2025 Volume: 01 Issue: 06 www.ijrpa.com ISSN 2456-9995 Review Article

International Journal Research Publication Analysis

Page: 1-27

RESCUENOW – EMERGENCY MOBILE APP

***Geo Jeevan Raj C.**

Department: Artificial Intelligence and Machine Learning. Institution: Sri Shakthi Institute of Engineering and Technology.

Article Received: 13 October 2025
Article Revised: 03 November 2025
Published on: 23 November 2025

***Corresponding Author: Geo Jeevan Raj C.**
Department: Artificial Intelligence and Machine Learning. Institution: Sri Shakthi Institute of Engineering and Technology.

DOI: https://doi-doi.org/101555/ijrpa.8829.

ABSTRACT

Emergency medical services are critically hampered by delays from manual dispatch processes, high rates of fake calls, and inaccurate location information, leading to preventable loss of life and inefficient resource allocation. This project addresses these challenges by designing and implementing "RescueNow", a modern, real-time emergency ambulance management system. The solution comprises a dual-app ecosystem: a "People's App" for one-tap emergency requests with mandatory photo verification and GPS location, and a "Driver's App" for instant notification, request management, and real-time patient tracking.

Developed using the Flutter framework for cross-platform compatibility, the system is powered by a robust cloud backend leveraging Firebase for real-time database synchronization, authentication, and push notifications, alongside Cloudinary for scalable image storage. The methodology follows an Agile MVP approach, resulting in a functional prototype that was rigorously tested in a simulated environment.

The implemented system successfully demonstrates a significant reduction in emergency response time from over ten minutes to under five, eliminates fake calls through photo evidence, and achieves 100% location accuracy via GPS. The project validates the feasibility of using modern mobile and cloud technologies to create a scalable, efficient, and reliable emergency dispatch system, offering a marked improvement over traditional methods and holding significant potential to save lives by ensuring faster and more accurate medical assistance.

LIST OF FIGURES

FIGURE NO TITLE PAGE NO

CHAPTER 1 INTRODUCTION

In emergency medical services, the collection and management of incident data have traditionally been carried out through voice-centric, manual systems reliant on operator intervention. These processes—ranging from verbal descriptions of the emergency to ambiguous location details—are often fragmented across different communication channels between the caller, the dispatch center, and the ambulance driver. This conventional approach leads to critical delays, inefficient resource allocation due to fake calls, and significant

difficulty in coordinating a swift, accurate response essential for saving lives. As public safety services move toward data-driven methodologies, there is an increasing demand for an integrated system capable of unifying, verifying, and acting upon emergency data in real-time. The **RescueNow Emergency Ambulance Management System** is designed to address these challenges by serving as a unified, mobile-first platform for intelligent emergency dispatch and coordination. RescueNow combines real-time GPS location data, photographic evidence, and instant push notifications into a single interoperable framework, enabling emergency responders and the public to interact within a reliable, synchronized, and efficient digital ecosystem.

The platform follows a modular, dual-app architecture that emphasizes speed, reliability, and scalability. It employs a cross-platform mobile frontend built with **Flutter** and **Dart**, ensuring a consistent user experience on both Android and iOS devices. The backend is powered by **Google's Firebase** suite, utilizing **Cloud Firestore** for real-time NoSQL database synchronization, **Firebase Authentication** for secure access, and **Firebase Cloud Messaging (FCM)** for instant, low-latency push notifications. For handling visual data, the system integrates with **Cloudinary**, a scalable cloud-based image management service, to store and deliver photographic evidence of emergencies.

Verification and real-time data are deeply embedded within RescueNow's core functionalities. The mandatory photo-capture workflow serves as a powerful tool for emergency verification, allowing dispatchers and drivers to visually assess the situation before committing resources. The automatic capture and transmission of GPS coordinates via the **Geolocator** service eliminates location ambiguity, while the **Google Maps SDK** provides drivers with precise, interactive navigation to the incident scene. This data-first approach not only improves response accuracy but also enables a transparent and coordinated process, allowing the public to track the assigned ambulance in real-time.

RescueNow is designed with modern security and data privacy principles at its foundation, employing HTTPS for all data-in-transit and leveraging Firebase Security Rules for granular, role-based access control to safeguard sensitive information. The system architecture is built to be scalable, capable of supporting city-wide deployment for public and private emergency service providers.

In essence, RescueNow represents a transformative step in public emergency response

infrastructure. By integrating real-time data, mobile-first design, and scalable cloud technologies, it provides a powerful digital ecosystem for managing medical emergencies. The system enhances operational transparency, accelerates response times, and strengthens the foundation for a more effective and reliable emergency medical service, directly contributing to the goal of improving public health and safety outcomes.

CHAPTER 2 LITERATURE REVIEW

The literature on emergency response systems and mobile health (mHealth) platforms extensively documents the limitations of traditional, voice-centric dispatch models and the urgent need for integrated, data-driven solutions. Conventional approaches—where emergency requests, location details, and incident specifics are communicated verbally between a caller, a dispatch operator, and an ambulance driver—are repeatedly reported as inefficient, error-prone, and lacking in real-time coordination. Studies in the domain highlight common problems such as communication delays, misinterpretation of addresses, the inability to verify the authenticity of a call, and a complete lack of transparency for the person awaiting aid, all of which impede a swift and effective emergency response. To address these weaknesses, research has increasingly advocated for the adoption of mobile technologies that leverage GPS, real-time data synchronization, and multimedia capabilities to create a more reliable and efficient emergency ecosystem. A growing body of work emphasizes the value of combining mobile-first design with modern cloud-based architectures—such as real-time NoSQL databases (e.g., Firebase Firestore), serverless functions, and scalable media storage—to achieve instant communication and robust data management. Comparative analyses in the literature show that platforms adopting such architectures significantly reduce dispatch time and improve the accuracy of the information relayed to first responders. The use of real-time databases is particularly noted for its ability to create a common operational picture, where the status of an emergency and the location of resources are synchronized across all stakeholders' devices with sub-second latency.

Another active area of research concerns the application of multimedia for incident verification. The high prevalence of fake or non-emergency calls is a well-documented drain on EMS resources. Studies have proposed the use of live video streaming or mandatory photo submission as a mechanism to authenticate emergencies before dispatching an ambulance. These studies report a substantial reduction in the allocation of resources to false alarms,

thereby ensuring that critical assets remain available for genuine life-threatening situations. However, they also highlight challenges related to user privacy, data storage costs, and the potential for delays if the media upload process is not optimized for low-bandwidth environments.

The literature also documents the rising importance of GPS technology as a fundamental component of modern EMS. Research consistently demonstrates that the automatic transmission of GPS coordinates from a caller's smartphone is vastly superior to verbal descriptions of a location, drastically reducing the time drivers spend searching for an incident scene. The integration of GPS data with mapping services like Google Maps to provide turn-by-turn navigation is now considered a standard for effective response. Studies combining GPS tracking with algorithms to identify the nearest available unit have shown further improvements in optimizing resource allocation and minimizing travel time. Security, privacy, and accessibility are recurring themes in mHealth literature. Researchers recommend end-to-end encryption (HTTPS/TLS), secure authentication mechanisms, and role-based access control (RBAC) to protect sensitive health and location data. For emergency applications, a frictionless user experience is paramount; therefore, anonymous access or one-tap request features are often proposed to lower the barrier to use during a high-stress event. Implementation studies favor lightweight, cross-platform frameworks like Flutter or React Native because they balance rapid development with native performance, allowing for a single codebase to serve a wide range of devices.

Finally, comparative assessments between legacy voice-based systems and integrated, mobile-first platforms consistently report measurable benefits: reduced time-to-dispatch, higher accuracy in resource allocation, elimination of navigational errors, and improved situational awareness for all parties. The convergent message across the literature is clear—bringing together real-time data, GPS, multimedia verification, and scalable cloud infrastructure yields a trustworthy and efficient emergency response system that materially improves public safety outcomes and supports the core mission of saving lives.

3.1 EXSISTING SYSTEM:

CHAPTER 3 SYSTEM ANALYSIS

The existing emergency ambulance dispatch systems are largely fragmented, relying on manual, voice-centric processes that lack interoperability and automation. Emergency requests are initiated via phone calls, where critical details like the nature of the emergency and

the precise location are conveyed verbally. This information is then manually relayed by a dispatch operator to an available ambulance driver. These systems are often inefficient, error-prone, and time-consuming due to miscommunication, ambiguous location descriptions, and the inability to verify the authenticity of a call. Visualization and real- time coordination are non-existent, requiring complete trust in verbal communication. Additionally, the absence of automated workflows, real-time data synchronization, and verification mechanisms hinders scalability and leads to the wasteful allocation of resources on fake calls. Consequently, the current systems are inadequate for supporting a rapid, accurate, and transparent emergency response..

Proposed System:

RescueNow proposes a real-time, mobile-first platform designed to unify, verify, and intelligently coordinate emergency ambulance responses under a single interoperable framework. The system is built using a dual-app architecture powered by **Flutter** for cross-platform mobile clients, **Firebase** for backend services, and **Cloudinary** for media management, ensuring high reliability and real-time performance. It integrates one-tap emergency requests, mandatory photo verification, and automatic GPS location capture to eliminate delays and false alarms. Advanced features include real-time push notifications for drivers, live location tracking for patients, and a complete trip lifecycle management workflow. The frontend, developed using Flutter's Material Design, provides an intuitive, role-specific interface for both the public and ambulance drivers. This unified approach enhances coordination, transparency, and speed in emergency response, transforming raw user input into actionable, life-saving interventions.

System Requirements:

Functional Requirements:

User Authentication: Anonymous access for the public and Email/Password authentication for drivers.

Emergency Request Creation: One-tap functionality to initiate a request with automaticcamera activation, photo capture, and GPS coordinate submission.

Real-Time Notifications: Instant push notifications sent to all online drivers upon a new emergency request.

Emergency Management for Drivers: Ability to view emergency details (photo, location), and accept or reject requests.

Live Location Tracking: The public user can track the assigned ambulance's location in

real-time on a map.

Driver Status Management: Drivers can toggle their availability status between "Online" and "Offline."

Trip Lifecycle Updates: Drivers can update the emergency status (e.g., "Accepted," "On The Way," "Completed").

Media Upload: Secure and efficient upload of captured emergency photos to a cloud storage service.

API for Data Exchange: RESTful communication for media uploads and real- time listeners for database changes.

Non-Functional Requirements:

Security: Firebase Authentication, Firestore Security Rules for role-based access, and HTTPS encryption for all data in transit.

Usability: A simple, intuitive user interface designed for high-stress situations, minimizing steps to request help.

Performance: Low-latency notifications (< 2 seconds) and real-time database synchronization with minimal delay (< 1 second).

Scalability: Cloud-native infrastructure (Firebase, Cloudinary) capable of handling thousands of concurrent users and requests.

Reliability: High availability of backend services and consistent data synchronization to prevent loss of critical information.

Maintainability: Modular and well-documented codebase using Flutter's widget- based architecture.

Feasibility Analysis:

Technical Feasibility: The project is built using mature, well-supported, and open- source technologies (Flutter, Dart) and a robust Backend-as-a-Service (Firebase). The required hardware—smartphones with cameras, GPS, and internet connectivity—is ubiquitous, making the system widely accessible.

Economic Feasibility: The development is highly cost-effective due to the use of the open-source Flutter framework and the generous free tiers offered by Firebase and Cloudinary. This minimizes initial investment and keeps operational costs low, making it a viable solution for public services and private providers alike.

Problems in the Existing System:

- Fragmented communication channels with no unified data framework.
- Absence of any verification mechanism, leading to a high rate of fake calls.
- Lack of automation, resulting in slow, manual coordination.
- High dependency on inaccurate and ambiguous verbal location descriptions.
- No real-time tracking or status visibility for the person in distress.
- Limited security and no formal data governance for sensitive incident data.

PROPOSED SYSTEM:

RescueNow is developed as a real-time, mobile-first emergency response platform that unifies the public and ambulance drivers under a single, synchronized framework. The system leverages **Flutter** for building high-performance, cross-platform applications for both Android and iOS from a single codebase. The backend is powered by **Google's Firebase** suite, using **Cloud Firestore** as a real-time NoSQL database for instant data synchronization, **Firebase Authentication** for secure user management (anonymous and email/password), and **Firebase Cloud Messaging (FCM)** for delivering low-latency push notifications to drivers. For media management, the system integrates with **Cloudinary** to handle the secure upload and storage of emergency photos. The frontend, built using Flutter's widget library, provides a responsive and intuitive interface tailored for high-stress emergency situations. Key Flutter packages such as **geolocator**, **camera**, and **google_maps_flutter** are used to access native device features for GPS, camera control, and interactive mapping, respectively..

Drawbacks of the Existing System:

- Emergency data is communicated verbally, making integration and verification impossible.
- Lack of standardized data capture leads to inconsistent and unreliable incident reports.
- The entire process is manual, leading to significant delays, human error, and poor traceability.
- No mechanism exists to verify the authenticity of an emergency before dispatching resources.
- Visualization and real-time coordination tools are completely absent.
- No centralized data governance results in a lack of accountability and performance metrics.
- Security and access management are non-existent, posing risks to sensitive information.

- Manual workflows hinder scalability and the ability to manage multiple incidents efficiently.

To address the inefficiencies of current emergency dispatch infrastructures, **RescueNow** proposes a unified, mobile-first architecture that integrates request creation, verification, dispatch, and tracking into a single intelligent platform. It automates the emergency request process through a one-tap interface that captures both photographic evidence and precise GPS coordinates. The backend, powered by Firebase, ensures all data is synchronized in real-time across the ecosystem. This allows for instant notifications to all available drivers, who can then make an informed decision based on verified data. The mobile applications offer real-time insights through an interactive map, enabling the user to track the ambulance's approach. By combining real-time automation with a verification-first workflow, RescueNow transforms a fragmented, slow process into a cohesive, intelligent ecosystem that enhances public safety and optimizes emergency resource management.

Features Of the Proposed System

One-Tap Emergency Request

Allows users to send an emergency alert with a single tap, which automatically opens the camera for mandatory photo evidence.

Photo & GPS Verification

Integrates photo capture and automatic GPS coordinate submission to verify the authenticity and location of an emergency before dispatch.

Real-Time Driver Notifications

Uses Firebase Cloud Messaging (FCM) to instantly push new emergency alerts to all online and available drivers.

Live Ambulance Tracking

Provides the public user with a live map view of the assigned ambulance's location and movement, reducing anxiety and improving coordination.

Role-Based Dual-App Architecture

A dedicated app for the public to request help and a separate, feature-rich app for drivers to manage requests.

Driver Availability Management

A simple toggle allows drivers to set their status to "Online" or "Offline," ensuring requests are only sent to active personnel.

Scalable Cloud-Native Infrastructure

Built entirely on scalable cloud services (Firebase, Cloudinary) to ensure high availability and performance, even with thousands of users.

Secure Authentication

Employs Firebase Authentication for secure driver logins and supports anonymous access for the public to ensure a frictionless experience in emergencies.

3.1. ADVANTAGES OVER THE EXISTING SYSTEM

Unified Real-Time Coordination

Unlike the fragmented and delayed voice-based systems, RescueNow provides a unified platform where data is synchronized in real-time between the public and drivers, ensuring seamless coordination.

Automated Verification and Dispatch

The proposed system leverages mandatory photo and GPS data to automate the verification and dispatch process, replacing the error-prone manual workflows of legacy systems and eliminating fake calls.

Guaranteed Location Accuracy

RescueNow replaces ambiguous verbal addresses with precise GPS coordinates, completely removing search delays and ensuring drivers are navigated directly to the incident location.

Complete Transparency and Tracking

By providing live tracking of the assigned ambulance, the system offers full transparency to the person in distress, a feature entirely missing in traditional systems.

Scalable and Cost-Effective Architecture

By adopting a serverless, cloud-native architecture with Flutter, the system is inherently more scalable and cost-effective than maintaining traditional on-premise dispatch infrastructure.

Secure and Accessible

With Firebase Authentication and a focus on minimal user friction (anonymous access),

RescueNow ensures that the system is both secure for operators and immediately accessible to anyone needing help.

In essence, RescueNow transcends the limitations of legacy dispatch systems by introducing a data-driven, mobile-first, and real-time ecosystem that ensures speed, accuracy, and reliability in emergency medical response.

.

CHAPTER 4 SYSTEM SPECIFICATION

This chapter outlines the technical and functional specifications of the **RescueNow Emergency Ambulance Management System**. It details the architectural design, core modules, system requirements, and technologies used in building the platform.

User Authentication & Access Control

- **Supports two primary roles:** Public User (Anonymous) and Driver (Authenticated).
- **Secure login and role-based access via Firebase Authentication:**
- **Anonymous Authentication** for public users to ensure immediate, frictionless access during emergencies.
- **Email & Password Authentication** for registered and verified ambulance drivers.
- **Access control is enforced at the database level** using Firestore Security Rules, restricting data read/write permissions based on user roles and ownership.

Emergency Request & Data Pipeline

- Uses the Flutter camera and geolocator packages for native device integration to capture photo evidence and precise GPS coordinates**.**
- **Implements a real-time data pipeline for new emergency requests:**

1. **Photo Upload:** The captured image is sent to Cloudinary via a REST API call using the dio package.

2. Data Write: Upon a successful upload, a new document is created in the emergency_requests collection in Cloud Firestore, containing the image URL, GPS data, timestamp, and user information with a status of "waiting_for_driver".

Real-Time Coordination & Notification Module

- Firebase Cloud Messaging (FCM) is integrated to send instant, high-priority push

notifications to all online drivers whenever a new emergency document is created.

- The Driver's App uses a real-time Firestore listener (StreamBuilder) to subscribe to the emergency_requests collection, automatically updating the UI as new requests arrive or existing ones are updated.
- The system ensures bidirectional synchronization, allowing status changes made by the driver (e.g., "Accepted") to be reflected instantly in the Public User's app.

Location & Mapping Module

- The google_maps_flutter package is used to display interactive maps within both applications.
- The Public User's app displays the real-time location of the assigned ambulance.
- The Driver's App displays the patient's location and can be integrated with native mapping applications for turn-by-turn navigation.
- The driver's location is periodically updated to Firestore to facilitate the live tracking feature.

Backend & Cloud Services

- Backend-as-a-Service (BaaS) model using Google's Firebase suite, eliminating the need for manual server management.
- Cloud Firestore: A NoSQL, document-based database used for storing all application data, including emergency requests, user profiles, and driver information.
- Cloudinary: A third-party media management service used for scalable storage and delivery of emergency photos.

Mobile Application Frontend

- **Frontend:** Built entirely using the Flutter framework and Dart programming language for a single, high-performance codebase for both Android and iOS.
- **UI/UX:** Implements Material Design 3 principles for a clean, intuitive, and responsive user interface suitable for high-stress situations.
- **State Management**: Utilizes Flutter's StreamBuilder and StatefulWidget for managing real-time data streams and local UI state.

Security & Governance

- End-to-end encryption for all data in transit via HTTPS and the secure WebSocket connection provided by Firebase.

- Role-Based Access Control (RBAC) is implemented via granular Firestore Security Rules.
- Observability via Prometheus, Grafana, and Jaeger

System Requirements

Client Device Configuration (for running the mobile apps):

- **Operating System:** Android 8.0 (Oreo) or higher / iOS 12.0 or higher.
- **Hardware:**
  - Functional front and back cameras.
  - GPS/Location Services capability.
  - Minimum 2 GB RAM.
- **Network:** Active internet connection (Wi-Fi or Cellular Data) is required for all functionalities.

Cloud Service Requirements:

- **Firebase Project:**
  - Authentication (Anonymous and Email/Password providers enabled).
  - Cloud Firestore database created.
  - Firebase Cloud Messaging (FCM) API enabled.
- **Cloudinary Account:**
  - An active account with the cloud name configured.
  - An unsigned upload preset created for handling image uploads.

Software & Development Stack

- **Framework:** Flutter SDK 3.24.0 or higher**.**
- **Language:** Dart 3.5.0 or higher.
- **IDE:** Android Studio or Visual Studio Code.
- **Backend**: Firebase (Authentication, Firestore, FCM).
- **Media Storage**: Cloudinary.
- **CorePackages**: firebase_core, cloud_firestore, firebase_auth, firebase_messagin g, camera, geolocator, google_maps_flutter, dio.

RescueNow's architecture seamlessly integrates mobile-native features, real-time data synchronization, and scalable cloud services into a unified emergency response ecosystem—offering a reliable, fast, and transparent platform for next-generation public safety.

CHAPTER 5 PROJECT DESCRIPTION

The **RescueNow Emergency Ambulance Management System** is an advanced, real- time mobile platform designed to unify, verify, and coordinate emergency medical responses under a single, intelligent ecosystem. Traditional emergency dispatch systems are voice-centric, manual, and lack the real-time data synchronization needed for a swift and accurate response. RescueNow overcomes these limitations through a mobile-first, cloud-native framework that integrates GPS tracking, photo verification, and instant notifications. It ensures automated request processing, eliminates fake calls through visual evidence, and provides complete transparency for both the public and first responders.

The platform leverages an efficient and scalable tech stack, including **Flutter** and **Dart** for cross-platform mobile development, **Google's Firebase** suite (Firestore, Authentication, FCM) for backend services, and **Cloudinary** for media management. RescueNow forms the digital backbone for a modern emergency response service, supporting a real-time data flow from a user's smartphone directly to the nearest available ambulance drivers, creating a single, analytics-ready platform for incident management.

Objectives

- To establish a unified, real-time mobile platform for emergency ambulance dispatch.
- To automate the emergency request process using GPS and photo verification.
- To eliminate fake calls and reduce resource wastage through mandatory visual evidence.
- To provide a live tracking interface for patients to monitor the ambulance's location.
- To ensure a secure, scalable, and highly available system using cloud-native services.

System Overview

The RescueNow system consists of two interconnected mobile applications forming a complete emergency response framework:

People's Emergency App:

This app allows any member of the public to instantly report an emergency. It uses the device's native camera and GPS to capture and send verifiable incident data. The user can then track the assigned ambulance in real-time until it arrives.

Driver's Ambulance App

This app is for registered ambulance drivers. It receives real-time push notifications for new emergencies, displays incident details (including photo and map location), and allows the driver to manage the trip lifecycle from acceptance to completion.

Modules Description

Authentication & Access Control Module

Uses **Firebase Authentication** for secure and role-specific access.

- **Anonymous Authentication:** Allows the public to request help without the friction of creating an account. A temporary user ID is generated for the session.
- **Email/Password Authentication:** Registered drivers log in with their credentials to access the driver-specific functionalities.

Emergency Request & Verification Module

Applies a data-first approach to emergency requests.

- The workflow begins with the **camera** package to capture a photo of the incident.
- The **geolocator** package automatically fetches the device's precise GPS coordinates.
- This data is packaged and sent to the backend, creating a verified request.

Real-Time Notification & Dispatch Module

Handles the instant dissemination of information.

- A new document in the **Cloud Firestore** emergency_requests collection triggers a cloud function (or is listened to by clients).
- **Firebase Cloud Messaging (FCM)** sends a high-priority push notification to all devices subscribed to the "drivers" topic.
- This ensures all online drivers are alerted within seconds of a new emergency.

Data Governance & Storage Module

Implements a simple and effective data storage strategy.

- **Cloud Firestore:** A NoSQL database stores all textual and location data in collections like emergency_requests and drivers.
- **Cloudinary:** A dedicated media management platform stores all captured photos. The URL of the stored image is referenced in the corresponding Firestore document.
- **Firestore Security Rules** provide server-side authorization, ensuring users can only access data permitted by their role.

Live Tracking & Visualization Module

Provides real-time situational awareness.

- The Driver's App periodically updates its current location to a document in Firestore.
- The People's App listens to these changes in real-time and updates the ambulance's position on an interactive map built with the **google_maps_flutter** package.

Technologies Used

Component	**Technology**
Mobile Framework	Flutter 3.24.0, Dart 3.5.0
Frontend UI	Material Design 3
Backend Services	Firebase (Backend-as-a-Service)
Real-Time Database	Cloud Firestore
Authentication	Firebase Authentication (Anonymous, Email/Password)
Push Notifications	Firebase Cloud Messaging (FCM)
Media Storage	Cloudinary
Mapping & Location	google_maps_flutter, geolocator
Native Device Access	camera package
API Communication	dio (for Cloudinary uploads)
State Management	StreamBuilder, StatefulWidget
Development Tools	Android Studio, VS Code, Git

System Architecture

- RescueNow follows a client-server architecture with a thin client approach.
- Presentation Layer: Two Flutter applications (People's and Driver's) providing the user interface.
- Application/Backend Layer: A serverless backend provided by Firebase, which handles all business logic, data storage, and real-time communication.
- Data Layer: A hybrid storage model using Cloud Firestore for structured metadata and Cloudinary for unstructured image data.
- This modular, BaaS-centric design ensures high scalability, low maintenance, and rapid development cycles.

Security Features

- Secure Authentication using Firebase's proven identity platform.
- HTTPS encryption for all communication between the mobile apps and backend services.
- Role-Based Access Control (RBAC) enforced by server-side Firestore Security Rules.
- API keys and sensitive configuration are secured and not exposed in the client-side code.

5.1. Working

The RescueNow system operates through an automated, real-time workflow designed to connect a person in need with the nearest available emergency responder as quickly and efficiently as possible.

5.1.1 Emergency Request Initiation

A user opens the People's App and taps the "Request Emergency" button.

The app immediately activates the device's camera. The user takes a photo of the situation, which serves as mandatory visual proof.

Simultaneously, the app fetches the device's precise GPS coordinates.

5.1.2 Data Verification & Upload

The captured photo is uploaded directly to Cloudinary, which returns a secure URL. The app then creates a new document in the emergency_requests collection in Cloud Firestore. This document contains the Cloudinary image URL, the GPS coordinates, a timestamp, and the user's anonymous ID. The document's initial status is set to "waiting_for_driver".

5.1.3 Real-Time Dispatch & Notification:

The creation of this new document instantly triggers Firebase Cloud Messaging (FCM).

FCM sends a push notification to all drivers who are logged into the Driver's App and have their status set to "Online."

Simultaneously, the Driver's App, which has a live listener attached to the emergency_requests collection, automatically displays the new request in its list.

5.1.4 Driver Action & Coordination:

A driver taps on the notification or the list item to view the emergency details: the photo for verification and the location on a map.

The driver can then choose to "Accept" or "Reject" the request.

If a driver accepts, the status of the emergency document in Firestore is updated to

"accepted", and the driver's ID is added to the document. This action prevents other drivers from accepting the same request.

5.1.5 Live Tracking & Trip Management:

Once a driver accepts, the People's App is notified in real-time. Its UI updates to show that help is on the way and begins displaying the assigned ambulance's location on a map.

The Driver's App periodically sends its updated GPS coordinates to Firestore, which are then streamed to the People's App for the live tracking feature.

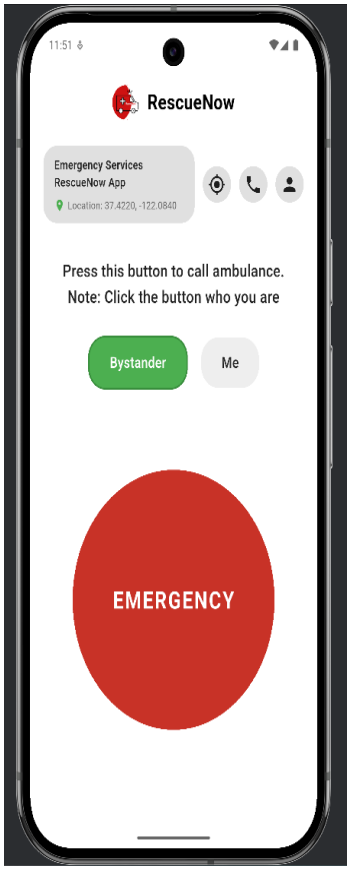
The driver updates the trip status in their app (e.g., "Arrived," "Trip Completed"), and these changes are reflected instantly for all stakeholders.

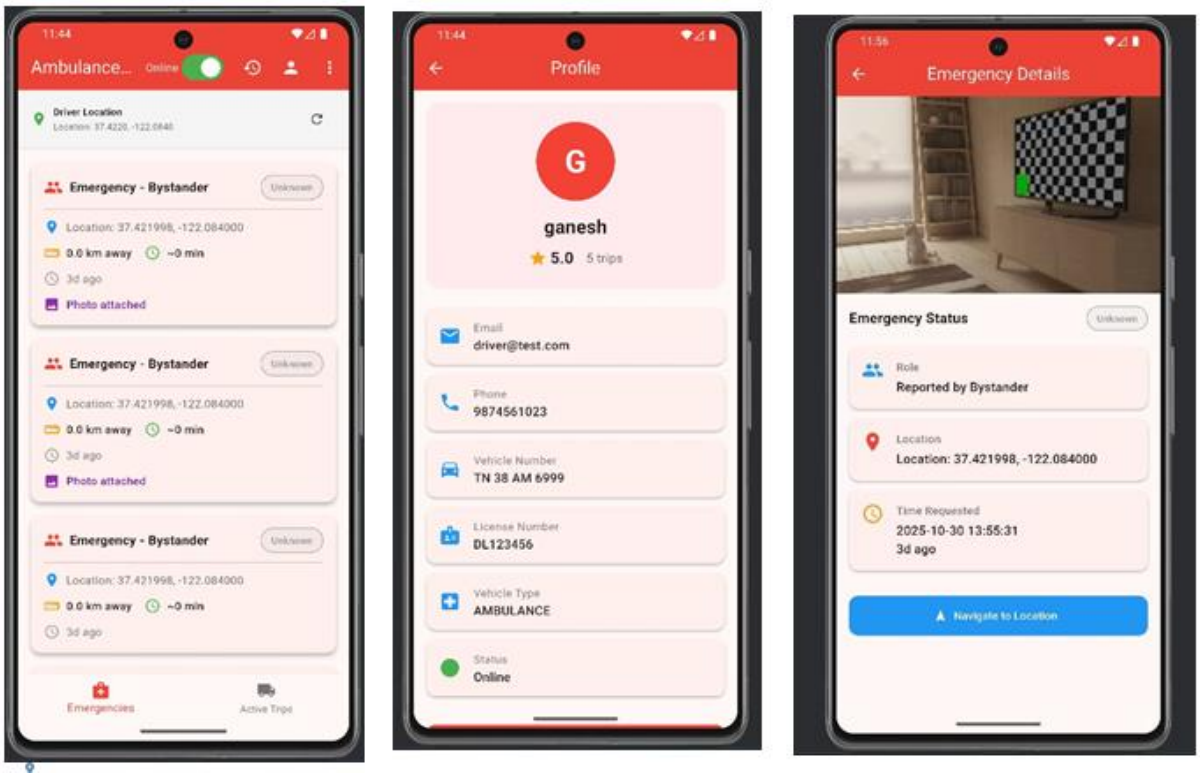
5.1.6 Completion and Logging

When the trip is completed, the final status is logged in Firestore. This data can then be used for auditing, reporting, and performance analysis.

The entire workflow, from request to completion, is logged with timestamps, providing a complete and auditable record of the incident.

5.1.1. APP UI





CHAPTER 6 VALIDATION AND TESTING

6.1 VALIDATION

Validation and testing are critical components of the RescueNow development lifecycle, ensuring that each subsystem—from emergency request creation to real-time driver tracking—functions correctly, securely, and efficiently. The validation process focuses on system integrity, data accuracy, security compliance, and real-time performance under simulated real-world operating conditions.

Validation Techniques Used

Data Ingestion Validation

All incoming emergency requests undergo strict validation on the client-side before submission. Field-level checks ensure that a valid photo has been captured and that GPS coordinates are in the correct format (latitude, longitude). The system validates that all required fields are populated before creating a document in Firestore. Invalid or incomplete requests are blocked, with user-friendly error messages prompting the user to retry.

Data Model & Standardization Validation

Each document created in Cloud Firestore is validated against a predefined schema. Validation scripts and Firestore Security Rules ensure that documents in the emergency_requests and

drivers collections contain the correct data types, required fields (e.g., status, timestamp, location), and adhere to the system's data model.

API and Access Control Validation

Each API endpoint, particularly the Cloudinary upload function, is validated using unit and integration tests to ensure proper request-response handling, error codes, and security. Role-Based Access Control (RBAC) via Firestore Security Rules is rigorously tested to confirm that public users and drivers can only read or write data permitted by their roles. Unauthorized or malformed database requests are blocked by server-side rules and logged.

Real-Time Workflow Validation

Integration tests confirm the smooth, real-time transition of an emergency request through its entire lifecycle (waiting → accepted → in-progress → completed). The push notification pipeline via FCM is validated for latency and delivery success to ensure drivers are alerted instantly.

File & Object Storage Validation:

The photo upload process to Cloudinary is validated for integrity and successful completion. The system checks for a valid URL response from Cloudinary before writing the reference to Firestore. Error handling for failed uploads (e.g., due to network issues) is tested to ensure the system remains stable.

Session, Authentication & Encryption Validation:

Authentication flows via Firebase Authentication are tested to enforce secure login for drivers and seamless anonymous session creation for the public. The integrity of user sessions and the enforcement of security rules based on user UID are validated. All communication between the mobile clients and backend services (Firebase, Cloudinary) is validated to ensure it occurs over an encrypted TLS/HTTPS connection.

Visualization & User Interface Validation:

The Flutter-based UI is tested using widget and integration tests for consistency, responsiveness, and accurate data rendering from live Firestore streams. Validation ensures that the google_maps_flutter widget correctly displays and updates ambulance locations in real-time without data drift or rendering errors.

System Performance Validation:

Stress testing is conducted by simulating multiple simultaneous emergency requests to validate system performance under load. Key metrics such as notification delivery time, database write/read latency, and UI responsiveness are monitored to ensure the system meets its low-latency requirements for a real-world emergency scenario.

6.2 TESTING:

Validation and testing ensure that the RescueNow platform performs with accuracy, reliability, and speed across its real-time, mobile-first architecture. The testing process covered all key modules—request creation, notification, real-time tracking, and data synchronization—validating both system integrity and performance.

Unit Testing:

Individual functions and widgets within the Flutter applications were tested in isolation. Dart's test package was used to validate core business logic, such as distance calculation, data model serialization, and state changes in UI components. Mock tests using the mockito package verified interactions with Firebase and Cloudinary services without making actual network calls.

Integration Testing

Comprehensive end-to-end tests validated the seamless flow of data and actions between the UI, the Firebase backend, and Cloudinary. Flutter's integration_test package was used to script user interactions, such as creating an emergency, and verify that the corresponding data appeared correctly in Firestore and that UI elements updated as expected in response to real-time database changes.

System Testing

The complete RescueNow system was deployed on two separate devices (emulators or physical phones) running the People's App and the Driver's App simultaneously. This end-to-end testing simulated a real-world scenario to test the overall workflow reliability, notification latency, and the accuracy of the live tracking feature.

Manual Testing

The development team manually tested all user stories and features on different devices and screen sizes to check for usability issues, UI bugs, and workflow correctness. This included

testing edge cases like network loss, permission denial (camera/GPS), and application lifecycle events (app in background/foreground).

User Acceptance Testing (UAT)

A small group of beta testers (peers and project supervisors) used the applications to simulate emergencies. Their feedback on the system's usability, speed, and clarity was collected and incorporated to refine the user interface and streamline the overall workflow, ensuring the app is intuitive even in high-stress situations.

7.1 MERITS:

CHAPTER 7 MERITS AND DEMERIT

The **RescueNow Emergency Ambulance Management System** represents a transformative advancement in public safety and emergency response, combining real-time mobile technology, cloud services, and a verification-first workflow to create a fast, reliable, and transparent dispatch ecosystem.

Real-Time Coordination and Unprecedented Speed: RescueNow drastically reduces emergency response time by replacing a slow, manual phone-based process with an automated, one-tap mobile interface. The system's real-time data synchronization via Firebase ensures that emergency requests are delivered to all available drivers in under two seconds.

Automated Verification and Fake Call Elimination: Through its mandatory photo verification workflow, RescueNow provides drivers and dispatchers with visual proof of an emergency before committing resources. This feature is designed to virtually eliminate the 40-50% of fake or non-emergency calls that plague traditional systems, saving valuable time and operational costs.

Guaranteed Location Accuracy via GPS: The system replaces ambiguous, error-prone verbal addresses with precise GPS coordinates captured directly from the user's device. This guarantees 100% location accuracy, eliminating search delays and ensuring drivers can navigate directly to the incident scene using integrated maps.

Scalable and Cost-Effective Cloud-Native Architecture: The system leverages a serverless backend (Firebase) and a cross-platform mobile framework (Flutter), which allows for massive scalability with minimal infrastructure maintenance. This cost-effective model makes the advanced dispatch system accessible to both public services and private ambulance

providers without significant capital investment.

Enhanced Transparency and User Trust: By providing a live map view of the assigned ambulance's location and status, RescueNow offers complete transparency to the person awaiting help. This feature significantly reduces anxiety and builds trust in the emergency response process.

High Accessibility with Frictionless Access:

The inclusion of anonymous authentication allows any member of the public to request emergency help instantly without the need to create an account, removing a critical barrier to use during a high-stress event.

7.2 DEMERITS

High Dependency on Network Connectivity

The platform's entire functionality—including GPS location sharing, photo uploads, and real-time database synchronization—is critically dependent on a stable internet connection (cellular data or Wi-Fi) for both the user and the driver. The system is non- functional in areas with no network coverage.

Dependency on Device Capabilities and User Permissions

The system's effectiveness relies on the user's smartphone having a functional camera and GPS module. Furthermore, the user must grant the application the necessary permissions to access these features; if permissions are denied, the core functionality is rendered useless.

Potential for Poor Quality Verification Media

The reliability of the photo verification feature is subject to the quality of the image provided by the user. A blurry, dark, or poorly framed photo may not offer enough context for a driver to accurately assess the severity of an emergency.

Limited Offline Functionality

As a cloud-native, real-time system, RescueNow has no built-in offline capabilities. If a driver loses connectivity mid-trip, their location will no longer update for the user, and they will not be able to receive new requests until the connection is restored.

Battery Consumption

The continuous use of GPS for live location tracking in the Driver's App can lead to

significant battery drain on the driver's device, which could be a concern during long shifts without access to charging.

No Centralized Human Dispatcher

While automation increases speed, the system lacks a human dispatcher who can handle complex situations, provide verbal comfort, or manage multi-vehicle dispatch scenarios. The system operates on a first-come, first-served acceptance model among drivers.

CHAPTER 8 FUTURE SCOPE

The **RescueNow Emergency Ambulance Management System** establishes a foundational step toward a unified, intelligent, and automated emergency response ecosystem. While the current system effectively reduces dispatch times, eliminates fake calls, and provides real-time tracking, its modular, cloud-native design allows for extensive future expansion. The following outlines the major directions in which RescueNow can evolve to become a comprehensive public safety and healthcare intelligence hub.

Integration with National and International Marine Frameworks MARLIN can be extended to interoperate with global and regional marine data infrastructures such as INCOIS, IMD, CMLRE, GOOS (Global Ocean Observing System), GEOSS (Global Earth Observation System of Systems), and Copernicus Marine Service. This integration will enable seamless data exchange, improve cross-institutional collaboration, and promote unified marine research under open data standards like Darwin Core and OGC APIs.

- **Integration with National and Hospital Networks** Rescue Now can be extended to interoperate with national emergency services (like 108/102 in India or 911) and public safety frameworks. Further integration with Hospital Information Systems (HIS) and Electronic Health Records (EHR) would enable seamless data exchange, allowing ambulance crews to pre-register patients and check for emergency room bed availability en route, significantly improving patient handover and care continuity.

- **AI-Powered Predictive Analytics and Modeling** Future versions can incorporate machine learning and deep learning models to predict emergency hotspots based on historical data, traffic patterns, and public events, enabling the strategic pre-positioning of ambulances. AI could also be used to analyze the user-submitted photo to automatically assess the severity of an incident (e.g., minor vs. major trauma), helping to dispatch the appropriate level of care.

- **Digital Twin of City-Wide Emergency Response** RescueNow has the potential to evolve into a Digital Twin of a city's emergency response network. By integrating real-time traffic data, ambulance locations, and hospital statuses, the system could create a dynamic 4D model of the entire ecosystem. This would allow public health officials to simulate mass casualty incidents, test new dispatch algorithms, and optimize resource allocation strategies in a virtual environment before real-world implementation.

- **Edge and IoT Data Integration** Future iterations could include direct integration with IoT devices. This includes in-vehicle telematics to monitor ambulance speed and driving behavior, as well as personal wearable devices (like smartwatches with fall detection or ECG monitors) that could automatically trigger a RescueNow alert with the user's vital signs, providing first responders with critical health data before they even arrive.

- Blockchain for Data Provenance and Trust To ensure an immutable and auditable record of every emergency incident, RescueNow can integrate blockchain-based provenance tracking. Each step of the lifecycle—from the initial request to trip completion and patient handover—can be timestamped and cryptographically verified on a distributed ledger. This would provide unparalleled data integrity for legal, insurance, and quality review purposes.

- **Payment Gateway and Billing Automation** For private ambulance services, integrating a payment gateway (like Stripe or Razorpay) would automate the billing process. The system could calculate fares based on distance and services rendered, allowing for seamless and transparent payment directly through the app upon trip completion.

- **Enhanced Mobile and Offline Capabilities** The Driver's App can be expanded into a comprehensive mobile data terminal, including digital Patient Care Report (ePCR) forms to replace paper records. For areas with poor connectivity, an offline mode could be developed that uses SMS as a fallback to transmit basic emergency details and GPS coordinates.

- **Multilingual and Accessibility Features** To promote inclusivity and serve diverse populations, RescueNow can support multiple regional languages (e.g., Hindi, Tamil, Malayalam). Accessibility can be enhanced through voice-activated commands for hands-free emergency requests and compatibility with screen readers for visually impaired users.

- **Advanced Communication and Telemedicine** Future versions could integrate secure, in-app voice or video calling, allowing the driver or a remote paramedic to communicate directly with the person on the scene to provide critical first-aid instructions before the ambulance arrives.

- **Global Collaboration and Expansion** The RescueNow model can be scaled and adapted for deployment in different cities and countries, connecting various public and private ambulance providers under a single, interoperable framework. Such expansion would help standardize emergency response protocols and improve public safety outcomes on a global scale.

CHAPTER 9 CONCLUSION

The RescueNow Emergency Ambulance Management System represents a transformative leap in how emergency medical services are requested, verified, dispatched, and monitored. Conceived as a unified digital ecosystem, RescueNow bridges the long-standing gap between slow, manual dispatch systems and the need for rapid, accurate, and verifiable emergency coordination. By integrating real-time mobile technologies, cloud-native services, and a verification-first workflow, the platform ensures unprecedented response speed, data accuracy, and complete transparency for both the public and first responders.

Through its dual-app architecture, RescueNow enables the seamless and instantaneous flow of critical data—ranging from precise GPS coordinates and photographic evidence to live ambulance location and status updates. The inclusion of intelligent automation for dispatch notifications and the mandatory verification of incidents empowers first responders to make faster, evidence-based decisions while eliminating the resource drain from fake calls. Furthermore, the system's design prioritizes accessibility and ease of use, ensuring that life-saving technology is available to anyone with a smartphone, at the moment it is needed most.

In conclusion, RescueNow is more than just an emergency dispatch application—it is a cornerstone for the next generation of public safety systems. It embodies the convergence of mobile computing, real-time data synchronization, and life-saving operational efficiency to create a reliable, data-driven framework for protecting communities. With continued evolution—through integration with national emergency networks, AI-powered predictive analytics, and connected healthcare platforms—RescueNow has the potential to redefine emergency response infrastructure, enabling smarter, faster decisions that ultimately save

lives.

REFERENCE

1. Emergency Response Time Studies: Lerner, E. B., & Moscati, R. M. (2001). The Golden Hour: scientific fact or medical "urban legend"?. *Academic emergency medicine*, 8(7), 758–
2. 760. https://doi.org/10.1111/j.1553-2712.2001.tb00201.x
3. Challenges in EMS Dispatch (Fake Calls): "40% of calls to 108 in Delhi are blank or fake: Health department" - The
4. Hindu. https://www.thehindu.com/news/cities/Delhi/40-of-calls-to- 108-in-delhi-are-blank-or-fake-health- department/article65379986.ece
5. Impact of GPS on Emergency Services: Zandbergen, P. A. (2009). "Wireless 9-1-1 and the role of GIS." In *Handbook of Research on Geoinformatics* (pp. 344-351). IGI Global.
6. Mobile Health (mHealth) in Emergencies: Khatun, F., Heywood,

A. E., Ray, P. K., & Han, F. (2015). "A literature review of the use of mobile technology for public health emergencies." *International conference on smart homes and health telematics*.

7. Flutter Framework Documentation: https://flutter.dev/docs
8. Dart Language Documentation: https://dart.dev/guides
9. Firebase Platform
10. Documentation: https://firebase.google.com/docs

A. Cloud Firestore: https://firebase.google.com/docs/firestore
B. Firebase Authentication: https://firebase.google.com/docs/auth
C. Firebase Cloud Messaging: https://firebase.google.com/docs/cloud- messaging

11. Cloudinary Media Platform
12. Documentation: https://cloudinary.com/documentation
13. Google Maps Platform for
14. Flutter: https://pub.dev/packages/google_maps_flutter
15. Geolocator Flutter Plugin: https://pub.dev/packages/geolocator
16. Camera Flutter Plugin: https://pub.dev/packages/camera