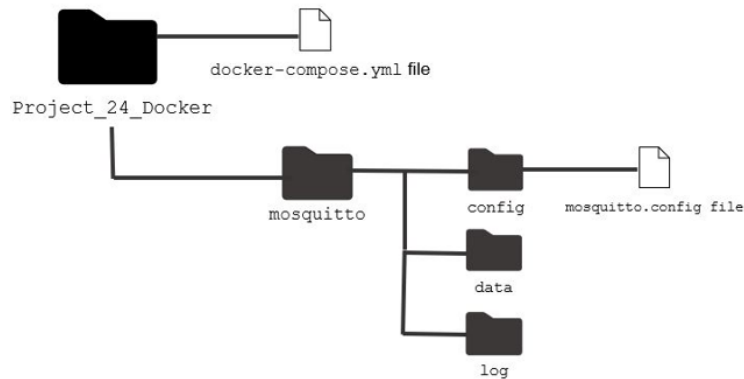


1. In a Terminal window, create a new folder called `Project_24_Docker`. Download the [docker-compose.yml](#) file and place it inside the `Project_24_Docker` folder. Inside the `Project_24_Docker` folder, create a folder titled `mosquitto`. Inside the `mosquitto` folder, create three more subfolders titled as follows: `config`, `data`, and `log`. Download the [mosquitto.config](#) file and place it inside the `config` folder. See the image below, which depicts the required folders and files:



Provide a screenshot to show that you correctly created all of the required folders and that you placed the `docker-compose.yml` and `mosquitto.config` files in the `Project_24_Docker` and `config` folders, respectively.

```

Directory: C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard,
Kafka\Project_24_Docker

Mode                LastWriteTime         Length Name
----                -
d-----            8/23/2023   12:41 PM             mosquitto
-a-----            8/23/2023   12:41 PM             289 docker-compose.yml

PS C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard, Kafka\Project_24_Docker> c
d .\mosquitto\
PS C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard, Kafka\Project_24_Docker\mo
squitto> ls

Directory: C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard,
Kafka\Project_24_Docker\mosquitto

Mode                LastWriteTime         Length Name
----                -
d-----            8/23/2023   12:42 PM             config
d-----            8/23/2023   12:41 PM             data
d-----            8/23/2023   12:41 PM             log

PS C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard, Kafka\Project_24_Docker\mo
squitto> cd .\config\
PS C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard, Kafka\Project_24_Docker\mo
squitto\config> ls

Directory: C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard,
Kafka\Project_24_Docker\mosquitto\config

Mode                LastWriteTime         Length Name
----                -
-a-----            8/23/2023   12:42 PM             134 mosquitto.conf

PS C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard, Kafka\Project_24_Docker\mo
squitto\config> |

```

2. In a Terminal window, navigate to the `Project_24_Docker` folder and run the command below to initialize your Mosquitto *container*.

`docker-compose up`

Provide a screenshot of your Docker GUI to show that you have successfully initialized the Mosquitto *container*.



3. In a local Terminal window, run the command below to install the Paho MQTT Python *client library* locally:

`pip install paho-mqtt`

Provide a screenshot to show that you have successfully installed the Paho MQTT Python *client library*.

```

PS C:\Users\mauri> pip install paho-mqtt
Requirement already satisfied: paho-mqtt in c:\users\mauri\appdata\local\programs\python\python311\lib\site-packages (1.6.1)
PS C:\Users\mauri> |

```

4. In a Terminal window, navigate to your home folder.
For Windows users:

Enter the command below to navigate to your home folder:

```
cd ~
```

For Mac users:

Enter the command below to navigate to your home folder:

```
cd
```

Inside of your home folder, create two folders named `.mytb-data` and `.mytb-logs`.

Provide a screenshot to show that you created the `.mytb-data` and `.mytb-logs` folders in your home folder.

```
Mode                LastWriteTime         Length Name
-----
d-----          8/23/2023  12:46 PM             .mytb-data

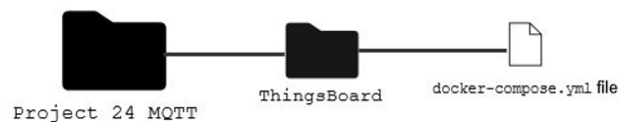
PS C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard, Kafka\Project_24_Docker> mkdir .mytb-logs

Directory: C:\Users\mauri\Documents\MIT xPro\Module 24 Handling Big Data with Mosquitto, ThingsBoard, Kafka\Project_24_Docker

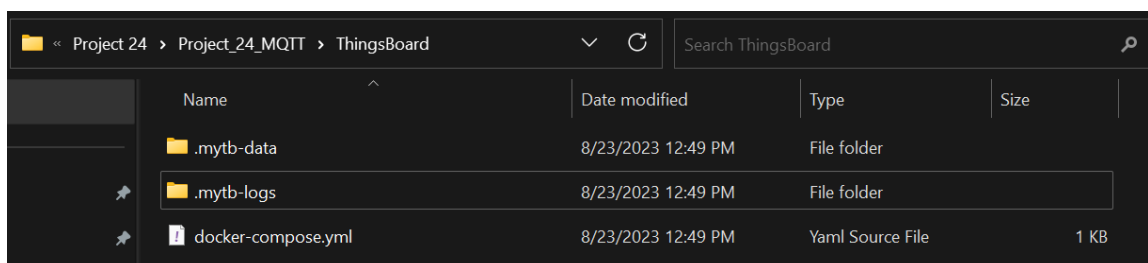
Mode                LastWriteTime         Length Name
-----
d-----          8/23/2023  12:46 PM             .mytb-logs
```

5. Inside the same location where you created the `Project_24_Docker` folder, create a new folder titled `Project_24_MQTT`. Inside the `Project_24_MQTT` folder, create a new subfolder titled `ThingsBoard`. Download the [docker-compose.yml](#) file and place it inside the `ThingsBoard` folder. Also inside of the `ThingsBoard` folder, create two folders named `.mytb-data` and `.mytb-logs`.

See the image below, which depicts the required folders and files:



Provide a screenshot to show that you correctly created all of the required folders and that you placed the `docker-compose.yml` inside the `ThingsBoard` folder.

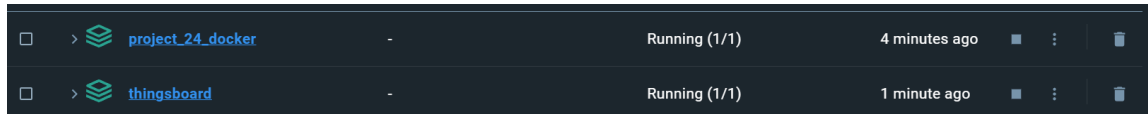


6. In a Terminal window, navigate to the `ThingsBoard` folder that you created in Step 4 and run the command below to initialize your ThingsBoard *container*:

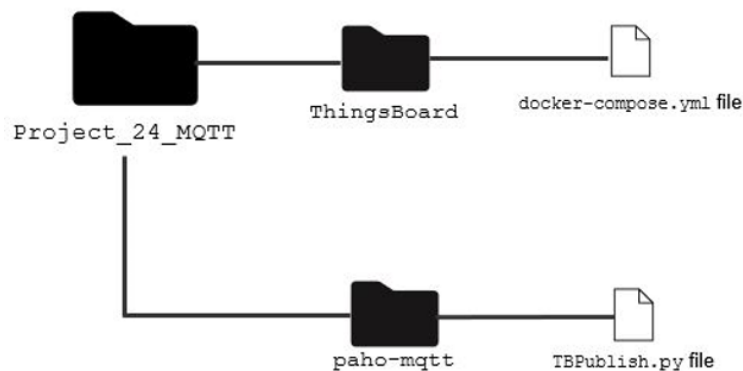
```
docker-compose up
```

If the ThingsBoard *container* does not spin up correctly, then change the ports on line 8 to `1883:1883` and try to spin up the *container* again.

Provide a screenshot of your Docker GUI to show that you have successfully initialized the ThingsBoard *container*.



7. Inside the `Project_24_MQTT` folder, create a new subfolder titled `paho-mqtt`. Download the [TBPublish.py](#) file and place it inside the `paho-mqtt` folder. Open the `TBPublish.py` file in VS Code. See the image below, which depicts the required folders and files:



Modify the `sensor_data` *dictionary* by adding another *key*, `humidity`, with a corresponding value equal to 0. Inside the `while` *loop*, add a *statement* to generate random integer values between 50 and 100. Assign these values to the `humidity` variable.

Provide a screenshot to show that you created the `paho-mqtt` folder and modified the code inside the `TBPublish.py` file to add the `humidity` *key* with the correct values assigned to the `humidity` variable.

```
pahopublish.py  TBPublish.py X  pahosubscribe.py
Project 24 > Project_24_MQTT > ThingsBoard > paho-mqtt > TBPublish.py > ...
1  import time
2  import random
3  import paho.mqtt.client as mqtt
4  import json
5  PORT = 9883
6  THINGSBOARD_HOST = 'localhost'
7  ACCESS_TOKEN = 'DHT11_DEMO_TOKEN'
8
9  # Data capture and upload interval in seconds.
10
11  sensor_data = {'temperature': 0, 'humidity': 0}
12
13  client = mqtt.Client()
14  # Set access token
15  client.username_pw_set(ACCESS_TOKEN)
16
17  # Connect to ThingsBoard using default MQTT port and 60 seconds keepalive interval
18  client.connect(THINGSBOARD_HOST, PORT, 60)
19  client.loop_start()
20
21  try:
22      while True:
23          temperature = random.randint(0, 100)
24          humidity = random.randint(50, 100)
25
26          print(f"Temperature: {temperature} humidity: {humidity}")
27          sensor_data['temperature'] = temperature
28          sensor_data['humidity'] = humidity
29
30          # Sending humidity and temperature data to ThingsBoard
31          client.publish('v1/devices/me/telemetry', json.dumps(sensor_data), 1)
32          time.sleep(3)
33  except KeyboardInterrupt:
34      pass
35
36  client.loop_stop()
37  client.disconnect()
```

8. Open a Terminal window in VS Code. Run the `TBPublish.py` file. Provide a screenshot to show that your code is correctly producing temperature and humidity data.

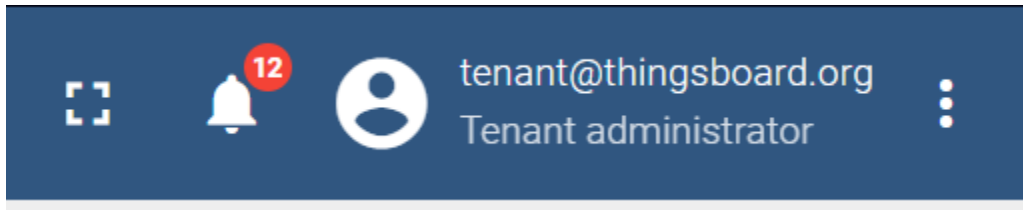
```
Temperature: 4 humidity: 99
Temperature: 94 humidity: 58
Temperature: 91 humidity: 87
Temperature: 22 humidity: 56
Temperature: 79 humidity: 75
Temperature: 33 humidity: 88
```

9. In a browser window, navigate to <http://localhost:8080/>. Log in to ThingsBoard using the credentials below:

Login: tenant@thingsboard.org

Password: tenant

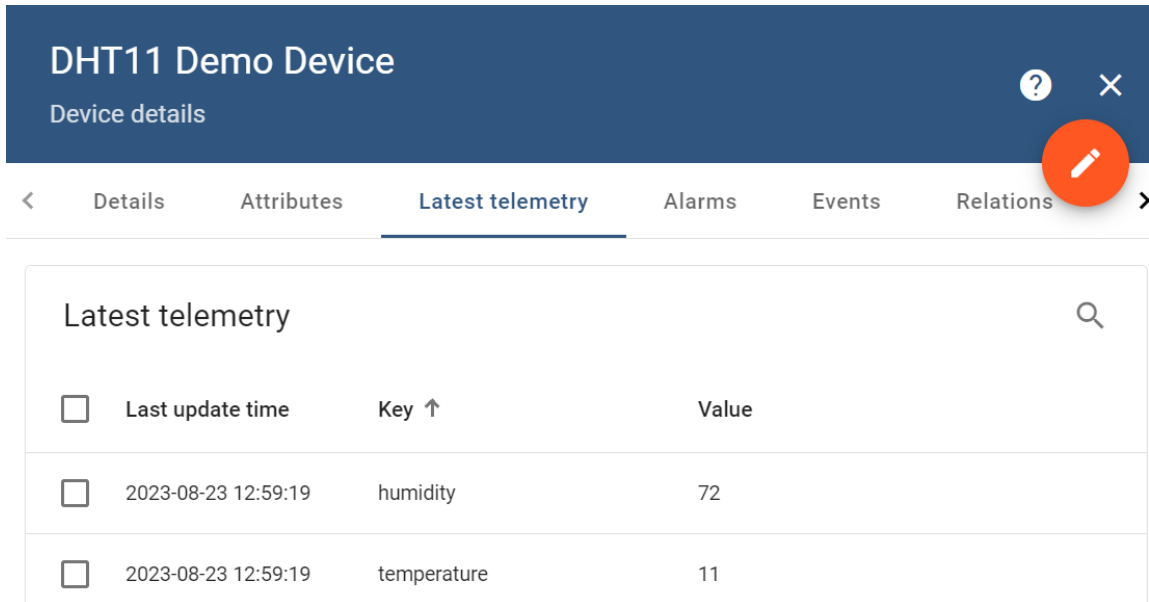
Provide a screenshot to show that you successfully logged in to ThingsBoard by using the credentials provided.



10. In ThingsBoard, from the menu on the left, select “*Devices*”. You should see an existing *device* called DHT11 Demo *Device*. This *device publishes* data produced by an MQTT protocol to ThingsBoard. In other words, this *device* is able to read the data produced by the `TBPublish.py` file and send it to ThingsBoard.

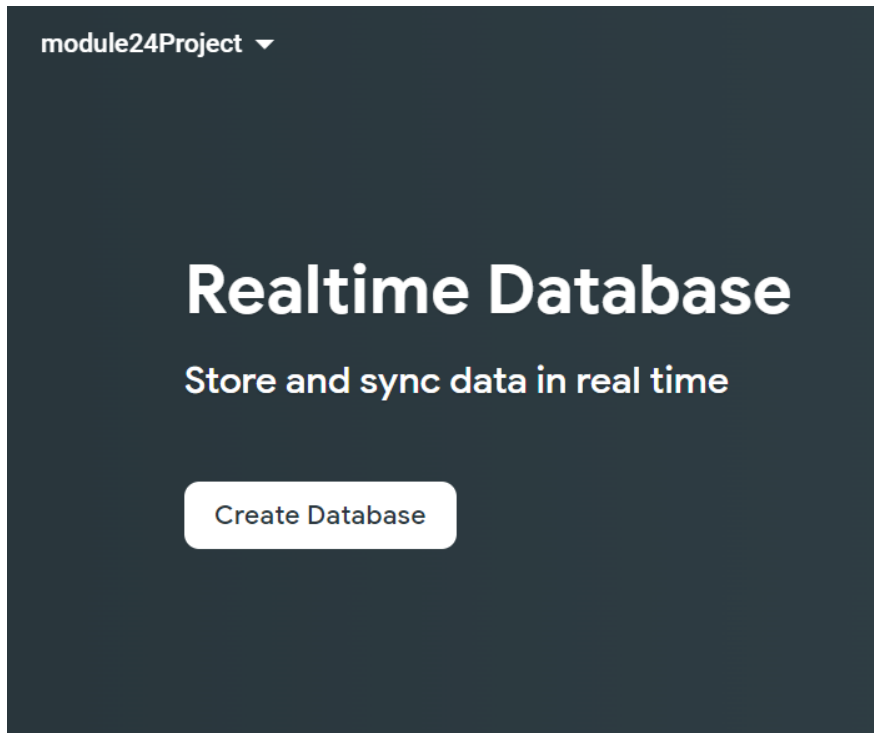
Open the DHT11 Demo *Device* by selecting it. Navigate to the Latest Telemetry tab to see the latest telemetry.

Provide a screenshot of the data in the latest telemetry tab to show that the DHT11 Demo *Device* is *publishing* the data produced by the `TBPublish.py` file to ThingsBoard.




11. Navigate to the main page of Firebase. Follow the steps in [Video 24.3](#) to add a new project called `module24Project`.

Provide a screenshot to show that you created the `module24Project` project in Firebase.

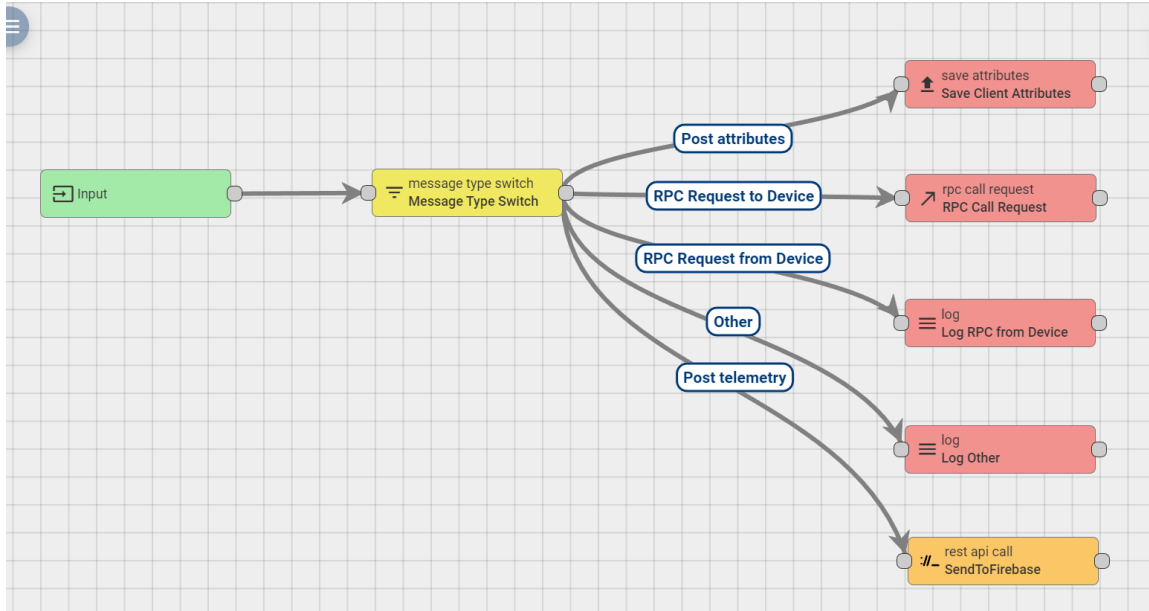


12. Follow the steps in [Video 24.3](#) to create a Realtime database in Firebase. Add a field in your database titled `temperature` and initialize the corresponding field to zero. Provide a screenshot to show that you created the `temperature` field inside your Realtime database.

 <https://module24project-b73c0-default-rtdb.firebaseio.com>

```
https://module24project-b73c0-default-rtdb.firebaseio.com/  
└── temperature: 0
```

13. Navigate to the Root Rule Chain in ThingsBoard. Add a “rest API call” *node* and name it Firebase. In the “Endpoint URL pattern” field, paste the link to your Realtime database from Firebase followed by `/temperature.json`. Connect the “Firebase” *node* to the “Message Type Switch” *node*. Add “Post telemetry” as the link label. Provide a screenshot to show that you have created the Firebase *node* correctly, connected it to the “Message Type Switch” *node*, and added “Post telemetry” as the link label.



14. Navigate to Firebase and display your temperature and humidity data by expanding the entries in your Realtime database. Provide a screenshot to show that your Realtime database is updating correctly and displaying your temperature and humidity data.

