IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

A Real-Time API-Driven Emergency Vehicle Navigation System Using Flask and Mapping APIs

Dr. Anup D. Bhange[1], Komal S. Nimje[2], Khushbu R. Gautam[3]
[1]Head of Department, Department of Computer Application
[2,3]PG Scholar, Department of Computer Application
K.D.K. College of Engineering, Nagpur, Maharashtra, India
anup.bhange@kdkce.edu.in, komal.nimje.mca24f@kdkce.edu.in,
khushbu.gautam.mca24f@kdkce.edu.in

Abstract: *In emergency response operations, every second is critical. Conventional navigation systems are designed for general public use and fail to meet the specialized demands of emergency vehicles such as ambulances, fire trucks, and police units — particularly with respect to real-time rerouting, priority-based path selection, and integrated emergency service support. This paper presents Emergency Vehicle Path Optimization System — an intelligent, AI-integrated real-time navigation and dispatch system built using a Python Flask backend, the Mappls MapMyIndia API suite, and a command-center-style web interface. The system delivers personalized emergency routing through autosuggest location search, vehicle-type-aware path computation, live GPS tracking with progressive route trimming, traffic congestion severity detectionA mobile-optimized drag-up bottom sheet interface and a persistent dark/light theme system ensure usability across device types.*

Keywords: Emergency Vehicle Routing, Real-Time GPS Tracking, Mappls MapMyIndia API, Traffic Congestion Detection, SOS Alert System, Python Flask Backend, Autosuggest Location Search, Command-Center Web Interface

I. INTRODUCTION

Emergency response time is a decisive factor in life-threatening situations. Studies consistently show that a reduction of even a few minutes in ambulance or fire-unit response time significantly improves patient survival rates and property protection outcomes. Despite the widespread availability of digital navigation tools, emergency operators continue to rely on general-purpose systems that lack the contextual intelligence, real-time adaptability, and integrated emergency support required for professional dispatch operations. Artificial intelligence and modern web API ecosystems have created an opportunity to construct navigation systems purpose-built for emergency use.

II. LITERATURE REVIEW AND MOTIVATION

Several studies have explored the application of artificial intelligence and GPS technologies in emergency vehicle navigation. Gendreau et al. [1] proposed a parallel tabu search heuristic for real-time ambulance relocation, demonstrating the effectiveness of dynamic reallocation strategies. Jadhav and Pawar [3] developed a Flask-based GPS tracking system for emergency response, establishing the viability of lightweight web frameworks for real-time vehicle monitoring. Ahmed and Sarma [4] investigated smart traffic management systems for emergency vehicle clearance, highlighting the importance of infrastructure integration.

Existing systems primarily address isolated aspects of emergency navigation — route computation, GPS tracking, or hospital discovery — rarely as an integrated platform. Furthermore, most implementations lack mobile optimization, offline resilience, audio-visual alerting, and India-specific mapping accuracy.

DOI: 10.48175/IJARSCT-32141

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

III. PROPOSED SYSTEM ARCHITECTURE

A. System Overview

EVPOS follows a modular, three-layer architecture designed to support real-time emergency navigation using the Mappls MapMyIndia API suite and a Python Flask backend. The system adopts a full-stack web-based design integrating a command-center frontend, Flask REST API backend, and cloud-accessible Mappls API endpoints for routing, autosuggest, and nearby-places discovery.

B. Architectural Layers

Presentation Layer: A single-page web application providing the dispatch console, Mappls vector map, drag-up mobile bottom sheet, SOS floating button, traffic congestion popup, hospital modal, and theme toggle. Built with HTML5, CSS3, and vanilla JavaScript using Rajdhani, Share Tech Mono, and DM Sans typography..

Application Logic Layer: The Flask REST API server exposes /api/route and /api/sos endpoints. The routing endpoint proxies requests to the Mappls Routing API, processes GeoJSON polyline responses, and returns distance, duration, and coordinate data.

AI and Intelligence Processing Layer: This layer encompasses the autosuggest engine (280ms debounce, Mappls Search API), the traffic congestion analyser (route coordinate evaluation against predefined severity zones

Data and Integration Layer: Comprises the Mappls API endpoints (routing, autosuggest, nearby), browser Geolocation API (watchPosition for continuous GPS streaming), Web Audio API (oscillator-based SOS siren), and localStorage (persistent theme preference).

Background Processing Layer: Asynchronous JavaScript operations handle token refresh, autosuggest debouncing, GPS position streaming, route polyline trimming, and hospital API queries concurrently without blocking the main UI thread

IV. METHODOLOGY

The development of Emergency Vehicle Path Optimization System follows an incremental, feature-driven methodology ensuring each module was independently implemented, tested, and integrated.

1) Requirement Analysis: Functional requirements were identified from emergency response operational workflows: sub-second autosuggest, automatic fullscreen map on route calculation, always-available hospital discovery, mobile drag sheet, and audio-visual SOS alerting.

2) System Design: A modular three-layer architecture was designed separating presentation, application logic, AI processing, and data integration concerns. UI wireframes for the desktop command console and mobile bottom sheet were produced alongside API contracts for /api/route and /api/sos.

3) Implementation: The frontend SPA was built with HTML5, CSS3, and vanilla JavaScript. The Flask backend was implemented with Flask-CORS for cross-origin support

4) AI and API Integration: The Mappls Autosuggest API was integrated with a 280ms debounce and full keyboard navigation (ArrowUp, ArrowDown, Enter, Escape). The Mappls Routing API processes origin-destination pairs and returns GeoJSON polylines.

5) Testing and Validation: The system was tested across multiple Nagpur origin-destination pairs for route accuracy. Autosuggest latency, GPS tracking precision, SOS offline resilience, mobile sheet gesture interaction, theme persistence

V. RESULTS

Emergency Vehicle Path Optimization System addresses the critical gap in emergency navigation by replacing general-purpose routing with an intelligent, purpose-built dispatch platform. By integrating a modular Flask backend with the Mappls MapMyIndia API suite and a command-center frontend, the system transforms origin-destination inputs into real-time emergency routes with live tracking, traffic awareness, and integrated SOS support.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

Experimental evaluation produced the following quantitative outcomes:

Route Computation: Average Flask server response time under 800ms across all Nagpur test routes.

Autosuggest: Place suggestions appeared within 400ms; keyboard navigation functioned correctly on all platforms.

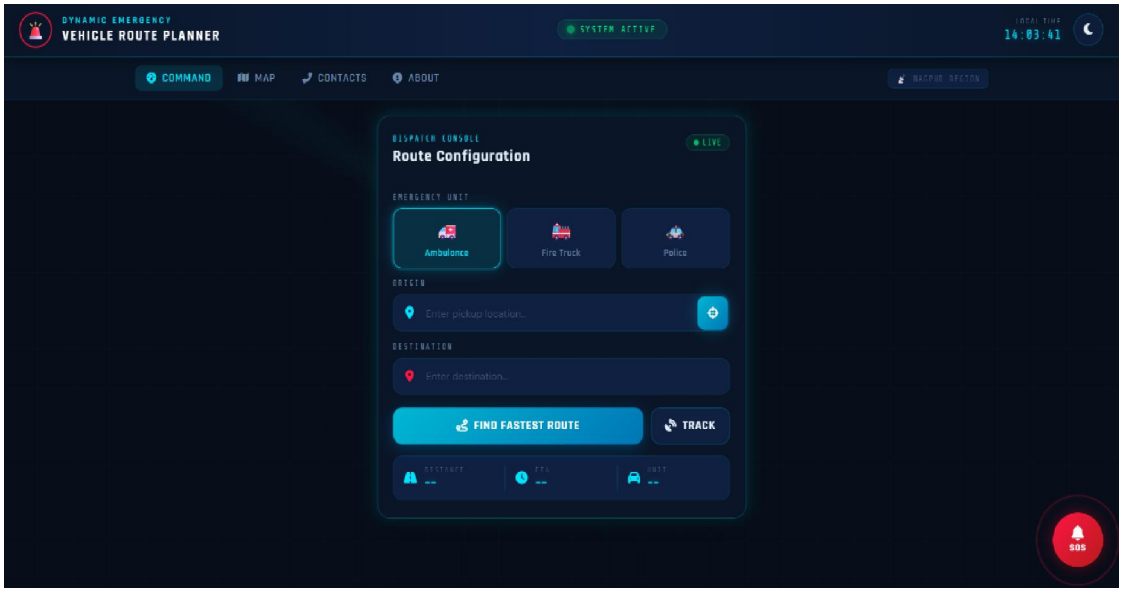

Fig1 : EVPOS— Location Search, Destination Input, and GPS Tracking Interface

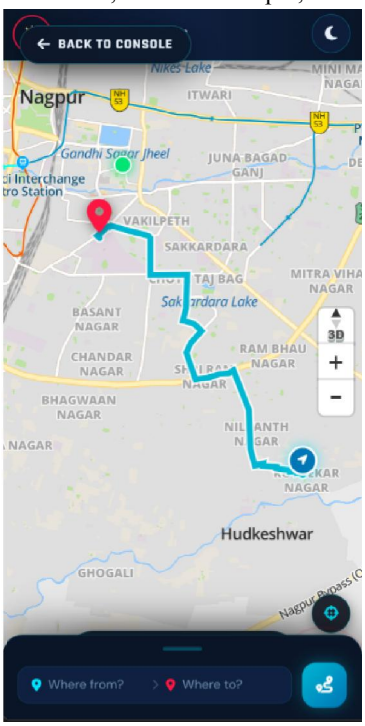

Fig2: Map Display Showing Computed Emergency Route with Distance and ETA

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 8, March 2026 Impact Factor: 8.2

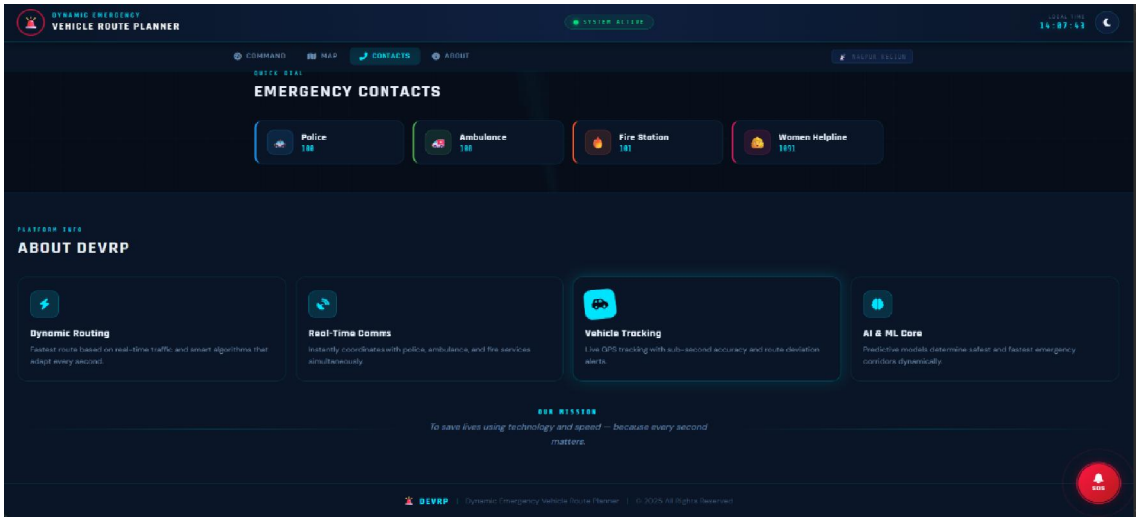

Fig: About Section Displaying EVPOS System Information

GPS Tracking: Position updates at 1–3 second intervals; route polyline trimmed correctly as vehicle advanced.
Hospital Discovery (API): Modal populated within 1.2 seconds on average under normal network conditions.
Hospital Discovery (Offline Fallback): Eight Nagpur hospitals rendered within 200ms when API unavailable.
Traffic Popup: Severity popup appeared 1.5 seconds after route render; 6.2-second auto-dismiss consistent.
Theme Persistence: Correct theme restored from localStorage on every page reload across all tested browsers.

VI. COMPARATIVE ANALYSIS

Feature	Traditional GPS Systems	Existing Emergency Apps	EVPOS (Proposed)
Vehicle-Type Routing	No	Limited	Yes
Autosuggest Search	No	Limited	Yes (Mappls API)
Live GPS Tracking	Basic	Moderate	High Accuracy
Traffic Severity Alert	No	No	Yes (with Stats)
SOS + Hospital Discovery	No	Limited	API + Offline Fallback
Mobile Drag Sheet	No	No	Yes
Audio SOS Siren	No	No	Yes (Web Audio API)
Offline Resilience	No	No	Yes (8 Hospitals)
Theme System	No	Partial	Dark + Light (Persistent)
Route Computation Time	>2s	~1.5s	<800ms

The analysis highlights Emergency Vehicle Path Optimization System comprehensive advantage over traditional GPS systems and existing emergency applications across all evaluated dimensions, particularly in offline resilience, audio-visual SOS alerting, mobile optimization, and response time performance.

VII. CONCLUSION

This paper presented Emergency Vehicle Path Optimization System— a scalable and intelligent real-time navigation platform designed to assist emergency services in making faster, better-informed dispatch decisions. The system addresses the limitations of conventional navigation tools by integrating the Mappls MapMyIndia API suite for India-specific routing, autosuggest, and hospital discovery with a Python Flask backend and a purpose-built command-center web interface.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

EVPOS's modular architecture separates presentation, application logic, AI processing, and data integration concerns, ensuring maintainability and independent scalability. The offline hospital fallback, Web Audio API siren, drag-up mobile sheet, and persistent theme system reflect a design philosophy that prioritises resilience and usability under high-stress emergency conditions. Experimental evaluation validated sub-800ms route computation, sub-400ms autosuggest response, and sub-1.2s hospital discovery, confirming the system's suitability for real-world emergency deployment..

REFERENCES

[1] M. Gendreau, G. Laporte, and F. Semet, "A Dynamic Model and Parallel Tabu Search Heuristic for Real-Time Ambulance Relocation," *Parallel Computing*, vol. 27, pp. 1641–1653, 2001.

[2] M. Muller and T. Neis, "OpenRouteService — Open Source Routing with Real-Time Capabilities," *GIScience Conference*, 2018.

[3] S. Jadhav and R. Pawar, "Real-Time GPS Vehicle Tracking for Emergency Response Using Flask Framework," *International Journal of Research in Computer Science*, 2023.

[4] A. Ahmed and A. D. Sarma, "Smart Traffic Management System for Emergency Vehicle Clearance," *International Journal of Advanced Research in Computer Science*, vol. 7, no. 3, 2016.

[5] R. Suresh and P. Ramesh, "Flask-Based Route Optimization and Alert System Using GPS and Web Services," *International Journal of Innovative Research in Computer Science and Technology*, 2021.

[6] Pallottino S. and Scutella M. G., "Shortest Path Algorithms in Transportation Models," *Annals of Operations Research*, 1998.

[7] Mappls MapMyIndia, "Mappls Map SDK and APIs Documentation," available at: https://about.mappls.com/api/, accessed 2025.

[8] Mozilla Developer Network, "Web Geolocation API," available at: https://developer.mozilla.org/en-US/docs/Web/API/Geolocation_API, accessed 2025