MINI PROJECT REPORT

ON

Embedded Smart Helmet for Soldier Safety using Arduino Uno

Submitted by

Nikhitha B A

NC.SC.U4CSE24121

Submitted to

Dr. V. Thanammal Indu

AP/CSE

For

23CSE201 – Procedural Programming Using C

III Semester

B.Tech CSE B

School of Computing

Amrita Vishwa Vidyapeetham, Nagercoil

Index

1. Abstract

This project presents the design and implementation of a Smart Helmet using Arduino to monitor a soldier's vital parameters and environmental conditions in real time. The helmet integrates a temperature sensor, gas sensor, and heart rate sensor to measure body temperature, detect toxic gases, and track heart rate. A GPS module (simulated here) provides location data, ensuring that a soldier's position can be identified during emergencies. The system also includes an LCD display for real-time monitoring and a buzzer alarm for immediate alerts. The objective is to enhance soldier safety by providing continuous health monitoring and instant notifications of abnormal conditions.

The project is implemented using Arduino Uno, programmed in Embedded C via the Arduino IDE, and tested using Tinkercad simulation. The outcome demonstrates a working prototype capable of detecting abnormal conditions such as high temperature, dangerous gas levels, and irregular heart rate, along with simulated GPS location updates. This project lays the foundation for a real-world solution where live data can be transmitted to a command center for remote monitoring and soldier safety assurance.

2. Introduction

Overview:
The safety and well-being of soldiers in the field is of paramount importance. A helmet embedded with health and environmental sensors provides a low-cost, effective solution to monitor vital signs and surroundings.

Relevance of Arduino and C Programming:
Arduino is chosen due to its affordability, ease of use, and ability to integrate multiple sensors. The programming is done in Embedded C using Arduino IDE, which makes the code portable and efficient for real-time monitoring.

Real-world applications:

- Soldier health monitoring in battlefields
- Worker safety in mines, chemical industries, or hazardous environments
- Disaster management (tracking rescuers)

- Personal protective equipment with IoT integration

Objectives:

- Monitor body temperature, heart rate, and environmental gases.
- Alert through buzzer in case of dangerous conditions.
- Display live data on LCD screen.
- Provide soldier's location using GPS (simulated in this project).

3. Literature Review / Background Study

Existing systems include:

- Wearable health bands (measure heart rate and temperature).
- Industrial gas detectors (measure gas leakage).
- GPS trackers for soldiers.

However, most existing solutions work **individually**. Our project is unique because it **integrates multiple sensors** into a **single helmet system** that simultaneously monitors health, environment, and location. It is cost-effective, modular, and scalable.

4. Problem Statement

Soldiers in hostile or remote environments face risks such as extreme temperatures, toxic gases, and irregular health conditions. Existing monitoring systems are bulky, expensive, and not integrated into wearable equipment.

Motivation:
To provide a compact, affordable helmet-based monitoring system that ensures soldier safety and provides real-time updates to their commanders.

5. System Requirements

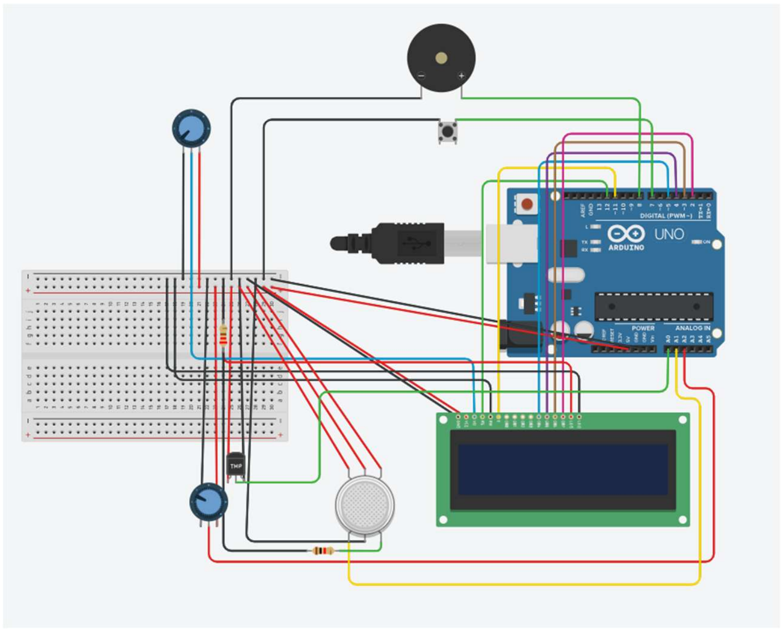
Hardware:

- Arduino Uno microcontroller
- LM35 Temperature Sensor
- MQ Gas Sensor
- Heart Rate Sensor (simulated with potentiometer)
- GPS Module (Neo-6M, simulated here with preset coordinates)
- 16x2 LCD display
- Buzzer for alerts
- Push button for location switching (simulation)
- Breadboard and connecting wires

Software:

- Arduino IDE (C programming)
- Tinkercad for simulation

6. System Design



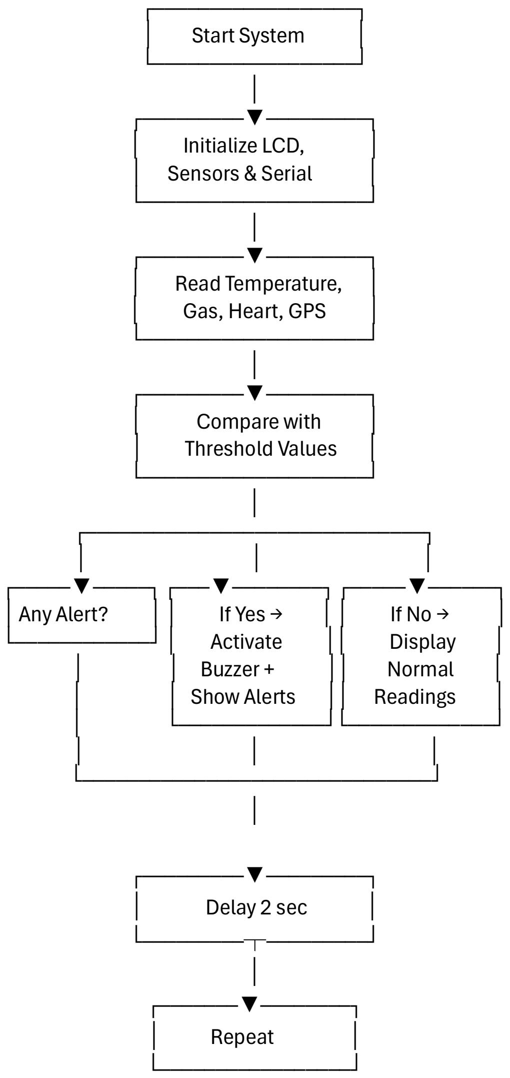
Explanation of Components:

- **Arduino Uno :** Acts as the brain of the project.
- **LM35:** Provides temperature in °C.
- **MQ Gas Sensor:** Detects harmful gas concentration.
- **Potentiometer/Heart Rate Sensor:** Measures beats per minute.
- **Potentiometer/Backlight:** This allows adjusting screen visibility/contrast.
- **Push Button (GPS Module):** Provides soldier's location.
- **Buzzer:** Alerts in emergencies.
- **LCD Display:** Shows readings and alerts.

Algorithm of the Project

1. Start the system.
2. Initialize LCD and Serial Monitor.
3. Read values from:
    - LM35 (Temperature)
    - MQ Sensor (Gas Level)
    - Potentiometer (Heart Rate)
    - Button (GPS simulation)
4. Compare each value with predefined thresholds.
5. If any value exceeds the threshold → Activate buzzer & show alert on LCD + Serial Monitor.
6. Else → Display normal readings.
7. Wait for 2 seconds and repeat.

Flowchart:


Start System
Initialize LCD, Sensors & Serial
Read Temperature, Gas, Heart, GPS
Compare with Threshold Values
Any Alert?
If Yes → Activate Buzzer + Show Alerts
If No → Display Normal Readings
Delay 2 sec
Repeat

7. Implementation

- **Code Development:**

```
#include <LiquidCrystal.h>


// LCD pins (non-I2C)

#define RS 12

#define EN 11

#define D4 5

#define D5 4

#define D6 3

#define D7 2

LiquidCrystal lcd(RS, EN, D4, D5, D6, D7);


// Pins

#define TEMP_PIN A0 // LM35

#define GAS_PIN A1

#define HEART_PIN A2

#define BTN_PIN 7

#define BUZZER 8


// Thresholds

const float TEMP_HIGH_ALERT = 40.0;

const float TEMP_LOW_ALERT  = 10.0;

const int GAS_ALERT = 150;

const int HEART_LOW = 50;

const int HEART_HIGH = 125;
```

```
// GPS locations
const char* gpsLocations[] = {
  "19.0760N,72.8777E",
  "28.6139N,77.2090E",
  "13.0827N,80.2707E",
  "26.9124N,75.7873E",
  "22.5726N,88.3639E"
};
int gpsIndex = 0;
int gpsCount = sizeof(gpsLocations) / sizeof(gpsLocations[0]);
unsigned long lastGpsChange = 0;
const unsigned long debounce = 200;

float readTemperature() {
  int raw = analogRead(TEMP_PIN);
  float voltage = (raw * 5.0) / 1023.0;
  return voltage * 100.0;
}

int readHeartRate() {
  int raw = analogRead(HEART_PIN);
  return map(raw, 0, 1023, 40, 180);
}

int readGas() {
  return analogRead(GAS_PIN);
}
```

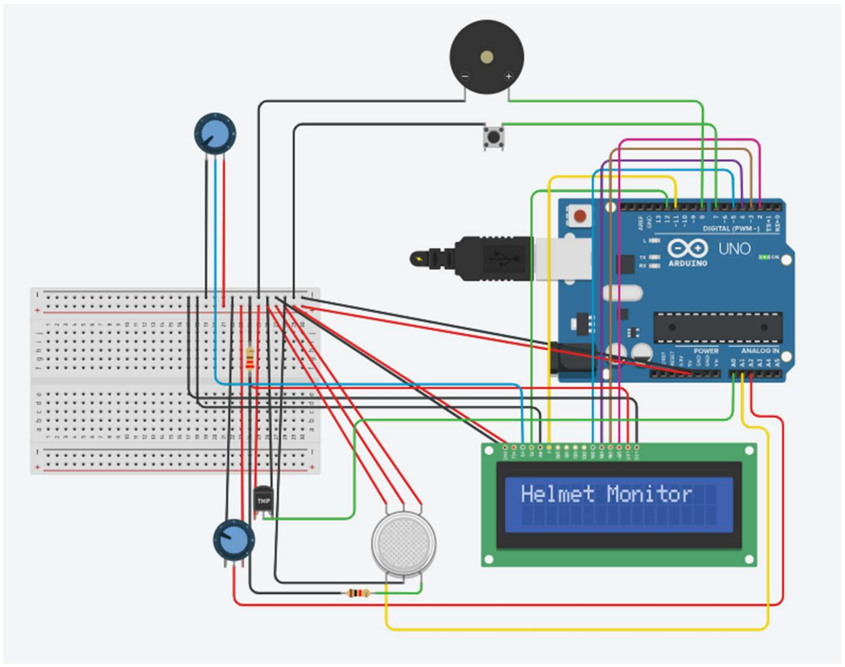
```
void setup() {
  Serial.begin(9600);
  lcd.begin(16, 2);
  pinMode(BTN_PIN, INPUT_PULLUP);
  pinMode(BUZZER, OUTPUT);

  lcd.setCursor(0,0);
  lcd.print("Helmet Monitor");
  delay(1500);
  lcd.clear();
}

void loop() {
  float tempC = readTemperature();
  int gasVal = readGas();
  int bpm = readHeartRate();

  if (digitalRead(BTN_PIN) == LOW) {
    if (millis() - lastGpsChange > debounce) {
      gpsIndex = (gpsIndex + 1) % gpsCount;
      lastGpsChange = millis();
    }
  } else {
    if (millis() - lastGpsChange > 10000) {
      gpsIndex = (gpsIndex + 1) % gpsCount;
      lastGpsChange = millis();
    }
  }
```

```
const char* gpsNow = gpsLocations[gpsIndex];


bool tempHighAlert = (tempC >= TEMP_HIGH_ALERT);
bool tempLowAlert  = (tempC <= TEMP_LOW_ALERT);
bool gasAlert      = (gasVal >= GAS_ALERT);
bool heartAlert    = (bpm < HEART_LOW || bpm > HEART_HIGH);


bool alert = tempHighAlert || tempLowAlert || gasAlert || heartAlert;
if (alert) {
  tone(BUZZER, 1200);
} else {
  noTone(BUZZER);
}


lcd.clear();
lcd.setCursor(0,0);
lcd.print("T:"); lcd.print(tempC,1); lcd.print("C ");
lcd.print("G:"); lcd.print(gasVal);
lcd.setCursor(0,1);
lcd.print("H:"); lcd.print(bpm);
lcd.print(" GPS:");
lcd.print(String(gpsNow).substring(0,5));


unsigned long t = millis() / 1000;
unsigned int sec = t % 60;
unsigned int min = (t / 60) % 60;
unsigned int hr = t / 3600;
```

```
  Serial.println("==================================================");
  Serial.print("[");
  if (hr < 10) Serial.print("0"); Serial.print(hr); Serial.print(":");
  if (min < 10) Serial.print("0"); Serial.print(min); Serial.print(":");
  if (sec < 10) Serial.print("0"); Serial.print(sec);
  Serial.print("]  Location: "); Serial.println(gpsNow);
  Serial.println("--------------------------------------------------");
  Serial.print("Temp  : "); Serial.print(tempC,1); Serial.println(" °C");
  Serial.print("Gas   : "); Serial.println(gasVal);
  Serial.print("Heart : "); Serial.print(bpm); Serial.println(" bpm");
  Serial.println("--------------------------------------------------");
  if (tempHighAlert) Serial.println("ALERT: High Temperature!");
  if (tempLowAlert) Serial.println("ALERT: Low Temperature!");
  if (gasAlert) Serial.println("ALERT: Dangerous Gas Level!");
  if (heartAlert && bpm < HEART_LOW) Serial.println("ALERT: Heart Rate Too Low!");
  if (heartAlert && bpm > HEART_HIGH) Serial.println("ALERT: Heart Rate Too High!");
  if (!alert) Serial.println("All conditions normal.");
  Serial.println("==================================================\n");

  delay(5000);
}
```

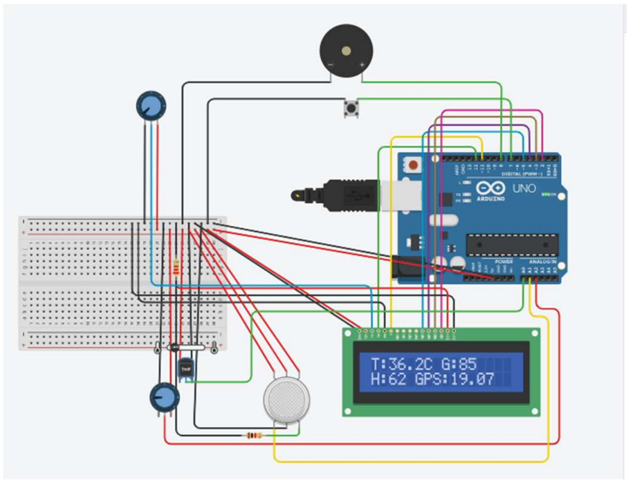
8. Results and Output

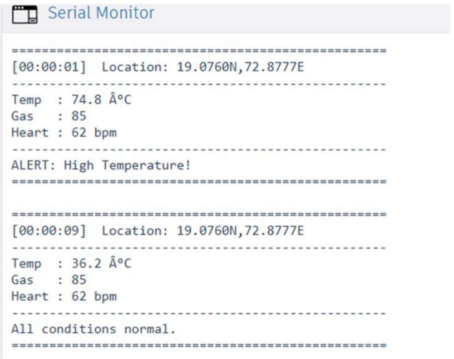
Step 1:



- The information and conditions are displayed on the LCD.

Step 2:

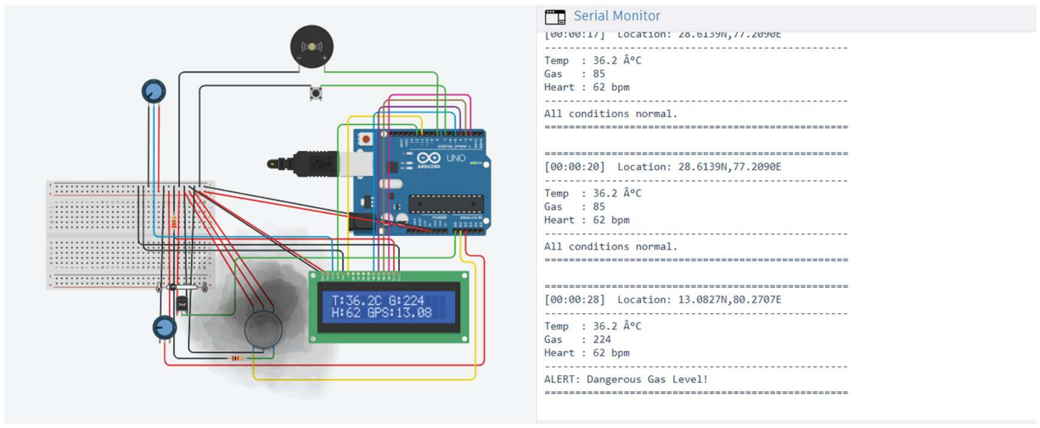




```
================================================
[00:00:01]  Location: 19.0760N,72.8777E
------------------------------------------------
Temp   : 74.8 Â°C
Gas    : 85
Heart  : 62 bpm
------------------------------------------------
ALERT: High Temperature!
================================================

================================================
[00:00:09]  Location: 19.0760N,72.8777E
------------------------------------------------
Temp   : 36.2 Â°C
Gas    : 85
Heart  : 62 bpm
------------------------------------------------
All conditions normal.
================================================
```

- When temperature is higher, it detects and buzzes
- When it is lowered, the temperature condition is normal.

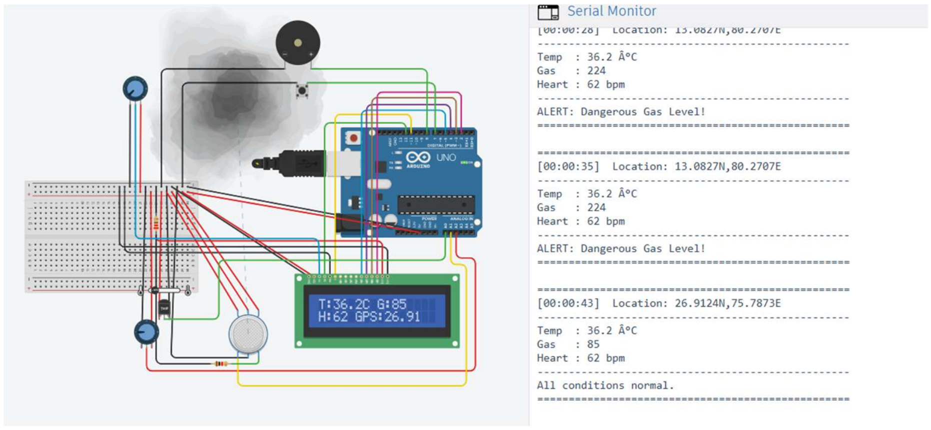
Step 3:



```
[00:00:17]  Location: 28.6139N,77.2090E
---------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 62 bpm
---------------------------------------------
All conditions normal.
=============================================

=============================================
[00:00:20]  Location: 28.6139N,77.2090E
---------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 62 bpm
---------------------------------------------
All conditions normal.
=============================================

=============================================
[00:00:28]  Location: 13.0827N,80.2707E
---------------------------------------------
Temp  : 36.2 Â°C
Gas   : 224
Heart : 62 bpm
---------------------------------------------
ALERT: Dangerous Gas Level!
=============================================
```

- **As the toxic gas passes near the helmet, it detects and gives alert.**

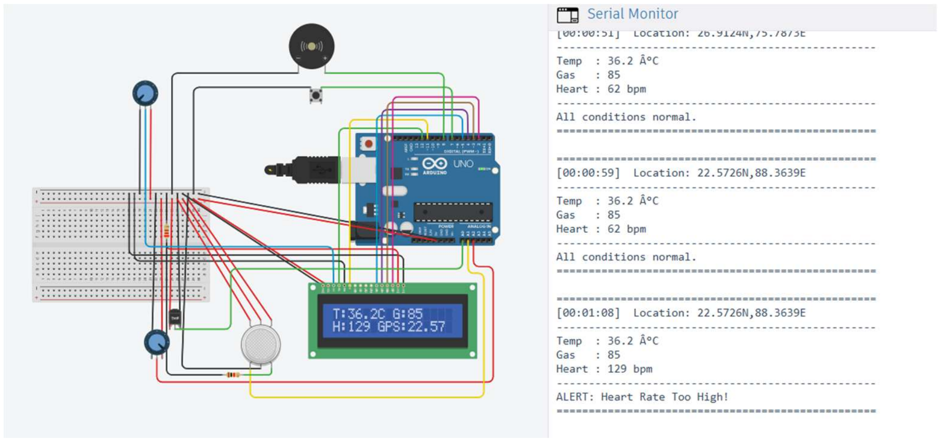
Step 4:



```
[00:00:28]  Location: 13.0827N,80.2707E
---------------------------------------------
Temp  : 36.2 Â°C
Gas   : 224
Heart : 62 bpm
---------------------------------------------
ALERT: Dangerous Gas Level!
=============================================

=============================================
[00:00:35]  Location: 13.0827N,80.2707E
---------------------------------------------
Temp  : 36.2 Â°C
Gas   : 224
Heart : 62 bpm
---------------------------------------------
ALERT: Dangerous Gas Level!
=============================================

=============================================
[00:00:43]  Location: 26.9124N,75.7873E
---------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 62 bpm
---------------------------------------------
All conditions normal.
=============================================
```

- **As the gas is away from the helmet, the gas condition is normal.**

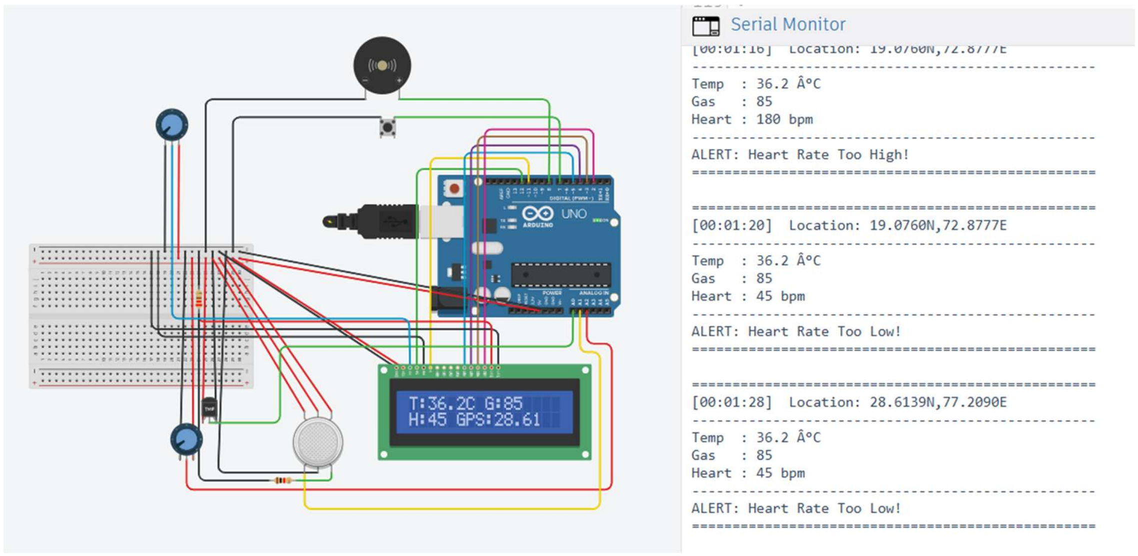
Step 5:



```
[00:00:51]  Location: 26.9124N,75.7873E
---------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 62 bpm
---------------------------------------------
All conditions normal.
=============================================

=============================================
[00:00:59]  Location: 22.5726N,88.3639E
---------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 62 bpm
---------------------------------------------
All conditions normal.
=============================================

=============================================
[00:01:08]  Location: 22.5726N,88.3639E
---------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 129 bpm
---------------------------------------------
ALERT: Heart Rate Too High!
=============================================
```

- **It detects and gives alert when the heart rate raises.**

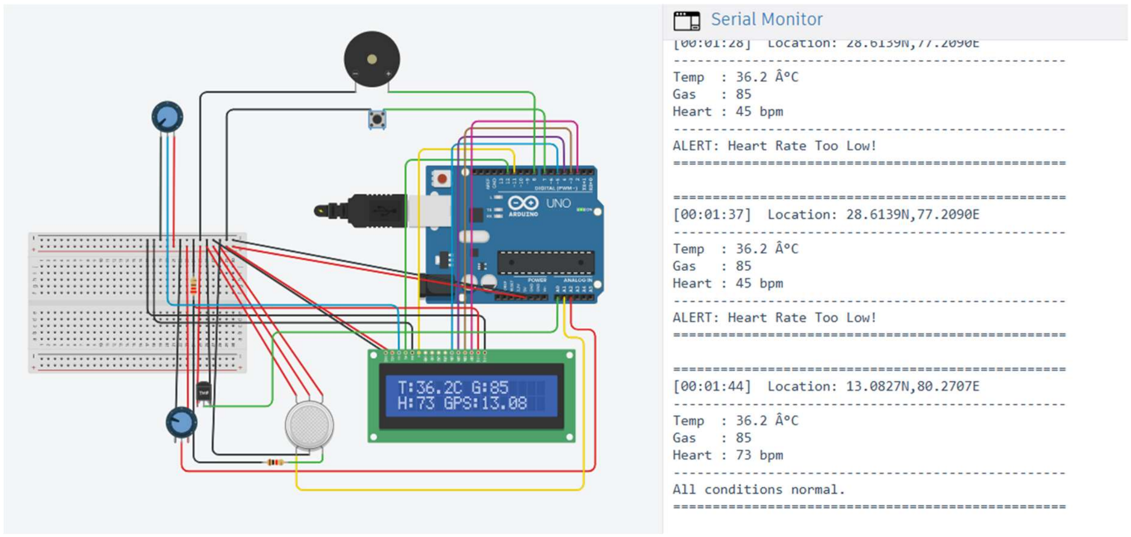
Step 6:



```
[00:01:16]  Location: 19.0760N,72.8777E
-----------------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 180 bpm
-----------------------------------------------------
ALERT: Heart Rate Too High!
=====================================================

=====================================================
[00:01:20]  Location: 19.0760N,72.8777E
-----------------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 45 bpm
-----------------------------------------------------
ALERT: Heart Rate Too Low!
=====================================================

=====================================================
[00:01:28]  Location: 28.6139N,77.2090E
-----------------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 45 bpm
-----------------------------------------------------
ALERT: Heart Rate Too Low!
=====================================================
```

- **It detects and gives alert when the heart rate is low.**

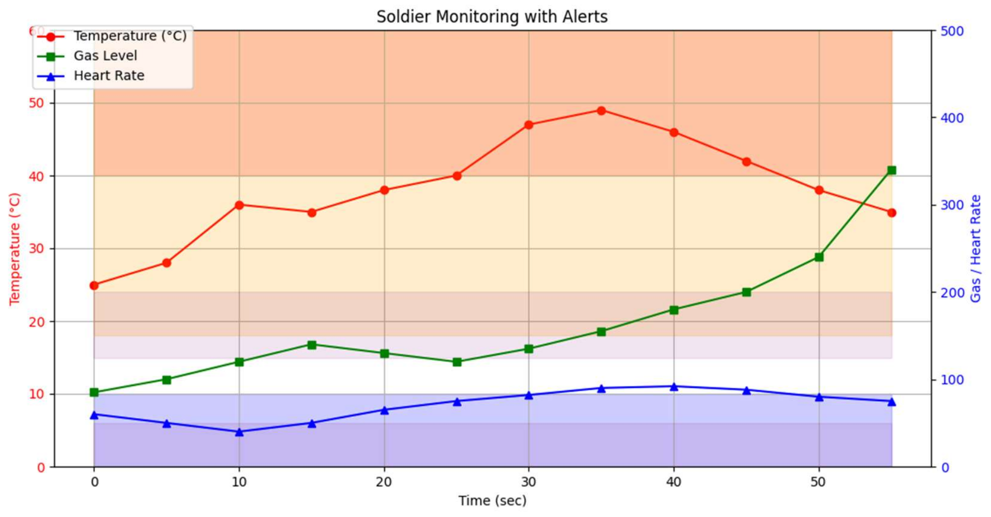
Step 7:



```
[00:01:28]  Location: 28.6139N,77.2090E
-----------------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 45 bpm
-----------------------------------------------------
ALERT: Heart Rate Too Low!
=====================================================

=====================================================
[00:01:37]  Location: 28.6139N,77.2090E
-----------------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 45 bpm
-----------------------------------------------------
ALERT: Heart Rate Too Low!
=====================================================

=====================================================
[00:01:44]  Location: 13.0827N,80.2707E
-----------------------------------------------------
Temp  : 36.2 Â°C
Gas   : 85
Heart : 73 bpm
-----------------------------------------------------
All conditions normal.
=====================================================
```

- **The GPS location is displayed when the button is pressed(for simulation).**

Graph of the temperature, gas and heart rate:



9. Discussion and Analysis

- The system successfully detects abnormal health and environmental conditions.
- Alerts are immediate and easy to notice (buzzer + LCD + Serial).
- Challenge: Gas sensor values in simulation don't exceed 350–400 (Tinkercad limitation). Solved by adjusting threshold values in simulation.
- GPS is simulated in Tinkercad. In real-world implementation, it will provide accurate live location.

10. Applications and Future Scope

Applications:

- Soldier monitoring on battlefields.
- Worker safety in mining, chemical plants, or gas pipelines.
- Emergency rescue missions.
- Industrial IoT and safety helmets.

Future Scope:

- Add wireless communication (GSM, LoRa, WiFi) to send live data to remote server.
- Integrate fall detection (accelerometer).
- Improve GPS to show real-time movement on Google Maps.
- Add solar charging for long-term field use.

11. Conclusion

This project successfully demonstrates a Smart Helmet that monitors vital health parameters (temperature, heart rate) and environmental conditions (gas levels), while providing simulated GPS location. Alerts are generated in case of abnormal readings, ensuring immediate attention. The project highlights the use of Arduino and sensors in safety applications, and it can be extended with real GPS and wireless communication for real battlefield use. Key learnings include sensor interfacing, embedded programming, and the integration of multiple systems for a real-world problem.

12. References

- S. Monk, *Programming Arduino: Getting Started with Sketches*, 2nd ed. New York, NY: McGraw-Hill, 2016.
- Arduino Official Website: https://www.arduino.cc
- Tinkercad Official Website: https://www.tinkercad.com
- Arduino Project Hub (sample projects): https://projecthub.arduino.cc

13. GitHub link of the project

https://github.com/nikhithaaa25/Smart-Military-Helmet-Arduino-And-C