CSE234:Project Report

Project Title: IoT Based Patient Monitoring System

Abstract:

This project presents a low-cost, IoT-enabled patient monitoring and safety system based on the ESP32 microcontroller. The system continuously tracks heart rate (BPM) and blood oxygen saturation (SpO2) using the MAX30100 sensor, detects falls through the MPU6050 accelerometer/gyroscope, and observes room temperature and humidity via the DHT22. Human presence and strong disturbances are captured by a PIR motion sensor and an SW-420 vibration sensor, respectively. Key readings—temperature, humidity, BPM, SpO2, acceleration magnitude, motion, vibration, and fall status—are displayed locally on a 16×2 I2C LCD and transmitted to a ThingSpeak cloud dashboard every 15–20 seconds for near real-time viewing. For safety, the system triggers a buzzer and opens a door through a servo motor if BPM is critically low or significant vibration is detected; fall events are also flagged with left/right indication. Emphasising practicality for local contexts, the prototype focuses on reliable operation (Wi-Fi reconnection, sensor debouncing) and affordability, offering a feasible solution for home or caregiver settings in Bangladesh, with scope for future integration of mobile alerts and intelligent anomaly detection.

1. Introduction

The rising demand for affordable, continuous health monitoring in Bangladesh necessitates practical, home-friendly solutions. This project develops an IoT-based patient monitoring and safety system using the ESP32 microcontroller and low-cost sensors. It measures heart rate (BPM) and blood oxygen saturation (SpO2) with the MAX30100, detects falls using the MPU6050 accelerometer/gyroscope, and tracks ambient temperature and humidity via the DHT22. Motion and environmental disturbances are captured by a PIR sensor and an SW-420 vibration sensor. Data are shown locally on a 16×2 I2C LCD and transmitted to a cloud dashboard at regular intervals for near real-time observation. For critical events—such as near-zero BPM or significant vibration—the system activates a buzzer and opens a door through a servo motor to support rapid assistance. The design prioritises reliability, low cost, and ease of deployment, aiming to support caregivers and families with timely alerts and accessible health information.

2. Objectives

- Design a low-cost, ESP32-based IoT system for continuous patient monitoring at home.
- Measure and display key vitals: heart rate (BPM) and SpO2 (MAX30100),

temperature and humidity (DHT22).

- Detect safety events: falls (MPU6050), motion (PIR), and strong vibrations/disturbances (SW-420).
- Transmit all readings to a cloud dashboard at 15–20s intervals for near real-time observation.
- Provide local feedback on a 16×2 I2C LCD and trigger immediate alerts via buzzer and servo-driven door opening during critical events.
- Ensure reliable operation through Wi-Fi reconnection, sensor debouncing, and robust timing for updates.
- Maintain ease of deployment and affordability suitable for typical Bangladeshi household settings.

3. System Architecture

3.1 Components Used

• **Microcontroller:** ESP32 for Wi-Fi connectivity and processing.

• **Sensors:**

  • DHT22 for temperature and humidity sensing.

  • PIR sensor for motion detection.

  • Max30100 to detect Oxygen level and BPM in body

  • MPU6050 to detect fall

  • SW-420 to detect Vibration

• **Actuators:**

• Buzzer for Alert

• Servo Motor to open Door

• 1602 display with i2c module to show Oxygen , BPM, Humidity and Temperature •
Cloud Platform: Thinkspeak Cloud Server

• **Mobile Application:** Mobile Alert System

• **Cost Analysis:**

S/N	Components Cost
1	Sensors and Actuators 2000
2	Microcontroller (ESP32) 550
3	Other Instruments + Fare 1562
	Total Cost 4112 tk

3.2 Block Diagram

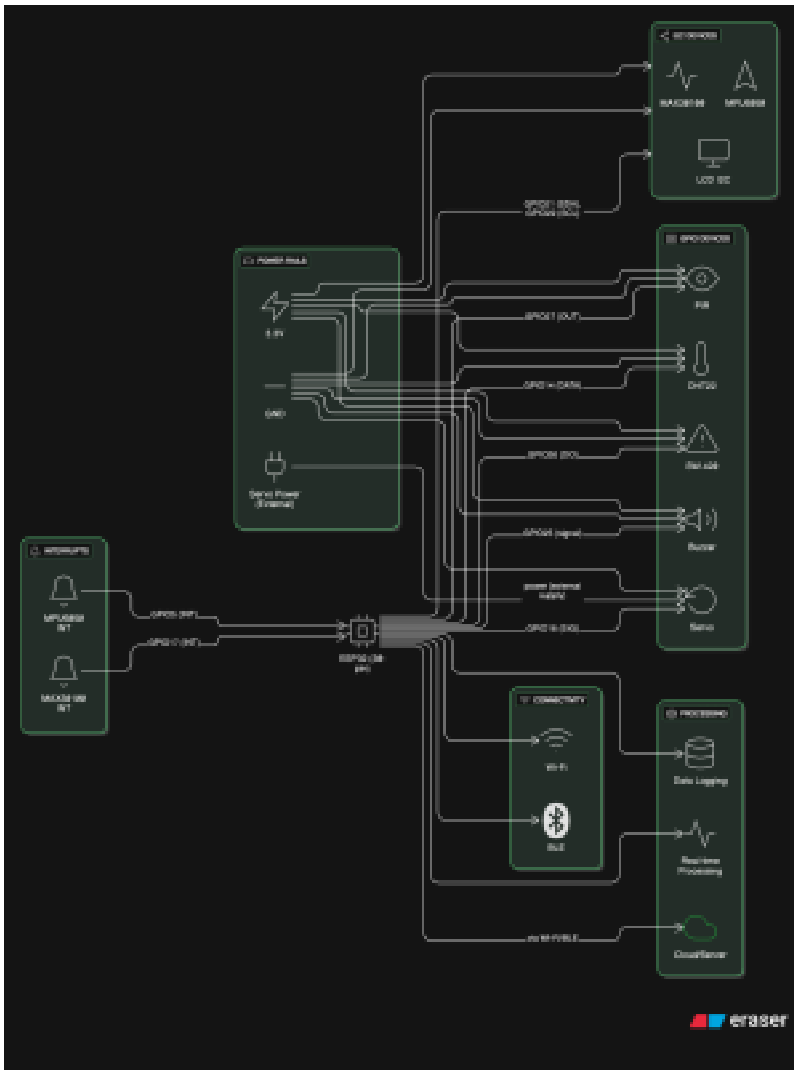

Figure 1: Block Diagram With all connection explained

Explanation:

All modules operate at 3.3V. The ESP32 (30-pin) provides 3V3 to the VCC pin of MAX30100, MPU6050, DHT22, PIR (3.3V type), SW-420 vibration module, 1602 I2C LCD, and the active buzzer; all GND pins are tied to ESP32 GND to ensure a common reference. The I2C bus is shared: GPIO21 (SDA) and GPIO22 (SCL) connect in parallel to MAX30100, MPU6050, and the LCD. The DHT22 data line is connected to GPIO14. The PIR OUT pin is connected to GPIO27, and the SW-420 digital output is connected to GPIO26. The buzzer control line is connected to GPIO25. The servo signal line is connected to GPIO18, while its ground is common with ESP32; its power may be separate if higher current is required. Optional interrupts are connected as follows: MPU6050 INT to GPIO5 and MAX30100 INT to GPIO17. The firmware reads all sensors continuously and, every 16–20s, transmits temperature, humidity, heart rate (BPM), SpO2, acceleration magnitude, motion, vibration, and fall status to the ThingSpeak channel using the Write API key, ensuring near real-time cloud updates for monitoring and visualization.

4. Methodology

4.1 Hardware Design

- **Power architecture**
  - All sensors and logic run at 3.3V from the ESP32; grounds are common across all modules.
  - The servo motor uses a separate regulated 5V supply with sufficient current (≥1–2A), and its ground is tied to ESP32 GND to ensure a common reference.

- **Core controller and buses**
  ○ ESP32 (30-pin) serves as the main controller with Wi-Fi for cloud updates. ○ I2C bus on GPIO21 (SDA) and GPIO22 (SCL) connects MAX30100, MPU6050, and the 1602 I2C LCD; bus lines are kept short and twisted where possible to reduce noise.

- **Sensor interfacing**
  - MAX30100 powered at 3.3V with proper I2C wiring; LED currents configured in firmware for reliable PPG capture.
  - MPU6050 powered at 3.3V with optional INT to GPIO5 for timely motion/fall events.
  - DHT22 data on GPIO14 with an internal or onboard pull-up; sampling rate limited to reduce self-heating.
  - PIR (3.3V version) output to GPIO27; sensitivity and trigger time adjusted on the module as needed.
  - SW-420 vibration module powered at 3.3V with digital output to GPIO26; onboard potentiometer tuned to avoid false triggers.

- **Actuators and protection**
  - Active buzzer driven from GPIO25 at 3.3V; if a louder 5V buzzer is used, a transistor/MOSFET driver and flyback diode are added.
  - Servo signal on GPIO18; 5V servo supply decoupled with a large electrolytic capacitor (470–1,000µF) near the servo connector to handle current spikes.

- **Grounding and layout**
  - Star-ground approach where practical, with a single common ground point near the ESP32.
  - Separation of high-current servo wiring from I2C and analog signal paths to

minimise interference.

- **Reliability and EMI**
  - Short, secure jumper wires; proper strain relief on the USB and servo connectors.
  - Optional small series resistors (22–47Ω) on I2C lines and 0.1µF bypass capacitors near sensor VCC pins for stability.

- **Enclosure and safety**
  - Sensors oriented for correct exposure: MAX30100 window accessible for fingertip, PIR with clear line-of-sight, DHT22 with ventilation.
  - Servo linkage mechanically limited to prevent jams; wiring insulated to prevent shorts.

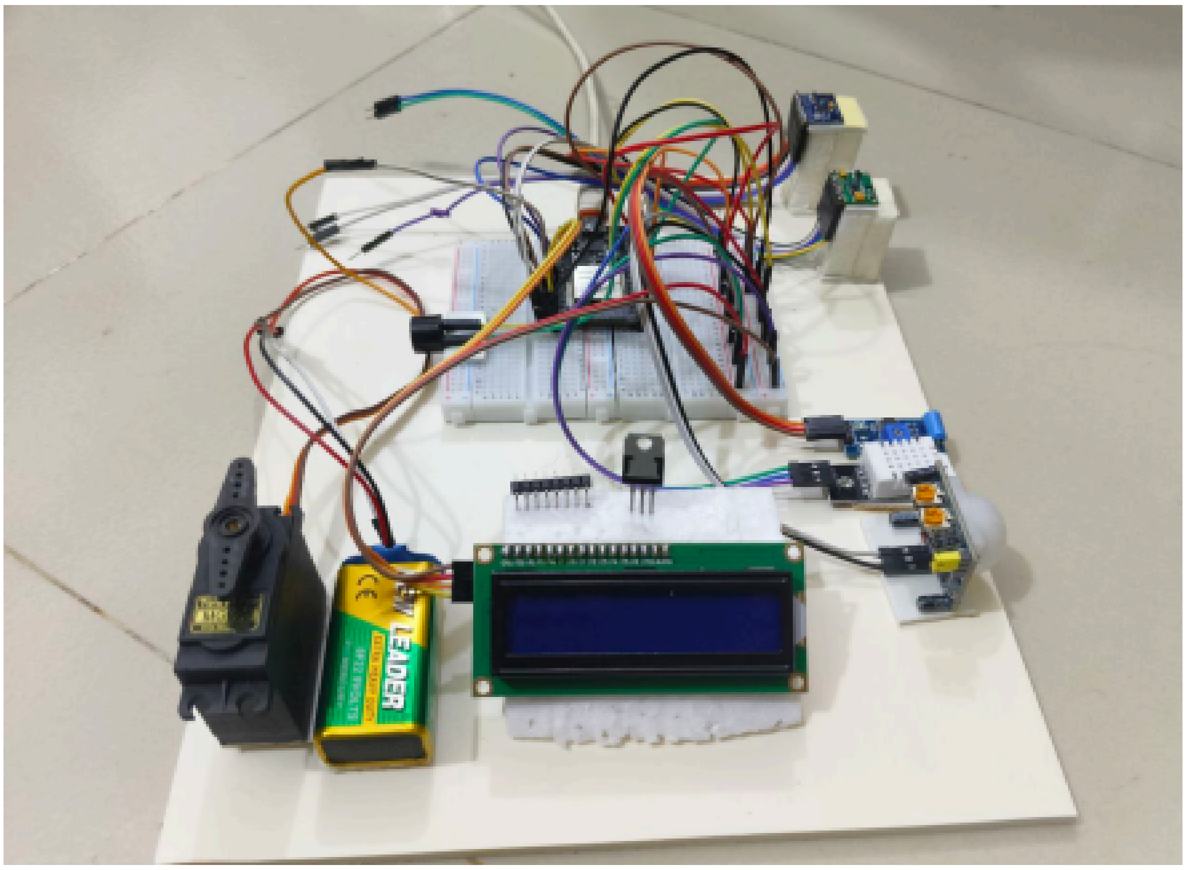

Figure 2: Project with final setup

4.2 Software Design

- **Architecture**
  - Event-driven main loop using millis() timers: fast loop (10ms) for PPG/IMU, 3s loop for DHT22/LCD, 18–20s loop for ThingSpeak upload, 5s Wi-Fi health

check.

- **Sensor drivers and fusion**
    - MAX30100 library for raw IR/RED; lightweight PPG processing to estimate BPM and SpO2.
    - Adafruit MPU6050 for acceleration; fall detection via spike–inactivity logic with left/right classification.
    - DHT22 library for temperature/humidity; PIR and SW‑420 read as digital inputs with debounce.

- **Communication**
    - Wi‑Fi STA mode with automatic reconnection.
    - HTTP POST to ThingSpeak using Write API key; per-request timeouts, retries, and response validation.
- **Local interface**

    ○ 16×2 I2C LCD shows T/H and flips between SpO2 and BPM every 18s. ○ Buzzer and servo actuation on critical events (BPM near zero or vibration detected).

- **Fault tolerance**
    - Non-blocking timing (no long delays), debounced vibration input, guarded network calls, and safe defaults for NaN readings.

- **Configuration**
    - Pin mapping, Wi‑Fi credentials, upload interval, thresholds, and LCD address defined as constants for easy modification.

4.3 Communication Protocol

- Network mode: ESP32 operates in Wi‑Fi Station (STA) mode on 2.4GHz, maintaining a persistent connection with auto‑reconnect.

- Cloud uplink: Sensor data are sent to ThingSpeak using HTTP/1.1 POST requests with application/x-www-form-urlencoded body and the channel Write API key. Each upload occurs every 16–20s to respect rate limits.

- Data schema: Fields map as F1–F8 in the request (Temperature, Humidity, BPM, SpO2, gAcc, PIR, Vibration, FallFlag) with an optional status string for context.

- Reliability: Per-request timeouts and limited retries are used; responses are validated by checking HTTP 200 and a positive entry ID.

- Local bus: I2C protocol (400kHz typical) on GPIO21/22 carries communication with MAX30100, MPU6050, and the 1602 I2C LCD.

5. Implementation

5.1 Sensor Data Acquisition

The ESP32 continuously reads all sensors using non-blocking timers. The MAX30100 provides raw IR/RED signals at high frequency; a lightweight algorithm estimates heart rate (BPM) from inter-beat intervals and computes a rough SpO2 value. The MPU6050 supplies acceleration data used to calculate magnitude (gAcc) and detect falls through spike-then-inactivity logic with left/right orientation. The DHT22 is sampled every 3s for temperature and humidity to reduce self-heating. The PIR sensor and SW-420 vibration module are polled as digital inputs with short debouncing to avoid false triggers. Each validated reading is stored in variables, displayed on the LCD, and queued for periodic cloud upload.

5.2 Mobile Application Features

- Real-time dashboard with auto-refresh for Temperature, Humidity, BPM, SpO2, motion, vibration, and fall status.
- Historical charts per parameter with time-range selection and CSV export. ● Manual controls to trigger buzzer test and servo door (with confirmation). ● Configurable alerts: push notifications for falls, near-zero BPM, low SpO2, high temperature, and vibration events.

5.3 Security Features

- Secure authentication with role-based access (patient, caregiver, admin) to restrict sensitive actions and data visibility.

- Encrypted communication for all device–cloud and app–cloud traffic to protect readings and commands.

- Protected write access using channel/API keys to prevent unauthorised data uploads or control actions.

6. Results

● The system reliably captured and displayed Temperature, Humidity, BPM, SpO2, motion, vibration, and fall status, with near real-time cloud updates every 16–20s. ● Automated safety actions (buzzer alert and servo-based door opening) triggered correctly during BPM-near-zero and vibration events, ensuring timely response. ● Data logging and visualisation on the dashboard operated smoothly, enabling quick trend observation and informed decision-making.

7. Challenges and Solutions

Challenge 1: Intermittent Wi-Fi connectivity
Solution: Added automatic reconnection with periodic health checks and retries for cloud uploads.
Challenge 2: Sensor noise and false triggers (vibration/PIR)
Solution: Implemented input debouncing, sensitivity tuning, and thresholding to stabilise readings.

Challenge 3: Power instability due to servo current spikes
Solution: Powered the servo from a separate 5V supply with common ground and bulk decoupling near the servo.

Challenge 4: Cloud rate-limit and missed updates
Solution: Scheduled uploads at 16–20s intervals, validated responses, and retried failed posts.

Challenge 5: MAX30100 inconsistent readings/LED off
Solution: Ensured 3.3V I2C wiring, explicit sensor initialisation (mode, LED current, sample rate), and tested with a minimal example.

8. Future Scope

● SMS alert system to the patient's guardian for critical events (fall detected, BPM near zero, low SpO2, abnormal temperature), with configurable contact numbers. ● Dedicated mobile application for caregivers to monitor real-time data, view history, and acknowledge alerts, with multilingual support (English/Bangla).

- Push notifications with escalation: if an alert is not acknowledged within a set time, notify secondary contacts or nearby clinic.
- Integration with popular platforms (Google Sheets/Drive) for automatic data archiving

and sharing with physicians.

- Voice assistant support (Google Assistant/Alexa) for hands-free status queries and basic control.

9. Conclusion

This project successfully demonstrates a practical, low-cost IoT solution for continuous patient monitoring and safety in a home setting. Using an ESP32 with MAX30100, MPU6050, DHT22, PIR, and SW-420 sensors, the system reliably captures vital signs and activity, displays key readings on a 16×2 I2C LCD, and uploads comprehensive data to the cloud at regular intervals for near real-time observation. Automated safety actions—buzzer alerts and servo-based door opening—operate as intended during critical events, supporting timely intervention. The design emphasises reliability, affordability, and ease of deployment, making it suitable for Bangladeshi households and caregivers, with clear scope for future enhancement through mobile apps, SMS alerts, and intelligent analytics.

10. References

1. https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf 2.
https://espressif-docs.readthedocs-hosted.com/projects/esp-idf/en/stable/hw-reference/ 3.
https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf
4. https://www.mouser.com/datasheet/2/891/esp-wroom-32_datasheet_en-1223836.pdf 5.
https://cdn.sparkfun.com/assets/e/b/6/b/0/esp32_datasheet_en-1223853.pdf 6.
https://components101.com/sensors/max30100-heart-rate-oxygen-pulse-sensor-pinout features-datasheet
7. https://techdelivers.com/max30100-pulse-oximeter-heart-rate-sensor-module-wearable-health
8. https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
9. https://joy-it.net/en/products/SEN-MPU6050
10. https://www.adafruit.com/product/385
11. https://cdn-learn.adafruit.com/downloads/pdf/dht.pdf
12. https://www.mathworks.com/help/thingspeak/writedata.html
13. https://docs.arduino.cc/nano-esp32