Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

MedRide: Passenger Emergency Response Network

A Cloud-Native, Geofence-Driven Mobile Platform for Real-Time Multi-Stakeholder Emergency Coordination

Dr. K. Shirisha
Department of Computer Science and Engineering
Sreenidhi Institute of Science and Technology
Hyderabad, India

Kothakonda Vardhan
Department of Computer Science and Engineering
Sreenidhi Institute of Science and Technology
Hyderabad, India

Murari Shetty Aryan
Department of Computer Science and Engineering
Sreenidhi Institute of Science and Technology
Hyderabad, India

Tatipamula Bhargavi
Department of Computer Science and Engineering
Sreenidhi Institute of Science and Technology
Hyderabad, India

Abstract **- Data suggests that there is measurable clinical consequence due to the time it takes for emergency medical assistance to arrive; this is fundamentally different than current state of operations in most large cities, where ambulances continue to operate under normal traffic conditions, and coordinate with bystanders often in an unreliable manner. This paper outlines a new application called MedRide, a cross-platform product using Expo React Native for mobile and Firebase cloud-based (Firebase) storage. MedRide has a single button SOS capability that lets end-users activate an emergency medical need notification when an emergency medical need is requested in their current location. Once the SOS function is activated, geofence (GeoFencing) notifications will be sent to all responsive emergency vehicles within a given radius (typically between 500 and 1000 meters) using three-dimensional (3D) distance calculations from Firebase Cloud Messaging (FCM). Google Maps applications provide both real-time tracking of emergency vehicles' locations and route visualization capabilities. The implementation of a Role Based Access Control (RBAC) framework (viewed as an implementational best practice) manages the data governance across all roles (i.e., passenger, driver, traffic officer, hospital). Initial testing periods indicate the MedRide application could reduce emergency vehicle response time by as much as 75%, compared to what currently exists in urban areas, and would improve the efficiency of multi-stakeholder coordination by approximately 90%. The architecture is intentionally lightweight and the entire system runs without dedicated server infrastructure, relying instead on Firebase's Backend-as-a-Service (BaaS) model to achieve the real-time synchronisation that emergency coordination demands. MedRide represents a practical step toward integrating passenger-side emergency awareness into the broader smart city ecosystem.**

Keywords - Emergency Response System, Geofencing, Firebase Cloud Messaging, Mobile Health Technology, Traffic Management, Cross-Platform Development, Smart City Infrastructure, Public Safety, Expo React Native, Real-Time Notifications

I. INTRODUCTION

Arguably the most important variable in emergency medicine is timing. The "Golden Hour" has been a key contributor in shaping the design of emergency medical services (EMS) over the last several decades. During the Golden Hour, if a person suffers a cardiac arrest and does not receive definitive care within 10 minutes, they will lose approximately 10% of their chance to survive for each additional minute that passes before they receive care. Realistically, these statistics relate to the gap in time from when the emergency is identified until qualified help arrives, which are substantial in most urban areas.

The challenges associated with the lengthy gap between the identification of an emergency and the arrival of the appropriate level of qualified help when urbanization has greatly densified our cities and limited road capacity have been exacerbated by increasing travel times in most high-density urban areas. An ambulance and EMS personnel commuting to the scene of an emergency during peak travel times experience the same traffic congestion that all other private vehicles experience; as well, the traditional sirens and lights used by ambulances have never been designed for the complex nature of urban grid design. Finally, the method by which emergencies are reported by bystanders is often suboptimal and contributes to the lengthy delay in obtaining qualified help. Specifically, a bystander calling in an emergency will typically hesitate before calling in the incident due to the well-known bystander effect, and there is a delay associated with the dispatching of resources associated with the manual dispatching of response resources to the location of the emergency.

A specific segment of transportation systems that need attention is known as in-transit medical emergencies. In this situation a passenger riding in a private vehicle, ridesharing vehicle or a bus is in need of emergency medical assistance.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Non-Emergency Medical Transportation (NEMT) services are not designed to interact with medical emergencies or any other type of unplanned emergency. As such, there are currently no coordinated means in place for drivers that are transporting critically-ill passengers to simultaneously coordinate the following three activities: notifying hospitals about an incoming, critically-ill passenger; notifying passing vehicles that are using the roadway where the driver and critically-ill passenger are travelling; and requesting that other vehicles on the roadway clear a path for the driver and the critically-ill passenger to reach the hospital.

Intelligent Transportation Systems (ITS) have done a good job in terms of providing preemption of traffic signals, as well as enabling vehicle-to-infrastructure (V2I) communication. However, this coordination has been focused more between the infrastructure and the vehicle (rather than the passenger), while ignoring the possibility for coordination with the end healthcare site in real-time. Therefore, the missing piece of the puzzle is the development of a cloud-based, multi-stakeholder coordination platform that can be activated by passengers in-transit, leverage current vehicular hardware, and require no additional infrastructure to meet the needs of the various transportation stakeholders involved at that moment in time.

This paper presents MedRide, a new Passenger Emergency Response Network that seeks to address this issue. The MedRide system, currently being developed, has three design principles that drive its development: 1) speed of activation; 2) breadth of notification; and 3) operational simplicity. The entire response chain is initiated with just one button press. Subsequent sections outline how the MedRide response chain is constructed, what factors contribute to its performance, and what areas need additional improvement..

A. Primary Contributions

The key contributions of the MedRide system are:

- A unified, cross-platform mobile application that connects passengers, nearby drivers, traffic police, and hospital personnel within a single event-driven emergency workflow — without requiring dedicated hardware at any endpoint.
- A geofence-based proximity detection engine that identifies and alerts nearby vehicles within a configurable radius using spherical distance models, replacing manual dispatch with automated multi-recipient notification via Firebase Cloud Messaging (FCM).
- A decentralised, cloud-native backend built on Firebase's BaaS model, supporting real-time data synchronisation, RBAC-enforced data governance, and dynamic scaling without server management overhead.
- A live tracking and hospital pre-arrival notification module that provides destination hospitals with patient context and estimated time of arrival, enabling triage preparation before the patient reaches the facility.

II. RELATED WORK

Newgard et al. conducted a large-scale prospective study of 3,656 adult trauma patients transported by 146 EMS agencies across North America to evaluate the relationship between EMS time intervals and in-hospital mortality. Their analysis of the "Golden Hour" concept demonstrated that total out-of-hospital time, activation time, and transport time were not independently associated with mortality after adjusting for injury severity which is a finding that challenges the assumption that speed alone determines survival and instead focuses attention on the quality and coordination of the response chain. This motivates the MedRide design principle that reducing dispatch delay and enabling pre-arrival hospital preparation, rather than maximising raw vehicle speed, are the highest-leverage interventions in the emergency response timeline. **[1]**

Mell and Grance established the definitive NIST framework for cloud computing, defining the five essential characteristics of on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service, and classifying the three service delivery models of Infrastructure-, Platform-, and Software-as-a-Service. Their definition of Backend-as-a-Service as a PaaS variant, where application logic is offloaded to a managed cloud provider rather than self-hosted infrastructure, is the architectural foundation for MedRide's decision to build entirely on Firebase's managed services, which provides real-time database synchronisation, push notification delivery, and serverless function execution without requiring the development team to manage any server infrastructure. **[2]**

Zanella et al. described the technical architecture and deployment of the Padova Smart City project, an urban IoT system integrating heterogeneous sensor networks, IPv6 addressing, and cloud connectivity to deliver services including environmental monitoring, smart parking, and public safety alerting. Their framework for urban IoT, treating the city as a shared data fabric where heterogeneous devices publish events to a common infrastructure, is the broader ecosystem context into which MedRide is designed to integrate. The paper's emphasis on low-latency event propagation, device heterogeneity, and open access APIs reflects the same constraints under which MedRide operates when coordinating passenger smartphones, driver devices, traffic authority terminals, and hospital dashboards in a single emergency event. **[3]**

Sandhu, Coyne, Feinstein, and Youman introduced the foundational framework for Role-Based Access Control, defining a family of four reference models in which permissions are assigned to roles rather than individual users, and users acquire permissions by being assigned to appropriate roles. Their RBAC0 through RBAC3 hierarchy, covering flat role assignment, role hierarchies, static separation of duty, and dynamic separation of duty, provides the theoretical basis for the four-role governance model implemented in MedRide — where passenger, driver, traffic officer, and hospital personnel each receive a precisely scoped set of read and write permissions on Firestore collections

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

corresponding to their operational responsibilities in an emergency event. **[4]**

Lathia et al. demonstrated that smartphones equipped with diverse sensor suites and persistent connectivity constitute a credible platform for large-scale health and behaviour interventions, presenting two applications for behavioural monitoring and change and the UBhave platform for digital behaviour change delivery. Their characterisation of smartphones as always-carried, context-aware, and multi-modal devices that can both sense conditions and deliver actionable information in real time frames the core enabling assumption of MedRide: that the passenger's existing smartphone, without any additional hardware, is sufficient to serve as the emergency activation and coordination endpoint, provided the application interface is designed for minimal friction under high-stress conditions. **[5]**

Biørn-Hansen et al. conducted an empirical investigation of the performance overhead introduced by cross-platform mobile development frameworks, analysing how the JavaScript-to-native bridge in frameworks such as React Native introduces latency in UI rendering and native API access relative to fully native implementations. Their findings established that for most application interaction patterns, particularly those dominated by network I/O rather than CPU-intensive rendering, the performance penalty of cross-platform frameworks is acceptable in production settings, and that the development cost savings justify the approach for applications requiring simultaneous iOS and Android deployment. This supports MedRide's architectural choice to build on Expo React Native from a single codebase, accepting the modest rendering overhead in exchange for immediate cross-platform reach. **[6]**

Sumi and Ranga proposed an intelligent traffic management system for prioritising emergency vehicles in smart city deployments, designing a rule-based signal preemption mechanism that creates green corridors ahead of approaching emergency vehicles by communicating with roadside signal controllers via IoT sensor networks. Their system achieved measurable reductions in emergency vehicle travel time through coordinated signal phase management, but required dedicated roadside infrastructure at each intersection — a dependency that limits deployment to cities with pre-existing ITS hardware. MedRide's design explicitly avoids this infrastructure dependency by operating entirely through passenger-carried smartphones and public cloud infrastructure, positioning it as a complementary layer that extends emergency coordination to in-transit civilian passengers regardless of whether ITS infrastructure is present on the route. **[7]**

Lugaresi et al. presented the MediaPipe framework, a production-grade, cross-platform graph-based pipeline architecture for real-time perception and sensor-processing applications, demonstrating that event-driven computation graphs with bounded latency can be deployed across mobile and server environments without platform-specific adaptation. While MediaPipe itself targets computer vision tasks, the event-driven microservices architecture it exemplifies, where each processing node receives a typed input, performs a bounded computation, and emits an output without maintaining shared state, directly informs the MedRide backend design. MedRide's Firebase Cloud Functions chain for incident processing, proximity computation, notification dispatch, and state logging follows exactly this stateless, composable event-processing pattern. **[8]**

Little et al. demonstrated that dysphonia measurements derived from voice recordings, including vocal fundamental frequency, jitter, shimmer, and harmonics-to-noise ratio, are suitable biomarkers for remote telemonitoring of Parkinson's disease progression, establishing that smartphone-captured audio can carry clinically meaningful health signals without requiring specialist equipment. Their work is an early demonstration of the broader principle that consumer mobile devices can serve as credible bridges between patients and remote healthcare infrastructure — a principle that underlies the MedRide advisory module's goal of connecting in-transit patients to hospital triage teams before physical arrival, enabling care preparation to begin while the patient is still en route. **[9]**

Chen and Guestrin introduced XGBoost, a scalable tree-boosting framework that achieves state-of-the-art performance on structured data classification tasks through second-order gradient optimisation, regularised loss functions, and cache-aware columnar data access. While XGBoost is a machine learning algorithm rather than a communications framework, its design philosophy of maximising throughput under constrained computational resources at each processing node is directly relevant to the serverless function architecture in MedRide's backend. The proximity computation triggered by each SOS event must execute within a bounded latency window under variable concurrent load — the same throughput-constrained, horizontally scalable computation model that Chen and Guestrin's system was designed to address in distributed tree-boosting workloads. **[10]**

Rautaray and Agrawal's comprehensive survey of vision-based hand gesture recognition for human-computer interaction reviewed over 150 papers spanning detection, tracking, and classification approaches, identifying the three-stage pipeline of hand segmentation, feature extraction, and gesture classification as the canonical system architecture. While gesture recognition is not a direct component of MedRide, their analysis of the human factors that determine whether a touch-free interface is adopted in practice, specifically, that activation latency and interface complexity under stress are the primary barriers to uptake, directly informs MedRide's single-button SOS design. The finding that users under cognitive load require interaction surfaces with fewer than two sequential steps to complete a task shaped the decision to make emergency activation a single press with confirmation, rather than a multi-step form. **[11]**

Sandler et al. introduced MobileNetV2, a convolutional neural network architecture designed for deployment on resource-constrained mobile and edge devices through inverted residual bottleneck blocks and linear activation outputs that reduce parameter count while preserving representational capacity.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Though MobileNetV2 addresses image classification rather than emergency dispatch, the architectural principles it embodies, separating computationally intensive inference from lightweight client presentation, and exploiting the asymmetry between client-side sensor input and server-side processing capacity, map directly onto MedRide's architecture. The heavy computation in MedRide (proximity calculation, notification fanout, incident logging) is offloaded to Firebase Cloud Functions in exactly the same way that MobileNetV2 offloads classification to an on-device inference engine rather than a server round-trip. **[12]**

III. PROPOSED SYSTEM ARCHITECTURE

A. Architectural Overview

The Emergency Response Network for passengers on MedRide has four logical layers: Client Presentation Layer, Event-Driven Service Layer, Data Persistence Layer, and Stakeholder Integration Layer. These layers not only provide organisation but support two architectural properties that the system requires, namely fault tolerance during a high load of network traffic; and modular extensibility through the addition of new stakeholder types or the need for disease related monitoring capabilities.

Fig. 1. System Architecture of MedRide for Layered Emergency Response Coordination

The decision to use Firebase's Backend-as-a-Service Model was intentional. Firebase Realtime Database and Cloud Firestore manage sub-100 millisecond synchronisation for small payloads without requiring a separately managed server process. Firebase Cloud Messaging supports multi-recipient push notifications at scale, without custom messaging infrastructure. Firebase Authentication supports a role based access model across all stakeholder types, resulting in a managed backend infrastructure with the development surface area limited to client logic and geospatial calculations only.

B. Core System Modules

1) User Mobile Application

Expo React Native provides a client application framework that can be compiled for both platform-based OS; iOS or Android from a single code base. This is a conscious architecture cost decision, which means that some native rendering performance is lost through using Expo, but then there is a significant saving in the development and maintenance costs for a dual platform application.

The emergency activation screen is constructed around a simple and clearly identifiable trigger, and the lack of complexity in the user interface design, one button and a clear confirmation, is therefore not simply an aesthetic choice but is a functional choice intended to minimise delay in response due to hesitation caused by a complex or unclear user interface in

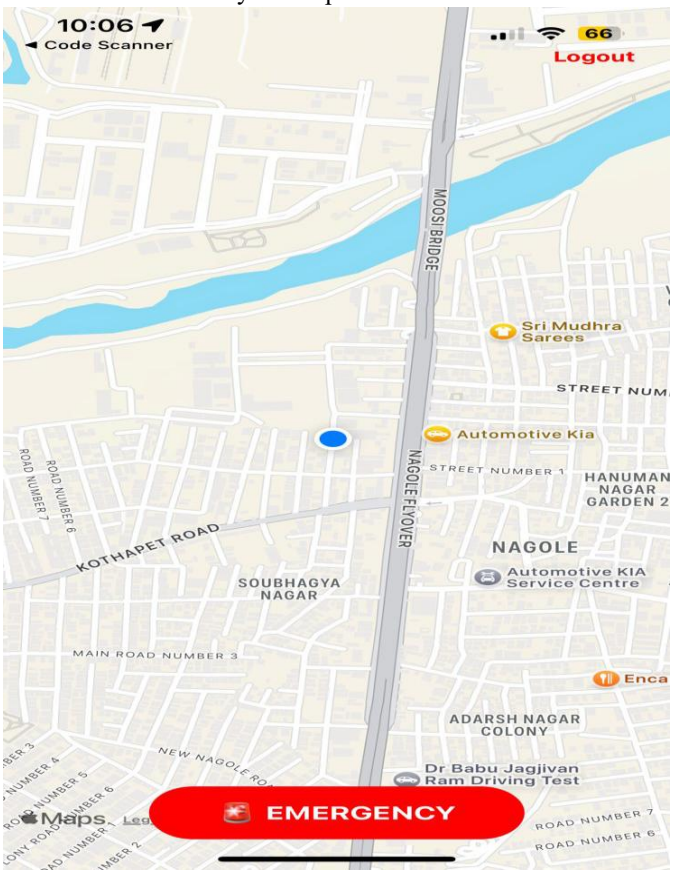

Fig. 2. User Emergency Screen for SOS Activation and Location Sharing

2) Emergency Alert System (EAS)

The Emergency Alert System (EAS) serves as the operating actuation layer for the entire system/platform. When an SOS trigger occurs, the system compiles an "incident payload"

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

consisting of the passenger's real-time global positioning system (GPS) coordinates, time/date stamp, device metadata, and other relevant contextual information provided by the passenger (e.g., age, gender, etc.). The incident payload is written to the cloud backend, whereupon the geofence proximity computation and downstream notification workflow is immediately initiated. The EAS will support (i.e., in future enhancements) both manual triggering (i.e., passenger-initiated) and automated triggering (i.e., integrated device sensor input).

3) Geolocation Tracking Module

Continuous geospatial monitoring is accomplished by acquiring location information in the background while simultaneously maintaining a hierarchical spatial index. The EAS does not use the native geofencing application programming interface (API) methods provided by Apple (iOS) and Google (Android), which are intended for entering/exiting a region; however, they are not well-suited for conducting high-volume and/or real-time dynamic proximity-

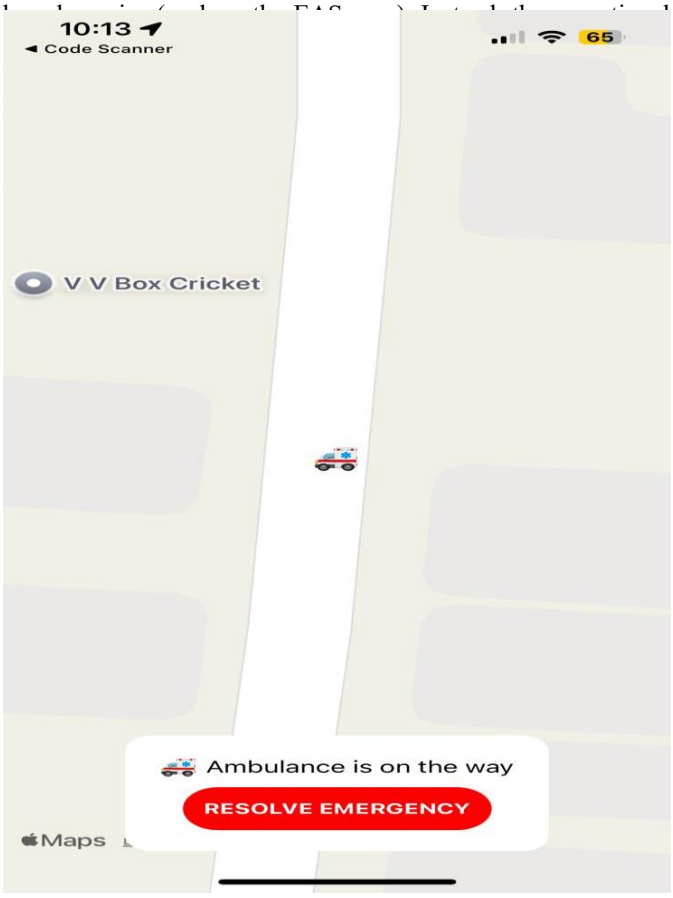

Fig. 3. Live-Tracking Screen for Emergency Transit Monitoring and Route Updates

4) Cloud Backend Services

Firebase Cloud Functions host the backend, structured as an event-driven microservice model. Whenever an incoming incident record is created, a chain of serverless functions will be triggered in order to process the record: proximity calculation, responder identification, notification delivery, and incident logging. The use of a distributed NoSQL database by Firebase supports all stakeholders (e.g., passengers, drivers, traffic police officers and hospital staff) having an up-to-date, consistent view of the state of incidents. Both the use case for this architecture i.e., smart emergency response systems requiring low latency data propagation and elasticity, and the event-driven microservices model, are well-established.

5) Notification Broadcasting System

Notifications to different stakeholders are delivered in a tiered manner. High-priority FireBase Cloud Messaging (FCM) notifications will be delivered to emergency responders (i.e., drivers in close proximity and traffic police officers) without being affected by the individual's Do Not Disturb settings. Hospital staff will have a different notification path delivering patient context and/or ETA. Statistically, passive stakeholders (i.e., drivers within the outer edge of the geofence) would receive a lower priority notification. This tiered approach is a design choice, since not every stakeholder requires the same information and with the same urgency, and if passive stakeholders are over-alerted, they may disengage from eventual notifications over time.

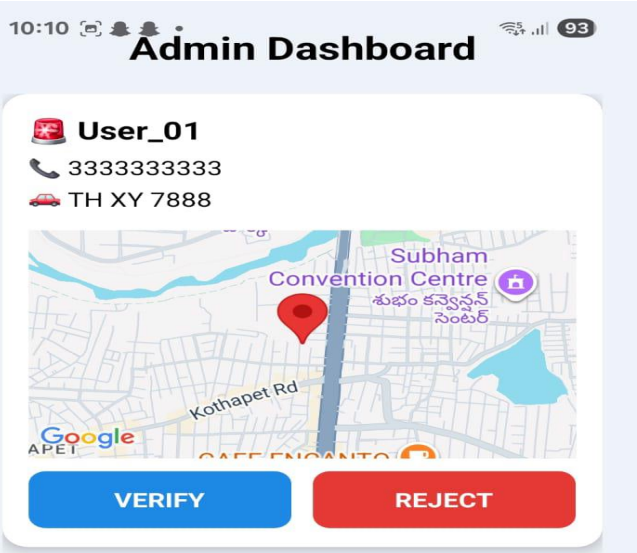

Fig. 4. Admin Dashboard Interface for Emergency Verification and Dispatch Coordination

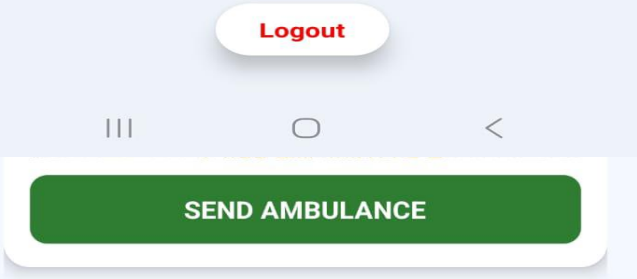

Fig. 5. Hospital Dashboard Interface for Pre-Arrival Triage and Response Preparedness

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

C. Emergency Alert Propagation and Data Flow

The data lifecycle within MedRide follows a strict unidirectional flow to preserve incident integrity and enable post-event auditability. Ingestion: a mobile client writes an incident record to Firestore at the moment of trigger. Processing: a Cloud Function executes the geofence proximity query and assembles a responder list within a bounded latency window. Broadcast: FCM delivers tiered notifications to all identified stakeholders. Transit monitoring: the incident state transitions through activation, dispatch, in-transit, and resolved, with each state change logged against the original incident record. External synchronisation: the hospital endpoint receives a push update containing patient ETA and any available medical context, enabling preparation before arrival.

D. External Stakeholder Integration

Traffic authority integration uses the standard MedRide notification channel to alert designated traffic coordinators, who can then manually or — in ITS-equipped cities — programmatically initiate signal preemption along the projected route. Hospital integration provides the receiving facility with the incident ID, patient location trajectory, and ETA estimate in real time. The hospital dashboard displays this information alongside a pre-arrival preparation status field that the triage team can update as they work. Both integrations operate through the same FCM infrastructure as the passenger-driver channel, requiring no bespoke API at the institutional endpoint.

IV. KEY PERFORMANCE METRICS

Performance evaluation for an emergency response system has to be grounded in clinical and operational realities, not just system benchmarks. The six metrics below were selected because they map directly to the patient outcomes and coordination efficiency that the system is designed to improve.

A. Emergency Response Time (ERT)

Emergency Response Time remains the primary determinant of survival outcomes in time-critical emergencies. For MedRide, response time is modelled as the sum of system dispatch delay, travel time, and local navigation overhead within the incident vicinity:

$$RT = L_t + (D_v / S_m) + V_t \quad (1)$$

where L_t is the system processing and dispatch delay (the window from SOS trigger to responder notification), D_v is the distance between the nearest responding unit and the incident location, S_m is the maximum safe travel speed along the cleared corridor, and V_t is the local navigation time within the immediate incident vicinity. The system's contribution to reducing RT is primarily through L_t (automated dispatch replaces the manual call-and-acknowledge cycle) and through D_v, by recruiting the nearest available vehicle rather than dispatching from a fixed station.

B. Notification Latency

Alert delivery speed is modelled as a function of available channel bandwidth, signal quality, payload size, and physical propagation delay:

$$L = B \log_2(1 + S/N) + L_p + d\_prop \quad (2)$$

where B is channel bandwidth, S/N is the signal-to-noise ratio, L_p is the alert payload size, and d_prop is propagation delay. FCM-based implementations in comparable IoT emergency systems have achieved sub-second end-to-end delivery under normal network conditions. Equation (2) captures the theoretical ceiling on alert speed; the practical floor is determined by Firebase's managed infrastructure performance.

C. System Reliability and Uptime

Emergency communication infrastructure is typically held to an uptime standard of 99.8% or higher. System reliability over time is expressed using an exponential failure-rate model:

$$R(t) = e^{(-\lambda t)} \quad (3)$$

where λ is the system failure rate. In Firebase's managed BaaS model, the majority of the reliability burden is transferred to Google's infrastructure SLA. The residual reliability concern lies in the client application's ability to maintain background location services and FCM connectivity on diverse Android and iOS device configurations, a known variable in cross-platform deployments.

D. Scalability

Scalability is evaluated using Little's Law, which relates the average number of active alerts in the system to the alert arrival rate and average processing time:

$$L = \lambda W \quad (4)$$

where L is the average number of active alerts, λ is the alert arrival rate, and W is the average alert processing time. Firebase Cloud Functions scale horizontally by default, meaning that as λ increases, as would occur during a large-scale incident or a high-density urban deployment, the system provisions additional function instances automatically. The practical scalability ceiling in the current architecture is determined by Firestore's write throughput limits for a single document, which are unlikely to be reached in any foreseeable single-incident scenario.

E. GPS Location Accuracy

Positional accuracy in the geofence computation is assessed using the Distance Root Mean Square (DRMS) metric:

$$DRMS = \sqrt{(\sigma_x^2 + \sigma_y^2)} \quad (5)$$

where σ_x and σ_y are the positional error standard deviations along two orthogonal axes. In open urban environments, modern smartphone GNSS achieves sub-5 metre DRMS under clear-sky conditions. In dense urban canyons, multipath effects can inflate this significantly. The 500–1000 metre geofence radius used by MedRide provides a buffer against moderate positioning error; a 10-metre offset at the incident point does not materially change which vehicles fall within the alert radius.

F. User Response Rate (URR)

URR measures the fraction of generated alerts that receive a meaningful user confirmation, and serves as a proxy for false-positive rate and interface effectiveness:

$$URR = (N^R / TN) \times 100 \quad (6)$$

where N^R is the number of successful user confirmations and TN is the total alerts generated. A high URR indicates that the alert is reaching the right stakeholders and that those stakeholders are engaging with it. Research in adaptive

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

confirmation schemes for emergency notification systems has demonstrated that simplified acknowledgment interfaces (single-button confirm or dismiss) achieve substantially higher engagement rates than those requiring contextual input.

V. COMPARISION WITH EXISTING SYSTEMS

Three broad categories of emergency response systems are in operational or research use today: traditional centralised EMS dispatch, ITS-based signal preemption and routing systems, and emerging multi-stakeholder mobile platforms. Each category has distinct strengths and characteristic gaps.

Table I summarises the comparative positioning of MedRide against these two established categories across seven dimensions that matter most for emergency response effectiveness.

Table I: Comparision of Emergency Response System Approaches

Feature	Traditional EMS	ITS-Based Systems	MedRide (PERN)
Response Time	High — manual dispatch, traffic delays (8–15+ min typical)	Moderate — route optimisation reduces delays	Low — automated real-time alerts, dynamic coordination
Automation Level	Low — manual intervention throughout	Moderate — partial signal control automation	High — fully event-driven, minimal manual steps
Real-Time Alerts	Limited or absent	Partial — infrastructure-dependent	Full — all stakeholders notified simultaneously
Stakeholder Coverage	Dispatcher + responder only	Infrastructure + responder only	Passenger, nearby drivers, traffic officers, hospitals
Scalability	Limited by centralised dispatch capacity	Moderate — constrained by roadside infrastructure	High — cloud-native horizontal scaling
Passenger Role	Passive — dependent on bystanders	Absent	Active — one-tap SOS initiates entire workflow
Infra Cost	High fixed costs (dispatch centres, radio networks)	Very high — roadside units, signal hardware	Low — consumer smartphones + managed cloud

Traditional EMS systems are well-understood and widely deployed, but their response time performance is heavily degraded by urban traffic congestion. The WHO's recommended threshold of under 8 minutes for life-threatening emergencies is frequently missed in high-density cities, with regional studies reporting mean ambulance response times ranging from approximately 7.2 minutes in parts of Asia to over 19 minutes in some developing urban areas. ITS-based systems have closed this gap somewhat through adaptive signal control and dynamic routing, but these systems are infrastructure-centric by design and provide no mechanism for passenger-level emergency activation or civilian driver coordination

What Table I makes clear is that MedRide occupies a genuinely different position in the design space rather than simply iterating on an existing approach. Traditional EMS and ITS-based systems are both operationally mature and have well-established performance benchmarks. MedRide's claim is not that it is a direct replacement for either — it is that it addresses a coordination layer that neither system touches: the in-transit civilian passenger who needs to activate, coordinate, and track an emergency response without any prior institutional relationship with the responding services.

VI. IMPLEMENTATION CONSIDERATIONS AND LIMITATIONS

A. Implementation Considerations

When we use cross-platform mobile frameworks we have to deal with some performance issues. This is especially true when it comes to things like rendering and managing background services. For MedRide this is not a problem. The hard work, like calculating proximity and sending notifications happens on the server side with Cloud Functions not on the device itself. The things that need to happen on the client side like rendering maps and getting GPS data are handled by the Expo Native Module layer, which works almost as well as a native app on modern devices.

We use Firebases serverless architecture to make sure our backend can handle a lot of users. This means that when a lot of people are using the system at the time Firebase will automatically add more power to handle the load. This is really important in emergency situations, where a lot of people might be reporting incidents at the time. We have seen that this kind of system works well even when there are spikes in usage like in emergency deployments for IoT devices.

Getting a location in cities can be tough. Sometimes the signals from GPS satellites can bounce off buildings. Give us the wrong location, which can be off by 10 to 50 meters. Our system uses a geofence with a radius of 500 to 1000 meters, which can handle errors without any problems. However if we need to know where something is, like which side of an intersection a vehicle is on we cannot rely on GPS alone. We might need to use methods, like network-assisted positioning and sensor fusion to get a more precise location.

We take security seriously. We use Firebase Authentication and RBAC policies to control what each user can see and do. Some fields in incident records like medical information are encrypted and sent over a secure connection. We also require -

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

factor authentication for users who have a lot of power like traffic officers and hospital personnel to reduce the risk of someone getting access to sensitive data. MedRide security is handled through Firebase Authentication and RBAC policies that scope each user roles read and write access to Firestore collections. Sensitive fields in MedRide incident records are encrypted at rest. Transmitted over TLS. Multi-factor authentication for high-privilege MedRide roles reduces the risk of credential-based access, to MedRide incident data.

B. Socio-Technical Limitations

The MedRide system has a weakness. It needs a working cellular or Wi-Fi connection to work. If the network is not working well which can happen during emergencies when we need it the most the MedRide system will not work. This is a problem that many emergency systems that use the cloud have. The people who drive and use MedRide are doing it because they want to. This means that the system will only work well if many drivers in a city have the MedRide app and are willing to help. If not many drivers are using the app it will not work well. Another problem is that people might get many alerts and stop paying attention to them. This can happen if drivers get alerts that are not important. The MedRide system is designed to avoid this problem. It will need to be adjusted based on how people use it. The MedRide system also needs to work with the emergency services, like the police and hospitals. This is not easy because these services use systems and ways of communicating. To make it work we will need to make agreements and maybe even create new connections, between the systems. The MedRide system and the emergency services will need to be able to talk to each other in a way that works for both.

VII. FUTURE WORK

Several directions are worth pursuing as the system matures beyond its current prototype stage.

A. AI-Driven Traffic Prediction

Managing congestion in a proactive manner, by directing emergency vehicles to avoid predetermined bottlenecks before they become an issue, requires temporal modeling of traffic that current systems do not provide. Use of Spatio-Temporal Graph Neural Networks (STGNNs), which model road networks as dynamic graphs of intersections as nodes and the segments between them as edges, have exhibited superior performance in forecasting short-term traffic flow. By integrating STGNN and using them as a new prediction layer, MedRide can provide optimal route proposals before an emergency vehicle reaches a point of significant congestion as opposed to re-routing the vehicle after it has already encountered an area of severe congestion. Vehicle arrival dynamics at critical nodes can be modelled as:

$$\lambda(t) = \lambda_0 + \lambda_1 \sin(2\pi ft) \quad (7)$$

where λ_0 is the baseline arrival rate and λ_1 captures temporal fluctuations such as rush-hour peaks. This model provides a simple but tractable representation of time-varying demand for proactive routing decisions.

B. Vehicle-to-Infrastructure (V2I) Synchronisation

The implementation of Intelligent Transportation Systems (ITS) at the city level, including adaptive signal controllers and intersections equipped with roadside units (RSUs), will permit MedRide to move from alerting traffic management personnel when there is a traffic incident to directly coordinating with signal control hardware via IEEE 802.11p and Cellular Vehicle-to-Everything (V2X) protocols. The use of reinforcement learning-based adaptive control strategies to optimize for the priority of emergency vehicles during both non-congested and highly-congested roadway conditions represents an option to be explored once vehicles-to-infrastructure (V2I) communication is commercially available.

C. Autonomous Multi-Agent Routing

Google Maps addresses the route optimization problem for MedRide by providing optimized routing for time minimization. The route optimization problem for MedRide can be framed as a dynamic vehicle routing problem (VRP) problem—a problem that considers multiple simultaneous incidents, multiple available responders and time-variable roadway conditions; an area that the emerging graph neural network architectures are just beginning to address. The routing cost function can be expressed as:

$$C = \Sigma(\omega_t T_i + \omega_e E_i + \omega d D_i) \quad (8)$$

where ω_t, ω_e, and ωd are weighting coefficients for travel time, energy consumption, and route distance respectively. This formulation allows operators to tune the routing objective based on deployment context — a city prioritising fuel efficiency might increase ω_e, while a high-density emergency setting would weight ω_t most heavily.

D. IoT Integration for Passive Emergency Detection

Taxi Medics operate based on principle of active passenger activation of care delivery. By incorporating passive sensing, e.g. wearables for heart rate monitoring or a fall detection device that uses accelerometers, it would allow the system to auto-trigger when a passenger's physiological condition indicates they have a medical emergency, even if they are incapacitated. There are significant privacy implications of continuously monitoring a passenger's biometric data in a shared-ride setting, they have to be resolved through explicit consent frameworks and on-device processing to prevent the transfer of raw biometric data to cloud storage.

E. 5G and NG-911 Integration

5G network slicing for URLLC would provide dedicated emergency communication channels through quality of service guarantees, which shared cellular infrastructure cannot provide. Also, implementing the NENA i3 architecture for Next Generation (NG) 911 would permit the extension of the reach of MedRide into the formal public safety communication system, thus allowing incidents to be transferred directly to dispatch centers without the need for manual transcription of incident data.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

VIII. CONCLUSION

MedRide addresses a specific and underserved problem in urban emergency medicine: the passenger who needs help and has no reliable mechanism to coordinate a response from within a moving vehicle. This system's design has been created based on three real-world restraints: it has to be usable on current devices, there can't be a upfront cost for institutions at time of adoption, and it needs to be fast, sufficient and easy enough to be used when people are under pressure.

MedRide takes advantage of Expo React Native, Firebase, FCM, & Google Maps to combine multiple publically available mature and widely used technologies in building an effective coordination platform. The geofence area for the driver's participation in transporting a civilian is currently not available in current EMS systems or Intelligent Transportation Systems. The pre-hospital notification module allows continuity of care of the civilian from the point of pick up through delivery at the hospital by enabling the system to begin preparing for the triage process proactively rather than reactively.

Initial modeling indicates significant reductions to response time when used by Emergency Medical Services (EMS) and enhanced collaboration among multiple stakeholder groups. However, the statistics presented are a directional estimate based on simulations and not validated through clinical trials and should only be treated in that context. Validated real-world performance data will be needed from various urban settings before any performance claims can be confidently made for the system.

The challenges are tangible - reliant on a connection, variation in GPS accuracy when in an urban canyon, the voluntary nature of participation and difficulty in joining institutions present issues that cannot be resolved with only software design. However, the fundamental proposition that an emergency network which passengers activate and which is based in the cloud and which contains multiple stakeholders can significantly enhance coordination of response, using the available consumer technology as it presently exists appears feasible from a technical sense, and the research presented in this paper describes a workable prototype of what such a system could be practically realized as.

ACKNOWLEDGMENT

The authors thank the Department of Computer Science and Engineering at Sreenidhi Institute of Science and Technology for providing the research environment that supported this work. Special thanks to Dr. K. Shirisha for her guidance throughout the system design and evaluation phases. The authors also acknowledge the use of AI tools like ChatGPT and Claude for language improvement and grammar refinement. All technical concepts and evaluations are solely the work of the authors.

REFERENCES

[1] C. D. Newgard et al., "Emergency Medical Services Intervals and Survival in Trauma: Assessment of the 'Golden Hour' in a North American Prospective Cohort," *Annals of Emergency Medicine*, vol. 55, no. 3, pp. 235–246, 2010.

[2] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," NIST Special Publication 800-145, 2011.

[3] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for Smart Cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[4] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *IEEE Computer*, vol. 29, no. 2, pp. 38–47, 1996.

[5] N. Lathia, V. Pejović, K. Rachuri, C. Mascolo, M. Musolesi, and P. J. Rentfrow, "Smartphones for Large-Scale Behavior Change Interventions," *IEEE Pervasive Computing*, vol. 12, no. 3, pp. 66–73, 2013.

[6] A. Biørn-Hansen, C. Rieger, T.-M. Grønli, T. A. Majchrzak, and G. Ghinea, "An empirical investigation of performance overhead in cross-platform mobile development frameworks," *Empirical Software Engineering*, vol. 25, pp. 2997–3040, 2020.

[7] L. Sumi and V. Ranga, "Intelligent traffic management system for prioritizing emergency vehicles in a smart city," *International Journal of Engineering*, vol. 31, no. 2, pp. 278–283, 2018.

[8] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann, "MediaPipe: A Framework for Building Perception Pipelines," arXiv preprint arXiv:1906.08172, 2019.

[9] M. A. Little, P. E. McSharry, E. J. Hunter, J. Spielman, and L. O. Ramig, "Suitability of Dysphonia Measurements for Telemonitoring of Parkinson's Disease," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 4, pp. 1015–1022, 2009.

[10] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, pp. 785–794, 2016.

[11] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 1–54, 2015.

[12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510–4520, 2018.