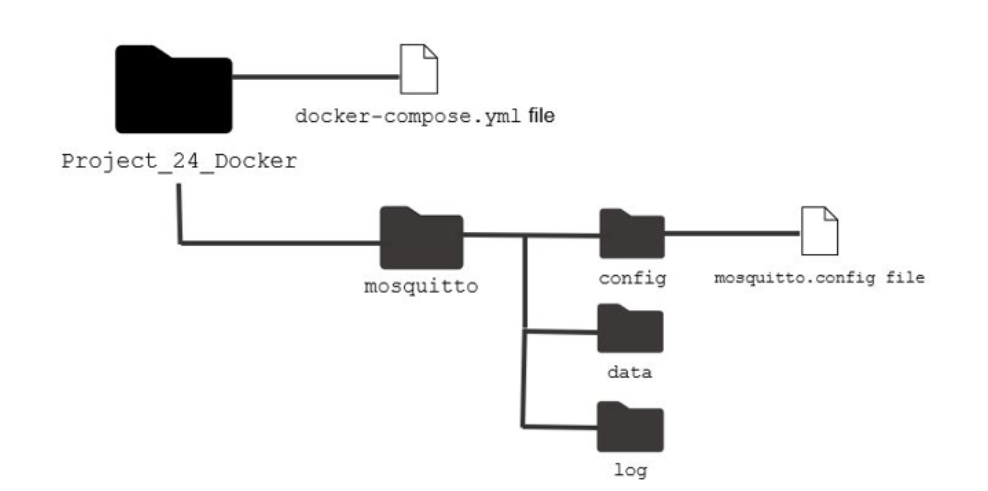
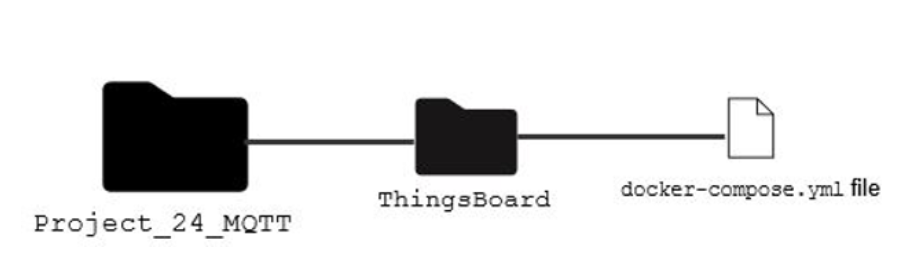
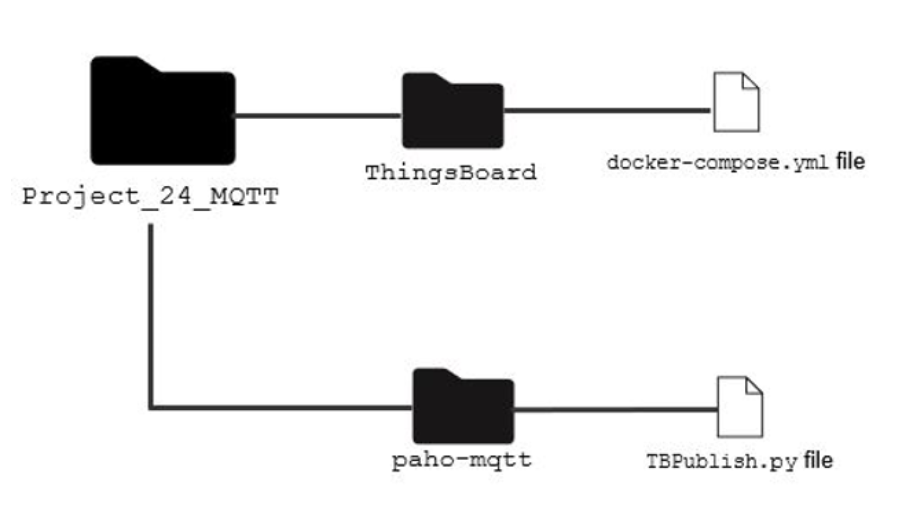
In the first part of this project, you will set up your Mosquitto, ThingsBoard, and Firebase environments. You will begin by setting up your MQTT protocol to produce temperature and humidity data. Next, you will ensure that the data produced by the MQTT protocol is *published* correctly to ThingsBoard. Finally, you will create a new project and a Realtime database in Firebase and send the temperature and humidity data to it.

This project is worth a total of 100 points.

**To complete this project, follow these steps:**

1. In a Terminal window, create a new folder called Project\_24\_Docker. Download the [docker-compose.yml](https://classroom.emeritus.org/courses/10605/files/3006931/download) 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](https://classroom.emeritus.org/courses/10605/files/3006937/download) file and place it inside the config folder.  
   See the image below, which depicts the required folders and files:  
   
2. Provide a screenshot showing that you correctly created all the required folders and placed the docker-compose.yml and mosquitto.config files in the Project\_24\_Docker and config folders, respectively.
3. 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*.
4. 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*.
5. 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 the 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.
6. 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](https://classroom.emeritus.org/courses/10605/files/3006887/download) 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 showing that you correctly created all the required folders and placed the docker-compose.yml inside the ThingsBoard folder.
7. 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 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*.
8. Inside the Project\_24\_MQTT folder, create a new subfolder titled paho-mqtt. Download the [TBPublish.py](https://classroom.emeritus.org/courses/10605/files/3006906/download) 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.
9. Open a Terminal window in VS Code. Run the TBPublish.py file.  
   Provide a screenshot showing that your code is correctly producing temperature and humidity data.
10. 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 showing that you successfully logged in to ThingsBoard by using the credentials provided.
11. 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* can 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.
12. Navigate to the main page of Firebase. Follow the steps in [Video 24.3](https://classroom.emeritus.org/courses/10605/pages/getting-started-with-firebase-09-17) to add a new project called module24Project.  
    Provide a screenshot to show that you created the module24Project project in Firebase.
13. Follow the steps in [Video 24.3](https://classroom.emeritus.org/courses/10605/pages/getting-started-with-firebase-09-17) 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.
14. 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 showing that you have created the Firebase *node* correctly, connected it to the “Message Type *Switch*” *node*, and added “Post telemetry” as the link label.
15. 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.

Now that you have set up your MQTT protocol, verified that it's *publishing* data to ThingsBoard, and created a new project in Firebase, you are ready to learn how to set up *alarms* in ThingsBoard to monitor your data in Part 2 of this project.

**Submission Instructions:**

Your submission for this project should be a Word document that includes the following screenshots, each labeled for the step that the screenshot represents:

1. Provide a screenshot showing that you correctly created all the required folders and placed the docker-compose.yml and mosquitto.config files in the Project\_24\_Docker and config folders, respectively.
2. Provide a screenshot of your Docker GUI to show that you have successfully initialized the Mosquitto *container*.
3. Provide a screenshot showing that you have successfully installed the Paho MQTT Python *client* *library*.
4. Provide a screenshot showing that you created the .mytb-data and .mytb-logs folders inside the home folder correctly.
5. Provide a screenshot showing that you correctly created all the required folders and placed the docker-compose.yml inside the ThingsBoard folder.
6. Provide a screenshot of your Docker GUI showing that you have successfully initialized the ThingsBoard *container*.
7. Provide a screenshot showing 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.
8. Provide a screenshot showing that your code is correctly producing data for the temperature and the humidity.
9. Provide a screenshot showing that you successfully logged in to ThingsBoard using the credentials provided.
10. Provide a screenshot of the data in the latest telemetry tab showing that the DHT11 Demo *Device* is *publishing* the data produced by the TBPublish.py file to ThingsBoard.
11. Provide a screenshot showing that you created the module24Project project in Firebase.
12. Provide a screenshot showing that you created the temperature field inside your Realtime database.
13. Provide a screenshot showing 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. Provide a screenshot showing that your Realtime database is updating correctly and displaying your temperature and humidity data.