Deliverable G

Prototype 2 and Client Feedback

GNG 1103
Design

To:
Professor David Knox
And
Pankaj Kumar Rathi/Chetan Kumbhar

By:
Jack Bridgeland
Wei Chen
Nicolas O'Brien
Balpreet Singh
Johann Wehrstedt

Team: Eternal Hoptimists
Submission Date: November 13th, 2022

University of Ottawa
Faculty of Engineering

Table of Contents

Introduction

In this report, our team will discuss the feedback received during client meeting 3 with Shane from Beyond the Pale, our team will be outlining the different prototypes used in phase 2 of our prototyping sequence and we will be outlining a new prototyping test plan for the final prototype.

Feedback from Client Meeting 3

On November 8, 2022, our team was able to have our 3rd client meeting with Shane from Beyond the Pale Brewing Company. This meeting involved a pitch presentation which was given by Jack Bridgeland. During this presentation, Jack pitched our current idea of using the difference in height and pressure to determine density which can then allow us to determine specific gravity. Jack was also able to ask Shane a couple questions and receive some feedback. Shane’s major comments and concerns were that our pipe casing which is in the tank must be food-safe due to regulations placed on companies like Beyond the Pale by the Canadian Food Inspection Agency and he also re-stated that he wants the device to be easily installable to simplify the lab techs job. To ensure easy installation, we will be using quick-connect wires to efficiently connect the different pieces of the system. To ensure our device is food safe, the housing for the pressure sensors will be made of stainless steel (The prototype will be made from aluminum due to the access we have to aluminum).

Prototyping

Clock Module

To prototype the clock module, I wired the real-time clock module to the Arduino using 4 jumper cables as seen below in figure 1. The first step to prototyping this module was to first set the time. Once the time was set, I unplugged the sensor for the Arduino and let it sit unplugged for 16 hours. After those 16 hours, I ran a code to display what day and time it was, and the sensor correctly displayed the correct time as seen in figure 2.

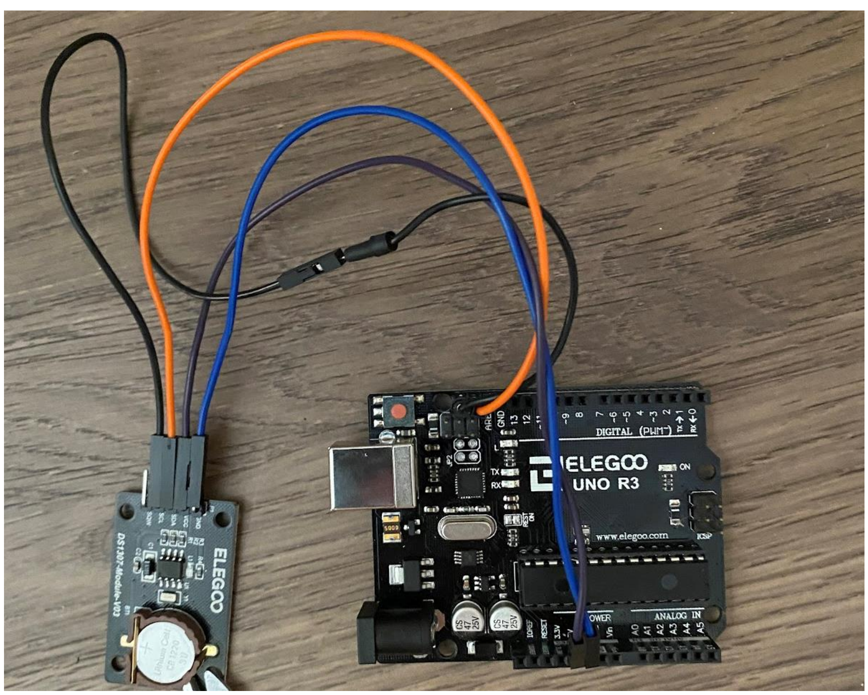

Figure 1: Clock Module Prototype

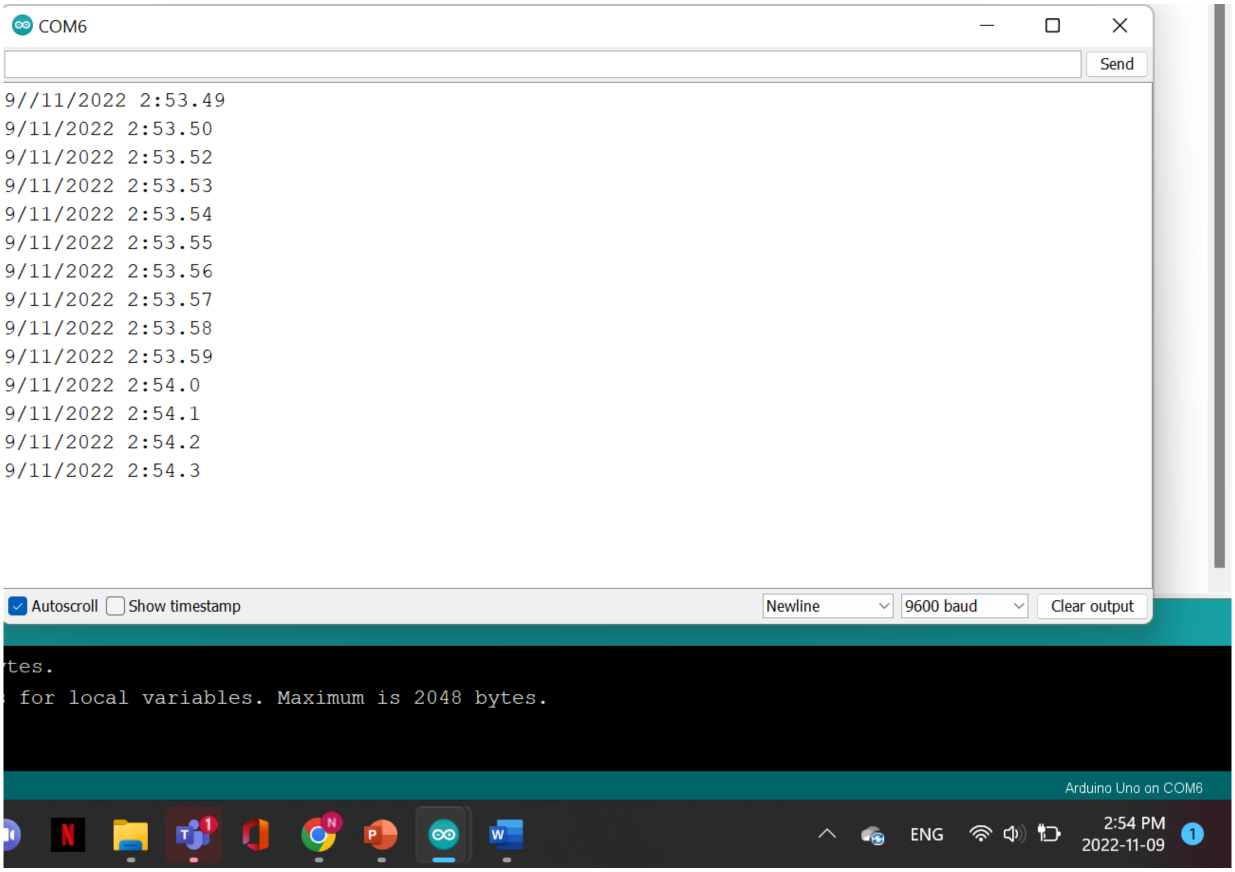

Figure 2: Date/Time Output from the Real-Time Clock Module vs Date/Time on Laptop

Code for Clock Module

```
#include <Wire.h>
#include <ds3231.h>

struct ts t;

void setup() {
 Serial.begin(9600);
 Wire.begin();
 DS3231_init(DS3231_CONTROL_INTCN);

 //t.hour=10; //Set following parameters to current time if clock isnt set up and uncomment
the lines
 //t.min=53;
 //t.sec=30;
 //t.mday=8;
 //t.mon=11;
 //t.year=2022+100;
 //DS3231_set(t);
```

```
}
void loop() {
  DS3231_get(&t);

  Serial.print(t.mday); //Displays current data and time
  Serial.print("/");
  Serial.print(t.mon);
  Serial.print("/");
  Serial.print(t.year);
  Serial.print(" ");
  Serial.print(t.hour);
  Serial.print(":");
  Serial.print(t.min);
  Serial.print(".");
  Serial.println(t.sec);

  delay(1000);
}
```

Bluetooth Sensor

To test our Bluetooth sensor, our HC-05 Bluetooth module will be connected to the Arduino as seen in figure 3. To ensure our system can connect to our software and transfer the correct data, we will be using dummy variables with known values, and we will test to see if our software is able to receive them. We will stop testing until the correct values are being received. Unfortunately, our team was only able to establish one-way connection from the module making me believe the transmission pin somehow got fried. We will look into other methods on how to send data to our computer. We were able to send commands from my tablet to the Arduino as seen in figure 4 and 4’ but commands were not being received from the Arduino to the tablet.

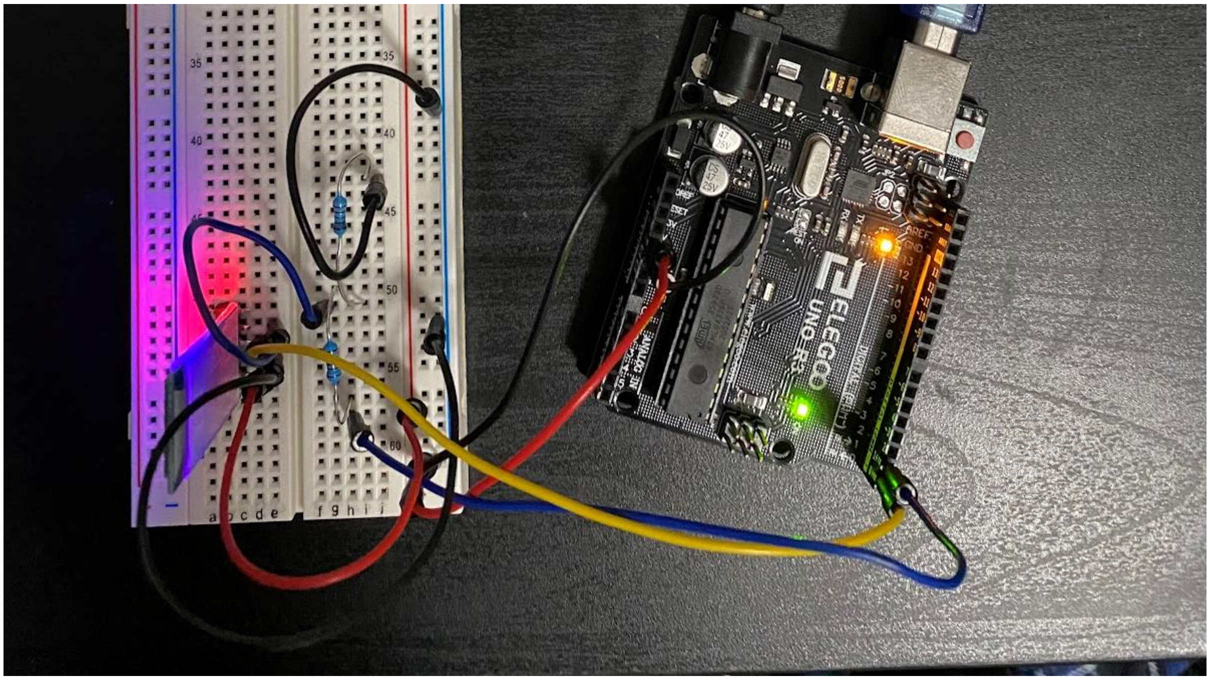

Figure 3: Bluetooth Prototype

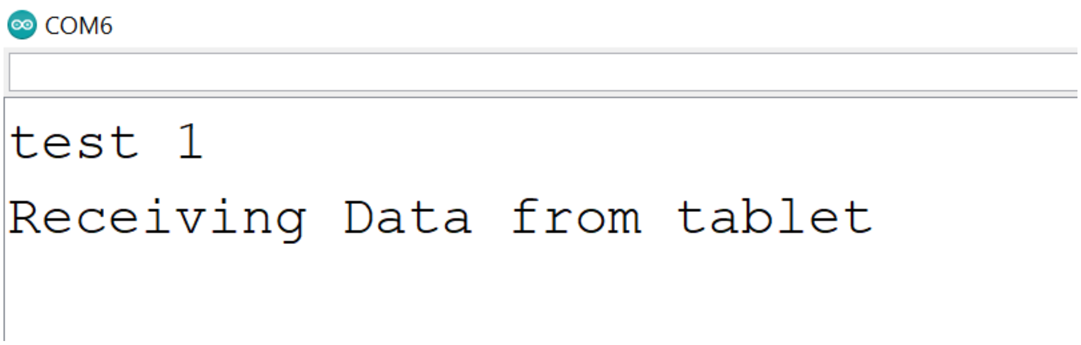

Figure 4: Data being Received by Bluetooth Connection

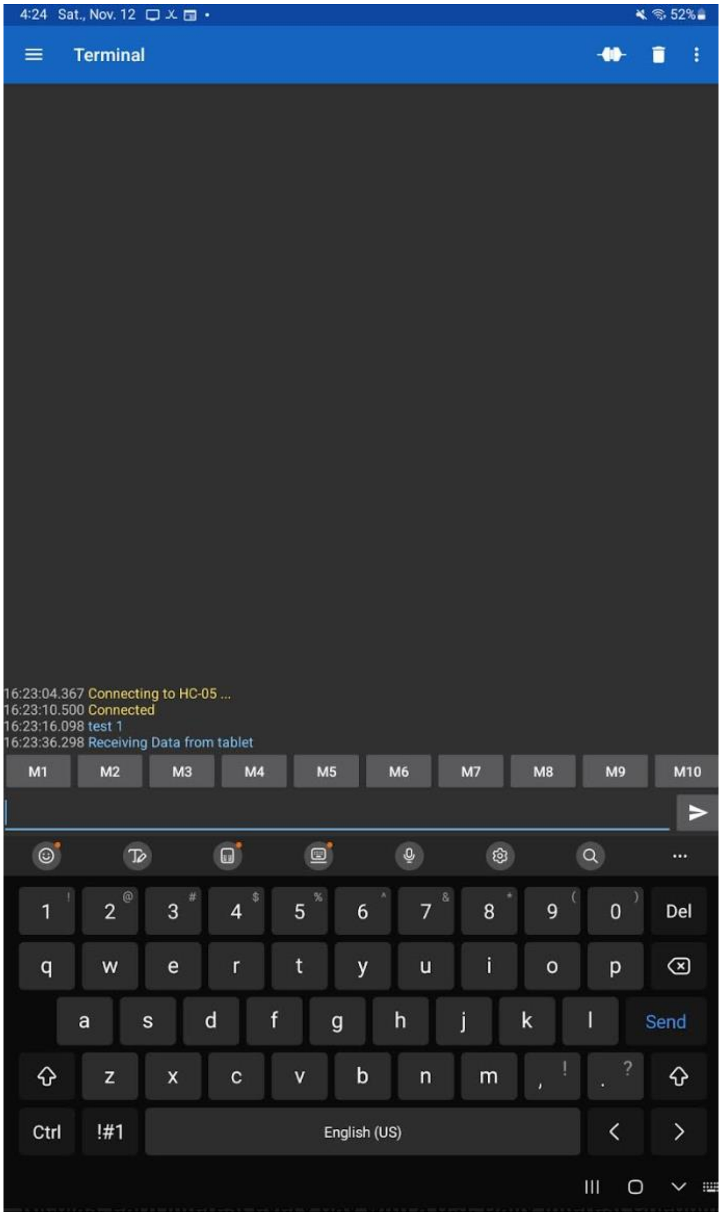

Figure 4': Data being sent to the Arduino

Pressure Sensor + Temperature Sensor

To prove that our concept works, we decided to prototype it by wiring our 2 pressure sensors and the temperature sensor to the Arduino using a breadboard and jumper wires as seen in Figure 6. To simulate the tank, we grabbed a bucket and filled it with water. We then submerged the bottom pressure sensor in the bottom of the bucket, and we held the other sensor 0.27305m higher than that point as seen in figure 5. By doing this, we were able to get specific gravity in a consistent range of 0.94-1.04. A screenshot of these results cannot be provided as our computer crashed after our prototype was taken apart deleting the COM port with the data, but we had our written data to keep track of our results. The expected result of this experiment was to be 0.99 so we had an error of approximately ±0.05. Since we were holding the sensor in place with our hands, our height was as precise as human error allows for so an error of ±0.05 with human error is very good as a proof of concept. It could have also been due to not having a perfect measurement of the height as well. It is our hope to be able to get a more accurate reading of the specific gravity in our next prototype test.

Figure 5: Pressure Sensor Prototype

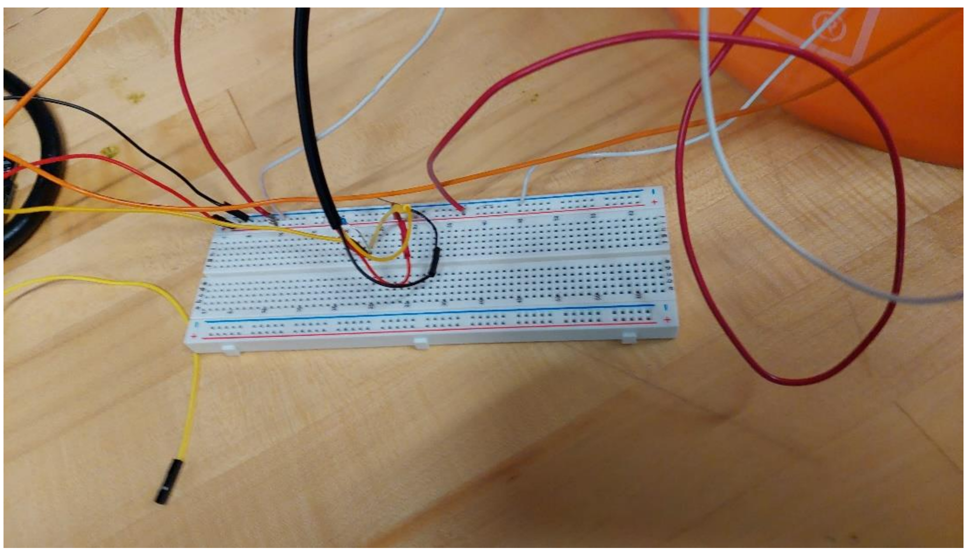

Figure 6: Prototype Wiring

Code for Pressure Sensor + Temperature Sensor

```
#include <Wire.h>
#include <OneWire.h>
```

```
#include <DallasTemperature.h>

int Psensor1 = A2;
int Psensor2 = A3;

float APZero = 102.4 ; //Analog Pressure Reading when P = 0 Psi
float APMid= 512; //Analog Pressure Reading when P=15 Psi
float APMax = 921.6; //Analog Pressure Reading when P = 30 Psi
int PressureMax = 30; // Max rated Pressure of Transducer is 30 Psi

float height = 0.27305; // This is the constant difference in Height in meters of the 2 Pressure
Transducers
float g = 9.81; //Gravitational Constant in N/kg or m/s^2
float row; // Value for the density of the fluid
float rowWater = 998.23; //Density of water at 20 degree Celsius in Kg/m^3
float SG; //Value will be used to Store SG value
float Plato; //Value will be used to store Plato conversion

float TopPressure;// Value will be used to store Psi value
float BottomPressure; // Value will be used to store Psi value

float PaTop; // Value will be used to convert and store Psi to Pa
float PaBottom;// Value will be used to convert and store Psi to Pa

float tempValue; // Value to Store Temperature Value
int tempPin=2;

OneWire oneWirePin(tempPin); //Defining temperature sensor
DallasTemperature sensors(&oneWirePin); //Passing sensor through Dallas Temperature

void setup() {

Serial.begin(9600);
sensors.begin();//Starting sensor to request data

}

void loop() {
TopPressure = analogRead(Psensor1); //Reads Analog value from the top pressure sensor
TopPressure = ((TopPressure - APZero)*PressureMax)/(APMax - APZero); //Converting Analog
Pressure to Psi
PaTop = TopPressure*6894.76; //Converting Psi to Pa

BottomPressure = analogRead(Psensor2); //Reads Analog value from bottom pressure sensor
```

```
BottomPressure = ((BottomPressure - APZero)*PressureMax)/(APMax - APZero); //Converting
Analog Reading to Psi
PaBottom = BottomPressure*6894.76; //Converting Psi to Pa

row = (PaBottom-PaTop)/(g*height);

SG = row/rowWater;

Plato = (-1*616.868)+(1111.14*SG)-(630.272*SG*SG)+(135.997*SG*SG*SG);

 sensors.requestTemperatures();
 tempValue = sensors.getTempCByIndex(0);// Getting Temperature Value

Serial.println("Pressure 1:");
Serial.println(TopPressure);
Serial.println("Pressure 2:");
Serial.println(BottomPressure);
Serial.println("SG:");
Serial.println(SG);
Serial.print("Temperature:");
Serial.println(tempValue);

delay(5000);

}
```

Final Prototype

Our final prototype will be completed for Prototype 3. Our code is currently all written and our prototype just needs to be wired.

Code for Final System

```
#include <IRremote.hpp>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <ds3231.h>

SoftwareSerial bluetooth(0, 1); //RX, TX

int Psensor1 = A2;
int Psensor2 = A3;
```

```
int tempPin = 2;
int IR_RECEIVE_PIN = 3; //Connect IR sensor to Pin 3
int buttonpin = 4; //Connect Button to Pin 4

int i; // Used to enter in loop to measure data

float APZero = 102.4 ; //Analog Pressure Reading when P = 0 Psi
float APMid = 512; //Analog Pressure Reading when P=15Psi
float APMax = 921.6; //Analog Pressure Reading when P = 30 Psi
int PressureMax = 30; // Max rated Pressure of Transducer is 30 Psi

float height = 1; // This is the constant difference in Height in meters of the 2 Pressure
Transducers
float g = 9.81; //Gravitational Constant in N/kg or m/s^2
float row; // Value for the density of the fluid
float rowWater = 998.23; //Density of water at 20 degree Celsius in Kg/m^3
float SG; //Value will be used to Store SG value
float Plato; //Value will be used to store Plato conversion

float TopPressure;// Value will be used to store Psi value
float BottomPressure; // Value will be used to store Psi value
float PaTop; // Value will be used to convert and store Psi to Pa
float PaBottom;// Value will be used to convert and store Psi to Pa

float tempValue; // Value to Store Temperature Value
int buttonState; // Value to Save Current State of the Button

struct ts t; //Array for RTC

OneWire oneWirePin(tempPin); //Defining temperature sensor
DallasTemperature sensors(&oneWirePin); //Passing sensor through Dallas Temperature
LiquidCrystal_I2C lcd = LiquidCrystal_I2C(0x27, 16, 2); //For 16 x 2 LCD Display

void setup()
{
  IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK); // Start the receiver
  Serial.begin(9600); //Start Serial monitor
  sensors.begin();//Starting Temp sensor to request data
  lcd.init(); //Initiallizing LCD Screen
  lcd.backlight();//Initiallizing LCD Screen
  pinMode(buttonpin, INPUT); // Set buttonpin as an input
  bluetooth.begin(9600); //Start Bluetooth
```

```
  Wire.begin();
  DS3231_init(DS3231_CONTROL_INTCN);

  //t.hour=10; //Next 7 Lines are used to set time. If clock dies, replace battery and replace
values with your time to reset time
  //t.min=53; //switches minutes to 53
  //t.sec=30; //switches seconds to 30
  //t.mday=8; //switches day to the 8
  //t.mon=11; //Switches month to the november
  //t.year=2022 // switches year to 2022
  //DS3231_set(t); //sets clock to selected time
}

void loop() {

  if (IrReceiver.decode()) // Loop to receive input from remote
  {
    switch (IrReceiver.decodedIRData.decodedRawData) {
      case 0xBA45FF00://This is for the Start button on the remote
        i = 1; //Set i=1 to enter in loop to start measuring and saving data
        break;
      case 0xB946FF00://This is for the vol+ button
        i = 0; //This is to stop the loop
        break;
    }
    IrReceiver.resume(); // Enable receiving of the next value
  }

  if (i == 0) // If vol+ button is hit, it stops measuring data and displays following message
  {
    lcd.setCursor(0, 0); // Sets cursor on first row, first column
    lcd.print("Hit Start to");
    lcd.setCursor(5, 1); //Sets curson on fifth row, first column
    lcd.print("begin");
  }

  if (i == 1) //If Start button is hit, enter this function to begin measurements
  {
    sensors.requestTemperatures();
    tempValue = sensors.getTempCByIndex(0); // Storing Temperature Value

    TopPressure = analogRead(Psensor1); //Reads Analog value from the top pressure sensor
    TopPressure = ((TopPressure - APZero) * PressureMax) / (APMax - APZero); //Converting
Analog Pressure to Psi
```

```
  PaTop = 6.89476 * TopPressure * 1000; //Converting Psi to Pa

  BottomPressure = analogRead(Psensor2); //Reads Analog value from bottom pressure sensor
  BottomPressure = ((BottomPressure - APZero) * PressureMax) / (APMax - APZero);
//Converting Analog Reading to Psi
  PaBottom = 6.89476 * BottomPressure * 1000; //Converting Psi to Pa

  row = (PaBottom - PaTop) / (g * height);
  SG = row / rowWater;
  Plato = (-1 * 616.868) + (1111.14 * SG) - (630.272 * SG * SG) + (135.997 * SG * SG * SG);

  buttonState = digitalRead(buttonpin); //Read current state of button

  if (buttonState == HIGH) {
    lcd.setCursor(0, 0); // Sets cursor on first row, first column
    lcd.print("Plato:");
    lcd.print(Plato); // Print specific gravity in Plato
    lcd.setCursor(0, 1); //Sets cursor on first row, second column
    lcd.print("SG:");
    lcd.print(SG);//Print Specific Gravity
    lcd.print("kg/m3");
    delay(500);
    lcd.clear();
  }
  else if (buttonState == LOW) {
    lcd.setCursor(0, 0); // Sets cursor on first row, first column
    lcd.print("Temperature :");
    lcd.print(tempValue); // Print Temperature value on the screen
    lcd.setCursor(0, 1); //Sets cursor on first row, second column
    lcd.print(""); //Print whatever we need
    delay(500);
    lcd.clear();
  }

  if (Serial.available() > 0)      // Send data only when you receive data:
  {
    DS3231_get(&t);

    bluetooth.print(t.mday); // send current day
    bluetooth.print("/"); // send a dash
    bluetooth.print(t.mon); // send current month
    bluetooth.print("/"); // send a dash
    bluetooth.print(t.year); // send current year
    bluetooth.print(" ");// send a space
```

```
    bluetooth.print(t.hour); // send current hour
    bluetooth.print(":"); // send a semi colon
    bluetooth.print(t.min); //send current minutes
    bluetooth.print("."); //send a dot
    bluetooth.println(t.sec); //send current seconds
    bluetooth.print(","); //Send a comma
    bluetooth.print(Plato); //Send value of Plato
    bluetooth.print(","); //Send a comma
    bluetooth.print(tempValue); //Send value of Temperature
    bluetooth.print(";"); //Send a semi-colon
   }
  }
}
```

Analysis

Pressure Sensor Analysis

During our test, we measured a high difference of 0.27305m between the bottom of the bucket and an arbitrary point we measure and marked on the bucket. Knowing that gravity is 9.81 m/s^2, we are able to determine the theoretical SG of our fluid which is water for this experiment according to the following equation:

SG = density of fluid/density of water
SG = density of water/density of water
SG = 1

Since the density of water is known, we can use the following equation to determine the expected difference in pressure:

$$\Delta P = p g \Delta h$$
$$\Delta P = 1000 * 9.81*0.27305$$
$$\Delta P \text{ theoretical} = 2678.621 \text{ Pa} = 0.3885 \text{ Psi}$$

During our test, we needed to hold the top pressure sensor with our hands which led to us getting values ranging between 0.94 – 1.04 for SG caused by human error due to fatigue of holding the sensor in place with shaky hands. This led to the following range of ΔP:

$$\Delta P \text{ min}=940*9.81*0.27305$$
$$\Delta P \text{ min}= 2517.9 \text{ Pa}$$

$$\Delta P \text{ max}=1040*9.81*0.27305$$
$$\Delta P \text{ min}= 2785.77 \text{ Pa}$$

$$2517.9 \text{ Pa}< \Delta P \text{ experimental} < 2785.77$$

To determine this initial error, we are able to use the following formula

$$\text{Error} = \frac{\Delta\text{Pexperimental} - \Delta\text{P theoretical}}{\Delta\text{P theoretical}} x100$$

$$\text{Low bound error} = \frac{2517.9 - 2678.621}{2678.621} x100$$

Low Bound Error = -6%

$$\text{Higher Bound error} = \frac{2785.77 - 2678.621}{2678.621} x100$$

Higher Bound Error = 4%

-6% < Error < 4%

Changes Since Deliverable F

Since the last deliverable, not much has changed in the objective of our project, the only change is regarding our Bluetooth module. The Makerspace/Maker Lab did not have the HC-05 module we needed so we needed to get one from Amazon. This change can be seen in our updated BOM below highlighted in yellow. The price of the sensors on Makerspace/Maker Lab was 12.99$ while on Amazon, it is 13.78$ so it is an increase of 0.79$.

Updated BOM

Table 1: Updated Bill of Materials

Parts	Quantity	Unit of Measure	Unit Cost ($)	Estimate Cost($)	Link
Hardware Required					
Mil. Spec Aluminum Tubing 1OD 1ft Length	2	FT	0	0	Owned
Aluminum Rod 1-5/8 Dia 1ft Length	1	FT	0	0	Owned
Aluminum 2x6x1-1/2	1	EA	0	0	Owned
Food Grade Silicone Sealant	1	EA	0	0	Owned
3D printed Electrical Compartment Bottom	1	EA	0	0	MakerLab (to be printed)
3D Printing Filament	1	EA	10	10	MakerLab
3D printed Electrical Compartment Top	1	EA	0	0	MakerLab (to be printed)
O-Rings	6	EA	0	0	Owned
Screws	8	EA	0	0	Owned

Electrical Components Required					
Arduino	1	EA	0	0	Owned
Breadboard	1	EA	0	0	Owned
LCD Screen	1	EA	0	0	Owned
Pressure Transducer Sensor 30 Psi	2	EA	13.49	26.98	See Deliverable F
Latching Buttons	1	EA	0.54	0.54	See Deliverable F
Bluetooth connection	1	EA	13.78	13.78	See Link Section
DS18B20 Temperature Sensor	1	EA	14	14	See Deliverable F
IR Sensor	1	EA	0	0	Owned
Clock Module	1	EA	10	10	See Deliverable F
Remote	1	EA	0	0	Owned
4.7 KΩ Resistor	1	EA	0	0	Owned
10 KΩ Resistor	1	EA	0	0	Owned
10 KΩ Potentiometer	1	EA	0	0	Owned
Wiring	50	EA	0	0	Owned
Software Required					
Arduino IDE	1	EA	0	0	Owned
OnShape	1	EA	0	0	Owned
Python	1	EA	0	0	Owned
Libraries					
Liquid Crystal	1	EA	0	0	See Library Links
OneWire	1	EA	0	0	See Library Links
Spi	1	EA	0	0	See Library Links
DS3231	1	EA	0	0	See Library Links
DallasTemperature	1	EA	0	0	See Library Links
Irremote	1	EA	0	0	See Library Links
Software Serial	1	EA	0	0	See Library Links
Total Cost (Without Tax)				75.30	
Total Cost(With Tax)				81.90	

Spreadsheet links:
https://docs.google.com/spreadsheets/d/1HLeKHimS2K6xT5xADkeC8T_iUnEW0eZaAFPuYZtyU8s/edit#gid=0

Updated Component Links

To see all component links, please see the spreadsheet link or see Deliverable F

Bluetooth Module: https://www.amazon.ca/Wireless-Bluetooth-Module-Transceive-Serial/dp/B08GXTJ2V9/ref=sr_1_2_sspa?gclid=Cj0KCQiAmaibBhCAARIsAKUlaKSiJHGQFjf6SyfgH6WRO96vIeFgJU-hos0QvWp5MAXcjAcd4FZ1qs8aAnwJEALw_wcB&hvadid=596027847757&hvdev=c&hvlocphy=1002376&hvnetw=g&hvqmt=e&hvrand=9460209702837723489&hvtargid=kwd-299124711286&hydadcr=4679_9338026&keywords=hc-05+wireless+bluetooth&qid=1667947866&qu=eyJxc2MiOiIxLjczIiwicXNhIjoiMC4wMCIsInFzcCI6IjAuMDAifQ%3D%3D&s=electronics&sr=1-2-spons&psc=1

Prototype Test Plan

Table 2: Remaining Prototype Tests

ID	Test Objective	Description of Prototype	Results to be Recorded	Duration of Test
1	Determine accuracy of pressure sensor	Our 2 pressure sensors will be connected to an Arduino and be tested in different liquids of known density at different heights to test accuracy	1.height of completed test 2.the fluid used 3.temperature of fluid 4.difference from theoretical value	Test will last until we are within our design spec accuracy
2	Determine if the whole system is working	This prototype will consist of 2 pressure sensors, 1 temperature sensor, 1 Arduino, 1 LCD screen, 1 IR receiver, 1 button, 1 Bluetooth module, 1 real-clock module.	1.Any bugs with our code 2.Input conditions	Test until the system works as planned

3D Printing Requirements

Our team will be printing the case shown in figure 7. This case contains a slot for the remote control to sit, a hole for the LCD screen, a hole for access to the battery, a hole for the button, a hole for the IR receiver and a threaded cylinder slot at the back of the case to connect it to the rest of the system as seen in figure 8. This print will take approximately 14 hours and 50 minutes.

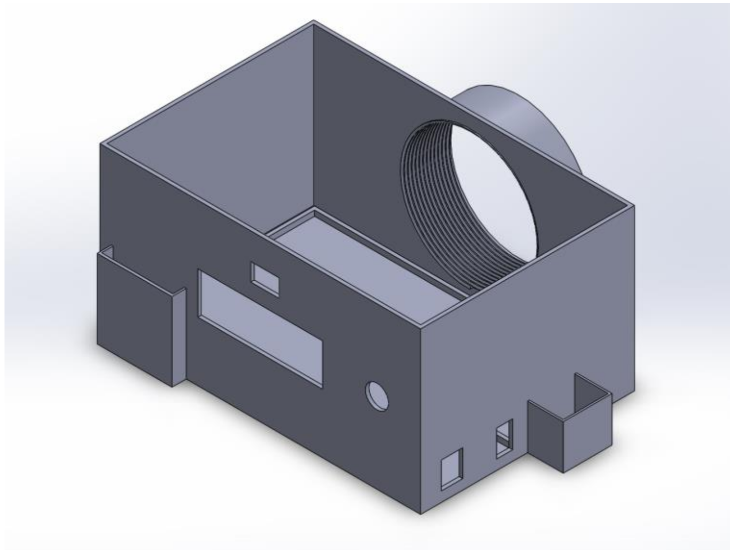

Figure 7: Solidworks Case Model

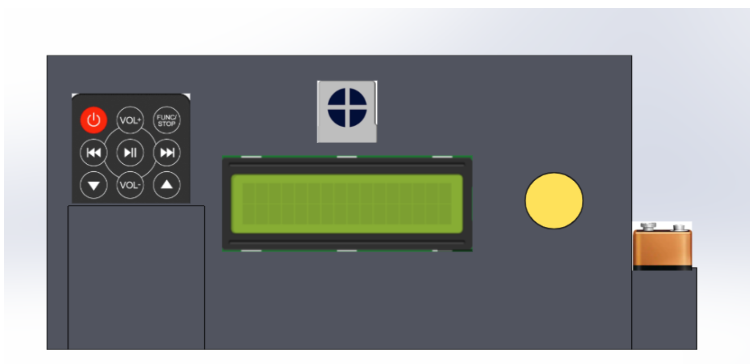

Figure 8: Animated Model

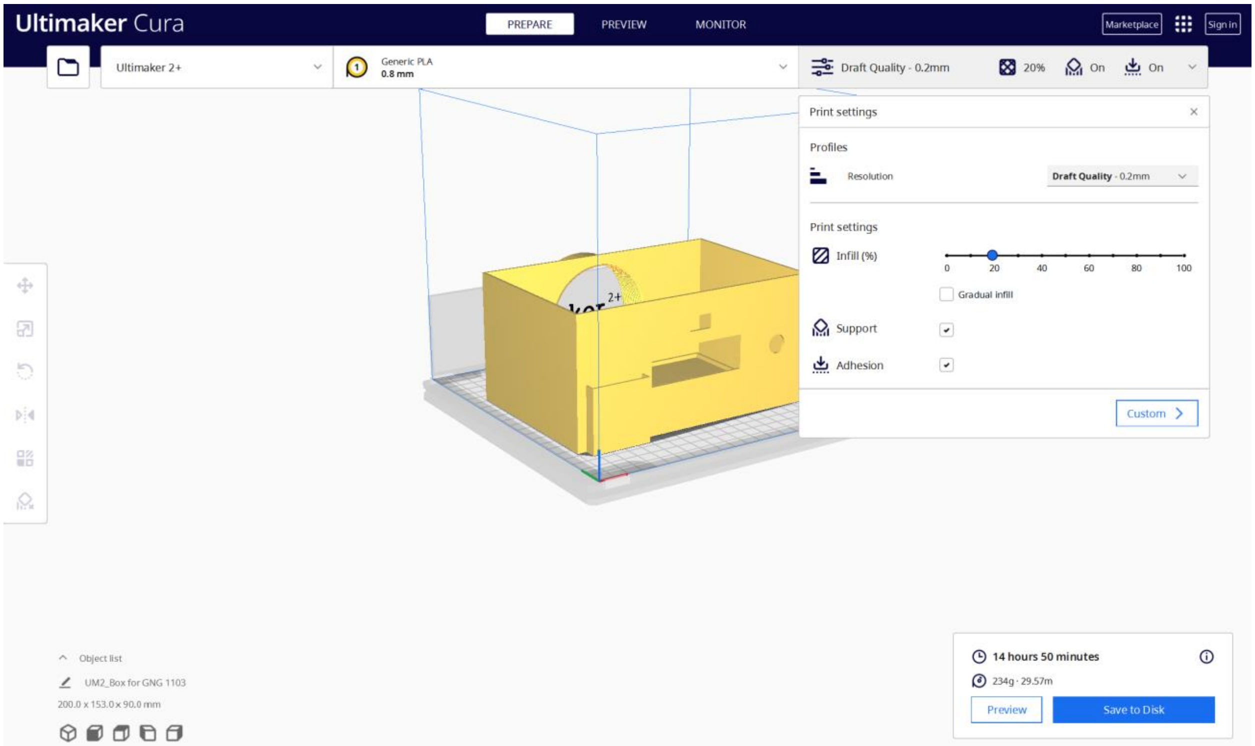

Figure 9: 3D Printing Slice Preview and Time Estimate

Wrike Link and Project Plan

Our project plan for the remaining weeks is to combine all sensors and modules that have all been tested into one big prototype to ensure the system works correctly and can be tested further. Our team will also start printing the casing for the final prototype.

https://www.wrike.com/workspace.htm?acc=4975842&wr=20#folder/966341779/list?filters=status%3Dactive&sidePanelItemId=985887791&sortOrder=1&spaceId=-1&viewId=108931260