Institute of Technology University of Moratuwa

Service recruitment and tool rental App Development

Software Requirements Specification

Version 1.0

Student Details:

MAR Ahamed	21IT0453
MRM Rishaf	21IT0521
WGHD Rathnayake	21IT0520
JM Naveeth	21IT0502
P Manovithan	21IT0498

Supervised by:

Dr.(Mrs). Kalpana Galappaththi

Division of Information Technology

Institute of Technology University of Moratuwa

April 2024

Table of contents

Table of figures

Introduction

The Software Requirements Specification (SRS) for our Service recruitment and tool rental App is designed to delineate the functional and non-functional necessities essential for the creation of a comprehensive mobile application. This app is crafted with the aim of providing users with a seamless and efficient platform for accessing various Service recruitment and tool rental services. With a focus on user convenience and service accessibility, our Service recruitment and tool rental App encompasses features such as rental tools, live chatbot support, pre-booking capabilities, and quick match tracking options.

The primary objective of the Service recruitment and tool rental app is to simplify and streamline the process of accessing Service recruitment and tool rental services, ensuring a hassle-free experience for both service providers and users. By integrating innovative functionalities like rental tools, users can easily locate and rent required tools for their tasks directly through the app. The inclusion of a live chatbot further enhances user experience by providing instant assistance and support whenever needed.

Moreover, the pre-booking option allows users to schedule appointments in advance, enabling efficient planning and allocation of resources for service providers. Additionally, the quick match tracking feature ensures real-time updates on service requests and their status, enabling users to stay informed throughout the service delivery process.

This SRS will comprehensively define the functionalities, user roles, security protocols, performance benchmarks, and integration requirements essential for the development of the Service recruitment and tool rental app. By adhering to these specifications, we aim to deliver a robust and user-centric solution that meets the evolving needs of both service providers and users in the handyman industry.

1.1 Purpose

The purpose of the Software Requirements Specification (SRS) for the Service recruitment and tool rental app is to clearly define and document both the functional and non-functional requirements of the application. This document serves as a guide for the development team, stakeholders, and users, outlining the specific objectives and features of the app.by detailing the functionalities such as rental tools option, live chatbot, pre-booking option, and quick match tracking option, the SRS ensures that everyone involved shares a common understanding of the app's capabilities and expectations

Moreover, the SRS acts as a communication tool, fostering discussions and resolving any uncertainties or misconceptions. It aids in project planning, estimation, and resource allocation by providing a clear roadmap and scope for the development team.

Ultimately, the Service recruitment and tool rental app's SRS ensures that the application is developed to meet the desired functionality, usability, security, and performance standards, thereby delivering a high-quality solution that fulfills the needs of both service providers and users.

1.2 Product Scope

The service recruitment and tool rental app facilitates seamless connections between users and handymen services, offering a convenient platform for pre-booking appointments and renting tools. With features like live chatbot assistance and quick match tracking, users can easily find and schedule services while tracking their progress in real-time. The app aims to enhance user experience, ensure service reliability through vetting processes, and generate revenue through premium features or commission-based services. Limitations include a focus solely on handymen services and tool rentals, with compatibility limited to iOS and Android devices, assuming stable internet connectivity and compliance with local regulations by registered handymen. Dependencies include integration with reliable chatbot platforms and calendar functionalities for effective service scheduling.

1.2.1 Aim and Objectives

Aim:

Finding service providers as well as getting access to required tools should be made as simple and efficient as possible with the help of the service recruitment and tool rental app. Improving productivity and efficiency for users inside the network or employer is our main objective. Our goal is to streamline the procedures of tool rental and service recruiting by providing a full of options mobile application, which will help consumers save time and money. Our main goal is to offer a user-friendly platform that includes robust pre-booking, search, and communication tools so that everyone involved has a smooth experience. Furthermore, our objective is to establish confidence and dependability on the platform by means of a review and rating system that encourages accountability and openness.

Objective:

1. Develop a user-friendly mobile application specifically for our community to speed up the tool rental and service recruiting procedures.
2. Offer a smooth platform where consumers can look up and retrieve details about verified service providers and available resources according to their needs.
3. Put in place a simple booking system that makes it easy for customers to reserve tools for their projects and make appointments with service providers.
4. Include a review and rating system that allows users to submit feedback on tool and service providers, fostering transparency and confidence among users.
5. Provide users with efficient channels for communication within the app, enabling smooth coordination and cooperation during the tool rental and service hiring procedures.
6. Increase customer happiness and convenience by providing committed customer service to quickly resolve questions, problems, or technical help.
7. Streamline workflows and reduce costs associated with traditional service recruitment and tool renting methods, ultimately improving efficiency and productivity.

8. Encourage user participation in the community and teamwork, creating an environment of mutual assistance and communication on the platform.
9. Improve overall user experience and satisfaction while supporting the growth and development of the organization.

By fulfilling these aims and objectives, the Handyman App will revolutionize the way home maintenance tasks are handled, ensuring efficiency, organization, and satisfaction for users and service providers alike. Through streamlined booking processes, seamless tool rentals, and real-time assistance, the app aims to simplify the entire handyman experience, making it more accessible and enjoyable for homeowners, handymen, and the platform administrators.

1.2.2 Project Boundary

The Software Requirements Specification (SRS) for the Service Recruitment and Tool Rental Application defines the following boundaries:

1. Application Scope: The SRS focuses on the development of a mobile application designed exclusively for managing handyman services and tool rentals. It will include features such as service booking, tool inventory management, administrative tasks, and integration with live chatbot assistance. However, it excludes features not directly related to the mobile platform, such as web-based functionalities or social media integration.

2. User Roles: The SRS identifies four main user roles. such as: service seekers, handymen, shop owners, and administrators. The system will cater to the specific needs of each role, allowing service seekers to browse and book handyman services as well as rent tools, handymen to manage their service offerings and tool inventory, shop owners to list and manage tools available for rent, and administrators to oversee platform operations, including user management and content moderation.
3. Quick Match Tracking and Live Chatbot: The SRS incorporates a quick match tracking feature to allow users to monitor the status of their service requests and bookings in real-time. Additionally, it includes integration of a live chatbot to provide instant assistance to users, including service inquiries, booking assistance, and general support. However, the scope does not extend to the integration of other IoT devices or systems beyond the match tracking and chatbot functionalities specifically related to the handyman services.

4. Platforms: The SRS focuses on the development of a mobile application tailored for handheld devices running on major platforms like iOS and Android. The scope does not include the development of a web application, desktop application, or compatibility with smart TVs or other platforms beyond mobile devices.

5. Localization: The SRS anticipates the development of the Handyman App in specific languages based on the target user demographics. However, it does not encompass extensive localization efforts beyond language translations and adapting date and time formats to suit different regions or cultures.

6. Scalability: The SRS focuses on developing a scalable system capable of accommodating a reasonable number of users, service requests, and administrators. However, it does not incorporate specific provisions for scaling the system to support a significantly larger user base or expanded tool rental inventory without additional considerations and performance optimizations.

By delineating these project boundaries, the SRS ensures that the development team and stakeholders possess a comprehensive comprehension of the app's scope, constraints, and key functionalities, enabling efficient planning and successful implementation of the service recruitment and tool rental app project.

2 Overall Description

2.1 Product Perspective

The Handyman App comprises several core components, primarily the mobile application serving as the primary interface for users to book services and manage their bookings. Additionally, there are features for quick match tracking and live chatbot assistance within the mobile app. The app's functionalities ensure a cohesive and user-friendly experience for both service seekers and providers.

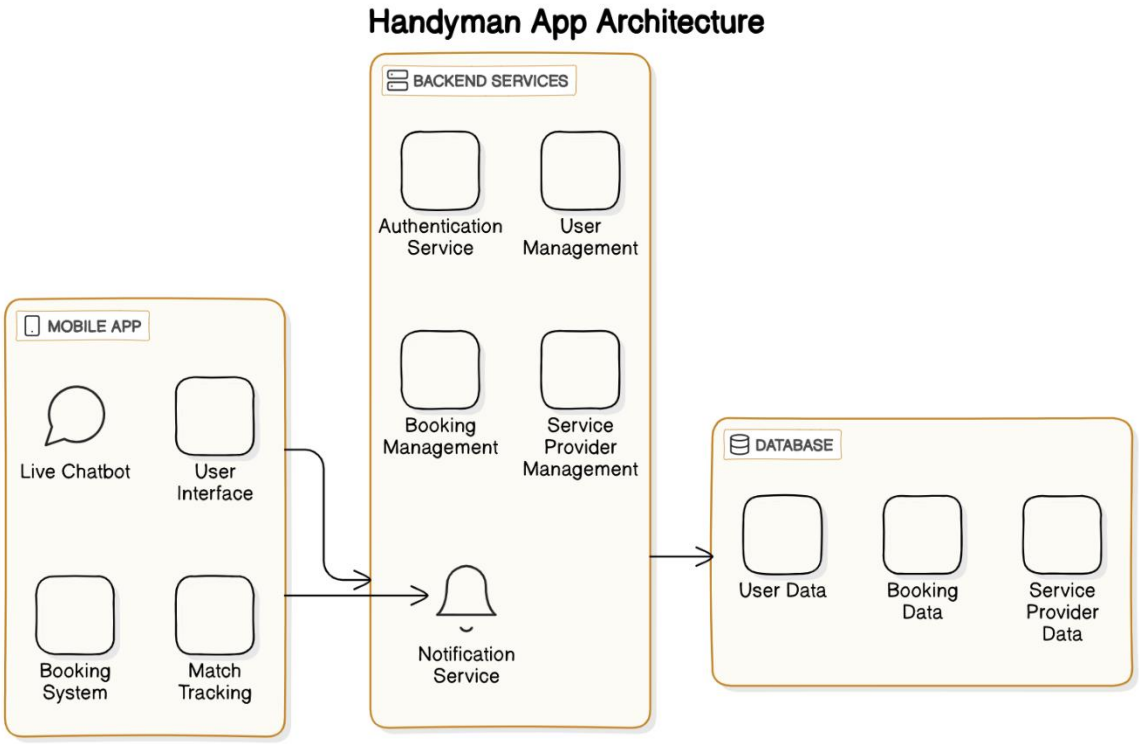

Figure 1:High -Level Diagram

2.2 User Classes and Characteristics

1. **Users:**
    - Users seeking handyman services for various home maintenance tasks.
    - Require easy-to-use interfaces for browsing services, booking appointments, and renting tools.
    - Value convenience, reliability, and quality in service provision.
    - May include homeowners, renters, and property managers.

2. **Service Providers:**
    - Skilled professionals offering a range of home maintenance and repair services.
    - Seek a platform to showcase their expertise, manage bookings, and promote their services.
    - Require tools for managing their schedules, communicating with customers, and tracking earnings.
    - Value visibility, credibility, and ease of use in the platform.

3. **Shop Owners (Tool Rental Providers):**
    - Owners or managers of shops offering tools and equipment for rent.
    - Interested in listing their inventory on the platform to attract customers and generate revenue.
    - Require tools for managing their tool listings, availability, and rental transactions.
    - Value visibility, trustworthiness, and seamless integration with their existing rental processes.

4. **Administrators:**
    - Platform managers responsible for overseeing the app's operations and ensuring smooth functioning.
    - Require tools for managing user accounts, resolving disputes, and monitoring platform activities.
    - Value efficiency, data security, and scalability in the platform's management capabilities.
    - May include customer support representatives, content moderators, and business managers.

2.3 Operating Environment

The proposed Handyman App operates within a dynamic and interconnected environment tailored to the home maintenance industry and its stakeholders. The app utilizes advanced technology to streamline interactions and transactions between homeowners, handymen, shop owners, and administrators. The operating environment encompasses mobile platforms, communication networks, and integrations with external systems.

The app is accessible solely through a mobile platform, ensuring convenience and accessibility for users on the go. It relies on a secure and robust infrastructure to maintain data security and user privacy. Additionally, the app interfaces with external applications, such as payment gateways and messaging platforms, to facilitate real-time information exchange and seamless integration with existing tools and services.

• Mobile Application Development:
We'll use Flutter for cross-platform mobile app development.

• User Interface Design:
Figma and Adobe Photoshop for UI wireframing.

• Version Control:
GitHub for collaborative code management.

• Database Management:
Firebase Firestore for real-time data sync.

• Deployment:
App stores (Google Play Store, Apple App Store) for distribution.

2.4 Design and Implementation Constraints

1. Data Privacy and Security:

   • Ensure compliance with data privacy regulations and implement robust security measures to protect user information.

   • Accessibility: Ensure the app is accessible to users with disabilities by adhering to accessibility standards and guidelines.

2. Real-time Functionality Requirements:

   • The app should be capable of real-time updates and responses to user actions, ensuring seamless user experience.

   • Integration with external systems or APIs should provide real-time data exchange for accurate information retrieval and processing

3. Resource Optimization:

   • The app should optimize memory usage to ensure efficient performance, especially on devices with limited resources.

   • This optimization is essential for smooth operation, particularly when handling large datasets or concurrent tasks.

4. Platform Compatibility:

   • The mobile application should be compatible with various devices and operating systems, ensuring seamless performance across different smartphones and tablets.

   • Compatibility testing should verify the app's functionality on different screen sizes, resolutions, and hardware configurations.

5. Battery Optimization:

   • The mobile application should optimize battery usage to prolong device runtime and ensure uninterrupted usage.

   • Implementing power-saving features and minimizing background processes can help conserve battery life, enhancing the user experience.

6. Scalability:

   • The app's backend infrastructure should be scalable to accommodate potential growth and increased user activity.

   • Ensure the architecture can handle a growing user base, additional features, and increased data volume without compromising performance or reliability.

7. App Maintenance and Support:

   • Ensure the mobile application receives regular updates and maintenance support to address bugs, security vulnerabilities, and compatibility issues.

   • Availability of technical support and timely responses to user inquiries and issues are crucial for ensuring a positive user experience and long-term satisfaction.

8. Security Measures:

   • Implement robust security measures within the mobile application to safeguard user data against unauthorized access or breaches.

   • Utilize encryption, secure authentication methods, and data encryption techniques to ensure the confidentiality and integrity of user information stored within the app.

9. Scalability and Performance

   • The mobile application must be designed to handle increasing user activity and data transactions without compromising performance. This necessitates employing scalable architecture and efficient database management techniques. Additionally, implementing caching mechanisms can optimize response times and reduce server load, ensuring a seamless user experience even during peak usage periods

10. User Interface (UI) Consistency and Usability

    - To deliver a seamless user experience, the mobile application should feature a consistent and intuitive user interface. Adhering to design principles and guidelines, such as those outlined in Material Design for Flutter, will ensure UI consistency and enhance usability across different screens and functionalities

11. Platform Compatibility:

• The app should be developed to be compatible with both Android and iOS mobile platforms, ensuring accessibility to a wide range of users. Utilizing cross-platform frameworks like Flutter will enable consistent functionality and user experience across different devices and operating systems, streamlining development and maintenance efforts

2.5 Assumption and Dependencies

- Internet Connectivity

  The Handyman App assumes that users, including service seekers, handymen, and shop owners, will have access to a reliable internet connection to interact with the app. Features such as real-time updates, live chatbot assistance, and online transactions rely on a stable internet connection for seamless functionality.

- User Device Compatibility

  The Handyman App assumes that users will possess compatible mobile devices, such as smartphones or tablets, capable of running the app. The application will be designed to support various devices, operating systems (iOS and Android), and screen sizes to ensure accessibility and usability for all users.

- User Authentication and Authorization

  The Handyman App assumes that users will authenticate themselves using their phone numbers. Upon entering their phone numbers, users will receive a verification code via SMS to their registered phone numbers. Once the verification code is entered correctly, users will be granted access to the app. User access rights and permissions will be managed based on their verified phone numbers to ensure proper authorization and data security.

3 External Interface Requirements

3.1 User Interfaces

Launch Page

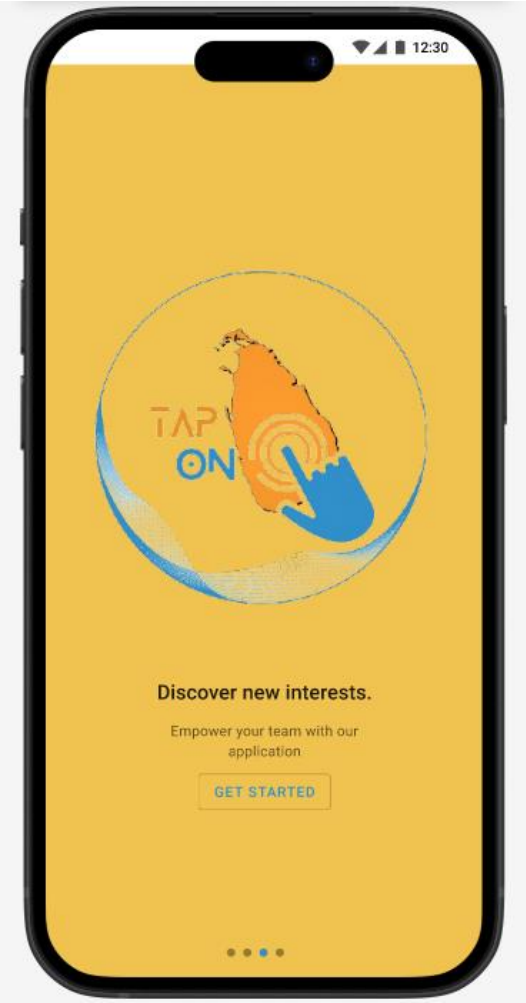

Figure 3: Launch page interface

Language Page

Figure 2: Language page interface

login Page

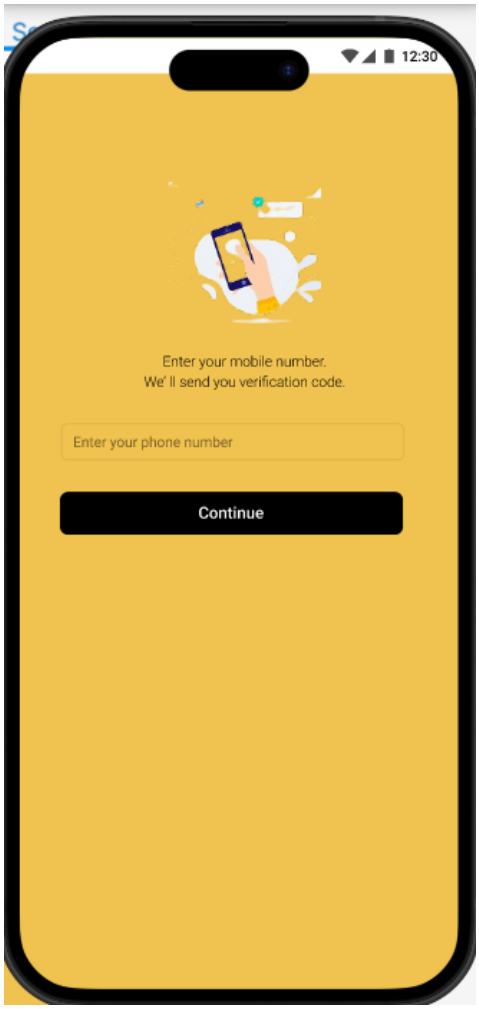

Figure 5: Login page interface

Verification Page

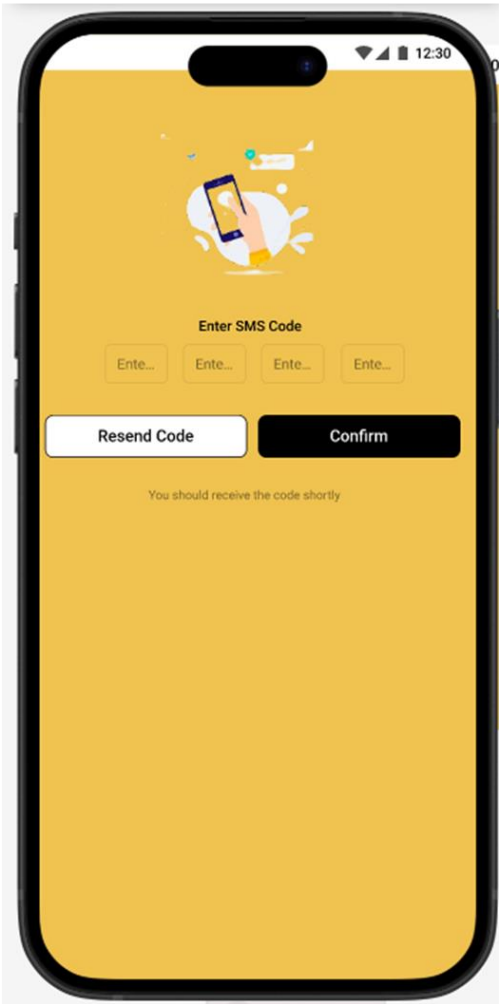

Figure 4: Verification page interface

Location Page

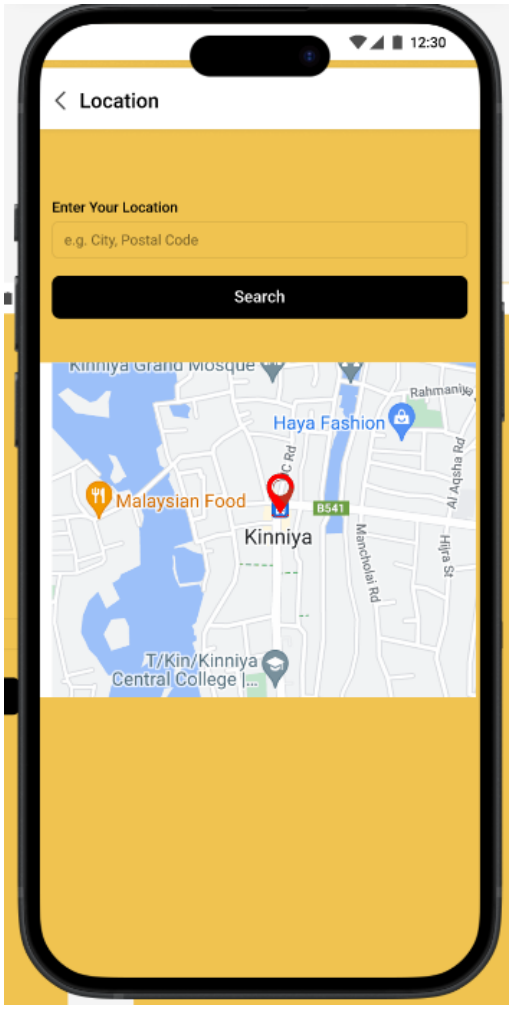

Figure 7: Location page interface

Home Page

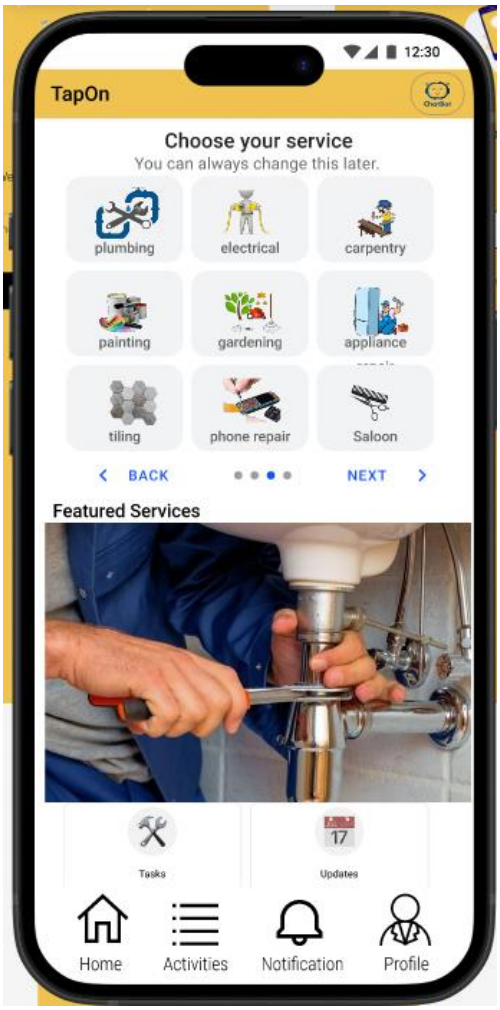

Figure 6: Home page interface

Quick Matching Page

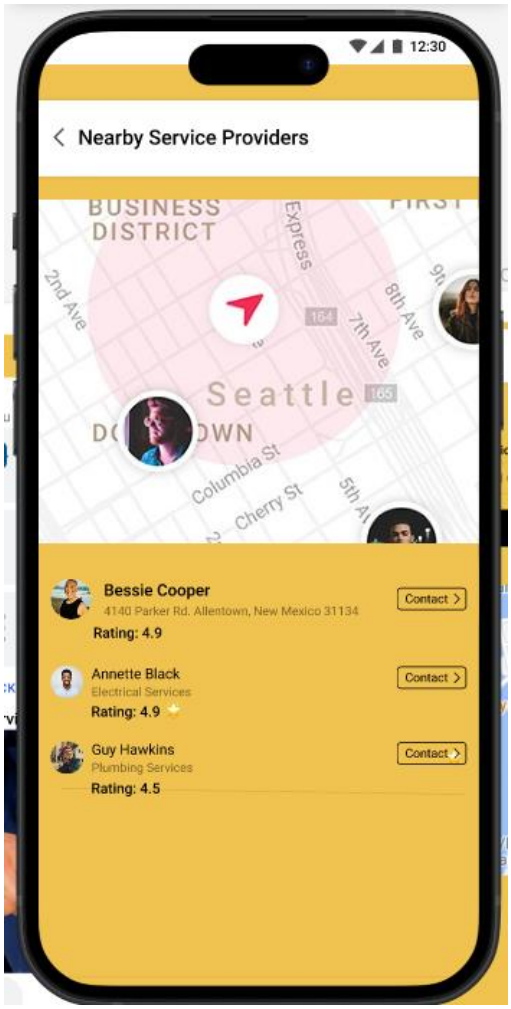

Figure 9: Quick matching interface

Service provider contact Page

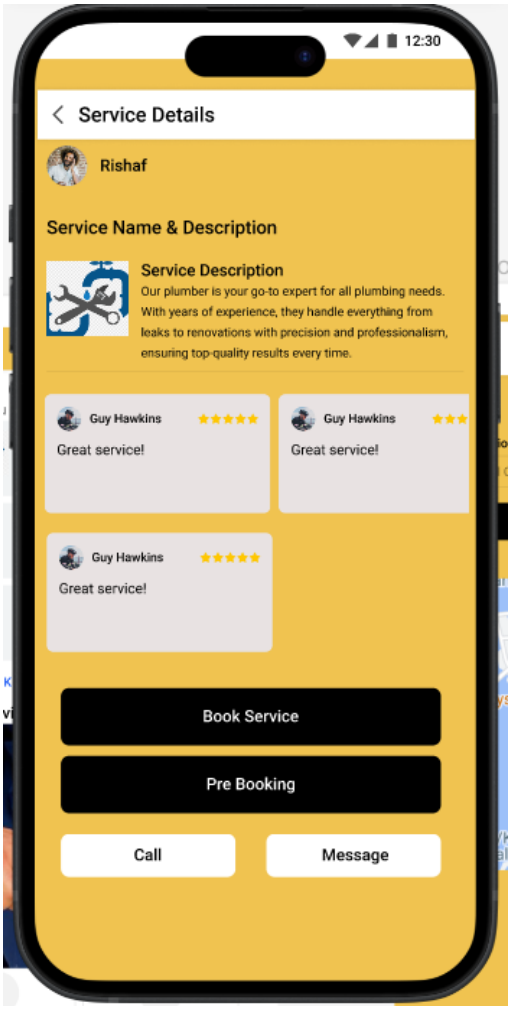

Figure 8: Service provider contact interface

Tracking Page

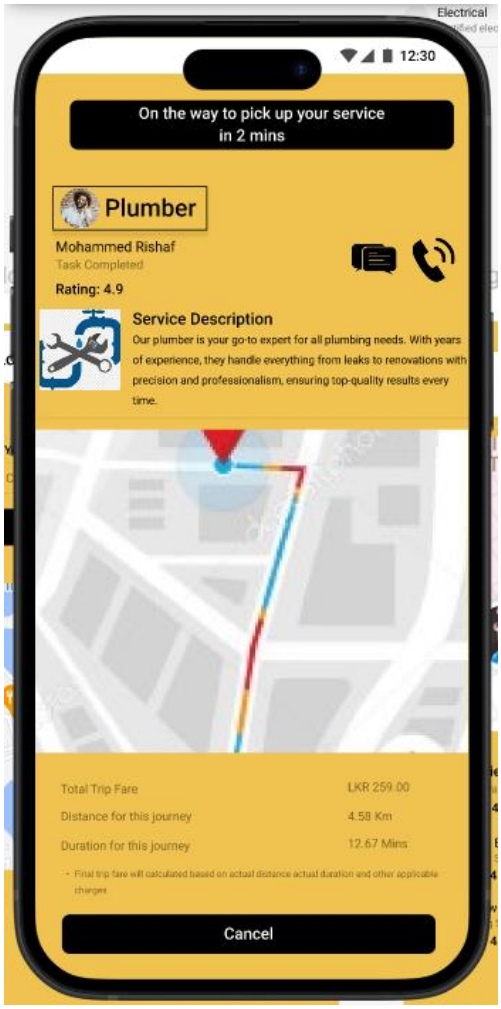

Figure 11: Tracking page interface

Edit profile Page

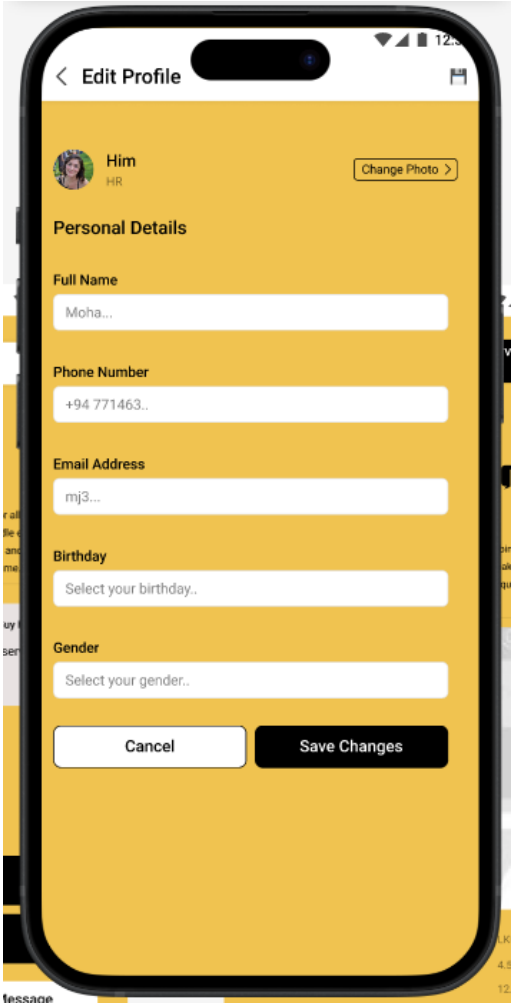

Figure 10: Edit profile interface

Live chatbot Page

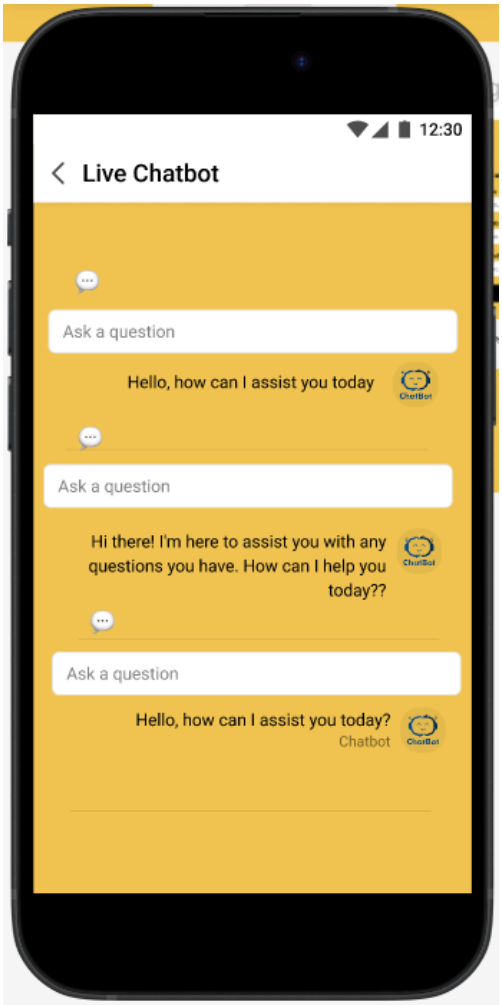

Figure 13: Live chatbot page interface

Pre-booking Page

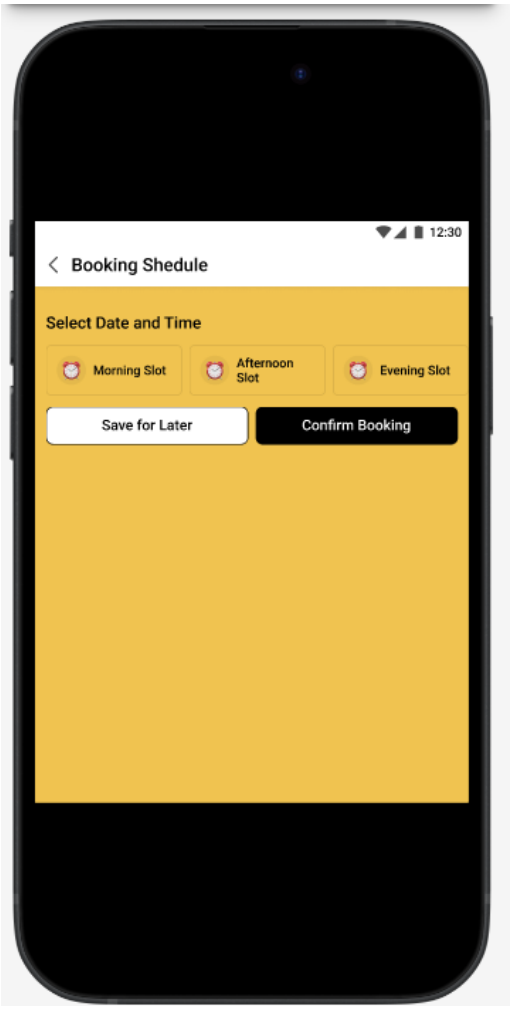

Figure 12: Pre-booking interface

3.2 Hardware interfaces

The Handyman App doesn't directly connect with hardware gadgets. However, it can work with them through links or APIs. It can also use some features on your phone, like GPS for finding your location or the camera for taking pictures. Making sure the app works well on different phones and systems is really important for it to run smoothly and be easy for everyone to use.

1. Services Management:
    - The app interacts with tools used for handyman services, facilitating booking and management. These tools may include scheduling software, inventory trackers, and communication devices.

2. User Devices:
    - The mobile app connects users through their smartphones or tablets. It's compatible with iOS and Android devices, utilizing features like touchscreens, cameras, and network connectivity for smooth operation. Users can book services, manage requests, and receive notifications seamlessly.

3. Cloud Infrastructure:
    - The app relies on cloud servers for hosting and running its services. This includes web servers like Apache HTTP Server or Nginx, which provide computational resources. It also interacts with database servers storing booking and user data. Communication with these servers happens through network infrastructure like routers and firewalls, ensuring reliable connections.

The hardware interfaces outlined above lay the groundwork for the Handyman App. The software's functionality encompasses supporting various communication protocols, ensuring device compatibility, and facilitating seamless data interactions to optimize the management of handyman services. Selecting specific hardware components and communication protocols should align with the app's requirements and objectives, ensuring efficient operation and integration with external systems.

3.3 Software Interfaces

User Interfaces: The Handyman App will provide a user-friendly interface exclusively through mobile applications. Users, including service seekers, handymen, and shop owners, will access and interact with the app's functionalities solely on their smartphones or tablets, ensuring convenience and accessibility.

- Database Management System (DBMS):
  The app interacts with a database management system to store and retrieve data related to user accounts, service bookings, tool rentals, and other relevant information. The DBMS facilitates efficient data management and retrieval, ensuring the app's responsiveness and scalability.

- Authentication and Authorization Services:
  The app integrates with authentication and authorization services to verify user identities and manage access rights. This ensures that only authenticated users can access specific features and data within the app, enhancing security and user privacy.

- Payment Gateway:
  Integration with a payment gateway allows users to make secure online payments for booking services or renting tools. The payment gateway facilitates transactions, ensures financial security, and provides a seamless payment experience for users and service providers.

- Messaging and Notification Services:
  The app interfaces with messaging and notification services to send alerts, updates, and reminders to users. This includes notifications for booking confirmations, service updates, and chatbot responses, enhancing user engagement and communication.

- Geolocation Services:
  Utilizing geolocation services enables the app to determine users' locations and provide relevant information based on their proximity to service providers or tool rental shops. This enhances the user experience by offering personalized recommendations and convenient access to nearby services.

- External APIs and Integrations:

  The app may integrate with external APIs and third-party services to access additional features or data sources. This could include integration with social media platforms for user authentication, map services for location-based features, or analytics services for monitoring app performance and user behavior.

3.4 Communication Interfaces

The Handyman App relies on various communication interfaces to ensure smooth interactions between its components, users, and external systems. These interfaces include functions and protocols for essential features like push notifications, in-app messaging, server communications, and real-time data synchronization. These interfaces play a crucial role in enhancing user experience and ensuring the app's functionality.

1. Mobile App Communication:

- The system utilizes HTTP (Hypertext Transfer Protocol) or its secure variant, HTTPS, for communication between the web application, mobile application, and backend servers. HTTP/HTTPS enables the exchange of data and requests/responses over the internet. It is used for actions such as user authentication, booking submissions, retrieving facility information, and fetching data from the server. The use of HTTPS ensures secure and encrypted communication to protect sensitive user information.

2. Email Notifications:

- To facilitate email notifications, the system integrates with Simple Mail Transfer Protocol (SMTP). SMTP is utilized for sending transactional emails, including booking confirmations, reminders, and updates, to users and administrators. It ensures reliable delivery of emails through appropriate email servers.

3. API Integration:

- The system exposes API endpoints to enable seamless integration with third-party services, such as payment gateways.

4 System Design

4.1 Use case Diagram

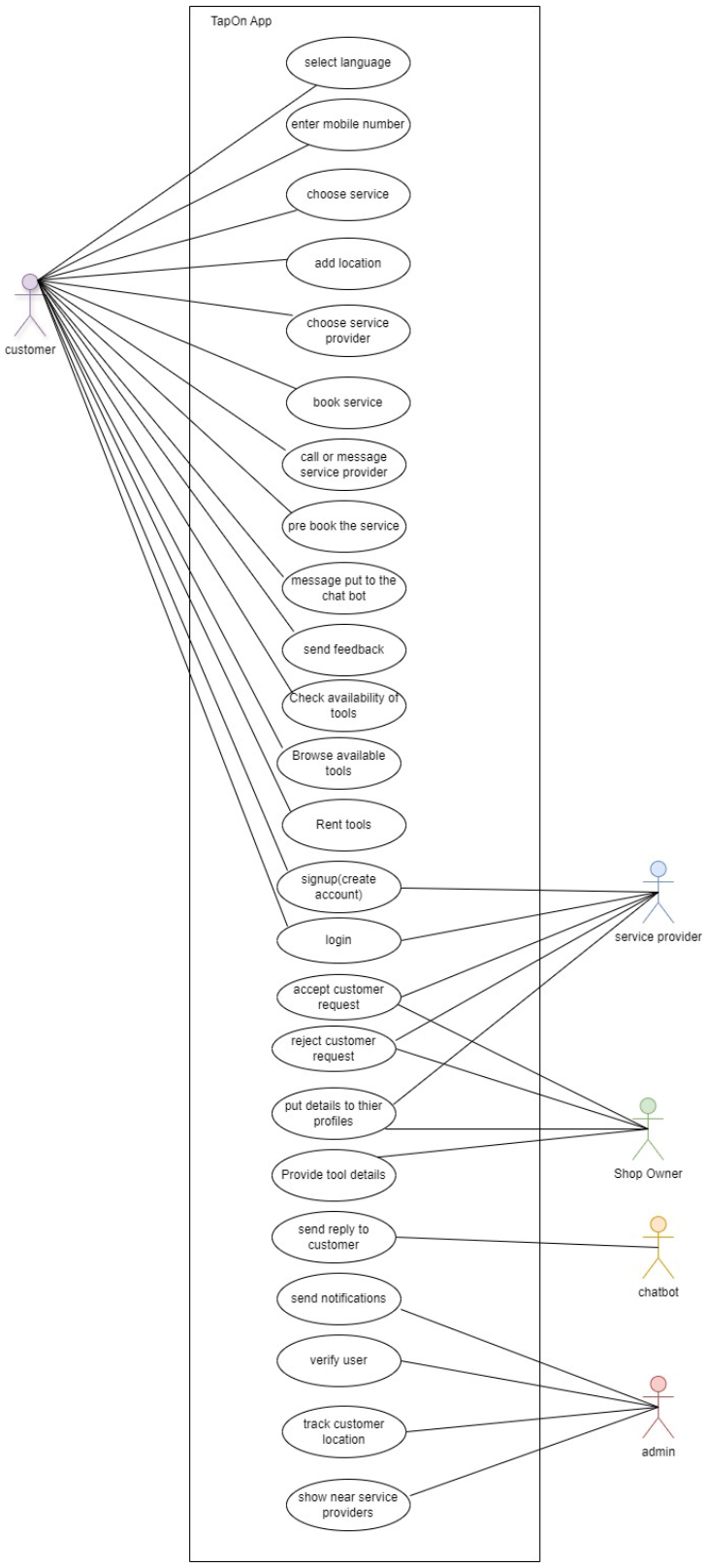

Figure 14: Use case Diagram

4.1.1 Use case Description

Table 1: Use case- Select Language

Use Case ID	UC001	
Use Case	Select Language	
Priority	High	
Primary System Actor	User	
Other participant Actors	None	
Description	This use case enables users to select their preferred language when accessing the Handyman App. Users can choose from available language options to customize their app experience..	
Pre-conditions	The Handyman App is accessible..	
Trigger	The user launches the Handyman App for the first time or accesses the language settings.	
Flow of Events	Actor Action	System Response
	1. User launches the Handyman App. 2. System prompts the user to select their preferred language. 3. User selects the desired language from the available options.	1. System detects the user's launch of the Handyman App. 2. System displays a language selection prompt to the user. 3. System presents a list of available language options to the user. 4. System updates the app's language settings

		according to the user's selection. 5.System confirms the language selection and updates the app's interface accordingly. 6. System provides the user with the app's interface displayed in the selected language. 7. User continues to interact with the app in their chosen language.
Post-conditions	The Handyman App interface is displayed in the user's selected language.	
Alternative Flows	None	
Assumptions	• The Handyman App supports multiple language options. • The language selection feature is properly integrated into the app's user interface. • The user's selected language preference is stored and applied throughout their app sessions.	

Table 2:Use case- Enter Mobile Number

Use Case ID	UC002	
Use Case	Enter Mobile Number	
Priority	High	
Primary System Actor	User	
Other participant Actors	None	
Description	This use case involves the user entering their mobile number to access the Handyman App. It is a crucial step for both new users registering for the app and existing users logging in.	
Pre-conditions	The Handyman App is installed on the user's device. The user has launched the app and is prompted to enter their mobile number.	
Trigger	The user initiates the process of logging in or registering by entering their mobile number	
Flow of Events	Actor Action	System Response
	1. User launches the Handyman App. 2. User is prompted to enter their mobile number. 3. User enters their mobile number. 4. User submits the mobile number. 5. System verifies the mobile number and sends a verification code to the user's mobile device.	1. System detects the user's action of launching the app. 2. System displays the mobile number entry screen. 3. System waits for the user to input their mobile number. 4. System validates the format of the mobile number. 5. System sends a verification code to the provided mobile number. 6. System awaits the user's input of the verification code. 7. System verifies the entered code.

	6. User receives the verification code on their mobile device. 7. User enters the verification code into the app. 8. User submits the verification code.	8. System grants access to the app upon successful verification.
Post-conditions	The user is successfully logged in or registered, and their mobile number is linked to their Handyman App account..	
Alternative Flows	If the user enters an incorrect mobile number, they are prompted to re-enter it. If the verification code is not received, the user can request to resend it.	
Assumptions	The user has a valid mobile number. The mobile device has a stable internet connection to receive the verification code. The user's mobile number is not already associated with another account in the Handyman App. The entered email address is unique and not already associated with another account.	

Table 3:Use case- Choose Service

Use Case ID	UC003	
Use Case	Choose Service	
Priority	High	
Primary System Actor	User	
Other participant Actors	None	
Description	This use case involves the user selecting a service from the Handyman App to address their specific needs.	
Pre-conditions	The user has launched the Handyman App and is logged into their account.	
Trigger	The user wants to find and select a service from the available options.	
Flow of Events	Actor Action	System Response
	1. The User accesses the Handyman App. 2. The System presents the main interface displaying available services. 3. The User selects the option to browse services. 4. The User browses through the list of available services. 5. The User selects a specific service category (e.g., plumbing,	1. System displays the main interface with available service options. 2. System presents the user with the option to browse services. 3. System retrieves and displays the list of available service categories. 4. System displays the list of services within the selected category. 5. System provides the user with the option to select a specific service.

	electrical, carpentry, etc.). 6. The User views the list of services within the chosen category. 7. The User selects a particular service they require assistance with.	6. System registers the user's selection and proceeds to the next step. 7. System detects the user's action of launching the app.
Post-conditions	The user has successfully chosen a service from the Handyman App, and the system proceeds to the next step in the service booking process.	
Alternative Flows	• If the user does not find the desired service category, they can go back to the previous screen or search for specific services using the search function.	
Assumptions	• The Handyman App has a stable internet connection for real-time browsing and selection of services. • The list of available services is regularly updated and maintained by the system administrators. • Users have the necessary permissions and credentials to access and browse services within the app. • The user interface of the Handyman App is intuitive and user-friendly, facilitating easy navigation and selection of services. • Service categories and descriptions are accurately presented, allowing users to make informed decisions when choosing a service. • The system can handle a large variety of service categories and options to cater to the diverse needs of users. • Users can easily backtrack or modify their selection if they change their mind during the service browsing process.	

	• The Handyman App provides clear instructions and guidance to users throughout the service selection process, ensuring a smooth user experience.

Table 4:Use case- book service

Use Case ID	UC004	
Use Case	book service	
Priority	High	
Primary System Actor	User	
Other participant Actors	Service Provider, Admin	
Description	This use case involves the user booking a service through the Handyman App. The user can browse available services, select a service provider, specify the booking details, and confirm the booking.	
Pre-conditions	The Handyman App is installed on the user's device. The user has successfully logged in to the app.	
Trigger	The user decides to book a service through the Handyman App.	
Flow of Events	Actor Action	System Response
	1. User accesses the Handyman App. 2. User navigates to the service booking section. 3. User browses available services or uses search filters to find a specific service. 4. User selects a service from the list. 5. User views details of the selected service, including service provider information, availability, and pricing.	1. System detects user's action of accessing the app. 2. System displays the service booking section. 3. System retrieves available services from the database and presents them to the user. 4. System waits for the user to select a service.

	6. User selects a service provider for the booking. 7. User specifies booking details such as date, time, and any additional requirements. 8. User confirms the booking request.	5. System retrieves detailed information about the selected service and displays it to the user. 6. System retrieves available service providers for the selected service and presents them to the user. 7. System awaits the user's input of booking details. 8. System processes the booking request and sends a confirmation to the user.
Post-conditions	The booking request is successfully submitted, and the user receives a confirmation of the booking. The service provider is notified of the booking request.	
Alternative Flows	If the user cancels the booking process at any step, the system returns to the previous screen or provides an option to cancel the booking. If the selected service provider is unavailable, the system notifies the user and suggests alternative providers.	
Assumptions	1. The Handyman App assumes that users have access to a stable internet connection to browse services, make bookings, and communicate with service providers. 2. It is assumed that service providers have properly registered and verified their profiles on the Handyman App before offering services to users.	

	3. The app assumes that service providers are available during the specified booking times unless otherwise indicated. 4. The Handyman App assumes that users will provide accurate information when booking services, including date, time, and specific requirements. 5. It is assumed that service providers will respond promptly to booking requests and communicate any changes or updates to users in a timely manner. 6. The app assumes that users will comply with the terms and conditions outlined by the service providers and the Handyman App platform. 7. It is assumed that the Handyman App platform will maintain the privacy and security of user data in accordance with relevant regulations and policies. 8. The app assumes that users will have compatible devices and operating systems to access and use the Handyman App effectively. 9. It is assumed that users will have basic knowledge of how to navigate and interact with mobile applications. 10. The Handyman App assumes that administrators will monitor and manage the platform to ensure smooth operation and resolve any issues that may arise.

Table 5:Use case- Send Feedback

Use Case ID	UC005	
Use Case	Send Feedback	
Priority	Medium	
Primary System Actor	User	
Other participant Actors	None	
Description	The Send Feedback feature allows users to provide feedback on their experience with the Handyman App, including suggestions for improvement or reporting issues.	
Pre-conditions	The user is logged into the Handyman App..	
Trigger	The user wants to provide feedback on the Handyman App.	
Flow of Events	Actor Action	System Response
	1. User accesses the Handyman App. 2. System presents the main menu options to the user. 3. User selects the "Send Feedback" option. 4. System displays a feedback form to the user. 5. User fills out the feedback form, including their comments, suggestions, or issues. 6. User submits the feedback form.	1. System acknowledges the successful submission of feedback. 2. User can continue using other features of the Handyman App.

Post-conditions	The user's feedback is recorded and may be reviewed by the Handyman App administrators for further action.
Alternative Flows	None
Assumptions	• The Handyman App provides a user-friendly feedback form for users to submit their feedback. • The system securely stores and manages user feedback data for analysis and review by administrators. • Users have the necessary permissions and credentials to access the "Send Feedback" feature within the Handyman App.

Table 6:Use case- Rent Tools

Use Case ID	UC006	
Use Case	Rent Tools	
Priority	High	
Primary System Actor	Customer	
Other participant Actors	Shop Owner	
Description	This use case enables customers to rent tools from shop owners through the Handyman App. Customers can browse available tools, select the desired items, specify rental duration, and confirm the booking.	
Pre-conditions	The Handyman App is installed and accessible on the customer's mobile device. The customer has a registered account on the Handyman App. Shop owners have listed their tools for rent on the Handyman App.	
Trigger	The customer wants to rent tools for a specific task or project.	
Flow of Events	Actor Action	System Response
	1. The customer opens the Handyman App on their mobile device. 2. The system displays the home screen with options to browse services. 3. The customer selects the "Rent Tools" option from the menu.	1. The system loads the home screen interface with service options. 2. The system presents the "Rent Tools" interface with tool categories. 3. The system retrieves and displays the list of available tools for rent.

	4. The system presents a list of available tools categorized by type. 5. The customer browses through the list and selects the desired tool(s). 6. The system prompts the customer to specify the rental duration (e.g., number of days). 7. The customer enters the rental duration and confirms the booking. 8. The system sends the booking request to the respective shop owner(s). 9. The shop owner(s) receive the booking request and confirm availability. 10. The system notifies the customer of the booking confirmation and provides details for pickup or delivery. 11. The customer can view the booking details and make payment if required. 12. The system updates the booking status and rental	4. The system highlights the selected tool(s) for rental. 5. The system prompts the customer to enter the rental duration. 6. The system validates the rental duration input and confirms the booking. 7. The system processes the booking request and sends notifications to the shop owner(s). 8. The system updates the booking status and notifies the customer of the pending confirmation. 9. The system delivers the booking request to the shop owner(s) via the app notification. 10. The system sends a confirmation notification to the customer with pickup/delivery details. 11. The system redirects the customer to the payment interface if payment is required. 12. The system updates the booking status and rental details in the customer's account.

	details in the customer's account. 13. The customer can continue using other features of the Handyman App.	13. The system maintains the customer's session and allows access to other app features.
Post-conditions	The customer has successfully booked the desired tools for the specified rental duration.	
Alternative Flows	None	
Assumptions	• Shop owners have listed their tools for rent on the Handyman App. • The Handyman App has a stable internet connection for real-time communication between customers and shop owners. • Customers have verified accounts and payment methods registered on the Handyman App. • Shop owners promptly confirm booking requests and communicate any changes or updates to customers. • Customers will comply with the terms and conditions outlined by the shop owners and the Handyman App platform.	

Table 7:Use case- Login/Signup

Use Case ID	UC007	
Use Case	Login/Signup	
Priority	High	
Primary System Actor	Service Provider	
Other participant Actors	None	
Description	The Login/Signup use case enables users to access the Handyman App by either logging in with their existing credentials or signing up for a new account. Users can access various features and functionalities of the app after successful authentication.	
Pre-conditions	The user has downloaded and installed the Handyman App on their mobile device. The user has a stable internet connection.	
Trigger	The user wants to access the Handyman App to browse services, make bookings, or manage their account.	
Flow of Events	Actor Action	System Response
	1.User Launches the App: The user launches the Handyman App on their mobile device. 2.Selects Login/Signup: The user selects the login/signup option from the app interface. 3.Enters Mobile Number: If the user already has an account, they enter their registered mobile number. If not, they enter their mobile number to sign up. 4.Receives Verification Code: The system sends a verification code	1. Login/Signup Interface: The Handyman App presents the login/signup interface to the user. 2. Verification Code Sent: The system sends a verification code to the user's mobile number for authentication. 3. Validation: The system validates the entered verification code and

	to the user's mobile number for authentication. 5. Enters Verification Code: The user enters the received verification code in the app. 6. Provides Additional Information: If signing up, the user provides additional information such as name, email (optional), and creates a password. 7. Validation: The system validates the entered information and verification code. 8. Access Granted: Upon successful validation, the system grants access to the user's account or creates a new account. 9. Main Interface: The user is directed to the main interface of the Handyman App.	additional information (if signing up). 4. Access Granted: Upon successful validation, the system grants access to the user's account or creates a new account. 5. Main Interface Displayed: The Handyman App displays the main interface with access to all features and functionalities.
Post-conditions	The user is logged in to the Handyman App and can access all available features and functionalities.	
Alternative Flows	None	
Assumptions	• Users have a valid mobile number for receiving verification codes. • Users have access to their mobile devices and are capable of receiving and entering verification codes. • The Handyman App has permission to send SMS messages for verification purposes.	

	• The mobile device has a stable internet connection for real-time communication with the Handyman App server.

Table 8:Use case- Account Management

Use Case ID	UC008	
Use Case	Account Management	
Priority	High	
Primary System Actor	Service Provider	
Other participant Actors	None	
Description	The Account Management feature allows users to create, access, and manage their accounts on the Handyman App. Users can perform actions such as registration, login, profile management, and password recovery.	
Pre-conditions	The user has access to the Handyman App and a stable internet connection.	
Trigger	The user wants to create, access, or manage their account on the Handyman App.	
Flow of Events	Actor Action	System Response
	1. User Accesses the App: The user launches the Handyman App on their device. 2. System Presents Login/Registration Interface: The system displays the login or registration interface to the user. 3. User Selects Account Management: The user chooses the account management option from the interface. 4. User Selects Registration: If the user opts to register, they select the registration option.	1. Interface Displayed: The system presents the login or registration interface to the user. 2. Options Displayed: The system displays options for login or registration based on the user's selection. 3. Form Displayed: If registration is selected, the system prompts the

	5. User Enters Personal Details: The user provides their personal details such as name, email, and phone number. 6. User Submits Registration Form: Upon completing the form, the user submits the registration details. 7. Verification Code Sent: The system sends a verification code to the user's provided phone number or email address. 8. User Enters Verification Code: The user enters the verification code received. 9. User Selects Login: If the user chooses to login instead, they select the login option. 10. User Enters Login Credentials: The user inputs their login credentials, typically a phone number and verification code. 11. User Selects Profile Management: Upon successful login, the user selects the profile management option. 12. User Updates Profile: If needed, the user can update their profile information. 13. User Changes Password: Alternatively, the user may choose to change their password.	user to enter their personal details. 4. Form Submitted: Upon form submission, the system validates the provided information and sends a verification code. 5. Verification Code Sent: The system successfully sends the verification code to the user. 6. Verification Code Validated: The system verifies the entered code and grants access upon successful validation. 7. Profile Management Options Displayed: The system presents options for profile management upon successful login. 8. Profile Updated: After the user updates their profile, the system validates the changes and updates the profile accordingly. 9. Password Changed: If the user changes their

	14. User Initiates Password Recovery: The user selects the password recovery option if they've forgotten their password. 15. Password Reset Link/Code Sent: The system sends a password reset link or code to the user's provided contact information. 16. User Follows Password Reset Instructions: The user follows the instructions to reset their password using the link or code. 17. Account Management Actions Completed: Once the user completes their desired account management actions, they can continue using the app.	password, the system verifies the credentials and updates the password. 10. Password Reset Instructions Sent: The system successfully sends instructions for password reset to the user. 11. Password Reset Verified: Upon following the reset instructions, the system verifies the reset link or code provided by the user. 12. Actions Confirmed: The system confirms the successful completion of the requested account management actions.
Post-conditions	• The user's account is created, accessed, or updated based on the performed actions.	
Alternative Flows	None	
Assumptions	• The Handyman App supports user registration, login, and account management functionalities. • The app has a secure authentication mechanism to protect user accounts from unauthorized access.	

	• Users have access to valid contact information (phone number or email) for receiving verification codes and password reset instructions. • The app's servers are operational and accessible to handle user registration, login, and account management requests.

Table 9:Use case- Provide Tool Details

Use Case ID	UC009	
Use Case	Provide Tool Details	
Priority	High	
Primary System Actor	Shop Owner (Tool Rental Provider)	
Other participant Actors	None	
Description	This use case allows Shop Owners to provide details of tools available for rent on the Handyman App. Shop Owners can list various tool categories, along with specific tool types, descriptions, pricing, and availability, to attract potential renters.	
Pre-conditions	The Shop Owner has registered and verified their profile on the Handyman App. The Shop Owner has access to the tool management interface within the Handyman App.	
Trigger	The Shop Owner wants to add or update details of tools available for rent	
Flow of Events	Actor Action	System Response
	1. The Shop Owner accesses the tool management section of the Handyman App. 2. The Shop Owner selects the option to provide tool details. 3. The Shop Owner enters the necessary information for the tool, including category, type, description, pricing, and availability.	1. The system presents the tool management interface to the Shop Owner. 2. The system validates the provided information. 3. If the information is valid, the system updates the tool listing with the provided details.

	4. The Shop Owner submits the entered information for validation.	4. The system sends a confirmation message to the Shop Owner regarding the successful addition or update of tool details. 5. If any validation errors occur, the system prompts the Shop Owner to correct the information and resubmit. 6. The Shop Owner can continue managing other tools or exit the tool management interface.
Post-conditions	The details of the tools are updated or added to the Handyman App, making them available for users to view and rent.	
Alternative Flows	None	
Assumptions	• The Handyman App platform supports the management of tool listings by Shop Owners. • Shop Owners have access to a stable internet connection to update tool details in real-time. • The Shop Owner's credentials and permissions allow them to manage tool listings within the Handyman App.	

Table 10:Use case- verify user

Use Case ID	UC010	
Use Case	verify user	
Priority	High	
Primary System Actor	Shop Owner (Tool Rental Provider)	
Other participant Actors	Administrator	
Description	This use case involves the verification of a user's identity when accessing the Handyman App. The user needs to confirm their identity to ensure secure access to the platform and prevent unauthorized usage.	
Pre-conditions	The user has installed the Handyman App on their mobile device. The user has provided their mobile phone number during the registration process. The system has stored the user's mobile phone number securely in the database.	
Trigger	The user attempts to log in to the Handyman App.	
Flow of Events	Actor Action	System Response
	1. User Accesses App: The user opens the Handyman App on their mobile device. 2. System Presents Login Interface: The system displays the login interface, prompting the user to enter their registered mobile phone number. 3. User Enters Mobile Number: The user inputs their registered mobile	1. Login Interface Displayed: The system presents the login interface, awaiting user input. 2. Verification Code Sent: Upon entering the mobile number, the system sends a verification code to

	phone number into the designated field. 4. System Sends Verification Code: Upon entering the mobile number, the system generates and sends a verification code to the user's phone number via SMS. 5. User Receives Verification Code: The user receives the verification code on their mobile device. 6. User Enters Verification Code: The user enters the received verification code into the app. 7. System Verifies Code: The system verifies the entered code against the one sent to the user's mobile number. 8. System Grants Access: If the verification code matches, the system grants access to the user's account. 9. User Accesses App Features: Upon successful verification, the user gains access to the features of the Handyman App and can proceed to utilize its functionalities.	the provided phone number. 3. Verification Code Received: The user receives the verification code on their mobile device. 4. Code Verification: The system verifies the entered code against the one sent to the user's mobile number. 5. Access Granted: If the verification code matches, the system grants access to the user's account and allows them to proceed. 6. Access Denied: If the verification code does not match or expires, the system denies access and prompts the user to retry or request a new verification code. 7. User Access Confirmed: Upon successful verification, the system confirms the

		user's access and enables them to utilize the app's features.
Post-conditions	The user successfully verifies their identity and gains access to the Handyman App.	
Alternative Flows	None	
Assumptions	• The user's mobile phone number is accurately registered during the sign-up process. • The SMS service used for sending verification codes is reliable and delivers messages promptly. • The user has access to their registered mobile phone to receive the verification code. • The system securely stores and handles user verification data to prevent unauthorized access.	

Table 11:Use case- Send Notifications

Use Case ID	UC010	
Use Case	Send Notifications	
Priority	High	
Primary System Actor	Administrator	
Other participant Actors	None	
Description	The Send Notifications feature enables administrators to send notifications to users to keep them updated on the status of their service requests, appointment reminders, and promotions.	
Pre-conditions	The administrator has the necessary credentials to access the system. The system has users registered with valid contact information to receive notifications.	
Trigger	The administrator needs to send notifications to users for various purposes.	
Flow of Events	Actor Action	System Response
	10. Accesses the mobile application. 11. Selects the "Send Notifications" option. 12. Fills in the notification details, including recipient(s), message content, and scheduling options. 13. Confirms the notification details and triggers the sending process.	1. Presents the administrator with the dashboard or notification management interface upon login. 2. Displays the notification composition form upon selecting the "Send Notifications" option.

		3. Validates the input and prepares the notification for sending. 4. Sends the notifications to the designated recipients upon confirmation by the administrator. 5. Confirms the successful delivery of the notifications and updates the notification status
Post-conditions	The designated users receive the notifications according to the administrator's instructions.	
Alternative Flows	None	
Assumptions	• The system has a robust notification delivery mechanism capable of reaching users through various channels (e.g., push notifications, emails, SMS). • Users have opted in to receive notifications and have provided valid contact information during registration. • The administrator has the authority to send notifications and access the notification management feature. • The system maintains user preferences and respects privacy settings regarding notification preferences.	

Table 12:Use case- show near service providers

Use Case ID	UC010	
Use Case	show near service providers	
Priority	High	
Primary System Actor	Administrator	
Other participant Actors	Service Providers, Location Tracking System	
Description	This feature enables users to view a list of service providers near their current location. The system utilizes the location tracking functionality to identify nearby service providers based on the user's location preferences..	
Pre-conditions	The user has the Handyman App installed on their mobile device. The user has granted the app permission to access their location.	
Trigger	The user wants to find service providers near their current location to book a service.	
Flow of Events	Actor Action	System Response
	14. The user opens the Handyman App on their mobile device. 15. The system prompts the user to enable location services if not already enabled. 16. The user selects the "Find Service Providers Near Me" option from the app's main menu. 17. The system accesses the user's current location using the device's GPS or location services. 18. The system retrieves a list of nearby service providers based on the user's location.	1. The system displays a prompt requesting the user to enable location services if not already enabled. 2. The system presents the user with a list of nearby service providers based on their current location. 3. The system retrieves and displays detailed information about the

	19. The user reviews the list of nearby service providers displayed on the app. 20. The user selects a service provider from the list to view detailed information. 21. The system displays the selected service provider's profile, including their services offered, ratings, reviews, and contact information. 22. The user can choose to book the service provider directly from their profile or contact them for more information. 23. The system updates the user's search results based on any changes in their location or preferences.	selected service provider. 4. The system updates the user's search results based on any changes in their location or preferences.
Post-conditions	The user has successfully viewed a list of service providers near their current location. The user can proceed to book a service or contact a service provider based on their preferences.	
Alternative Flows	If the user denies access to their location, the system prompts the user to manually enter their location or provide a zip code to search for service providers in the specified area. If there are no service providers available near the user's location, the system notifies the user and suggests expanding their search radius or trying again later.	

Assumptions	• The Handyman App has access to the user's device location through GPS or location services. • Service providers have provided accurate location information in their profiles. • The location tracking system used by the app is accurate and reliable for identifying nearby service providers. • The user's mobile device has a stable internet connection to retrieve location-based information in real-time.The system maintains user preferences and respects privacy settings regarding notification preferences.

5. System Features

5.1 Tool Rental Listing and Management

5.1.1 Description and Priority

The Tool Rental Listing and Management feature enables seamless listing and efficient management of tools available for rent by Shop Owners. It holds a high priority as it ensures a smooth rental process and enhances user satisfaction by providing access to a variety of tools for service seekers.

5.1.2 Stimulus/Response Sequences

- Tool Listing Creation:
  Stimulus: Shop Owner initiates the process to list a new tool for rent.

  Response: The app prompts the Shop Owner to input details such as tool type, description, rental price, and availability. After submission, the tool listing is created and added to the inventory.

- Tool Availability Update:
  Stimulus: Shop Owner updates the availability status of a tool (e.g., tool rented out, tool returned).

  Response: The app's system detects the update in tool availability and reflects the changes in real-time, ensuring that users browsing for tools see accurate availability status.

- Tool Information Modification:
  Stimulus: Shop Owner edits the details of an existing tool listing (e.g., updating description, changing rental price).

  Response: The app allows the Shop Owner to modify the tool information as needed. Upon submission, the changes are updated in the tool listing.

- Tool Removal:
  Stimulus: Shop Owner removes a tool listing from the inventory (e.g., tool no longer available for rent).

  Response: The app prompts the Shop Owner to confirm the removal action. Upon confirmation, the tool listing is deleted from the inventory.

- Rental Request Notification:
  Stimulus: Service Seeker sends a rental request for a tool.

  Response: The app notifies the Shop Owner of the incoming rental request, providing details of the requested tool and rental duration.

- Rental Request Approval/Rejection:
  Stimulus: Shop Owner reviews a rental request and makes a decision (approve or reject).

  Response: The app updates the status of the rental request accordingly. If approved, the tool availability is adjusted, and the Service Seeker is notified. If rejected, the Service Seeker is informed of the rejection.

- Communication with Service Seeker:
  Stimulus: Shop Owner initiates communication with the Service Seeker regarding the rental request (e.g., clarification on rental terms, arranging pickup/drop-off).
  Response: The app provides a platform for seamless communication between the Shop Owner and the Service Seeker, facilitating discussions and arrangements related to the tool rental.

- Mobile-based Tool Management:

  Stimulus: User accesses the mobile application for tool management.

  Response: The app presents the user with options to manage tools, including listing new tools, updating tool availability, or removing outdated listings. Users can easily perform these actions using the intuitive interface provided by the mobile application, enhancing convenience and efficiency in tool management.

5.1.3 Functional Requirements

REQ-1: The system shall provide a user interface for Shop Owners to list tools for rent and manage tool inventory.

REQ-2: The system shall enable communication between the app and Shop Owners to send tool availability updates and receive rental requests.

REQ-3: The system shall update tool availability dynamically based on rental status and Shop Owners' actions.

REQ-4: Shop Owners shall be able to manually update tool availability and modify tool listings via the mobile application.

REQ-5: The system shall provide real-time feedback on tool availability status and any changes made by Shop Owners.

REQ-6: The system shall handle and display error messages for any communication failures or inventory management issues.

5.2 live chatbot

5.2.1 Description and Priority

The live chatbot feature provides users with real-time assistance and support through an interactive chat interface. It is of high priority as it enhances user engagement and satisfaction by offering immediate responses to queries, troubleshooting assistance, and guidance on tool rental processes. The live chatbot serves as a vital tool for providing seamless customer support, resolving issues promptly, and ensuring a positive user experience within the app ecosystem.

5.2.2 Stimulus/Response Sequences

- Initiate Chatbot Interaction:

  Stimulus: User initiates interaction with the chatbot by accessing the chat interface.

  Response: The chatbot greets the user and prompts them to specify their query or request.

- Select Inquiry Type:

  Stimulus: User selects a specific category or type of inquiry from the available options (e.g., general information, troubleshooting, booking assistance).

  Response: The chatbot displays options related to the selected inquiry type and awaits further input from the user.

- Provide Query Details:

  Stimulus: User provides details of their query or request within the chosen inquiry type.

  Response: The chatbot acknowledges the provided information and processes the query to generate a relevant response.

- Retrieve Information:

  Stimulus: Chatbot processes the user's query and retrieves relevant information or assistance.

  Response: The chatbot presents the user with the requested information, troubleshooting steps, booking options, or other assistance as applicable.

- Engage in Conversation:

  Stimulus: User engages in conversation with the chatbot, asking follow-up questions or seeking further clarification.

  Response: The chatbot responds to the user's inquiries in a conversational manner, providing additional details or guidance as needed.

- Offer Assistance Options:

  Stimulus: User expresses the need for specific assistance options (e.g., speak to a human agent, receive a callback).

  Response: The chatbot presents the user with available assistance options, such as transferring to a human agent or initiating a callback request.

- Conclude Interaction:

  Stimulus: User indicates the conclusion of the interaction with the chatbot.

  Response: The chatbot acknowledges the end of the conversation and offers further assistance if needed, or gracefully exits the interaction.

5.2.3 Functional Requirements

REQ-1: Chatbot Integration

- The app shall integrate a live chatbot functionality, allowing users to interact with the chatbot for assistance, inquiries, or support.

REQ-2: User Interaction

- Users shall be able to initiate conversations with the chatbot by accessing the chat interface within the app.

REQ-3: Inquiry Handling

- The chatbot shall handle user inquiries regarding tool availability, service bookings, rental pricing, and other related queries.

REQ-4: Service Provider Information

- The chatbot shall provide users with information about available service providers, including their profiles, offered services, and availability.

REQ-5: Tool Rental Information

- Users shall receive information from the chatbot regarding available tools for rent, including tool types, descriptions, rental prices, and availability.

REQ-6: Reservation Assistance

- The chatbot shall assist users in reserving specific tools or services by guiding them through the reservation process.

REQ-7: Error Handling

- If there is an error during the interaction with the chatbot, such as invalid input or system issues, the chatbot shall provide appropriate error messages to the user.

REQ-8: User Authentication

- Users shall be authenticated before accessing certain functionalities of the chatbot, ensuring security and personalized assistance.

REQ-9: Integration with Customer Support

- The chatbot shall seamlessly integrate with customer support systems, allowing users to escalate inquiries or issues to human agents if needed.

REQ-10: Natural Language Processing

- The chatbot shall utilize natural language processing techniques to understand and respond to user queries in a conversational manner.

REQ-11: Contextual Awareness

- The chatbot shall maintain contextual awareness during conversations, allowing it to remember previous interactions and provide relevant responses.

These functional requirements ensure the effective implementation and operation of the Live Chatbot feature within your app, enhancing user engagement and support capabilities.

5.3 Pre-Booking tools and services

5.3.1 Description and Priority

The pre-booking feature enables users to browse tool availability, select preferred rental periods, and make reservations in advance. It is of high priority as it constitutes a fundamental functionality of the system, ensuring users can efficiently plan and secure tool rentals according to their needs.

5.3.2 Stimulus/Response Sequences

• Select Category:

Stimulus: User selects a specific category (tools or services) from the available options.

Response: The system displays the selected category's details, including tool types or service types, descriptions, pricing, and availability.

• Select Item:

Stimulus: User selects a specific tool or service from the available options within the chosen category.

Response: The system presents a list of available tools or services, along with their availability status, descriptions, and pricing.

• Select Date/Time:

Stimulus: User chooses a preferred date and time for the booking.

Response: The system confirms the selected date and time, displays the available options for the chosen tool or service, and presents the user with booking choices.

• Select Quantity/Duration:

Stimulus: User specifies the quantity (for tools) or duration (for services) for the booking.

Response: The system confirms the selected quantity or duration, calculates the corresponding booking fee, and presents the user with the total cost.

• Add to Cart:

Stimulus: User adds the selected item to the booking cart.

Response: The system updates the booking cart with the selected item, date, time, quantity, or duration, allowing the user to proceed to checkout.

5.3.3 Functional Requirements

REQ-1: User Registration

- The app shall provide a user registration functionality, allowing users to create an account by providing their personal information, including name, phone number, and password.

REQ-2: User Login

- Registered users shall be able to log in using their phone numbers and passwords to access the pre-booking tools and services.

REQ-3: Tool Rental Listing and Management

- The system shall display a listing of available tools for rent, including tool types, descriptions, rental prices, and availability.

- Users shall be able to view and manage their tool rentals, including adding, updating, or removing listings.

REQ-4: Service Provider Listing and Management

- The app shall provide a listing of available service providers, including their profiles, offered services, and availability.

- Service providers shall be able to manage their profiles, including adding/editing services offered, updating availability, and responding to service requests.

REQ-5: Availability of Time Slots

- The system shall present a schedule of available time slots for each tool/service, indicating their availability for booking.

- Users shall be able to check the available time slots and choose a suitable time for their tool rental or service booking.

REQ-6: Tool/Service Reservation

- Users shall be able to reserve a specific tool/service for their chosen time slot.

- The system shall validate the availability of the tool/service and time slot before confirming the reservation.

REQ-7: Error Handling

- If there is an error during the payment process, such as an invalid payment method or insufficient funds, the system shall display an appropriate error message to the user.

REQ-12: Confirmation and Receipt Generation

- Upon successful payment, the system shall generate a payment confirmation and receipt for the user, providing details such as transaction ID, booking details, payment amount, and timestamp.

These functional requirements ensure the smooth operation of the Pre-Booking Tools and Services feature within your app, providing users with convenient access to tools and services for their needs.

6. Other Nonfunctional Requirements

6.1 Performance Requirements

- The app should be capable of supporting a minimum of 100 simultaneous user connections without any noticeable degradation in response time, ensuring smooth user experience during peak usage periods.
- The app should generate reports and perform complex calculations within 5 seconds for a dataset of up to 1000 tools listed by Shop Owners, ensuring efficient operation and responsiveness for users managing tool rentals.

6.2 Safety Requirements

- The app should enforce user authentication through phone number verification to ensure secure access and prevent unauthorized usage.
- It should implement access control mechanisms to restrict access to sensitive user data, such as personal information and payment details, to authorized personnel only.
- Regular data backups should be performed to safeguard user information and ensure data availability in the event of system failures or disasters.
- The app should comply with relevant safety regulations and guidelines for tool rental facilities, ensuring the safety of users during tool pickups and returns.
- It should provide safety guidelines and tips within the app to promote safe handling and usage of rented tools, minimizing the risk of accidents or injuries.

6.3 Security Requirements

- The system should encrypt all sensitive user data, including passwords, personal information, and payment details, using industry-standard encryption algorithms.
- The system should have a secure login mechanism, enforcing password complexity and session timeout to protect against unauthorized access.
- The system should undergo regular security audits and vulnerability assessments to identify and address any potential security vulnerabilities.

6.4 Software Quality Attributes

- Data Encryption:
  The app must encrypt all sensitive user data, such as phone numbers and rental details, using industry-standard encryption algorithms to ensure confidentiality and integrity.

- Secure Authentication:
  Implement a secure login mechanism using phone number verification, enforcing robust password policies and session management to prevent unauthorized access.
  Ensure that user sessions expire after a specified period of inactivity to minimize the risk of unauthorized access.

- Regular Security Audits:
  Conduct regular security audits and vulnerability assessments to identify and address any potential security vulnerabilities in the app's infrastructure and codebase.
  Implement measures to patch known vulnerabilities promptly and continuously monitor for emerging threats.

7. References

[1] O. Mendis and G. Rathnayake, "GoHandy - Handyman Service Booking Application," 2020, doi: 10.1109/ICIP48927.2020.9367349.

[2] "A Survey of Cloud Database Systems | IEEE Journals & Magazine | IEEE Xplore." https://ieeexplore.ieee.org/abstract/document/6401099 (accessed Jun. 27, 2023).

[3] C. Te Lee, L. B. Chen, H. M. Chu, and C. J. Hsieh, "Design and Implementation of a Leader-Follower Handyman Service IEEE Access, vol. 10, pp. 28066–28079, 2022, doi: 10.1109/ACCESS.2022.3158494.

[5] R. Sahtyawan and A. I. Wicaksono, "Application for Control of Handyman Service Using Microcontroller Nodemcu Esp 8266 Compiler, vol. 9, no. 1, 2020, doi: 10.28989/compiler.v9i1.627.

[6] K. Y. Chen, M. C. Chen, and W. Y. Liu, "Designing data warehouses for equipment management system with genetic algorithms," Int. J. Prod. Res., vol. 46, no. 21, 2008, doi: 10.1080/00207540701222776.

[7] W. Yang, "Analysis on Online Payment Systems of Handyman Service E-Commerce," Oulu Univ. Appl. Sci., 2017.

[8] "Research on Online Payment Pattern and Security Strategy of Handyman Service E-Commerce | IEEE Conference Publication | IEEE Xplore." https://ieeexplore.ieee.org/abstract/document/5566314 (accessed Jun. 27, 2023).