Deliverable H - Prototype III and Customer Feedback

Nada El Rayes - 300143185
Amanda Beraldo Brandao de Souza - 300211045
DongYu Wang - 300114760
Aaron MacNeil - 300199522
Shayleen Ghanaat - 300198724

GNG 1103

University of Ottawa

March 28th, 2021

Table of contents

Introduction

JAMZ Automated Delivery is a drone delivery service focused mainly on the shipment of food from restaurants to the client. Their drones are ready for use, however, some essential features, like a climate sensor inside the package, are not functional yet. The main goal of this project is to develop a reliable solution that generates information about the content of the package during delivery. The device should provide valid and consistent data on the temperature and humidity of the food and send a warning to the drone's microcontroller (Raspberry Pi) when the conditions inside the package are not ideal. In this document, group D8 presents the improvements done based on client feedback, a detailed description of the third prototype, the testing process and the analysis of the results obtained.

Feedback

Table 1: Feedback obtained from the TA

Concept Title	Description	Feedback
Basic principles observed and reported	Knowledge generated underpinning hardware project concepts and presentation.	For the last prototype, the group has focused on coding the prototype. The group has started to work on the code for one DHT22 sensor. The TA has noted that the group is on the right track for the assembly of prototype 3. They suggest working on the code for one DHT22 sensor. When the code response was positive, we should rewrite the code for two DHT22 sensors.
Analytical and experimental critical function and/or characteristic proof of concept	Analytical studies place the technology in an appropriate context and laboratory demonstrations, modelling and simulation validate analytical prediction.	The group has faced problems while writing the code for the two DHT22 sensors. The group has asked for help from TA. The TA suggested multiple solutions. To check the wiring, the code has changed and been reviewed multiple times. Eventually, by updating the liberty the code worked. Both sensors were measuring the temperature and humanity.

		It was also suggested by the TA that this device will be better if it's able to send a warning to the drone's microcontroller (Raspberry Pi) when the conditions inside the package are not ideal. It was noted to decide on a range of temperature by measuring a sample of specific foods. Getting an average from this chosen temperature then decided above or below a specific temperature the device should send a warning to raspberry pie.

Project updates

Since the last prototype, we have received our new DHT22 sensors. We have updated the code to implement 2 sensors, and to report back the average values of the temperature and humidity readings. We have decided to wire together all components, and have decided to mount the fan into the housing. We also tested the code for reliable data readings and refined the code to only display the data readings with a warning message in the condition that the temperature or humidity data falls outside the acceptable threshold. As well, we updated the code to include a communication from one Arduino to another.

Prototype III

Description

Our third prototype called "SAAND III" is a physical comprehensive prototype composed of two sensors DHT22, one microcontroller Arduino Uno, a housing LeMotech ABS IP65 Junction Box and a cooling fan San Ace 60 103P0605H701.

Objectives

The objectives of this prototype are to develop and test a code that receives the measurements of temperature and humidity from the two sensors, analyzes this data and sends a warning to another microcontroller indicating that the temperature and/or humidity are not in the ideal range.

Analysis of critical components

Critical components for the sensor

- 2×sensors DHT22

Critical components for the microcontroller

- Arduino Uno

Critical components for the housing

- Housing LeMotech ABS
- Cooling fan San Ace 60 103P0605H701.

Analysis of system integration

The sensors will stay outside the housing to properly produce data of humidity and temperature, so it should be connected to the Arduino Uno through wires. A waterproof tube will connect the wiring from the microcontroller Arduino Uno that is inside the waterproof box to the sensor. The data received will be interpreted using a code, and a warning will be sent to the drone's microcontroller if the temperature is below or above the accepted range and/or the humidity is above a certain limit value. The fan will be on to control the temperature of the components inside the housing.

Feasibility analysis

The financial feasibility analysis has been done in the last deliverable, and the total project cost is $50.84 because the group already has the microcontroller. In addition, all the components are readily available, usually in online stores or the local equipment store, therefore it will be easy and economical even for future production. Thus, our design is both economically feasible and accessible.

Risk mitigation

Table 2: Risk description and mitigation approach

Risk Description	Mitigation Approach
Temperature sensor fails to operate completely	To solve the issue temperature sensors output voltage should be continuously monitored by an ADC or redundancy can be introduced (i.e use of 2 temperature sensors) and better calibration can be done by comparing outputs of 2 temperature sensors.

Temperature sensor is enabled when calibration data is changing	There can be an issue as the chip can stay in reset or can be stuck in reset loop, thus not allowing the chip to boot and execute the application program
Syntax errors in code	Have another program open which you can refer to, as in most cases the syntax and formatting are the same between different programs. For further help can refer to TA's
Unable to connect the sensor to the Arduino	Purchasing another sensor that the individual is familiar with
Fan does not respond	Check the wiring Maybe the fan inner part system has a disconnection

Stopping criteria for tests

- No substantial change (± 2 °C for temperature and ±5 points for humidity) on the readings from the sensors after one hour of testing
- Arduino used as Master receives the data sent from the Arduino used as Slave while testing the I2C communication

Tests

Tests were performed to evaluate the parameters of the prototype previously described. In this section, we describe in detail how the tests were held and what are the results obtained.

- **Code for the sensor**

FINAL_MASTERCODE_20210324

```
#include <Wire.h>

void setup(){
  Wire.begin();
  Serial.begin(9600);
}

void loop(){
  //request data from slave
  Wire.requestFrom(2,20);
  String string, string1, string2;// initialize strings
  do{
    char c = Wire.read();
    string = string + c; // whole string temp + hum
    string1 = string.substring(0,8); // string for temperature
    string2 = string.substring(9);// string for humidity
  } while(Wire.available());{ //print the warnig on serial monitor
    Serial.print("WARNING!");
    Serial.println();
    Serial.println();
    Serial.println();
    Serial.print("Temperature: " );
    Serial.print(string1);
    Serial.print(""C");


    Serial.println();
    Serial.print("Humidity: " );
    Serial.print(string2);
    Serial.print("% ");
    Serial.println("");
    Serial.println("");
    Serial.println("");

    delay(500);
  }
  delay(500);
  }
```

FINAL_SLAVECODE_20210324

```
//SLAVECODE WITH NUMBERS
// Example testing sketch for various DHT humidity/temperature sensors
// Written by ladyada, public domain

// REQUIRES the following Arduino libraries:
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit_Sensor

#include <DHT.h>
#include<Wire.h>

#define DHTPIN1 8
#define DHTPIN2 7
// Digital pin connected to the DHT sensor
// Feather HUZZAH ESP8266 note: use pins 3, 4, 5, 12, 13 or 14 --
// Pin 15 can work but DHT must be disconnected during program upload.

// Uncomment whatever type you're using!
#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321


// Connect pin 1 (on the left) of the sensor to +5V
// NOTE: If using a board with 3.3V logic like an Arduino Due connect pin 1
// to 3.3V instead of 5V!
// Connect pin 2 of the sensor to whatever your DHTPIN is
// Connect pin 3 (on the right) of the sensor to GROUND (if your sensor has 3 pins)
// Connect pin 4 (on the right) of the sensor to GROUND and leave the pin 3 EMPTY (if your sensor has 4 pins)
// Connect a 10K resistor from pin 2 (data) to pin 1 (power) of the sensor

// Initialize DHT sensor.
// Note that older versions of this library took an optional third parameter to
// tweak the timings for faster processors.  This parameter is no longer needed
// as the current DHT reading algorithm adjusts itself to work on faster procs.
DHT dht1(8, DHTTYPE);
DHT dht2(7, DHTTYPE);

#define SLAVE_ADDR 9  //define Slave I2C Address
 //Define Slave answer size


char wy[7];
char ix[8];

float tavg; //average temperature
float havg; //average humidity
```

Figure 1.1 - Code that reads data from DHT22 and transmits it to the second microcontroller used

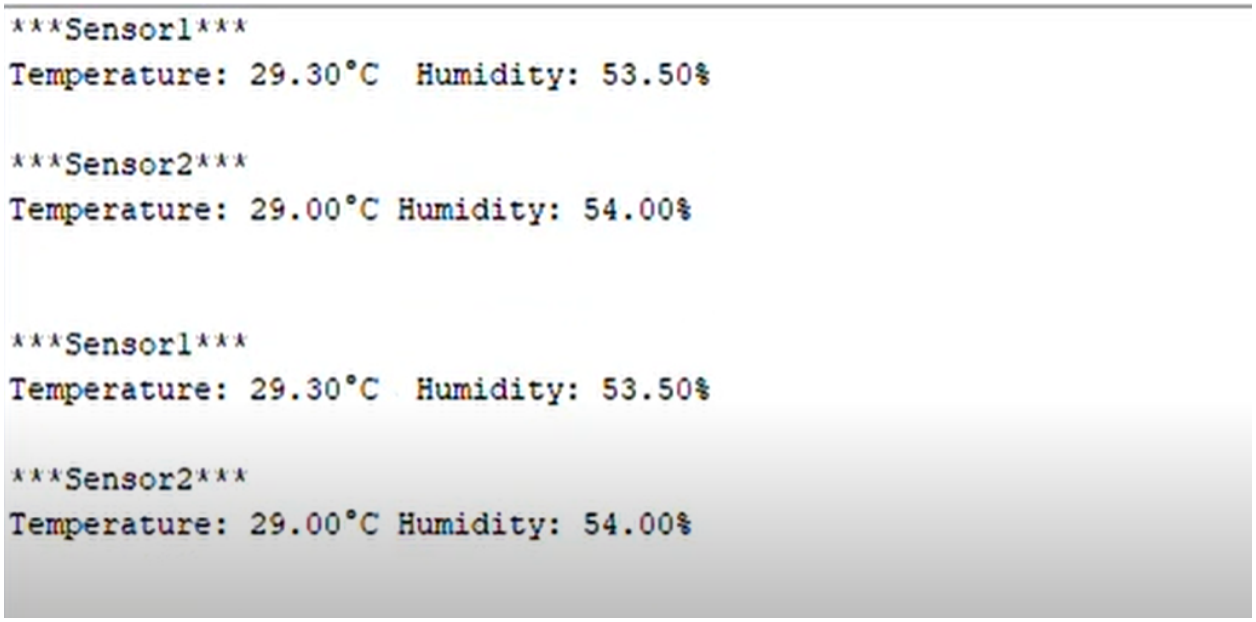

Figure 1.2 - Data received from the sensor DHT22 on the Arduino's Uno serial monitor

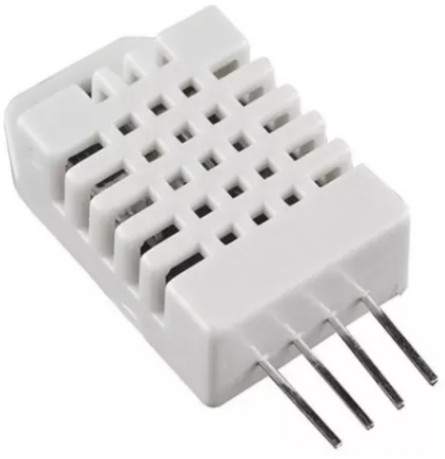

Figure 1.3 - Sensor DHT22

Testing method

Analytical Prototype Testing

Estimate Duration Time

1 hour

Description of Prototype

Code that receives the information from the sensor and prints it on the serial monitor. For this code we don't have a specific condition for the transmission, so the communication happens all the time.

Description of results to be recorded and how these results will be used

Accurate readings from the two sensors DHT22. This data will be read in the first Arduino and transmitted to the other microcontroller.

- **Communication between microcontrollers**

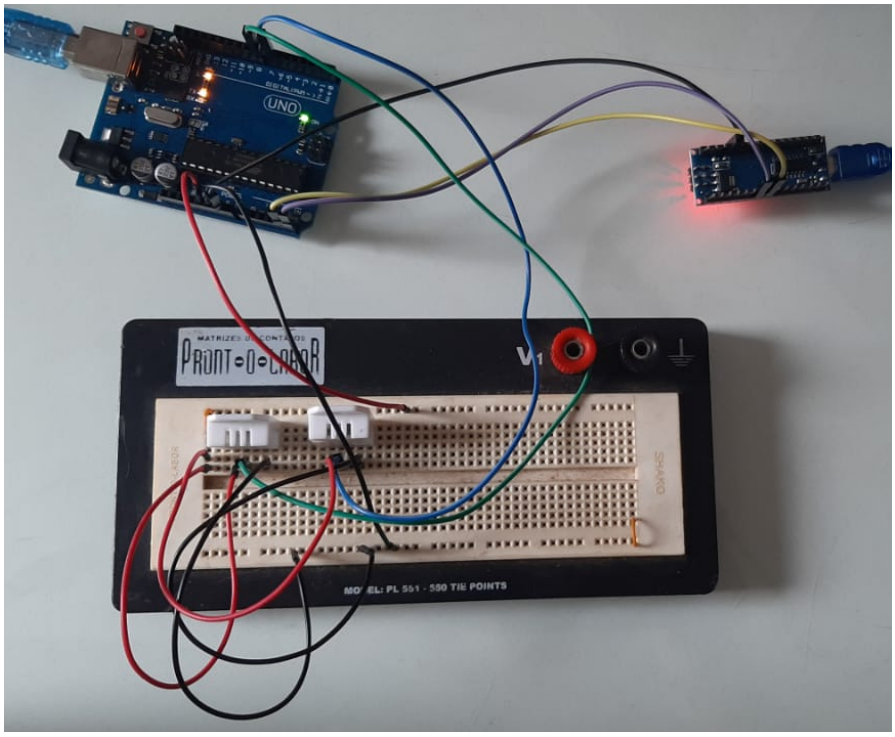

Figure 2.1 - Circuit using two DHT22 sensors, one Arduino Uno and one Arduino Mini

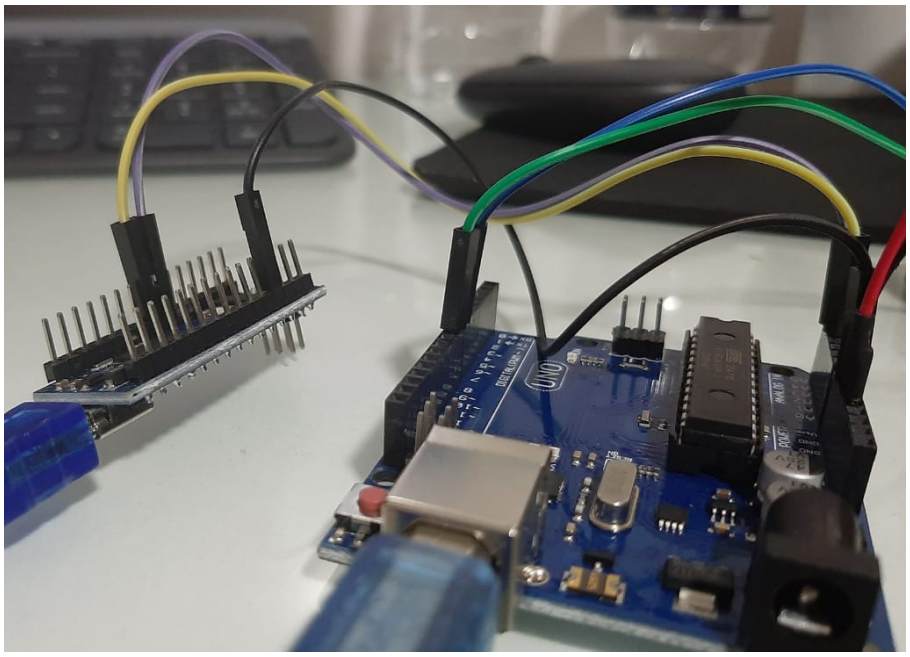

Figure 2.2 Arduino Uno connected to the Arduino Mini using the GND, A4 and A5 pins

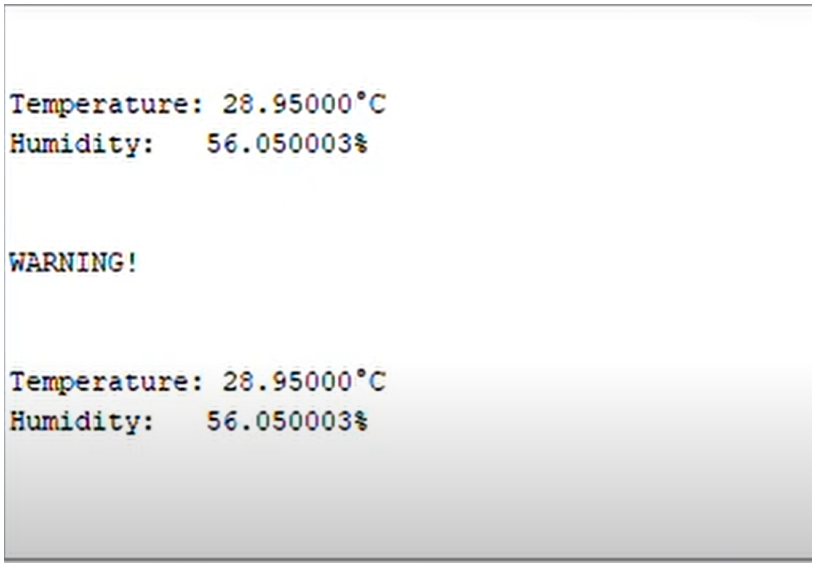

Figure 2.3 - Warning message and temperature and humidity readings on the Arduino's Mini serial monitor

Testing method

Analytical Prototype Testing

Estimate Duration Time

1 hour

Description of Prototype

Code that sends the information from the Arduino Uno to the Arduino Mini using I2C communication

Description of results to be recorded and how these results will be used

Warning and temperature and humidity readings on the Arduino's Mini serial monitor. These results will define if the data and the warning message are effectively transmitted from the Arduino Uno to the Arduino Mini.

Conclusion

After the tests were performed, it was concluded that the readings from the sensors were accurate and that the communication between the microcontrollers is efficient. We defined that the average between the readings of the two sensors will be the value that activates the communication between microcontrollers. This activation will only happen when this average reading is between 0°C and 19°C and when the humidity is above 60%. For getting these values we used sample measurements of specific foods (examples: Ice cream, toast, tea etc) using a thermometer. To define this range, most of the ideal

temperatures tested were above 19°C and below 0°C. The humidity value was obtained by measuring the humidity when drinks would spill in a controlled environment. For the next steps, we will implement the condition of the temperature in the code and assembly the components.