BPS - Bat Preservation Society

Deliverable F: Prototype II and Customer Feedback

Team C03-14

Engineering Design GNG1103

Team Members:

Jett Nguyen (300418721)

Kian O'Reilly (300438056)

Jake Easthouse (300404970)

Syamalan Krishnakumar (300434069)

Tawfiq Muhmmad (300371225)

Faculty of Engineering

November 10th, 2024

Table of Contents

1. Introduction *4*

2. Prototype II *4*

2.1. Bat Box **4**

2.2. Load Cell Sensor **4**

2.3. Camera **5**

2.4. WiFi **5**

2.5. Data Application **7**

3. Feedback *9*

3.1. Customer Feedback **9**

3.2. Feedback For Other Groups **9**

3.3. What is Next? **9**

4. Project Updates *10*

4.1. Updated Project Plan **10**

4.2. Updated Bill of Materials ..... Error! Bookmark not defined.

4.3. Prototype III Test Plan ..... Error! Bookmark not defined.

5. Conclusion *13*

6. References *13*

List of Figures

Figure 1.Wiring of load sensor 4

Figure 2. Prototype II Camera Code 5

Figure 3. Prototype II Data Application 8

List of Tables

Table 1. Updated Project Plan 10

Table 2. Updated Bill of Materials 11

Table 3. Prototype III Test Plan 12

Abstract

In the previous deliverable, Prototype I was created, and it was presented to our client during client meet 3. The feedback that was provided by the client was used to modify our prototype to create prototype II. This deliverable highlights various tests prototype III underwent and will be further modified to produce prototype III>

1. Introduction

In the previous deliverable, Prototype I was developed and was presented to our client for feedback. The feedback received from our client was used to change and improve our design from Prototype I to create Prototype II. The purpose of this deliverable is to present Prototype II and analyze its performance and functionality. Project updates and feedback that was given to our team and others will also be discussed to improve our design.

2. Prototype II

2.1. Bat Box

For the bat box we were unable to construct a new prototype for the whole box due to a heavy workload in other subjects.

2.2. Load Cell Sensor

For our first prototype we were unable to construct and test we designed since the parts hadn't come in yet. This week we were able to wire the sensor load sensor and get a chance to test the code we've been writing over the past couple weeks.

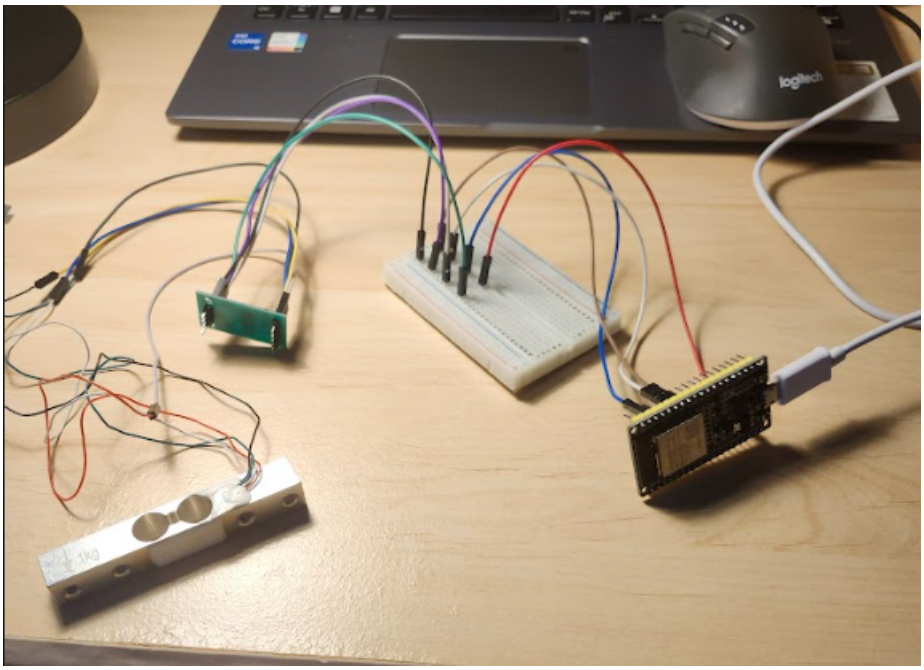


Figure 1.Wiring of load sensor

After using the example code associated with the load cell library we were able to test if our wiring of the load sensor worked correctly and if it would be able to detect a weight similar to what a bat would apply upon landing on the pad.

2.3. Camera

Last week, the camera was able to take multiple pictures and display them but used a large amount of code to do a relatively simple task. If there was an error in this code it would have been very difficult to find. It also had to be instructed by a computer to take a picture and could not store it. To fix this the code was completely deleted and restarted. Research was done on YouTube, GitHub, and various other online tech forums to figure out how the camera module works. The code below is what was created.

sketch_camera_module.ino

```
#include <Wire.h>
#include <ArduCAM.h>
#include <OV7670.h>
#include "HX711.h"

#define CS_PIN 10
#define LOADCELL_DOUT_PIN 3
#define LOADCELL_SCK_PIN 2
#define WEIGHT_THRESHOLD 1000  // Adjust this value based on your needs

ArduCAM myCAM(OV7670, CS_PIN);
HX711 scale;

void setup() {
  Serial.begin(115200);
  Wire.begin();

  // Initialize camera
  myCAM.init();
  Serial.println("Camera initialized.");

  // Initialize load cell
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
  scale.set_scale();  // Adjust to your calibration factor
  scale.tare();  // Reset the scale to 0
  Serial.println("Load cell initialized.");
}

void loop() {
  float weight = scale.get_units(10);  // Get average of 10 readings
  Serial.print("Weight: ");
  Serial.println(weight);

  if (weight > WEIGHT_THRESHOLD) {
    Serial.println("Weight threshold exceeded!");
    captureImage();
    delay(5000); // Delay to avoid multiple captures
  }
}

void captureImage() {
  myCAM.flush_fifo();
  myCAM.clear_fifo_flag();
  myCAM.start_capture();

  while (!myCAM.get_bit(ARDUCHIP_TRIG, CAP_DONE_MASK));
  Serial.println("Capture Done.");

  // Send image data to ESP32
  myCAM.CS_LOW();
  myCAM.set_fifo_burst();
  while (myCAM.get_fifo_length()) {
    Serial.write(SPI.transfer(0x00));  // Send data to ESP32 via Serial
  }
  myCAM.CS_HIGH();
  Serial.println("Image data sent to ESP32.");
```

Figure 2. Prototype II Camera Code

The code works by activating if the weight on the sensor is higher than the weight threshold and if it is the camera turns on. However, the camera is meant for sending a livestream, so this problem is resolved by setting a delay to the pictures of five seconds, so we don't get multiple pictures of the same bat entering. Next the code sends the data to the wifi module where it can be uploaded.

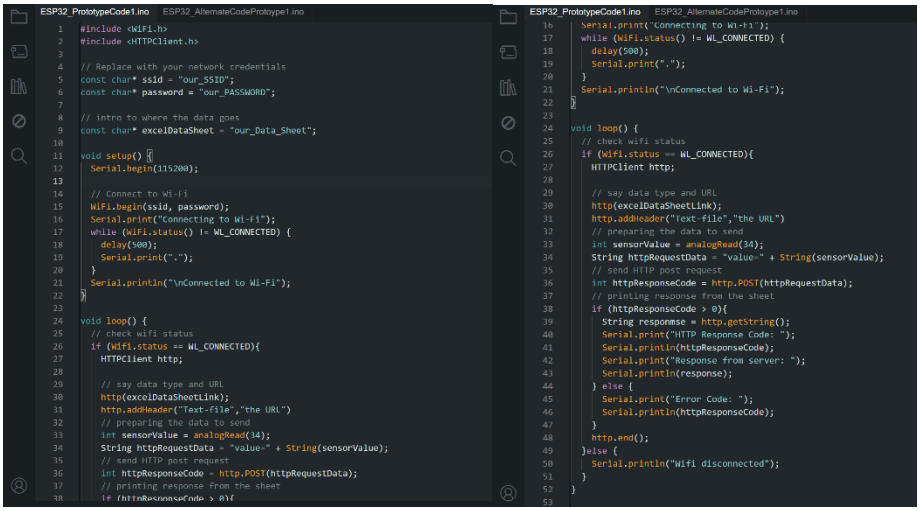
While we received the camera module for our final prototype, we were unable to test it as we did not have enough of the right type of wires to use both it and the load cell. Since our code requires the sensor to be triggered to activate the load cell testing wasn't possible. We will get more of the correct wires so we can test it before the final prototype.

2.4. Wi-Fi

The code for the Wi-Fi module stayed relatively unchanged, but we created an alternate code that uses a URL from a site called IFTTT, that acts as a liaison between multiple different sources. The liaison would transmit the data received from the weight sensor through the site itself and

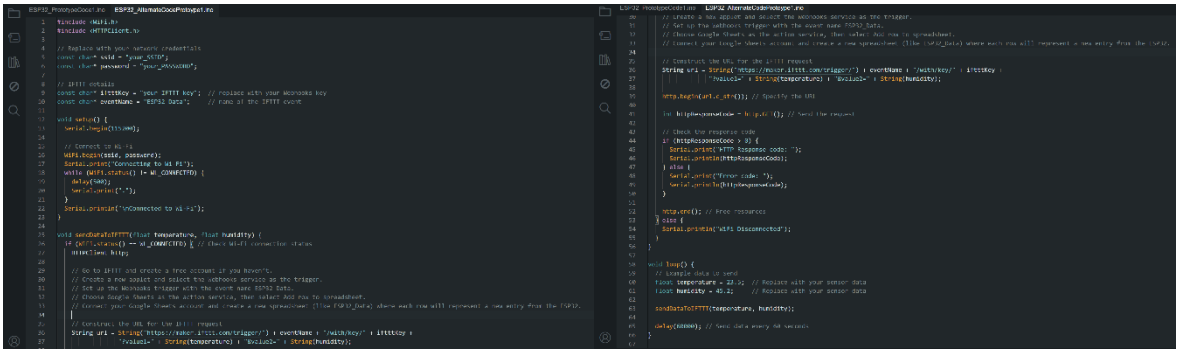
then to google sheets, the app selected by the user to send the data. This code will be used as an alternative to our original Wi-Fi module's code. Here is the original and the alternative code:

Original:



```
#include <WiFi.h>
#include <HTTPClient.h>

// Replace with your network credentials
const char* ssid = "our_SSID";
const char* password = "our_PASSWORD";

// intro to where the data goes
const char* excelDataSheet = "our_Data_Sheet";

void setup() {
  Serial.begin(115200);

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nConnected to Wi-Fi");
}

void loop() {
  // check wifi status
  if (WiFi.status == WL_CONNECTED){
    HTTPClient http;

    // say data type and URL
    http(excelDataSheetLink);
    http.addHeader("Text-file","the URL")
    // preparing the data to send
    int sensorValue = analogRead(34);
    String httpRequestData = "value=" + String(sensorValue);
    // send HTTP post request
    int httpResponseCode = http.POST(httpRequestData);
    // printing response from the sheet
    if (httpResponseCode > 0){
      String responmse = http.getString();
      Serial.print("HTTP Response Code: ");
      Serial.println(httpResponseCode);
      Serial.print("Response from server: ");
      Serial.println(response);
    } else {
      Serial.print("Error Code: ");
      Serial.println(httpResponseCode);
    }
    http.end();
  }else {
    Serial.println("Wifi disconnected");
  }
}
```

Alternative:



```
#include <WiFi.h>
#include <HTTPClient.h>

// Replace with your network credentials
const char* ssid = "your_SSID";
const char* password = "your_PASSWORD";

// IFTTT details
const char* iftttKey = "your IFTTT Key";  // replace with your Webhooks key
const char* eventName = "ESP32 Data";     // name of the IFTTT event

void setup() {
  Serial.begin(115200);

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nConnected to Wi-Fi");
}

void sendDataToIFTTT(float temperature, float humidity) {
  if (WiFi.status() == WL_CONNECTED) { // Check Wi-Fi connection status
    HTTPClient http;

    // Go to IFTTT and create a free account if you haven't.
    // Create a new applet and select the Webhooks service as the trigger.
    // Set up the Webhooks trigger with the event name ESP32_Data.
    // Choose Google Sheets as the action service, then select Add row to spreadsheet.
    // Connect your Google Sheets account and create a new spreadsheet (like ESP32_Data) where each row will represent a new entry from the ESP32.

    // Construct the URL for the IFTTT request
    String url = String("https://maker.ifttt.com/trigger/") + eventName + "/with/key/" + iftttKey +
                 "?value1=" + String(temperature) + "&value2=" + String(humidity);

    http.begin(url.c_str()); // Specify the URL

    int httpResponseCode = http.GET(); // Send the request

    // Check the response code
    if (httpResponseCode > 0) {
      Serial.print("HTTP Response code: ");
      Serial.println(httpResponseCode);
    } else {
      Serial.print("Error code: ");
      Serial.println(httpResponseCode);
    }

    http.end(); // Free resources
  } else {
    Serial.println("WiFi Disconnected");
  }
}

void loop() {
  // Example data to send
  float temperature = 22.5;  // Replace with your sensor data
  float humidity = 45.2;     // Replace with your sensor data

  sendDataToIFTTT(temperature, humidity);

  delay(60000); // Send data every 60 seconds
}
```

Although, to fix the original code, there must now be a function to allow the photos to be sent through the Wi-Fi module as well.

Along with updating the code for the Wi-Fi application we were also able to test the functionality of the microcontroller's Wi-Fi capabilities by checking to see if it could pick up on and connect to nearby wireless networks. This test was successful, confirming that the microcontroller's Wi-Fi module works.

2.5. Data Application

In our last prototype, we began developing our data application which was able to navigate between different windows that were outlined in our detailed design. It followed the application flowchart that we created which navigated through the following windows: login, login credentials, data, review/sort, and live footage.

In this prototype, we further modified our application to have: a working device registration that sends login info to an excel file, a valid login credentials screen that only works if the entered data is found in the login info excel file, a table that displays data retrieved from an excel sheet in the review/sort window, working buttons that show the number of entries from the past week, month, or all season. To make our application more aesthetically pleasing, backgrounds were changed to white, button colours were changed to grey, fonts were changed to a larger size in Times New Roman, and all screens were centered. Messages were also created to be displayed if errors were encountered, e.g. "Login Failed", "Invalid device number or password. Please try again.".

The Prototype II Data Application successfully passed multiple tests. It was able to validate login credentials of existing registered devices, register new devices, display data from an excel file, and display the amount of entrances that the device collected.

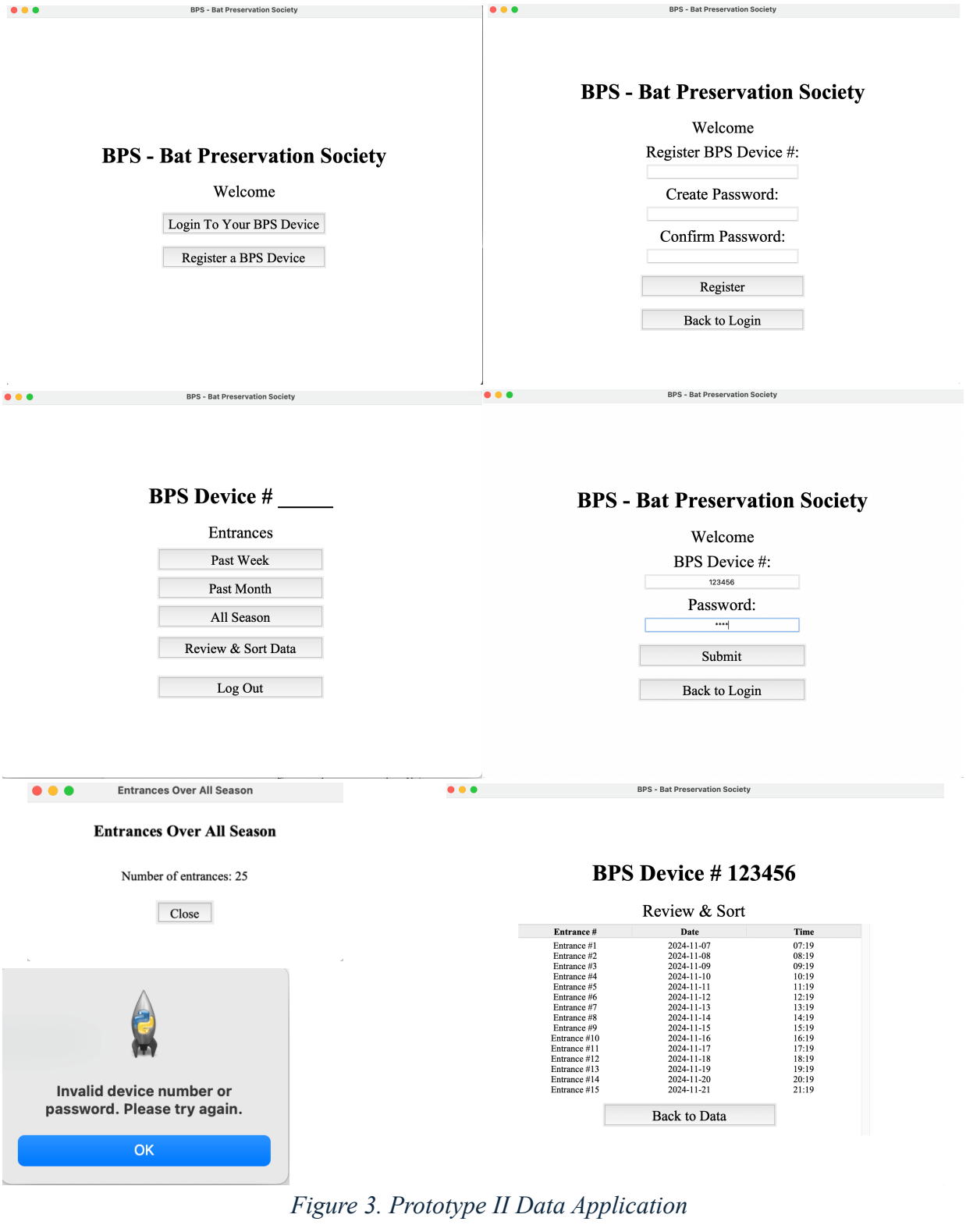


Figure 3. Prototype II Data Application

3. Feedback

3.1. Customer Feedback

After our third client meeting, where we presented our first prototype, since we didn't have any physical representation of our project, we were told to build our model as soon as possible to ensure we get the best feedback from him as well as our classmates. He also told us to get rid of the live footage function of our model, as it would take too much battery power and take up too much storage space. We were also told to discuss the limitations of our design, one of which could be power usage. We need to find out which components have the largest need of power, and how we could reduce that. Lastly, there is the issue of whether the impact of the bats landing would activate the sensor. Testing the minimum impact force will be required to ensure all entries are logged.

Summarized:

- Create a physical model to receive more feedback
- Scrap the live footage function
- Power usage of each component
- A bat's activation of the weight sensor

3.2. Feedback For Other Groups

Some feedback that the client gave during the 3rd client meeting that could benefit us was to not save the building of our prototype to the last minute, and rather finish it as soon as possible. As a result, we decided to switch from using batteries to have it plugged into an outlet instead. The client also emphasized the importance of understanding the power consumption of our device. This ultimately saved us quite some time, allowing to spend more time on building our final prototype as well as cut down power consumption. In addition, this helped us cut down our overall budget as well.

3.3. What is Next?

Next, we will combine all our updated codes and test to make sure they run and perform their functions on the physical components that we will also combine to form our first physical prototype of our sensor system. We will test these functions by dropping a weight that is like a bat onto the weight sensor, this will cause an entry to be tallied and sent to the data application. The imitated bat's weight will be altered to find the minimum amount of force needed to log an entry.

4. Project Updates

4.1. Updated Project Plan

Table 1. Updated Project Plan

Item	Description	Teammates	Date Due	Duration
Bat Box Prototype 3	Based on the testing, construct a new prototype or modify the existing one to meet requirements of test.	Tawfiq	November 16	>3 hours
Data Application Prototype 3	Based on the testing, construct a new prototype or modify the existing one to meet requirements of test.	Jett	November 16	6 days
Load Cell Sensor 3	Based on the testing, construct a new prototype or modify the existing one to meet requirements of test.	Syamalan	November 16	6 days
Wi-Fi Module prototype 3	Based on the testing, construct a new prototype or modify the existing one to meet requirements of test.	Kian	November 16	6 days
Camera Module Prototype 3	Based on the testing, construct a new prototype or modify the existing one to meet requirements of test.	Jake	November 16	6 days
Prototype Testing	Tests will be performed on each aspect of the design to determine which aspects works and which need to change.	Whole Team	November 22	6 days
Design Day	Team will prepare a presentation and present it to the client at design day	Whole Team	November 27	5 days
User Manual	Team will write a manual for potential users.	Whole Team	Dec 3	6 days

4.2. Updated Bill of Materials

Table 2. Updated Bill of Materials

Item/material	Description	Quantity	Size/Dimensions	Est. Cost per	Total est. Cost
Legend					
Green	We have the item				
Yellow	We might not the item				
Grey	We might not need				
Bat Box					
MDF	box panels and landing pad	2	12" x 24	$ 2.50	$ 5.00
Screws	Used to assemble the bat box	1	1" long	$ 3.97	$ 3.97
Sensor					
Load Sensor	Used to track bat entires by attaching to landing pad	1	3.0" x 0.5" x 0.5"	$17.99	$ 17.99
Camera	Takes snapshot when load cell is triggered. Used for additional information	1		$ 9.99	$ 9.99
Wi-Fi (ESP32)	Used to send data to data application	1		$ 12.00	$ 12.00
Bread Board	Electric circuit for our device	1		$ 5.00	$ 5.00
Wires	Used to wire sensors	3		$ 1.00	$ 3.00
Software					
Python v3.14	Programming software used to create our data application	1	N/A	N/A	free
tkinter library	Python library used in data application code	1	N/A	N/A	free
Arduino IDE	used to create our sensor	1	N/A	N/A	free
DFRobot_HX711.h	Library used for our load cell	1	N/A	N/A	free
ArduImageCapture plugin	Library used for our camera	1	N/A	N/A	free
Ada_fruitOV7670	Library used for our camera	1	N/A	N/A	free
ESP32	Library used for WiFi module	1	N/A	N/A	free
LiveOV7670	Library used for our camera	1	N/A	N/A	free
openpyxl library	Python library used in data application code	1	N/A	N/A	free
datetime library	Python library used in data application code	1	N/A	N/A	free
TOTAL (with tax)		$ 64.35			

4.3. Prototype III Test Plan

Table 3. Prototype III Test Plan

Test ID	Objective	Test Description and Materials needed	Description of results to be recorded	Test duration
1	Validating Spec 5	The Sensor will be placed on a prototype of the bat box and an object with a similar weight will be thrown at it. We will need a prototype of the bat box, an object with the weight of a bat, the Arduino/ Arduino components, and the load sensor.	We will record whether the sensor can track bats. The test will be concluded when we know that it can accurately track bats. As with most of these tests, the results will be determined by our confidence.	<2 hours
2	Validating Spec 3	The camera will be hooked up to the sensor and placed in various locations on the bat box while the sensor test is performed. The same materials are needed.	We will record whether or not the camera can take pictures and see where on the box the camera should be placed. Test will be concluded when we both know the ideal location for the camera and know its takes pictures.	>3 hours
3	Validating Spec 4	The Wi-Fi module will also be attached while the sensor test is being performed.	See if the Wi-Fi module can upload images to a cloud storage system. Test will be completed when we can prove that the module can	>2 hours

			upload data reliably.	
4	Validating Spec 3	Data will be uploaded to the storage system and attempted to be accessed through the system. No materials are needed.	Test the usability of the storage system and see if it stores data. Test will be completed when we can be confident that the storage system stores data.	>2 hours
5	Validating Spec 4	A more realistic bat box prototype will be constructed. We will need a material harder than cardboard.	Test the time it takes and challenges associated with constructing it. Test will be completed when the box has been constructed.	>2 hours

5. Conclusion

In conclusion, Prototype II was successful through various tests. It consists of multiple modifications to sensor code that was tested after successfully wiring the ESP32 to our load cell sensor as well as a functional data application. For our Prototype III, feedback will continue to be gathered and implemented into our design. Additional testing will be performed to ensure our we have a functional final prototype that can be presented to our client and others on design day. We will continue to modify and improve our product as we work towards the completion of our project.

6. References

Pelayo, Roland, and Roland Pelayo. “Arduino Camera (OV7670) Tutorial.” Microcontroller Tutorials, October 10, 2023. https://www.teachmemicro.com/arduino-camera-ov7670-tutorial/#google_vignette.

"How to Use OV7670 Camera Module With Arduino ," n.d. https://circuitdigest.com/microcontroller-projects/how-to-use-ov7670-camera-module-with-arduino.

Indrek. "OV7670 Camera Module With Arduino: Color Image to PC (Step-By-Step Guide)," December 2, 2018. https://www.youtube.com/watch?v=L9DTW1ulsT0.