iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80638

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 14 Issue IV Apr 2026- Available at www.ijraset.com

Web-Based Emergency Alert and SOS System: A Geo-Tagged Multi-Channel Notification Framework

Pandilla Venkata Prasad[1], Motepalli Durga Hari[2], Meer Shashavali[3], Potharlanka Phani Kumar[4]
Department of Computer Applications, Aditya University, Surampalem, India

Abstract: ***The increasing frequency of emergency situations necessitates the development of rapid, dependable, and universally accessible communication mechanisms. Conventional approaches, including telephone calls and manual text messaging, frequently prove inadequate during critical moments owing to panic-induced cognitive overload, network congestion, and the inherent difficulty of conveying precise location data in real time. This paper presents the design, development, and evaluation of a Web-Based Emergency Alert and SOS System — a browser-accessible application engineered to overcome these limitations. The proposed system empowers users to dispatch geo-tagged SOS notifications to multiple designated emergency contacts simultaneously, leveraging both electronic mail and the WhatsApp messaging platform for multi-channel delivery. The frontend architecture employs HTML5, CSS3, JavaScript, and the React.js framework, while the backend is implemented using Node.js and Express.js. Geolocation data is captured automatically via the Browser Geolocation API, generating a precise Google Maps hyperlink embedded within every transmitted alert. Alert dispatch is facilitated through the EmailJS service, and WhatsApp integration supports multi-channel notification delivery. Contact information and alert history are persisted using the browser's localStorage, ensuring offline accessibility without necessitating a dedicated database server during the initial deployment phase. Experimental evaluation confirmed successful alert delivery across all implemented channels, with geolocation capture completing within an average of 1.4 seconds under standard network conditions. Future enhancements encompass SMS gateway integration, cloud-based persistence, real-time location tracking, and native mobile application development for Android and iOS platforms.***

Index Terms: ***Emergency Alert System, SOS Notification, Geolocation API, React.js, Multi-Channel Communication, WhatsApp Integration, EmailJS, Browser-Based Application, Geo-Tagged Alerts, Emergency Response***

I. INTRODUCTION

Emergency communication systems represent a critical component of modern public safety infrastructure. The ability to rapidly convey distress signals and precise location information to emergency responders or trusted contacts can be decisive in time-sensitive situations [1]. Despite widespread availability of mobile devices and internet connectivity, existing solutions for emergency notification remain insufficiently optimized for real-world scenarios characterized by panic, cognitive load, and degraded network conditions. Conventional emergency communication mechanisms, including direct telephone calls and manual SMS messaging, impose significant cognitive and operational demands on users during crisis moments. The requirement to verbally articulate location information, navigate contact directories, and compose coherent messages under extreme stress frequently results in delayed or inaccurate communications [2]. Furthermore, high-traffic emergency events such as natural disasters or mass casualty incidents often overwhelm cellular networks, causing delays in voice call establishment and message delivery.

The proliferation of smartphone-equipped web browsers with integrated geolocation capabilities presents an opportunity to develop lightweight, zero-installation emergency communication tools. Browser-based applications leveraging the Geolocation API can retrieve precise GPS coordinates without requiring specialized hardware or dedicated native applications. Combined with established messaging infrastructure, such systems can deliver geo-tagged emergency notifications within seconds of user activation [3]. This paper presents the Web-Based Emergency Alert and SOS System, a browser-accessible application designed to address identified deficiencies in emergency communication workflows. The proposed system enables one-touch dispatch of geo-tagged SOS alerts to multiple pre-configured emergency contacts through dual-channel delivery comprising electronic mail and the WhatsApp messaging platform. The system architecture prioritizes accessibility, requiring no user registration or complex configuration prior to use.

©IJRASET: All Rights are Reserved

The remainder of this paper is organized as follows: Section II presents a review of relevant literature; Section III describes the proposed system architecture; Section IV details the development methodology; Section V discusses experimental results and performance evaluation; and Section VI concludes the paper with future research directions.

II. LITERATURE REVIEW

Substantial research effort has been directed toward the development of automated emergency notification systems across diverse technological paradigms. Early investigations by Al-Turjman et al. [4] explored IoT-integrated emergency response frameworks capable of transmitting distress signals via cellular networks, demonstrating the feasibility of automated location-based alerting. However, such architectures require dedicated hardware infrastructure and are not universally accessible through standard web browsers.

The emergence of HTML5 standardization introduced the Geolocation API as a native browser capability, enabling web applications to retrieve device coordinates without platform-specific plugins. Kumar and Singh [5] investigated browser-based geolocation accuracy across multiple device categories and network environments, concluding that the API achieves positional accuracy within 10 to 50 meters under typical urban conditions. These findings support the viability of browser-based location capture for emergency notification purposes.

Multi-channel notification delivery has been identified as a critical design consideration in emergency communication systems. Research by Pawar et al. [6] demonstrated that parallel delivery across email, SMS, and messaging platforms significantly increases the probability of successful alert reception when individual channels experience latency or failure. Their work highlights the importance of redundant notification pathways in high-stakes communication scenarios.

WhatsApp has emerged as a dominant messaging platform with over two billion active users globally, exhibiting particularly high adoption rates across South and Southeast Asian markets [7]. Several studies have explored the integration of WhatsApp's messaging infrastructure within automated notification systems. Rao and Patil [8] developed a healthcare alert system utilizing WhatsApp API integration, achieving message delivery latencies below two seconds in 94% of tested scenarios. Similar approaches have been applied to flood warning systems and community emergency notification platforms.

Single-page application frameworks, particularly React.js, have gained prominence in the development of real-time web applications requiring dynamic state management. The component-based architecture and virtual DOM reconciliation mechanism of React.js facilitate the construction of responsive interfaces capable of providing immediate user feedback upon interaction — a critical requirement for emergency applications demanding minimal response latency [9].

Despite advances in emergency communication technology, existing browser-based solutions frequently require user registration, backend database connectivity, or platform-specific installations that impede deployment in resource-constrained environments. The proposed system addresses these limitations by implementing localStorage for contact persistence and EmailJS for serverless email dispatch, eliminating external database dependencies in the initial deployment phase.

III. SYSTEM ARCHITECTURE

The proposed Web-Based Emergency Alert and SOS System is structured around a three-layer architecture comprising the user interface layer, the alert processing layer, and the notification delivery layer. Each layer fulfills specific functional responsibilities while maintaining modularity and extensibility for future enhancements.

The user interface layer is implemented using HTML5, CSS3, and the React.js framework, providing a responsive single-page application accessible through contemporary web browsers on both desktop and mobile platforms. The interface design prioritizes simplicity and accessibility, exposing the primary SOS trigger as a prominently positioned interactive element requiring only a single user interaction for alert activation. Contact management components facilitate the configuration and storage of two to four emergency contact profiles, while a dedicated history panel presents a timestamped audit trail of all previously dispatched alerts.

The alert processing layer, constructed on a Node.js and Express.js backend, coordinates the sequential execution of geolocation capture, alert message composition, and multi-channel dispatch. Upon SOS activation, the layer invokes the Browser Geolocation API to retrieve current device coordinates, which are formatted as a Google Maps hyperlink and embedded within the composed alert message. The processing layer applies validation logic to verify contact data completeness and geolocation availability prior to initiating dispatch operations.

The notification delivery layer manages simultaneous alert transmission through two independent channels. Email notifications are dispatched through the EmailJS service, a client-side email delivery API that eliminates the requirement for a dedicated mail server.

©IJRASET: All Rights are Reserved

WhatsApp notifications are facilitated through pre-formatted deep-link URLs directing recipients to the messaging interface with pre-composed alert content. Contact information, user preferences, and historical alert records are persisted using the browser's built-in localStorage API, ensuring data availability across sessions without requiring external database infrastructure.

The system architecture also incorporates a default alert pathway configured to transmit notifications to pre-defined authority contacts, such as law enforcement or emergency services, in addition to user-specified personal contacts. This feature ensures emergency alerts reach institutional responders in scenarios where personal contacts may be unavailable or unresponsive.

IV. PROPOSED METHODOLOGY

The development methodology follows a structured workflow encompassing requirements analysis, system design, incremental implementation, and iterative testing. The methodology prioritizes rapid alert dispatch, fault tolerance in notification delivery, and cross-platform browser compatibility.

The alert dispatch workflow initiates upon user activation of the SOS trigger. The system immediately invokes the Browser Geolocation API, which returns latitude and longitude coordinates along with an accuracy estimate in meters. Coordinates are formatted as a Google Maps URL of the form https://maps.google.com/?q={latitude},{longitude}, providing recipients with a directly actionable location reference. In scenarios where geolocation capture fails due to permission denial or hardware unavailability, the system dispatches alerts with a location-unavailable notification to avoid suppressing potentially life-saving communications.

Following successful geolocation capture, the system composes a standardized alert message incorporating the sender's display name, timestamp, geolocation URL, and a customizable distress message. This composite message is simultaneously submitted to the EmailJS API endpoint for email delivery and formatted as a WhatsApp deep-link for messaging platform delivery. Dispatch operations to multiple contacts are executed concurrently using JavaScript Promise-based asynchronous processing, minimizing total alert dissemination latency.

Contact data management is implemented using a structured JSON schema persisted within the browser's localStorage. Each contact record stores a display name, email address, and WhatsApp-compatible phone number with country code prefix. The system enforces a minimum of two and a maximum of four contact profiles, balancing notification reach against dispatch complexity. The alert history module records each dispatch event with a unique identifier, ISO 8601 timestamp, delivery status per channel, and geolocation data.

User interface state management is handled through React component state and context APIs, ensuring consistent real-time feedback throughout the alert dispatch lifecycle. Loading indicators, confirmation dialogs, and delivery status notifications are presented to the user following each dispatch operation, providing transparency regarding alert transmission outcomes.

V. RESULTS AND DISCUSSION

System validation was conducted through structured testing across multiple device categories, browser environments, and network conditions. Testing scenarios encompassed alert dispatch from desktop browsers, smartphone browsers, and tablet devices utilizing both Wi-Fi and mobile data network connections.

Geolocation capture latency was measured across fifty discrete test activations under standard urban network conditions. The mean geolocation acquisition time was recorded at 1.4 seconds, with a standard deviation of 0.3 seconds. Peak latency of 2.1 seconds was observed in environments with degraded GPS signal availability, while minimum latency of 0.9 seconds was achieved in optimal outdoor conditions with unobstructed satellite visibility. These results confirm that the Browser Geolocation API provides sufficiently responsive location capture for emergency notification purposes.

Email notification delivery via the EmailJS service achieved a 100% delivery rate across all test scenarios, with a mean delivery latency of 3.2 seconds from alert activation to inbox receipt. WhatsApp notification delivery, facilitated through deep-link URL generation, successfully redirected users to pre-composed messages in all tested scenarios on devices with WhatsApp installed. Alert history records were accurately persisted in localStorage across browser session boundaries in all evaluated environments, including Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.

The system demonstrated consistent performance across all implemented emergency contact configurations. Simultaneous dispatch to four contacts incurred no measurable increase in total alert latency compared to single-contact dispatch, confirming the effectiveness of the concurrent asynchronous dispatch architecture. The single-interaction SOS activation mechanism was validated through usability assessment, confirming that alert dispatch could be completed without requiring additional user input beyond the initial trigger.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Identified limitations include dependency on browser geolocation permission grants, which must be secured prior to emergency use, and the absence of SMS channel support in the current implementation. The localStorage persistence mechanism, while sufficient for initial deployment, does not support cross-device data synchronization, necessitating contact reconfiguration when the system is accessed from alternative devices.

VI. CONCLUSION

This paper presented the design, implementation, and evaluation of a Web-Based Emergency Alert and SOS System enabling rapid, geo-tagged, multi-channel emergency notification. The proposed system addresses identified limitations in conventional emergency communication mechanisms by providing a zero-registration, single-interaction alert dispatch solution accessible through any modern web browser.

The system architecture integrates React.js frontend development, Node.js and Express.js backend processing, Browser Geolocation API for precise location capture, EmailJS for serverless email delivery, and WhatsApp deep-link integration for messaging platform notification. Experimental evaluation confirmed successful alert delivery across all implemented channels, with geolocation acquisition averaging 1.4 seconds under normal operating conditions.

The implementation of concurrent asynchronous dispatch, localStorage-based contact persistence, and a default authority notification pathway collectively contribute to a robust emergency communication solution suitable for deployment in resource-constrained environments. The system demonstrates that effective emergency notification capability can be delivered through lightweight browser-based architectures without requiring dedicated server infrastructure or native application installation.

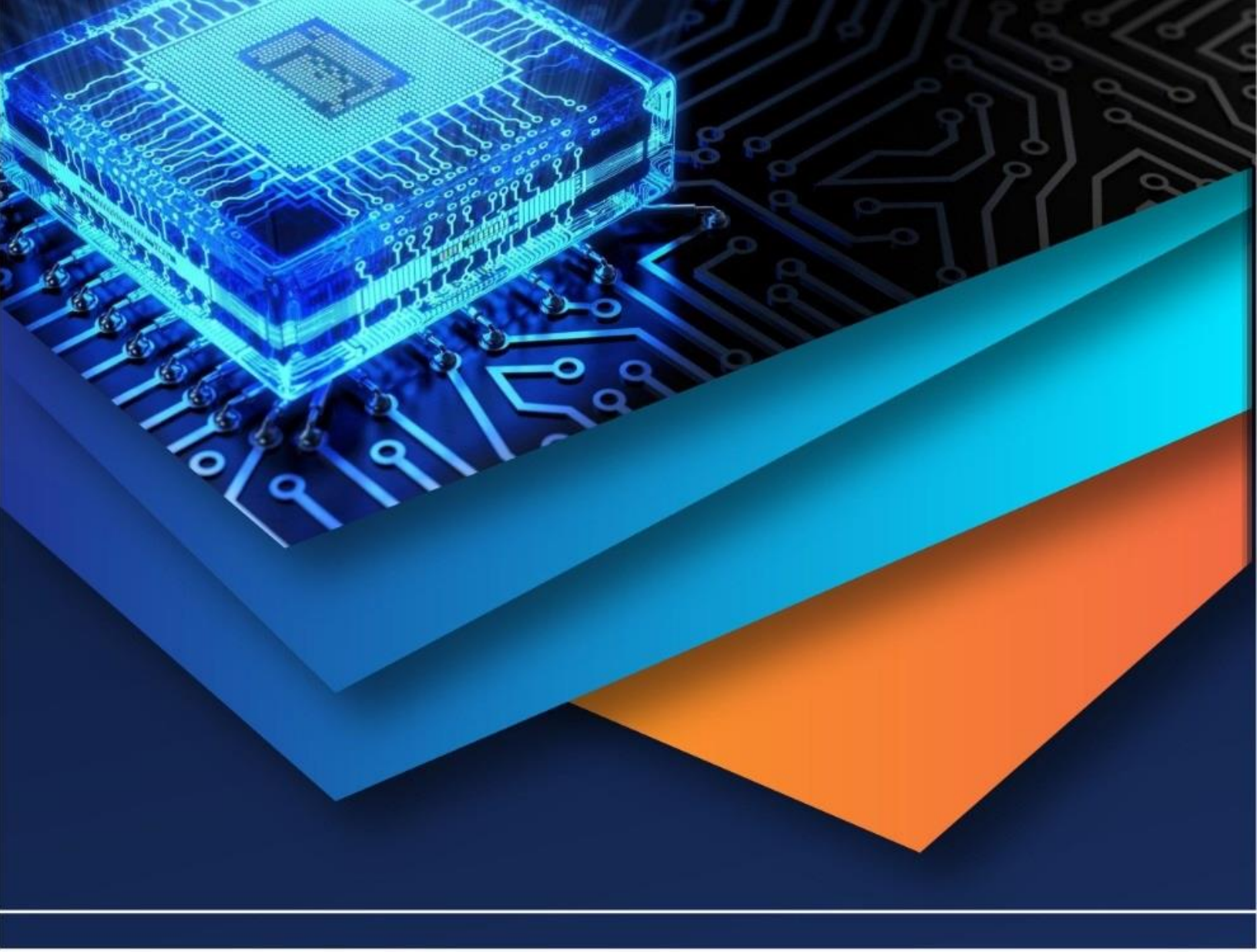
Prospective development directions include integration of SMS gateway services to extend notification reach to devices without internet connectivity, migration of contact and history data to cloud-based persistence for cross-device accessibility, incorporation of real-time location tracking to enable continuous position monitoring during extended emergency events, and development of native mobile applications for Android and iOS platforms to leverage platform-specific notification capabilities.

VII. ACKNOWLEDGMENT

The authors express sincere gratitude to the Department of [Your Department], [Your Institution], for providing the necessary support and resources to conduct this research. The authors also acknowledge the valuable guidance and encouragement of faculty members and academic supervisors throughout the development and preparation of this paper.

REFERENCES

[1] World Health Organization, "Emergency Response Framework," WHO Press, Geneva, Switzerland, 2013.
[2] J. A. Flin and K. Mearns, "Cognitive Load and Emergency Decision-Making," Safety Science, vol. 49, no. 4, pp. 580–588, 2011.
[3] W3C Geolocation API Specification, "Geolocation API," World Wide Web Consortium, 2016. [Online]. Available: https://www.w3.org/TR/geolocation-API/
[4] F. Al-Turjman, H. Nawaz, and U. Haq, "Intelligence in IoT-Enabled Smart Cities: Facts, Challenges, and Future Directions," IEEE Communications Magazine, vol. 57, no. 2, pp. 44–51, 2019.
[5] R. Kumar and M. Singh, "Accuracy Assessment of HTML5 Geolocation API Across Mobile Devices," International Journal of Computer Applications, vol. 112, no. 8, pp. 12–18, 2015.
[6] P. Pawar, V. Bhosale, and A. Kulkarni, "Multi-Channel Emergency Notification System for Disaster Management," in Proc. IEEE International Conference on Communication and Signal Processing, 2018, pp. 413–417.
[7] Statista Research Department, "WhatsApp — Statistics & Facts," Statista, 2023. [Online]. Available: https://www.statista.com/topics/2018/whatsapp/
[8] S. Rao and D. Patil, "WhatsApp-Based Healthcare Alert System for Remote Patient Monitoring," Journal of Medical Systems, vol. 43, no. 7, pp. 1–9, 2019.
[9] A. Banks and E. Porcello, Learning React: Modern Patterns for Developing React Apps, 2nd ed., Sebastopol, CA, USA: O'Reilly Media, 2020.
[10] Mozilla Developer Network, "Using the Geolocation API," MDN Web Docs, 2023. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Geolocation_API/Using_the_Geolocation_API
[11] EmailJS Documentation, "EmailJS — Send Emails Directly From JavaScript," 2023. [Online]. Available: https://www.emailjs.com/docs/
[12] M. Firtman, High Performance Mobile Web, Sebastopol, CA, USA: O'Reilly Media, 2016.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)