GNG1103
Deliverable H: Prototype III and Customer Feedback

Submitted by
Erin Cox, 300051053
Carly Dawe, 300060960
James Lu, 300060342
Lynne Ngo, 300068874
Sophia Ruberto, 300052105

-

1. Introduction

In the following report, we will be discussing the third prototype as well as the feedback gathered from it. First, we will be reflecting on results from the previous prototype and determine how to improve this prototype. We will then outline our prototype test plan which will include the systems we are testing as well as the method used to test the systems. This will include an analysis of the subsystems and have a defined stopping criteria to allow us to end the test once satisfied with the results. Last we will be displaying feedback gathered directly from CEED employees. This report will contain the project's final prototype along with its test plan and feedback.

2. Results From the Previous Prototype

From the previous prototype, we learned that CEED users prefer a button style prototype, we will continue on with this UI style for prototype III and our final product. CEED users said that the button design was easy to understand and users took a small amount of time to determine the status of the printer. In order to further improve our design, we will add a small amount of text in order to distinguish status more efficiently, and also make the interface more accessible to people who are colorblind. From the previous prototype we distinguished that the motion sensor should work, in this prototype we will confirm that the motion sensor works, and we will create a functional design, as well as construct communication between our Arduino and Dashboard.

3. Prototyping Test Plan

3.1 Purpose of Test

For prototype 3, we would like to have our entire subsystem, this includes our dashboard from the previous prototype as well as a functional Arduino code and we would like to have our nodemcu connected to our dashboard with the wifi. Our entire system should work. The purpose is to ensure all of our individual components are compatible.

3.2 Test Objectives Description

-Things we want to test

- If nodemcu is able to return 0 if no motion
- If nodemcu is able to return 1 if motion
- If the arduino code runs without delay in the serial monitor
- If the dashboard is able to receive correct parameter
- If dashboard changes based on the parameter it receives
- If a user is able to understand and use our system
- if we are able to successfully connect Arduino to Dashboard

3.3 What exactly is being learned or communicated with the prototype?

- How nodemcu interacts with the motion sensor

- How nodemcu picks up information from PIR sensor
- If a listener server corrects translates information
- If our motions sensors will work

3.4 Possible types of results

- Serial monitor prints 1
- Serial monitor prints 0
- Serial monitor prints nonsense (indicating an error in the circuit)
- Dashboard doesn't receive parameter and doesn't change
- Dashboard receives parameter and doesn't change color
- Dashboard receives parameter and doesn't change color but incorrectly
- Dashboard receives parameter and doesn't change color correctly
- Listening server code works
- Listening server not working

3.5 How the results will be used to make decisions or select concepts

At the end of this prototype, we will know if all of our individual components correctly interact with each other. We will know if one of our components isn't compatible that we will have to change or adjust it. Also, we will learn how to increase the aesthetics by receiving feedback from potential users.

3.6 Criteria for test success or failure

- Success for the Arduino code is it returns the correct value for if motion is detected or not
- Failure the Arduino code is it returns incorrect values for if motion is detected or not
- The success the listener server is it is able to constantly receive proper information from nodemcu
- Failure for the listening server is it is unable to constantly receive correct information from the nodemcu
- Success for the dashboard it is able to change button color based on the information it receives
- Failure for the dashboard is it either incorrectly changes color or it has no effect when the parameters change

3.7 What is going on and how is it being done

- Prototype 2 will be a physical prototype
- Further work will be finished with the interactions between Dashboard and Arduino, a listening server will be created a tutorial will be followed from bright space
- Nodemcu code will be finished in Arduino, this will be done by researching existing codes as well as asking for opinions from experienced users of the software.

3.8 Testing process

The first step is to create the code in the Arduino ide, once that is completed we can test if it properly is detecting motion by using the serial monitor to see if it is printing 1 for motion and 0 for no motion. We test it by using our hands to create motion and no motion and see if it properly prints correctly in the serial monitor. Next, we will create the listening server and then we will use this to see if the dashboard button changes based on whether it receives parameters indicating motion or no motion.

3.9 Information being measured

3.9.1 Functional

- Response time of the Arduino code, should be the 1-second timestep its set to and not lagging

3.9.2 Non-functional

- How quickly users are able to understand and use our prototype, this is based on the time it takes them to comfortably use it

3.10 Information being observed and how is it being recorded

3.10.1 Functional

- If the listener server is able to correctly get information from the nodemcu
- If nodemcu is able to properly detect motion or not

3.10.2 Non-Functional

- New user is easily able to understand how to use the user interface and don't need to ask many questions

3.11 What materials are required and what is the approximate estimated cost?

3.11.1 Material we need to purchase

- No more materials to be purchased

3.11.2 Estimated cost

- $0, no new cost for this prototype

3.11.3 University material

3-D printer, CEED employee

3.12 Work to be done

- We will need to do research on motion sensor code involving nodemcu
- More research on pinout for the nodemcu as well as wiring it to the PIR sensor
- Create a function code for the nodemcu using the Arduino ide
- Create a listening server
- Test final prototype as a whole

3.13 Prototype test plan results

The Dashboard was successful and changes the attributes of the button. The color and text changes based on the correct parameter it receives. This change of color and text is instantaneous. The listener server was a success and is able to correctly use wifi to transfer data from the nodemcu to the Dashboard it sends correctly a 1 or a 0, we know this because we can observe the information on the serial monitor in Arduino and compare it to the information on Dashboard. The last subsection is the motion sensor and nodemcu. This is where the majority of our challenges have occurred. The subsystem correctly prints 1 or 0 based on if it receives a signal of low or high, however, the system is quite finicky occasionally it will give false data.We believe that our code is correct. The difficulty with this subsystem is the PIR motion sensor has a very difficult time actually detecting if motion is occurring. We didn't select a very good sensor, it was too cheap.

4. Prototype

4.1 Motion Sensor

In the last prototype, we did a thorough analytical analysis of the motion sensors, and comparing to specs and features we found online, along with benchmarking of other motion sensors, we had predicted that the motion sensors should work. We had also listed a list of contingency plans in case something did not work, like if the spool moved too slowly or if the motion sensor range, sensitivity and delay time was too short.

For our final prototype, we did a physical test of our PIR motion sensor connected to our Arduino NodeMCU. During our testing, we encountered multiple problems. The first problem that we faced was that we did not choose the exact board type to pick in the Arduino. So to solve this we chose the esp8266 board. We then had the problem of nothing printing in the serial monitor, and to solve this we adjusted the baud rate and we had proper "1" and "0" print. Our final and biggest problem was that we did not have enough voltage/power to power our PIR motion sensors. The PIR motion sensors we have, require a minimum voltage of 5V to run. Our initial idea of plugging the wires into the 3.3V input pin and Vic input pin did not work because both inputs produced only 3.3-4.5 V. We then bought an external battery pack but that too only produced 4.5V. We finally fixed this by buying a 9V battery which pumped out enough power to supply the motion sensors, giving us the "1" and "0" variations in the serial monitor. So now in our final prototype, we have our Arduino NodeMcu, connected to our PIR motion sensors, that are both supplied by an external 9V battery. As for the Arduino code, we did not change much code-wise, but we did ask experts to check for potential code errors.

Our objective for this prototype was to very if our code works with experts, then determine if there is motion or no motion could be detected through hand movements, and verify if this is translated in to "1" and "0" code. And for this prototype, we have consulted countless experts to fix errors or adjust codes for the project as a whole to run better and correctly.

4.2 Dashboard Interface

The dashboard is similar to what it was in the previous prototype. In this prototype, the text was added that will change alongside the color of the button. Both the button color changing code and the text changing code are fully functional. The codes are very similar, they both receive information from both the node MCU and the manual buttons that the CEED employees can press. Multiple nested if statements are used in order to effectively change the color and text.

Our overall goal for Dashboard was to successfully take in the "1" and "0" that the Arduino is sending and when it does receive the numbers, correctly change the text box to our designated color. And after testing this, we can confirm that our Dashboard works correctly.

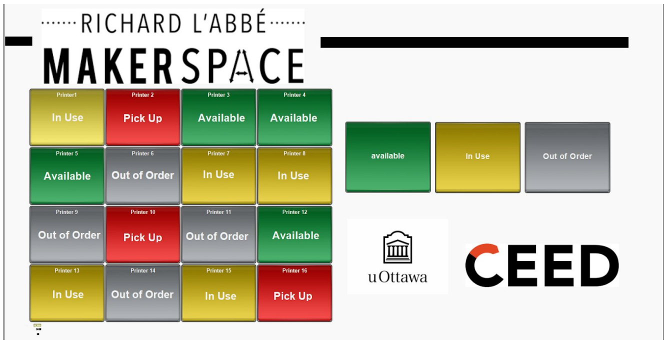
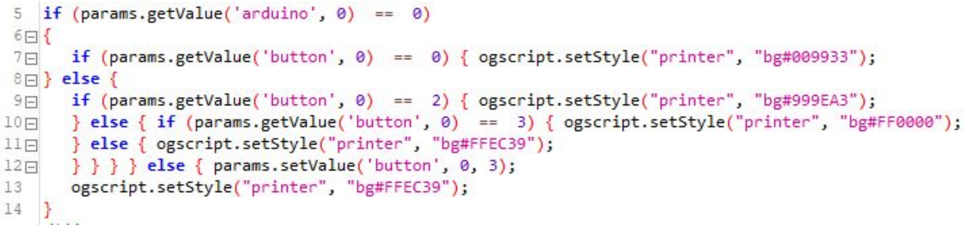
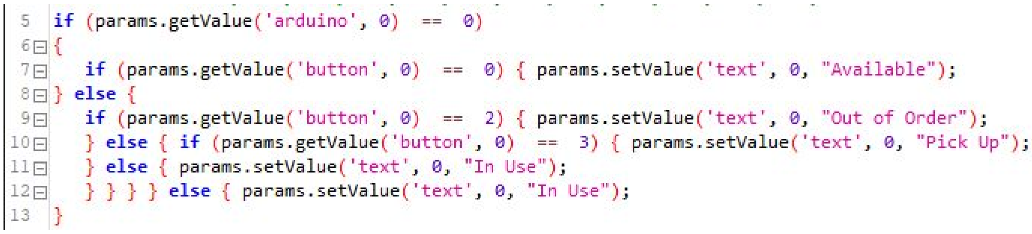


Figure 1: Dashboard Interface



```
if (params.getValue('arduino', 0)  ==  0)
{
   if (params.getValue('button', 0)  ==  0) { ogscript.setStyle("printer", "bg#009933");
} else {
   if (params.getValue('button', 0)  ==  2) { ogscript.setStyle("printer", "bg#999EA3");
   } else { if (params.getValue('button', 0)  ==  3) { ogscript.setStyle("printer", "bg#FF0000");
   } else { ogscript.setStyle("printer", "bg#FFEC39");
   } } } } else { params.setValue('button', 0, 3);
   ogscript.setStyle("printer", "bg#FFEC39");
}
```

Figure 2: Dashboard Colour Code



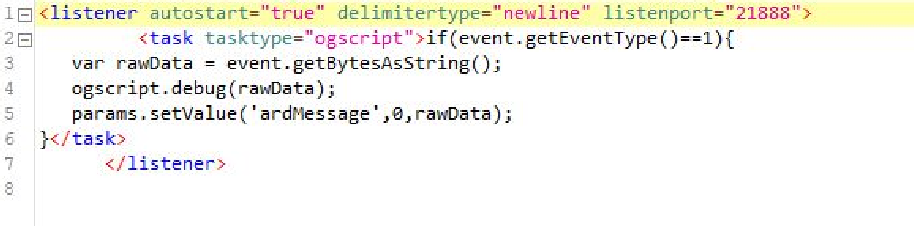
```
if (params.getValue('arduino', 0)  ==  0)
{
   if (params.getValue('button', 0)  ==  0) { params.setValue('text', 0, "Available");
} else {
   if (params.getValue('button', 0)  ==  2) { params.setValue('text', 0, "Out of Order");
   } else { if (params.getValue('button', 0)  ==  3) { params.setValue('text', 0, "Pick Up");
   } else { params.setValue('text', 0, "In Use");
   } } } } else { params.setValue('text', 0, "In Use");
}
```

Figure 3: Dashboard Text Code

4.3 Listener Server

The listener server is used to connect the Dashboard interface to Arduino. The Arduino receives "0" and "1" from the PIR motion sensor and using the client .println function it sends the number to the dashboard which stores the value into a variable called rawData in the text parameter "ardMessage" from this it changes the button color accordingly.

Our overall objective was to correctly write the code and add our specific Arduino code to it. After consulting with our TA and other experts, we can confirm that we have our listening server code working.



```
<listener autostart="true" delimitertype="newline" listenport="21888">
         <task tasktype="ogscript">if(event.getEventType()==1){
   var rawData = event.getBytesAsString();
   ogscript.debug(rawData);
   params.setValue('ardMessage',0,rawData);
}</task>
      </listener>
```

Figure 4 : Listening Server Code; Dashboard

4.4 Complete Arduino Code

The following code is used to send the information received through the motion sensor to the dashboard using a listener server on the Arduino IDE.

```
#include <ESP8266WiFi.h>
#include <ESP8266WiFiMulti.h>

#ifndef STASSID
#define STASSID "NameRouter" // Erin changed this
#define STAPSK  "VodkaCran136" //Erin changed this
#endif

int sensor = D7; // These may need to be global definitions
int state = LOW;

const char* ssid     = STASSID;
const char* password = STAPSK;

const char* host = "192.168.2.27"; // Erin changed this it used to be  10.192.68.118
192.168.2.42
const uint16_t port = 21888;

ESP8266WiFiMulti WiFiMulti;

void setup() {
  Serial.begin(9600);
  pinMode(sensor, INPUT);
  // We start by connecting to a WiFi network
  WiFi.mode(WIFI_STA);
  WiFiMulti.addAP(ssid, password);

  Serial.println();
  Serial.println();
  Serial.print("Wait for WiFi... ");
  while (WiFiMulti.run() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  delay(500);
}


void loop() {
  Serial.print("connecting to ");
  Serial.print(host);
  Serial.print(':');
  Serial.println(port);

   WiFiClient client;

    if (!client.connect(host, port)) {
    Serial.println("connection failed");
    Serial.println("wait 5 sec...");
    delay(5000);
    return;
  }
     int state = digitalRead(sensor); // Why is this a long? digitalRead outputs an int
  if(state == HIGH){
    //digitalWrite (Status, HIGH);
    Serial.println("1");
    client.println("1");
    delay(1000);
  }
   else{
    Serial.println("0");
    client.println("0");
    delay(1000);
}
}
```

5. Feedback

5.1 CEED Employee 1 Feedback:

Employee 1 liked the dashboard interface, liked the different colors and that the status of the 3D printer is also given in the text. Said it’s a nice and simple design, very intuitive and easy to use. They also liked that the boxes and buttons are large it’s easy to see from away if needed. Also won’t misclick any buttons but it would be easy to change it back

5.2 CEED Employee 2 Feedback:

Employee 2 liked the box that the motion sensor and Arduino is placed in because it’s simple and holds all the components so that space doesn’t get too messy with wires, motion sensors, and Arduino if the system was implemented on every 3D printer.

6. Conclusion

After the third prototype, we will continue to work out the last few imperfections to ensure all of the subsystems are working perfectly. From this prototype, we have completed the motion sensor circuit that connects to the Arduino. This then has a listening server that sends information to the dashboard in order to properly display the printer status for the CEED employee. Our next objective will be to continue to test our system as a whole to ensure it functions. While doing this final prototype, we learned that many things in our project did not work such as the motion sensors, Arduino code errors, dashboard code errors and finally listener server errors. Nevertheless, our group prevailed and nagged many TA, CEED employees and experts for their advice and we have managed to solve this while also learning a lot.