Tribhuvan University

Academia International College

Final Year Project Report
On
Ambulance Management System (AMS)
[CSC 412]

Under the supervision of

"Mr. Ananda Adhikari"

Submitted by

Arjun Raut (T.U. Exam Roll No. 26479/077)

Manish Kr Mandal (T.U. Exam Roll No. 26497/077)

Rohan Maharjan (T.U. Exam Roll No. 26507/077)

Submitted to

Department of Computer Science and Information Technology

Academia International College

Institute of Science and Technology

Tribhuvan University

January, 2025

Tribhuvan University

Academia International College

Final Year Project Report

On

Ambulance Management System (AMS)

[CSC 412]

A final year project submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Computer Science and Information Technology awarded by Tribhuvan University

Submitted by

Arjun Raut (T.U. Exam Roll No. 26479/077)

Manish Kr Mandal (T.U. Exam Roll No. 26497/077)

Rohan Maharjan (T.U. Exam Roll No. 26507/077)

Submitted to

Department of Computer Science and Information Technology Academia International College

Institute of Science and Technology

Tribhuvan University

January, 2025

Tribhuvan University

Institute of Science and Technology

Academia International College

Department of Computer Science and Information Technology

Email: mail@academiacollege.edu.np

Supervisor's Recommendation

I hereby recommend that the project work report prepared under my supervision by Mr. Arjun Raut (26479/077), Mr. Manish Kr Mandal (26497/077), and Mr. Rohan Maharjan (26507/077) entitled "Ambulance Management System (AmbulanceNow)" be accepted as fulfilling in partial requirements for the degree of Bachelors of Science in Computer Science and Information Technology. In my best knowledge, this is an original work in Computer Science and Information Technology.

.....

Mr. Ananda Adhikari

Project Supervisor

Department of Computer Science and Information Technology

Academia International College

Gwarko, Lalitpur

Tribhuvan University

Department of Computer Science and Information Technology

Academia International College

Certificate of Approval

This is to certify that this project prepared by Mr. Arjun Raut, Mr. Manish Kr Mandal and Mr. Rohan Maharjan entitled "Ambulance Management System (AmbulanceNow)" in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Information Technology has been well studied. In our opinion, it is satisfactory in the scope and quality as a project for the required degree.

…….. Mr. Ananda Adhikari Project Supervisor Department of Computer Science and IT Academia International College	…….. Mr. Bishwas Mathema HOD/Program Coordinator Department of Computer Science and IT Academia International College
…….. Internal Examiner Academia International College	…….. External Examiner Central Department of CSIT Tribhuvan University

Acknowledgement

The support, guidance and assistance that we received from various individuals and organizations were instrumental in successfully completing this project. We would like to extend our profound gratitude to each and every one of them.

We are highly obliged to **Academia International College** for their regular guidance, supervision and provision of the essential supporting ICT infrastructure and environment required to enable this project to be conducted. We also express our gratitude to the B.Sc. CSIT Coordinator **Mr. Santosh Rijal** whose continuous support was crucial in making this project possible.

We are extremely grateful to our project supervisor **Mr. Ananda Adhikari** for his guidance, feedback, and professional expertise during this project. His contribution was key to create a working/functional webapp. We also pay our grateful thanks to **Mr. Bishwas Mathema**, for his continued encouragement and support in making sure this report followed the university academic criteria and conventions.

We are thankful and fortunate enough to get constant support from our seniors and every teaching staff of B.Sc. CSIT department which helped us successfully complete our project. We would also like to extend our regards to all the non-teaching staff of B.Sc. CSIT department for their timely support.

Last but not the least, our thanks and appreciations also go to each and every one of our friends for their encouragement and support in developing the project.

Abstract

Information and Communication Technology (ICT) is revolutionizing lives around the world, with emerging innovations perpetually evolving to improve our tactics for addressing common dilemmas. Few clicks are needed to replace human interaction, filling gaps with technological tools and solutions. The Ambulance Management System is an online application to enable the typical or normal person to have a fast, accurate, and smooth switch for the ambulance.

The Ambulance Management System is a web application whose main objective is getting ambulance services during emergencies in a short time. This system connects the general public, ambulance drivers and hospitals together to make sure that user can book an ambulance with a matter of few clicks. Not only does the web app gives the quickest route for the drivers to reach and pick up a patient, it also allows real time tracking of an ambulance such that anyone who calls an ambulance knows precisely where their ride is. The system, along with the nearest hospitals and best available routes displayed, guarantees that the patient is taken to the right place at minimum time.

The Ambulance Management System takes benefit of technology to change emergency medical service delivery into routine and quicker access. It provides quick response and optimized routes, connecting the public to ambulance drivers and then to hospitals. Designed to expedite response-associated delays, conserve lives and improve health delivery — the system lives its motto: "Faster Response, Safer Journeys, Better Care."

Table of Contents

List of Figures

List of Tables

List of Abbreviation

DFD	Data Flow Diagram
EMS	Emergency Medical Service
UI	User Interface
AMS	Ambulance Management System
MySQL	My Structured Query Language
CSS	Cascading Style Sheets

Chapter 1: INTRODUCTION

1.1. Introduction

The Ambulance Management System is a webapp created to deliver effective emergency medical aid to individuals requiring urgent attention. In a nation such, as Nepal where receiving help during emergencies can pose a significant hurdle this platform strives to connect the public with hospitals and ambulance services seamlessly. By offering a user interface for people to swiftly call for ambulances it facilitates responses, during urgent situations. In addition, to that feature of the system is giving ambulance drivers the best routes, in time to quickly and effectively reach patients in need.

By using functions, like tracking ambulances in time and identifying the hospital while optimizing routes efficiently, the Ambulance Management System guarantees a quick connection between patients and medical professionals. With the help of technology and communication tools this system facilitates communication, between ambulance teams and hospitals thereby enhancing the coordination of emergency responses overall.

This initiative is focused on reducing wait times, for emergency assistance, a factor that can make a difference between life and death situations. It seeks to tackle the issues encountered by individuals and healthcare professionals by enhancing the availability, clarity and promptness of emergency services. The Ambulance Management System corresponds with the increasing demand for successful healthcare provision in Nepal, in urban and remote regions where immediate medical assistance may be scarce. In the end, goal of the system is to save lives and boost healthcare results by improving the speed and effectiveness of emergency services.

1.2. Problem Statement

The situation with ambulance services in Nepal continues to be in a critical state as it is expounded by several problems that affect the delivery of timely and efficient emergency medical services. These hardships are more apparent where ambulances are hard to come by, mainly in the rural and remote areas where infrastructures are poorly developed. In urban areas traffic jam and poor traffic management only make the situation worst meaning that a delay situation can actually become fatal. Due to inadequate national standards of consolidating and organizing the ambulance services, such issues arise and lead to misuse of resources, poor interface between organizations and unequal service delivery. As all these problems persist, the outcomes of emergency circumstances in Nepal stay bleak, on top of threatening the lives of patients who require attentive care.

Challenges in Accessing Ambulance Services in Nepal

- Lack of Centralized Coordination
- Poor Communication Infrastructure
- Traffic Congestion in Urban Areas
- Absence of Real-Time Tracking
- Limited Public Awareness

1.3. Objectives

- To identify the ambulance locations on the road network in real-time using GPS coordinates.
- To display the hospitals within the radius
- To locate the available ambulances
- To find the fastest routes from all the accidents to reach the hospital immediately.

1.4. Scope and Limitation

1.4.1. Scopes

- Quick Access to Emergency Care

  The Ambulance Management System is designed to enhance the speed and efficiency of emergency medical services in Nepal. By enabling users to request ambulances with just a few clicks, the system ensures quick access to emergency care, especially in critical situations.
- Route Optimization

The system also optimizes routes for ambulance drivers, calculating the fastest and most efficient paths to the patient's location. This helps reduce response times and ensures that patients receive timely medical attention. Additionally, the application displays nearby hospitals, assisting ambulance drivers in quickly navigating to the most appropriate healthcare facility for the patient.

- Widespread Access

  Initially, the system will focus on urban areas with well-developed infrastructure, and as internet connectivity and road access improve, it will expand to rural and remote regions. The ultimate goal of the Ambulance Management System is to provide reliable, accessible, and efficient ambulance services across Nepal, contributing to faster emergency responses and better healthcare outcomes for all citizens.

1.4.2. Limitations

The Ambulance Management System provides a good and workable solution to the problem but there are limitations also:

- Dependency on Internet Connectivity – The system will require both internet and a stable connection for real-time monitoring & communication. This may become difficult in remote or rural context where the network coverage is less.
- Ambulance Availability: Due to limited resources, there may not be enough ambulances available in some areas and this can impede the success of the system.
- Road Infrastructure: Sufficient road infrastructure will play the major role for routes optimization by the system; however, it may differ with provinces of Nepal, especially towards rural or hill regions.
- User Awareness: Getting everyone in the country to follow and use this system isn't going to be an easy road, especially in remote areas where people don't know much about everything related to technology.

1.5. Development Methodology

The Ambulance Management System follows the Agile Software Development Methodology because of its flexibility and iterative approach that allows for adaptation, to changes, throughout the projects lifecycle to meet evolving needs and feedback requirements effectively in a manner by delivering functional increments at each project stage.

Phases of Development

1. Requirement Gathering and Analysis

Discussions and meetings were held to gather information about the project requirements like information about hospitals, ambulance services, and the general public. Key challenges in the existing emergency response system were identified, and essential functionalities such as monitoring of ambulance locations and optimizing routes for dispatch were pinpointed. The focus was on addressing user needs for both ambulance services and hospital facility.

2. Design

After discussing all requirements, the system's architecture was defined, and a modular structure was established. It included both top-level architecture and detailed database designs. The user interface (UI) of the web application was designed considering easy handling by ambulance drivers and general users. Cultured database design facilitated unobtrusive information transfer across the different components of the system.

3. Implementation

The implementation phase involved developing features iteratively. Primary focus was placed on building the ambulance request feature. Iterative development allowed for prompt testing and feedback, ensuring that each feature functioned correctly. The Dijkstra algorithm for route optimization was integrated, along with geo-hashing, to enhance ambulance dispatch accuracy and reduce response times.

4. Testing:

Different testing like unit testing, integration testing, and system testing were carried out to ensure the system functioned as expected. The route optimization algorithm was rigorously tested to verify its effectiveness in providing the fastest routes for ambulances. This phase

helped identify and resolve any issues, ensuring that the features performed correctly within the larger system.

5. Deployment and Feedback:

The completion of feature implementation paved the way for system deployment for real-world use by ambulance drivers, hospital management, and the general public. Feedback was collected from users on areas needing improvement. The collected feedback was then utilized to perform iteration on further improvements and modifications to the existing system, so as to ensure that system meets the users' needs in the next development cycles.

6. Maintenance and Updates:

After deploying the system was periodically undergone updates and maintenance to include new functionalities and enhancement in its performance. New features were added, system optimizations were implemented, and any issues reported by users were addressed.

The basic Principles of Agile Methodology

- Customer Collaboration Over Contract Negotiation:
- Focuses on continuous collaboration with users to ensure the system meets their evolving needs.
- Responding to Change Over Following a Plan:
- Allows flexibility for changes throughout the project based on real-time feedback.
- Delivering Working Software Frequently:
- Delivers small, functional parts of the system regularly to get early user feedback.
- Simplicity:
- Focuses on implementing essential features to deliver value, avoiding unnecessary complexity.
- Self-Organizing Teams:
- Encourages team autonomy for decision-making, promoting collaboration and efficiency.
- Continuous Improvement:
  Promotes regular reflection and improvement after each iteration to enhance the system's performance.

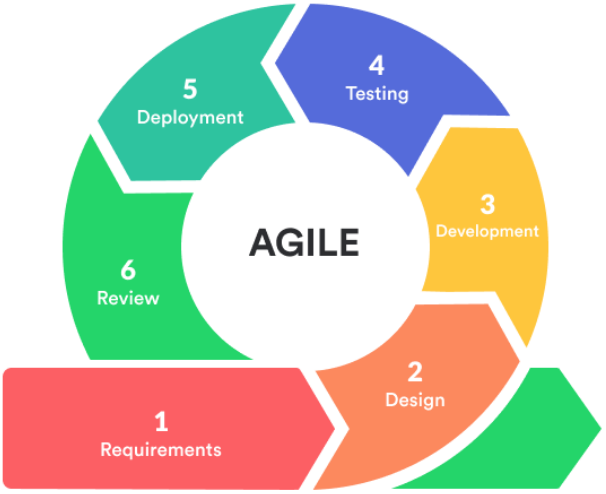


Figure 1 1.1: Agile Development Methodology

1.6. Report Organization

After successfully completing the project, the documentation of the project has been prepared. The starting section of the report consists a Title Page, Certificate Page, Acknowledgement, Abstract, Table of Contents, and lists of Abbreviations, Figures, and Tables.

The main report is organized into 6 chapters, aligning with their respective headings and content, which includes the following:

Chapter 1: Introduction

The first chapter provides a brief overview of the project by sketching aspects such as project introduction,problem statement, objectives, scope and limitations, and methodology.

Chapter 2: Background Study and Literature Review

It talks about the background study of the project and reviews existing literature, summarizing relevant projects, papers, and articles.

Chapter 3: System Analysis

It centers on system analysis, covering requirements and feasibility analysis. Functional requirements of the system are defined through a use case diagram. A Gantt chart is also used to clarify the time taken for various tasks in the project.

Chapter 4: System Design

It delves into system details, focusing on the implementation process and designing the model architecture, database, interface, and forms. Also, it provides insights into the algorithms used in the system.

Chapter 5: Implementation and Testing

It explains the implementation and testing details, including an overview of the tools and dependencies utilized for implementing the system.

Chapter 6: Conclusion and Recommendations

It sums up and concludes the project and explores possibilities for future enhancementsof the project.

The report’s main section contains References following IEEE standards and Appendices containing system screenshots and essential source code snippets.

Chapter 2: BACKGROUND STUDY AND LITERATURE REVIEW

2.1. Background Study

In Nepal, timely land transportation access to emergency medical services continues to be a distant reality, primarily for remote and rural communities. And the life-threatening delays of medical assistance can be attributed to Nepal's poor road infrastructure, deficient ambulance services and most importantly lack of coordination between hospitals and ambulances. In addition, the absence of live information and route optimizing makes it worse for emergency response times.

However, with the promise of rapid digitalization in Nepal, there is an opening to bridge these gaps in the healthcare system. With more than half the population using a mobile phone while almost the same fraction of people having internet connectivity, even in rural areas; there can be a space created for technology-driven solutions which can enhance emergency medical services as well. So, by developing this Ambulance Management System, we try to fill these missing links where the general public can directly communicate with the ambulance drivers and hospitals through an easy-to-use web application during any emergency.

The integrated Ambulance Management System aims to enhance emergency response by utilizing real-time tracking, optimized routing, and efficient communication to reduce the time it takes for ambulances to reach their intended destination, enabling medical treatment as soon as possible and saving valuable lives. In a country like Nepal, where quick access to healthcare services can make the difference between life and death. Thus, this is potentially set to revolutionize emergency medical services in underserved, difficult to access places within the nation.

2.2. Literature Review

The literature on Ambulance Management Systems reveals a continuous evolution driven by technological advancements and the need for efficient emergency response. While significant progress has been made, challenges remain in optimizing response times, ensuring system integration, and protecting data security. Studies, highlight the shift towards automated, real-time systems that integrate with emergency call centers and hospital databases to enhance response efficiency. The evolution of communication technologies has significantly improved the data transmission and connectivity of AMS.

Algorithms for dynamic prioritization play a crucial role in handling multiple emergency requests. Brown in 2020 explore various dynamic priority queue models that adjust priorities based on the severity of emergencies and resource availability. Efficient routing algorithms, Dijkstra is essential for minimizing travel time. Wilson in 2021 review how real- time traffic data is integrated into these algorithms to adapt routes dynamically, thereby reducing delays.

2.3. Study of Relevant Projects

Inefficient system has often been hampering Emergency Medical Services (EMS) in Nepal, especially concerning ambulance dispatch and coordination. Conventional ambulance services focus more on manual operations, while the users have to call for requesting ambulances. It introduces lags because of communication problem, real-time updates unavailability and route optimization absence. Ambulance drivers do not necessarily get the routes convenient for them to take, and without a centralized platform in place, hospitals will function separately with ambulance services, making response times longer and logistics slower.

While government and private ambulance service providers are available, they operate with various levels of coverage and technological constraints. Most ambulance services operated by government paramedics are located in cities with none to little existing infrastructure in rural areas. Though much easier to access, private services do not offer real-time tracking, optimization of routes or coordination with hospitals

Many private ambulance services have picked up some basic mobile applications like GPS to catch the location, but such solutions are usually not end-to-end or convenient. Current systems do not offer optimal routing or real-time communication between all key

stakeholders, including users, drivers and hospitals. Moreover, there is no system which informs user regarding nearest hospital or closest available ambulance. The limited penetration of these tests in rural areas, compounded with the inefficiencies euphemized, cause delays that can compromise patient care outcomes.

Hence, The Ambulance Management System aims to address these challenges by offering an integrated, technology-driven platform that ensures faster response times, optimized routes, and better coordination between ambulance services and healthcare providers.

Chapter 3: SYSTEM ANALYSIS

3.1. System Analysis

System analysis evaluates a system to comprehend its parts, operations and goals. When it comes to "Ambulance Management System", system analysis includes examining the existing emergency response procedures spotting areas of inefficiency and establishing what enhancements are needed. This step centers on grasping the demands of users, like patients, ambulance providers and medical facilities while guaranteeing that the system adequately caters to these needs.

3.1.1. Requirement Analysis

i. Functional Requirements

a. User Registration & Login

The system has an account registration feature that allows users (user and driver) to create a new separate account by providing unique username and password. Users or Drivers who are already registered would be able to log in with their username/password securely. It also allows new users to register, enter their name, contact details and location.

b. Request and Dispatch for Ambulance

This system makes it as easy for a user to request an ambulance with the least amount of information such as type of emergency, current location, and details about patient. As soon as the request is made, the system find out which facility, where ambulance supply should go, closest to this patient location and then it will send that ambulance.

c. Ambulance Driver Route Optimization

The system includes route optimization algorithms to help ambulance drivers find the fastest and most efficient route possible, while considering traffic conditions, road closures, and distance from the patient's location. This allows faster feedback response and better resource management.

d. Direction and Information of Hospital

The system shows a set of hospitals around the patient in relation to the geographic space where he is. It also points to the nearest appropriate hospital based on the patient's condition as well as hospital availability

e. Ambulance Availability

The system allows ambulance operators to put a mark as to whether their vehicle is available or not and it also provide drivers with the ability to change their status in the event of delays, breakdowns or other circumstances that impact service.

f. User Profile Management

When a user clicks on the sign-out, system ask them to confirm that intent and permission because it can be accidental too. It could be displayed as 'Are you sure that you want to sign out?' which will help prevent accidental logouts.

g. Sign-Out

The system provides user with a way to sign out of the system securely thus protecting their personal and medical information when they are done with the system. Sign out properly kills the user session so they cannot access in between the action.

ii. Non-Functional Requirements

a. User Friendly

The system has user-friendly and easy to navigate User Interface (UI) even for people with limited technical knowledge. The User Interface is intuitive, with minimal clicks required to make an ambulance request or view real-time tracking. The system is designed for accessibility, supporting users with disabilities.

b. Performance

The performance of the Ambulance Management System is not affected with the increase in number of concurrent users. The System serves the user requests in a timely manner like time taken to respond to ambulance request will be less than 5 sec. The system also facilitates route optimization while producing appropriate route suggestions even under peak loads.

c. Reliability

The system is highly reliable and capable of operating continuously, with minimal downtime. The emergency web app is available all the time for users to access ambulance services, no matter how serious the situation is

d. Scalability

The system is scalable as the number of users increases and also when more ambulances and hospitals are included into the system. It is able to accommodate the growing number of requests and users without compromising service quality. It also manages an ever-growing volume of requests for ambulance dispatch.

e. Security

Sensitive data, such as user information, driver and ambulance information along with hospital name are secured by the Ambulance Management System. The system utilizes encryption protocols and use strong authentication mechanisms to prevent unauthorized

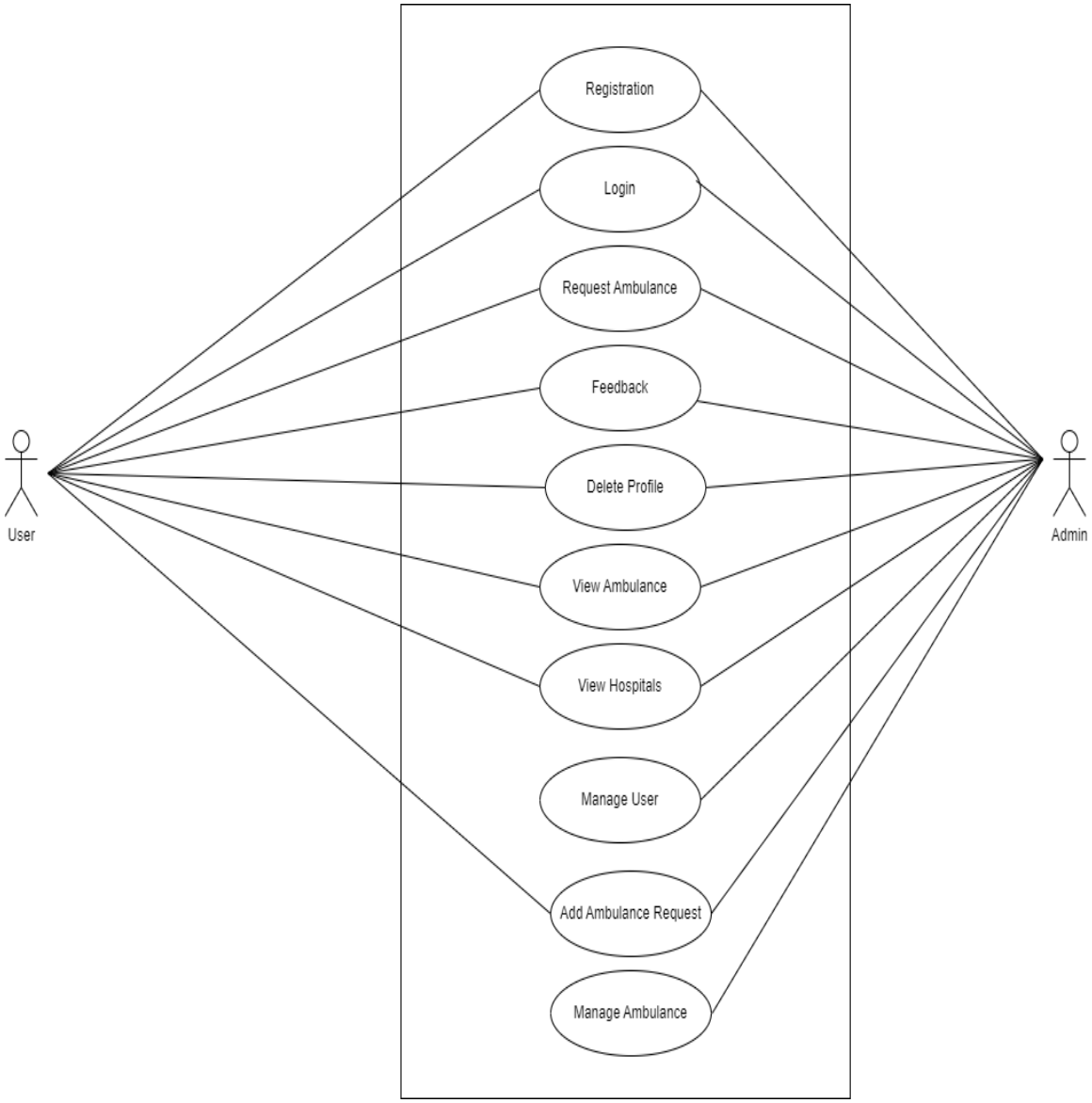


Figure 2 3.1: Use Case Diagram of AMS

3.1.2. Feasibility Analysis

Feasibility analysis helps determine whether the project is viable and if it can be executed successfully within the given constraints of time, resources, and budget. The feasibility study also evaluated the viability of the suggested system from multiple angles: technical, operational, and economic. It is useful to help determine the potential challenges and risks ensuring that the project can be delivered successfully while fulfilling user needs and expectations. This analysis also makes sure that the system can be supported indefinitely and continues to benefit the target user group over time.

i. Technical Feasibility

The technical feasibility of the Ambulance Management System is high since it includes universal technologies like GPS, web app, and storage system. Route optimization, and user registration are all achievable with current technologies, including Open Street Map API and Leaflet.js for location services, as well as tracking. The major challenge lies in ensuring the system performs reliably, especially in remote areas with limited internet connectivity

ii. Operational Feasibility

The Ambulance Management System is operationally feasible as it aligns exactly with what is actually required by EMS service, hospitals and users. The system provides a simple and easy way to request an ambulance, tracking of the dispatched ambulance, details about nearby hospitals, which will be easily accepted by end-users. It will also require training ambulance drivers and user to use the system. For successful implementation, it would require co-operation between ambulance services and hospitals while the system must be scalable to accommodate spikes in demand, such as during a natural disaster or major incident.

iii. Economic Feasibility

From an economic standpoint, the Ambulance Management System offers significant benefits. While the initial development and deployment may incur costs related to tracking system, and maintenance, the long-term operational costs are relatively low. The system could also help save costs for ambulance services by utilizing resources more effectively and minimizing operational inefficiencies.

iv. Schedule

Task	Start Date	End Date	duration
Study and Analysis	2-Aug	19-Aug	17
Data Collection	25-Aug	13-Sep	19
Implementation	13-Sep	25-Nov	73
Testing	6-Nov	29-Nov	23
Documentation	13-Aug	14-Jan	154
Review	4-Dec	23-Dec	19
Presentation	4-Jan	14-Jan	10

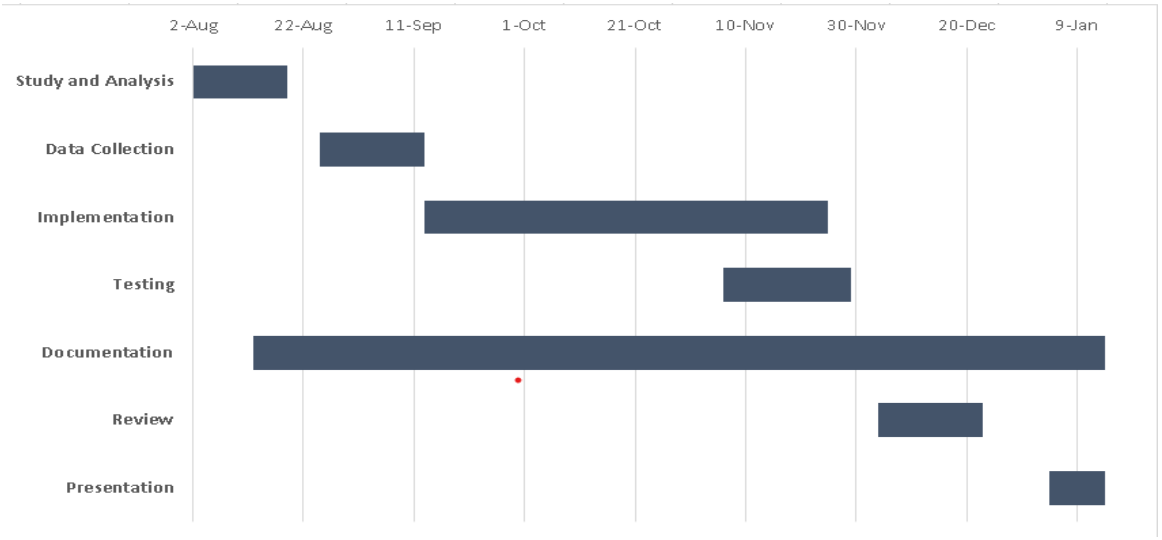


Figure 3 3.2: Gantt Chart

3.1.3. Data Modelling using ER Diagrams

The following Entity-Relationship (ER) diagram visually represent the relationships between entities or concepts within the system, providing a detailed conceptual view of a database.

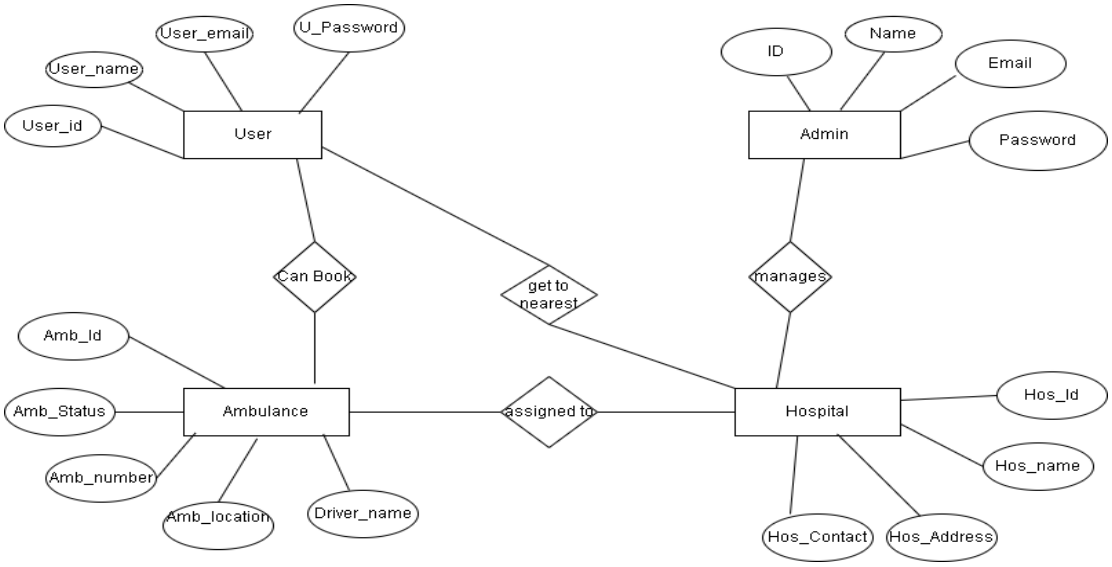


Figure 4 3.3: ER Diagram of AMS

3.1.4. Process Modelling Using DFD

The below Data Flow Diagram (DFD) visually represent the flow of processes used to capture, manipulate, store, and distribute data between the system and between its components.

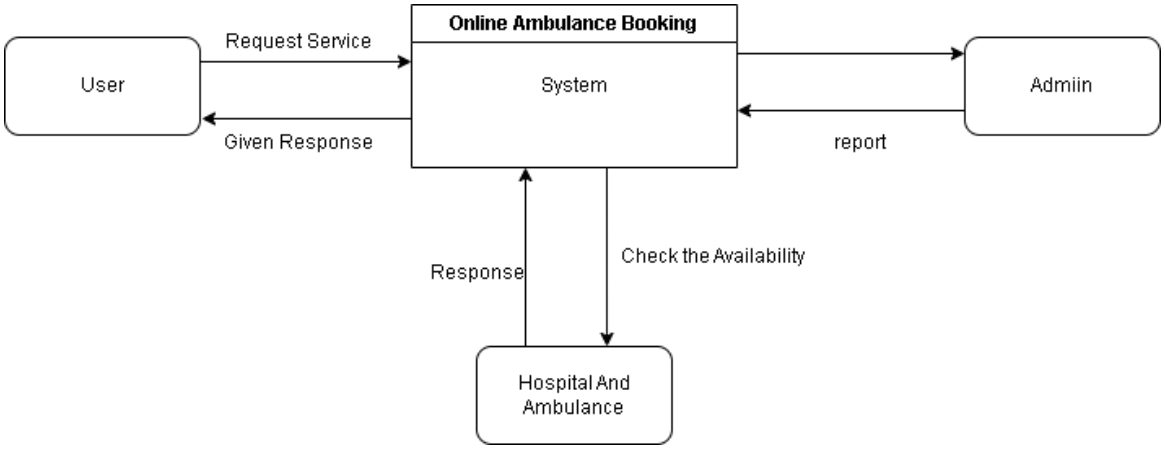


Figure 5 3.4: Level-0 DFD (context Diagram) of AMS

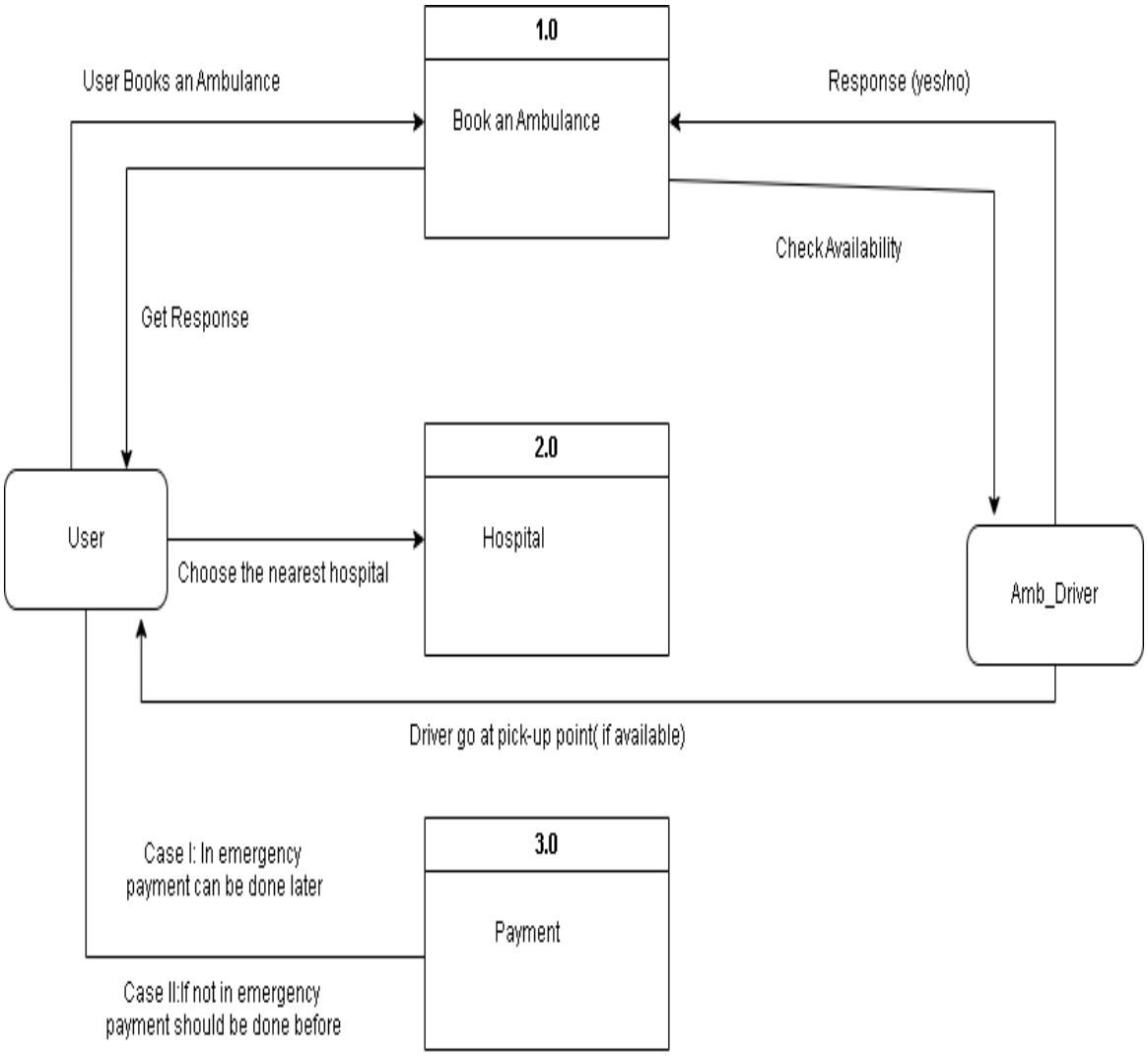


Figure 6 3.5: Level-1 DFD (Context Diagram) of AMS

Chapter 4: SYSTEM DESIGN

4.1. Design

The development of the AMS focused on defining the system's components, including its structure, sections, and interactions, along with the flow of information between them. The theoretical framework, or system architecture, outlined the system's organization and functionality. After analyzing the requirements, the system design was carefully detailed through a database schema to represent the overall design.

4.1.1. Database Design

The success of the AMS heavily relies on its designed database, which is needed to store user information, ambulance requests, and other related data. MySQL is employed as a relational database management system.

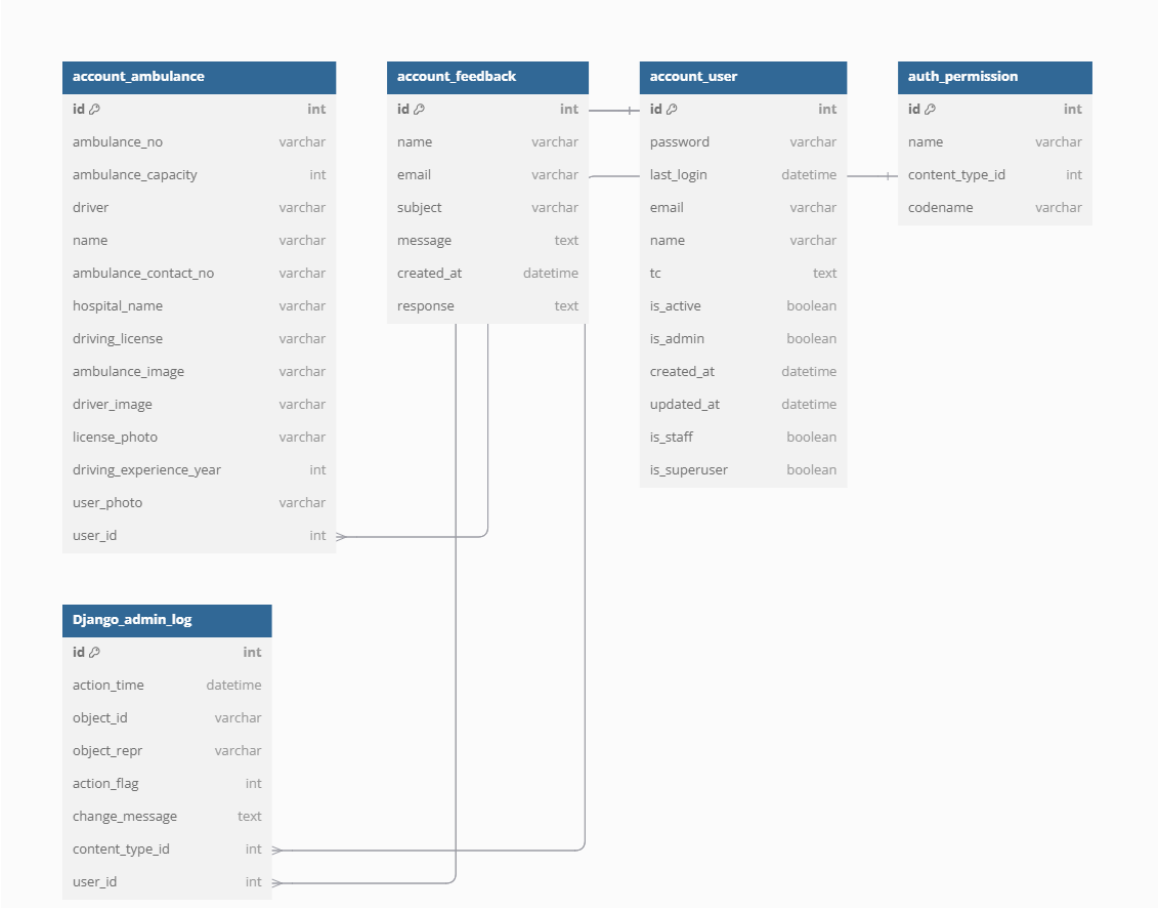


Figure 7 4.1: Database Schema of AMS

4.1.2. Forms and Interface Design

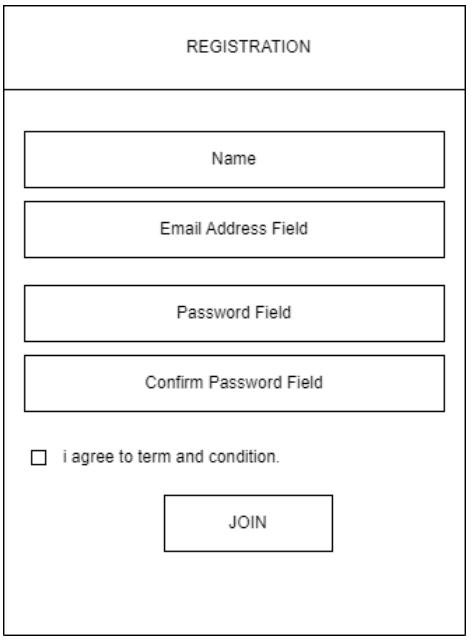


Figure 8 4.2: Registration Form

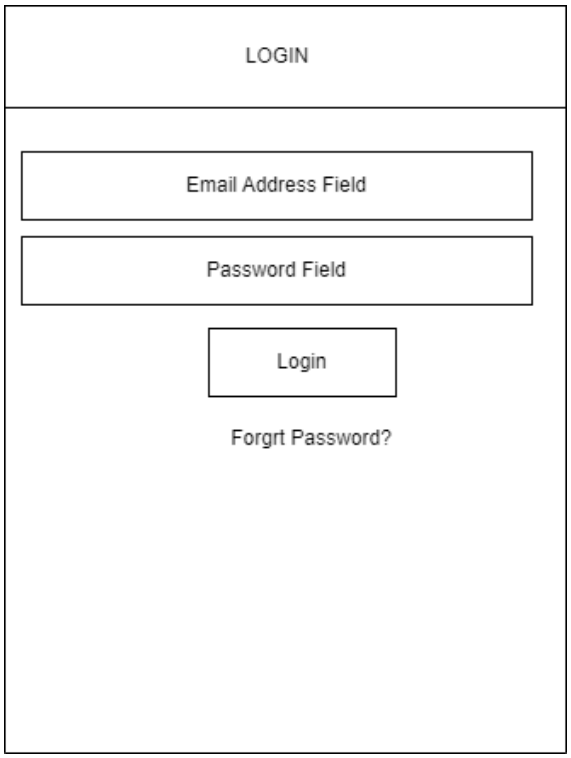


Figure 9 4.3: Login

Figure 10 4.4: Dashboard

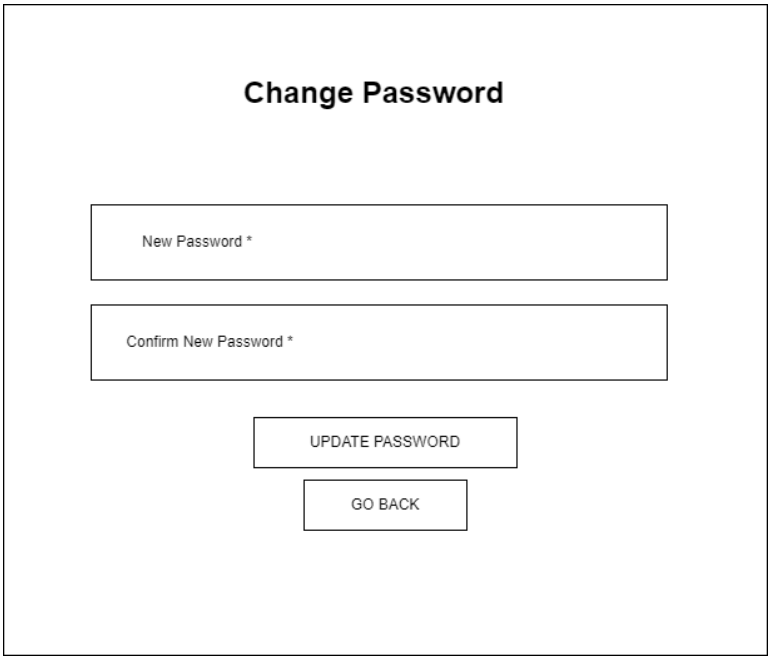


Figure 11 4.5: Change Password

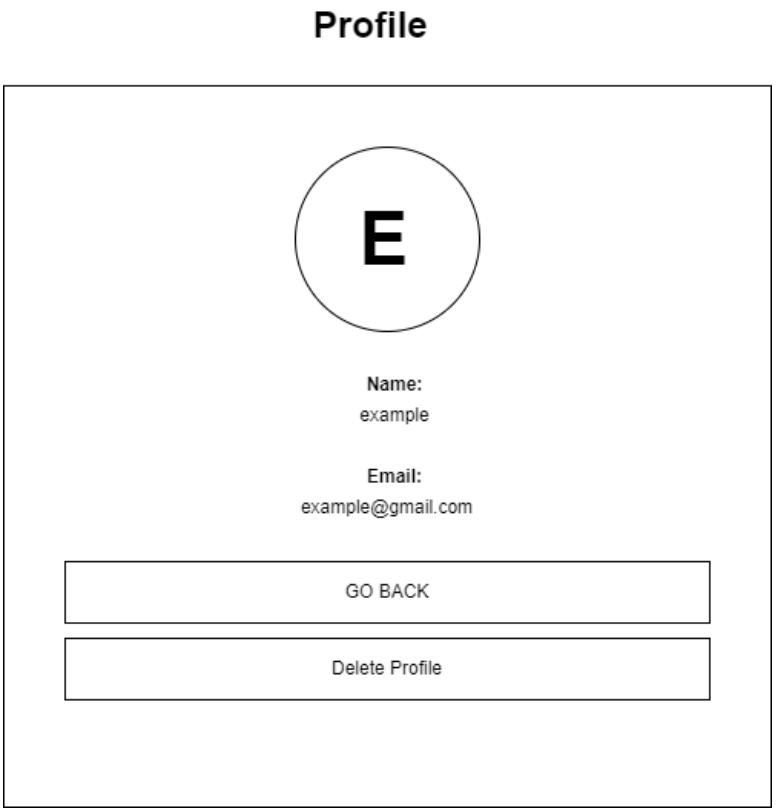


Figure 12 4.6: Delete Profile

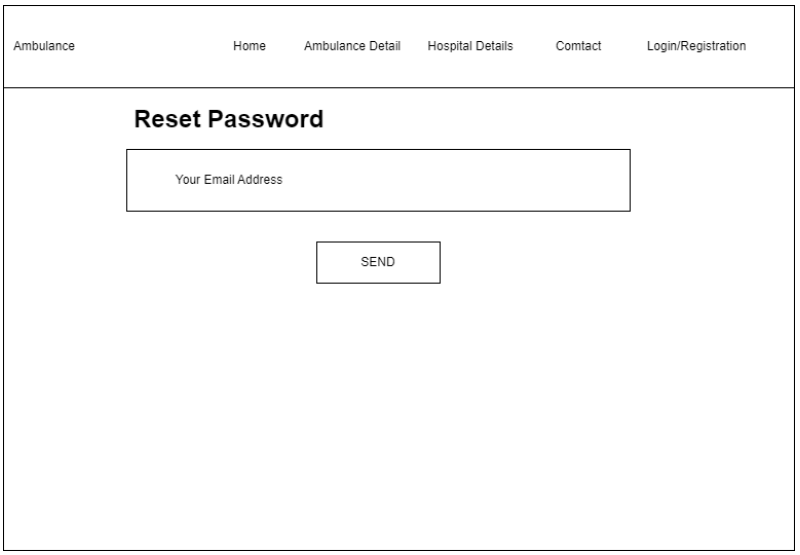


Figure 13 4.7: Reset Password

4.2. Algorithm Details

4.2.1. Dijkstra Algorithm

Dijkstra's algorithm is a popular graph search algorithm used to find the shortest path from a source node to all other nodes in a weighted graph. The algorithm works by progressively "relaxing" the edges of the graph, updating the shortest known distances to each node, and selecting the closest unvisited node until the algorithm has processed all nodes. [1]

Steps of Dijkstra’s Algorithm

1. Initialization: Set the distance of the source node to 0, and the distances of all other nodes to infinity. Mark all nodes as unvisited.
2. Selection: Select the unvisited node with the smallest tentative distance as the current node.
3. Relaxation: Update the distances to each neighboring node of the current node if a shorter path is found.
4. Mark as Visited: Mark the current node as visited and move to the next unvisited node with the smallest tentative distance.
5. Repeat: Continue until all nodes have been visited, or if the smallest tentative distance among unvisited nodes is infinity (indicating no reachable nodes are left).

Formula

For each edge (u,v) between nodes u and v is updated if the distance through u is shorter

$$d(v)=\min(d(v),d(u) + w(u,v))$$

where:

d(v) is the current known shortest distance to node v

d(u) is the current known shortest distance to node u

w(u,v) is the weight (or cost) of the edge between nodes u and v

4.2.2. Haversine Formula

The Haversine formula is used to calculate the great-circle distance between two points on the basis of given latitudes and longitudes. [2] The formula is as follows:

$$haversine(\theta) = sin^2(\frac{\theta}{2})$$

where, $\theta = (\frac{d}{r})$ is the central angle between the two points and *sin* is the sine function. *d* is the great-circle distance between the two points and *r* is the radius of earth

The haversine of the central angle θ is determined by the formula:

$$haversine\ (\theta) = haversine(lat_2\text{-}lat_1) + cos(lat_1).cos(lat_2).haversine(lon_2\text{-}lon_1)$$

where lat_1, lon_1, lat_2, lon_2 are the respective latitudes and longitudes of the first and second point in radians.

The central angle θ is calculated using the following formula:

$$sin\ \frac{\theta}{2} \quad \sqrt{\frac{haversine(lat2-lat1) + cos(lat1).cos(lat2).haversine(lon2-lon1)}{2}}$$

Then calculate actual great-circle distance *d* between the two points using the formula

$$d = r \bullet \theta$$

Chapter 5: IMPLEMENTTAION AND TESTING

5.1. Implementation

Implementing ‘Ambulance Now’ required multiple stages which included requirements defining, choosing a platform, planning, frontend components building, implementing backends, testing the system and quality assurance, and optimizing the system contents

5.1.1. Tools Used

- Django: Used for building robust and scalable web applications
- MySQL: A RDBMS used for storing and managing data
- CSS: A stylesheet language used to describe the presentation of a web page
- ReactJS: Used for building interactive user interfaces and dynamic web app
- VS Code: Used for writing and editing code with various programming languages
- Dbdiagram: Used for visualizing and designing database diagrams
- Draw.io: Used for creating flowcharts, wireframes, and other visual diagrams
- Microsoft Word: used for creating, editing, and formatting project documents
- MS Excel: Used to prepare Gantt Charts
- Git: Version Control
- Postman: Used for unit testing, integration testing and overall system testing

5.1.2. Implementation Detail of Modules

The project includes following modules:

- Admin Module: This module allows administrators to execute actions such as user management, ambulance management, etc. They can manage user accounts, ambulance requests, hospitals' information through the module.
- User Module: This module covers general features such as user sign up, log in, ambulance requests, and ambulance search.
- Registration Module: This function allows a user to register for an account by providing the respective registration information such as username, email, and password. It verifies the data by looking for common accounts but saves all the new details to the user account database.

- Login Module: This function caters for both user admin and user login in checking the provided username and password against available user account data. If the credentials supplied by the user qualify, then they will be accepted into the system as their own.

5.2. Testing

Testing is crucial phase in the implementation of all systems. Thus, different types of tests were performed on this system i.e. Ambulance Now to ensure its performance, usability and functionality.

5.2.1. Test Case for Unit Testing

In unit testing, different test scenarios, testable units of code, at the function or method level, were tested to ensure their correctness and proper functioning.

The following test scenarios were employed for conducting unit testing.

Table 5.1: Test Case for User Login

Test ID	Test Scenario	Test Data	Expected Result	Observed Result	Test Status
1	User Login With invalid credentials	Email: test@gmail.com Password: demo@123	Sign-in Failed; Display error message	Same as expected result	Pass
2	User Login With validinputs	Email: test@gmail.com Password: testing@123	Signed-in Successfully; User redirects to the homepage.	Same as expected result	Pass

Table 5.2: Test Cases for Admin Login

Test ID	Test Scenario	Test Data	Expected Result	Observed Result	Test Status
1	Admin Login With invalid credentials	Email: admin@gmail.com Password: test@123	Signed-in Successfully; Admin redirects to the Admin dashboard.	Same as expected result	Fail
2	Admin Login With validinputs	Email: admin@gmail.com Password: admin@123	Signed-in Successfully; Admin redirects to the Admin dashboard.	Same as expected result	Pass

Table 5.3: Test Cases for Deleting User Profile

Test ID	Test Scenario	Test Data	ExpectedResult	Observed Result	Test Status
1	Delete user profile	Navigate to dashboard, click delete profile button	User profile deleted successfully	Same as expected result	Pass

Table 5.4: Test Cases for Adding Ambulance Request

Test ID	Test Scenario	Test Data	ExpectedResult	Observed Result	Test Status
1	Add a new ambulance request and click on submit button	24x7 Ambulance Service, Kathamndu Contact:9840186344	Ambulance Request successfully sent and shown on screen	Same as expected result	Pass
2	Add a new ambulance request without filling information and click on submit button	24x7 Ambulance Service, Kathamndu Contact:9840186344	Ambulance Request failed; shows error message.	Same as expected result	Pass

Table 5.5: Test Cases for Searching Hospital

Test ID	Test Scenario	Test Data	ExpectedResult	Observed Result	Test Status
1	Search nearest hospital	Patan Hospital	Donorslist successfully displayed.	Same as expected result	Pass

5.2.2 Test Cases for System Testing

After completing the unit testing, system testing was performed to check the collective functionality of the system as a unified project. The working of each component within the system was individually examined

The following test scenarios were examined for system testing.

Table 5.6: Test Cases for Responsiveness

Test ID	Test Scenario	Test Data	Expected Result	Observed Result	Test Status
TR_01	Responsiveness of screen size	Devices with different dimensions i.e. width x height	Response to different dimensions	Same as expected result	Pass
TR_02	React on click	Website icons and buttons	Reacts when clicked on different icons and buttons, performing the expected actions	Same as expected result	Pass

5.3. Result Analysis

The analysis of the results of a system involved reviewing test results, recording defects, and making decisions. The main activities here were analyzing results, error detection, root cause investigation, impact assessment, and presenting the results. Identified issues during testing were rolled back to the development phase to fully prepare the system for real-life deployment. Flawless compatibility across the possibly error-free component of the system was essential. The system had undergone integration which had been tested with the already predefined test cases on various devices to attest that it was performing according to expectations.

Chapter 6: Conclusion and Future Recommendations

6.1. Conclusion

In conclusion, "Ambulance Now" is not just a web-based application but it has also the potential to save lives by allowing fast and efficient access to ambulance services. This system effectively manages the challenges of locating and requesting an ambulance, providing the shortest route to both the patient and the ambulance, along with estimated arrival times. By utilizing technologies such as Dijkstra algorithm, geo-hashing, React, Django, JavaScript, CSS, OpenStreetMap API, and Leaflet JS, the application ensures reliable and accurate performance. The Agile development approach was essential in adapting to evolving needs, utilizing iterative cycles to achieve the desired project. Thorough proper testing, including component-level and full-system tests, confirmed the application's functionality and readiness. The successful integration and testing of the system, coupled with its ability to meet all the defined requirements, ensured the project's timely and successful completion.

6.2. Future Recommendations

- Implement an interface for registering ambulances within the system to improve fleet management and operational efficiency.
- Introduce telemedicine feature for real time consultation with doctors while on the way to hospital.
- Develop a user-friendly mobile app to request and track ambulance in real time and receive update on medical advice during the wait.

•

Bibliography

[1] "geeksforgeeks," 2024. [Online]. Available: https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/.

[2] J. A. Akshay Upadhyay, "IGIS," [Online]. Available: https://www.igismap.com/haversine-formula-calculate-geographic-distance-earth/.

[3] N. E. Consultancy, 2022. [Online]. Available: nepalec.edu.np.

[4] P. Johny Jose, "Studocu," Ambulance Management System," 2023, [Online]. . [Online]. Available: https://www.studocu.com/in/document/guru-gobind-singh-indraprastha-university/bachelor-of-technology/ambulance-management-system/59961626?origin=search-results..

[5] V. W. S. A. S. P. V. Patil, " "EMERGENCY AMBULANCE BOOKING APPLICATION"," April-2022.

[6] D. S. M. Chirico, "geohashTools," 2023.

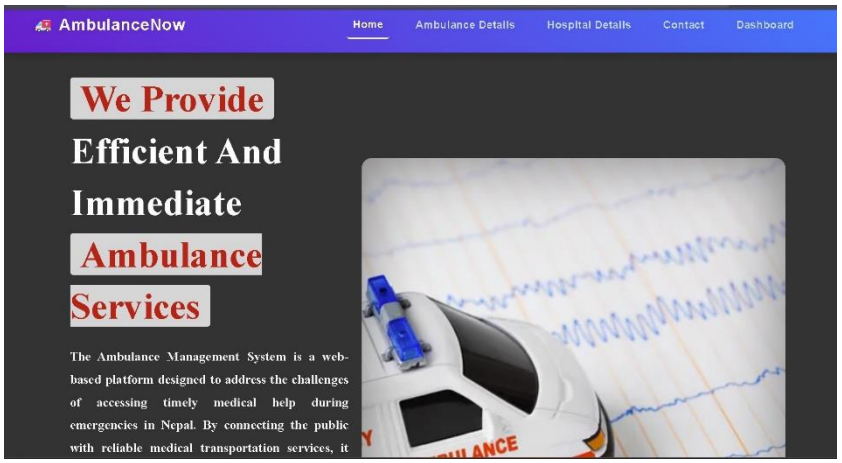
[7] A. A. M. M. S. R. S. G. D. P. P. S, "The Smart Ambulance Services," International Journal of Research Publication and Reviews," 2024.

[8] "namastesindhupalchowk," 2023. [Online]. Available: https://namastesindhupalchowk.com/blog/ambulance-service-nepal.

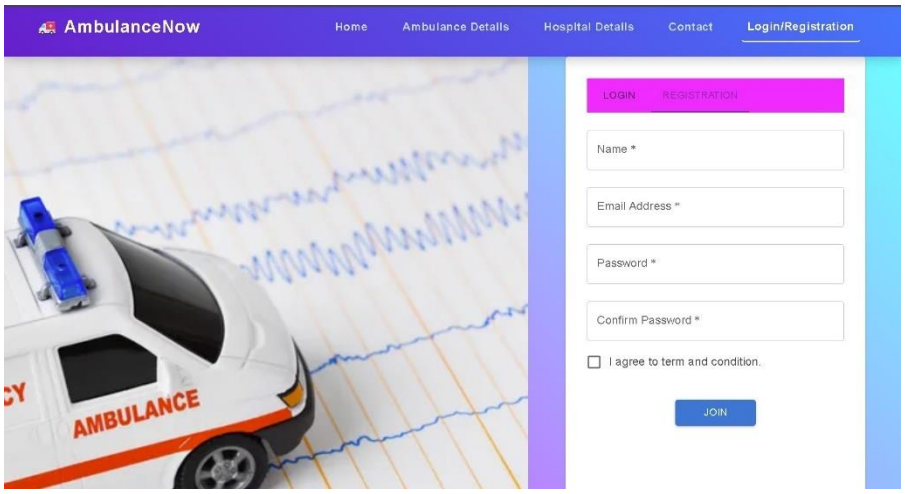
Appendices

Screenshot of Appendices

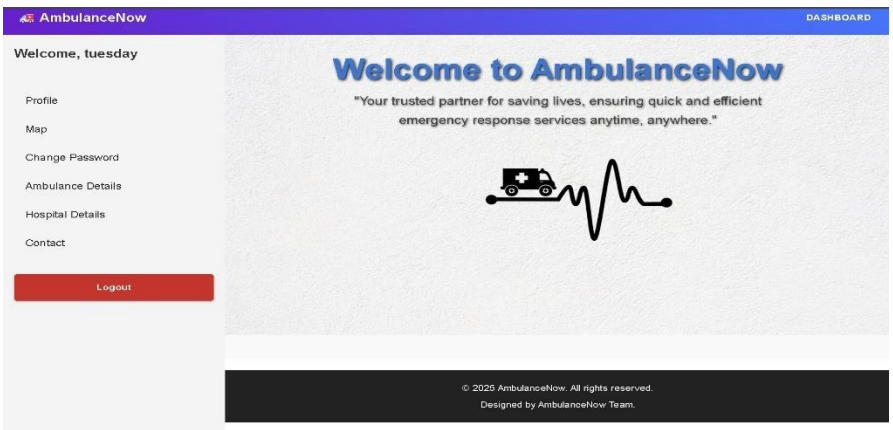
a. Login/Register Page



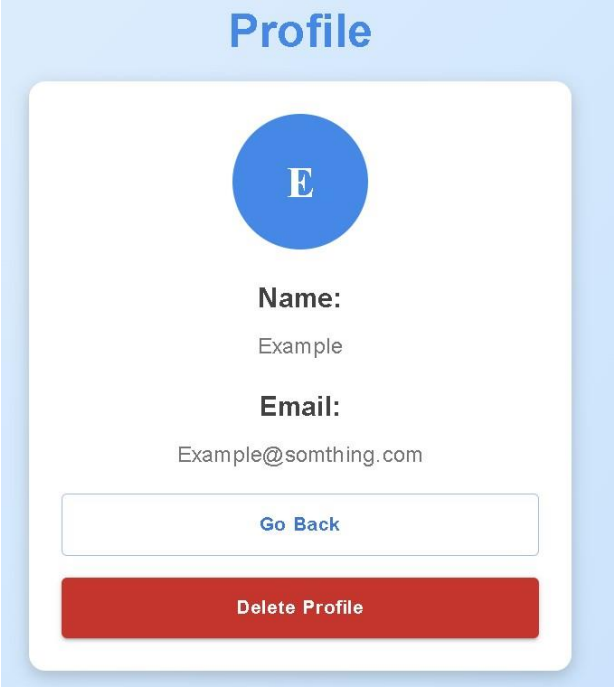
b. Landing Page



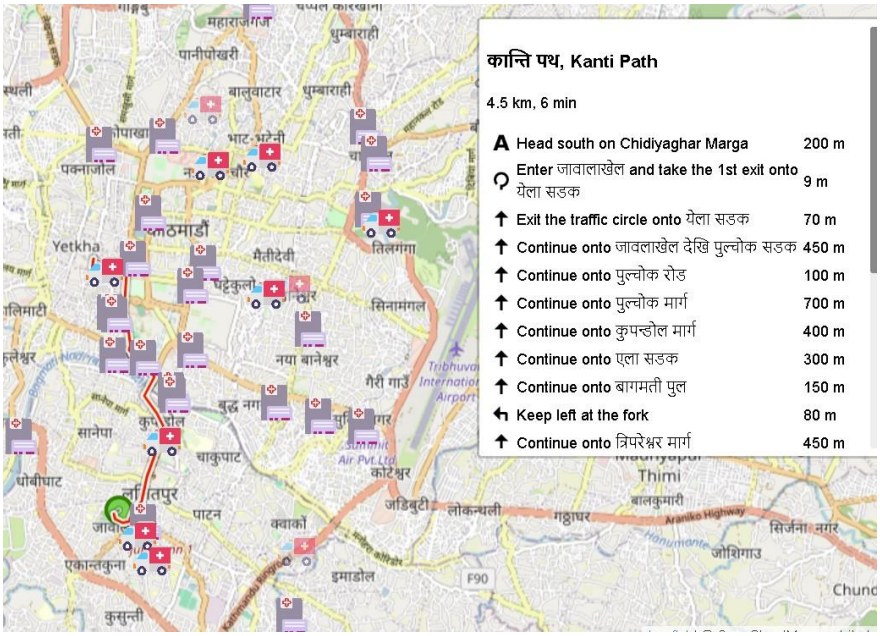
c. Dashboard



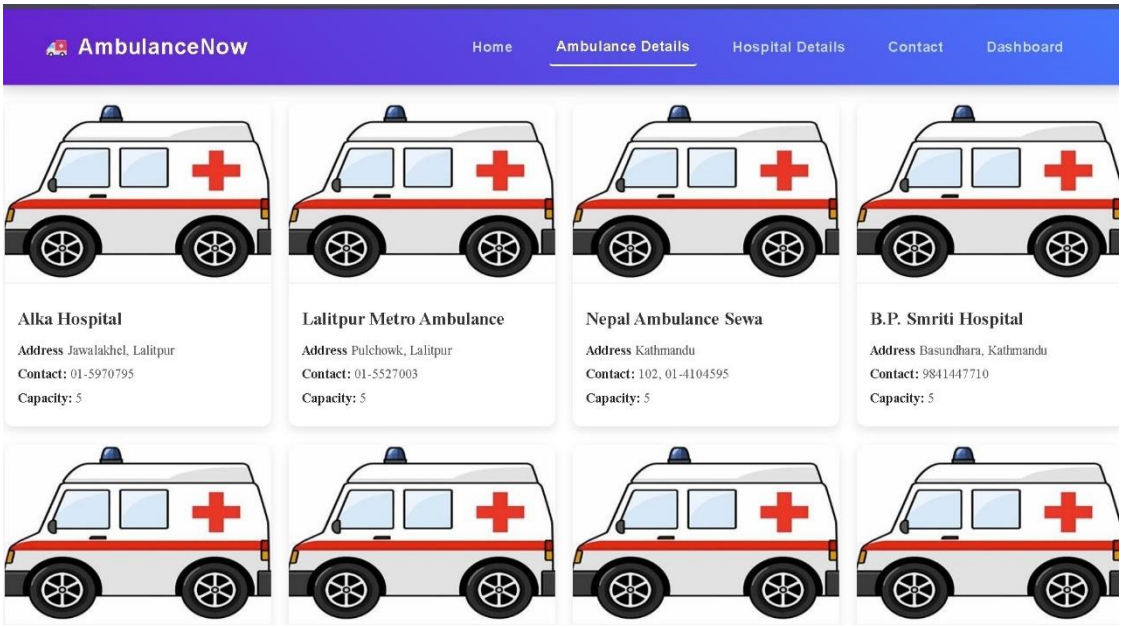
d. Profile Page



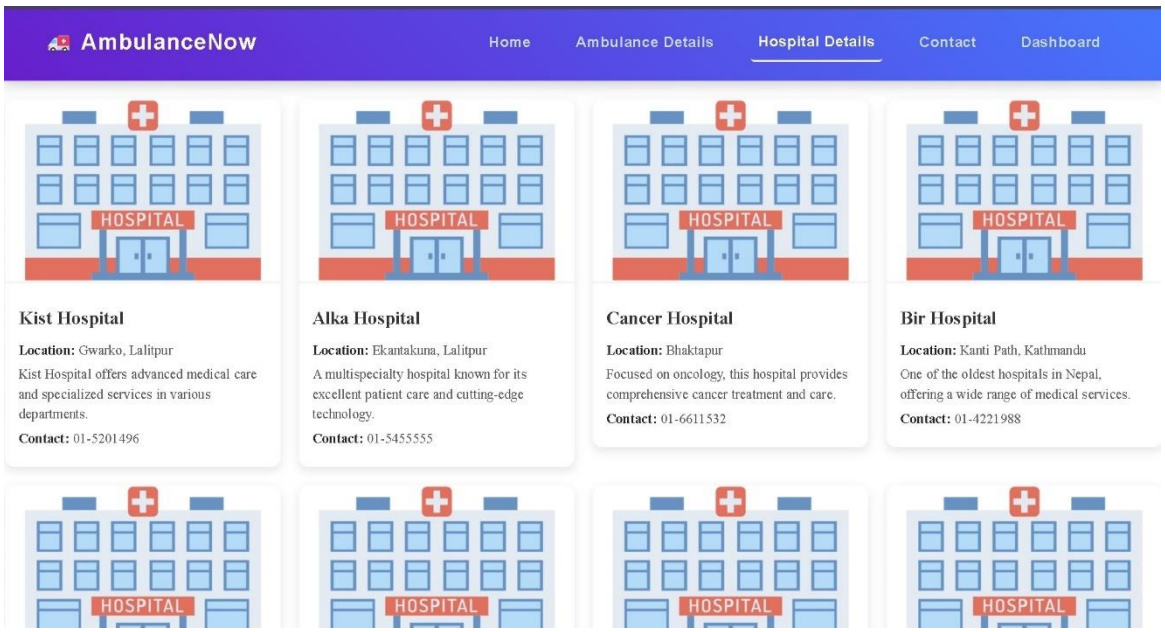
e. Map



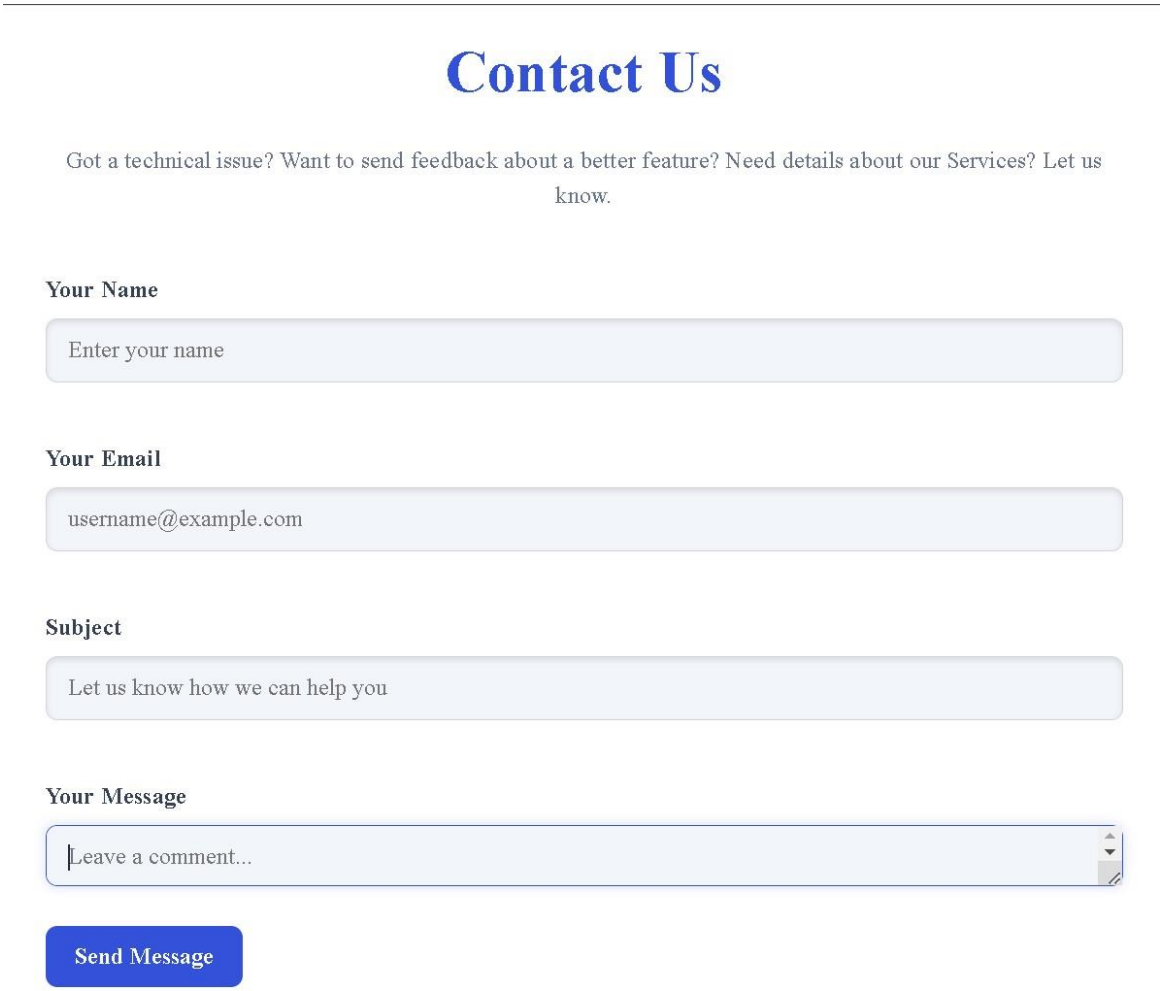
f. Ambulance List



g. Hospital List



h. Contact Us


Contact Us
Got a technical issue? Want to send feedback about a better feature? Need details about our Services? Let us know.
Your Name
Enter your name
Your Email
username@example.com
Subject
Let us know how we can help you
Your Message
Leave a comment...
Send Message

Code Implementation of Dijkstra and Haversine

```
// Function to calculate the distance between two points in km
    function calculateDistance(lat1, lon1, lat2, lon2) {
      const R = 6371; // Radius of the Earth in km
      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));
      return R * c; // Returns distance in km
    }


    // Dijkstra's algorithm to find the shortest path
    function dijkstra(graph, start, end) {
      const distances = {};
      const previous = {};
      const nodes = new PriorityQueue();

      // Initialize distances and nodes
      graph.forEach((node, i) => {
        if (i === start) {
          distances[i] = 0;
        } else {
          distances[i] = Infinity;
        }
        previous[i] = null;
        nodes.enqueue(i, distances[i]);
      });

      while (!nodes.isEmpty()) {
        const smallest = nodes.dequeue();
```

```
            if (smallest === end) {
                const path = [];
                let currentNode = end;
                while (currentNode !== start) {
                    path.push(currentNode);
                    currentNode = previous[currentNode];
                }
                return path.reverse();
            }

            const neighbors = graph[smallest];
            neighbors.forEach((neighbor, index) => {
                const alt = distances[smallest] + calculateDistance(neighbor.lat, neighbor.lng,
graph[smallest].lat, graph[smallest].lng);
                if (alt < distances[neighbor]) {
                    distances[neighbor] = alt;
                    previous[neighbor] = smallest;
                    nodes.enqueue(neighbor, distances[neighbor]);
                }
            });
        }

        return [];
    }

    function findNearestAmbulance() {
        if (userLat && userLng) {
            let nearestAmbulance = null;
            let minDistance = 10; // 10 km range
            let nearestMarker = null;

            ambulances.forEach((ambulance, index) => {
                const distance = calculateDistance(userLat, userLng, ambulance.lat,
ambulance.lng);
```

```
            if (distance <= minDistance) {
                minDistance = distance;
                nearestAmbulance = ambulance;
                nearestMarker = ambulanceMarkers[index];
            }
        });

        if (nearestAmbulance) {
            map.setView([nearestAmbulance.lat, nearestAmbulance.lng], 13);
            nearestMarker.openPopup();
        } else {
            alert("No ambulance available within 10 km.");
        }
    } else {
        alert("Unable to determine your current location. Please try again.");
    }
}
function findNearestHospital() {
    if (userLat && userLng) {
        let nearestHospital = null;
        let minDistance = 15; // Start with an infinite distance

        hospitals.forEach(hospital => {
            const distance = calculateDistance(userLat, userLng, hospital.lat,
hospital.lng);
            if (distance < minDistance) {
                minDistance = distance;
                nearestHospital = hospital;
            }
        });

        if (nearestHospital) {
            map.setView([nearestHospital.lat, nearestHospital.lng], 13);
            map.eachLayer(layer => {
```

```
                if (layer instanceof L.Marker &&
layer.getLatLng().equals([nearestHospital.lat, nearestHospital.lng])) {
                    layer.openPopup();
                }
            });
        } else {
            alert("No hospital available.");
        }
    } else {
        alert("Unable to determine your current location. Please try again.");
    }
}
```