Project Overview

Design, develop, and iterate a reminder device for patients with neurological disorders

Conner Cherry
conerc22@proton.me
Mentor Michael Revel
mrevel@johnbapst.org

Description:

The proposed device will be a portable solution to a common disconnect in medical adjacent engineering. Existing solutions are clunky, unintuitive, and do not cater to patients in late stages of neurological disorders such as Alzheimer's. The device will incorporate visual and auditory cues to direct the patient to take their medicine. It will incorporate an audio recording which can be modified by the patient's caretaker or loved ones. Familiarity from the recorded message will bring comfort to patients experiencing neurological disorders with an anxiety component. The device will be open-source and easily accessible on Github, able to be recreated by anyone.

All prototypes will be made from 3D-Printed PLA plastic with inexpensive electronics and few additional materials to expand accessibility and affordability.

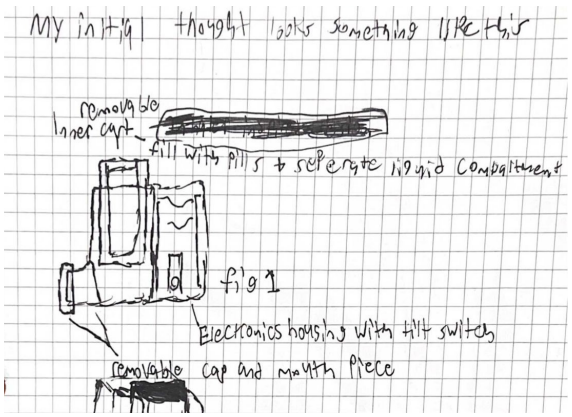
Background:

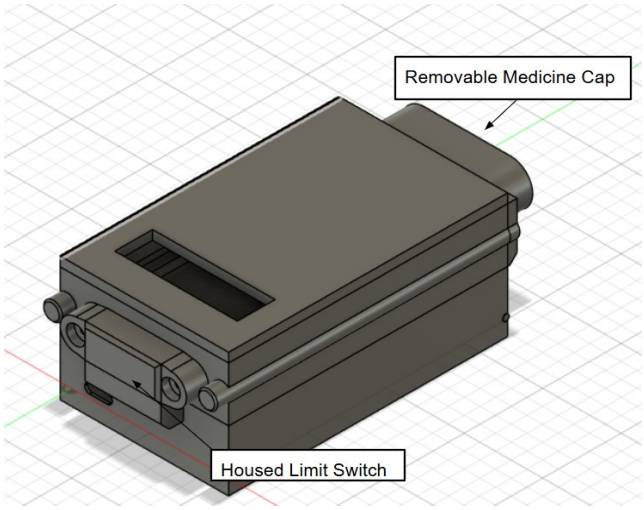
As the son of a nurse, I often hear about common issues in the world of healthcare. Engineering has not seen as large an impact on end of life care as it has in other sectors of healthcare, largely due to a lack of monetary force driving the development of medical devices. My grandmother has late stage Alzheimer's and is unable to care for herself. Through firsthand experience I've learned that one of the greatest challenges with end of life care is losing autonomy. Patients want independence. One way I believe we can give patients such as my grandma as much independence as possible, is through medical devices. As an amateur builder of anything from electronics to furniture, I wanted to try my hand at developing and iterating one of these medical devices and explore the engineering involved. In early 2025, I had the idea to make a portable, intuitive medical device that could be set up over wifi and played a prerecorded message to patients, providing them with auditory and visual cues to take their medicine. From then on I completed three design-prototype cycles to finally end with a finished device.

Initial Prototyping

V1

Initially meant to look like an inhaler, and include water in the device to help with swallowing
- Quickly pivoted to a boxier shape, with
one single large capsule





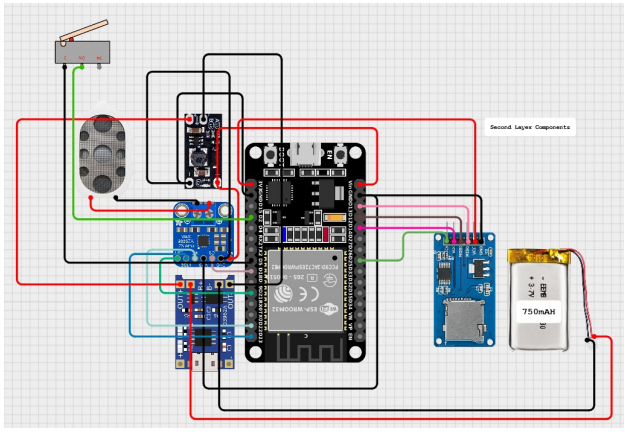
V1 Wiring

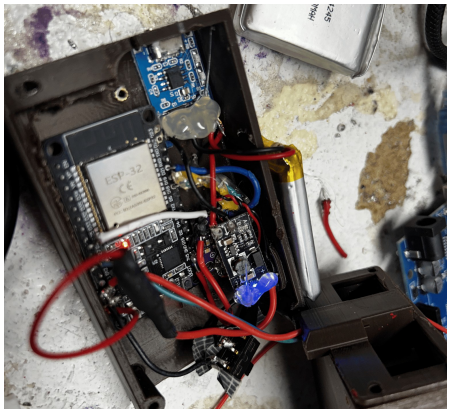
I had initially planned to use an SD Card module to store audio files. Instead, I pivoted to using source code uploaded by Xtronical (https://www.xtronical.com/) which uploads audio files as HEX code to be stored directly on the board.

-Decided to use ESP32 Wroom CE for eventual WiFi Capabilities and inexpensiveness

-Attempted to use millis() from my ESP32 Wroom CE board as an RTC, which caused time drift

-Modules held with heat set inserts.





Programming V1

-Programmed in Arduino IDE
Initial “HEX to WAV” code modified from XTronical- WAV files are loaded on the board as HEX. Alarm time is set in the programming IDE. Not ideal- more of a proof of concept.

```
void loop()
{
  switchval = digitalRead(limitpin);
  if (switchval==1){
    digitalWrite(mutepin,HIGH);
  }
  uint8_t Mono[4];                          // This holds the data we actually send to the I2S if mono sound
  const unsigned char *Data;                // Points to the data we are going to send
  size_t BytesWritten;                      // Returned by the I2S write routine, we are not interested in it

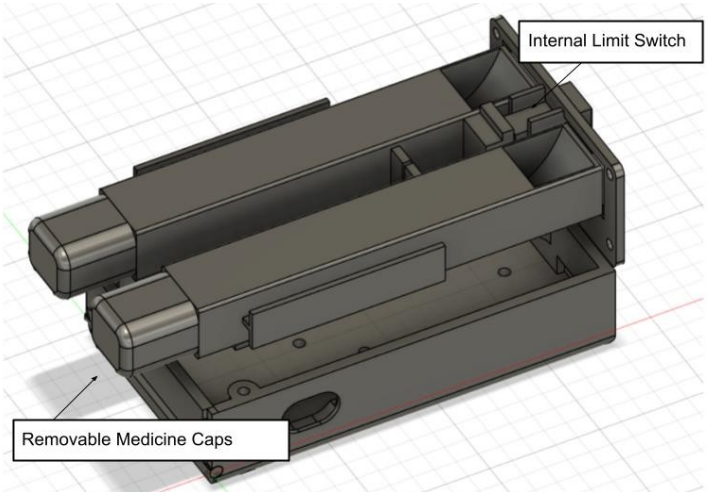
  // The WAV Data could be mono or stereo but always 16 bit, that's a data size of 2 byte or 4 bytes
  // Unfortunatly I2S only allows stereo, so to send mono we have to send the mono sample on both left and right
  // channels. It's a bit of a faf really!
  if(WavHeader.NumChannels==1)     // mono
  {
    Mono[0]=*(TheData+DataIdx);               // copy the sample to both left and right samples, this is left
    Mono[1]=*(TheData+DataIdx+1);
    Mono[2]=*(TheData+DataIdx);               // Same data to the right channel
    Mono[3]=*(TheData+DataIdx+1);
    Data=Mono;
  }
  else                            // stereo
    Data=TheData+DataIdx;

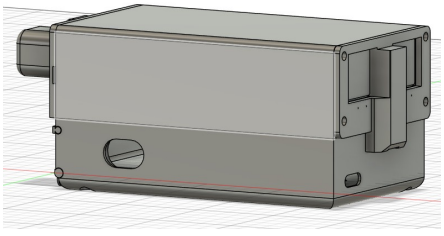
  i2s_write(i2s_num,Data,4,&BytesWritten,portMAX_DELAY);
  DataIdx+=WavHeader.BlockAlign;                        // increase the data index to next next sample
  if(DataIdx>=WavHeader.DataSize)           // If we gone past end of data reset back to beginning
    DataIdx=0;
}
```

V1 Problems
-Only able to store one dose of medication at a time
-No WiFi integration
-Programming the device requires the Arduino IDE and knowledge of the Arduino language
-Slider tension too high- hard to push down

V2

Changed from initial design:
-Now two compartments/ two sliders for pill storage
-Internal spring mechanism for sliders to pop up
-Front face and slider retainers now contain limit switch to monitor if medicine is taken





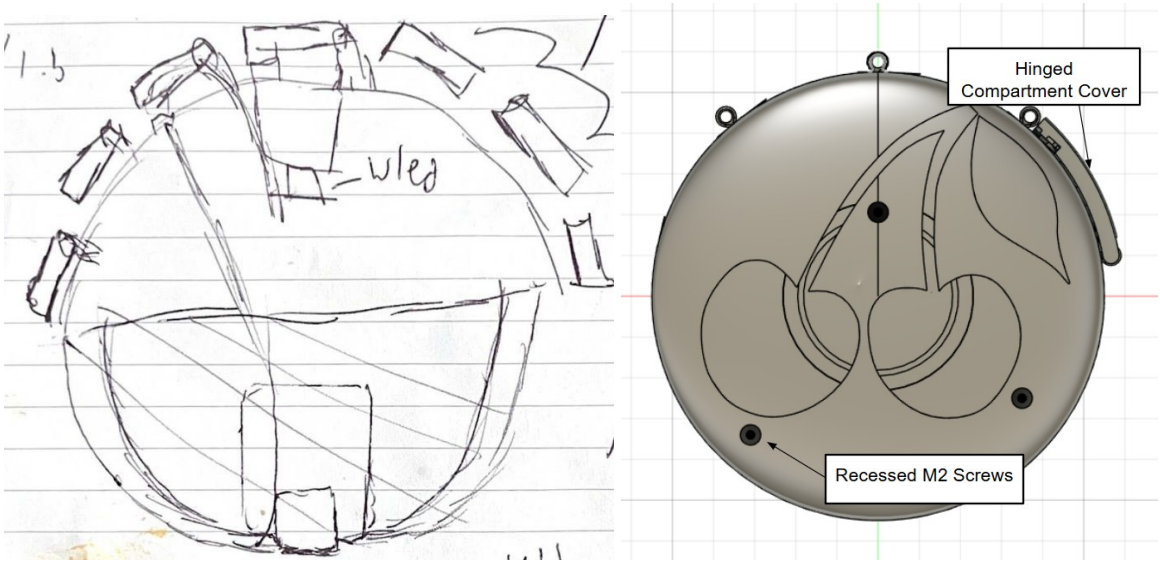
Programming V2

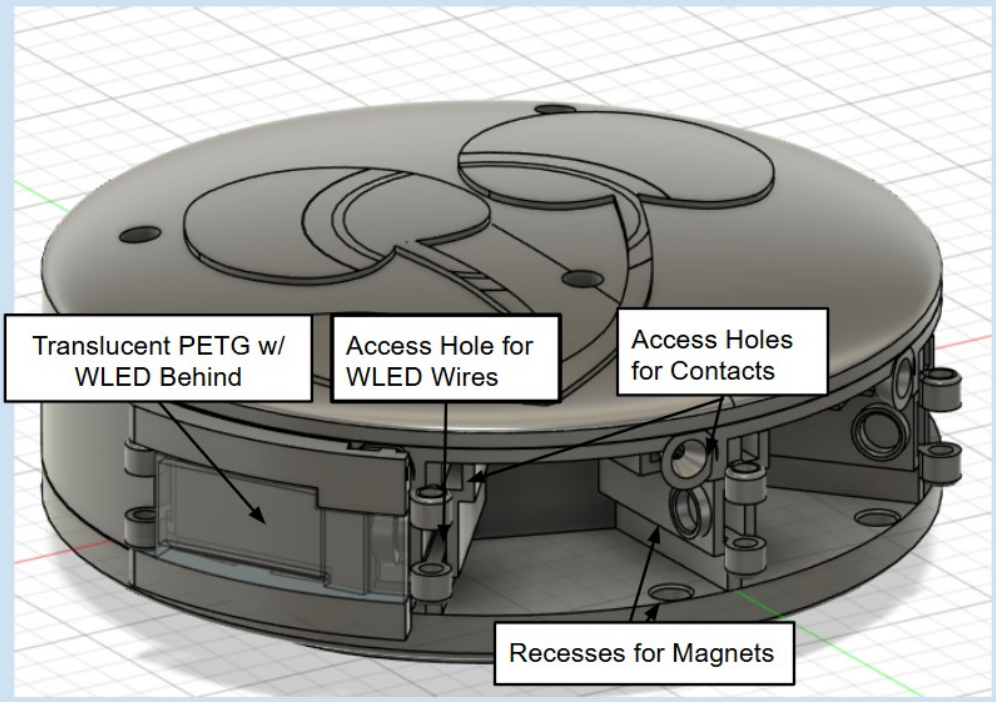
Programming stayed mostly the same- with the integration of new HEX voice commands, specifically text-to-speech commands telling the user what to do.
-Wiring stayed the same, with the addition of an RTC module to keep track of time.

Final problems
As part of the engineering design process, once I finished my first version of the device it came time to iterate again. I tested the first device out and found a couple issues

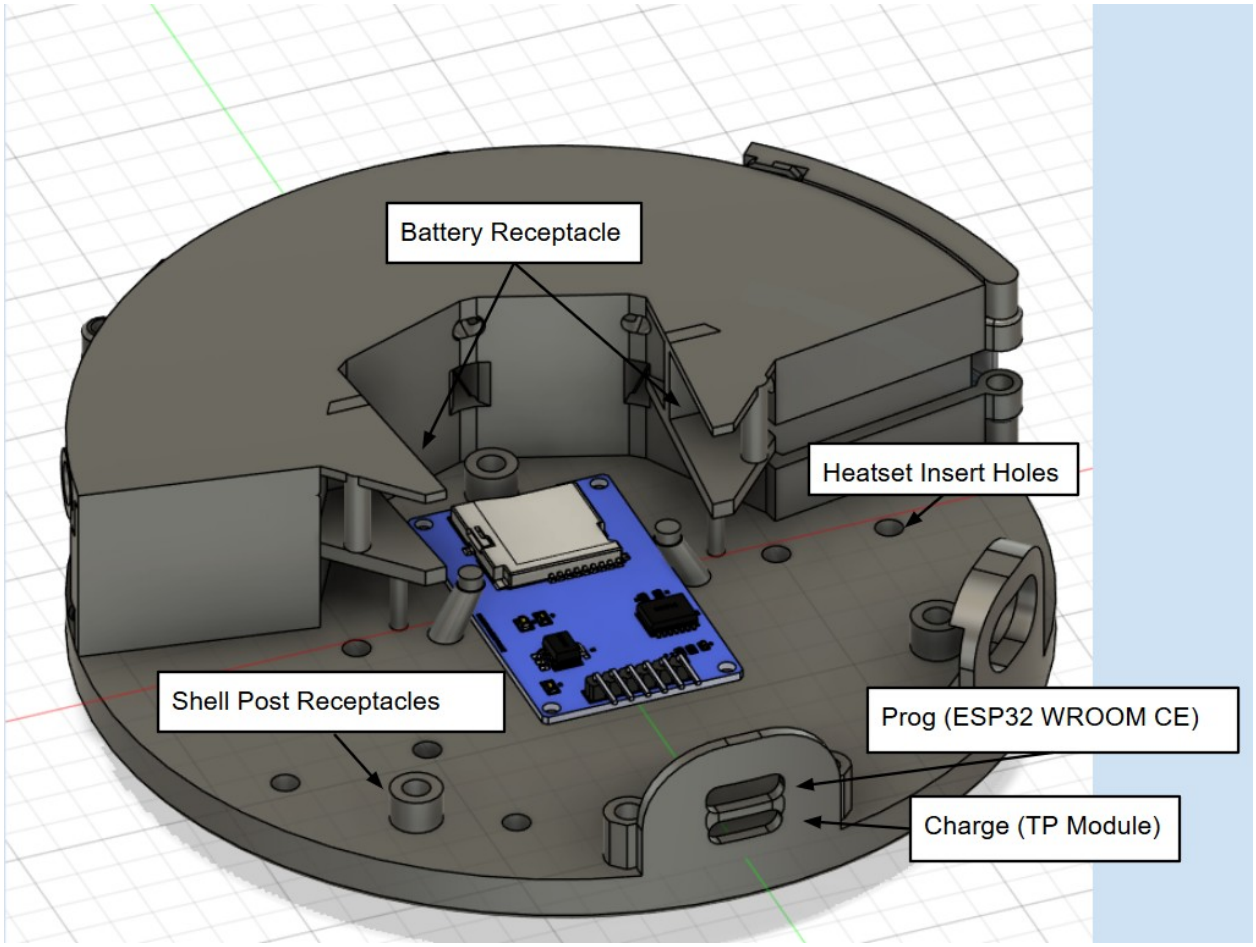
Problems:
-Capsules can fit few doses
-Only two different capsules can be inserted
-Medicine dispensing is complicated
-Internals are hard to reach

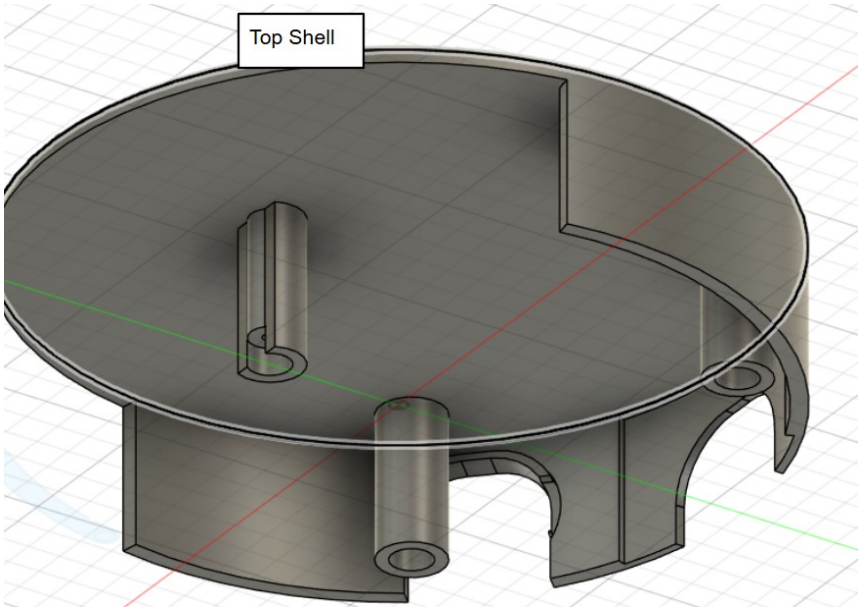
Final Iteration





I heat set the magnets into the PLA plastic, dropped and smoothed a ball of solder into contact receptacles with stripped wires waiting (carefully without melting the plastic)





Physical Changes Made-

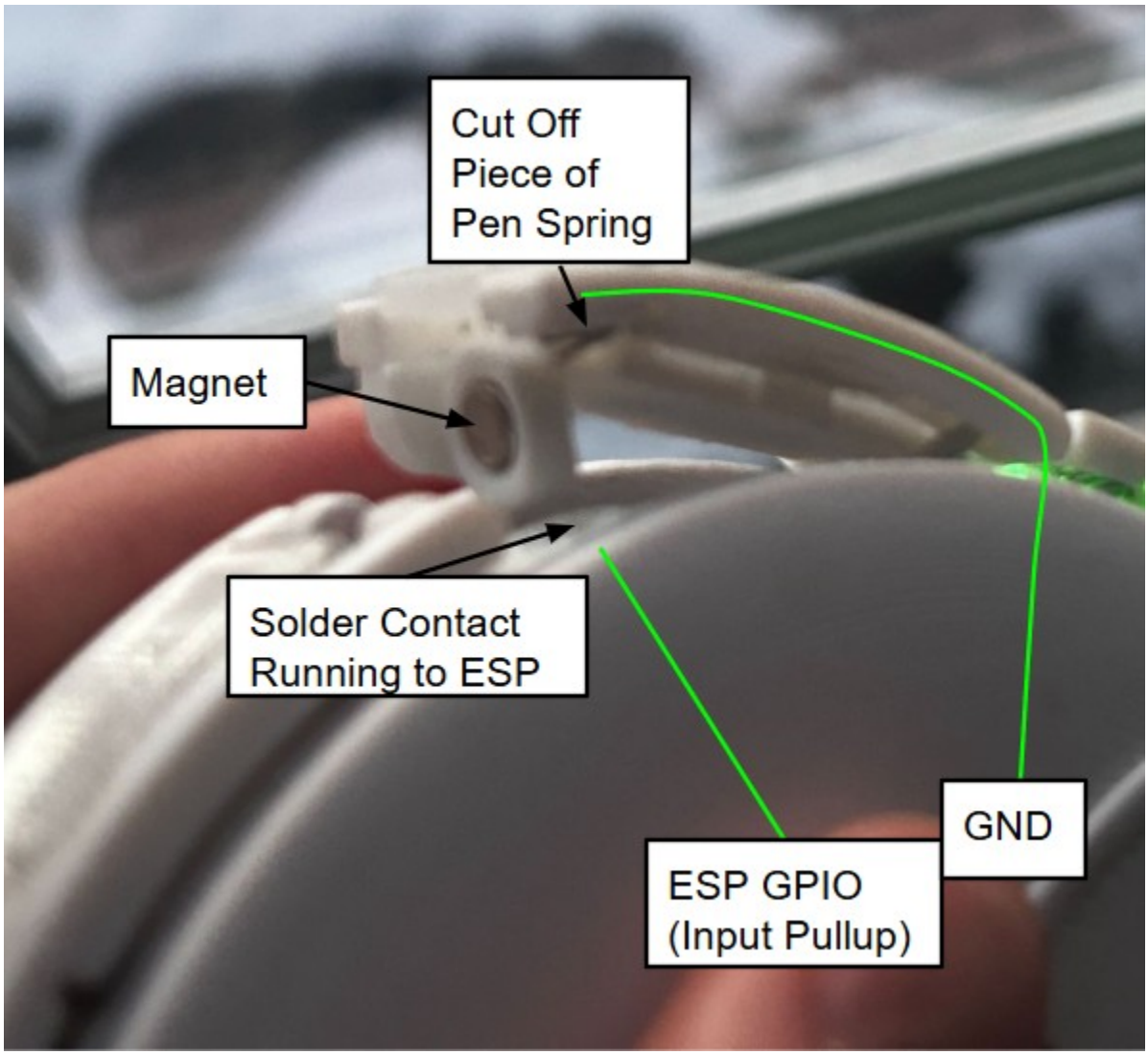
-New design I changed to be two halves with no wires connecting! Can be disassembled easily. I was inspired by a Sovol 3D Printer I was fixing at work- the top half of the heating assembly was just a shell with outward jutting posts.

-I incorporated color theory, or the idea that late stage patients may still associate certain colors with bad, and others with good. I used this principle by having the device blink red, for bad, when medicine is left alone, and finally shine green, for good, once medicine is taken.

-I transitioned to a circular design to double available dosage.

-I transitioned to a hinge design with metal contacts and magnets.

Contacts- Run through hinged compartment covers to pen springs which meet solder contacts when closed. Attached to sensor pins on ESP32



Final Iteration Programming and Wiring

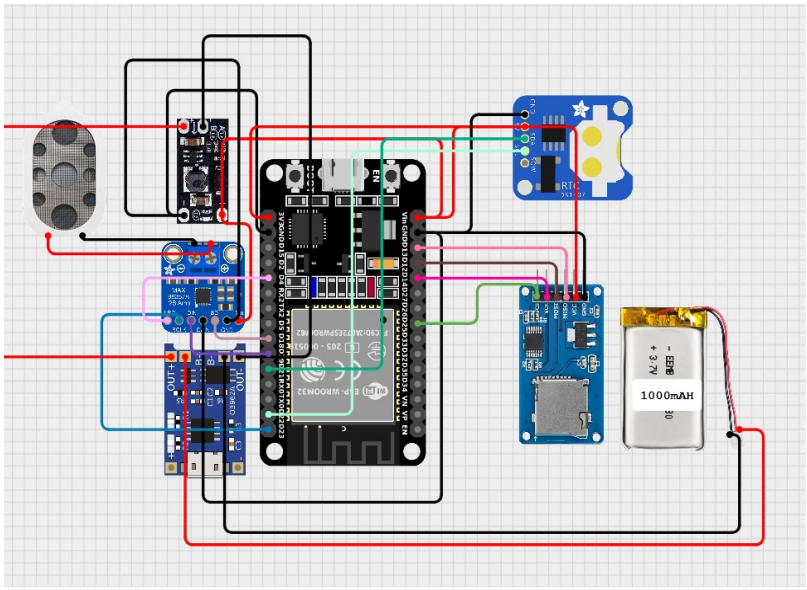
Wiring Changes Made

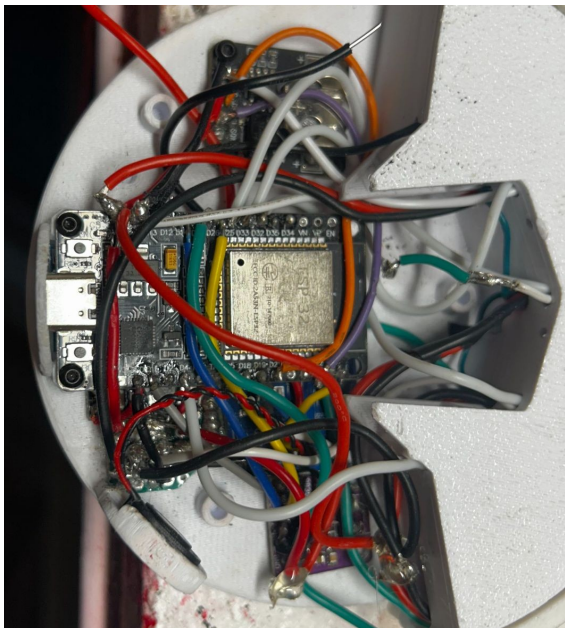
-I incorporated an SD card to allow for larger and more complex audio files

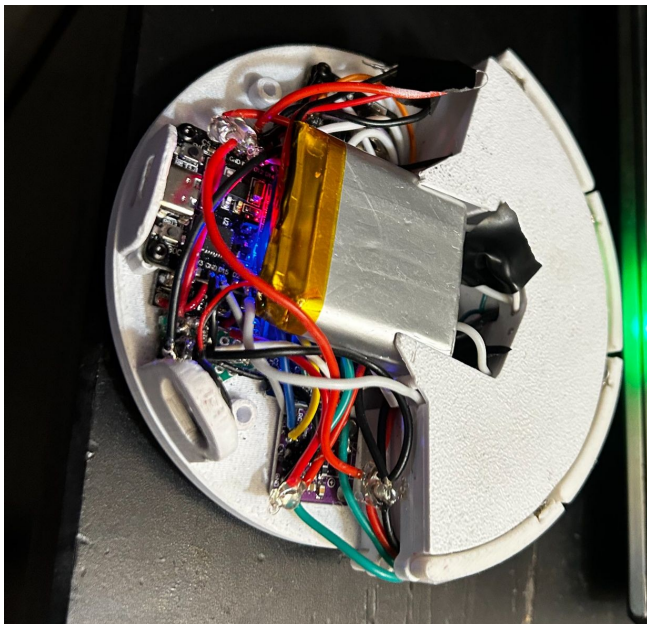
-I incorporated a WSLED strand to provide visual cues to the patient

-I incorporated wired contacts for sensing pills taken instead of a limit switch

Same wiring diagram as last time with the addition of an RTC module hooked into SCL and SDA+bigger battery+4 new sensor pins instead of just 1 (none shown)







Programming Changes Made

-Setting up the device is now much easier, through the use of a captive portal, allowing simple WiFi connection

I wanted to make the website look like an old terminal for the prototype

```
void loop() {
  if (Serial.available()) {
    String cmd = Serial.readStringUntil('\n');
    cmd.trim();
    if (cmd.equalsIgnoreCase("reset")) {
      Serial.println("Rebooting...");
      delay(200);
      ESP.restart();   // hardware-level reset
    }
  }

  if (!setupdevice){
  dnsServer.processNextRequest();
  server.handleClient();
  }
  if (setupdevice){
    DateTime now = rtc.now();
    Serial.println(now.minute());
    if (now.hour()==alarm1Hour && now.minute()==alarm1Minute && alarm1done == 0){
      Serial.println("alarm1trigger");
      strip.setPixelColor(0, strip.Color(255,0,0));
      strip.show();
      alarm1();
    }
    if (now.hour()==alarm2Hour && now.minute()==alarm2Minute && alarm2done == 0){
      strip.setPixelColor(1, strip.Color(255,0,0));
      strip.show();
      alarm2();
    }
    if (now.hour()==alarm3Hour && now.minute()==alarm3Minute && alarm3done == 0 ){
      strip.setPixelColor(2, strip.Color(255,0,0));
      strip.show();
      alarm3();
    }
    if (now.hour()==alarm4Hour && now.minute()==alarm4Minute && alarm4done == 0){
      strip.setPixelColor(3, strip.Color(255,0,0));
      strip.show();
      alarm4();
    }
    delay(200);
  }
}
```





The new setup is simple. Connect to MEDREMINDER on any wifi capable device and set alarm times.

-This solves the problem of target demographic likely not having access to the Arduino IDE or the skills required to set up the program.

Summary

Through completing this project, I dove deep- researching the problems that face the global medical industry. I learned about the challenges that face engineers working to assist the industry. In creating this device, I was able to work in the world of medical engineering, learning more about what it means to be an engineer in a medical adjacent field.

If I were to continue iterating the design, I would expand functionality to allow dependents to be alerted if medicine is not taken. I would also include the ability to upload audio files over the internet or USB connection, as the current way of removing the SD card is unintuitive. Additionally, as I've seen through my research, an important part of finalizing a device such as this one for marketing purposes would include optimizing the build for large scale development- which would likely involve injection molded plastic and a full PCB instead of individual modules.