FALL DETECTION AND ALERT SYSTEM FOR ELDERLY

Mini-Project/ Electronic Design workshop (Hardware Part)

EC-681 April-2025

Maulana Abul Kalam Azad University of Technology (formerly known as West Bengal University of Technology)

By

ANSHIKA SHARMA (MAKAUT Roll No: 14800322015)

SANJUKTA CHAKRABORTY (MAKAUT Roll No: 14800322017)

JOYITA CHAKRABORTY (MAKAUT Roll No: 14800323130)

&

TANMAYEE GOUDA (MAKAUT Roll No:14800322065)

Under the mentorship of

Professor Abhijit Kumar Pal

Professor Dr. Koushik Sarkar

(Assistant Professor, Dept of ECE, FIEM)

Department of Electronics & Communications Engineering

Future Institute of Engineering & Management

Sonarpur Station Road, Kolkata-700150

2025

Acknowledgement

It is a great pleasure to express our deepest gratitude and indebtedness to our mentor(s), Prof Abhijit Kumar Pal & Prof Dr. Koushik Sarkar, Department of Electronics and Communications Engineering, Future Institute of Engineering & Management, Kolkata for his/her supervision, constant help and encouragement throughout the entire period.

We are also thankful to all other faculty members and technical staffs of the department for their help and support.

ANSHIKA SHARMA

SANJUKTA CHAKRABORTY

JOYITA CHAKRABORTY

TANMAYEE GOUDA

Future Institute of Engineering & Management

Affiliated to

Maulana Abul Kalam Azad University of Technology (formerly West Bengal University of Technology)

*Certificate of Approval**

The foregoing mini project synopsis is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned don't necessarily endorse or approve any statement made opinion expressed or conclusion drawn therein but approve the synopsis only for the purpose for which it is submitted.

Signature of the Mentor:

…………………………………………….

(Assistant Professor, Dept of ECE, FIEM)

****Only in the case the thesis is approved***

Department of Electronics & Communications Engineering

Future Institute of Engineering & Management

Sonarpur Station Road, Kolkata-700150

2025

INDEX

ABSTRACT

Falls are one of the leading causes of injury and fatality among the elderly, often resulting in serious physical, psychological, and financial consequences. To address this critical issue, this project proposes a smart, real-time Fall Detection and Alert System specifically designed for elderly individuals. The system integrates motion sensors (such as accelerometers and gyroscopes) with intelligent fall detection algorithms to accurately distinguish between normal daily movements and actual falls. Upon detecting a fall, the system immediately sends an automated alert—including location data—to pre-designated caregivers or emergency contacts via SMS or mobile app notification. The solution is designed to be non-intrusive, wearable, and highly responsive, ensuring user comfort and reliability. Advanced features such as machine learning-based motion pattern recognition and cloud integration for health data tracking may also be included to enhance the system's accuracy and scalability. This project aims to improve the quality of life and safety of the elderly by enabling faster response times in emergencies and reducing the risk of unattended falls

Chapter-1: INTRODUCTION

1.1 OVERVIEW:

One of the most frequent and dangerous health hazards that older people face is falls. Older adults are more prone to falls even during everyday activities because their physical strength, balance, and reflexes tend to deteriorate with age. Falls can have serious repercussions, such as fractures, brain damage, and diminished independence. More importantly, if a fall happens while the person is by themselves and unable to call for assistance, the delay in getting medical help may result in potentially fatal consequences.

Fall detection and alert systems have been created as a proactive way to improve care for the elderly in order to address this problem. These devices are made to automatically track an elderly person's movements, detect falls, and promptly alert emergency services or caregivers.

This makes it possible to react quickly, which could save lives and avoid major health effects.

The main way the system functions is by identifying unusual movement patterns that correspond to the features of a fall. A fall usually entails an abrupt loss of equilibrium, quick descent to the ground, and a pause in motion. The system can differentiate between real falls and everyday activities (such as sitting or lying down) by using sophisticated sensing technology and processing algorithms. When a fall is detected, the system automatically notifies pre-designated emergency contacts, which may include information such as the incident's time and location.

These systems are particularly helpful for people who are at a high risk of falling, have chronic illnesses, or are elderly and live alone. Fall detection systems not only lessen the fear of falling but also enable older adults to retain their sense of independence and confidence in their everyday lives by guaranteeing that assistance is available when needed.

These systems keep getting smaller, more precise, and more integrated with other health monitoring solutions as technology develops. They provide users and their families with safety, security, and peace of mind, and they are a major step toward intelligent, tech-enabled eldercare.

1.2 OBJECTIVE:

In order to reduce health risks and encourage independent living, the primary goal of a fall detection and alert system is to automatically detect fall incidents in senior citizens and promptly start an alert mechanism to ensure timely assistance.

Specific Objectives:

- Precise Identification of Falls: To accurately identify fall incidents by employing motion sensors, posture analysis, or sophisticated algorithms to continuously track the movements of senior citizens.

- Immediate Alert Production: To ensure timely intervention by automatically and instantly alerting family members, caregivers, or emergency services when a fall is detected.

- Reduction in False Alarms: To reduce false positives and needless alerts by employing trustworthy algorithms to distinguish between real falls and typical everyday movements (such as sitting or lying down).

- Improving the Safety and Quality of Life of Elderly People: To give senior citizens a safe environment, enhancing their general quality of life and providing users and their families with peace of mind.

1.3 APPLICATION:

1. Medical Facilities and Rehab Facilities:

- Used for senior citizens recuperating from physical therapy, surgery, or stroke.

- Aids in avoiding complications or re-injury from unintentional falls during recovery.

2. Elderly with Chronic Illness or Mobility Issues:

- Applied to individuals suffering from conditions like Parkinson's disease, arthritis, or post-stroke weakness.

- Helps track falls and prevents serious health consequences from delayed assistance.

3. Medical Facilities and Rehab Facilities:

- Used for senior citizens recuperating from physical therapy, surgery, or stroke.

- Aids in avoiding complications or re-injury from unintentional falls during recovery.

4. Residents in Assisted Living or Nursing Homes:

- Distributed among several residents to provide centralized oversight.

- Helps employees react quickly to fall incidents, particularly in low-visibility areas or at night.

1.4 DESCRIPTION OF COMPONENTS WITH SPECIFICATIONS:

1. ESP-32 →

The ESP32 is the central microcontroller used in a Fall Detection and Alert System for the Elderly. It plays a critical role in processing sensor data, detecting falls, and sending alerts to caregivers or medical services. Here's a complete and detailed explanation of the ESP32's involvement in such a system:

Core Function in the System

The ESP32 serves as the brain of the fall detection system. It collects real-time motion data from sensors such as accelerometers and gyroscopes, processes the data using programmed algorithms to detect falls, and initiates alert mechanisms like sending messages, triggering buzzers, or updating cloud dashboards.

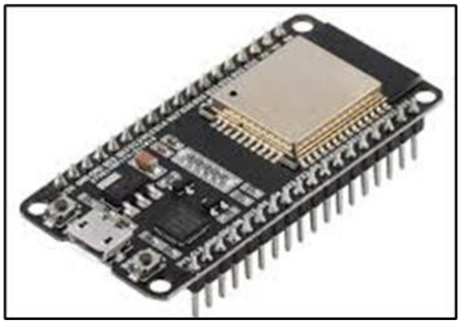


Fig 1.4.1: ESP32

Sensor Integration

The system typically includes motion sensors like the MPU6050, which combines a 3-axis accelerometer and a 3-axis gyroscope. These sensors provide continuous movement data. The ESP32 communicates with these sensors using I2C protocol through its GPIO pins. It interprets changes in motion, orientation, or acceleration that match the patterns of a fall.

ADC Interface: Converts analog signals from other optional sensors to digital for processing.

Digital I/O: Reads digital sensor outputs (e.g., from gyroscope modules).

Data Processing and Fall Detection Algorithm

The ESP32 is equipped with a dual-core processor running up to 240 MHz, allowing it to handle high-speed computations. Algorithms are embedded into its flash memory that evaluate sensor data in real-time. These may include:

Threshold-based algorithms: Checking if acceleration exceeds a preset fall threshold.

Machine learning models: Some systems implement basic ML models that the ESP32 can run locally.

2. MPU6050 →

In a Fall Detection and Alert System for the Elderly, the MPU6050 sensor plays a critical role in detecting movements, orientation, and sudden impacts. It is an Inertial Measurement Unit (IMU) that combines a 3-axis accelerometer and a 3-axis gyroscope in a single chip, making it ideal for motion tracking applications.

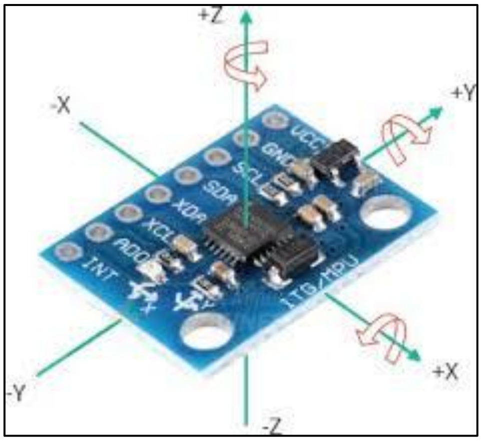


Fig 1.4.2: MPU6050

What is MPU6050?
The MPU6050 is a low-cost, highly accurate motion tracking device developed by InvenSense. It can measure linear acceleration (via accelerometer) and angular velocity (via gyroscope) across three axes: X, Y, and Z. It communicates with microcontrollers like ESP32 via the I2C protocol, which makes it easy to integrate into embedded systems
Role of MPU6050 in Fall Detection System
Motion Monitoring
The MPU6050 constantly monitors the elderly person's movement and orientation. It measures:

Acceleration: sudden changes in movement that might indicate a fall.

Rotation: tilt or angular motion that could reflect loss of balance
Fall Detection Algorithm Input
Data from the MPU6050 is fed into a fall detection algorithm on the ESP32. A typical fall involves:

A sudden high acceleration (impact),

Followed by a period of no movement (lying still),

And a change in body orientation.

The MPU6050 provides the raw data to detect this sequence accurately.
Orientation Detection

The gyroscope helps track how the person's body is rotating. A quick or unusual rotation followed by stillness can help confirm a fall.

3. Buzzer →

In a Fall Detection and Alert System for the Elderly, the buzzer is used as an immediate local alert mechanism. When a fall is detected, the buzzer emits a loud sound to alert nearby people, such as caregivers, neighbors, or passersby, enabling quick assistance. It also serves as a notification to the user, indicating that a fall has been registered and alerts are being sent.

Fig 1.4.3: Buzzer

What is a Buzzer?
A buzzer is an audio signaling device that converts electrical signals into sound. It can be active (has an internal oscillator and sounds with just power) or passive (requires a PWM signal to produce sound). For fall detection systems, an active buzzer is usually preferred for its simplicity.
Role of Buzzer in the System
Fall Alert Notification
When a fall is detected by the ESP32 (using data from the MPU6050), the buzzer is activated to:

Alert people nearby.

Inform the user that an alert has been triggered.

Give the user a chance to cancel the alert in case of a false alarm (if the system has a button for that)
Emergency Confirmation Window
The buzzer can sound for a few seconds before sending a remote alert, giving the user a moment to press a reset/cancel button if they are okay. This helps avoid false emergency messages.
Status Indication
Different sound patterns (like long or short beeps) can be programmed to indicate:

Fall detected

Alert sent successfully

System ready or booting

4. Jumper wires →

In a Fall Detection and Alert System for the Elderly, jumper wires are essential for establishing electrical connections between various components such as the ESP32, MPU6050 sensor, buzzer, power supply, and any other modules (like GSM or GPS, if used). They enable a flexible, solder-free way to build and test the circuit, especially during prototyping on a breadboard.

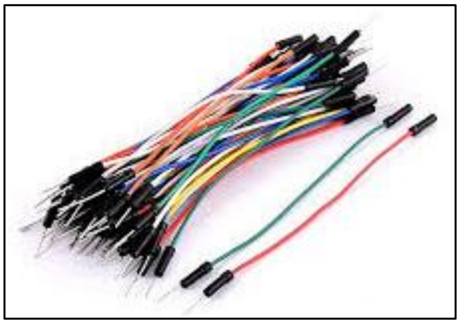


Fig 1.4.4: Jumper Wires

What Are Jumper Wires?
Jumper wires are small, insulated wires with metallic connectors (pins) at both ends, used to connect the pins of different electronic components. They come in three types:

Male-to-Male (M-M): Connect components on breadboards.

Male-to-Female (M-F): Connect microcontroller pins to modules with female headers.

Female-to-Female (F-F): Connect between modules with male pins.

5. Breadboard →

A breadboard is a reusable plastic board with a grid of holes connected internally with metal strips. It allows you to plug in electronic components and jumper wires to create circuits without soldering. It's commonly used for experimenting, debugging, and learning.

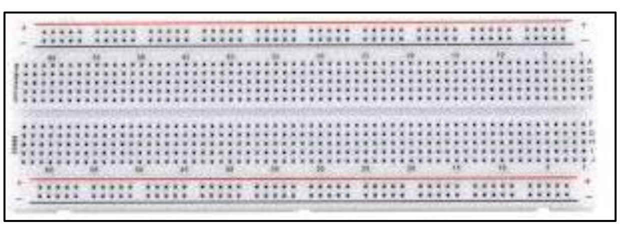


Fig 1.4.5: Breadboard

The breadboard is an essential part of the fall detection system's development and testing phase. It enables quick assembly, troubleshooting, and updates to the circuit layout before finalizing the design. Though not used in the final wearable version, it plays a critical role in validating and refining the system's functionality, making it a vital tool for engineers, students, and hobbyists working on elderly safety and monitoring projects.

6. Max 30102 →

The **MAX30102** is a digital biosensor that measures heart rate and blood oxygen (SpO_2) levels using red and infrared light. In your fall detection project, it is used to **improve the accuracy of fall detection by monitoring vital signs** that may suddenly change during or after a fall. For example, a drop in oxygen saturation or a spike/drop in heart rate can indicate physical distress, helping the system confirm that a real fall has occurred. This sensor works alongside the **MPU6050**, which detects sudden movements or impacts, and the **ESP32** processes the data and sends alerts via **Blynk**. Together, these components make your system more reliable by cross-verifying physical movement with physiological signals, reducing false positives and increasing the effectiveness of real-time health monitoring.

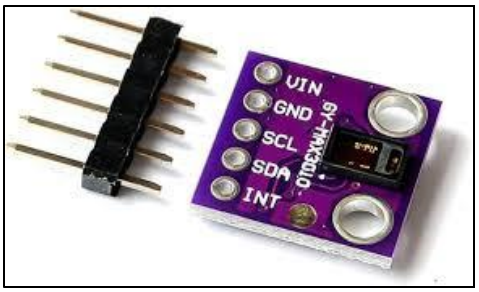


Fig 1.4.6: Max30102

7. Neo-6M-0-001 →

The NEO-6M-0-001 (often referred to as NEO-6M GPS module) is a compact, high-performance GPS receiver that provides accurate location data including latitude, longitude, altitude, speed, and time. In your fall detection project, the NEO-6M is used to track the exact location of the user when a fall is detected. This is especially important for elderly care or remote monitoring, as it allows caregivers or family members to know where the person is during an emergency. The GPS module communicates with the ESP32 via serial (UART) and provides real-time coordinates that can be sent to the Blynk app. By integrating GPS data, your system becomes more effective, enabling not just fall detection but also location-based emergency response.

Fig 1.4.7: Neo-6M-0-001

Chapter-2: LITERATURE SURVEY

Previous Research Work on Fall Detection and Alert Systems →

Over the past decade, fall detection systems have gained significant attention due to the increasing aging population and the high risk of injuries from falls. Researchers have explored a variety of methods to detect and respond to falls in elderly individuals, aiming to enhance safety and reduce medical emergencies.

1. Wearable Sensor-Based Systems:

Many early studies focused on wearable devices using accelerometers and gyroscopes. These sensors measure motion and orientation to detect abnormal changes indicative of a fall.

Example: Bourke et al. (2007) developed a threshold-based system using a tri-axial accelerometer to detect impact and post-fall inactivity.

Pros: Lightweight, low power.

Cons: Require the user to always wear the device.

2. Vision-Based Systems:

Some research used camera systems (like RGB or depth cameras) to monitor movement.

Example: Nait-Charif and McKenna (2004) used ceiling-mounted cameras and HMM (Hidden Markov Models) to analyze fall patterns.

Pros: No need to wear devices.

Cons: Privacy concerns and performance affected by lighting or occlusion.

3. Machine Learning and AI Integration:

Recent work integrates machine learning models (e.g., SVM, neural networks) for more accurate and intelligent detection.

Example: Khan et al. (2014) trained classifiers using wearable sensor data to distinguish between normal activities and falls.

Pros: Better accuracy and adaptability.

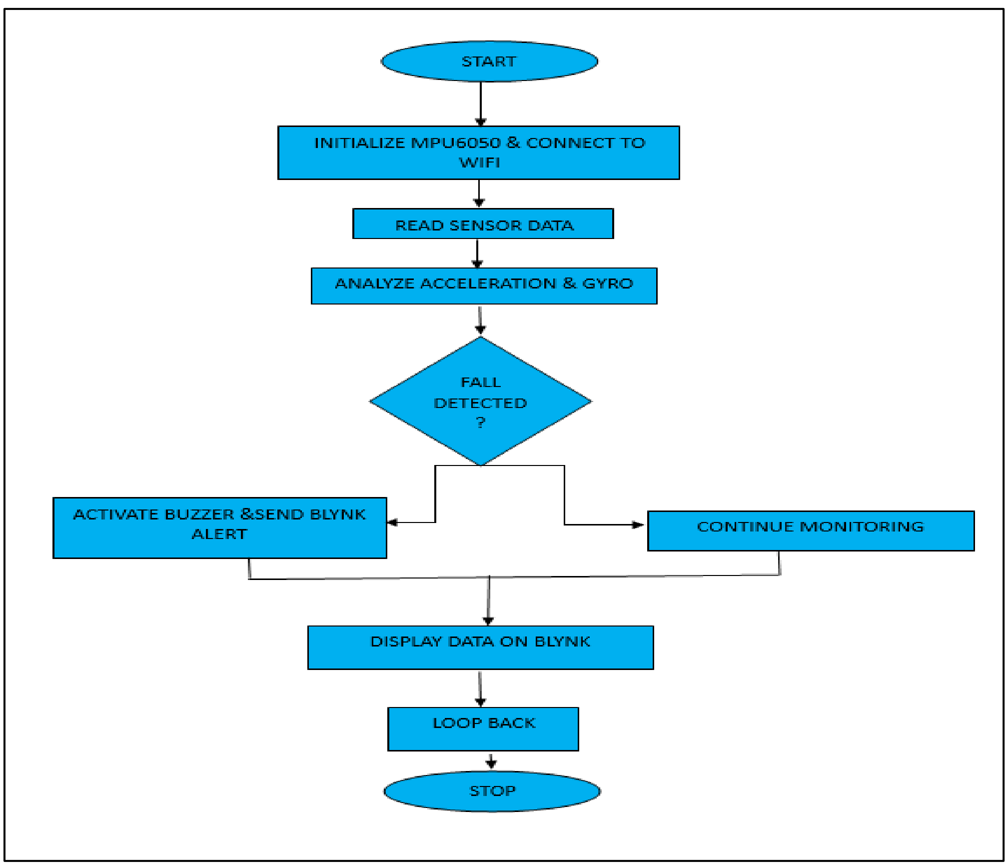
Cons: Requires large datasets and training time.

5. Alert Mechanisms:

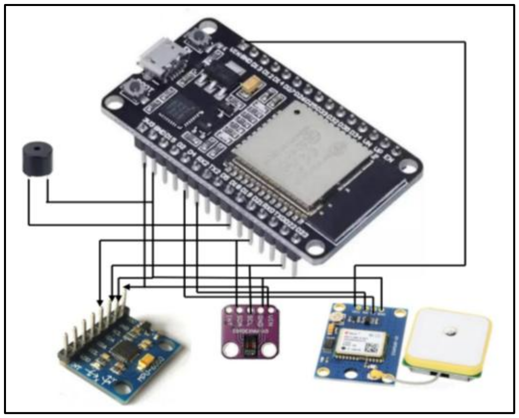
Across most studies, automatic alert systems were included via SMS, mobile apps, or emergency call services, once a fall was detected.
Some recent systems integrate IoT and cloud platforms for real-time monitoring and remote alerts to caregivers.

Chapter -3

3.1 FLOWCHART:



3.2 <u>CIRCUIT DIAGRAM:</u>



3.3 <u>Working Principle:</u>

The fall detection system operates by continuously tracking real-time motion data of a person using the **MPU6050** sensor, which combines a 3-axis accelerometer and gyroscope. The **ESP32** microcontroller collects this motion data and applies an algorithm to detect abnormal movement patterns indicating a fall.
When the combined acceleration exceeds a defined threshold (like 18 m/s^2), or if any single-axis acceleration (ax, ay, az) or angular velocity (gx, gy, gz) surpasses typical limits, a fall is inferred. Upon confirmation, the system instantly triggers an output alert via a **buzzer** and sends a notification to the **Blynk app** using Wi-Fi.
To improve detection accuracy, the system integrates the **MAX30102** sensor, which monitors the user's **heart rate and blood oxygen (SpO_2) levels**. Sudden changes in these vitals (e.g., irregular heartbeat or no finger detected) are used as additional indicators to validate a potential fall, especially useful in reducing false positives.
Furthermore, the **NEO-6M GPS module** provides **real-time location tracking**, enabling the system to share the user's exact location through a **Google Maps link** in the Blynk alert. This helps caregivers or responders quickly locate the individual in case of a fall.
This advanced system ensures a **multi-layered safety mechanism** through:

- **Motion analysis (MPU6050)**
- **Vital monitoring (MAX30102)**
- **Location tracking (NEO-6M GPS)**
- **Remote alerts (Blynk IoT)**
- **Local alert (Buzzer on GPIO25)**

Fall Detection Algorithm
- Thresholds used: totalAccel > 18, ax/ay > 2.0, az < 0.3, and gyro values > 2.
- On fall detection:
  - Buzzer and LED are turned on for 3 seconds.
  - Alert sent to Blynk with message: "Fall Detected! Please check."

Blynk IoT Platform
- Displays real-time sensor values using gauges.
- Sends alerts when fall is detected.
- Accessible from anywhere via smartphone app.

Buzzer (Output Alert)
- Connected to ESP32 on pin 18.
- Activated for 3 seconds when a fall is detected to alert nearby people.

Power Supply
- System powered through USB or battery for portability.
- Powers ESP32 and connected modules (MPU6050, Buzzer).

3.4 <u>RESULT AND DISCUSSION:</u>

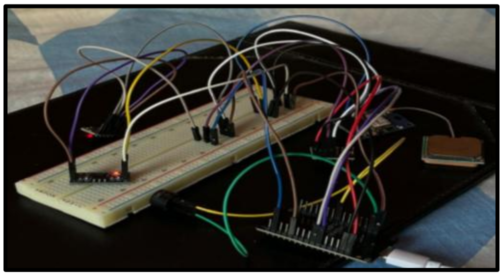


Fig: 3.4.1 HARDWARE

Fig: 3.4.2 SERIAL MONITOR OUTPUT

Fig: 3.4.3 SOFTWARE BLYNK MOBILE NOTIFICATION OUTPUT

Chapter-4: FUTURE ADVANCEMENT

The fall detection system developed in this project can be further enhanced by integrating additional features and technologies to provide a more robust and comprehensive health monitoring and emergency alert system for the elderly or patients. Some of the key future advancements include:

1. Call Facility Using GSM Module:

A **GSM module (e.g., SIM800L)** can be added to automatically make a voice call to a caregiver or family member if a fall is detected. This ensures that the person receives assistance even if they are unconscious or unable to interact with a smartphone or app. The system can be pre-configured with emergency contact numbers.

2. SMS Alert System:

Besides calling, the GSM module can also send **SMS alerts** containing critical information such as “Fall Detected at [Time]”. This can serve as a redundant alert method in case internet connectivity is not available or the user is in a remote area.

3. Integration with Health Monitoring Dashboard:

The system can be extended to work with a **centralized health monitoring dashboard** accessible via a mobile or web application. It can display heart rate, fall history, and location data, giving caregivers a complete picture of the user’s physical condition over time.

4. Machine Learning-Based Fall Detection:

Instead of using fixed threshold values, a **machine learning model** can be trained using accelerometer and gyroscope data to **recognize different types of falls vs normal activities**. This would significantly **improve accuracy and reduce false alarms**. TensorFlow Lite or Edge Impulse can be used for lightweight deployment on ESP32.

5. Voice Activation Support:

A **voice command system** (e.g., using offline voice recognition modules like Elec house or Arduino Voice Recognition Module) could allow the user to **manually send an emergency alert** by saying a keyword like "Help" in case the fall detection algorithm fails or they are conscious after a fall.

6. Battery Backup and Power Monitoring:

Integrating a **rechargeable battery** with **battery level monitoring** ensures the system works during power outages. An alert can be triggered when the battery is low to avoid device failure during emergencies.

7. Emergency Contact Prioritization:

Allowing users or caregivers to **configure a contact list with priority levels** so that if the first contact doesn't respond, the system automatically alerts the next contact. This improves the reliability of the alert system.

8. Wearable or Compact Design:

Developing the system as a **compact wearable device** (like a wristband or clip-on) would make it more user-friendly and portable. It would also reduce the risk of damage or accidental removal during a fall.

9. Mobile App Notifications with Health Tips:

The Blynk app or a custom app could also send **real-time health tips** or **reminders for medication**, hydration, or walking. This turns the device into a **complete elderly care assistant**.

REFERENCE

1. *MPU-6050 | TDK InvenSense*. (2024, March 13). TDK InvenSense. https://invensense.tdk.com/products/motion-tracking/6-axis/mpu-6050/
2. Impulse Tech. (2023, May 11). IoT Fall Detection System using MPU6050, ESP32, and Blynk - Protect Your Loved Ones! #iot [Video]. YouTube. https://www.youtube.com/watch?v=ycAf2J4Ll3ohttps://ieeexplore.ieee.org/document/9043779
3. ESP32 Resources | Espressif Systems. (n.d.). https://www.espressif.com/en/products/socs/esp32/resources
4. Introduction | Blynk Documentation. (n.d.). https://docs.blynk.io/
5. Patel, H., & Patel, M. (2023, July 16). *Sensors for Falls and Fall Detection Techniques: From the past to the future*. https://pubs.thesciencein.org/journal/index.php/jist/article/view/a575
6. Ruiz-Garcia, J. C., Tolosana, R., Vera-Rodriguez, R., & Moro, C. (2023, July 11). CareFall: Automatic Fall Detection through Wearable Devices and AI Methods. arXiv.org. https://arxiv.org/abs/2307.05275

CONCLUSION

Designed for the elderly, the fall detection system based on ESP32, MPU6050, and Blynk offers a strong and dependable means of guaranteeing safety and prompt intervention in the event of a fall. Accurate fall detection depends on real-time movement monitoring made possible by the combination of accelerometer and gyroscope sensors. The system provides a user-friendly interface to show acceleration and gyro values on three axes using gauges by linking these parts to the Blynk platform, therefore facilitating simple tracking of the person's movement.

Including a buzzer in the system gives immediate notice in case of a detected fall, therefore adding extra layer of safety. Real-time feedback from the system guarantees instant identification of any fall event, therefore facilitating quick response and improving the general safety of elderly people.

Moreover, the initiative has been effectively carried out in a logical sequence, therefore guaranteeing simplicity of comprehension and reproduction. The combination of sensors with the ESP32 microcontroller and the use of the Blynk platform shows a sensible way to create an Internet of Things (IoT) solution for healthcare applications.

By offering a technological tool to track and react to falls—a frequent and possibly harmful event among the elderly—the project greatly advances the field of geriatric care. It not only encourages independence for senior people but also gives carers and family members peace of mind knowing they will be notified in an emergency.
Ultimately, the fall detection system is a creative and useful use of IoT technology that provides a consistent way to guarantee elderly safety and lower fall-related hazards.

APPENDIX

```
#define BLYNK_TEMPLATE_ID "TMPL3Zb8YpMPG"
#define BLYNK_TEMPLATE_NAME "Fall Detection"
#define BLYNK_AUTH_TOKEN "FuLQov4q7LZdazngTC3mzLcyXq6ZDwJv"

#define BLYNK_PRINT Serial

#include <Wire.h>
#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include <MPU6050_tockn.h>
#include <TinyGPSPlus.h>
#include <HardwareSerial.h>
#include "MAX30105.h"
#include "heartRate.h"

// WiFi credentials
char ssid[] = "vivo Y02t";
char pass[] = "Ashok_2019";

// Pin Definitions
#define BUZZER_PIN 25
#define GPS_RX 16
#define GPS_TX 17

// Objects
MPU6050 mpu6050(Wire);
BlynkTimer timer;
```

```
TinyGPSPlus gps;
HardwareSerial SerialGPS(1);
MAX30105 particleSensor;

// Globals
bool fallDetected = false;
bool abnormalHeartRateDetected = false;
unsigned long fallTime = 0;
float lastLat = 0.0, lastLon = 0.0;
float beatsPerMinute;
int beatAvg;
unsigned long lastBeat = 0;

const byte RATE_SIZE = 4;
byte rates[RATE_SIZE];
byte rateSpot = 0;

// Buzzer
void buzz() {
  digitalWrite(BUZZER_PIN, HIGH);
  delay(3000);
  digitalWrite(BUZZER_PIN, LOW);
}

// GPS Update
void updateGPS() {
  while (SerialGPS.available()) {
    gps.encode(SerialGPS.read());
  }

  if (gps.location.isValid()) {
    lastLat = gps.location.lat();
    lastLon = gps.location.lng();
  }
}

// Heart Rate Check
void checkHeartRate() {
  long irValue = particleSensor.getIR();

  if (irValue < 50000) {
    Serial.println("⚠ No finger detected on MAX30102!");
    Blynk.virtualWrite(V6, irValue);
    Blynk.virtualWrite(V7, 0);
    Blynk.logEvent("heart_rate_alert", "⚠ No finger detected on heart rate sensor!");
    abnormalHeartRateDetected = true;
    return;
  }

  if (checkForBeat(irValue)) {
    long delta = millis() - lastBeat;
    lastBeat = millis();
```

```
    beatsPerMinute = 60.0 / (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;

      Serial.print("💓 BPM: ");
      Serial.print(beatsPerMinute);
      Serial.print(" | Avg: ");
      Serial.println(beatAvg);

      Blynk.virtualWrite(V6, irValue); // IR
      Blynk.virtualWrite(V7, beatAvg); // BPM

      if (beatAvg > 100 || beatAvg < 50) {
        abnormalHeartRateDetected = true;
        String alert = "⚠ Abnormal Heart Rate Detected! BPM: " + String(beatAvg);
        Serial.println(alert);
        Blynk.logEvent("heart_rate_alert", alert);
        buzz();
      } else {
        abnormalHeartRateDetected = false;
      }
    }
  } else {
    Serial.print("IR: "); Serial.print(irValue);
    Serial.println(" - No beat detected");

    String alert = "⚠ Abnormal Heart Rate Detected! BPM: 0";
    Blynk.virtualWrite(V6, irValue);
    Blynk.virtualWrite(V7, 0);
    Serial.println(alert);
    Blynk.logEvent("heart_rate_alert", alert);
    abnormalHeartRateDetected = true;
    buzz();
  }
}

// Main sensor function
void sendSensorData() {
  mpu6050.update();

  float ax = mpu6050.getAccX();
  float ay = mpu6050.getAccY();
  float az = mpu6050.getAccZ();
  float gx = mpu6050.getGyroX();
  float gy = mpu6050.getGyroY();
```

```
  float gz = mpu6050.getGyroZ();
  float totalAccel = sqrt(ax * ax + ay * ay + az * az);

  Serial.print("Accel: ");
  Serial.print(ax); Serial.print(" ");
  Serial.print(ay); Serial.print(" ");
  Serial.println(az);

  Serial.print("Gyro: ");
  Serial.print(gx); Serial.print(" ");
  Serial.print(gy); Serial.print(" ");
  Serial.println(gz);

  Serial.print("Total Accel: ");
  Serial.println(totalAccel);

  // Send to Blynk
  Blynk.virtualWrite(V0, ax);
  Blynk.virtualWrite(V1, ay);
  Blynk.virtualWrite(V2, az);
  Blynk.virtualWrite(V3, gx);
  Blynk.virtualWrite(V4, gy);
  Blynk.virtualWrite(V5, gz);

  // Fall detection
  if ((az < 0.3 || ax > 2.0 || ay > 2.0 || totalAccel > 18) &&
      (abs(gx) > 2 || abs(gy) > 2 || abs(gz) > 2)) {
    if (!fallDetected) {
      fallDetected = true;
      fallTime = millis();

      String locationLink = (lastLat != 0.0) ? "https://maps.google.com/?q=" + String(lastLat, 6) + "," + String(lastLon, 6) : "Location not available";
      String alert = "⚠ Fall Detected!\n" + locationLink;
      Serial.println(alert);
      Blynk.logEvent("fall_alert", alert);
      // Test log event for fall alert
      Blynk.logEvent("fall_alert", "🚨 Test Fall Alert");
      buzz();
    }
  }

  if (fallDetected && millis() - fallTime > 5000) {
    fallDetected = false;
  }

  // Heart rate check
  checkHeartRate();

  // Combined alert
  if ((millis() - fallTime < 5000) && abnormalHeartRateDetected) {
    String alert = "⚠ Fall and Abnormal Heart Rate Detected!\nLocation: " +
```

```
      (lastLat != 0.0 ? "https://maps.google.com/?q=" + String(lastLat, 6) + "," + String(lastLon, 6) :
"Location not available");
    Serial.println(alert);
    Blynk.logEvent("fall_and_heart_rate_alert", alert);
    // Test log event for combined fall and heart rate alert
    Blynk.logEvent("fall_and_heart_rate_alert", "🚨 Test Combined Alert");
    buzz();
    abnormalHeartRateDetected = false;
  }
}

void setup() {
  Serial.begin(115200);
  Wire.begin(21, 22);
  pinMode(BUZZER_PIN, OUTPUT);

  Serial.println("Initializing MPU6050...");
  mpu6050.begin();
  mpu6050.calcGyroOffsets(true);
  Serial.println("✅ MPU6050 Initialized");

  Serial.println("Connecting to WiFi...");
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
  Serial.println("✅ WiFi Connected");

  SerialGPS.begin(9600, SERIAL_8N1, GPS_RX, GPS_TX);
  Serial.println("✅ GPS Initialized");

  if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) {
    Serial.println("MAX30102 was not found. Please check wiring.");
    while (1);
  }

  particleSensor.setup();
  particleSensor.setPulseAmplitudeRed(0x0A); // IR
  particleSensor.setPulseAmplitudeGreen(0);  // No green
  Serial.println("✅ Heart Rate Sensor Initialized");

  timer.setInterval(1000L, sendSensorData);
  timer.setInterval(2000L, updateGPS);
}

void loop() {
  Blynk.run();
  timer.run();
}
```

Signature of the Mentor