IoT Chicken Coop

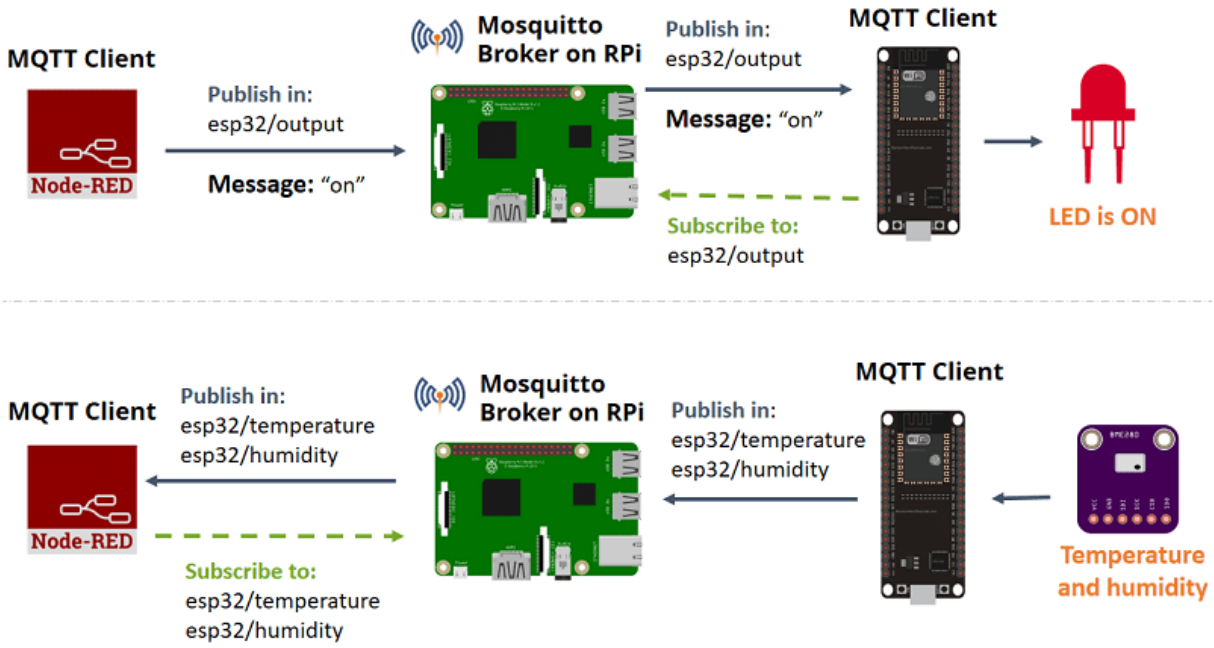
By Ruben Goovaerts IoT Student

In this paper, we will see a detailed step-by-step guide on how to make an IoT Chicken Coop project. Keep in mind that this is still a prototype, so some parts are changed to a LED instead of an actual object. (e.g The automated gate is visualized by a red LED.)

Project overview:

In this scenario, a Node-RED application on a Raspberry Pi governs the outputs of an ESP32 and gathers sensor data from the ESP32 via MQTT communication protocol. Utilizing the Mosquitto broker hosted on the Raspberry Pi, messages are routed, filtered, and distributed to relevant subscribers. The broker acts as a central hub, receiving messages, determining their recipients, and disseminating them to all subscribed clients.

The forthcoming diagram provides an overview of the tasks we will undertake in this tutorial. (In our case we will use the DHT11 sensor instead of the BME280, and will add an extra topic called esp32/light_level, that will read the BH1750 sensor value)



Preparation:

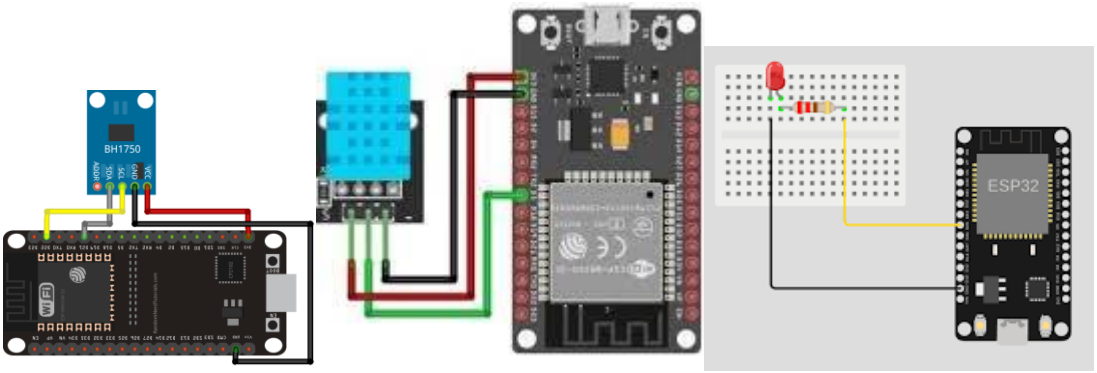
- Be familiar with the Raspberry Pi 4
- Have a Linux running on your Raspberry Pi
- Have Node-RED running on you Raspberry Pi
- Have access to the Node-RED Dashboard
- Be familiar with MQTT (Mosquitto)

Items you will need:

- ESP32
- DHT11 Temperature and humidity sensor
- BH1750 light sensor
- Raspberry Pi 4
- Wire
- LED's

Schematics:

Wire the pins of the sensor to the ESP32 as such: (Beware that the pins in your code may be defined differently than the pins in the picture)



Arduino IDE preparation:

To integrate the BH1750 sensor into your project, follow these steps:

- Launch your Arduino IDE.
- Navigate to Sketch > Include Library > Manage Libraries. This action will launch the Library Manager.
- In the search bar of the Library Manager, enter "BH1750" and press Enter.-
- Look for the BH1750 library developed by Christopher Laws in the search results.
- Click on the "Install" button adjacent to the BH1750 library to commence the installation process.

To incorporate the PubSubClient library into your project, follow these steps (or simply search the exact name as stated in the #iniclude part of the code):

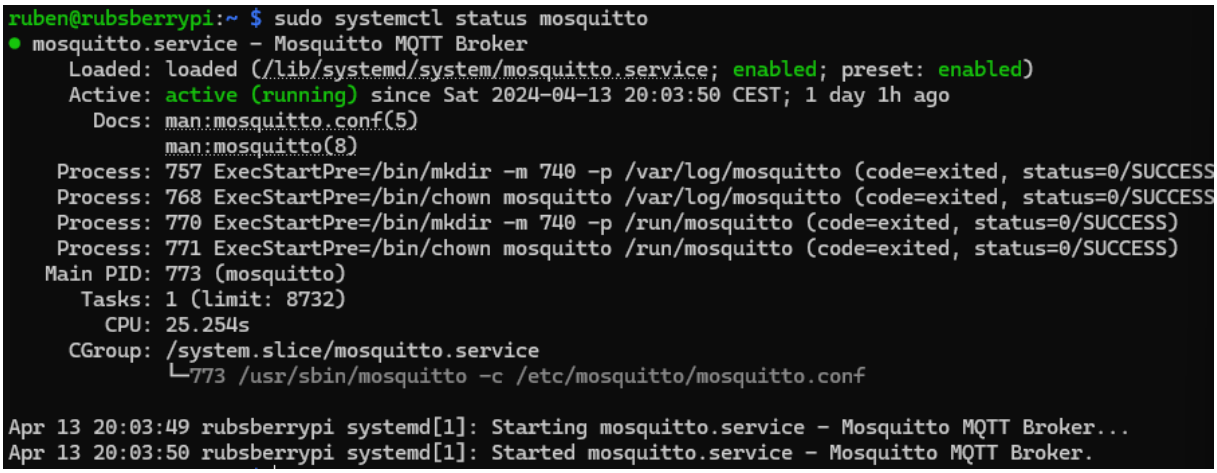
- Download the PubSubClient library by clicking here. Once downloaded, you'll have a .zip folder in your Downloads directory.
- Unzip the downloaded .zip folder, which will yield a folder named pubsubclient-master.
- Rename the folder from pubsubclient-master to pubsubclient.
- Move the pubsubclient folder to the libraries folder within your Arduino IDE installation directory.
- Finally, restart your Arduino IDE to ensure that the library is properly recognized and accessible for your projects.

Now simply upload the code for the ESP32 (provided in github) in the IDE. Line-by-line explanations are provided in the code.

Mosquitto (MQTT):

To make sure MQTT is running properly, you can check with the following command:

"Sudo systemctl status mosquito"



If everything goes right, your prompt will look like the image above. In case your MQTT is not active, try activating it with *"sudo systemctl start mosquito".* You can check your ESP32 is connected to the broker by checking in the serial monitor. It should say:

```
Connecting to GOOVAERTSBUREAU
..
WiFi connected
IP address:
192.168.0.157
Attempting MQTT connection...connected
```

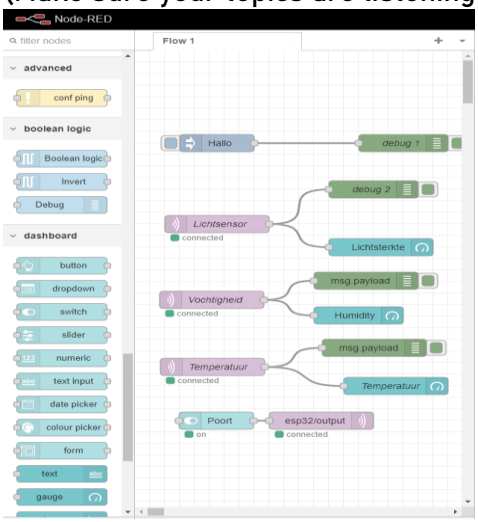
Node-Red:

After you installed Node-RED, Node-RED-Dashboard and the Mosquitto broker (the links to step-by steps guides are linked under the preparation header.)

Import the Node-RED-flow code to Options<Import<Clipboard (Flow can be found in the github files)

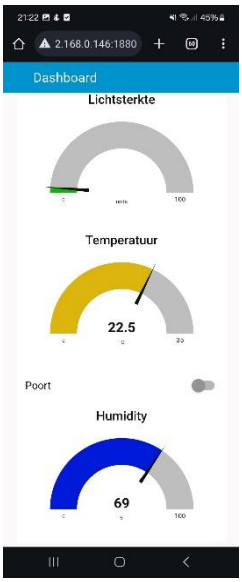
Once the flow is imported you should be able to get this (I added the BH1750 "Lichtsensor" after the float):

(Make sure your topics are listening on the right port, 1883 by default)



Once all the sensors, topics, visualizers and payloads are connected, you can see your sensor values connected as such: (Node-RED-Dashboard can be found using http://Your_RPi_IP_address:1880/ui)

Dashboard should look something like this (Depending on the type of visuals you selected to use):



(For this project I used some Dutch terminology, as this project is originally made in Dutch)

In the Arduino IDE serial monitor you should be able to see the topic messages, weather the switch(gate of the coop) is opened or closed (LED is either on or off), and the sensor values.

```
Temperature: 19.60
Humidity: 46.00
Light Level: 0.83
```

These are the values that will be sent to the broker, and sent to the Node-RED dashboard.

```
Message arrived on topic: esp32/output. Message: off
Changing output to off
```

When toggeling the virtual button on the dashboard, you will get a message from the topic in the serial monitor in your IDE.

Congratulations! You now have completed the project. Feel free to add as many sensors as you want. ☺