Software Requirements Specification
for
Mechanic App

Prepared by:
Mahfuzur Rahman Rafi
Sultana Sharmin
Shatabdi Mondol Shila
Raihan Aziz

Introduction:

Overview of the project: Our app name is **Mechanic**. In our daily life, if something like : AC, car, TV, washing machine or any water line of the house is broken, we have to face a lot of difficulties. Finding the right mechanic at the right time becomes difficult. To solve this problem we thought of making our app Mechanic. Through this app, users can find a specific mechanic for their problem with their location.

Its purpose and goals: Through our app, many problems of daily life of common people will be solved. people don't find mechanics when they need them. As a result they faced many difficulties. This app will remove these difficulties for them. People can easily find mechanics from this app. There will be 3 panels. Admin panel, Mechanic and user. This app will make our daily life easier.

Functional requirements:

Both user Mechanic and customer will be able to create accounts, log in, and manage their profiles. Customers will be able to submit service requests detailing their issues. Customers will be able to search for mechanics based on location, availability, services offered, ratings, and reviews. Also they can book appointments with mechanics and re-schedule service appointments. Customers can communicate with mechanics through in-app messaging or calls. Customers will be able to rate and review mechanics based on their experience.

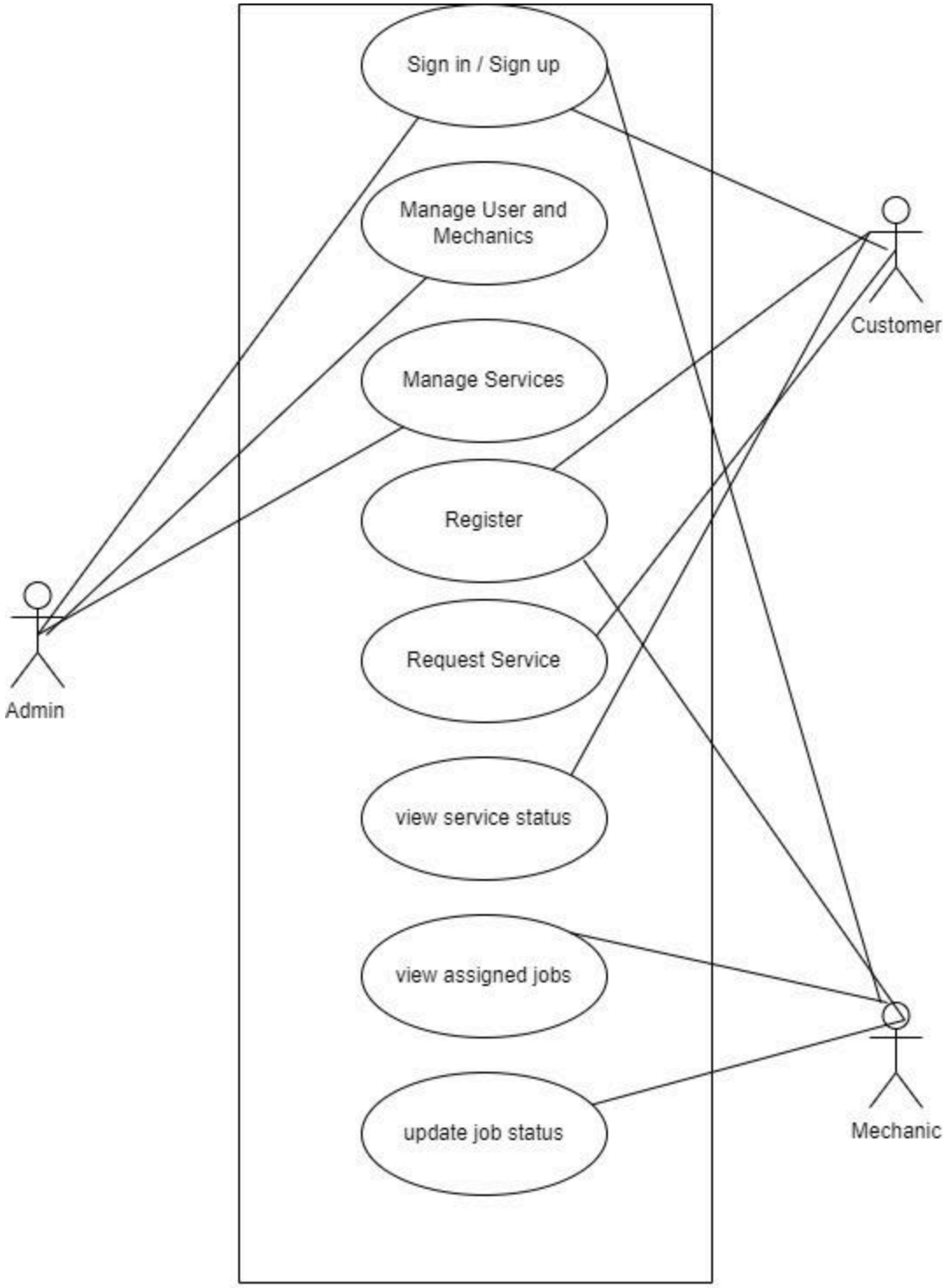
Non-functional requirements:

The app will load quickly and respond promptly to user interactions, even during peak usage times. The app will be able to handle a growing number of users and service requests without significant degradation in performance. The app will be available and reliable, with minimal downtime or service interruptions. The app will implement robust security measures to protect user data, including encryption of sensitive information and secure authentication mechanisms. It will be easy to maintain and update, with well-documented code and modular architecture. It will collect and analyze performance metrics, such as response times and error rates, to continuously improve its performance.

User story:

When a user opens the Mechanic app on their smartphone. They are presented with a search bar where they can enter their location or use GPS to detect it automatically. they enter their location and tap the search button. The app displays a list of nearby mechanics along with their ratings and reviews. They browse through the list and select a mechanic with high ratings and positive reviews. After selecting a mechanic, customers see their contact details and a button to request service. They tap the button to request a service appointment. The app prompts them to enter details about their issue that they are experiencing. Customers provide the necessary information and submit the request. The mechanic receives customers' requests and confirms the appointment through the app. Customers also can give feedback, review and rating about the service through the app.

Use case diagram:


Sign in / Sign up
Manage User and Mechanics
Manage Services
Register
Request Service
view service status
view assigned jobs
update job status
Admin
Customer
Mechanic

Client-Side (Mobile App):
- Developed using Flutter for cross-platform compatibility (iOS and Android).
- Utilizes Flutter widgets for the user interface, following the design we've planned.

User Authentication:
- Integrates Firebase Authentication for secure user registration and login.

Database:
- Firebase Firestore or Realtime Database stores user profiles, service provider information, service requests, and other relevant data.

User Interface:
- Adheres to the UI/UX design we've created, with responsive layouts for different devices.

UX/UI Design guidelines:

Homepage:
- Feature a search bar prominently for users to search for services

- Display categories or popular services for quick access.

User Authentication:
- Create clean and straightforward screens for user registration and login.
- Include options for both customers and service providers to register.

Customer Dashboard:
- Show recent searches and suggested services.
- Include a section for ongoing or bookmarked services.

Admin Interface:

- Design an interface for admin to review service provider registrations, manage user data, and monitor system activity.

Profile Pages:

- Both customers and service providers should have editable profile pages with relevant details.

Booking/Request Flow:

- Design an intuitive process for customers to request services and for service providers to respond.

Ratings:

- Include a system for users to leave feedback and ratings for service providers.

Settings:

- Create a settings page for users to manage account preferences, notifications, and privacy setting