Smart Water Bottle Attachment Documentation:

Chandini Kalidindi

CS 3651

Goal/Motivation:

Proper hydration is important for many bodily functions and prevents dehydration which can lead to further complications. My smart bottle attachment can attach to any bottle and track hydration levels by keeping track of how often you drink water and how much water you drink. The user will be able to view this information on a dashboard to clearly see their trends. Many people tend to forget about drinking water as it gets pushed to the back of their mind. I hope this device will allow people to be more mindful of staying hydrated.

Inputs:

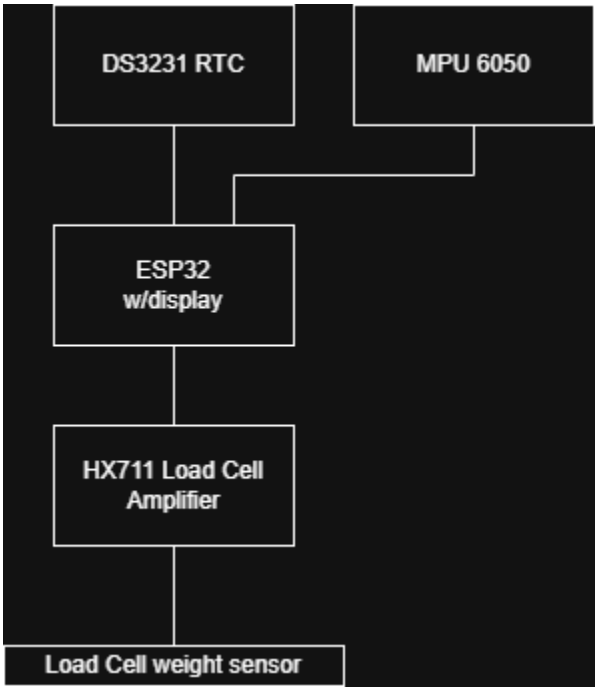
For my microcontroller, I decided to use the ESP32-S3 with a tft display. This would allow the user to not have to charge the bottle as often since it uses less power than other microcontrollers and is able to do wifi connectivity so it can connect to my server to upload all drink information.

Weight(water level):
For my project, I needed a way to read water levels reliably over time. To do this I used a load cell which measures the weight of the water bottle. By calibrating to the empty weight and the full weight of the water bottle, I can correlate these values to the total capacity of water for the bottle. This means I will be able to measure how much water is in the bottle based on the weight.

Accelerometer:
In order to determine when an actual drink happens, I use the mpu 6050 to measure the acceleration along the z axis to approximate the tilt. When most of the acceleration is along the z axis that means it is upright. When assembling my project, I made sure that the mpu 6050 was mounted upward.

```
bool isUpright(float az) {
  return (az > 5.5);
```

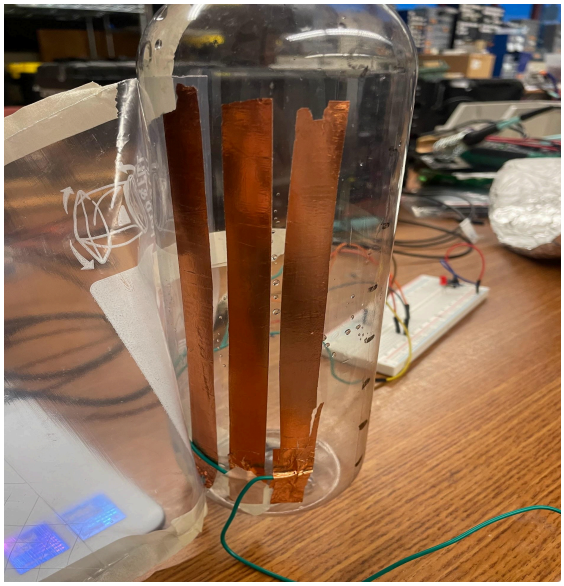
RTC: I used the rtc to log the time of the drink events to give the user a proper timeline of their hydration levels.



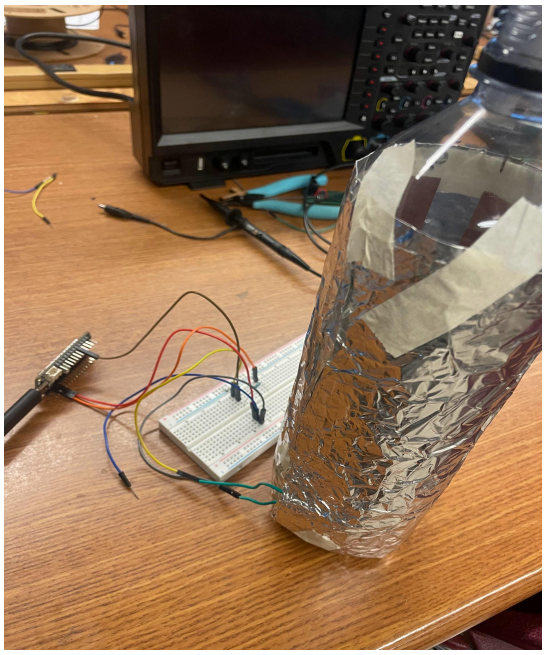
Iterative Process:

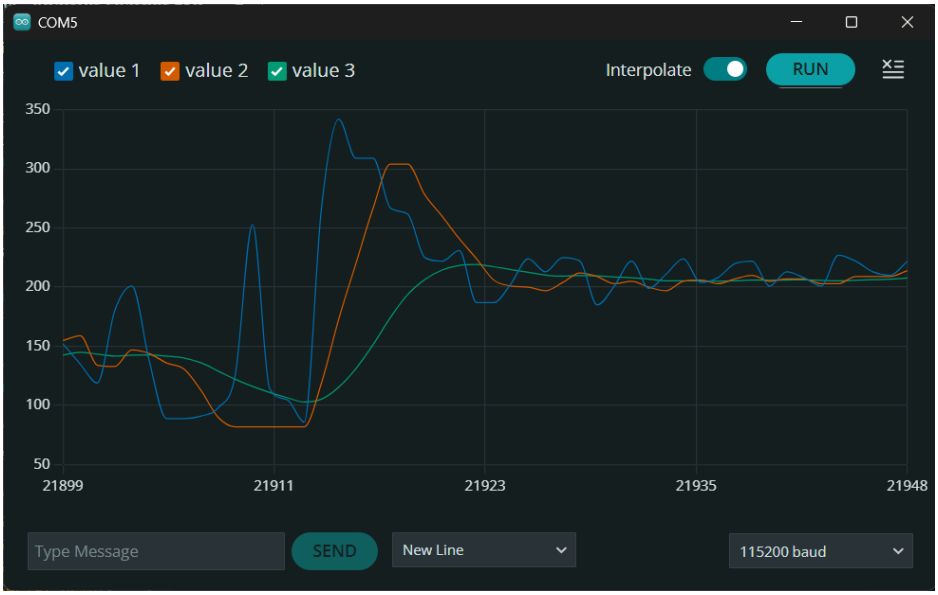
Measuring Water Level:

For my project I originally wanted to use capacitance to measure water level by putting copper strips on the outside of the bottle. I originally tried using touchRead() to read capacitance but the values were too noisy and it was hard to measure any accurate values. I was able to get a slight linear relationship when pouring in water, but as soon as I moved the bottle slightly or brought my hand closer to the strips, it would mess up the values making it unusable.

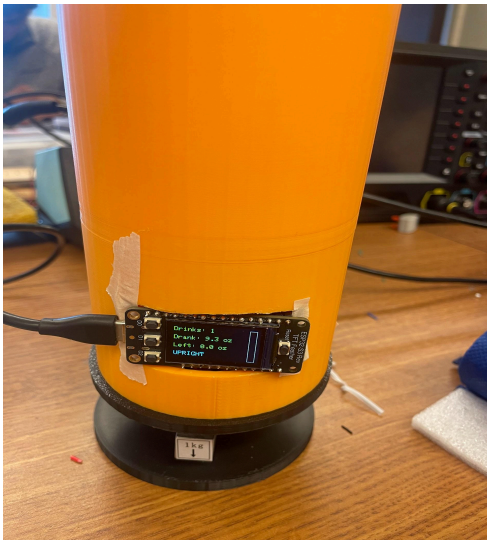


I also tried using a method similar to sd4 where I would take the amount of time it took the strips to charge based on the water level. The higher the water level, the longer the strips would charge before discharging. This method was a bit more promising after adding a grounded foil surrounding my bottle(not touching copper wires) and incorporating aggressive smoothing with the values. This method took a lot of experimenting with different resistor values to make the rc charging faster or slower. I also experimented with the size and placement of the copper strips. Overall after this process, I was able to detect changes in water level more reliably than my first method, but it still wasn't reliable enough for long term measurements and the values would completely drift to different values. I believe this can be due to the changing humidity in the bottle and other environmental factors.



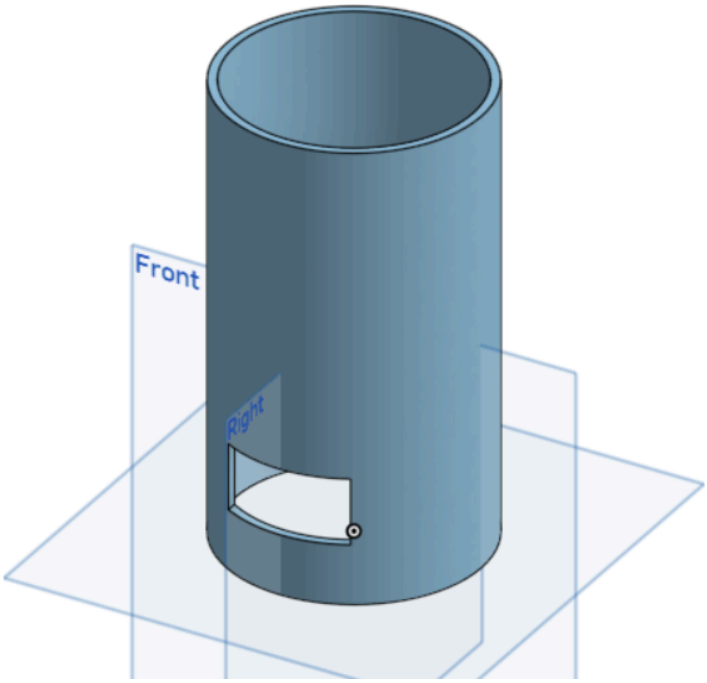


Overall, I concluded that using capacitance wouldn't be a reliable way to measure water level for my project so I pivoted to my backup plan of using a load cell to measure weight which would be a much more reliable metric long term.



Outputs: For my outputs I used the tft display on the esp32. This allows me to stay more compact as I would not need additional hardware to mount on the outside of the bottle. On the tft display, the user can see how much water they have drank, how much they have left, and how many drinks they have to drink throughout the day. I also displayed this information on a hydration dashboard that the user can access to more easily view long term hydration information.

3D Printed Bottle Stand:



Libraries:

Display:
Adafruit_GFX: I used this library for drawing text and shapes on my tft display
Adafruit_ST7789: TFT Display driver

Sensors:
Adafruit_MPU6050: For accelerometer and gyro data.
Adafruit_Sensor
HX711: Library to read data from the load cell amplifier adc.
RTClib: Library to interface with rtc using datetime.

Communication/data:
WiFi.h: Handles connecting the esp32 to wifi.
HTTPClient.h: sends http post requests to fastapi backend.
Fastapi
Pydantic
Sqlite3
Datetime

Esp32:
Wire.h : Handles i2c communication for the mpu6050 and rtc.
SPI.h: Handles communication for tft display

Resources:

DIY Capacitive Water Level Sensor using ESP32 Touch Read function

Non-contact Capacitive Liquid Level Sensing using FDC1004 - Hackster.io

GitHub - Infineon/mtb-example-psoc4-capsense-liquid-level-sensing

Capacitive liquid level sensing and shielding using arduino - Other Hardware / Sensors - Arduino Forum

Value I added:

- Most of the similar projects online used the touchread() method but it was done on a stationary tank of water instead of a mobile device like a water bottle. I also saw examples that utilized a designated capacitance to digital chips. I was not able to find these parts online and I wanted to measure capacitance manually like we did in sd4. I was able to experiment myself using my own conclusions and thought process while iterating through my project.