WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

HUMAN SAFETY ALERT SYSTEM (HASS) A Mobile-Based Personal Safety and Emergency Assistance System

Adarsh R. Mhaske[1], Devendra K. Botkewar[2], Atharva R. Meshram[3], Tanushree S. Kawadkar[4], Vaishnavi R. Madne[5], Prof. Yogesh B. Malode*[6]

Department of Information Technology, Priyadarshini Bhagwati College Of Engineering, Nagpur 440009, Maharashtra, India
*Corresponding Author (Mentor)

ABSTRACT :

Personal safety applications support emergency communication by enabling users to send alerts, share location information, and coordinate assistance. Previous research has explored mobile alerting and real-time location monitoring. This paper presents the proposed Human Safety Alert System (HASS), a mobile-based application that integrates emergency alerts, location sharing, and trusted-contact coordination. Its main objective is to provide a structured process for initiating alerts through a five-second SOS countdown, emergency category selection for medical emergencies, accidents, and harassment, prioritized notifications to primary and secondary trusted contacts, and an escalation workflow. The proposed methodology includes requirement analysis, system design, application development, service integration, and functional and security testing. React Native, Expo, Supabase, and PostgreSQL can support application development and data management. Mapping services, battery and network information, and potential AI assistance through the Gemini API may further support emergency response. HASS is intended to complement India's Emergency Response Support System (ERSS), accessible through emergency number 112, rather than replace official emergency services. Its reliability, security, and practical usefulness must be established through systematic testing and evaluation.

Keywords: Human Safety Alert System, HASS, Personal Safety, Emergency SOS, GPS, Location Sharing, Trusted Contacts, Safety Check-In, Mobile Application, Emergency Assistance.

I. INTRODUCTION

Personal safety is an important concern during medical emergencies, road accidents, harassment, and other potentially dangerous situations. Such incidents may occur unexpectedly and require individuals to communicate their situation and seek assistance promptly. During an emergency, providing accurate location information and notifying trusted individuals can help coordinate an appropriate response. However, communicating the nature of an incident, sharing a location, or contacting multiple people may be difficult under stressful conditions. Mobile applications offer a way to support emergency communication by combining alert mechanisms with location-related information and contact coordination [2][3]. These capabilities provide the motivation for developing mobile-based systems that aim to make emergency information easier to share.

In India, the Emergency Response Support System (ERSS) provides a unified emergency response mechanism through the national emergency number 112 [1]. It enables citizens to request assistance for emergencies involving services such as police, fire and rescue, and health services. The official ERSS framework supports multiple channels for reporting emergencies, including telephone calls and digital alert mechanisms [1]. Personal safety applications can complement such services by providing additional tools for communicating with family members, trusted contacts, or other designated recipients. However, these applications should not be considered replacements for official emergency services. Their role is to provide supplementary communication and coordination features that may be useful during an emergency.

The Human Safety Alert System (HASS) is proposed as a mobile-based personal safety application that integrates emergency alerts, location sharing, and trusted-contact coordination within a single platform. The main objective is to provide a structured process through which users can initiate an emergency alert and communicate relevant information. The proposed system includes an SOS button with a five-second countdown, allowing a short interval before alert initiation. It also provides emergency category selection for medical emergencies, accidents, and harassment. If the user does not select a category, the system is designed to prepare a general alert. These features aim to simplify the emergency initiation process and help organize the information associated with an alert.

An important component of HASS is its proposed trusted-contact notification and escalation workflow. Contacts are divided into primary and secondary groups so that notifications can follow a predefined priority sequence. This arrangement is intended to organize communication and provide designated individuals with an opportunity to respond to an emergency. The system also proposes an escalation mechanism for situations in which assistance is not received during the initial notification stages. The usefulness of this workflow depends on the accuracy of contact information, successful notification delivery, network availability, and correct implementation of the escalation rules. Therefore, the intended sequence and timing must be verified through functional testing rather than assumed to operate successfully in every situation.

HASS also proposes location sharing, identification of nearby hospitals and police stations, and a helper assistance mechanism. Mapping and location services can support the display of geographic information and help users identify nearby facilities [7][8][9]. Sharing available location information may help trusted contacts or helpers understand where assistance is required. The helper mechanism is intended to support coordination between a person requesting assistance and an individual willing to help. In addition, device battery level and network connectivity information may provide useful context about the device during an emergency [12][13]. The availability and accuracy of these capabilities depend on device permissions, location-service availability, connectivity, and the successful integration of the relevant services.

The proposed application can be developed using technologies suited to mobile interfaces, authentication, data management, and location-related functionality. React Native and Expo provide a foundation for mobile application development [10][11]. Supabase offers backend services, including authentication and database integration, while PostgreSQL supports structured data storage [15][16][18]. Mapping services can support location-related functions and nearby facility searches [7][8][9]. The Gemini API may also be explored for supplementary AI-based safety guidance, subject to appropriate implementation and validation [19]. The final technology configuration should reflect the actual system architecture, and only components that are implemented or formally proposed should be described as part of the project.

Privacy, security, and operational reliability are important considerations because a personal safety application may handle sensitive information, including location details, account information, and trusted-contact records. Appropriate access controls and secure data-handling practices should be considered during development, with relevant security guidance provided by the OWASP Mobile Application Security Verification Standard [20].

The system must also account for practical limitations such as weak network connectivity, unavailable location services, low battery levels, and delayed or unsuccessful notifications. These conditions may affect the availability of emergency information and the delivery of alerts. Consequently, the proposed features require appropriate verification, and their presence in the design alone should not be interpreted as proof of reliability or security.

The main objective of HASS is to provide an integrated approach to personal safety by combining SOS activation, emergency category selection, prioritized trusted-contact notifications, location sharing, and access to nearby emergency facilities. The system is intended to complement India's ERSS emergency number 112 rather than replace official emergency response services [1]. Its development and evaluation should consider functional correctness, notification delivery, location accuracy, usability, privacy, and security. Testing under different device and network conditions can help identify limitations and areas requiring improvement. The overall contribution of the proposed system is its integration of emergency communication and assistance-related functions within one mobile application, with its practical effectiveness to be determined through implementation and systematic evaluation.

II. PROBLEM STATEMENT

During emergencies, people may need to communicate their situation, share their location, and notify trusted contacts [2][3]. Managing emergency notifications and providing location information can be challenging when permissions, network connectivity, or location services are unavailable [7][11][13].

The proposed Human Safety Alert System (HASS) aims to address these challenges through an SOS mechanism, prioritized trusted-contact notifications, and location sharing. Since the application may handle personal and location data, appropriate security controls and testing are also necessary [20].

Problem Statement: To develop a mobile-based system that integrates emergency alerts, location sharing, and trusted-contact coordination while evaluating its functionality, security, and limitations.

III. LITERATURE REVIEW

The literature review examines existing emergency alert approaches, mobile location technologies, application development frameworks, backend services, artificial intelligence tools, and mobile application security standards relevant to the proposed Human Safety Alert System (HASS). The selected references include research publications, official service information, and technical documentation related to these areas.

3.1 Existing Emergency Alert and Personal Safety Systems

India's Emergency Response Support System (ERSS) provides access to emergency assistance through the 112 emergency number [1]. HASS is proposed as a complementary application for emergency alerts and trusted-contact coordination [1].

Mowbray et al. reviewed mobile alerting for public emergencies and factors influencing people's responses to emergency messages [2]. Gowda et al. described an Android application combining real-time location monitoring with emergency alerts, providing a relevant basis for the proposed HASS workflow [3].

A 2025 conference publication explored an IoT-based wearable safety device using GSM, GPS, and ESP32-CAM, representing a hardware-oriented approach to personal safety [4]. The 2026 SafeGuard publication identifies offline-first operation, multi-modal detection, and face authentication as its main areas of focus [5]. Indalkar et al. discussed mobile applications for community safety involving emergency alerts, live tracking, and monitoring [6].

3.2 Location Monitoring, Maps, and Nearby Emergency Services

Google's documentation describes location-related functionality and map integration for Android applications [7][8]. The Google Places API Nearby Search supports searching for nearby places using geographical information and search criteria [9].

Expo provides location-related functionality for mobile applications, which is relevant to the proposed location-sharing feature of HASS [11]. These technologies can support location display and nearby facility searches, although actual results depend on permissions, implementation, and available data [7][9][11].

3.3 Mobile Application Development Technologies

React Native provides a framework for mobile application development, while Expo documents additional device capabilities [10][11]. Expo's Battery and Network APIs provide access to supported battery and network information, which may be used by relevant HASS features [12][13]. TypeScript provides language features that can help structure application code and define data types [14]. These technologies provide a potential development foundation for the HASS interface and device-related functions, but system reliability must be established through testing.

3.4 Backend Services, Authentication, and Data Management

Supabase provides backend capabilities for application development, and its authentication services can support user account management [15][16]. Supabase Realtime provides functionality for receiving data changes, which may support updates between connected application components when properly configured [17].

PostgreSQL provides a relational database system suitable for organizing structured records according to the application's data model [18]. These technologies can support the proposed HASS backend, authentication, and emergency data management, depending on the final implementation [15][16][17][18].

3.5 Artificial Intelligence and Mobile Application Security

The Gemini API provides a technical option for integrating supported generative AI capabilities into applications [19]. HASS may use this capability for supplementary safety guidance, subject to implementation, validation, and appropriate safety controls [19].

The OWASP Mobile Application Security Verification Standard (MASVS) provides a framework for assessing mobile application security [20]. It is relevant to evaluating the protection of user information, authentication, data storage, and communication within HASS [20].

3.6 Summary of the Literature and Research Gap

The selected references cover emergency response, mobile alerting, location monitoring, wearable safety devices, application development, backend services, AI, and mobile security [1][2][3][4][5][6][7][8][9][10][14][15][16][19][20].

The proposed HASS aims to combine SOS activation, emergency category selection, location sharing, and prioritized trusted-contact notifications within one mobile application. However, the selected references do not establish that this complete combination has been successfully implemented and evaluated in HASS.

Therefore, the project focuses on implementing and testing the integrated workflow, including notification handling, location availability, connectivity limitations, and security. Its practical effectiveness and reliability must be determined through testing rather than assumed in advance.

IV. RESEARCH GAP, AIM AND OBJECTIVES

4.1 Research Gap

Previous studies have explored mobile emergency alerts, real-time location monitoring, and mobile-based safety applications [2][3][6]. Existing technical resources also provide tools for location services, mapping, backend development, and mobile application security [7][8][15][20].

However, the selected references do not establish that the complete set of features proposed for HASS has been integrated and evaluated together in one application. Therefore, this project focuses on combining SOS activation, emergency category selection, location sharing, and prioritized notifications to trusted contacts within a single mobile-based system.

The project will also consider permission handling, connectivity limitations, and security during testing [11][13][20]. The effectiveness and reliability of the proposed system must be determined through implementation and evaluation.

4.2 Aim

The main aim of this project is to develop a mobile-based Human Safety Alert System (HASS) that supports emergency alert generation, location sharing, and coordinated notifications to trusted contacts through a structured workflow.

4.3 Objectives

The main objectives of HASS are:

1. To provide an SOS button with a five-second countdown.

2. To support medical, accident, and harassment emergency categories.
3. To manage primary and secondary trusted contacts.
4. To implement a prioritized emergency notification workflow.
5. To support location sharing and display nearby emergency-related facilities using mapping services [7][8][9].
6. To implement suitable authentication and backend data management [15][16][18].
7. To evaluate system functionality, security, and limitations through testing [20].

4.4 Expected Outcome

The expected outcome is a mobile application prototype that integrates the implemented HASS features into a structured emergency assistance workflow. Its actual reliability and effectiveness will be assessed through testing.

Challenge	Proposed Solution
User cannot manually call for help	SOS button with countdown
Emergency type is unclear	Emergency category selection
Contacts may miss alerts	Prioritized contact notification
User location is unknown	Location sharing with permission
Network connection fails	Handle connection failure and alert status
Personal data may be exposed	Authentication and access controls

Table 1: Challenges and proposed solutions in HASS

V. PROPOSED METHODOLOGY AND REQUIREMENTS

5.1 Proposed Methodology

The proposed Human Safety Alert System (HASS) follows a structured development approach consisting of requirement analysis, system design, implementation, and testing.

1. Requirement Analysis: The system requirements are identified for SOS activation, emergency categories, trusted-contact management, location sharing, and emergency notifications.
2. System Design: The application workflow and architecture are planned to connect the mobile interface, backend services, database, and supported external services.
3. Application Development: The mobile application can be developed using React Native and Expo, with TypeScript for application code [10][11][14]. Supabase and PostgreSQL can support authentication and structured data management [15][16][18].
4. Feature Integration: The proposed features include the five-second SOS countdown, emergency category selection, prioritized trusted-contact notifications, and location information. Mapping and nearby-place functionality can use suitable Google Maps and Places services [7][8][9].
5. Testing and Evaluation: The implemented features should be tested for functional correctness, permission handling, connectivity limitations, notification behaviour, and security requirements [13][20]. Performance results should be reported only after testing has been completed.

5.2 Functional Requirements

The main functional requirements of the proposed HASS are:

1. The system shall provide an SOS button with a five-second countdown.
2. The system shall support medical, accident, and harassment emergency categories.
3. The system shall generate a general alert if no emergency category is selected.
4. The system shall support primary and secondary trusted-contact management.
5. The system shall follow configured priorities when notifying trusted contacts.
6. The system shall support location sharing and display relevant location information when permissions and services are available.
7. The system shall support viewing nearby emergency-related facilities through mapping services [8][9].
8. The system shall provide suitable authentication and storage for application data [15][16][18].

5.3 Non-Functional Requirements

The proposed system should meet the following non-functional requirements:

1. Usability: Emergency functions should be easy to identify and operate.
2. Reliability: The application should handle unavailable location data and notification failures appropriately.
3. Security: User accounts, contact details, and location information should be protected using suitable security controls [20].
4. Compatibility: The application should be tested on supported devices and operating-system versions.
5. Connectivity Handling: The application should respond appropriately to network interruptions and failed service requests [13].
6. Maintainability: The application code and data structures should be organized to support future updates [14].

5.4 Requirements Summary

The proposed system requires a compatible mobile device, location permissions, network access for online services, a backend for supported application functions, and access to configured mapping or notification services. Actual requirements depend on the final implementation and the external services integrated into HASS.

VI. SYSTEM ARCHITECTURE AND DESIGN

6.1 System Architecture

The proposed Human Safety Alert System (HASS) follows a layered architecture consisting of a mobile application, backend, database, and external services. React Native, Expo, and TypeScript can support mobile development, while Supabase and PostgreSQL can manage authentication and application data [10][11][14][15][16][18].

The application manages SOS activation, emergency categories, and trusted contacts. Mapping and location features can use Google Maps and Places services [7][8][9].

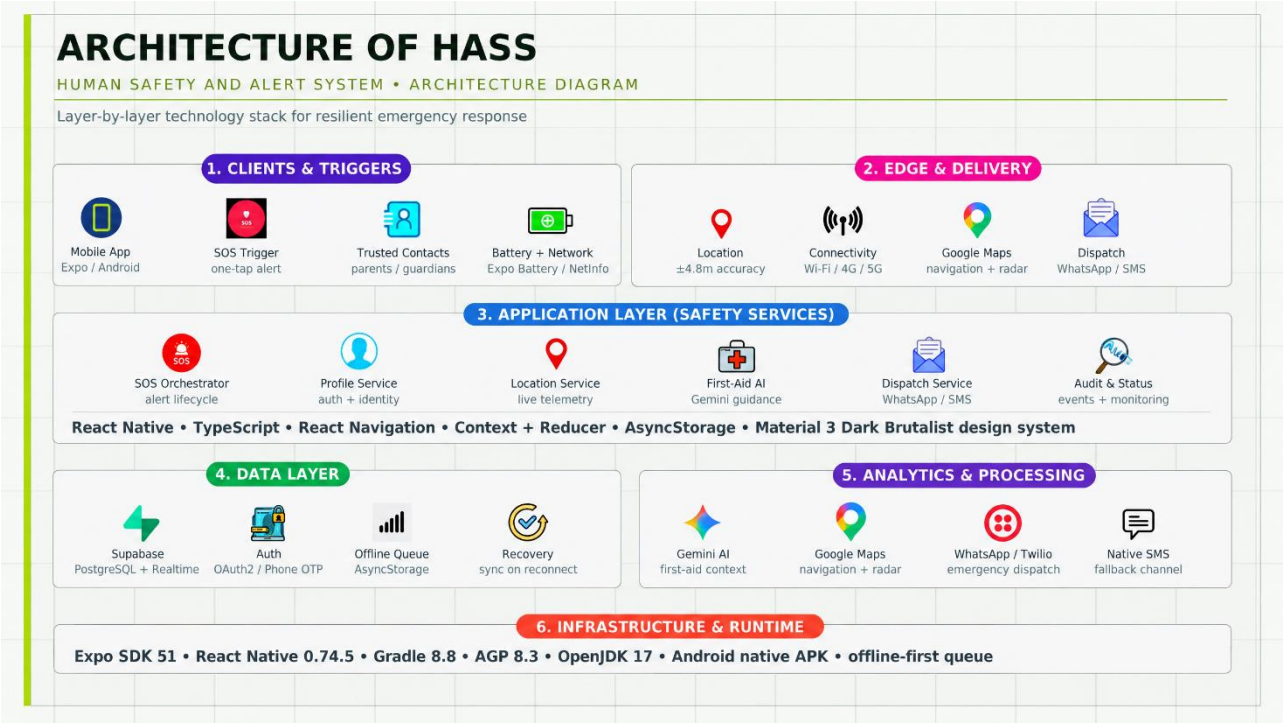


Figure 1: Architecture of Human Safety Alert System

6.2 Main Architectural Components

1. Mobile Application Layer: Provides the SOS interface, emergency categories, and trusted-contact management.
2. Backend Layer: Processes application requests and data operations [15].
3. Database Layer: Stores user profiles, trusted contacts, and emergency records [18].
4. Location and Mapping Layer: Supports location sharing, maps, and nearby-places [7][8][9][11].
5. Notification Layer: Manages emergency notifications according to the configured workflow.

6. Security Layer: Supports authentication, access controls, and data protection [16][20].

6.3 System Design

The proposed workflow begins when the user activates SOS, selects an emergency category, and completes the five-second countdown. The system then prepares emergency information and initiates the configured notification process.

Primary contacts are prioritized before secondary contacts according to the escalation rules. Location information is included when available and permitted. The system must also account for network interruptions, unavailable permissions, and notification failures [11][13].

The modular architecture allows individual components to be developed and tested separately before integration. The final capabilities must be verified against the implemented application.

Data Item	Purpose
User ID	Identifies the registered user
Emergency Type	Indicates the reported situation
Location Coordinates	Helps identify the user's location
Alert Timestamp	Records when the alert was created
Trusted Contact Details	Identifies whom to notify
Alert Status	Tracks the notification process

Table 2: Data Required During Emergency

VII. IMPLEMENTATION

The implementation of the proposed Human Safety Alert System (HASS) involves developing the mobile interface, integrating backend services, managing application data, and configuring emergency-related features. React Native, Expo, and TypeScript provide the development technologies for the mobile application [10][11][14].

7.1 Mobile Application Implementation

The mobile interface provides access to the SOS button, emergency categories, and trusted-contact management. The proposed SOS workflow includes a five-second countdown, with medical, accident, harassment, and general alert options.

7.2 Backend and Database Implementation

Supabase Auth can support user authentication, while Supabase and PostgreSQL can manage user profiles, trusted contacts, and emergency records [15][16][18]. Supabase Realtime may support live updates when configured appropriately [17].

7.3 Location and Device Information

Expo Location and mapping services can support location sharing and map display [7][8][11]. Google Places Nearby Search can help identify nearby facilities [9]. Expo Battery and Network APIs can provide supported device battery and network information [12][13].

7.4 Emergency Notification Workflow

The proposed workflow prioritizes primary contacts before secondary contacts. Alerts may include the emergency category and available location information. SMS, WhatsApp, and other notification delivery depend on the services integrated and require testing.

7.5 AI Assistance and Security

The Gemini API may support an AI-based safety assistant, subject to implementation and validation [19]. Security should be evaluated using relevant OWASP MASVS guidelines [20].

The implementation status of each feature must be confirmed through code inspection and testing. Planned features should not be presented as successfully implemented.

VIII. SYSTEM WORKING AND FUNCTIONAL MODULES

8.1 System Working

The proposed HASS follows these steps to coordinate emergency assistance:

1. SOS Activation: The user presses SOS to start a five-second countdown.
2. Emergency Selection: The user selects medical, accident, or harassment; otherwise, a general alert is prepared.
3. Alert Generation: The application prepares emergency details and available location information.
4. Contact Notification: Primary contacts are prioritized before secondary contacts.
6. Assistance Coordination: The proposed helper interface supports accepting requests and communicating with the user.
7. Location Support: Maps can display location information and nearby facilities [7][8][9].
8. Status Monitoring: Available emergency and helper status information is displayed.

Actual functionality depends on implementation, permissions, and network connectivity.

Stage	Action	Purpose
Stage 1	SOS Activation	Initiates the emergency process
Stage 2	Emergency Classification	Identifies the type of emergency
Stage 3	Primary Contact Alert	Notifies priority contacts
Stage 4	Secondary Contact Alert	Extends the alert to additional contacts
Stage 5	Helper Assistance	Enables a helper to respond
Stage 6	Emergency Escalation	Moves to the next response stage if assistance is not received

Table 3: Emergency Response Escalation Workflow

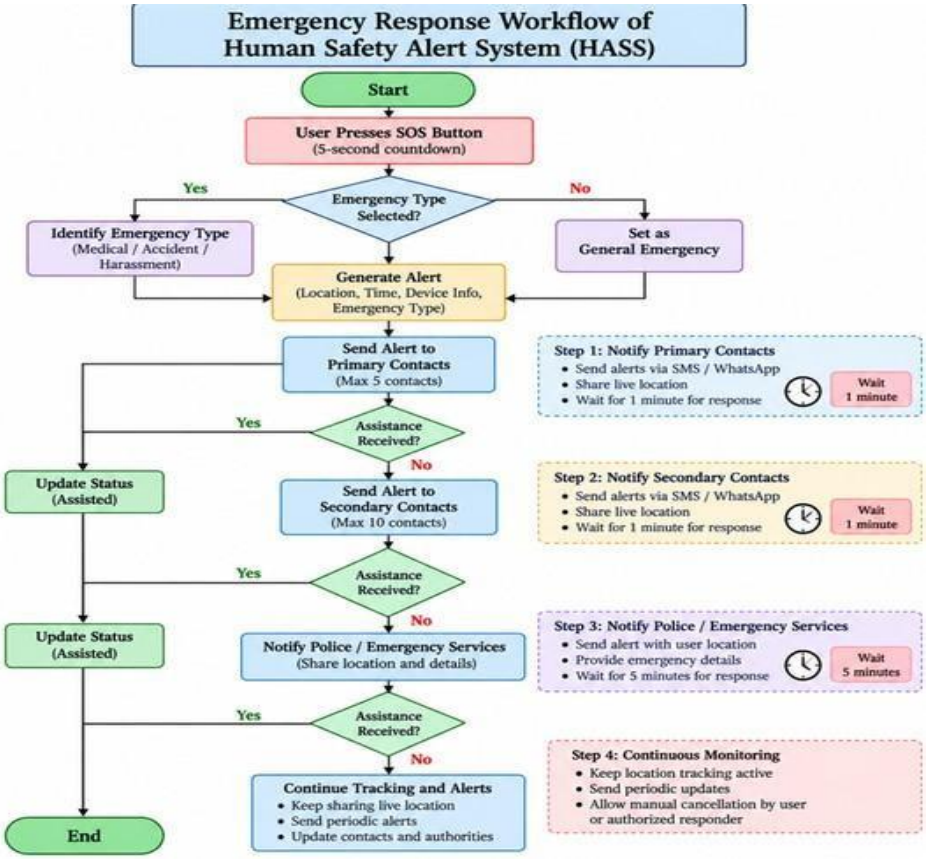


Figure 2: Emergency Response Escalation Workflow

8.2 Functional Modules

1. SOS Emergency Module: Initiates the emergency workflow with a five-second countdown.
2. Emergency Category Module: Supports medical, accident, harassment, and general alerts.
3. Trusted Contacts Module: Manages primary and secondary contacts according to notification priorities.
4. Location and Maps Module: Supports location display and nearby facility searches [7][8][9][11].
5. Notification Module: Coordinates configured emergency alert channels.
6. Helper Assistance Module: Supports helper acceptance and communication.
7. AI Safety Assistance Module: May provide supplementary guidance using the Gemini API [19].
8. Authentication and Data Module: Supports user authentication and structured data management [15][16][18].

These modules describe the intended HASS functionality. Each feature must be tested before claiming successful implementation.

IX. APPLICATION INTERFACE AND SCREENSHOTS

The Human Safety Alert System (HASS) is designed to provide a user-friendly interface for accessing emergency alert and safety-related functions. The interface is intended to simplify SOS activation, emergency category selection, trusted-contact management, and access to location-related information.

The application interface can be presented through screenshots of its main screens. These screenshots should represent the actual application implementation and demonstrate the available functions.

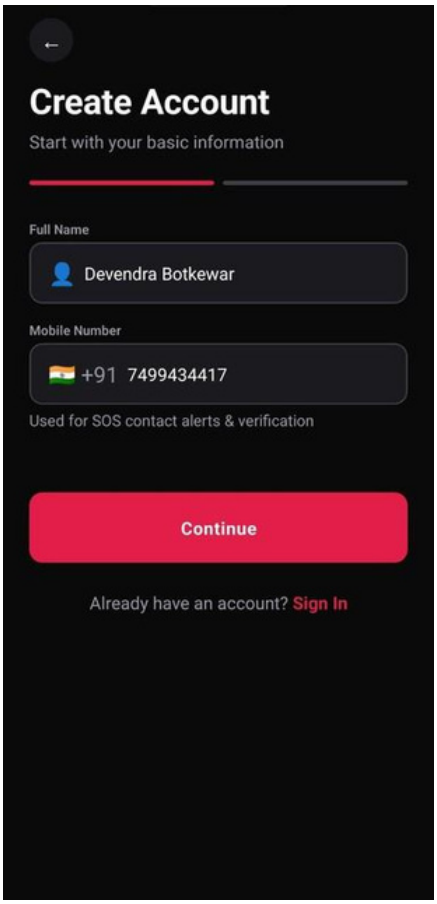


Figure 3. User Registration Screen: New user registration interface.

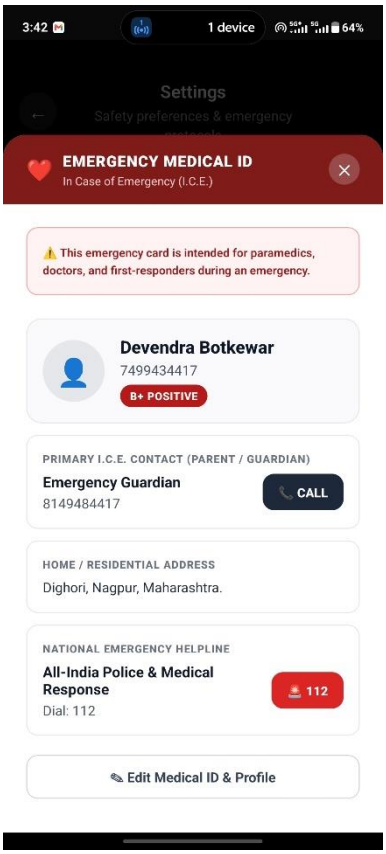


Figure 4. User Profile Screen: User details and account settings.

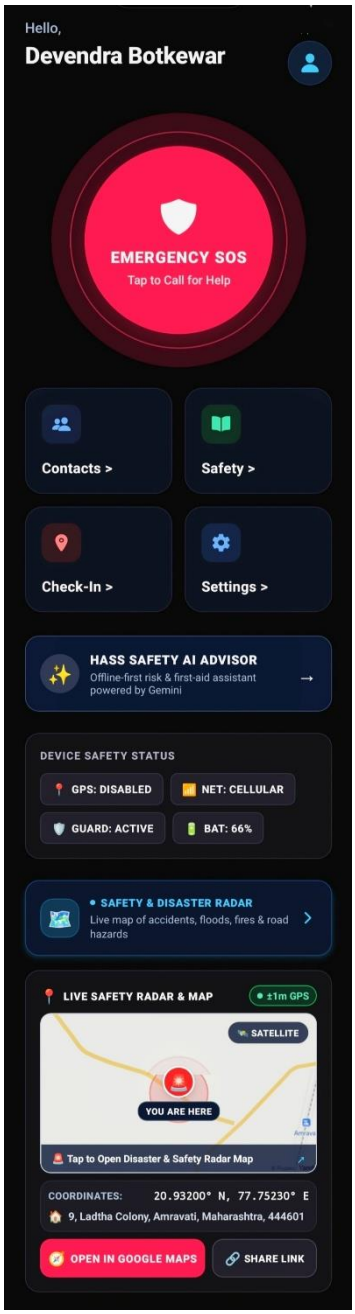


Figure 5. Home Screen: Displays the main dashboard and SOS button for initiating the emergency workflow.

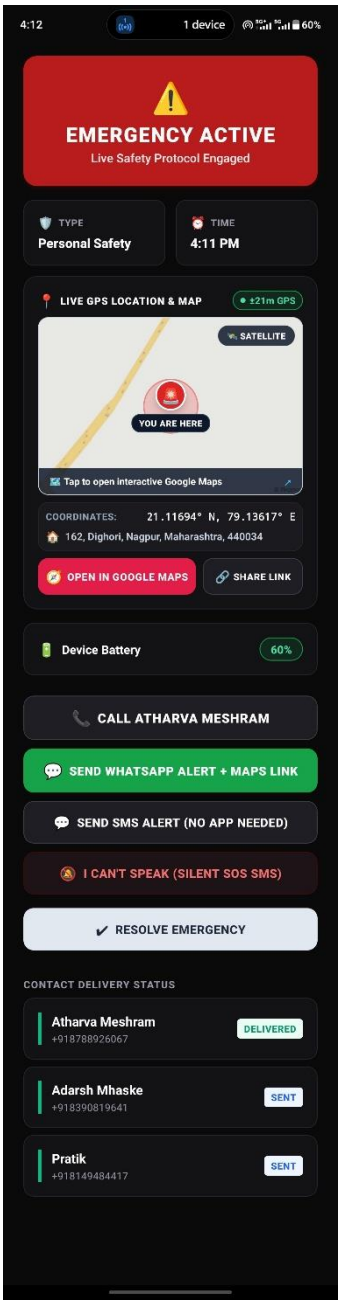


Figure 6. Emergency Alert Screen: Shows the SOS countdown and available emergency categories.

The screenshots provide a visual overview of the application interface. Only screens and features that are available in the actual application should be included in the final research paper.

X. VERIFICATION, DISCUSSION AND LIMITATIONS

10.1 Verification and Testing

HASS requires functional testing to verify its emergency workflow through the following checks:

1. SOS Functionality: Test the five-second countdown and alert initiation.
2. Emergency Categories: Verify medical, accident, harassment, and general alerts.
3. Trusted Contacts: Check contact management and notification priorities.

4. Location Services: Test location permissions and map display [7][11].
5. Notifications: Check delivery status for configured channels.
6. Security and Connectivity: Test authentication, data protection, and network interruptions [13][16][20].

Actual results must be recorded after testing; successful verification must not be claimed without evidence.

10.2 Discussion

HASS proposes integrating SOS alerts, location sharing, and trusted-contact coordination. Existing research on mobile alerts and location monitoring provides relevant context [2][3]. Its practical usefulness depends on reliable implementation and testing.

10.3 Limitations

1. Network Dependency: Online features may fail during connectivity interruptions.
2. Location Availability: Location data depends on permissions and device services.
3. Notification Delivery: Generating an alert does not guarantee its delivery or response.
4. External Services: Mapping and AI features depend on configured services.
5. Privacy and Security: Personal and location data require appropriate protection.
6. Unverified Effectiveness: Reliability and emergency outcomes require systematic testing.

These limitations should guide future improvements and evaluation of HASS.

Current Limitation	**Possible Future Improvement**
Poor network connectivity	Improve offline alert handling
Location permission disabled	Provide clear permission guidance
Notification delivery may fail	Add delivery-status checks and retry mechanisms
Incorrect or outdated location	Improve location validation
Security risks	Conduct security testing
Limited real-world evaluation	Perform user trials and performance testing

Table 4: Limitations and Future Improvements

XI. CONCLUSION AND FUTURE SCOPE

11.1 Conclusion

The Human Safety Alert System (HASS) is proposed as a mobile safety application integrating SOS alerts, emergency categories, location sharing, and trusted-contact coordination. Its design draws on existing research and technical resources for emergency alerting, mobile development, mapping, and security.

HASS aims to complement India's ERSS emergency service, accessible through 112, rather than replace it. Its reliability and effectiveness depend on proper implementation, notification delivery, connectivity, permissions, and data security. Further development and systematic testing are necessary to evaluate its practical usefulness.

11.2 Future Scope

HASS can be enhanced through the following improvements:

1. Improved Notifications: Better handling of delayed or failed alerts.
2. Offline Support: Explore offline functionality during network interruptions.
3. Enhanced Location Services: Improve location sharing and nearby facility searches.
4. AI Safety Assistance: Explore validated AI guidance using the Gemini API.
5. Security Enhancements: Strengthen authentication and data protection.
6. Performance Testing: Evaluate reliability under different device and network conditions.

These improvements are future possibilities until implemented and tested.

XII. REFERENCES

The references below were selected for the main areas covered by HASS: emergency alerting, real-time location sharing, personal safety research, mobile application development, mapping, AI assistance, and application security. Research papers are included alongside official technical documentation.

1. Ministry of Home Affairs, Government of India. (n.d.). Emergency Response Support System (ERSS) – 112 India. https://112.gov.in/
2. Mowbray, F., et al. (2024). A systematic review of the use of mobile alerting to inform the public about emergencies and the factors that influence the public response. Journal of Contingencies and Crisis Management, 32(1), e12499. https://doi.org/10.1111/1468-5973.12499
3. Gowda, D. V., Prasad, K. D. V., Shekhar, R., Srinivas, R., Srinivas, K. N. V., & Lakineni, P. (2023). Development of a real-time location monitoring app with emergency alert features for Android devices. 2023 4th IEEE Global Conference for Advancement in Technology (GCAT). https://doi.org/10.1109/GCAT59970.2023.10353310
4. An IoT-based smart wearable safety device using GSM, GPS, and ESP32-CAM. (2025). 2025 3rd International Conference on Sustainable Computing and Data Communication Systems (ICSCDS). https://doi.org/10.1109/ICSCDS65426.2025.11167200
5. SafeGuard: An offline-first emergency response system with multi-modal detection and face authentication. (2026). 2026 International Conference on AI-Driven Smart Systems and Ubiquitous Computing (ICAUC). https://doi.org/10.1109/ICAUC68182.2026.11441021
6. Indalkar, A. S., Tamgave, P. A., Khalate, S. P., Gurav, S. D., & Pawar, S. T. (2025). Enhancing community safety through real-time mobile applications: Emergency alerts, live tracking, and monitoring systems. International Journal of Emerging Technologies and Innovative Research, 12(4), l592–l597. https://www.jetir.org/papers/JETIR2504B73.pdf
7. Google for Developers. (n.d.). Location data | Maps SDK for Android. https://developers.google.com/maps/documentation/android-sdk/location
8. Google for Developers. (n.d.). Maps SDK for Android overview. https://developers.google.com/maps/documentation/android-sdk/overview
9. Google for Developers. (n.d.). Nearby Search | Places API. https://developers.google.com/maps/documentation/places/web-service/nearby-search
10. React Native. (n.d.). React Native documentation. https://reactnative.dev/docs/getting-started
11. Expo. (n.d.). Location – Expo SDK documentation. https://docs.expo.dev/versions/latest/sdk/location/
12. Expo. (n.d.). Battery – Expo SDK documentation. https://docs.expo.dev/versions/latest/sdk/battery/
13. Expo. (n.d.). Network – Expo SDK documentation. https://docs.expo.dev/versions/latest/sdk/network/
14. Microsoft. (n.d.). TypeScript handbook. https://www.typescriptlang.org/docs/handbook/
15. Supabase. (n.d.). Supabase documentation. https://supabase.com/docs
16. Supabase. (n.d.). Auth documentation. https://supabase.com/docs/guides/auth
17. Supabase. (n.d.). Realtime documentation. https://supabase.com/docs/guides/realtime
18. The PostgreSQL Global Development Group. (n.d.). PostgreSQL documentation. https://www.postgresql.org/docs/current/
19. Google AI for Developers. (n.d.). Gemini API documentation. https://ai.google.dev/gemini-api/docs
20. OWASP Foundation. (n.d.). Mobile Application Security Verification Standard (MASVS). https://mas.owasp.org/MASVS/