MQTT Connection with an ESP32 to Adafruit IO and Paramiko-PAHO

Carlos Ignacio Elias Contreras
Prof: Luis Emilio Tonix Gleason

October 28, 2025

1 Project Objectives

The following objectives were defined for this project, based on the course's requirements:

1. Establish a robust local communication channel using the MQTT protocol with the Mosquitto Broker.
2. Develop a Python script to consume data from the MQTT topic and log the connections and message metadata into a CSV file for later analysis.
3. Implement a Publisher application using the provided code base to send simulated telemetry data (temperature and GPS coordinates) to a remote cloud platform (Adafruit IO).
4. Visualize the received data on an Adafruit IO Dashboard, specifically plotting the geographic coordinates to form a star shape over the Jabil building in Guadalajara, alongside displaying the randomly generated temperature.

2 Project Description

This project involves setting up a complete Internet of Things (IoT) data pipeline, starting from a local MQTT broker to a remote cloud visualization dashboard. The core components include the Mosquitto broker for local messaging, a Python client for data logging, and an adapted publisher script for communication with Adafruit IO.

Repository Link

The code used for the publisher and subscriber clients is available in the following repository (Link to be added after push):
¡INSERT REPOSITORY LINK HERE¿

3 Procedure and Implementation

A. Local MQTT Broker Setup (Mosquitto)

This stage focused on establishing the core local communication channel. The Mosquitto MQTT Broker was used, running on the local machine (or a local server), acting as the central message hub. This setup ensures that both the Python logger and the final publisher application can reliably exchange data.

- **Tool Used:** Mosquitto (Local Installation).
- **Configuration:** Default port 1883 was utilized for unsecured TCP communication.
- **Test:** The broker's functionality was confirmed by running the built-in Mosquitto test client subscriber/publisher utilities.

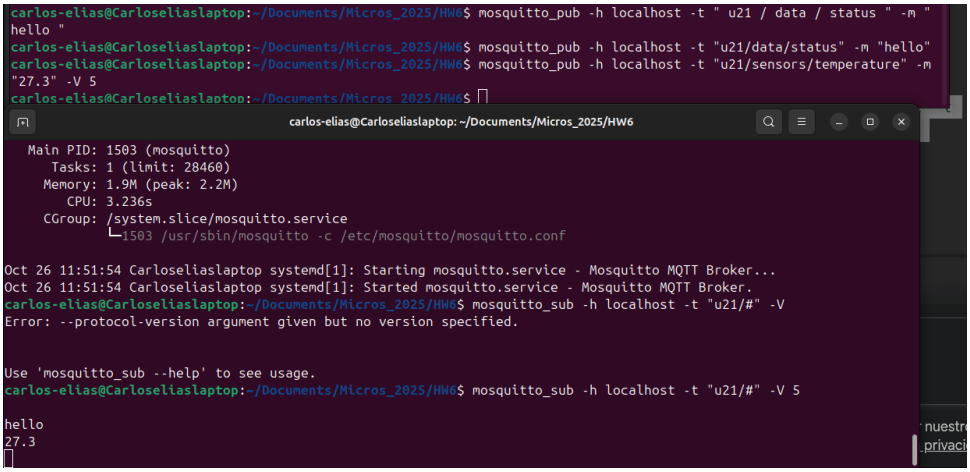

Figure 1: Screenshot of the running Mosquitto Broker and test connection.

B. Python Subscriber for CSV Data Logging

A Python script was developed based on the provided assignment code to subscribe to a specific MQTT topic. The primary function of this script was to monitor the network for message activity and log connection events into a CSV file.

- **Library Used:** Paho-MQTT (Python client).
- **Logging Mechanism:** The script captured timestamps, topic information, and connection status every time a message was received or a client connected/disconnected.
- **Output:** The data was written to a file named ‘log_conexiones.csv‘.

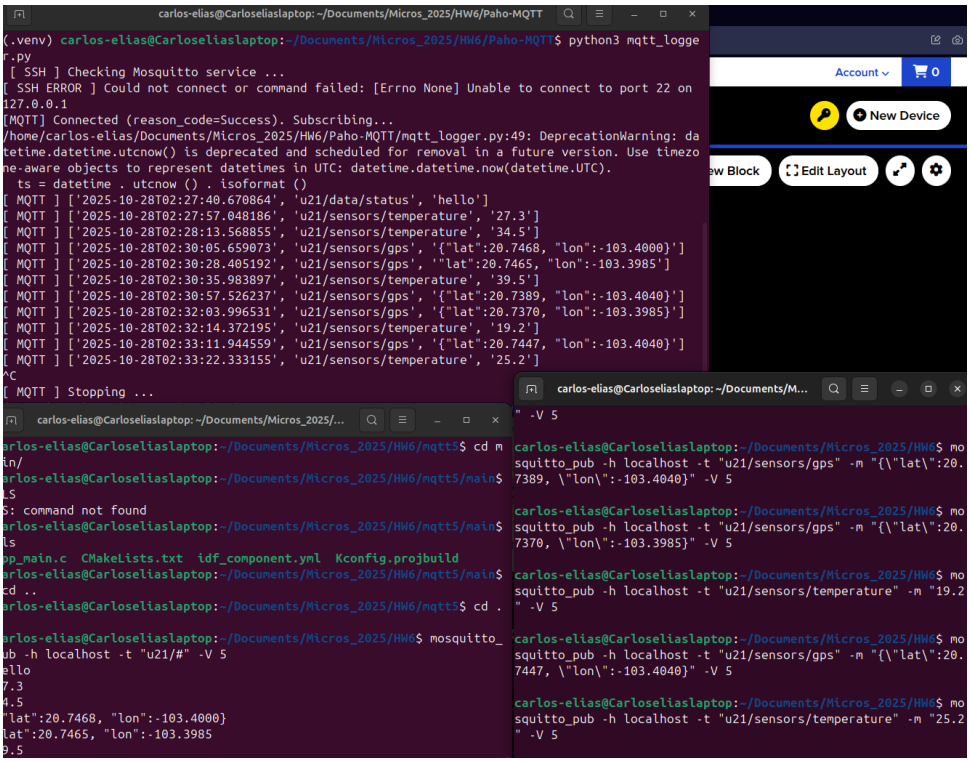

Figure 2: Screenshot showing the Python script running and the contents of the generated CSV log.

C. Adafruit IO Visualization and Star Coordinates

The final step involved adapting the publisher code to send telemetry data to the Adafruit IO cloud platform. This required configuring credentials (username and AIO key) and defining the specific geographic logic.

- **Platform:** Adafruit IO.
- **Data Feeds:** Dedicated feeds were created for GPS coordinates and temperature.
- **Geographic Simulation:** The code was configured to cycle through five specific latitude and longitude pairs designed to form a five-pointed star shape centered over the Jabil building in Guadalajara, Jalisco. This was combined with randomly generated temperature values.
- **Visualization:** An Adafruit IO Dashboard was used to display the coordinates on a map block (showing the star formation) and the temperature on a gauge block.

Created at	Value	Locat
2025/10/27 05:52:23PM	20.6748,-103.3525	
2025/10/27 05:52:18PM	20.6736,-103.3440	
2025/10/27 05:52:12PM	20.6840,-103.3801	

Figure 3: Adafruit IO Dashboard showing the randomly generated temperature feed.

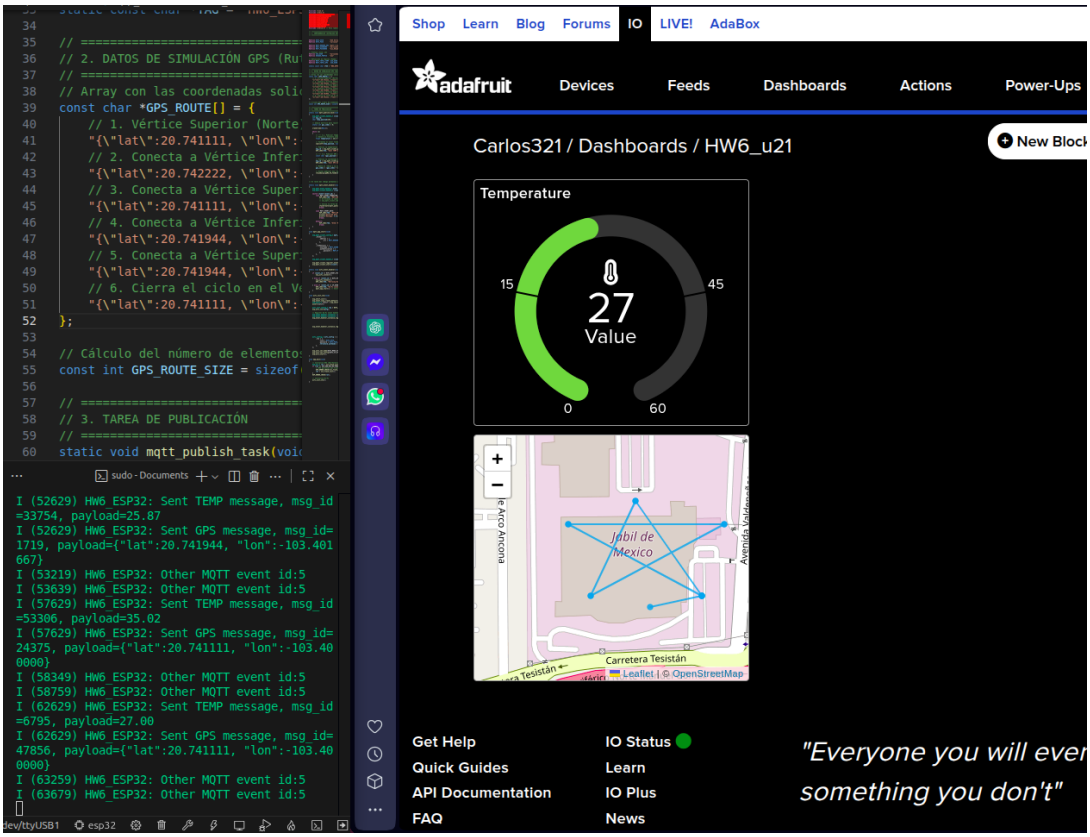

Figure 4: Adafruit IO Dashboard Map showing the five points forming the star over Jabil, Guadalajara.

4 Conclusion

The project successfully demonstrated the implementation of a complete and layered IoT communication system. The objectives were fully met by establishing local MQTT connectivity with Mosquitto, ensuring data integrity through a Python-based CSV logging utility, and finally, pushing the data to a remote cloud for visualization. The creative requirement of plotting a star shape over a specific geographic location in Guadalajara on the Adafruit IO map was successfully executed, validating the control over the data payload and its presentation layer. This exercise provided practical experience in handling asynchronous messaging, data logging, and cloud service integration vital for industrial and personal IoT applications.