EE 250 Final Project: Autonomous Temperature and Humidity MQTT System

Project Description

The purpose of this project is to create an autonomous IOT system that can inform the temperature and humidity of VHE 205 to any incoming person. More specifically, the LED will light up and display the temperature and humidity, and any subscripted laptop (presumably the incoming person's device) will also display the pair of data. The main topics of focus are MQTT and the Grove Pi sensors. The block diagram for the system is shown in Figure 1 on page 2.

Components, Protocols, and Data Processing

On the publisher's side, an RPI with Grove Pi components are employed. The DHT11 sensor is used for temperature and humidity data collection, the Ultrasonic Ranger will be directed towards the door and used to measure the distance between the podium and an incoming person, a LED will flash in certain patterns to indicate the system status, and the LCD will display whether there is a person in range or not, if so, the temperature and humidity data will be shown. To run the publisher block, the paho.mqtt, time, grovepi, datetime, and socket libraries are required. All data processing is completed with this publisher RPI. Temperature, humidity, and distance datasets are collected every second. However, data are only sent to the broker and displayed on the LED and subscribe device if a person is 70 cm away from the podium.

For the subscriber, the paho.mqtt library is required, and is subscribed to the channels of 'temp' and 'humidity.' Once a data pair is received, it will be displayed on the terminal. Since the eclipse USC server is down, in order to set up the MQTT, a second RPI is employed as the broker, which is connected to the same hotspot as with other devices. We have set the client.connect() function to connect to the broker RPI IP address, which intermediates data transfer.

Reflection: Limitation and Lessons Learned

The limitation for our system is that the distance measurement is most accurate when it is below 100 cm. In addition, the broker RPI will go to sleep if the program has not been ran in a while, so the user may have to compile the subscriber program (project_sub.py) twice. Most importantly, all devices must be on the same network for this system to work.

The most important lesson that we learned in this class is to efficiently leverage MQTTs and use RPI as brokers whenever a default server is unavailable. We also learned the modules that are required for us to implement RPI as a broker.

Project Block Diagram:

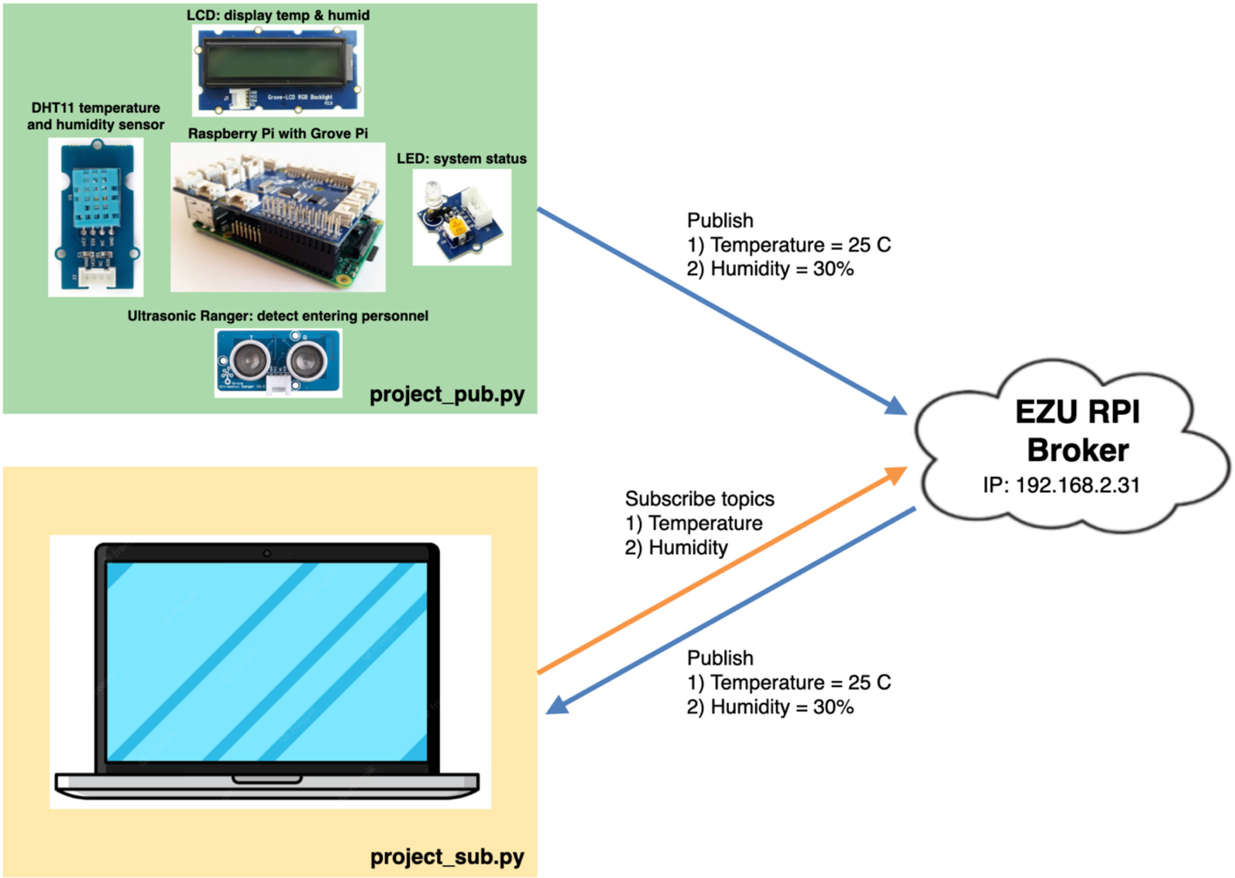

Figure 1. Block diagram for the Autonomous Temperature and Humidity MQTT System