Tribhuvan University

Faculty of Humanities and Social Sciences

A Project Report on
Rapid Reach
(Real-Time Emergency Locator)

Submitted to

BCA department

Nepal Kasthamandap College

In partial fulfilment of the requirements for the bachelor's in computer application

Submitted by

Ambhoj Adhikari

[Reg No.: 6-2-144-01-2020]

November 2025

Under the Supervision of

Shyam Maharjan

Tribhuvan University

Faculty of Humanities and Social Sciences

Nepal Kasthamandap College

Supervisor's Recommendation

I hereby recommend that this project prepared under my supervision by **Ambhoj Adhikari** entitled **"Rapid Reach"** (Real-Time Emergency Locator) in partial fulfilment of the requirement for the degree of Bachelor's in Computer Application is recommended for the final evaluation.

Supervisor

Shyam Maharjan

Nepal Kasthamandap College

Tribhuvan University

Faculty of Humanities and Social Sciences

Nepal Kasthamandap College

Letter of Approval

This is to certify that this project prepared by **Ambhoj Adhikari** entitled **"Rapid Reach"** (Real-Time Emergency Locator) in partial fulfilment of the requirements for the degree of Bachelor's in Computer Application has been evaluated. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

Supervisor **Shyam Maharjan** Lecturer Nepal Kasthamandap College	BCA Coordinator **ER.Sujan Poudel** Nepal Kasthamandap College
Internal Examiner **Roshan Poudel**	External Examiner **Roshan Tandukar** Patan Multiple Campus

Abstract

In Nepal, accessing timely and reliable emergency services such as hospitals, police stations, fire services, ambulances, and blood banks remains a critical challenge. Rapid Reach (A Real-Time Emergency Locator) is developed as a platform to bridge this gap by providing users with accurate, location-based information about nearby emergency services. The system addresses problems such as delays in finding help, lack of verified service information, and the absence of a centralized platform for quick access to emergency resources. Through a user-friendly interface, Rapid Reach enables users to detect their real-time location, search for categorized services, view distance and availability, and directly connect through the "Call Now" feature.

The platform is designed to deliver essential details like service availability, working hours, verified status, and user ratings, ensuring reliability and efficiency. By centralizing emergency resources, Rapid Reach reduces response time and enhances public safety in critical situations. Although similar tools exist globally, the system stands out as a much-needed solution in the context of Nepal, where localized, real-time emergency service locators are limited.

Keywords: Real-Time Emergency Locator, Priority Scoring Algorithm, Socket IO, Web RTC

Acknowledgement

I take this opportunity to express my heartfelt gratitude to everyone who has been instrumental in the successful completion of my project, Rapid Reach. This project was undertaken as part of my BCA Eighth Semester curriculum, and it would not have been possible without the support and guidance of many individuals.

First and foremost, I would like to express my sincere thanks to my supervisor, Mr. Shyam Maharjan, for providing me with the golden opportunity to work on this project. His valuable guidance, insightful suggestions, and constant encouragement were a great source of motivation throughout the development process. Under his mentorship, I was able to enhance my technical knowledge and gain skills essential for both academic and professional growth.

I am also deeply grateful to Nepal Kasthamandap College for providing the necessary infrastructure, resources, and a supportive learning environment. The availability of computer labs, library resources, and access to essential software contributed greatly to the smooth progress of my work. I would also like to extend my gratitude to all the faculty members of the BCA Department for their constructive feedback, academic advice, and encouragement during this journey.

Finally, I would like to thank my friends and classmates for their continuous support, encouragement, and motivation during both the challenging and successful moments of this project. Their companionship kept me inspired and focused.

This project has been a significant learning experience for me, and I remain sincerely thankful to everyone who contributed, directly or indirectly, to the successful completion of Rapid Reach.

Yours sincerely,
Ambhoj Adhikari

Table of Contents

List of Figures

List of Tables

List of Listings

List of Abbreviations

Abbreviation	**Description**
API	Application Programming Interface
CSS	Cascading Style Sheet
HTML	HyperText Markup Language
JS	JavaScript
JSON	JavaScript Object Notation
JSX	JavaScript XML
MERN	Mongo Db, Express JS, React JS, Node JS
RTC	Real-Time Communication
SQL	Structured Query Language
URL	Uniform Resource Locator
XML	Extensible Mark UP Language

Chapter 1 Introduction

1.1 Introduction

Rapid Reach is a real-time emergency locator platform designed to connect users with nearby essential services when urgency strikes. Whether someone needs a hospital, police station, fire service, ambulance, or blood bank, the website provides instant access based on the user's current location. It simplifies the process of finding help by allowing users to search by category and view crucial details such as distance, availability, and working hours.

The platform also includes features like “Call Now” functionality, verified service status, and user ratings to ensure quick and reliable support. By offering accurate and up-to-date information in just a few clicks, Rapid Reach empowers people to take fast action in critical situations, saving valuable time and potentially lives.

1.2 Problem Statement

In the context of Nepal, peoples are facing numerous challenges related to health services. The problems such as:

- No centralized platform for services like Fire Department, Health Service and Police Departments.
- Lack of real-time access and direct communication.
- Delay in response time.

1.3 Objectives

The objectives of this project which are mainly focused are mentioned below:

- To develop a real-time live tracking system for ambulance, fire truck and police vehicle.
- To reduce emergency response time by applying call now feature.
- To recommend nearby departments like hospitals, fire department and police department along with vehicles services like ambulance, fire truck and police vehicle.

1.4 Scope and Limitation

1.4.1 Scopes

- It provides the services with the help of real-time search, location and contact.
- It displays essential details like availability, working hours, distance, verification status, and ratings.
- It enables direct calling to emergency services from within the platform.

1.4.2 Limitations

- It does not provide medical, legal, or emergency advice.
- It is only made for two municipality including Kathmandu and Chandragiri.

1.5 Development Methodology

The Agile methodology will be used for the system because in "Agile method" the task breaks into smaller iterations, or parts do not directly involve long term planning. In agile method the system can change requirements, even in later stages and can be kept things simple as possible.

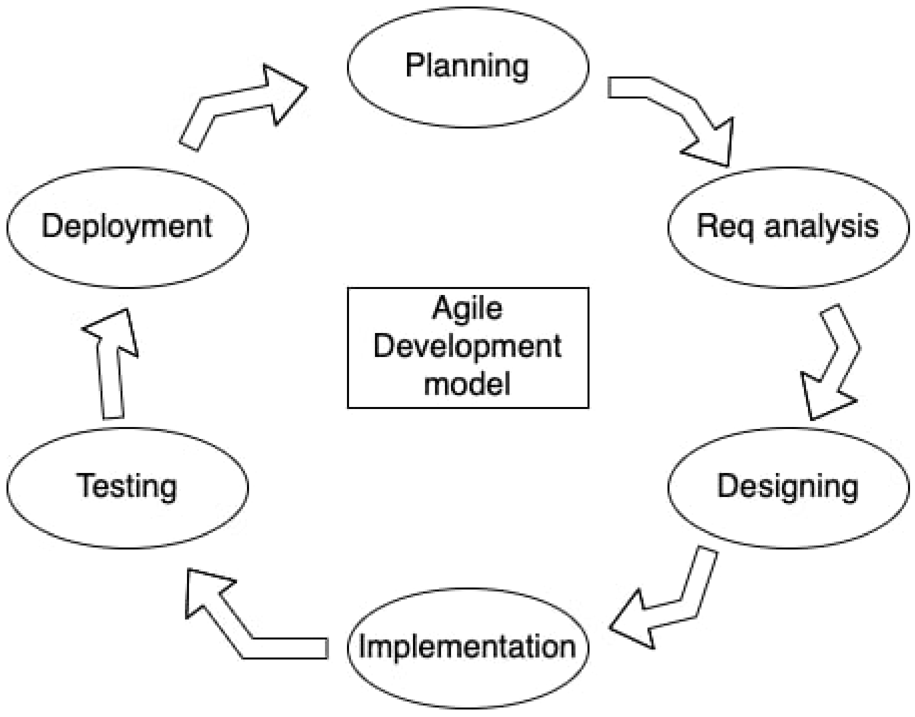


Figure 1.1: Agile Methodology

1.6 Report Organization

Introduction

It outlines the main goals of the project and the specific criteria that must be met during its execution.

Background Study and Literature Review

It involves examining the current system, conducting a detailed analysis, and gaining a comprehensive understanding of its history and context.

System Analysis and Design

It outlines both the functional and non-functional requirements of the system, including data modelling, process workflows, database schemas, and the overall architectural design.

Implementation and Testing

It specifies the tools and technologies that will be utilized for the system's development, along with the testing procedures to be followed, including the creation and execution of test cases.

Conclusion and Future Recommendation

It provides a comprehensive summary of the project through documentation, highlighting key features and suggesting areas for potential system improvement.

Chapter 2 Background Study and Literature Review

2.1 Background Study

In Nepal, timely access to emergency services such as hospitals, ambulances, police stations, fire services, and blood banks is crucial but often challenging. Many people struggle to quickly find and contact the nearest emergency help due to a lack of centralized, real-time information. This problem is more pronounced in rural and remote areas, where digital infrastructure and service visibility are limited, leading to dangerous delays during critical situations.

Currently, individuals rely on outdated directories, word-of-mouth, or random internet searches, which are inefficient and unreliable when every second counts. Even in urban areas, the absence of verified, up-to-date details about service availability, working hours, and direct contact options causes confusion and mistrust. This gap in emergency response accessibility puts lives at risk and highlights the urgent need for a comprehensive digital solution.

Rapid Reach is developed to address these challenges by providing a real-time emergency locator platform that connects users with verified emergency services based on their location. With features like live GPS tracking, categorized searches, instant calling, and information on working hours and ratings, Rapid Reach empowers users to act quickly and confidently during emergencies. By enhancing accessibility and trust, Rapid Reach aims to improve emergency response outcomes and save lives across Nepal.

2.2 Literature review

The Nepal Ambulance Service (NAS) is a non-profit initiative dedicated to the establishment of an emergency medical response system (EMS) in the greater Kathmandu and Patan municipalities, later to be expanded nationwide. This system will provide rapid ambulance transport to hospitals along with life-saving medical care by trained emergency medical technicians (EMTs) for sick and injured people regardless of ability to pay. NAS aims to operate fully equipped and staffed ambulances via a central dispatch facility with radio communication between area hospitals and ambulances in order to ensure rapid transport and treatment for individual patients. NAS EMTs will be trained by Stanford University School of Medicine (USA) experts from Stanford Emergency Medicine International (SEMI). [1]

Ashal Chhimeki Nepal-ACN (Good Neighbours Nepal-GNN) is a non-government organization, established in 2002 at Kathmandu, Nepal with the aim of transforming the community for the sustainable development holistically. The major areas of interventions are Livelihood, Health, Gender Equality & Social Inclusion, Sanitation & WASH, Education, Network building and Partnerships, Emergency response, and Advocacy. Nepal remains one of the poorest countries in the world where 25% of the people are living on less than 100 Rupees (less than $1) a day. Hence, GNN seeks to work on supporting marginalized people in the overall community development. In addition to this, GNN also provides relief support during natural disasters such as the recent earthquake we faced in 2015. The projects are designed on need based principle outlined by the target group/community. [2]

Hospital for Advanced Medicine & Surgery (HAMS) is a multi-disciplinary tertiary care hospital situated in Dhumbarahi, Kathmandu. With over 25 years of experience and expertise, we have been providing quality and affordable healthcare to the community. We are proud of our highly experienced clinicians, technicians & administrators, backed by state-of-the-art technology and dependable infrastructure. Our hospital is fostered by highly trained and caring nurses who strive to give you the best patient care and experience the town has to offer. [3]

Alka Hospital, established in 2006, evolved from Alka Pharmacy (1995) and Alka Polyclinic (2000). With 100 beds, advanced diagnostic, curative facilities, and a commitment to quality healthcare. To Start with, Alka hospital had its footprints in the form of Alka Pharmacy that was established in 2052 BS (1995 AD) with the aim of supplying sufficient and proper quality medicine to the People within its vicinity. Within two year, with god's grace and endless effort of the staff, the management team thought to expand its services resulting the alka poly clinic (2057BS,2000AD) along with its pharmacy. [4]

Nepal Police, established in 1951, plays a crucial role in maintaining law and order, ensuring public safety, and upholding the rule of law in Nepal. Scholarly literature highlights the organization's efforts in modernizing its services, especially in response to evolving security challenges and democratic transitions. Studies have examined its role during periods of political instability, the civil conflict (1996–2006), and its reforms in community policing and human rights training. However, critiques often focus on issues of corruption, political interference, and inadequate resources, which hinder effective service delivery. Recent research also emphasizes the importance of institutional reform and capacity-building to enhance public trust and accountability within the force. [5]

Existing blood management system in Nepal is manual, cumbersome and inefficient. Most blood banks record the information on blood collection/supply manually in registers. Maintaining blood stock inventory is tedious with laborious back-office paperwork and managing information on availability and shortage of blood is a tall task. A social initiative for a smart, transparent and holistic blood management service from collection to supply. When it comes to blood, right information at the right time can be the answer to a life and death situation. [6]

FIRE AID was founded in 2012 in a café in Kent, where a group of like-minded experts, dedicated to supporting first responders globally, came together with a shared vision. Their goal was to create a central hub that would streamline the exchange of knowledge and resources to enhance emergency response efforts worldwide. In early 2013, like-minded organizations gathered at the House of Commons for a meeting chaired by Jim Fitzpatrick, who was a serving MP at the time. During this meeting, funding was secured through the FCDO to establish FIRE AID's database and website. Jim served as Chair until September 2023. FIRE AID’s first collaborative project was launched in Ukraine, followed closely by projects in Moldova and Tajikistan. [7]

On 14 February 2002, a medical branch was established at the Armed Police Force headquarters in Kathmandu to provide healthcare to APF personnel and their families. Despite limited resources, it offered services like OPD, emergency care, X-ray, physiotherapy, and pharmacy. For specialized care, renowned hospitals like Teaching Hospital and Bir Hospital were consulted. On 15 December 2003, with the expansion of APF offices nationwide, the branch became APF Hospital. By 16 February 2005, due to increased patient volume, the government approved the construction of a new 110-bed hospital at Balambu, Kathmandu, reserving 5 beds each for the National Investigations Department and local residents, with 100 beds allocated for APF staff. [8]

Sketchfab is revolutionizing creativity by making it easy for anyone to publish and discover 3D content online. As the largest platform for immersive, interactive 3D models, it boasts a vibrant community of millions of creators. With seamless integration across major 3D tools, and support for VR, AR, and all devices, Sketchfab ensures a smooth experience for both creators and buyers. Its real-time viewer, model inspector, and robust APIs allow for confident transactions and flexible embedding, empowering a new era of 3D innovation. [9]

Three.js is a powerful and widely used JavaScript library that enables the creation of 3D content for the web, utilizing the capabilities of WebGL. It abstracts the complexities of WebGL, making it easier for developers to render 3D graphics and animations directly in web browsers without needing to deal with low-level code. With Three.js, developers can create interactive 3D scenes, visualizations, and immersive environments that are compatible across all major browsers and devices. The library supports a wide range of features, including lighting, textures, camera control, shaders, and more, making it a go-to tool for 3D web development. [10]

GitHub is a web-based platform for version control and collaborative software development. It uses Git, a distributed version control system, to help developers track changes to code, collaborate on projects, and manage code repositories. GitHub provides tools for branching, merging, and reviewing code, making it easier for teams to work together, even remotely. It also supports open-source projects by allowing anyone to contribute to repositories, share code, and document projects with features like issues, wikis, and pull requests. GitHub has become a central hub for developers, hosting millions of repositories for various programming languages and software projects. [11]

A renderer is a software component or system responsible for converting data into visual output, typically in the context of graphics, web pages, or 3D environments. In graphics, a renderer processes models, textures, lighting, and other visual elements to generate images or animations, often using techniques like ray tracing or rasterization. In web development, a renderer refers to the browser's engine that interprets HTML, CSS, and JavaScript to display a webpage. Essentially, the renderer takes complex data and turns it into something visually perceivable for the user. [12]

Chapter 3: System Analysis and Design

3.1 System Analysis

The system analysis of the system is done by conducting requirement analysis and feasibility analysis as follows:

3.1.1 Requirement Analysis

Requirements analysis is a crucial step for determining the success of a system or software project. Requirements are generally split into two types.

a. Functional requirements

For User:

- User can register and login the system through email and password.
- User can locate nearby services by the help of Haversine Algorithm.
- User can contact to available services from Call Now feature.

For Hospitals:

- Hospital's user can locate nearby users.
- The name of the Hospital will be displayed in the map along with location.
- The operating hours will be displayed.

For Ambulance:

- The ambulance user's locations will be tracked live.
- The ambulance user will be contacted by nearby users.
- The ambulance can locate nearby hospitals and users.

For Police Department:

- Department's user can locate nearby emergency.
- Department can contact to police vehicle crews.
- Department's will be contacted by users.

For Police Vehicles:

- The vehicles crew's location will be tracked live.
- The vehicles crew will be informed when a nearby emergency occurred.
- In emergency situations the vehicles crew's can call for backup from department.

For Fire Department:

- The fire department will be notified or called for emergency.
- The fire department will dispatch vehicles for rescue to the location.
- The fire department can track the live location of fire truck.

For Fire Truck:

- The vehicles crew's location will be tracked live.
- The vehicles crew will be informed when a nearby emergency occurred.
- In emergency situations the vehicles crew's can call for backup from department.

For Blood Bank:

- The name of the Blood Bank will be displayed in the map along with location.
- The Blood bank will be contacted by users and hospital's users for blood.
- The operating hours will be displayed.

For Admin:

- Admin can login to the system.
- Admin can see the total number of accounts.
- Admin will be allowed to manage accounts.
- Admin can view the live tracker.

Use Case Diagram (Hospital and Blood Bank)

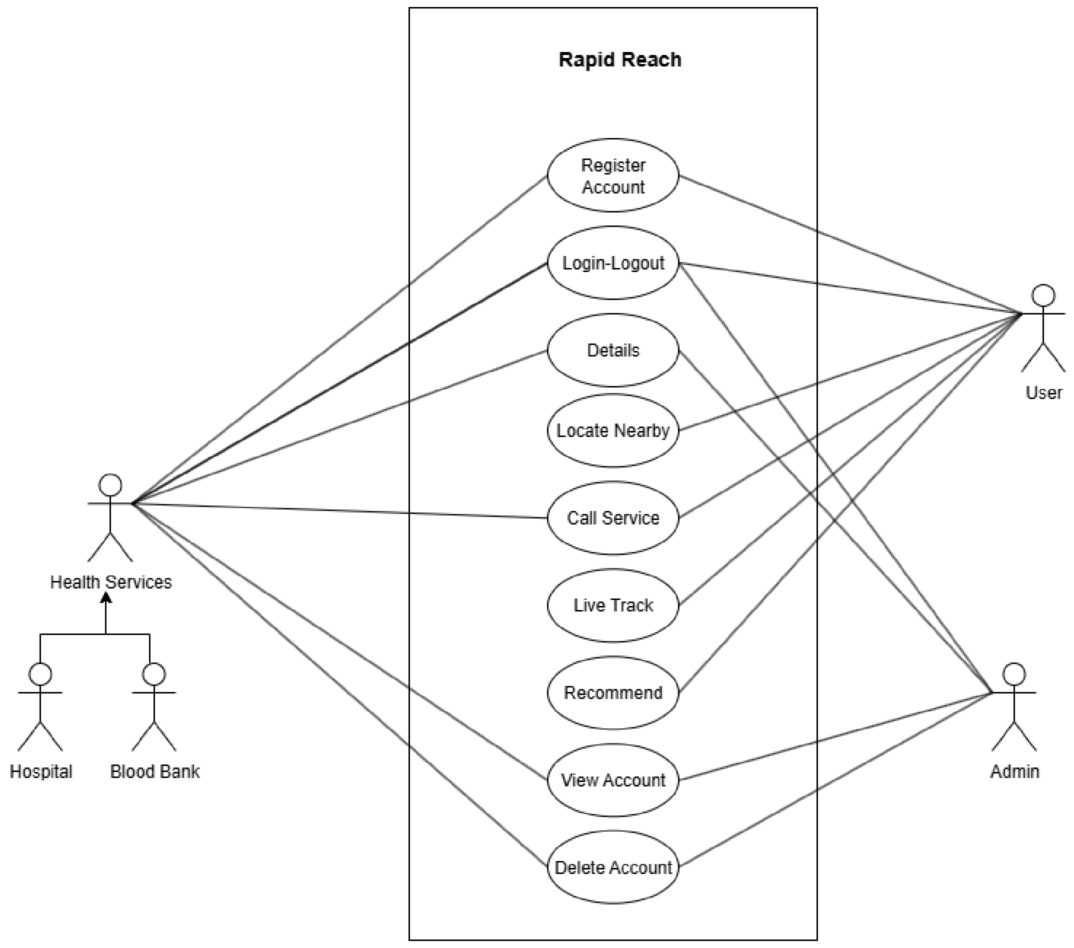


Figure 3.1: Use case diagram for Hospital and Blood Bank of Rapid Reach

This is a use case diagram of Rapid Reach, which explains how users can interact with the system. First, a user can register for a new account and later log in or log out. After logging in, they can manage their personal details, view their account information, or even delete the account if needed. The main services include the ability to locate nearby emergency services, call those services directly, and even track them live. Additionally, users have the option to recommend services to others. Overall, the diagram highlights both the account management features and the emergency service functionalities, showing how the system is designed to support users during urgent situations.

Use Case Diagram (Police department and Fire department)

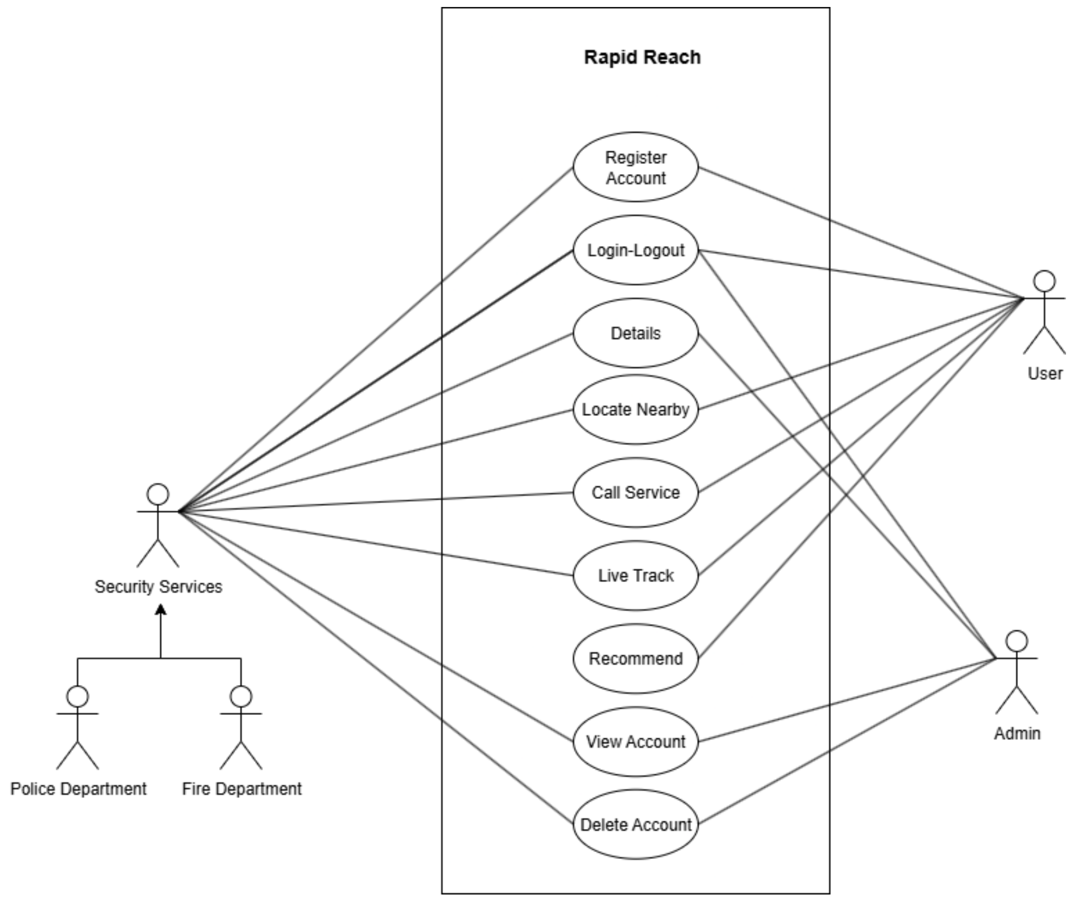


Figure 3.2: Use case diagram for Police and Fire Department of Rapid Reach

This use case diagram shows how different emergency departments interact with the Rapid Reach system. A Police Department can be contacted by users in case of crimes or emergencies, and users can also live with a police response. The Fire Department can be reached directly when there is a fire or hazard, and users can locate nearby fire stations and request help instantly. A Hospital is available for medical emergencies, where users can view hospital details, call for medical help, and track ambulance services live. Similarly, a Blood Bank helps users to locate available blood supplies, request blood, and get contact details. All departments can be recommended by users to others for reliability, and users can still manage their accounts (register, log in/out, view, or delete).

Use Case Diagram (Police vehicle, Fire truck and Ambulance)

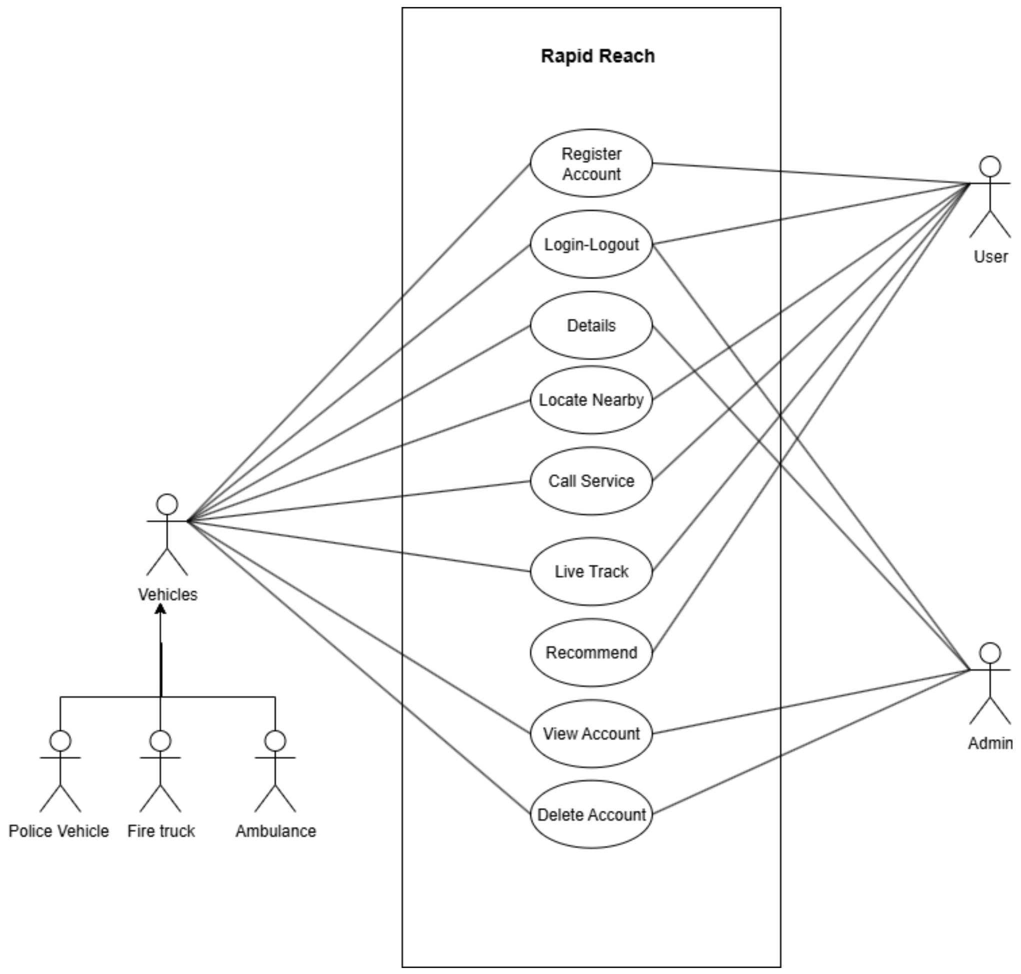


Figure 3.3: Use case diagram for Emergency Vehicles of Rapid Reach

This use case diagram would represent how users can interact with emergency vehicles through the system. A Police Vehicle can be requested when users report an incident, and they can track the vehicle live until it arrives. A Fire Truck can be called during fire emergencies, where users can locate the nearest fire station, request a truck, and monitor its movement in real-time. Similarly, an Ambulance can be contacted for medical emergencies; users can call it directly, track its live location, and view details like availability and estimated arrival time. Just like before, users still have the ability to register/login, manage accounts, and recommend services.

b. Non-Functional Requirement

i. Availability

It will be available as a website. The system works on multiple web browsers like Chrome, Mozilla Firefox and Opera and will be available 24 hours when it is deployed.

ii. Security

The system has accounts for its users and only authorized users can access the system with username and password. The register system contains form validations so that non-authorized users cannot access.

iii. Performance

This system will be designed for smooth performance with optimization and good response.

3.1.2 Feasibility Study

i. Technical feasibility

The website will be developed with the help of MERN Stack along with the help of Haversine and Priority scoring Algorithms for better user experience, which helps to recommend nearby services along with top rated services.

ii. Operational feasibility

The help of current developing tools and deployment tools, the web site can run smoothly in both desktop devices along with mobile devices.

iii. Economic feasibility

As per the research, system doesn't need any funding while it is being developed. However, to build the system it needs software like Visual Studio Code which is free on website.

iv. Schedule feasibility

The scheduled time for the project can be seen below with the help of Gantt Chart table.

Table 3.1: Gantt chart Table for Rapid Reach

Task	May 15	June 30	Aug 15	Aug 26	Sep 27	Oct 3	Nov 10	Estimation
Planning	■	■						20 days
Research		■	■					15 days
Design			■	■				25 days
Implementation				■	■			35 days
Testing					■	■		25 days
Documentation	■	■	■	■	■	■	■	180 days

3.1.3 Object Modeling using Class and Object Diagram

The Class Diagram for the project represents three main classes: User, Post, and Comment. The User class includes attributes like userId, userMail, and userImage, and methods for creating posts and comments. The Post and Drive class has attributes like Id, Content, and references to the User, and it allows adding comments. The Comment class contains attributes like commentId, commentContent, and references to both the User and Post. The diagram also defines the relationships: a User can create many Posts, each Post can have many Comments, and a User can write many Comments.

The relationships among these classes were also clearly defined: a User can create many Posts, each Post can have multiple Comments, and a User can write many Comments. These relationships were represented using one-to-many associations between the respective classes.

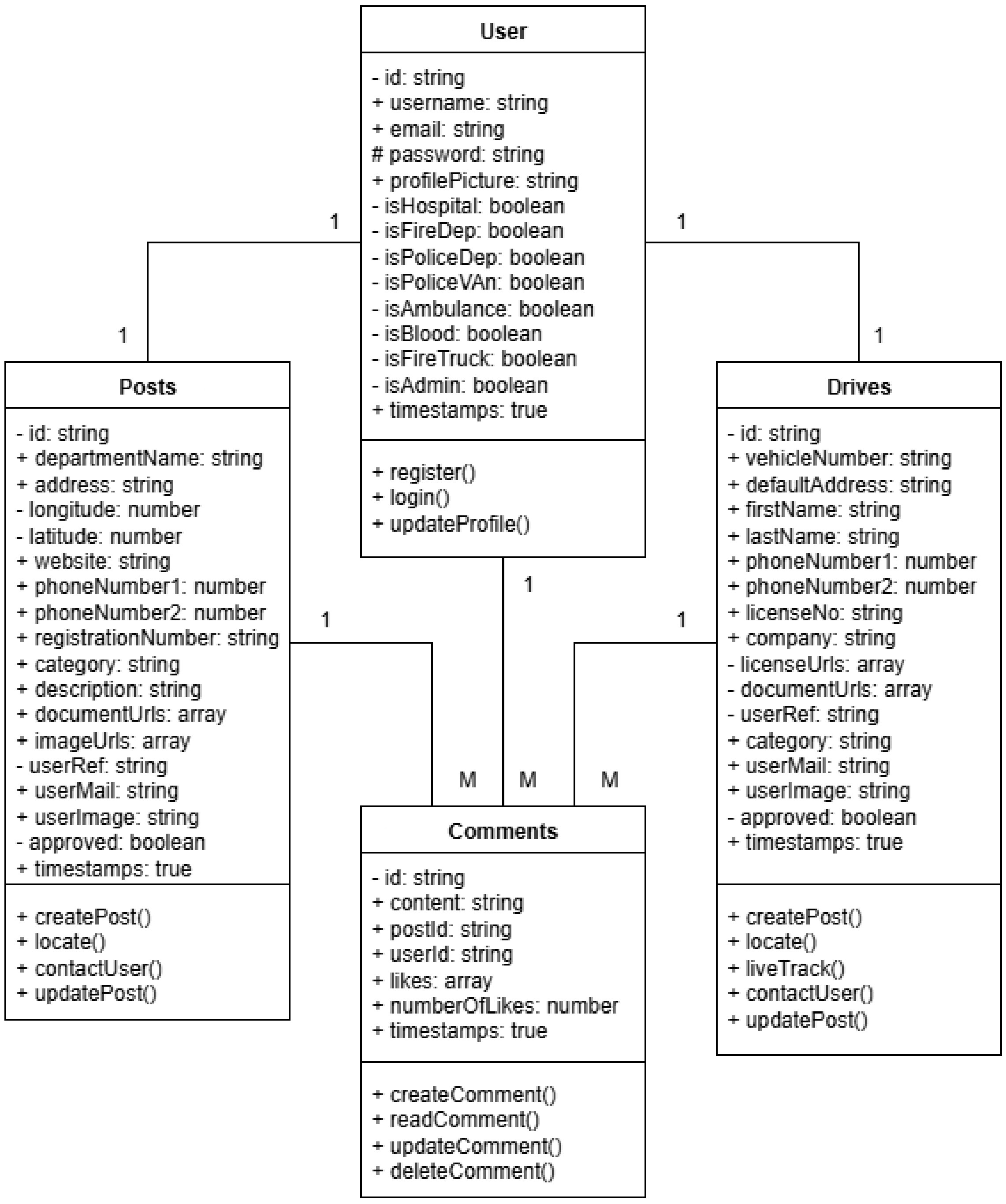


Figure 3.4: Class diagram for Rapid Reach

3.1.4 Dynamic Modelling using State and Sequence Diagram

Dynamic modeling represents how a system behaves and how objects interact over time. It shows the changes in the system's state during execution, complementing static models like class diagrams that focus on structure. Dynamic models, such as sequence diagrams, demonstrate how objects communicate and how data flows between them.

In the context of Rapid Reach, a sequence diagram can be used to illustrate the interactions between the User, Post, Role, Contact, and Authentication (Auth) systems. When a User logs in, the Auth system verifies their credentials, checking the username and password. Once verified, the Auth system returns the User's Role, such as Admin or User, to define the level of access.

The diagram illustrates the flow of actions: first, the login request, then the post creation, and finally the contact addition. Each interaction shows how objects, such as User, Post, and Contact, change state or communicate with each other during these processes.

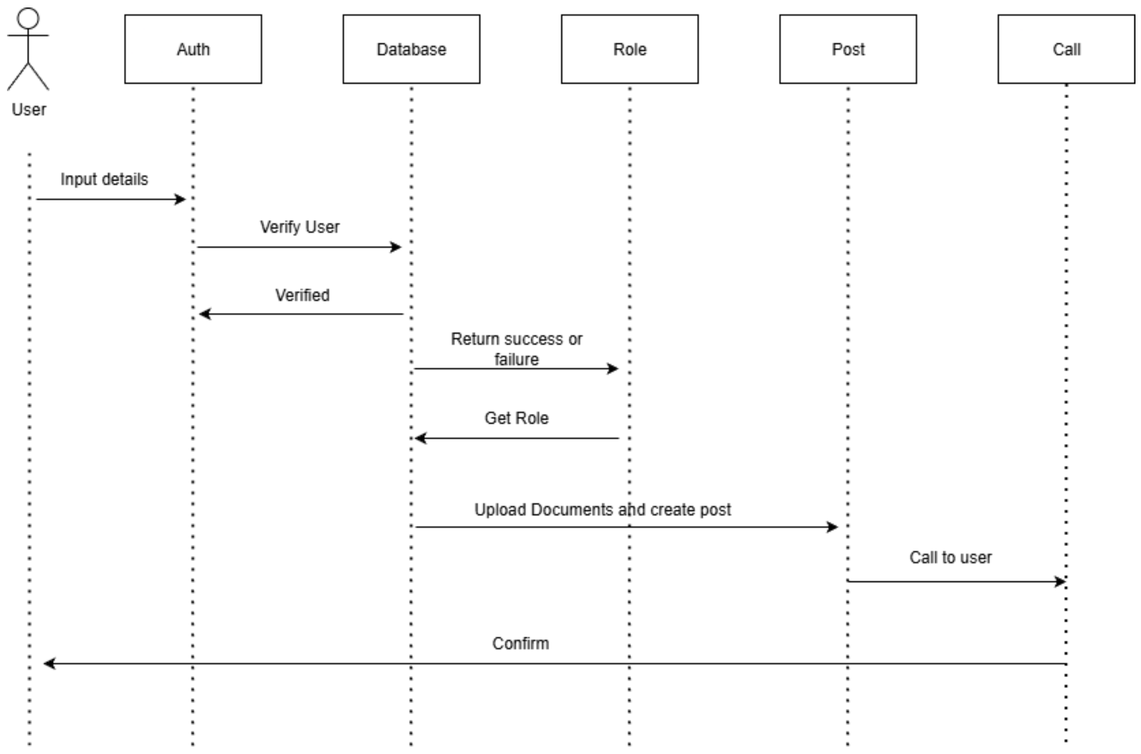


Figure 3.5: Sequence diagram for Rapid Reach

3.1.5 Process Modelling using Activity Diagrams

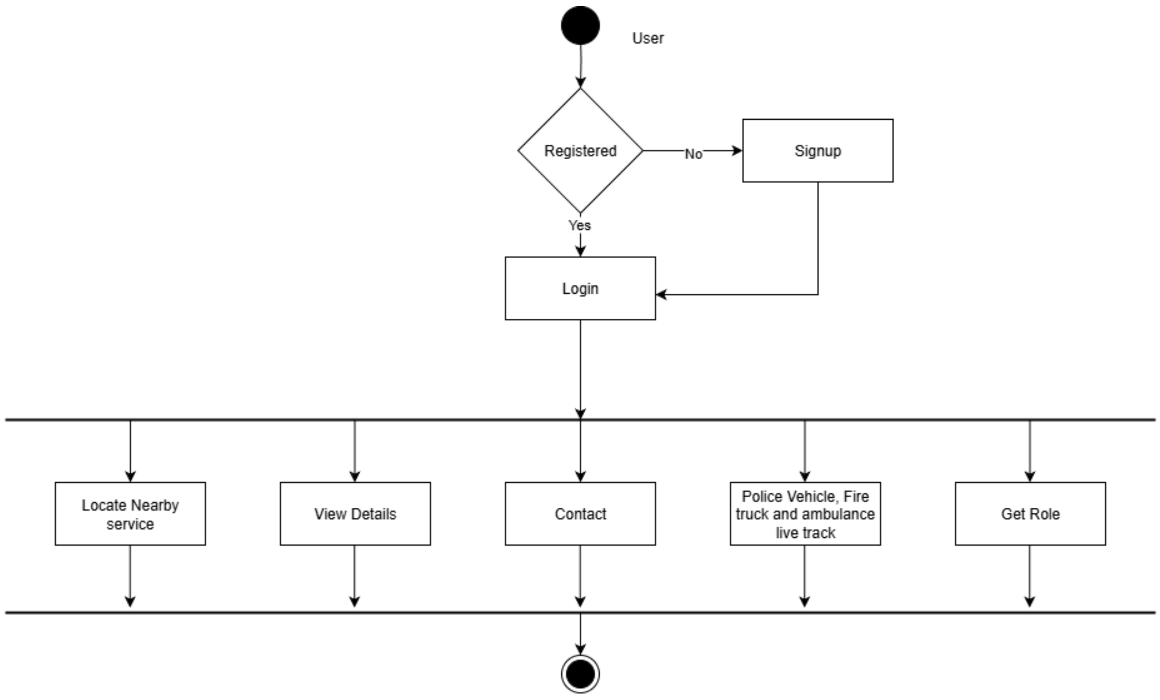


Figure 3.6: Activity diagram of User for Rapid Reach

The diagram models the user process flow in the system. It begins by checking whether a user is registered, if not, they must go through the signup process before logging in, while registered users proceed directly to login. After login, users can access key features such as commenting, viewing maps, contacting others, locating nearby users, and live tracking.

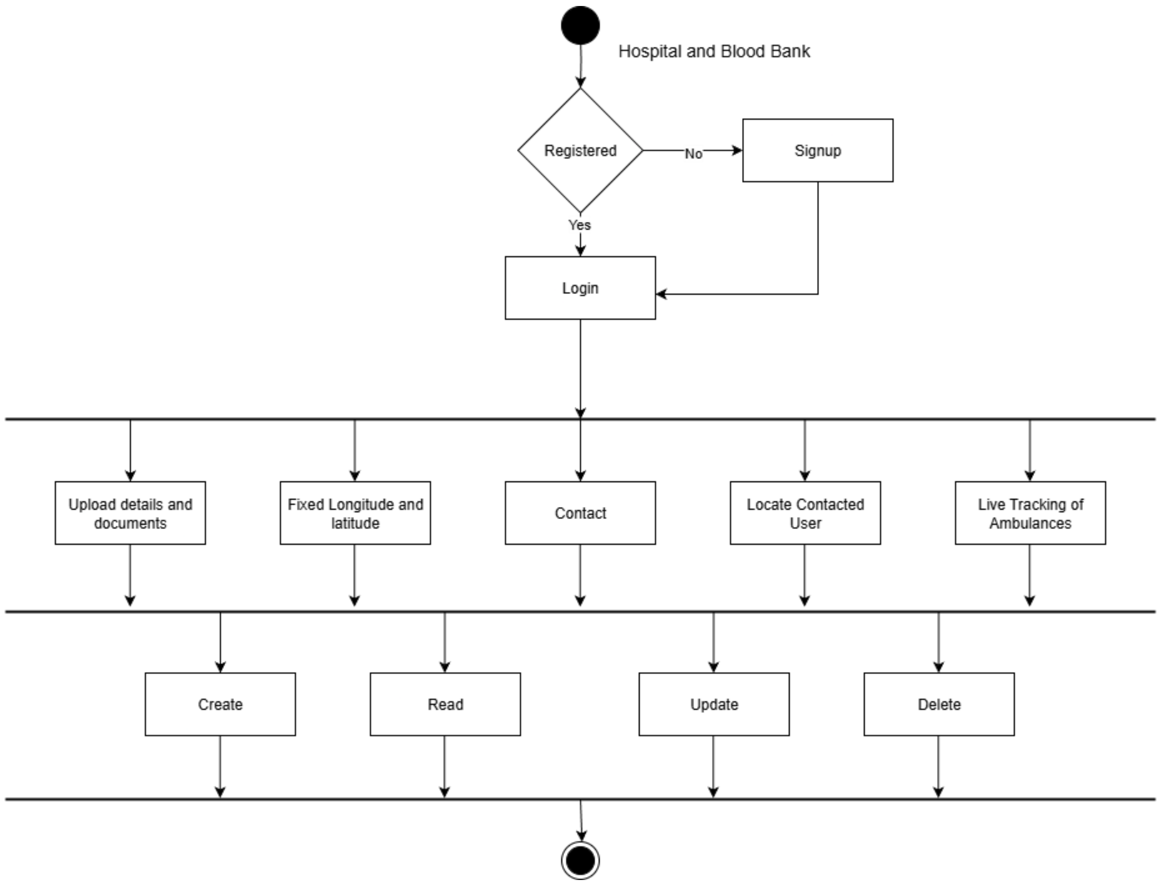


Figure 3.7: Activity diagram of Hospital and Blood Bank for Rapid Reach

The diagram models the Hospital and Blood bank users process flow in the system. It begins by checking whether a user is registered, if not, they must go through the signup process before logging in, while registered users proceed directly to login. After login, users need to upload their documents to access key features such as viewing maps, contacting others, locating nearby users, and live tracking. Finally, the system supports full CRUD operations (Create, Read, Update, Delete), allowing users to manage and interact with their data seamlessly.

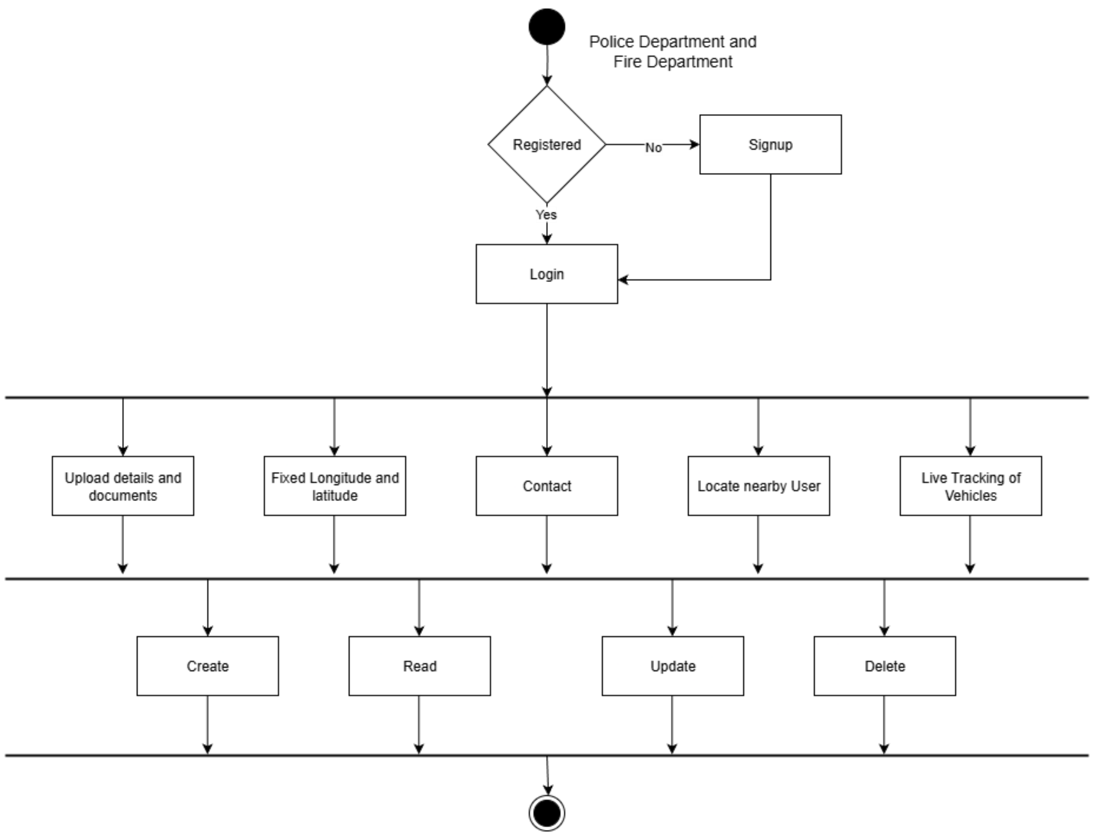

Police Department and Fire Department
Registered
No
Signup
Yes
Login
Upload details and documents
Fixed Longitude and latitude
Contact
Locate nearby User
Live Tracking of Vehicles
Create
Read
Update
Delete

Figure 3.8: Activity diagram of Police and Fire Department for Rapid Reach

The diagram models the Police Department and Fire Department users process flow in the system. It begins by checking whether a user is registered, if not, they must go through the signup process before logging in, while registered users proceed directly to login. After login, users need to upload their documents to access key features such as viewing maps, contacting others, locating nearby users, and live tracking. Finally, the system supports full CRUD operations (Create, Read, Update, Delete), allowing users to manage and interact with their data seamlessly.

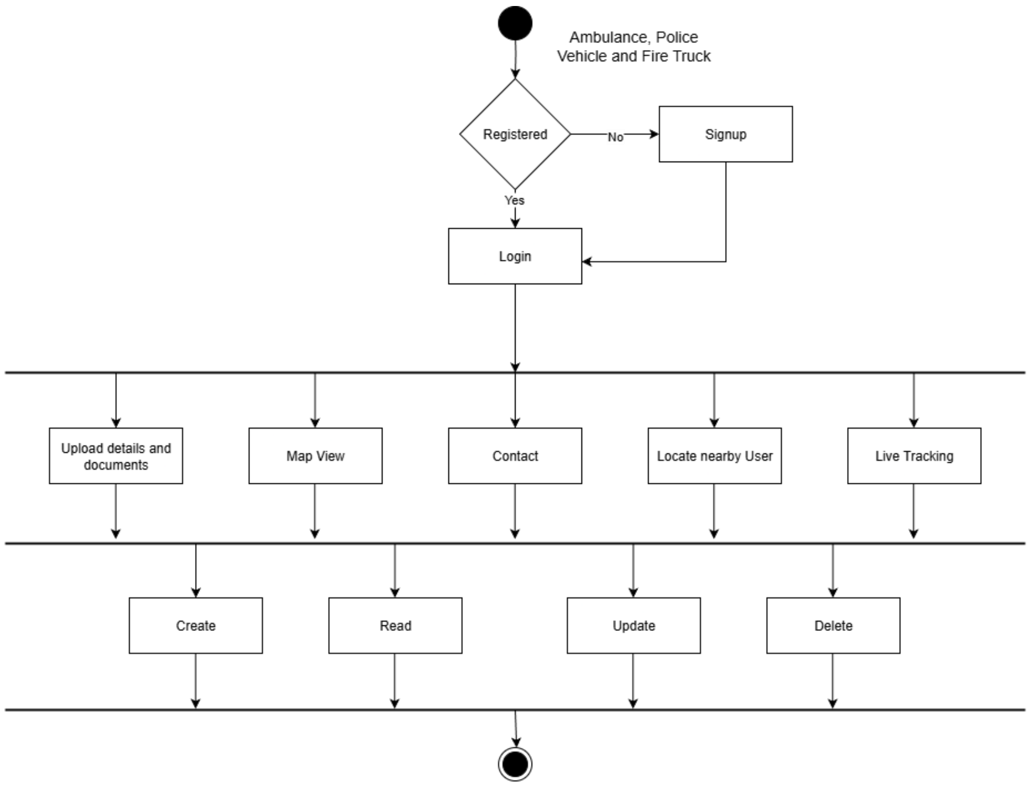


Figure 3.9: Activity diagram of Emergency Vehicle for Rapid Reach

This diagram shows how the emergency locator system is operated. First, it is checked whether the user is registered. If the user is not registered, they are directed to the Signup page; if they are registered, the Login page is displayed. After logging in, several features are made available, such as details and documents being uploaded, the map being viewed, contact options being accessed, nearby users being located, and live tracking being used. At the bottom, the system's information is handled through Create, Read, Update, and Delete (CRUD) operations, which are performed in the background to support all functionalities.

3.2 System Design

3.2.1 Refinement of Class, Object, State, Sequence and Activity diagram

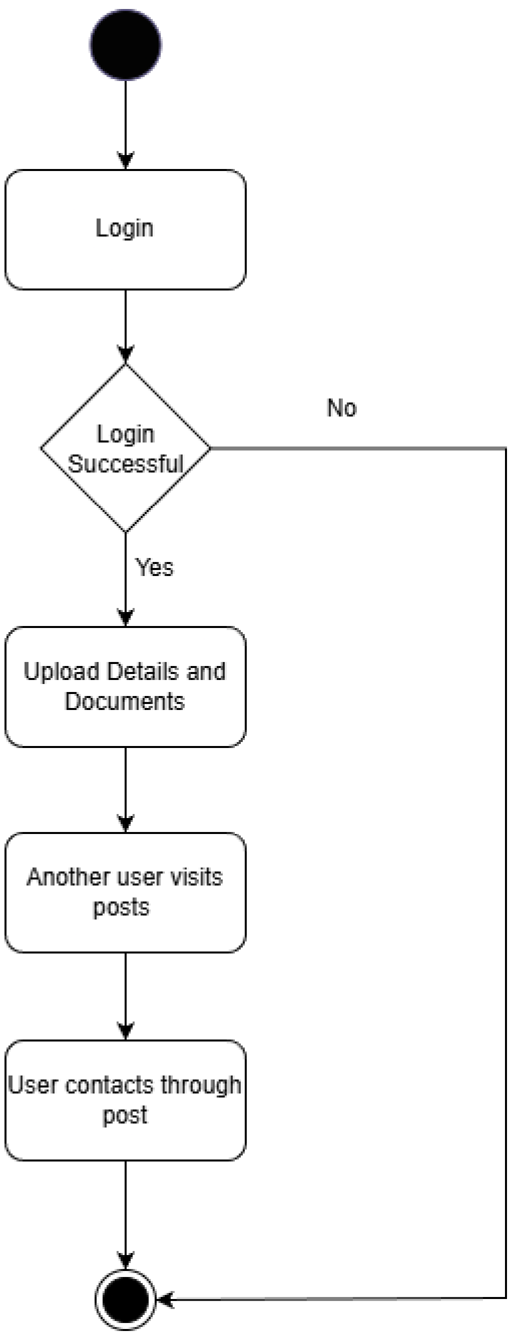

Login
Login Successful
No
Yes
Upload Details and Documents
Another user visits posts
User contacts through post

Figure 3.10: Activity diagram of Rapid Reach

The activity diagram outlines the process of a User logging into the system, creating a Post, and interacting with it. First, the User logs in, and a check is made to determine if the login credentials are valid. If the login is invalid, an error message is displayed, and the process ends. If the login is successful, the User proceeds to create a Post. The Post is then saved in the database. Once saved, another User may view the post and can write a Comment.

The Comment is then saved in the database, and the process ends. The diagram helps visualize the flow of actions and decisions that take place during this sequence.

3.2.2 Component Diagram

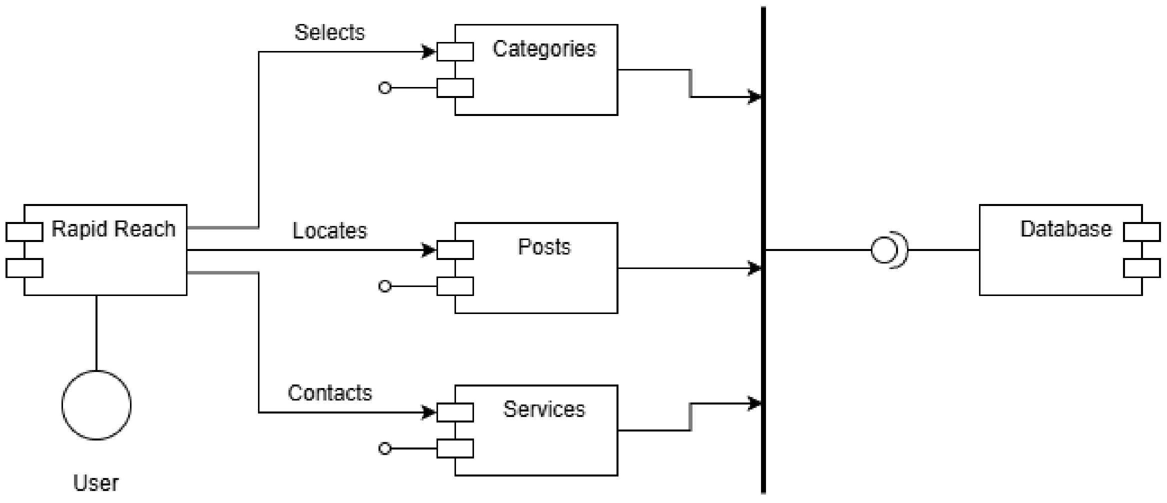


Figure 3.11: Component Diagram of User for Rapid Reach

The component diagram for Rapid Reach outlines a modular system designed for efficient user interaction and content management. It features distinct components for user, post, and location management, all secured through a centralized authentication module. Each of these components communicates with a shared database, ensuring streamlined data flow and integrity.

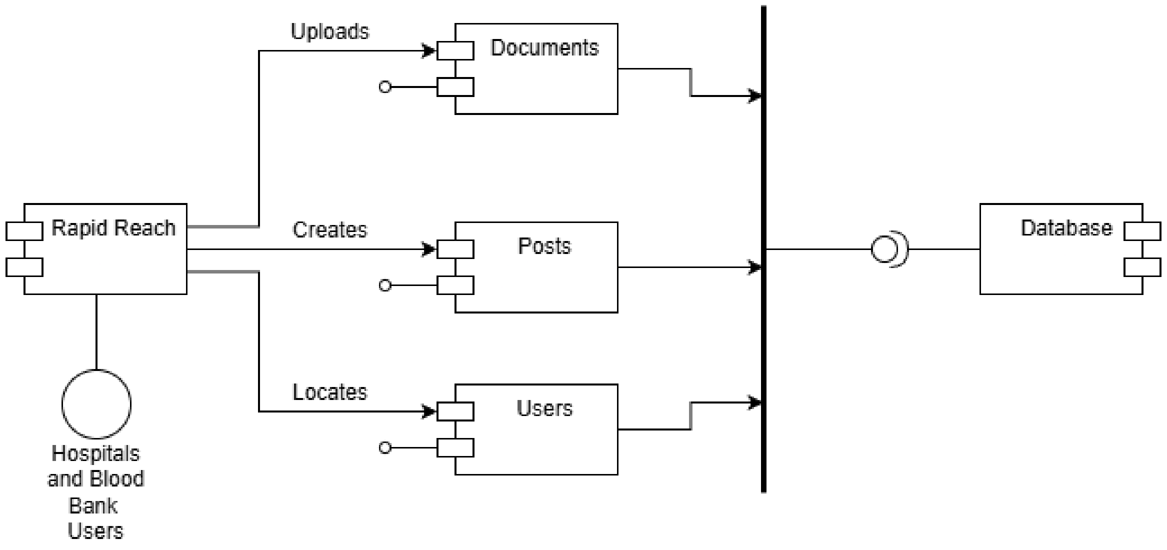


Figure 3.12: Component Diagram of Health Services for Rapid Reach

The above diagram shows the component flow for Hospital and Blood bank users. Users are required to upload documents which are later converted to posts. The documents are stored in the database. The posts then are visible on a map and through which the users can contact the owner of the post. The contacted users can be located on the map.

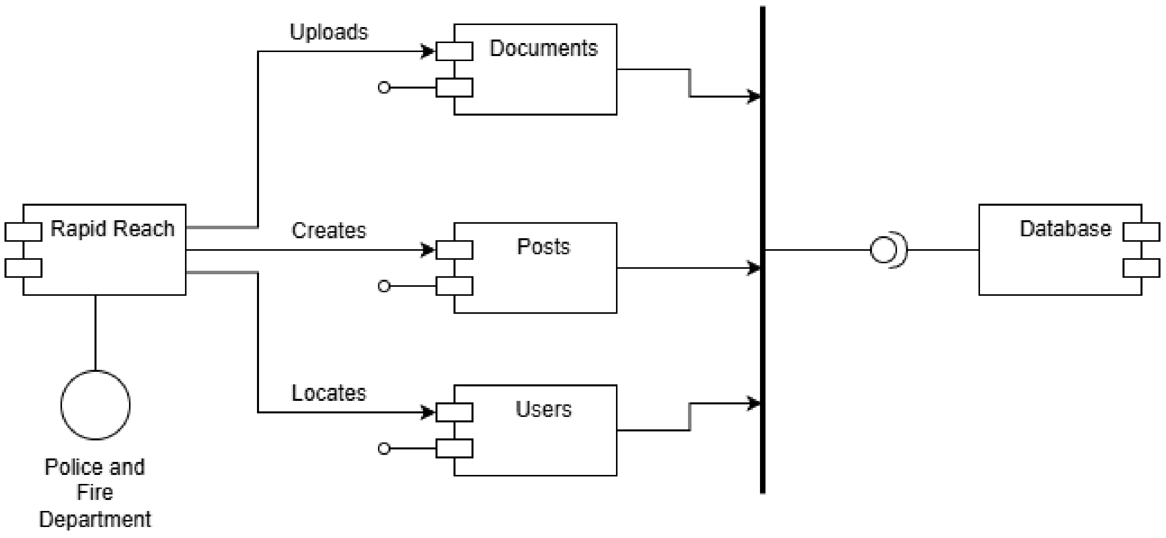


Figure 3.13: Component Diagram of Emergency Services for Rapid Reach

The above diagram shows the component flow for Police and Fire Department users. Users are required to upload documents which are later converted to posts. The documents are stored in the database. The posts then are visible on a map and through which the users can contact the owner of the post. The contacted users can be located on the map.

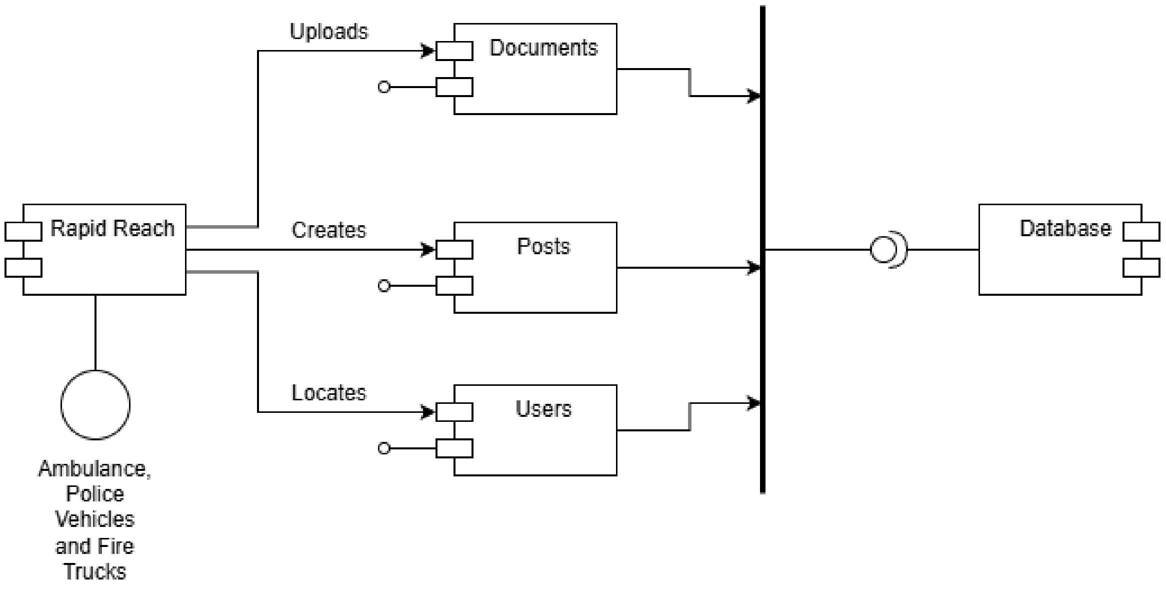


Figure 3.14: Component Diagram of Emergency Vehicles for Rapid Reach

The above diagram shows the component flow for Ambulance, Police Vehicles and Fire Truck users. Users are required to upload documents which are later converted to posts. The documents are stored in the database. The Users then are visible on a map and if only the users are online from which the users can contact the owner of the post. The owner of the post and the users both can be live tracked.

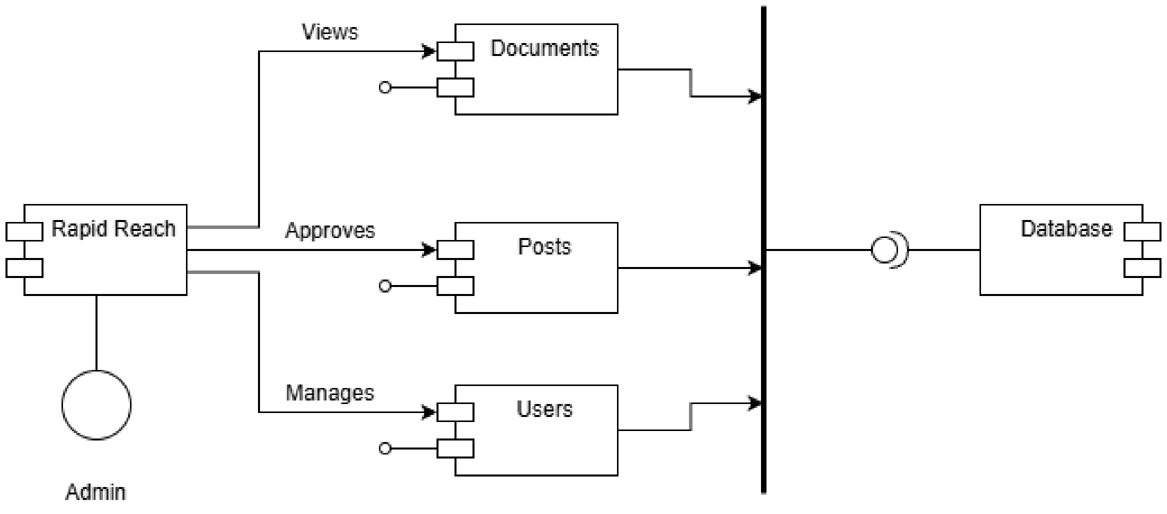


Figure 3.15: Component Diagram of Admin for Rapid Reach

The above diagram shows the component flow for Admin. Admin views the uploaded documents. Admin approves the document which later converts into a post. Admin are allowed to manage users. Admin can contact any users.

3.2.3 Deployment Diagram

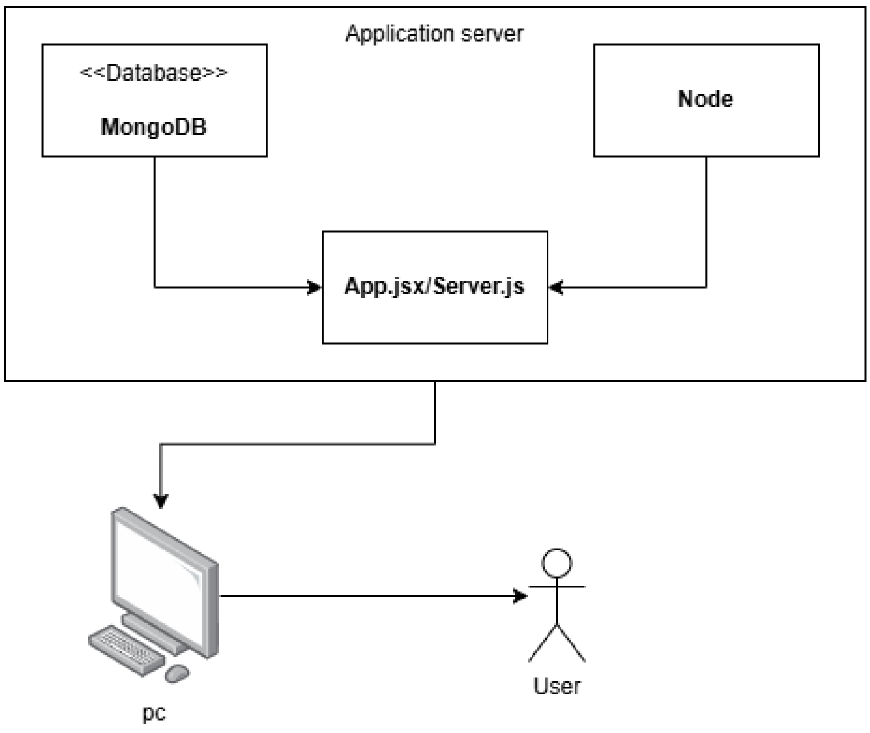


Figure 3.16: Deployment Diagram for Rapid Reach

The deployment diagram illustrates the backend structure of a web application built with Node.js. At its core, the server-side logic is handled by App.jsx or Server.js, which runs within the Node runtime environment. The application connects to a MongoDB database to store and retrieve data. A client device (represented by a PC) interacts with the server, sending requests that are processed by the Node-powered scripts and responded to using data from MongoDB. This setup highlights a typical full-stack JavaScript application using Node.js.

3.2.4 Wireframing

A wireframe is considered the basic blueprint of a digital project, focusing on layout and content placement without visual details like colors or fonts. It serves as a skeletal structure that helps the functionality and flow of a website, app, or dashboard to be mapped out, ensuring that ideas are organized before the design phase begins. The foundation is laid to ensure that everything fits together logically and smoothly.

Homepage:

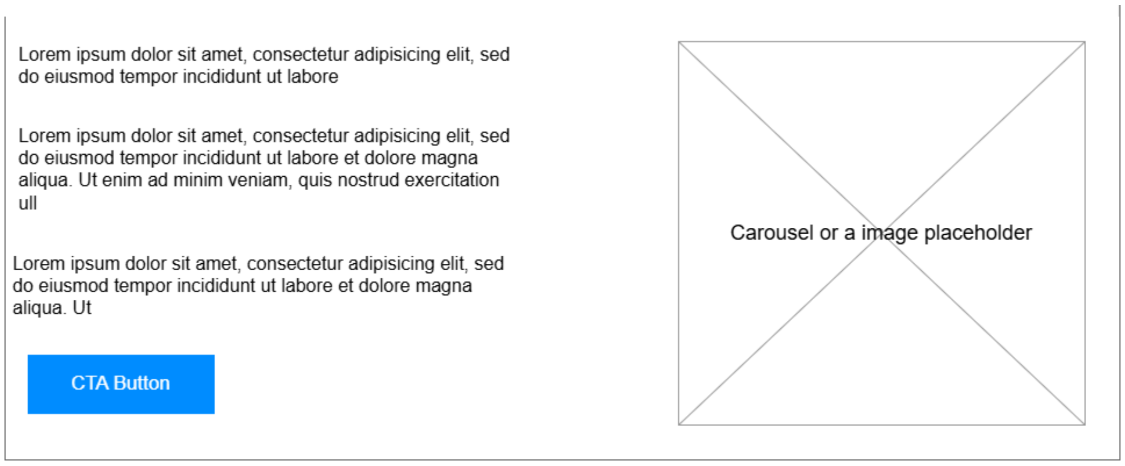

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ull
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
CTA Button
Carousel or a image placeholder

Figure 3.17: Wireframing Design of Homepage

This is a homepage wireframe of the Rapid Reach. The right side of the page is left for image, and a left side of the page is left for short description on website.

Login Page:

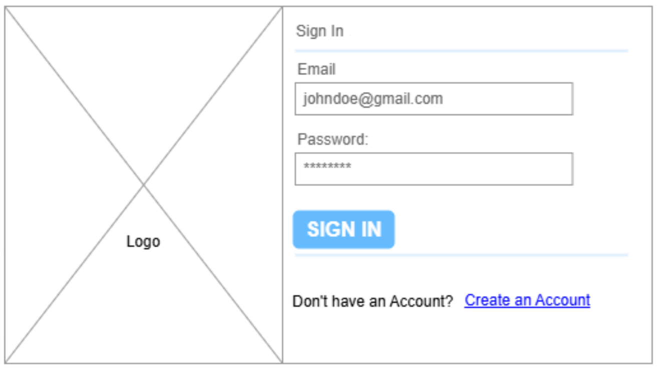


Figure 3.18: Wireframing Design of Login Page

This is a login page wireframe of the Rapid Reach. The right side of the page is left for form, and left side of the page is left for logo on website.

Profile Page:

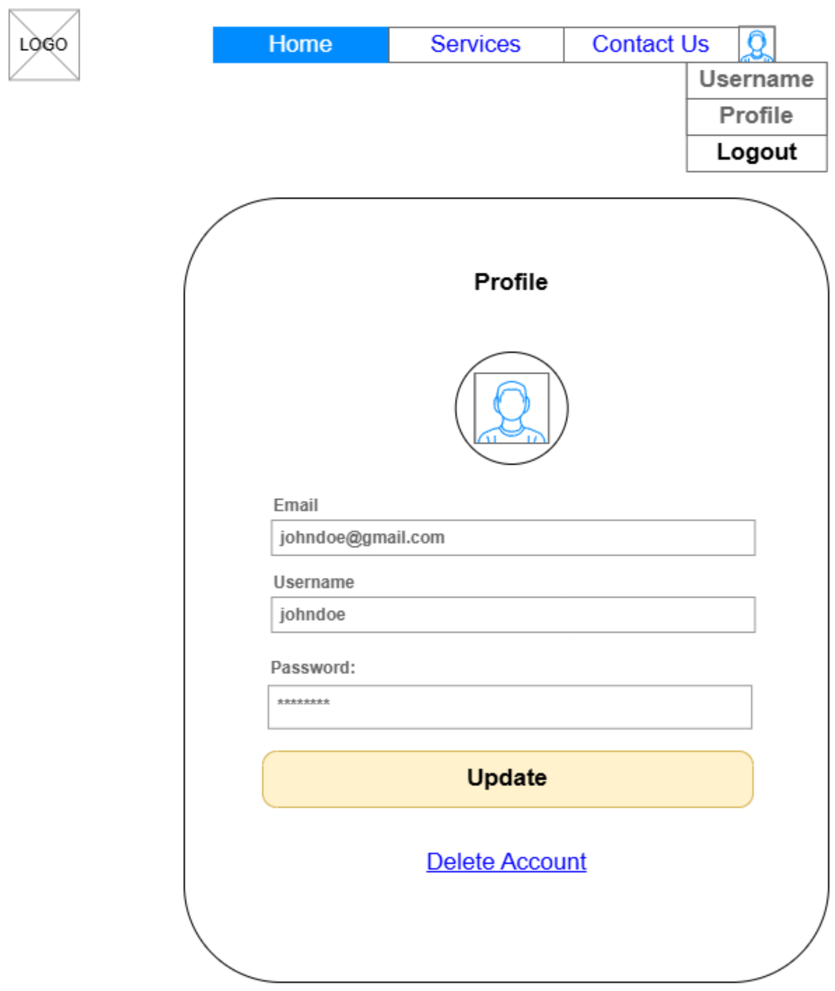


Figure 3.19: Wireframing Design of Profile Page

This is a Profile page wireframe of the Rapid Reach. It is followed by the sidebar which also works as navigation bar for the page on the right side.

Post page:

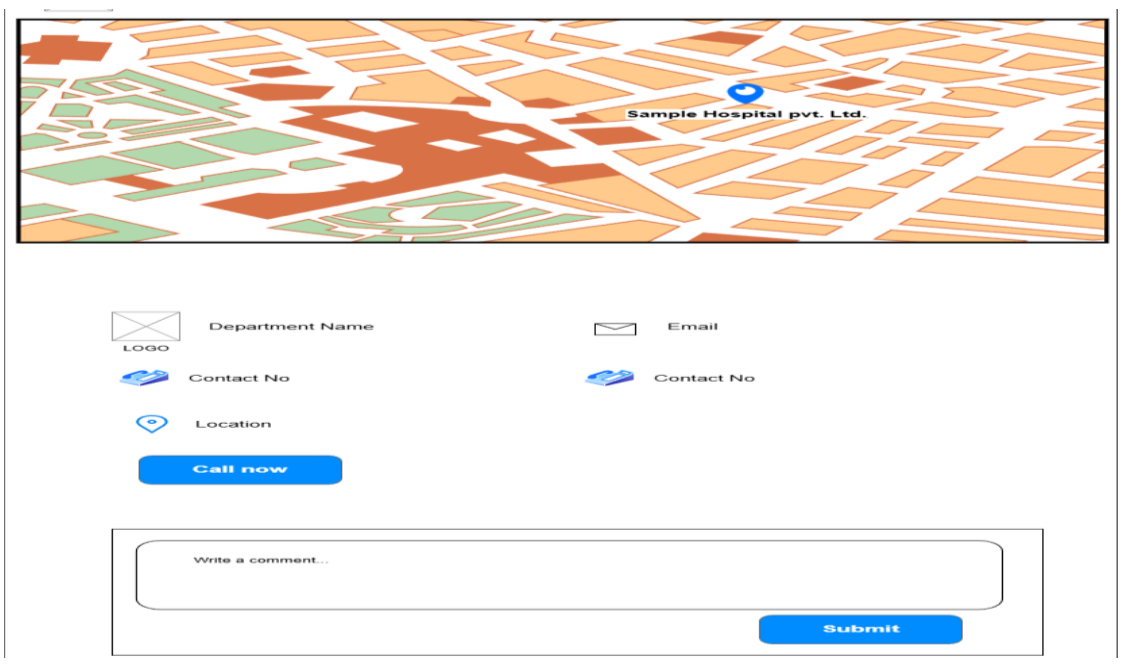


Figure 3.20: Wireframing Design of Post Page

This is a post page wireframe of the Rapid Reach. The upper part of the page shows the map with the location of the post and below are the details of the post followed by the comment section and call now button.

3.2.5 Interface Design (UI Interface/Interface Structure Diagrams)

Homepage:

The page begins with a homepage. The homepage in Rapid Reach contains some details and a 3D model of a city.

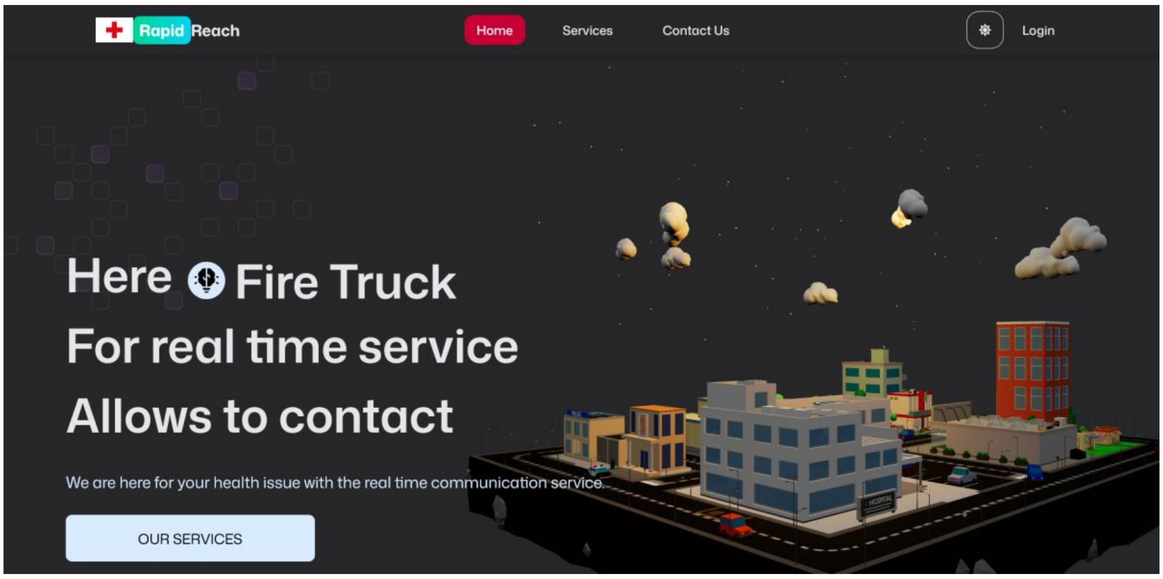


Figure 3.21: Interface Design of Homepage

Login Page:

This is the login page of the website. After this a Cookie is created to store data.

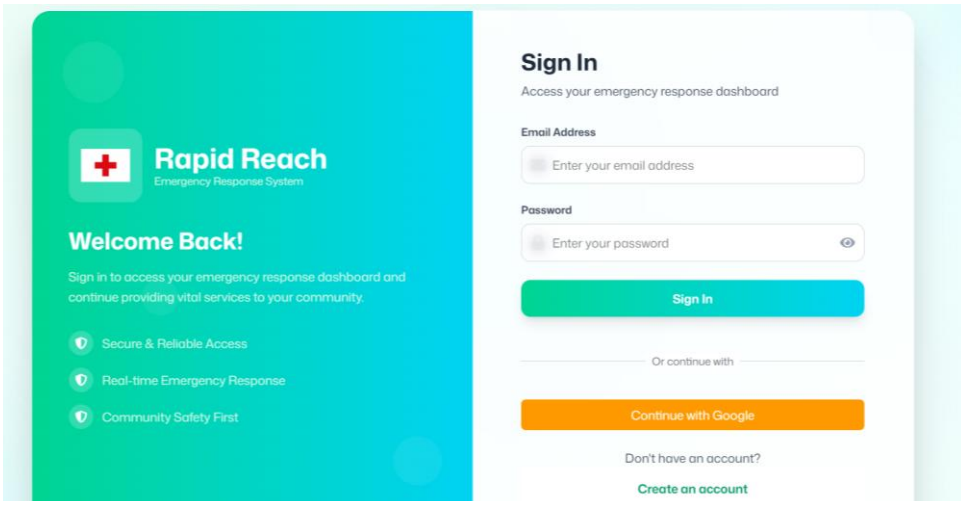
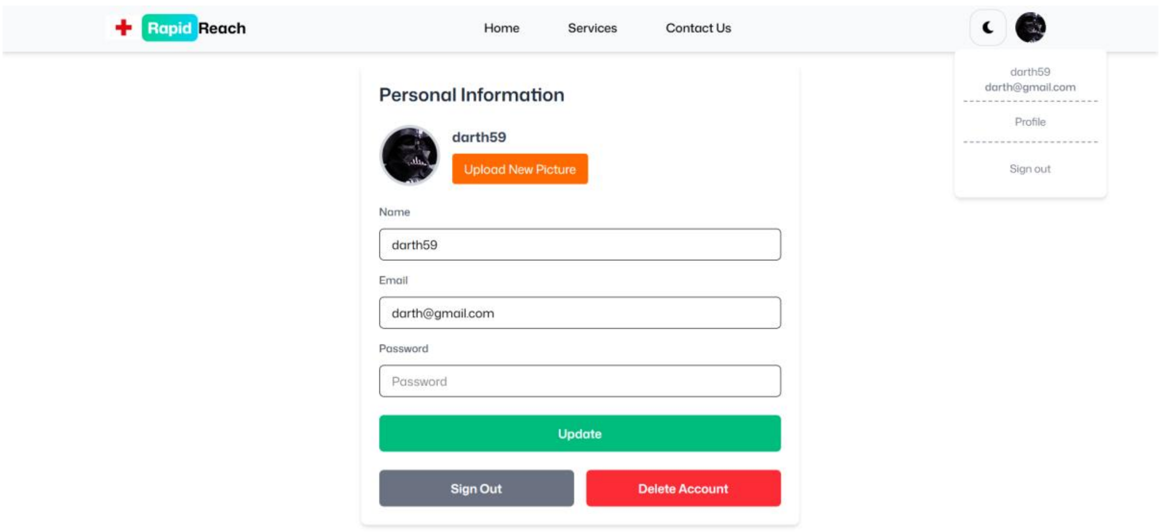


Figure 3.22: Interface Design of Login Page

Profile Page:

The Profile Page shows the information of the login user in which they can Update or delete their account.



Personal Information

darth59

Upload New Picture

Name: darth59

Email: darth@gmail.com

Password: Password

Update

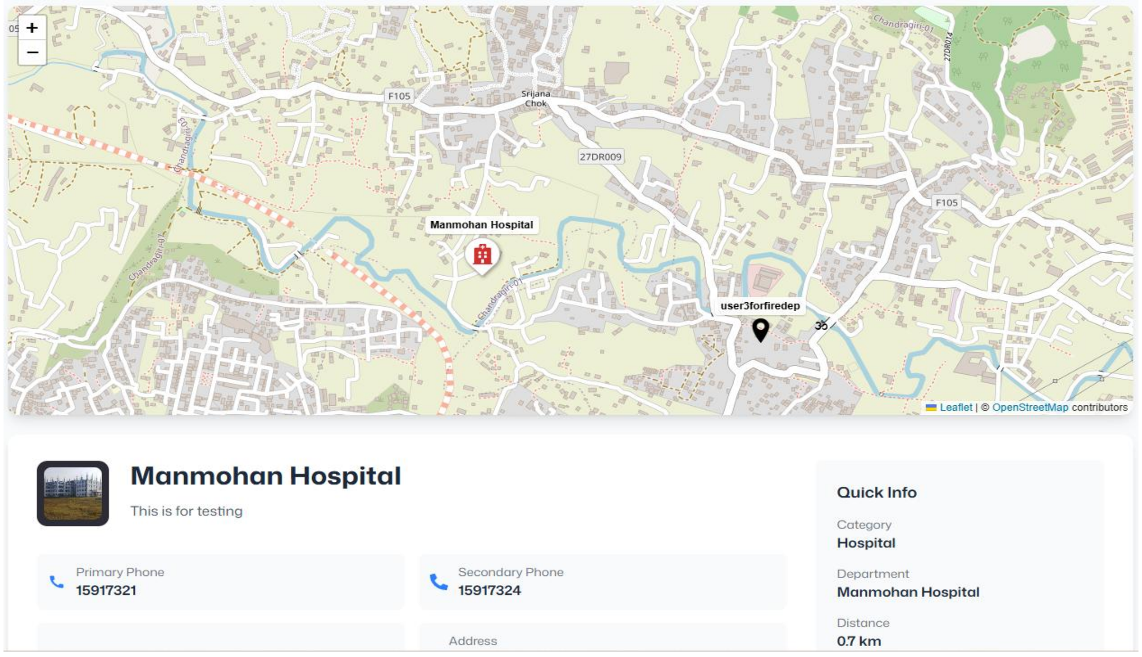
Sign Out

Delete Account

Figure 3.23: Interface Design of Profile Page

Post Page:

The post page shows the location of posts along with the map and live tracking.



Manmohan Hospital

This is for testing

Primary Phone: 15917321

Secondary Phone: 15917324

Address

Quick Info

Category: Hospital

Department: Manmohan Hospital

Distance: 0.7 km

Figure 3 24: Interface Design of Post Page

3.3 Algorithm Details

Algorithms are used in websites to deliver functionality, personalization, performance, and security.

3.4.1 Haversine Algorithm

In the Rapid Reach system, the Haversine algorithm is used to calculate the shortest distance between the user's current location and nearby emergency services. By applying this formula to the latitude and longitude coordinates, accurate distance values are generated, which are then displayed to the user. This ensures that emergency services are listed in order of proximity, allowing quicker access to the nearest available help.

Steps to Implement the Algorithm

Location Data Collection

- The user's real-time geographic coordinates (latitude and longitude) are retrieved using the browser's geolocation API.
- Each service provider or department also stores predefined coordinates in the database.
- These coordinate pairs serve as input for distance calculation.

Coordinate Conversion

- The latitude and longitude values are converted from degrees to radians.
- This is essential because trigonometric functions in the Haversine formula operate on radian values.

Calculate Differences

- The differences in latitude and longitude between two points are computed:
- $\Delta lat = lat_2 - lat_1$
- $\Delta lon = lon_2 - lon_1$

These represent the angular distance between the points.

Apply the Haversine Formula

- $\phi 1, \phi 2$ be the latitudes of point 1 and point 2 (in radians)
- $\lambda 1, \lambda 2$ be the longitudes of point 1 and point 2 (in radians)

- r is the radius of the Earth (mean radius ≈ 6,371 km)

Then the Haversine formula is:

$$a = sin^2\left(\frac{\Delta\emptyset}{2}\right) + \cos(\emptyset 1).\cos(\emptyset 2).sin^2\left(\frac{\Delta}{\lambda}\right)$$

$$c = 2.archtan2(\sqrt{a}, \sqrt{1-a})$$

$$d = r.c$$

Where:

- R = 6371 km (radius of the Earth)
- d = distance between the two locations in kilometers

Assigning Distance to Service Providers

After calculating the distance, the value is attached to each service post.

```
distance = haversineDistance(userLocation, {
   lat: post.latitude,
   lng: post.longitude
})
```

Sorting Based on Distance

- The calculated distance values are used to sort services from nearest to farthest.
- This helps users easily identify the closest emergency service.

Filtering and Ranking

- The system can filter services by distance thresholds (e.g., within 10 km or 50 km).

Final Output

- The nearest services or top-ranked services based on distance are presented to the user.

3.4.2 Priority Scoring Algorithm

In the Rapid Reach system, a priority scoring method is applied to rank emergency services based on multiple factors such as distance, availability, verified status, and user ratings. Scores are automatically calculated, and services are displayed in order of priority, ensuring that the most reliable and accessible options are highlighted first. This approach enables users to make faster and more informed decisions during emergencies.

Steps to Implement the Algorithm

Real-Time Availability Detection

- The system uses WebSockets to detect whether a service provider (Hospital, Police, Fire, Ambulance, etc.) is currently online.
- When a provider connects to the platform, the socket server marks them as "online".
- Availability is a crucial factor because emergency help is only useful if providers are active.

Data Collection

The system collects the following service attributes:

From each provider (post):

- Category (Hospital, Blood Bank, Fire Department, etc.)
- Location (latitude & longitude)
- Approval status
- Created date
- Online/offline status
- Department details

From the user:

- User’s current GPS location (via Geolocation API)

Distance Calculation (Haversine Formula)

To measure how close a service is to the user, the system calculates the geographical distance using the Haversine Distance Formula.

Priority Score Calculation

Each service is evaluated and given a score between 0 and 5, based on these weighted factors:

a. Online Availability (+1.5)

- Services that are currently online receive a higher score.

b. Approval Status (+1.25)

- Approved and verified services are considered more reliable.

c. Freshness of the Post (+1.0 → 0.5)

- The system calculates how recent the service post is:
- Posts within 30 days receive full freshness score
- Posts between 30–90 days get a reduced score
- Older posts receive minimal or zero freshness score
- This ensures that updated services are ranked higher.

d. Distance Factor (0.75 → weighted)

- Services within 10 km receive maximum distance scores
- Services between 10–50 km receive gradually reduced score
- Farther services receive no distance score
- This allows the system to prioritize services near the user.

e. Category Priority Weighting (0.3 → 0.5)

Different service types have different priority levels:

Hospital	0.5
Fire Department	0.5
Police Department	0.5

Blood Bank	0.4
Ambulance	0.4
Fire Truck	0.3
Police Vehicle	0.3

Priority Score Formula:

- n be the number of factors (criteria)
- xi be the score for the i-th factor
- wi be the weight assigned to the i-th factor (where $\sum_i^n w_i = 1$

$$Priority\ Score = \sum_{i=1}^{n} (w_i . x_i)$$

or written out:

$$Priority\ Score = w_1 x_1 + w_2 x_2 + \cdots + w_n . x_n$$

5. Sorting & Ranking

- After calculating the priority score for all services:
- Services are sorted in descending order based on priority score.
- The top 3 services are shown under “Best Services for You”.
- Remaining services are displayed according to user-selected filters or sorting preferences such as:
  - Nearest
  - Latest
  - Oldest
  - Availability

6. Filter and Search Processing

The system also supports:

- Searching by name, address, category
- Filtering by availability (online/offline)
- Sorting by distance or date
- URL-based dynamic filtering for sharing or navigation
- Filtering ensures users can quickly find the type of service they need.

7. Final Recommendation

- The system presents the user with:
- Top Ranked (High Priority) Services
- Nearby services based on distance
- Filtered results based on user input

Chapter 4 Implementation and Testing

4.1 Implementation

4.1.1 Tools Used

Various tools were implemented for the development of this project. Following are the tools that are used for the development of this project:

ReactJs

ReactJS is a popular open-source JavaScript library used for building user interfaces, especially for single-page applications. Developed by Facebook, it allows developers to create reusable UI components that update efficiently when data changes. React uses a virtual DOM to improve performance and makes it easier to manage the state of an application. It's widely used for building fast, dynamic, and responsive web applications.

Node.js

Node.js is an open-source, server-side JavaScript runtime built on Chrome's V8 engine. It allows developers to run JavaScript outside the browser, making it possible to build fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model, which makes it efficient and suitable for real-time applications like chat apps, APIs, and streaming services. It's commonly used with frameworks like Express.js for building web servers and backend services.

Express.js

Express.js is a lightweight and flexible web application framework for Node.js. It simplifies the process of building web servers and APIs by providing a set of easy-to-use tools and features, such as routing, middleware support, and request/response handling. Express allows developers to quickly create robust and scalable backend applications, and it's widely used in building RESTful APIs and full-stack web apps when combined with frontend frameworks like React.

CSS

Tailwind CSS is a utility-first CSS framework used for rapidly building custom user interfaces. Instead of writing custom CSS, developers use pre-defined utility classes directly in HTML to style elements. This approach makes styling faster, consistent, and easier to maintain. Tailwind is highly customizable and works well with modern frontend frameworks like React, Vue, and Angular. It helps developers create responsive and clean designs without leaving their HTML.

Mongodb

MongoDB is a popular open-source NoSQL database designed for storing large volumes of unstructured or semi-structured data. Unlike traditional relational databases, MongoDB stores data in flexible JSON-like documents (called BSON), which makes it easier to work with dynamic data structures. It is highly scalable, supports high performance, and is commonly used in modern web applications for storing user data, posts, comments, and more. MongoDB integrates well with Node.js and is often used in the MERN stack

Multer

Multer is a Node.js middleware used with Express to handle file uploads, especially multipart/form-data. It processes incoming files, stores them in memory or on disk, and makes them accessible through req.file or req.files. This allows developers to easily manage user-uploaded content such as images, documents, and media within their applications.

Socket.io

Socket.IO is a JavaScript library that enables real-time, bidirectional communication between web clients and servers. Built on top of WebSockets, it allows data to be sent and received instantly without refreshing the page. Socket.IO is commonly used in applications like chat apps, live notifications, online games, and collaborative tools. It works with Node.js on the backend and can be easily integrated with frontend frameworks like React or plain JavaScript to create interactive, real-time user experiences.

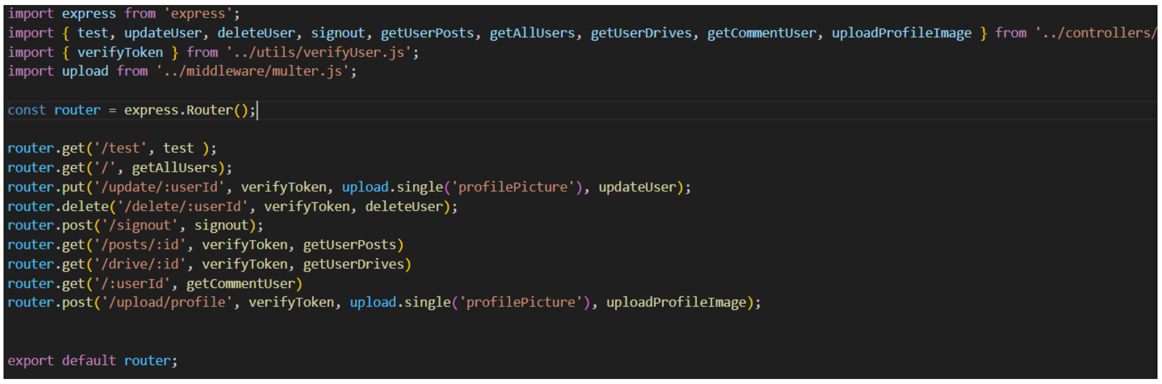
WebRTC

WebRTC (Web Real-Time Communication) is an open-source technology that enables real-time audio, video, and data sharing directly between browsers and devices without needing plugins or third-party software. It allows users to make video calls, voice calls, or share files peer-to-peer over the internet.

4.1.2 Implementation Details of Modules

The system’s modules are shown below:

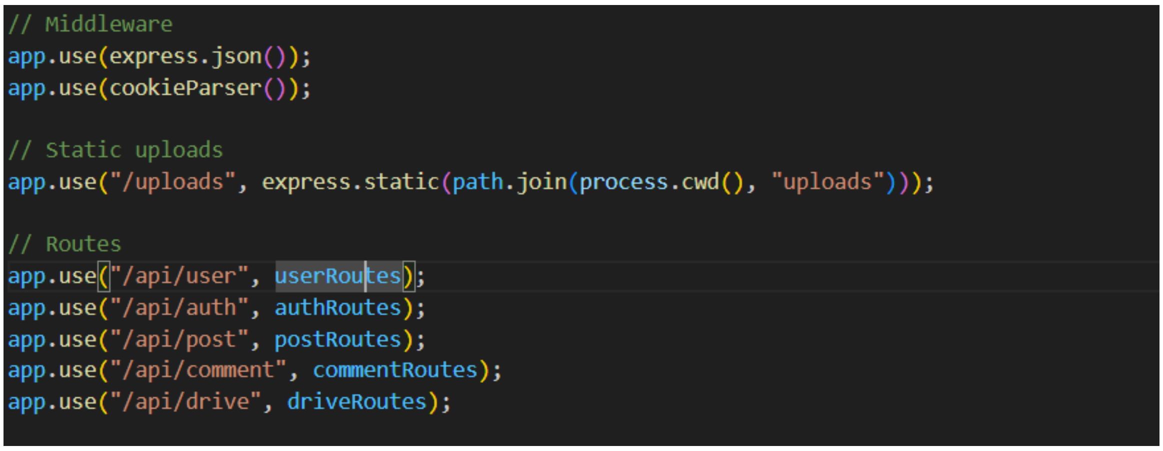
- **User Route**



```
import express from 'express';
import { test, updateUser, deleteUser, signout, getUserPosts, getAllUsers, getUserDrives, getCommentUser, uploadProfileImage } from '../controllers/
import { verifyToken } from '../utils/verifyUser.js';
import upload from '../middleware/multer.js';

const router = express.Router();

router.get('/test', test );
router.get('/', getAllUsers);
router.put('/update/:userId', verifyToken, upload.single('profilePicture'), updateUser);
router.delete('/delete/:userId', verifyToken, deleteUser);
router.post('/signout', signout);
router.get('/posts/:id', verifyToken, getUserPosts)
router.get('/drive/:id', verifyToken, getUserDrives)
router.get('/:userId', getCommentUser)
router.post('/upload/profile', verifyToken, upload.single('profilePicture'), uploadProfileImage);


export default router;
```

Listing 4.1: Code snippet to User Route

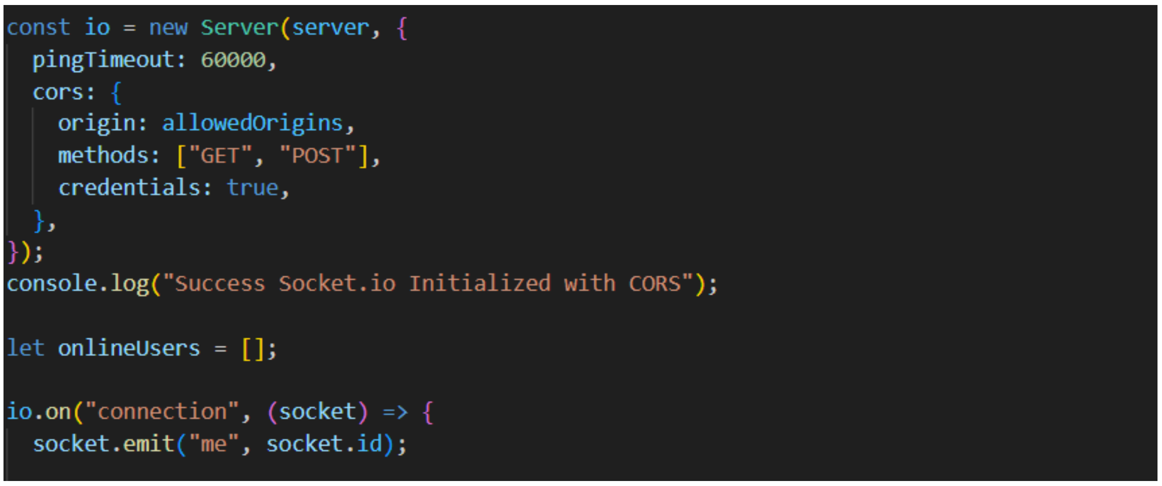
- **Backend Index**



```
// Middleware
app.use(express.json());
app.use(cookieParser());

// Static uploads
app.use("/uploads", express.static(path.join(process.cwd(), "uploads")));

// Routes
app.use("/api/user", userRoutes);
app.use("/api/auth", authRoutes);
app.use("/api/post", postRoutes);
app.use("/api/comment", commentRoutes);
app.use("/api/drive", driveRoutes);
```

Listing 4.2: Code snippet to Backend Index

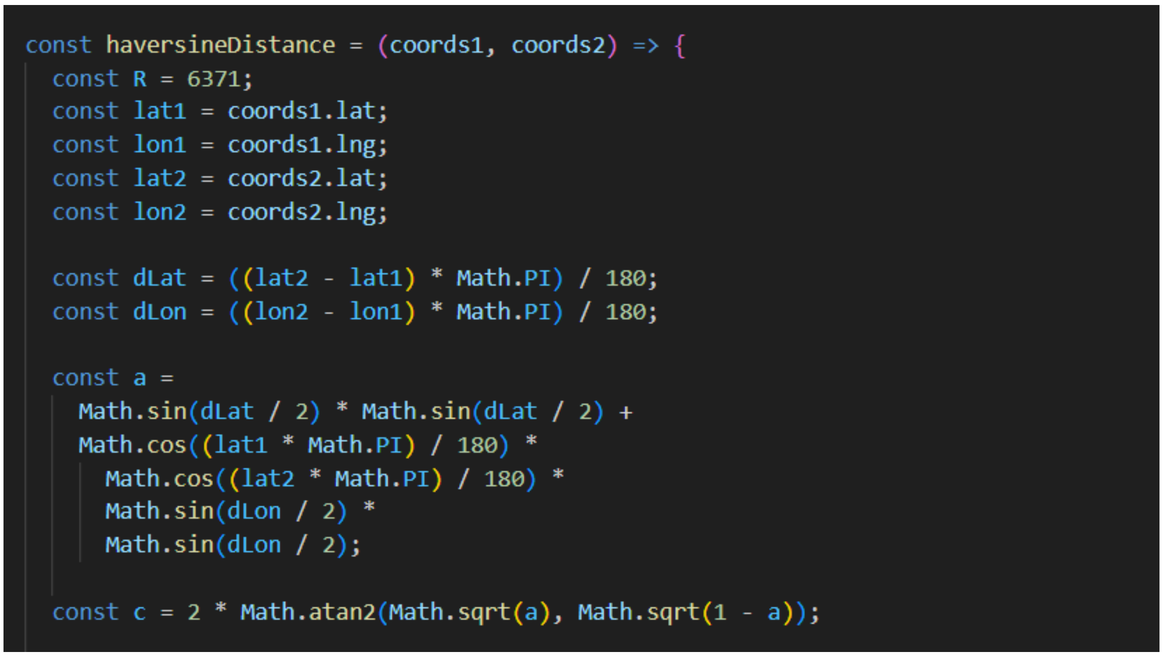
- **Socket Connection**



```
const io = new Server(server, {
  pingTimeout: 60000,
  cors: {
    origin: allowedOrigins,
    methods: ["GET", "POST"],
    credentials: true,
  },
});
console.log("Success Socket.io Initialized with CORS");

let onlineUsers = [];

io.on("connection", (socket) => {
  socket.emit("me", socket.id);
```

Listing 4.3: Code snippet to Socket Connection

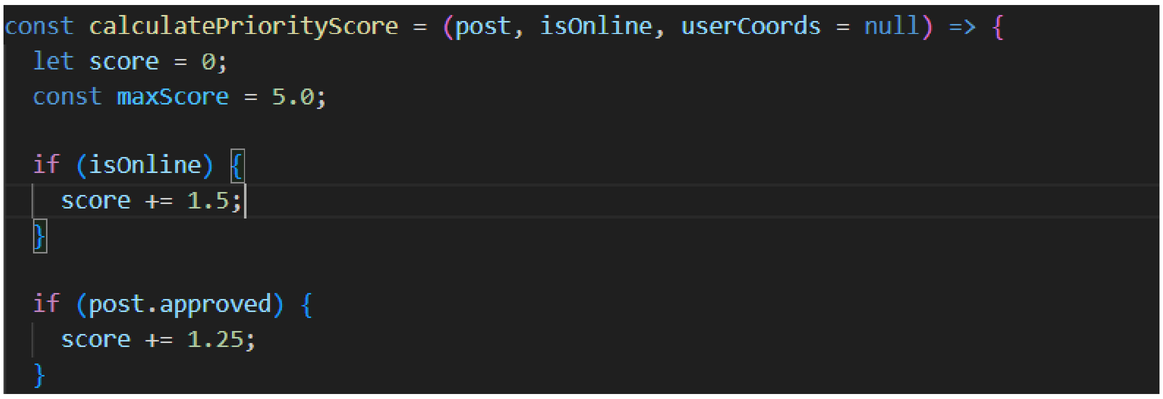
- **Haversine Algorithm**



```
const haversineDistance = (coords1, coords2) => {
  const R = 6371;
  const lat1 = coords1.lat;
  const lon1 = coords1.lng;
  const lat2 = coords2.lat;
  const lon2 = coords2.lng;

  const dLat = ((lat2 - lat1) * Math.PI) / 180;
  const dLon = ((lon2 - lon1) * Math.PI) / 180;

  const a =
    Math.sin(dLat / 2) * Math.sin(dLat / 2) +
    Math.cos((lat1 * Math.PI) / 180) *
      Math.cos((lat2 * Math.PI) / 180) *
      Math.sin(dLon / 2) *
      Math.sin(dLon / 2);

  const c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
```

Listing 4.4: Code snippet to Haversine Algorithm

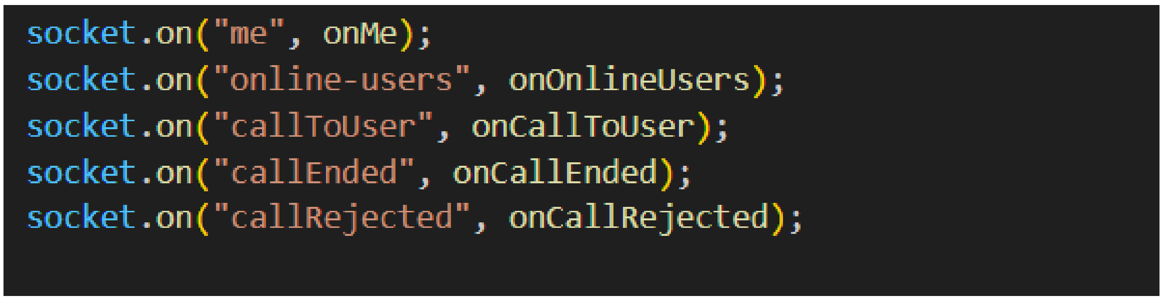
- **Priority Scoring Algorithm**



```
const calculatePriorityScore = (post, isOnline, userCoords = null) => {
  let score = 0;
  const maxScore = 5.0;

  if (isOnline) {
    score += 1.5;
  }

  if (post.approved) {
    score += 1.25;
  }
```

Listing 4.5: Code snippet to Priority Scoring Algorithm

- **Video Call**



```
socket.on("me", onMe);
socket.on("online-users", onOnlineUsers);
socket.on("callToUser", onCallToUser);
socket.on("callEnded", onCallEnded);
socket.on("callRejected", onCallRejected);
```

Listing 4.6: Code snippet to Video call

4.2 Testing

4.2.1 Unit Testing

Unit testing in Rapid Reach is a crucial step to ensure the reliability and accuracy of individual modules within the system. It involves designing and executing test cases that simulate different scenarios for both users and emergency service providers. By conducting unit tests, we verify that each feature, such as location detection, service search, availability display, and call functionality, performs as expected. The system is developed in a modularized pattern, and each module is tested independently to confirm proper functionality. Until the desired accurate output is achieved from a module, testing and refinement continue. Input forms and search fields are also thoroughly tested to ensure they do not accept invalid or incomplete data, thereby maintaining system integrity and reliability.

Table 4.1: Unit testing for Rapid Reach

S.N.	Test Name	Input	Expected Outcome
1.	Open Application	http://localhost:5173/	Homepage

2.	Enter invalid username, email, password, confirm password & click register button	Username= user2forbloodbank
		Email=godfather59@gmail.com
		Password=
			Fill out required field

3.	Enter credentials & click register button	Username=user2forbloodbank
		Email=user2@gmail.com
		Password=user2
			User registered successfully

4.	Enter email & invalid password	Email= user2@gmail.com
		Password=wilson2022
			User not found & refresh the fields in same page

	& click login button		

5.	Enter email & valid password & click login button	Email= user2@gmail.com
		Password= user2
			Login successful & redirect to home page

6.	Upload documents and Create post	http://localhost:5173/create-post	Please fill up all the details.
7.	Redirect to post page and map view	http://localhost:5173/post/post._id	Details displayed along with Map.
8.	Haversine Algorithm implementati-on	http://localhost:5173/gridview?category=hospital	On selected nearest the nearest service along with the distance is shown
9.	Open Admin	http://localhost:5173/dashboard?tab=dashb	Rapid Reach admin login

10.	Enter email & invalid password & Click login button	Email=notadmin@gmail.com
		Password=admin2022
			User not found & redirect to login page

11.	Enter email & valid password & Click login button	Email=gp16373@gmail.com
		Password=gp16373
			Welcome & redirect to Homepage.

4.2.2 System Testing

System testing plays a crucial role in ensuring the smooth integration and functionality of all modules and components within Rapid Reach. This testing phase involves validating the system to ensure seamless cooperation among its various parts.

Table 4.2: System testing for Rapid Reach

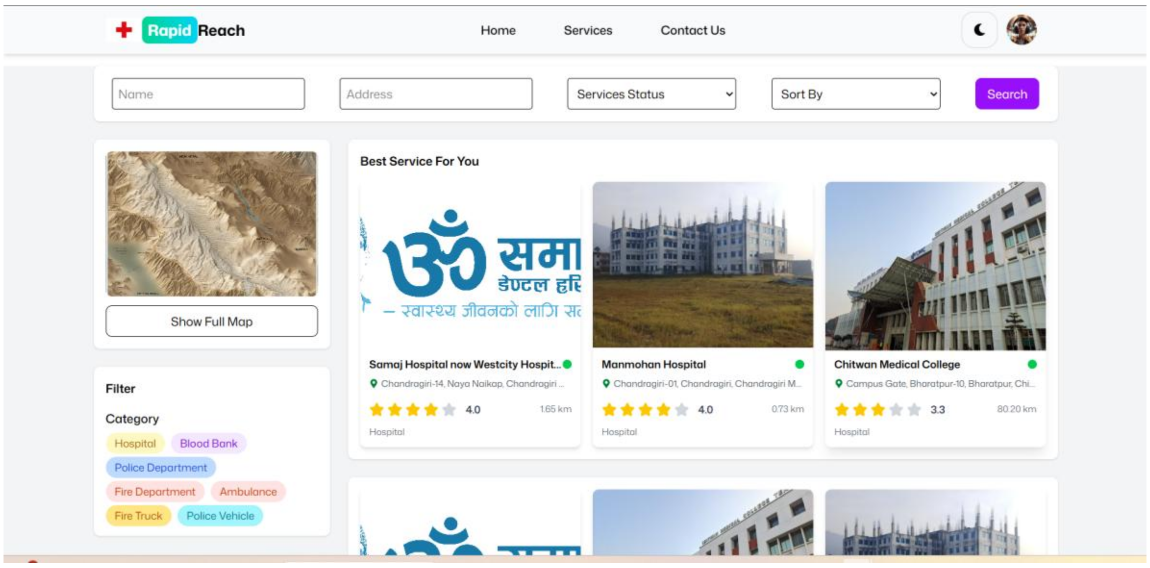
S.N.	Test Case	Expected Outcome	Test Result
1.	Launch website	Homepage	Pass
2.	Entered valid email and password	Go to home page with user image in headings.	Pass
3.	Did not filled all the inputs.	Please fill out this field	Pass
4.	Upload image other than JPEG and PNG	Error: Please upload an image file	Pass
5.	Selected Nearest in Dropdown.	Nearest is shown with the distance	Pass
6.	On click the call now button in post page.	A popup appears with username and image for call.	Pass

4.3 Result Analysis

The system is designed to provide real-time tracking of emergency vehicles, including ambulances, fire trucks, and police vehicles. It aims to reduce emergency response times by incorporating a "call now" feature, allowing immediate assistance to be requested. Additionally, the system recommends nearby departments such as hospitals, fire stations, and police stations, along with the appropriate emergency vehicles, ensuring rapid coordination and response in critical situations.

Output 1 (Services Page)

The Service page displays all the posts that are approved by the admin along with the Priority Scoring in stars. The page also allows the user to search the post.



Best Service For You

Samaj Hospital now Westcity Hospit...
Chandragiri-14, Naya Naikap, Chandragiri ...
4.0 1.65 km
Hospital

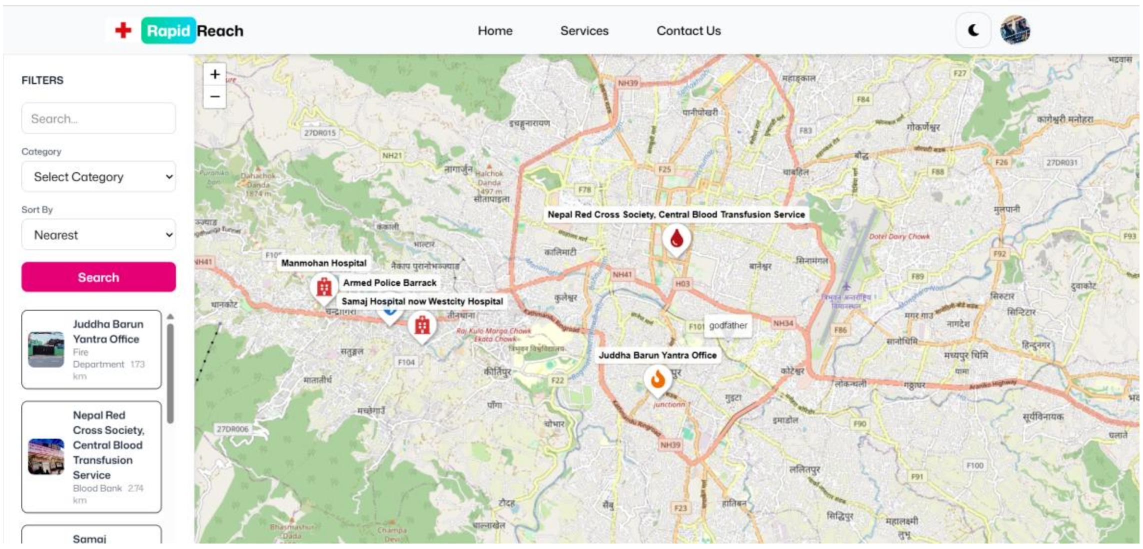
Manmohan Hospital
Chandragiri-01, Chandragiri, Chandragiri M...
4.0 0.73 km
Hospital

Chitwan Medical College
Campus Gate, Bharatpur-10, Bharatpur, Chi...
3.3 80.20 km
Hospital

Figure 4.1: Service Page with Priority Scoring Algorithm

Output 2 (Map Page)

The map page allows user to live track the location of the departments, Vehicles and users. It also allows the user to select the nearest service from the sidebar which uses the Haversine Algorithm.

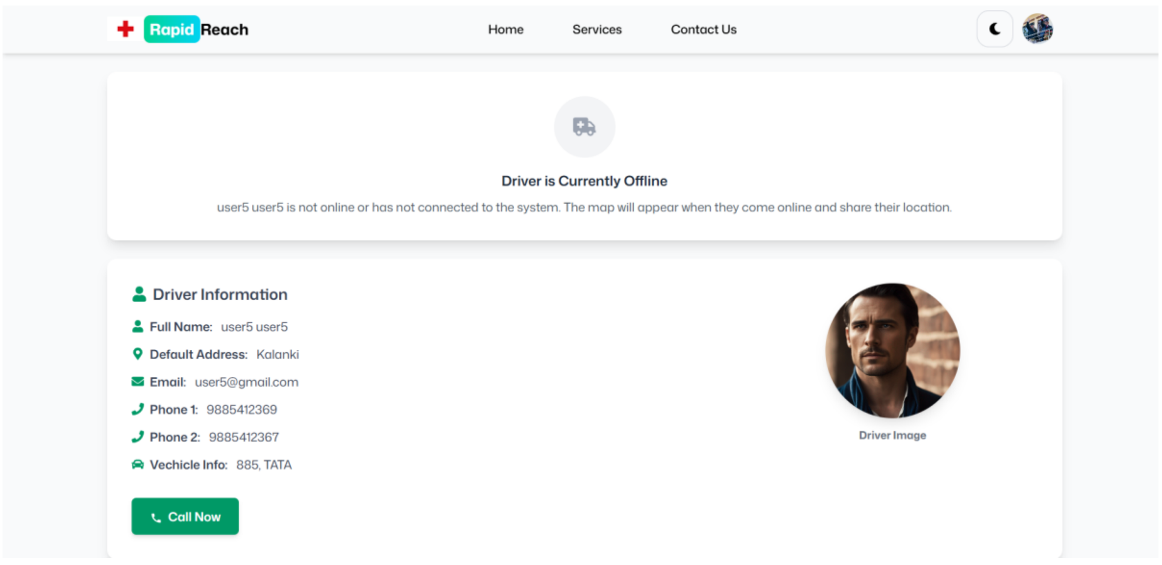


Rapid Reach
Home
Services
Contact Us
FILTERS
Search...
Category
Select Category
Sort By
Nearest
Search
Juddha Barun Yantra Office
Fire Department 1.73 km
Nepal Red Cross Society, Central Blood Transfusion Service
Blood Bank 2.74 km
Samaj
Nepal Red Cross Society, Central Blood Transfusion Service
Manmohan Hospital
Armed Police Barrack
Samaj Hospital now Westcity Hospital
Juddha Barun Yantra Office

Figure 4.2: Map Page with Haversine Algorithm

Output 3 (Drive Post Page)

The Drive Post page allows the user to live track the location of the vehicle if the user is online and shares the location of the vehicle. It also allows the user to directly call the vehicle.


Rapid Reach

Home Services Contact Us

Driver is Currently Offline

user5 user5 is not online or has not connected to the system. The map will appear when they come online and share their location.

Driver Information

Full Name: user5 user5

Default Address: Kalanki

Email: user5@gmail.com

Phone 1: 9885412369

Phone 2: 9885412367

Vechicle Info: 885, TATA

Call Now

Driver Image

Figure 4.3: Drive Post Page using Socket

Output 4 (Video Call)

The video call is done with the help of the Web RTC after the post page is clicked. It allows the user to communicate with each other.

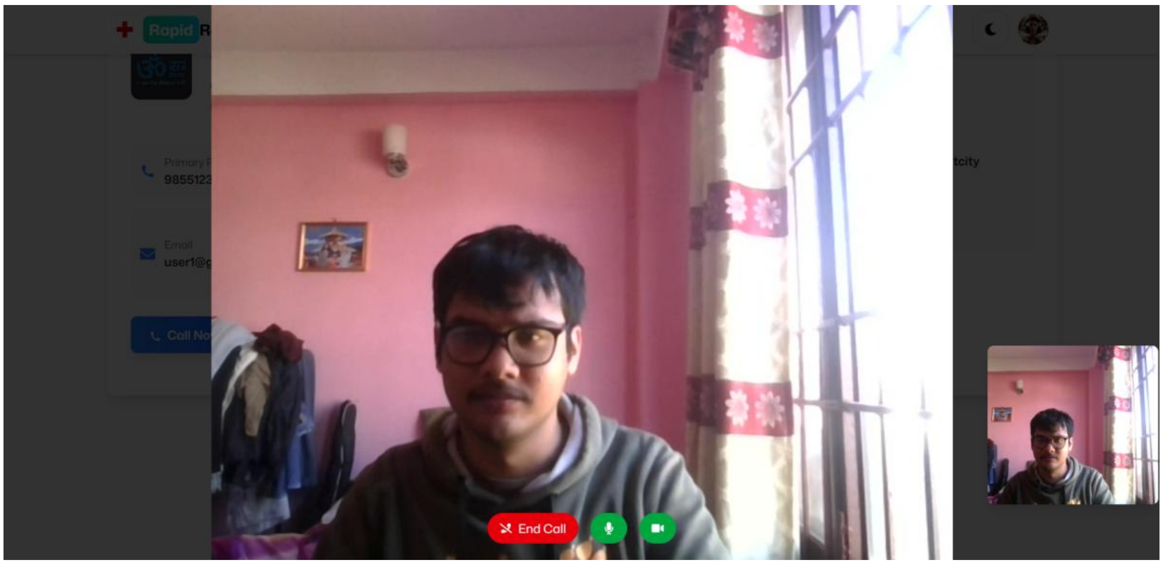


Figure 4.4: Video Call using Web RTC

Output 5 (Contact us)

This page allows users to contact the admin by using EmailJs. The message will be directed to the admin's Mail.

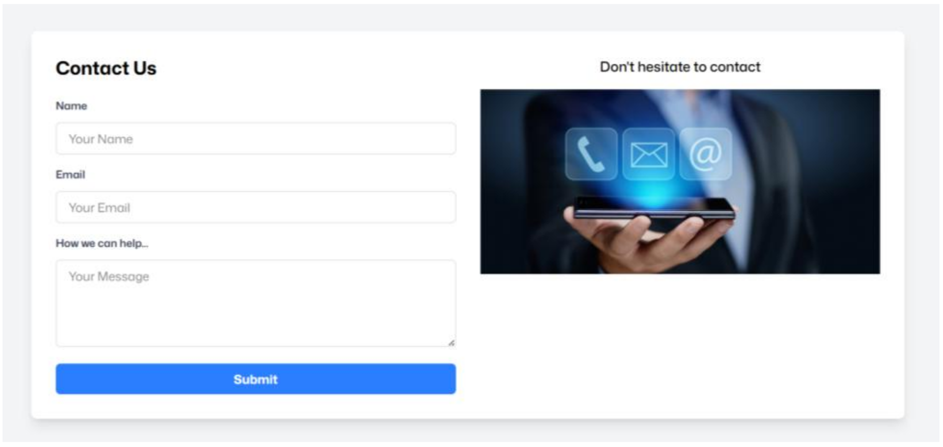


Figure 4.5: Contact us using EmailJs

Chapter 5 Conclusion and Future Recommendations

5.1 Conclusion

The Rapid Reach system provides an efficient and reliable platform for locating and contacting emergency services in real time. By integrating geolocation, priority scoring, online availability detection, and video call functionality, the system ensures that users can quickly identify and directly communicate with the most suitable and nearby services during critical situations. Algorithms such as the Haversine Distance for proximity calculation and the Priority Scoring Algorithm for ranking services enhance decision-making, reduce response time, and improve accessibility. The filtering, sorting, and real-time update features further make the system user-friendly and adaptable to various emergency scenarios. Overall, Rapid Reach demonstrates how technology can improve both the accessibility and immediacy of emergency response.

5.2 Outcome

The main outcome of the Rapid Reach project is a functional, responsive system that provides users with the most relevant emergency services based on multiple factors such as availability, proximity, category priority, verified status, and user ratings. Users can search, filter, and sort services efficiently, and the system highlights the top services through priority scoring. Additionally, the video call feature allows direct communication with service providers, enabling faster coordination during emergencies. By combining real-time updates, priority ranking, and instant contact options, Rapid Reach effectively reduces the time and effort required to access emergency assistance, enhancing both safety and reliability for end users.

5.3 Future Recommendations

The project is completed as per the date but there are some portions from which the project can be upgraded in near future. Some of the adjustments are:

- Develop a mobile app for real-time tracking of emergency vehicles, notifications, and one-click service requests.
- Adding a feature that notify and educate the public about ongoing emergencies, traffic disruptions, or safety hazards.

- Introduce a ratings and reviews system so users can provide feedback on emergency services, enhancing quality and accountability.

References

[1] P. Aacharya, “Nepal Ambulance Service: Saving lives along the way,” Nepal Ambulance Service, 2022. [Online]. Available: https://nepalambulanceservice.org/.

[2] A. Timilsina, “Good Neighbours Nepal: Ashal Chhimeki Nepal-ACN,” Good Neighbours Nepal, 2016. [Online]. Available: https://goodneighbours.org.np/.

[3] B. R. Pant, “Hospital for Advanced Medicine & Surgery : A multi-disciplinary tertiary care boutique hospital,” Hospital for Advanced Medicine & Surgery, 2024. [Online]. Available: https://hamshospital.com/.

[4] S. B. Silwal, “Alka Hospital Pvt. Ltd: We care to cure,” Alka Hospital Pvt. Ltd, 2023. [Online]. Available: https://www.alkahospital.com/.

[5] D. Thapa, “Nepal Police: Truth, Service and Security,” Nepal Police, 2022. [Online]. Available: https://www.nepalpolice.gov.np/.

[6] S. Chettri, “Hamro Life Bank:Vein to Vein,” Hamro Life Bank, 2019. [Online]. Available: https://hamrolifebank.com/.

[7] J. Fitzpatrick, “Fire-Aid: We send in specialist teams to support communities,” Fire-Aid, 2019. [Online]. Available: https://www.fire-aid.org/.

[8] R. Maharjan, “Nepal APF Hospital: Armed Police Force,” Nepal APF Hospital, 2021. [Online]. Available: https://hospital.apf.gov.np/.

[9] C. Pinson, “Sketchfab: The leading platform for 3D & AR on the web,” Sketchfab, 2019. [Online]. Available: https://sketchfab.com/.

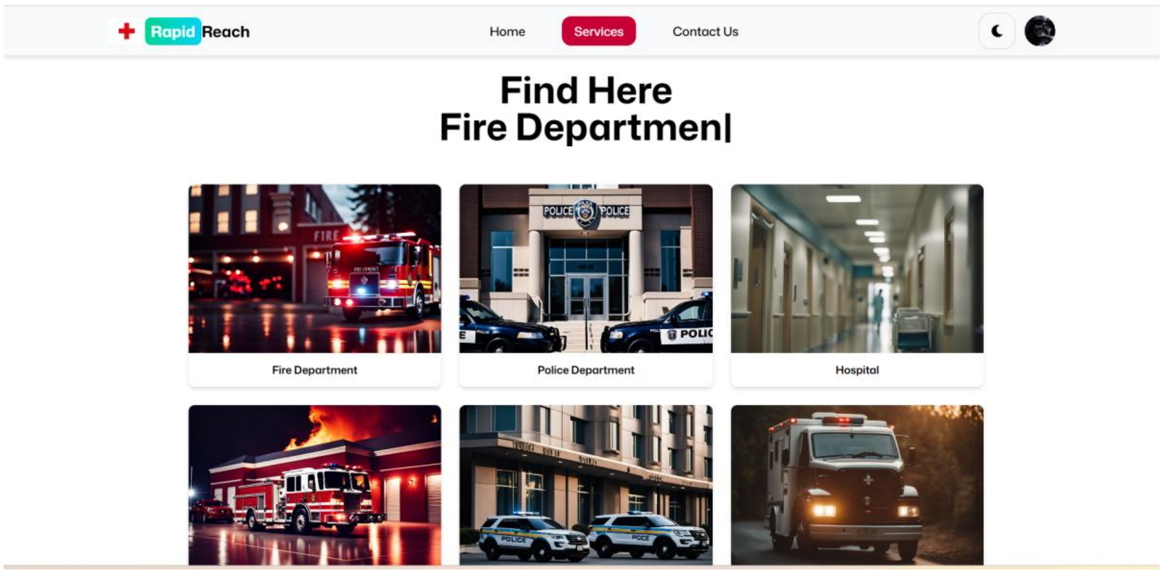
[10] R. Cabello, “Threejs,” Three.js: JavaScript 3D Library, 2022. [Online]. Available: https://threejs.org/.

[11] T. Dohmke, “Github: The complete developer platform to build, scale, and deliver secure software.,” Github, 2021. [Online]. Available: https://github.com/.

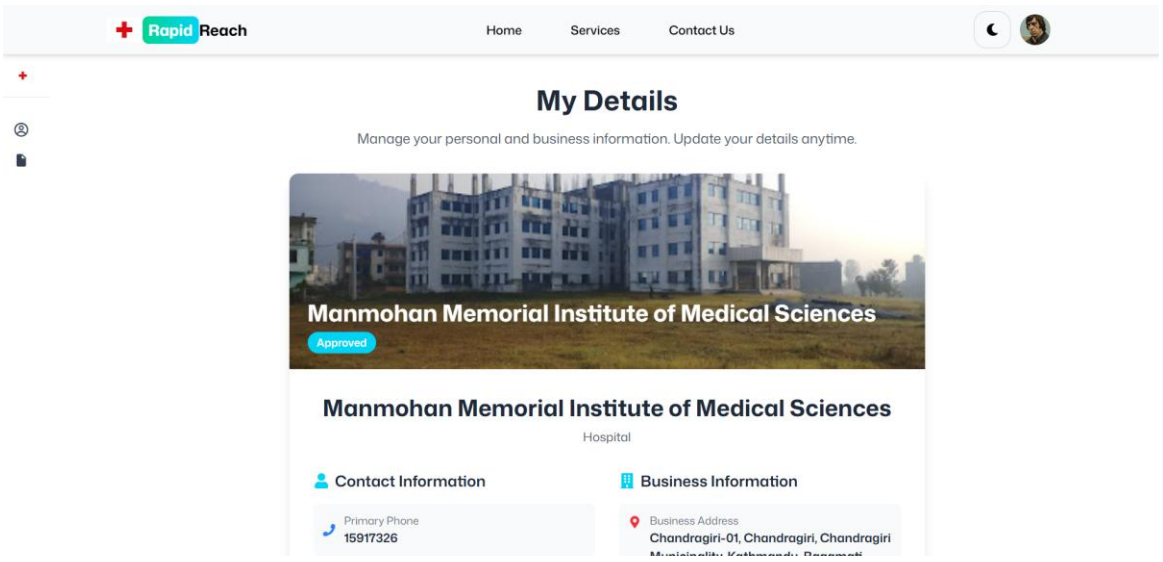
[12] E. Catmull, “Render: Your fastest path to production,” Render, 2023. [Online]. Available: https://render.com/.

Appendices

Category Page



Document Page



My Details

Manage your personal and business information. Update your details anytime.

Manmohan Memorial Institute of Medical Sciences

Manmohan Memorial Institute of Medical Sciences

Hospital

Contact Information

Primary Phone
15917326

Business Information

Business Address
Chandragiri-01, Chandragiri, Chandragiri

Source Code

```
import { io } from "socket.io-client";
let socket;
const getSocket = () => {
  if (!socket) {
    const url = import.meta.env.VITE_API_SOCKET_URL || "http://localhost:3000";
    socket = io(url, {
      withCredentials: true,
      transports: ["websocket", "polling"],
    });
    // Basic diagnostics
    socket.on('connect', () => {
      console.log('[socket] connected', socket.id);
    });
    socket.on('connect_error', (err) => {
      console.log('[socket] connect_error', err?.message);
    });
    socket.on('disconnect', (reason) => {
      console.log('[socket] disconnected', reason);
    });
  }
  return socket;
}

const setSocket = () => {
  socket = null;
}

export default {
  getSocket, setSocket
}
```

```
import AnimatedCounter from "../components/AnimatedCounter.jsx";
import Button from "../components/Button"
import HeroExperience from "../components/HeroModels/HeroExperience"
import { words } from "../constants/index.js"
import { useGSAP } from "@gsap/react";
import gsap from 'gsap';
const Home = () => {
  return (
    <section id="hero" className="relative overflow-hidden">
        <div className="absolute top-0 left-0 z-10">
            <img src="/images/bg.png" alt="background" />
        </div>

        <div className="hero-layout">
            {/* LEFT: HERO CONTENT */}
            <header className="flex flex-col justify-center md:w-full w-screen md:px-20 px-
5">
                <div className="flex flex-col gap-7">
                    <div className="hero-text">
                        <h1>Here
                            <span className="slide">
                               <span className="wrapper">
  {words.map((word) => (
    <span
      key={word.text}
      className="flex items-center md:gap-3 gap-1 pb-2"
    >
      <word.icon
        className="xl:size-12 md:size-10 size-7 md:p-2 p-1 rounded-full bg-white-50 text-
blue-600"
      />
      <span>{word.text}</span>
    </span>
```

```
 ))}
</span>
                    </span>
                  </h1>
                  <h1>For real time service</h1>
                  <h1>Allows to contact</h1>
                </div>
                <p className="text-gray-600 dark:text-white-50 md:text-xl relative z-10
pointer-events-none:">
                  We are here for your health issue with the real time <br />
                  communication service.
                </p>
                <Button
                className="md:w-80 md:h-16 w-60 h-12"
                id="button"
                text="Our Services"
                />
              </div>
            </header>
            {/* RIGHT: 3D MODEL */}
            <figure>
              <div className="hero-3d-layout ">
              <HeroExperience />
              </div>
            </figure>
          </div>
          <AnimatedCounter />
   </section>
 )}
export default Home
```