A white and red logo on a black background

AI-generated content may be incorrect.

**CSC2106**

**IoT Protocols and Networks**

**Project Interim Report**

|  |  |
| --- | --- |
| **SIT ID** | **Name** |
| 2300945 | Abdul Hakam Bin Hasbullah |
| 2301110 | Balavignesh Suresh Kumar |
| 2300946 | Chang Zhi Hao Nelson |
| 2303383 | Christian Angelo Lau Candelario |
| 2300938 | Mohamed Faizal Mohamed Fawaz |

# Section I: Literature Review

## Current Limitations in Fall Detection Systems

Existing fall detection systems have several flaws that limit their effectiveness, including false alarms, high power consumption, poor connectivity, and scalability issues. While commercial products like Apple Watch, Fitbit, and Garmin smartwatches include fall detection features, these devices trigger alerts via Wi-Fi, LTE, or Bluetooth, requiring constant internet access and frequent charging, making them impractical for continuous, long-term monitoring.

IoT-based solutions using Wi-Fi, Bluetooth, or Zigbee attempt to improve fall detection, but they suffer from:

1. **High Power Consumption**
   * Wi-Fi-based fall detection systems drain battery quickly due to continuous network association and data transmission (Rodrigues et al., 2020).
   * Apple Watch and Fitbit require daily charging due to their always-on Bluetooth and Wi-Fi connectivity.
   * Bluetooth-based devices, while more energy-efficient, need smartphone proximity, limiting independence for elderly users.
2. **Connectivity Issues**
   * Apple Watch and Fitbit rely on Wi-Fi or LTE for cloud-based alerts. If the network is down or unavailable, alerts cannot be sent, posing a safety risk (Khan et al., 2022).
   * IoT-based fall detection using Wi-Fi struggles in rural areas or places with poor signal coverage. Zigbee-based solutions also require a centralized hub, making them less flexible.
3. **Limited Scalability**
   * Wi-Fi and Bluetooth cannot effectively handle large-scale deployments in hospitals, elderly care homes, or smart home setups due to network congestion and interference (Eridani et al., 2021).

Given these issues, existing Wi-Fi and Bluetooth-based fall detection solutions are unreliable, power-hungry, and unsuitable for large-area monitoring. Our project overcomes these limitations using LoRa and ESP-NOW, which provide low power consumption, improved range, and better scalability.

## Why LoRa and ESP-NOW are Better for Fall Detection in our project?

Our system leverages ESP-NOW for local, low-latency communication and LoRa for long-range data transmission, combining the strengths of both protocols.

1. **Low Power Consumption**
   * ESP-NOW operates without constant network association, unlike Wi-Fi, allowing devices to wake up, send data, and return to sleep, significantly extending battery life (Espressif Systems, 2023).
   * LoRa’s ultra-low power operation allows sensor nodes to send data across kilometers while using minimal energy (LoRa Alliance, 2021).
2. **Reliable Communication in Any Environment**
   * ESP-NOW eliminates Wi-Fi dependency, enabling direct peer-to-peer communication between sensors. If a device detects a fall, it can immediately send an alert without internet access (Eridani et al., 2021).
   * LoRa extends communication range far beyond Wi-Fi’s limit, ensuring that fall alerts reach caregivers even in remote areas with weak network coverage (Petäjäjärvi et al., 2017).
3. **Scalability and Multi-Hop Network Support**
   * ESP-NOW enables efficient short-range device-to-device communication, making it ideal for smart homes and assisted living facilities (Espressif Systems, 2023).
   * LoRa supports multi-hop relay nodes, allowing coverage across entire buildings or communities with minimal infrastructure (LoRa Alliance, 2021).
   * Unlike Apple Watch and Fitbit, which only monitor individual users, our system scales to monitor multiple users simultaneously in hospitals, elderly care homes, or large residences.

## Conclusion

By integrating ESP-NOW for low-latency local networking and LoRa for long-range communication, our solution eliminates dependency on internet connectivity, extends monitoring range, and reduces battery drain. This ensures faster response times, improved reliability, and energy-efficient monitoring, making it a superior choice for IoT-based fall detection in large-scale and remote environments.

# Section II: Design

## Overall Architecture of the System

The system is designed to perform fall detection using a hybrid wireless network that combines a Wi‑Fi mesh network and long‑range multi-hop LoRa communication. It also incorporates a Flask server with MongoDB and a Web Application to store and display data to Caregivers and Flat Security Management personnels.

Devices and Sensors Used

* **M5StickCPlus** – Built‑in accelerometer and gyroscope (for fall detection).
* **LilyGO T3-S3** (with SX1280 LoRa Module) – Primarily used as a data relay/forwarding node and equipped with an OLED display for local status and message display.

Device Connection Outline

* **Local Mesh Networking** - M5Stick devices, which are worn or placed to monitor movement, collect sensor data (e.g., accelerometer/gyroscope data) and format it as JSON. These devices connect via a Wi‑Fi mesh network (using the PainlessMesh library) to a central “starting” LilyGO node.
* **Long‑Range Communication using LoRa** - The starting LilyGO receives the JSON data over the mesh and then forwards it using plain LoRa (via an SX1280 module). Optionally, a middleman node (another LilyGO) acts as a relay in a multi‑hop LoRa mesh, ensuring reliability and scalability.
* **Data Forwarding over Wi‑Fi to a Central Server** - The final destination LilyGO (which receives the LoRa transmission) then leverages its built‑in Wi‑Fi to send the collected JSON data to a Flask server.
* **Centralized Data Management & Web Interface** - The Flask server receives the data via HTTP and stores it in a MongoDB database. A web application is built on top of this backend to allow users to view real‑time data, historical logs, and receive alerts (e.g., fall detection notifications).

A diagram of a computer network

AI-generated content may be incorrect.

Data Collection and Communication Protocols

* **Data Collection** – Sensor data is captured on M5Stick devices and packaged in JSON (e.g. including fields like "node\_id": "3666078289", "status": "FALL DETECTED", "timestamp": "2025-03-04T12:34:56Z") Additional data fields can be captured depending on the border needs of this project.
* **Communication Protocols** –
  + **Wi‑Fi Mesh (PainlessMesh)** – M5Sticks and the starting LilyGO exchange data over a peer‑to‑peer mesh network.
  + **LoRa Communication** – The starting LilyGO forwards the JSON messages via the SX1280 module operating in LoRa mode. Optional relay nodes (middleman) can extend the coverage and for reliability in ensuring data is always sent.
  + **Wi‑Fi to Flask Server** – The final destination LilyGO uses its Wi‑Fi to send the received JSON to a Flask server. This can be done via HTTP POST requests.

## Data Storage and Processing

Storage

* **On the IOT Devices** – Both the M5StickCPlus and the LilyGO T3-S3 may temporarily display or log data locally.
* **On the Server** - The Flask server receives incoming data from the final destination LilyGO node. Data is then stored in a MongoDB database, which can handle large-scale, flexible, NoSQL storage of sensor events.

Processing

* The Flask server processes incoming data (e.g., verifying fall events, triggering alerts).
* Data can be further processed by background tasks (e.g., aggregating events, running analytics).

## User Interaction and Integration

* **Local Interaction** – Elderly can interact with the M5StickCPlus buttons to “turn off” fall detection message if they are able to recover themselves. Additionally, can interact with the button in the event a false flag occurs for the fall detection.
* **Remote Interaction** - A web application interfaces with the Flask server. Users can log into a dashboard to view real‑time data, historical logs, and receive notifications or alerts about fall detection events.
* **Integration with Third‑Party Applications** – The Flask server can expose APIs (e.g., REST endpoints) for integration with mobile apps, SMS gateways, or other notification services for the border idea of this project.

## Security Protocols

Device-Level Security

* The Wi‑Fi mesh network uses secure protocols (WPA2) for encrypted communication between devices.
* Data transmitted over LoRa can be encrypted at the application layer (e.g. using AES).

Server and Network Security

* Communication from the final destination LilyGO to the Flask server can be secured using HTTPS (TLS encryption).
* The Flask server and MongoDB should be secured with strong authentication, access control, and regular patching.

# References

Apple. (2022). Apple Watch Fall Detection. Retrieved from <https://www.apple.com/watch>

Eridani, D. et al. (2021). Comparative Study of ESP-NOW, Wi-Fi, and LoRa Protocols. IEEE.

Espressif Systems. (2023). ESP-NOW: Low-Power Wireless Communication Protocol. Retrieved from <https://docs.espressif.com>

Khan, M. et al. (2022). Challenges in IoT-Based Remote Health Monitoring. Sensors, 22(6), 1025.

LoRa Alliance. (2021). LoRaWAN Network Performance and Scalability. Retrieved from <https://www.lora-alliance.org>

Petäjäjärvi, J. et al. (2017). Evaluation of LoRa for Remote Health Monitoring. Wireless Communications, 23(5), 56-63.

Rodrigues, J. et al. (2020). Fall Detection Systems: A Systematic Review. IEEE Access, 8, 80034-80049.

Wu, F. et al. (2022). Machine Learning for Fall Detection in IoT Networks. Future Internet, 14(3), 78