Software Design Specification
for

Ambulance Service Application

Version 1.0 approved

Prepared by:

Maaz Alvi 1912353

Ali Nasir 1812347

Shaheed Zulfikar Ali Bhutto Institute of Science and Technology

January 01, 2023

Software Design Specification

The Software Design Specification Outline

1. Introduction

1.1 Purpose of this document

This document intends to define the requirements for Ambulance Service mobile application. We will be the system developers and engineers and, we will also do requirement elicitation on one's own by identifying and documenting functional, performance, or other requirements for the Ambulance service android based application. There is no real occurring monetary cost associated with the project. The project is expected to be delivered by the end of December 2023.

1.2 Scope of the development project

- With an increasing emphasis on promoting independent living today, having access to the nearest ambulance to you can provide much needed peace of mind in a worst case scenario.
- To improve quality of healthcare services and provide a platform to help users to find health solutions that are convenient and useful enough to continue for a longer time.
- To integrate city emergency ambulance transportation for patients and partner ambulance drivers onto a platform that is convenient, transparent and provides immediate emergency service fulfillment

1.3 Definitions, acronyms, and abbreviations

Serial #	**Abbreviations**	Definitions
1	IDE	Integrated Development Environment
2	DB	Database
3	FR	Functional Requirement
4	NFR	Non-Functional Requirement
5	GPS	Global Positioning System
6	API	Application Programmable Interface
7	GUI	Graphical User Interface

1.4 References

- https://www.msdsb.net/images/EMS/policies/Ambulance%20Service%20Documentation%20Standards%20April2000.pdf
- https://jdcwelfare.org/campaigns/ambulance-services/

- https://bmcemergmed.biomedcentral.com/articles/10.1186/s12873-019-0297-3
- https://www.icrc.org/en/document/best-practices-ambulance-services-excellent-health-care-patients

1.5 Overview of document

The purpose of this document is to describe the requirements of the system completely, accurately, and unambiguously in a Technology-independent manner. It will provide a detailed description of the project's functional, non-functional requirements, and other requirements for the system, and all the features which will be present in the system will also be expressed in this document

2. System architecture description

2.1. Section Overview

This section provides an overview and rationale for the program's data and architectural design decisions.

2.2. General Constraints

Here is the list of general constraints:

i. Reliable Network Connectivity: The application heavily relies on network connectivity for communication, location tracking, and data exchange. Ensure that the infrastructure provides reliable and robust network connectivity to minimize downtime and disruptions.

ii. Scalability: The ambulance service application should be designed to handle a potentially large number of concurrent users and emergency requests. Scalability considerations should be taken into account to accommodate future growth and sudden surges in demand.

iii. Security and Privacy: As the application deals with sensitive user information and communication, implement robust security measures to protect data privacy and prevent unauthorized access. Encryption, user authentication, and secure communication protocols are essential components to address security concerns.

iv. Compatibility and Interoperability: The ambulance service application needs to be compatible with various devices and systems within the SDS ecosystem. Consider interoperability standards and protocols to ensure seamless integration with other components, such as GPS devices, communication systems, and emergency response infrastructure.

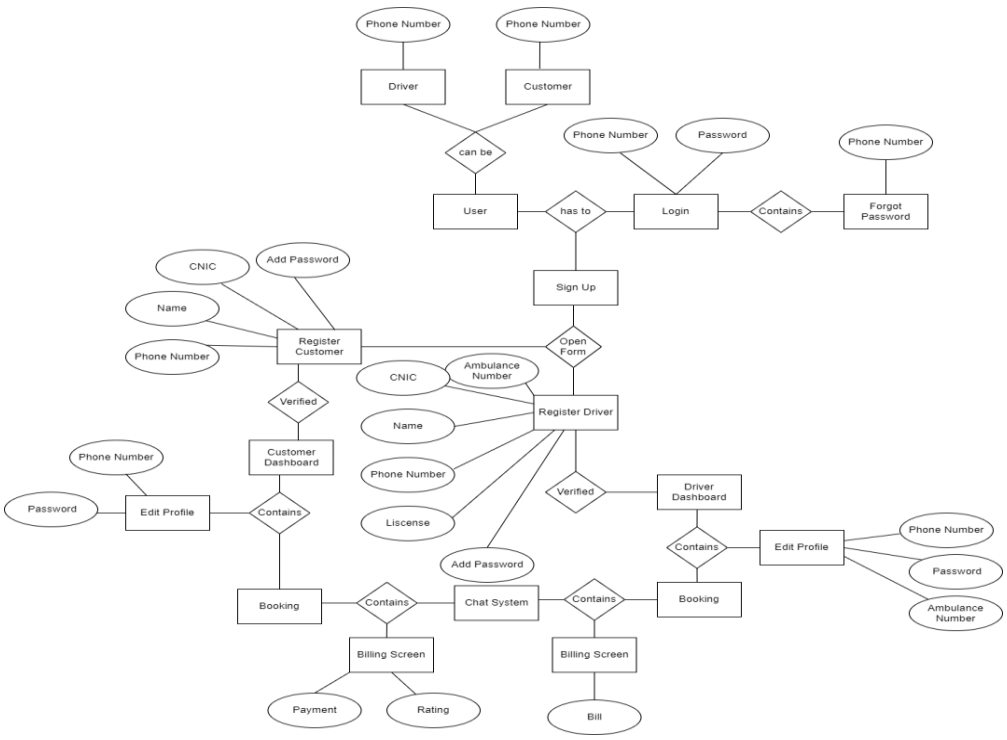
v. User Experience and Training: The application should be designed with a user-centric approach, considering the needs of emergency responders,

dispatchers, and end-users. User experience (UX) principles should guide the design to ensure ease of use and minimize errors during high-stress situations. Additionally, provide training and support for users to familiarize themselves with the application and its features.

vi. Performance and Reliability: The ambulance service application should be highly available and reliable, capable of handling peak loads and critical situations without interruptions or delays. Implement measures such as load balancing, fault tolerance, and disaster recovery strategies to ensure continuous operation.

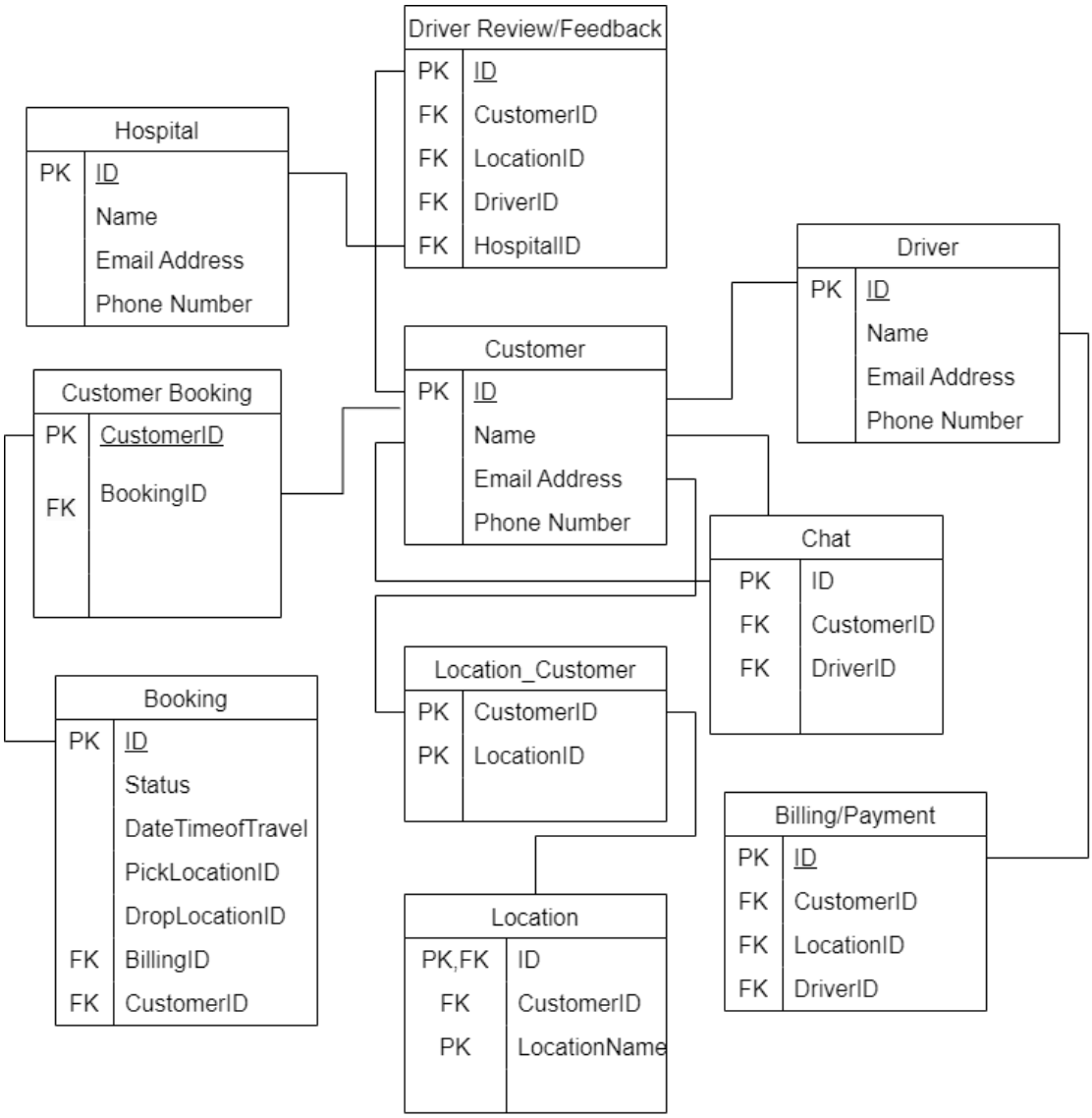
vii. Data Management and Analytics: The application should have mechanisms to capture, store, and analyze relevant data generated during emergency incidents. This data can be used for improving response times, optimizing resource allocation, and enhancing overall system performance.

2.3. Data Design



System includes different entities which stores data into database. The database used for our application is MangoDB. MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. Adding live GPS tracking functionality for safety purpose.

2.4. **Program Structure**



Class Customer has 4 attributes i.e., Attributes Name, Email and password is type string and phone number is type int. Class Driver has 3 attributes. Attributes Name, Email of type string and phone number is type int.

2.5. **Alternatives Considered**

Not Included

3. **Detailed description of components**

3.1. **Section Overview**

3.2. **Component n Detail (include a sub-section for each component)**

3.2.1. Description

In this section, each component's detailed procedural structure and working is shown in a more systematic way.

3.2.2. Data Members (include type, visibility, and description)

S.NO	Attribute	Type	Visibility
1.	DATE	Datetime (Integer)	Visible to all members
2.	NAME	string	Visible to all members
3.	GPS	Integer	Visible to all members
4.	EMAIL	Char(56)	Not Visible to all members
5.	PASSWORD	Char(15)	Private
6.	MESSAGE	string	Visible to only the member who has sent message and the other who receive it

3.2.3. Methods (include English or psuedocode descriptions for each one)

```
FUNCTION requestAmbulance(patientLocation, patientCondition)
   IF patientLocation is valid AND patientCondition is valid
      CALL ambulanceServiceAPI to request an ambulance with patientLocation and
patientCondition
      IF ambulanceServiceAPI response is successful
         DISPLAY "Ambulance request successful. Help is on the way!"
      ELSE
         DISPLAY "Sorry, unable to process ambulance request at the moment. Please
try again later."
      END IF
   ELSE
```

```
      DISPLAY "Invalid patient location or condition. Please provide valid
information."
   END IF
END FUNCTION


FUNCTION trackAmbulance(ambulanceID)
   IF ambulanceID is valid
      CALL ambulanceServiceAPI to get the current location and status of
ambulanceID
      IF ambulanceServiceAPI response is successful
         DISPLAY "Ambulance is currently at <location>. Status: <status>"
      ELSE
         DISPLAY "Unable to track ambulance at the moment. Please try again later."
      END IF
   ELSE
      DISPLAY "Invalid ambulance ID. Please provide a valid ID."
   END IF
END FUNCTION


FUNCTION rateAmbulance(ambulanceID, rating)
   IF ambulanceID is valid AND rating is valid
      CALL ambulanceServiceAPI to submit the rating for ambulanceID
      IF ambulanceServiceAPI response is successful
         DISPLAY "Thank you for rating the ambulance service."
      ELSE
         DISPLAY "Sorry, unable to submit rating at the moment. Please try again
later."
      END IF
   ELSE
      DISPLAY "Invalid ambulance ID or rating. Please provide valid information."
   END IF
END FUNCTION


FUNCTION viewAmbulanceHistory(patientID)
   IF patientID is valid
      CALL ambulanceServiceAPI to retrieve the ambulance history for patientID
      IF ambulanceServiceAPI response is successful
         DISPLAY "Ambulance history for patientID:"
         FOR EACH ambulanceRecord in ambulanceServiceAPI response
            DISPLAY "Ambulance ID: <ambulanceID> - Date: <date> - Location:
<location>"
         END FOR
      ELSE
         DISPLAY "Unable to retrieve ambulance history at the moment. Please try
again later."
      END IF
   ELSE
```

```
    DISPLAY "Invalid patient ID. Please provide a valid ID."
  END IF
END FUNCTION
```

SDS component template

COMPONENT 1

Identification	Customer
Type	Customer can book ride
Purpose	Customer can book ride and call or chat with the driver
Function	Customer can book ride and call or chat with the driver
Subordinates	--
Dependencies	Customer is dependent on Rider
Interfaces	----------
Resources	Android Smartphones with Processor Minimum 1 GHZ RAM Minimum 1 GB Storage Minimum 4GB should be used
Processing	----------
Data	Data is a customer's data that is filled in registration time

COMPONENT 2

Identification	Driver
Type	Driver picks ride and drop off the customer
Purpose	Driver picks ride and drop off the customer
Function	Driver picks ride and drop off the customer
Subordinates	--
Dependencies	Driver is dependent on customer
Interfaces	----------
Resources	Android Smartphones with Processor Minimum 1 GHZ RAM Minimum 1 GB Storage Minimum 4GB should be used
Processing	----------
Data	Data is a driver's data that is filled in registration time

4. User Interface Design

4.1. **Section Overview**

Provide a summary of the contents of this section.

4.2. Interface Design Rules

When designing the interface for an ambulance service application within a Smart Delivery System (SDS), it's important to prioritize usability, efficiency, and clear communication of critical information. Here are some interface design rules to consider:

1. Keep it Simple: Strive for a clean and uncluttered interface design that focuses on the most important elements. Avoid excessive visual noise and unnecessary complexity that can distract users during critical situations.

2. Prioritize Key Information: Display essential information prominently and make it easily noticeable. Highlight critical details such as emergency call buttons, location information, and estimated arrival times for ambulances.

3. Clear Navigation and Hierarchy: Ensure intuitive navigation with logical grouping of features. Use clear labels and visual cues to guide users through the application. Follow established design patterns and conventions for consistency and familiarity.

4. Responsive Design: Design the interface to be responsive and adaptable to different screen sizes and device orientations. Ambulance staff may use the application on tablets, smartphones, or in-vehicle systems, so optimize the interface for each platform.

5. Consistent Visual Language: Establish a consistent visual language, including typography, color scheme, and iconography, throughout the application. This creates a cohesive user experience and helps users quickly understand the interface elements.

6. Clear Call-to-Action Buttons: Use large, easily tappable buttons for important actions like emergency calls or updating the status. Ensure that these buttons are easily accessible and visually distinct from other elements.

7. Color Coding and Alerts: Utilize color coding to convey critical information and status updates. For example, use red for emergency alerts, green for available ambulances, and yellow for in-progress calls. Be mindful of color blindness and ensure sufficient contrast for readability.

8. Input Validation and Error Handling: Implement input validation to prevent errors during data entry. Provide clear error messages and feedback to guide users in correcting mistakes. Consider using inline validation to provide real-time feedback as users input information.

9. Feedback and Notifications: Provide clear feedback and notifications to keep users informed about the progress of their requests. This can include

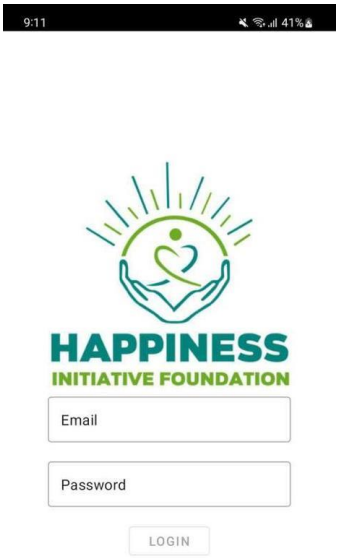
confirmation messages for successful actions, progress indicators for ongoing processes, and notifications for updates or changes in status.

10. Accessibility Considerations: Ensure the interface is accessible to users with disabilities. Follow WCAG (Web Content Accessibility Guidelines) standards, including providing alternative text for images, supporting keyboard navigation, and using appropriate color contrasts.

11. User Testing and Iteration: Conduct user testing and gather feedback from ambulance staff and other stakeholders to identify pain points and areas for improvement. Iterate on the interface design based on user insights to enhance usability and efficiency.

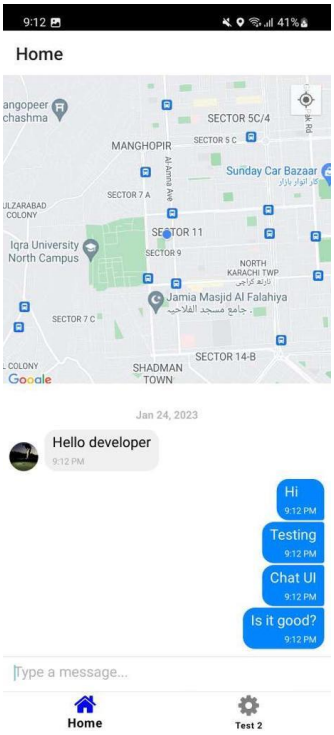
4.3. **GUI Components**

Registration Screen

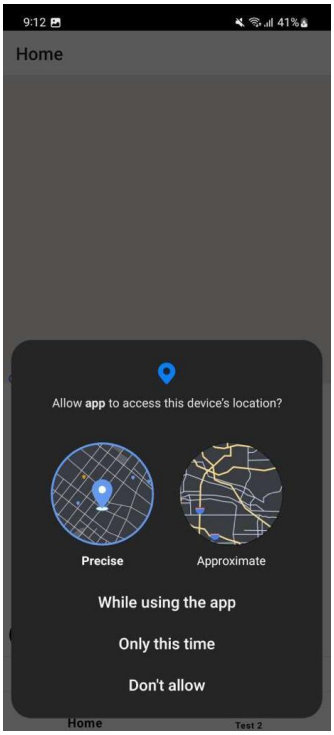


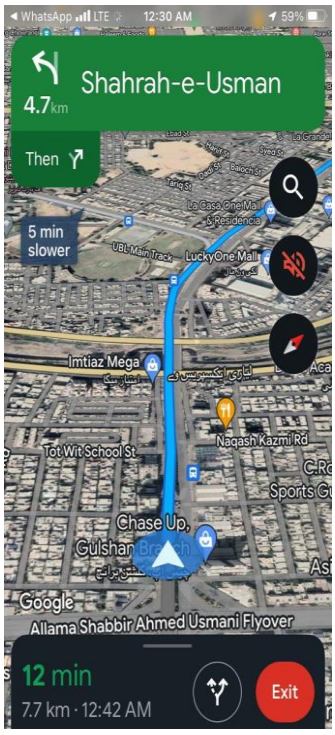
Login Screen

Main/Home Screen



Message Screen





GPS Screen

4.4. **Detailed Description**

The ambulance service application focuses on providing a platform for users to register as drivers and facilitating communication between drivers and customers who have booked the ambulance service. Additionally, the application enables customers to access GPS functionality and share their location for improved coordination. Here's a detailed description of the application's features:

1. User Registration:
   - The application allows users to register as ambulance drivers by providing their relevant information, such as name, contact details.
   - During the registration process, users may need to undergo verification and provide necessary documentation to ensure their eligibility as drivers.

2. Driver-Customer Chat:
   - Once a customer books an ambulance service, the application enables direct communication between the driver and the customer through a chat feature.
   - The chat functionality allows for real-time text-based communication, facilitating coordination and addressing any specific requirements or concerns.

3. GPS and Location Sharing:
   - The application integrates GPS functionality, enabling customers to access real-time location tracking.
   - Customers can share their location with the driver, allowing for efficient and accurate dispatching of the ambulance to the customer's current location.
   - This feature enhances coordination between the driver and the customer, ensuring that the ambulance reaches the customer promptly.

4. Booking Management:
   - The application includes a booking management system that allows customers to request ambulance services, specifying their location and any additional details or requirements.
   - Drivers can view and manage incoming booking requests, accepting or rejecting them based on their availability and proximity to the customer's location.

5. 7. Notifications and Alerts:

- The application sends notifications and alerts to both drivers and customers to keep them informed about booking confirmations, driver assignments, and trip updates.
- These notifications can be delivered via in-app messages, SMS, or push notifications to ensure timely communication.

5.0 Reuse and relationships to other products
NOT INCLUDED

6.0 Design decisions and tradeoffs

- The UI should be intuitive and user-friendly, with clear navigation and easily accessible features. Consider the use of large buttons, clear icons, and minimal text for quick comprehension during emergency situations.

- Implement a prominent and easily accessible emergency call button within the application. This button should connect the user directly to emergency services to report an incident or request immediate assistance.

- Integration with GPS technology is crucial to track the user's location accurately and enable real-time tracking of ambulances. This allows dispatchers to allocate the nearest available ambulance efficiently.

- Include features for seamless communication between the user, emergency dispatchers, and ambulance staff. This can be through in-app messaging, voice calls, or video calls to provide necessary information and updates.
- Provide a section for users to input their medical information, such as allergies, pre-existing conditions, and medications. This information can be crucial for emergency responders to provide appropriate care.

- Develop an algorithm or system to prioritize emergency calls based on the severity of the situation. This can help allocate ambulances efficiently and ensure that critical cases receive immediate attention.

- If the ambulance service is part of a larger Smart Delivery System, ensure seamless integration with other components of the system. This may include coordinating with logistics, traffic management, and emergency services to optimize response times.

- Implement robust security measures to protect user data, including personal information and medical records. Adhere to relevant data protection regulations and ensure secure communication channels for sensitive information.

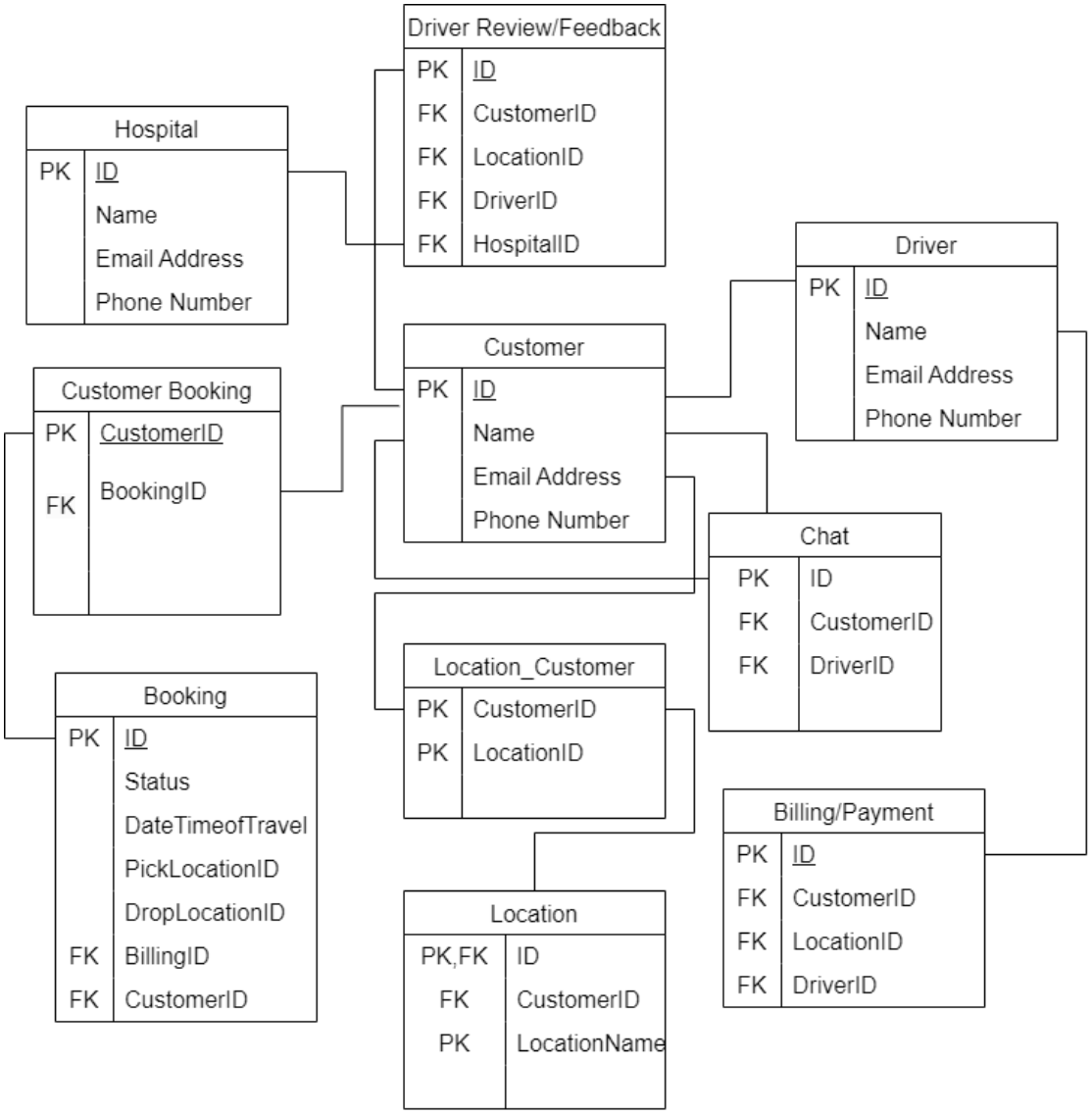
- Ensure the application is compatible with a wide range of devices, including smartphones, tablets, and wearable devices. This allows for broader accessibility and availability during emergencies.

- The application should be designed to handle high traffic and provide reliable service during critical situations. Consider load testing, redundant server architecture, and efficient data handling to ensure smooth operation.

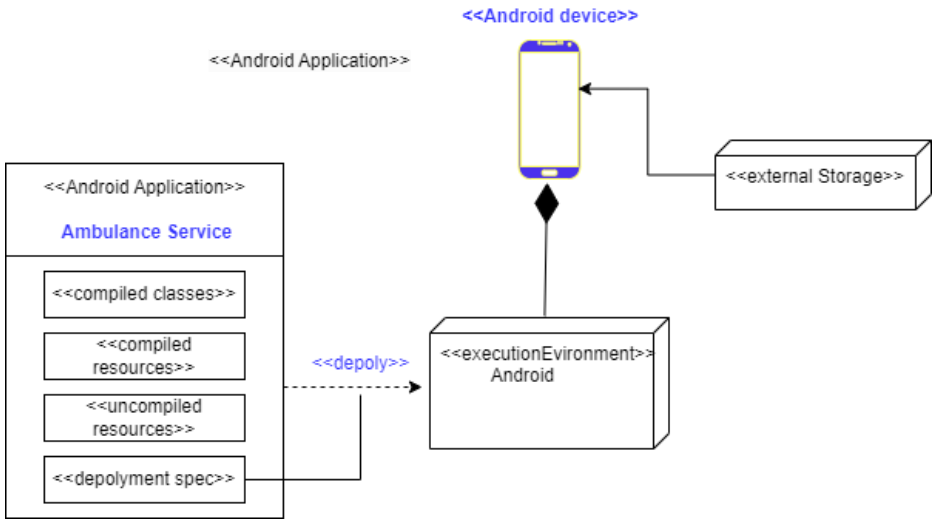
7.0 Pseudocode for components

```
function contactAmbulance(ambulanceID):
    // Connect to the ambulance staff
    connectToAmbulance(ambulanceID)

    // Establish voice or video call with ambulance staff
    establishCommunication(ambulanceID)

function sendUpdateToAmbulance(ambulanceID, update):
    // Send an update to the ambulance staff
    sendMessage(ambulanceID, update)

function receiveUpdateFromAmbulance():
    // Receive updates from the ambulance staff
    update = receiveMessage()

    return update
```

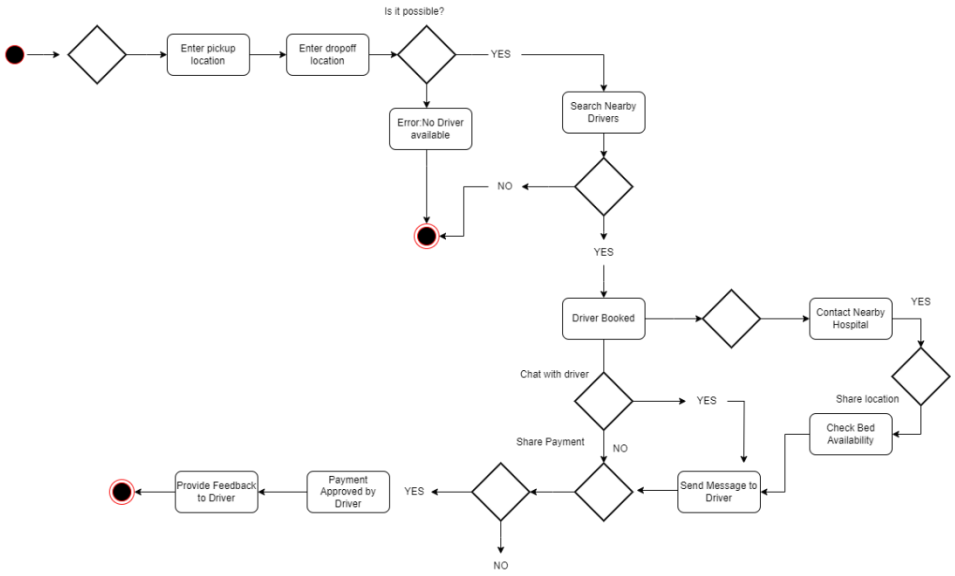
8.0 Appendices



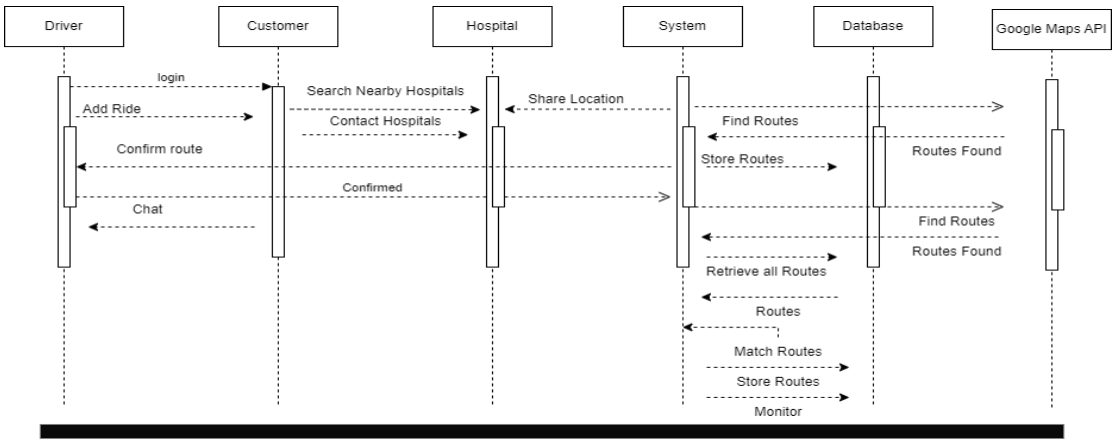
Class Diagram



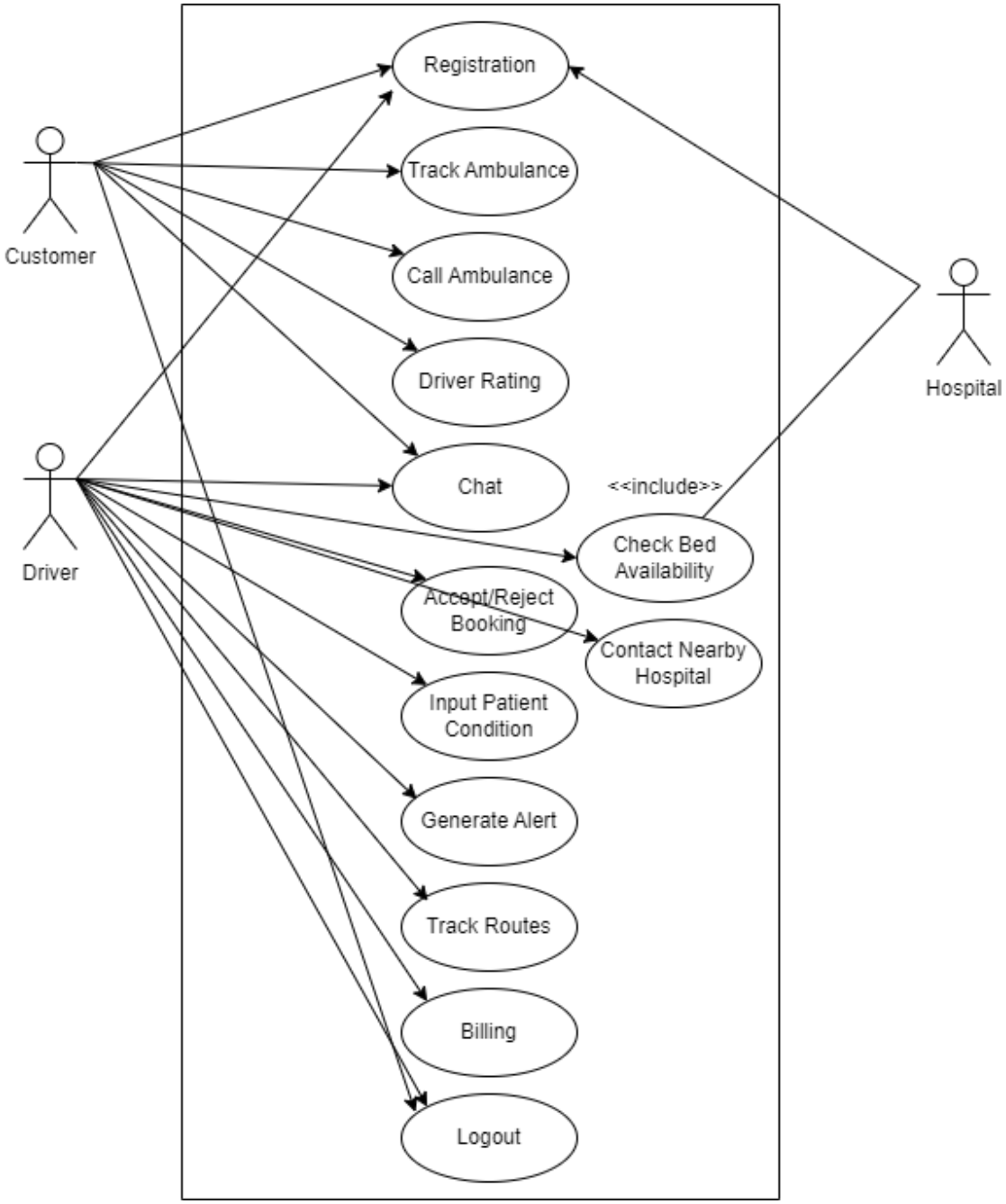
Deployment Diagram



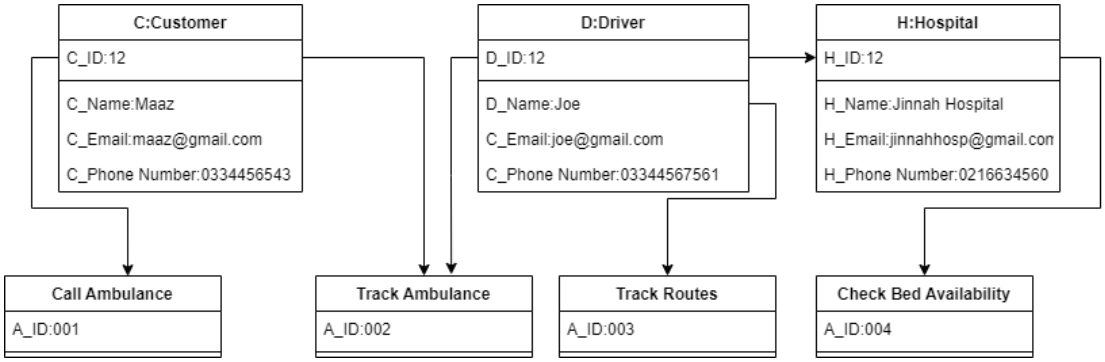
Activity Diagram



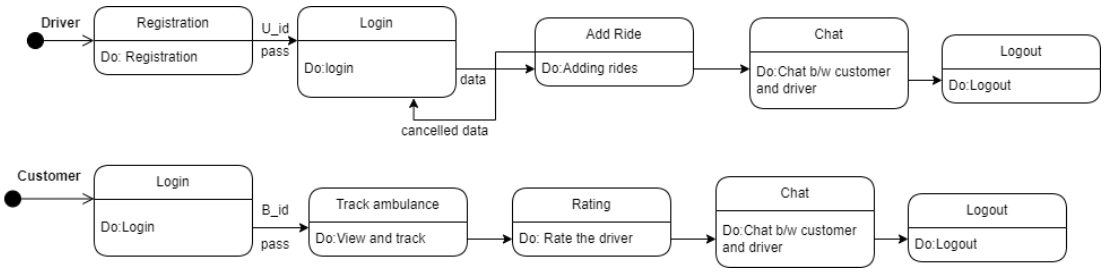
System Block Diagram



Use-case Diagram



Object Diagram



Statechart Diagram