ECE Laboratory

**DREXEL UNIVERSITY**

**To: Dr. Peters**

**From: Ehi Simon**

**Re: ECE 304 Lab 8 – Node Red**

**PURPOSE:**

The purpose of this week’s lab is to learn the basics of nose red and nose red dashboard. It was also to create a functional node red page that monitors/controls both of your circuits via the graphical user interface using Node Red Dashboard.

**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 first had to install the node.js software to my mac. After this, I had to install node red via my terminal. Once I had my node red installed, I entered the directory and installed mode red dashboard. A figure showing all the terminal commands I used for installation for this project can be found below.

A picture containing text, font, white, screenshot

Description automatically generated

*Fig. 2. Figure Showing Commands Used to Install Necessary Software for Projects*

**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:

A screen shot of a computer program

Description automatically generated with low confidence

*Fig. 3. Figure Showing Initialization of Libraries and Variable Definitions*

A picture containing text, screenshot, software

Description automatically generated

*Fig. 4. 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. At the end of the function, the JSONVar class is sent to the server to be displayed on the webpage in JSON format.

A screen shot of a computer program

Description automatically generated with low confidence

*Fig. 5. Figure Showing Handle\_OnConnect Function*

A screen shot of a computer program

Description automatically generated with low confidence

*Fig. 6. Figure Showing Setup and Loop Functions*

The figure above shows the setup function of the file. 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.

The rest of the functions that are not shown or explained did not need to be edited from the code that was provided.

**Node Red**

Node red first has to be initialized from my computer’s terminal app. This process can be found n the figure below.

*A screenshot of a computer

Description automatically generated with medium confidence*

*Fig. 7. Figure Showing Initialization of Node Red on my Computer*

After this, it can be opened in a web browser by typing “localhost:1880”, because 1880 is the port that node red runs on. On the node web page, I was able to choose buttons and chain them. These chains were able to create webpages using information pulled from the ESP32 local server, as well as weatherapi.com. This can all be found in the figures below.

**A screenshot of a computer

Description automatically generated with medium confidence**

*Fig. 8. Figure Showing Node Red Form to Create Yaw, Pitch, and Roll Monitors*

**A screenshot of a computer

Description automatically generated with medium confidence**

*Fig. 9. Figure Showing Node Red Form to Create Environmental Monitors*

**A screenshot of a computer

Description automatically generated with medium confidence**

*Fig. 9. Figure Showing Node Red HTTP GET Request to Weatherapi.com*

**A screenshot of a computer

Description automatically generated with medium confidence**

*Fig. 10. Figure Showing Final Webpage Produce Using Node Red*

**Conclusion**

In this lecture, I learned about Node-RED, which is a flow-based development tool for visual programming. It allows me to wire hardware devices, APIs, and online services together as part of the Internet of Things. Node-RED is web browser-based and built on Node.js. The flows I create are stored as JSON. To visualize and control my project, I created a web interface using the Node-RED Dashboard. I learned how to create tabs, groups, and configure nodes to display and control data. Overall, this lecture provided me with a comprehensive introduction to Node-RED, and I gained hands-on experience in controlling devices, retrieving sensor data, creating a web interface, and integrating external APIs.