IJIRAE:: International Journal of Innovative Research in Advanced Engineering ISSN: 2349-2163
Volume 12, Issue 06, June 2025 https://www.ijirae.com/archives

Real-Time Emergency Response System Using Geolocation, Data Analytics, and Machine Learning

Harshit Prakash Tiwari, Deepak Kumar, Kartik Jha
Department of Information Technology,
Greater Noida Institute of Technology, (Engg. Institute),
Greater Noida, India
Priyanka, SK Wasil Umar, Puneet Upadhyay
Department of Information Technology,
Greater Noida Institute of Technology, (Engg. Institute),
Greater Noida, India
Dr.Vikas Singhal
Professor & Head, Department of Information Technology,
Greater Noida Institute of Technology, (Engg. Institute),
Greater Noida, India

Publication History
Manuscript Reference No: IJIRAE/RS/Vol.12/Issue06/JNAE10082
Research Article | Open Access | Double-Blind Peer-Reviewed | Article ID: IJIRAE/RS/Vol.12/Issue06/JNAE10082
Received: 20, May 2025, Revised: 01, June 2025, Accepted: 09, June 2025, Published Online: 13, June 2025.
https://www.ijirae.com/volumes/Vol12/iss-06/01.JNAE10082.pdf
Article Citation: Harshit,Deepak,Kartik,Priyanka,Wasil,Puneet,Dr.Vikas (2025), Real-Time Emergency Response System Using Geolocation, Data Analytics, and Machine Learning. IJIRAE: International Journal of Innovative Research in Advanced Engineering, Volume 12, Issue 06 of 2025 pages 287-291
Doi:> https://doi.org/10.26562/ijirae.2025.v1206.01
BibTeX Key: Dr.Vikas@2025Real

Abstract: In critical emergency situations, rapid response can significantly reduce casualties and damage. This research proposes an intelligent emergency response system that leverages geolocation APIs and machine learning to optimize dispatch and resource allocation. By integrating real-time location data from users and emergency services, the system accurately identifies incident locations and dynamically selects the nearest and most suitable emergency units. Machine learning models are employed to predict incident severity, estimate response times, and continuously improve decision-making based on historical data. The proposed system demonstrates improved efficiency, reduced response times, and enhanced situational awareness, offering a scalable and adaptive solution for modern emergency management infrastructures. A GPS (Global Positioning System) modem embedded in the ambulance allows the hospital's portal servers to continuously track its live location. WebRTC (Web Real-Time Communications) live video streaming provides real-time doctor-patient interaction during emergencies, enabling seamless audio and video communication.
Keywords: Geolocation API, Machine Learning, Real-time Data, GPS, WebRTC

I. INTRODUCTION

Timely and efficient emergency response is crucial in saving lives and minimizing damage during critical incidents such as medical emergencies, natural disasters, or accidents. Traditional emergency response systems often rely on manual reporting, static resource allocation, and delayed communication, which can result in slow response times and ineffective service delivery. With the advancement of geolocation technologies and the rise of intelligent systems, there is an opportunity to modernizeemergency response mechanisms to become faster, smarter, and more adaptive. This research introduces a novel emergency response framework that integrates geolocation APIs with machine learning algorithms to enhance decision-making and reduce response time. Geolocation services provide accurate, real-time location tracking of both users in distress and available emergency units, enabling the system to identify and dispatch the closest and most appropriate resources.

Machine learning models are incorporated to analyze historical incident data, predict the severity of incoming alerts, estimate travel times, and continuously improve system performance through learning. The primary objective of this study is to design and evaluate a scalable, real-time emergency response system that leverages intelligent data processing and spatial analysis to optimize resource deployment. The proposed system is intended to assist emergency operators, healthcare professionals, and disaster management agencies in delivering timely assistance with greater efficiency. This paper discusses the architecture, methodologies, implementation challenges, and performance evaluation of the proposed system, aiming to contribute to the development of next-generation emergency management solutions. Despite technological advancements, many emergency response systems still struggle with fragmented data, limited automation, and lack of predictive capabilities. These shortcomings can result in delayed assistance, inefficient use of emergency units, and poor outcomes in critical scenarios. The integration of geolocation APIs allows real-time tracking and spatial mapping, which is essential for identifying the exact location of incidents and available responders.

When combined with mobile applications and Internet of Things (IoT) devices, geolocation technology provides a robust infrastructure for situational awareness and rapid communication between victims, responders, and control centre. Furthermore, machine learning plays a vital role in enhancing the intelligence of the response system. By analysing patterns from historical data such as call volume, incident types, traffic conditions, and response success rates machine learning models can assist in prioritizing calls, predicting outcomes, and allocating resources proactively. Techniques such as classification, clustering, and regression analysis enable the system to continuously evolve and adapt to changing urban dynamics and emergency trends. This intelligent, data-driven approach holds the potential to revolutionize emergency management by transitioning from reactive to proactive response strategies.

II. LITERATURE REVIEW

Several related works support the proposed system by emphasizing efficient dynamic routing and improving the speed and precision of ambulance dispatch. Dijkstra's algorithm, often used with dynamic reference positions, is a key routing method. AHP (Analytic Hierarchy Process) reduces response time by analyzing resource data and providing optimal route decisions [1].

Some approaches merge Dijkstra's algorithm with FCFS (First Come First Serve). Servers utilize GPS modules to acquire latitude and longitude of target sites. GPS and GIS systems are employed to locate users and addresses, with information stored on cloud servers via internet access [2].

Similarly, GPS and GSM modules enable real-time tracking of public transport, where coordinates are sent via GSM as SMS to emergency numbers like 108 in case of accidents, some systems integrate Renesas microcontrollers for hardware control [3].

Hardware-based models aim to resolve traffic bottlenecks, incorporating GPS tracking, ambulance registration, and Google Maps-based real-time visualization [4].

A different system uses Arduino Uno with a GPS module to detect accident points, and the GSM module sends alerts to control rooms and emergency contacts. Crash and vibration detection relies on piezoelectric sensors [5].

Another framework applies client-server architecture and the TCP/IP protocol to portable medical devices, supporting real-time doctor access to emergency sites and enabling bio-signal and image transmission through GSM networks [6].

A mobile GPS application called 'Traveler's Sidekick' uses the TSDC algorithm to find shortest routes, with Google Maps JavaScript API V3 used for distance and map rendering [7].

A novel method applies circular trilateration to locate moving targets. The Seidel model simulates signal propagation in wireless environments. This solution emphasizes LBS (Location-Based Services), using GSM-based triangulation to track ambulances in defined regions [8].

More advanced systems monitor ambulance movement and real-time patient data like heartbeat, temperature, and blood pressure, transmitting the data to hospitals via GSM for better preparedness [9].

Some IoT-based systems integrate GPS and GSM with cloud computing, uploading coordinates to the cloud to recommend the most efficient hospital-to-incident route [10].

Table 2.1

Problem	Limitations	Solution
Static Dijkstra algorithm routing	Not responsive to changing conditions	Dynamic routing with machine learning and live GPS updates
GSM Based Data Transmission	Slow, insecure, and limited data transfer	Real-time, secure video transmission via WebRTC with SSL
Dependence on hardware systems	Limited scalability and flexibility	Cloud-based system providing scalability and adaptability
Basic crash detection systems	False alarms and low accuracy	Machine learning enhances detection precision with sensor data
Delay in transmitting patient vitals	Hospitals receive alerts too late	Real-time synchronization of vital signs using GSM and cloud
No predictive traffic routing	Routes are based on past data, not real-time	Machine learning models predict traffic and optimize routing

The Ant Colony System Algorithm is also employed to optimize emergency routing and minimize delays [11].

WebRTC is incorporated in emergency solutions to provide live video, voice, and medical image transmission such as x-rays. One implementation uses Amazon Web Services and Elastic Beanstalk for deployment, ensuring data security via SSL/TLS and enabling peer-to-peer streaming with WebRTC Data Channel and RTC Peer Connection [12].

Other solutions apply machine learning, such as Decision Tree Regression and Linear Support Vector Regression, to stabilize video streams, using WebRTC for 4G-based real-time streaming [13].

Raspberry Pi models B and B+ are utilized for live ambulance video feeds via RTMP to hospital systems [14].

Educational platforms also adopt WebRTC for interactive remote learning. These use a Node.js backend with MongoDB and STUN/TURN servers within MCU units to manage WebRTC communication [15].

Another approach blends WebRTC with social network APIs, forming peer-to-peer overlays that enable web-based applications to serve as clients for live video sharing in distributed P2P systems [16].

III. METHODOLOGY

The proposed emergency response system is developed using a component based web architecture with React as the frontend framework. The system leverages real-time geolocation APIs, cloud-based databases, and machine learning models to create an intelligent, responsive, and user-friendly interface for both emergency users and response operators. The methodology consists of four main phases: system architecture design, frontend development, backend integration, and machine learning model deployment. The system is built on a client-server model. The frontend is developed using React, enabling fast rendering, modular code, and responsive design. The backend is implemented using Node.js and Express for API handling, while a MongoDB or Firebase database stores user profiles, emergency logs, and responder locations. Google Maps API or Mapbox is used to provide real-time geolocation services, allowing both users and responders to view and share locations. The architecture supports real-time communication using WebSockets or Firebase Realtime Database.

React is used to create interactive user interfaces with reusable components. Key UI elements include: A user dashboard to report emergencies with one-click geolocation access. A live map view for responders to track incident locations. A dispatch center panel for operators to monitor all ongoing incidents and assign responders. React Hooks and Context API are used for state management, while libraries like Axios are used to fetch data from the backend. The frontend ensures real-time updates through WebSocket connections or Firebase listeners for immediate changes in location and response statuses.

The backend serves as the bridge between the React frontend, the database, and the machine learning engine. RESTful APIs handle emergency requests, location updates, user authentication, and responder dispatching. Location data is processed and stored securely. Machine learning predictions (e.g., severity assessment or ETA) are exposed as APIs that can be called from the frontend to enhance the decision-making process.

Machine learning models are trained offline using historical emergency and geolocation data. Models such as Logistic Regression, Random Forest, or K-Nearest Neighbors are used to: Predict the urgency/severity of reported emergencies. Recommend the best responder based on proximity and availability. Estimate response time using real-time and historical data. These models are hosted on a Python Flask or FastAPI server, and endpoints are integrated into the main application for predictions. The models continuously improve as new incident data is added.

The application will function based on a First Come First Serve (FCFS) approach. Notifications will be sent to all hospitals within a designated range using the Flutter notification package. The hospital that responds the quickest will be the one to dispatch the ambulance. To verify the emergency, the hospital can contact the patient using the phone number provided by the user, and may store the information in their hospital database to manage ambulance payment details. During the call, the hospital can confirm the type of ambulance needed, the patient's condition, and any additional survival requirements. The default range is set to 3 km, but it may be increased if ambulance availability is low. The live coordinates, transmitted through Node.js and socket.io, will be tracked by the Google Maps API, following the markers for both the source and destination locations. WebRTC will be used for video calls or live streams during emergencies, serving only as an emergency assistance tool and not for scheduling doctor appointments. By integrating WebRTC into Emergency Response System, we enable real-time transmission of live video, audio, and data between individuals in the field—such as victims or responders—and the central command unit.

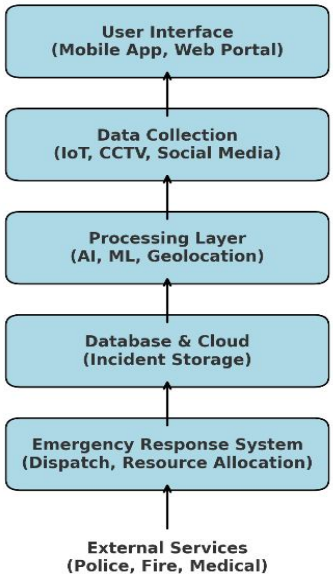

Fig. 3.1 Work Flow Diagram

IV. DATA RESOURCES

The implementation of the emergency response system relies on a robust combination of hardware and software resources to ensure seamless operation, real-time data handling, and system scalability. On the hardware side, GPS-enabled mobile devices such as smartphones and tablets are essential for both end-users and responders to interact with the system. These devices facilitate emergency reporting, real-time location sharing, and communication. For development and deployment, standard computers or laptops with at least 8GB RAM, SSD storage, and a reliable internet connection are used by developers. Additionally, cloud-based servers hosted on platforms like Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure are employed to manage backend services, machine learning models, and the application's real-time infrastructure. On the software side, the frontend of the application is developed using React.js, chosen for its component-based architecture and efficient state management.

Which ensures a smooth and responsive user experience. The backend is powered by Node.js and Express.js, which handle API requests, authentication, data processing, and integration with other system components. For data storage and retrieval, MongoDB or Firebase is used to manage user records, incident logs, and responder locations. Real-time geolocation and mapping functionalities are implemented using Google Maps API or Mapbox, allowing accurate positioning and route calculation. Machine learning models, developed using Python with libraries like scikit-learn, pandas, and NumPy, are hosted using lightweight web frameworks such as Flask or FastAPI and accessed via RESTful APIs from the frontend. These models predict emergency severity, estimated response time, and optimize resource deployment. Cloud services also support features like real-time synchronization, push notifications, and auto-scaling to handle high traffic during critical situations. Finally, tools such as Git and GitHub are used for version control and team collaboration, while Postman and Jest are utilized for API testing and frontend unit testing, respectively.

V. PURPOSE

The primary purpose of this research is to design, develop, and evaluate an intelligent emergency response system that leverages geolocation technologies and machine learning algorithms to significantly reduce emergency response time and improve resource allocation. Traditional emergency systems often suffer from delays caused by manual location identification, inefficient routing, and lack of predictive support. This research aims to address these limitations by integrating real-time location tracking through geolocation APIs with smart decision-making powered by machine learning. By building the system using modern web technologies such as React.js, the research also seeks to create a responsive, user-friendly interface for both emergency victims and first responders. The study explores how data-driven insights derived from historical incidents, traffic data, and user behaviourcan be used to forecast incident severity, estimate optimal response times, and suggest the most efficient dispatch strategy. Ultimately, the goal is to contribute to the modernization of emergency management infrastructure by offering a scalable, real-time, and intelligent solution that can be adapted to various types of emergencies and geographic regions.

VI. USER INTERFACE

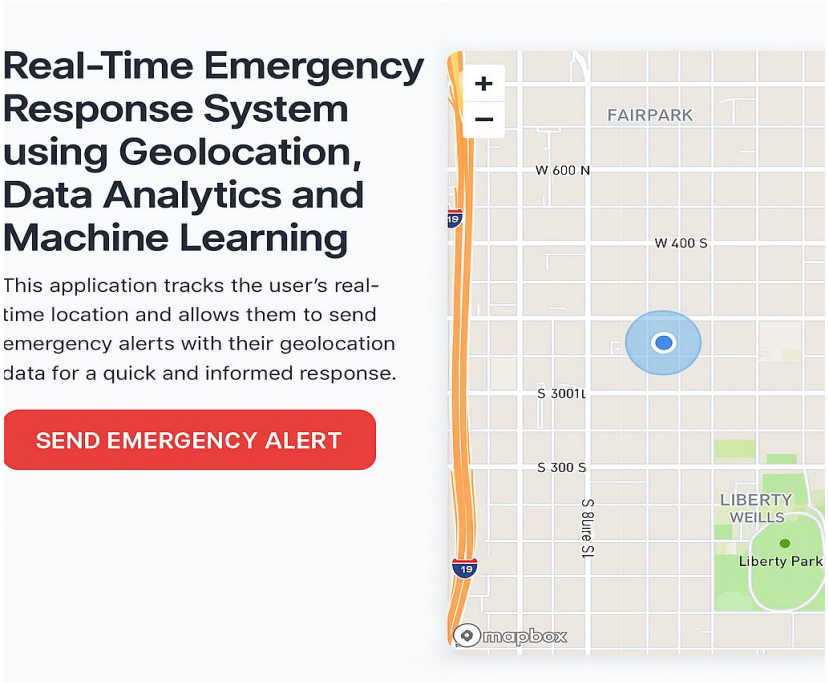

Fig. 6.1 Front Page

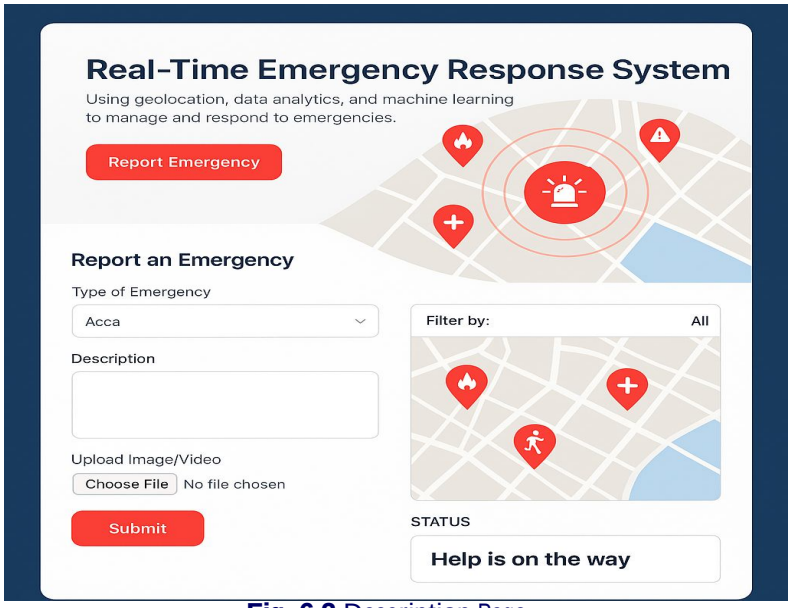

Fig. 6.2 Description Page

VII. CONCLUSION

In conclusion this research, we proposed and developed a real-time emergency response system that combines the power of geolocation APIs, machine learning, and a React-based frontend to enhance the speed, accuracy, and efficiency of emergency services. The system addresses critical challenges in traditional emergency management such as delayed response times, manual decision-making, and poor resource coordination by offering automated, data-driven solutions. Through real-time location tracking, intelligent incident analysis, and predictive modeling, the system ensures that the nearest and most appropriate responder is dispatched promptly, ultimately saving time and potentially saving lives. The use of modern web technologies such as React provides a responsive, user-friendly interface for both users and emergency operators, while the integration of machine learning models offers continuous system improvement based on historical data patterns. The project demonstrates that smart, location-aware, and predictive systems can play a transformative role in emergency response infrastructure. Future enhancements may include integration with IoT devices, support for multilingual communication, and large-scale deployment in urban and rural settings to further expand the system's impact and adaptability.