Cover and Title Page

Indoor Environment Monitoring IOT System

By Chris Kruger, Vanessa Huerta, Liamjohn Velazquez
April 29th
E-122 Section N

The group pledges its honor that it has abided by the Stevens Honor System
Chris Kruger, Vanessa Huerta, Liamjohn Velazquez

Abstract:

The Design II project was to help the team understand the idea of the “Internet of Things” (IOT). By allowing the team to acquire field data in raw data form, it would then be uploaded to a server. From there, the team then downloaded the data from another location and incorporated the data into graphical interpretations. The goal of this project was to fully understand the logic behind an IOT system, and why it can be useful in scenarios where those that are part of a team may not be located in the same vicinity. This was also unintentionally a great way to see how work can be done remotely from home due to any situation such as the COVID-19 pandemic. Innovations that are specific to the group were making a visual interpretation in LabView that is readable to the layman as well as making a cheap yet effective housing for the WeMos board. Please see **Figure A.1** for this visual interpretation and **Figure C.1** for the WeMos housing. The team’s design performed exceptionally, with no faults and always gave data that was expected. From looking at the final results being visualized and interpreted, the group found results that are expected; the light sensor showed lower results as the night time came, the temperature went down as the night time came, and the humidity went up as the day time came. Although expected, the team found these findings interesting to see visualized. Due to the system working properly, the only issue that came about was setting up the WeMos board which gave some trouble due to poor quality wires in some cases. Concluding this effort, the team found it to be interesting how intuitive it is to use IOT when delegating different tasks when in separate locations as well as seeing the different values of the sensors change as the day went on.

Table of Content:

Introduction:

The purpose of this report is to reflect on the outcome of the project and examine what the team did right as well as places the team could have done better. In this project the team learned how to use programs such as LabView, Arduino IDE, and Solidworks. The team was able to analyze data and create graphs and tables in order to interpret the data better. The group was able to not only work on creating a system designed to be able interpret data, but also work on the mechanical and software design by setting up a WeMos board to interpret data. Overall, the objective of this project is to create a software system with sensors that remotely monitors an environment and use software tools to collect, store and visualize data clearly. While doing this project the team faced restrictions due to the COVID-19 situation and being required to complete the project from home. The team had to learn how to better communicate and use the resources available to them in order to complete their assignments. Please refer to the organization chart **(Pg.4)** to see individual roles played by each team member.

Organization chart:

Team Member	**Role**
Christopher Kruger	Setting up the Software
Vanessa Huerta	Leading the Written Work
Liamjohn Velazquez	Setting up the Hardware

Discussion:

Requirements

The team is required to design, test, and operate a system with sensors (humidity, light, and temperature) that remotely monitors another user's environment. The selected sensors will measure time varying physical quantities and will enable data-driven operations to be stored. Following this, the team will be required to use IOT protocols to publish the data to a remote server in the Stevens Institute of Technology campus. By using tools such as LabView and Arduino IDE, the team downloads then analyzes the data. By doing this, the team will be able to collect, store, interpret, and visualize the data.

Following the publication of data, the team must then collaborate effectively with teammates to make sure every process is done correctly and without fault. The team then must conduct a presentation to report on their findings. By doing this, the team must practice oral and written communications skills as well as practice professionalism in a shared lab environment.

System Design

The project took roughly 14 weeks to be completed. The first week involved the installation of LabView and getting accustomed to the User Interface. The second week was connecting the potentiometer to the WeMos, developing a LabView Virtual Instrument, and verifying the sensitivity and position measurement. Having done this allowed the team better knowledge in having to troubleshoot for problems that may arise in the future. The third week involved using a CdS photocell, making the group become more aware of the sensors that will be used for the final project. The fourth week involved a similar process of becoming more aware of what sensors will be used and how they work, namely the temperature sensor and

humidity sensor. Week five taught the team the methods of using a IOT system. By configuring the sensors and subscribing to a network, the team was able to publish and record the data to Stevens Institute of Technology server. This was also the beginning of learning how to properly visualize spatial and temporal changes in the environment. Weeks six through seven began the journey in using the 3D printer. By testing out individual projects of the team member's choice, the team was now beginning to understand how to properly set up a model for 3D printing, how to estimate a time to print, and how to calculate the cost of printing an item. Please refer to **Figure D.1** to see the results. This helped the team learn that certain orientations of a model may reduce/increase the time to print as well as the cost to produce. Week eight involved the team's members downloading data from example data given to test a LabView program to see the visualization of the program. An example of this visualized data can be seen in **Figure(s) E.1-E.3**. Weeks nine through ten allowed the team to plan and sort out requirements for the final project. The block diagram for the system can be seen in **Figure H.1** and a photo of the system can be seen in **Figure I.1**. This involved test validations, measuring power draws, and making sure all integrations of the IOT system were working. Week eleven was used for testing field data and field maintenance. Weeks twelve through thirteen involved the devices doing proper field work, taking in data points using the onboard sensors. The team conducted the field work by letting the WeMos run for a minimum of a day in two seperate rooms with slightly different environments. Both tests were recorded in the same city and state of Fairfield, CT. The first data set was recorded on the first floor of the first house, the system was set up in a bed room taped to a window. The data recorded included the light from outside and the temperature and humidity from inside. The second data set was taken in similar conditions taped to a window of a room on the first floor. The data again was of the light from the outside and of the temperature

and humidity of the inside. Week fourteen was the end of the project, allowing the team to recover the device, analyze the data, and conduct an oral report.

Mechanical Design

An enclosure was designed in SolidWorks in order to hold and protect the system. The wemos board would reside in the bottom half with the sensors in the top half. Holes were created to allow for the power cable. The enclosure was designed such that it required as little material as possible while still being strong enough to support the system. The material is 0.175 inches thick at anypoint. The enclosure is 4 inches long and 1.25 inches wide.The bottom half of the enclosure is tall enough to fit the wemos and long enough so that it fits snuggly. The top half allows for room for the sensors and the system is thin enough that the wires could wrap around to the upper portion. Please look at **Figure D.2** for these measurements.

Software Design and Coding

The software aspect of this project was accredited to two different programs: LabView and the Arduino IDE.

The code/block diagram for LabView can be seen in **Figure(s) B.1-B.2.** The program began by taking in the excel document that was created from the WeMos uploading data to the server. The program then takes the file from the MQTT server and separates it into Light, Temperature, and Humidity arrays. The LabView program, or Virtual Instrument (VI), reads the “comma-separated-variable” file from excel and separates it into a 2D array. In doing so, a “delimeter” was used to tell the VI where to create a new cell. The program then reads the size

of the array, making note of how many times the loop will need to run. This loop will separate the data into 3 different arrays, named “Temp,” “Humidity,” and “Light.” Another loop then removes any blank data points and will ignore any null data points. The program then “scrubs” the arrays, turning the values into condensed arrays. These arrays are then converted from a string into an integer. Following this, the integers are then manipulated mathematically to show average values, max values, and min values. Look at **Figure E.4** for a flow chart of this progress.

The code for the Arduino IDE can be seen in **Figure F.1.** The code for the Arduino IDE had a different use compared to the VI. This code was used to configure the WeMos board to use its sensors to continuously upload its data to a Stevens MQTT remote server. The began by inputting a team member’s personal wifi name and password to use the internet. By connecting to the internet, it then used the Steven server’s username and password to upload the data. Before it was taking in data, it had some fail safeties, for example if it ever failed to connect, it would try again in five seconds. This allowed the WeMos board to never fail. From there it would publish its data. The flowchart is shown in **Figure F.2**. The actions of the weather station could be seen in the serial monitor from its connection to the server to the data it collects. It is useful in identifying errors and troubleshooting the system. Initially the board wasn’t able to connect to the internet but by using the serial monitor we were able to identify and fix the problem. Photos of the serial monitor are provided in **Figure G.1**.

Electrical and Wiring Design

The board consists of a WeMos D1 R1 board, a DHT11 sensor board, a CdS photocell, a 10K Ohm Resistor, wires, and a USB B Micro Cable 3’. The code and power is provided to the

board through the Micro Cable. The DHT 11 sensor board is connected to the board through three wires. The first goes from the ground on the DHT to ground on the WeMos, the second goes from Data on the DHT to D6 on Wemos, and the third goes from VCC on the DHT to 5V on the Wemos board. To set up the photo sensor, three wires plus a resistor is required. One end of the photosensor and one end of the resistor are tied together and then connected to a wire that leads to the A0 port. The loose end of the photo sensor is connected to the 5V port and the loose end of the resistor is connected to the GND port.

Final Evaluation

Overall the system performed the way it was supposed to. Comparing the two data sets the team got consistent results. Looking at the data, the team realized that the temperature, humidity, and light graphs showed a distinction between day and night. The temperature, humidity and light dropped during the night time which can be clearly seen in the images of the graphs. The second bedroom also showed similar results. The average temperature for bedroom 1 is 62.372 is slightly higher than bedroom 2 which is 59.516. The average humidity for bedroom 1 is 44.3375 which is lower than the average humidity in bedroom 2 at 53.2359. The average light in bedroom 1 was 1.6133 which was higher than the average light in room two, which was 1.00849. **Figure(s) E.1 - E.7** (not including E.4) show the final results from the team's findings. The first three figures show the Temperature, Humidity, and Light data from the first bedroom. This bedroom is located in Fairfield, CT.

Software tools (Excel and LabView) allowed the team to properly analyze the data. Excel allowed the team to collect the data they received from the WeMos system, while LabView allowed the team to display the data in order to look for trends.

Conclusion and Recommendations:

The team was able to successfully create a system that monitored its environment and used software tools that collected and visualized the data. While the team successfully completed the project there are some recommendations the team would make in order to improve the team's system. It would have been better to have the 3D design the group created to hold the WeMos board, but that was not possible given the COVID-19 situation. Overall the team believes they did a good job in creating a functioning system that accurately collected data. One recommendation the team would make in regards to improving the project in terms of enhancing the students basic knowledge would be to give the students more practice with the softwares, especially LabView, before starting the project so that they gain more knowledge on how to properly use it. Another recommendation the team came up with is to use the system to test on something, not just receive data. Doing this would make the project more impactful and would help the students remember the project in the future. Overall this project allowed the team to learn valuable skills including increasing their knowledge of software applications such as LabView, Arduino IDE, and Solidworks which will be useful knowledge for future design classes as well as research the team might do. This project also allowed the team to collaborate with each other. Especially since they had to work remotely, good communication was necessary in order to successfully complete this project. Good communication and teamwork is a skill that the team will use in future classes and jobs once out of college. Finally the team also practiced their oral communication skill, which is necessary in future design classes in order to communicate results. Overall, the team successfully completed the project and gained many skills from the experience as well.

Attachments:

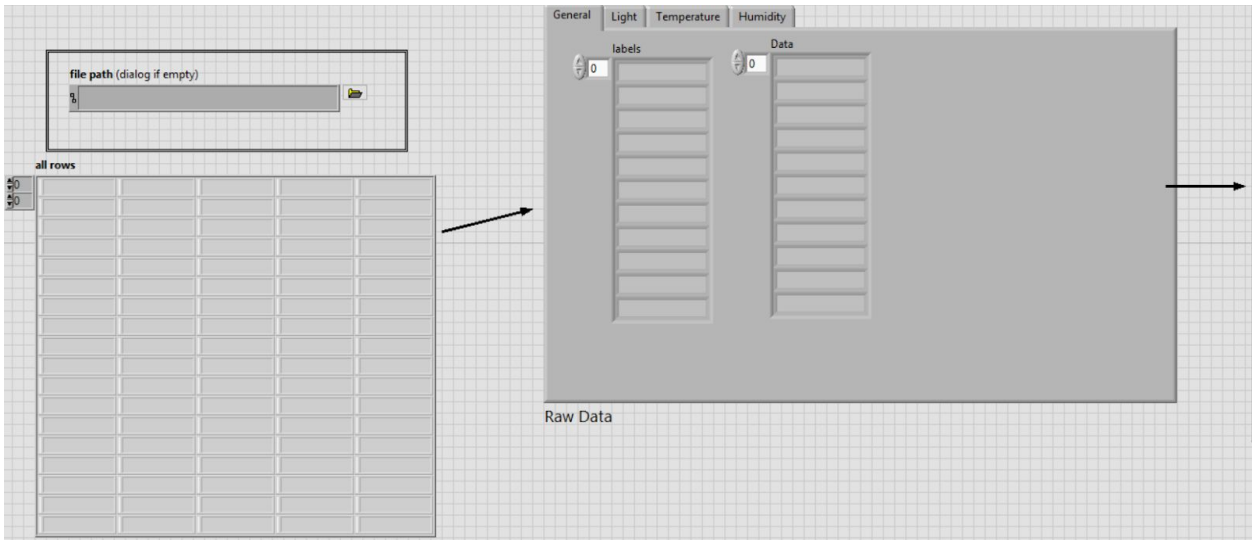


Figure A.1 - Note Guidance Arrows and Text Elements

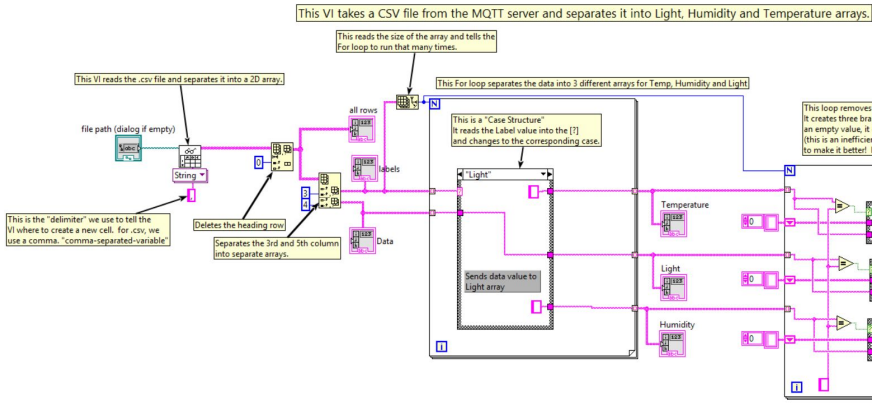
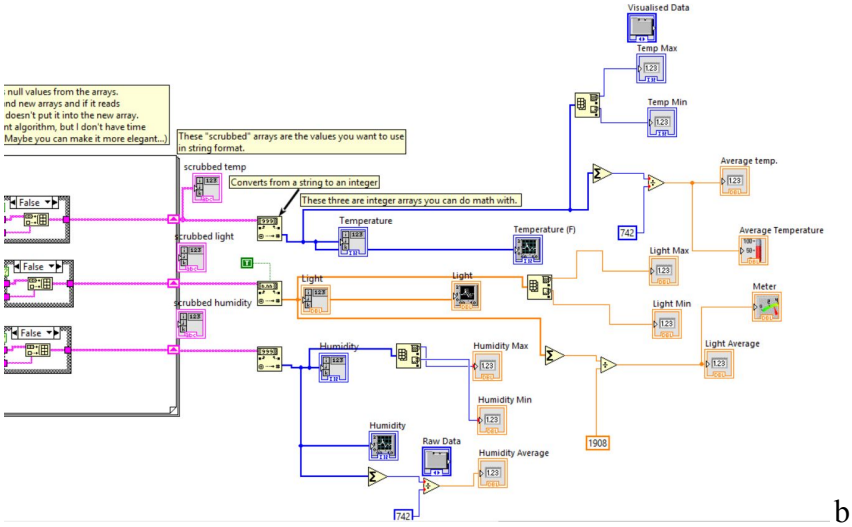


Figure B.1 - Block Diagram of the Logic Behind Our LabView



b

Figure B.2 - Block Diagram of the Logic Behind Our LabView

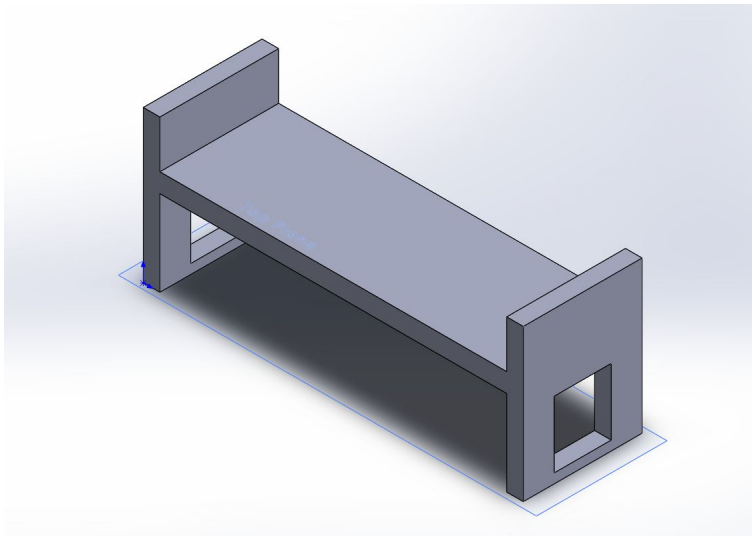


Figure C.1 - Isometric View of the Solidworks Part

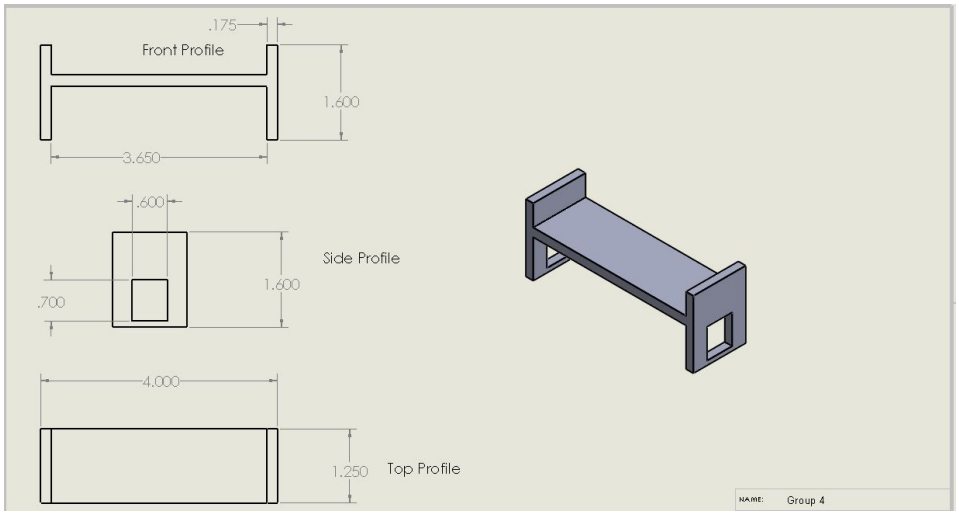
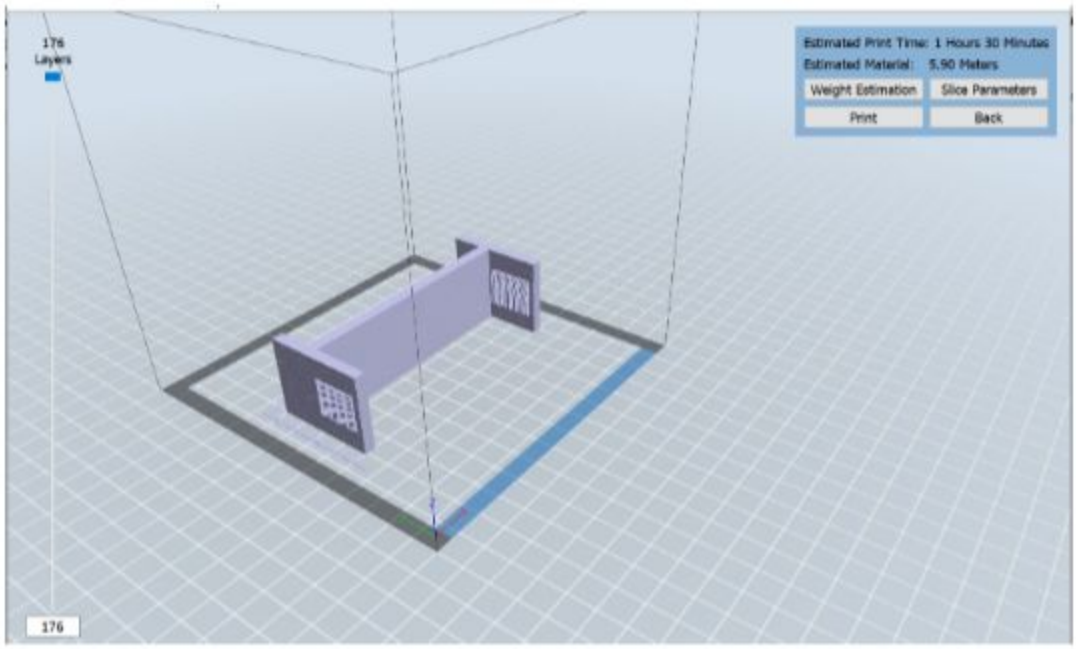


Figure C.2 - Measurements of the Solidworks Drawing

Estimated Length : 5.9 m

Time to Print : 1.5 hours



Estimated Length : 8.21 m

Time to Print : 2.2 hours

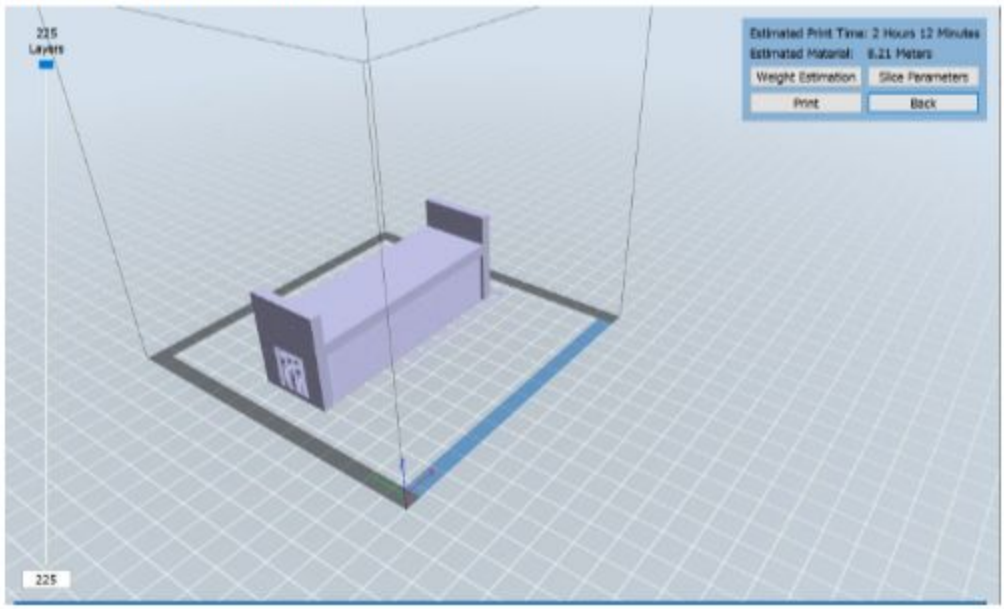


Figure D.1 - The Estimated Length and Cost Required to Print the Team's Enclosure

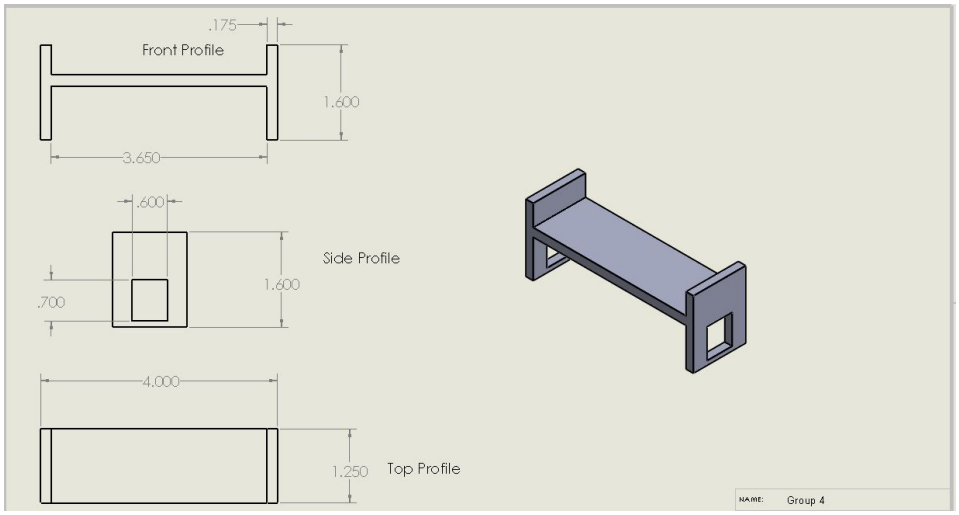
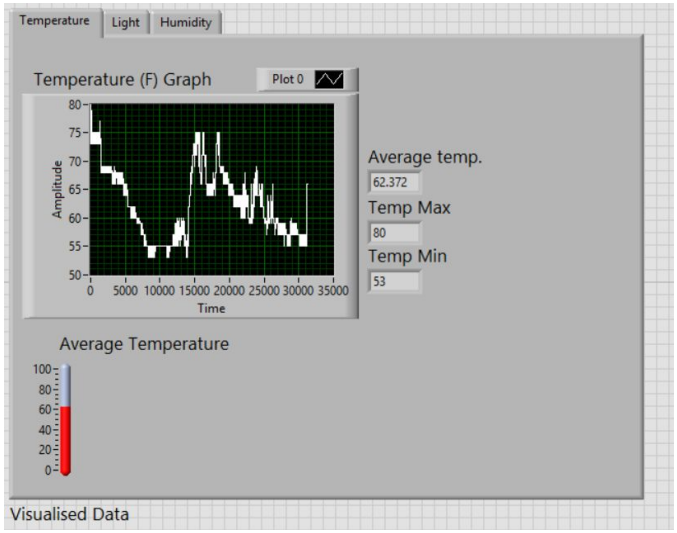
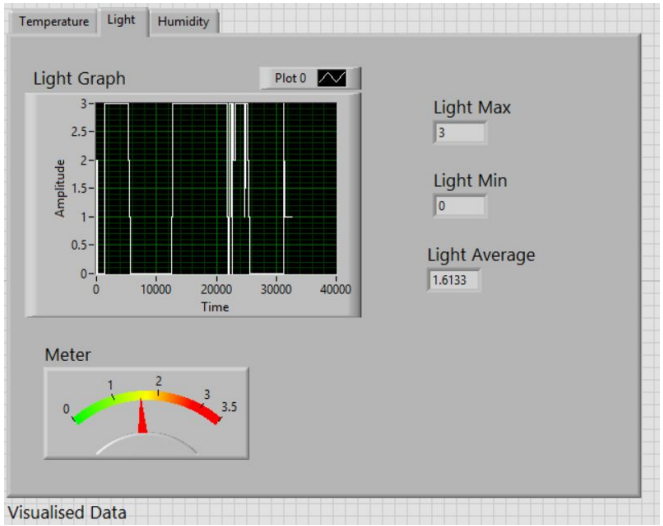


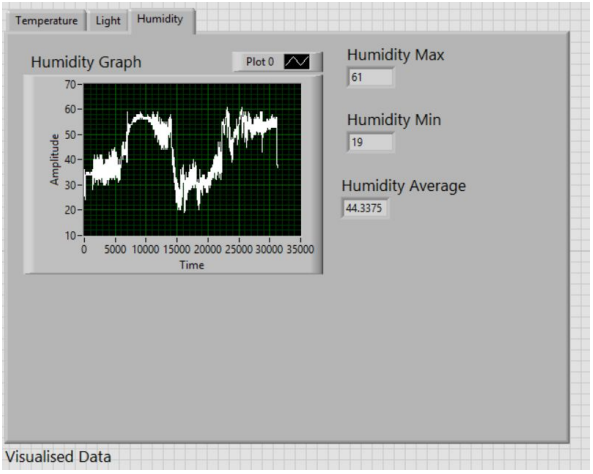
Figure D.2 - Measurements for the Enclosure



E.1 - Visual Graph of the Temperature Data Points (BEDROOM 1)



E.2 - Visual Graph of the Light Data Points (BEDROOM 1)



E.3 - Visual Graph of the Humidity Data Points (BEDROOM 1)

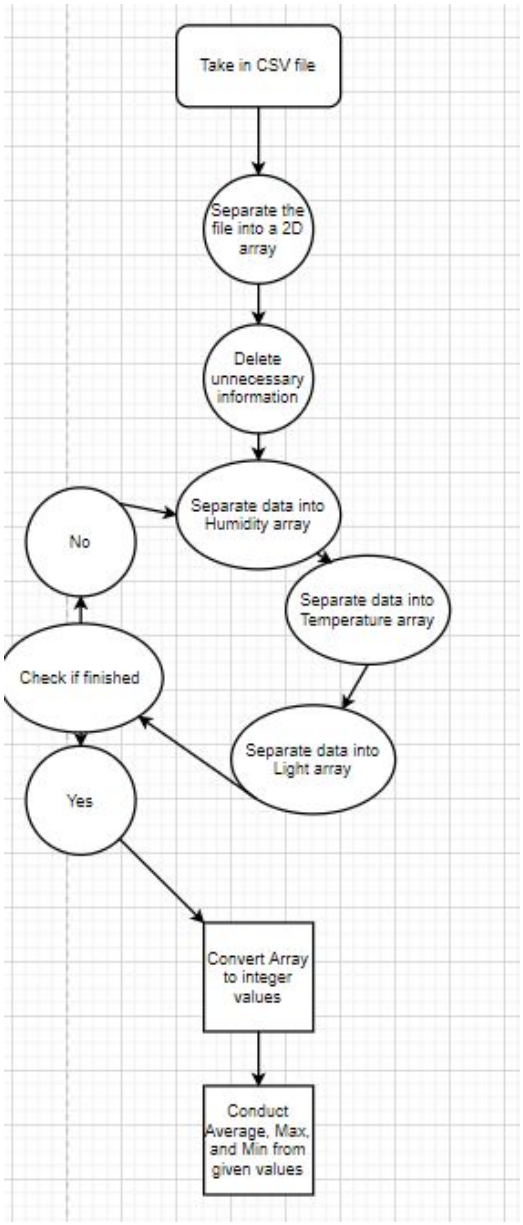


Figure E.4 - Flowchart of LabView program

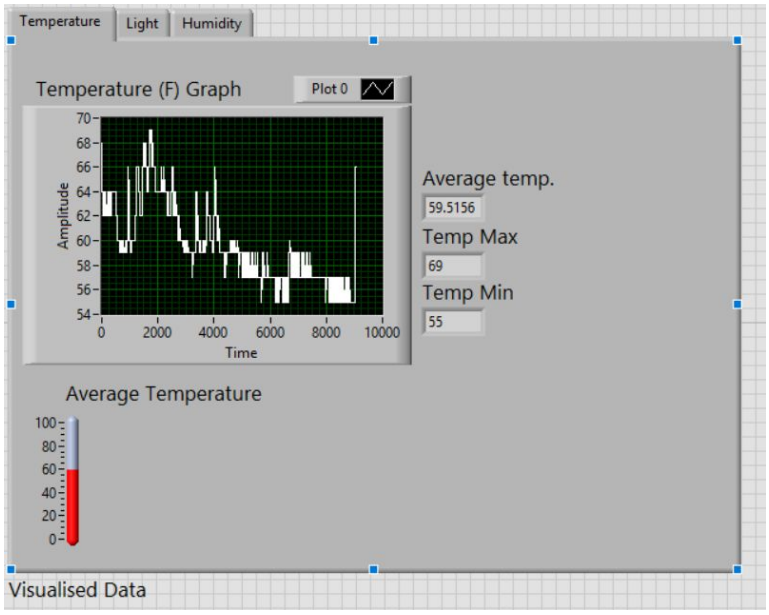


Figure E.5 - Visual Graph of the Temperature Data Points (BEDROOM 2)

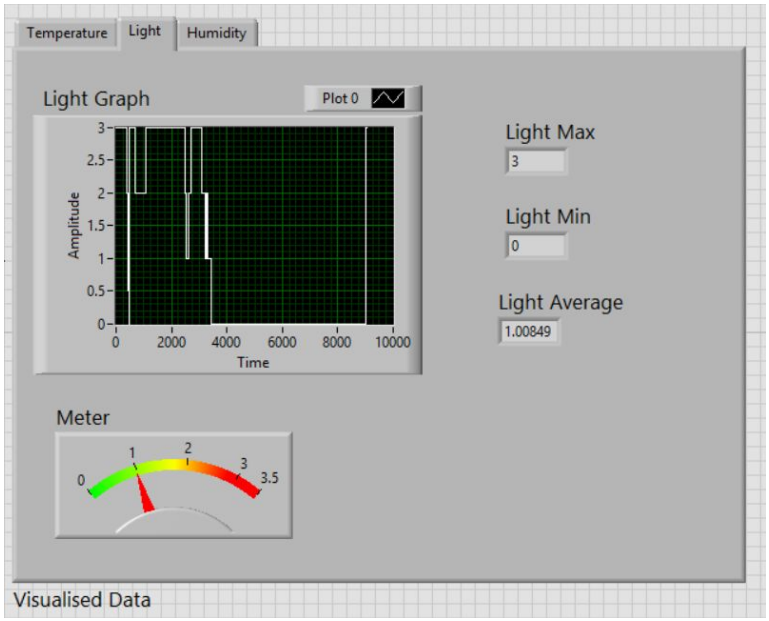


Figure E.6 - Visual Graph of the Light Data Points (BEDROOM 2)

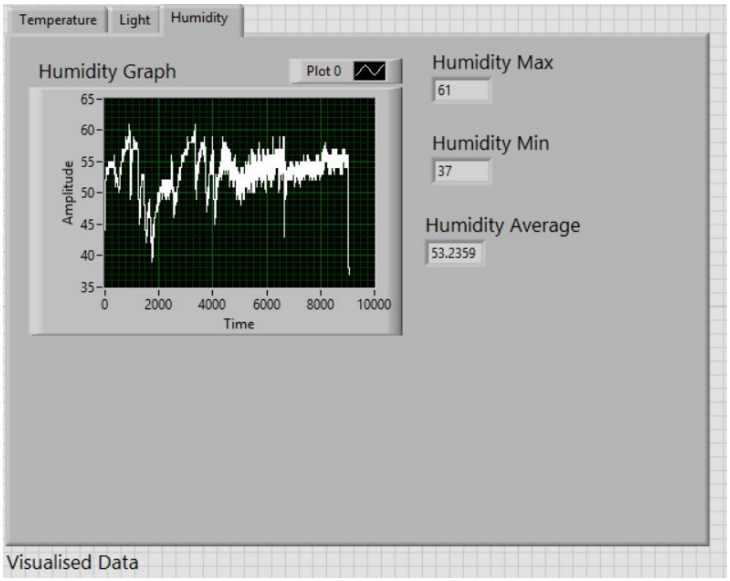


Figure E.7 - Visual Graph of the Humidity Data Points (BEDROOM 2)

```
#include <ESP8266WiFi.h>
#include "PubSubClient.h"
#include "info.h"
#include "WiFiManager.h"
#include "DHT.h"

  //The ESP8266 recognizes different pins than what is labelled on the WeMos D1
  #if defined(d1)  //Defines Wemos D1 R1 pins to GPIO pins
    #define D0 3
    #define D1 1
    #define D2 16
    #define D8 0
    #define D9 2
    #define D5 14
    #define D6 12
    #define D7 13
    #define D10 15
  #endif
  #if defined(d1_mini) //Defines Wemos D1 R2 pins to GPIO pins
    #define A0 0
    #define D0 16
    #define D1 5
    #define D2 4
    #define D3 0
    #define D4 2
    #define D5 14
    #define D6 12
    #define D7 13
    #define D8 15
  #endif
```

```
//Set up the DHT11
#define DHTPIN D6  //PLUG THE DHT 11 ONLY INTO D5, D6, D7 on either D1-R1 or D1-R2 Board.
                   // DO NOT PLUG INTO OTHERS AS THE MAPPING IS NOT THE SAME. DHT11 WILL BURN.
#define DHTTYPE DHT11
DHT dht(DHTPIN,DHTTYPE);


// CONFIGURATION SETTINGS ....BEGIN

//Wifi Settings
//const char* ssid = "DLabsPrivate1";
//const char* password = "L3tsM@keSometh1n";
const char* ssid = "Stevens-Media";
const char* password = "Stevens1870";
// ANY WEMOS WITH A STICKER ON THE BACK IS REGISTERED TO THE NETWORK...
// THOSE WITHOUT THE STICKERS DO NOT WORK...


//MQTT Settings
const char* mqtt_server = "155.246.62.110";
const char* MQusername = "jojo";
const char* MQpassword = "hereboy";
//const char* mqtt_server = "broker.hivemq.com"; //This is a public server - no username/pwd

//MQTT Publish Topics
const char* MQtopic1 = "E122/FD25/Temperature";
const char* MQtopic2 = "E122/FD25/Humidity";
const char* MQtopic3 = "E122/FD25/Light";


//MQTT Data Model
const int DataModel = 1; // 1: Sin others are not implement
const int DataTempMin = 20;
const int DataTempMax = 50; // Make sure max > min.
const int DataHumMin = 30;
const int DataHumMax = 60; // Make sure max > min.
const int DataCyclePeriod = 2; // In minutes
//const int DataSampling = 30; // This is hardcoded on this wemos.
//
//Note that since this is a real Wemos board -- it runs forever as opposed to
// the fakemos -- http://www.prooflab.stevens.edu/fakemos/

// CONFIGURATION SETTINGS 129
//END

WiFiClient espClient;
info board_info;

PubSubClient client(espClient);
long lastMsg = 0;
char msg1[20],msg2[20],msg3[20];
int value = 0;
float temp, hum,freq, light;
int tm = 0;
int dt,ta,ha;
```

```
void setup_wifi() {

  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  randomSeed(micros());

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

void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();
```

```
  // Switch on the LED if an 1 was received as first character
  if ((char)payload[0] == '1') {
    digitalWrite(BUILTIN_LED, LOW);   // Turn the LED on (Note that LOW is the voltage level
    // but actually the LED is on; this is because
    // it is acive low on the ESP-01)
  } else {
    digitalWrite(BUILTIN_LED, HIGH);  // Turn the LED off by making the voltage HIGH
  }

}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP8266Client-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str(),MQusername,MQpassword)) {
//    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      //client.publish(MQtopic1, "00000");
      // ... and resubscribe ---- Dont subscribe KP
      // client.subscribe("inTopic");
      // #KP - No announments --- No Subscribes.
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);


void setup() {
  pinMode(BUILTIN_LED, OUTPUT);     // Initialize the BUILTIN_LED pin as an output
  Serial.begin(115200);
  Serial.println("Wemos POWERING UP ......... ");
  Serial.print("Mac Address:");
  Serial.println(board_info.mac());
  setup_wifi();
  dht.begin();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

void loop() {

  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  long now = millis();
  dt = now - lastMsg ;
  if (dt > 2000) {
  }
   setup_dhtt();
}
```

```
void setup_dhtt()
{
  // put your main code here, to run repeatedly:
  Serial.println("Temperature and Humidity");
    temp = dht.readTemperature(true);
    hum = dht.readHumidity();
    light = (float) analogRead(A0);
    light = light/310;
    snprintf (msg1, 20, "%d", (int) temp);
    snprintf (msg2, 20, "%d", (int) hum);
    snprintf (msg3, 20, "%f", light);
    Serial.println(msg1);
    Serial.println(msg2);
    Serial.println(msg3);
    delay(5000);

    //tm = tm + dt/1000;
    //lastMsg = now;
    //ta = (DataTempMax-DataTempMin)/2.0;
    //ha = (DataHumMax-DataHumMin)/2.0;
    //freq = (float) 2.0*3.14169267/(DataCyclePeriod*60.0);
    //temp = DataTempMin + ta*(sin(freq*tm));
    //hum = DataHumMin+ha*(sin(freq*tm));
    //snprintf (msg1, 20, "%d", (int) temp);
    //snprintf (msg2, 20, "%d", (int) hum);

    Serial.print("Published :" );
    Serial.print(MQtopic1);
    Serial.print(" with value: " );
    Serial.println(msg1);
    client.publish(MQtopic1, msg1);
    Serial.print("Published :" );
    Serial.print(MQtopic2);
    Serial.print(" with value: " );
    Serial.println(msg2);

    Serial.print("Published :" );
    Serial.print(MQtopic3);
    Serial.print(" with value: " );
    Serial.println(msg3);
    client.publish(MQtopic3, msg3);

}
```

Figure F.1 - Code of the Arduino IDE

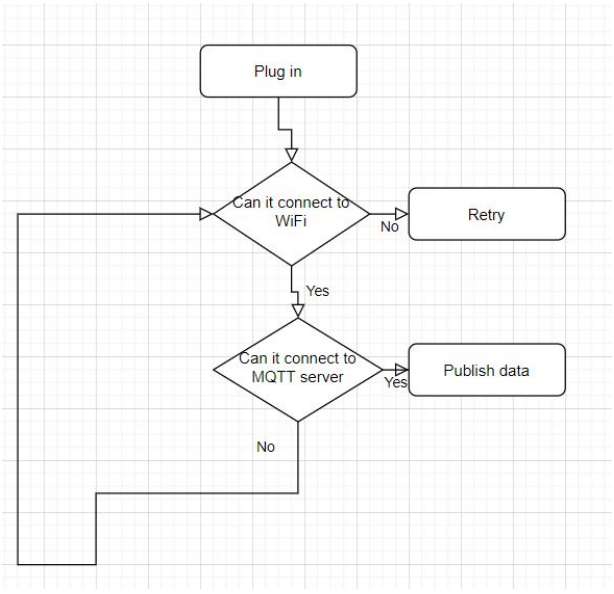


Figure F.2 - Flowchart for Arduino IDE

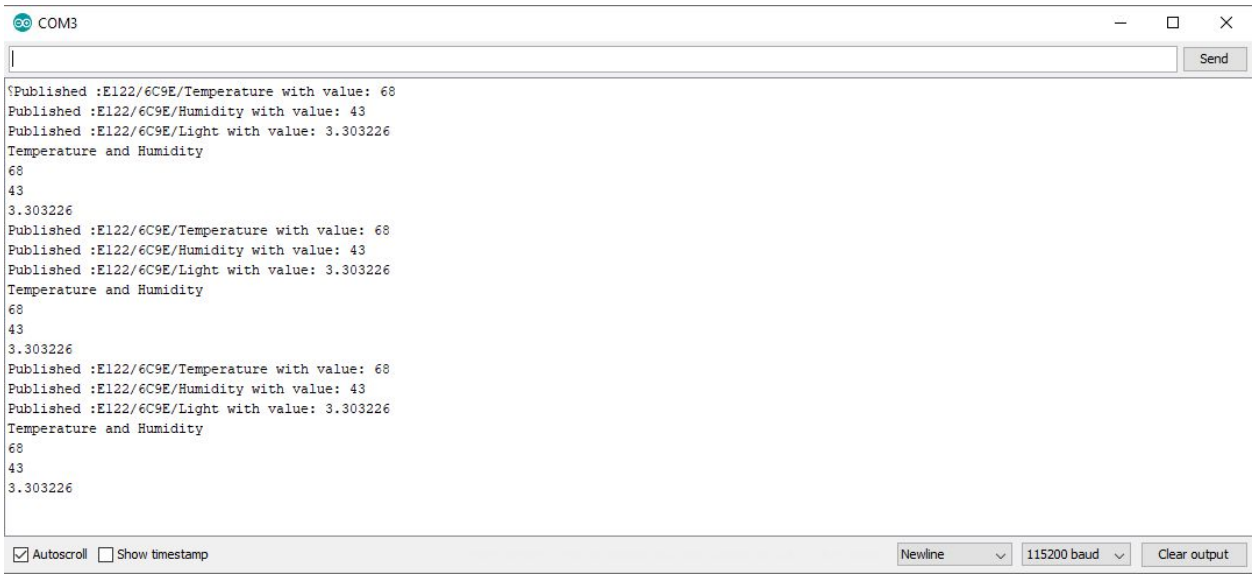


Figure G.1- Serial monitor

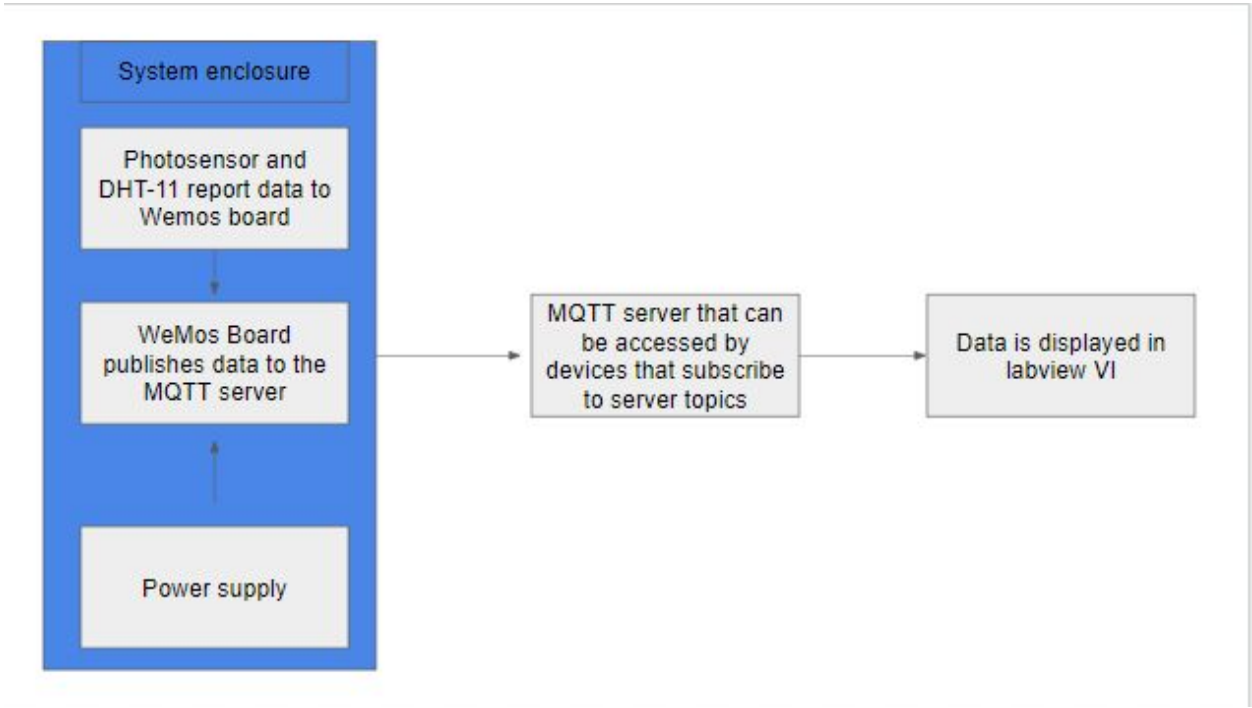


Figure H.1

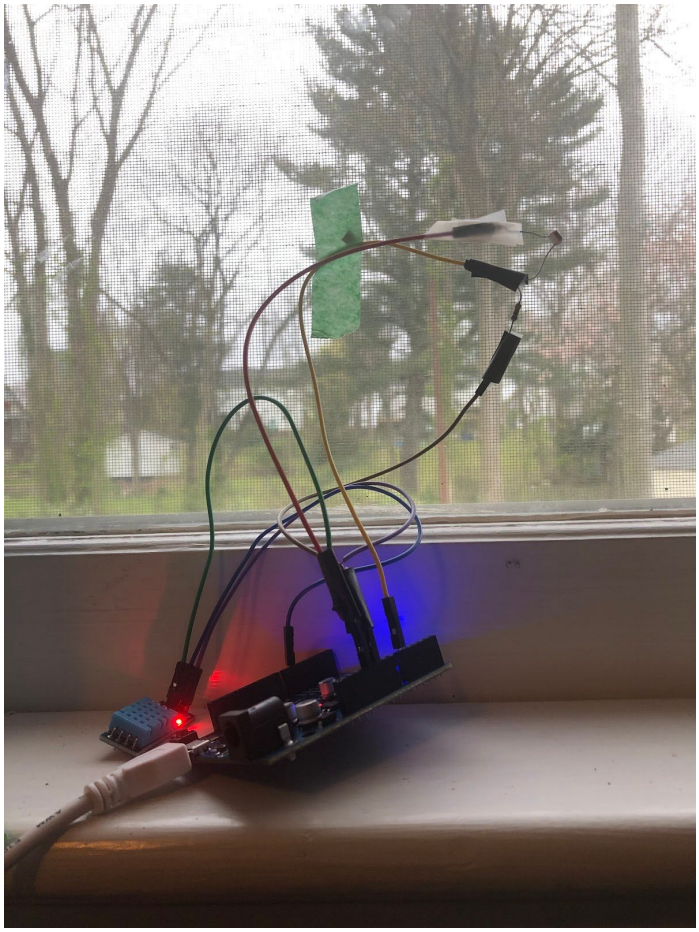


Figure I.1