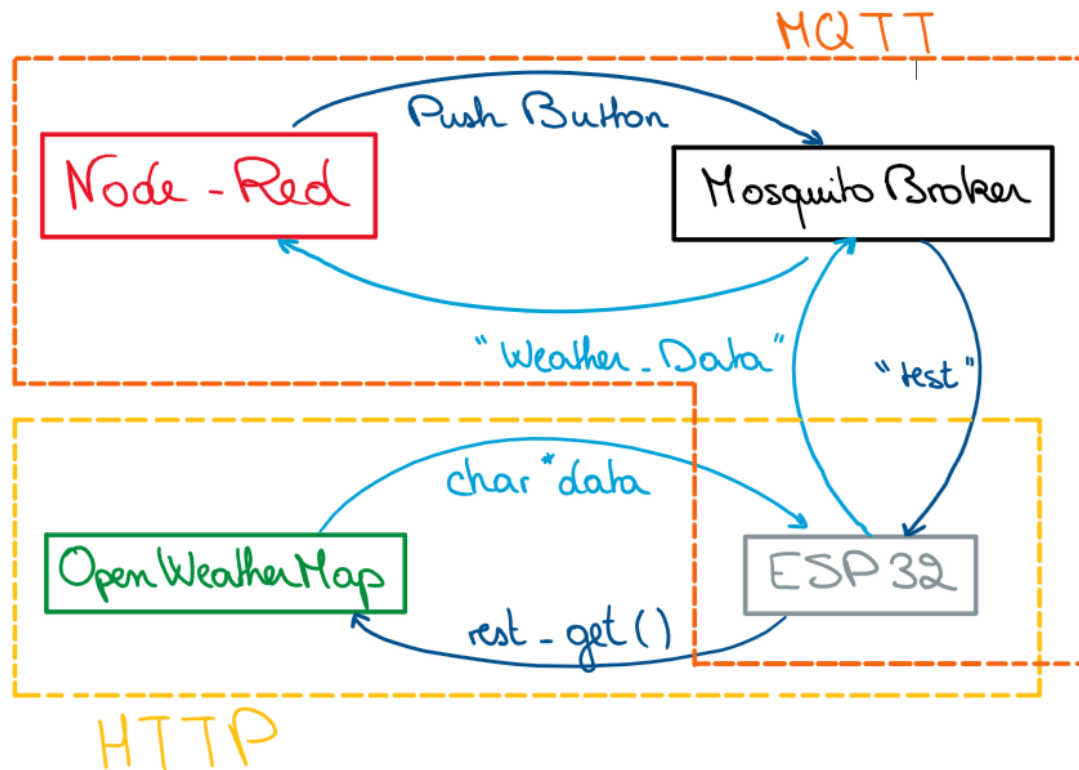


General Diagram



ESP32 HTTP Client

We first started with the HTTP part, the goal was to successfully retrieve the data from the OpenWeatherMap and print them on the terminal of VSCode in the format requested in the instructions.

- The `client_event_get_handler()` function retrieves the data and stores it inside a variable called `data`. Seeing that the received data is going to be too large, it will be sent to the esp in two parts, so we used the `strncat()` function to concatenate those two parts together
- The `rest_get()` function is used to do the API call using the API KEY we got after signing up to the weather app
- We added the function `print_data()` which would be responsible for selecting the data we need from the "data" variable and displaying it in the right format.
 - We first used the `cJSON_Parse()` on the "data" variable to turn it into a cJSON recognizable variable called "json" in order to work on it with the functions specific to cJSON
 - We used the `cJSON_GetObjectItem()` and to `cJSON_GetArrayItem()` extract the information needed and printed it using the `printf()` function

MQTT Mosquitto Broker

Installing and testing the MQTT Mosquitto broker was straight forward.

We used three terminals, the first one runs mosquitto in the background, the second one is the publisher whom sends a message through a specified topic, and the third one is the subscriber who subscribes to that topic and listens to the messages sent on that topic.

```
Command Prompt - mosquitto.exe -v
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User>cd C:\Program Files\mosquitto

C:\Program Files\mosquitto>mosquitto.exe -v
1670430703: mosquitto version 2.0.15 starting
1670430703: Using default config.
1670430703: Starting in local only mode. Connections will only be possible from clients running on this machine.
1670430703: Create a configuration file which defines a listener to allow remote access.
1670430703: For more details see https://mosquitto.org/documentation/authentication-methods/
1670430703: Opening ipv4 listen socket on port 1883.
1670430703: Opening ipv6 listen socket on port 1883.
1670430703: mosquitto version 2.0.15 running
1670430765: New connection from ::1:60418 on port 1883.
1670430765: New client connected from ::1:60418 as auto-565DEAB0-2BE1-8029-0AA3-884370577F6E (p2, c1, k60).
1670430765: No will message specified.
1670430765: Sending CONNACK to auto-565DEAB0-2BE1-8029-0AA3-884370577F6E (0, 0)
1670430765: Received SUBSCRIBE from auto-565DEAB0-2BE1-8029-0AA3-884370577F6E
1670430765: TEST (QoS 0)
1670430765: auto-565DEAB0-2BE1-8029-0AA3-884370577F6E 0 TEST
1670430765: Sending SUBACK to auto-565DEAB0-2BE1-8029-0AA3-884370577F6E
1670430826: Received PINGREQ from auto-565DEAB0-2BE1-8029-0AA3-884370577F6E
1670430826: Sending PINGRESP to auto-565DEAB0-2BE1-8029-0AA3-884370577F6E
1670430885: Received PINGREQ from auto-565DEAB0-2BE1-8029-0AA3-884370577F6E
1670430885: Sending PINGRESP to auto-565DEAB0-2BE1-8029-0AA3-884370577F6E
1670430897: New connection from ::1:60468 on port 1883.
1670430897: New client connected from ::1:60468 as auto-3322566B-D11F-FFF5-2998-2685E48F72C6 (p2, c1, k60).
1670430897: No will message specified.
1670430897: Sending CONNACK to auto-3322566B-D11F-FFF5-2998-2685E48F72C6 (0, 0)
1670430897: Received PUBLISH from auto-3322566B-D11F-FFF5-2998-2685E48F72C6 (d0, q0, r0, m0, 'TEST', ... (5 bytes))
1670430897: Sending PUBLISH to auto-565DEAB0-2BE1-8029-0AA3-884370577F6E (d0, q0, r0, m0, 'TEST', ... (5 bytes))
1670430897: Received DISCONNECT from auto-3322566B-D11F-FFF5-2998-2685E48F72C6
1670430897: Client auto-3322566B-D11F-FFF5-2998-2685E48F72C6 disconnected.
```

```
Command Prompt
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User>cd C:\Program Files\mosquitto

C:\Program Files\mosquitto>mosquitto_pub.exe -t TEST -m "HELLO" -d
Client null sending CONNECT
Client null received CONNACK (0)
Client null sending PUBLISH (d0, q0, r0, m1, 'TEST', ... (5 bytes))
Client null sending DISCONNECT

C:\Program Files\mosquitto>
```

```
Command Prompt - mosquitto_sub.exe -t TEST -d
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User>cd C:\Program Files\mosquitto

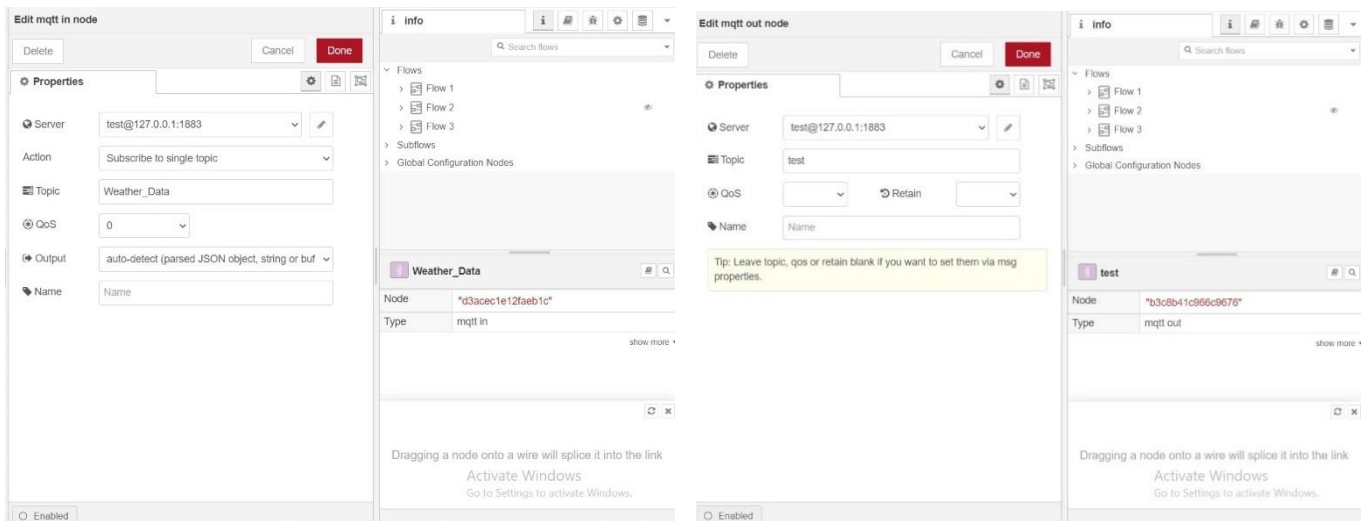
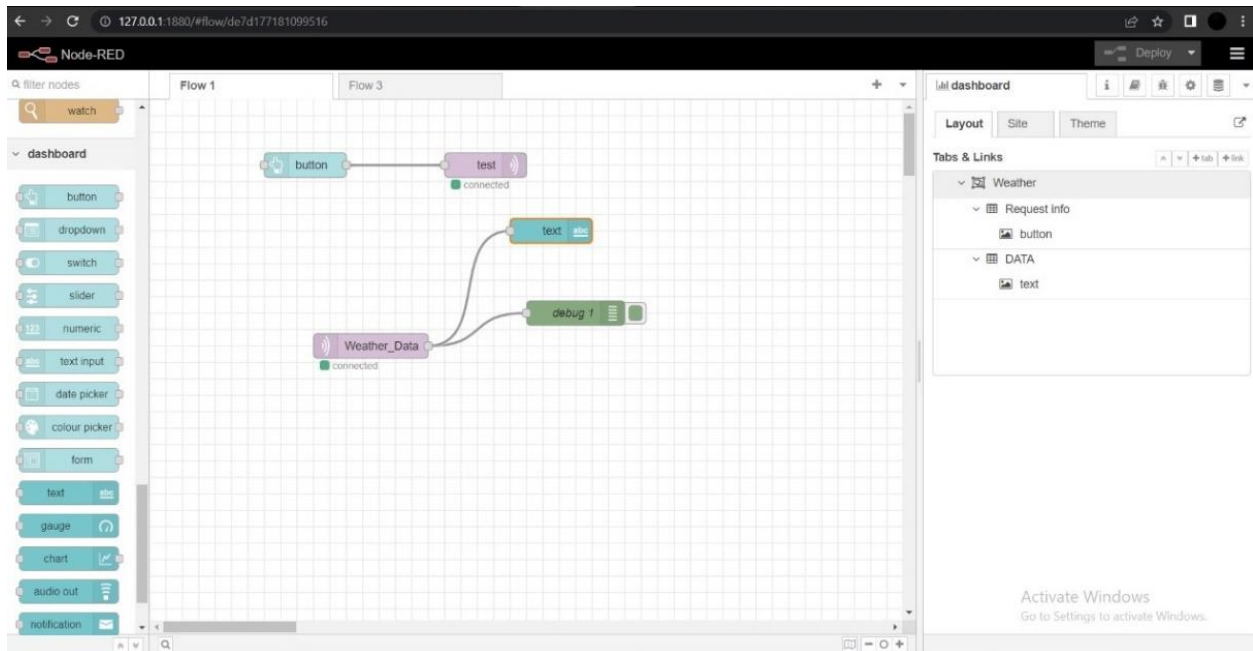
C:\Program Files\mosquitto>mosquitto_sub.exe -t TEST -d
Client null sending CONNECT
Client null received CONNACK (0)
Client null sending SUBSCRIBE (Mid: 1, Topic: TEST, QoS: 0, Options: 0x00)
Client null received SUBACK
Subscribed (mid: 1): 0
Client null sending PINGREQ
Client null received PINGRESP
Client null sending PINGREQ
Client null received PINGRESP
Client null received PUBLISH (d0, q0, r0, m0, 'TEST', ... (5 bytes))
HELLO
```

All of the above is done locally on the PC with no connection to the esp or Node-RED whatsoever.

For the purpose of our project we needed to allow the broker to communicate with the esp32 which was causing problems because the broker would only allow communication with the local host. To fix that problem we had to go to ProgramFiles\mosquitto and add a test.conf file which contained the two lines "listener 1883" and "allow_anonymous true".

Node-RED MQTT Client and Dashboard

To set up Node-RED we connected a push button to an MQTT publish node named "test" below. It was assigned the topic "test", the request for the data will be sent through it. On the other hand, there is the MQTT receive node which will be the one receiving the data through the topic "Weather_Data" and will display it using the "text" node on the page. We can also see below that both the servers for "test" and "Weather_Data" are the local host because they are communicating with the broker which is on the PC itself.



ESP32 MQTT Client

In the function `mqtt_event_handler_cb()`, in the case of `MQTT_EVENT_CONNECTED`, we put as a condition to check that the topic received is called "test" (the one we want to subscribe to) and in the case `MQTT_EVENT_DATA`, after making sure we are receiving a call through the topic we subscribed to, we will call the function `rest_get()` defined earlier to get the data from OpenWeatherMap and send it through the topic "Weather_Data" to the broker which will then send it back to Node_RED.

One more thing to note is the uri in the function `mqtt_app_start()` which needs to be modified every once in a while, seeing that the IP keeps changing "mqtt://MY_IP:1883".

We merged this code with the one we wrote for the HTTP part, but we modified the function `print_data()` in the latter so that it concatenates all the data we want to send in order, adding icons and in the right format, puts them in a buffer variable and sends them in a variable called "send_data".

Final Result

To run the program we just worked on, we need to have Node-RED and the Mosquitto broker running in the background through the command prompt as demonstrated below

```
Command Prompt - mosquitto -v -c test.conf
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User>cd C:\Program Files\mosquitto

C:\Program Files\mosquitto>mosquitto -v -c test.conf
1670360769: mosquitto version 2.0.15 starting
1670360769: Config loaded from test.conf.
1670360769: Opening ipv6 listen socket on port 1883.
1670360769: Opening ipv4 listen socket on port 1883.
1670360769: mosquitto version 2.0.15 running
1670360777: New connection from 127.0.0.1:57503 on port 1883.
1670360777: New client connected from 127.0.0.1:57503 as test (p2, c1, k60).
1670360777: No will message specified.
1670360777: Sending CONNACK to test (0, 0)
1670360777: Received SUBSCRIBE from test
1670360777:   Weather_Data (QoS 0)
1670360777: test 0 Weather_Data
1670360777: Sending SUBACK to test
```

```
node-red
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User>node-red
7 Dec 18:41:13 - [info]

Welcome to Node-RED
=====

7 Dec 18:41:13 - [info] Node-RED version: v3.0.2
7 Dec 18:41:13 - [info] Node.js version: v18.12.1
7 Dec 18:41:13 - [info] Windows_NT 10.0.19044 x64 LE
7 Dec 18:41:19 - [info] Loading palette nodes
7 Dec 18:41:30 - [info] Dashboard version 3.2.3 started at /ui
7 Dec 18:41:31 - [info] Settings file : C:\Users\User\.node-red\settings.js
7 Dec 18:41:31 - [info] Context store : 'default' [module=memory]
7 Dec 18:41:31 - [info] User directory : \Users\User\.node-red
7 Dec 18:41:31 - [warn] Projects disabled : editorTheme.projects.enabled=false
7 Dec 18:41:31 - [info] Flows file : \Users\User\.node-red\flows.json
7 Dec 18:41:31 - [warn]

-----
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
-----

7 Dec 18:41:31 - [info] Starting flows
7 Dec 18:41:31 - [info] Started flows
7 Dec 18:41:31 - [info] [mqtt-broker:0ad345c02312d8a3] Connection failed to broker: test@mqtt://127.0.0.1:1883
7 Dec 18:41:31 - [info] Server now running at http://127.0.0.1:1880/
7 Dec 18:41:46 - [info] [mqtt-broker:0ad345c02312d8a3] Connected to broker: test@mqtt://127.0.0.1:1883
```

We also need the esp32 to be connected, with the code uploaded on it.

Finally, all that's left is to open the Node-RED page and press the button shown in the picture below, and a few seconds later the requested data will be displayed.

