Deliverable K

User Manual

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: December 7th, 2022

University of Ottawa
Faculty of Engineering

Table of Contents

List of Figures

List of Tables

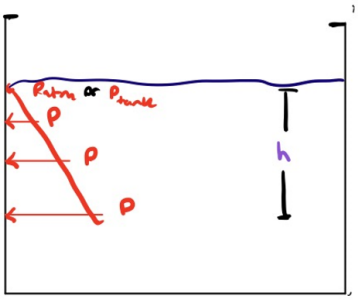
List of Acronyms and Glossary

Table 1. Acronyms

Acronym	Definition
BTP	Beyond the Pale
CAD	Computer Aided Design
IR	Infared
BOM	Bill of materials
LCD	Liquid Crystal Display

1 Introduction

Beyond the Pale needs a food-safe device that can accurately measure brewing metrics and log specific gravity vs time in a cost-effective, automated system. In this report, our team will go through the steps we took to solving this design problem. Our solution team decided to use the concepts of fluid statics as seen in figure 1. The concept of fluid statics explains that the pressure in a fluid at any point in the fluid is proportional to its density. By determine the pressure at any 2 points with a known height difference between these 2 points, density can easily be isolated and calculated. The purpose of this document is to familiarize any new user with this device's use and to allow any other engineers to expand on our current work to further the products accuracy or optimize our design without starting from nothing. The report will take you through every process of the design phase, it will go into detail on how the product works for any user trying to learn to use this device and it will explain in detail how the device was built from wiring diagrams to CAD models, to C++ and python code explanations to aid anyone trying to improve this device. There are no security or privacy considerations associated with the use of this user manual.



$$S.G = \Delta P / g h \rho_{H2O,4°C}$$

S.G - Specific gravity of wort

$\rho_{H2O,4°C}$ - Calculated density of water at 4°C

ΔP - Difference in pressure of the two sensors

g - The gravitational gravity constant 9.81 m s^{-2}

h - The difference in the height

Figure 1: Concept of Fluid Statics

2 Overview

Beyond the Pale (BTP) is a brewing company located in Ottawa Ontario, Canada, that came to us to create an accurate, automated density measuring device for them to use in their fermentation and production of their beer. Breweries often use density (specific gravity or degrees plato) to measure both the sugar and alcohol content of their beer to get the precise taste. BTP currently uses refractometers and hydrometers to measure to do this process, which is time consuming for lab techs and wastes potential product due to requiring a sample from the beer. Other methods of measuring density such as a Tilt bar was not able to be used due to the device being free floating and often getting damaged or lost when alcohol was drained for the fermentation tank.

After meeting with the co-owner and head brewer at BTP, Shawn Clark, they several needs that were identified. The product needed to be safe use and made from food grade material as the device. Density needed to be accurate as the change of density over time tells BTP lab technicians when the beer is done fermenting. It was also noted that recording, logging, and displaying data in easy and efficient way was a great asset, as it would allow BTP to look back previous brews and able to replicate it more efficiently than before. There were also many different physical constraints and finical we had to consider while building our device. It needed to fit in a 3-inch hole, must be able to run for 2 to 3 weeks at a time and need to not be free floating in the tanks. They were also financial constraints as well, with our device needing to be within $20 000 for all 16 fermentation tanks.

Our device is a stationary using the pressure sensors to measure density using the principle of hydrostatic pressure, unlike many sensors on the market. We focused on the installability of our prototype in to BTP fermentation tanks, and how easy it is to use for BTP technicians. We also focused on accuracy and readability of the density measurement. We have both have a simple display screen to show the density and temperature measurement that can be switch between using a button near the screen. Data can be displayed on a computer through a graphical manner showing both the temperature and SG over time.

Figure 2: Final Prototype

The pressure/temperature sensor is housed in an aluminium unit, that is four sperate parts that attached together either through threading, O-rings, or a combination of both. The second part is the Electronic and display housing unit, which contains the Arduino and display screen, which is made using a 3D printer. This is attached to the pressure sensor via a screw or/and a well fitted tube. The main screen can switch between temperature and SG measurements through a button on the left of the screen. The whole prototype would ideally be located at the 3-inch hole of the fermentation tank with the pressure sensors housing is made for liquid environments and the electronic/display housing outside the tank.

2.1 Cautions & Warnings

Caution: Make sure housing parts for the pressure sensors are properly sealed and put together. If there is a leak, it would short out the wires and electronics of the Arduino board. Make sure also to be able to seal the holes you made as water may leak through and cause hazard to other electronics or tripping hazards for people walking by.

3 Getting started

Top construct the pressure sensor housing unit, the following list of material is required to assemble the device. Some of the materials need to made or obtained. Drawings of hardware are appendix I.

Table 1: List of Material

List of Materials	Quantity
T- Joint	1
Pressure Caps	2
Vertical Connectors (Threads on both ends)	2
Horizontal Connector (O-ring)	1
O-rings	6
Sealing tape	1
Electrical Housing	1
Arduino	1
Bread Board	1
LCD Screen	1
I2C Module	1
Pressure Transducer	2
Latching Buttons	1
IR Sensor	1
Clock Module	1
Remote	2
4.7 KΩ Resistor	1
10 KΩ Resistor	1
10 KΩ Potentiometer	1
Wiring	50

1. Place sealing tape on the thread of the pressure transducer. Thread a pressure transducer into one of the pressure caps. Tighten by hand with a socket and wrench. Repeat this step for the other pressure transducer and cap.

Figure 3: Step 1

2. Slide the wires of the pressure transducer through the vertical connector. Place a O-rings on each end of the vertical connector. After thread, the cap on to the vertical connector. Tight just by hand. Repeat this step with the other pressure transducer and vertical connector.

Figure 4 Step 2: One left untightens to show O-ring

3. Place the wires of one pressure transducer through one of the threaded holes on the T-Joint. The wires should exit the T-joint through the hole with no threads. Simply thread the vertical connector on the T-joint. Tighten by hand. Repeat this step with the other vertical connector.

Figure 5 Step 3:

4. Wire all the electrical components on the bread board and Arduino and place inside the electrical housing. See wiring diagram below for guidance. Do not connect the pressure sensors yet.

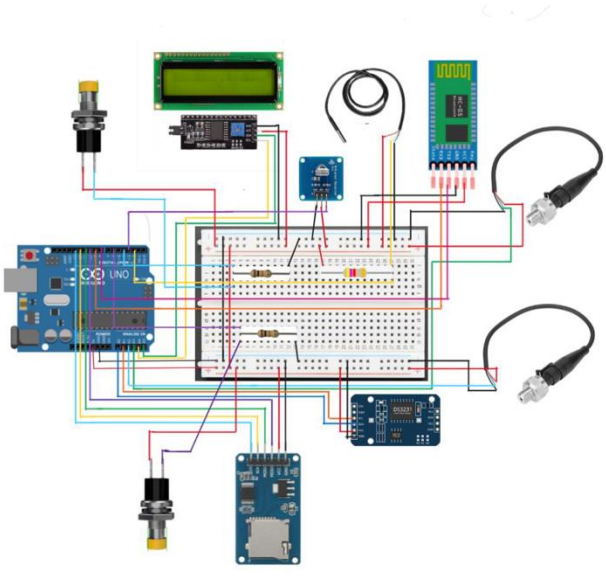


Figure 6 Step 4:

5. Place 2 O-rings on the horizontal connector. Place the horizontal connect end with the O-rings on the T-Joint.

Figure 7 Step 5:

6. Place the device in the desired bucket/tank. Then connect the electrical housing to the horizontal connector that is outside the bucket/tank. Then wire the pressure sensors into the breadboard and Arduino.

7. Upload code to the Arduino from a laptop using the USB connection. Once the code is uploaded hit start on the remote and the device will begin to start reading specific gravity.

3.1 User Access Considerations

This device is engineered to measure the density of liquids in specific gravity. The main group of users will be the breweries. It is important for breweries to know the specific gravity when producing beer as this helps make decisions on the fermentation process. The use of the device would be useful in any situation that requires to know the density of a liquid.

3.2 Accessing/setting-up the System

Once the device is wired and installed in the tank plug in the battery. The user will simply just upload the Arduino code to the Arduino board. Then they can hit start when indicated on the LCD screen. There is no user ID or log in system required to run the device

3.3 System Organization & Navigation

When the device is running the LCD will display the readings in plato and specific gravity. To switch to the temperature readings. Simply hit the button on the front of the electrical and the LCD will switch to showing the temperature.

3.4 Exiting the System

To turn the system off simple hit stop on the remote. Once the device isn't displaying any of the results and the graphing is saved. Just unplug from the computer and disassemble the device by following the assemble steps in reverse.

4 Using the System

The following sub-sections will provide a detailed explanation of our device's interaction with the user. The user can interact with our device through the display of our device or through the logging/Graphing of the data being measured.

4.1 Specific Gravity and Temperature Display

When the system is turned on, you will first see the message "Hit Start to begin" as seen in figure 8. Once the red power button is hit on the remote control, the system will begin to measure specific gravity and temperature. On the LCD display, you will now see the specific gravity in kg/m^3 and in Plato as seen in figure 9. To view the temperature, you hit the white button to switch the display as seen in figure 10. If the button is already pressed, you may see the temperature instantly instead of specific gravity. You can again use the button to switch back and forth. To reset/pause the system to the initial state, we hit the Vol + button on the remote as seen in figure 11. To restart the system, you would again hit the power button on the remote. While the system is running, data will be sent to the serial monitor as seen in figure 11 in the format of Date, Plato, and Temperature. The data being sent to the will be useful for the logging and graphing function.

If your LCD screen is experiencing issues where random characters are being shown as shown in figure 12, you may need to reset the system by either hitting the reset button located on the Arduino or re-uploading the code the system.

Note: the following figure were taken when no sensors were connected to the system so numbers being displayed will not make any sense.

Figure 8: System after First Being Turned On

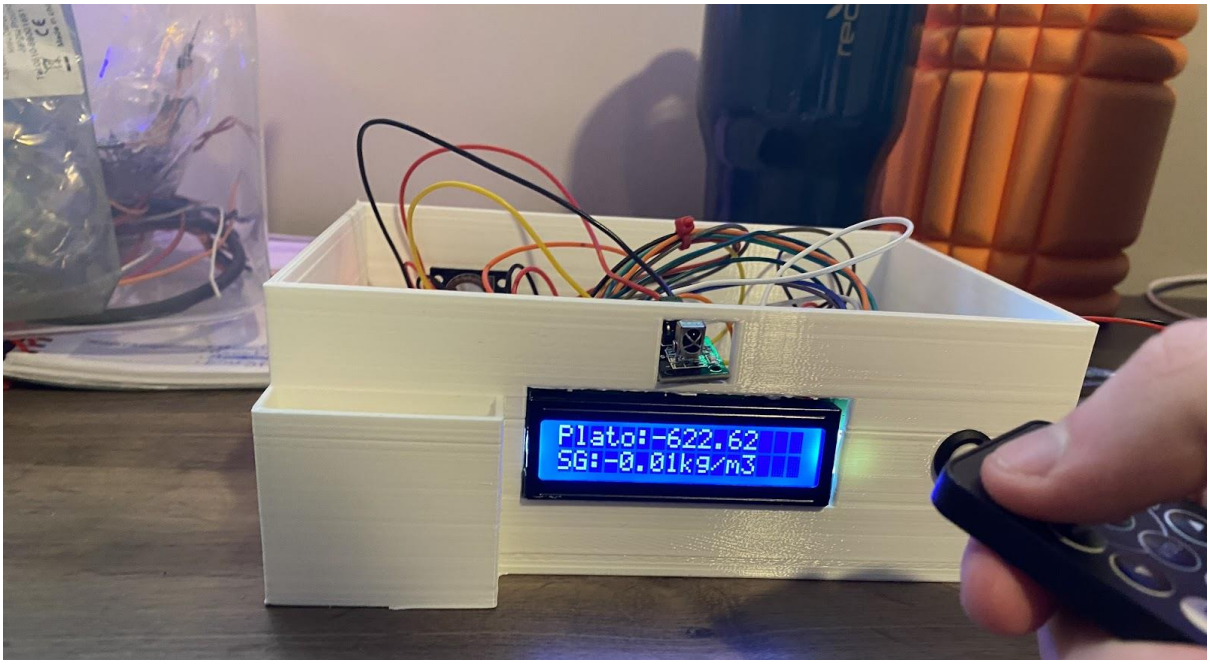


Figure 9: Display after System is started

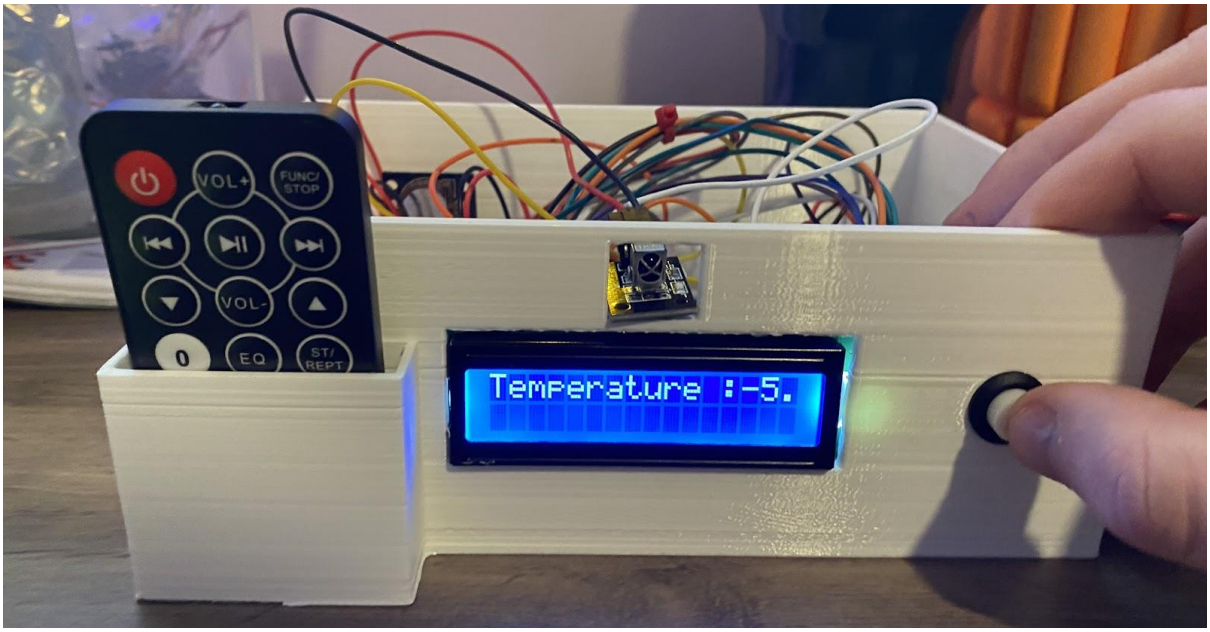


Figure 10: Display after Button was Pressed

Figure 11: Button after Vol+ was hit

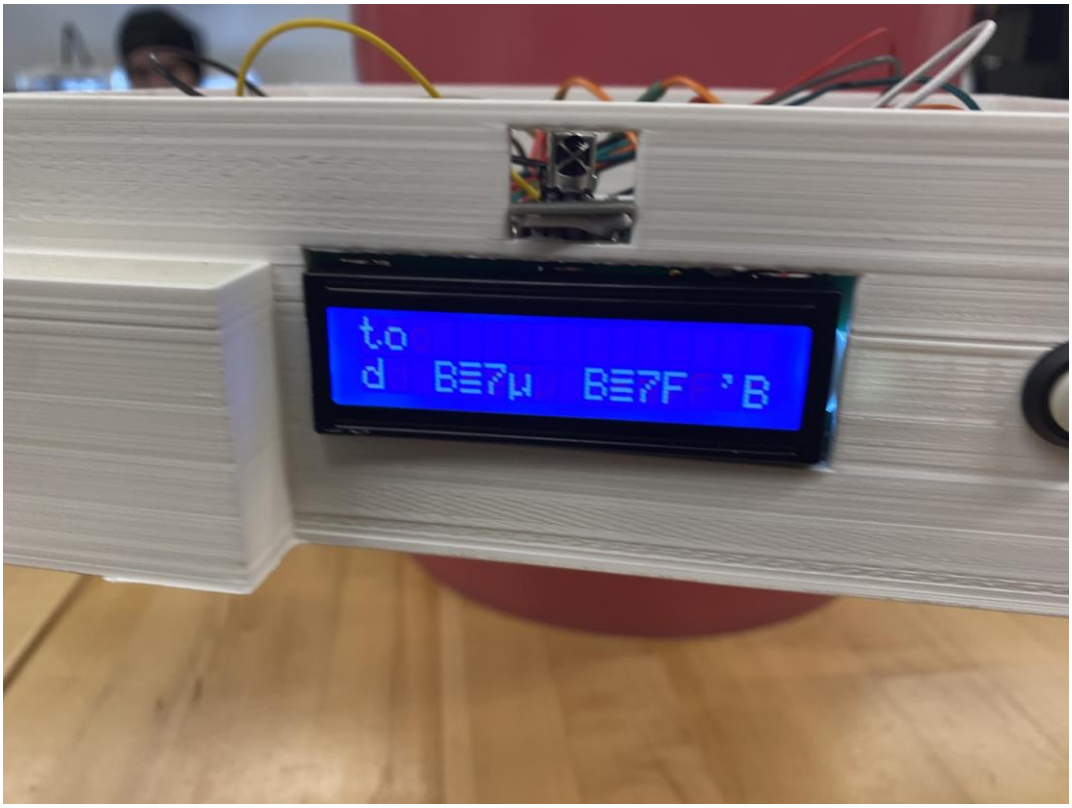


Figure 12: LCD Glitch

4.2 Logging and Graphing of Specific Gravity and Temperature

To log and graph the temperature and specific gravity of the brew, in addition to the time and date at which the data points were taken, you must plug in your laptop into the side of the device.

Make sure to install Python by downloading the appropriate file at https://www.python.org/downloads/ and then installing the required libraries by running the command "pip3 install –r requirements.txt" in the terminal while under the same folder in the Appendix. The requirements.txt file contained the necessary libraries used in the Python script.

Once connected to the Arduino, you should be able to read serial data on the serial monitor in the format of [Date Time, SG in Plato, Temperature in Celsius;] as seen in figure 13. Once you have established that your laptop is receiving the serial data from the Arduino, you can run our Python script found in Appendix 1: Design Files – Table 3: Referenced Documents – data_plot.py. This code takes the serial data and saves it into a CSV file as seen in figure 14. This file will be saved to your laptop which can then be referenced by the user in the future. In addition to saving the data to a CSV file, the python script will also graph specific gravity and temperature vs time which will help users control the take-off of fermentation curves in real time.

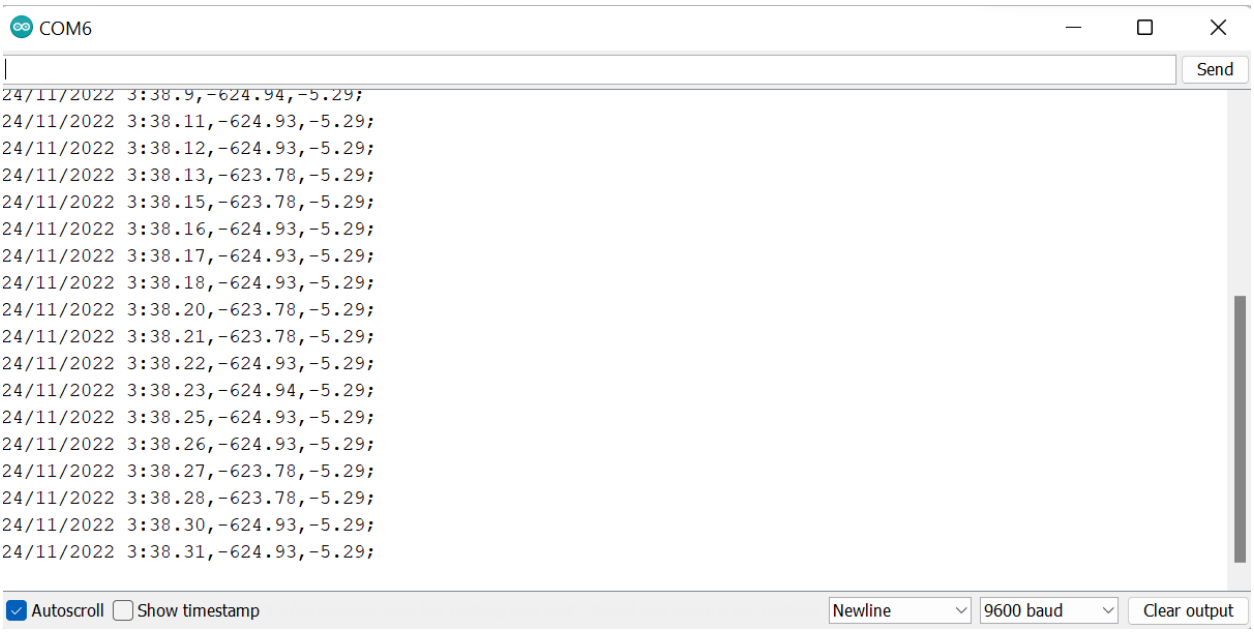


Figure 13: Serial Monitor Format

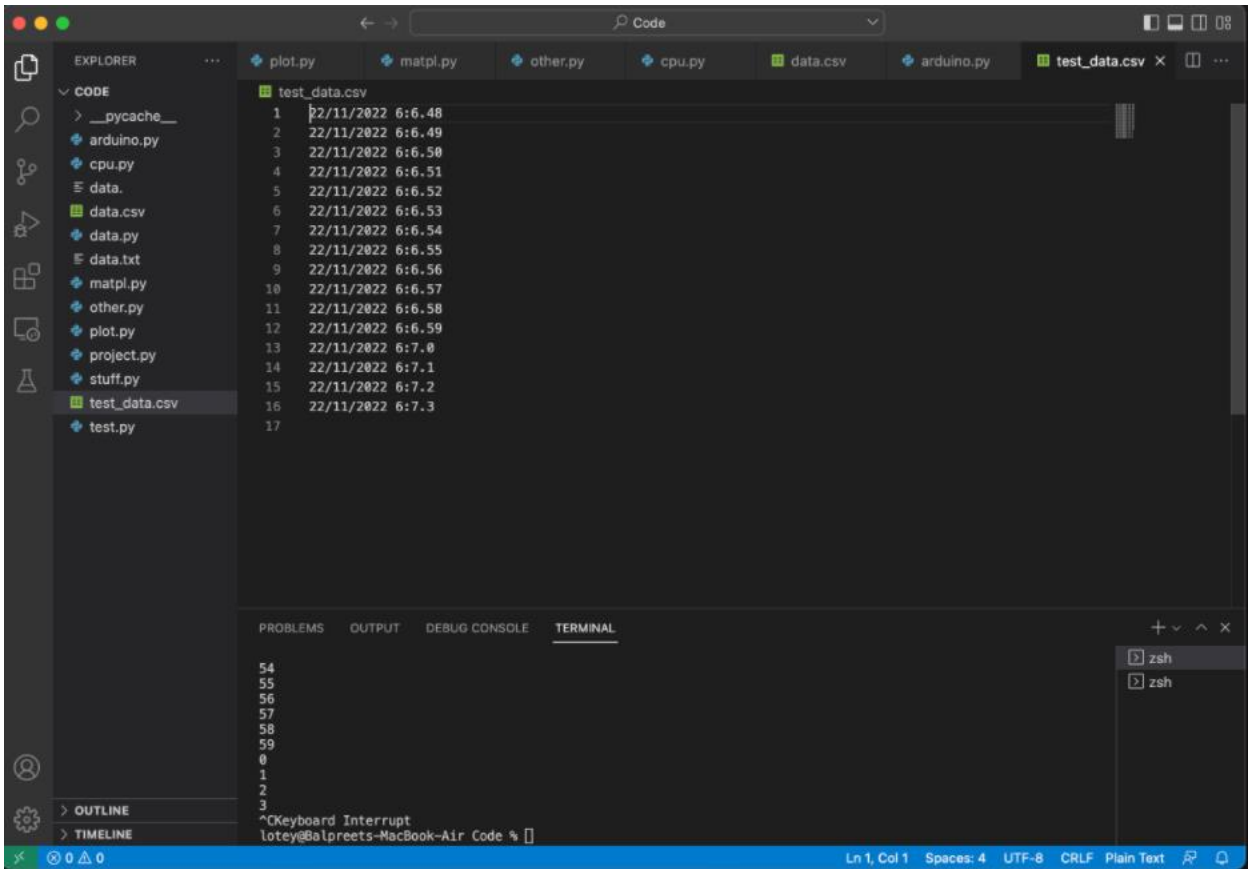


Figure 14 Data being Saved to CSV File

5 Troubleshooting & Support

5.1 Error Messages or Behaviors

5.1.1 Common issues with Arduino

Compilation error: '[DallasTemperature]' does not name a type; did you mean 'temperature'?

The [library] has not been installed. Ensure that the correct library by the correct publisher of the library is installed from the Library Manager in Sketch > Include Library > Manage Libraries… > Install (under the correct library) and recompile the sketch.

Failed uploading: no upload port provided

The correct COM port has not been selected. It needs to be picked from the drop-down menus of available ports as seen below:

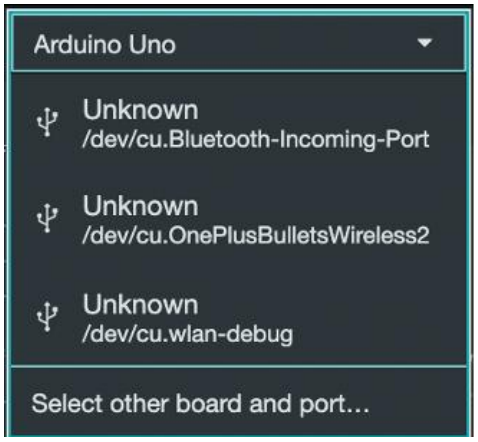


Figure 15: Available Arduino COM ports

5. 1. 2 Common issues with Python

[Errno 2] could not open port [/dev/cu.usbmodem1101]:

An incorrect USB port or a mistyped entry for the USB port has been entered. Ensure that the correct USB port is entered in line 14 by checking with the Arduino IDE what COM port is currently being used. See “Failed Uploading...” under 5.1.1

ImportError: No module named [csv] / NameError: name 'serial' is not defined:

The library has not been installed. Type the following command to install the library correctly:

```
pip3 install [library]
```

5.2 Special Considerations

- Ensure that the device is installed perpendicular to the ground and that it is not at an angle from normal to the ground. This can be done by checking the stainless-steel tubes and twisting them into position before fully tightening them.
- The brewery tank must be fully cleaned before the device can be installed to prevent leaving unwanted contaminants in the process of sealing the device, that would be difficult to remove later.
- The product requires a computer that can be hard-wired to the product 24/7. The data would also stop logging once the device goes to sleep. Ensure that such a computer is available and is always connected to a power supply.
- Primary components of the product are powered by a 9V battery which is prone to discharging over time. Therefore, the casing supports the external containing of the battery to allow for the removal and exchange of the battery with a newer one regularly. Rechargeable batteries may also be used.
- When installing any software or files, a stable internet connection would be required.

5.3 Maintenance

To ensure our device is always getting accurate reading, this device must be uninstalled after every batch and the pressure sensors must be removed from the assembly. Once removed from the assembly, each pressure sensor must be placed in a cleaning solution to remove any built-up yeast that may have gotten caught in the sensor. All stainless-steel tube should also be scrubbed down to remove any risk of cross contamination by using food-safe solution. Once the pressure sensors have been thoroughly soaked and are deemed to be clean, you may re-install the pressure sensors into the device by reconnecting the quick connect cable and re-installing the device into the tank.

5.4 Support

5.4.1 Arduino Help

To troubleshoot issues with Arduino sketches, visit https://docs.arduino.cc/learn/starting-guide/troubleshooting-sketches.

The Arduino Help Center (https://support.arduino.cc/hc/en-us) can also be visited for answers to FAQs on common errors.

Arduino also has Forum Support (https://forum.arduino.cc/c/using-arduino/installation-troubleshooting/18) available to ask questions to the Arduino community rrelated to the above that can aid troubleshooting.

5.4.2 Python Help

It would help to understand some of the basics of the libraries used in the Python code to understand how the data is being transferred. The documentation also helps with troubleshooting common issues that the user might encountering when trying to run the Python code. Below are the official documentations on each of the implemented libraries:

pySerial: https://pyserial.readthedocs.io/en/latest/pyserial.html#overview
matplotlib: https://matplotlibguide.readthedocs.io/en/latest/
csv: https://docs.python.org/3/library/csv.html

5.4.3 Contact Support

For further information and assistance with the product, gain more information about the product and gain insight on its creation by visiting our site: https://makerepo.com/nobri040/1292.gng1103b2specific-gravity. You can also contact us directly by visiting our individual pages on the same page:

Figure 16: Further Contact information

6 Product Documentation

6.1 Pressure/Temperature System

6.1.1 BOM (Bill of Materials)

Table 1: Updated Bill of Materials for Presure/Temperature Housing

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
O-Rings	6	EA	0	0	Owned
Electrical Components Required					
Pressure Transducer Sensor 30 Psi	2	EA	13.49	26.98	See Deliverable F
DS18B20 Temperature Sensor	1	EA	14	14	See Deliverable F
Wiring	6	EA	0	0	Owned
Software Required					
On Shape	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

6.1.2 Equipment list

Equipment Name	Quantity
Wire Cutters	1
Machine Shop	1
Socket wrench set	1
Soldering Iron	1

6.1.3 Instructions

The T-Joint, pressure caps, vertical connectors and the horizontal connector will be sent to a machine shop to be manufactured. Note in the prototype we used aluminium, but stainless steel would work, though it is much more expensive. If one cannot find a machine shop, PBC piping could work as well. After all parts are accessible follow the steps in getting started.

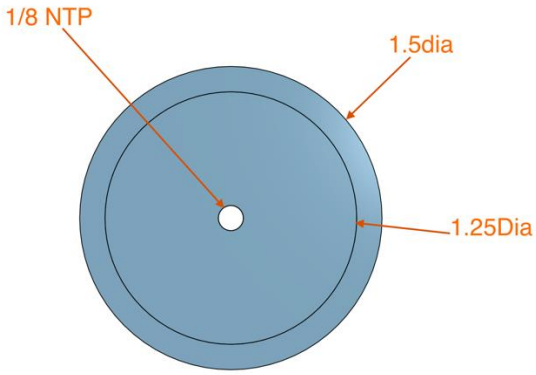


Figure 17: Pressure Cap front

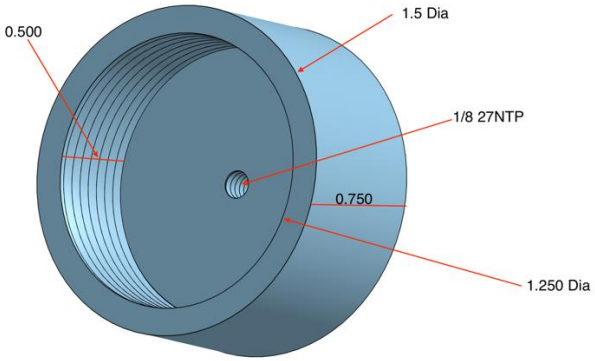


Figure 18: Pressure Cap Side

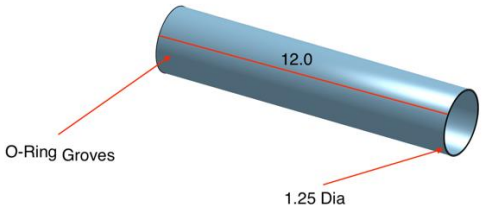


Figure 19: Horizonal Connector

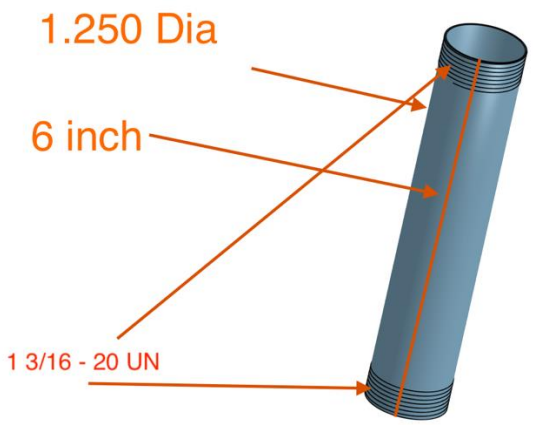


Figure 20: Vertical Connector

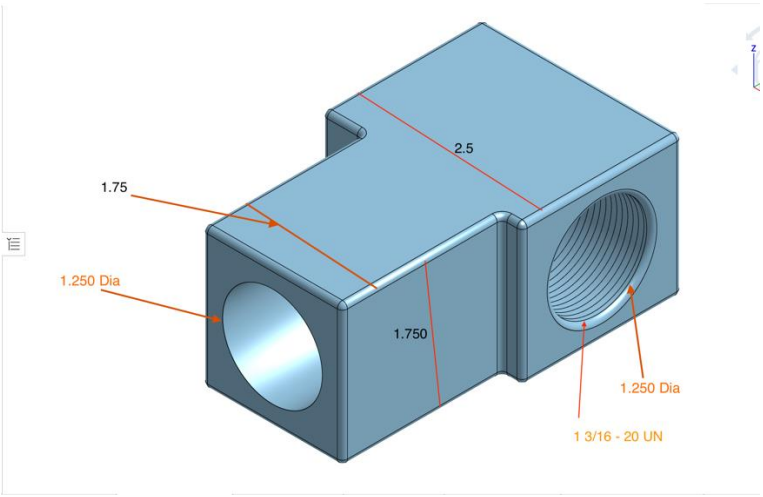


Figure 21: T-Joint

6.2 Display/Logging System

6.2.1 BOM (Bill of Materials)

Table 1: Updated Bill of Materials

Parts	Quantity	Unit of Measure	Unit Cost ($)	Estimate Cost($)	Link
Hardware Required					
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)
Electrical Components Required					
Arduino	1	EA	0	0	Owned
Breadboard	1	EA	0	0	Owned
LCD Screen	1	EA	0	0	Owned
Latching Buttons	1	EA	0.54	0.54	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

6.2.2 Equipment list

Equipment Name	Quantity
Computer	1
3D printer	1
Pliers	1

6.2.3 Instructions

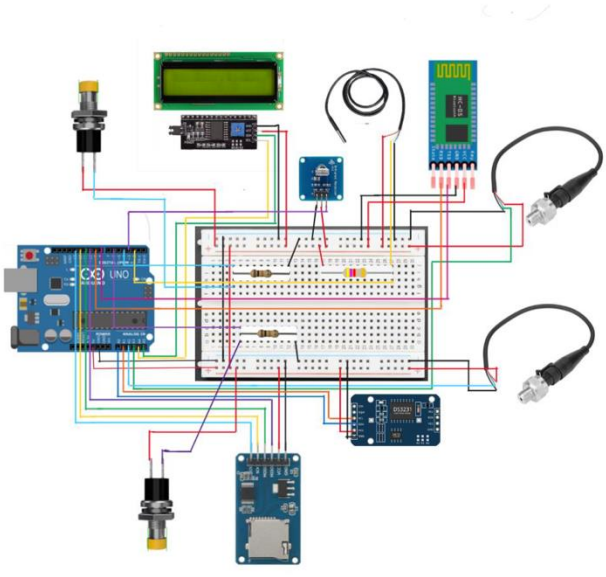
To manufacture the electrical housing

1. Upload the file of the electrical housing found below to an SD card
2. Upload the file in the 3D printer.
3. Once the electrical housing has been printed use the pliers to pull out the supports.

Next, obtain the following materials and wire them in the electrical housing unit as shown in the figure below:

Table 2: List of Material

List of Materials	Quantity
Electrical Housing	1
Arduino	1
Bread Board	1
LCD Screen	1
I2C Module	1
Pressure Transducer	2
Latching Buttons	1
IR Sensor	1
Clock Module	1
4.7 KΩ Resistor	1
10 KΩ Resistor	1
10 KΩ Potentiometer	1
Wiring	50



6.3 Testing & Validation

6.4.1 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 23. 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 22.

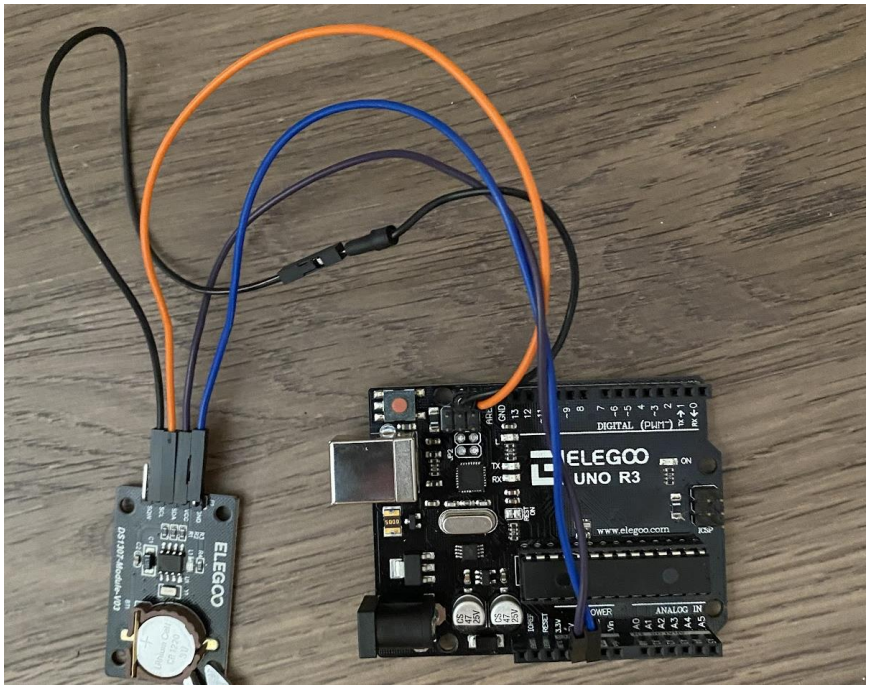


Figure 22: Clock Module Prototype

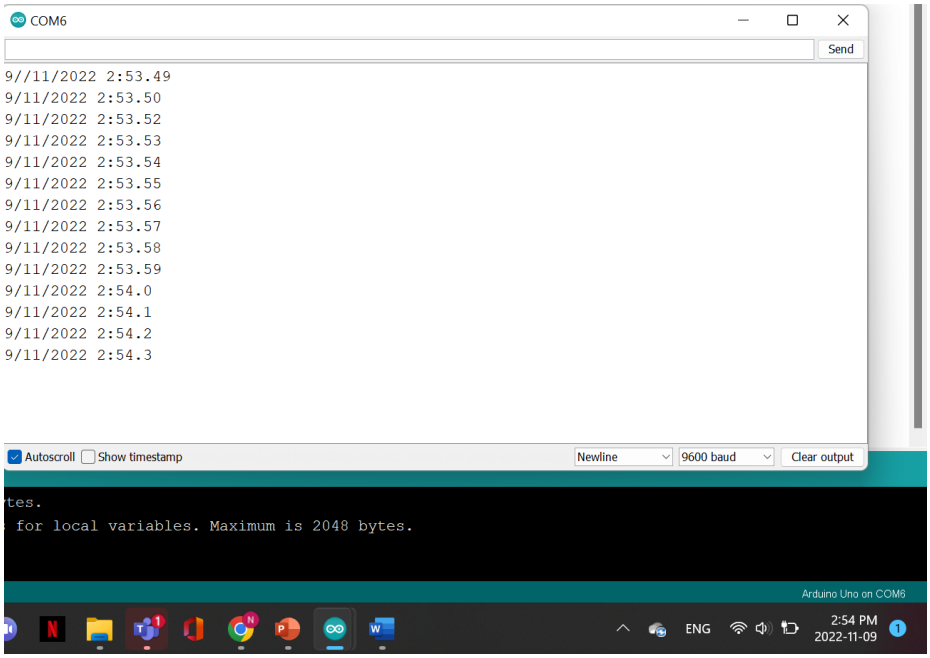


Figure 23: Date/Time Output from the Real-Time Clock Module vs Date/Time on Laptop

6.4.2 IR Sensor Testing

To test the IR sensor, I connected the receiver to the Arduino as seen in figure 24. This prototype consists of an IR receiver, a remote, an Arduino, jumper cables and a breadboard. The goal of this prototype was to ensure the IR receiver is working and accepts the correct commands. To begin, once the code was set up, I opened the serial monitor and printed all raw data being received by the receiver, I individually hit every button and recorded the reading that was being shown on the monitor. Once I had these values, I edited my code to begin doing commands for different buttons. Since our project only requires this as a start and stop button, I set i=1 when the power button was hit and i=0 when the vol+ button was hit. This i integer will be used to control whether the Arduino will enter the loop to begin measuring data or continue awaiting a command from the remote. When I uploaded my code to my Arduino when I hit the power button, the message “Data is being measured” popped up on my serial monitor continuously and when the vol+ was hit, the message stopped rolling in which is exactly what our system will do. Found below in table 3was the values of each button on our remote.

Table 3. Button values

Button	Raw Value of Each Button
Power	0xBA45FF00
Vol+	0xB946FF00
FUNC/STOP	0xB847FF00
Back	0xBB44FF00
Play/Pause	0xBF40FF00

Forward	0xBC43FF00
Down	0xF807FF00
Vol-	0xEA15FF00
Up	0xF609FF00
0	0xE916FF00
EQ	0xE619FF00
ST/REPT	0xF20DFF00
1	0xF30CFF00
2	0xE718FF00
3	0xA15EFF00
4	0xF708FF00
5	0xE31CFF00
6	0xA55AFF00
7	0xAD52FF00
8	0xAD52FF00
9	0xB54AFF00

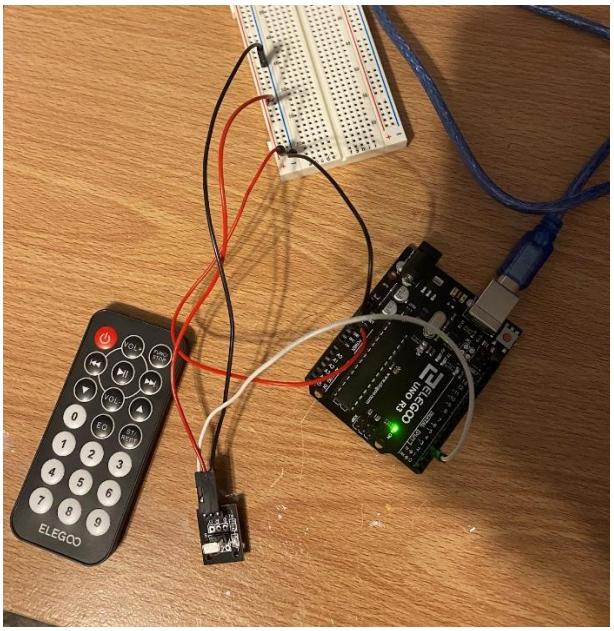


Figure 24: IR receiver and Remote Testing Prototype

6.4.3 LCD Screen + Button

To test the LCD Screen and the button, I wired the LCD screen and the button to a breadboard which was connected to an Arduino. The main goal of this test was to ensure that when the button was pressed, the screen would switch to display other quantities and it would clear the screen of the other quantity. In addition to the LCD screen and the button, I also needed a 10KΩ resistor for the button and a 10KΩ potentiometer for the LCD screen. The potentiometer is needed while we wait for the I2C module to be approved for purchase. The prototype can be seen in figure 25. When the button isn't pressed, you can see case 1 in figure 26 and when the button is pressed, you can see case 2 in figure 27.

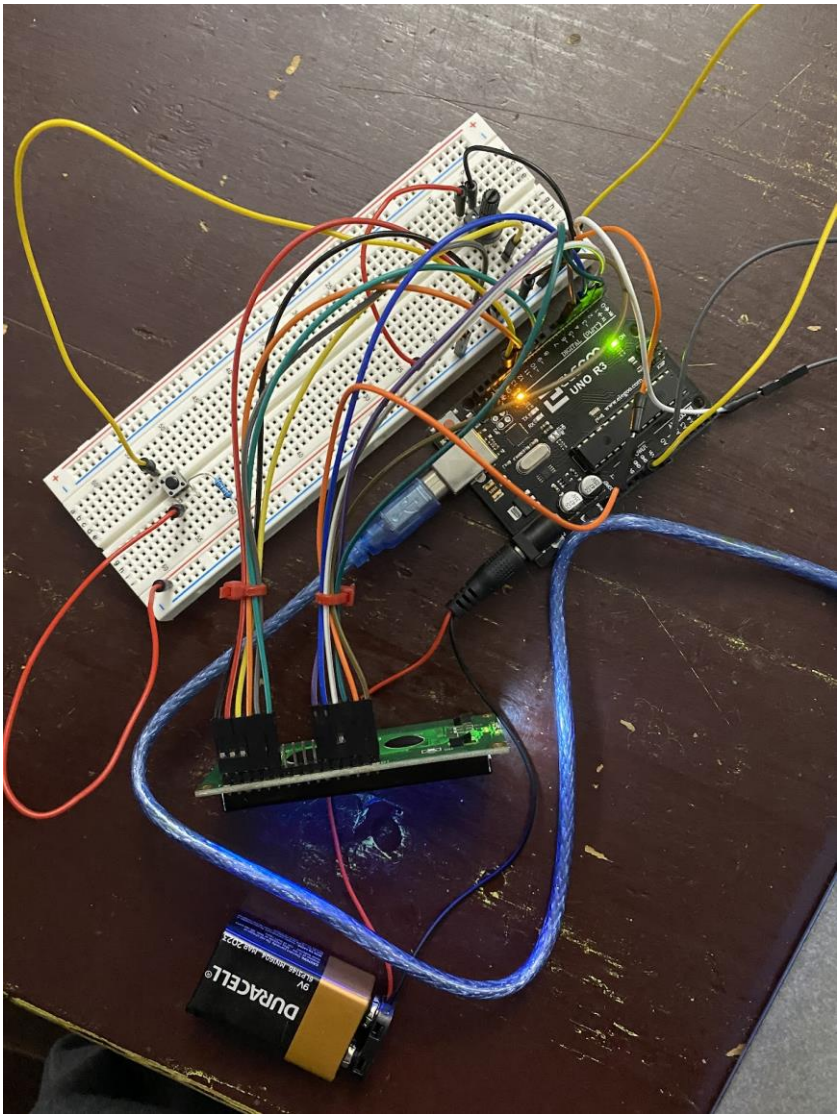


Figure 25: LCD Screen Prototype

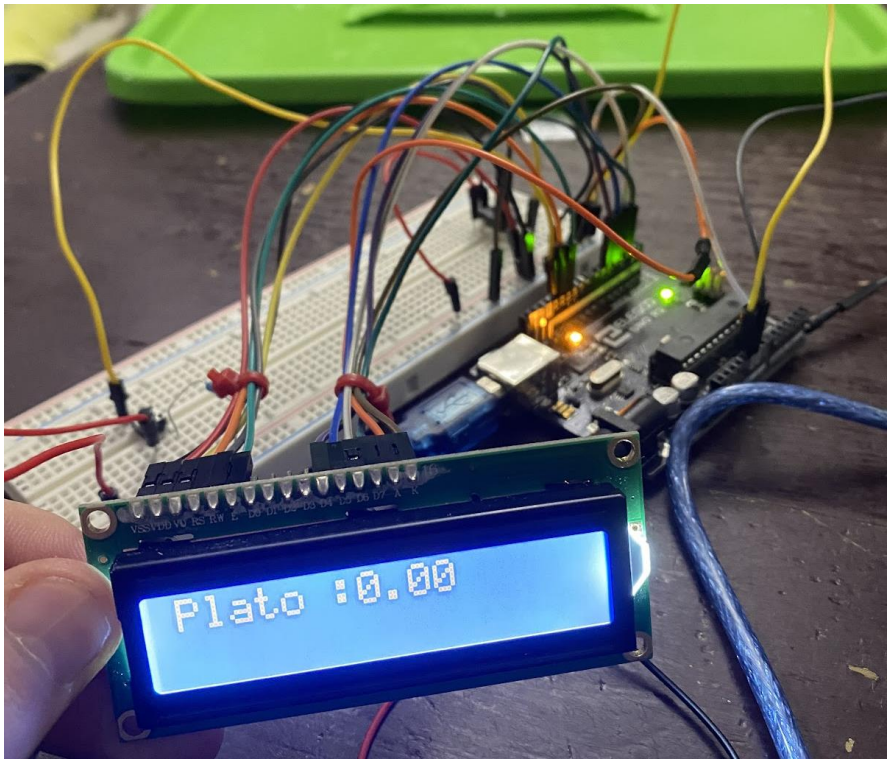


Figure 26: Case 1 - The button was not pressed.

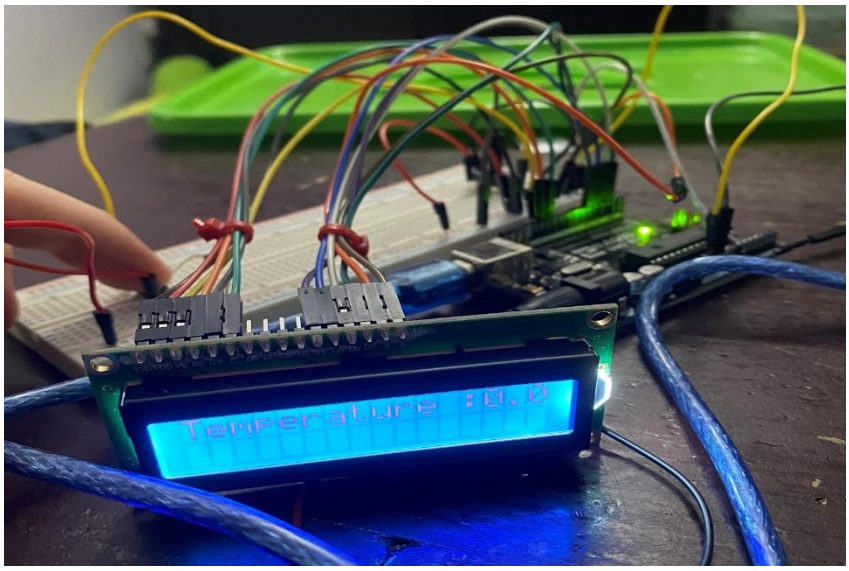


Figure 27 Case 2 - The button was pressed

6.4.4 Temperature Sensor Prototype

To prototype the temperature sensor, our temperature sensor was wired up to the Arduino using a 5kΩ resistor, wires and a breadboard as seen in figure 28. The goal of this prototype was to see if our temperature sensor was working. To test if our temperature sensor is working, we put a pot of water on the stove and filled a glass full of water and ice. After letting the water heat up on the stove and letting the ice melt to mix with the room-temperature water. Using the serial monitor, we were able to display the readings of the temperature sensor as seen in Figures 29 and 30 and we were able to cross-reference these readings with a digital thermometer.

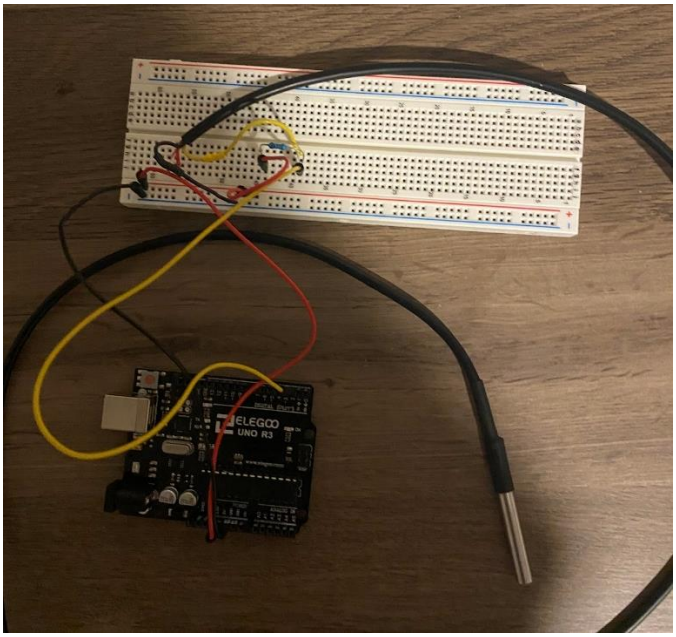


Figure 28: Temperature Sensor Prototype

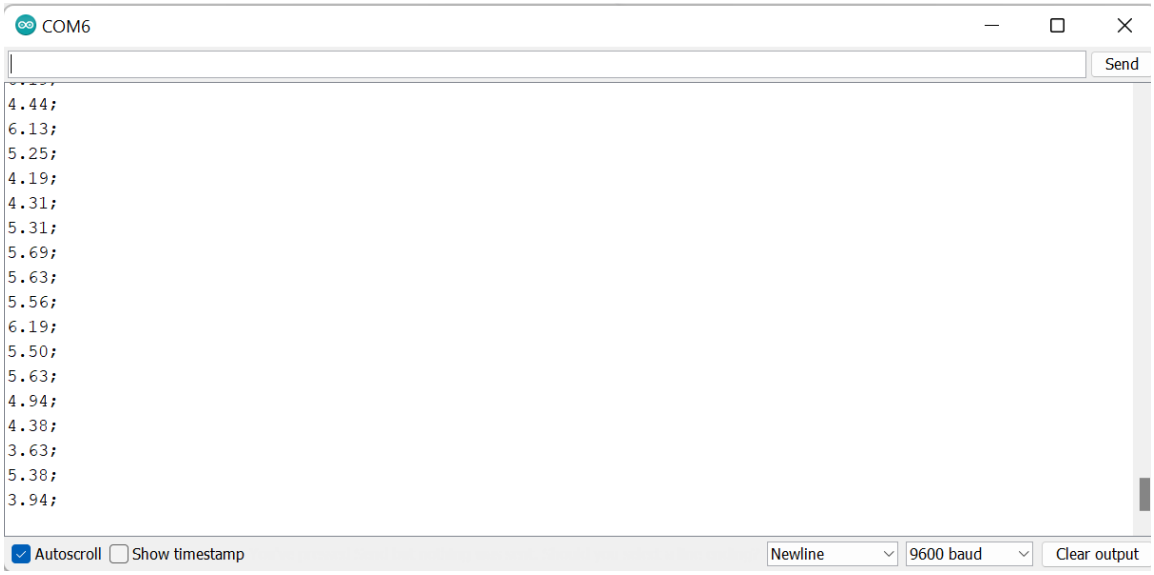


Figure 29: Temperature sensor in almost freezing water

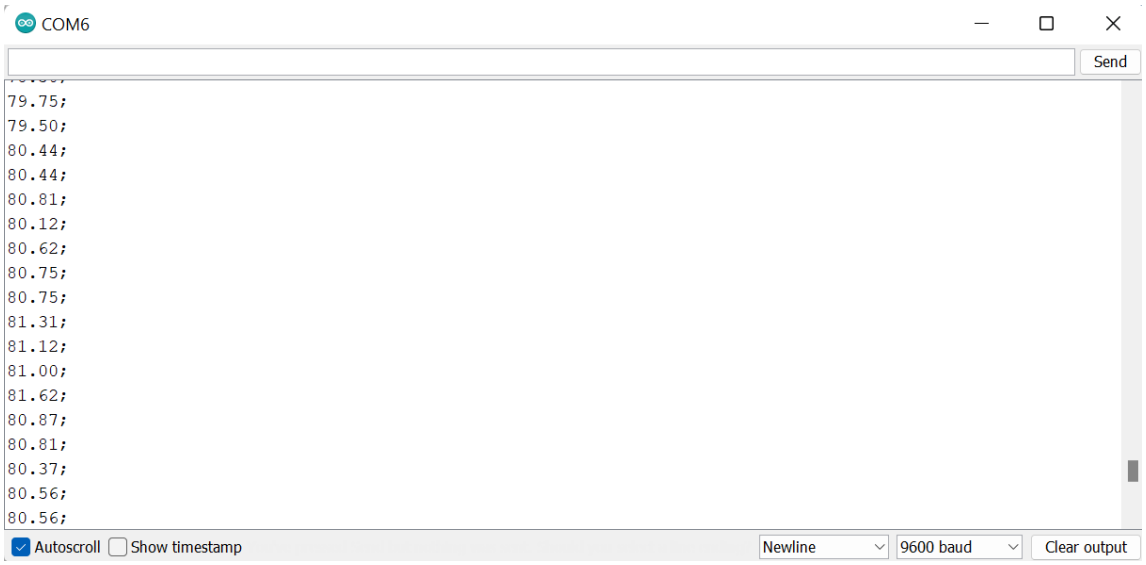


Figure 30: Temperature sensor in almost boiling water

6.4.5 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 32. 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 31. 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 ±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 exceptionally 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 31: Pressure Sensor Prototype

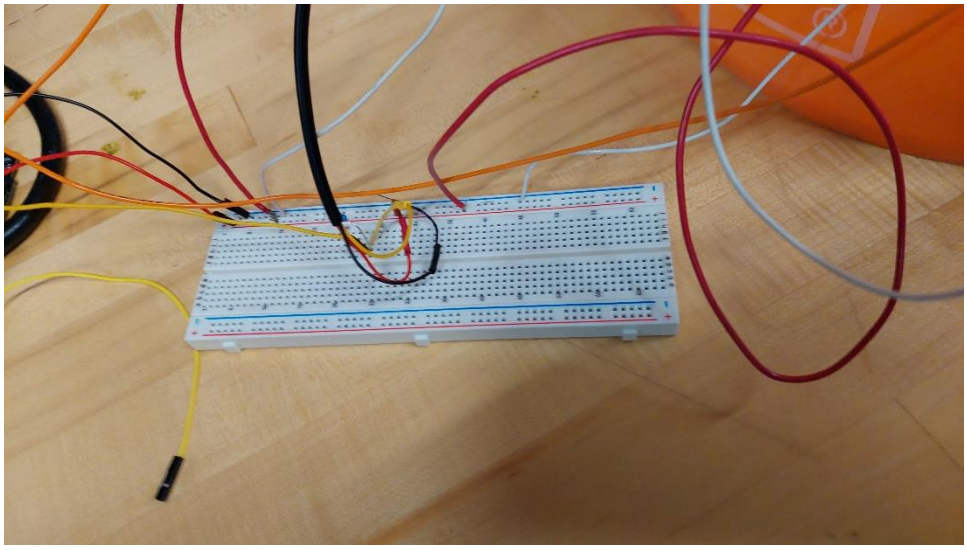


Figure 32: Prototype Wiring

7 Conclusions and Recommendations for Future Work

Throughout this term, or team learn many lessons working on this product. We learnt that long 3D prints are not a very viable option making us optimize our electronic housing as much as possible to reduce our print time to under 8 hours. Our team learnt the basics of Arduino, how to code a microcontroller to do what you want it to do and how to interconnect sensors with an Arduino microcontroller. In the end, we were able to get a fully working, high-fidelity prototype that had all measuring and display features fully working using the concept of fluid static by measuring the pressure difference between 2 reference points in the tank. Our recommendation to improve this product would be to get more accurate pressure sensors and to re-test with those. Our sensors were cheap sensors which led to some poor results at times since the pressure change is so small, our sensors at time would not pick it up and cause our

pressure reading to be an approximation. Our sensors were only able to measure changes to 0.1 Psi (689.476 Pa). Since our sensors can only detect the changes in multiples of 0.1 Psi (689.476 Pa) leaving room for error. If for example our pressure change is 1000Pa, it will only detect a 0.1Psi change and cause a very inaccurate reading since it would not detect a change until the pressure hit 1378.952 Pa. If we had a few more months, our team would add Bluetooth connectivity to our device. We had to abandon using a Bluetooth connection to transfer data due to a burn pin on our HC-05 module which did not allow for the transfer of data. With more time and money, we believe our team could have achieved this feature. Bluetooth connectivity was an important feature for us it is not feasible to have a different computer plugged into every single device if a brewery of 100 tanks was to purchase this device. A Bluetooth or Wi-Fi module would enable the use of one central computer to receive the data of all the tanks at once instead of relying on 100 different computers.

8 Bibliography

Insert your list of references here.

Component Links

Pressure Transducer Sensor 30 Psi: https://www.amazon.ca/Thread-Stainless-Pressure-Transducer-Compatible/dp/B08G4ZX4LY/ref=sr_1_11?keywords=pressure%2Btransducer&qid=1666567830&qu=eyJxc2MiOiIzLjk5IiwicXNhIjoiMy4yNiIsInFzcCI6IjIuNTAifQ%3D%3D&s=industrial&sprefix=Pressure%2BTrans%2Cindustrial%2C153&sr=1-11&th=1

Latching Buttons: https://edu-makerlab.odoo.com/shop/product/push-button-switch-81#attr=148

Bluetooth connection: https://edu-makerlab.odoo.com/shop/product/bluetooth-module-9?search=bluetooth#attr=255

DS18B20 Temperature Sensor: https://edu-makerlab.odoo.com/shop/product/grove-temperature-sensor-47?search=temperature+sensor#attr=

Clock Module: https://www.amazon.com/AT24C32-Replace-Arduino-Batteries-Included/dp/B07Q7NZTQS/ref=sr_1_3?keywords=Arduino+RTC&qid=1667612354&qu=eyJxc2MiOiIzLjY1IiwicXNhIjoiMy41NyIsInFzcCI6IjMuNDUifQ%3D%3D&sr=8-3

Library Links

Liquid Crystal: https://www.arduino.cc/reference/en/libraries/liquidcrystal/

OneWIre: https://www.arduino.cc/reference/en/libraries/onewire/

Spi: https://www.arduino.cc/reference/en/language/functions/communication/spi/

DS3231: https://www.arduino.cc/reference/en/libraries/ds3231/

DallasTemperature: https://www.arduino.cc/reference/en/libraries/dallastemperature/

Irremote: https://www.arduino.cc/reference/en/libraries/irremote/

Software Serial: https://docs.arduino.cc/learn/built-in-libraries/software-serial

APPENDICES

APPENDIX I: Design Files

Found below in table 3. Referenced Documents, you will find all relevant design files. The first document in the table labelled Box for GNG 1103.SLDPRT is the Solidworks file for our electrical housing. The second file named Box for GNG 1103_take2.STL is the sliced Solidworks file which we inserted into the 3D printer to print our casing. The file named Full_Code.ino is the code which was used to run our Arduino Uno. This code controls all the electronics in our system. The file named Wiring Diagram.pdf is an easy-to-follow guide on how we wired our Arduino. By following our wiring diagram, the Arduino code will not require any changes. The final design file is data_plot.py which is a python script which can take serial data and save it into a CSV file and graph the data as it is streamed into the script.

Table 2. Referenced Documents

Document Name	Document Location and/or URL	Issuance Date
Box for GNG 1103.SLDPRT	https://makerepo.com/nobri040/1292.gng1103b2specific-gravity	2022-11-29
Box for GNG 1103_take2.STL	https://makerepo.com/nobri040/1292.gng1103b2specific-gravity	2022-11-29
Full_Code.ino	https://makerepo.com/nobri040/1292.gng1103b2specific-gravity	2022-11-29
Wiring Diagram.pdf	https://makerepo.com/nobri040/1292.gng1103b2specific-gravity	2022-11-29
data_plot.py	https://makerepo.com/nobri040/1292.gng1103b2specific-gravity	2022-11-29
requirements.txt	https://makerepo.com/nobri040/1292.gng1103b2specific-gravity	2022-11-29