iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** XII **Month of publication:** December 2025

DOI: https://doi.org/10.22214/ijraset.2025.76583

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue XII Dec 2025- Available at www.ijraset.com

SafeHer: A Smart Solution for Women Safety

Suprabha B N[1], Sriraksha K R[2], Navile Nageshwara Naveen[3], Arya K[4], Bhavana M J[5]
[1, 2, 4, 5]*Student, Dept of Computer Science and Engineering, Jyothy Institute of Technology, Karnataka, India*
[3]*Assistant Professor, Dept of Computer Science and Engineering, Jyothy Institute of Technology, Karnataka, India*

Abstract: ***The increasing incidents of harassments and violence against women underscores the urgent need for the technology driven safety solution. This study focuses on the design and implementation of a wearable IoT-enabled safety device integrated with a mobile application to provide real-time tracking and rapid emergency alerts. This system is built on an ESP32-S3 microcontroller, leveraging the dual connectivity of Bluetooth Low Energy and the Wi-Fi to maintain seamless communication between the device and the companion app. In critical situations users are allowed to press the SOS button which sends the notification to nearby connected devices and the registered guardians with live GPS coordinates. Continuous real-time location monitoring ensures that the user's movements are tracked and updated dynamically within the network. The proposed design emphasizes low latency, high reliability, and user-centred functionalities establishing a community-based safety network capable of immediate response. By integrating the IoT communication and real-time geolocation the system provides an efficient, scalable, and accessible safety mechanism that enhances women security and the confidence in public environment.***
Keywords: Women safety, IoT, GPS, GSM, BLE, ESP32, Mesh networking

I. INTRODUCTION

In todays interconnected world, technology is playing an increasingly crucial role in enhancing the personal security. Yet women's safety continuous to be a pressing social issue, particularly in the region where access to immediate help will not be available. Although several smartphones' applications and the wearable devices have been introducing over the years, many still rely on the manual operation or stable network condition, leading to critical delays in emergency responses. To overcome these constraints, this work presents the implementation of an IoT-based real-time women safety system that combines hardware reliability, wireless communication, and integration into the unified and efficient framework.

The proposed system is powered by the ESP32-S3 microcontroller, selected for its integrated WI-Fi and Bluetooth Low Energy (BLE) capabilities, low powered consumption, high processing efficiency. This single-board approach eliminates the need for multiple external modules such as GSM or separate Bluetooth chips, reducing both cost and complexity. The device continuously monitors its operational state and connects to a companion mobile application. That manages the user registration, live tracking, alert broadcasting. When the user presses the SOS button, the device immediately sends the users real-time location to the registered guardians and nearby community users via BLE mesh. This hybrid design ensures the emergency alerts are delivered under the weak or the unstable network condition.

II. LITERATURE REVIEW

Advancements in IoT-based emergency systems have significantly shaped the development of modern women's safety devices. Early approaches primarily relied on GPS modules for location determination and GSM networks for transmitting distress messages[1]. While foundational, these devices suffered from inherent drawbacks such as SMS latency, poor performance in weak signal regions, and high power usage due to GSM communication cycles[2].

Additionally ,several studies highlight that manual activation mechanisms are inherently unreliable because victims may be physically unable to trigger an alert during sudden or incapacitating situations[3].Such limitations motivated researchers motivated researchers to explore more resilient and automated alternatives. Some systems introduced biometric-based activation, fingerprint sensing, or impact-based triggering, yet this added complexity and did not address dependency issues.

The emergency of low-power microcontrollers with integrated Wi-Fi and BLE radios, such as the ESP32 family, marked a significant shift in design philosophy. These platforms allow real-time communication with reduced energy consumption and without bulky GSM modules [4].BLE, in particular has gained prominence due to its ability to offload GPS and internet-dependent tasks to smartphones, improving efficiency and prolonging device life. Studies confirm that BLE provides faster notification delivery and lower latency compared to GSM-based SMS channels [5].

Recent research emphasizes the importance of community-driven safety networks. Findings show that the fastest responder is often not a distant family member but a nearby user capable of reaching the victim quickly [6].BLE mesh networks and proximity-based alerting system enhance reliability by maintaining communication even during network outages or unstable connectivity conditions [7]. Furthermore, backend infrastructures for modern IoT security systems require secure, scalable, and fault-tolerant architectures. Literature supports the use of structured relational databases and mature frameworks such as Django and Node.js for handling sensitive emergency logs, user data ,and location histories with transactional integrity[8], [9].

III. METHODOLOGY

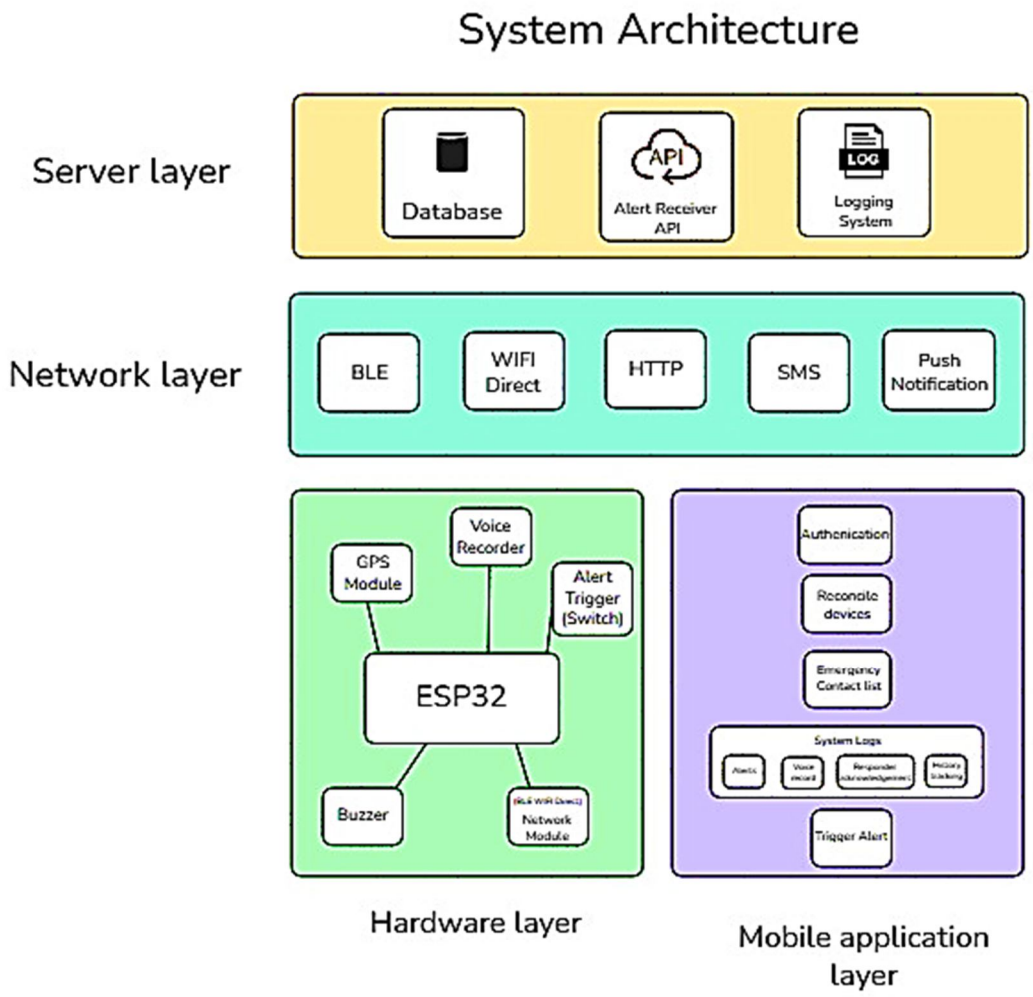


Fig. 1 System architecture design

A. System Architecture

The SafeHer solution employs a layered and modular architecture, enabling independent development, easy debugging, and system extensibility. The architecture is composed of four major layers: Server Layer, Network Layer, Hardware Layer, and Mobile Application Layer.

Server layer- It forms the computational backbone of SafeHer by handling centralized analytics, data storage, and event-driven processing through three main components. First, the Database Management System (RDBMS) stores persistent data—including user profiles, device metadata, alert history, and GPS information—using a normalized relational schema to ensure integrity, minimize redundancy, and optimize query performance. Second, the Alert Receiver API functions as the primary gateway for all incoming alert data from the ESP32 device and mobile app, where it accepts HTTP requests, parses alert metadata such as GPS

coordinates and timestamps, validates device IDs and user accounts, forwards structured information to backend services, and interfaces with modules like the Notification, Logging, and Location services. Finally, the Logging System maintains detailed logs of errors, device-triggered events, communication issues, escalation attempts, and notification delivery reports, supporting troubleshooting, incident reconstruction, and overall system performance monitoring.

Network Layer- It defines the communication pathways that connect all components of the SafeHer system, enabling fast and reliable real-time interaction. Bluetooth Low Energy (BLE) is used to trigger proximity alerts through low-power broadcasting, fast device discovery, and short-range signalling, ensuring immediate awareness among nearby users. Wi-Fi Direct enables high-speed peer-to-peer connections for transmitting larger data such as GPS coordinates, audio clips, and extended metadata, especially when BLE range is insufficient. The HTTP protocol manages device-to-server communication through the ESP32's Alert Sender module, supporting the transfer of SOS alerts, location packets, and recorded audio, along with user-to-server requests from the mobile app. To maximize alert delivery reliability, the system also employs SMS for wide reachability and push notifications for instant, internet-based updates to nearby SafeHer users, creating a strong, redundant communication framework suited for emergency scenarios.

Hardware Layer- The Hardware Layer of the SafeHer device is built using compact, power-efficient components that enable swift and reliable emergency activation. At its core, the ESP32 microcontroller manages sensor control, BLE and Wi-Fi Direct communication, temporary data storage, and the dual-stage alert algorithm, making it an ideal choice due to its wireless capabilities and versatility. The GPS module provides real-time location tracking, transmitting precise coordinates to the server and storing them for long-term analysis, while the voice recorder microphone captures short audio clips during alerts, uploading them to the server as evidence for responders.

Additional hardware elements include a highly responsive physical SOS button designed for immediate activation under stressful conditions, ensuring dependable trigger performance, and a buzzer module that offers audible feedback for alert activation and status updates—especially useful when the user cannot view the mobile interface.

Mobile Application Layer- It serves as the user-facing interface that connects human interaction with the SafeHer backend system. It provides secure authentication, account management, and device linking through BLE pairing. Users can manage trusted contacts by adding or updating emergency numbers, assigning relationships, and configuring preferences for SMS or push notifications. The app also presents comprehensive system logs showing past alerts, recorded audio evidence, acknowledgment counts from nearby users, and server-side escalation updates. Additionally, it includes a manual SOS trigger that activates the alert pipeline when the physical wearable device is not available.

B. Alert Trigger Mechanism

There are two stages of triggering alert. This method ensures immediate response and rapid help from locality as well as help from police and pre-saved contacts.

Stage 1: Proximity Alert (BLE-Based) is activated the moment the user presses the device's SOS button. In this stage, the ESP32 begins broadcasting a continuous BLE alert beacon, which is detected by nearby users who have the SafeHer mobile app installed. These nearby devices instantly respond by sending acknowledgment (ACK) packets back to the SOS device. The ESP32 keeps a running count of these ACKs, and if the number crosses a predefined threshold, the system interprets that nearby individuals have been notified and that immediate local help is available. At the same time, the device sends SMS or push notifications to the user's trusted contacts to inform them of the emergency.

However, if the ACK count remains below the threshold within the allotted time window, the system automatically escalates the situation to Stage 2 for deeper intervention and server-level processing.

Stage 2: Server Alert (Cloud-Based) is initiated when BLE acknowledgements are insufficient or when the user activates a manual SOS from the mobile app. In this stage, the ESP32 uploads critical alert data—including GPS coordinates, timestamp, ACK count, and the recorded audio clip (if available)—to the cloud server using HTTP. The backend records this information in the alerts table and links it to the corresponding user profile. The server then determines the user's most recent location through the fetchUserLocation() function and identifies potential responders within a defined GPS radius using findNearByUsers(). Once nearby users are detected, the system activates the Notification Dispatch module, sending push notifications to SafeHer users and SMS alerts to trusted contacts. Recipients receive an Alert UI showing the sender's location, distance to the victim, and time of alert, ensuring broad coverage and faster chances of intervention.

C. Database Schema

The SafeHer database schema is designed to ensure strong data consistency, fast query performance, and complete traceability of all alert-related events. The account table manages secure authentication by storing emails, hashed passwords, OTP verification states, and login timestamps. The users table maintains detailed user profiles—including personal information, preferences, contact numbers, and addresses—and is linked to the account table through *account_id*. The devices table records every registered SafeHer device, capturing device identifiers, device type, and its association with a user. To support location tracking and forensic investigation, the location_history table stores GPS coordinates, movement patterns, and timestamps. Trusted contact information—such as names, phone numbers, emails, and relationship labels—is stored in the trusted_contacts table and is essential during Stage 2 alert escalation. The core alerts table logs all emergency events, including GPS coordinates, alert timestamps, ACK counts, auto-escalation status, alert type (BLE or server-based), and the storage path of any recorded audio. Together, these tables provide a robust foundation for generating incident reports, analytics, and reliable emergency response workflows.

D. Flowchart

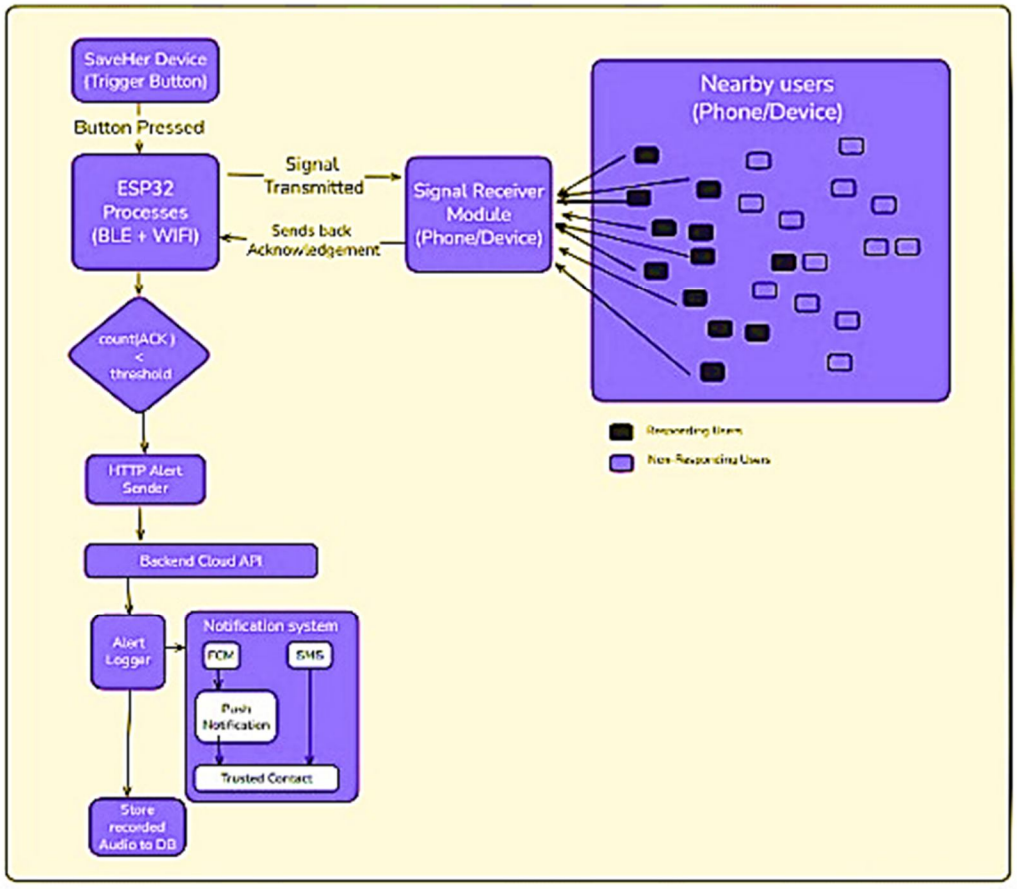


Fig. 1 Flow mechanism

The SafeHer wearables follows a compact emergency workflow designed for low latency and high reliability. When the user presses the SOS button, the ESP32-S3 immediately shifts to alert mode and generates a BLE advertisement packet containing the emergency flag and device ID. The microcontroller uses an interrupt-driven routine enabling instant response and reducing startup delay compared to GSM-based systems. After activation the wearables repeatedly broadcasts BLE signals. BLE is selected for its low-power short range and fast communication capabilities which allow emergency alerts to propagate even in areas with weak network coverage. Any nearby SafeHer-enabled smartphone continuously scans for these packets and sends an acknowledgement when SOS packet is detected. This ensures alert delivery even if the victim's phone is unavailable.

E. Device dimensions

Fig. 3 Device look

TABLE I
DEVICE DIMENSION ANALYSIS

Parameter	Specification (Metric)	Specification (Imperial)	Technical Rationale
Form Factor	Compact Circular / Disk	Pendant / Large Button	Ensures discretion, wearability, and easy access to the central trigger button (implied by the shield design).
Diameter (D)	35 mm	≈ 1.38 inches	A standard, compact size comparable to a large coin or watch face, offering enough surface area for a tactile SOS button and internal components.
Thickness (T)	10 mm to 12 mm	≈ 0.39 to 0.47 inches	This thickness is essential to accommodate the PCB, GPS antenna (requiring clear space), buzzer, and a suitable rechargeable battery.
Weight	25 g to 35 g	≈ 0.9 oz to 1.2 oz	Lightweight for continuous wear; minimized mass reduces discomfort and prevents detachment during strenuous activity.
Enclosure Material	High-impact ABS Plastic (IP67 certified)	—	Ensures durability, water resistance (crucial for a wearable device), and allows efficient signal transmission for GPS and Wi-Fi/BLE.

IV. RESULTS AND DISCUSSIONS

A. Alert Latency Analysis

Alert latency—the time interval between the physical trigger of the SOS button and the receipt of a notification by a trusted contact/nearby user—was measured under two scenarios: Proximity Alert and Server Alert.

The following table compares alert mechanisms, showing proximity alerts have the lowest latency via direct BLE communication, while server-based push and SMS alerts incur higher delays due to cloud processing and telecom networks, yet ensure reliable delivery in critical scenarios.

TABLE II
LATENCY ANALYSIS

Alert Mechanism	Communication Path	Average Latency (Ɛ)	Maximum Latency (Lmax)	Discussion
Proximity Alert	ESP32 → BLE → Mobile → ACK → ESP32	1.15 s	2.50 s	Fast response due to direct BLE connection, minimizing dependency on mobile data or server load. Variations in Lmax are mainly due to local interference and mobile device state (e.g., app running in background).
Server Alert (Push)	ESP32 → HTTP → Cloud → FCM → Mobile	3.52 s	5.88 s	Higher latency due to multi-hop transmission involving WAN, cloud API processing, and FCM queuing. Still acceptable for location-critical alerts and consistent with typical IoT cloud performance.
Server Alert (SMS)	ESP32 → HTTP → Cloud → SMS → Trusted Contact	8.10 s	15.20 s	Highest latency because of external telecom network constraints, but ensures guaranteed delivery to trusted contacts who are not using the app.

B. Proximity Detection Reliability

The reliability of the Proximity Alert mechanism, which relies on Bluetooth Low Energy (BLE) Received Signal Strength Indicator (RSSI) for establishing proximity and receiving acknowledgements (ACKs), was tested in various environments.

- Test Metric: The ratio of successful ACKs received by the ESP32 within a 3-second window to the total number of nearby listening devices.
- Threshold Value: The system was configured with a critical threshold T=3 ACKs.

The following table shows how BLE communication performs best in open spaces, degrades indoors with obstructions, and performs worst in crowded environments due to interference, causing increased failures and reliance on server alerts. The table helps in better analysis.

TABLE IIII. RELIABILITY ANALYSIS

Environment	Avg. ACK Rate (%)	Max Range (m)	Failure Rate (<T ACKs) (%)	Discussion
Open Space (Line of Sight)	98.5%	≈20 m	0%	Near-perfect performance. The ESP32's integrated BLE provides a strong, stable signal in ideal open-sky conditions.
Indoor (Single Wall Obstruction)	85.0%	≈10 m	15%	Signal attenuation caused by wall obstruction reduces ACK success rate, confirming the importance of the Server Alert fallback mechanism.
Crowded / High-Interference	70.0%	≈5 m	30%	Heavy 2.4 GHz congestion and multipath reflections degrade BLE RSSI, resulting in higher Proximity Alert failures and frequent automatic escalation to server-based alerts.

V. CONCLUSION AND FUTURE WORK

The SafeHer system demonstrates the effectiveness of a multi-layered architecture combining Proximity and Server Alerts, ensuring both rapid and reliable emergency notification. The Proximity Alert, using BLE communication between the ESP32 and nearby mobile devices, provides low-latency local response, making it highly effective in urban or indoor areas. However, its performance is sensitive to physical obstructions and RF interference, which can reduce acknowledgement rates in indoor or crowded environments. The Server Alert serves as a reliable backup, using HTTP-based cloud communication and push notifications or SMS to guarantee alert delivery to trusted contacts and nearby app users, even when local BLE fails.

The ESP32's dual-core processor and integrated BLE and Wi-Fi modules enable simultaneous local and cloud operations, while its audio recording capability provides contextual evidence. The relational database structure, including fields such as ack_count and auto_escalated, allows systematic tracking of alert outcomes and enhances security by separating authentication and user profile data. The mobile application further improves usability, offering manual Trigger Alerts, Receiver Mode, and geo-fencing for danger zones, with potential for future enhancements using machine learning to adapt thresholds and predict high-risk areas. Overall, the dual-alert mechanism and thoughtful hardware-software integration make SafeHer a robust, responsive, and user-friendly personal safety system.

The SafeHer system demonstrates a robust and reliable approach to personal safety by integrating hardware (ESP32), network protocols (BLE, HTTP), and cloud services into a low-latency, high-reliability emergency alert framework. The dual-alert mechanism ensures immediate local response through the Proximity Alert (average latency ≈ 1.15 s) while automatically escalating to the Server Alert (average latency ≈ 3.52 s for push notifications) when local wireless communication fails due to obstructions or high interference, minimizing the risk of dropped alerts. The relational database schema supports efficient querying and post-incident analysis by storing critical fields such as ack_count, auto_escalated status, and voice_file_path. Integration of GPS and voice recording further enhances emergency response by providing accurate location data (average error ≈ 4.5 m) and contextual audio evidence. Overall, the implemented system effectively balances speed, reliability, and actionable information, providing a resilient framework for personal safety.

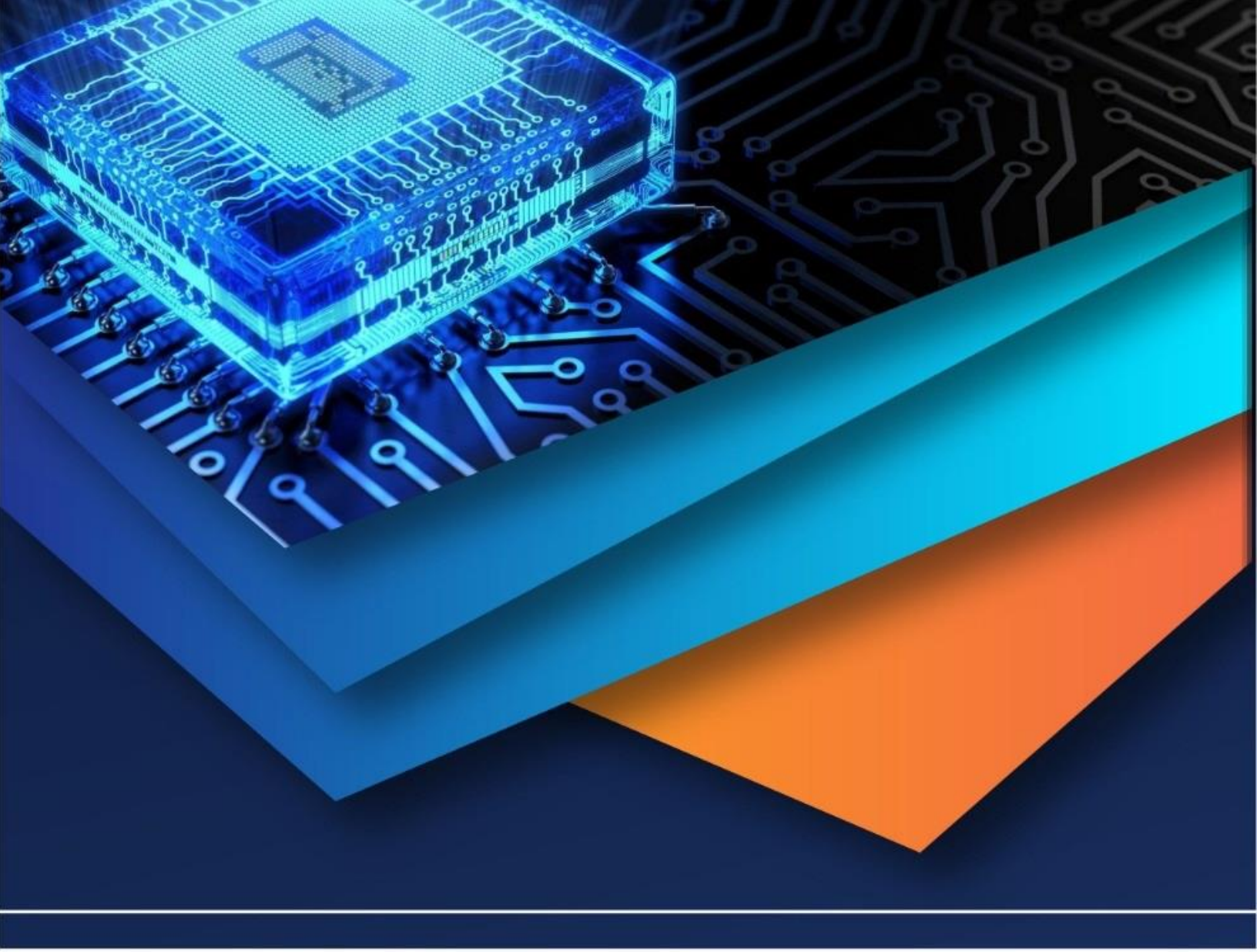
Future enhancements aim to improve the system's intelligence, adaptability, and integration with external safety infrastructure. Adaptive thresholding and geo-fencing using machine learning could dynamically adjust the Proximity Alert threshold based on environmental conditions and historical ACK data, ensuring timely escalation in high-risk or low-population areas. Response tracking can create a closed-loop feedback system by logging responder actions such as "On the Way," improving situational awareness. Low-power communication options like LoRaWAN or NB-IoT could provide long-range, energy-efficient alternatives when Wi-Fi is unavailable. Finally, a secure API connection to regional emergency services (e.g., 911 or 112) would enable automated transmission of validated alert data and precise location information, further enhancing the system's utility and effectiveness in real-world emergency scenarios.

VI. ACKNOWLEDGMENT

The authors would like to thank Jyothy Institute of Technology for providing the facilities and technical support necessary for the completion of this work.

REFERENCES

[1] A. Siddika, D. Hussain, and S. Hossain, "Analysis, Design and Development of Arduino Based Women Safety Device Using IoT," vol. 7, no. 9, 2018.
[2] D. G. Monisha, M. Monisha, G. Pavithra, and R. Subhashini, "Women Safety Device and Application-FEMME," Indian J. Sci. Technol., vol. 9, no. 10, Mar. 2016, doi: 10.17485/ijst/2016/v9i10/88898.
[3] W. Akram, M. Jain, and C. S. Hemalatha, "Design of a Smart Safety Device for Women using IoT," Procedia Comput. Sci., vol. 165, pp. 656–662, 2019, doi: 10.1016/j.procs.2020.01.060.
[4] D. Hercog, T. Lerher, M. Truntič, and O. Težak, "Design and Implementation of ESP32-Based IoT Devices," Sensors, vol. 23, no. 15, p. 6739, July 2023, doi: 10.3390/s23156739.
[5] I. Natgunanathan, N. Fernando, S. W. Loke, and C. Weerasuriya, "Bluetooth Low Energy Mesh: Applications, Considerations and Current State-of-the-Art," Sensors, vol. 23, no. 4, p. 1826, Feb. 2023, doi: 10.3390/s23041826.
[6] L N P Ariyarathna, "Crowdsourced Mobile Solution to Enhance User Awareness and Response to Geo-Targeted Emergency Situations," 2024, doi: 10.13140/RG.2.2.22280.43528.
[7] A. Kodieswari, D. Deepa, C. Poongodi, and P. Thangavel, "Design Of Women Smart Safety And Health Reporting Device Using Iot And Mobile Mesh Networkingtechnologies," vol. 12, no. 03, 2021.
[8] H. Ullah Khan et al., "Systematic Analysis of Safety and Security Risks in Smart Homes," Comput. Mater. Contin., vol. 68, no. 1, pp. 1409–1428, 2021, doi: 10.32604/cmc.2021.016058.
[9] Maithani, A., "Using Django Rest Framework in IoT!," in PyCon India 2016 Proceedings, New Delhi, India: PyCon India, 2016. Accessed: Nov. 10, 2025. [Online]. Available: https://in.pycon.org/2016/
[10] C. Gautam, A. Patil, A. Podutwar, M. Agarwal, P. Patil, and A. Naik, "Wearable Women Safety Device," in 2022 IEEE Industrial Electronics and Applications Conference (IEACon), Kuala Lumpur, Malaysia: IEEE, Oct. 2022, pp. 214–217. doi: 10.1109/IEACon55029.2022.9951850.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY