Deliverable H

Group 8

Thomas Alkhoury, Laura Godfrey, James Hight, Cecilia Lou & Julian Ward

GNG 1103

November 24, 2019

Contents

1. Introduction 3
2. Reflection 3
3. Objective 4
4. Stopping Criteria 4
5. Plans/Procedure 4
6. Results and Analysis 5
   6.1 Results 5
   6.2 Analysis 6
7. Proof of Concept 8
8. User Feedback 10
9. Conclusion 11

1. Introduction

During this deliverable, the third prototype of this project will be analyzed. This will be accomplished through reflection on previous prototypes to help improve the current one. An objective, stopping criteria, and a plan and procedure will also be established for the third prototype. The results of the third prototype will then be discussed and analyzed. Finally, a proof of concept of our prototype will be provided. This will include user feedback about the prototype in order to identify what could be improved upon that may have previously been missed. This combination of goals will ensure that the prototype is fully completed to the best of our abilities.

2. Reflection

After disappointing results from the previous prototype, there remained a lot of work to cover in this prototype. Additionally, there were some issues in effectively communicating what was actually attempted and tested during the prototyping through the deliverable. The Raspberry Pi (RPi) was mistakenly tested with a voltmeter, the results of this led the team to believe that the RPi was not producing an output. This put the project in a standstill for a couple of days, seriously delaying our progress. From Prototype 2, the team learned the importance of proper communication and ample testing. Discovering the RPi was working was an easy connection to a monitor displaying that the RPi is functioning. This would have been done had the team communicated with each other and heavily tested the RPi, not relying on one test.

In Prototype 2, we faced difficulties in connecting the RPi with Dashboard as the RPi continued to say connection failed when attempting to bind the port and the host. The issue was fixed by updating our RPi and rebooting it whenever the connection failed issue arose. Once the connection failed issue was fixed and we were able to connect RPi to Dashboard we attempted to send messages between the two coding languages. We had difficulties sending the proper replies from the RPi to Dashboard as we had included our reply within a for loop that would close the connection before sending the reply. To fix this particular issue we removed the for loop and replaced it with an if statement to avoid having the connection closed before the reply was sent.

Moving forward, in Prototype 3, the prototype will be heavily tested, and progress will be communicated. The team also plans on Prototype 3 to be a combination of different parts of the final prototype. The team currently has a functioning Dashboard code, parts of the RPi code, and a physical set up of the relay. In Prototype 3, the team plans on putting all the parts together.

3. Objective

This prototype aims to finally create a working relay module using instructions found online and with help from the TA. Testing this relay and connecting it with the already coded Dashboard was also a major focus. Once the code has been tested, work on soldering the relay module together can be started, as well as attaching the relay to the outlet that will carry current to the chosen machine. The objective of this prototype is that the machine will only turn on if the student number is listed as with training.

4. Stopping Criteria

For this deliverable, the stopping criteria must be more specific than the one described in Prototype II. It must therefore do the following;

- Register arbitrary input in Ross Video representing barcode. This will be done by manually inputting it into the system.
- Access or store a list of student IDs and their corresponding training for each student ID. This will be done by saving and editing it locally in the interface.
- Relay must activate (shut power off) when the user’s student ID does not match the required field for the machine in question. This will be done by referring to the list of student IDs and their corresponding training in the interface (by communicating with live feed).
- Many issues have occurred as a result of testing different methods and having technical difficulties with the Raspberry Pi. As a result, our verdict was to use the Raspberry Pi and wire it to the relay, therefore the relay must receive input from the Raspberry Pi.

The stopping criteria for the test was reached after a continuous 10 attempts, where the LED simulating the machine finally turned on after many reconfigurations of the breadboard setup.

5. Plans/Procedure

For this prototype we will be working on our dashboard and RPi as well as soldering our relay to a PCB. For the dashboard we have to create a pin pad for the students sign in. We will do this by connecting the buttons on the dashboard to the student number display which will show the number pressed and remove the previous number when the backspace is pressed. When the enter button is pressed the number in the student number display will be sent to the RPi. When the number is sent to the Dashboard, the function will display whether or not the machine turns on. This will be determined by the message received from RPi. If the message is “Electricity On” accompanied by the GPIO message to turn on the electricity, then the machine will turn on. If the message is “Electricity Off” accompanied by a GPIO message to turn off the electricity the machine will remain turned off. This will be displayed in the message area of the Dashboard display.

For the RPi we had to take the code from Brightspace that connects dashboard to Raspberry Pi. Then we researched a code that connects the RPi to the relay. This code takes student number input from dashboard. The code contains a list of student numbers that do have the training for machine 1. The code then checks the list for the student number and if the list contains the given student number then the Raspberry Pi and relay will turn on the electricity to the machine due to an output from the GPIO pins. If the student number is not in the given list, then the machine will remain turned off due to output from the GPIO pins. To test this part of the project we sent 10 student numbers to the dashboard and recorded the data.

Finally, for our prototype we had to solder the relay to the PCB. We made a circuit diagram for the relay and recreated the diagram on a breadboard. This way we could test it before making the final soldered board. The way the relay works is the RPi connects to the coil of the relay and sends signals telling the relay to switch on or not. The main connection for the machine is connected directly to the four pins on the relay.

6. Results and Analysis

6.1 Results

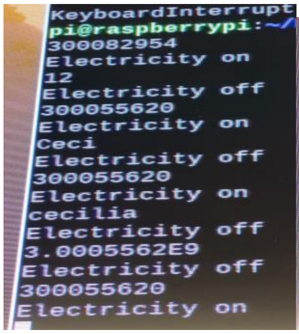
Student Number	Has Training?	Electricity
300055620	Yes	On
300082954	Yes	On
300076084	Yes	On
300102218	Yes	On
300066794	Yes	On
Cecilia Lou	No	Off
Laura Godfry	No	Off
123456789	No	Off
12	No	Off

!124&103	No	Off

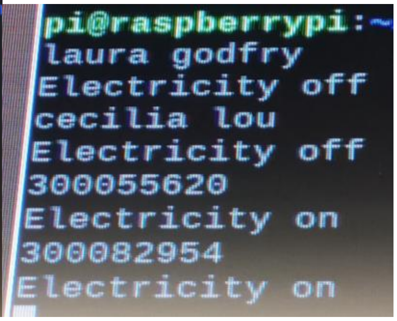
During the testing process of Prototype 3, multiple different entries were used. The purpose of this was to test the objective that the machine will only turn on if the student number is listed as with training. The stopping criteria for the test was after a continuous 10 entry accuracy. This was achieved as seen in the table above. The result can also be observed in the video attached. When an off signal is determined at the Raspberry pi, an audible click can be heard as the relay switches so that electricity will be off. The visual representation of this can be observed with the LED light that switches on when electricity is flowing and switched off when electricity is not.

6.2 Analysis

Looking back at the results, it can be reflected that Prototype 3 is functioning correctly. Prototype 3 involves a Dashboard code, a python code on Raspberry pi, and a physical aspect in terms of the relay connection. The testing conducted in this part of the project required that every aspect of the project to be fully functioning in order to produce a good outcome. By analyzing the results, we can conclude that the dashboard is successfully transmitting the student number to the Raspberry Pi. The Raspberry Pi is also differentiating whether the user has training and then sending a signal to the relay which will switch off the current flow if the user does not have training. After ensuring the functionality of the project, the next step of the project is to ensure that the aesthetics of the project will match the function. For design day, the team plans on working on the display of Dashboard and make it more user friendly, make a box to contain the plugs and wires and solder everything together. The final test will be to ensure that the project is still functioning after the soldering process.



```
KeyboardInterrupt
pi@raspberrypi:~/
300082954
Electricity on
12
Electricity off
300055620
Electricity on
Ceci
Electricity off
300055620
Electricity on
cecilia
Electricity off
3.0005562E9
Electricity off
300055620
Electricity on
```



```
pi@raspberrypi:~
laura godfry
Electricity off
cecilia lou
Electricity off
300055620
Electricity on
300082954
Electricity on
```

Figure 6.1: Proof of Testing Figure 6.2: Proof of Testing

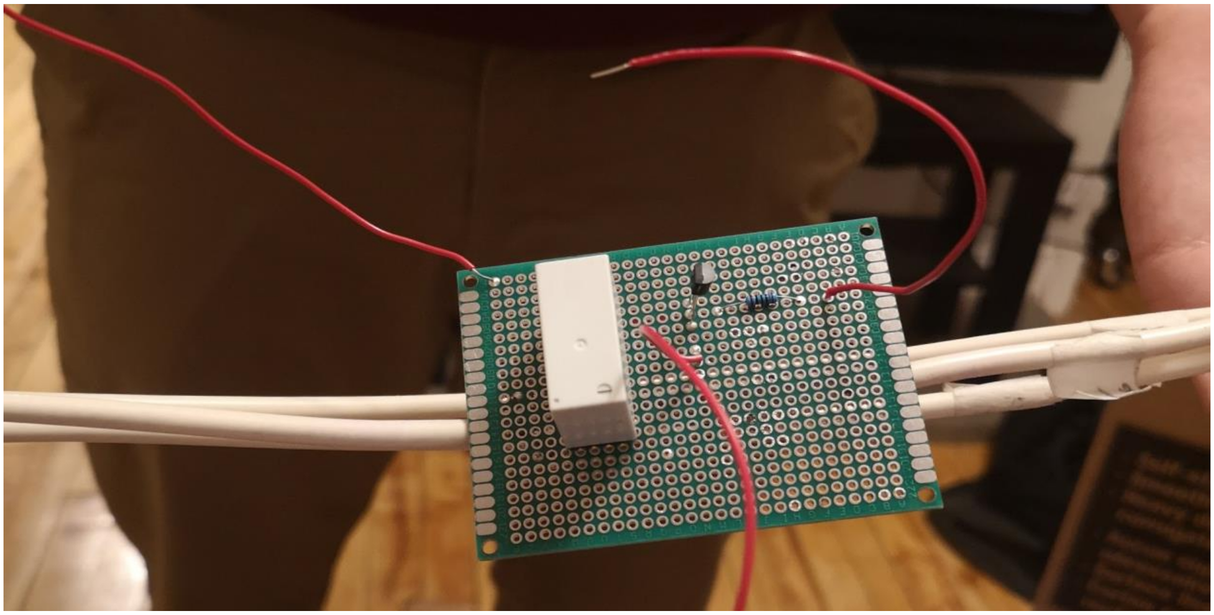


Figure 6.3: Image of PCB with Relay and Wiring Attached

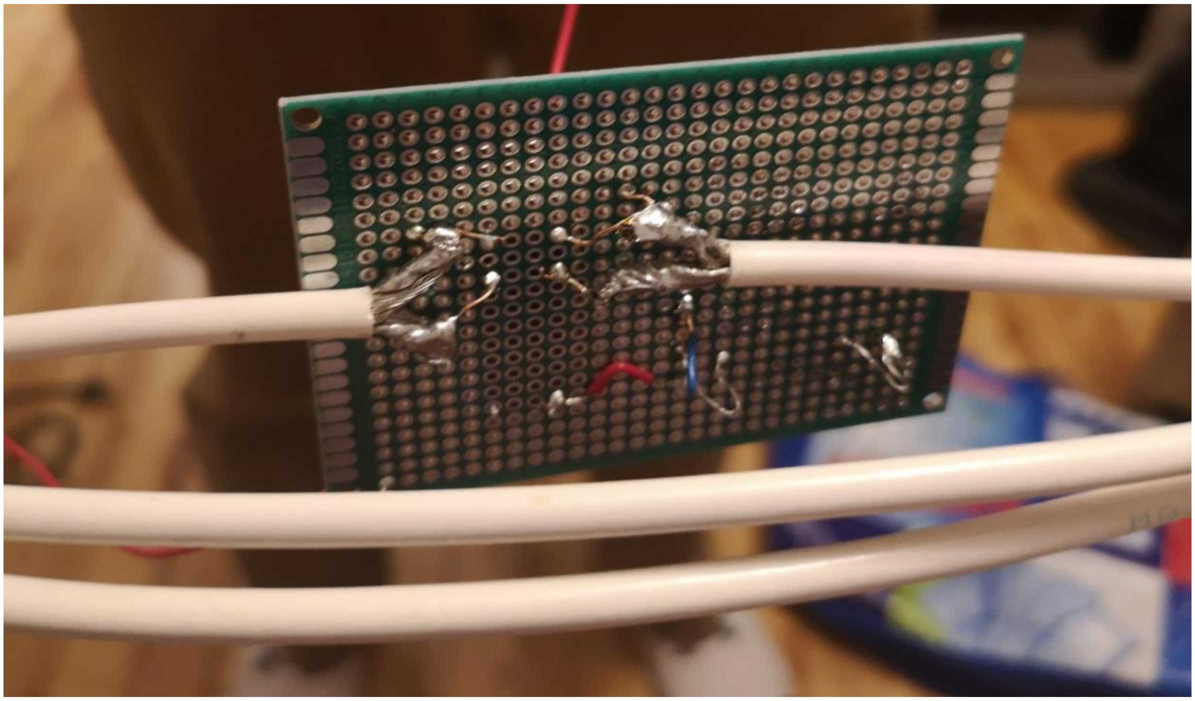


Figure 6.4: Image of PCB with Relay and Wiring Attached

7. Proof of Concept

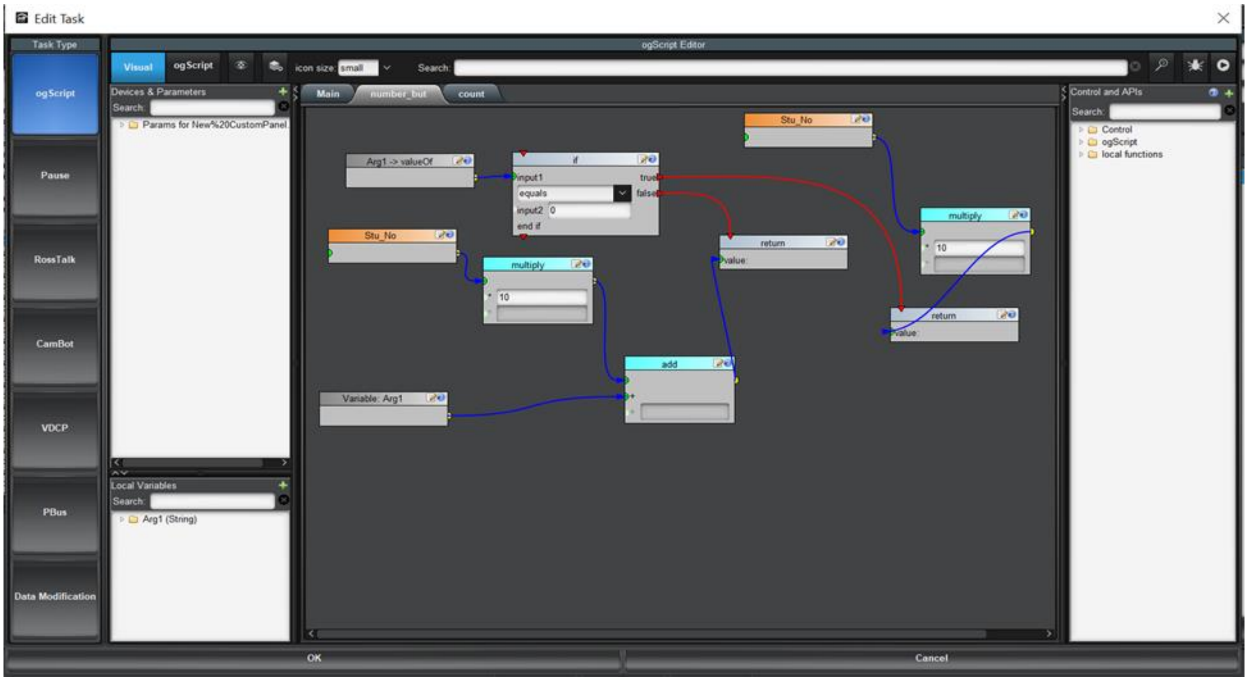


Figure 7.1: Proof of Dashboard Code

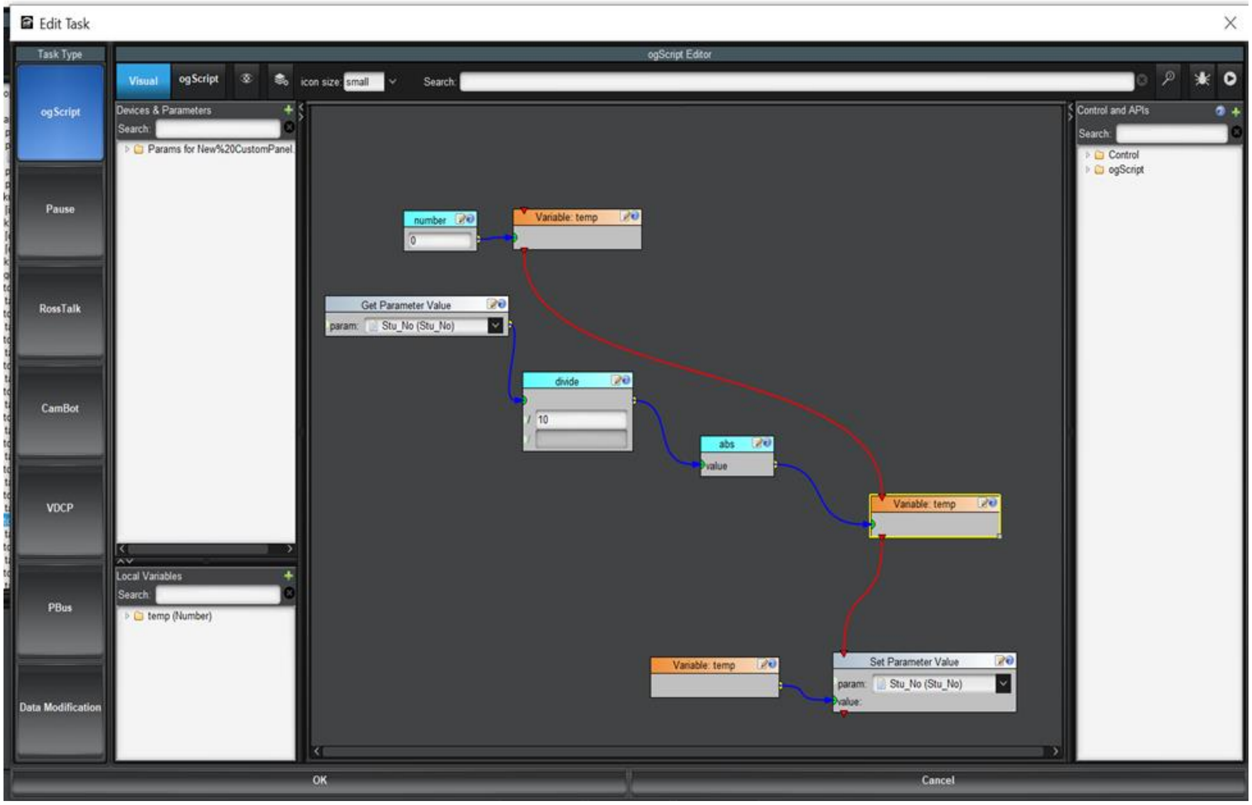
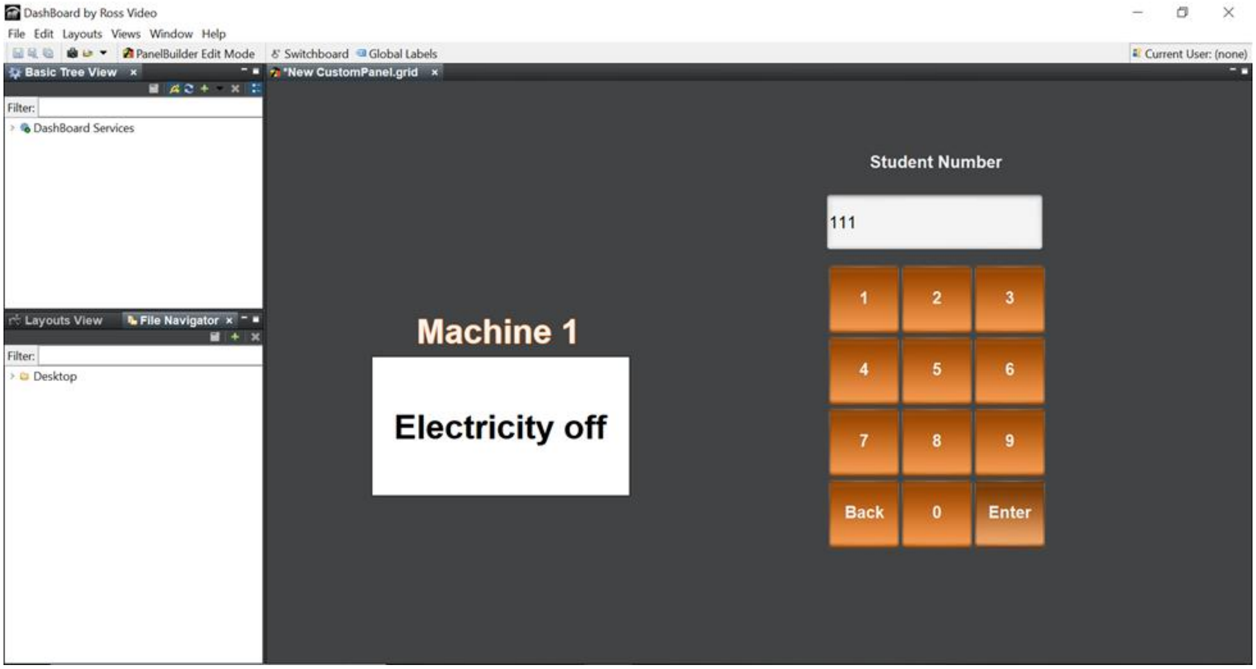


Figure 7.2: Proof of Dashboard Code



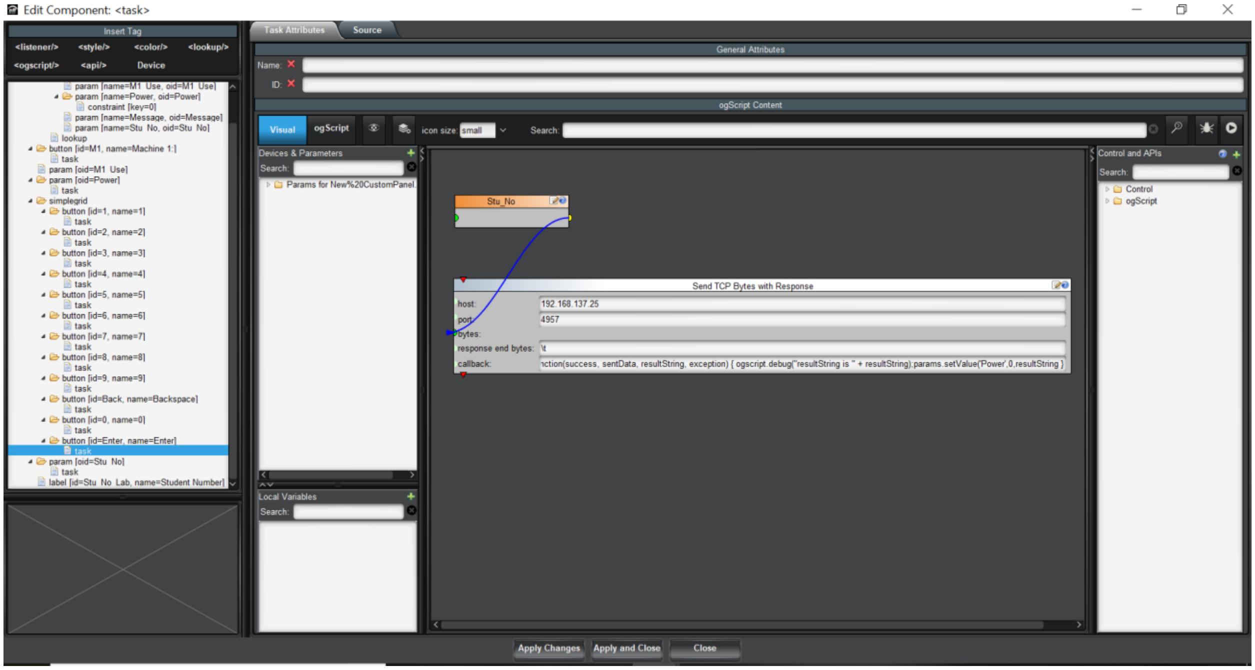
Student Number

111

Machine 1

Electricity off

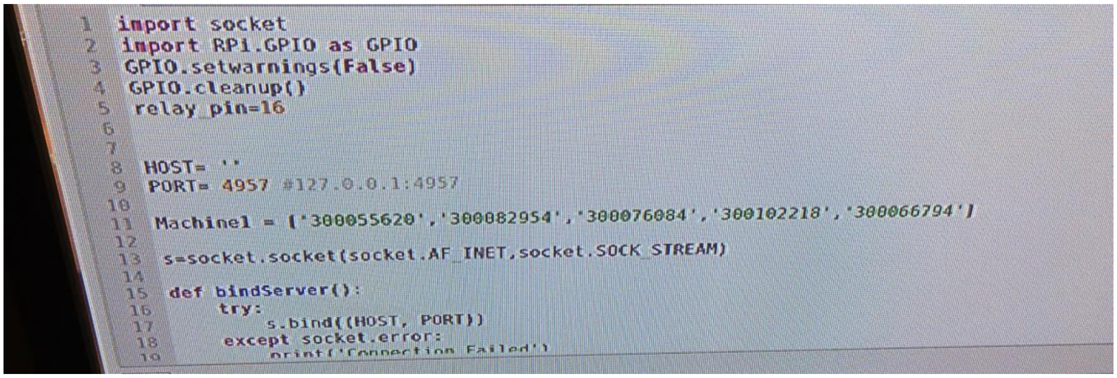
Figure 7.3: Proof of Dashboard UI



param [name=M1 Use, oid=M1 Use]
param [name=Power, oid=Power]
constraint [key=0]
param [name=Message, oid=Message]
param [name=Stu No, oid=Stu No]
lookup
button [id=M1, name=Machine 1:]
task
param [oid=M1 Use]
param [oid=Power]
task
simplegrid
button [id=1, name=1]
task
button [id=2, name=2]
task
button [id=3, name=3]
task
button [id=4, name=4]
task
button [id=5, name=5]
task
button [id=6, name=6]
task
button [id=7, name=7]
task
button [id=8, name=8]
task
button [id=9, name=9]
task
button [id=Back, name=Backspace]
task
button [id=0, name=0]
task
button [id=Enter, name=Enter]
task
param [oid=Stu No]
task
label [id=Stu No Lab, name=Student Number]

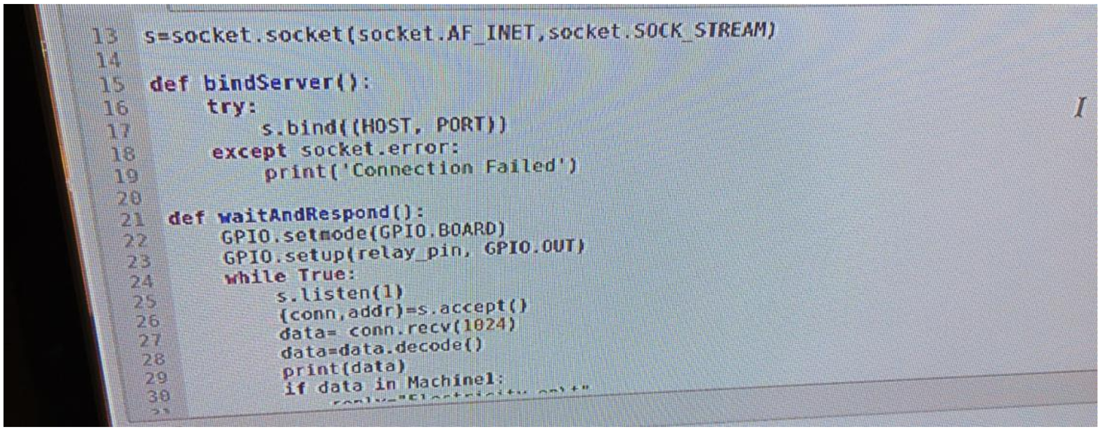
host: 192.168.137.25
port: 4957
bytes:
response end bytes: \t
callback: ınction(success, sentData, resultString, exception) { ogscript.debug("resultString is " + resultString);params.setValue('Power',0,resultString }

Figure 7.4: Proof of Dashboard Code



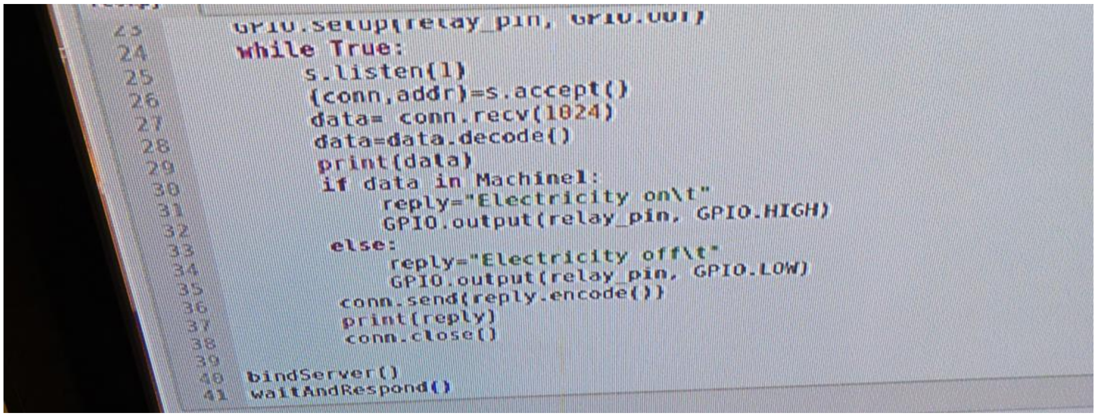
```
import socket
import RPi.GPIO as GPIO
GPIO.setwarnings(False)
GPIO.cleanup()
relay_pin=16


HOST= ''
PORT= 4957 #127.0.0.1:4957

Machine1 = ['300055620','300082954','300076084','300102218','300066794']

s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)

def bindServer():
    try:
        s.bind((HOST, PORT))
    except socket.error:
        print('Connection Failed')
```

Figure 7.5: Proof of RPi Code



```
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)

def bindServer():
    try:
        s.bind((HOST, PORT))
    except socket.error:
        print('Connection Failed')

def waitAndRespond():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(relay_pin, GPIO.OUT)
    while True:
        s.listen(1)
        (conn,addr)=s.accept()
        data= conn.recv(1024)
        data=data.decode()
        print(data)
        if data in Machine1:
            reply="Electricity on\t"
```

Figure 7.6: Proof of RPi Code



```
    GPIO.setup(relay_pin, GPIO.OUT)
    while True:
        s.listen(1)
        (conn,addr)=s.accept()
        data= conn.recv(1024)
        data=data.decode()
        print(data)
        if data in Machine1:
            reply="Electricity on\t"
            GPIO.output(relay_pin, GPIO.HIGH)
        else:
            reply="Electricity off\t"
            GPIO.output(relay_pin, GPIO.LOW)
        conn.send(reply.encode())
        print(reply)
        conn.close()

bindServer()
waitAndRespond()
```

Figure 7.7: Proof of RPi Code

8. User Feedback

We have received user feedback from Katherine Hall. She used our UI while connected to the raspberry pi. Unfortunately, we were unable to connect to the relay as it was not finished being soldered yet. Katherine was impressed with the UI and found it extremely easy to use. However, she did not enjoy the colour orange that was used on the buttons. She found it to be aggressive and “a brutal assault” on her eyes.

We will be taking this feedback into consideration and changing the button colour from orange to a potential forest green. Katherine thinks the final project will be an excellent way to control who can use the machines.

9. Conclusion

We have followed through with our objective, stopping criteria, and our planning and procedures to achieve the desired results. Our code is fully capable of sending and receiving the proper messages from the Raspberry Pi to Dashboard. The next step to be completed by design day is to complete the soldering of the final product. This will allow the relay to receive the correct messages to turn on and off the electricity that is flowing through it. Once that is complete, the product will be fully finished and ready for use.