***Report of the Final Project Done in complete Fulfilment of the Requirements for the Award of***

***One Month Internship completion certification***

**HUMAN ELDERLY MONITORING SYSTEM**

**IOT-Based Real-Time Location and Alerting**

**FINAL PROJECT REPORT**

**Done by: TEAM AJCE**

**Jishin Bijumon George Joel James**

**Rohan P Thomas Sharon Thomas**

*Under the guidance of*

**Mrs. Induja & Mrs. Aishwarya**



JUNE-2025

# ABSTRACT

# As the global elderly population continues to rise, ensuring their safety and well-being has become a critical concern, especially for individuals living alone or with limited supervision. This project presents a smart, wearable elderly monitoring system that combines health tracking, fall detection, and emergency alerting in one compact and efficient solution. Built around the ESP32 microcontroller, the device integrates sensors such as the MPU6050 accelerometer for fall detection, the MAX30102 for monitoring heart rate and blood oxygen levels, and the DHT11 for measuring temperature and humidity. In the event of a fall or abnormal health readings, or when the user manually presses an SOS button, the system immediately triggers an alert by sounding a buzzer and blinking an LED for local attention. Simultaneously, it sends a detailed emergency message, including GPS coordinates and vital health data, to predefined contacts via the SIM800L GSM module. The system also supports real-time data transmission over WiFi . Designed with portability, reliability, and ease of use in mind, this wearable solution offers peace of mind for both elderly users and their caregivers, making it a valuable contribution to assistive healthcare technology.

# CHAPTER 1

# INTRODUCTION

With the continuous rise in the global aging population, ensuring the safety, health, and autonomy of elderly individuals has become an increasingly important challenge. While many senior citizens prefer to live independently, they are more vulnerable to health complications, sudden medical emergencies, and accidental falls that demand timely intervention. Traditional monitoring approaches often fall short in providing real-time updates or immediate emergency responses.

To address these critical concerns, this project presents an **IoT-Based Elderly Health and Safety Monitoring System with Real-Time Location and Alerting**. This wearable system is designed to continuously monitor essential physiological and environmental parameters, detect abnormal events such as falls, and trigger real-time alerts—including geolocation—directly to caregivers or emergency contacts. It aims to bridge the gap between independence and safety by providing a smart, responsive, and portable solution.

The system is built around the **ESP32 microcontroller**, which offers built-in Wi-Fi and Bluetooth connectivity with low power consumption—making it ideal for wearable applications. It integrates a range of sensors including the **MAX30102**, which continuously tracks heart rate and SpO₂ levels, and the **DHT11 sensor**, used to monitor the user’s surrounding temperature and humidity. Fall detection is handled by the **MPU6050** accelerometer and gyroscope module, which identifies abrupt movements that indicate a fall. For real-time location tracking, the **NEO-6M GPS module** retrieves the user's coordinates, while the **SIM800L GSM module** is responsible for sending SMS alerts or placing emergency calls during critical events.

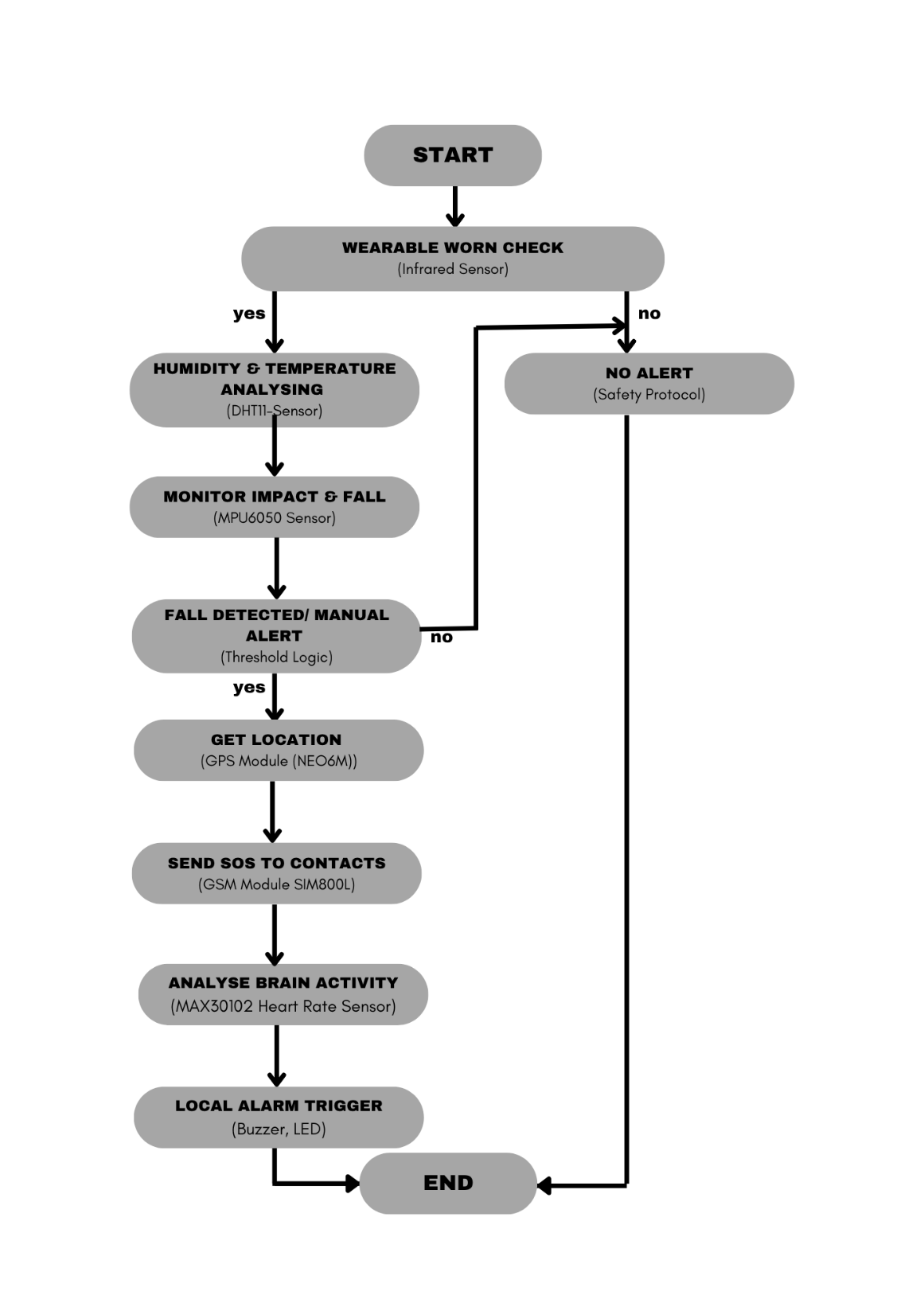
The system includes a **buzzer and LED indicator** for local alert signalling, ensuring immediate attention from nearby individuals. All data including health vitals and environment readings , enabling remote access and monitoring via a mobile or web dashboard by family members or caregivers. Powered by a rechargeable lithium-ion battery, the entire unit is compact, portable, and tailored for continuous operation.

This project thus serves as a comprehensive solution aimed at improving the quality of life and safety for elderly individuals, particularly those living alone, by combining wearable technology with the power of IoT-based real-time monitoring and alerting.

# CHAPTER 2

# WORKING OF SYSTEM

**2.1. System Modelling**



*Figure 2.1 Block diagram of Human Elderly Monitoring System*

The modelling illustrates a **wearable safety system** designed for elderly individuals, integrating multiple sensors and modules with an **ESP32 microcontroller**. The system begins with an **infrared sensor** check to confirm the device is being worn. If active, it monitors **temperature and humidity (DHT11)**, detects falls using the **MPU6050 accelerometer**, and tracks **heart rate and SpO₂ (MAX30102)**.

In the event of a fall or when the **SOS button** is pressed, the system fetches the user’s location via the **NEO-6M GPS module** and sends emergency alerts using the **SIM800L GSM module**. Simultaneously, it triggers a **buzzer and LED** for local alerts. All health and environment data are updated, enabling remote monitoring by caregivers. The device is powered by a rechargeable battery, making it portable and reliable for continuous elderly care.

## **2.2. Description of System Components**

1. **ESP32 Microcontroller**



***Figure 2.2 ESP 32 Module***

The ESP32 acts as the brain of the system, reading sensor values and controlling the output devices. It connects to WiFi and transmits data to Firebase, enabling remote monitoring. Based on sensor data, it activates the LED and buzzer when gas leakage is detected.

1. **NEO-6M GPS Module**



***Figure 2.3 NEO-6M GPS***

The NEO-6M GPS Module is a GPS receiver that provides real-time location data such as latitude, longitude, and altitude. It uses the u-blox NEO-6M chip and communicates via UART, making it easy to interface with microcontrollers like the ESP32. The module features a built-in antenna and backup battery for quick satellite acquisition and reliable tracking.

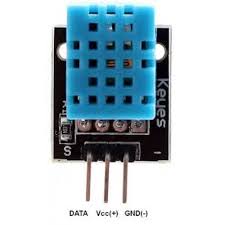
1. **MAX30102 Heart Rate Sensor**



***Figure 2.4 MAX Pulse Oximeter***

The MAX30102 is an integrated pulse oximeter and heart rate sensor that measures heart rate and blood oxygen (SpO₂) levels using infrared and red LEDs. It is compact, low-power, and ideal for wearable health monitoring applications. The sensor communicates with microcontrollers like the ESP32 via the I²C interface for real-time data transmission.

1. **DHT11 Temperature Sensor**



***Figure 2.5 DHT11 Humidity & Temperature Sensor***

The DHT11 is a basic, low-cost digital sensor used to measure temperature and humidity. It provides reliable readings with moderate accuracy and is easy to interface with microcontrollers like the ESP32 via a single-wire digital signal. The sensor is suitable for simple monitoring applications where high precision is not required.

1. **MPU6050 Accelerometer & Gyroscope**



***Figure 2.6 MPU 6050***

The MPU6050 is a 6-axis motion tracking sensor that combines a 3-axis accelerometer and a 3-axis gyroscope in a single chip. It detects orientation, acceleration, and angular velocity, making it ideal for applications like fall detection. The sensor communicates via the I²C interface and is widely used in wearable and motion-sensitive IoT projects.

1. **SIM800L GSM Module**

****

***Figure 2.7 SIM800L GSM Module***

The SIM800L GSM Module is a compact quad-band GSM/GPRS module used for enabling communication via SMS, voice calls, and mobile internet. It allows microcontrollers like the ESP32 to send alerts or receive commands over a cellular network. The module communicates through serial (UART) and is ideal for remote IoT applications requiring mobile connectivity.

1. **Buzzer**

****

***Figure 2.8 Buzzer***

A buzzer is an audio signalling device that produces a sound when powered, commonly used for alarms or notifications. In electronic systems, it serves as an alert mechanism to indicate warnings, such as system errors or emergencies. Buzzers can be easily controlled by microcontrollers like the ESP32 using digital output pins.

1. **IR Sensor**



***Figure 2.9 IR Sensor***

An IR (Infrared) Sensor detects objects or obstacles by emitting and receiving infrared light. It is commonly used for proximity sensing, motion detection, and obstacle avoidance in various electronic systems. The sensor outputs a digital signal and can be easily interfaced with microcontrollers like the ESP32 for automation or safety applications.

1. **Lithium Ion Battery**



***Figure 2.10 Lithium Ion Battery***

A Lithium-Ion Battery is a rechargeable power source known for its high energy density, lightweight, and long cycle life. It is commonly used in portable electronic devices and IoT systems due to its compact size and reliable performance. In microcontroller-based projects, it provides stable power to ensure uninterrupted operation of sensors and communication modules.

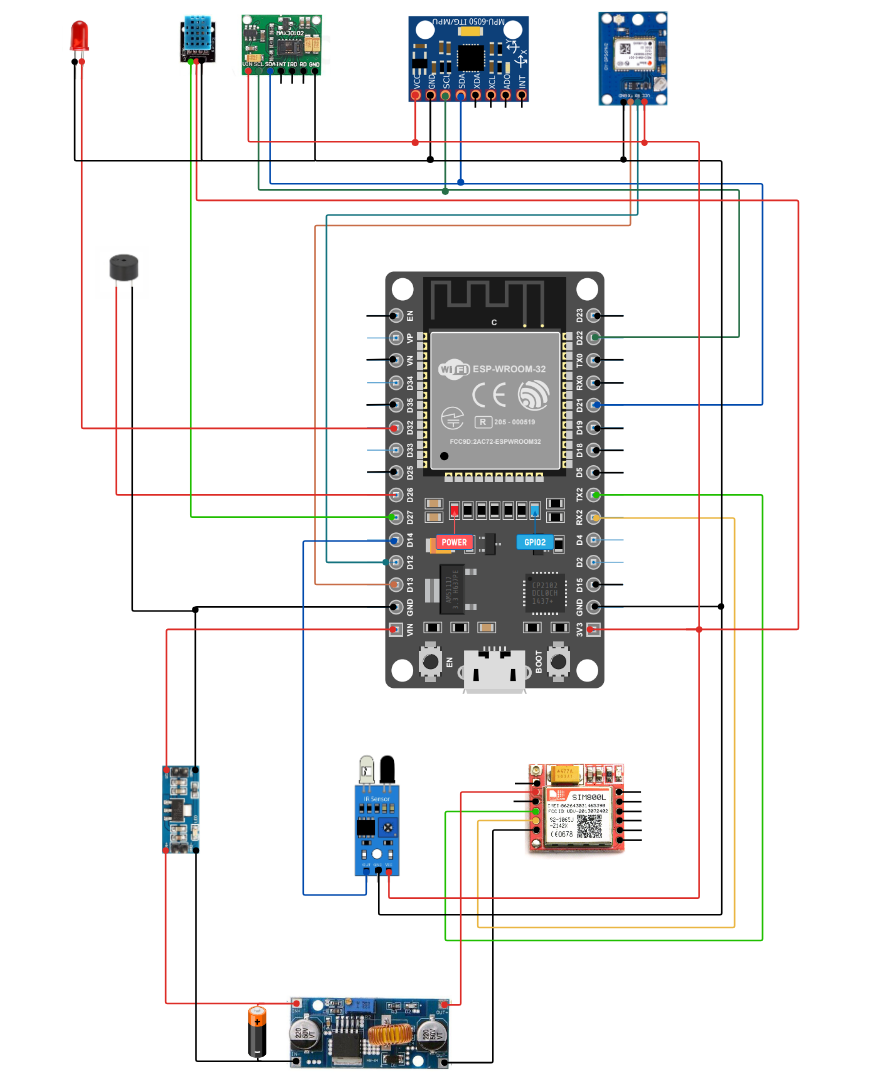
1. **DC-DC Buck Converter**



***Figure 2.11 DC-DC Buck Converter***

The DC-DC buck converter is a power regulation module designed to step down a higher input voltage to a lower, stable output voltage. It operates using a switching regulator that efficiently converts power with minimal energy loss, making it ideal for battery-powered and embedded systems

**2.3. Circuit Diagram**

******

***Figure 2.12 Circuit Diagram***

**2.4 Description of the Working of the System**

The Wearable Elderly Monitoring System is designed to monitor an elderly person’s health and safety conditions in real-time and provide timely alerts during emergencies such as falls or sudden health abnormalities. The entire system is built around the ESP32 microcontroller, which serves as the central processing unit, collecting and processing data from various sensors and modules.

The system begins its operation by checking if the device is being worn using an Infrared (IR) sensor. This ensures that the sensors only operate when the device is properly placed on the user. Once it detects that the device is being worn, the ESP32 starts collecting data from all active sensors.

The DHT11 sensor measures environmental parameters such as temperature and humidity. This helps caregivers monitor the user’s surrounding comfort levels and identify if the person is exposed to unsafe heat or cold conditions. Simultaneously, the MPU6050 sensor, which combines a gyroscope and accelerometer, continuously monitors the user's body movement. If it detects a sudden impact, high acceleration, or free fall beyond a defined threshold, the system assumes a fall has occurred.

In parallel, the system tracks the user's heart rate and blood oxygen level (SpO₂) using the MAX30102 sensor. This helps detect abnormal heart activity or low oxygen saturation, which could indicate a serious medical issue. If any emergency is detected—either due to a fall, abnormal vitals , the system immediately goes into alert mode.

At this point, the NEO-6M GPS module is activated to determine the user’s current location. The SIM800L GSM module then sends a detailed emergency SMS to preconfigured contacts. This message includes the location (in Google Maps link format), the detected health readings, and an alert that immediate help is required.

In addition to remote alerts, the system also triggers a buzzer and LED on the device to alert nearby people, ensuring both local and remote response. All data collected—temperature, humidity, heart rate, SpO₂, fall status, and emergency triggers—are send to the preset contacts. This allows caregivers or family members to monitor the elderly user.

The system is powered by a rechargeable Li-ion battery and is designed to be lightweight, compact, and wearable for daily use. The entire operation runs continuously in the background, automatically responding when any condition matches a danger or emergency scenario. This proactive methodology ensures real-time health tracking, reliable communication, and quick response, making it a valuable solution for elderly care and safety monitoring.

**3.5. Cost of Materials**

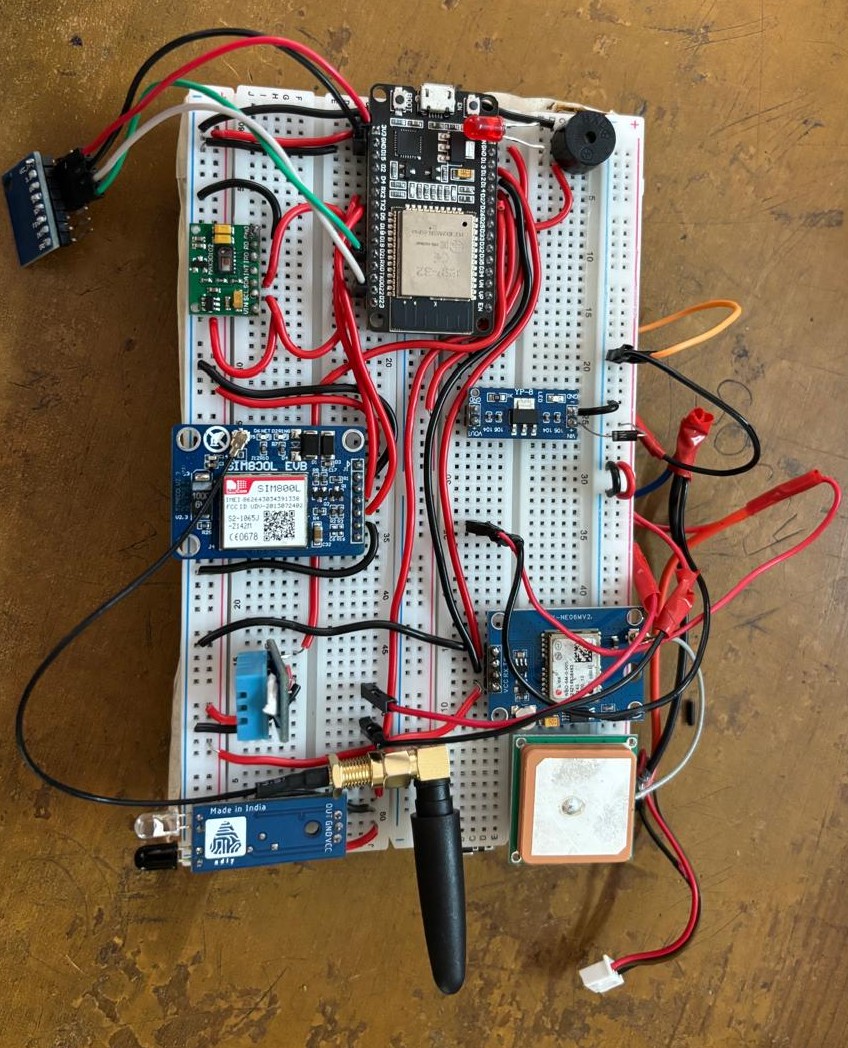
*Table 3.1. Cost of Materials*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.**  **No.** | **Components** | **Specifications** | **Quantity** | **Cost (Rs)** |
| 1. | ESP32 Dev  Module | ESP-WROOM-32,  Wi-Fi & Bluetooth | 1 | 600 |
| 2. | GPS  Module | NEO-6M GPS | 1 | 250 |
| 3. | GSM  Module | SIM800L | 1 | 200 |
| 4. | MPU6050  Sensor | 3-Axis Accelerometer & Gyroscope | 1 | 130 |
| 5. | DHT-11 | Temperature & Humidity Sensor | 1 | 45 |
| 6. | Infrared  Sensor | IR Proximity Sensor | 1 | 30 |
| 7. | Heart Rate  Sensor | Pulse Sensor  MAX30102 | 1 | 120 |
| 8. | Voltage  Regulator | |  | | --- | |  |  |  | | --- | | 3.3V/5V Power Regulator | | 1 | 20 |
| 9. | Lithium Ion  Battery | 2000mAh Li-ion ,  Rechargeable Lithium  -ion Battery, 3.7V | 1 | 500 |
| 10. | Jumper Wire | Male-to-Male | 20 | 100 |
|  |  |  | **Total Cost** | **1995/-** |

## **CHAPTER 3**

## **RESULT**

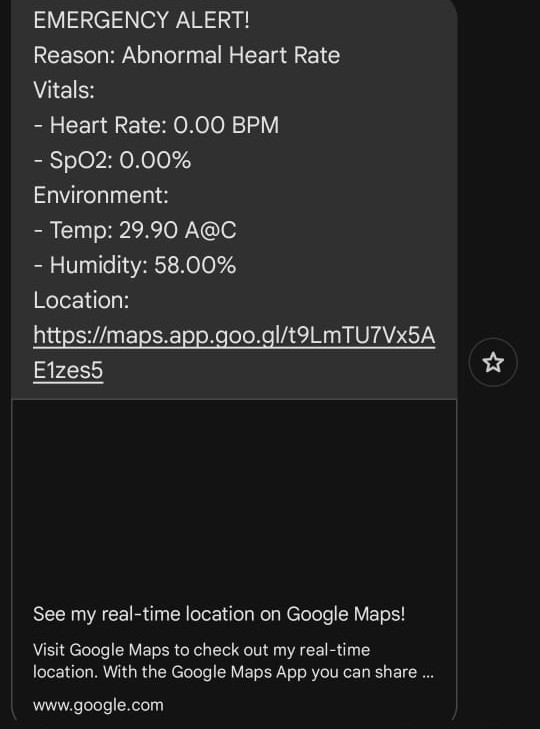
The developed Wearable Elderly Monitoring System successfully achieved its goal of providing real-time health monitoring, fall detection, and emergency alerting for elderly individuals through a compact, portable, and reliable design. The integration of the ESP32 microcontroller with sensors such as the MPU6050, MAX30102, and DHT11 enabled accurate tracking of movement, heart rate, SpO₂ levels, temperature, and humidity. The system effectively detected simulated falls and triggered emergency responses when a fall was detected. Upon activation, the system successfully retrieved GPS coordinates using the NEO-6M module and transmitted emergency messages via the SIM800L GSM module to predefined contacts, confirming reliable communication. The buzzer and LED provided immediate local alerts, ensuring attention from nearby individuals. The device operated efficiently on a rechargeable lithium-ion battery, maintaining stable performance during extended use.



**Figure 3.1 Built Circuit**

****

*Figure 3.2 Trial run*

****

*Figure 3.3 obtained outcome*

**CONCLUSION**

This project successfully demonstrates how technology can be used to support and protect elderly individuals in their daily lives. By combining health monitoring, fall detection, and real-time alerting in a compact wearable system, we were able to build a reliable solution that can respond quickly during emergencies. The system continuously checks vital signs, environmental conditions, and sudden movements, and instantly notifies caregivers through SMS. The addition of visual/audible alerts makes it even more user-friendly and responsive. It worked well during testing, showing accurate sensor readings, reliable GPS tracking, and prompt message delivery. The wearable design and battery-powered operation make it convenient for real-world use. Overall, the project proved to be an effective and thoughtful step toward making elderly care more accessible, connected, and safe.

**CODE OF THE PROGRAM**

**#include <Wire.h>**

**#include "MAX30105.h"**

**#include "heartRate.h"**

**#include <DHT.h>**

**#include <Adafruit\_MPU6050.h>**

**#include <Adafruit\_Sensor.h>**

**#include <SoftwareSerial.h>**

**#include <TinyGPS++.h>**

**// Sensor Definitions**

**#define DHTPIN 27**

**#define DHTTYPE DHT11**

**#define IR\_SENSOR\_PIN 14**

**#define BUZZER\_PIN 25**

**#define LED\_PIN 26**

**// Thresholds**

**#define HIGH\_HEART\_RATE 120**

**#define LOW\_HEART\_RATE 45**

**#define HIGH\_TEMP 39.0**

**#define LOW\_TEMP 34.0**

**#define LOW\_SPO2 90.0**

**#define FALL\_THRESHOLD 20.0**

**// Objects**

**MAX30105 particleSensor;**

**DHT dht(DHTPIN, DHTTYPE);**

**Adafruit\_MPU6050 mpu;**

**TinyGPSPlus gps;**

**// GSM and GPS Setup**

**SoftwareSerial gsmSerial(16, 17); // RX, TX for SIM800L**

**SoftwareSerial gpsSerial(4, 5); // RX, TX for NEO-6M GPS**

**// System Variables**

**const char\* phoneNumbers[] = {"+919037360398"}; // Replace with emergency contact**

**bool deviceWorn = false;**

**bool emergencyTriggered = false;**

**unsigned long lastSensorRead = 0;**

**const unsigned long SENSOR\_INTERVAL = 2000;**

**unsigned long lastDisplayUpdate = 0;**

**const unsigned long DISPLAY\_INTERVAL = 1000;**

**unsigned long lastGpsUpdate = 0;**

**const unsigned long GPS\_INTERVAL = 5000;**

**// Health Metrics**

**const byte RATE\_SIZE = 4;**

**byte rates[RATE\_SIZE];**

**byte rateSpot = 0;**

**long lastBeat = 0;**

**float beatsPerMinute;**

**int beatAvg;**

**float spo2Value = 0.0;**

**// Environmental Data**

**float currentTemp = 0.0;**

**float currentHumidity = 0.0;**

**// GPS Data**

**float gpsLatitude = 0.0;**

**float gpsLongitude = 0.0;**

**bool gpsValid = false;**

**String googleMapsLink = "";**

**void setup() {**

**Serial.begin(115200);**

**delay(1000);**

**Serial.println("==========================================");**

**Serial.println(" ELDERLY MONITORING SYSTEM v3.0 ");**

**Serial.println("==========================================");**

**Wire.begin();**

**gsmSerial.begin(9600);**

**gpsSerial.begin(9600);**

**// Initialize all sensors**

**initMAX30102();**

**initDHT11();**

**initMPU6050();**

**// Configure pins**

**pinMode(BUZZER\_PIN, OUTPUT);**

**pinMode(LED\_PIN, OUTPUT);**

**pinMode(IR\_SENSOR\_PIN, INPUT);**

**// Test alerts and GSM**

**testAlerts();**

**initGSM();**

**Serial.println("\nSystem Ready - Waiting for device to be worn...");**

**Serial.println("------------------------------------------");**

**}**

**void loop() {**

**checkDeviceStatus();**

**// Read GPS data**

**while (gpsSerial.available() > 0) {**

**if (gps.encode(gpsSerial.read())) {**

**updateGPSData();**

**}**

**}**

**if (!deviceWorn) {**

**delay(500);**

**return;**

**}**

**if (millis() - lastSensorRead >= SENSOR\_INTERVAL) {**

**lastSensorRead = millis();**

**readAllSensors();**

**checkEmergencies();**

**}**

**if (millis() - lastDisplayUpdate >= DISPLAY\_INTERVAL) {**

**lastDisplayUpdate = millis();**

**displayReadings();**

**}**

**if (emergencyTriggered) {**

**handleEmergency();**

**}**

**delay(10);**

**}**

**void updateGPSData() {**

**if (gps.location.isValid()) {**

**gpsLatitude = gps.location.lat();**

**gpsLongitude = gps.location.lng();**

**gpsValid = true;**

**// Generate Google Maps link**

**googleMapsLink = "http://maps.google.com/maps?q=" + String(gpsLatitude, 6) + "," + String(gpsLongitude, 6);**

**if (millis() - lastGpsUpdate >= GPS\_INTERVAL) {**

**lastGpsUpdate = millis();**

**Serial.println("GPS Updated: " + googleMapsLink);**

**}**

**} else {**

**gpsValid = false;**

**googleMapsLink = "GPS signal not available";**

**}**

**}**

**// Sensor Initialization Functions (same as before)**

**void initMAX30102() {**

**if (!particleSensor.begin(Wire, I2C\_SPEED\_STANDARD)) {**

**Serial.println("MAX30102 not found!");**

**while(1);**

**}**

**particleSensor.setup();**

**particleSensor.setPulseAmplitudeRed(0x3F);**

**particleSensor.setPulseAmplitudeIR(0x3F);**

**Serial.println("- MAX30102 Initialized");**

**}**

**void initDHT11() {**

**dht.begin();**

**delay(2000);**

**if (isnan(dht.readTemperature())) {**

**Serial.println("- DHT11 Initialization Failed!");**

**} else {**

**Serial.println("- DHT11 Initialized");**

**}**

**}**

**void initMPU6050() {**

**if (!mpu.begin()) {**

**Serial.println("- MPU6050 not found!");**

**while(1);**

**}**

**mpu.setAccelerometerRange(MPU6050\_RANGE\_4\_G);**

**mpu.setGyroRange(MPU6050\_RANGE\_500\_DEG);**

**mpu.setFilterBandwidth(MPU6050\_BAND\_21\_HZ);**

**Serial.println("- MPU6050 Initialized");**

**}**

**void initGSM() {**

**delay(1000);**

**gsmSerial.println("AT");**

**delay(1000);**

**gsmSerial.println("AT+CMGF=1");**

**delay(1000);**

**Serial.println("- GSM Module Ready");**

**}**

**// Core Functions (updated with GPS)**

**void checkEmergencies() {**

**if (beatsPerMinute > HIGH\_HEART\_RATE || beatsPerMinute < LOW\_HEART\_RATE) {**

**triggerEmergency("Abnormal Heart Rate");**

**}**

**if (spo2Value < LOW\_SPO2 && spo2Value > 0) {**

**triggerEmergency("Low Blood Oxygen");**

**}**

**if (!isnan(currentTemp)) {**

**if (currentTemp > HIGH\_TEMP) {**

**triggerEmergency("High Temperature");**

**} else if (currentTemp < LOW\_TEMP) {**

**triggerEmergency("Low Temperature");**

**}**

**}**

**}**

**void triggerEmergency(String reason) {**

**if (emergencyTriggered) return;**

**emergencyTriggered = true;**

**Serial.println("\n!!! EMERGENCY ALERT !!!");**

**Serial.println("Reason: " + reason);**

**String message = "EMERGENCY ALERT!\n";**

**message += "Reason: " + reason + "\n\n";**

**message += "VITAL SIGNS:\n";**

**message += "- Heart Rate: " + String(beatsPerMinute) + " BPM\n";**

**message += "- SpO2: " + String(spo2Value) + "%\n\n";**

**message += "ENVIRONMENT:\n";**

**message += "- Temperature: " + String(currentTemp) + "°C\n";**

**message += "- Humidity: " + String(currentHumidity) + "%\n\n";**

**if (gpsValid) {**

**message += "LOCATION:\n";**

**message += "- Latitude: " + String(gpsLatitude, 6) + "\n";**

**message += "- Longitude: " + String(gpsLongitude, 6) + "\n";**

**message += "- Google Maps: " + googleMapsLink + "\n";**

**} else {**

**message += "LOCATION: GPS signal not available\n";**

**}**

**message += "\nPlease check on the person immediately!";**

**for (int i = 0; i < sizeof(phoneNumbers)/sizeof(phoneNumbers[0]); i++) {**

**sendAlert(phoneNumbers[i], message);**

**}**

**}**

**void sendAlert(String number, String message) {**

**Serial.println("Sending alert to: " + number);**

**Serial.println("Message content:\n" + message);**

**gsmSerial.println("AT+CMGF=1");**

**delay(500);**

**gsmSerial.println("AT+CMGS=\"" + number + "\"");**

**delay(500);**

**gsmSerial.print(message);**

**delay(500);**

**gsmSerial.write(26); // CTRL+Z to send**

**delay(2000);**

**// Check if message was sent successfully**

**while (gsmSerial.available()) {**

**String response = gsmSerial.readString();**

**Serial.println("GSM Response: " + response);**

**}**

**}**

**// Other functions remain the same as your original code**

**void checkDeviceStatus() {**

**bool status = (digitalRead(IR\_SENSOR\_PIN) == LOW);**

**if (status != deviceWorn) {**

**deviceWorn = status;**

**if (deviceWorn) {**

**Serial.println("\nDevice detected on wrist!");**

**Serial.println("Starting monitoring...");**

**} else {**

**Serial.println("\nDevice removed - Monitoring paused");**

**emergencyTriggered = false;**

**digitalWrite(LED\_PIN, LOW);**

**noTone(BUZZER\_PIN);**

**}**

**}**

**}**

**void readAllSensors() {**

**readVitals();**

**readEnvironment();**

**checkMovement();**

**}**

**void readVitals() {**

**long irValue = particleSensor.getIR();**

**if (checkForBeat(irValue)) {**

**long delta = millis() - lastBeat;**

**lastBeat = millis();**

**beatsPerMinute = 60 / (delta / 1000.0);**

**if (beatsPerMinute < 255 && beatsPerMinute > 20) {**

**rates[rateSpot++] = (byte)beatsPerMinute;**

**rateSpot %= RATE\_SIZE;**

**beatAvg = 0;**

**for (byte x = 0; x < RATE\_SIZE; x++) beatAvg += rates[x];**

**beatAvg /= RATE\_SIZE;**

**}**

**}**

**// SPO2 Calculation**

**long redValue = particleSensor.getRed();**

**if (irValue > 50000 && redValue > 0) {**

**float ratio = (float)redValue / (float)irValue;**

**spo2Value = 110.0 - (ratio \* 25.0);**

**spo2Value = constrain(spo2Value, 70, 100);**

**} else {**

**spo2Value = 0;**

**}**

**}**

**void readEnvironment() {**

**currentTemp = dht.readTemperature();**

**currentHumidity = dht.readHumidity();**

**}**

**void checkMovement() {**

**sensors\_event\_t a, g, temp;**

**mpu.getEvent(&a, &g, &temp);**

**float totalAccel = sqrt(a.acceleration.x\*a.acceleration.x +**

**a.acceleration.y\*a.acceleration.y +**

**a.acceleration.z\*a.acceleration.z);**

**if (totalAccel > FALL\_THRESHOLD) {**

**triggerEmergency("Fall Detected");**

**}**

**}**

**void displayReadings() {**

**Serial.println("\n============== CURRENT READINGS ==============");**

**// Health Data**

**Serial.println("HEALTH METRICS:");**

**Serial.print(" Heart Rate: ");**

**if (beatsPerMinute > 0) Serial.print(beatsPerMinute); else Serial.print("--");**

**Serial.print(" BPM (Avg: ");**

**Serial.print(beatAvg);**

**Serial.println(")");**

**Serial.print(" SpO2: ");**

**if (spo2Value > 0) Serial.print(spo2Value); else Serial.print("--");**

**Serial.println(" %");**

**// Environment Data**

**Serial.println("\nENVIRONMENT:");**

**Serial.print(" Temperature: ");**

**if (!isnan(currentTemp)) Serial.print(currentTemp); else Serial.print("--");**

**Serial.println(" °C");**

**Serial.print(" Humidity: ");**

**if (!isnan(currentHumidity)) Serial.print(currentHumidity); else Serial.print("--");**

**Serial.println(" %");**

**// GPS Data**

**Serial.println("\nLOCATION:");**

**if (gpsValid) {**

**Serial.print(" Latitude: "); Serial.println(gpsLatitude, 6);**

**Serial.print(" Longitude: "); Serial.println(gpsLongitude, 6);**

**Serial.print(" Google Maps: "); Serial.println(googleMapsLink);**

**} else {**

**Serial.println(" GPS signal not available");**

**}**

**// Motion Data**

**sensors\_event\_t a, g, temp;**

**mpu.getEvent(&a, &g, &temp);**

**Serial.println("\nMOTION:");**

**Serial.print(" Accel - X:"); Serial.print(a.acceleration.x);**

**Serial.print(" Y:"); Serial.print(a.acceleration.y);**

**Serial.print(" Z:"); Serial.println(a.acceleration.z);**

**Serial.println("==========================================");**

**}**

**void handleEmergency() {**

**static unsigned long lastAlert = 0;**

**if (millis() - lastAlert >= 500) {**

**lastAlert = millis();**

**digitalWrite(LED\_PIN, !digitalRead(LED\_PIN));**

**tone(BUZZER\_PIN, 2000, 300);**

**}**

**}**

**void testAlerts() {**

**for (int i = 0; i < 3; i++) {**

**digitalWrite(LED\_PIN, HIGH);**

**tone(BUZZER\_PIN, 2000, 300);**

**delay(500);**

**digitalWrite(LED\_PIN, LOW);**

**delay(300);**

**}**

**}**