Internet Project Description

Deadline {Wednesday} 05/01/2022

In this project you will understand:

1- A client – Server architecture.
2- Socket programming.
3- Multi-threading.
4- Building a simple IoT application.
5- Using ThingsBoard platform.

Project description:

Using Internet of Things (IOT), we can control any electronic equipment in homes and industries. Moreover, you can read a data from any sensor and analyze it graphically from anywhere in the world.

Assume you have an IoT sensor (ex: temperature, light, RFID…) connected to the analog input of an ESP8266 Wi-Fi module (you will only need to get the ESP8266 for the project) which will periodically send the temperature readings to the server, using client server module of socket programming. The data received by the server will then be sent to ThingsBoard platform for visualization and any analytical usage in the future.

Milestone 4:

In this milestone, you will build up on the previously implemented client-server architecture (ESP8266 client and the python server). This building on will be done by uploading the data (by the server) collected from the sensors to an online live platform called ThingsBoard (Only data collected from one ESP8266 is needed).

This milestone can be viewed as 3 main parts:

1- The first part of the milestone will be setting up ThingsBoard. ThingsBoard is an open-source IoT platform which is used for data collection, processing, visualizing and for device management. ThingsBoard is easy to use and enables connectivity via standard IoT protocols like HTTP and MQTT. ThingsBoard also combines scalability, fault-tolerance and performance which ensure the perseverance of the data. ThingsBoard is quite easy to set-up and use. Some helpful instructions will be provided below.

2-The second part of the milestone will be communicating with ThingsBoard using the python server. Python communicates with ThingsBoard using the MQTT protocol. The MQTT protocol is a standard IoT communication protocol. It is extremely lightweight and efficient which makes it useful in networks with minimal bandwidth and on small microcontrollers. It also provides scalability, bi-directional communications, reliable message delivery and secure communications. Further resources about MQTT will be provided below.

3- The third part will be modifying the python server (accepting the data from the ESP) -which was implemented in the previous milestones- to be able to communicate and send the data to ThingsBoard platform.

By the end of this milestone, you are expected to have a full working IoT network between the ESP8266, python server and ThingsBoard where:

- The ESP8266 should read the data from the sensor (temperature, light... etc.)
- The ESP8266 will send the data to the python server using a TCP socket.
- The python server should then forward the data towards ThingsBoard.
- The data should be stored and visualized on ThingsBoard.

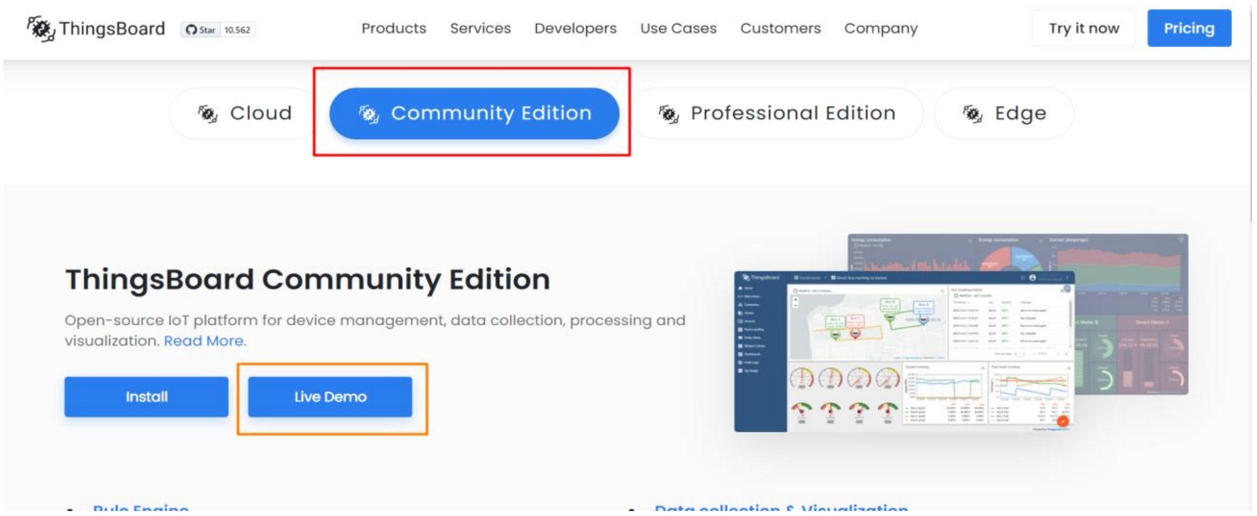
Deliverables:

1- Arduino Code for the ESP8266 which sends the data to Python server
2- Python Code accepting the data from the ESP and forwarding the data to ThingsBoard
3- Working ThingsBoard dashboard with real time values.

Some Helpful Instructions:

Setting Up ThingsBoard and Dashboard:

1- Start by visiting ThingsBoard website using the following link " https://thingsboard.io/"
2- Press on "Try it now" on the upper right corner.
3- Choose the "Community Edition" tab and then press on Live Demo.

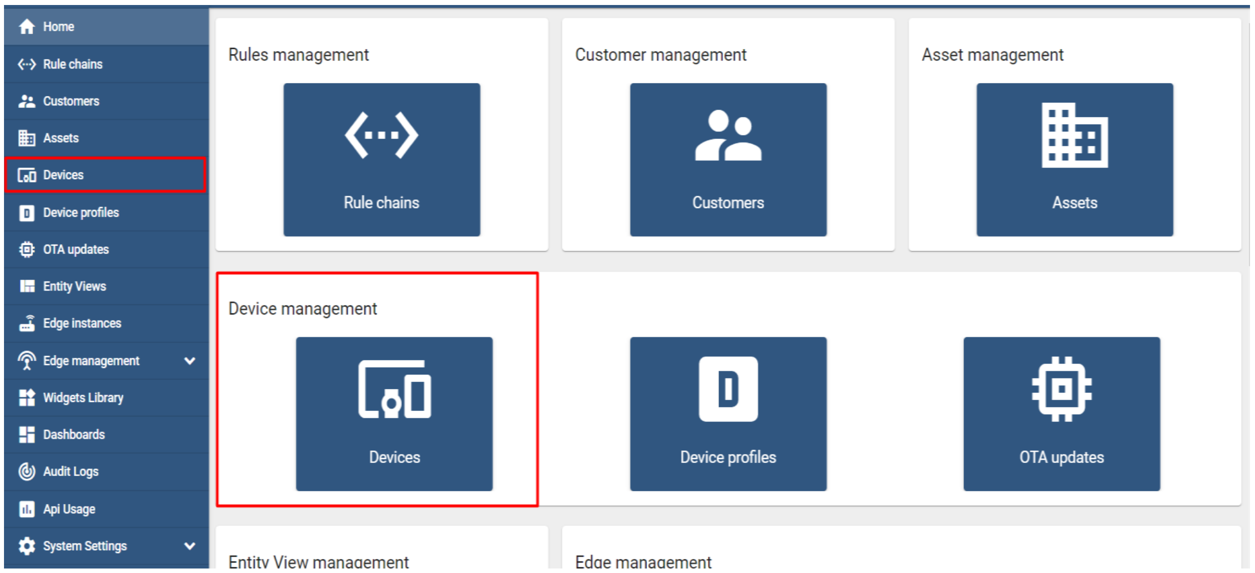


ThingsBoard Community Edition

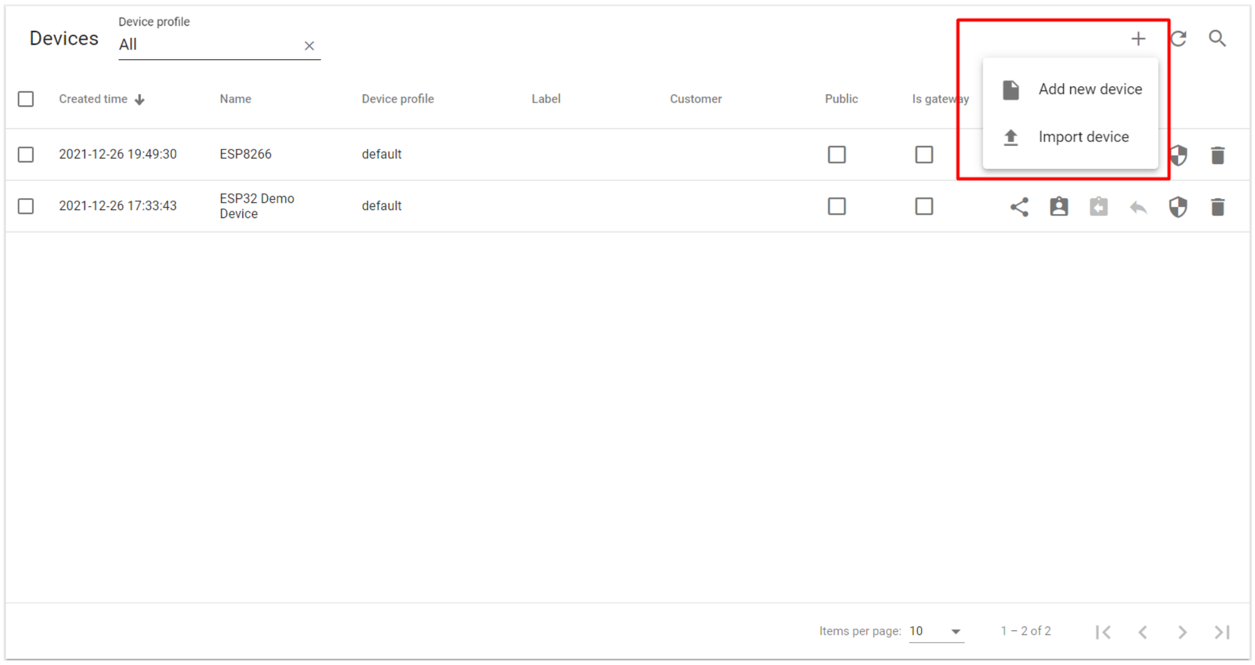
Open-source IoT platform for device management, data collection, processing and visualization. Read More.

4- Sign up with a new account to be able to use ThingsBoard.

5- Start with adding a new device by choosing “Devices” from the left panel or from “Home”.

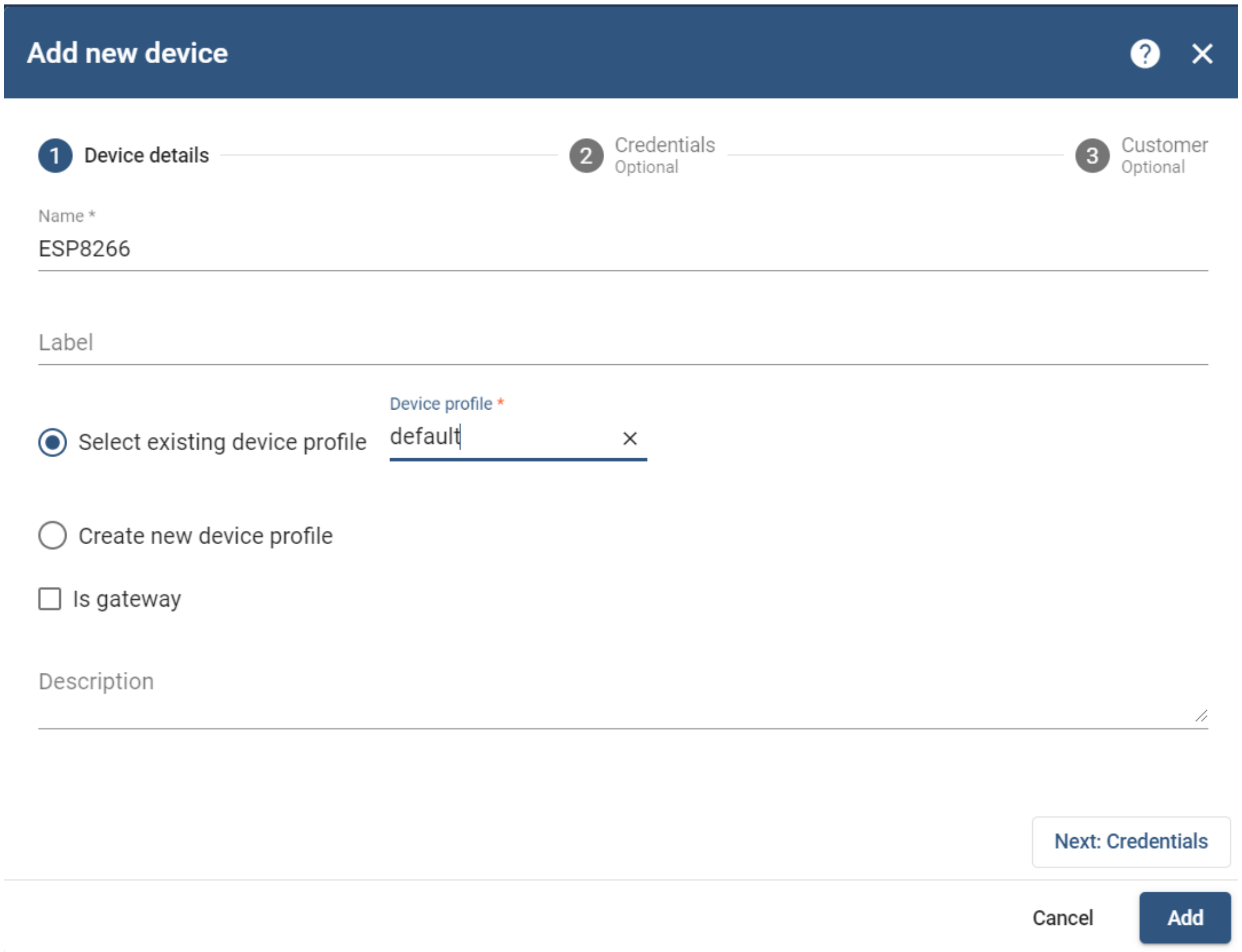


6- Add a new device by pressing on the “+” button and choosing add new device(You will have an empty list unlike this screenshot)



Created time	Name	Device profile	Label	Customer	Public	Is gateway
2021-12-26 19:49:30	ESP8266	default			☐	☐
2021-12-26 17:33:43	ESP32 Demo Device	default			☐	☐

7- Give the device a name (ex: ESP8266) and give it the ‘default’ device profile and an optional description and press “Add”.


Add new device

1 Device details — 2 Credentials Optional — 3 Customer Optional

Name *
ESP8266

Label

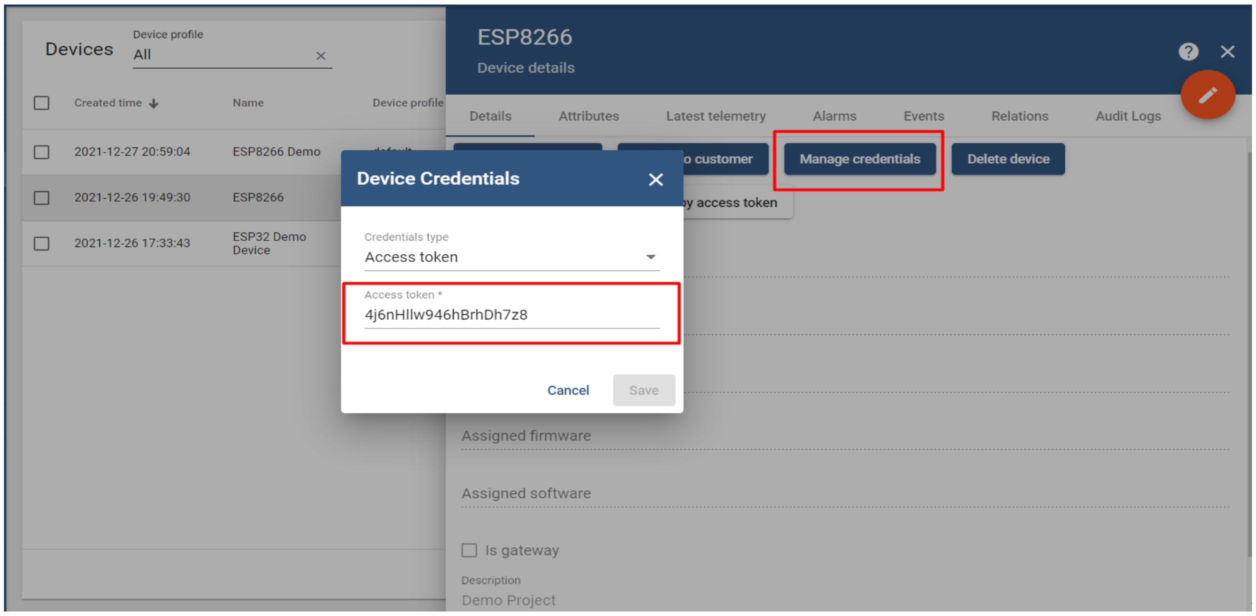
(●) Select existing device profile — Device profile *: default

( ) Create new device profile

☐ Is gateway

Description

8- By selecting the device and pressing “Manage credentials”, we can get the access token which will be useful later.


Devices — Device profile: All

	Created time	Name	Device profile
☐	2021-12-27 20:59:04	ESP8266 Demo	default
☐	2021-12-26 19:49:30	ESP8266	
☐	2021-12-26 17:33:43	ESP32 Demo Device	

ESP8266
Device details

Device Credentials

Credentials type
Access token

Access token *
4j6nHllw946hBrhDh7z8

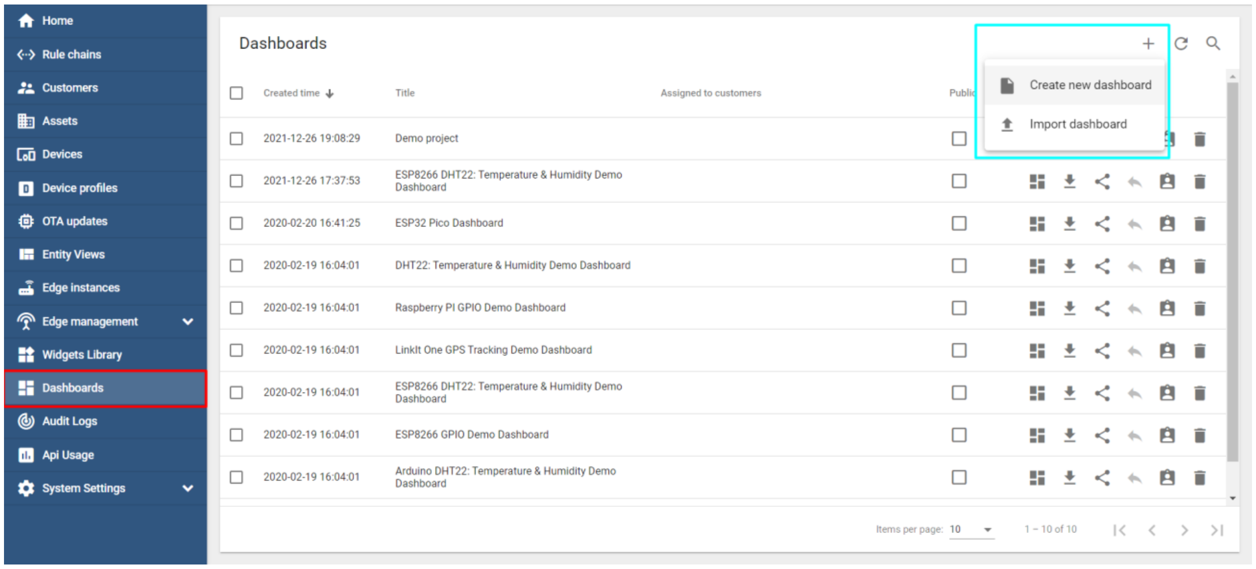
Assigned firmware

Assigned software

☐ Is gateway

Description
Demo Project

9- We will create the dashboard by choosing dashboards on the left panel and add a new dashboard by pressing the "+" button and choosing "create new dashboard" (Again your dashboards should be empty like the Devices earlier).

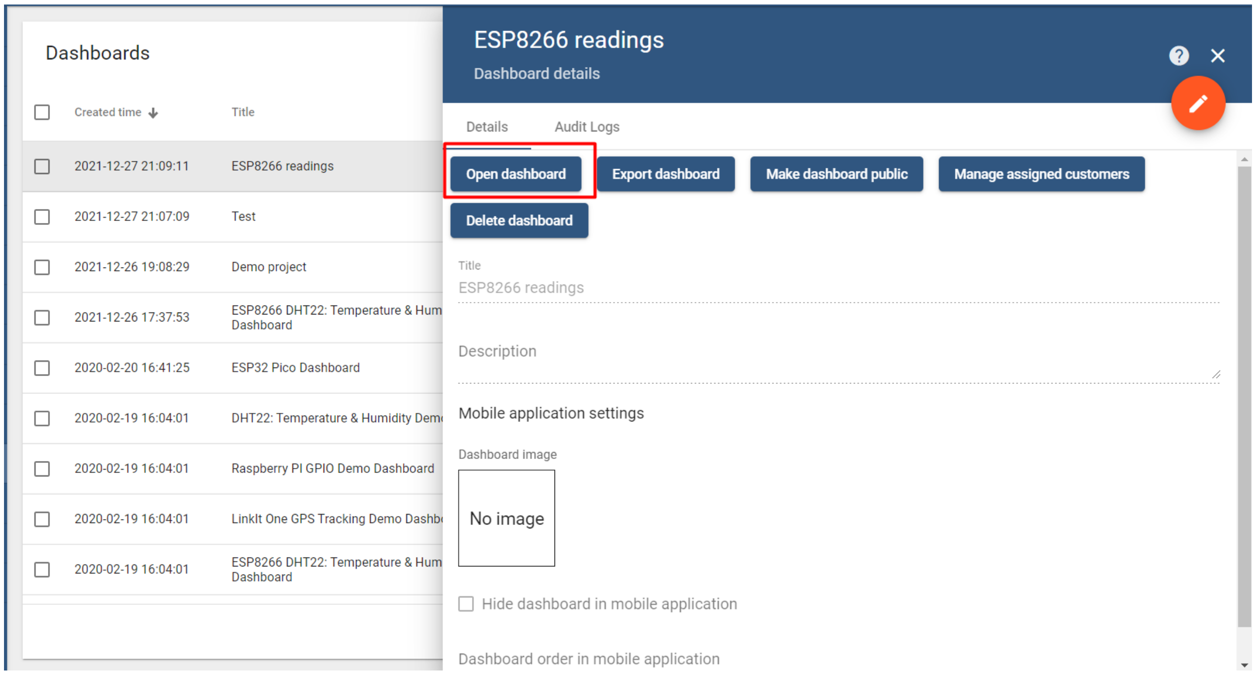


Dashboards

Created time	Title	Assigned to customers	Public
2021-12-26 19:08:29	Demo project		☐
2021-12-26 17:37:53	ESP8266 DHT22: Temperature & Humidity Demo Dashboard		☐
2020-02-20 16:41:25	ESP32 Pico Dashboard		☐
2020-02-19 16:04:01	DHT22: Temperature & Humidity Demo Dashboard		☐
2020-02-19 16:04:01	Raspberry PI GPIO Demo Dashboard		☐
2020-02-19 16:04:01	LinkIt One GPS Tracking Demo Dashboard		☐
2020-02-19 16:04:01	ESP8266 DHT22: Temperature & Humidity Demo Dashboard		☐
2020-02-19 16:04:01	ESP8266 GPIO Demo Dashboard		☐
2020-02-19 16:04:01	Arduino DHT22: Temperature & Humidity Demo Dashboard		☐

10- Give the dashboard a name (ex: ESP8266 readings) and click "Add"

11- Open the dashboard by choosing the dashboard and pressing "Open dashboard"



Dashboards

Created time	Title
2021-12-27 21:09:11	ESP8266 readings
2021-12-27 21:07:09	Test
2021-12-26 19:08:29	Demo project
2021-12-26 17:37:53	ESP8266 DHT22: Temperature & Hum Dashboard
2020-02-20 16:41:25	ESP32 Pico Dashboard
2020-02-19 16:04:01	DHT22: Temperature & Humidity Dem
2020-02-19 16:04:01	Raspberry PI GPIO Demo Dashboard
2020-02-19 16:04:01	LinkIt One GPS Tracking Demo Dashb
2020-02-19 16:04:01	ESP8266 DHT22: Temperature & Hum Dashboard

ESP8266 readings
Dashboard details

Title
ESP8266 readings

Description

Mobile application settings

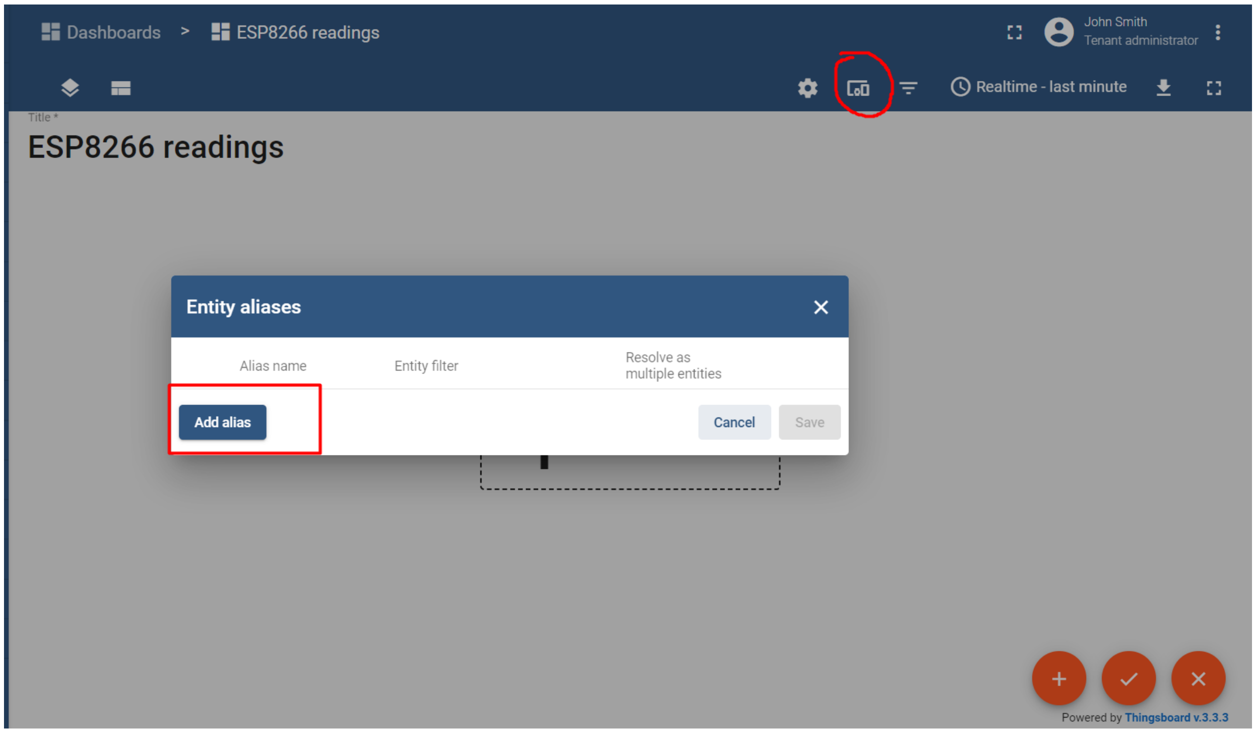
Dashboard image
No image

☐ Hide dashboard in mobile application

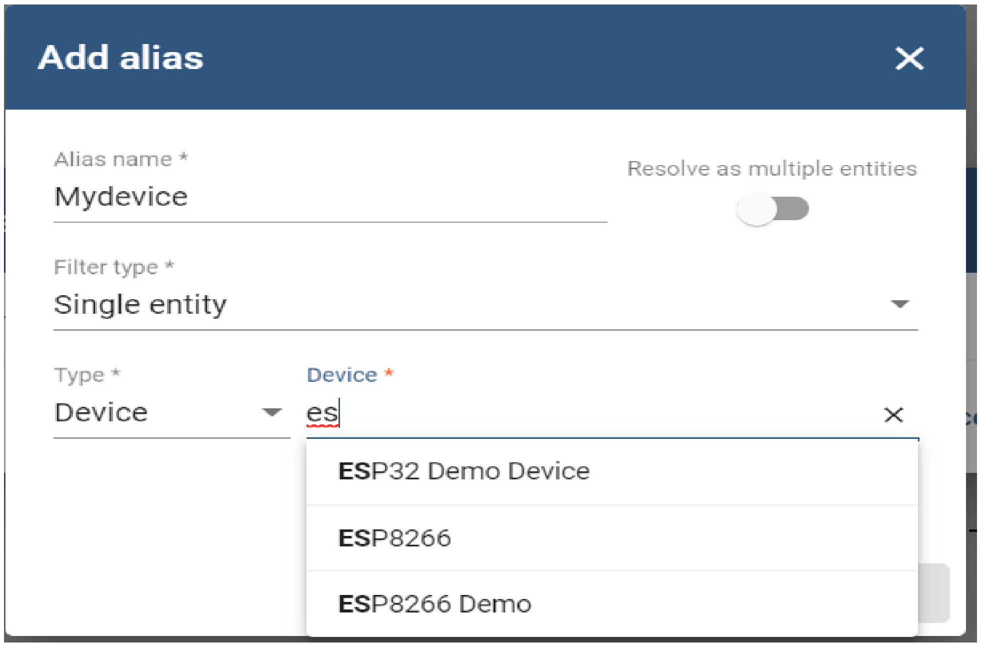
Dashboard order in mobile application

12- Press on the pencil icon on the lower right corner to start editing the dashboard

13- Start by adding the device which will be monitored and visualized by pressing on the Entity aliases and choose “Add alias”



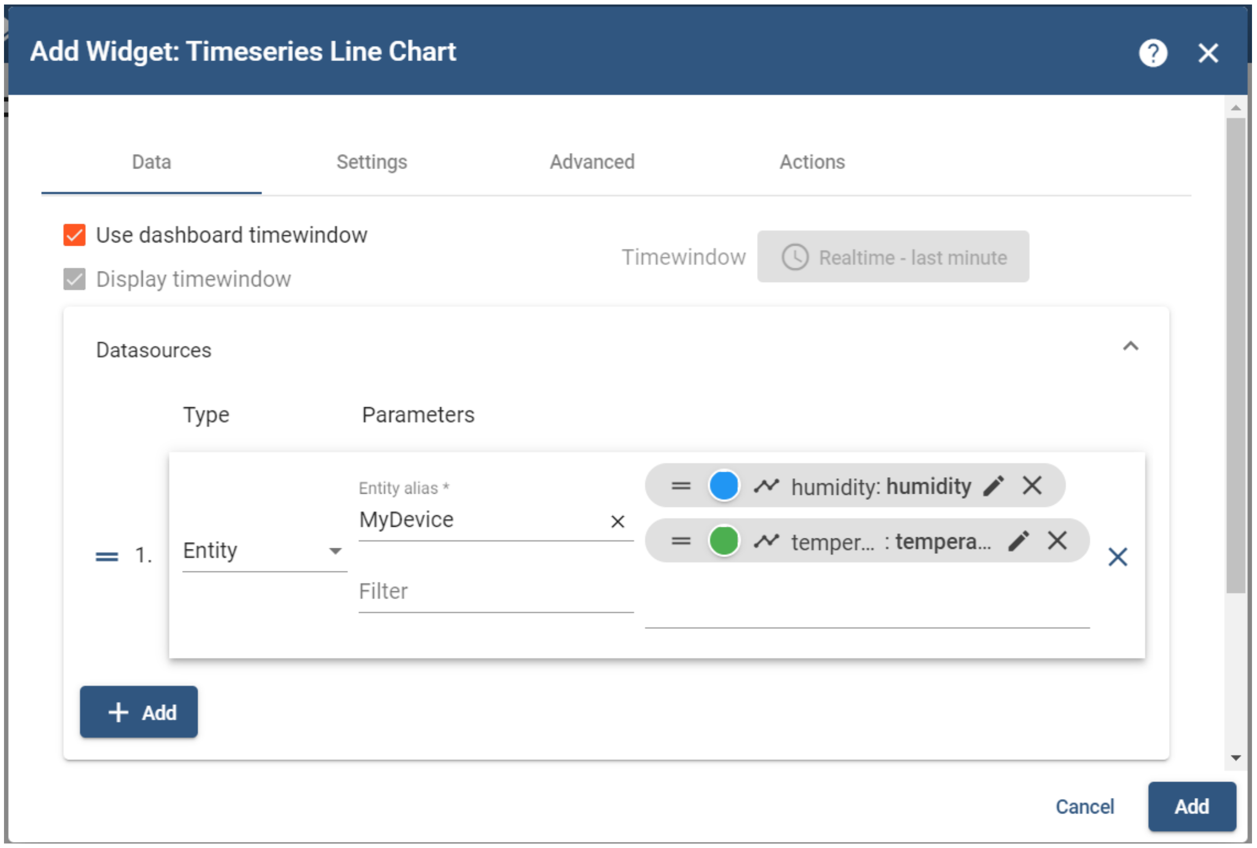
14- Give it any name (ex:MyDevice) , select Single Entity , under type choose Device and write the name of the device you added in step 7.



Alias name *: Mydevice
Resolve as multiple entities
Filter type *: Single entity
Type *: Device
Device *: es

15- Add a new widget by pressing “Add new Widget”. There are lots of widget types all used to visualize the data, you can test out with the different one but let’s just add “Timeseries Line Chart” under “Charts”.

16- Add the datasources as follows (This can be done after sending the actual data from the python server)



Add Widget: Timeseries Line Chart

Data | Settings | Advanced | Actions

- [x] Use dashboard timewindow
- [x] Display timewindow

Timewindow: Realtime - last minute

Datasources

	Type	Parameters	
1.	Entity	Entity alias *: MyDevice; Filter	humidity: humidity; temper... : tempera...

Tips for Sending MQTT message from Python Server:

1- The “paho mqtt” library in python enables sending the mqtt easily. Make sure you have the library installed before starting.

2- The access token from ThingsBoard will be used in the python code to identify the device on ThingsBoard.

3- Host name will also be required in the python code which will be set to “demo.thingsboard.io”.

4- A sample code will be provided in the useful links which can help in implementing this part.

Some Useful Links & Resources:

Reading about MQTT:

- https://mqtt.org/getting-started/

Getting started with ThingsBoard:

- https://thingsboard.io/docs/getting-started-guides/helloworld/

Sample Python Code (Python Server→ThingsBoard)

- https://bytesofgigabytes.com/thingsboard/sending-data-to-thingsboard-using-python/

BONUS:

It is only needed to have one ESP8266 working and sending its data to the python server which is then forwarded to ThingsBoard. This does not require the use of Multi-threading.

Having the two ESP8266's working and sending their data with the use of multithreading and having both data forwarded to ThingsBoard will be counted as a BONUS. Note: You should be able to differentiate between the readings of the two ESP8266's both in the python server and ThingsBoard.