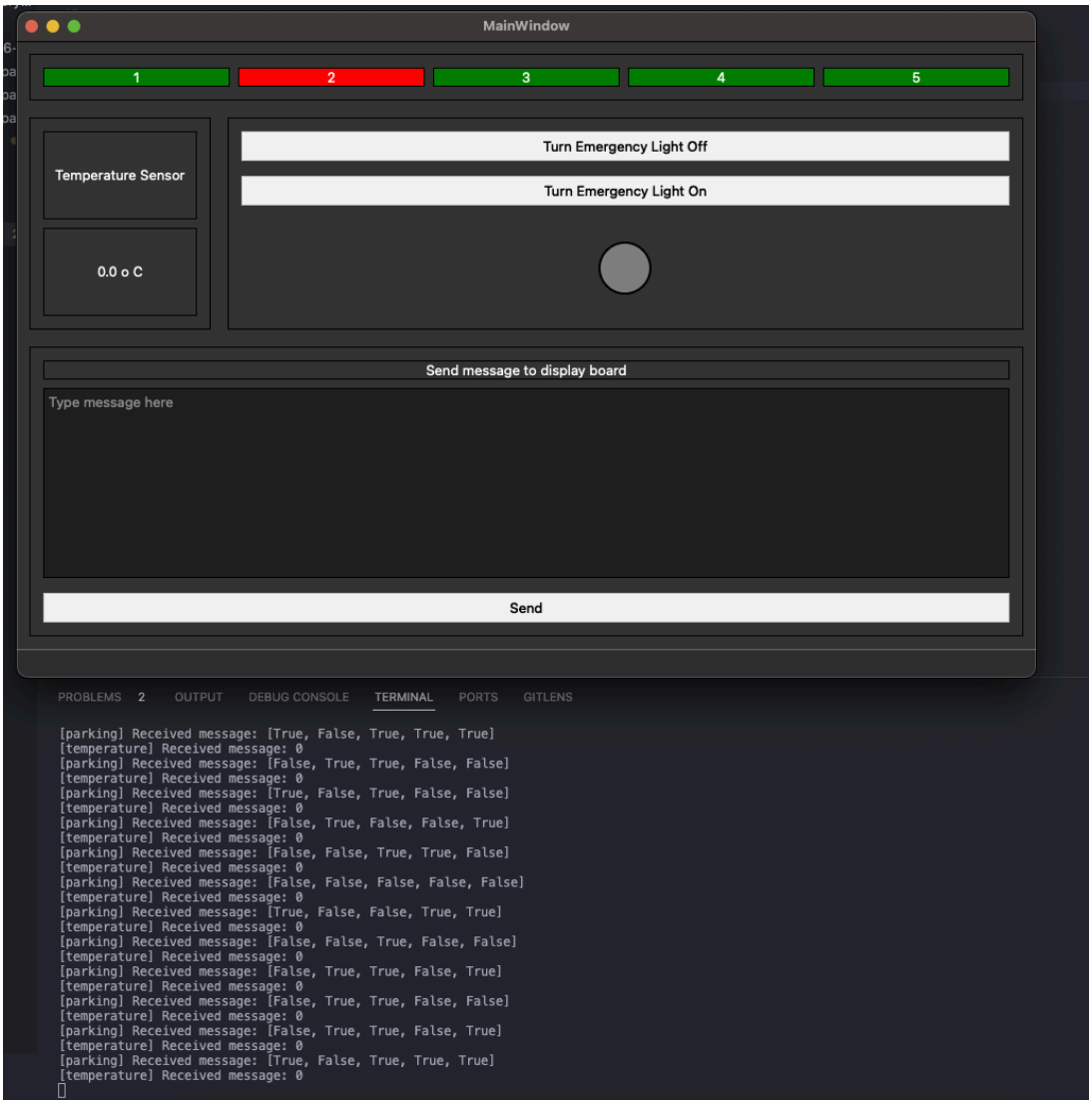
CME466 Lab 03
Vince Ruiz
11274971
vmr846

Code + Output

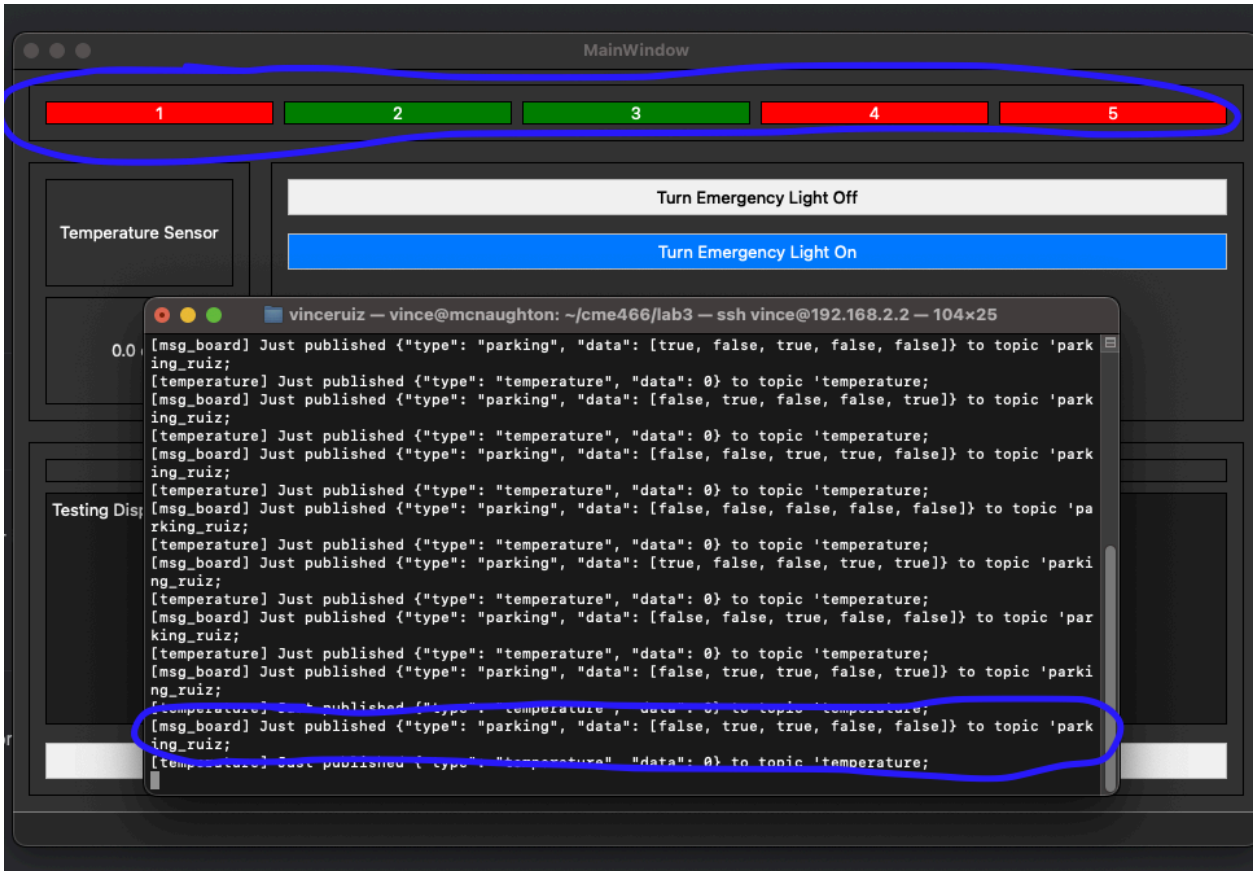
General

My system consists of two clients: PC and Pi. The PC client runs the GUI, publishes the emergency light signal, and publishes the display board message. The Pi client handles the sensing, subscribes to the emergency light topic (called “emergency”) and the display board topic (called “msg_board”), and publishes the temperature sensor value and parking status (to which the PC client subscribes). While the GUI was designed using Qt Designer, most of the GUI code was refactored manually for better organization, consistency, and functionality with styles and events. Please refer to the provided .py files for code implementation details.



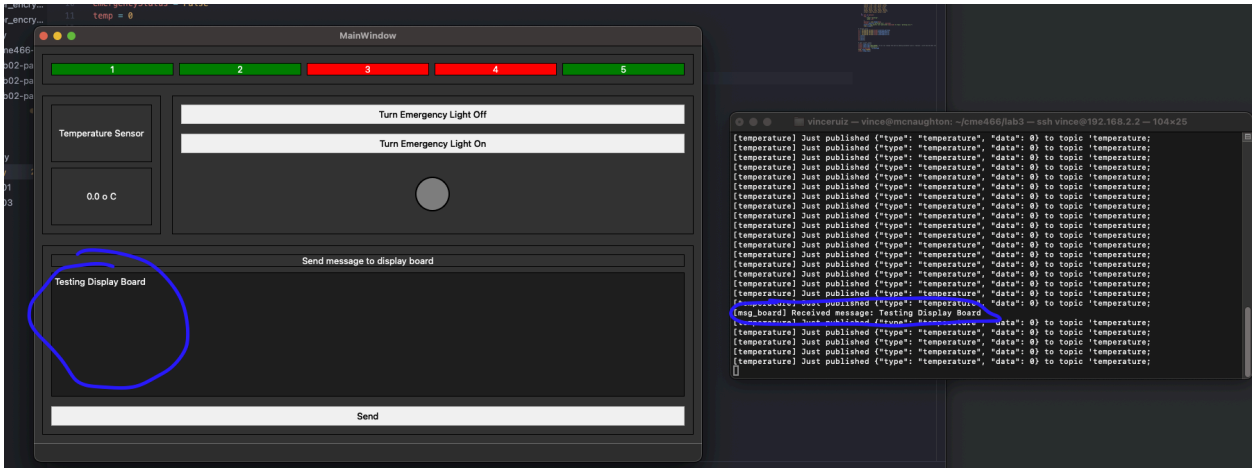
Parking Simulation

The Pi client simulates the parking status using a hard coded dataset, where a new set is published to the “parking” topic every few seconds. The PC client subscribes to this “parking” topic and updates the stylesheet of the parking boxes to either display a red background (for taken) or green (for vacant).



Display Board

The PC client simply obtains the value of the messageBoardInput QTextEdit object and publishes its value to the “msg_board” topic. The Pi client subscribes to this topic and prints its contents to the terminal.



Emergency Light

The PC client handles turning the emergency light on/off using two buttons on the GUI. A preview of the emergency light status is also shown as a circle. The light preview works by running a thread that continuously reads an emergencyStatus variable. If the variable reads True, the circle blinks between red and green every 0.25 seconds. When the buttons are pressed, an appropriate helper function sets the emergencyStatus variable either True or False, and activates another function that handles publishing the status signal to the “emergency” topic. The Pi client uses similar logic to determine the emergencyStatus and blink the LED by setting the appropriate GPIO pinout (HIGH/LOW) every 0.25 seconds if emergencyStatus is True, or continuously LOW otherwise.

Temperature Sensor

The Pi client uses a thread to handle the temperature sensing and converts the adc value to ºC before publishing it to the “temperature_ruiz” topic in a separate thread that continuously publishes a new temperature reading every 5 seconds.

Libraries Used

- smbus - for I2C
- PyQt5 - for GUI
- sys - for running the pc client program
- RPi.GPIO - for GPIO interaction
- time - for delay using time.sleep()
- threading - leveraging parallel computing to run both light, display board, temperature, and mqtt functions simultaneously
- math - to convert the adc value to ºC
- paho.mqtt for mqtt communication across clients
- json to format payload