Contents

[Project Overview 1](#_Toc156232056)

[My Progress 2](#_Toc156232057)

[Next Steps 6](#_Toc156232058)

[Barriers Encountered 7](#_Toc156232059)

Edgar Muniz

RPI 2025

Electrical Engineering

# Project Overview

The Mercer X Lab project is a collaborative project aimed at provoking interest in the new and renovated Mercer X Lab. This innovative project comprises two main components: an autonomous greeter robot and interactive signs strategically placed throughout the center.

**Autonomous Greeter Robot:**

The primary objective of the autonomous greeter robot is to engage with students and guide them to the Mercer X Lab. To accomplish this, the robot is equipped with navigation capabilities to traverse the Jonsson Engineering Center autonomously. By seamlessly interacting with students, the robot serves as a friendly guide, fostering a welcoming atmosphere and facilitating easy navigation to the lab.

**Interactive Signs:**

Complementing the efforts of the greeter robot, interactive signs play a crucial role in providing valuable information about the Mercer X Lab. These signs are strategically placed to capture the attention of passersby and generate interest in the lab's offerings. Key features of the signs include speech capabilities, LED matrices displaying dynamic content, and real-time updates on the location and activities of the autonomous greeter robot within the Jonsson Engineering Center. The goal is to create an engaging and informative environment that showcases the lab's significance and encourages exploration.

**Contributions:**

My contribution to this project focused on the electronics and communication aspects of the interactive signs. This included setting up a Mosquitto server for MQTT protocol usage and deploying various services like NodeRED as Docker images within a Linux environment. Additionally, I coded for Adafruit's Matrix Portal M4 co-processor board, enabling the display of dynamic images on LED matrices. Lastly, I collaborated on developing code for the ESP8266 NodeMCU to empower the signs with the ability to send and receive data via the MQTT protocol.

In summary, the Mercer X Lab project brings together robotics, autonomous navigation, and interactive displays to redefine the visitor experience within the Jonsson Engineering Center. By combining the capabilities of an autonomous greeter robot with engaging interactive signs, the project aims to not only guide students to the lab but also spark curiosity and interest in the lab's offerings.

# My Progress

Server:

The Mosquitto Server has been setup in addition to NodeRED. Both of these run inside of docker containers which makes implementation fairly easy. I chose to include NodeRED because it’s an extremely useful flow based programing tool for IoT applications. In the event that data needs to be tracked or saved by some means other than the existing electronics, all that would need to be done is create and configure a new node. It also makes debugging simple as you can see all incoming and outgoing traffic to different nodes.

LED Matrices and Adafruit Matrix Portal M4:

The code for the Adafruit Matrix Portal M4 is nearly complete. To begin, the matrix portal can cycle through pre-saved gifs that are in the matrix portals on board storage. The time in between gifs can be configured to whatever is desired.

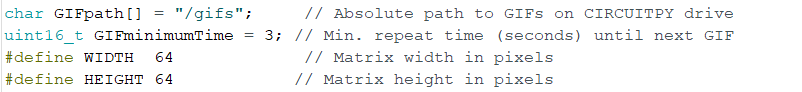


Figure Demonstrates the folder path containing the gifs as well as the variable used to track time between gifs.

The code is also written to allow the matrix portal m4 to connect the internet.

A white screen with black text

Description automatically generated

Figure demonstrates an initialization which allows the on-board esp32 processor to be utilized.



Figure Shows the storage and the RPI netreg SSID

A screen shot of a computer code

Description automatically generated

Figure Shows the setup code that connects the on board esp32 to the internet

Internet connection allows the matrix portal to receive and also send data via the mosquito broker server. In my own testing, the code is able to send out data to the broker server while simultaneously displaying and cycling through gifs.

A screen shot of a computer code

Description automatically generated

Figure Illustrates the use of the mqtt client object in order to connect the esp to the server

Therefore, the LED matrix functionality is nearly completed except for a few things that will be mentioned later in this document.

ESP 8266:

Similar to the Matrix Portal M4, the code written for the esp8266 gives the esp internet connection, as well as the ability to publish and receive data from the broker server.

A computer code with text

Description automatically generated with medium confidence

Figure RPI NetReg SSID storage for ESP8266

A screenshot of a computer code

Description automatically generated

Figure MQTT connect for esp8266

A screen shot of a computer code

Description automatically generated

Figure Demonstrates an example of publishing data using the esp8266

Possible small signs (UI):

A blueprint of a building

Description automatically generated

# Next Steps

* The current code for Adafruit matrix functionality involves cycling through pre-downloaded gifs/images in a specified folder. It would be beneficial to expand its capabilities to incorporate other types of data, such as weather information, and display them on the matrices. The code is also designed to support 32 x 32 matrices, but there's a provision for 64 x 32 matrices. To enable the latter, it's necessary to solder together two address pins on the Matrix Portal M4. The matrix portal will also need to be re-registered with RPI-NetReg. The MAC address can be found by running the m4\_gif\_test code and opening up a serial connection within the Arduino environment. It would also be beneficial to make the code more memory efficient in order to avoid an complications when incorporating more complicated functions in addition to the gif display and mqtt publish.
* The ESP 8266 is intended for use with the big signs. Therefore, they will all need to be re-registered with RPI Netreg. Similar to the matrix portal, there will be code provided to get each module’s MAC address. In addition, publishing capabilities are setup for both the esp8266 and the matrix portal. However, the subscription function is not yet complete and is a tad more involved than simply publishing data.
* As for the overall project, Nikolas built the support legs for the big signs. Therefore, the rest of the frame for the signs would need to be completed. Electronics would then need to be mounted, and cables would need to be cable managed. We have purchased power supplies, and the only other thing we would need is one Raspberry Pi to be purchased for the big signs to serve as the server for the entire project. Nikolas was also working on writing code for the servo motors as well as the individual 7-segment display modules. To the best of my knowledge, this was not completed and would therefore need some work.
* If the small signs are to be pursued, then both a backend and frontend would need to be developed. In addition to building the frames for these small signs. Computer power could be as simple as a raspberry pi zero. As the 512 MB of ram should be plenty.

# Barriers Encountered

* One of the main barriers that was encountered was the navigation of Adafruit’s documentation. Adafruit mixes their tutorials & product documentation which could be beneficial for someone new to these types of projects. However, for someone who is simply looking for pin-outs and library documentation, it makes navigation 10 times harder. A large chunk of time was spent figuring out how to initialize different functionalities on the matrix portal board, such as co-processor communication.