ECE Laboratory

**DREXEL UNIVERSITY**

**To: Dr. Peters**

**From: Ehi Simon**

**Re: ECE 304 Lab 5 - Web Server Creation Using Flask**

**PURPOSE:**

The purpose of this week’s lab is to develop basic web server using Python and Flask, develop basic HTML templates and place my local computer and ESP32 on my Local Area Network.

**Discussion:**

**A picture containing electronics, electronic engineering, computer component, computer hardware

Description automatically generated**

*Fig. 1. Circuit Connection for Project 2*

The circuit for the lab was built like the one above. It consists of 2 330 resistors, a red LED, a blue LED, a BME280 Environmental Sensor, an Adafruit BNO085 IMU, and an ESP32S microcontroller.

The setup of the project was very tasking. I downloaded “requirements.txt” to install all the required libraries and created a virtual environment in my project directory that I copied it into. From this directory, using the virtual environment I was able to connect to <http://127.0.0.1:5000/>. This page had my climate information. I was also able to connect to <http://127.0.0.1:5000/led>. This page had my LED command. To connect to these pages, I first had to run a python file named In\_Class\_Experiment.py with the command “$ python3 In\_Class\_Experiment.py”. For all this to work, I still had to build and upload my ESP32 program. Figures showing this process can be found below.

**A screenshot of a computer program

Description automatically generated with medium confidence**

*Fig. 2. Figure Showing Creation of Virtual Environment and Installation of Required Libraries*

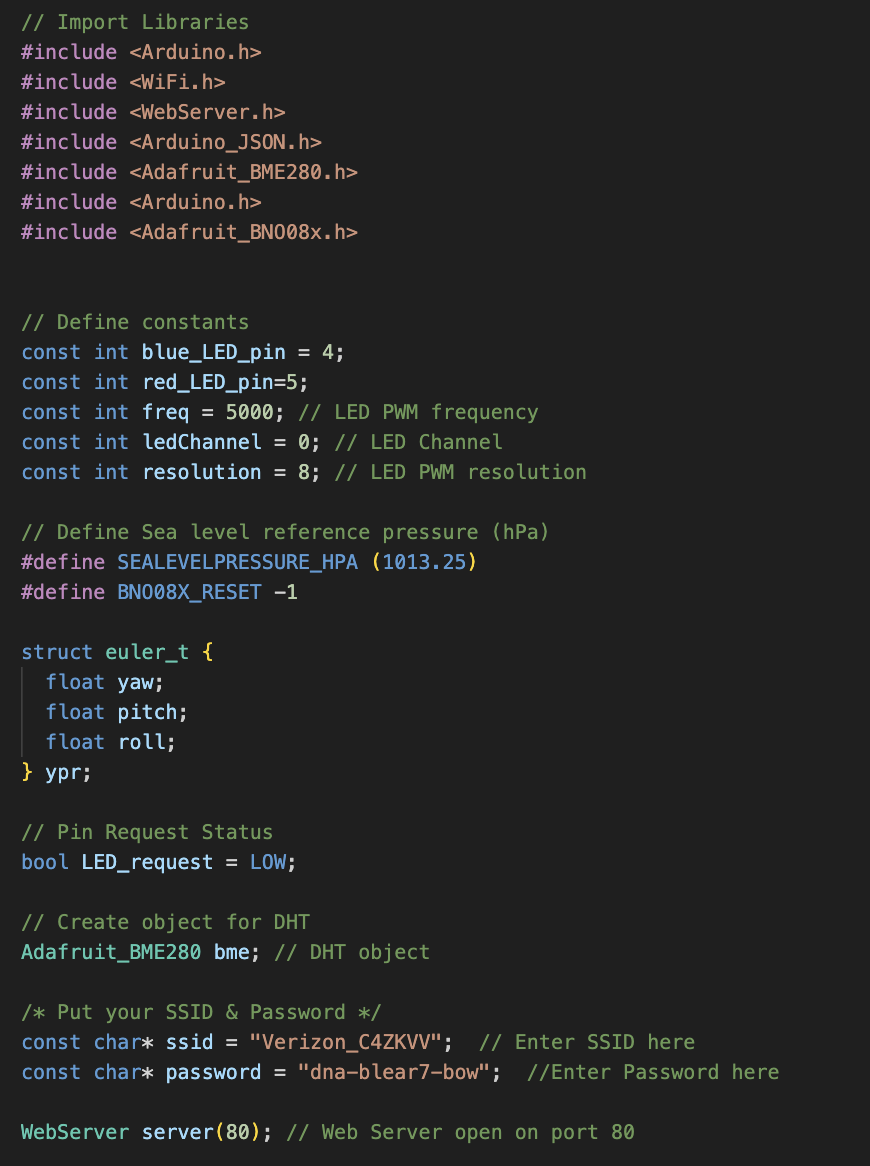
**A screenshot of a computer code

Description automatically generated with low confidence**

*Fig. 3. Figure Showing Opening and Interacting with Webpage*

**Main.cpp**

In my main.cpp file, I initialized the multiple libraries that were needed for the sensors to work and provide readings. I also included libraries to get the ESP32 to connect to STA, and libraries needed to interact with a server in JSON. I had to install the Adafruit JSON library prior. The LEDs are defined, the sea level reference pressure is defined and the BNO08X chip is reset. The BME object is created and the network SSID and password. The web server is then opened on port 80. This can all be found in the figure below.



*Fig. 4. Figure Showing Initialization of Libraries and Variable Definitions*

A picture containing text, screenshot, software

Description automatically generated

*Fig. 5. Figure Showing Setup for Yaw, Pitch, and Roll Values*

The figure above shows the setup to obtain the yaw, pitch, and roll. The BNO08X chip is reset, and all the necessary initializations and calculations are performed to obtain the values.

In the figure below, I modified the handle\_OnConnect function to include the yaw, pitch, and roll values in the doc under the JSONVar class. This class also contains the temperature, humidity, altitude, and pressure taken from the BME 280 sensor. I also have a set\_LEDs function that interacts with the server and uses the client’s response to update the status of the LEDs.



*Fig. 6. Figure Showing Handle\_OnConnect Function and set\_LEDs Function*

The figure below shows the setup and loop functions. This setup function does more than those in previous projects. It sets the LED pin modes. It also initializes the BME280 sensor and the BNO08X chip. It then connects to the local Wi-Fi network and turns on the server. The setup function makes it possible to open the webpage and GET or POST from the page. The loop function simply keeps handling the client.

A screen shot of a computer program

Description automatically generated with low confidence

*Fig. 7. Figure Showing Setup and Loop Functions*

The final result consists of multiple webpages and changes from GET/POST requests. They can be found in the figures below.

A picture containing text, screenshot, design, algebra

Description automatically generated

*Fig. 8. Figure Showing Climate Information Webpage*

*A screenshot of a computer

Description automatically generated with medium confidence*

*Fig. 9. Figure Showing LED Commander Webpage*

**Conclusion**

In this lecture, I learned about Flask, which is a lightweight web application framework in Python. I discovered the basic components of a Flask implementation, such as initializing an application instance, mapping URLs to Python functions, and starting the server. I also set up my computer by obtaining the IP address and creating a "templates" subfolder. The lecture demonstrated how to create dynamic web pages using Flask and interact with the ESP32 device. Overall, it was an informative session that provided hands-on experience with web server creation using Flask.