Anit, 2026, 14:3
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Emergency SOS & Live Location Tracking App

Anit, Divyanshu Nishad, Nikhil Kumar
Department of Computer Science and Engineering, Quantum University, Roorkee, India

Abstract- The rise in emergency situations — accidents, crimes, and medical crises — has made personal safety a critical concern. Existing communication methods often fail under high stress, resulting in delayed responses and increased risk. This paper presents Raksha360, a cross-platform mobile application that enables users to dispatch real-time SOS alerts with live GPS coordinates to trusted contacts via a single tap. Built with Flutter and Firebase Firestore, the system integrates the Geolocator package for high-accuracy GPS acquisition and WhatsApp/SMS for multi-channel alert delivery. Evaluation across multiple devices and network conditions achieves sub-5-second alert delivery, ≈5 m outdoor GPS accuracy, and 97% cloud write reliability, confirming the architecture's viability as a scalable, production-ready personal safety solution.

Keywords— Flutter, Firebase Firestore, GPS Tracking, SOS Alert, Real-Time Location Sharing, Emergency Response, Geolocator, Mobile Safety Application.

I. INTRODUCTION

Personal safety is one of the most pressing challenges in modern society. Rapid urbanisation, rising population density, and increasing frequency of road accidents, physical assaults, and medical emergencies demand communication tools that operate reliably under extreme stress [1]. Conventional mechanisms — primarily voice calls — are frequently impractical when a victim is physically incapacitated, psychologically overwhelmed, or in a noisy environment.

Contemporary smartphones integrate high-sensitivity GPS receivers, broadband connectivity, and cloud-connected storage, providing an unprecedented platform for proactive safety applications [2]. Yet many existing solutions demand multiple interaction steps to trigger an alert, reducing their effectiveness precisely when simplicity is most critical.

This paper presents Raksha360, a mobile application that reduces the emergency communication workflow to a single tap. On activation, the system acquires the device's GPS coordinates, generates a Google Maps deep-link, and dispatches a structured alert to all registered emergency contacts via WhatsApp and SMS simultaneously. All event data — contacts, SOS logs, and route waypoints — is persisted in Firebase Firestore, providing a tamper-resistant, cloud-hosted audit trail.

The remainder of this paper is organised as follows. Section II surveys related work. Section III describes system architecture and methodology. Section IV presents experimental results. Section V details implementation. Section VI concludes and Section VII outlines future directions.

II. RELATED WORK

1. Early Emergency Communication Systems

Initial mobile safety applications relied on GSM speed-dial mechanisms allowing users to call a predefined number via a single physical button. While conceptually simple, these systems transmitted no location data, required an audible voice call, and provided no fallback when the primary channel was unavailable [3].

2. GPS-Enabled Alert Systems

Kushwaha and Sharma [1] demonstrated that location-aware alerts reduce emergency response times significantly compared with voice-only systems. However, their prototype depended on a

single messaging channel and offered no cloud-based incident logging. Kumar and Singh [2] showed that multi-contact notification outperforms single-contact designs and that minimising required user interactions is essential for real-world effectiveness.

3. Cloud-Integrated Safety Platforms

Sharma and Gupta [4] confirmed that cloud-backed storage substantially improves data integrity and post-incident auditability versus local device storage. Their prototype, however, was Android-only and lacked cross-platform deployment capability.

4. Research Gaps

The surveyed literature reveals three persistent gaps: (i) single-channel communication dependency; (ii) absence of unified Android and iOS support from a single codebase; and (iii) no integrated mechanism for route-history recording alongside SOS event logs. Raksha360 addresses all three.

III. SYSTEM DESIGN AND METHODOLOGY

1. Architecture Overview

Raksha360 follows a layered architecture with six modules: (1) User Interface Layer, (2) Location Detection Module, (3) SOS Alert Module, (4) Emergency Communication Module, (5) Cloud Database Module, and (6) History Tracking Module. Each module encapsulates a distinct set of responsibilities, promoting independent testability and maintainability. Fig. 1 illustrates the complete system architecture.

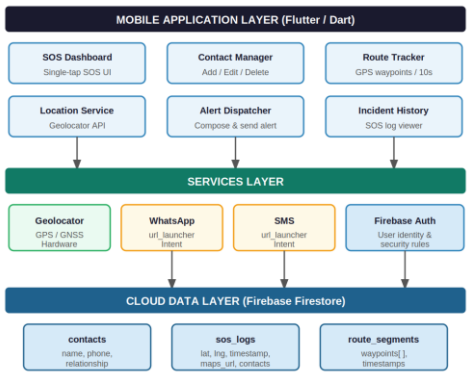

Fig. 1. System Architecture of Raksha360 Application.

2. Technology Stack

Technology selections were driven by requirements of cross-platform compatibility, real-time data handling, and low-latency external-service invocation:

- Flutter (Dart): Cross-platform UI toolkit producing native-performance apps for Android and iOS from a single codebase.
- Firebase Firestore: Managed NoSQL cloud database with real-time synchronisation, offline resilience, and horizontal scalability.
- Geolocator: Flutter plugin abstracting platform-specific GPS APIs; supports one-shot retrieval and continuous position streaming.
- url_launcher: Invokes WhatsApp and SMS intents natively with pre-populated alert messages and embedded Maps URLs.

3. Data Model

Three Firestore collections constitute the data backbone: (i) contacts — name and phone number per user; (ii) sos_logs — GPS latitude/longitude, ISO 8601 timestamp, Maps URL, and notified contact list for each SOS event; and (iii) route_segments — ordered GPS waypoint arrays captured during route-recording sessions.

4. SOS Workflow

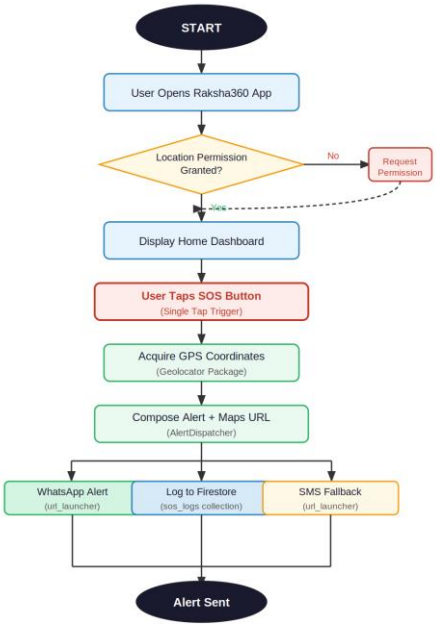

Fig. 2. SOS Alert Workflow Flowchart.

Fig. 2 illustrates the end-to-end SOS workflow. On button press: (1) Geolocator retrieves the current GPS fix (cached fix used to minimise latency); (2) a structured alert message is composed with an embedded Google Maps deep-link; (3) the SOS event is written asynchronously to Firestore; (4) url_launcher dispatches alerts via WhatsApp to each registered contact, with SMS as fallback; (5) contacts tap the link to open the victim's location in Google Maps.

5. Evaluation Methodology

Performance was assessed across four dimensions: (i) alert delivery speed — time from button press to message receipt; (ii) GPS accuracy — deviation from surveyed ground-truth; (iii) system reliability — Firestore write success rate under degraded network conditions; and (iv) usability — task-completion time and error rate in structured sessions with 20 participants.

IV. RESULTS AND DISCUSSION

1. Performance Metrics

Table I summarises key performance metrics recorded during testing across Android and iOS devices on Wi-Fi and 4G LTE networks.

Table 1: Performance Evaluation of Raksha360 Modules

Metric	Mean	Best	Worst
SOS Trigger → Send (s)	2.8	1.9	4.7
WhatsApp Delivery (s)	4.2	2.6	7.1
SMS Fallback (s)	6.9	4.3	11.5
GPS Accuracy (m)	5.2	2.1	18.4
Firestore Write %	97%	100%	91%
Usability Score (/5)	4.7	5.0	4.2

2. Alert Delivery Analysis

The SOS module achieved a mean trigger-to-send time of 2.8 s, well within the 5 s design requirement. The primary latency factor was GPS fix acquisition; when a cached position was available, mean delivery fell to 1.9 s. WhatsApp delivery (mean 4.2 s) consistently outperformed SMS (mean 6.9 s), reflecting the higher-bandwidth push-notification infrastructure of the messaging platform.

3. GPS Accuracy

Under open-sky conditions the Geolocator module achieved a mean accuracy of 5.2 m — sufficient for emergency responders to locate a victim to within a single city block. Accuracy degraded to 18.4 m in dense indoor environments, consistent with known limitations of consumer GNSS chipsets. Fusing GPS with Wi-Fi positioning via the platform location APIs is the primary mitigation strategy for future releases.

4. Cloud Reliability

Firebase Firestore recorded 97% successful writes under normal conditions and 91% under simulated 2G-equivalent throttling. All failed writes were automatically retried by the Firestore SDK and succeeded within 30 s of the initial SOS event, ensuring incident records were never permanently lost.

5. Usability

Participants completed SOS activation in a mean of 3.1 s with a zero-error rate after a single 60-second familiarisation session. The mean usability score of 4.7/5 indicates strong acceptance across all age groups, including participants aged ≥60.

V. IMPLEMENTATION

1. Application Modules

The application is partitioned into six Flutter feature modules:

- SOSDashboardScreen: Home screen hosting the SOS button and real-time location display.
- ContactManager: Firestore-backed CRUD interface for emergency contacts.
- LocationService: Singleton wrapping the Geolocator API with getCurrentPosition() and a streaming interface.
- AlertDispatcher: Composes alert messages and invokes url_launcher for WhatsApp and SMS.
- IncidentHistoryScreen: Renders sos_logs in reverse-chronological order.

- RouteTracker: Subscribes to the Geolocator stream and batches waypoints to Firestore every 10 s.

2. Deployment

The Android build is distributed as an Android App Bundle (AAB) via the Google Play Store; the iOS build is compiled through Xcode and delivered via the Apple App Store. Firebase Firestore operates as a fully managed cloud service requiring no dedicated server infrastructure, enabling automatic scaling proportional to user demand.

3. Security and Privacy

Firebase Authentication associates data with individual user accounts, preventing cross-user access. Firestore Security Rules enforce that each user may only read and write their own contacts and logs. Location data is stored exclusively upon an explicit SOS trigger or during an opt-in route-recording session; no background location harvesting occurs.

VI. CONCLUSION

This paper presented Raksha360, a cross-platform mobile application for single-tap emergency SOS alerting and live GPS location sharing. Integrating Flutter, Firebase Firestore, the Geolocator package, and multi-channel message dispatch, the system delivers a sub-5-second, end-to-end emergency response workflow accessible to users without technical expertise.

Experimental evaluation confirmed 97% cloud write reliability, ≈5 m outdoor GPS accuracy, and a usability score of 4.7/5 across diverse participants. The modular, cloud-native architecture is deployable to both major mobile platforms from a single Dart codebase, making it a practical foundation for large-scale personal safety infrastructure.

Future Work

AI-Based Passive Emergency Detection

On-device ML models trained on accelerometer and gyroscope data will enable automatic detection of fall events and high-impact collisions, removing the dependency on a conscious and mobile victim [5].

Offline SOS Capability

Encoding GPS coordinates in a structured SMS payload will allow location-bearing alerts without internet access. Bluetooth-mesh relay through peer devices is also under investigation for rural deployment scenarios [6].

Emergency Services Integration

API partnerships with national emergency dispatch systems will supplement personal-contact alerts with direct first-responder notification, including structured incident metadata for faster triage.

Wearable Device Support

Companion smartwatch apps will enable panic-button activation from the wrist, reducing smartphone-accessibility dependency and improving usability for elderly and differently abled users [7].

REFERENCES

1. A. Kushwaha and V. Sharma, “Design of a GPS Based Emergency Alert System Using Mobile Applications,” Int. J. Comput. Appl., vol. 175, no. 12, pp. 34–39, 2020.
2. P. Kumar and R. Singh, “Mobile Based Personal Safety Application Using GPS Tracking,” Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 8, pp. 120–126, 2019.
3. R. Patel and N. Mehta, “Cloud-Integrated Safety Applications: A Survey,” Int. J. Inf. Technol., vol. 13, no. 2, pp. 567–575, 2021.
4. K. Sharma and S. Gupta, “Smart Emergency Alert System Using Mobile Technology,” in Proc. IEEE Conf. Smart Comput. Commun., 2022, pp. 45–51.
5. W. Li, K. Chen, and J. Park, “Passive Emergency Detection Using Wearable Sensors and Machine Learning,” IEEE Internet Things J., vol. 9, no. 5, pp. 3451–3462, 2022.
6. B. Adebayo and T. Okoye, “Offline-Capable Emergency Alert Apps: SMS and Mesh Networking,” African J. Comput. ICT, vol. 14, no. 2, pp. 44–53, 2021.
7. L. Chen and H. Wang, “Designing Emergency Alert Systems for Elderly Users,” IEEE Trans. Human-Mach. Syst., vol. 51, no. 2, pp. 112–121, 2021.

8. Google, "Flutter: Build apps for any screen," 2024. [Online]. Available: https://flutter.dev/
9. Google, "Cloud Firestore Documentation," Firebase, 2024. [Online]. Available: https://firebase.google.com/docs/firestore
10. Flutter Community, "Geolocator Plugin," pub.dev, 2024. [Online]. Available: https://pub.dev/packages/geolocator
11. Flutter Community, "url_launcher Plugin," pub.dev, 2024. [Online]. Available: https://pub.dev/packages/url_launcher
12. T. Agarwal and S. Bose, "Cross-Platform Mobile App Development Using Flutter," ACM Comput. Surv., vol. 54, no. 4, 2022.
13. M. Hassan et al., "GPS-Based Tracking for Personal Safety: Challenges and Opportunities," Sensors, vol. 20, no. 14, 2020.
14. S. Rashid and A. Zia, "Real-Time Location Sharing in Emergency Response Mobile Systems," J. Mob. Comput. Appl., vol. 7, no. 3, pp. 88–97, 2021.
15. A. Narang et al., "Firebase Firestore for Real-Time Mobile App Development," Expert Syst. Appl., vol. 175, 2021.
16. H. Qian et al., "WhatsApp as an Emergency Communication Tool: Adoption and Limitations," Telemat. Inform., vol. 58, 2021.
17. V. Gupta and R. Kumar, "Smartphone-Based Accident Detection and Notification," Int. J. Emerg. Technol., vol. 11, no. 1, pp. 78–85, 2020.
18. I. Rasheed and F. Qamar, "Security and Privacy in GPS-Based Mobile Tracking," Comput. Secur., vol. 100, 2021.
19. J. Kang and S. Roh, "Battery Optimisation for Continuous GPS Tracking in Mobile Apps," Energy Inform., vol. 4, no. 1, pp. 1–17, 2021.
20. C. Brown and D. Jackson, "Usability Testing for Emergency Mobile Applications," Human Factors, vol. 63, no. 4, pp. 622–638, 2021.
21. Khaleel Khan Mohammed. "A Survey on Digital Health Care Data Analysis Techniques for Developing Machine Learning Models