Panic Attack Monitor

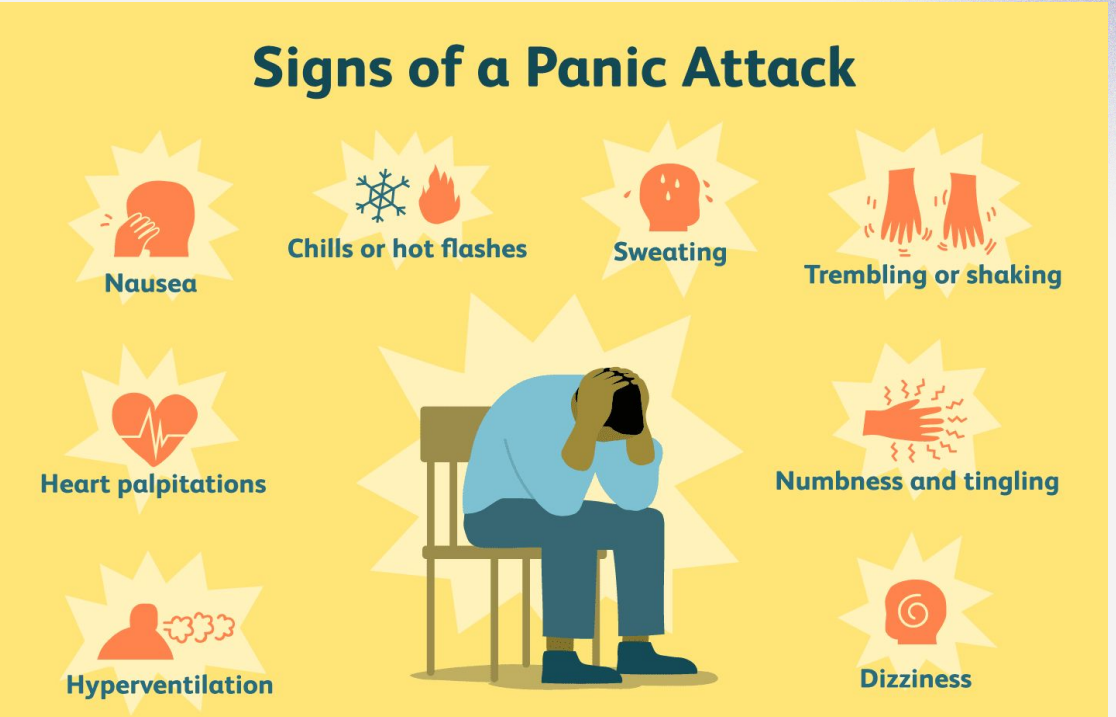
Github Link:

https://github.com/hzc0726/514-FINAL-PROJECT-Chaney-he

Chaney He

Background

In many cases, a panic attack triggers a fast heart rate, also known as tachycardia. The heart rate may speed up to 200 beats per minute or even faster. A fast heart rate can make you feel lightheaded and short of breath. Or you might feel fluttering or pounding in your chest



Proposed Solution

Our proposed solution aims to address the acute needs of individuals experiencing panic attacks through a health monitoring device. By leveraging a PPG sensor to detect physiological markers associated with panic episodes, such as rapid heart rate, the device provides real-time data to users, enabling early intervention and management of symptoms.

The Sensor Device

Sensors:
PPG: detect the pulse of user and update to the display device.

Microcontroller:
An **ESP32**, which is powerful enough for basic DSP/ML tasks and has built-in Wi-Fi for wireless communication.

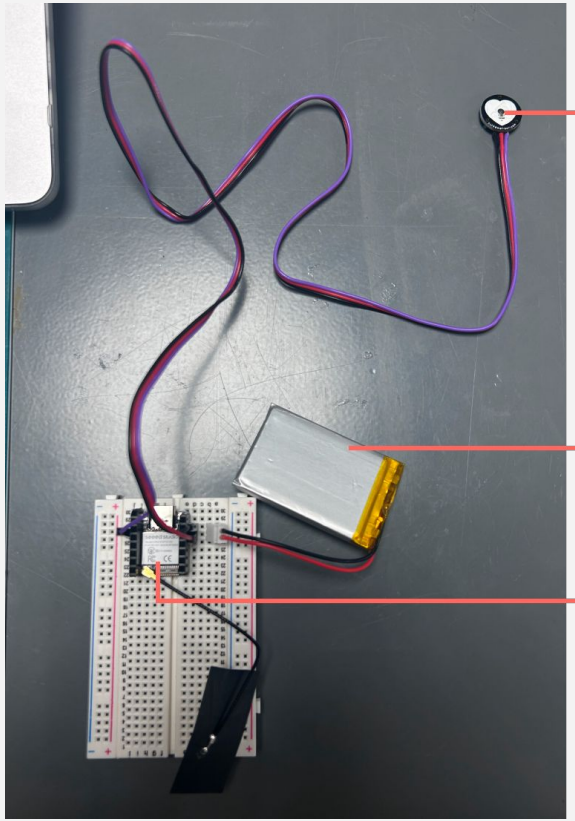
Display Device

LED: when the switch on, the LED turn on.

Stepper Motor Driven Gauge: For the ppg pulse level display.

OLED Screen: displays the data of the pulse and prompts to helps patients take deep breaths when they have a panic attack

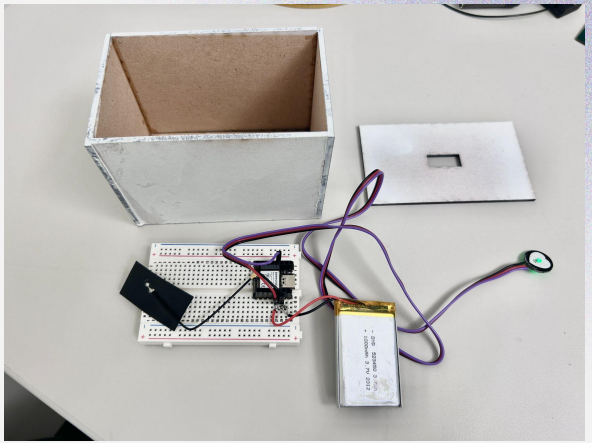
Sensing Device

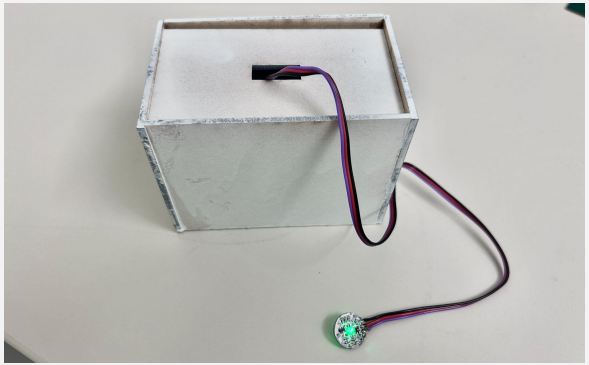


PPG Pulse Sensor

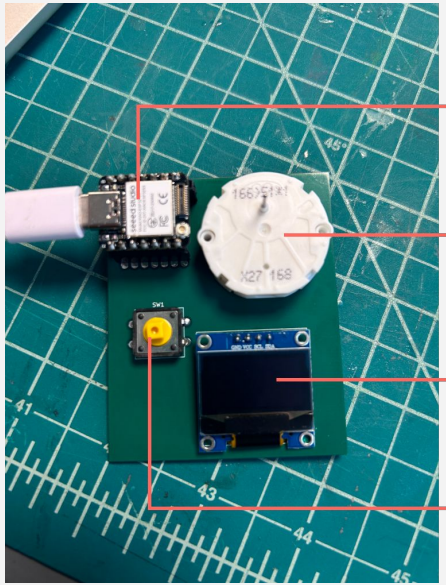
BORMIO 523450 Lipo Rechargeable Battery with JST Connector for Household Appliances 1000mAh 3.7V

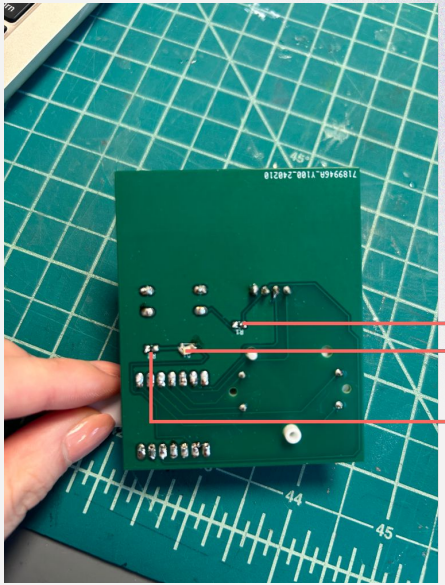
ESP32S3 xiao

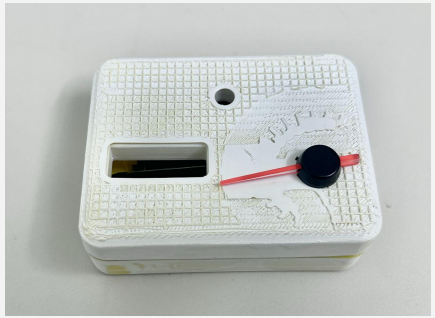


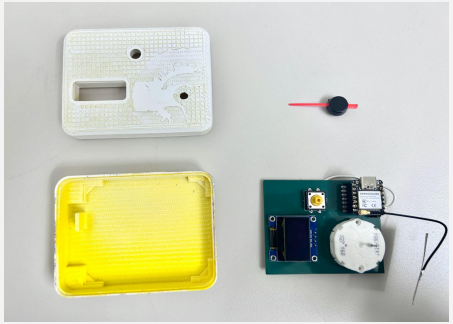


Display Device





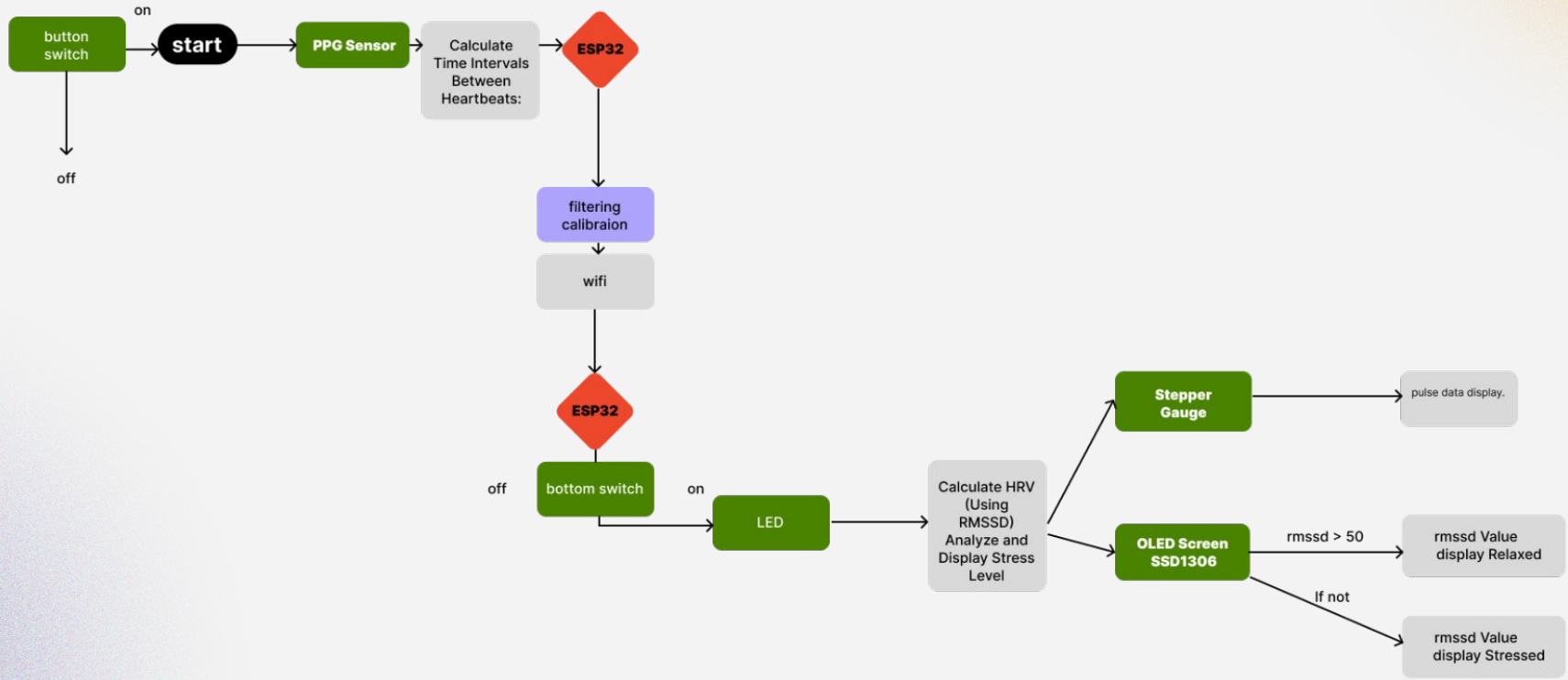




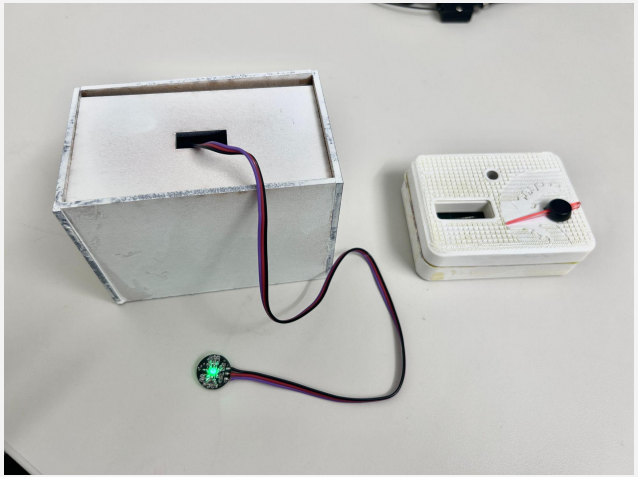
System architecture

The Sensor Device

The Display Device



Power Solution



BORMIO 523450 Lipo Rechargeable Battery with JST Connector for Household Appliances 1000mAh 3.7V

Capacity and Voltage: With a capacity of 1000mAh and a voltage of 3.7V, this LiPo battery provides a suitable power supply for the ESP32 Seeed XIAO and the PPG sensor, ensuring that the device can operate for extended periods without needing frequent recharges. This is particularly beneficial for wearable health monitoring devices that require continuous data collection.

Rechargeability: Being rechargeable, this battery offers convenience and cost-effectiveness over the long term. You can easily recharge the battery once depleted, eliminating the need to constantly purchase and dispose of single-use batteries, which is both economically and environmentally beneficial.

Size and Portability: The BORNMIO 523450 model's compact size is ideal for portable and wearable devices. Its form factor allows for easy integration into your sensing device without adding significant bulk or weight, making it comfortable for users to wear throughout the day.

Compatibility with Household Appliances: The battery's design for use in household appliances indicates its reliability and safety in various applications. This reliability is crucial in health monitoring devices, where consistent and dependable power supply is essential for accurate data collection and analysis.**JST Connector**: The inclusion of a JST connector simplifies the connection process to the ESP32 Seeed XIAO board and other components. This plug-and-play approach reduces the complexity of the device assembly and ensures a secure connection, minimizing the risk of power disconnections during use.

Overall, the BORNMIO 523450 Lipo battery is chosen for its balance of capacity, rechargeability, size, and reliability, making it an excellent power solution for a portable sensing device aimed at continuous health monitoring.

signal processing

The Sensor Device

```
#include <Arduino.h>
#include <BluetoothSerial.h>

BluetoothSerial BTSerial;

const int PPG_PIN = 34; // Assuming PPG sensor is connected to GPIO 34
unsigned long lastBeatTime = 0;
```

```
void setup() {
  Serial.begin(115200);
  BTSerial.begin("PPG_Sensing_Device"); // Bluetooth device name
  pinMode(PPG_PIN, INPUT);
}

void loop() {
  int sensorValue = analogRead(PPG_PIN);
  unsigned long currentTime = millis();
```

```
  // Dummy condition for heartbeat detection; replace with your algorithm
  if (sensorValue > 512 && (currentTime - lastBeatTime > 200)) { // Simple threshold and debounce
logic
    unsigned long beatInterval = currentTime - lastBeatTime;
    lastBeatTime = currentTime;
```

```
    // Send interval to display device
    BTSerial.println(beatInterval);
  }

  delay(1); // Short delay to prevent spamming; adjust based on your needs
}
```

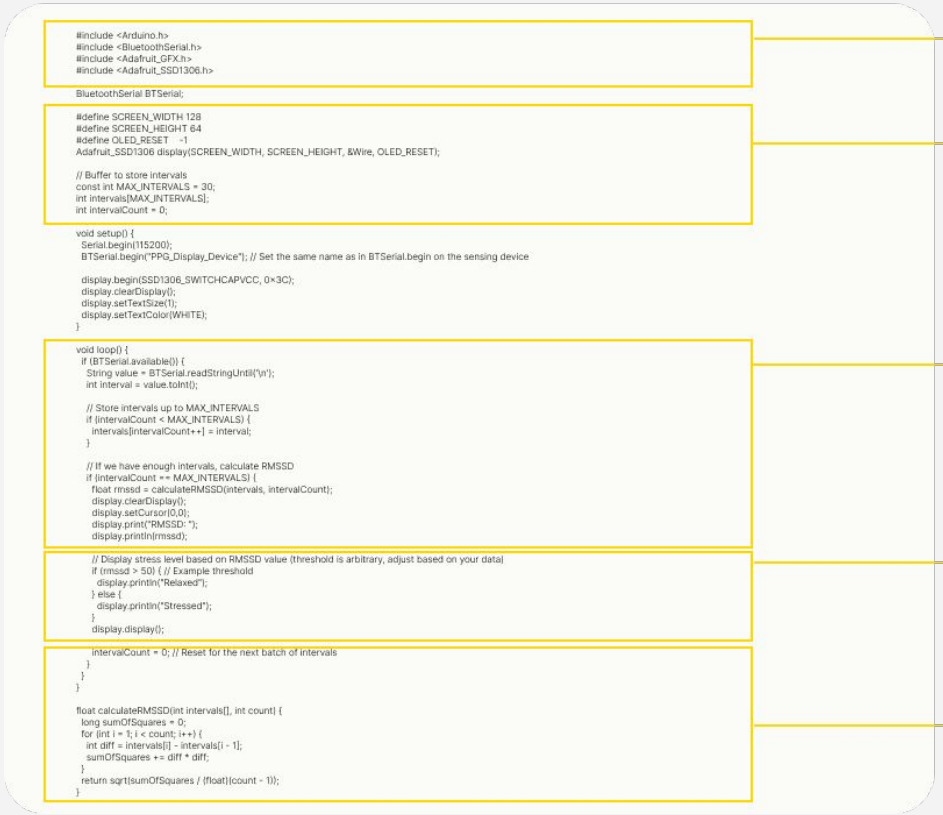
1. Initialize System and Bluetooth Communication:
   - Set up serial communication for debugging purposes.
   - Initialize Bluetooth with a specific name to allow the display device to pair and connect.

2. Read PPG Sensor Data:
   - Continuously read data from the PPG sensor connected to a specific GPIO pin.
   - Implement a simple method to detect heartbeats. In a real application, this would involve more complex signal processing to accurately identify each heartbeat from the PPG signal.

3. Calculate Time Intervals Between Heartbeats:
   - Upon detecting a heartbeat, calculate the time interval since the last detected heartbeat.
   - This interval is crucial for HRV analysis, as HRV is determined by the variability in these time intervals.

4. Transmit Heartbeat Intervals via Bluetooth:
   - Send the calculated time intervals between heartbeats to the display device using Bluetooth.
   - Ensure that each interval is sent as a newline-terminated string for easy parsing by the display device.

The Display Device



```
#include <Arduino.h>
#include <BluetoothSerial.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

BluetoothSerial BTSerial;

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET    -1
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

// Buffer to store intervals
const int MAX_INTERVALS = 30;
int intervals[MAX_INTERVALS];
int intervalCount = 0;

void setup() {
  Serial.begin(115200);
  BTSerial.begin("PPG_Display_Device"); // Set the same name as in BTSerial.begin on the sensing device

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
}

void loop() {
  if (BTSerial.available()) {
    String value = BTSerial.readStringUntil('\n');
    int interval = value.toInt();

    // Store intervals up to MAX_INTERVALS
    if (intervalCount < MAX_INTERVALS) {
      intervals[intervalCount++] = interval;
    }

    // If we have enough intervals, calculate RMSSD
    if (intervalCount == MAX_INTERVALS) {
      float rmssd = calculateRMSSD(intervals, intervalCount);
      display.clearDisplay();
      display.setCursor(0,0);
      display.print("RMSSD: ");
      display.println(rmssd);

      // Display stress level based on RMSSD value (threshold is arbitrary, adjust based on your data)
      if (rmssd > 50) { // Example threshold
        display.println("Relaxed");
      } else {
        display.println("Stressed");
      }
      display.display();

      intervalCount = 0; // Reset for the next batch of intervals
    }
  }
}

float calculateRMSSD(int intervals[], int count) {
  long sumOfSquares = 0;
  for (int i = 1; i < count; i++) {
    int diff = intervals[i] - intervals[i - 1];
    sumOfSquares += diff * diff;
  }
  return sqrt(sumOfSquares / (float)(count - 1));
}
```

1. Initialize System, Bluetooth Communication, and OLED Display:
   - Set up serial and Bluetooth communication.
   - Initialize the OLED display for visual feedback.

2. Receive and Store Heartbeat Intervals:
   - Listen for incoming Bluetooth connections and read the transmitted heartbeat intervals.
   - Store these intervals in an array until enough data is collected for HRV analysis (e.g., 30 intervals).

3. Calculate HRV (Using RMSSD):
   - Once enough intervals are collected, calculate the RMSSD (Root Mean Square of Successive Differences) as a measure of HRV.
   - RMSSD is calculated by taking the square root of the mean of the squares of the differences between successive heartbeat intervals.

4. Analyze and Display Stress Level:
   - Based on the calculated RMSSD, determine the user's stress level. You might set a threshold to categorize the RMSSD values into "Stressed" or "Relaxed" states.
   - Display the calculated RMSSD value and the corresponding stress level analysis on the OLED screen.

5. Loop and Update:
   - Continuously update the stress level analysis as new heartbeat interval data is received and processed.

signal processing (code)

Interaction Between Sensing and Display Devices

- The sensing device focuses on accurately capturing heartbeat data from the PPG sensor and calculating the intervals between heartbeats. These intervals are then transmitted to the display device via Bluetooth.
- The display device receives these intervals, performs HRV analysis by calculating the RMSSD, and uses this information to provide feedback on the user's stress level. This feedback is displayed visually on an OLED screen.
- This setup allows for real-time monitoring and analysis of HRV as an indicator of stress or relaxation, providing valuable feedback to the user.

Budget Summary and future work

Item	Price	number
PPG sencor	24.5	1
OLED screen	9.9	1
stpper motor	9.9	1
	Sum	44.3

. The affordability of the components suggests that the project is cost-effective, making it accessible for personal use or for scaling to a larger production if needed.

The future work for this health monitoring project involves refining the accuracy and reliability of the PPG sensor data through advanced signal processing techniques, enhancing the user interface for better user experience, and incorporating wireless charging for increased convenience. Further development will also focus on implementing machine learning algorithms to predict potential health issues proactively, improving the portability and ergonomics of the wearable device, and ensuring data privacy and security. Additionally, expanding the device's capabilities to monitor other vital signs, such as blood oxygen saturation and temperature, could provide a more comprehensive overview of the user's health status. Collaborating with healthcare professionals for clinical validation and considering user feedback for iterative design improvements will also be pivotal in the evolution of the project.

https://youtube.com/shorts/F1cB1bTlh64