a action's target dashboard and target dashboard state.

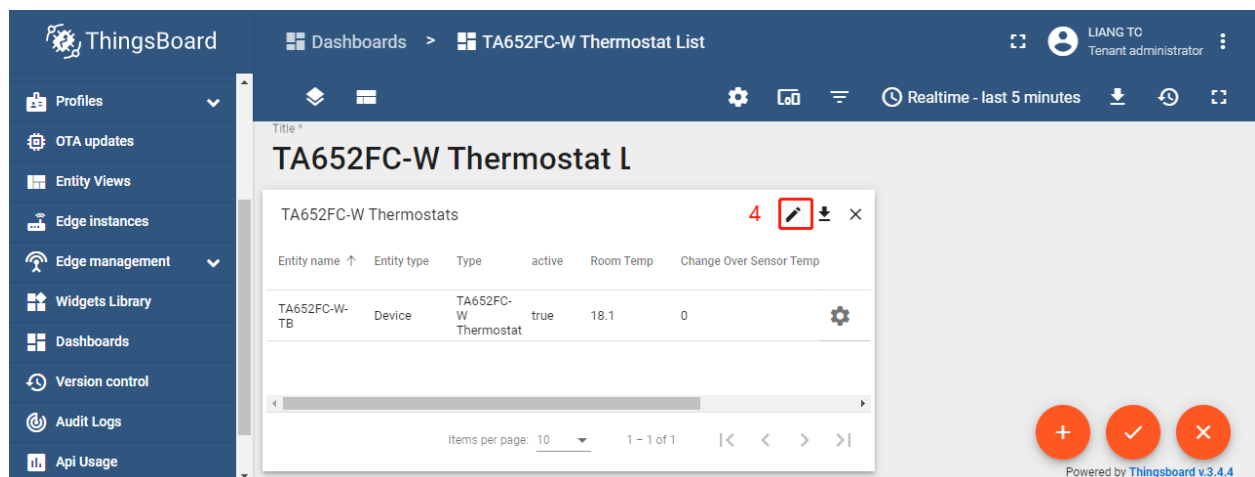
- **Dashboards** → Click *my list dashboard*



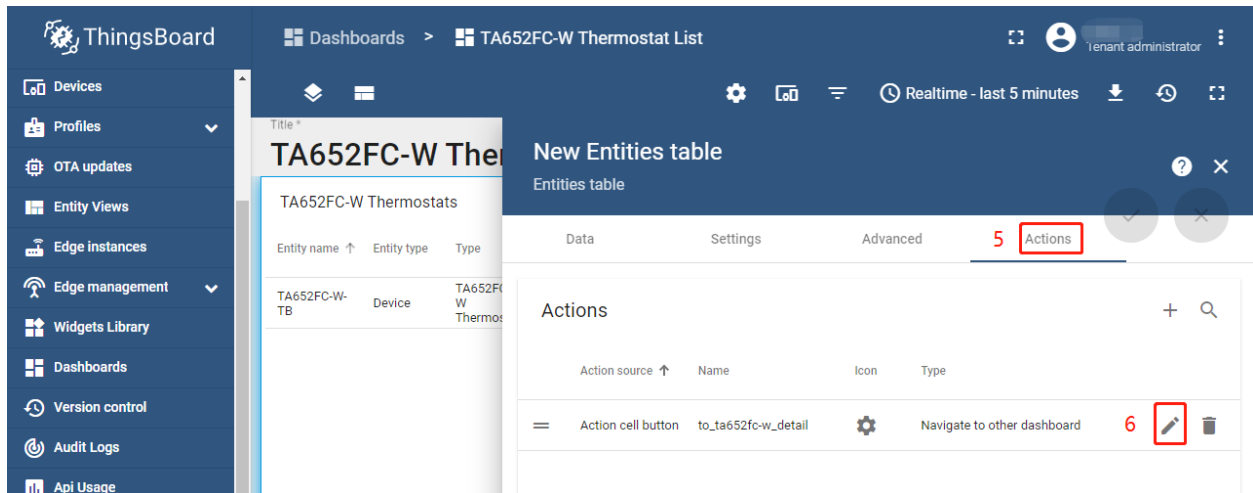
- **Edit** (red icon on the bottom and right)



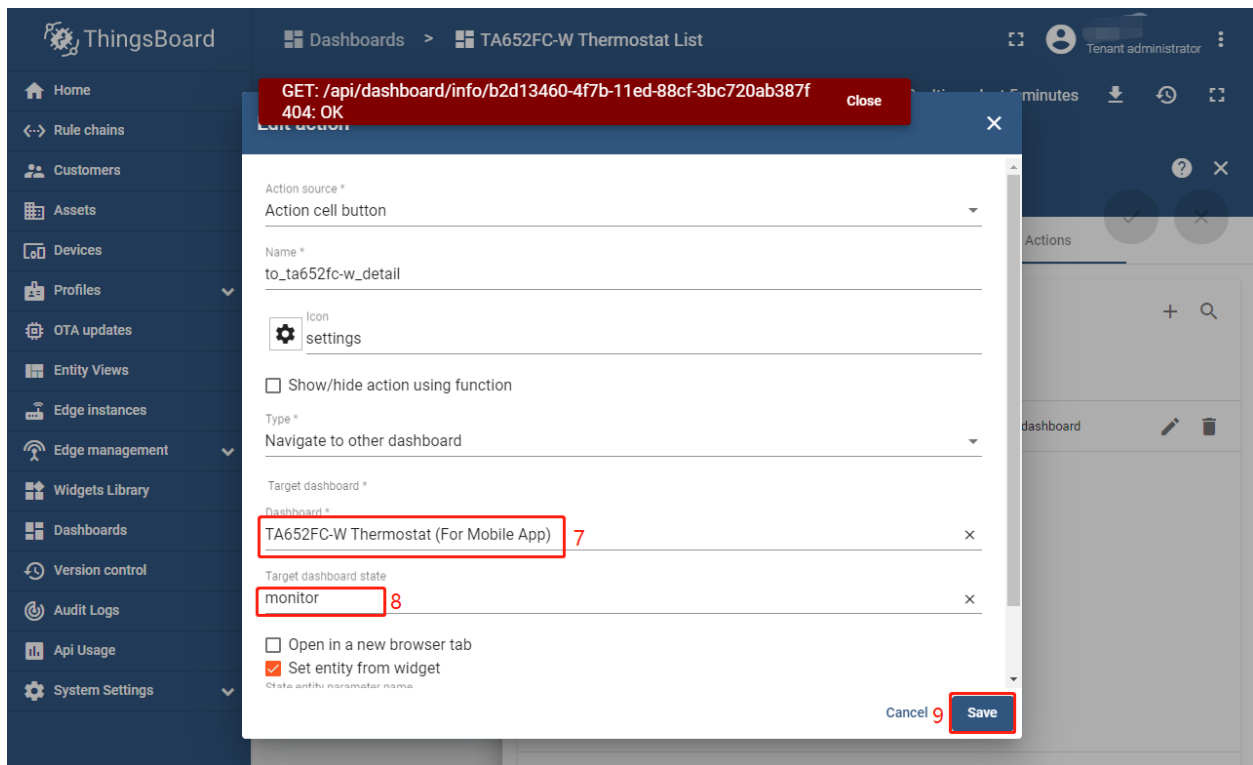
- Enter *Edit Dashboard Mode* → **Edit Widget** (icon)



- **Action** → **Edit Action** (icon)



- Modify **Target dashboard** -> modify **Target dashboard state** -> **Save**



These values are shown in the following table:

Field	Value
Target dashboard	TA652FC-W Thermostat (For Mobile App)
Target dashboard state	monitor

- **Apply changes** (red icon)

ThingsBoard

Dashboards > TA652FC-W Thermostat List

GET: /api/dashboard/info/b2d... 404: OK

TA652FC-W Thermostat List

New Entities table

Entities table

Actions

Action source	Name	Icon	Type
Action cell button	to_ta652fc-w_detail	⚙️	Navigate to other dashboard

10

- **Apply changes** (red icon on the bottom and right)

ThingsBoard

Dashboards > TA652FC-W Thermostat List


TA652FC-W Thermostat List

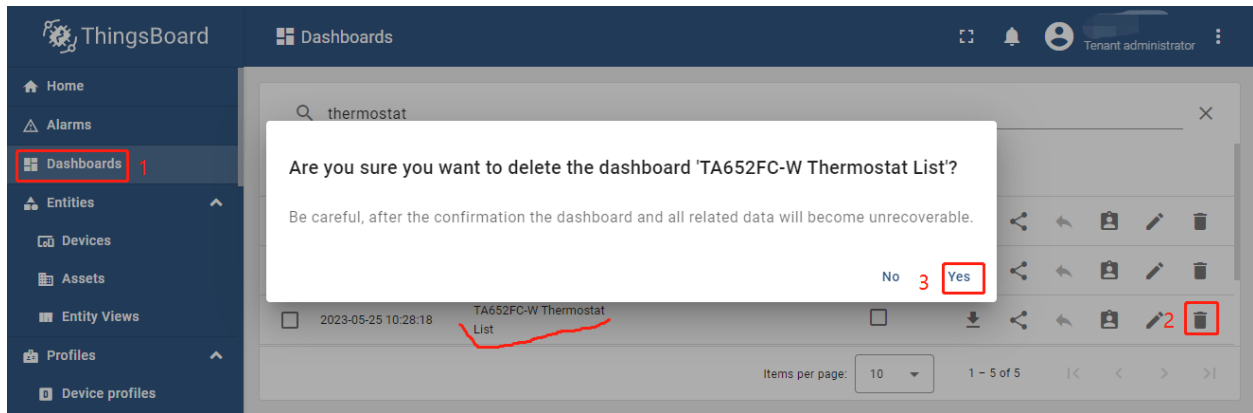
Entity name	Entity type	Type	active	Room Temp	Change Over Sensor Temp
TA652FC-W-TB	Device	TA652FC-W Thermostat	true	18.3	0

11

Powered by Thingsboard v3.4.4

### Update List Dashboard

- First, delete this dashboard: **Dashboards** → Click  in the row of TA652FC-W Thermostat List → **Popup dialog: Are you sure you want to delete ...?** → **Yes**.



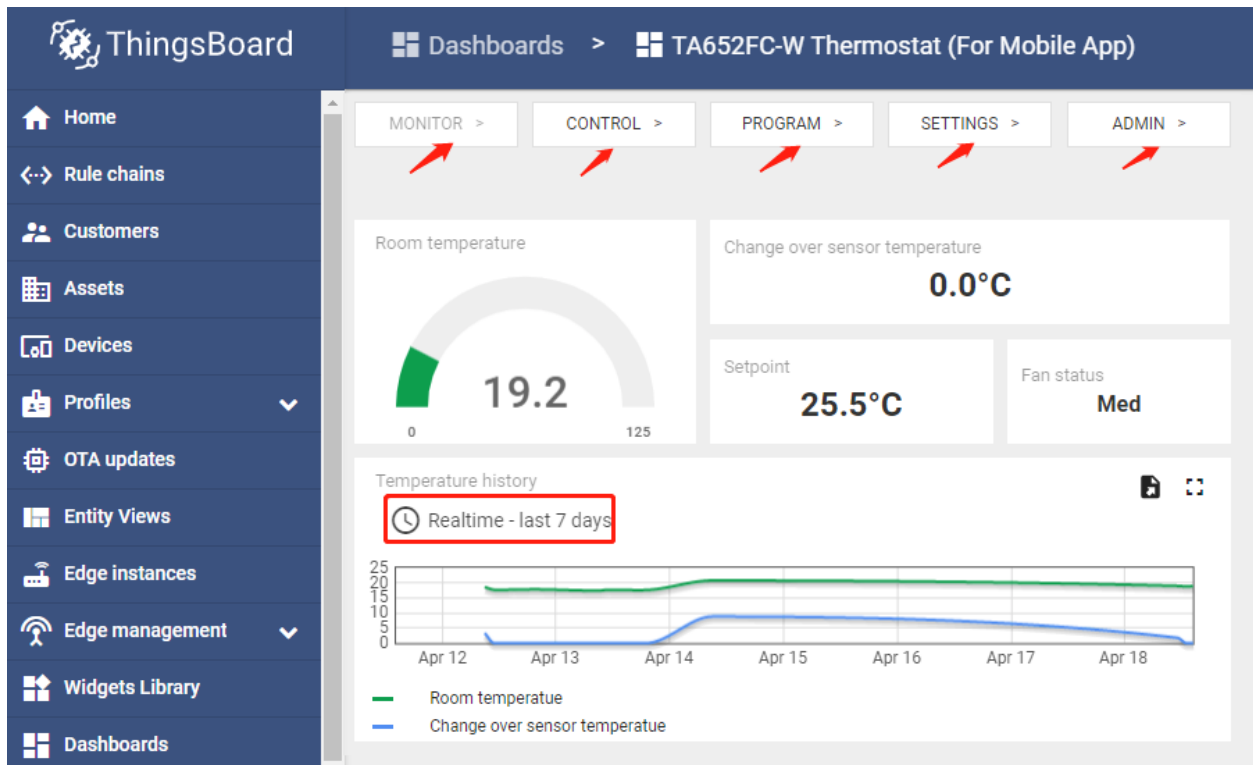
- Next, *import TA652FC-W List Dashboard*.

### 4.5.3 TA652FC-W Thermostat (For Mobile App)


#### Dashboard states

##### Monitor state

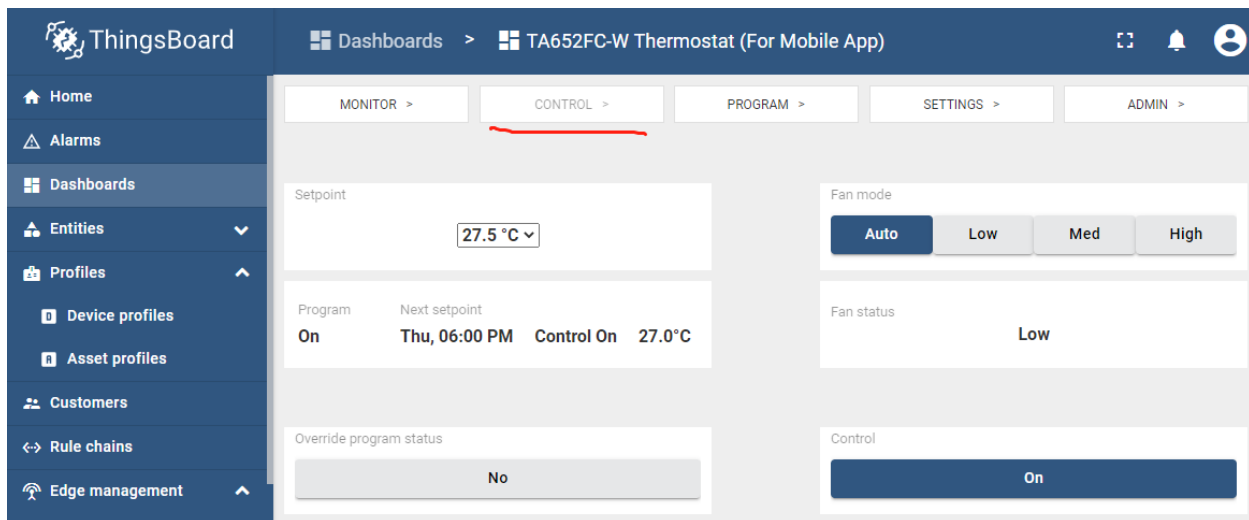
Monitor state is root state.



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Widget	Description
MONITOR	skip to <a href="#">Monitor state</a>
CONTROL	skip to <a href="#">Control state</a>
PROGRAM	skip to <a href="#">Program state</a>
SETTINGS	skip to <a href="#">Settings state</a>
ADMIN	skip to <a href="#">Admin state</a>
Room Temperature	room temperature
Change Over Sensor Temperature	change over sensor temperature
Setpoint	current setpoint value
Fan Status	“Off”, “Low”, “Med” or “High”
Temperature history	Room temperature & Change Over Sensor temperature history. Click  Realtime - last 7 days to edit this timewindow. Refer to <a href="#">Default state</a>

## Control state

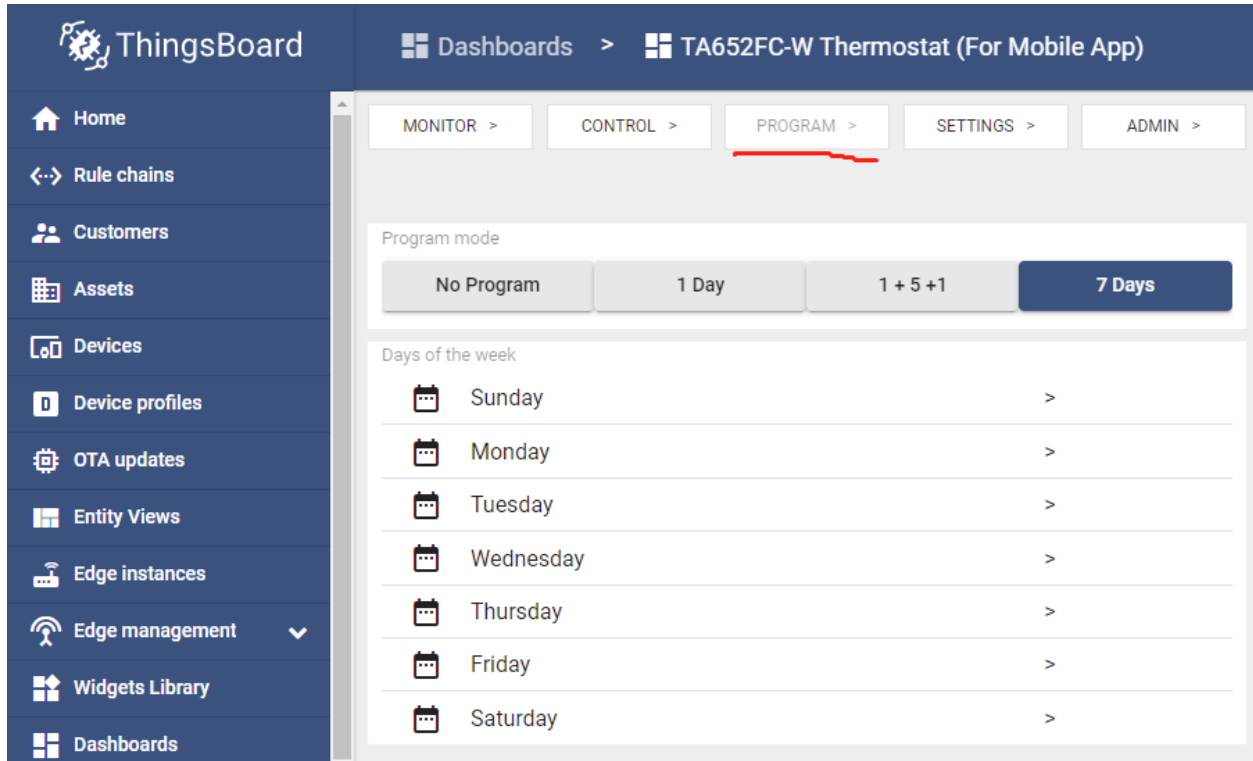


- **Dashboard bar:**  
Hidden. Refer to [Default state](#).
- **Widgets:**

Widget	Description
Setpoint	If you adjust <i>setpoint</i> , <i>override program status</i> is YES (true)
Program	program on or off
PRG next setpoint	next program time & setpoint
Override program status	“YES”(true) or “NO”(false)
Fan Mode	“Auto”, “Low”, “Med” or “High”
Fan Status	“Off”, “Low”, “Med” or “High”
Control Mode	“Off” or “On”



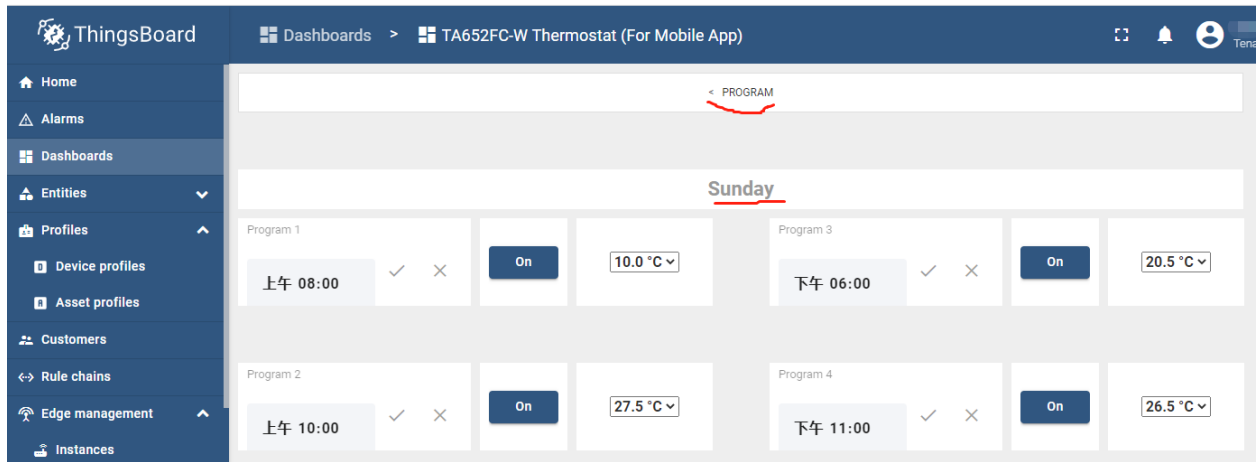
## Program state



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Program Mode	Description
NO PROGRAM	Program disabled
1 DAY (MON)	Using 4 set points of Monday every day
1+5+1 (SUN+MON+SAT)	Using 4 set points of Monday from Monday to Friday
7 DAYS (SUN~SAT)	Using 4 set points every day
Sunday, ...	Skip to <i>Program_setpoints state</i>

## Program\_setpoints state



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

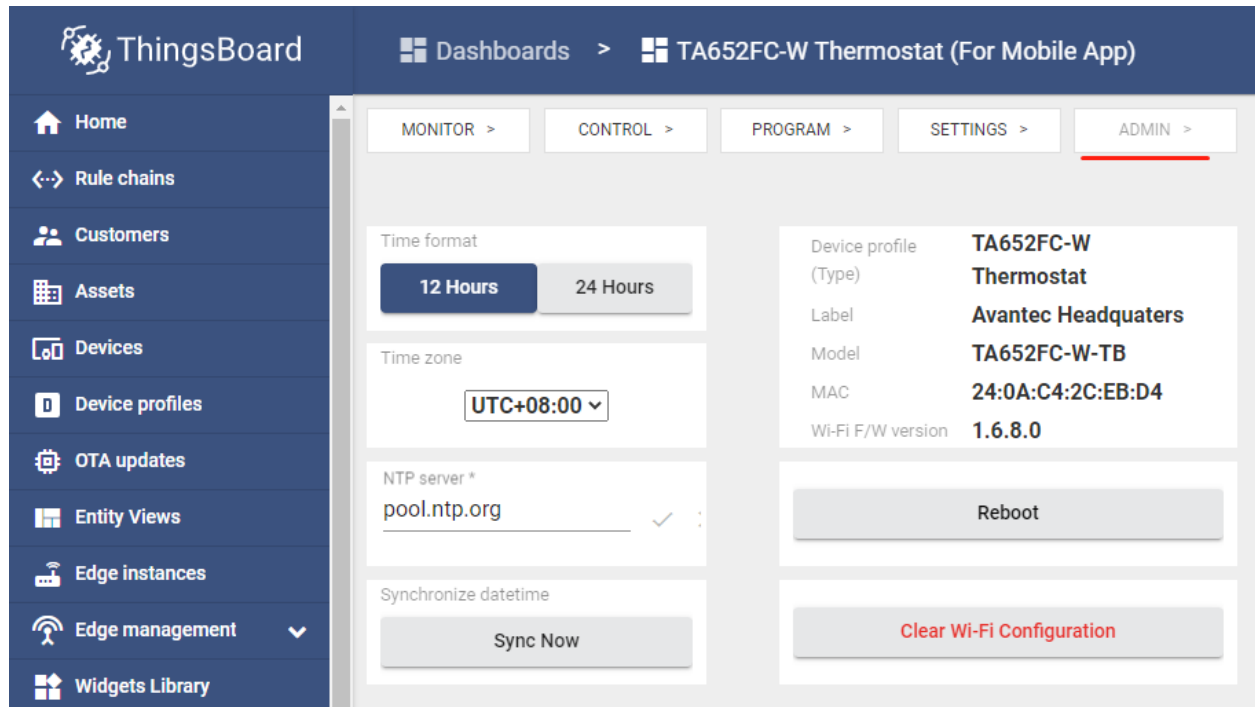
Widget	Description
Program 1 ~ Program 4	time, hour:minute
Control Mode 1 ~ Control Mode 4	On, Off
Setpoint 1 ~ Setpoint 4	setpoint value, temperature

## Settings state

- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Widget	Description
Temp Unit	“°C” or “°F”. <b>Reboot the device to take effect</b>
Change Over Mode	“Heat”, “Cool” or “Auto”
Change Over Temp Heating	Change over temperature heating
Change Over Temp Cooling	Change over temperature cooling
Force Ventilation	Used in automatic <i>Fan Mode</i>
Temp Offset(Internal Sensor)	Internal sensor temperature offset
Switching Diff Heating	Switching differential heating
Switching Diff Cooling	Switching differential cooling

## Admin state



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Widget	Description
Time Format	“12 Hours” or “24 Hours”
Timezone	See <a href="#">Step 4.2 Add shared attributes of new device</a>
NTP Server	SNTP protocol server URL, e.g. pool.ntp.org, 0.pool.ntp.org, 1.pool.ntp.org, time.nist.gov, ... see <a href="#">Step 4.2 Add shared attributes of new device</a>
Sync Time	Sync time per syncTimeFreq seconds. If you change <i>Timezone</i> or <i>NTP Server</i> , you have to do it. See <a href="#">Step 4.2 Add shared attributes of new device</a>
Device attributes	Device name, device profile (type), device label, model, MAC, device Wi-Fi Module F/W version, device Main MCU F/W version
Reboot	Reboot device
Clear Wi-Fi Config	Clear device’s Wi-Fi configuration

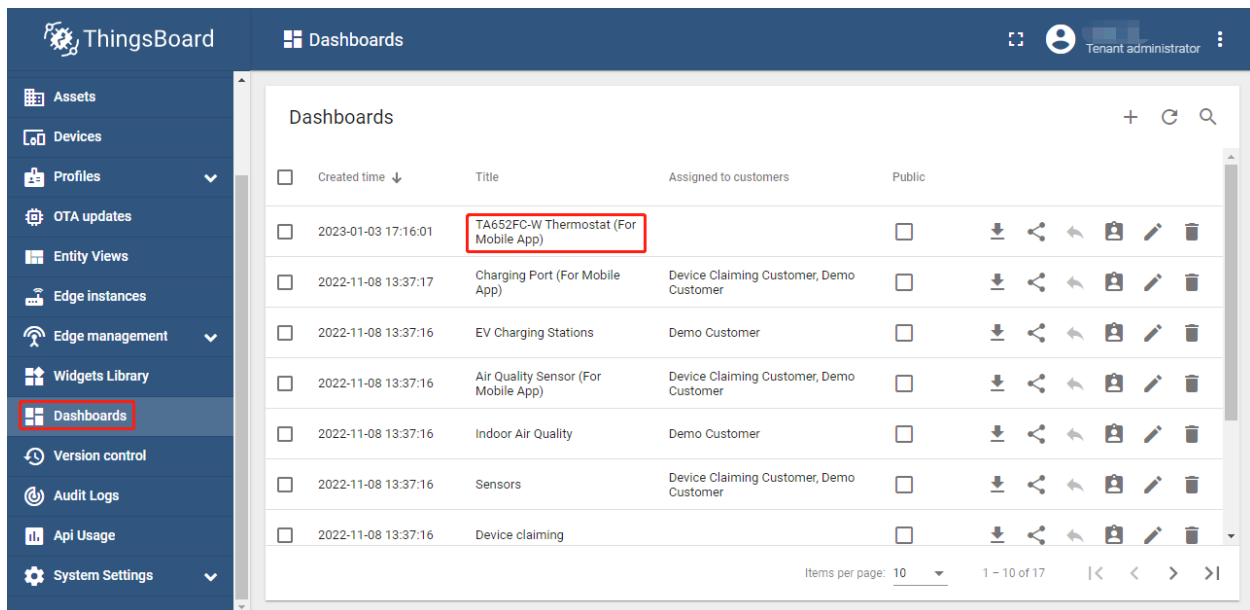
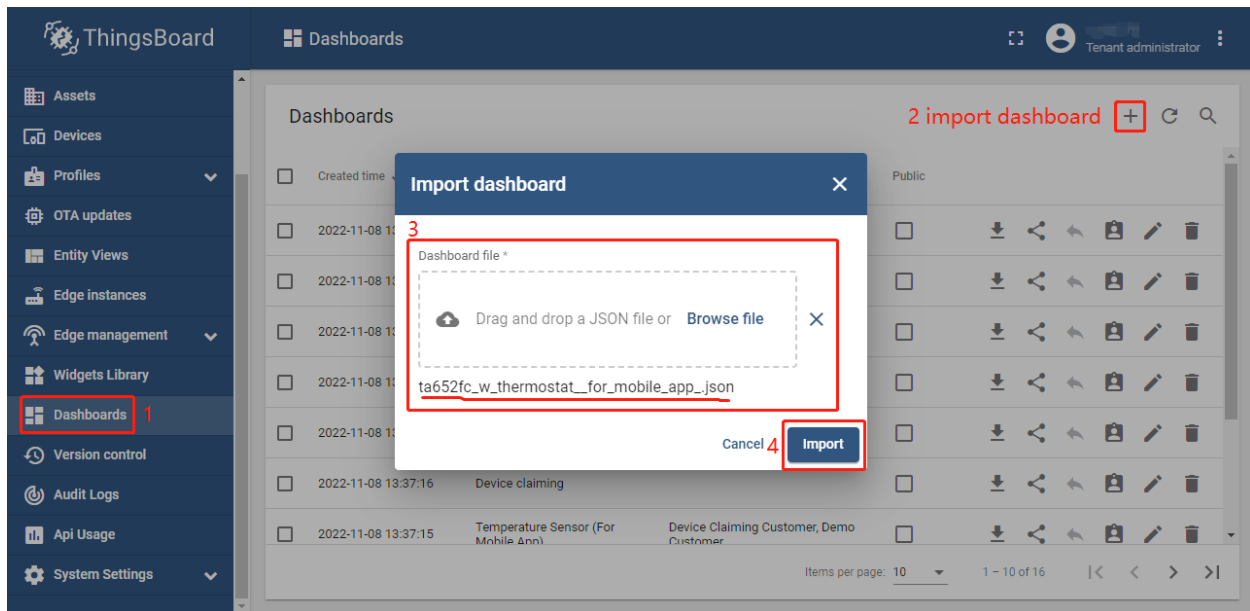
## Import Detail Dashboard

**Tip:** A *Dashboard file* can only be imported once. If you have already imported it, you don’t need and cannot repeat the import.

If you have already imported it, you can skip this step.


In order to use this dashboard, you must to create **TA652FC-W Thermostat Device Profile**. If it doesn’t exist, you can import it. See [Import Device Profile of TA652FC-W Thermostat](#).

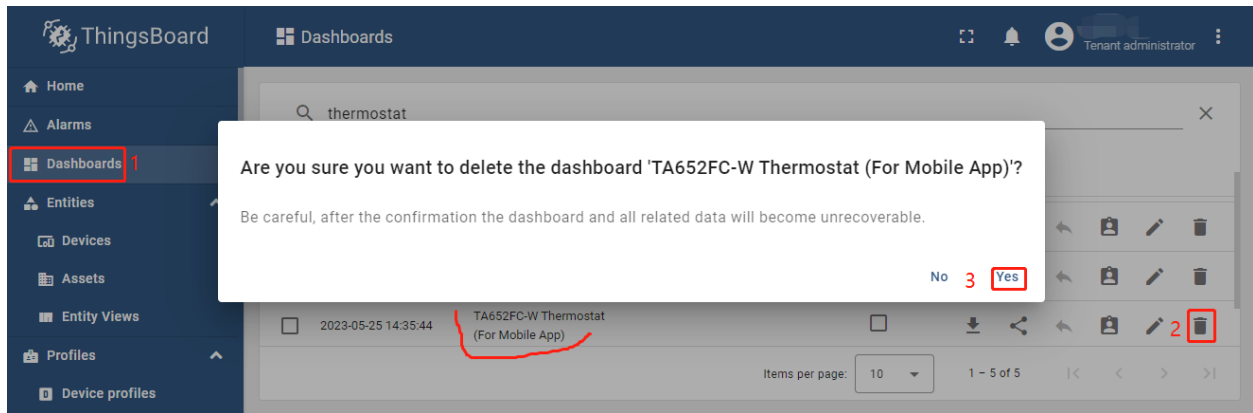
- Download `ta652fc_w_thermostat__for_mobile_app_.json`.
- **Dashboards** → **+** → **Popup dialog: Import dashboard** → Drag and drop *detail dashboard File* → **Import**.



- Optional, This dashboard can be set as TA652FC-W Thermostat Device Profile's mobile dashboard. See *Modify TA652FC-W Thermostat device profile's mobile dashboard*.

### Update Detail Dashboard

- First, *clear TA652FC-W Thermostat device profile's mobile dashboard*.
- Next, delete this dashboard: **Dashboards** → Click  in the row of TA652FC-W Thermostat (For Mobile App) → **Popup dialog: Are you sure you want to delete ...? → Yes**.



- Then *import TA652FC-W Detail Dashboard*.

## 4.6 TA652FC-W MQTT API

### Tip:

- This section applies to both TA652FC-W and TA652FH-W.
- Unless otherwise specified, all specifications applicable to TA652FC-W are also applicable to TA652FH-W.

### 4.6.1 Overview

TA652FC-W & TA652FH-W MQTT API is an implementation of *ThingsBoard MQTT Device API* (MQTT is a lightweight publish/subscribe messaging protocol).

### 4.6.2 Features

- MQTT protocol:
  - Support MQTT over TCP
  - **NOT support MQTT over SSL with mbedtls, MQTT over Websocket, MQTT over Websocket Secure**
- Base on *ThingsBoard MQTT Device API*:
  - Support telemetry upload API
  - Support client-side & shared attributes, and attributes API:
    - \* Request attribute values from the server
    - \* Publish attribute update to the server
    - \* Subscribe to attribute updates from the server
  - Support PRC API:
    - \* Server-side RPC: one-way and two-way
    - \* Client-side RPC: one-way and two-way
  - Support Claiming devices API

- Support Firmware API
- **NOT support Device provisioning API**

### 4.6.3 MQTT Special

- Currently support mqtt schemes
- Currently **NOT** support mqttts, ws, wss schemes
- MQTT over TCP samples:
  - mqtt://mqtt.eclipse.org : MQTT over TCP, default port 1883:
  - mqtt://mqtt.eclipse.org:1884 : MQTT over TCP, port 1884:

### 4.6.4 Flow Chart

---

**Tip:** The range of values for message fields is referred to in the following sections, see *Telemetry (Time-series data)*, *Shared attributes*, *Client-side attributes* and *Server-side RPC*.

---

#### TELE.01 Time-series Data Upload

**Chart:**

**Message:**

```
// Message Type: Telemetry upload (MQTT, PUBLISH)
// Topic:        v1/devices/me/telemetry
// Payload - TA652FC-W:
{"roomTemp":26.2,"changeOverTemp":26.3}
// Payload - TA652FH-W:
{"roomTemp":26.2,"floorTemp":26.3}
```

See *roomTemp*, *changeOverTemp* (only for TA652FC-W), *floorTemp* (only for TA652FH-W).

See *uploadFreq*.

See *uploadThreshold*.

#### CTRL.01 Control Mode

**Chart:**

**Message 1:**

```
// Message Type: publish client-side attributes update to the server (MQTT, ↵
↵PUBLISH)
// Topic:        v1/devices/me/attributes
// Payload:
{"controlMode":"On"}
```

**Message 2:**



```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteSetControlMode","params":"Off"}
```

See *controlMode* and *remoteSetControlMode*.

## CTRL.02 Fan Mode & Fan Status (only for TA652FC-W)

Chart:

Message 1:

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/attributes
// Payload - TA652FC-W:
{"fanMode":"Auto"}
```

Message 2:

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/attributes
// Payload - TA652FC-W:
{"fanStatus":"Low"}
```

Message 3:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload - TA652FC-W:
{"method":"remoteSetFanMode","params":"Med"}
```

See *fanMode* (only for TA652FC-W), *fanStatus* (only for TA652FC-W) and *remoteSetFanMode* (only for TA652FC-W).

## CTRL.03 Set Point & Override Status

Chart:

Message 1:

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/attributes
// Payload:
{"spValue":27.5}
```

Message 2:

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/attributes
```

(continues on next page)

(continued from previous page)

```
// Payload:  
{"overrideStatus":false}
```

### Message 3:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)  
// Topic:         v1/devices/me/rpc/request/$request_id  
// Payload:  
{"method":"remoteSetSpValue","params":34}
```

### Example 4:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)  
// Topic:         v1/devices/me/rpc/request/$request_id  
// Payload:  
{"method":"remoteSetOverrideStatus","params":{}}
```

See *spValue*, *overrideStatus*, *remoteSetSpValue* and *remoteSetOverrideStatus* .

## PRG.01 Program Mode & Program Status

### Chart:

#### Message 1:

```
// Message Type: publish client-side attributes update to the server (MQTT, ↵  
↵PUBLISH)  
// Topic:         v1/devices/me/attributes  
// Payload:  
{"prgMode":"Every-day"}
```

#### Message 2:

```
// Message Type: publish client-side attributes update to the server (MQTT, ↵  
↵PUBLISH)  
// Topic:         v1/devices/me/attributes  
// Payload:  
{"prgNextEnable":true}
```

#### Message 3:

```
// Message Type: publish client-side attributes update to the server (MQTT, ↵  
↵PUBLISH)  
// Topic:         v1/devices/me/attributes  
// Payload:  
{"prgNextCtrlMode":"On"} //{"prgNextCtrlMode":"Off"}
```

#### Message 4:

```
// Message Type: publish client-side attributes update to the server (MQTT, ↵  
↵PUBLISH)  
// Topic:         v1/devices/me/attributes  
// Payload:  
{"prgNextSetpoint":24.5}
```

**Message 5:**

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/attributes
// Payload:
{"prgNextDaysTime":"Wed, 06:00 PM"}
```

**Message 6:**

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteSetPrgMode","params":"Sun_mon-fri_sat"}
```

See *prgMode*, *prgNextEnable*, *prgNextCtrlMode*, *prgNextSetpoint*, *prgNextDaysTime* and *remoteSetPrgMode*.

**PRG.02 Program Setpoint & Time****Chart:****Message 1:**

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/attributes
// Payload:
{"prgSpTime00":"10:00"}
```

**Message 2:**

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/attributes
// Payload:
{"prgCtrlMode00":"On"} //{"prgCtrlMode27":"Off"}
```

**Message 3:**

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/attributes
// Payload:
{"prgSpValue00":27.5}
```

**Message 4:**

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteSetPrgSpTime27","params":"23:00"}
```

**Message 5:**

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteSetPrgCtrlMode27","params":"Off"} //{"method":
↪ "remoteSetPrgCtrlMode00","params":"On"}
```

**Message 6:**

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteSetPrgSpValue14","params":21.5}
```

See *prgSpTimeXX*, *prgCtrlModeXX*, *prgSpValueXX*, *remoteSetPrgSpTimeXX*, *remoteSetPrgCtrlModeXX* and *remoteSetPrgSpValueXX*.

**SET.01 Upload Device Attributes when the device is started****Chart:****Message 1:**

```
// Message Type: publish client-side attributes update to the server (MQTT, ↪
↪ PUBLISH)
// Topic:         v1/devices/me/attributes
// Payload:
{"model":"TA652FC-W-TB","mac":"24:0A:C4:2C:EB:C8",
"wifiFWVersion":"1.5.4.0","mcuFWVersion":"1.4.4.1",
"wifiRSSIMin":0,"wifiRssiMax":255,"wifiRssiStep":1,
"uploadFreqMin":2,"uploadFreqMax":2592000,"uploadFreqStep":1,
"syncTimeFreqMin":1800,"syncTimeFreqMax":2592000,"syncTimeFreqStep":1}
```

**Message 2:**

```
// Message Type: publish client-side attributes update to the server (MQTT, ↪
↪ PUBLISH)
// Topic:         v1/devices/me/attributes
// Payload:
{"supportCtrlModeInSchedule":"Yes",
"currentTempUnit":"°C",
"envirTempMin":0,"envirTempMax":50,"envirTempStep":0.1,
"spValueMin":5,"spValueMax":40,"spValueStep":0.5,
"internalOffsetMin":-5,"internalOffsetMax":5,"internalOffsetStep":0.5,
"uploadThresholdMin":0.2,"uploadThresholdMax":5,"uploadThresholdStep":0.1}
```

**Message 3 - TA652FC-W:**

```
// Message Type: publish client-side attributes update to the server (MQTT, ↪
↪ PUBLISH)
// Topic:         v1/devices/me/attributes
// Payload - TA652FC-W:
{"switchingDiffHeatingMin":1,"switchingDiffHeatingMax":4,"switchingDiffHeatingStep
↪ ":0.5,
"switchingDiffCoolingMin":1,"switchingDiffCoolingMax":4,"switchingDiffCoolingStep
```

(continues on next page)

(continued from previous page)

```

↪":0.5,
"changeOverTempHeatingMin":27,"changeOverTempHeatingMax":40,
↪"changeOverTempHeatingStep":0.5,
"changeOverTempCoolingMin":10,"changeOverTempCoolingMax":25,
↪"changeOverTempCoolingStep":0.5}

```

**Message 3 - TA652FH-W:**

```

// Message Type: publish client-side attributes update to the server (MQTT, ↪
↪PUBLISH)
// Topic:          v1/devices/me/attributes
// Payload - TA652FH-W:
{"floorTempLimitedMin":20,"floorTempLimitedMax":40,"floorTempLimitedStep":0.5,
"switchingDiffHeatingMin":1,"switchingDiffHeatingMax":4,"switchingDiffHeatingStep
↪":0.5,
"switchingDiffCoolingMin":1,"switchingDiffCoolingMax":4,"switchingDiffCoolingStep
↪":0.5}

```

See *model*, *mac*, *wifiFWVersion*, *mcuFWVersion*, *wifiRSSIMin*, *wifiRssiMax*, *wifiRssiStep*, *uploadFreqMin*, *uploadFreqMax*, *uploadFreqStep*, *syncTimeFreqMin*, *syncTimeFreqMax* and *syncTimeFreqStep*.

See *supportCtrlModeInSchedule*, *currentTempUnit*, *envirTempMin*, *envirTempMax*, *envirTempStep*, *spValueMin*, *spValueMax*, *spValueStep*, *internalOffsetMin*, *internalOffsetMax* and *internalOffsetStep*, *uploadThresholdMin*, *uploadThresholdMax* and *uploadThresholdStep*.

See *floorTempLimitedMin* (only for TA652FH-W), *floorTempLimitedMax* (only for TA652FH-W), *floorTempLimitedStep* (only for TA652FH-W), *switchingDiffHeatingMin*, *switchingDiffHeatingMax*, *switchingDiffHeatingStep*, *switchingDiffCoolingMin*, *switchingDiffCoolingMax*, *switchingDiffCoolingStep*, *changeOverTempHeatingMin* (only for TA652FC-W), *changeOverTempHeatingMax* (only for TA652FC-W), *changeOverTempHeatingStep* (only for TA652FC-W), *changeOverTempCoolingMin* (only for TA652FC-W), *changeOverTempCoolingMax* (only for TA652FC-W) and *changeOverTempCoolingStep* (only for TA652FC-W).

**SET.02 Settings****Chart:****Message 1a:**

```

// Message Type: publish client-side attributes update to the server (MQTT, ↪
↪PUBLISH)
// Topic:          v1/devices/me/attributes
// Payload:
{"tempUnit":"°C"}

```

**Message 1b:**

```

// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteSetTempUnit","params":"°F"}

```

**Message 2a:**

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)  
// Topic: v1/devices/me/attributes  
// Payload:  
{"timeFormat": "12hours"}
```

### Message 2b:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)  
// Topic: v1/devices/me/rpc/request/$request_id  
// Payload:  
{"method": "remoteSetTimeFormat", "params": "24hours"}
```

### Message 3a:

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)  
// Topic: v1/devices/me/attributes  
// Payload:  
{"method": "remoteSetInternalOffset", "params": -3.5}
```

### Message 3b:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)  
// Topic: v1/devices/me/rpc/request/$request_id  
// Payload:  
{"internalOffset": -3.5}
```

### Message 4a:

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)  
// Topic: v1/devices/me/attributes  
// Payload:  
{"switchingDiffHeating": 3.5}
```

### Message 4b:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)  
// Topic: v1/devices/me/rpc/request/$request_id  
// Payload:  
{"method": "remoteSetSwitchingDiffHeating", "params": 3.5}
```

### Message 5a:

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)  
// Topic: v1/devices/me/attributes  
// Payload:  
{"switchingDiffCooling": 2.5}
```

### Message 5b:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteSetSwitchingDiffCooling","params":2.5}
```

**Message 6a - TA652FH-W:**

```
// Message Type: publish client-side attributes update to the server (MQTT, ↪PUBLISH)
// Topic:         v1/devices/me/attributes
// Payload - TA652FH-W:
{"systemMode":"Cool"}
```

**Message 6b - TA652FH-W:**

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload - TA652FH-W:
{"method":"remoteSetSystemMode","params":"Heat"}
```

**Message 7a - TA652FH-W:**

```
// Message Type: publish client-side attributes update to the server (MQTT, ↪PUBLISH)
// Topic:         v1/devices/me/attributes
// Payload - TA652FH-W:
{"sensorMode":"Internal"}
```

**Message 7b - TA652FH-W:**

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload - TA652FH-W:
{"method":"remoteSetSensorMode","params":"External"}
```

**Message 8a - TA652FH-W:**

```
// Message Type: publish client-side attributes update to the server (MQTT, ↪PUBLISH)
// Topic:         v1/devices/me/attributes
// Payload - TA652FH-W:
{"floorTempLimited":29.5}
```

**Message 8b - TA652FH-W:**

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload - TA652FH-W:
{"method":"remoteSetFloorTempLimited","params":29.5}
```

**Message 9a - TA652FH-W:**

```
// Message Type: publish client-side attributes update to the server (MQTT, ↪PUBLISH)
```

(continues on next page)

(continued from previous page)

```
// Topic:          v1/devices/me/attributes
// Payload - TA652FH-W:
{"adaptiveControl":false}
```

### Message 9b - TA652FH-W:

```
// Message Type:  receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/rpc/request/$request_id
// Payload - TA652FH-W:
{"method":"remoteSetAdaptiveControl","params":true}
```

### Message 10a - TA652FC-W:

```
// Message Type:  publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/attributes
// Payload - TA652FC-W:
{"forceVent":true}
```

### Message 10b - TA652FC-W:

```
// Message Type:  receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/rpc/request/$request_id
// Payload - TA652FC-W:
{"method":"remoteSetForceVent","params":false}
```

### Message 11a - TA652FC-W:

```
// Message Type:  publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/attributes
// Payload - TA652FC-W:
{"changeOverMode":"Heat"}
```

### Message 11b - TA652FC-W:

```
// Message Type:  receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/rpc/request/$request_id
// Payload - TA652FC-W:
{"method":"remoteSetChangeOverMode","params":"Auto"}
```

### Message 12a - TA652FC-W:

```
// Message Type:  publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/attributes
// Payload - TA652FC-W:
{"changeOverTempHeating":27}
```

### Message 12b - TA652FC-W:

```
// Message Type:  receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:          v1/devices/me/rpc/request/$request_id
```

(continues on next page)



(continued from previous page)

```
// Payload - TA652FC-W:
{"method":"remoteSetChangeOverTempHeating","params":27}
```

**Message 13a - TA652FC-W:**

```
// Message Type: publish client-side attributes update to the server (MQTT, PUBLISH)
// Topic: v1/devices/me/attributes
// Payload - TA652FC-W:
{"changeOverTempCooling":11.5}
```

**Message 13b - TA652FC-W:**

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic: v1/devices/me/rpc/request/$request_id
// Payload - TA652FC-W:
{"method":"remoteSetChangeOverTempCooling","params":10}
```

See *tempUnit* and *remoteSetTempUnit*, *timeFormat* and *remoteSetTimeFormat*, *internalOffset* and *remoteSetInternalOffset*, *switchingDiffHeating* and *remoteSetSwitchingDiffHeating*, *switchingDiffCooling* and *remoteSetSwitchingDiffCooling*.

See *systemMode* and *remoteSetSystemMode*, *sensorMode* and *remoteSetSensorMode*, *floorTempLimited* and *remoteSetFloorTempLimited*, *adaptiveControl* and *remoteSetAdaptiveControl*.(only for TA652FH-W)

See *forceVent* and *remoteSetForceVent*, *changeOverMode* and *remoteSetChangeOverMode*, *changeOverTempHeating* and *remoteSetChangeOverTempHeating*, *changeOverTempCooling* and *remoteSetChangeOverTempCooling*.(only for TA652FC-W)

**ADM.01 Request all remote parameters when the device is started****Chart:****Message 1:**

```
// Message Type: request attribute values from the server (MQTT, PUBLISH)
// Topic: v1/devices/me/attributes/request/$request_id
// Payload:
{"sharedKeys":"uploadFreq,uploadThreshold,syncTimeFreq,timezone,timeNTPServer"}
```

**Message 2:**

```
// Message Type: receive response (MQTT, PUBLISH)
// Topic: v1/devices/me/attributes/response/$request_id
// Payload:
{"shared":{"uploadFreq":300,"uploadThreshold":1.5,"syncTimeFreq":86400,
"timezone":480,"timeNTPServer":"pool.ntp.org"}}
```

See *uploadFreq*, *uploadThreshold*, *syncTimeFreq*, *timezone* and *timeNTPServer*.

## ADM.02 Timer Parameters & upload threshold

Chart:

Message 1:

```
// Message Type: receive attribute update from the server (MQTT, PUBLISH)
// Topic:        v1/devices/me/attributes
// Payload:
{"uploadFreq":300}
```

Message 2:

```
// Message Type: receive attribute update from the server (MQTT, PUBLISH)
// Topic:        v1/devices/me/attributes
// Payload:
{"uploadThreshold":1.5}
```

Message 3:

```
// Message Type: receive attribute update from the server (MQTT, PUBLISH)
// Topic:        v1/devices/me/attributes
// Payload:
{"syncTimeFreq":86400}
```

See *uploadFreq*, *uploadThreshold* and *syncTimeFreq*.

## ADM.03 Remote Sync Time

Chart:

Message 1:

```
// Message Type: receive attribute update from the server (MQTT, PUBLISH)
// Topic:        v1/devices/me/attributes
// Payload:
{"timezone":480}
```

Message 2:

```
// Message Type: receive attribute update from the server (MQTT, PUBLISH)
// Topic:        v1/devices/me/attributes
// Payload:
{"timeNTPServer":"pool.ntp.org"}
```

Message 3:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:        v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteSyncTimeRequest","params":{}}
```

See *timezone*, *timeNTPServer* and *remoteSyncTimeRequest*.

## ADM.04 FUOTA (firmware update over the air)

The flow is to download the firmware from your HTTP server. For the flow of downloading firmware from ThingsBoard server, please refer to *Firmware API*.

### Chart:

#### Message 1a:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteWiFiFUOTA",
"params":"http://192.168.1.106/TA652FC-W_WiFi.ino.bin"}
```

#### Message 1b:

```
// Message Type: send response (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/response/$request_id
// Payload:
{"method":"remoteWiFiFUOTA","results":{"result":"success"}}
```

#### Message 2a (NOT implemented):

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteMcuFUOTA",
"params":"http://192.168.1.106/TA652FC-W_MCU.bin"}
```

#### Message 2b (NOT implemented):

```
// Message Type: send response (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/response/$request_id
// Payload:
{"method":"remoteMcuFUOTA","results":{"result":"success"}}
```

See *remoteWiFiFUOTA* and *remoteMcuFUOTA*.

## ADM.05 Remote Get Memory Usage

### Chart:

#### Message 1a:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteGetMemoryUsage"}
```

#### Message 1b:

```
// Message Type: send response (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/response/$request_id
// Payload:
{"iram":162592,"spiram":4194252}
```

See *remoteGetMemoryUsage*.

## ADM.06 Remote Reboot Device

Chart:

Message 1:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteRebootDevice","params":{}}
```

See *remoteRebootDevice*.

## ADM.07 Remote Clear Wi-Fi Config

Chart:

Message 1:

```
// Message Type: receive server-side RPC request from the server (MQTT, PUBLISH)
// Topic:         v1/devices/me/rpc/request/$request_id
// Payload:
{"method":"remoteClearWiFiConfig","params":{}}
```

See *remoteClearWiFiConfig*.

## Claiming

Refer to *Claiming API*.

## Firmware update with ThingsBoard Server

Refer to *Firmware API*.

## 4.6.5 Telemetry (Time-series data)

---

**Tip:**

- All of these telemetry (timeseries data) is uploaded every *uploadFreq* seconds.
  - If these telemetry (timeseries data) change exceeds *uploadThreshold*, upload immediately.
- 

**roomTemp**

**changeOverTemp**

**floorTemp**

**wifiRssi**

Table 3: Telemetry (Time-series data)

Time-series	Type	Unit	Min	Max	Step/Pre-Value	TA652 FC-W	TA652 FH-W	Memo
roomTemp	float	<i>current-TempUnit</i>	<i>en-virTemp-Min</i>	<i>en-virTemp-Max</i>	<i>en-virTemp-Step</i>			Room temperature
changeOverTemp	float	<i>current-TempUnit</i>	<i>en-virTemp-Min</i>	<i>en-virTemp-Max</i>	<i>en-virTemp-Step</i>			Change Over Temperature
floorTemp	float	<i>current-TempUnit</i>	<i>en-virTemp-Min</i>	<i>en-virTemp-Max</i>	<i>en-virTemp-Step</i>			Floor Temperature
wifiRssi (deprecated)*	int		<i>wifiRssiMin</i>	<i>wifiRssiMax</i>	<i>wifiRssiStep</i>			Received Signal Strength Indicator

**Tip:** In order to reduce the load on ThingsBoard server, *wifiRssi* is no longer sent.

**4.6.6 Shared attributes**

**Tip:** All of these shared attributes may be obtained from your ThingsBoard server.

**uploadFreq****uploadThreshold****syncTimeFreq****timezone**

## timeNTPServer

Table 4: Shared attributes

Sharec at- tribute	Type	Unit	Min	Max	Step/Pr	Value	TA652 FC- W	TA652 FH- W	Memo
up- load- Freq	int	sec- ond	<i>up- load- Fre- qMin</i>	<i>up- load- Fre- qMax</i>	<i>up- load- Fre- qStep</i>	Default: 300			Time-series (Telemetry) up- load Frequency. see <i>Step 4.2 Add shared attributes of new device.</i>
up- load- Thresh- old	dou- ble	tem- per- a- ture	<i>up- load- Thresh- old- Min</i>	<i>up- load- Thresh- old- Max</i>	<i>up- load- Thresh- old- Step</i>	Default: 1.5			Time-series (Telemetry) up- load Threshold. see <i>Step 4.2 Add shared attributes of new device.</i>
sync- Time- Freq	int	sec- ond	<i>sync- Time- Fre- qMin</i>	<i>sync- Time- Fre- qMax</i>	<i>sync- Time- Fre- qStep</i>	Default: 86400 (24*3600)			timer period of sync datetime. see <i>Step 4.2 Add shared at- tributes of new device.</i>
time- zone	int	minut				Default: 480 (8*60)			offset UTC. see <i>Step 4.2 Add shared attributes of new de- vice.</i>
ti- meNTF	string					Default: pool.ntp.org (127 char+'0')			SNTP server, eg: pool.ntp.org . see <i>Step 4.2 Add shared at- tributes of new device.</i>

## 4.6.7 Client-side attributes

## Client-side attribute (static/fixed)

model

mac

wifiFWVersion

**mcuFWVersion**

Table 5: Client-side attribute (static/fixed)

Client-side attribute (static/fixed)	Type	Unit	Value	TA652 FC-W	TA652 FH-W	Memo
model	string		“TA652FC-W-TB”, “TA652FH-W-TB”			Product Model
mac	string		eg: “34:02:86:5F:23:A9”			Mac Address
wifiFWVersion	string		eg: “1.5.5”			WiFi Module F/W version
mcuFWVersion	string		eg: “1.5.4”			Main MCU F/W version

**Client-side attribute (static/fixed, metadata)****wifiRssiMin****wifiRssiMax****wifiRssiStep****uploadFreqMin****uploadFreqMax****uploadFreqStep****syncTimeFreqMin****syncTimeFreqMax**

**syncTimeFreqStep**

Table 6: Client-side attribute (static/fixed, metadata)

Client-side attribute (static/fixed, metadata)	Type	Unit	Value	TA652 FC-W	TA652 FH-W	Memo
wifiRssiMin	int		0			the minimum value of <i>wifiRssi</i>
wifiRssiMax	int		255			the maximum value of <i>wifiRssi</i>
wifiRssiStep	int		1			the step value of <i>wifiRssi</i>
uploadFreqMin	int	sec- ond	2			the minimum value of <i>uploadFreq</i>
uploadFreqMax	int	sec- ond	30*24*3600 (2592000)			the maximum value of <i>uploadFreq</i>
uploadFreqStep	int	sec- ond	1			the step value of <i>upload-Freq</i>
syncTimeFreqMin	int	sec- ond	30*60			the minimum value of <i>syncTimeFreq</i>
syncTimeFreqMax	int	sec- ond	30*24*3600			the maximum value of <i>syncTimeFreq</i>
syncTimeFreqStep	int	sec- ond	1			the step value of <i>sync-TimeFreq</i>

**Client-side attribute (semi-static)****supportCtrlModelInSchedule****currentTempUnit****tempResolution****envirTempMin****envirTempMax****envirTempStep****spValueMin****spValueMax****spValueStep****internalOffsetMin****internalOffsetMax**



internalOffsetStep

uploadThresholdMin

uploadThresholdMax

uploadThresholdStep

floorTempLimitedMin

floorTempLimitedMax

floorTempLimitedStep

switchingDiffHeatingMin

switchingDiffHeatingMax

switchingDiffHeatingStep

switchingDiffCoolingMin

switchingDiffCoolingMax

switchingDiffCoolingStep

changeOverTempHeatingMin

changeOverTempHeatingMax

changeOverTempHeatingStep

changeOverTempCoolingMin

changeOverTempCoolingMax

## changeOverTempCoolingStep

Table 7: Client-side attribute (semi-static)

Client-side attribute (semi-static)	at (semi-static)	Type	Unit	Value	TA652 FC-W	TA652 FH-W	Memo
supportCtrlModeIn-Schedule		string		“Yes” / “No”			Control On/Off in schedule
currentTempUnit		string		“°C” / “°F”			Centigrade, Fahrenheit
envirTempMin		float	<i>current-TempUnit</i>	0.0 (°C) / 32 (°F)			the minimum value of <i>roomTemp changeOverTemp, floorTemp</i>
envirTempMax		float	<i>current-TempUnit</i>	50.0 (°C) / 120 (°F)			the maximum value of <i>roomTemp changeOverTemp, floorTemp</i>
envirTempStep		float	<i>current-TempUnit</i>	0.1 (°C) / 0.5 (°F)			the step value of <i>roomTemp changeOverTemp, floorTemp</i>
spValueMin		float	<i>current-TempUnit</i>	5.0 (°C) / 40 (°F)			the minimum value of <i>spValue prgSpValueXX</i>
spValueMax		float	<i>current-TempUnit</i>	40.0 (°C) / 104 (°F)			the maximum value of <i>spValue prgSpValueXX</i>
spValueStep		float	<i>current-TempUnit</i>	0.5 (°C) / 1.0 (°F)			the step value of <i>spValue prgSpValueXX</i>
internalOffsetMin		float	<i>current-TempUnit</i>	-5.0 (°C) / -10 (°F)			the minimum value of <i>internalOffset</i>
internalOffsetMax		float	<i>current-TempUnit</i>	5.0 (°C) / 10 (°F)			the maximum value of <i>internalOffset</i>
internalOffsetStep		float	<i>current-TempUnit</i>	0.1 (°C) / 0.5 (°F)			the step value of <i>internalOffset</i>
uploadThreshold-Min		float	<i>current-TempUnit</i>	0.2 (°C) / 32 (°F)			the minimum value of <i>upload-Threshold</i>
uploadThreshold-Max		float	<i>current-TempUnit</i>	5.0 (°C) / 41 (°F)			the maximum value of <i>upload-Threshold</i>
uploadThresholdtep		float	<i>current-TempUnit</i>	0.1 (°C) / 0.5 (°F)			the step value of <i>uploadThreshold</i>
floorTempLimited-Min		float	<i>current-TempUnit</i>	20.0 (°C) / 68 (°F)			the minimum value of <i>floorTempLimited</i>
floorTempLimited-Max		float	<i>current-TempUnit</i>	5.0 (°C) / 10 (°F)			the maximum value of <i>floorTempLimited</i>
floorTempLimited-Step		float	<i>current-TempUnit</i>	40.0 (°C) / 104 (°F)			the step value of <i>floorTempLimited</i>
switchingDiffHeatingMin		float	<i>current-TempUnit</i>	0.5 (°C) / 1 (°F)			the minimum value of <i>switchingDiffHeating</i>
switchingDiffHeat-		float	<i>current-</i>	4.0 (°C) / 8			the maximum value of <i>switch-</i>

**Client-side attribute (application state)****fanStatus****overrideStatus****prgNextEnable****prgNextCtrlMode****prgNextDaysTime****prgNextSetpoint**

Table 8: Client-side attribute (application state)

Client-side attribute (application state)	Type	Unit	Value	TA652 FC-W	TA652 FH-W	Memo
fanStatus	string		“Off”, “Low”, “Med”, “High”			
overrideStatus	bool		true, false			see <i>spValue</i>
prgNextEnable	bool		true, false			Next program enabled
prgNextCtrlMode	string		“On”, “Off”			Next program control On/Off
prgNextDaysTime	float					Next program weekday & time
prgNextSetpoint	float	<i>current-TempUnit</i>				Next program set point

**Client-side attribute (change by server-side RPC, settings)****tempUnit****timeFormat****systemMode****sensorMode****internalOffset****floorTempLimited****switchingDiffHeating****switchingDiffCooling**

**adaptiveControl**

**forceVent**

**changeOverMode**

**changeOverTempHeating**

## changeOverTempCooling

Table 9: Client-side attribute (change by server-side RPC, settings)

Client-side attribute	Type	Unit	Min	Max	Step/ Precision	Value	TA65 FC- W	TA65 FH- W	Memo
tempUnit	string					“°C” / “°F”			Centigrade, Fahrenheit, see <i>remoteSetTempUnit</i>
timeFormat	string					“12hours”, “24hours”			see <i>remoteSetTimeFormat</i>
systemMode	string					“Heat”, “Cool”			see <i>remoteSetSystemMode</i>
sensorMode	string					“Internal”, “External”, “Combined”			see <i>remoteSetSensorMode</i>
internalOffset	float	<i>currentTempUnit</i>	<i>internalOffsetMin</i>	<i>internalOffsetMax</i>	<i>internalOffsetStep</i>	“Internal”, “External”, “Combined”			Internal Sensor Temperature Offset, see <i>remoteSetInternalOffset</i>
floorTempLimited	float	<i>currentTempUnit</i>	<i>floorTempLimitedMin</i>	<i>floorTempLimitedMax</i>	<i>floorTempLimitedStep</i>				floor temperature limited (combined mode), see <i>remoteSetFloorTempLimited</i>
switchingDifferentialHeating	float	<i>currentTempUnit</i>	<i>switchingDifferentialHeatingMin</i>	<i>switchingDifferentialHeatingMax</i>	<i>switchingDifferentialHeatingStep</i>				Switching Differential Heating, see <i>remoteSetSwitchingDifferentialHeating</i>
switchingDifferentialCooling	float	<i>currentTempUnit</i>	<i>switchingDifferentialCoolingMin</i>	<i>switchingDifferentialCoolingMax</i>	<i>switchingDifferentialCoolingStep</i>				Switching Differential Cooling, see <i>remoteSetSwitchingDifferentialCooling</i>
adaptiveControl	bool					true, false			see <i>remoteSetAdaptiveControl</i>
forceVent	bool					true, false			Force Ventilation, see <i>remoteSetForceVent</i>
changeOverMode	string					“Heat”, “Cool”, “Auto”			see <i>remoteSetChangeOverMode</i>
changeOverTempHeating	float	<i>currentTempUnit</i>	<i>changeOverTempHeatingMin</i>	<i>changeOverTempHeatingMax</i>	<i>changeOverTempHeatingStep</i>				Change Over Temp Heating, see <i>remoteSetChangeOverTempHeating</i>
changeOverTempCooling	float	<i>currentTempUnit</i>	<i>changeOverTempCoolingMin</i>	<i>changeOverTempCoolingMax</i>	<i>changeOverTempCoolingStep</i>				Change Over Temp Cooling, see <i>remoteSetChangeOverTempCooling</i>

## Client-side attribute (change by server-side RPC, control &amp; program)

**controlMode****fanMode****spValue****prgMode****prgSpTimeXX**

0 &lt;= XX &lt;= 27, prgSpTime00 ~ prgSpTime27

**prgCtrlModeXX**

0 &lt;= XX &lt;= 27, prgCtrlMode00 ~ prgCtrlMode27

**prgSpValueXX**

0 &lt;= XX &lt;= 27, prgSpValue00 ~ prgSpValue27

Table 10: Client-side attribute (change by server-side RPC, control &amp; program)

Client-side attribute	Type	Unit	Min	Max	Step/Precision	Value	TA652 FC-W	TA652 FH-W	Memo
controlMode	string					“Off”, “On”			see <i>remoteSet-ControlMode</i>
fanMode	string					“Auto”, “Low”, “Med”, “High”			see <i>remoteSet-FanMode</i>
spValue	float	current-TempUnit	sp-Val-ueMin	sp-Val-ueMax	spVal-ueStep				see <i>remoteSet-SpValue</i> , see <i>overrideStatus</i>
prgMode	string					“No-program”, “One-day”, “Sun_mon-fri_sat”, “Every-day”			see <i>remoteSet-PrgMode</i>
prgSpTimeXX	string					“hh:mm”, eg: “23:50”			see <i>remoteSet-PrgSpTimeXX</i>
prgCtrlModeXX	string					“On”, “Off”			see <i>remoteSet-PrgCtrlModeXX</i>
prgSpValueXX	float	current-TempUnit	sp-Val-ueMin	sp-Val-ueMax	spVal-ueStep				see <i>remoteSet-PrgSpValueXX</i>

## 4.6.8 Server-side RPC

### Server-side RPC (remote change client-side attribute)

---

**Tip:**

- All of these server-side RPC are **one-way**, no response
  - Request format of these server-side RPC: {"method": "remoteSetTempUnit", "params": "°F"}
  - **params** value see *Client-side attribute (change by server-side RPC, settings)* & *Client-side attribute (change by server-side RPC, control & program)*
- 

remoteSetTempUnit

remoteSetTimeFormat

remoteSetSystemMode

remoteSetSensorMode

remoteSetInternalOffset

remoteSetFloorTempLimited

remoteSetSwitchingDiffHeating

remoteSetSwitchingDiffCooling

remoteSetAdaptiveControl

remoteSetForceVent

remoteSetChangeOverMode

remoteSetChangeOverTempHeating

remoteSetChangeOverTempCooling

remoteSetControlMode

remoteSetFanMode

remoteSetSpValue

remoteSetPrgMode

### **remoteSetPrgSpTimeXX**

0 <= XX <= 27, remoteSetPrgSpTime00 ~ remoteSetPrgSpTime27

### **remoteSetPrgCtrlModeXX**

0 <= XX <= 27, remoteSetPrgCtrlMode00 ~ remoteSetPrgCtrlMode27

### **remoteSetPrgSpValueXX**

0 <= XX <= 27, remoteSetPrgSpValue00 ~ remoteSetPrgSpValue27



Table 11: Server-side RPC (remote change client-side attribute)

Server-side RPC (remote change client-side attribute)	params value type	params value	TA652 FC-W	TA652 FH-W	Memo
remoteSetTempUnit	string	“°C” / “°F”			<i>tempUnit</i>
remoteSetTimeFormat	string	“12hours” “24hours”			<i>timeFormat</i>
remoteSetSystemMode	string	“Heat” “Cool”			<i>systemMode</i>
remoteSetSensorMode	string	“Internal” “External” “Combined”			<i>sensorMode</i>
remoteSetInternalOffset	float				<i>internalOffset</i>
remoteSetFloorTempLimited	float				<i>floorTempLimited</i>
remoteSetSwitchingDiffHeating	float				<i>switchingDiffHeating</i>
remoteSetSwitchingDiffCooling	float				<i>switchingDiffCooling</i>
remoteSetAdaptiveControl	bool	true false			<i>adaptiveControl</i>
remoteSetForceVent	bool	true false			<i>forceVent</i>
remoteSetChangeOverMode	string	“Heat” “Cool” “Auto”			<i>changeOverMode</i>
remote-SetChangeOverTempHeating	float				<i>changeOverTempHeating</i>
remote-SetChangeOverTempCooling	float				<i>changeOverTempCooling</i>
remoteSetControlMode	string	“Off” “On”			<i>controlMode</i>
remoteSetFanMode	string	“Auto” “Low” “Med” “High”			<i>fanMode</i>
remoteSetSpValue	float				<i>spValue</i>
remoteSetPrgMode	string	“No-program” “One-day” “Sun_mon-fri_sat” “Every-day”			<i>prgMode</i>
remoteSetPrgSpTimeXX	string	“hh:mm”, eg: “23:50”			remoteSetPrgSpTime00 ~ remoteSetPrgSpTime27, see <i>prgSpTimeXX</i>
remoteSetPrgCtrlModeXX	string	“On” “Off”			remoteSetPrgCtrlMode00 ~ remoteSetPrgCtrlMode27, see <i>prgCtrlModeXX</i>
remoteSetPrgSpValueXX	float				remoteSetPrgSpValue00 ~ remoteSetPrgSpValue27, see <i>prgSpValueXX</i>

**Server-side RPC (remote control)**

**remoteSetOverrideStatus**

**remoteSyncTimeRequest**

**remoteClearWiFiConfig**

**remoteRebootDevice**

**remoteWiFiFUOTA**

**remoteMcuFUOTA**

## remoteGetMemoryUsage

Table 12: Server-side RPC (remote control)

Server-side RPC	one-way   two-way	Request	Response	TA652FC-W	TA652FH-W	Memo
remote-SetOverrideStatus	one-way	{ "method": "remoteSetOverrideStatus", "params": {} }				Assign a <i>prgSpValueXX</i> to <i>spValue</i> , 0 <= XX <= 27
remoteSyncTimeRequest	one-way	{ "method": "remoteSyncTimeRequest", "params": {} }				
remoteClearWiFiConfig	one-way	{ "method": "remoteClearWiFiConfig", "params": {} }				
remoteRebootDevice	one-way	{ "method": "remoteRebootDevice", "params": {} }				
remoteWiFiFUOTA	two-way	{ "method": "remoteWiFiFUOTA", "params": { "http://192.168.1.2/x.png" } }	{ "method": "remoteWiFiFUOTA", "results": { "result": "success" } }, or { "method": "remoteWiFiFUOTA", "results": { "result": "failure", "description": "xxx" } }			
remoteMcuFUOTA (NOT implemented)	two-way	{ "method": "remoteMcuFUOTA", "params": { "http://192.168.1.1/y.png" } }	{ "method": "remoteMcuFUOTA", "results": { "result": "success" } }, or { "method": "remoteMcuFUOTA", "results": { "result": "failure", "description": "xxx" } }			
remoteGetMemoryUsage	two-way	{ "method": "remoteGetMemoryUsage", "params": {} }	{ "method": "remoteGetMemoryUsage", "results": { "iram": 123123, "spiram": 2345678 } }			



## TA652FH-W WI-FI THERMOSTAT

These references will help you learn more about TA652FH-W Wi-Fi Thermostat, operate it, and even realize your personalized Dashboard.

- [Specification](#)
- [Add to ThingsBoard](#) | [Connect to ThingsBoard](#) | [Demo Dashboards](#)
- [MQTT Device API](#)

### 5.1 TA652FH-W — Floor Heating Wi-Fi Thermostat

**Caution:**

1. Turn off all electrical devices (e.g. heater, cooler) that are connected to the unit before installation and maintenance.
2. The installer must be a trained service personnel
3. Disconnect the power supply before maintenance.
4. It must be mounted on a dry clean indoor place.
5. Do not expose this unit to moisture.
6. Do not expose this unit to dipping or splashing.

#### 5.1.1 Introduction

T65 is a controller that controls heater/cooler on or off to maintain room temperature at a desired level.

When **Heat mode** is used, **Internal sensor**, **Floor sensor** and **combined sensor** can be selected for different application.

When **Cool mode** is selected, Only **Internal sensor** will be used.

### 5.1.2 Feature List

- Voltage supply: 230Vac
- Temperature display in °C or °F
- Temperature measurable range : 0 - 50 °C
- Selection of Heat/Cool
- Adaptive control
- 7days / 5+1+1days, 1day program, no program.
- EEPROM stores all settings
- Adjustable control span
- Short cycle protection for compressor

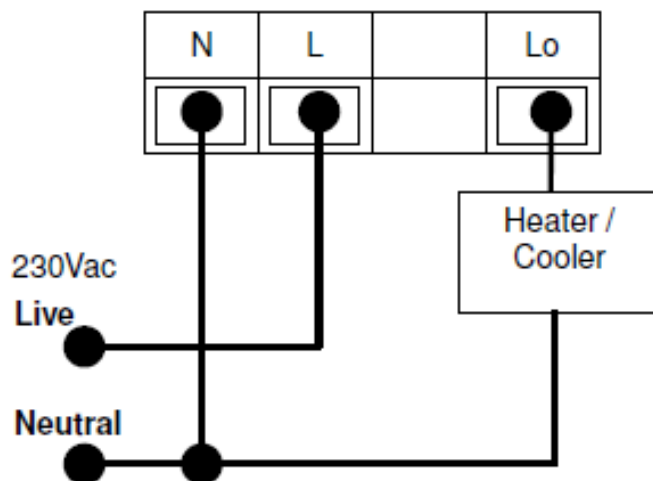
### 5.1.3 Wiring

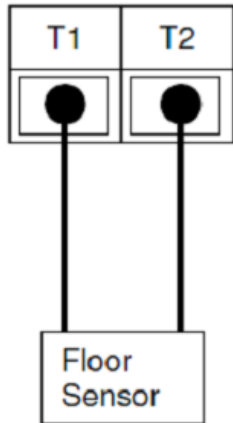
**NOTE:** Power supply of TA652FH-W is 230Vac.

Terminals	Device
L	230Vac Live
N	230Vac Neutral
LO	Heater / Cooler
T1	Floor Sensor 1
T2	Floor Sensor 2

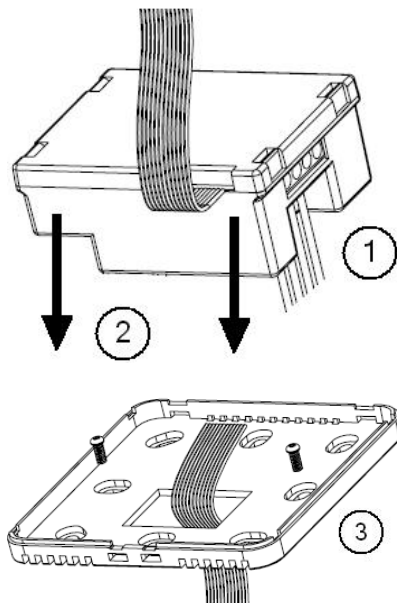
Pull all cables back into the wall beforehand to avoid trapping of wires. Do not use any metal conduits or cables provided with metal sheaths.

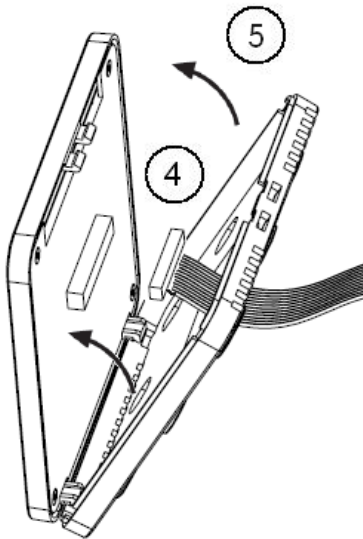
Recommend adding fuse or protective device in the live circuit.





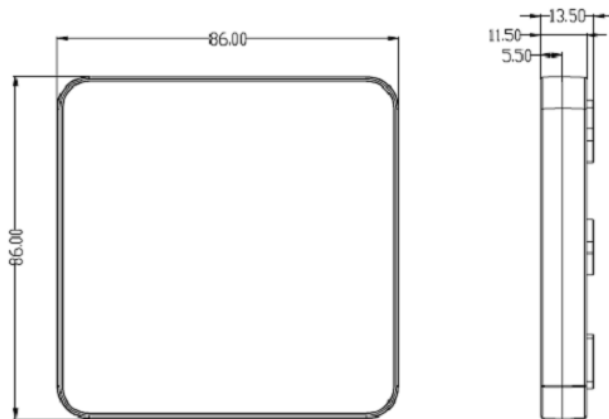
### 5.1.4 Mounting



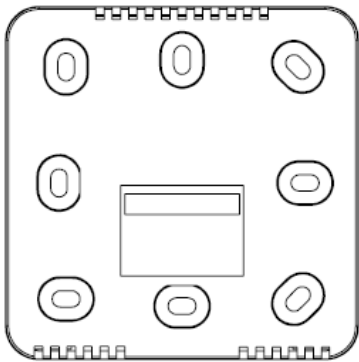


1. Wiring the terminals.
2. Put into junction box.
3. Mount the bottom plate of LCD board into junction box.
4. Connect the wire to the LCD board.
5. Assemble the Top and bottom plate of LCD Board.

### 5.1.5 Dimension in mm:

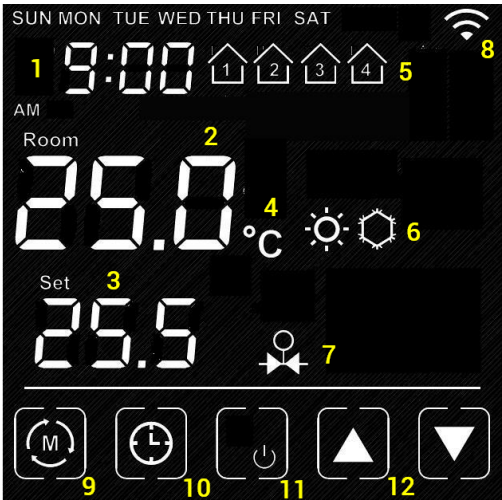







5.1.6 LCD Interface

LCD Indication













sn	item
1	Time
2	Room Temperature
3	Current Set Point
4	Temperature Unit
5	Current Program
6	Heat / Cool Mode
8	Output is On (when appear)
8	Wi-Fi (appear when connected to router)
9	Mode Key: Press to <b>internal setting 1</b> . Long hold to <b>internal setting 2</b> .
10	Clock Key: Press to <b>set clock</b> . Hold to <b>Program</b> the Schedule
11	Short Press: Fan Key, Long Hold: On/Off Key
12	Up/Down key: Adjust Set point or Value of setting.
13	Blank: the area outside of the previous five keys

### Turn On/Off the thermostat

Hold  to turn On / Off the thermostat. When the thermostat is Off. No Output will be activated.

### Clock setting

Normally the clock is automatically set once Wi-Fi is connected and synchronize for each day. So manual set is not necessary when it is online.











- Press  to start the setting
- Press  /  to change the day of week
- Press  again to confirm day of week setting and start to adjust hour
- Press  /  to change the hour
- Press  again to confirm hour setting and start to adjust minutes
- Press  /  to change the minutes
- Press  again to confirm minutes setting and start to adjust day of week
- Press [blank] to confirm or leave the clock setting. Or return after no key pressed for 20 seconds.

### Clock synchronization

When Wi-Fi is connected and time synchronization is need. Please use the App for time synchronization.



### Schedule Programming

When **1 day / 5+1+1 day / 7day program** is selected in internal setting.

- Hold  to start the setting.
- Press  /  to adjust the day of week
- Press  to confirm.
- Press  /  to adjust the time of schedule
- Press  to confirm.
- Press  /  to adjust the setpoint
- Press  to confirm.

- Press **[blank]** to confirm return.

## Override Temperature

The Set point can be adjusted by  / .




When it is in program mode, the set point will be overwritten until the next time slot.



can be pressed to release the override status.

## Internal parameter setting 1

### • Operation:

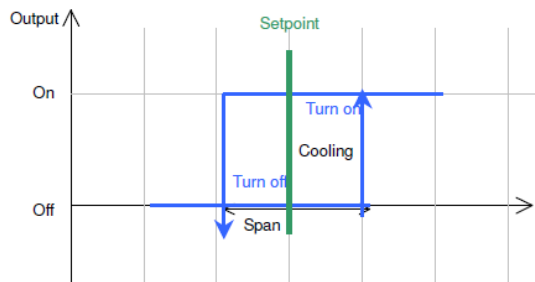
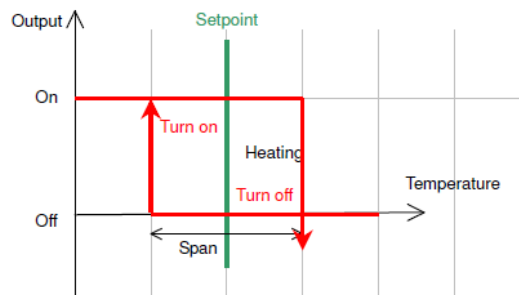
- Press  key to start the setting
- Press  /  to adjust the value
- Press **[blank]** to confirm and move to next setting

ID	Items	Value	Default Value
P00	User Interface Screen Saver	0-3	0
P01	Screen Saver Count down	0-120	20
P02	Display unit	°C / °F	°C
P03	Time Display unit	12/24	12
P04	Temperature Offset	-5°C - 5°C, -10°F-10°F	0°C
P05	Switching Differential Heat	2 - 4°C, 4 - 8°F	2°C
P06	Switching Differential Cool	2 - 4°C, 4 - 8°F	2°C
P07	Program mode	No program (0) / 1day program (1) / 5+1+1 program (2) / 7day program (3)	3
P08	Adaptive Control	Enable / Disable	Disable
P09	System Mode	Heat / Cool	Heat
P10	Sensor Mode	Internal Sensor / External Sensor / Combined mode	Internal Sensor
P11	Floor temperature limited	20-40°C, 68-104°F	40°C

### • User Interface Screen Saver:

The thermostat will go to screen saver mode after no key for certain period.

- **Mode 0:** Nothing will be displayed in screen saver mode.
- **Mode 1:** Only room temperature will be displayed in screen saver mode.
- **Mode 2:** Room temperature and Time will be displayed in screen saver mode.
- **Mode 3:** Display all in screen saver mode.
- **Screen Saver Count Down:**  
The count down time (in seconds) to screen saver mode.
- **Display Unit:**  
Temperature unit in Celsius or Fahrenheit.
- **Time Display Unit:**  
12/24.
- **Temperature offset:**  
The temperature of internal sensor can be calibrated from  $-5^{\circ}\text{C}$  -  $+5^{\circ}\text{C}$  in case there is temperature difference between actual value and thermostat.
- **Switching Differential:**  
The difference between switching the heating or controller on and off



- **Program Mode:**
  - 0: **No Program** Mode, the thermostat control the temperature simply according to single setpoint.
  - 1: **1 day** program, the thermostat control the temperature according to single schedule.
  - 2: **5+1+1 day** program, the thermostat control the temperature according to 5 +1+1 schedule (Mon to Fri, Sat, Sun).
  - 3: **7 days** program, the thermostat control the temperature according to 7day program (individual program for each day).
- **Adaptive control**  
When this function is enable, the thermostat learns the time taken to reach the desired setpoint and turn on the heating / cooling earlier so that the room temperature will reach the setpoint at desired schedule. This is no effect when **No program** is selected.

- **Heat / Cool Mode**

When **Heat mode** is selected, the thermostat control the room temperature with heating. **Room Sensor**, **Floor Sensor** or **Combined sensor** can be selected.

When **Cool mode** is selected, the thermostat control the room temperature with cooling. Only **Room Sensor**, will be used.

- **Sensor Mode**

There are 3 different settings of sensor control for Heat Mode. (For Cool Mode. Only Room sensor will be used)

- **Room sensor**

Thermostat control the room temperature based on Room Sensor

- **Floor sensor**

Thermostat control the room temperature based on Floor Sensor

- **Combined Floor-/Room sensor**




Thermostat control the room temperature based on Room Sensor. And the output will be off if floor temperature above “floor temperature limited” for protection.

- **Floor temperature limited**

It is the temperature limited for floor sensor. When the **Heat Mode** and **Combined Sensor** are selected. The output will turn off when floor sensor sense the temperature to be higher than **floor temperature limited**.

## Internal parameter setting 2

- **Operation:**

- Hold  key to start the setting
- Press  /  to adjust the value
- Press **[blank]** to confirm and move to next setting

ID	Items	Value	Default Value
P19	Clear Wi-Fi Configuration	Yes or No	No
P20	Clear Parameter setting (restore default)	Yes or No	No

- **Clear Wi-Fi Configuration:**

When set to yes, the SSID and Password stored in the thermostat will be cleared so another SSID and Password can be set again.

- **Clear Parameter setting:**

When set to yes, all internal parameter setting will be restored to default value in next power on (reset)

### Minimum off time

The minimum off time for Heat mode is 5 seconds and 4 minutes for Cool mode.

### Technical Data

Power supply:	195-250 Vac
Relay Contact Voltage:	230Vac Max. 50/60 Hz
Relay Contact Current:	16A Max.
Sensing Element:	103AT
Terminals:	2 sq. mm Cable
Operating Temperature:	32 - 122 °F / 0 - 50 °C
Storage Temperature:	23 - 122 °F / -5 - 50 °C
Operating Humidity:	5-95%RHnon-condensing

## 5.2 Add TA652FH-W to ThingsBoard

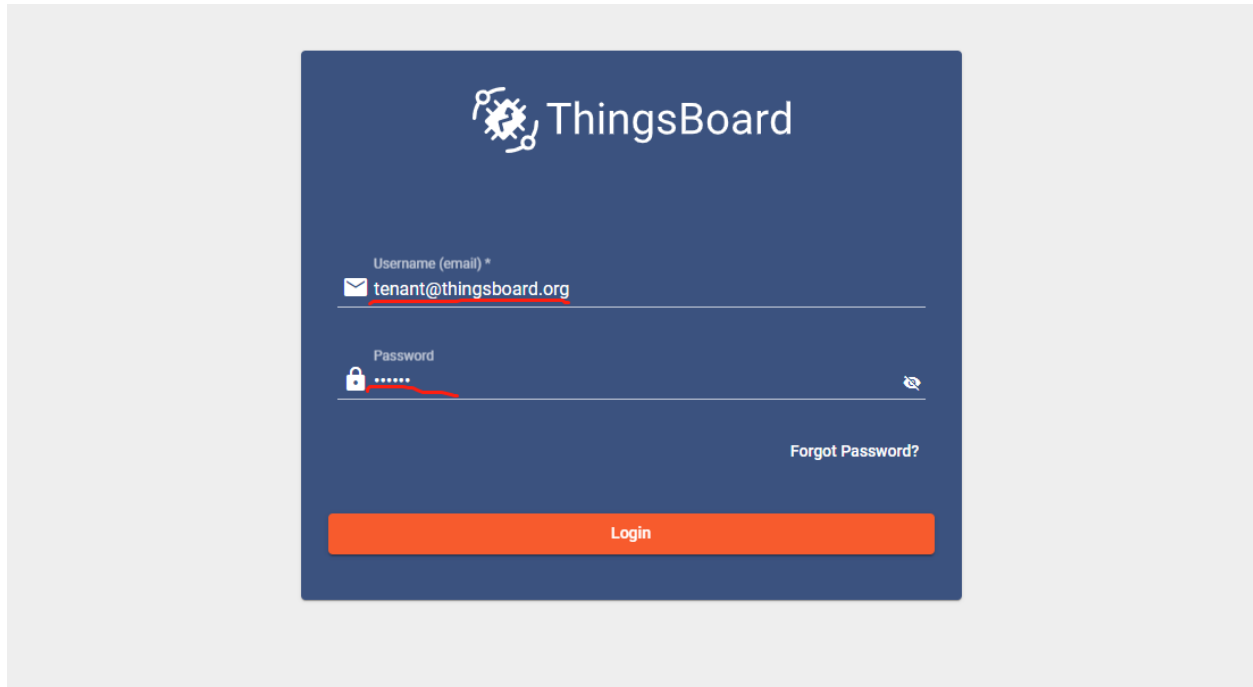
---

### Tip:

- This section applies to the situation where you add TA652FH-W to ThingsBoard Server.
  - If you are adding the first Avantec HVAC device to ThingsBoard Server, please refer to [Get Started](#).
- 

### 5.2.1 Step 1. Tenant Login

- Open ThingsBoard Web UI in browser, e.g. <http://localhost:8080>
- Tenant Administrator login ThingsBoard.



Tenant default username and password, refer to *Some important parameters*.

## 5.2.2 Step 2. Import Detail Dashboard of TA652FH-W

See *Import TA652FH-W Detail Dashboard*.

## 5.2.3 Step 3. Import List Dashboard of TA652FH-W

See *Import TA652FH-W List Dashboard*.

## 5.2.4 Step 4. Provision TA652FH-W device

### Step 4.1 Add device

- **Devices** -> **+** -> **Add new device** -> **Popup Dialog** -> Input **Name, Label & Description**, select **device profile** -> **Add**.

**Add new device**

1 Device details 2 Credentials Optional 3 Customer Optional

Name \* A8:48:FA:57:D5:20

Label AVANTEC Headquarters

Device profile \* TA652FH-W Thermostat

☒ Select existing device profile ☐ Create new device profile

☐ Is gateway

Description A Thermostat for floor heating

Next: Credentials

Cancel Add

Field	Value
Name*	My device name, e.g. TA652FH-W-TB, A8:48:FA:57:D5:20
Device profile*	<b>TA652FH-W Thermostat</b>
Label	My device label, e.g. Avantec Manufacturing Plant
Description	My device description, e.g. A Thermostat for floor-heating

**Note:** The field with \* must be filled in.

- Now my device should be listed first, since the table sort devices using the time of the creation by default.

**Devices**

Device profile All

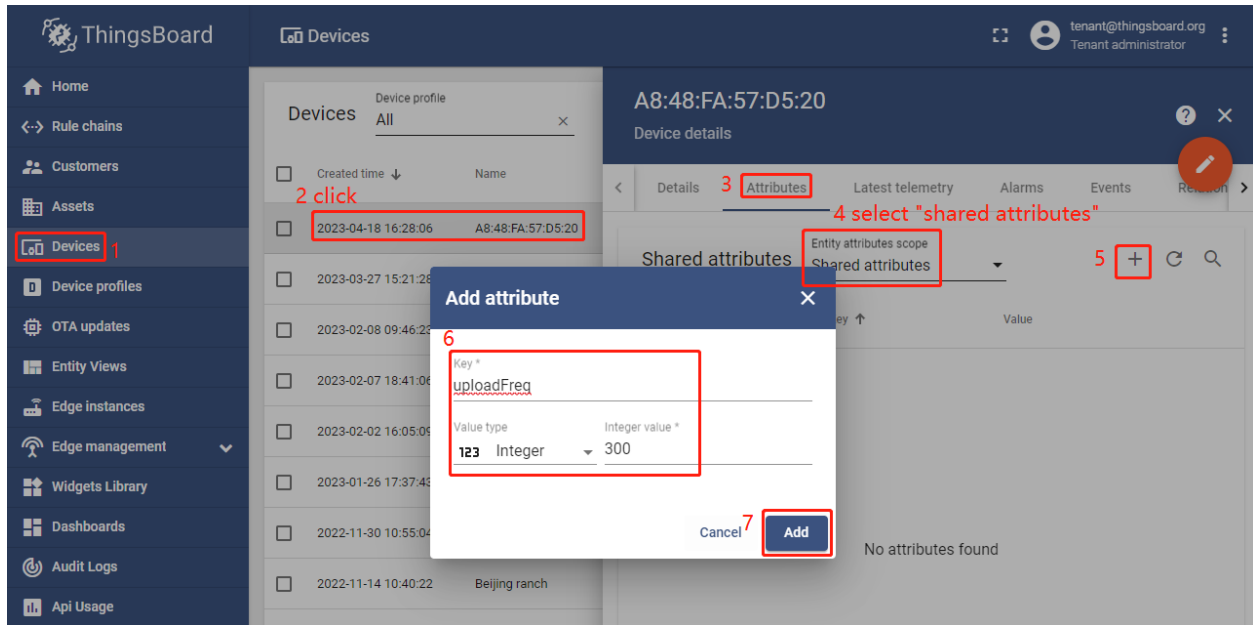
<input type="checkbox"/>	Created time ↓	Name	Device profile	Label	Customer	Publ
<input type="checkbox"/>	2023-04-18 16:28:06	A8:48:FA:57:D5:20	TA652FH-W Thermostat	AVANTEC Headquarters		
<input type="checkbox"/>	2023-03-27 15:21:28	A8:48:FA:57:60:A4	TA652FH-W Thermostat	Fai sir's		
<input type="checkbox"/>	2023-02-08 09:46:23	9C:9C:1F:19:4D:98	TA652FC-W Thermostat	left	Customer A	
<input type="checkbox"/>	2023-02-07 18:41:06	9C:9C:1F:18:72:B0	TA652FC-W Thermostat	right	Customer A	
<input type="checkbox"/>	2023-02-02 16:05:09	24:0A:C4:2C:EB:D4	TA652FC-W Thermostat	old_board	Customer A	
<input type="checkbox"/>	2023-01-26 17:37:43	MY_DEVICE_NAME	TA652FC-W Thermostat	MY_DEVICE_LABEL		
<input type="checkbox"/>	2022-11-30 10:55:04	Thermometer A-1	default	Thermometer A-1		

Items per page: 10 1 - 10 of 20



### Step 4.2 Add shared attributes of new device

- **Devices** → Click *my device* → **Attributes** → **Shared attributes** → + → **Popup Dialog** → Input Key, Value type & value → **Add**



Please add the following Shared attributes of **TA652FH-W**:

Table 1: Add shared attributes of TA652FH-W

Key*	Value Type*	Value*	Memo
<i>uploadFreq</i>	Integer	300	5*60. Telemetry per uploadFreq seconds
<i>uploadThresh-old</i>	Double	1.5	1.5°C. If the temprature (Telemetry data) change exceeds it, up-load immediately!
<i>syncTimeFreq</i>	Integer	86400	24*3600. Sync time per syncTimeFreq seconds
<i>timezone</i>	Integer	480	<b>Please replace with your value.</b> The time offset from UTC, minutes. For example Hongkong is UTC+8:00 time zone, this offset is 480 minutes (8*60)
<i>timeNTPServer</i>	String	pool.ntp.org	SNTP Server URL, e.g. pool.ntp.org, 0.pool.ntp.org, 1.pool.ntp.org, uk.pool.ntp.org, hk.pool.ntp.org, time.nist.gov, ...

**Note:** The field with \* must be filled in.

- Now the shared attributes of my device is like:

The screenshot displays the ThingsBoard web interface. On the left, the 'Devices' menu item is selected. The central panel shows a list of devices under the 'All' profile. A specific device with MAC address A8:48:FA:57:D5:20 is highlighted. The right panel shows the details for this device, specifically the 'Attributes' tab. A table of 'Shared attributes' is visible, listing various system and device parameters.

Entity attributes scope	Key	Value
Shared attributes	Last update time	
Shared attributes	syncTimeFreq	86400
Shared attributes	timeNTPServer	pool.ntp.org
Shared attributes	timezone	480
Shared attributes	uploadFreq	300
Shared attributes	uploadThreshold	1.5

You may also use:

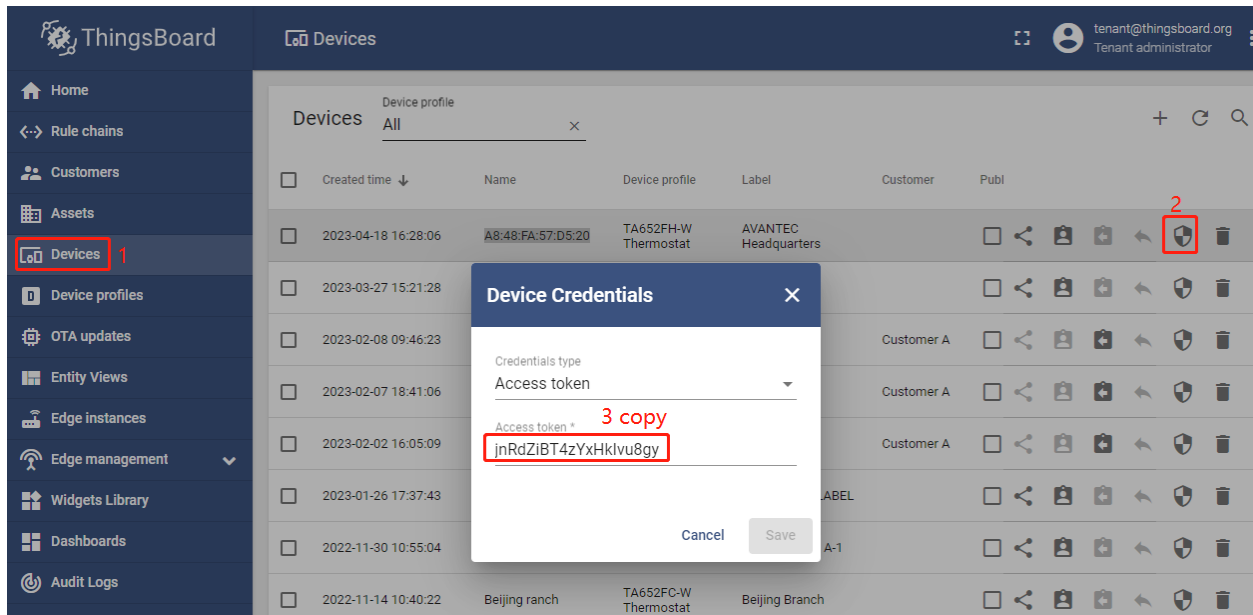
- [Bulk provisioning](#) to provision multiple devices from a CSV file using UI.
- [Device provisioning](#) to allow device firmware to automatically provision the device, so you don't need to configure each device manually.
- [REST API](#) to provision devices and other entities programmatically.

### 5.2.5 Step 5. Connect TA652FH-W device

#### Step 5.1 Copy credentials of new device

To connect the device you need to get the device credentials first. ThingsBoard supports various device credentials. We recommend using default auto-generated credentials which is access token for this guide.

- **Devices** -> **Manage credentials (icon)** -> **Popup Dialog** -> **Select Access Token, Ctrl + C**.



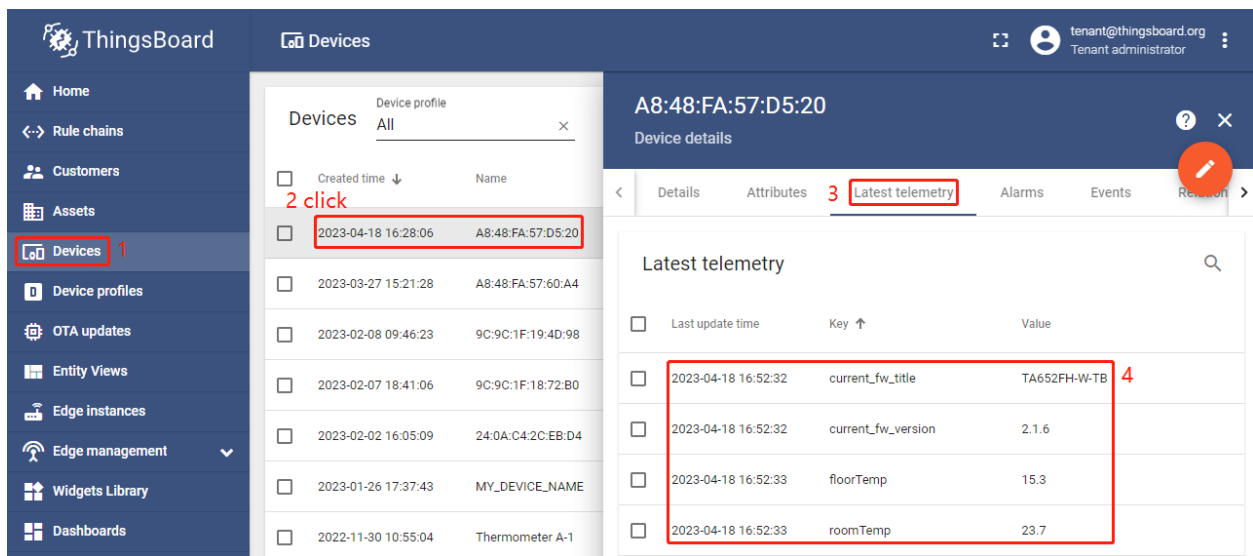
**Tip:** The Credentials (Access Token), which you need to use when you're configuring your hardware, for example, *j9JiCkID9E7uE1WhKxnc, lMTQLZ7VSRQSD7ls*.

## Step 5.2 Connect device to ThingsBoard

See *Connect TA652FH-W to ThingsBoard*.

## Step 5.3 Publish data to ThingsBoard

Now your device has already published telemetry data to ThingsBoard. You should immediately see them in the Device Telemetry Tab:



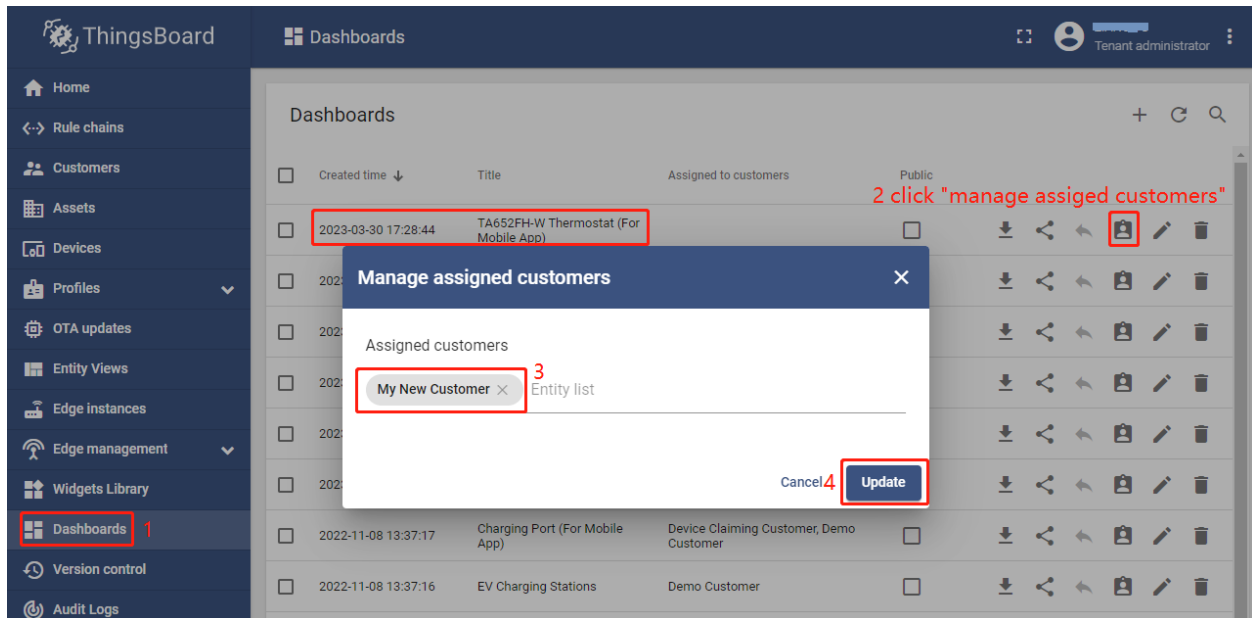
## 5.2.6 Step 6. Assign Device and Dashboards to Customer

One of the most important ThingsBoard features is the ability to assign Dashboards to Customers. You may assign different devices to different customers. Then, you may create a Dashboard(s) and assign it to multiple customers. Each customer user will see his own devices and will not be able to see devices or any other data that belongs to a different customer.

Refer to *Step 7.1 Create customers*, *Step 7.4 Create customer user* & *Step 7.5 Activate customer user*.

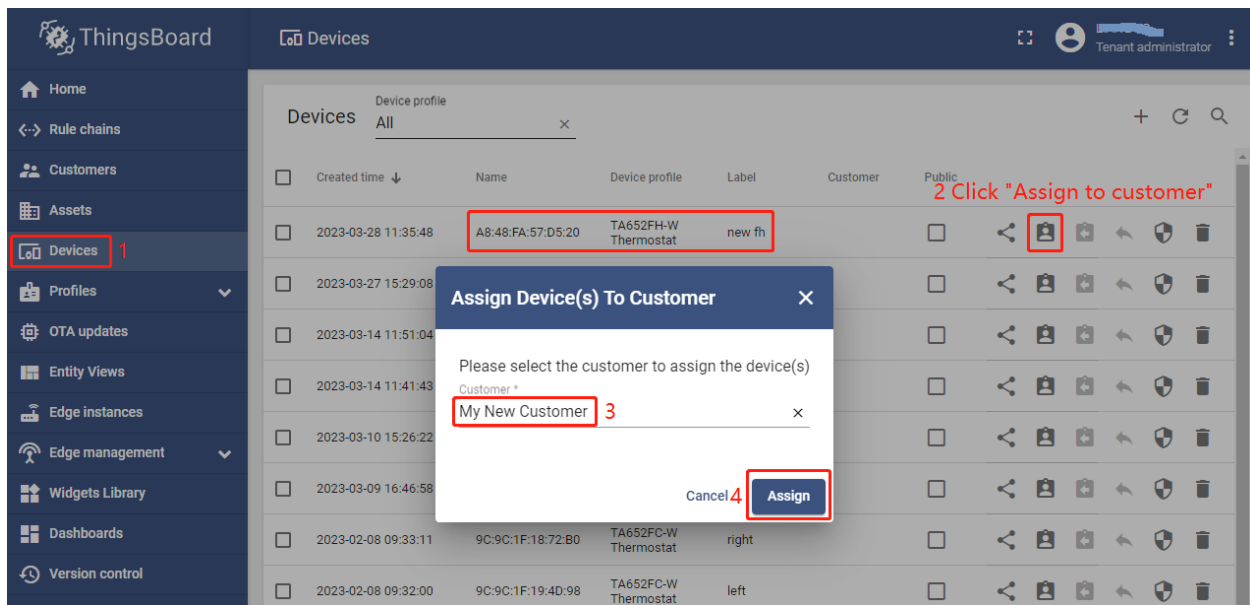
### Step 6.1 Assign dashboards of TA652FH-W to Customer

- Assign *Detail dashboard* to Customer: **Dashboards** → Click **Manage assigned customers** (icon) in *Detail dashboard* line → **Popup Dialog** → Select *My New Customer* → **Update**.

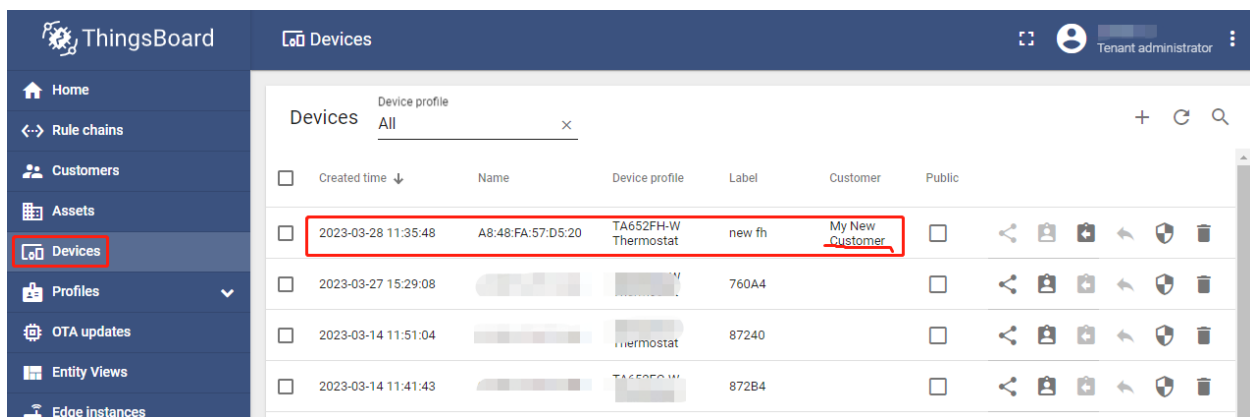


- Assign *List dashboard* to Customer: **Dashboards** → Click **Manage assigned customers** (icon) in *List dashboard* line → **Popup Dialog** → Select *My New Customer* → **Update**.





- It's like this now.



### 5.2.7 Step 7. Open Dashboards of TA652FH-W

- You are logged in as a Customer User or a Tenant user.
- **Dashboards** → click *my list dashboard*

ThingsBoard Dashboards

2 click

Created time ↓	Title	Assigned to customers	Public
2023-03-30 17:28:32	TA652FH-W Thermostat List	My New Customer	
2023-02-02 12:19:22			
2023-01-09 14:34:32			
2023-01-03 17:45:42		mer	
2023-01-03 17:16:01			
2022-11-08 13:37:17			

Items per page: 10 1 - 10 of 22

- Select my device → **Settings** (icon)

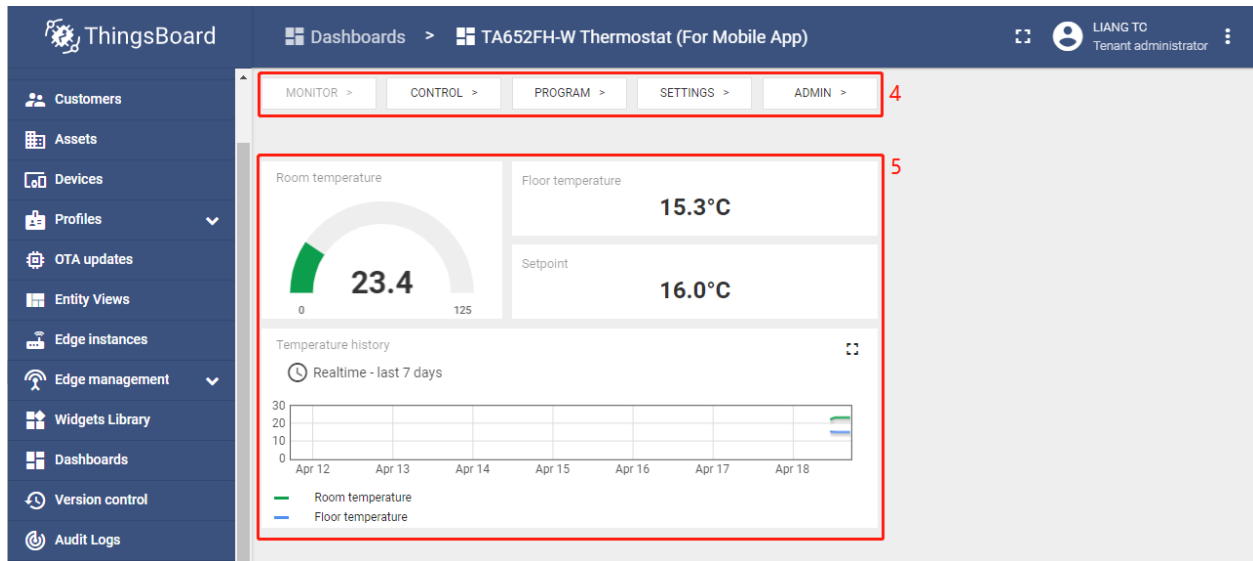
ThingsBoard Dashboards > TA652FH-W Thermostat List

TA652FH-W Thermostat List TA652FH-W Thermostat List Entities Realtime - last minute

Device name ↑	Label	Type	active	Room Temp	Floor Temp	Setpoint	Unit
A8:48:FA:57:5C:68	75C68	TA652FH-W Thermostat	false	23.7	26.2	21	°C
A8:48:FA:57:60:A4	760A4	TA652FH-W Thermostat	false	23.9	26.6	16	°C
A8:48:FA:57:D5:20	new fh	TA652FH-W Thermostat	false	23.4	15.3	16	°C

Items per page: 10 1 - 4 of 4

- Switch page → Operation



See *TA652FH-W Demo Dashboards Usage*.

### 5.2.8 Your feedback

Don't hesitate to star Avantec on [github](#) to help us spread the word.

## 5.3 Connect TA652FH-W to ThingsBoard

See *Connect TA652FC-W to ThingsBoard*.

## 5.4 TA652FH-W Thermostat – Demo Device Profile Usage

### 5.4.1 Import device profile

**Tip:** A *Device Profile file* can only be imported once. If you have already imported it, you do not need and cannot repeat the import.

If you have already imported it, you can skip this step.

- Download `ta652fh_w_thermostat.json`.
- **Profiles** -> **Device profiles** -> **+** -> **Popup dialog: Import device profile** -> Drag and drop *my device profile File* -> **Import**.



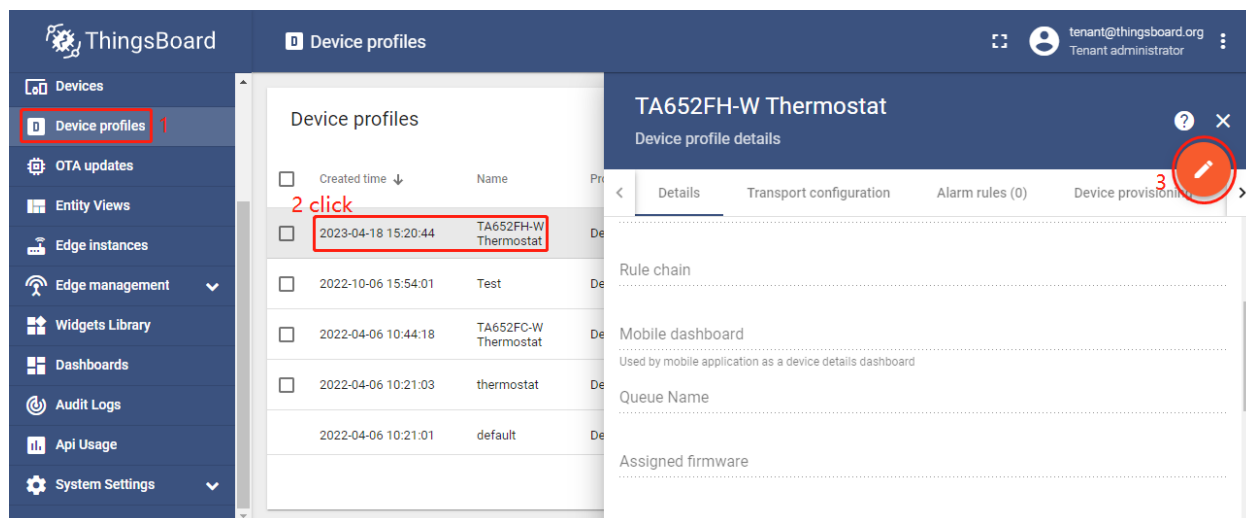
The top screenshot shows the ThingsBoard interface with the 'Device profiles' menu item selected. A dialog box titled 'Import device profile' is open, showing a file named 'ta652fh\_w\_thermostat.json' selected for import. The bottom screenshot shows the 'Device profiles' table with the imported profile listed.

Created time	Name	Profile type	Transport type	Description	Default
2023-04-18 15:20:44	TA652FH-W Thermostat	Default	Default	Avantec Thermostat Floor Heating	<input type="checkbox"/>
2022-10-06 15:54:01	Test	Default	Default	tttt	<input type="checkbox"/>
2022-04-06 10:44:18	TA652FC-W Thermostat	Default	Default		<input type="checkbox"/>
2022-04-06 10:21:03	thermostat	Default	Default	Thermostat device profile	<input type="checkbox"/>
2022-04-06 10:21:01	default	Default	Default	Default device profile	<input checked="" type="checkbox"/>

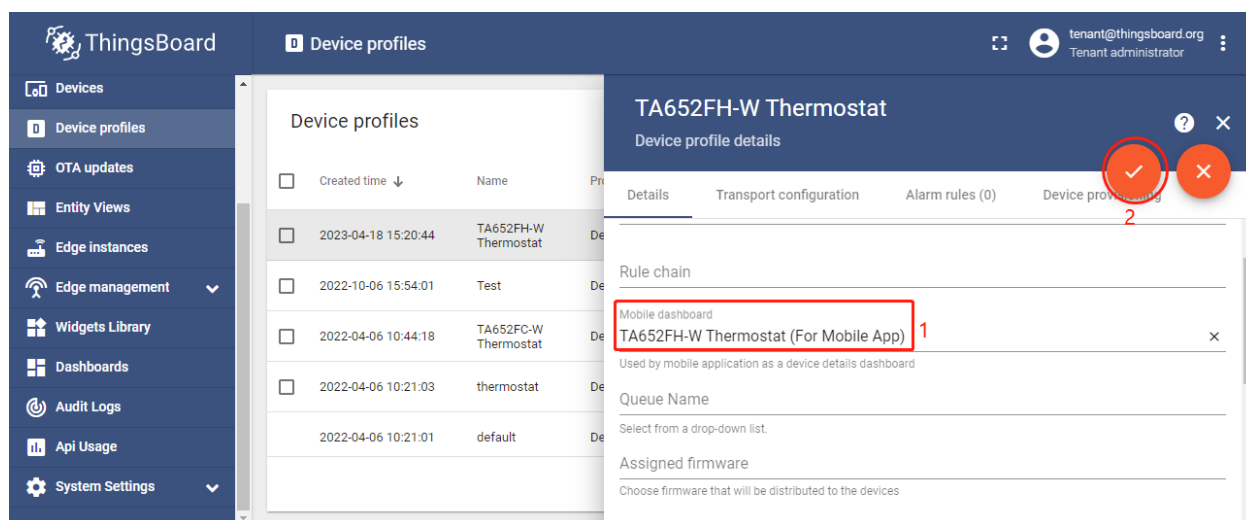
## 5.4.2 Modify device profile's mobile dashboard

Device profile's mobile dashboard is for ThingsBoard Mobile Application or ThingsBoard PE Mobile Application.

- Profiles → Device profiles → click *my device profile* → **Toggle edit mode** (red icon)



- Modify *Mobile dashboard* → **Apply changes** (red icon)



These values are shown in the following table:

Field	Value
Mobile dashboard	TA652FH-W Thermostat (For Mobile App)

### 5.4.3 Clear device profile's mobile dashboard

Sometimes if TA652FH-W Thermostat device profile's mobile dashboard is cleared, TA652FH-W Thermostat (For Mobile App) can only be deleted.

- **Profiles** → **Device profiles** → click *my device profile* → **Toggle edit mode** (red icon)

The screenshot shows the ThingsBoard interface. On the left sidebar, the 'Device profiles' menu item is highlighted with a red box and a '1' next to it. In the main panel, the 'Device profiles' table is visible. A row with the name 'TA652FH-W Thermostat' and creation time '2023-04-18 15:20:44' is highlighted with a red box and a '2 click' label. To the right, the 'TA652FH-W Thermostat' details page is open. The 'Device provisioning' tab is selected, and a red circle with a pencil icon and a '3' is placed over the 'Apply changes' button.

- Clear *Mobile dashboard* → **Apply changes** (red icon)

This screenshot shows the 'TA652FH-W Thermostat' details page with the 'Device provisioning' tab selected. The 'Mobile dashboard' field is highlighted with a red box and a '1' next to it. The field contains the text 'TA652FH-W Thermostat (For Mobile App)'. A red circle with a checkmark icon and a '2' is placed over the 'Apply changes' button. The left sidebar shows the 'Device profiles' menu item highlighted with a red box and a '1' next to it.

## 5.5 TA652FH-W Demo Dashboards Usage

### 5.5.1 Overview

There are some dashboards related to TA652FH-W, namely TA652FH-W Thermostat List, TA652FH-W Thermostat (For Mobile App) and Office center - TA652FH-W Thermostats. We open the former to start operating TA652FH-W.

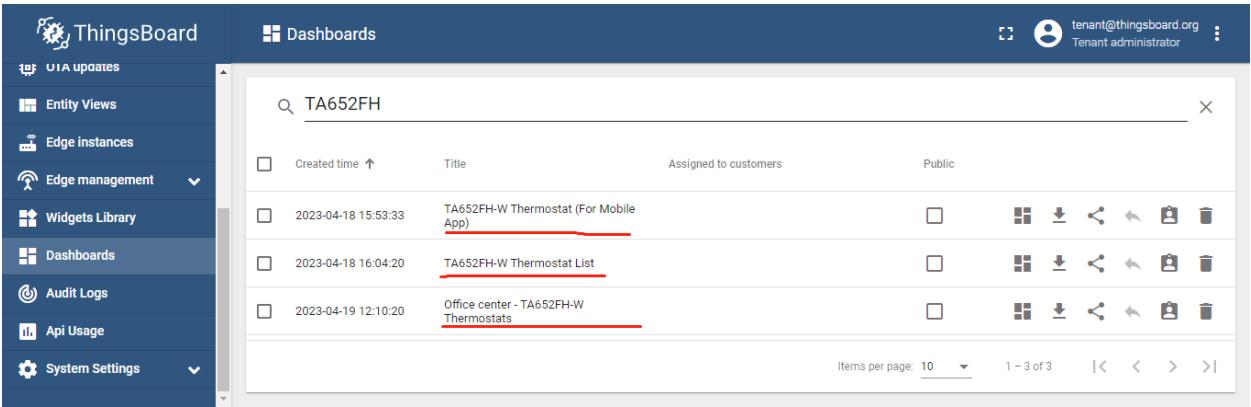


Table 2: TA652FH-W Demo Dashboards

Dashboard	Description	For Web UI	For Mobile App	Entry*
TA652FH-W Thermostat List*	list	Yes	No	Yes
TA652FH-W Thermostat (For Mobile App)	details	Yes	Yes	No
Office center - TA652FH-W Thermostats**	list & details	Yes	Yes	Yes

Hint:

- If *Entry* is *Yes*, then directly enter the Dashboard and there will be data displayed.
- If *Entry* is *No*, there will be no data display when entering this Dashboard directly, and you need to jump to this Dashboard from other Dashboards.
- TA652FH-W Thermostat List depends on TA652FH-W Thermostat (For Mobile App).
- Office center - TA652FH-W Thermostats can be used independently.

5.5.2 TA652FH-W Thermostat List

Dashboard states

Default state

Default state is root state.

- **Dashboard bar:**

- **TA652FH-W Thermostat List** : Click here to skip to **root state**. Since **default state** is *root state*, click here and there is no response.
- : Click the two ICONS in the upper left corner to display the page in full screen.
- **Realtime - last minute** : Edit time window.

- **Thermostats widgets:**

- **Fields:**

- \* Device name, Label, Type, active.
- \* Room temperature, Floor Temperature, Setpoint, Unit: Refer to *Monitor state*.

- **Actions:**

- \* : skip to *TA652FH-W Thermostat (For Mobile App)*.
- \* : Popup dialog to editing a device's label.

## Import List Dashboard

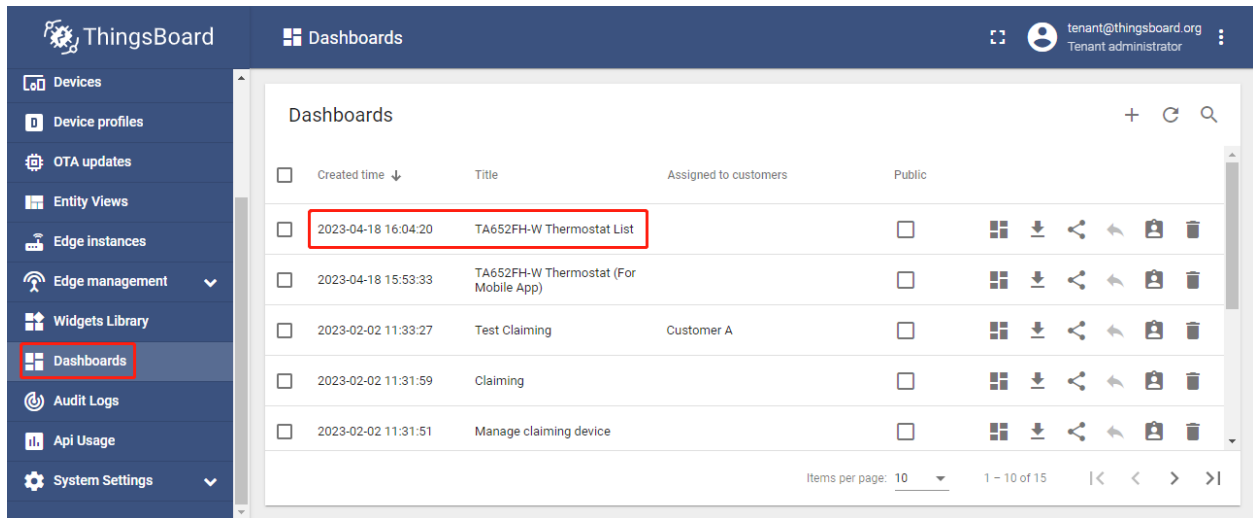
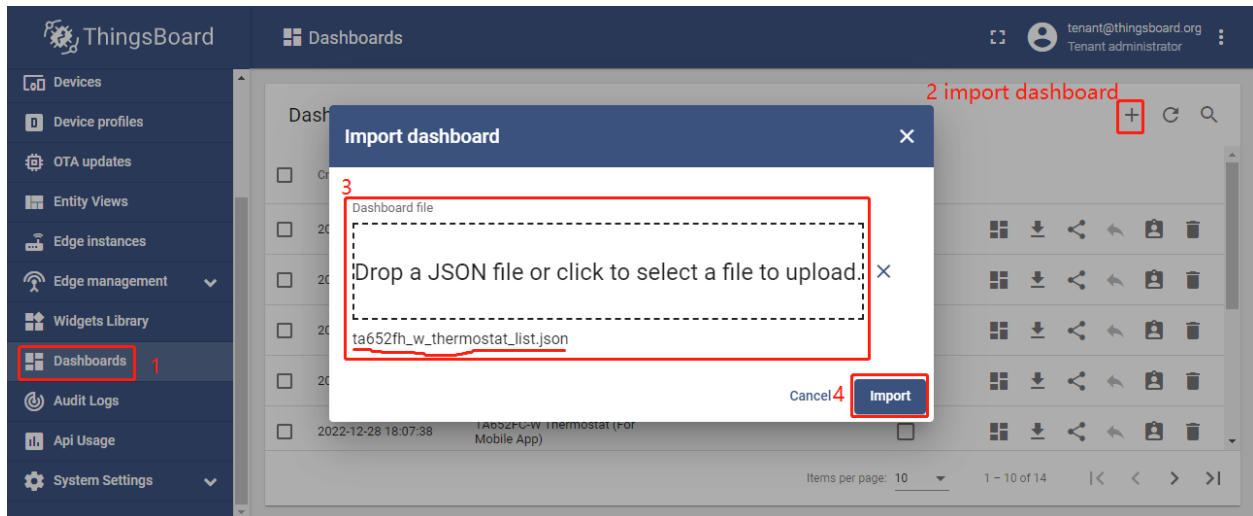
**Tip:** A Dashboard file can only be imported once. If you have already imported it, you do not need and cannot repeat the import.

If you have already imported it, you can skip this step.

In order to use this dashboard, you must to create TA652FH-W Thermostat Device Profile and TA652FH-W Thermostat (For Mobile App). If they don't exist, you can import them. See *Import Device Profile of TA652FH-W Thermostat* or *Import TA652FH-W Detail Dashboard*.

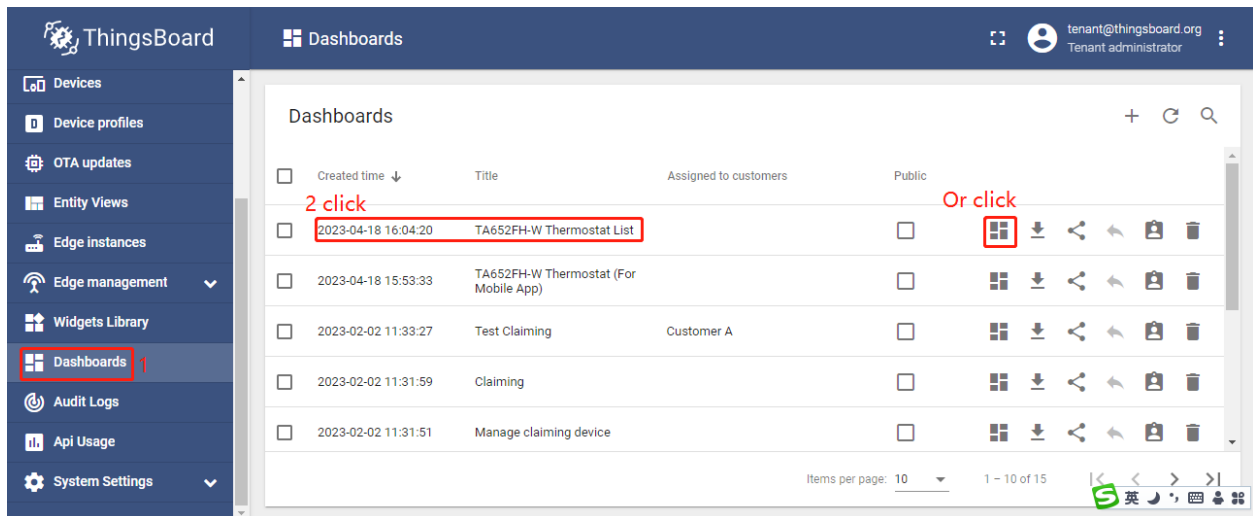
First, you can import this dashboard.

- Download `ta652fh_w_thermostat_list.json`.
- **Dashboards** → + → **Popup dialog: Import dashboard** → Drag and drop *list dashboard File* → **Import**.



Next, modify a action's target dashboard and target dashboard state.

- **Dashboards** → Click *my list dashboard*



- **Edit** (red icon on the bottom and right)

ThingsBoard

Dashboards > TA652FH-W Thermostat List

tenant@thingsboard.org  
Tenant administrator

TA652FH-W Thermostat List

TA652FH-W Thermostats

Device name ↑	Label	Type	active	Room Temp	Floor Temp	Setpoint	Unit
A8:48:FA:57:60:A4	Fai sir's	TA652FH-W Thermostat	false	22.6	23.3	16	°C
A8:48:FA:57:D5:20	new fh	TA652FH-W Thermostat	false	22.1	15.3	16	°C
Plant Floor Heating	Avantec Plant	TA652FH-W Thermostat	false	20	22.3	16	°C

Items per page: 10 1 - 3 of 3

3

Powered by

- Enter *Edit Dashboard Mode* → **Edit Widget** (icon)

ThingsBoard

Dashboards > TA652FH-W Thermostat List

tenant@thingsboard.org  
Tenant administrator

TA652FH-W Thermostat List

TA652FH-W Thermostats

Device name ↑	Label	Type	active	Room Temp	Floor Temp	Setpoint	Unit
A8:48:FA:57:60:A4	Fai sir's	TA652FH-W Thermostat	false	22.6	23.3	16	°C
A8:48:FA:57:D5:20	new fh	TA652FH-W Thermostat	false	22.1	15.3	16	°C
Plant Floor Heating	Avantec Plant	TA652FH-W Thermostat	false	20	22.3	16	°C

Items per page: 10 1 - 3 of 3

4

Powered by

- **Action** → **Edit Action** (icon)

ThingsBoard

Dashboards > TA652FH-W Thermostat List

tenant@thingsboard.org  
Tenant administrator

TA652FH-W Thermostat List

TA652FH-W Thermostats

Device name ↑	Label	Type
A8:48:FA:57:60:A4	Fai sir's	TA652FH-W Thermostat
A8:48:FA:57:D5:20	new fh	TA652FH-W Thermostat
Plant Floor Heating	Avantec Plant	TA652FH-W Thermostat

5

New Entities table

Entities table

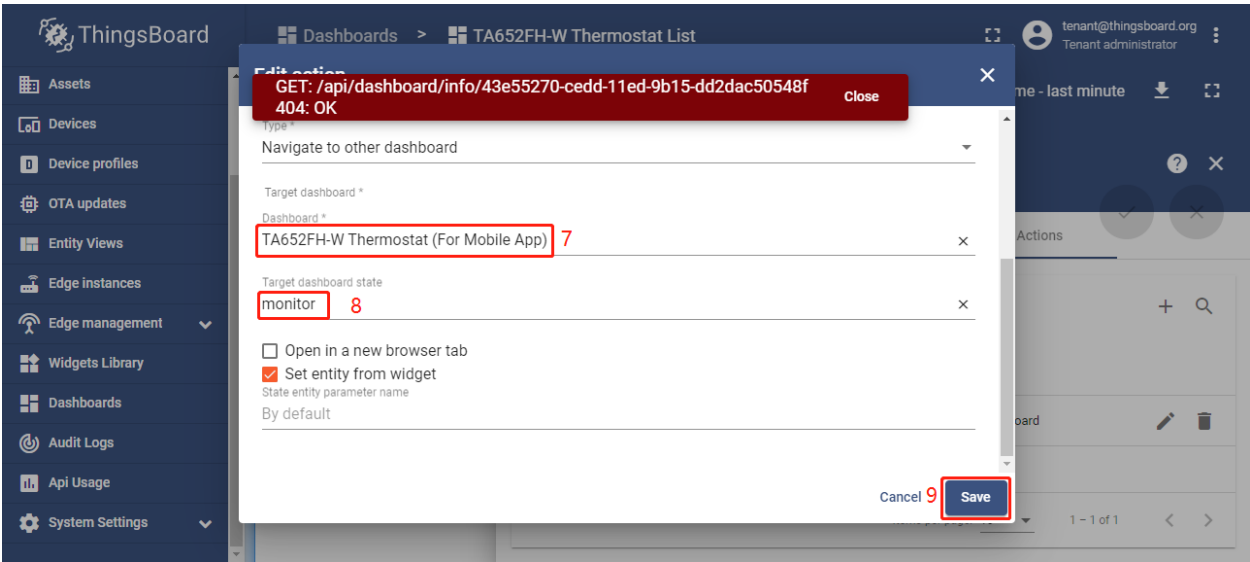
Data Settings Advanced

Actions

Action source ↑	Name	Icon	Type
Action cell button	to_ta652fh-w_detail		Navigate to other dashboard

6

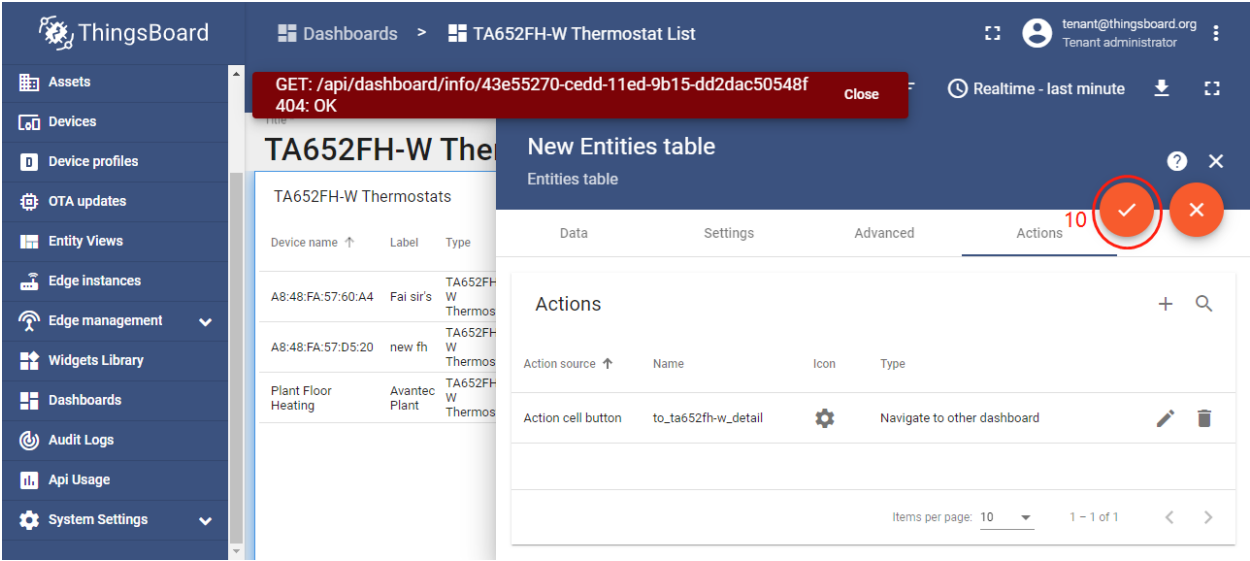
- Modify **Target dashboard** -> modify **Target dashboard state** -> **Save**



These values are shown in the following table:

Field	Value
Target dashboard	TA652FH-W Thermostat (For Mobile App)
Target dashboard state	monitor


- **Apply changes** (red icon)



- **Apply changes** (red icon on the bottom and right)



## Update List Dashboard

- First, delete this dashboard: **Dashboards** → Click  in the row of TA652FH-W Thermostat List → **Popup dialog: Are you sure you want to delete ...? → Yes.**

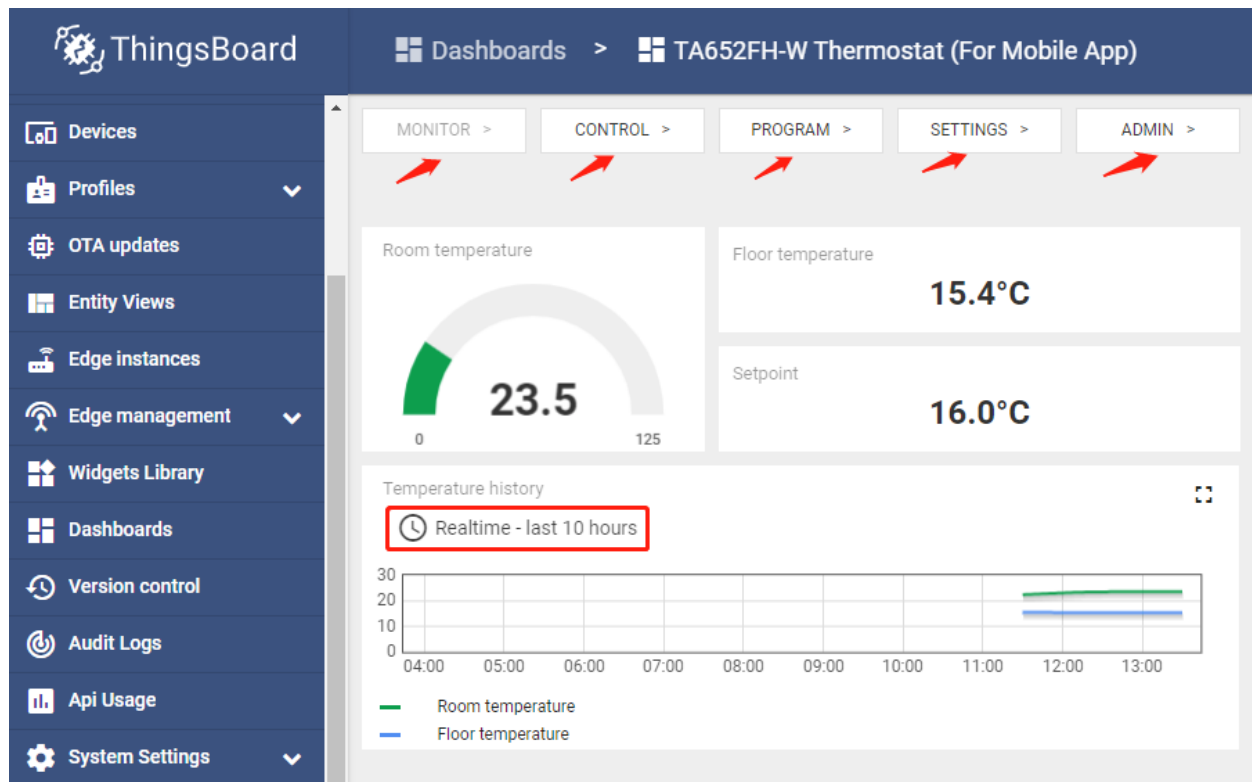
- Next, *import TA652FH-W List Dashboard.*

### 5.5.3 TA652FH-W Thermostat (For Mobile App)


#### Dashboard states

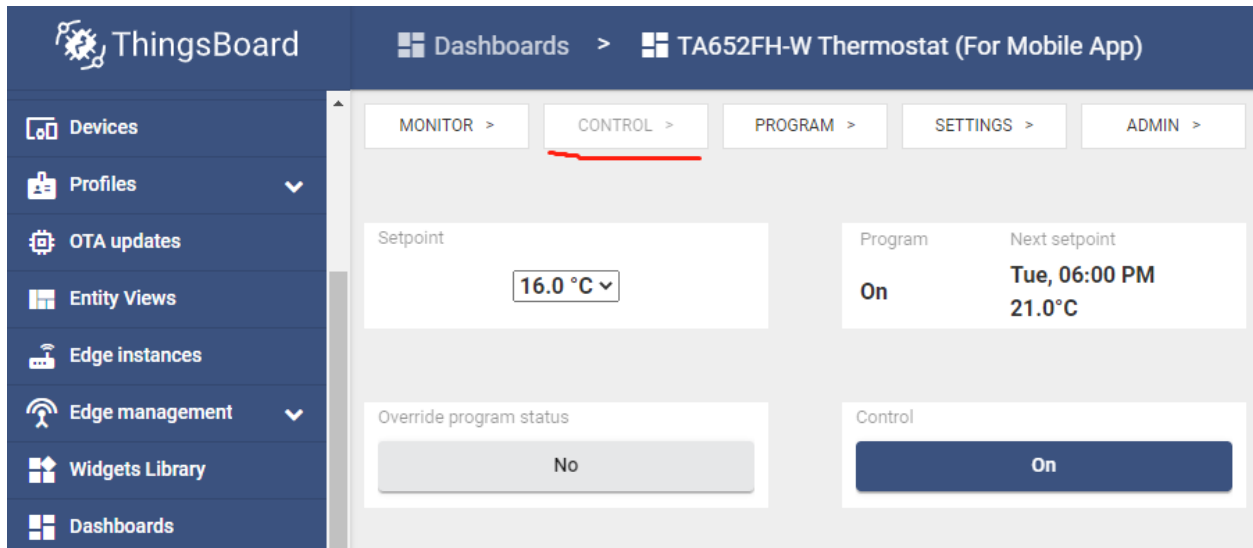
#### Monitor state

Monitor state is root state.



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Widget	Description
MONITOR	skip to <i>Monitor state</i>
CONTROL	skip to <i>Control state</i>
PROGRAM	skip to <i>Program state</i>
SETTINGS	skip to <i>Settings state</i>
ADMIN	skip to <i>Admin state</i>
Room Temperature	room temperature
Floor Temperature	floor temperature
Setpoint	current setpoint value
Temperature history	<p>Room temperature &amp; Change Over Sensor temperature history. Click  to edit this timewindow. Refer to <i>Default state</i></p>

**Control state**

- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

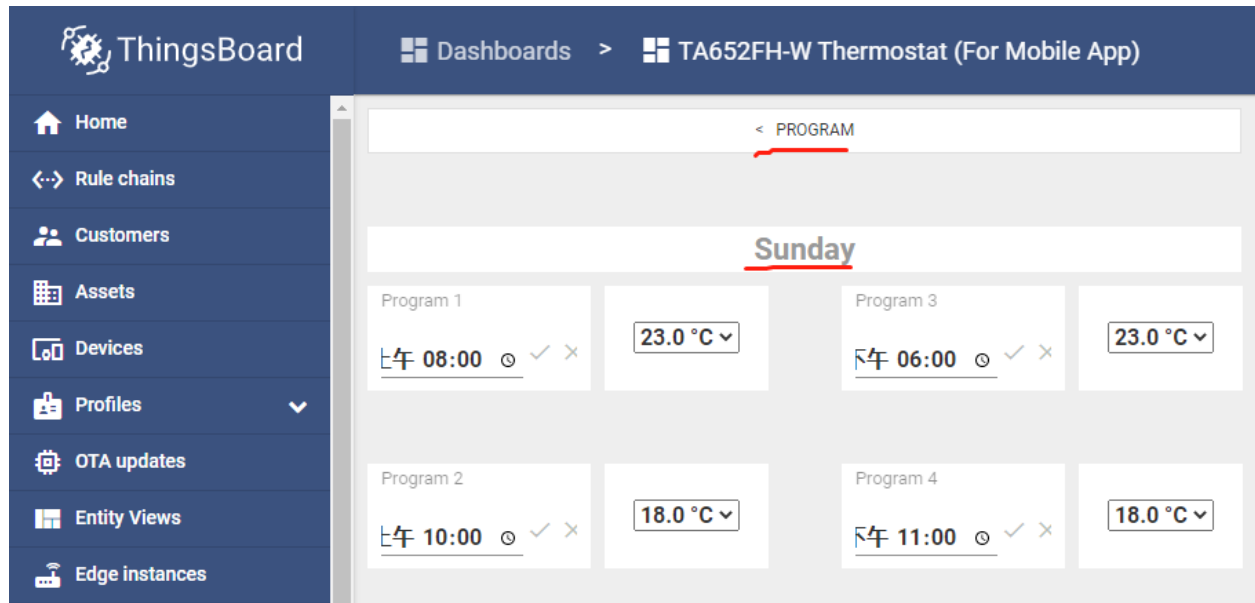
Widget	Description
Setpoint	If you adjust <i>setpoint</i> , <i>override program status</i> is YES (true)
Program	program on or off
PRG next setpoint	next program time & setpoint
Override program status	“YES”(true) or “NO”(false)
Control Mode	“Off” or “On”

## Program state

- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Program Mode	Description
NO PROGRAM	Program disabled
1 DAY (MON)	Using 4 set points of Monday every day
1+5+1 (SUN+MON+SAT)	Using 4 set points of Monday from Monday to Friday
7 DAYS (SUN~SAT)	Using 4 set points every day
Sunday, ...	Skip to <i>Program_setpoints state</i>

## Program\_setpoints state



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

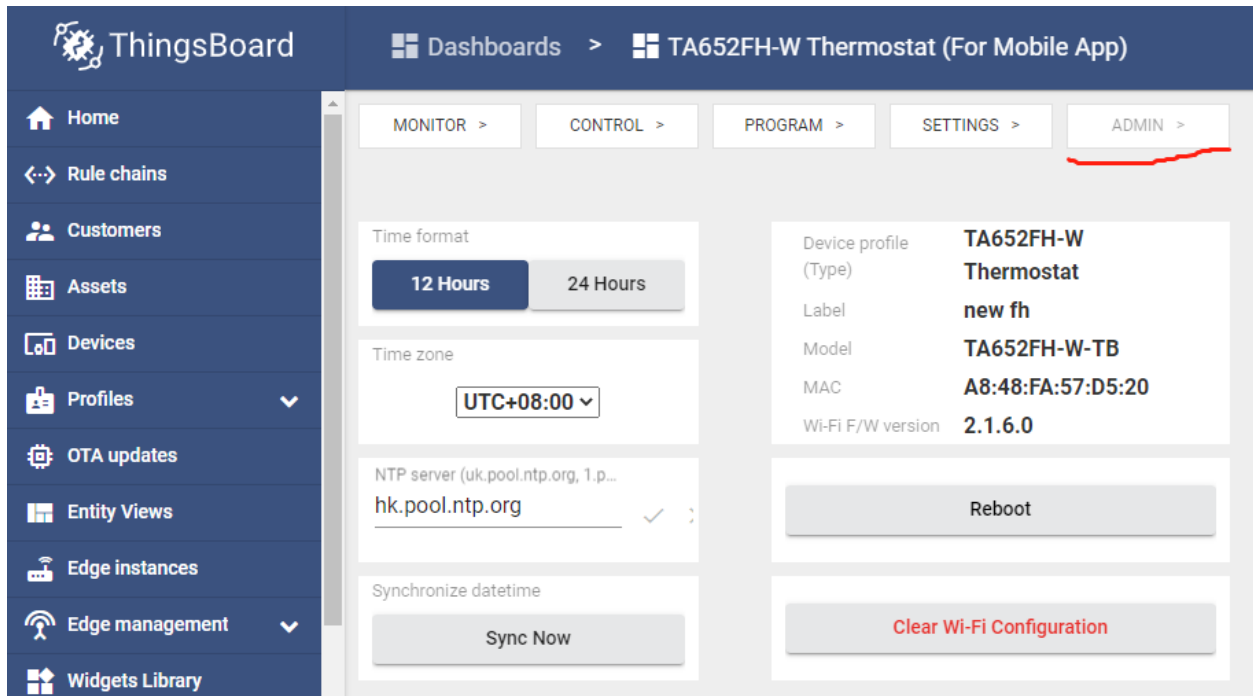
Widget	Description
Program 1 ~ Program 4	time, hour:minute
Setpoint 1 ~ Setpoint 4	setpoint value, temperature

## Settings state

- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Widget	Description
Temp Unit	“°C” or “°F”. <b>Reboot the device to take effect</b>
Adaptive control	Enabled or disabled
System Mode	“Heat” or “Cool”
Sensor Mode	<p>“Internal” / “External” / “Combined” sensor can be selected when it is in “Heat” mode.</p> <p>Only “Internal” Sensor will be used when it is in “Cool” mode.</p>
Floor temperature limited	external sensor temperature offset
Temp Offset(Internal Sensor)	Internal sensor temperature offset
Switching Diff Heating	Switching differential heating
Switching Diff Cooling	Switching differential cooling

## Admin state



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Widget	Description
Time Format	“12 Hours” or “24 Hours”
Timezone	See <i>Step 4.2 Add shared attributes of new device</i>
NTP Server	SNTP protocol server URL, e.g. pool.ntp.org, 0.pool.ntp.org, 1.pool.ntp.org, time.nist.gov, ... see <i>Step 4.2 Add shared attributes of new device</i>
Sync Time	Sync time per syncTimeFreq seconds. If you change <i>Timezone</i> or <i>NTP Server</i> , you have to do it. See <i>Step 4.2 Add shared attributes of new device</i>
Device attributes	Device name, device profile (type), device label, model, MAC, device Wi-Fi Module F/W version, device Main MCU F/W version
Reboot	Reboot device
Clear Wi-Fi Config	Clear device’s Wi-Fi configuration

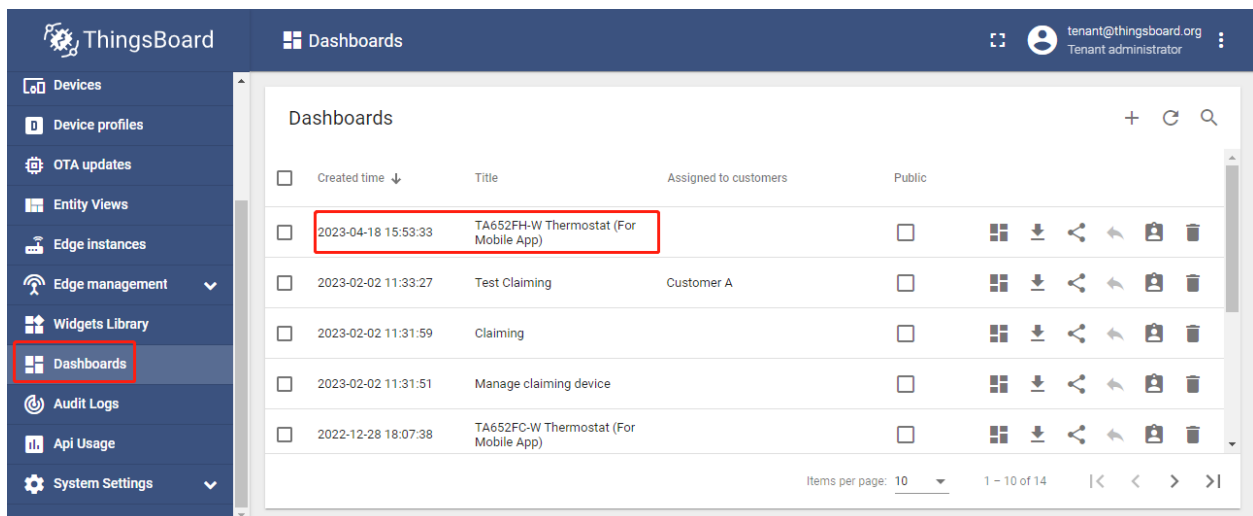
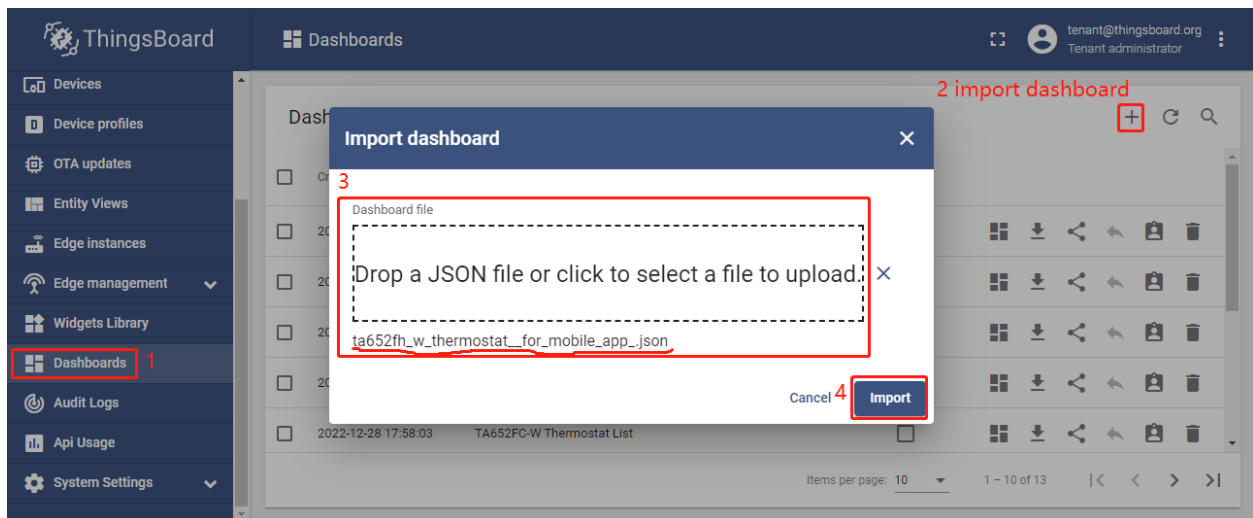
### Import Detail Dashboard

**Tip:** A *Dashboard* file can only be imported once. If you have already imported it, you do not need and cannot repeat the import.

If you have already imported it, you can skip this step.

In order to use this dashboard, you must to create TA652FH-W Thermostat Device Profile. If it doesn't exist, you can import it. See *Import Device Profile of TA652FH-W Thermostat*.


- Download `ta652fh_w_thermostat__for_mobile_app_.json`.
- **Dashboards** → + → **Popup dialog: Import dashboard** → Drag and drop *detail dashboard File* → **Import**.

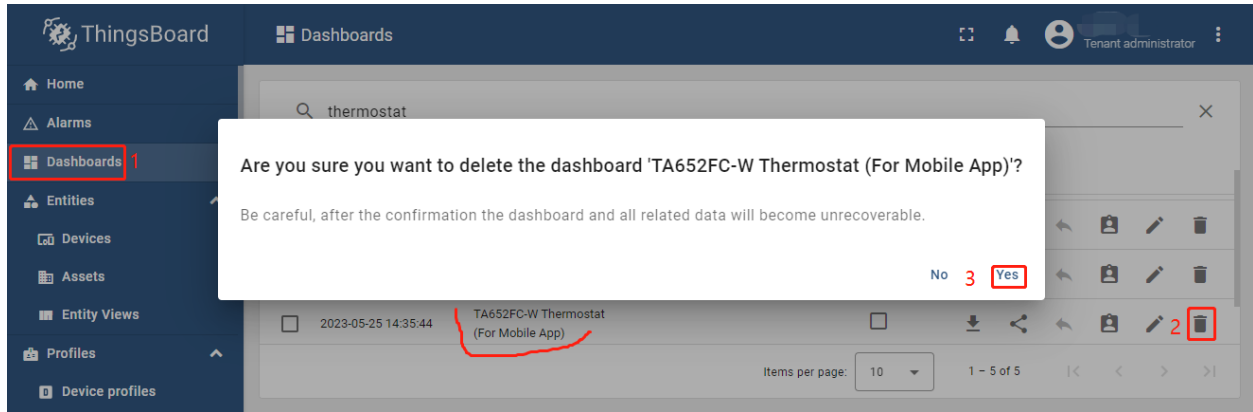


- Optional, This dashboard can be set as TA652FH-W Thermostat Device Profile's mobile dashboard. See *Modify TA652FH-W Thermostat device profile's mobile dashboard*.



## Update Detail Dashboard

- First, *clear TA652FH-W Thermostat device profile's mobile dashboard.*
- Next, delete this dashboard: **Dashboards** → Click  in the row of TA652FH-W Thermostat (For Mobile App) → **Popup dialog: Are you sure you want to delete ...? → Yes.**



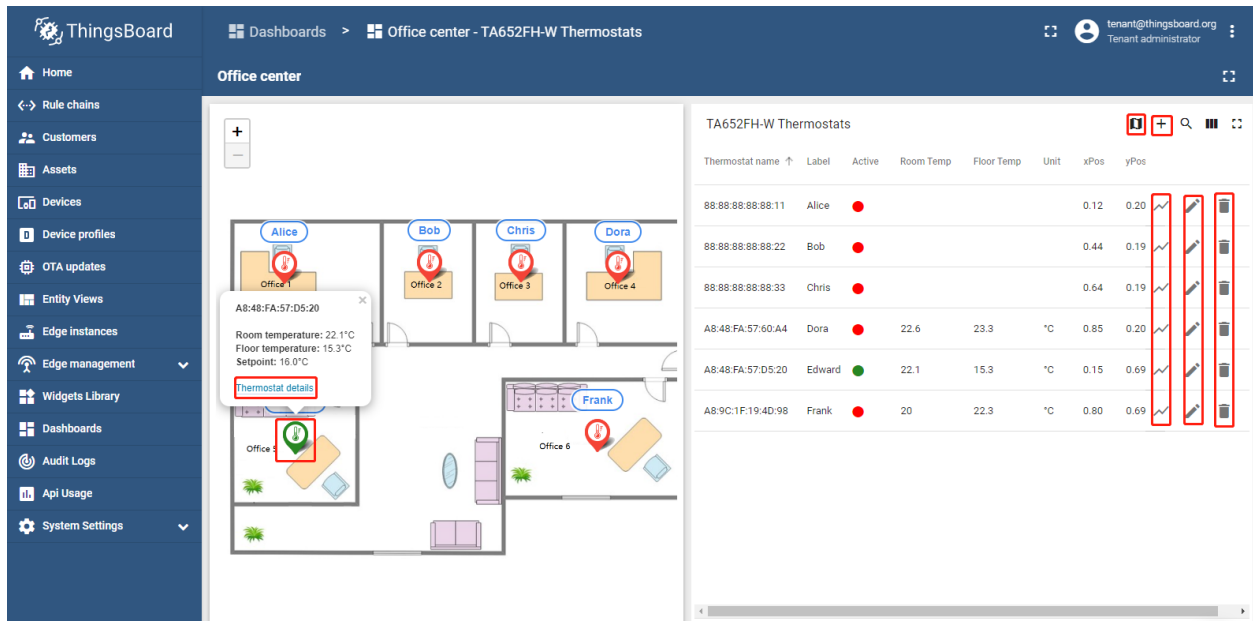
- Then *import TA652FH-W Detail Dashboard.*

## 5.5.4 Office center - TA652FH-W Thermostats

### Dashboard states

### Office state

Office state is root state.



- Dashboard bar:**

## Office center






- : Click here to skip to **root state**. Since **default state** is *root state*, click here and there is no response.

### • Thermostats List:


#### – Fields:

- \* Thermostat name, Label, active.
- \* Room temperature, Floor Temperature, Setpoint, Unit, xPos, yPos.

#### – Actions:

- \*  : Skip to *Map state*.
- \*  : Open a dialog, to add a new thermostat.
- \*  : Skip to *Chart state*.
- \*  : Open a dialog, to edit a thermostat.
- \*  : Open a dialog, to edit a thermostat.


### • Map widget:

-  : Open a box, to show some text.
- [Thermostat details](#) : Skip to *Chart state*.

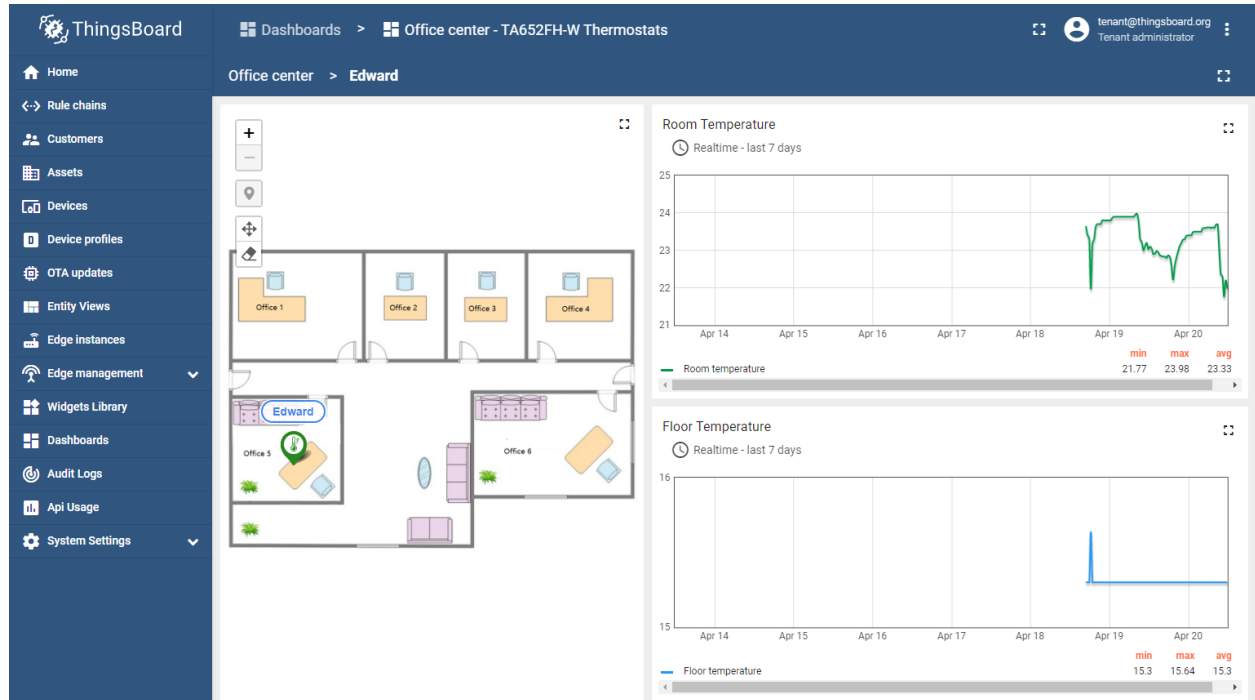
## Map state



- **Map widget:**

-  : Click to enter drag-drop mode, you can modify the position of the thermostat. Click again to save the modification and exit drag-drop mode.

## Chart state



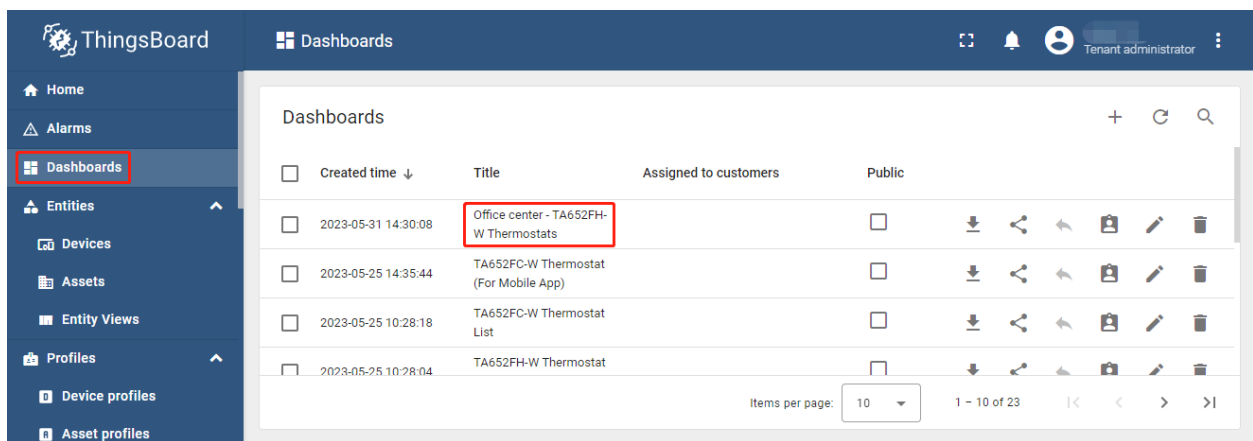
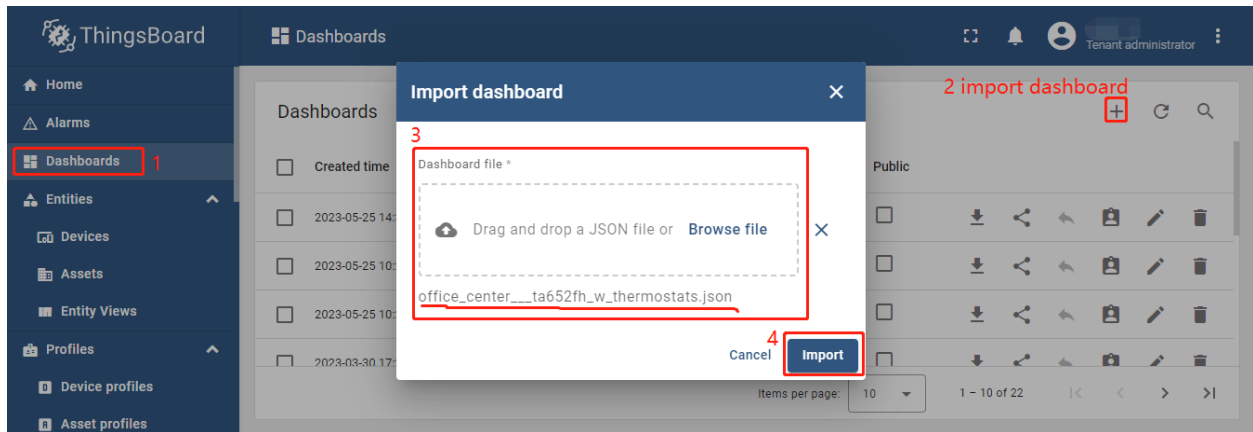
## Import Office Center Dashboard


**Tip:** A *Dashboard file* can only be imported once. If you have already imported it, you do not need and cannot repeat the import.

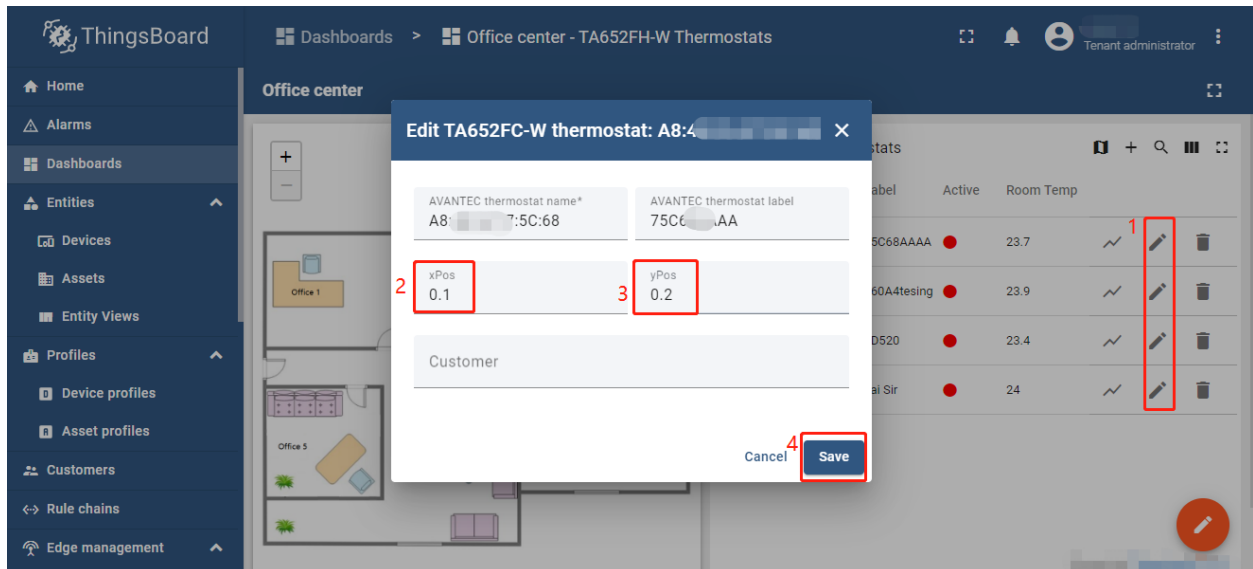
If you have already imported it, you can skip this step.

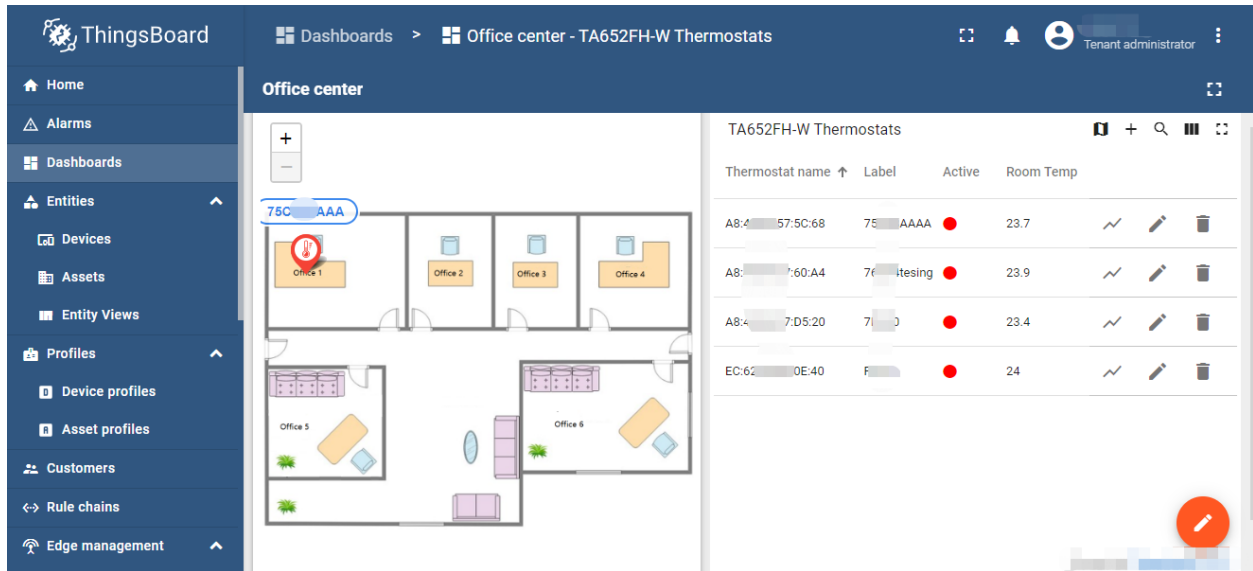
In order to use this dashboard, you must to create **TA652FH-W Thermostat Device Profile**. If it doesn't exist, you can import it. See [Import Device Profile of TA652FH-W Thermostat](#).

- Download `office_center___ta652fh_w_thermostats.json`.
- **Dashboards** → + → **Popup dialog: Import dashboard** → Drag and drop *Office center dashboard File* → **Import**.



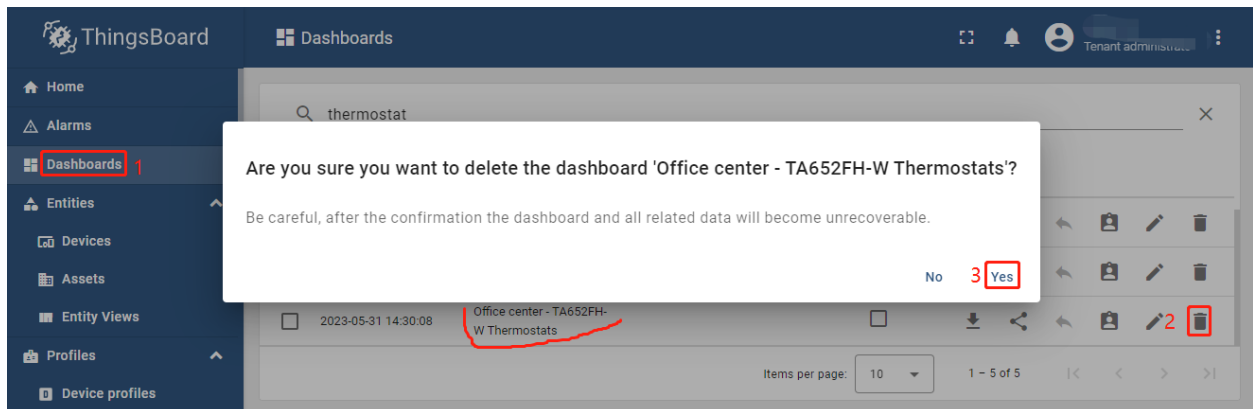
- Edit the position: Click  in a row → Enter xPos/yPos in Popup dialog → **Save**.





## Update Office Center Dashboard

- First, delete this dashboard: **Dashboards** → Click  in the row of Office center - TA652FH-W Thermostats → **Popup dialog: Are you sure you want to delete ...? → Yes.**



- Next, *import TA652FH-W List Dashboard.*

## 5.6 TA652FH-W MQTT API

See *TA652FC-W MQTT API.*



## TA692FC-L-5 LORAWAN THERMOSTAT

These references will help you learn more about TA692FC-L-5 LoRaWAN Thermostat, operate it, and even realize your personalized Dashboard.

- [\*Specification\*](#)
- [\*Add to ThingsBoard | Demo Dashboards\*](#)
- [\*LoRaWAN Device API\*](#)

### 6.1 TA692FC-L – FCU Thermostat Series

Operating Voltage	230 VAC $\pm 10\%$
Measurable range	0 - 40 °C, 0.1°C
LoRaWAN	Class C
EU868 band	868.1 MHz ~ 868.5 MHz
AS923 band (Optional)	923.2 MHz ~ 923.4 MHz

#### 6.1.1 Features

- Wireless thermostats for fan coil units
- 1.5" VA TN with backlit - lite grey text on dark background
- Touch keys x 5
- Flush-mount installation in an 86 x 86 / British single-gang wall-box
- White gloss housing with light grey silk-printed keys
- **Controls:**
  - 3-speed fan
  - One DC 0...10V valve actuators
- **Used in systems with:**
  - Fan coil units
  - Heating and cooling appliances

## 6.1.2 Technical Specification

Transmitting power	21.0dBm
Receiving sensitivity	-140dBm
Effective range outdoors	TBD
Measuring temperature	0 ~ 40°C
Controlling temperature	5 ~ 35°C
Adjustable span	1.0°C ~ 4.0°C
Sensing Element	103AT
Storage Temperature	-5 ~ 50°C
Measuring accuracy/resolution	±0.5°C
On/Off Relay Contact Rating	230VAC 2(1)A max
AO Contact Rating	10VDC 1mA max
Terminals	2 mm <sup>2</sup> cable
Operating Temperature	0 ~ 50°C
Operating Voltage	230VAC ±10%
Operating Humidity	5 ~ 95%R.H. non-condensing

## 6.1.3 Order Code

Symbols	Fan Control	Heating	Cooling	LoRa	Frequency
TA692FC-L-1	3-Speed	On/Off heater	On/Off valve	LoRaWAN	endpoint 868.1M~868.5MHz, or 920M~925MHz
TA692FC-L-2	0~10V	On/Off heater	On/Off valve	LoRaWAN	endpoint 868.1M~868.5MHz, or 920M~925MHz
TA692FC-L-3	0~10V	On/Off heater	0~10V modulating	LoRaWAN	endpoint 868.1M~868.5MHz, or 920M~925MHz
TA692FC-L-4	0~10V	0~10V modulating	0~10V modulating	LoRaWAN	endpoint 868.1M~868.5MHz, or 920M~925MHz
TA692FC-L-5	3-Speed	—	0~10V modulating	LoRaWAN	endpoint 868.1M~868.5MHz, or 920M~925MHz

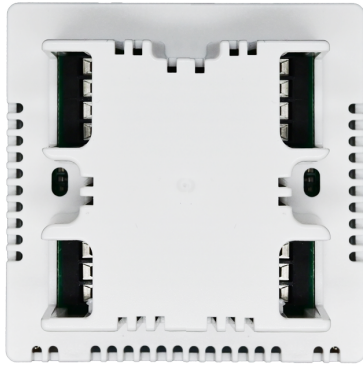
## 6.1.4 Dimensions / Outline

- Protruding part - 86.0mm(W) x 86.0mm(H) x 16.5mm(D)
- Concealed part - 64.0mm(W) x 66.5mm(H) x 26.6mm(D)

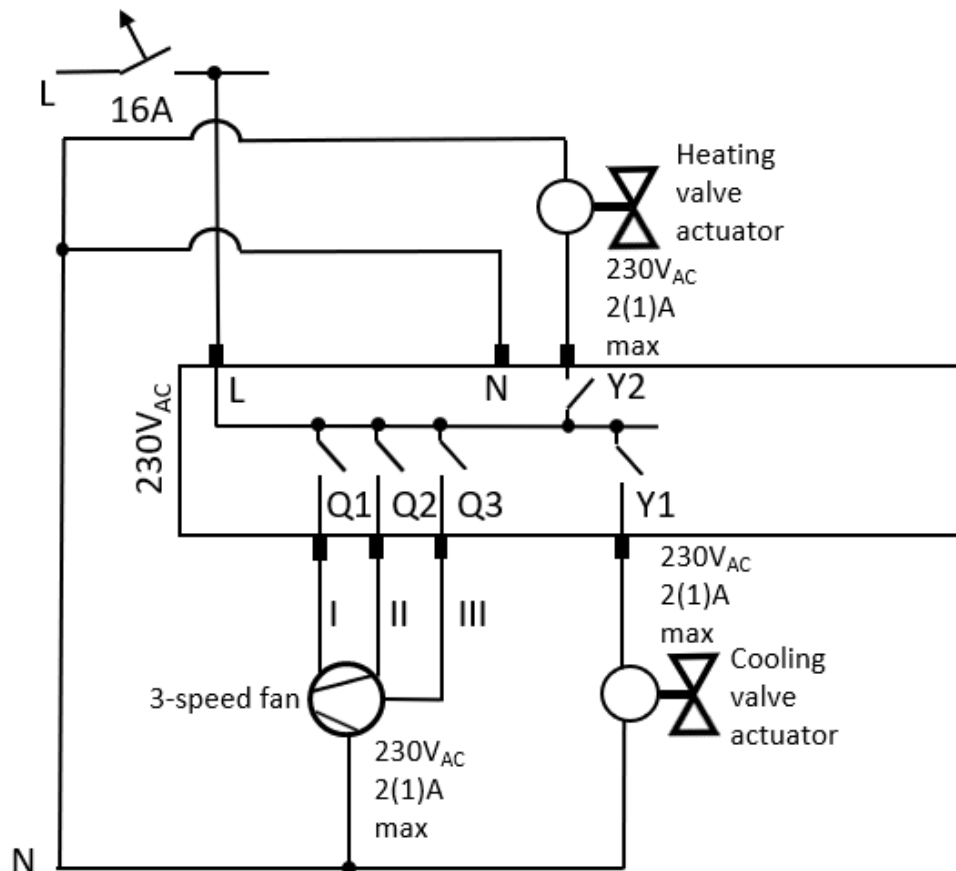




### 6.1.5 Product pictures

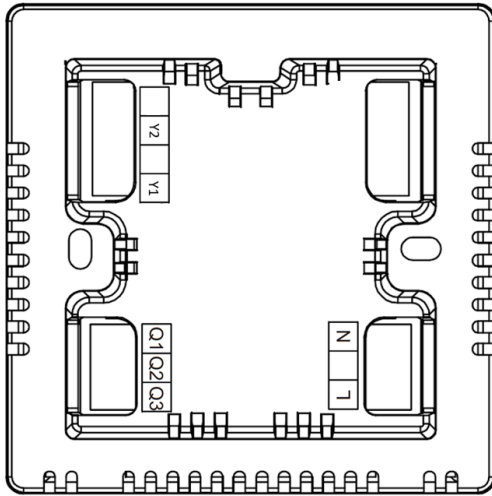


## 6.1.6 Wiring Example for TA692FC-L-1

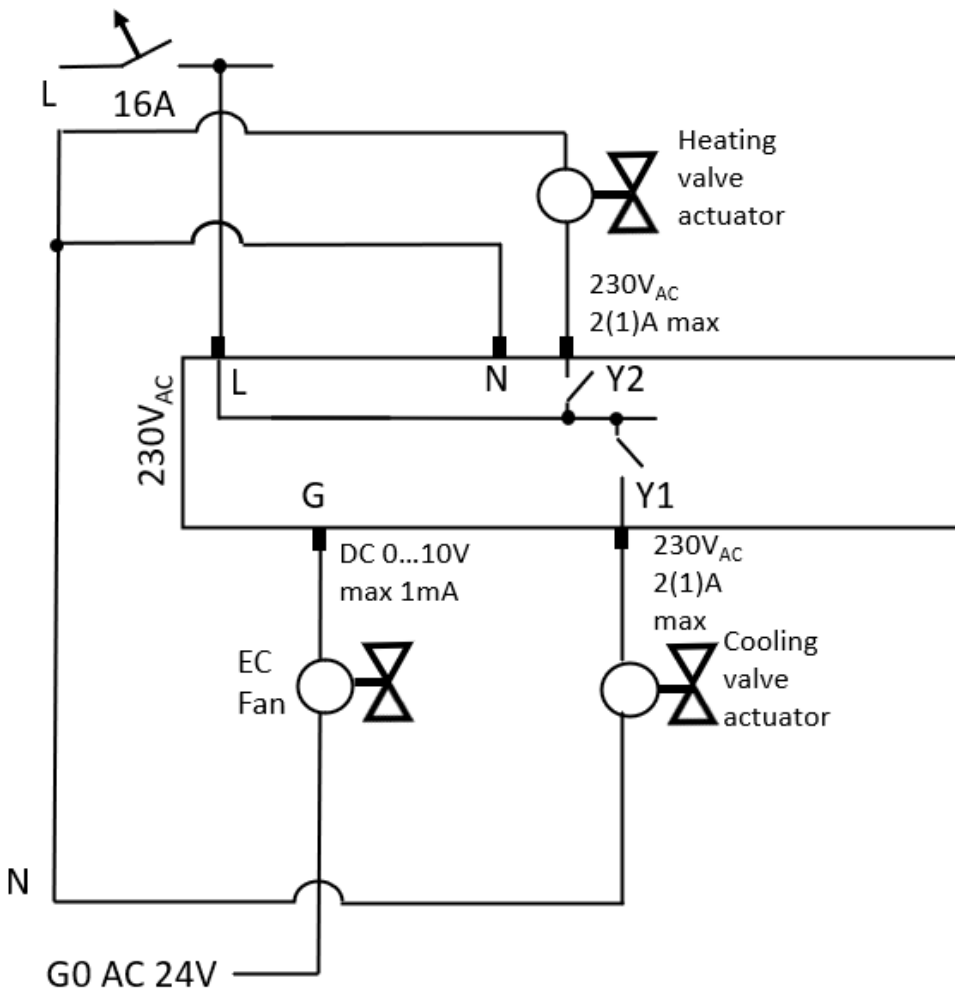


Symbols	Terminals
L	Live
N	Neutral
Q1	Control output Fan speed 1, 230VAC
Q2	Control output Fan speed 2, 230VAC
Q3	Control output Fan speed 3, 230VAC
Y1	Control output for Cool Valve ON/OFF, 230VAC
Y2	Control output for Heater ON/OFF, 230VAC

### 6.1.7 Terminal Labels on TA692FC-L-1

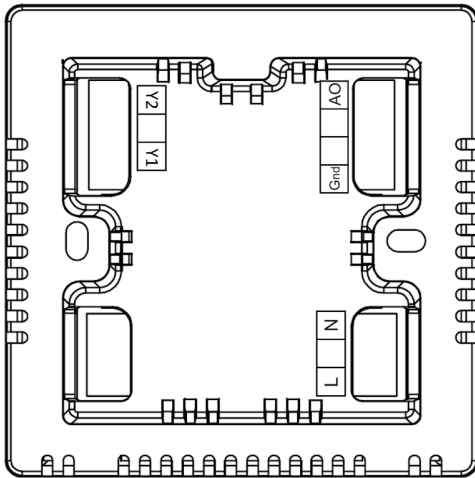


### 6.1.8 Wiring Example for TA692FC-L-2

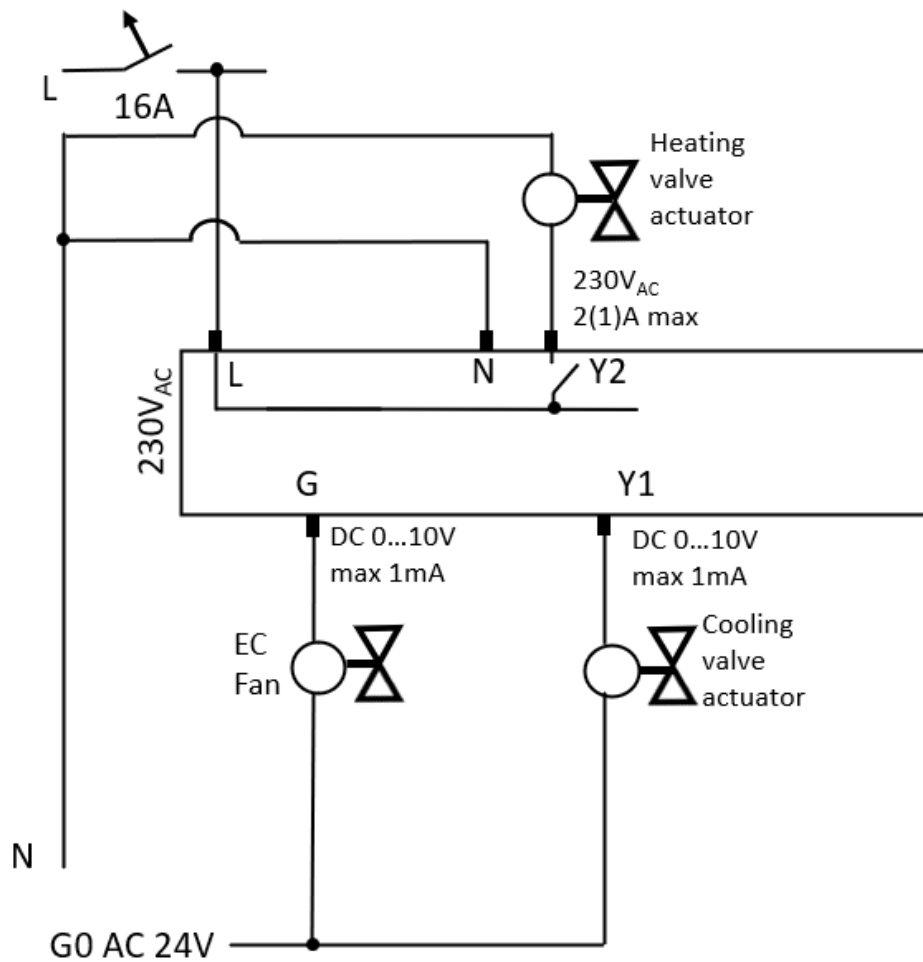


Symbols	Terminals
L	Live
N	Neutral
G	Control output to EC Fan 0...10VDC
Y1	Control output Cool valve ON/OFF. 230VAC
Y2	Control output Heater ON/OFF. 230VAC

### 6.1.9 Terminal Labels on TA692FC-L-2

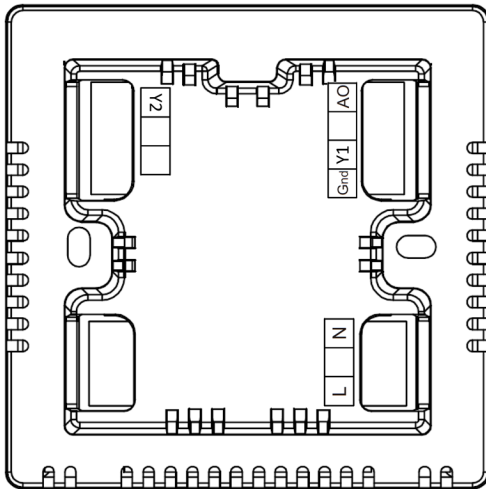


## 6.1.10 Wiring Example for TA692FC-L-3

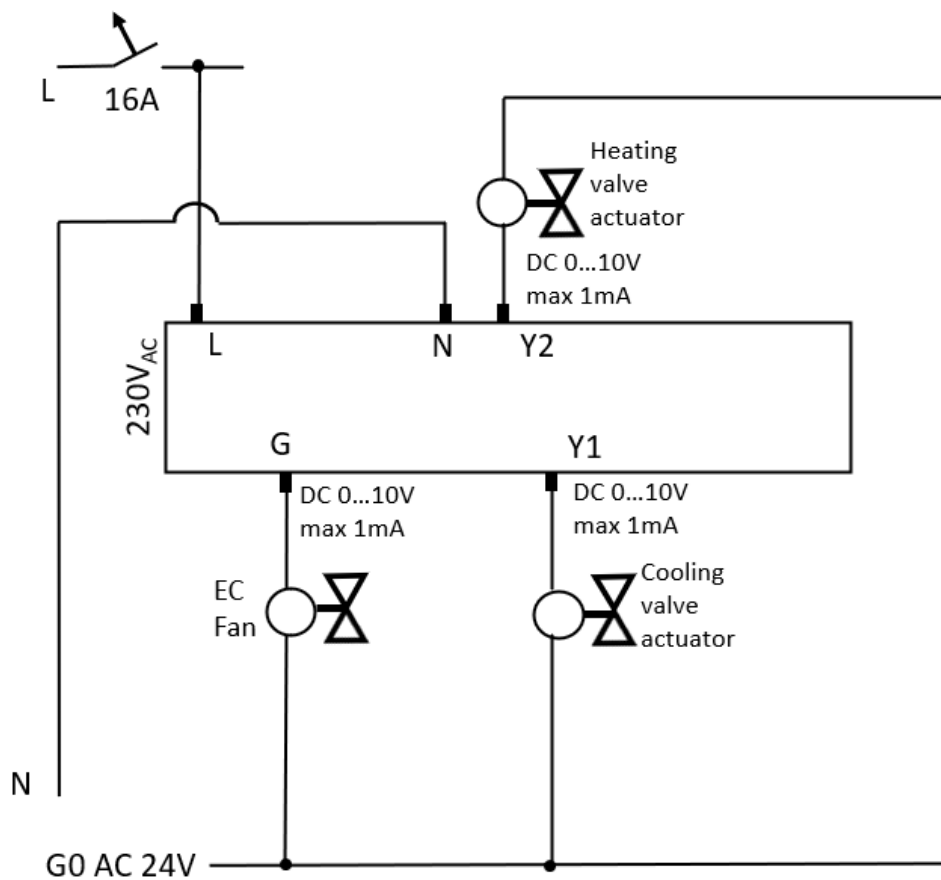


Symbols	Terminals
L	Live
N	Neutral
G	Control output to EC Fan 0...10VDC
Y1	Modulating control to Cool valve 0...10VDC
Y2	Control output Heater ON/OFF. 230VAC

### 6.1.11 Terminal Labels on TA692FC-L-3

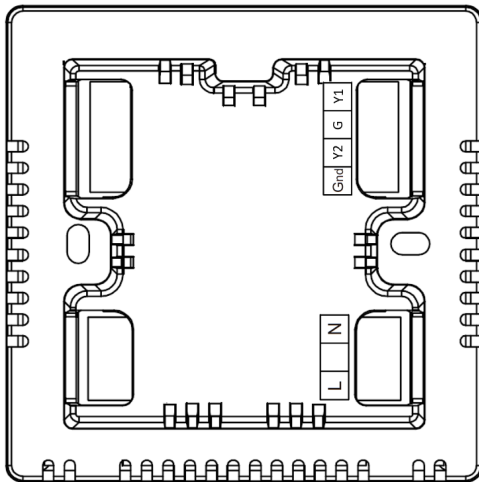


### 6.1.12 Wiring Example for TA692FC-L-4



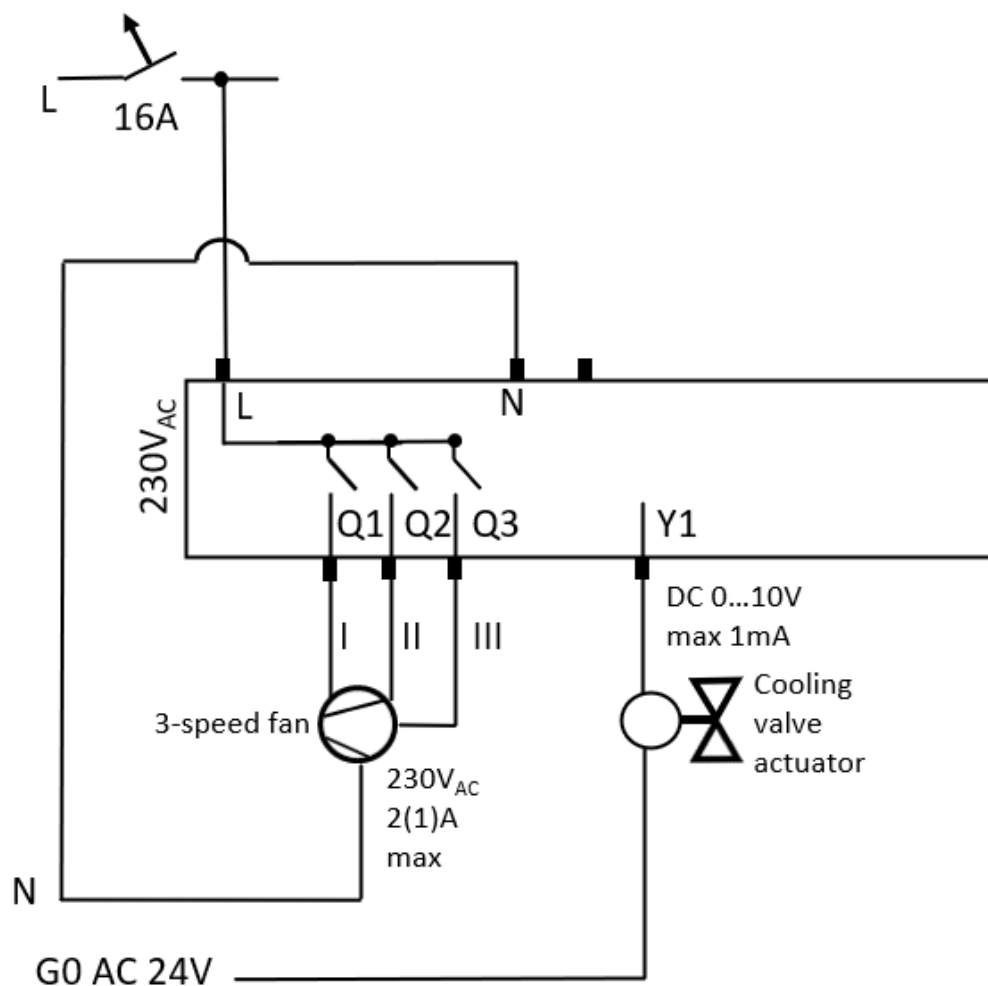
Symbols	Terminals
L	Live
N	Neutral
G	Control output to EC Fan 0...10VDC
Y1	Modulating control to Cooling valve 0...10VDC
Y2	Modulating control to Heating valve 0...10VDC

### 6.1.13 Terminal Labels on TA692FC-L-4



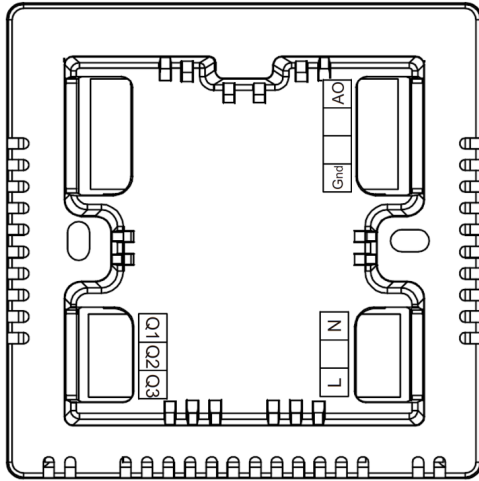


## 6.1.14 Wiring Example for TA692FC-L-5



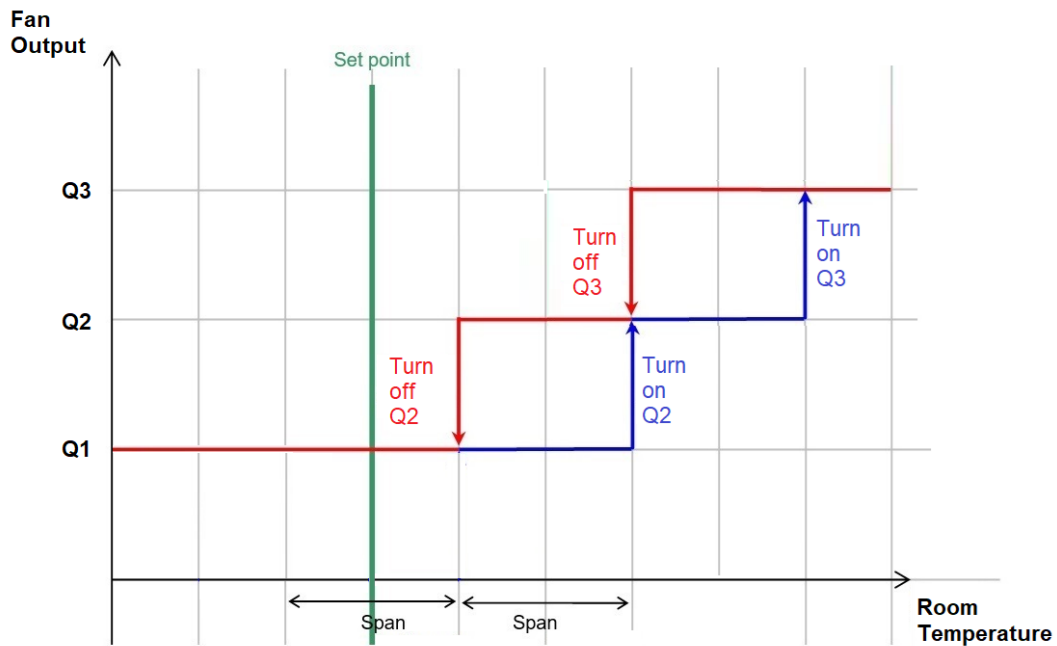
Symbols	Terminals
L	Live
N	Neutral
Q1	Control output Fan speed 1, 230VAC
Q2	Control output Fan speed 2, 230VAC
Q3	Control output Fan speed 3, 230VAC
Y1	Control output to Cooling valve 0...10VDC

### 6.1.15 Terminal Labels on TA692FC-L-5




### 6.1.16 Output diagrams

- **Fan controls -  $Q_1$   $Q_2$   $Q_3$  - in Auto Fan Mode.** Applicable to TA692FC-L-1, TA692FC-L-5 Except when Power Off, TA692FC-L is always running at low-fan ( $Q_1$  On).

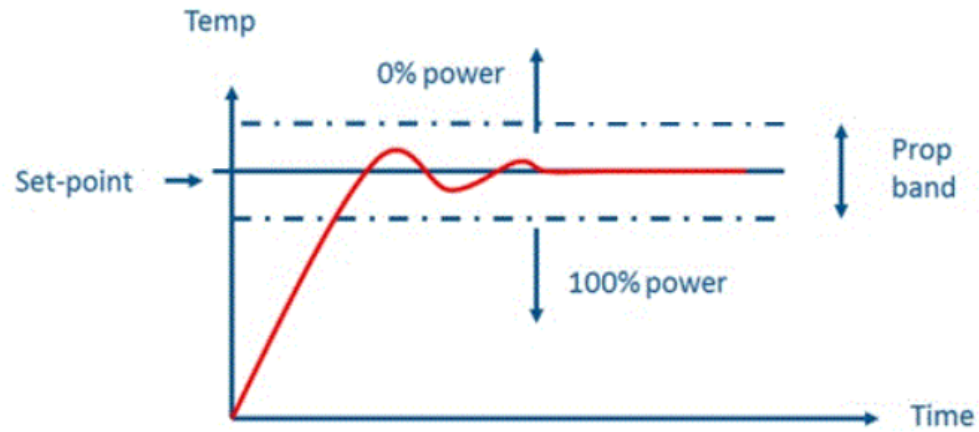


- **Cooling Valve ( $Y_1$ )**  
 PI control of Cooling Valve ( $Y_1$ ) in Cool Mode.  
 Applicable to TA692FC-L-3, TA692FC-L-4, TA692FC-L-5.  
 TA692FC-L employs proportional-integrative modulating control (PI).  
 Diagram shows changing in temperature difference versus  $Y_1$  voltage level over time.



Ion  blinks when output power is under 70%; persistently on at 100%; disappears at 0%.


Refer to subsequent sections for K-Factor, P-band and I-time settings.

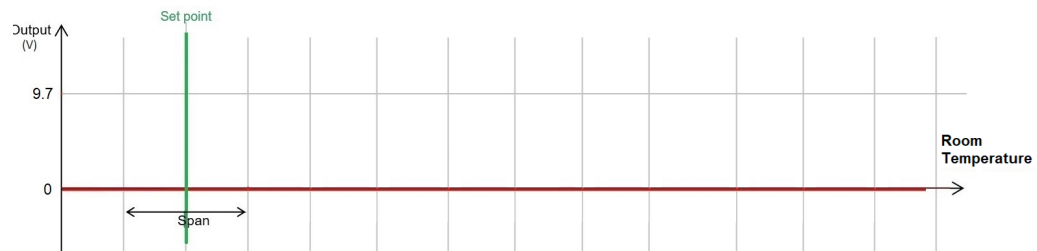


- **Cooling Valve (Y1) in Fan-Only Mode**

Applicable to TA692FC-L-3, TA692FC-L-4, TA692FC-L-5.




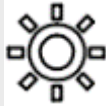




If Fan-Only Mode is selected, Y1 simply shuts off.  disappears.








### 6.1.17 LCD Display Content



## Icons

Label	Description
6	Room temperature
7	Temperature Setpoint
9	<p>System Mode icon</p>  <p>Cool mode</p>  <p>Heat mode</p> <p>no icon - <b>Fan-Only mode</b></p>
12	Y1 output status indicator
13	Y2 output status indicator
14	<p>Fan status indicator</p>  <p>Auto Fan Mode</p> <p>no icon - Manual Fan Mode</p>
	High Fan Speed indicator
	Med Fan speed indicator
	Low Fan speed indicator

## Buttons

Keys	Function
	Menu Key Short press: change mode Press-n-hold: Internal setting
	Fan Speed Short press: cycle-through L->M->H->Auto->L
	Power On/Off Key
	Traverse Up in Setting Menu
	Traverse Down in Setting Menu

## 6.1.18 Internal Parameter Menu in TA692FC-L-5

Items	Selection	Default
System Mode (P00)	Cool / Fan-only (CL/FAN)	Cool (CL)
Calibration (P04)	-4°C ~ 4°C	0°C
Span for Cool (P09)	1.0°C ~ 4.0°C	1.0°C
K-Factor(1/K) (P10)	1 ~ 9	3
P-band Cool (P12)	1.0°C ~ 4.0°C	4.0°C
I-Time Cool (P14)	5 ~ 180 sec	30 sec

## 6.1.19 Advanced Parameter Menu in TA692FC-L-5

Items	Selection	Default
Restore Default on the next power-cycle (P20)	Disabled/Enabled (DIS/EN)	Disabled(DIS)
LoRa status (P36)	Active/disconnect (on/dis)	Active (on)
Dev EUI (P37 ~ P44)	HEX. Read-only	—

## 6.2 Add TA692FC-L-5 to ThingsBoard

### Tip:

- This section applies to the situation where you add a TA692FC-L-5 to the ThingsBoard PE. It implement two-way communication between a TA692FC-L-5 and a ThingsBoard PE.
- Only [ThingsBoard PE](#) supports **Platform Integrations** feature.

### Tip:

- If you only need one-way communication from TA692FC-L-5 to ThingsBoard, you can use **chirpstack v3 + ThingsBoard CE** or **chirpstack v4 + ThingsBoard CE**.
- Refer to [ThingsBoard getting started for ChirpStack v3](#) and [ThingsBoard Integration for ChirpStack v3](#).
- Refer to [ThingsBoard getting started for ChirpStack v4](#) and [ThingsBoard Integration for ChirpStack v4](#).

### 6.2.1 Introduction

**Note:** The frequency of LoRaWAN device and gateway must match!

**Warning:** ChirpStack v4, the latest version, doesn't handle downlink data from ThingsBoard PE v3.5.x.

The ChirpStack open-source LoRaWAN Network Server stack provides open-source components for LoRaWAN networks. After integrating ChirpStack with ThingsBoard, you can connect, communicate, process and visualize data from TA692FC-L-5 thermostat in the ThingsBoard IoT platform.

Item	Description
LoRaWAN Device	TA692FC-L-5, Frequency 868 MHz*
LoRaWAN Gateway	MTCAP-868-041A, Frequency 868 MHz*
LoRaWAN Network Server	ChirpStack v3**
LoRaWAN Application Server	ThingsBoard PE v3.5.x**

### 6.2.2 Prerequisites

**Tip:** You need a ChirpStack instance that can be accessed by your ThingsBoard PE instance.

- If your ThingsBoard PE instance is installed in a LAN, you may also install a ChirpStack instance in the same LAN.
- If your ThingsBoard PE instance is installed in the cloud, you may also install a ChirpStack instance on the corresponding cloud host.
- Obtain the following TA692FC-L-5 LoRaWAN Parameters from your equipment vendor.

Item	Paramter
Model	TA692FC-L-5, Frequency 868 MHz
LoRaWAN	Class C
EU868 band	868.1MHz ~ 868.5MHz
DevEUI/AppEUI/JoinEUI*	<i>YOUR_DEV_EUI</i> , eg: 00:12:BD:FF:FE:02:AD:04
AppKey/Application Key/Network Key*	<i>YOUR_APP_KEY</i> , eg: 72357538782F413F4428472B4B625065

**Note:** These parameters are different for every thermostat.

- **Setup the MTCAP-868-041A**
  - *First-Time Setup of Gateway*
  - *Optional: Firmware Upgrade*
- **Install a ChirpStack v3 instance on**
  - *Amazon AWS*, or
  - *Microsoft Azure*, or
  - *Google Cloud*, or
  - *Debian/Ubuntu*, or
  - *Docker Compose*
- **Subscribe or install a ThingsBoard PE instance**
  - *Use ThingsBoard Cloud*, or
  - *Install your own ThingsBoard PE instance*

### 6.2.3 Step 1. MTCAP configuration

- *Configuring LoRa Packet Forwarder.*

### 6.2.4 Step 2. ChirpStack configuration

- *Connect gateway to ChirpStack.*
- *Connect device to ChirpStack.*

### 6.2.5 Step 3. Integrating ChirpStack with ThingsBoard PE

Refer to [ChirpStack Integration](#).



### Step 3.1 Uplink Converter

Before creating the integration, you need to create/import an Uplink converter in Data converters. Uplink is necessary in order to convert the incoming data from the device into the required format for displaying them in ThingsBoard. To view the events, enable **Debug**. In the function decoder field, specify a script to parse and transform data.

**NOTE** Although the Debug mode is very useful for development and troubleshooting, leaving it enabled in production mode may tremendously increase the disk space, used by the database, because all the debugging data is stored there. It is highly recommended to turn the Debug mode off when done debugging.

#### sample uplink message

Let's review sample uplink message from ChirpStack:

```
{
  "applicationID": "1",
  "applicationName": "TA692FC-L-5-Application",
  "deviceName": "Sales-Office",
  "devEUI": "ABK9//4CrQQ=",
  "rxInfo": [{
    "gatewayID": "AIAAAAACDgs=",
    "time": null,
    "timeSinceGPSEPOCH": null,
    "rssi": -52,
    "loRaSNR": 8.5,
    "channel": 2,
    "rfChain": 0,
    "board": 0,
    "antenna": 0,
    "location": {
      "latitude": 22.31025463915414,
      "longitude": -245.77515719803597,
      "altitude": 0,
      "source": "UNKNOWN",
      "accuracy": 0
    },
    "fineTimestampType": "NONE",
    "context": "LZVORA==",
    "uplinkID": "p7k/E6nyQpeMTwedtMqHgA==",
    "crcStatus": "CRC_OK"
  ]},
  "txInfo": {
    "frequency": 868500000,
    "modulation": "LORA",
    "loRaModulationInfo": {
      "bandwidth": 125,
      "spreadingFactor": 12,
      "codeRate": "4/5",
      "polarizationInversion": false
    }
  },
  "adr": true,
  "dr": 0,
```

(continues on next page)

(continued from previous page)

```

    "fCnt": 114,
    "fPort": 10,
    "data": "A0AA12QCAwIBKAAeAw==",
    "tags": {},
    "confirmedUplink": false,
    "devAddr": "ABCDuw==",
    "publishedAt": "2023-06-15T08:24:14.436509221Z",
    "deviceProfileID": "e87f230b-51a7-407a-aa6c-468308601139",
    "deviceProfileName": "TA692FC-L-5-868 Thermostat"
  }

```

## Device fields

- As you can see the device EUI arrives in the **devEUI** field. We will use it as a **device name** in ThingsBoard.
- As you can see the device profile name arrives in the **deviceProfileName** field. We will use it as a **device Type (device profile name)** in ThingsBoard.
- As you can see the device name arrives in the **deviceName** field. We will use it as a **device label** in ThingsBoard.

Table 1: ChirpStack fields v.s. ThingsBoard fields in this case :widths:  
auto :header-rows: 1

No.	ChirpStack field	Editable	ThingsBoard field	Editable
1	devEUI	No	deviceName	No
2	deviceProfileName	No	deviceType (deviceProfileName)	Yes <sup>1</sup>
3	deviceName	Yes <sup>2</sup>	deviceLabel	Yes <sup>2</sup>

## Notes

In the converter it will be indicated like this:

```

var deviceLabel = data.deviceName; // "Sales-Office"
var deviceName = base64ToHexWithoutUppercase(data.devEUI); // "0012bdfffe02ad04", unique
var deviceType = data.deviceProfileName; // "TA692FC-L-5-868 Thermostat"

var result = {
  deviceName: deviceName,
  deviceType: deviceType,
  deviceLabel: deviceLabel
}

```

<sup>1</sup> In this case, if **deviceType (deviceProfileName)** of the device is modified, the device may not be able to receive messages from ChirpStack or send messages to ChirpStack.

<sup>2</sup> Both are the same only when ThingsBoard automatically creates the device. They are not automatically kept in sync afterwards.

## Device data

Device data is encoded in the “data” field. The Base64 encoded data here is:

```
"data": "AOAA12QCAwIBKAAeAw=="
```

Let's convert them into roomTemperature, setTemperature, fanMode and fanState, etc.

In the decoded form we have the following string: **00 E0 00 D7 64 02 03 02 01 28 00 1E 03**

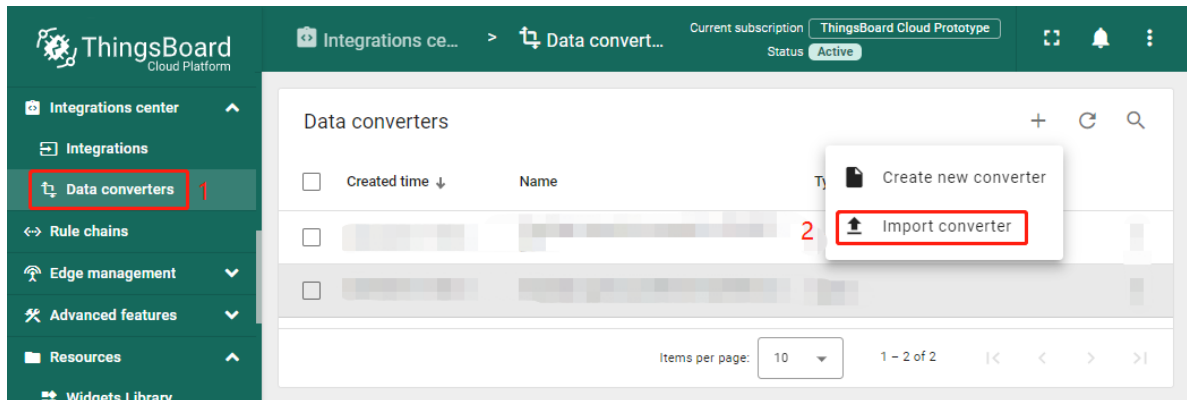
- **00 E0** is the value for roomTemperature, 22.4.
- **00 D7** is the value for setTemperature, 21.5.
- **64** is the value for coolProportionalOutput, 100%.
- **02** is the value for fanMode, MED.
- **03** is the value for fanState, HIGH.
- **02** is the value for threshold, 0.2.
- **01** is the value for systemMode, COOL.
- **28** is the value for coolPBand, 4.0.
- **00 1E** is the value for coolITime, 30.
- **03** is the value for kFactor, 3.

In the converter it will be indicated like this:

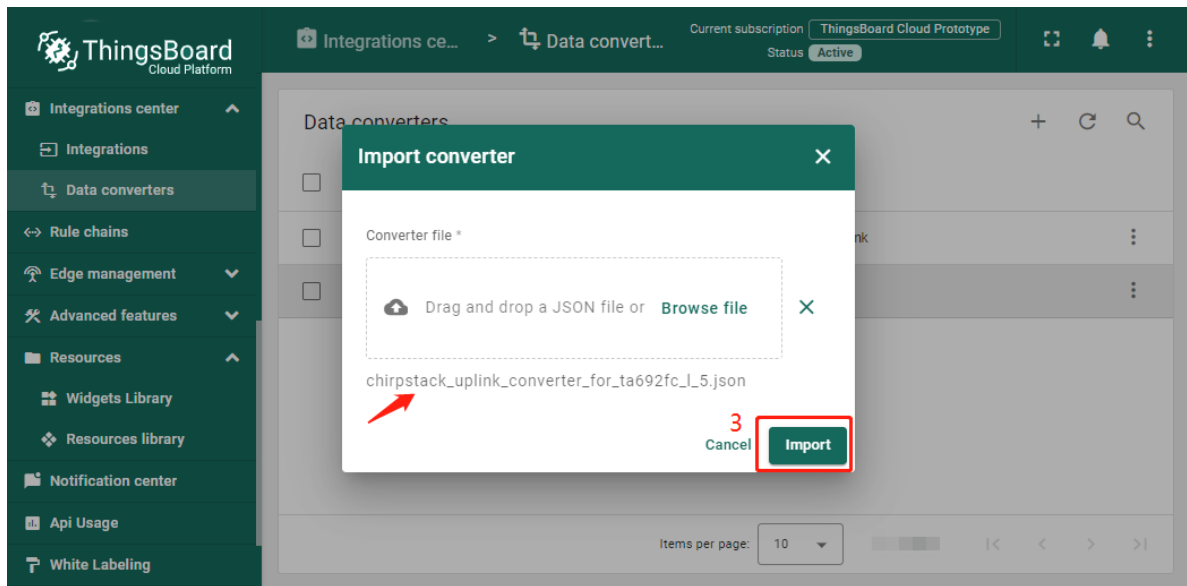
```
var result = {
  attributes: {
    setTemperature: parseInt(incomingHexData.substring(4, 8), 16)/10,
    coolProportionalOutput: parseInt(incomingHexData.substring(8, 10), 16)/100,
    fanMode: fanModeStateMeta[parseInt(incomingHexData.substring(10, 12), 16)],
    fanState: fanModeStateMeta[parseInt(incomingHexData.substring(12, 14), 16)],
    threshold: parseInt(incomingHexData.substring(14, 16), 16)/10,
    systemMode: systemModeMeta[parseInt(incomingHexData.substring(16, 18), 16)],
    coolPBand: parseInt(incomingHexData.substring(18, 20), 16)/10,
    coolITime: parseInt(incomingHexData.substring(20, 24), 16),
    kFactor: parseInt(incomingHexData.substring(24, 26), 16)
  },
  telemetry: {
    roomTemperature: parseInt(incomingHexData.substring(0, 4), 16)/10
  }
}
```

## Importing uplink Converter

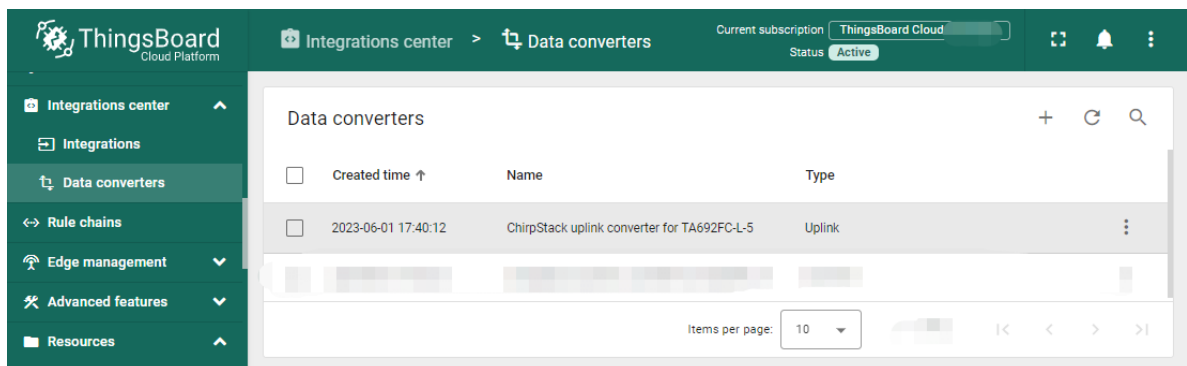
- Download ChirpStack uplink converter for TA692FC-L-5.
- **Data converters** -> + -> **Import converter**.



- Popup dialog: Import converter → Drag and drop the converter file → **Import**.



- Show it in the list of Data Converters.



You can change the decoder/encoder function while creating the converter or after creating it. If the converter has already been created, then click on the “pencil” icon to edit it. Copy the configuration example for the converter (or your own configuration) and insert it into the decoder/encoder function. Save changes by clicking on the “checkmark” icon.

### Step 3.2 Downlink Converter

You can customize the downlink according to your configuration. Let's consider an example where we send an shared attribute update message - **remoteSetSetTemperature**.

```
data: msg.remoteSetSetTemperature
```

Also, indicate the required parameters in the metadata:

```
metadata: {
  "cs_devEUI": "$Device_EUI"
}
```

Example for downlink converter:

```
var remoteSetSetTemperature = msg.remoteSetSetTemperature;
var fPort = 91;
var content = Math.round(remoteSetSetTemperature * 10);
var contentBase64 = Uint16ToBase64(content);

// Result object with encoded downlink payload
var result = {
  // downlink data content type: JSON, TEXT or BINARY (base64 format)
  contentType: "TEXT",

  // downlink data
  data: contentBase64, //JSON.stringify(data),

  // object: {...}, //ChirpStack v4 // decoded object (when application coded has
  // been configured)
  // Optional metadata object presented in key/value format
  metadata: {
    DevEUI: metadata.cs_devEUI, //ChirpStack v3
    fPort: fPort                //ChirpStack v3
  }
};

function Uint16ToBase64(value) {
  let myArr = new Uint8Array(2);
  myArr[0] = value >> 8; // High byte
  myArr[1] = value >> 0; // Low byte

  let myStr = Uint8ArrayToString(myArr);
  return btoa(myStr);
}

function Uint8ArrayToString(fileData){
  var dataString = "";
  for (var i = 0; i < fileData.length; i++) {
    dataString += String.fromCharCode(fileData[i]);
  }

  return dataString;
}
```

(continues on next page)

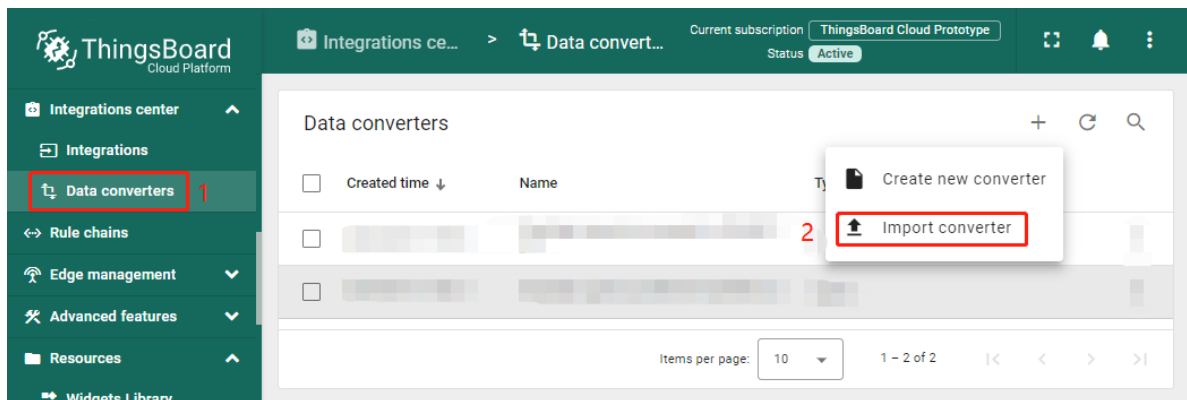
(continued from previous page)

```
return result;
```

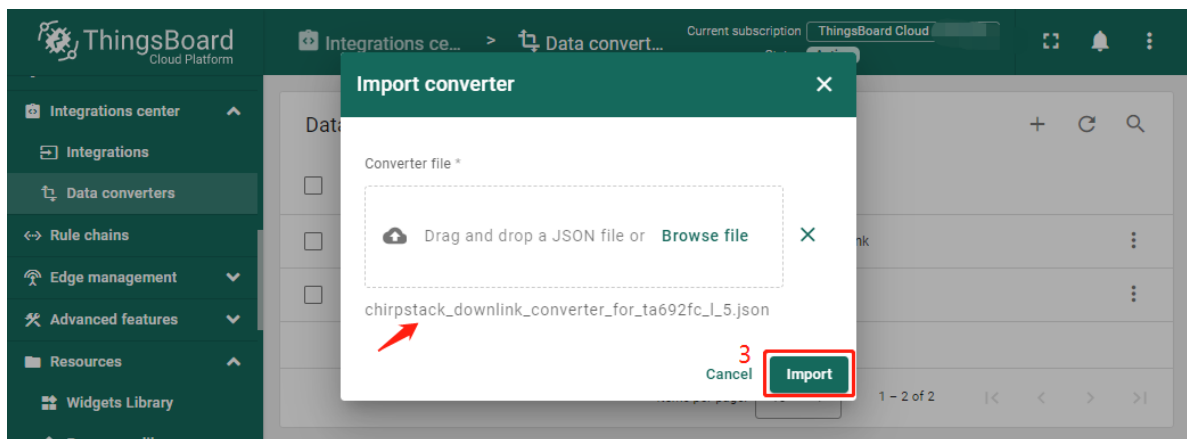
Where `cs_devEUI` is device EUI, it will be taken from the device uplink message.

## Importing downlink Converter

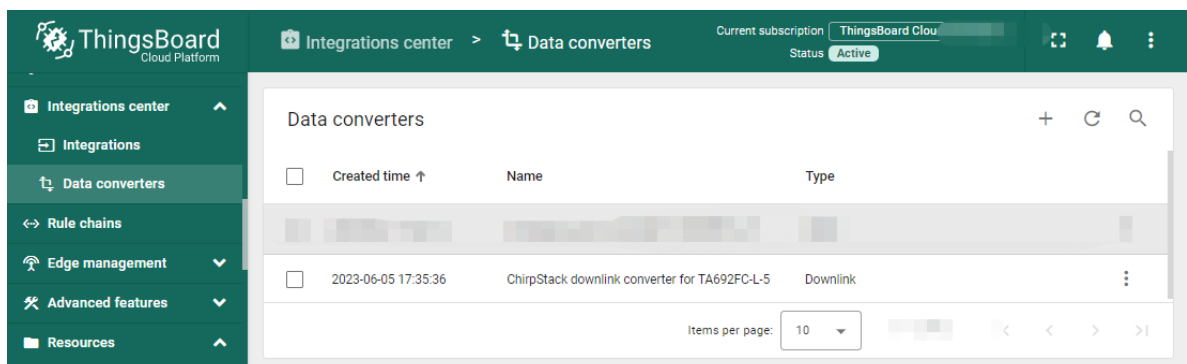
- Download ChirpStack downlink converter for TA692FC-L-5.
- **Data converters** → + → **Import converter**.



- Popup dialog: Import converter → Drag and drop the converter file → **Import**.



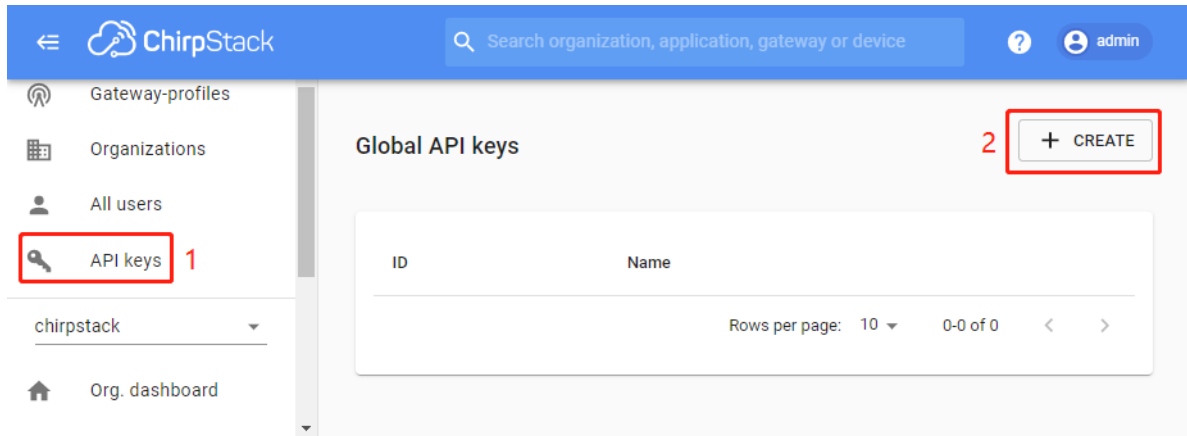
- Show it in the list of Data Converters.



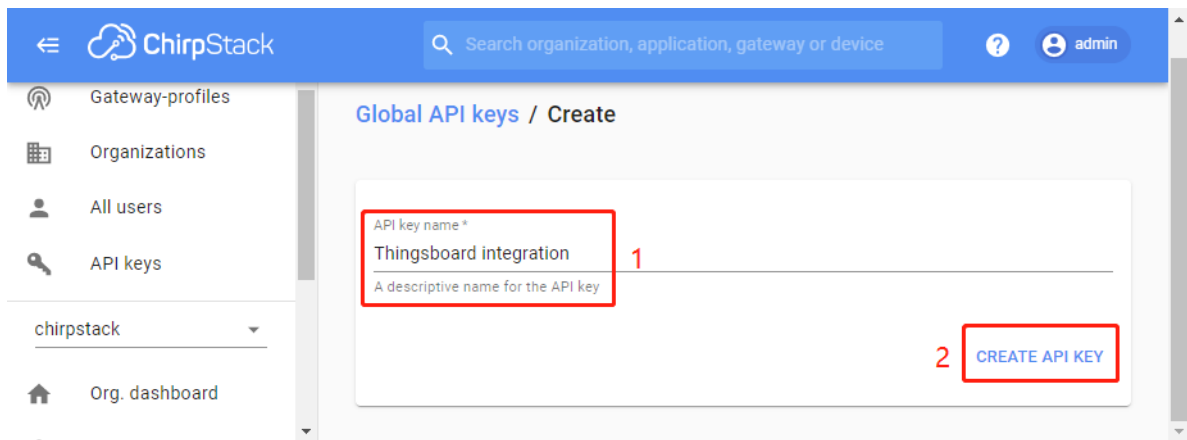
### Step 3.3 Create Integration

#### Step 3.3.1 Get Application API key from ChirpStack

- To get the API key we need to open Application server UI, open **API keys** tab from the left top menu and **Create** an API key.



- Input your API key name → **Create API key**.



- Copy your token.

ChirpStack

Search organization, application, gateway or device

admin

Gateway-profiles

Organizations

All users

API keys

chirpstack

Org. dashboard

Org. users

Org. API keys

Service-profiles

Device-profiles

### Global API keys / Create

API key ID  
a27375da-b694-4f88-a702-8ca8454f2b87

API key name  
Thingsboard integration

Token  
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcGlfa2V5X2lkIjoIYTl3Mzc1ZGZlYjY5NC00Zjg4LWE3MDItOGNhODQ1NGYyYjg3IiwiaXNkIjoIYXMiLCJpc3MiOiJhcylsIm5iZiI6MTY4NzE1NDZlNiwiOiJhbnR5cCI6IkpXVCJ9.y1ZP450BqntwWit\_nmx-MgSLRQgWUFD9ONKnXVaJtk

Use this token when making API request with this API key. This token is provided once.

- Show it in the list of global API keys.

ChirpStack

Search organization, application, gateway or device

admin

Org. users

Org. API keys

Service-profiles

Device-profiles

Gateways

Applications 1

### Applications

+ CREATE

ID	Name	Service-profile	Description
2	TA692FC-L-5-Application	localhost service profile	TA692FC-L-5-868 Thermostat, TA692FC-L-5-915 Thermostat

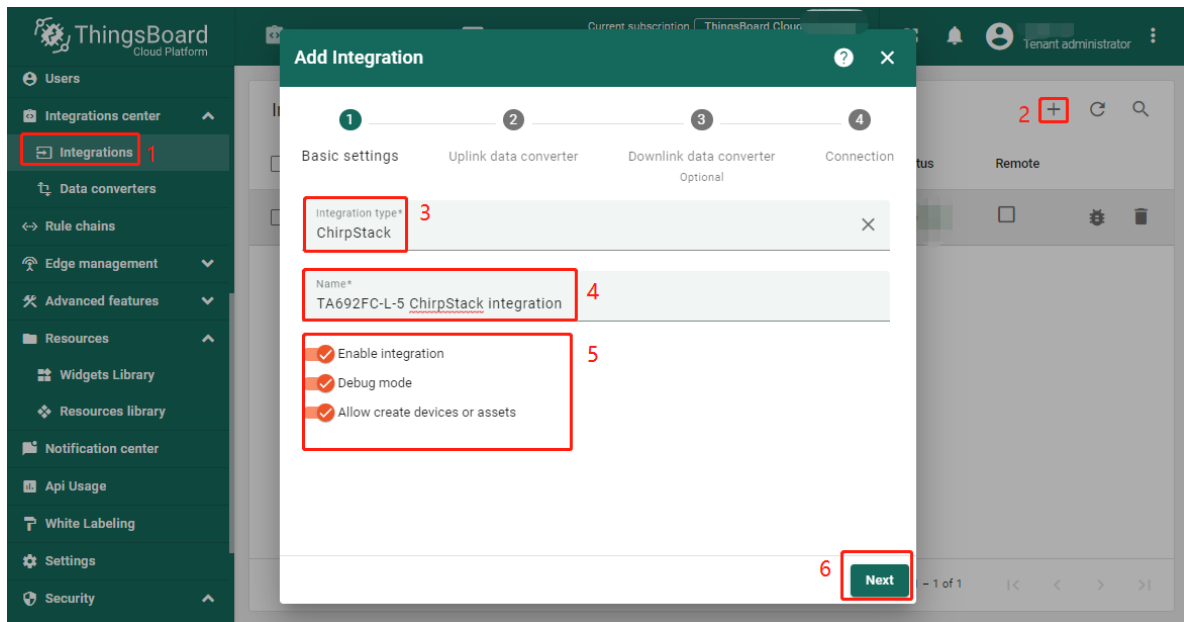
Rows per page: 10 1-1 of 1

### Step 3.3.2 Adding ChirpStack intergration on ThingsBoard

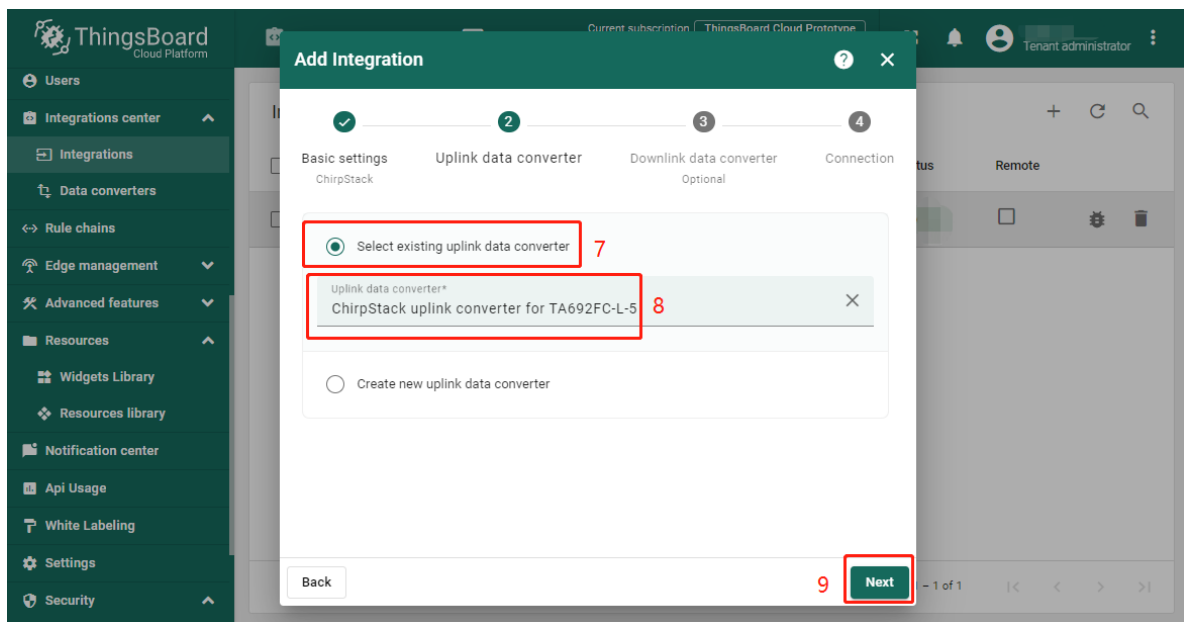
Now that the Uplink converter and Downlink converter have been created, and we have all required data, it is possible to create an integration.

- **Integrations** → select a integration type: **ChirpStack** → input name: *TA692FC-L-5 ChirpStack integration* → enable **integration**, **debug mode** and **allow create devices or assets** → **Next**.

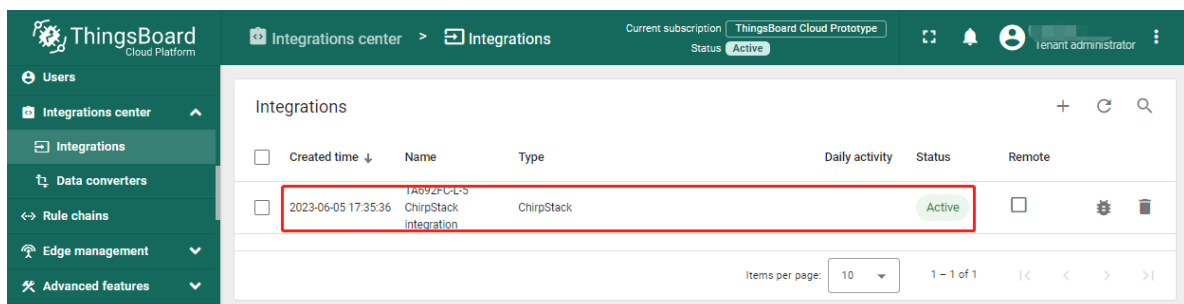
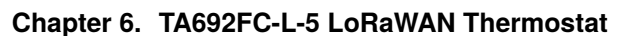




- Select Uplink data convert: *TA692FC-L-5 downlink from ChirpStack*.



- Select Downlink data convert: *TA692FC-L-5 uplink from ChirpStack*.

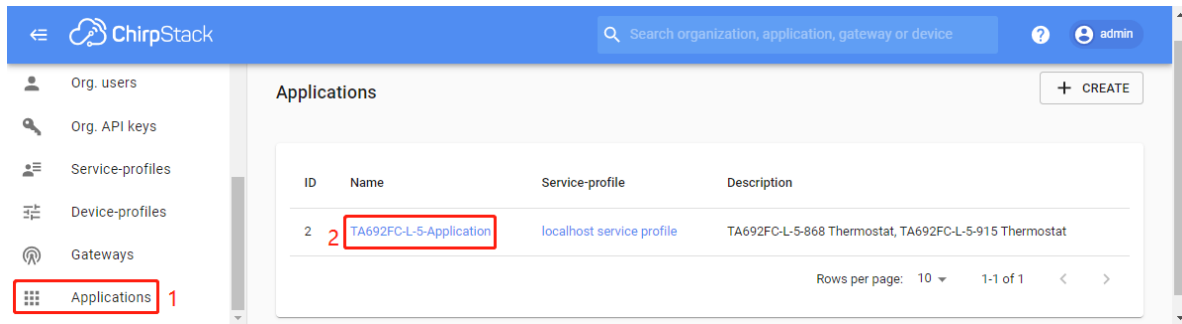


**NOTE:** It is recommended to enable Debug mode for debug purposes to see uplink/downlink events on integration.

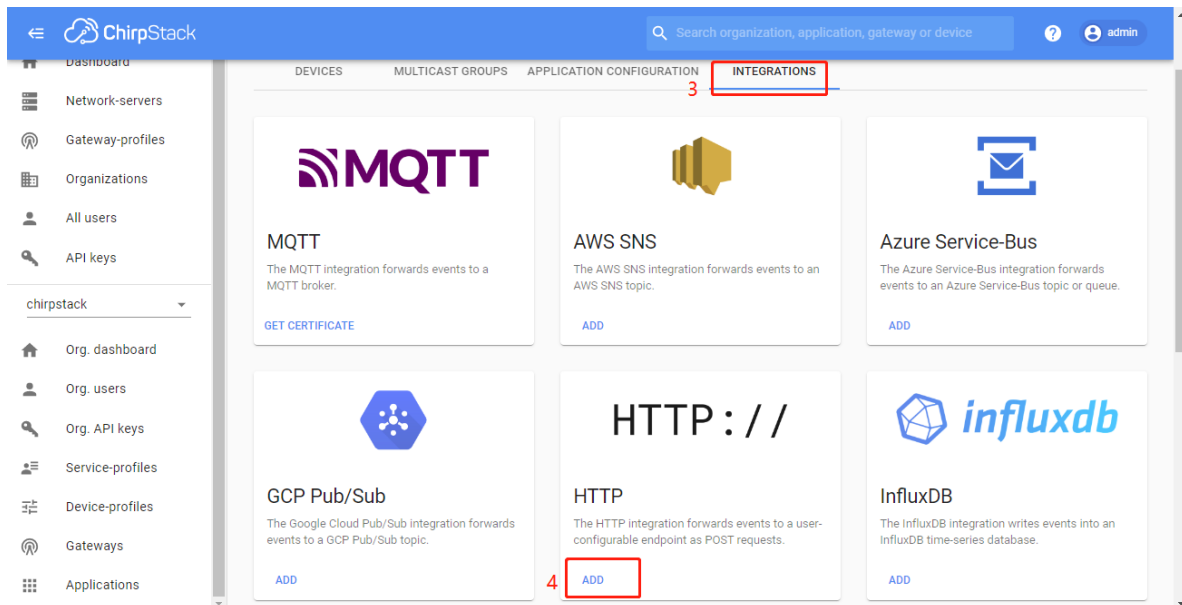
### Step 3.3.3 Configure an Integration for your ChirpStack application

To create integration on ChirpStack Network server stack, we need to do the following steps:

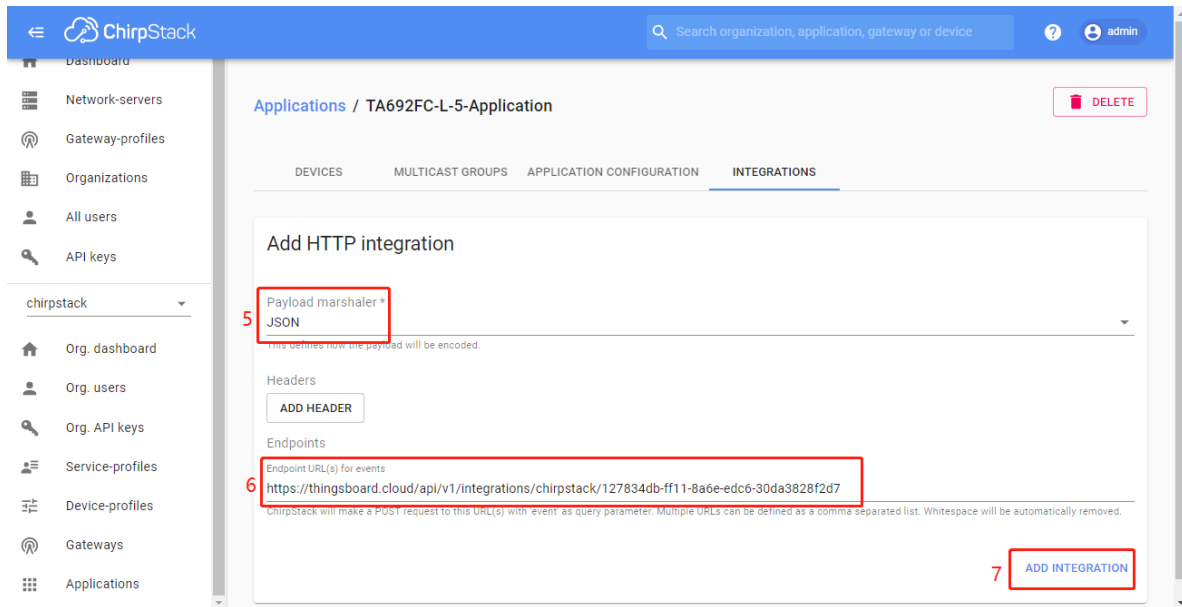
- Login to ChirpStack Network server stack user interface (Default login/password - **admin/admin**).
- We go to the tab **Applications** in the left menu and open our application (our application is named Application).



- Open the **Integrations** tab and create a **HTTP** integration.



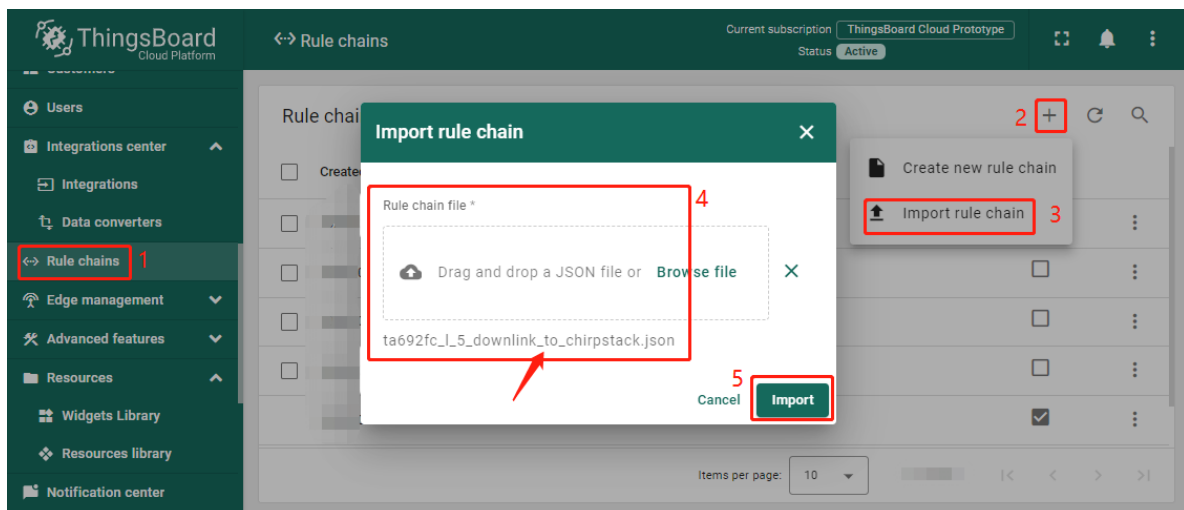
- Let's go to the **Integrations** tab in ThingsBoard. Find your ChirpStack integration and click on it. There you can find the HTTP endpoint URL. Click on the icon to copy the url.
- Fill the fields with endpoint url from ThingsBoard integration:



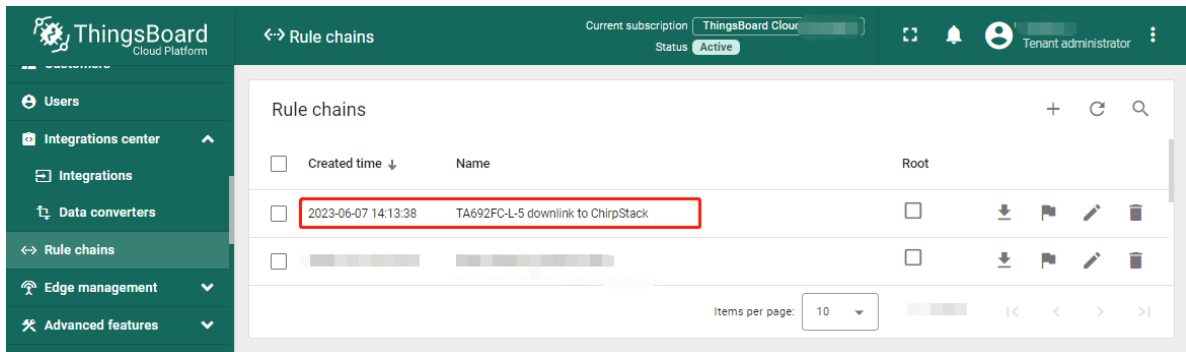
### Step 3.3.4 Importing rule chain for Downlink

In order to send Downlink, we use the rule chain to process shared attribute update. To get **devEUI** from device we have to import rule-chain.

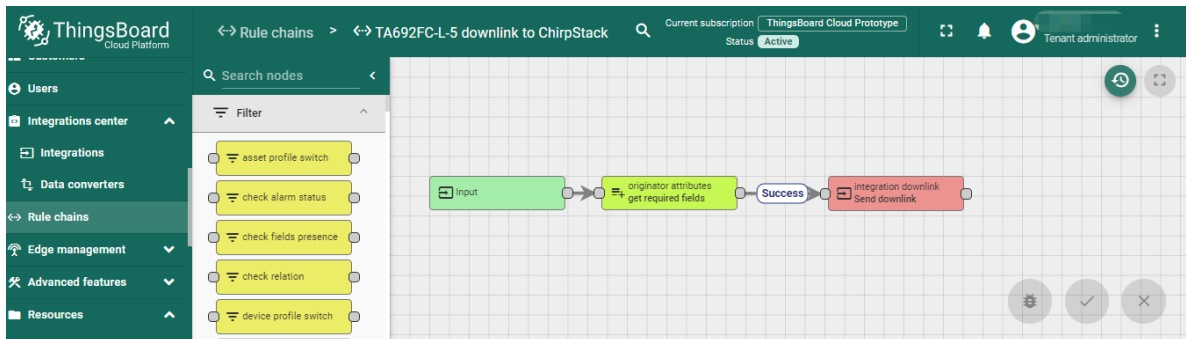
- Download Rule chain: TA692FC-L-5 downlink to ChirpStack.
- **Rule chains** -> **+** -> **Import rule chain** -> Popup dialog: Import rule chain -> Drag and drop the Rule chain file -> **Import**.



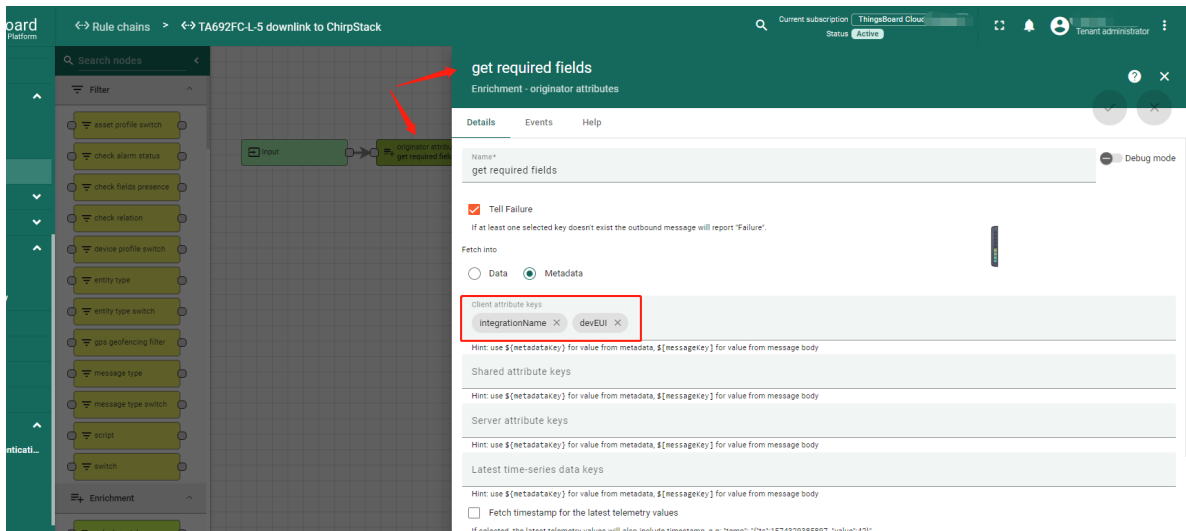
- Show it in the list of Rule chains -> Click on the row.



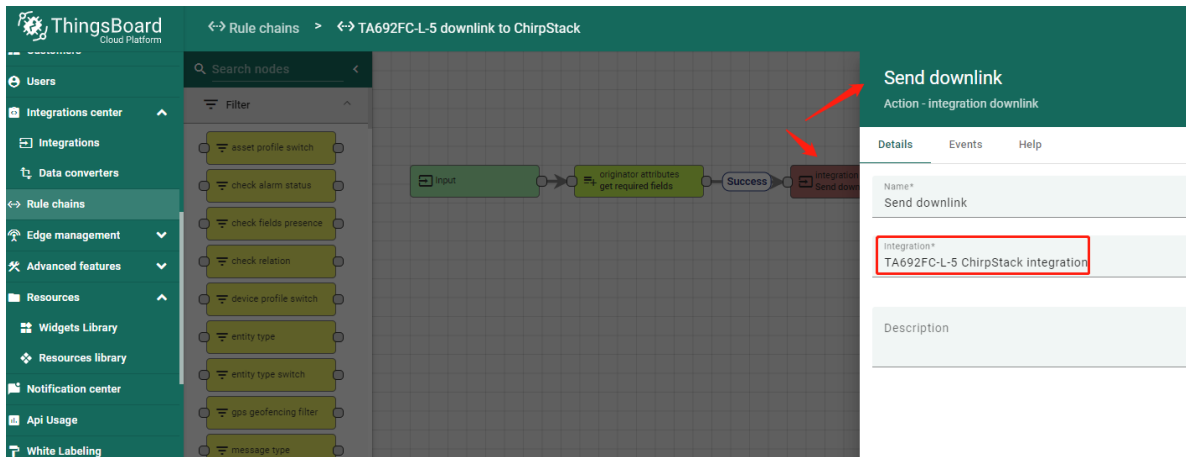
- Show the rule chain details.



- Check the node of **get required fields**.

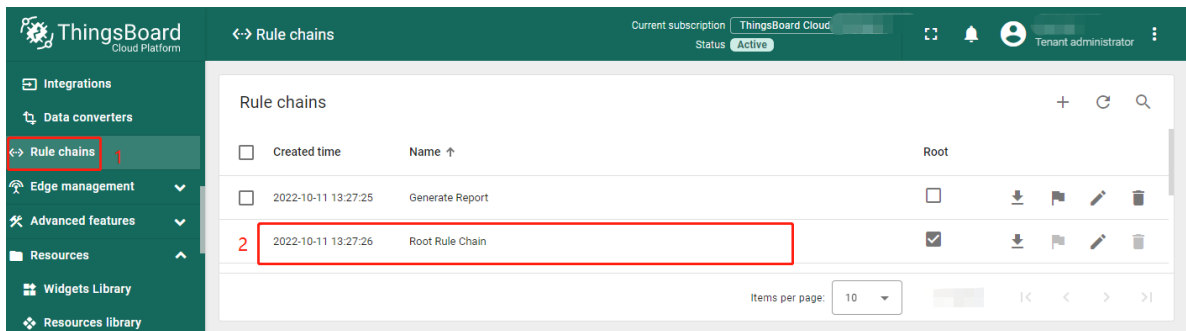


- Check the node of **Send downlink**.

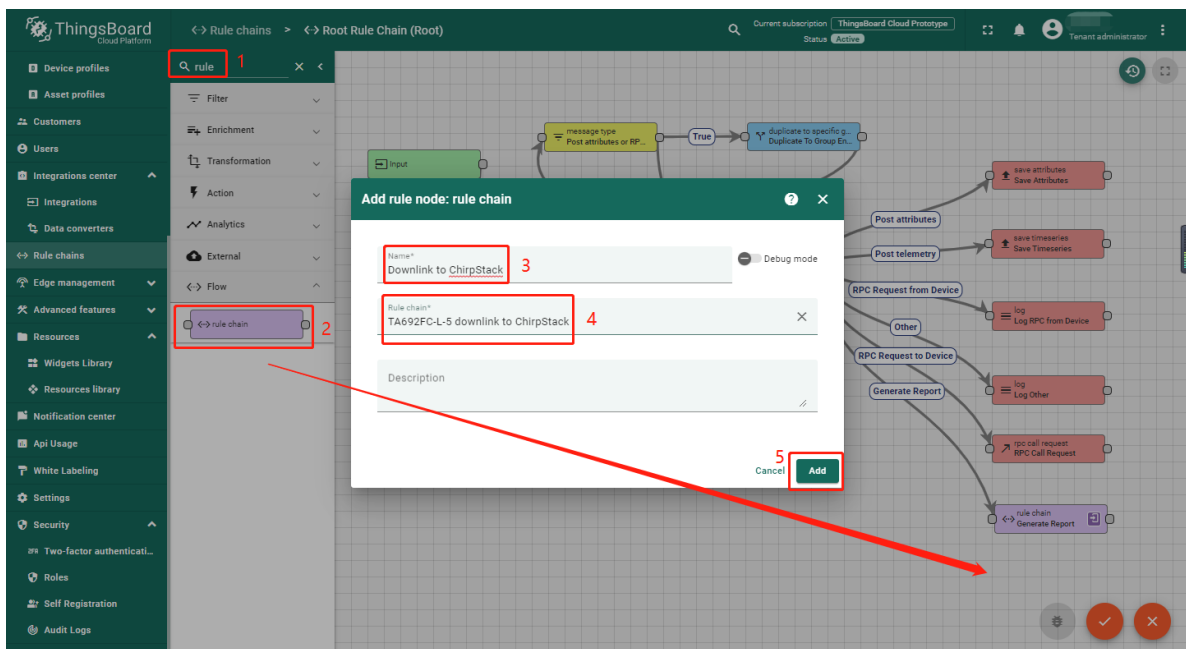


### Step 3.3.5 Configure the root rule-chain

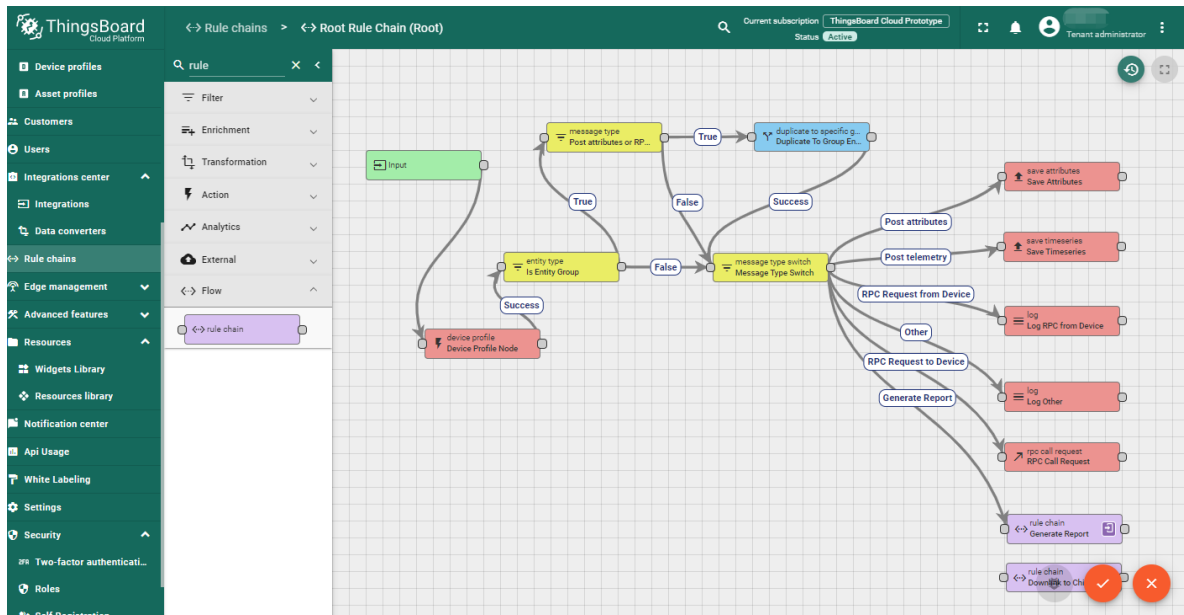
- **Rule chains** → Click on the row.



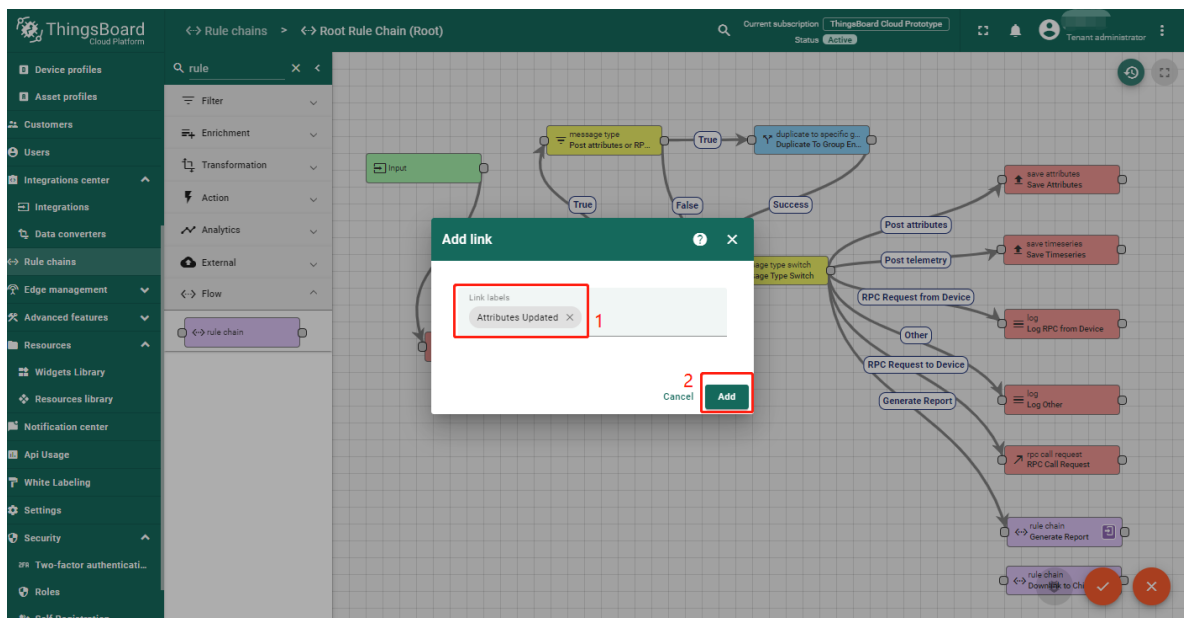
- Drag and drop the **Rule Chain** node → Popup dialog: **Add rule node: rule chain** → Input your node name, eg: *Downlink to ChirpStack* → Select the Rule Chain, eg: *TA692FC-L-5 downlink to ChirpStack* → **Add**.



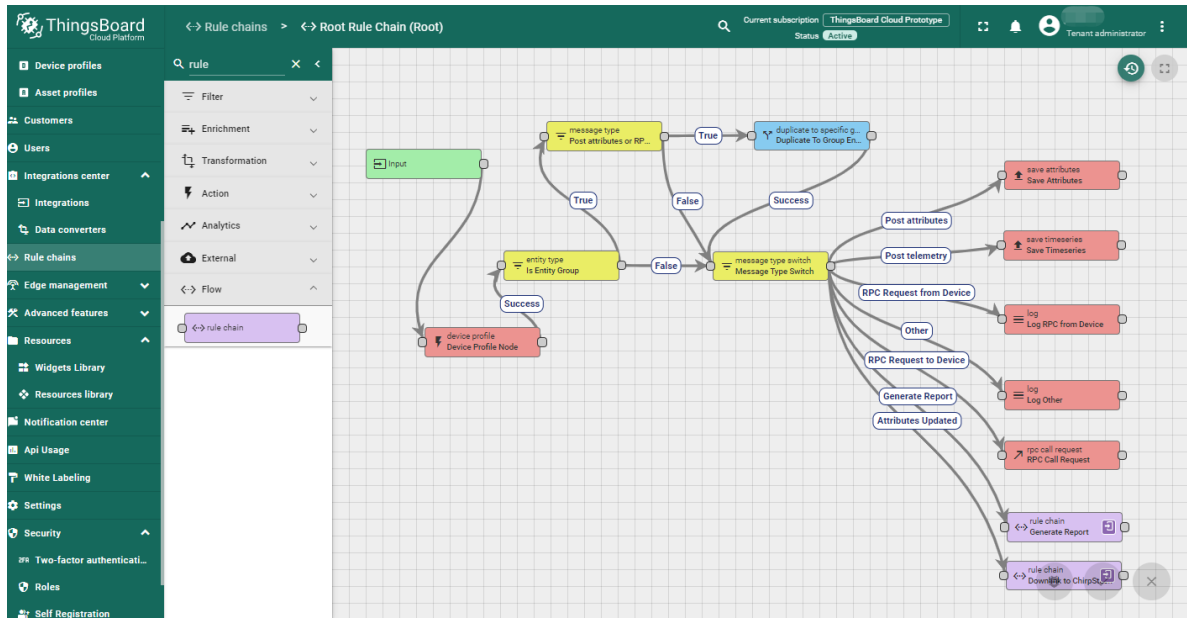
- Now, root rule chain looks like this:



- Add link from **Message Type Switch** to **Downlink to ChirpStack** → Popup dialog: Select a link label: **Attributes updated** → **Add**.



- Save the root rule chain.

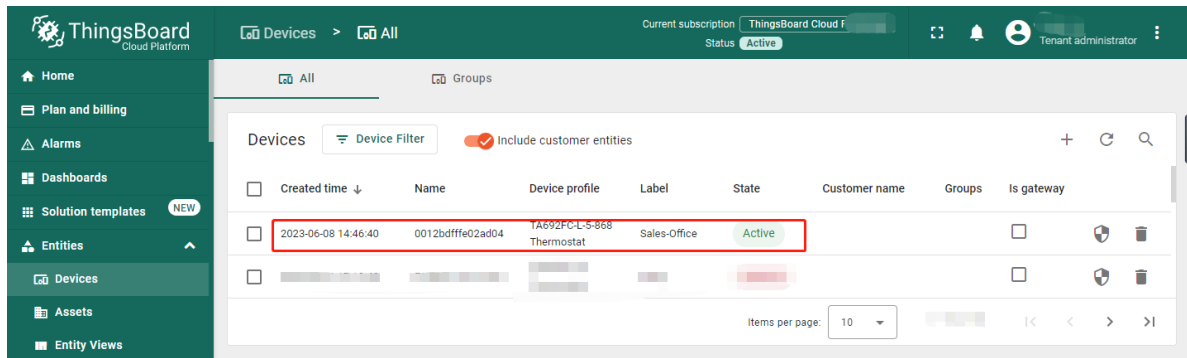


### Step 3.4 Processing Uplink message

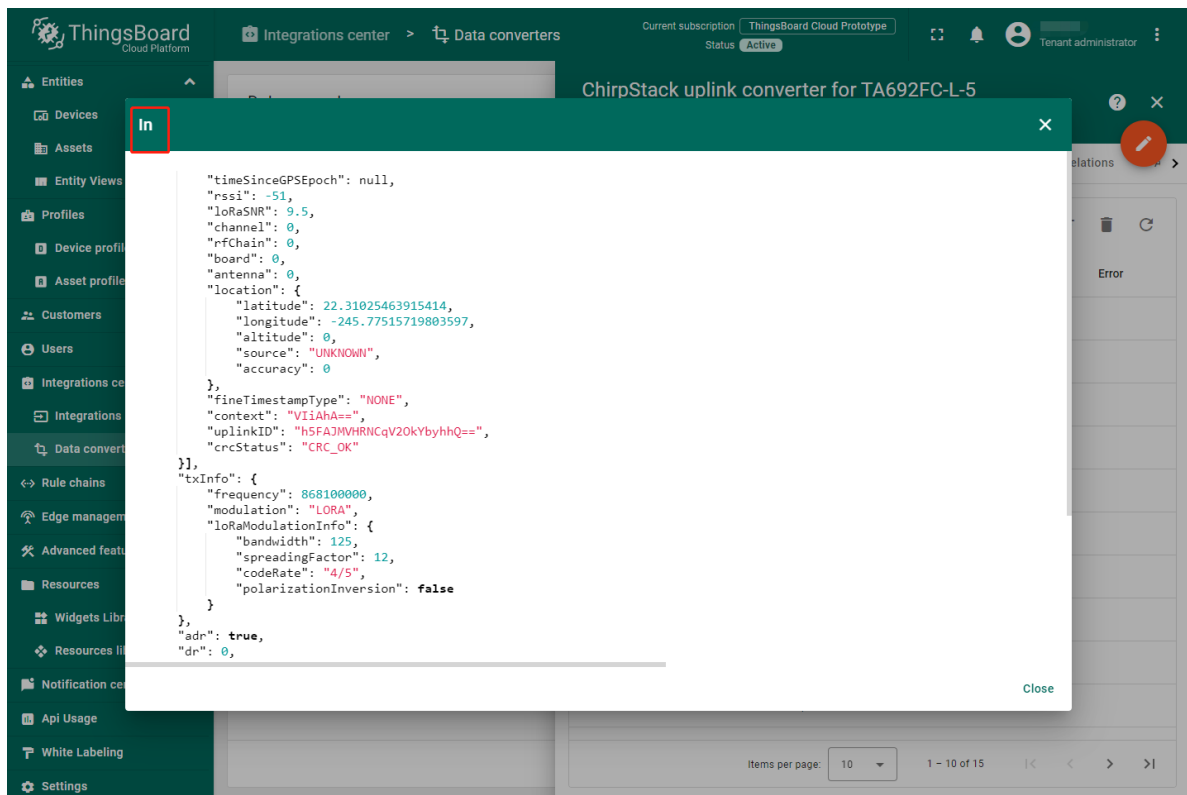
- When device sends uplink message, you will receive an uplink event on integration and data from the device.

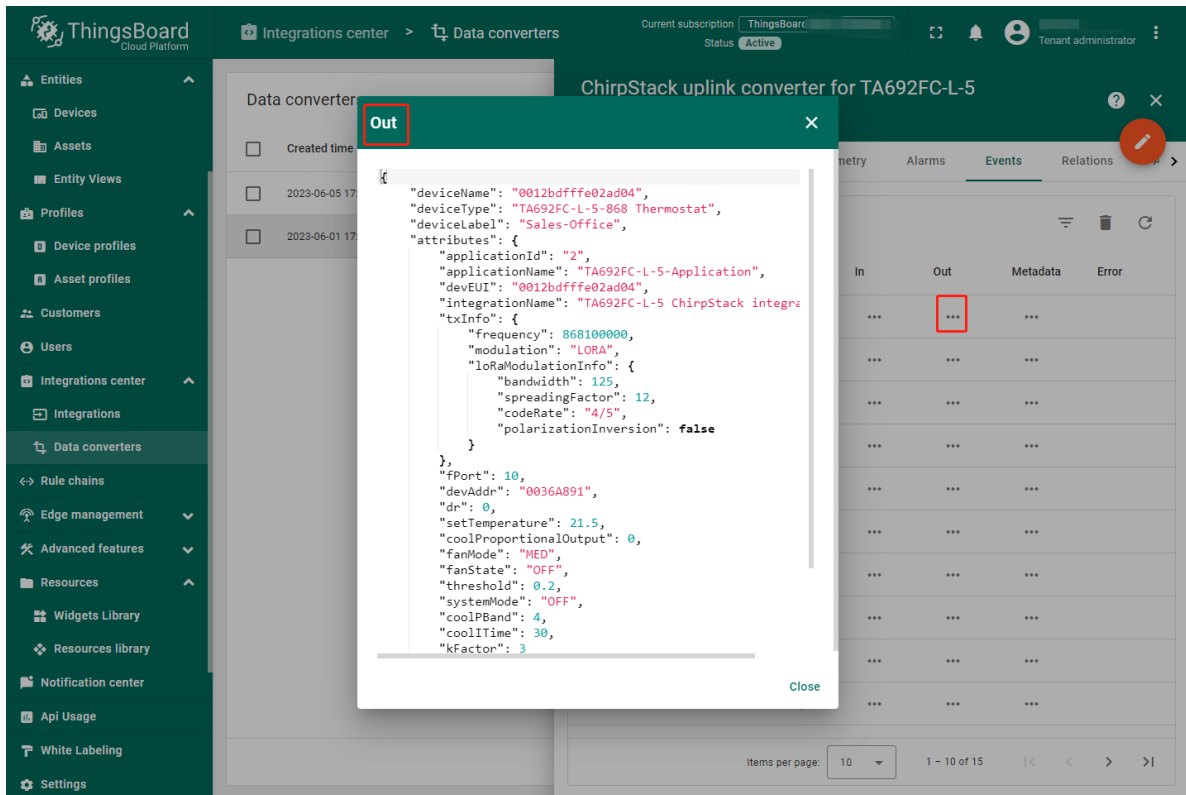
- The created device with data can be seen in the section **Device groups** → **All**.





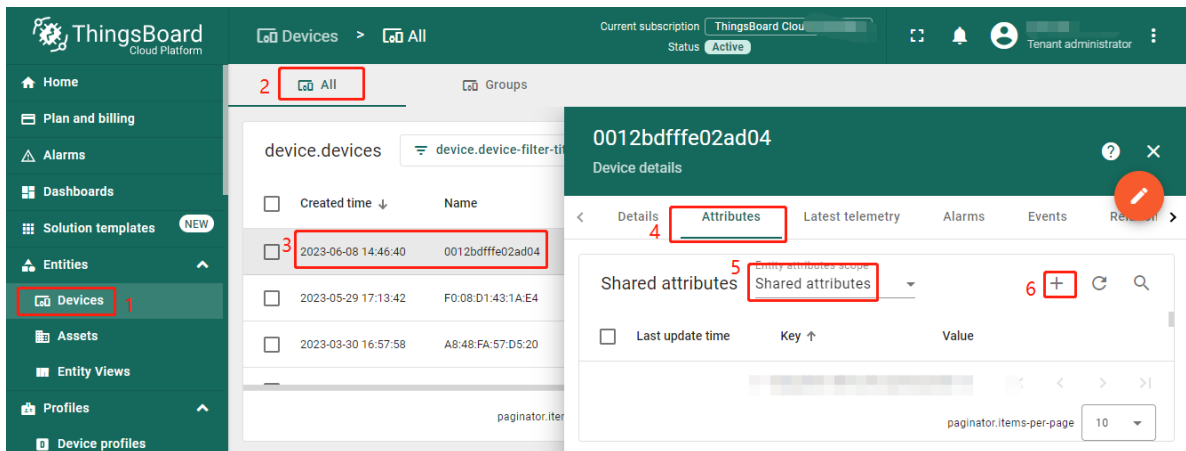
- Received data can be viewed in the Uplink converter. In the “In” and “Out” blocks of the Events tab:

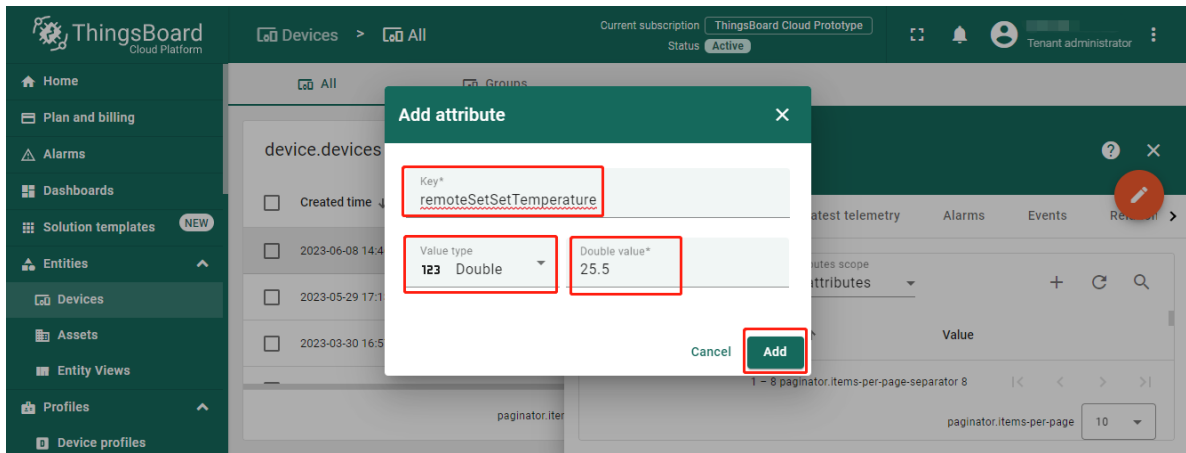




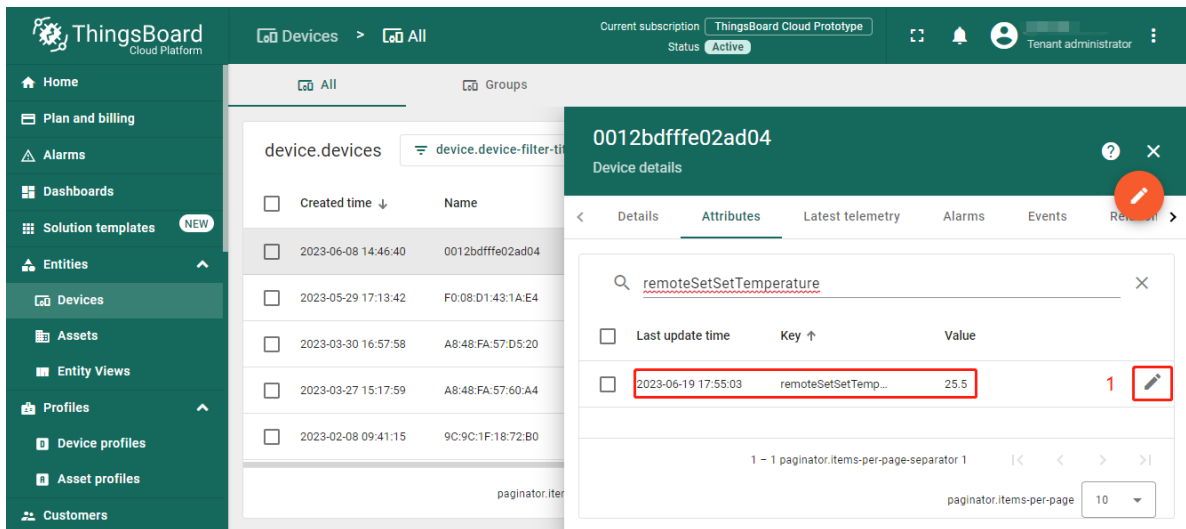
### Step 3.5 Processing Downlink message

- We go to the **Device group** section in the **All** folder, to see this with an example. We add a **remoteSetSetTemperature** of the device in the **Shared attributes** (initialize the **remoteSetSetTemperature** to 25.5).

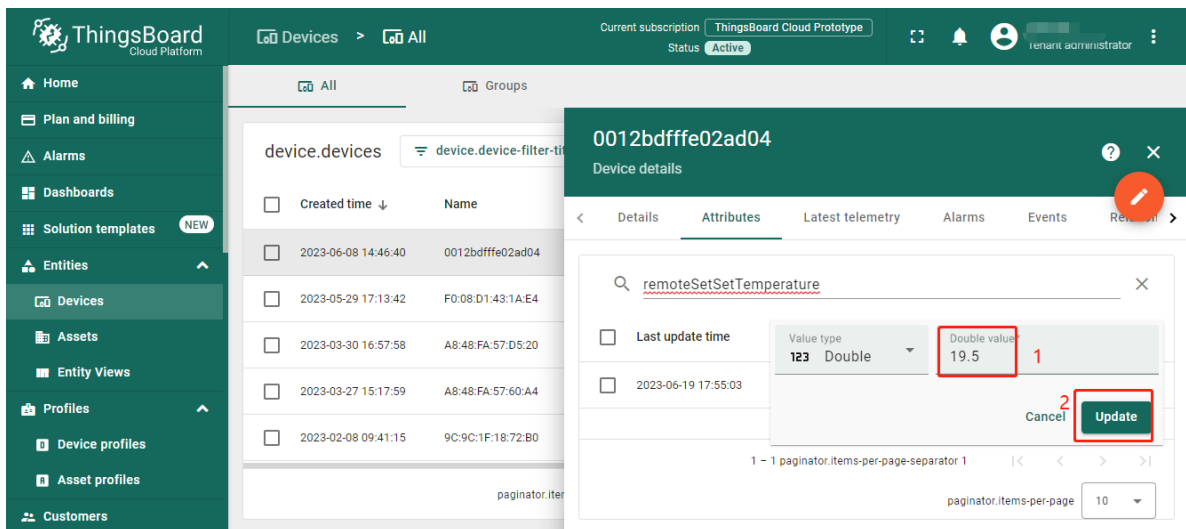




- Now, We have indicated the **remoteSetSetTemperature** of the device in the **Shared attributes**. Now we edit it by clicking on the “pencil” icon.



- Then we make changes to the attribute (change the **remoteSetSetTemperature** to **19.5**) and save the data.



- Received data and data that was sent can be viewed in the downlink converter. In the **In** block of the Events tab,

we see what data entered and the **Out** field displays messages to device:

The top screenshot shows the ThingsBoard interface with the 'Data converters' section open for the 'ChirpStack downlink converter for TA692FC-L-5'. A modal window displays the 'In' field with the following JSON message:

```
{
  "msg": {
    "remoteSetTemperature": 19.5
  },
  "metadata": {
    "cs_integrationName": "TA692FC-L-5 ChirpStack integr...",
    "userLastName": "Lei",
    "scope": "SHARED_SCOPE",
    "userEmail": "office@avantec.com.hk",
    "userFirstName": "Louis",
    "cs_devEUI": "0012bdfffe02ad04",
    "userName": "office@avantec.com.hk",
    "userId": "64f1a8d0-4925-11ed-88cf-3bc720ab387f"
  }
}
```

The bottom screenshot shows the same interface, but the 'Out' field is highlighted. A modal window displays the 'Out' field with the following JSON message:

```
{
  "contentType": "TEXT",
  "data": "AMM=",
  "metadata": {
    "DevEUI": "0012bdfffe02ad04",
    "fPort": 91
  }
}
```

### Step 3.6 Visual Data

Use the Dashboards to work with data. Dashboards are a modern format for collecting and visualizing data sets. Visibility of data presentation is achieved through a variety of widgets.

1. *Update Avantec Widgets.*
2. *Import TA692FC-L-5 Detail Dashboard.*
3. *Import TA692FC-L-5 List Dashboard.*
4. *Modify TA692FC-L-5-868 Thermostat device profile's mobile dashboard.*

For more information about the dashboard, please refer to [here](#).

## 6.3 MultiTech Conduit® MTCAP-868-041A

### 6.3.1 MTCAP Series

Refer to [MultiTech Conduit® AP Access Point for LoRa® Technology \(MTCAP Series\)](#).

The MultiTech Conduit® AP conveniently provides deep in-building connectivity and improved performance for network operators and enterprises connecting thousands of IoT assets by harnessing the power of the LoRaWAN® protocol.

Easy to deploy, the Conduit AP access point extends LoRa® connectivity in commercial buildings like hotels, convention centers, offices and retail facilities providing coverage in difficult to reach areas cell tower or rooftop deployments may not penetrate.

### 6.3.2 MTCAP-868-041A

Refer to:

- [WebSite: MTCAP-868-041A](#)
- [Data Sheet: MultiTech Conduit® AP Access Point for LoRa® Technology \(EU868\)](#)
- [Quick Start Guide: Conduit AP Quick Start Guide](#)
- [Application Note: Configuring mDot with Conduit AP using LoRa](#)
- [Software Guide: mPower™ Edge Intelligence Conduit AEP Software Guide](#)

Ethernet Only mPower Programmable Access Point with external LoRa antenna and EU/UK Accessory Kit.

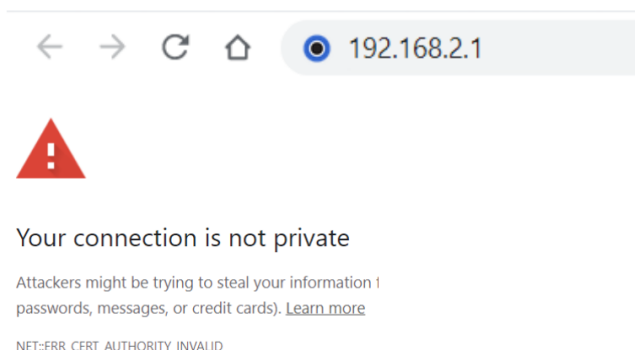
Accessory kit includes power supply and blade(s), LoRa antenna, Ethernet cable, mounting bracket and Quick Start Guide.

### First-Time Setup of Gateway

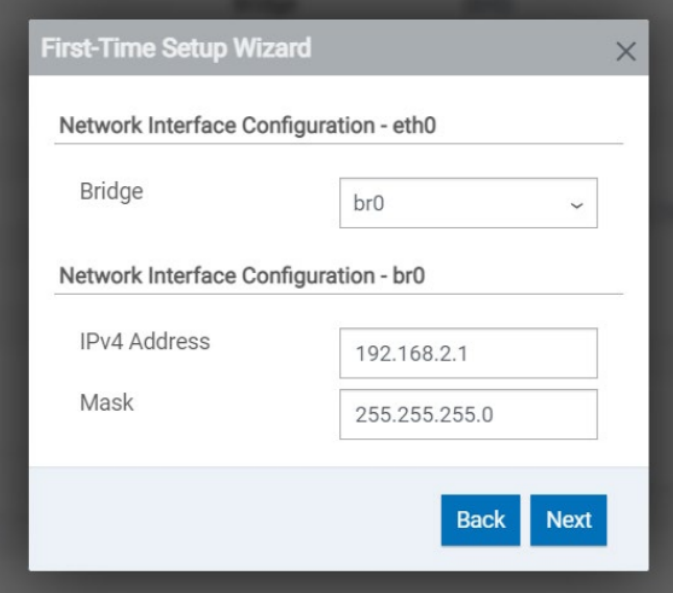
**Note:** This configuration works with the LoRa packet forwarder. You may choose to use Ethernet with LoRa or Cellular with LoRa (if your model has a cellular radio).

This setup wizard helps you configure the main features of your device for initial setup. In most cases, you can accept the provided defaults. See **First-Time Setup** in the [software guide](#) for more details.

1. Go to your browser's address line and enter the default IP address for the gateway to access the UI: 192.168.2.1. Most browsers display a warning about HTTP addresses being unsafe. Click on **Advanced** and continue to 192.168.2.1.



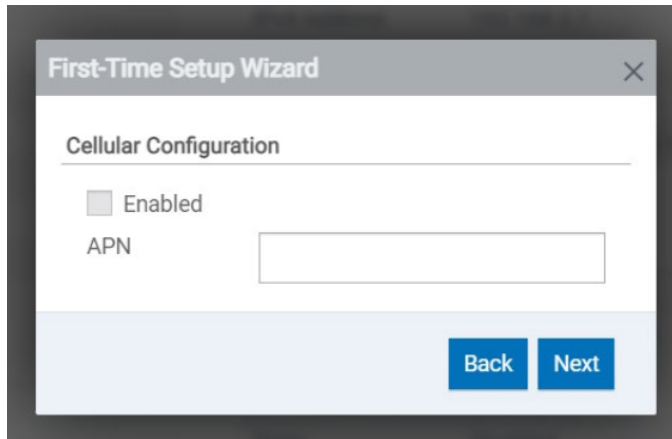
2. Upon power up for the first time, the device will be in commissioning mode. The system requires you to set up an administrative user. Enter your desired username and click **OK**.
3. Enter a desired password for the administrative user and click **OK**. This password must be of sufficient length and strength (with a mix of character classes such as letters, numbers, and symbols). Enter the password again to confirm. Click **OK**.
4. The login page appears. Enter username and password.
5. **First-Time Setup Wizard** appears.
6. **For Call Home,**
  - a. Accept all default settings (disabled).
  - b. Click **Next**.
7. **Set the date, time, and time zone.**
  - a. If the information is correct, accept the default values.
  - b. Otherwise, update **Date**, **Time**, and/or **Time Zone**.
  - c. Click **Next**.
8. **Configure LAN network interfaces Eth0 and Br0.**
  - a. Accept all default settings - **eth0** assigned to the bridge **br0** (with DHCP set automatically). **NOTE:** You will need to make additional configuration changes for Ethernet under Network Interfaces after **First-Time Setup**. See *Using Ethernet with LoRa Packet Forwarder*.
  - b. Click Next.



The screenshot shows a window titled "First-Time Setup Wizard" with a close button (X) in the top right corner. The window is divided into two sections. The first section is titled "Network Interface Configuration - eth0" and contains a "Bridge" label followed by a dropdown menu showing "br0". The second section is titled "Network Interface Configuration - br0" and contains two input fields: "IPv4 Address" with the value "192.168.2.1" and "Mask" with the value "255.255.255.0". At the bottom right of the window are two blue buttons labeled "Back" and "Next".

9. **Configure your device's Cellular connection.**
  - a. If you have no cellular radio (Ethernet only) in your device or plan to use Ethernet with the LoRa packet forwarder, accept all defaults with **Enabled** deactivated (Cellular is disabled) and APN left blank.
  - b. If you have a cellular radio model and plan to use **Cellular** with the LoRa packet forwarder, select **Enabled** (Cellular is enabled).
  - c. If required by your network carrier, enter your **APN** (Some carrier networks may set it automatically via OTA registration. Leave it blank in that case).

- d. Click **Next**.

The image shows a 'First-Time Setup Wizard' window with a title bar containing a close button. The main content area is titled 'Cellular Configuration'. It features a checkbox labeled 'Enabled' which is currently unchecked. Below the checkbox is a text input field labeled 'APN'. At the bottom right of the window, there are two blue buttons: 'Back' and 'Next'.

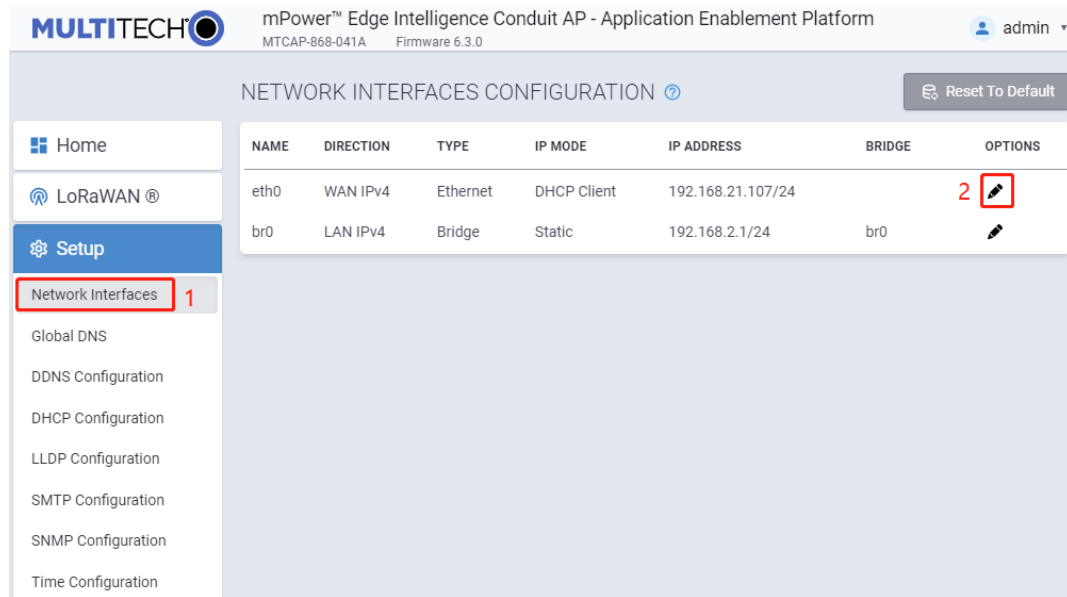
10. **For Cellular Authentication,**
  - a. Accept all defaults (NONE).
  - b. Click **Next**.
11. **For Remote Management,**
  - a. Accept all defaults (disabled).
  - b. Click **Next**.
12. **For HTTP/HTTPS Access,**
  - a. Accept all defaults (enable HTTP to HTTPS via LAN).
  - b. Click **Next**.
13. **For Bootloader Protection (setting a u-boot password),**
  - a. Disable Bootloader Protection (defaults vary with firmware version).
  - b. Click **Finish**.
14. To save your changes, click **Save and Apply**.

## Configuring LoRa Packet Forwarder

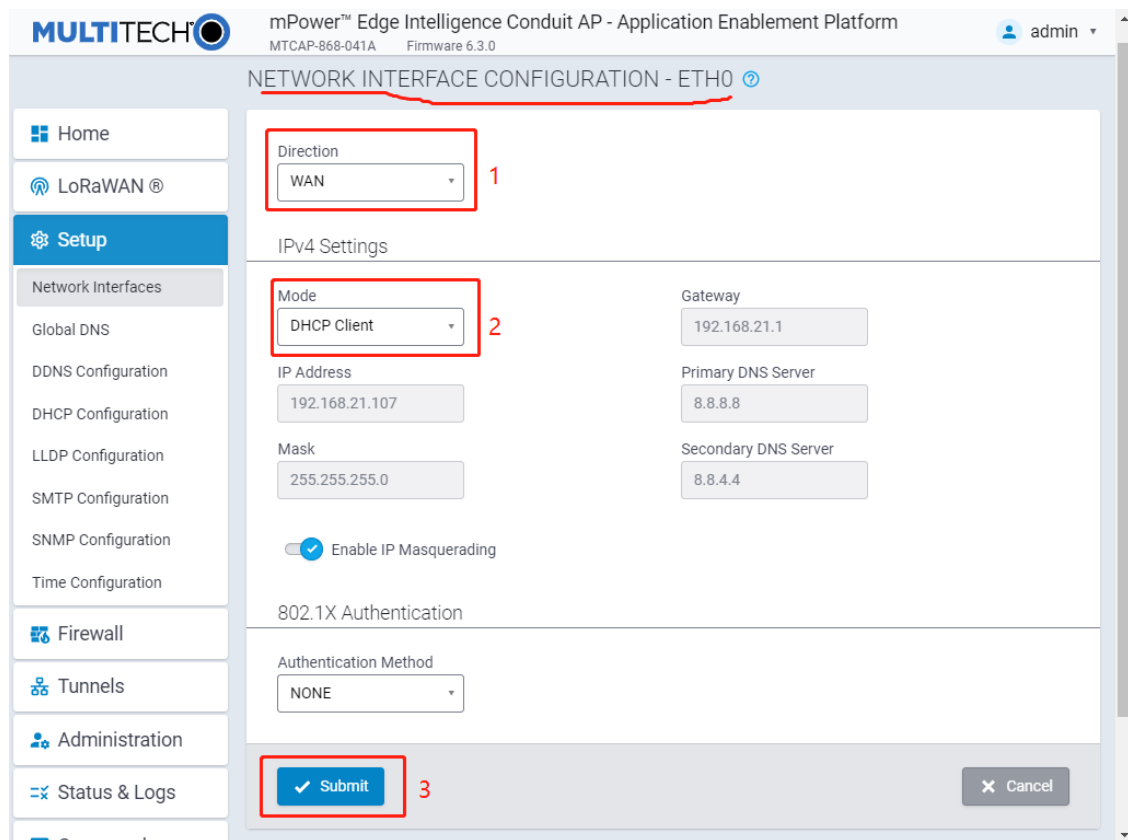
### Using Ethernet with LoRa Packet Forwarder

If you are planning to use Ethernet with the LoRa Packet Forwarder, then you must make this configuration change below before configuring and running Packet Forwarder. If you are using a Cellular connection with LoRa, you can skip these steps.

1. Go to **Network Interfaces**. Click the pencil for the **eth0** interface.

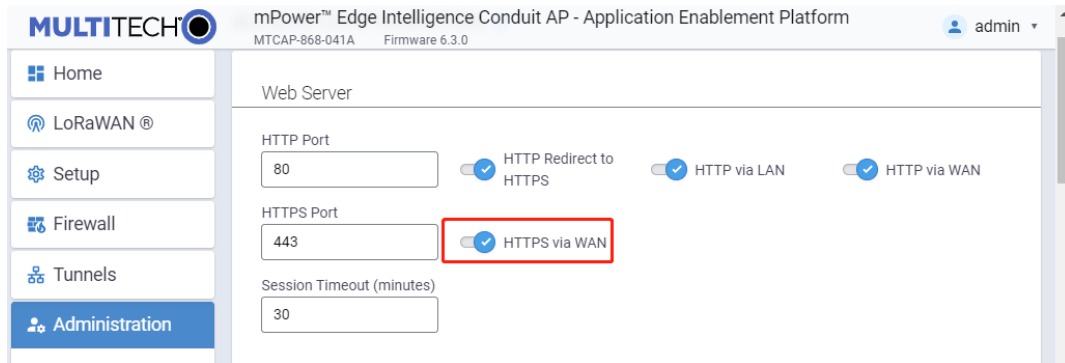


2. Under Network Interface ETH0,
  - a. change **Direction** to **WAN**.
  - b. Under **Mode**, select **DHCP Client**.
  - c. Click **Submit**.



3. Go to **Administration** > **Access Configuration** > **HTTPS**, Enable HTTPS via WAN, then click **Submit**.



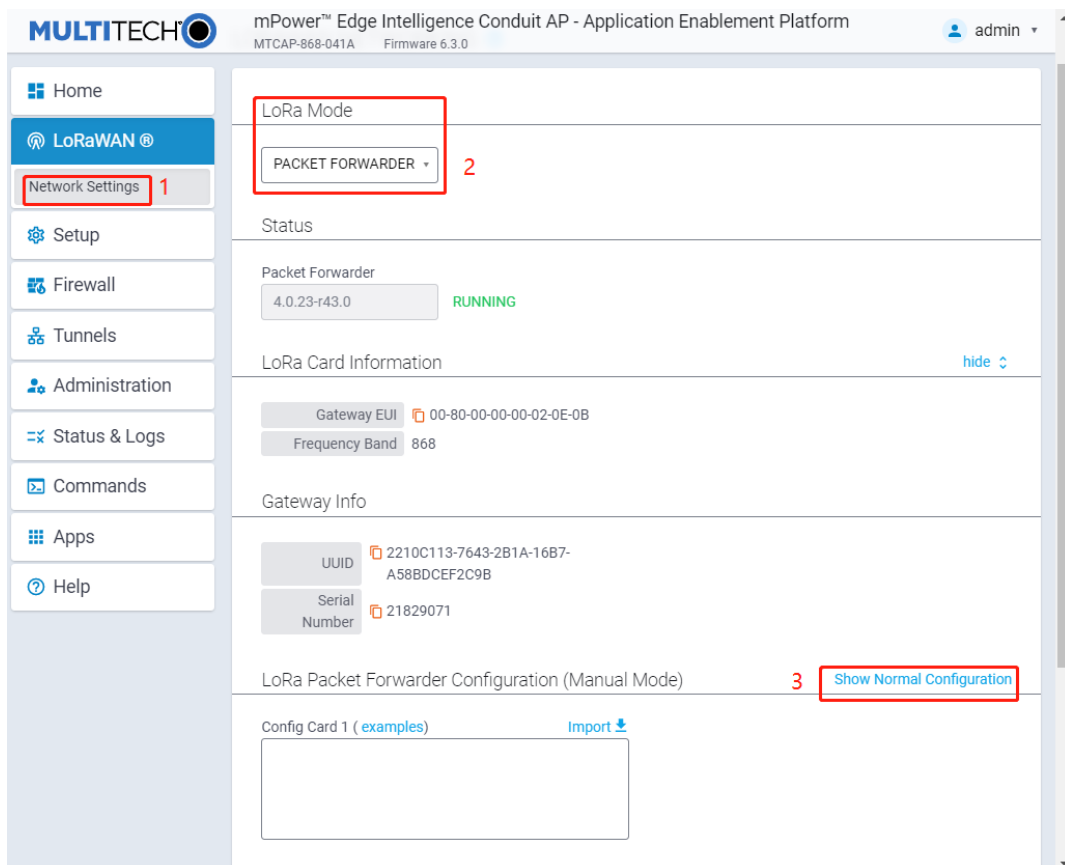


4. Click Save and Apply.

## Configuring the Gateway

To activate **LoRa Packet Forwarder Mode** on your device:

1. For **LoRaWAN > Network Settings > LoRa Mode**, select **PACKET FORWARDER** under **Mode**. If **Manual Configuration** is showing, click **Normal Configuration** to switch.



2. For **LoRaWAN > Network Settings > LoRa Packet Forwarder Configuration**,
  - a. You should select the network to use from the drop-down under **Network**: **Manual**, **Radio Bridge**, **ChirpStack**, **The Things Network**, **Senet**, or **Loriot**. In this case, select **Manual**.
  - b. Select the appropriate **Channel Plan** for the Packet Forwarder. Choose from the drop-down menu:

**US915: 915, AU915: 915, AS923-1: 915, AS923-2: 915, AS923-3: 915, AS923-4: 915, KR920: 915, EU868: 868, IN865: 868, RU864: 868, or ISM2400: 2400.** In this case, select **EU868**.

c. Type your LoRaWAN Network **Server address** - your ChirpStack IP.

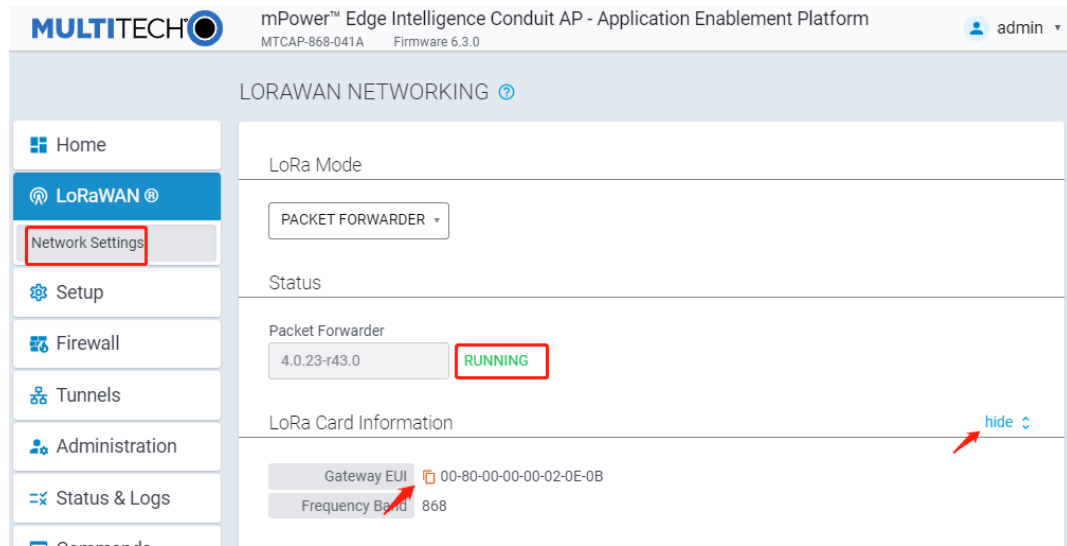
d. Type **upstream port & downstream port**: 1700.

e. Click **Submit**.

The screenshot shows the 'LoRa Packet Forwarder Configuration (Normal Mode)' page in the mPower Edge Intelligence Conduit AP web interface. The page is titled 'MULTITECH mPower™ Edge Intelligence Conduit AP - Application Enablement Platform' with 'MTCAP-868-041A' and 'Firmware 6.3.0' displayed. A user 'admin' is logged in. The configuration is divided into several sections: 'Network Settings' (containing 'Network' set to 'Manual' and 'Channel Plan' set to 'EU868'), 'Basics', 'Server Settings' (containing 'Server Address' set to '13.4.7.149' and 'Upstream Port' and 'Downstream Port' both set to '1700'), 'Forward CRC', 'Channel Config', 'Duty Cycle', and 'Intervals'. At the bottom, there is a 'ChirpStack Gateway Bridge Configuration' section with a toggle switch for 'Enabled' and a 'Submit' button. A red arrow points to the 'LoRa Packet Forwarder Configuration' title, and red boxes with numbers 1 through 5 highlight the 'Network' dropdown, 'Channel Plan' dropdown, 'Server Address' text input, the 'Upstream Port' and 'Downstream Port' text inputs, and the 'Submit' button respectively.

3. Then, click **Save and Apply**.

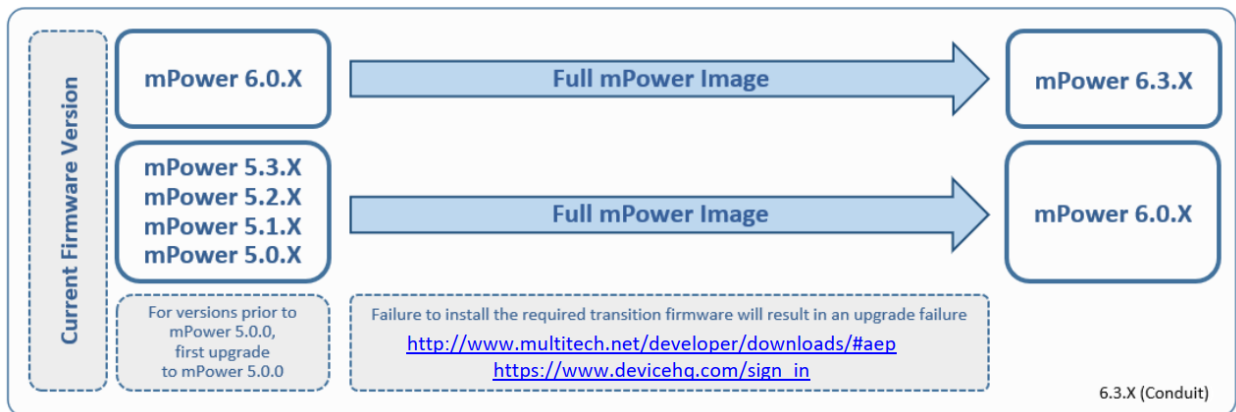
4. Confirm that the Packet Forwarder is now running under **Status**. Show the **LoRa Card Information** and copy the **Gateway EUI** (save for later).



- Make sure to properly add your gateway and any end devices to the network based on their specific system and instructions. For this example, we will configure the gateway and end device using **Manual**.

### Optional: Firmware Upgrade

**Tip:** To install mPower 6.3.0, the Conduit gateway must be upgraded to mPower 6.0.0 or higher. Customers that are running earlier versions of mPower should use the following upgrade process.



- Download a new firmware form [here](#).

**MultiTech Developer Resources**  
A Community of Developers

**MULTITECH**

Privacy Statement Terms of Use Sitemap News Products Software Downloads RED Compliance Forums

Source Code MultiTech.com Type text to search here...

### Downloads

#### Conduit: mPower Models (MTCDT-X-2XXA or MTCDTIP-x-2xxA)

Firmware Files: ([Changelog](#) | See mPower software guide for upgrade instructions)

- MTCDT 6.3.0 (signed firmware) with Release Notes
- MTCDT 6.0.4 (signed firmware) with Release Notes
- MTCDT 5.3.8s-s1 (signed firmware) with Release Notes
- Previous Versions

#### Conduit Access Point and Conduit IP67 200 Series: mPower Models (MTCAP and MTCDTIP2)

- MTCAP and MTCDTIP2 6.3.0 (signed firmware) with Release Notes
- MTCDTIP2 6.0.4 (signed firmware) with Release Notes
- MTCAP 6.0.1 (signed firmware) with Release Notes
- MTCAP 5.3.8s-s1 (signed firmware) with Release Notes
- Previous Versions

#### Conduit: mLinux Model

MTCDT Commissioning Image: ([Changelog](#) | [Upgrade Instructions](#))

- mLinux 6.3.0 with Release Notes TBD
- mLinux 6.0.1 with Release Notes
- mLinux 5.3.31 with Release Notes
- mLinux 5.3.5 with Release Notes
- mLinux 5.3.0b with Release Notes

User

- Register
- Log in

Sitemap

- View all pages

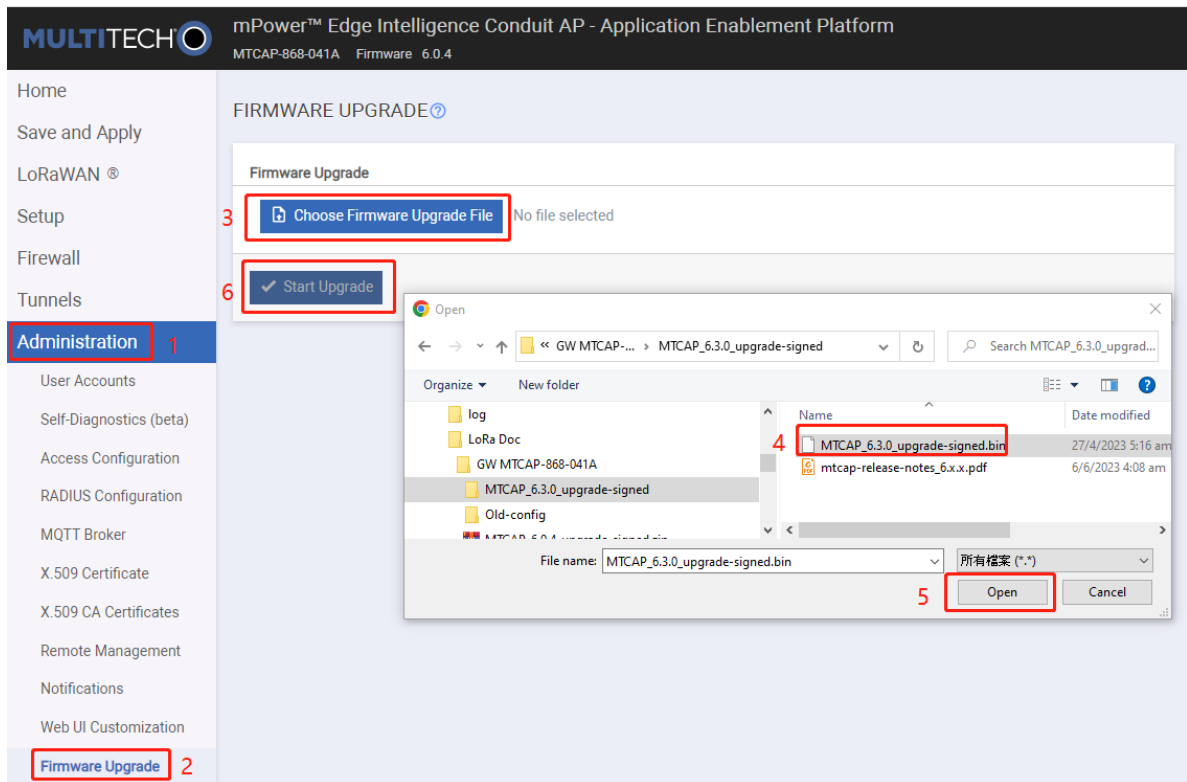
Pages

- Privacy Statement
- Terms of Use
- Sitemap
- Search Results
- Home
- News
- Products
- Software
- Downloads

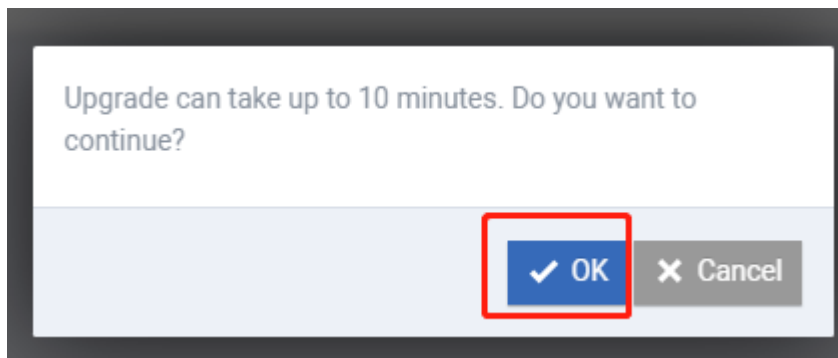
Forums

- Device HQ
- Dragonfly
- General
- Lora Network Server
- mDot/xDot
- Conduit: AEP Model
- Conduit: mLinux Model

- Upgrade: For **Administration > Firmware Upgrade > Choose Firmware Upgrade File**, select *a new firmware* > **Open > Start Upgrade >**



- Click **OK** in a popup dialog.



- Wait a few minutes. After the firmware upgrade is successful, MTCAP will automatically restart. If the upgrade is successful, you will see the new version number after logging in.

The screenshot displays the 'mPower™ Edge Intelligence Conduit AP - Application Enablement Platform' web interface. The top header shows the 'MULTITECH' logo, the device model 'MTCAP-868-041A', and the firmware version 'Firmware 6.3.0'. A user 'admin' is logged in. The main section is titled 'DEVICE INFORMATION' and is divided into three columns. The left column contains a sidebar menu with options: Home, LoRaWAN®, Network Settings, Setup, Network Interfaces, Global DNS, DDNS Configuration, DHCP Configuration, LLDP Configuration, SMTP Configuration, SNMP Configuration, and Time Configuration. The middle column shows 'WAN' settings for 'Ethernet (eth0)' with fields for Mode (DHCP Client), MAC Address (00:08:00:4B:F9:16), IPv4 Address (192.168.21.107), Mask (255.255.255.0), Gateway (192.168.21.1), DNS (8.8.8.8, 8.8.4.4), and 802.1X Auth Type (None). The right column shows 'LAN' settings with a message 'No network interface configured as LAN' and 'LoRa' settings including Model Number (MTCAP-LORA-868), Hardware (MTCAP-LORA-1.5), Frequency Band (868), and EUI (00-80-00-00-00-02-0E-0B). A 'Current Time' of 2023/6/14 下午12:10:22 and 'Up Time' of 00:23:50 are also displayed. The bottom status bar indicates 'Last update: 下午12:10:23'.

## 6.4 ChirpStack v3

### 6.4.1 Quick start Amazon AWS

Refer to [Quickstart Debian or Ubuntu](#).

This section describes the steps needed to setup the ChirpStack stack including all requirements on a single machine. It has been tested on the following distributions (but with non or minimal modifications it will work on other versions too):

- Ubuntu 20.04 LTS

Please refer to the other install pages for more generic installation instructions.

### Assumptions

Many configurations of these packages are possible. Dependent software packages could be installed on any number of remote servers, and the packages themselves could be on their own servers. However, in order to simplify the initial installation, we will assume the following deployment architecture:

- All ChirpStack components and their dependencies will be installed on a single server instance.
- The [ChirpStack Gateway Bridge](#) component will be installed on the server, but can also be installed on the gateway itself.
- No firewall rules are setup.

Of course, optimizations may need to be made depending on the performance of your systems. You may opt to move the PostgreSQL database to another server. Or you may decide to put your MQTT broker on a different system, or even use a different server than the one recommended in this document. These and other installation changes are beyond the scope of this document. However, you should be able to find the information here that would make these changes relatively straight-forward.

## Prerequisites

1. Launch an Amazon EC2 instance:

Item	Description
Software Image (AMI)	Ubuntu Server 20.04 LTS (HVM), SSD Volume Type, 64-bit (x86)
Virtual server type (instance type)	t3.micro, Family: t3, 2 vCPU, 1 GiB Memory, Current generation: true
Firewall (security group)	ssh; https; 8080, tcp, http; 1700, udp, LoRaWAN uplink
Configure storage	1 x 8 GiB, Volume type: gp2, Root volume (Not encrypted)

2. Allocate Elastic IP address.
3. Associate Elastic IP address with the EC2 instance.

## Install dependencies

- **MQTT broker** - A publish/subscribe protocol that allows users to publish information under topics that others can subscribe to. A popular implementation of the MQTT protocol is [Mosquitto](#).
- **Redis** - An in-memory database used to store relatively transient data.
- **PostgreSQL** - The long-term storage database used by the open source packages.

Use the package manager apt to install these dependencies:

```
sudo apt update
sudo apt install mosquitto mosquitto-clients redis-server redis-tools postgresql
```

## Setup PostgreSQL databases and users

To enter the command line utility for PostgreSQL:

```
sudo -u postgres psql
```

Inside this prompt, execute the following queries to set up the databases that are used by the ChirpStack stack components. It is recommended to change the usernames and passwords. Just remember to use these other values when updating the `chirpstack-network-server.toml` and `chirpstack-application-server.toml` configuration files. Since these two applications both use the same table to track database upgrades, they must have separate databases.

```
-- set up the users and the passwords
-- (note that it is important to use single quotes and a semicolon at the end!)
create role chirpstack_as with login password 'dbpassword';
create role chirpstack_ns with login password 'dbpassword';

-- create the database for the servers
create database chirpstack_as with owner chirpstack_as;
create database chirpstack_ns with owner chirpstack_ns;

-- change to the ChirpStack Application Server database
```

(continues on next page)

(continued from previous page)

```
\c chirpstack_as
-- enable the pq_trgm and hstore extensions
-- (this is needed to facilitate the search feature)
create extension pg_trgm;
-- (this is needed to store additional k/v meta-data)
create extension hstore;

-- exit psql
\q
```

### Setup ChirpStack software repository

ChirpStack provides a repository that is compatible with the Ubuntu apt package system. First make sure that both `dirmngr` and `apt-transport-https` are installed:

```
sudo apt install apt-transport-https dirmngr
```

Set up the key for this new repository:

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 1CE2AFD36DBCCA00
```

Add the repository to the repository list by creating a new file:

```
sudo echo "deb https://artifacts.chirpstack.io/packages/3.x/deb stable main" | sudo tee /
→etc/apt/sources.list.d/chirpstack.list
```

Update the apt package cache:

```
sudo apt update
```

### Install ChirpStack Gateway Bridge

Note: If you intend to run the [ChirpStack Gateway Bridge](#) only on gateway(s) themselves, you can skip this step.

Install the package using apt:

```
sudo apt install chirpstack-gateway-bridge
```

log output:

```
-----
The configuration file is located at:
/etc/chirpstack-gateway-bridge/chirpstack-gateway-bridge.toml

Some helpful commands for chirpstack-gateway-bridge:
Start:
$ sudo systemctl start chirpstack-gateway-bridge

Restart:
$ sudo systemctl restart chirpstack-gateway-bridge
```

(continues on next page)



(continued from previous page)

```

Stop:
$ sudo systemctl stop chirpstack-gateway-bridge

Display logs:
$ sudo journalctl -f -n 100 -u chirpstack-gateway-bridge
-----

```

The configuration file is located at `/etc/chirpstack-gateway-bridge/chirpstack-gateway-bridge.toml`. The default configuration is sufficient for this guide.

Start the ChirpStack Gateway Bridge service:

```

# start chirpstack-gateway-bridge
sudo systemctl start chirpstack-gateway-bridge

# start chirpstack-gateway-bridge on boot
sudo systemctl enable chirpstack-gateway-bridge

```

## Install ChirpStack Network Server

Install the package using apt:

```
sudo apt install chirpstack-network-server
```

log output:

```

-----
The configuration file is located at:
/etc/chirpstack-network-server/chirpstack-network-server.toml

Some helpful commands for chirpstack-network-server:
Start:
$ sudo systemctl start chirpstack-network-server

Restart:
$ sudo systemctl restart chirpstack-network-server

Stop:
$ sudo systemctl stop chirpstack-network-server

Display logs:
$ sudo journalctl -f -n 100 -u chirpstack-network-server
-----

```

The configuration file is located at `/etc/chirpstack-network-server/chirpstack-network-server.toml` and must be updated to match the database and band configuration. See below two examples for the **EU868** and **US915** band. For more information about all the ChirpStack Network Server configuration options, see [here](#) or [ChirpStack Network Server configuration](#).

After updating the configuration, you need to restart the ChirpStack Network Server and validate that there are no errors.

Start the ChirpStack Network Server service:

```
# start chirpstack-network-server
sudo systemctl start chirpstack-network-server

# start chirpstack-network-server on boot
sudo systemctl enable chirpstack-network-server
```

Print the ChirpStack Network Server log-output:

```
sudo journalctl -f -n 100 -u chirpstack-network-server
```

### EU868 configuration example

```
[general]
log_level=4

[postgresql]
dsn="postgres://chirpstack_ns:dbpassword@localhost/chirpstack_ns?sslmode=disable"

[network_server]
net_id="0000000"

[network_server.band]
# name="EU_863_870"
name="EU868"

[[network_server.network_settings.extra_channels]]
frequency=867100000
min_dr=0
max_dr=5

[[network_server.network_settings.extra_channels]]
frequency=867300000
min_dr=0
max_dr=5

[[network_server.network_settings.extra_channels]]
frequency=867500000
min_dr=0
max_dr=5

[[network_server.network_settings.extra_channels]]
frequency=867700000
min_dr=0
max_dr=5

[[network_server.network_settings.extra_channels]]
frequency=867900000
min_dr=0
max_dr=5
```

**US915 configuration example sub-band 1 (125kHz channels 0 - 7 & 500kHz channel 64)**

```
[general]
log_level=4

[postgresql]
dsn="postgres://chirpstack_ns:dbpassword@localhost/chirpstack_ns?sslmode=disable"

[network_server]
net_id="0000000"

[network_server.band]
# name="US_902_928"
name="US915"

[network_server.network_settings]
enabled_uplink_channels=[0, 1, 2, 3, 4, 5, 6, 7, 64]
```

**US915 configuration example sub-band 2 (125kHz channels 8 - 15 & 500kHz channel 65)**

This is the same channel-plan as used by The Things Network.

```
[general]
log_level=4

[postgresql]
dsn="postgres://chirpstack_ns:dbpassword@localhost/chirpstack_ns?sslmode=disable"

[network_server]
net_id="0000000"

[network_server.band]
# name="US_902_928"
name="US915"

[network_server.network_settings]
enabled_uplink_channels=[8, 9, 10, 11, 12, 13, 14, 15, 65]
```

**Install ChirpStack Application Server**

Install the package using apt:

```
sudo apt install chirpstack-application-server
```

log output:

```
-----
The configuration file is located at:
/etc/chirpstack-application-server/chirpstack-application-server.toml
```

(continues on next page)

(continued from previous page)

```

Some helpful commands for chirpstack-application-server:
Start:
$ sudo systemctl start chirpstack-application-server

Restart:
$ sudo systemctl restart chirpstack-application-server

Stop:
$ sudo systemctl stop chirpstack-application-server

Display logs:
$ sudo journalctl -f -n 100 -u chirpstack-application-server
-----

```

The configuration file is located at `/etc/chirpstack-application-server/chirpstack-application-server.toml` and must be updated to match the database configuration. See below a configuration example which matches the database which we have created in one of the previous steps. For more information about the ChirpStack Application Server configuration options, see [ChirpStack Application Server configuration](#).

```

[general]
log_level=4

[postgresql]
dsn="postgres://chirpstack_as:dbpassword@localhost/chirpstack_as?sslmode=disable"

[application_server.external_api]
jwt_secret="M9LoHX3wPQlcB2ziakV6qs/F2vLOvkAtrRv1yTu5Kks="

```

Note: you must replace the `jwt_secret` with a secure secret! You could use the command `openssl rand -base64 32` to generate a random secret.

Start the ChirpStack Application Server service:

```

# start chirpstack-application-server
sudo systemctl start chirpstack-application-server

# start chirpstack-application-server on boot
sudo systemctl enable chirpstack-application-server

```

Print the ChirpStack Application Server log-output:

```

sudo journalctl -f -n 100 -u chirpstack-application-server

```

## 6.4.2 Connecting a gateway

Login into the ChirpStack Application Server web-interface. The default credentials are:

- Username: admin
- Password: admin

### Optional: Adding a Network Server

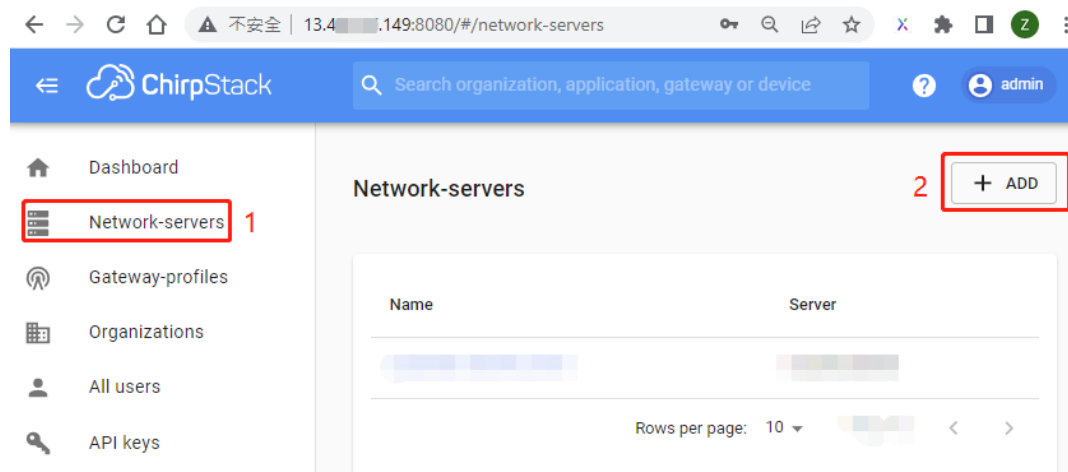
Refer to [Network servers](#).

ChirpStack Application Server is able to connect to one or multiple ChirpStack Network Server instances. Global admin users are able to add new Network Servers to the ChirpStack Application Server installation.

**Note:** once a Network Server is assigned to a [Service Profile](#) or [Device Profile](#), a Network Server can't be removed before deleting these entities, it will return an error.

When creating a new Network Server, ChirpStack Application Server will create a Routing Profile on the given Network Server, containing the `hostname: ip` of the ChirpStack Application Server installation. In case your ChirpStack Application Server installation is not reachable on `localhost`, make sure this `hostname: ip` is configured correctly in your [ChirpStack Application Server Configuration](#). This Routing Profile is updated on Network Server updates and deleted on Network Server deletes.

1. Go to **Network-servers** -> **+Add**.



2. Type some parameters -> **ADD NETWORK-SERVER**.

Item	Description
Network-server name	localhost network server
Network-server server	localhost:8000

3. Show Network servers.

Name	Server
localhost network server	localhost:8000

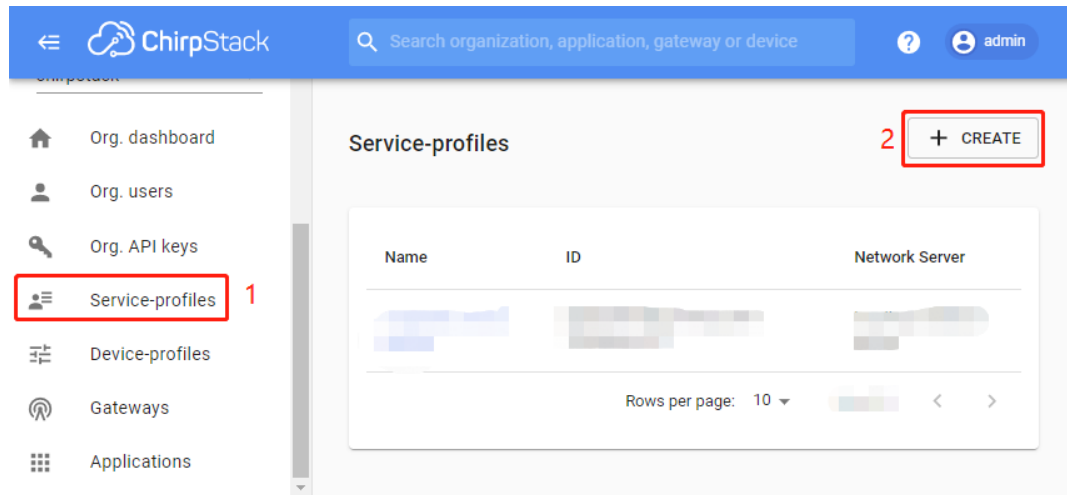
### Optional: Creating a Service-profile

Refer to [Service profiles](#).

The Service Profile can be seen as the “contract” between an user and the network. It describes the features that are enabled for the user(s) of the Service Profile and the rate of messages that can be sent over the network.

When creating a Service Profile, ChirpStack Application Server will create the actual profile on the selected Network Server, and will keep a reference record so it knows to which organization it belongs.

1. Go to **Service-profiles** → **+Create**.



2. Type some parameters → **CREATE SERVICE-PROFILE**

Item	Description
Service-profile name	localhost service profile
Network-server name	localhost network server
Add gateway meta-data	Enable

**Service-profiles / Create**

1 **Service-profile name \***  
localhost service profile  
A name to identify the service-profile.

2 **Network-server \***  
localhost network server  
The network-server on which this service-profile will be provisioned. After creating the service-profile, this value can't be changed.

3 ☒ **Add gateway meta-data**  
GW metadata (RSSI, SNR, GW geoloc., etc.) are added to the packet sent to the application-server.

☐ **Enable network geolocation**  
When enabled, the network-server will try to resolve the location of the devices under this service-profile. Please note that you need to have gateways supporting the fine-timestamp feature and that the network-server needs to be configured in order to provide geolocation support.

Device-status request frequency  
0  
Frequency to initiate an End-Device status request (request/day). Set to 0 to disable.

Minimum allowed data-rate \*  
0  
Minimum allowed data rate. Used for ADR.

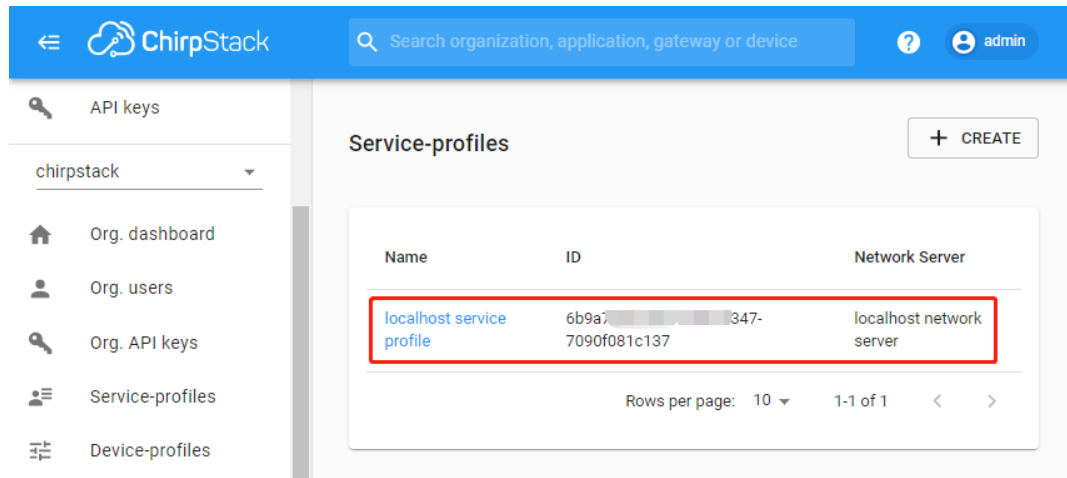
Maximum allowed data-rate \*  
0  
Maximum allowed data rate. Used for ADR.

☐ **Private gateways**  
Gateways under this service-profile are private. This means that these gateways can only be used by devices under the same service-profile.

4 **CREATE SERVICE-PROFILE**

3. Show Service profiles.





## Adding a gateway

Refer to [Connecting gateway](#).

This guide describes how to connect your gateway to ChirpStack and how to validate that it is successfully communicating with the ChirpStack Network Server. At this point it is expected that you have the ChirpStack Application Server and ChirpStack Network Server components up and running.

## Requirements

Before continuing, please make sure that you have installed both a packet-forwarder and the [ChirpStack Gateway Bridge](#). The packet-forwarder that is installed on your gateway and the steps needed to install the ChirpStack Gateway Bridge vary per gateway vendor and model. In some cases you must also install the ChirpStack Gateway Bridge on the gateway. Please refer to the ChirpStack Gateway Bridge Gateway [installation documentation](#), which contains instructions for various gateway models.

## Packet-forwarders

There are different packet-forwarder implementations. The packet-forwarder that is installed on your gateway depends on the gateway vendor and model. The packet-forwarders that are compatible with ChirpStack:

- [ChirpStack Concentratord](#)
- [Semtech UDP Packet Forwarder](#)
- [Semtech BasicStation](#)

## Configuration

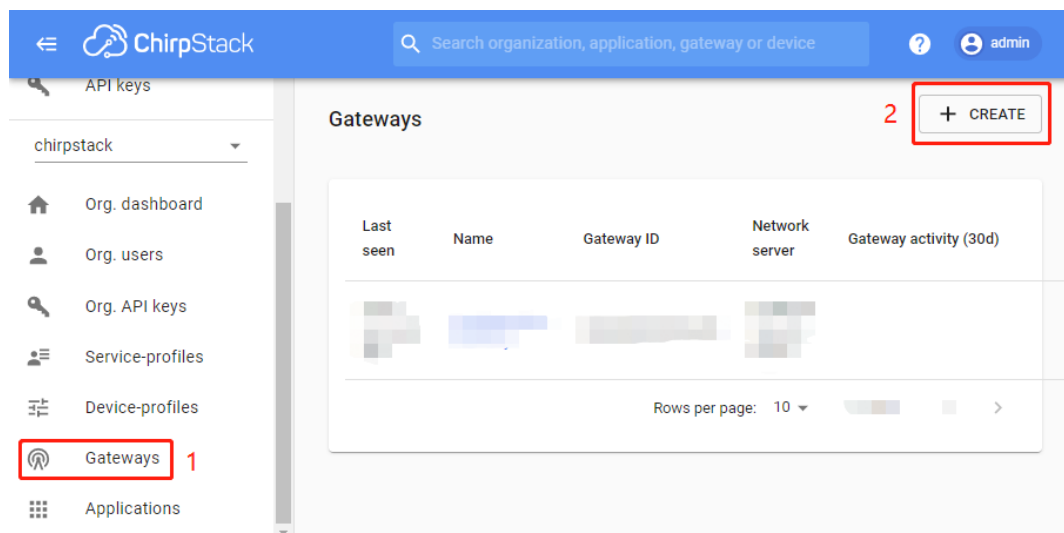
### Packet-forwarder

The packet-forwarder that is configured on your gateway must forward its data to the ChirpStack Gateway Bridge. As it controls the LoRa® chipset of the gateway, it also must be configured for the correct frequencies. A mismatch in frequencies means that the gateway will not receive uplinks sent by a device and / or is unable to send downlink payloads when the downlink frequency is outside the configured frequency range. Usually gateway vendors provide configuration examples for various bands. Please validate that the configuration matches the band and channels in the [ChirpStack Network Server Configuration](#).

### Add gateway

**Tip:** If you have not yet connected your [ChirpStack Application Server](#) instance with a [ChirpStack Network Server](#) instance, you must do this first. See [Optional: Adding a Network Server](#). Also you must connect the organization with the network-server by [Optional: Creating a Service-profile](#).

1. Go to **Gateways** in the web-interface, and click **+Create**.



2. Complete the form. Make sure that the **Gateway ID** field is equal to the Gateway ID of your gateway. If this value is incorrectly configured, data received by your gateway will be rejected. Then click **Create Gateway**.

Item	Description
Name	Headquarters-Gateway
Description	MTCAP-868-041A<
Gateway ID (EUI64)	<i>YOUR_GATEWAY_ID, eg:0080000000020e0b</i>
Network-server name	localhost network server
Service-profile name	localhost service profile

### 3. Show Gateways.

admin

chirpstack

Org. dashboard

Org. users

Org. API keys

Service-profiles

Device-profiles

Gateways

Applications

Gateways

+ CREATE

Last seen	Name	Gateway ID	Network server	Gateway activity (30d)
a few seconds ago	<a href="#">Headquarters-Gateway</a>	0080C...0e0b	localhost network server	

Rows per page: 10

1-1 of 1

< >

## Validate

There are a few ways to validate if your gateway is correctly configured.

### Last seen at

Event when no LoRa(WAN) data is received by the gateway, it will send gateway statistics periodically. Usually this stats interval is configured to 30 seconds. As ChirpStack Application Server will update the gateway **Last seen at** timestamp when it receives statistics, this is the easiest way to validate that the gateway is correctly configured.

**Note:** it might take a short while before statistics are sent by your gateway. You must refresh the page in order to see the (new) **Last seen at** value.

The screenshot shows the ChirpStack web interface. In the left sidebar, the 'Gateways' menu item is highlighted with a red box and labeled '1'. The main content area displays a table of gateways. The first gateway, 'Headquarters-Gateway', has a 'Last seen' column value of 'a few seconds ago', which is highlighted with a red box and labeled '2'. The gateway ID is '00800...20e0b' and the network server is 'localhost network server'.

## LoRaWAN frames

After opening the overview page of your gateway, you will see a **LoRaWAN frames** tab. This will show all LoRaWAN frames that are received and sent by your gateway. In case of received frames, this means that you will also see received frames from devices that are not yours and / or that are not yet configured. Therefore this screen is useful to validate if your gateway is able to receive LoRaWAN frames and forward these to ChirpStack.

The screenshot shows the ChirpStack web interface for a specific gateway. The 'LIVE LORAWAN FRAMES' tab is highlighted with a red box and labeled '1'. Below the tabs, there is a table of received frames. The first row of the table is highlighted with a red box and labeled '2'. The frame data includes timestamp, type, frequency, bandwidth, power, and device address.

Timestamp	Type	Freq	BW	Power	FCnt	DevAddr
Jun 15 2:24:14 PM	UnconfirmedDataUp	867.1 MHz	SF12	BW125	FCnt: 106	DevAddr: 001083bb
Jun 15 2:09:14 PM	UnconfirmedDataUp	867.5 MHz	SF12	BW125	FCnt: 105	DevAddr: 001083bb
Jun 15 1:57:04 PM	UnconfirmedDataUp	867.7 MHz	SF12	BW125	FCnt: 104	DevAddr: 001083bb
Jun 15 1:42:04 PM	UnconfirmedDataUp	868.1 MHz	SF12	BW125	FCnt: 103	DevAddr: 001083bb
Jun 15 1:27:04 PM	UnconfirmedDataUp	867.1 MHz	SF12	BW125	FCnt: 102	DevAddr: 001083bb

## Troubleshooting

See [Troubleshooting gateway](#).

### 6.4.3 Connecting a device

Refer to [Connecting device](#).

This guide describes how to connect your LoRaWAN device with ChirpStack and how to validate that it can successfully activate. At this point it is expected that you have the ChirpStack Network Server and ChirpStack Application Server components installed and that you have successfully connected a LoRa gateway to it.

## Requirements

Before continuing, there are a couple things you need to know about your device. This information is usually provided by the device vendor.

- DevEUI
- LoRaWAN MAC version implemented by the device
- Regional Parameters revision implemented by the device
- OTAA: Device root-keys (when no external join-server is used)

## Login

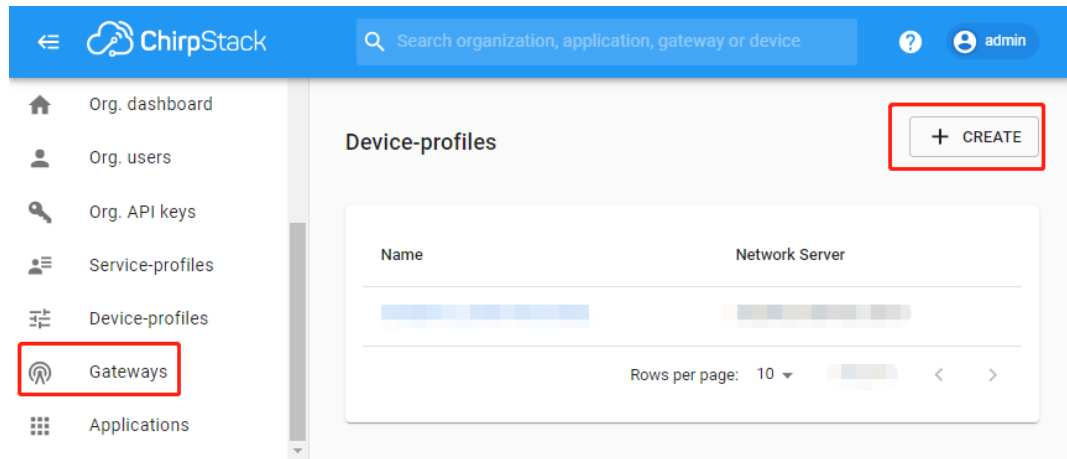
Login into the ChirpStack Application Server web-interface. The default credentials are:

- Username: admin
- Password: admin

## Optional: Creating a Device profile

Before you can add the device to ChirpStack, you have to create a [Device-profile](#) if you haven't done this already. In general it is a good practice to create separate device-profiles for different types of devices. A device-profile contains the capabilities of your device. For example if it uses ABP or OTAA for activation, which LoRaWAN version and Regional Parameters revision is implemented by the device, etc... It can also be configured with a function to decode the payloads sent by the devices using the device-profile.

1. Within the ChirpStack Application Server web-interface, click **Gateways** and then **Create**.



2. Under the **General** tab, fill in the required fields.

Item	Description
Name	TA692FC-L-5-868 Thermostat
Network-server	localhost network server
Region	EU868
LoRaWAN MAC version	LoRaWAN 1.0.3
LoRaWAN Regional parameters revision	A
ADR algorithm	Default ADR algorithm (LoRa only)
Uplink interval (seconds)	1000
Device-status request frequency (req/day)	1

**Device-profiles / Create**

**GENERAL** | JOIN (OTAA / ABP) | CLASS-B | CLASS-C | CODEC

- Device-profile name \*  
TA692FC-L-5-868 Thermostat  
A name to identify the device-profile.
- Network-server \*  
localhost network server  
The network-server on which this device-profile will be provisioned. After creating the device-profile, this value can't be changed.
- LoRaWAN MAC version \*  
1.0.3  
The LoRaWAN MAC version supported by the device.
- LoRaWAN Regional Parameters revision \*  
A  
Revision of the Regional Parameters specification supported by the device.
- ADR algorithm \*  
Default ADR algorithm (LoRa only)  
The ADR algorithm that will be used for controlling the device data-rate.
- Uplink interval (seconds) \*  
1000  
The expected interval in seconds in which the device sends uplink messages. This is used to determine if a device is active or inactive.

Max EIRP \*  
0  
Maximum EIRP supported by the device.

[CREATE DEVICE-PROFILE](#)

3. Under the **Join (OTTA/ABP)** tab, fill in the required fields.

Item	Description
Join (OTAA / ABP)	yes, Device supports OTAA

**Device-profiles / Create**

GENERAL | **JOIN (OTAA / ABP)** | CLASS-B | CLASS-C | CODEC

- ☒ Device supports OTAA

[CREATE DEVICE-PROFILE](#)

4. Under the **Class-B** tab, fill in the required fields.

Item	Description
Supports Class-B	no

ChirpStack

Search organization, application, gateway or device

admin

Org. dashboard  
Org. users  
Org. API keys  
Service-profiles  
Device-profiles  
Gateways  
Applications

Device-profiles / Create

GENERAL JOIN (OTAA / ABP) **CLASS-B** CLASS-C CODEC

☐ Device supports Class-B

CREATE DEVICE-PROFILE

5. Under the **Class-C** tab, fill in the required fields.

Item	Description
Supports Class-C	yes
Class-C confirmed downlink timeout (seconds)	300

ChirpStack

Search organization, application, gateway or device

admin

All users  
API keys  
chirpstack

Org. dashboard  
Org. users  
Org. API keys  
Service-profiles  
Device-profiles  
Gateways  
Applications

Device-profiles / Create

GENERAL JOIN (OTAA / ABP) CLASS-B **CLASS-C** CODEC

1 ☒ Device supports Class-C  
Select this option when the device will operate as Class-C device immediately after activation. In case it sends a DeviceModelInd mac-command when it changes to Class-C, do not select this option.

2 Class-C confirmed downlink timeout \*  
300  
Class-C timeout (in seconds) for confirmed downlink transmissions.

CREATE DEVICE-PROFILE

6. Under the **Codec** tab, fill in the required fields. Then click **create device-profile**.

**Note:** The data here may not be decoded. It's here just for debugging convenience.

Item	Description
Payload codec	Custom JavaScript codec functions

```
// Decode decodes an array of bytes into an object.
// - fPort contains the LoRaWAN fPort number
// - bytes is an array of bytes, e.g. [225, 230, 255, 0]
// - variables contains the device variables e.g. {"calibration": "3.5"}
// (both the key / value are of type string)
// The function must return an object, e.g. {"temperature": 22.5}
function Decode(fPort, bytes, variables) {
    var dataX = {};
```

(continues on next page)



(continued from previous page)

```

var fanModeStateMeta = {
0: "OFF",
1: "LOW",
2: "MED",
3: "HIGH",
4: "AUTO"
};
var systemModeMeta = {
0: "OFF",
1: "COOL",
2: "FAN-ONLY"
};
if(fPort==10){
dataX.roomTemperature = ((bytes[0] << 8) + bytes[1])/10;
dataX.setTemperature = ((bytes[2] << 8) + bytes[3])/10;
dataX.coolProportionalOutput = bytes[4]/100;
dataX.fanMode = fanModeStateMeta[bytes[5]];
dataX.fanState = fanModeStateMeta[bytes[6]];
dataX.threshold = bytes[7]/10;
dataX.systemMode = systemModeMeta[bytes[8]];
dataX.coolPBand = bytes[9]/10;
dataX.coolItime = (bytes[10] << 8) + bytes[11];
dataX.kFactor = bytes[12];
return {
  data: {
    roomTemperature: dataX.roomTemperature,
    setTemperature: dataX.setTemperature,
    coolProportionalOutput: dataX.coolProportionalOutput,
    fanMode: dataX.fanMode,
    fanState: dataX.fanState,
    threshold: dataX.threshold,
    systemMode: dataX.systemMode,
    coolPBand: dataX.coolPBand,
    coolItime: dataX.coolItime,
    kFactor: dataX.kFactor
  }
};
}
}

```

```

// Encode encodes the given object into an array of bytes.
// - fPort contains the LoRaWAN fPort number
// - obj is an object, e.g. {"temperature": 22.5}
// - variables contains the device variables e.g. {"calibration": "3.5"}
// (both the key / value are of type string)
// The function must return an array of bytes, e.g. [225, 230, 255, 0]
function Encode(fPort, obj, variables) {
return [];
}

```

**Device-profiles / Create**

GENERAL JOIN (OTAA / ABP) CLASS-B CLASS-C **CODEC** TAGS

1 **Custom JavaScript codec functions**

2

```
1 // Decode decodes an array of bytes into an object.
2 // - fPort contains the LoRaWAN fPort number
3 // - bytes is an array of bytes, e.g. [225, 230, 255, 0]
4 // - variables contains the device variables e.g. {"calibration": "3.5"} (both the key / value are of type string)
5 // The function must return an object, e.g. {"temperature": 22.5}
6 function Decode(fPort, bytes, variables) {
7   return {};
8 }
```

The function must have the signature `function Decode(fPort, bytes)` and must return an object. ChirpStack Application Server will convert this object to JSON.

3

```
1 // Encode encodes the given object into an array of bytes.
2 // - fPort contains the LoRaWAN fPort number
3 // - obj is an object, e.g. {"temperature": 22.5}
4 // - variables contains the device variables e.g. {"calibration": "3.5"} (both the key / value are of type string)
5 // The function must return an array of bytes, e.g. [225, 230, 255, 0]
6 function Encode(fPort, obj, variables) {
7   return [];
8 }
```

The function must have the signature `function Encode(fPort, obj)` and must return an array of bytes.

4 **CREATE DEVICE-PROFILE**

### 7. Show Device profiles.

**Device-profiles** **+ CREATE**

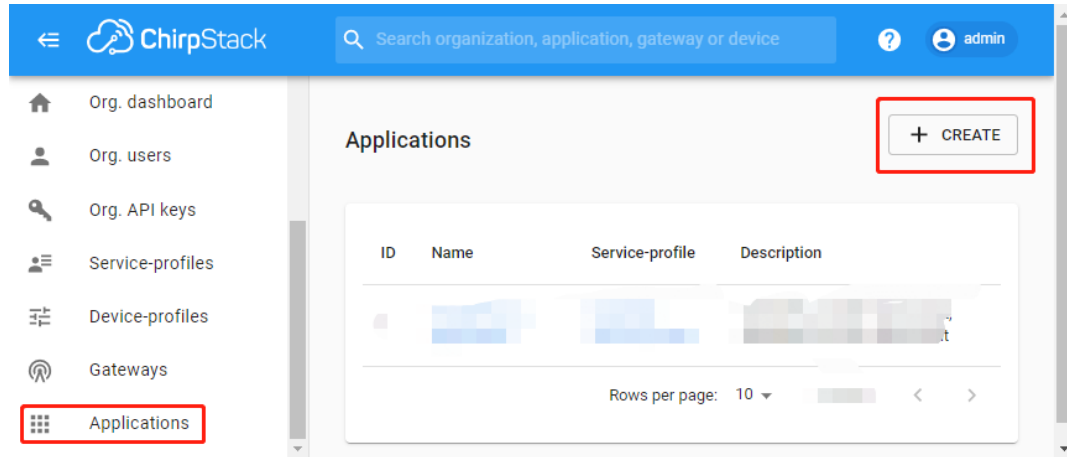
Name	Network Server
TA692FC-L-5-868 Thermostat	localhost network server

Rows per page: 10 1-1 of 1

### Optional: Adding an Application

Devices are grouped by applications. For example you could group your temperature sensors under one application and weather stations under an other application.

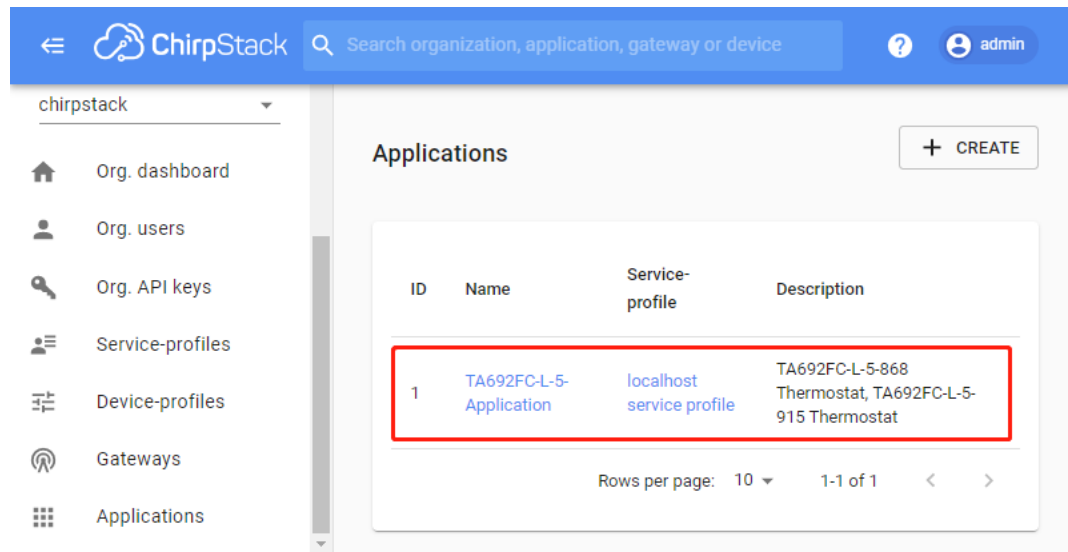
1. If you haven't created an application yet to which you want to add the device, click **Applications**, then click **Create**.



2. Fill in the required fields and **Create Application**.

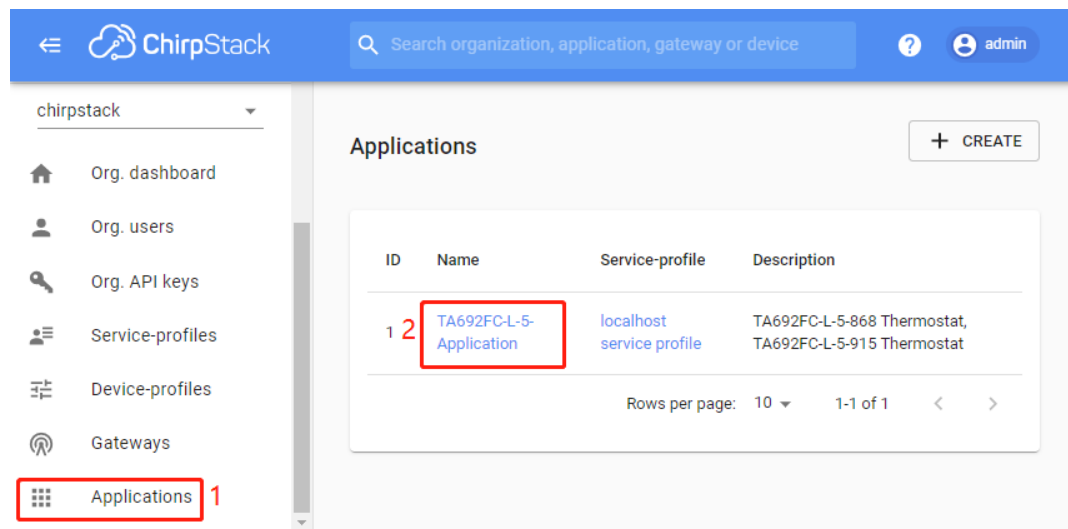
Item	Description
Name	TA692FC-L-5-Application
Description	TA692FC-L-5-868 Thermostat, TA692FC-L-5-915 Thermostat
Service-profile name	localhost service profile

3. Show Applications.

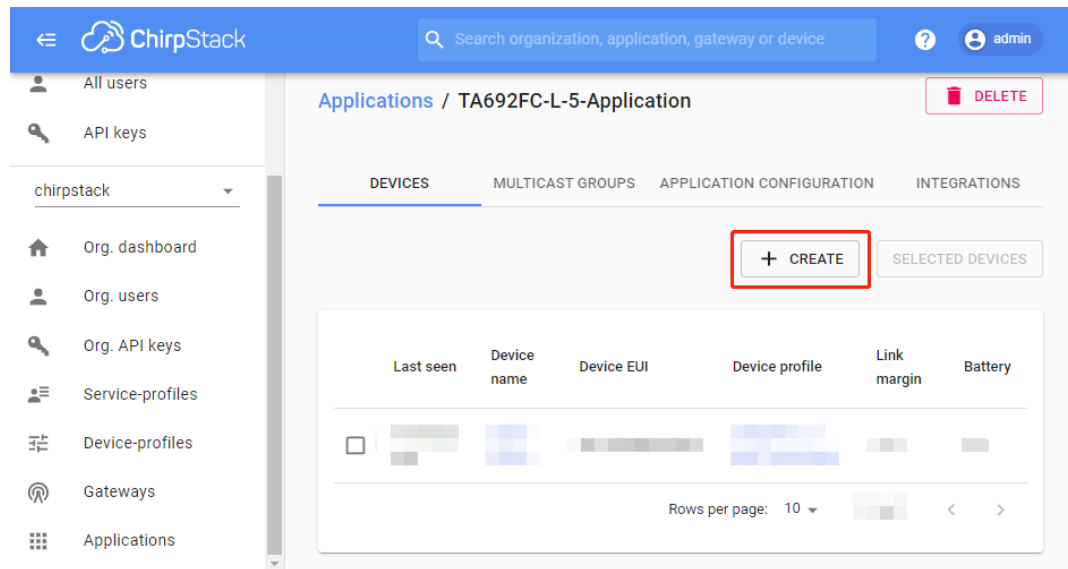


## Creating a device

1. Click the (newly created) **application** to which you want to add your device.



2. Under the **Devices** tab, click **Create**.



3. Fill in the required fields and select the device-profile that you want to associate with your device and save the device.

Item	Description
Name	Sales-Office
Description	TA692FC-L-5-868 device
Device EUI (EUI64)	<i>YOUR_DEVICE_EUI, eg:0012bdfffe02ad04</i>
Device profile	TA692FC-L-5-868 Thermostat

ChirpStack

Search organization, application, gateway or device

admin

Applications / TA692FC-L-5-Application / Devices / Create

GENERAL VARIABLES TAGS

1 Device name \*  
Sales-Office

The name may only contain words, numbers and dashes.

2 Device description \*  
TA692FC-L-5-868 device

3 Device EUI \*  
00 12 bd ad 04

MSB

4 Device-profile \*  
TA692FC-L-5-868 Thermostat

☐ Disable frame-counter validation

Note that disabling the frame-counter validation will compromise security as it enables people to perform replay-attacks.

☐ Device is disabled

ChirpStack Network Server will ignore received uplink frames and join-requests from disabled devices.

5 CREATE DEVICE

4. Depending the device-profile is configured for OTAA or ABP, the next page will ask you to enter the device root-keys (OTAA) or device session-keys (ABP).

In case your ChirpStack Network Server is configured with a join-server and your (OTAA) device will use this join-server for activation, then there is no need to enter the root-keys.

Item	Description
Application key	<i>YOUR_DEVICE_EUI, eg:72357538782F413F4428472B4B625065</i>

ChirpStack

Search organization, application, gateway or device

admin

Applications / TA692FC-L-5-Application / Devices / Sales-Office

DELETE

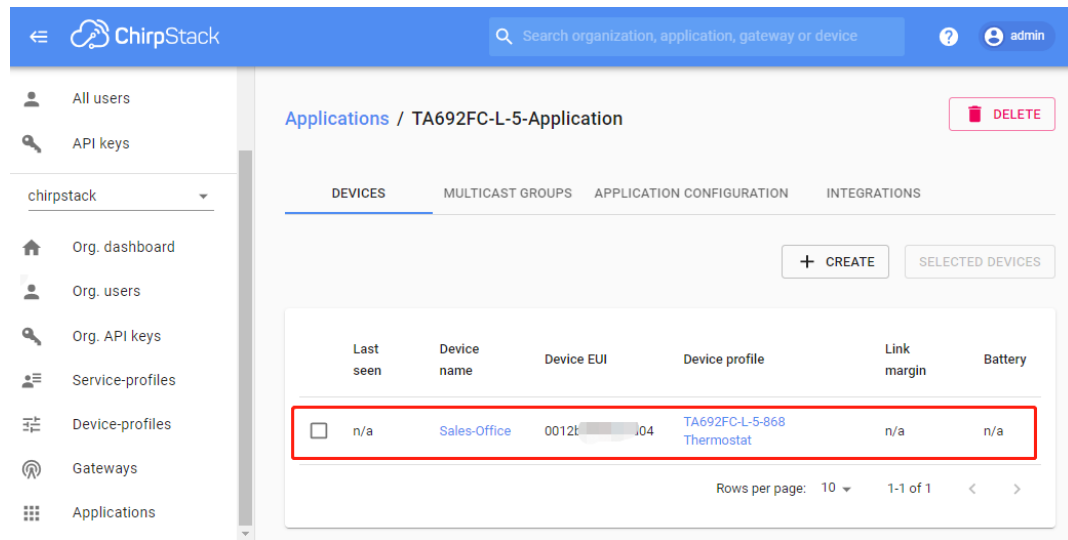
DETAILS CONFIGURATION KEYS (OTAA) ACTIVATION DEVICE DATA LORAW >

1 Application key \*  
72 35 75 38 78 2F 41 2B 4B 62 50 65

MSB

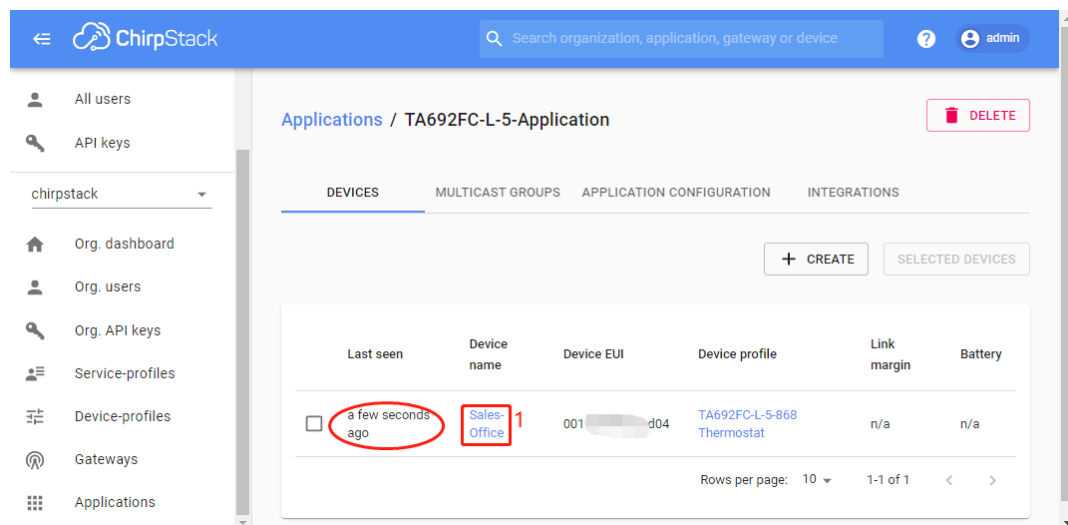
2 SET DEVICE-KEYS

5. Show **Devices**.

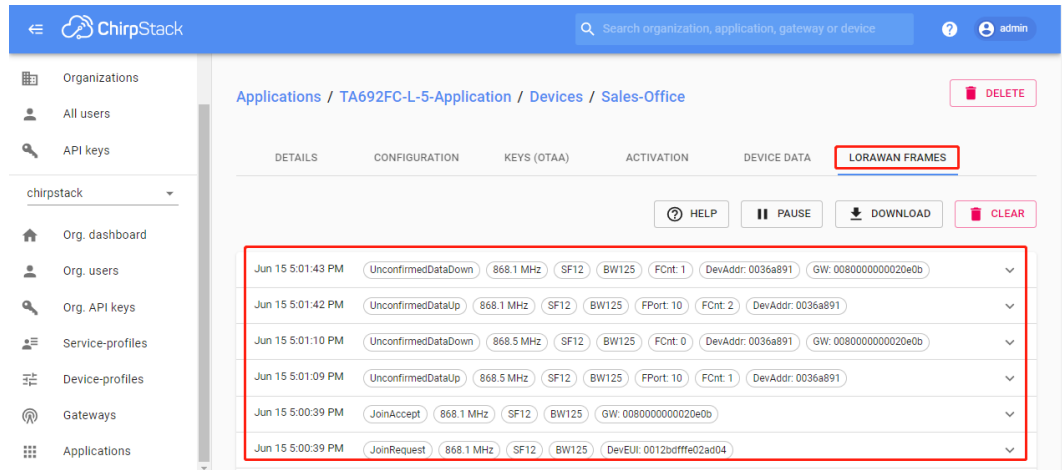


## Validate

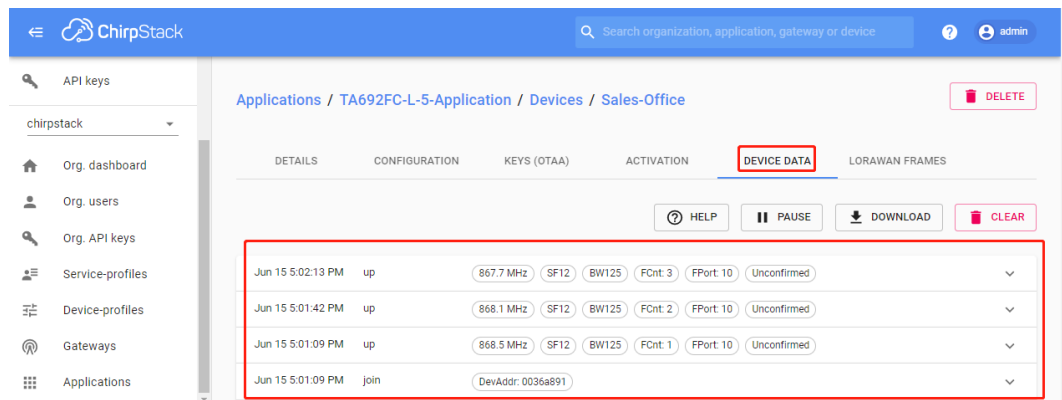
1. After adding your LoRaWAN device to ChirpStack, validate that your device is able activate (in case of OTAA) and send data. Clicking the device in the ChirpStack Application Server web-interface.



2. Open in one window the **Device data** and in an other window the **LoRaWAN frames** tab. Then turn on your device or trigger an uplink transmission. In case of an OTAA device you should first see a JoinRequest followed by a JoinAccept message in the **LoRaWAN frames** tab.



3. When the device sends its first data payload, you should also see a Join and Up event in the **Device data** tab.



## Troubleshooting

See [Troubleshooting device](#).

## 6.5 TA692FC-L-5-868 Thermostat – Demo device profile usage

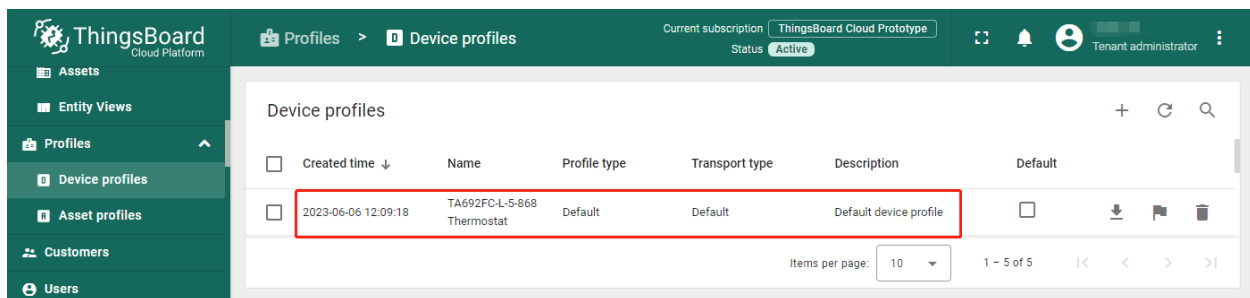
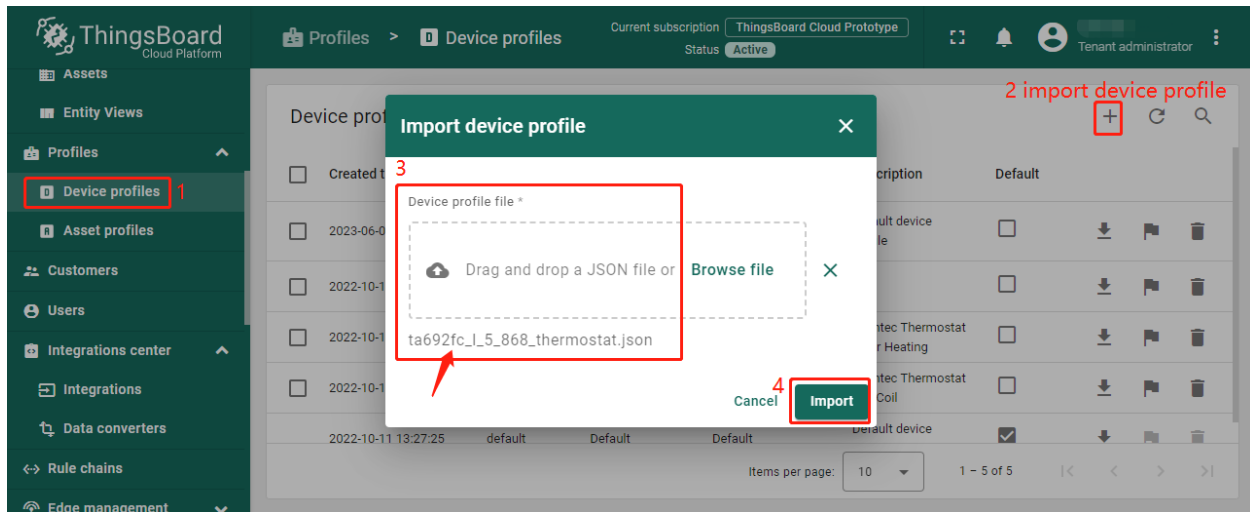
### 6.5.1 Import device profile

**Tip:** A *Device Profile* file can only be imported once. If you have already imported it, you do not need and cannot repeat the import.

If you have already imported it, you can skip this step.

- Download `ta692fc_l_5_868_thermostat.json`.
- **Profiles** → **Device profiles** → + → **Popup dialog: Import device profile** → Drag and drop *my device profile File* → **Import**.

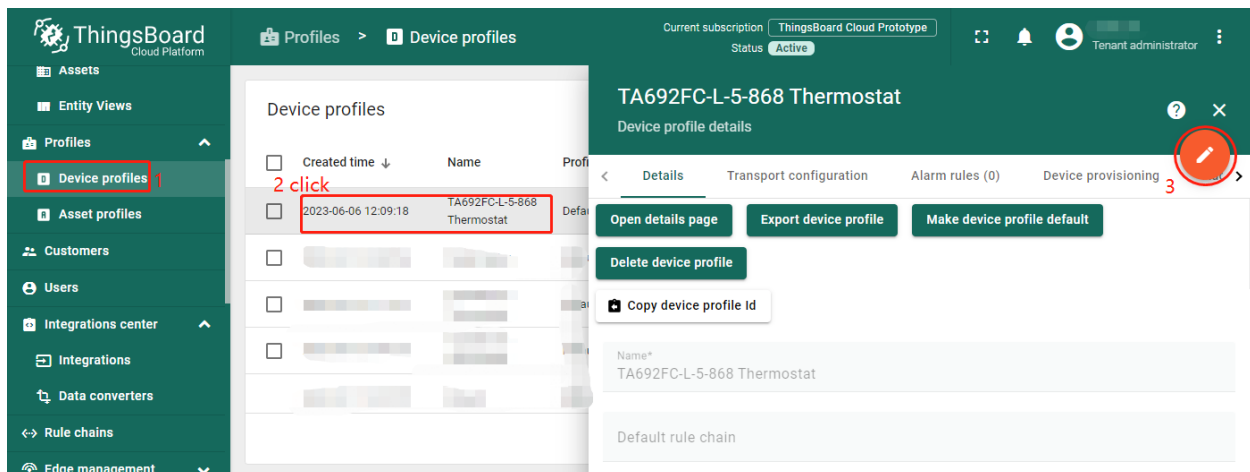




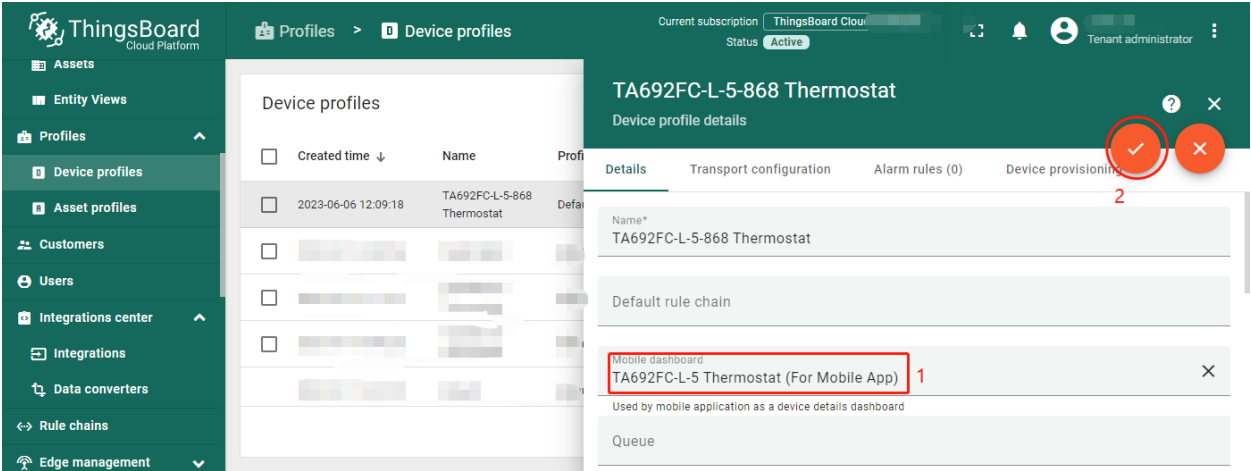
### 6.5.2 Modify device profile's mobile dashboard

Device profile's mobile dashboard is for ThingsBoard Mobile Application or ThingsBoard PE Mobile Application.

- Profiles → Device profiles → click my device profile → Toggle edit mode (red icon)



- Modify Mobile dashboard → Apply changes (red icon)



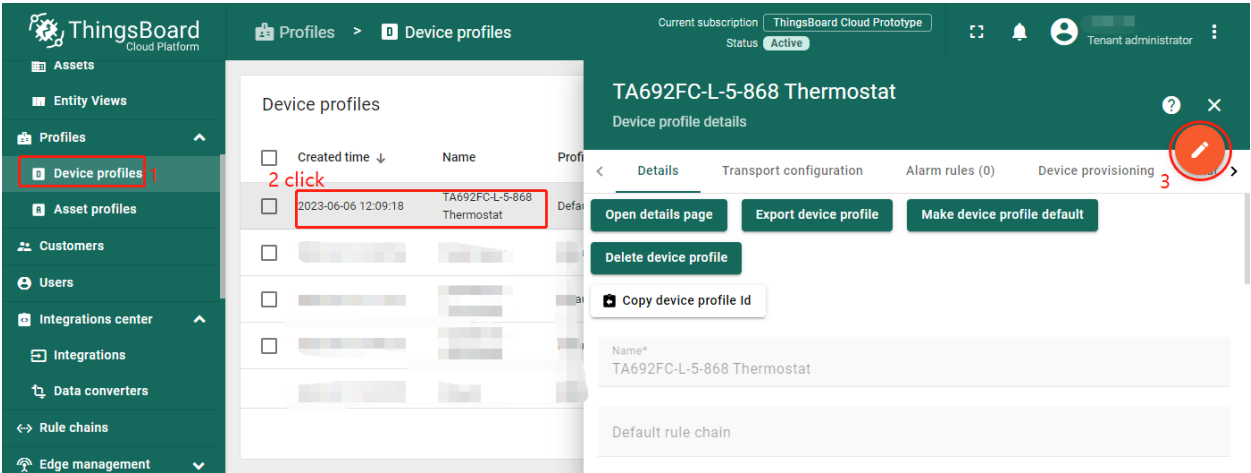
These values are shown in the following table:

Field	Value
Mobile dashboard	TA692FC-L-5 Thermostat (For Mobile App)

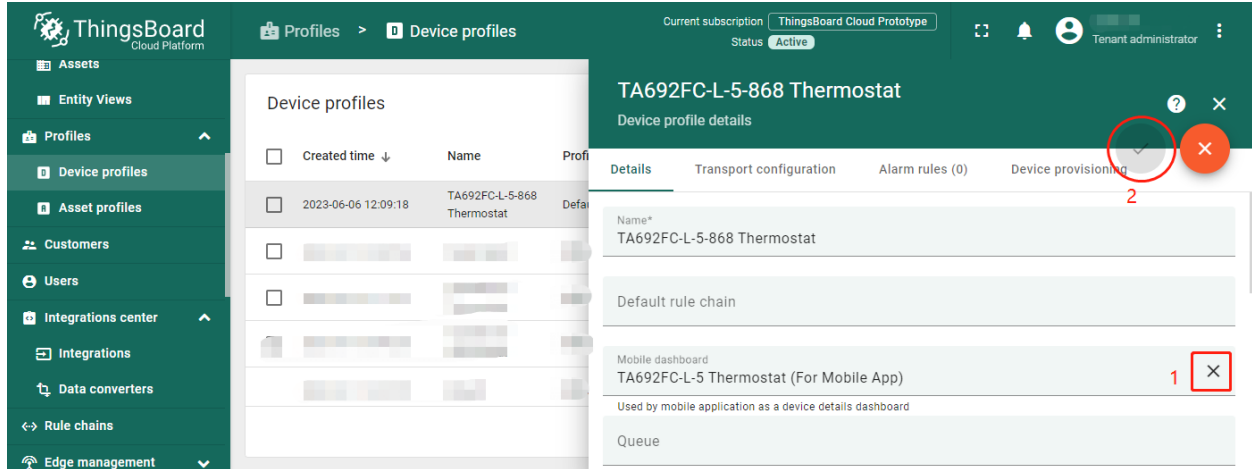
6.5.3 Clear device profile’s mobile dashboard

Sometimes if TA692FC-L-5-868 Thermostat device profile’s mobile dashboard is cleared, TA692FC-L-5 Thermostat (For Mobile App) can only be deleted.

- Profiles -> Device profiles -> click my device profile -> Toggle edit mode (red icon)



- Clear Mobile dashboard -> Apply changes (red icon)



## 6.6 TA692FC-L-5 Demo Dashboards Usage

### 6.6.1 Overview

There are two dashboards related to TA692FC-L-5, namely TA692FC-L-5 Thermostat List and TA692FC-L-5 Thermostat (For Mobile App). We open the former to start operating TA692FC-L-5.

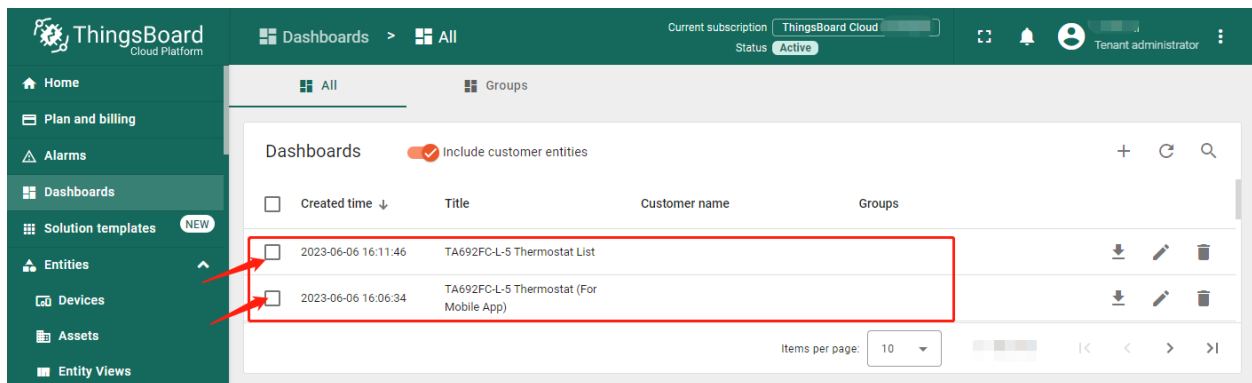


Table 2: TA692FC-L-5 Demo Dashboards

Dashboard	Description	For Web UI	For Mobile App	Entry*
TA692FC-L-5 Thermostat List	list	Yes	No	Yes
TA692FC-L-5 Thermostat (For Mobile App)	details	Yes	Yes	No

#### Hint:

- If *Entry* is *Yes*, then directly enter the Dashboard and there will be data displayed.
- If *Entry* is *No*, there will be no data display when entering this Dashboard directly, and you need to jump to this Dashboard from other Dashboards.

## 6.6.2 TA692FC-L-5 Thermostat List

### Dashboard states

#### Default state

Default state is root state.

ThingsBoard Cloud Platform

Dashboards > All > TA692FC-L-5 Thermostat List

Current subscription: ThingsBoard Cloud, Status: Active

Tenant administrator

TA692FC-L-5 Thermostat List

Entities

Realtime - last 5 minutes

TA692FC-L-5 Thermostats

Device name ↑	Label	Type	Active	Room Temperature(°C)	Set Temperature(°C)	System Mode	Fan State
0012bdfffe02ad04	Sales-Office	TA692FC-L-5-868 Thermostat	true	23	19.5	COOL	HIGH

Items per page: 10, 1 - 1 of 1

- **Dashboard bar:**

- **TA692FC-L-5 Thermostat List** : Click here to skip to **root state**. Since **default state** is *root state*, click here and there is no response.
- : Click the two ICONS in the upper left corner to display the page in full screen.
- **Realtime - last 5 minutes** : Edit time window.

- **Thermostats widgets:**

- **Fields:**

- \* Device name, Label, Type, active.
- \* Room temperature, Set Temperature, System Mode, Fan status: Refer to *Monitor state*.

- **Actions:**

- \* : skip to *TA692FC-L-5 Thermostat (For Mobile App)*.
- \* : Popup dialog to editing a device's label.

## Import List Dashboard

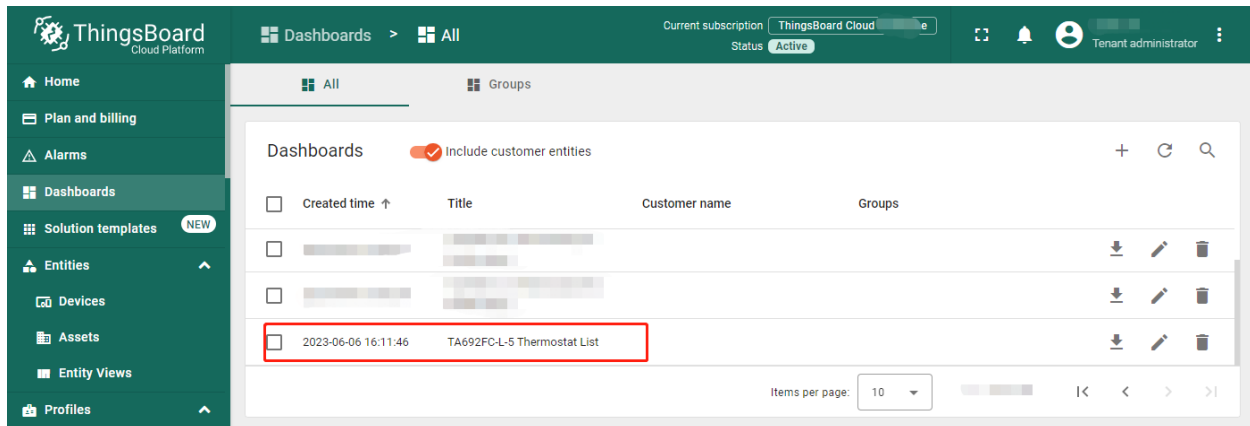
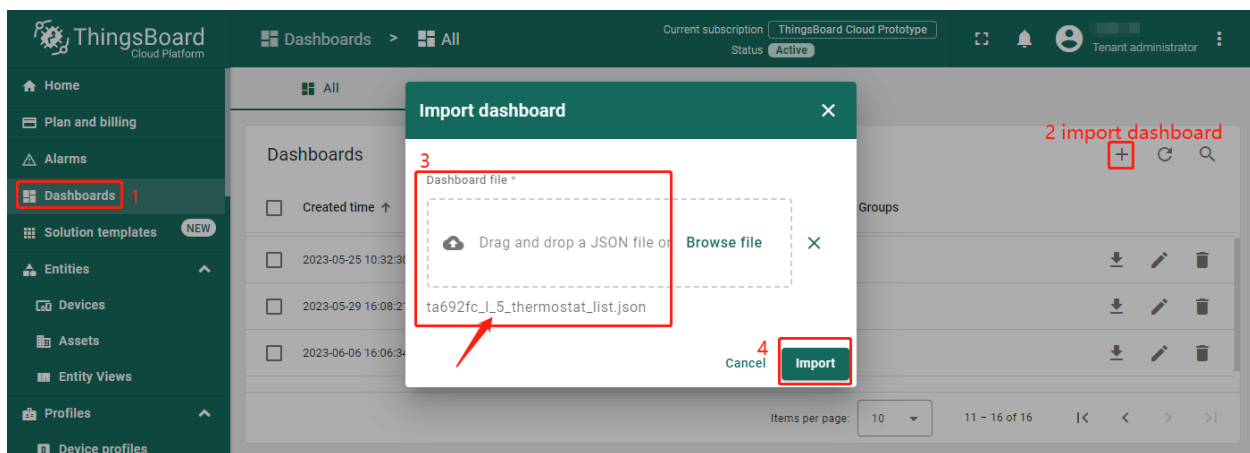
**Tip:** A *Dashboard file* can only be imported once. If you have already imported it, you do not need and cannot repeat the import.

If you have already imported it, you can skip this step.

In order to use this dashboard, you must to create TA692FC-L-5 Thermostat Device Profile and TA692FC-L-5 Thermostat (For Mobile App). If they don't exist, you can import them. See [Import Device Profile of TA692FC-L-5 Thermostat](#) or [Import TA692FC-L-5 Detail Dashboard](#).

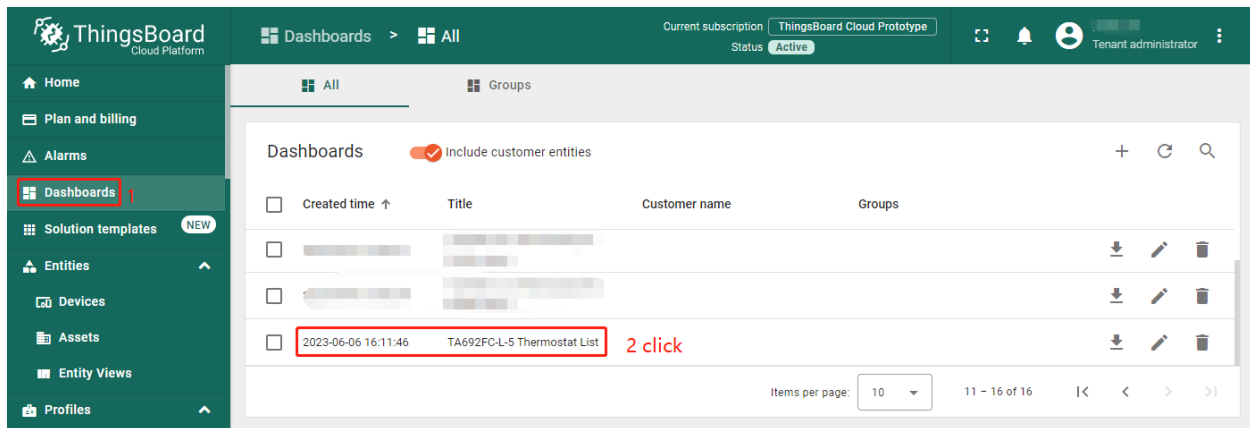
First, you can import this dashboard.

- Download `ta692fc_l_5_thermostat_list.json`.
- **Dashboards** → + → **Popup dialog: Import dashboard** → Drag and drop *list dashboard File* → **Import**.

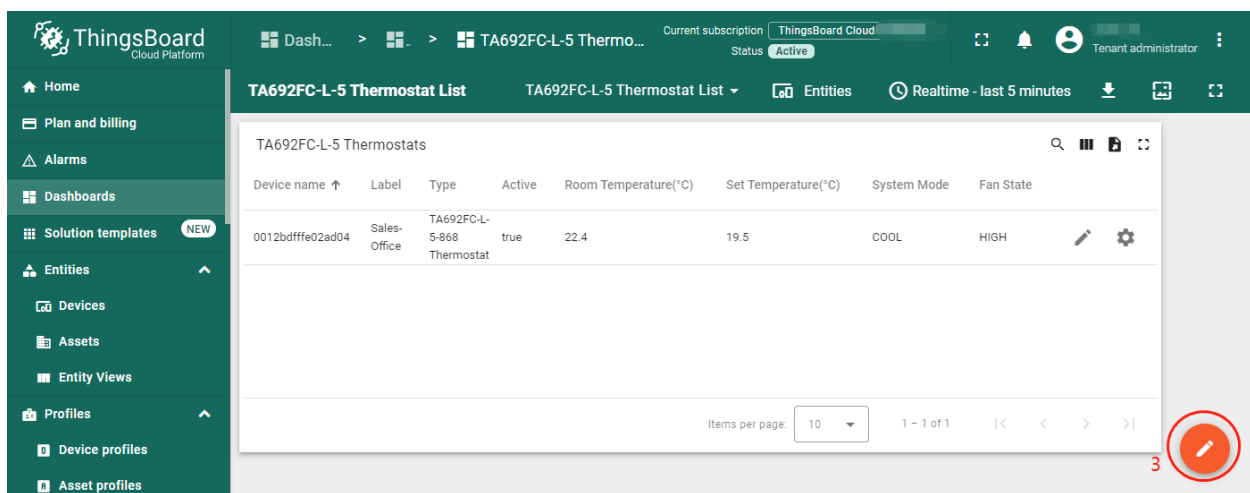


Next, modify a action's target dashboard and target dashboard state.

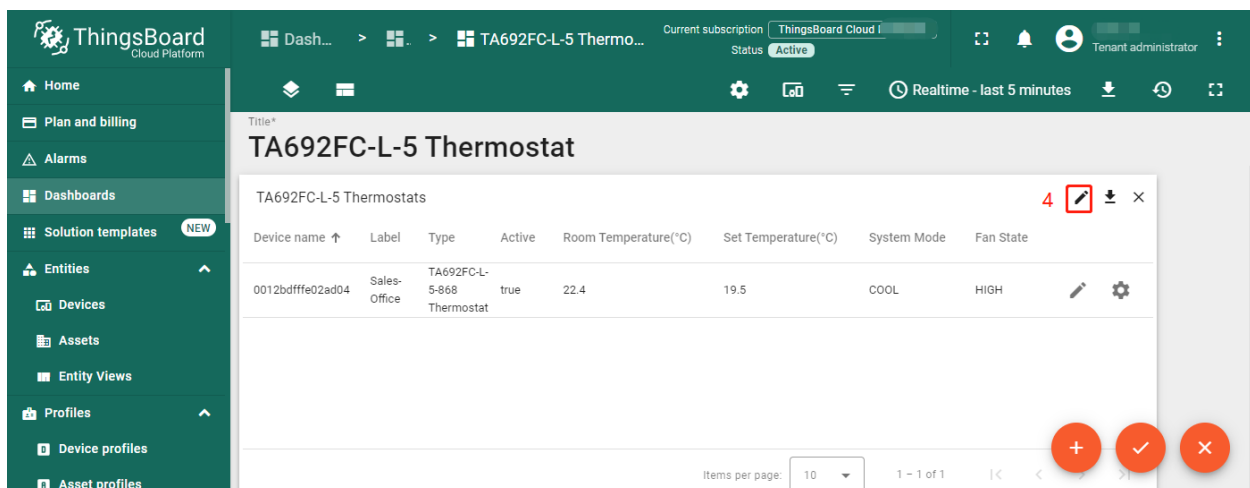
- **Dashboards** → Click *my list dashboard*



- **Edit** (red icon on the bottom and right)



- Enter *Edit Dashboard Mode* → **Edit Widget** (icon)



- **Action** → **Edit Action** (icon)

ThingsBoard Cloud Platform

Current subscription: ThingsBoard Cloud Prototype, Status: Active

Tenant administrator

Realtime - last 5 minutes

TA692FC-L-5 Thermostats

Entities table

Device name ↑ Label Type

0012bdf0e02ad04 Sales-Office TA692FC-L-5-868 Thermostat

Actions

Action source ↑	Name	Icon	Type
⌵	Action cell button	✎	Custom action (with HTML template)
⌵	Action cell button	⚙️	Navigate to other dashboard

Items per page: 10 1 - 2 of 2

- Modify Target dashboard → modify Target dashboard state → Save

ThingsBoard Cloud Platform

Current subscription: ThingsBoard Cloud Prototype, Status: Active

Tenant administrator

Realtime - last 5 minutes

TA692FC-L-5 Thermostats

Edit action

Type\*  
Navigate to other dashboard

Target dashboard \*

Dashboard\*  
TA692FC-L-5 Thermostat (For Mobile App)

monitor

☐ Open in a new browser tab

☒ Set entity from widget

State entity parameter name  
By default

Cancel Save

Items per page: 10 1 - 2 of 2

These values are shown in the following table:

Field	Value
Target dashboard	TA692FC-L-5 Thermostat (For Mobile App)
Target dashboard state	monitor

- Apply changes (red icon)

ThingsBoard Cloud Platform

Current subscription: ThingsBoard Cloud Prototype, Status: Active

Tenant administrator

Realtime - last 5 minutes

TA692FC-L-5 Thermostats

Entities table

10

Device name	Label	Type
0012bdffe02ad04	Sales-Office	TA692FC-L-5-868 Thermostat

Actions

Action source	Name	Icon	Type
Action cell button	Edit thermostat		Custom action (with HTML template)
Action cell button	Thermostat detail		Navigate to other dashboard

Items per page: 10, 1 - 2 of 2

- **Apply changes** (red icon on the bottom and right)

ThingsBoard Cloud Platform

Current subscription: ThingsBoard Cloud, Status: Active

Tenant administrator

Realtime - last 5 minutes

TA692FC-L-5 Thermostat

Device name	Label	Type	Active	Room Temperature(°C)	Set Temperature(°C)	System Mode	Fan State
0012bdffe02ad04	Sales-Office	TA692FC-L-5-868 Thermostat	true	22.3	19.5	COOL	HIGH

Items per page: 10, 1 - 1 of 1

11

### Update List Dashboard

- First, delete this dashboard: **Dashboards** → Click in the row of TA692FC-L-5 Thermostat List → **Popup dialog: Are you sure you want to delete ...? → Yes.**

ThingsBoard Cloud Platform

Current subscription: ThingsBoard Cloud Prototype, Status: Active

Tenant administrator

Are you sure you want to delete the dashboard 'TA692FC-L-5 Thermostat List'?

Be careful, after the confirmation the dashboard and all related data will become unrecoverable.

No 3 Yes

2023-06-06 16:11:46 TA692FC-L-5 Thermostat List

Items per page: 10, 11 - 16 of 16



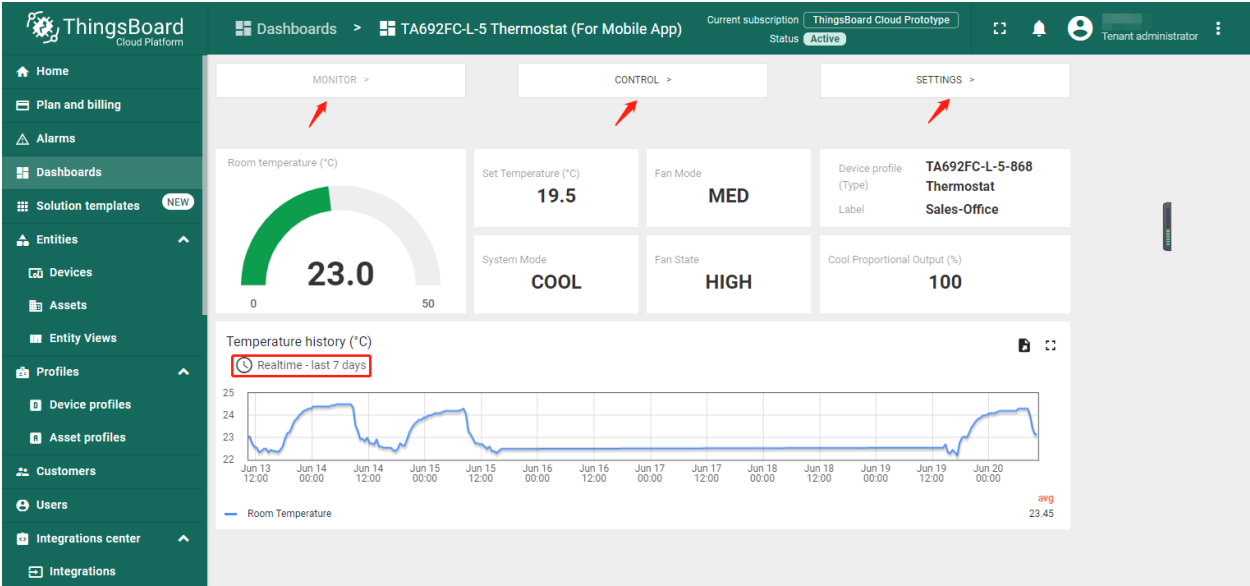
- Next, *import TA692FC-L-5 List Dashboard*.

6.6.3 TA692FC-L-5 Thermostat (For Mobile App)

Dashboard states

Monitor state

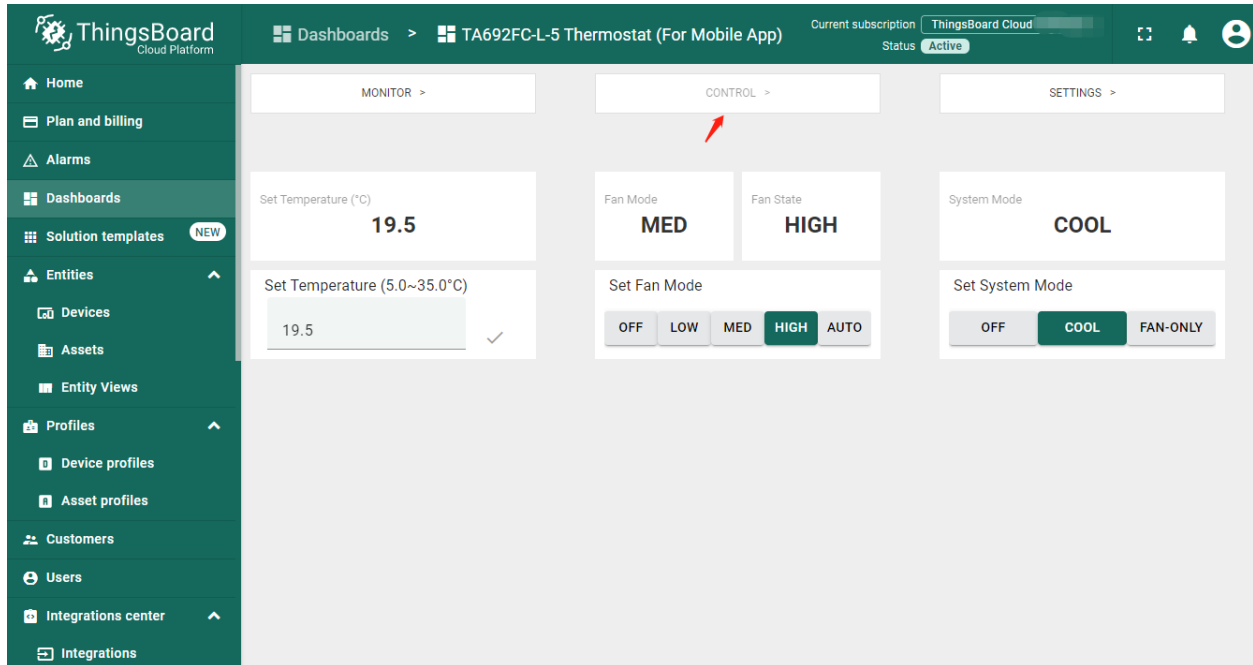
Monitor state is root state.



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Widget	Description
MONITOR	skip to <i>Monitor state</i>
CONTROL	skip to <i>Control state</i>
SETTINGS	skip to <i>Settings state</i>
Room Temperature	room temperature
Set Temperature	current setpoint value
System Mode	“OFF”, “OFF”, “COOL” or “FAN-ONLY”
Fan Mode	“OFF”, “LOW”, “MED”, “HIGH” or “AUTO”
Fan Status	“OFF”, “LOW”, “MED” or “HIGH”
Cool Proportional Output	0 ~ 100%
Temperature history	Room temperature history. Click ⌚ Realtime - last 7 days to edit this timewindow. Refer to <i>Default state</i>

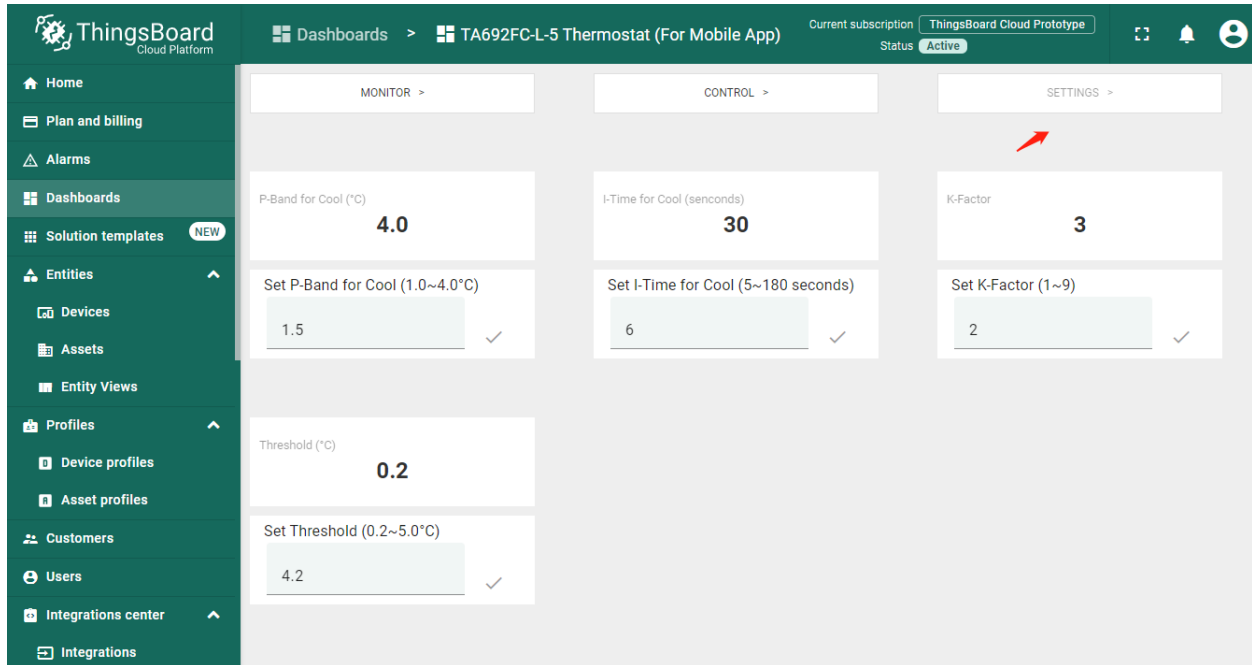
## Control state



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Widget	Description
Set Temperature	current setpoint value
Set Fan Mode	change fan mode
Set System Mode	change system mode

## Settings state



- **Dashboard bar:**  
Hidden. Refer to *Default state*.
- **Widgets:**

Widget	Description
Set P-Band for Cool	1.0 ~ 4.0 °C
Set I-Time for Cool	5 ~ 180 seconds
Set K-Factor	1 ~ 9
Set Threshold	0.2 ~ 5.0 °C

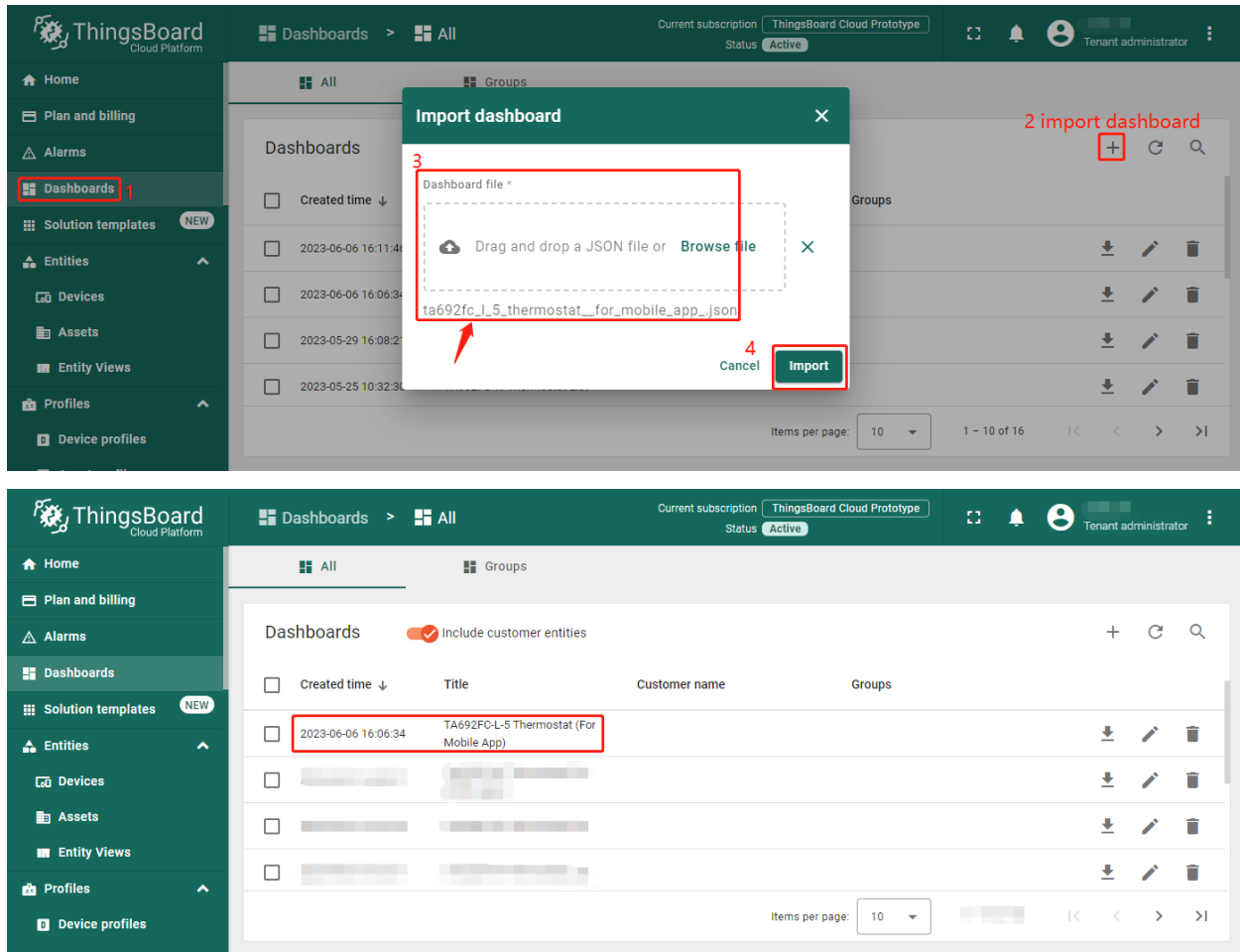
## Import Detail Dashboard

**Tip:** A *Dashboard file* can only be imported once. If you have already imported it, you don't need and cannot repeat the import.

If you have already imported it, you can skip this step.

In order to use this dashboard, you must create `ta692fc-l-5 Thermostat Device Profile`. If it doesn't exist, you can import it. See *Import Device Profile of ta692fc-l-5 Thermostat*.

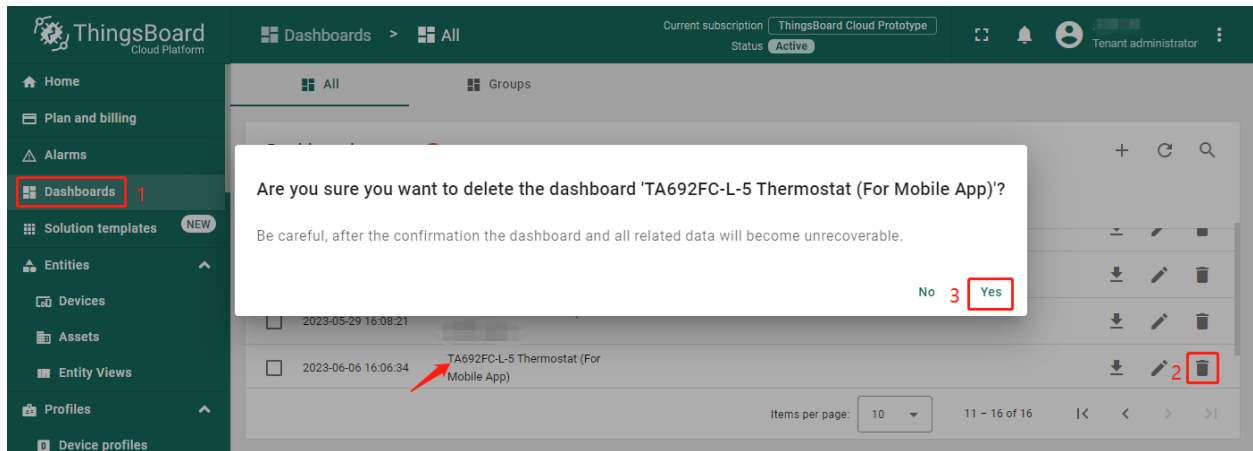
- Download `ta692fc_l_5_thermostat__for_mobile_app_.json`.
- **Dashboards** → **+** → **Popup dialog: Import dashboard** → Drag and drop *detail dashboard File* → **Import**.



- Optional, This dashboard can be set as ta692fc-l-5 Thermostat Device Profile's mobile dashboard. See *Modify ta692fc-l-5 Thermostat device profile's mobile dashboard*.

### Update Detail Dashboard

- First, *clear TA692FC-L-5-868 Thermostat device profile's mobile dashboard*.
- Next, delete this dashboard: **Dashboards** → Click  in the row of TA692FC-L-5 Thermostat (For Mobile App) → **Popup dialog: Are you sure you want to delete ...? → Yes**.



- Then *import TA692FC-L-5 Detail Dashboard*.

## 6.7 TA692FC-L-5 LoRaWAN API

### Tip:

- This section applies to TA692FC-L-5.

### 6.7.1 Overview

### 6.7.2 Payload format in LoRA packet used by TA692FC-L-5

#### Uplink | port 10

Byte	Data	Content	Range
0	data.RoomTemperature (High Byte)	Room Temperature(°C) = D_Room_Temperature/10	0 ~ 400
1	data.RoomTemperature (Low Byte)		
2	data.SetTemperature (High Byte)	Set Temperature(°C) = D_Set_Temperature/10	50 ~ 350
3	data.SetTemperature (Low Byte)		
4	data.CoolProportionalOutput	Cool Proportional Output : 0-100%	0 ~ 100
5	data.FanMode	0:OFF 1:LOW 2:MED 3:HIGH 4: AUTO	0 ~ 4
6	data.FanState	0:OFF 1:LOW 2:MED 3:HIGH	0 ~ 3
7	data.threshold (*)	Temperature change: 0.2°C ~ 5.0°C	2 ~ 50
8	data.SystemMode	0:OFF 1:COOL 2:FAN-ONLY	0 ~ 2
9	Data.CoolPBand	P-Band for Cool : 1.0°C ~ 4.0°C	10 ~ 40
10	Data.CoolItime (High Byte)	I-Time for Cool : 5 ~ 180	5 ~ 180
11	Data.CoolItime (Low Byte)		
12	Data.Kfactor	K-Factor : 1 ~ 9	1 ~ 9

## Downlink | port 90

Byte	Data	Content	Range
0	data.SetTemperature (High Byte)	Set Temperature(°C) = D_Set_Temperature/10	50 ~ 350
1	data.SetTemperature (Low Byte)		
2	data.FanMode	0:OFF 1:LOW 2:MED 3:HIGH 4:AUTO	0 ~ 4
3	data.threshold <sup>1</sup>	Temperature change: 0.2°C ~ 5.0°C	2 ~ 50
4	data.SystemMode	0:OFF 1:COOL 2:FAN-ONLY	0 ~ 2
5	Data.CoolPBand	P-Band for Cool : 1.0°C ~ 4.0°C	10 ~ 40
6	Data.CoolItime (High Byte)	I-Time for Cool : 5 ~ 180	5 ~ 180
7	Data.CoolItime (Low Byte)		
8	Data.Kfactor	K-Factor : 1 ~ 9	1 ~ 9

## Downlink | port 91

Byte	Data	Content	Range
0	data.SetTemperature (High Byte)	Set Temperature(°C) = D_Set_Temperature/10	50 ~ 350
1	data.SetTemperature (Low Byte)		

## Downlink | port 92

Byte	Data	Content	Range
0	data.FanMode	0:OFF 1:LOW 2:MED 3:HIGH 4:AUTO	0 ~ 4

## Downlink | port 93

Byte	Data	Content	Range
0	data.threshold <sup>2</sup>	Temperature change: 0.2°C ~ 5.0°C	2 ~ 50

<sup>1</sup> D\_update\_threshold determines the minimum change in ambient room temp required to trigger a send event i.e. uplink. The range is from 0.2 to 5 centigrade. However, this parameter is limited by another named, “sending interval”, hardcoded 15 seconds.

e.g. if change in temp > 0.2°C, or, fan status change, or user press a button etc., sends uplink immediately

<sup>2</sup> D\_update\_threshold determines the minimum change in ambient room temp required to trigger a send event i.e. uplink. The range is from 0.2 to 5 centigrade. However, this parameter is limited by another named, “sending interval”, hardcoded 15 seconds.

e.g. if change in temp > 0.2°C, or, fan status change, or user press a button etc., sends uplink immediately

**Downlink | port 94**

Byte	Data	Content	Range
0	data.SystemMode	0:OFF 1:COOL 2:FAN-ONLY	0 ~ 2

**Downlink | port 95**

Byte	Data	Content	Range
0	Data.CoolPBand	P-Band for Cool : 1.0°C ~ 4.0°C	10 ~ 40

**Downlink | port 97**

Byte	Data	Content	Range
0	Data.CoolItime (High Byte)	I-Time for Cool : 5 ~ 180	5 ~ 180
1	Data.CoolItime (Low Byte)		

**Downlink | port 95**

Byte	Data	Content	Range
0	Data.Kfactor	K-Factor : 1 ~ 9	1 ~ 9





## RELEASE NOTES

Release notes about documents, MQTT protocol, widgets, dashboard, etc.

- *Release Notes*
- *Upgrade instructions*

### 7.1 Release Notes

#### 7.1.1 v2.3.4 (Nov 8, 2023)

- Modify two wrong links in *Add TA692FC-L-5 to ThingsBoard*.

#### 7.1.2 v2.3.3 (Sep 25, 2023)

- **Avantec Dashboard**
  - New widget - Buttons navigation bar.
  - Update widget - Tabs navigation bar.
  - Update widget - Update time value with pattern key.
  - Update widget - Setting list.
  - Update dashboard - TA692FC-L-5 Detail Dashboard
  - Update dashboard - TA652FC-W Detail Dashboard
  - Update dashboard - TA652FH-W Detail Dashboard

#### 7.1.3 v2.3.2 (July 5, 2023)

- **Avantec Dashboard**
  - Updated TA692FC-L-5 Detail Dashboard
  - Updated TA652FC-W Detail Dashboard
  - Updated TA652FH-W Detail Dashboard

### 7.1.4 v2.3.1 (July 5, 2023)

- **Avantec Widgets**
  - New widget - Entities cards.

### 7.1.5 v2.3 (June 20, 2023)

- **Add TA692FC-L-5's documents**
  - Add TA692FC-L-5 Specification
  - Add TA692FC-L-5 LoRaWAN Device API
  - New application note: Add TA692FC-L-5 to ThingsBoard
  - Add TA692FC-L-5 List Dashboard
  - Add TA692FC-L-5 Detail Dashboard
- **Avantec Widgets**
  - New widget - Update shared string attribute with segmented switch.

### 7.1.6 v2.2 (June 1, 2023)

- **TA652FC-W MQTT API/Protocol**
  - **New feature: support (control mode) on/off in schedule.**
    - \* New client-side attributes: supportCtrlModeInSchedule (string), prgNextCtrlMode (string), prgCtrlModeXX (string).
    - \* New Server-side RPC: remoteSetPrgCtrlModeXX.
- **Avantec Widgets**
  - New widget for on/off in schedule - Styled button of string value with pattern key.
  - Fixed bug: primary color of some widgets about button can't be used in ThingsBoard v3.5.1.
  - Fixed bug: Setting list is not displayed properly in ThingsBoard v3.5.1.
- **TA652FC-W Dashboards**
  - **TA652FC-W List Dashboard**
    - \* New feature: support for modifying device label.
  - **TA652FC-W Detail Dashboard**
    - \* New feature: support (control mode) on/off in schedule.
- **TA652FH-W Dashboards**
  - **TA652FH-W List Dashboard**
    - \* New feature: support for modifying device label.
  - New dashboard - Office Center Dashboard

### 7.1.7 v2.1 (Apr 18, 2023)

- **TA652FC-W/TA652FH-W MQTT API/Protocol**
  - New shared attributes: `uploadThreshold` (double).
  - New client-side attributes: `uploadThresholdMin`, `uploadThresholdMax`, `uploadThresholdStep` (double).
- Refactor all documentation
- Refactor all widgets
- Refactor all dashboards

### 7.1.8 v1.0 (Jul 24, 2020 / Dec 20, 2022)

Initial version.

## 7.2 Upgrade instructions

- **Update Widgets Bundles:**
  - *Update Avantec Widgets*
- **Update Dashboards:**
  - **TA652FC-W**
    - \* *Update TA652FC-W List Dashboard*
    - \* *Update TA652FC-W Detail Dashboard*
  - **TA652FH-W**
    - \* *Update TA652FH-W List Dashboard*
    - \* *Update TA652FH-W Detail Dashboard*
    - \* *Update Office Center Dashboard*
  - **TA692FC-L-5**
    - \* *Update TA692FC-L-5 List Dashboard*
    - \* *Update TA692FC-L-5 Detail Dashboard*
- **Update F/W:**
  - *OTA Updates*



## AVANTEC AND THE PROJECT

Learn about the project and the company.

- [About us](#) | [Copyrights and Licenses](#)

### 8.1 About us

**Avantec Manufacturing Limited** was founded in 1983. We specialize in designing and manufacturing HVAC, telecom and VoIP products.

The company's headquarter is located in Kwun Tong, Hong Kong. This owned property hosts 40 staffs from R&D, Marketing, Shipping, Purchasing and Accounting Departments. Our products are distributed worldwide for years with CE, FCC, PTT and ROHS approvals.

We have a representative office and product development center in downtown Shenzhen.

We also work for customers on OEM projects. We have different R&D teams responsible for product design, software / PCB / tooling development and product approvals. We follow the rapid technology change and provide the products meeting customer's expectation and requirements.

With the abundant experience in this industry, our expertise is guaranteed. We are confident in providing you with innovative products meeting your requirements and in high quality. We are definitely looking forward to establishing a long term business partnership with you!

<http://www.avantec.com.hk>

### 8.2 Copyrights and Licenses

#### 8.2.1 Copyrights

All original source code & document in this repository is Copyright (C) 2023 Avantec Manufacturing Limited. This source code is licensed under the Apache License 2.0 as described in the file [LICENSE](#).

Additional third party copyrighted code & document is included under the following licenses.

Where source code & document headers specify Copyright & License information, this information takes precedence over the summaries made here.

## 8.2.2 ThingsBoard License

`thingsboard/thingsboard.github.io` is licensed under the [Apache License 2.0](#).