

**CHINMAYA INSTITUTE OF TECHNOLOGY**  
**KANNUR**



**MINI PROJECT REPORT**

**On**

**SKILL SEEKER**

**Presented by**

**Ms. RIYA M GIJO**

**Reg No : C3GMCA2050**

**THIRD SEMESTER MCA(2023-2025)**

**SCHOOL OF COMPUTER SCIENCE AND  
INFORMATION TECHNOLOGY**

# **CHINMAYA INSTITUTE OF TECHNOLOGY KANNUR**

## **SCHOOL OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY**

### **DECLARATION**

I, **Riya M Gijo**, third Semester MCA, Student of Chinmaya Institute of Technology, do hereby declare that the Project Report entitled **Skill Seeker** is the original work carried out by me under the supervision of **Mr. Rohit Nambiar** towards partial fulfilment of the requirement of MCA Degree at Kannur University, Kannur.

(Signature of the Student)

Place: Kannur

Date

# CHINMAYA INSTITUTE OF TECHNOLOGY KANNUR

## SCHOOL OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY



### CERTIFICATE

This is to certify that the project entitled “**Skill Seeker**” submitted in partial fulfilment of the requirement for the award of M.C.A Degree of Kannur University, Kannur, is a result of Bonafide work carried out by **Riya M Gijo** (University Register Number **C3GMCA2050**) during the third semester in the year 2023-25.

#### External Examiners

1.

2.

**Faculty in charge**

**Place:** Kannur

**Date:**

**Head of department**

# ACKNOWLEDGMENT

I would like to express my deepest gratitude to all those who have helped and supported me during the development of this project.

First and foremost, I extend my sincere thanks to my project guide, Mr. Rohit Nambiar, for his invaluable guidance, encouragement, and constructive feedback throughout the project. His expertise and suggestions have played a vital role in shaping the project and ensuring its success.

I would also like to thank the MCA department for providing me with the necessary resources and environment to complete this project successfully. The facilities and infrastructure provided were essential in carrying out the research and development work.

Special thanks to my friends and colleagues for their support, ideas, and continuous motivation throughout the duration of this project. Their encouragement and feedback have been greatly appreciated.

This project would not have been possible without the support of all these individuals, and I am deeply grateful for their assistance.

**RIYA M GIJO**

# **ABSTRACT**

Finding a tradesman for small jobs has become increasingly time-consuming, and traditional word-of-mouth referrals are often unreliable. The Skill Seeker project is a web-based platform designed to connect property owners with skilled tradesmen for a variety of services. The system allows users to search for tradesmen based on specific job categories and geographical locations, enabling them to book appointments effortlessly. Tradesmen can create detailed profiles, manage their service offerings, and choose to accept or reject booking requests. This platform effectively bridges the gap between tradesmen and customers, offering a streamlined solution for service bookings. Additionally, Skill Seeker incorporates a review and rating system to help customers make informed decisions, along with a complaint management feature to resolve any issues. Thus, the Skill Seeker platform provides a comprehensive set of tools to ensure smooth interactions and effective management for both customers and tradesmen.

# TABLE OF CONTENTS

<b>Chapters</b>	<b>Contents</b>	<b>Page no</b>
1.0	Introduction	1
1.1	Existing and proposed system	2
1.2	Salient features of the system.	3
2.0	Motivation and Survey of problem area	4
3.0	Problem Formulation	6
3.1	Main Objective	7
3.2	Specific Objective	7
3.3	Methodology	8
3.4	Platform	9
4.0	System Analysis and Design	12
4.1	Feasibility Analysis	12
4.2	Requirement Analysis	15
4.3	System requirement specification software requirements	15

<b>Chapters</b>	<b>Contents</b>	<b>Page no</b>
4.4	Gantt Chart	18
4.5	Use Case diagram	19
4.6	Activity diagram	23
4.7	Sequence diagram	26
4.8	Data Flow Diagram	28
4.9	Interface design	33
4.91	Input design	34
4.92	Output Design	36
4.93	Menu design	38
4.10	Code design	39
4.11	Database design	43
4.12	ER diagram	48
4.13	Validation and checks	50
5.0	Implementation of System and Testing	52

<b>Chapters</b>	<b>Contents</b>	<b>Page no</b>
6.0	Conclusion	56
7.0	Suggestion for future Work	57
8.0	Bibliography	58
9.0	Appendices	59



## LIST OF DIAGRAMS

SI No	Diagrams	Page No
1	Use Case diagram	19
2	Activity diagram	23
3	Sequence diagram	26
4	Data Flow Diagram	28
5	ER diagram	48

## 1.0 INTRODUCTION

Skill Seeker is a dedicated platform designed to facilitate connections between property owners and skilled tradesmen. This website provides an efficient solution for homeowners and property managers seeking qualified professionals for various trades jobs. Users can register on the site, search for tradesmen by specifying job types and choose tradesmen based on location, detailed reviews and ratings from previous clients. The platform also allows users to book appointments directly with their chosen tradesmen.

For tradesmen, Skill Seeker offers the opportunity to register, create profiles, and manage their bookings through the site. Tradesmen can update their availability, view appointment requests, and provide services as needed.

By bridging the gap between property owners and trades professionals, Skill Seeker enhances job accessibility and streamlines the process of finding and hiring skilled tradesmen, ultimately benefiting both parties by ensuring efficient job completion and expanding job opportunities.

## **1.1 EXISTING AND PROPOSED SYSTEM**

### **EXISTING SYSTEM**

In existing system, finding a tradesman is through inefficient word-of-mouth referrals. This approach is often dependent on one's social network and lacks consistency. Meanwhile, tradesmen struggle to secure jobs due to limited networking opportunities. Additionally, the absence of alternative options means that if a tradesman is unavailable, property owners face delays because they have no immediate access to other qualified professionals. This gap highlights the need for a more efficient and comprehensive solution for connecting property owners with tradesmen.

Moreover, existing online platforms do not permit users to select their preferred tradesperson. Instead, a random registered tradesperson is assigned to the user. This approach restricts tradespeople from creating profiles that customers can view and review.

### **PROPOSED SYSTEM**

To address the limitations of existing systems, the proposed tradesman-finding website introduces a comprehensive platform designed to seamlessly connect users with skilled tradesmen. This system offers robust registration features, allowing both users and tradesmen to create and manage their accounts with ease. Users can utilize a simple interface to search for tradesmen based on specific skills, ensuring that they find the most suitable professionals for their requirements.

Additionally, the website incorporates a review and rating system, empowering users to provide feedback upon the completion of work. This feature not only helps users make informed decisions but also promotes tradesmen to deliver high-quality services to maintain

positive ratings. The system also enables customers to select tradesmen according to their location, ensuring convenience and accessibility. By combining these advanced functionalities, the proposed system bridges the gap between demand and supply in the tradesman industry, creating a reliable and efficient environment for both property owners and skilled professionals.

## **1.2 SALIENT FEATURES OF THE SYSTEM.**

- Both users and tradesmen can create and manage their accounts on the platform.
- Users can search for tradesmen based on skills and location to find the most suitable professionals.
- Users can request services by booking tradesmen directly through the platform.
- Tradesmen can approve or reject booking requests based on their availability.
- Users can submit complaints about services, and admins can view and reply to these complaints.
- Users can provide feedback on tradesmen, and tradesmen can view their reviews to assess their service quality.
- Tradesmen can generate and update bills for services, which users can view and pay directly through the platform.
- Admins can approve tradesmen registrations, monitor registered users, and ensure platform integrity.
- The platform ensures a seamless experience for all stakeholders.

## **2.0 MOTIVATION AND SURVEY OF PROBLEM AREA**

### **Motivation**

This project was inspired by the need to bridge the gap between property owners and skilled professionals effectively. In today's fast-paced world, property owners often encounter significant challenges in finding reliable tradesmen for tasks such as plumbing, electrical repairs, or construction work. Market research revealed that property owners are often limited by the small pool of tradesmen they are aware of, leading to delays, especially when their preferred tradesmen are unavailable for urgent tasks.

Conversely, tradesmen struggle to secure consistent work due to their limited network and lack of visibility. Many rely on word-of-mouth recommendations, which restricts their reach to a small circle of clients. This often results in missed opportunities for tradesmen and prolonged inconvenience for property owners. These challenges highlight a pressing need for a comprehensive platform that seamlessly connects property owners with skilled professionals, ensuring accessibility, reliability, and convenience for all parties involved.

### **Survey of Problem Area**

The tradesman industry currently faces several challenges that hinder efficiency and reliability for both customers and service providers. Key issues include:

#### **1. Difficulty in finding tradesmen**

Property owners often have a limited network of tradesmen for each type of work. When their preferred tradesmen are unavailable, they struggle to find alternatives, leading to delays, even for urgent tasks. In other booking systems, tradesmen are assigned by the company, leaving

customers with no choice in selecting professionals based on their preferences. This approach creates dissatisfaction among customers who value trust and familiarity in the services they receive.

Additionally, word-of-mouth systems, a common method for finding tradesmen, only provide reviews from people the customer knows. This limits the ability to evaluate tradesmen based on a wider pool of feedback, resulting in incomplete or biased information about the quality of service.

## 2. Lack of flexibility for tradesmen

In current booking systems, tradesmen are unable to reject incoming requests from customers, even when they have prior commitments or scheduling conflicts. This lack of flexibility is particularly inconvenient during emergencies or unforeseen circumstances, forcing tradesmen to either overcommit or cancel at the last moment, leading to dissatisfaction for both parties.

Similarly, customers also lack the ability to cancel appointments in case of emergencies, leaving them bound to unnecessary commitments or potentially incurring penalties. This rigidity in booking systems increases frustration and undermines the overall user experience for both tradesmen and property owners.

## 3. Absence of a robust feedback mechanism

Many existing systems fail to incorporate a robust feedback mechanism. Users often lack the ability to share public reviews or rate services, making it difficult for others to assess the quality of tradesmen's work. In word-of-mouth systems, feedback is limited to a small network, depriving customers of a broader perspective on the tradesman's performance.

Furthermore, there is often no system for users to lodge complaints against tradesmen or seek resolution for unsatisfactory services. This creates a lack of accountability, as tradesmen have no motivation to maintain high standards when poor service cannot be documented or addressed.

### 3.0 PROBLEM FORMULATION

In today's world, property owners face significant challenges in finding reliable and skilled tradesmen to address their repair and maintenance needs. Existing platforms for hiring tradesmen often lack essential features that allow users to efficiently search for and book specific tradesmen of their choice. Most websites focus only on general service categories without providing users the ability to select a preferred tradesman, making it difficult to establish a sense of trust and personalization. Additionally, these platforms do not always provide sufficient transparency in terms of reviews, ratings, and availability, further complicating the decision-making process.

Tradesmen also encounter difficulties in showcasing their expertise, building trust with potential clients, and managing their bookings effectively. The absence of a centralized, user-friendly system prevents seamless communication and coordination between property owners and tradesmen. This gap creates inefficiencies, wastes time, and reduces customer satisfaction.

The Skill Seeker project aims to bridge this gap by providing an innovative platform where property owners can search for tradesmen based on specific job types and even book a particular tradesman of their choice. The platform allows users to browse profiles, view reviews and ratings, and schedule appointments with ease. Tradesmen, in turn, can register, create and manage profiles, handle bookings, and build credibility through customer feedback. By offering transparency, convenience, and personalization, Skill Seeker addresses the limitations of existing solutions and provides an all-encompassing service for both property owners and tradesmen.

### 3.1 Main Objective

- Facilitate seamless connections - Create a platform that connects property owners with skilled tradesmen efficiently and effectively.
- Enable location based search - Allow users to search for tradesmen based on their location and job requirements, ensuring convenience and accessibility.
- Incorporate advanced search features - Provide options for users to filter tradesmen by category, location, and customer reviews.
- Enhance tradesman flexibility - Offer tradesmen the ability to manage their schedules, accept or decline bookings, and grow their visibility to a broader audience.
- Introduce feedback and rating mechanisms - Enable users to share feedback, rate services, and lodge complaints to ensure transparency and accountability.
- Streamline booking and payments - Provide an efficient booking system integrated with secure and modern payment options, eliminating the need for manual processes.
- Promote high quality services - Encourage tradesmen to maintain high standards of service to achieve positive ratings and reviews.
- Bridge the Gap Between Demand and Supply - Address the challenges of finding skilled tradesmen and securing consistent work for professionals, fostering a reliable and efficient ecosystem.

### 3.2 Specific Objective

- Customer freedom to choose tradesmen - Empower customers to select tradesmen based on their preferences, such as reviews, ratings, and location, instead of being assigned a tradesman by the system.
- Tradesman flexibility to approve or reject requests - Provide tradesmen with the ability to accept or decline job requests, ensuring they have control over their schedules and commitments.



- Flexible booking system - Enable customers to cancel or reschedule appointments when necessary, fostering convenience and adaptability for all parties involved.

### 3.3 Methodology

This project adopts the Agile methodology, a flexible approach to software development that promotes collaboration, adaptability, and continuous improvement. The primary reason for choosing Agile is its ability to easily adapt to new changes during the development process. Initially, the project was planned as a simple platform to book tradesmen. However, as the project progressed, I realized the potential to enhance its functionality by incorporating additional features. These included options for users to view upcoming, previous, and today's bookings, as well as a feature to show the request status.

These changes and feature additions were seamlessly integrated into the project thanks to agile's iterative approach. Agile allowed me to prioritize tasks, implement changes incrementally, and continuously improve the system based on new ideas and feedback. This adaptability ensured the project evolved in alignment with the growing requirements, demonstrating the effectiveness of the agile methodology in dynamic and evolving projects like this one.

#### **Iteration:**

The various iterations performed in the project include:

1. Planning and documentation: Define project scope, requirements, and system design.
2. Customer management: Develop user registration, login, and profile management.
3. Tradesman management: Implement tradesmen registration and profile setup.
4. Category management: Create and manage job categories for filtering tradesmen.
5. Booking request management: Build the system for managing and tracking service requests.
6. Review and rating management: Enable users to provide feedback and rate tradesmen.
7. Complaint management: Develop a system for lodging and resolving customer complaints.
8. Payment management: Integrate secure payment methods, including QR code-based payments.

#### **Key points of agile methodology:**

- Iterative development
- Collaboration and constant communication between stakeholders to ensure that everyone is aligned and involved throughout the project.
- Flexibility and adaptability
- Continuous testing and integration

### **3.4 Platform**

The platform used in this project forms the foundation for the architecture, design, and implementation of the web application. The tools and technologies used in this software program are as follows:

- Operating System: Windows
- Technology: Frontend – HTML, CSS; Backend – Python (with Django framework); Database – MySQL; Server – Apache Server

#### **Frontend**

In this project, HTML and CSS were used to build and style the frontend of the web application. HTML is a markup language used for structuring and presenting content on the world wide web. HTML was used to structure the content of the platform, creating essential elements such as navigation menus, forms for customer and tradesman registration, booking forms, and profile pages. The HTML markup formed the backbone of the user interface, ensuring that all components of the site were organized and accessible.

CSS was then applied to style these HTML elements, ensuring a visually appealing and user-friendly experience. CSS is a style-sheet language used to control the presentation and formatting of HTML documents. It allows developers to define styles, such as colors, fonts, spacing, and positioning, providing a consistent and visually appealing design across multiple web pages. Through CSS, I was able to manage aspects like color schemes, typography, and

layout designs. Together, HTML and CSS allowed for a clean, intuitive, and attractive user interface that enhances the overall user experience.

## **Backend**

In this project, Python Django was used as the backend framework to handle the core functionality of the web application. Django is a high-level web framework built with Python language that allows for rapid development of secure and maintainable web applications. It provides built-in features like URL routing, database models, authentication, and session management, which significantly speed up development.

I used Django to manage the backend logic of the platform, including user registration and login, handling booking requests, and processing payment transactions. Django's Model-View-Template (MVT) architecture was employed to separate the business logic from the presentation layer, ensuring that the codebase remains clean and maintainable. The Django ORM (Object-Relational Mapping) was utilized to interact with the MySQL database, enabling efficient data storage and retrieval for user profiles, tradesman details, bookings, reviews, and payment records. Additionally, Django's powerful templating engine allowed me to dynamically generate HTML content and render it on the frontend, providing a seamless user experience.

## **Database**

For this project, MySQL was used as the relational database management system, and SQLyog Community Edition was employed as the database management tool. SQLyog provided a user-friendly interface that made it easier to manage and interact with the MySQL database. This tool helped streamline database administration tasks, such as creating tables, querying data, and optimizing performance.

MySQL offers several advantages, including:

- **High Performance:** MySQL is known for its fast data processing and query execution, ensuring the platform remains responsive even with large datasets.

- **Scalability:** MySQL can handle large amounts of data efficiently, which is essential for a growing platform with increasing user activity.
- **Data Integrity:** MySQL's ACID compliance ensures that the data is consistently accurate and reliable, supporting critical operations such as bookings and transactions.
- **Cost-Effective and Open-Source:** Being open-source, MySQL offers a cost-effective solution without licensing fees, making it a suitable choice for both small and large-scale projects.

Using MySQL alongside SQLyog enabled the efficient management of user profiles, tradesman details, booking information, reviews, ratings, and payment transactions. The database schema was designed to ensure that the data remained well-organized and easily accessible, supporting the platform's scalability and reliability.

### **Server**

WAMP (Windows, Apache, MySQL, PHP) is a popular software stack used to create a local web development environment on Windows operating systems. It combines Apache (a widely-used web server), MySQL (as the relational database management system), and PHP (a server-side scripting language). WAMP simplifies the process of setting up a web server environment for development purposes, making it an ideal choice for projects like mine.

In this project, I used WAMP Server to host the local development environment, allowing for easy integration of the Apache web server with the MySQL database. Apache was used to serve the web pages and handle HTTP requests, while MySQL was employed for data storage, managing user profiles, tradesman details, booking records, and payment transactions. WAMP's simplicity in setting up the server and database made it easier to test the backend functionalities of the application before deploying it to a live environment. Additionally, WAMP provided tools to manage and configure the server and database settings, ensuring smooth development and testing of the application locally.

## 4.0 SYSTEM ANALYSIS AND DESIGN

### 4.1 Feasibility Analysis

Feasibility analysis is a crucial process in determining whether a project is viable in terms of its operations, technical aspects, and economics. It helps in assessing if the project can be successfully developed and deployed within the required time and budget, while also ensuring that it meets the needs and objectives of the stakeholders.

#### Market Feasibility

Market research conducted with potential customers, including property owners and tradesmen, has revealed a significant demand for a tradesman-finding website. Property owners are concerned about the limited number of tradesmen they are aware of and the inconvenience of waiting for extended periods when their preferred tradesman is unavailable, even for urgent jobs. Conversely, tradesmen face challenges in securing jobs due to their small network. This indicates a clear need to bridge the gap between property owners and tradesmen. This research confirms a compelling market opportunity for a specialized tradesman-finding website to address these issues and effectively meet the needs of both property owners and tradesmen.

### REVENUE GENERATION

The Skill Seeker platform provides opportunities for revenue generation by leveraging multiple monetization strategies. These strategies are designed to benefit both the platform and its users, ensuring sustainability and growth. Key revenue streams include:

1. Booking Commission

Skill Seeker can charge a small commission on each successful booking made through the platform. This ensures a continuous flow of revenue proportional to the platform's

usage, creating a scalable revenue stream.

## 2. Payment Processing Fee

Since the platform integrates a payment system, a nominal fee can be charged for processing payments made through the system, particularly for QR code-based transactions.

The main type of feasibility studies done for this project are:

### **Operational Feasibility:**

Operational feasibility evaluates whether the system can be effectively implemented and used in a real-world environment. It assesses the ability to integrate the system with existing processes and workflows.

The project is highly operationally feasible, as it is designed to seamlessly integrate into real-world environments. The platform provides an intuitive user interface that allows customers to easily book tradesmen and review their services, while also giving tradesmen the flexibility to manage their bookings and availability. The system's simplicity ensures that both customers and tradesmen can efficiently use the platform without the need for extensive technical knowledge. Therefore, the operational needs of the users are fully addressed, ensuring smooth functionality and ease of use.

### **Technical Feasibility:**

Technical feasibility examines the technical resources and skills required to develop and deploy the project. This includes evaluating the technology stack, the availability of tools, and the ability to build the system within the specified technical constraints.

From a technical standpoint, the project is entirely feasible due to the use of proven, well-supported technologies. The combination of Python and the Django framework for backend development, along with MySQL for data management, offers a robust and scalable foundation for the system. These technologies are reliable, widely used, and provide a strong community of support. Additionally, PyCharm was used as the integrated development environment (IDE) for Python and Django development, providing powerful tools for code management,

debugging, and testing, which enhanced the development process. Furthermore, the use of WAMP server for local development ensures that the project can be easily tested and refined before deployment.

**Economic feasibility:**

Economic feasibility evaluates the financial aspects of the project, ensuring that the costs associated with development, implementation, and maintenance are within the available budget and provide good value for the investment.

Economically, the project is viable due to its cost-effective nature. By utilizing open-source technologies like Python, Django, and MySQL, the project significantly reduces the need for costly software licenses and tools. Additionally, the local development environment provided by WAMP server minimizes infrastructure costs during the development phase. Overall, the project offers an affordable solution that can generate value for all stakeholders involved.

**Behavioural Feasibility:**

Behavioural Feasibility refers to the assessment of how well a system aligns with the behaviours, needs, and expectations of its users. It evaluates whether the users will adopt and effectively interact with the system. This aspect ensures that the platform is user-friendly, intuitive, and provides a positive experience that encourages engagement and sustained use.

This project is behaviourally feasible because it is designed to meet the specific needs and expectations of its users (property owners and tradesmen) while ensuring ease of use and engagement. The user interface is simple and intuitive, allowing property owners to easily search for, book, and review tradesmen, making the booking process hassle-free. For tradesmen, the flexibility to manage their availability and reject or accept bookings according to their schedules ensures that they can use the platform without disrupting their existing commitments.

Moreover, the inclusion of features like reviews, ratings, and complaint management fosters trust and accountability. These features align with users' natural expectations of having control over their choices and experiences, leading to greater adoption and sustained usage.

## 4.2 Requirement Analysis

Requirement analysis for the Skill Seeker project involves identifying, documenting, and validating the needs of stakeholders to ensure the platform aligns with user expectations and business objectives. The primary stakeholders include property owners seeking skilled tradesmen, tradesmen offering services, and administrators managing the platform. The analysis focuses on functional requirements such as user registration, tradesman search, booking systems, payment integration via QR codes, and review mechanisms. It also covers non-functional requirements, including performance, scalability, security, usability, and maintainability. By thoroughly analysing and prioritizing these requirements, Skill Seeker ensures a seamless user experience, robust system architecture, and alignment with the project's goal of connecting property owners with tradesmen efficiently.

## 4.3 SYSTEM REQUIREMENT SPECIFICATION SOFTWARE REQUIREMENTS

### Software requirements

Web Server: Apache (WAMPP stack for local development).

Backend Technology: python (Django framework).

Database: MySQL (to store user, product, order, and transaction data).

Frontend Technologies: HTML5, CSS3

Operating System: Windows

### Hardware requirements (Minimum)

Processor: i5 10<sup>th</sup> Gen

SSD Storage: 256 GB

RAM: 2.00GB



## Development tools

- IDE: Visual Studio Code
- Browser: Google Chrome, Mozilla Firefox.

## Functional requirements

Functional requirements define the specific functionalities that the Skill Seeker platform must provide to ensure it meets the needs of its users effectively. The functional requirements of this project are:

### 1. User Management

- Registration:
  - Property owners and tradesmen must be able to register on the platform.
  - Collect details such as name, email, password, and contact information.
  - Tradesmen need additional details such as trade category.
- Login/Authentication:
  - Secure login system with username and password.
- Profile Management:
  - Property owners can update their personal details and view their booking history.
  - Tradesmen can create, update, and showcase profiles, including profile pictures, trade expertise, certifications, and availability.

### 2. Search Functionality

- Property owners must be able to search for tradesmen based on Trade category, Ratings and Reviews.
- Show a list of tradesmen matching the search criteria.

### 3. Tradesman Selection and Booking

Property owners must be able to:

- Select a specific tradesman of their choice.

- View the tradesman's detailed profile, including reviews, ratings, and availability.
- Book appointments by selecting a service date and time.

#### 4. Payment System

- Integration of a secure payment gateway to handle transactions.
- Maintain transaction history for users.

#### 5. Review and Rating System

- Property owners must be able to submit reviews and ratings for tradesmen after availing of their services.
- Ratings should be on a scale (e.g., 1 to 5 stars) and include text review.
- Tradesmen can view the reviews and ratings to improve their service quality.
- 

#### 6. Booking and Appointment Management

- Tradesmen must be able to view and manage upcoming bookings via a dashboard.
- Property owners can cancel bookings based on their availability

#### 7. Admin Dashboard

Provide administrators with tools to manage the platform, such as:

- Monitoring registered customers and tradesman.
- Approving or rejecting tradesmen registrations.
- Send reply to customer complaints.

### **Non-Functional Requirements**

Non-functional requirements define the quality attributes of the system and ensure that it operates effectively, efficiently, and reliably. These requirements complement functional requirements and focus on aspects such as performance, usability, security, and scalability.

#### 1. Performance Requirements

- The platform must respond to user actions (e.g., searches, bookings, or profile updates) within 2-3 seconds under normal conditions.

## 2. Scalability Requirements

- The platform must be able to scale horizontally to accommodate a growing number of users, including property owners, tradesmen, and administrators.
- Database design should support an expanding dataset for user profiles, bookings, reviews, and payment records.

## 3. Usability Requirements

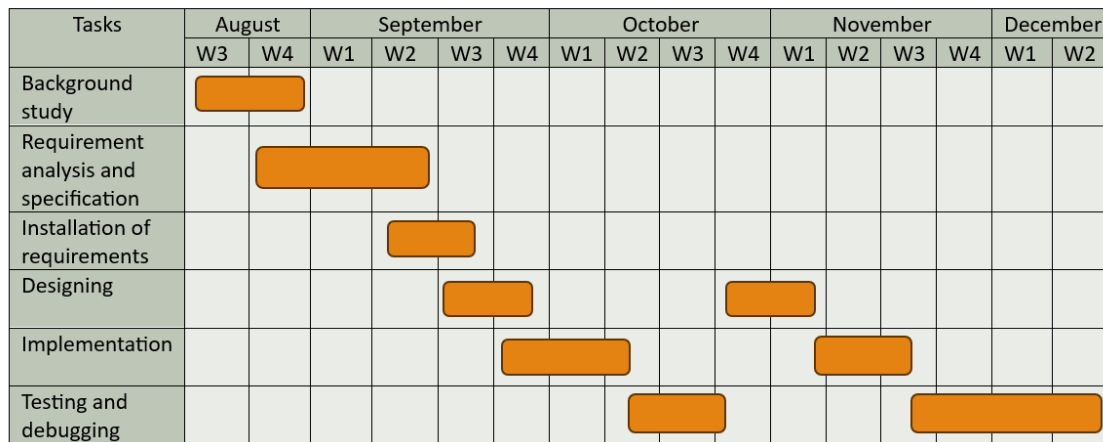
- The platform must provide an intuitive and user-friendly interface for both property owners and tradesmen, designed using responsive web design principles.
- Provide clear error messages and guidance for users in case of invalid input or system errors.
- Maintain a consistent visual design across all pages and features.

## 4. Maintainability and Extensibility Requirements

- The codebase must be modular and well-documented to facilitate easy updates and debugging.
- The architecture should allow new features to be added with minimal impact on existing functionality.

### 4.4 Gantt Chart

The following Gantt chart shows the time allocated for completing each task of the project. Since the Agile model is used, new requirements can be added or modified as needed, allowing for flexibility throughout the development process.



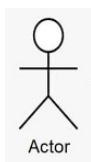
As depicted in the Gantt chart, requirement analysis and specification, final testing and debugging took up the majority of the time. This was due to the need for thorough analysis of evolving requirements, frequent revisions, and ensuring the system met all user needs. Additionally, final testing and debugging required substantial time to ensure the system was fully functional, bug-free, and aligned with project goals, as any issues discovered were addressed iteratively in multiple testing phases.

## 4.5 Use Case diagram

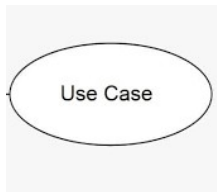
A Use case diagram is a visual representation of the functional requirements of a system, showing how users (or "actors") interact with the system to achieve specific goals or tasks. It helps to capture the system's functional requirements and the relationships between different users and the system's processes.

Components of a Use Case Diagram:

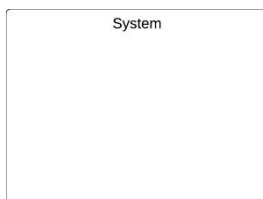
- **Actors:** These are the external entities that interact with the system. It is represented as



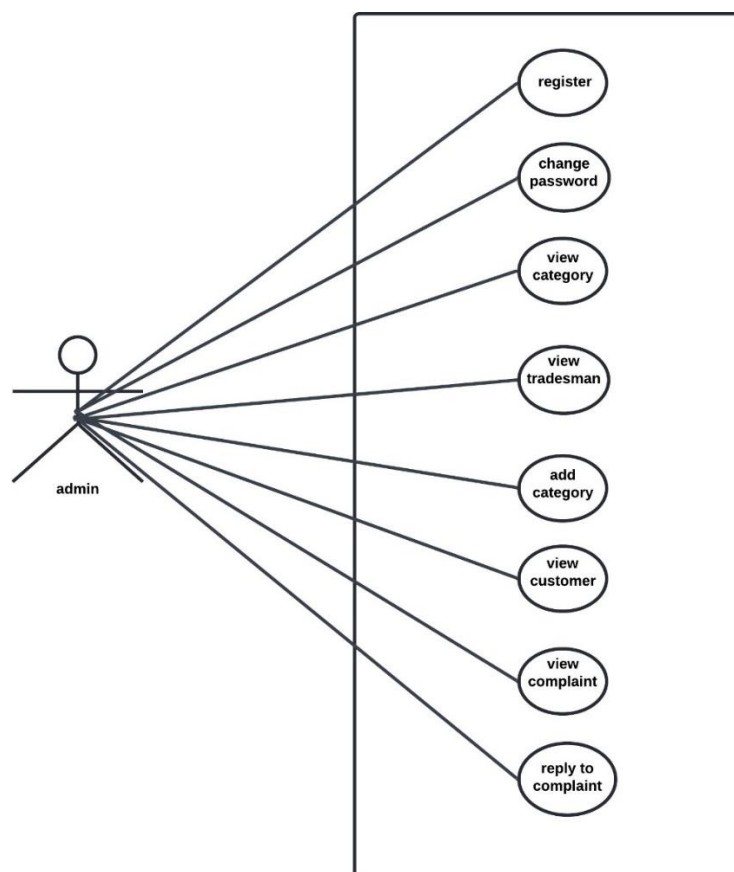
- **Use Cases:** A use case represents a specific function of the system. It is represented as

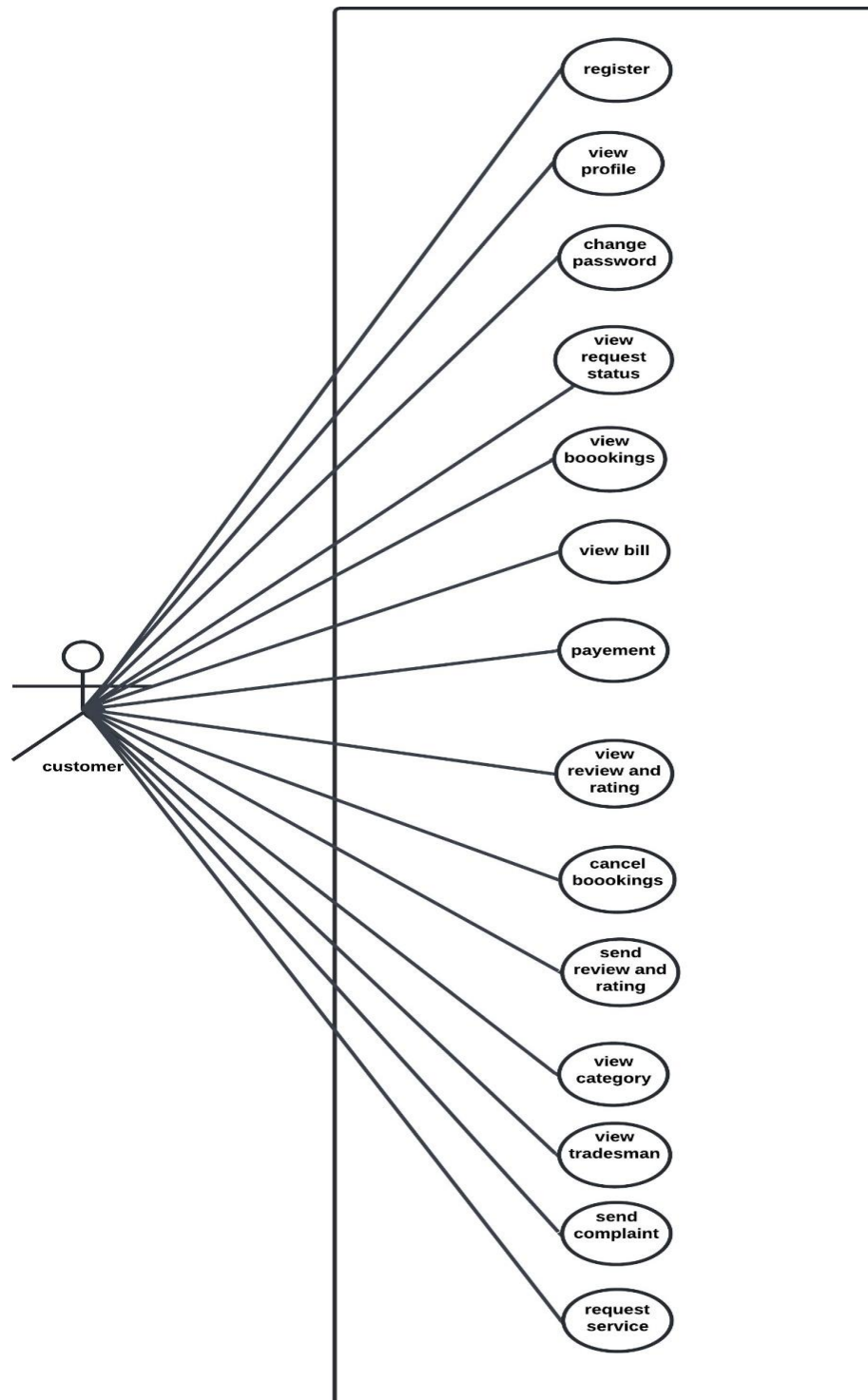


- System: The boundary or scope of the system. It is represented as



### Admin Use Case:

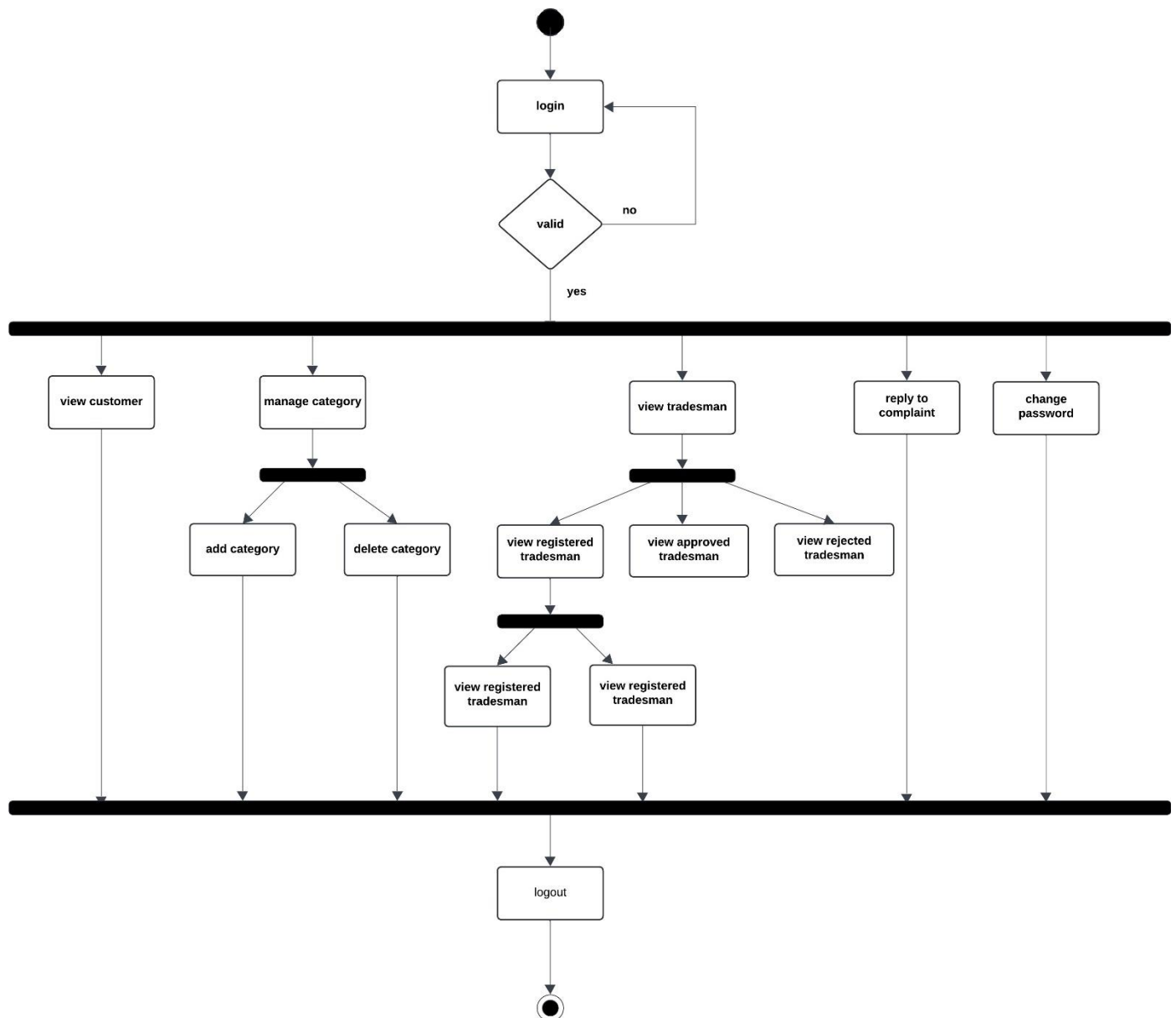


**Customer Use Case:**

**Tradesman Use Case:**

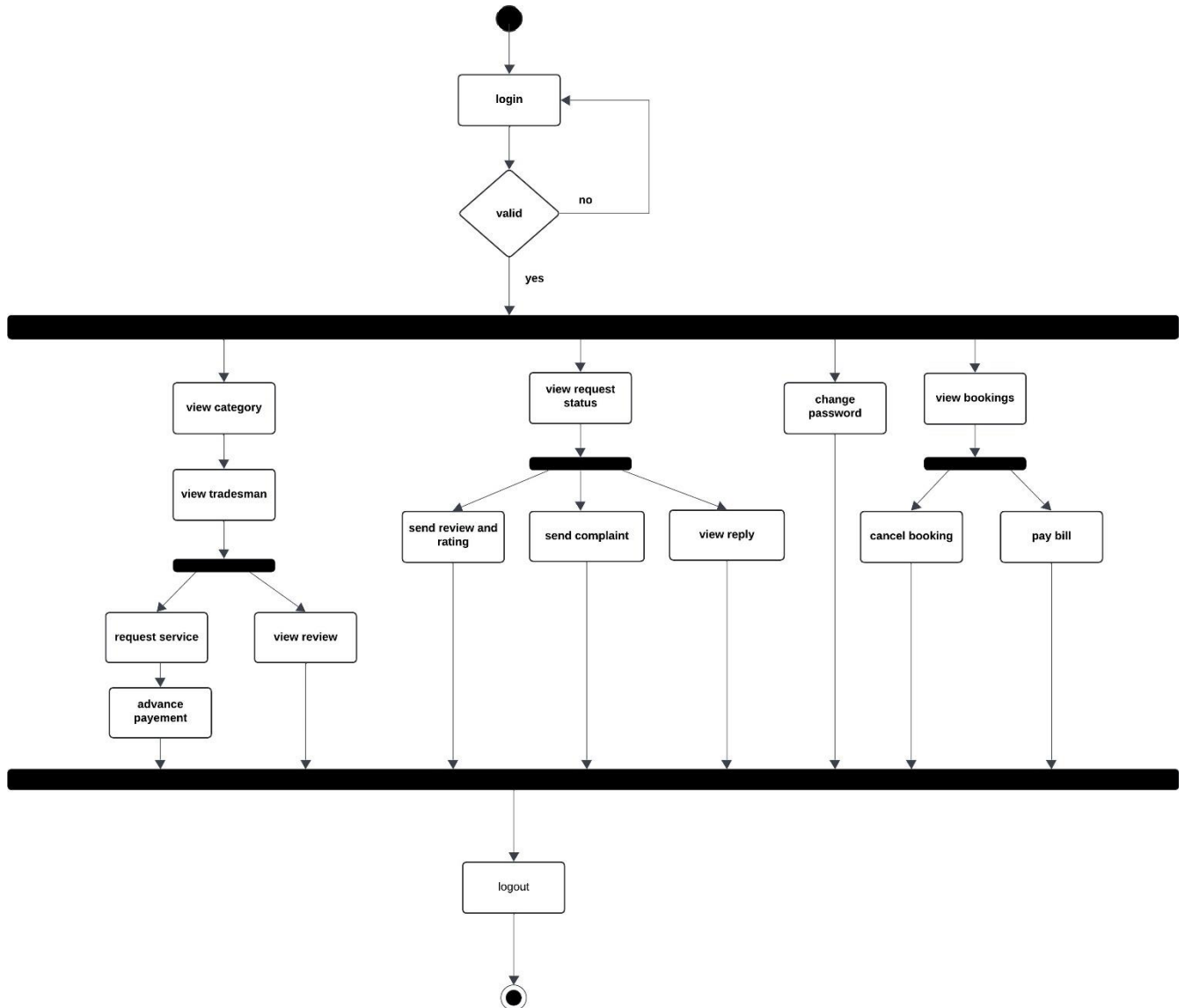
## 4.6 Activity diagram

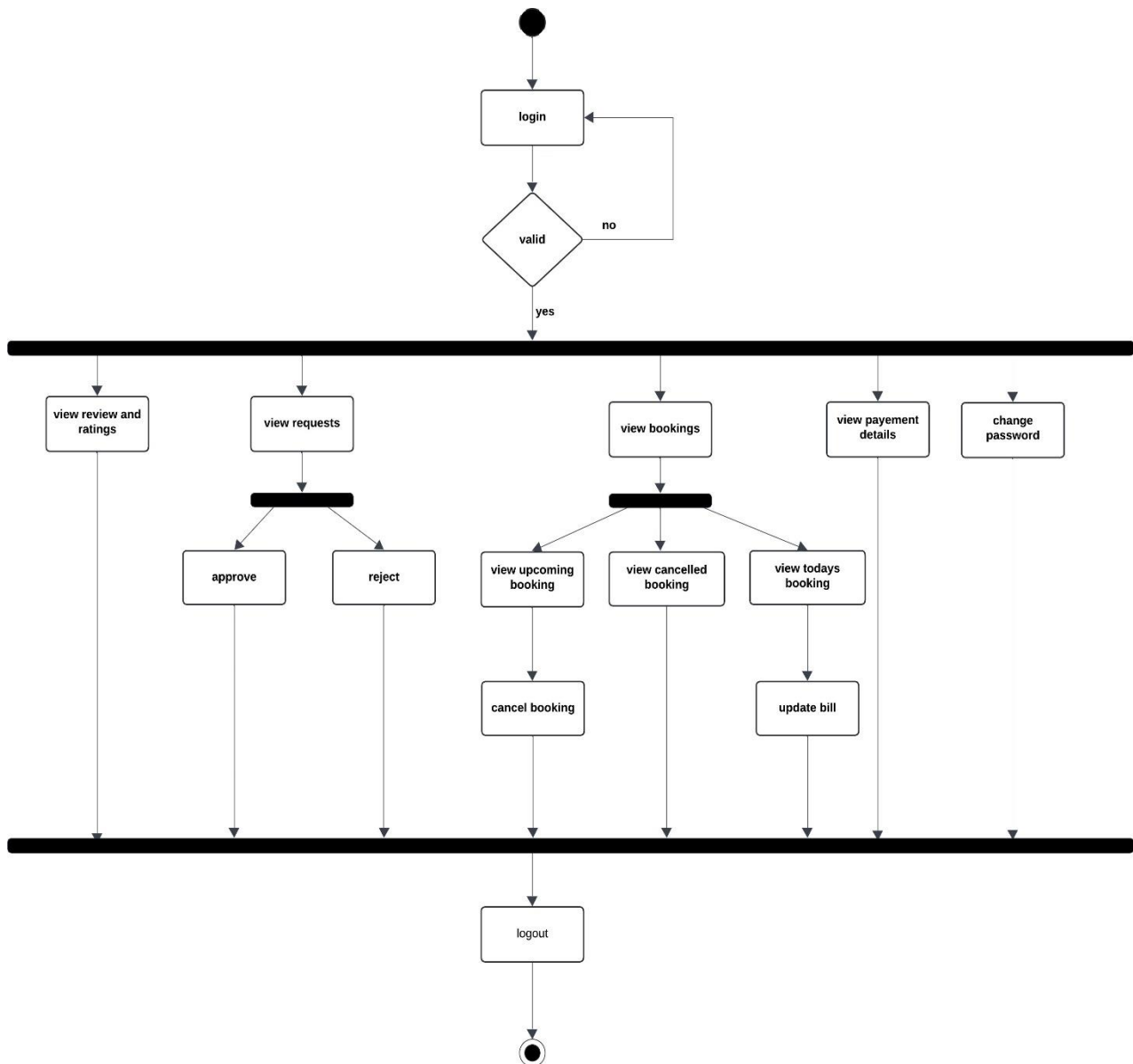
### Admin Activity Diagram:





### Customer Activity Diagram:

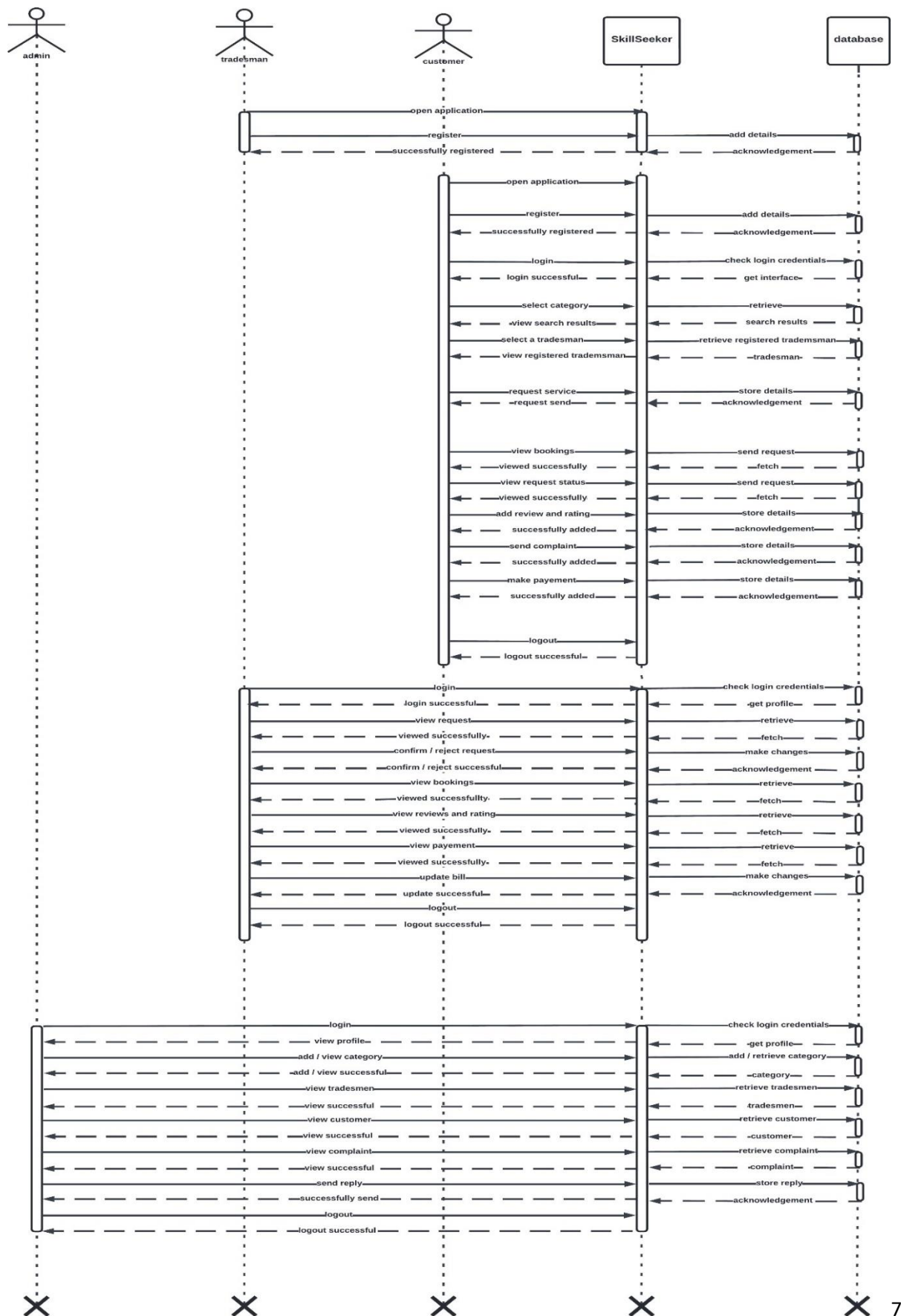


**Tradesman Activity Diagram:**

## 4.7 Sequence diagram

A Sequence Diagram is a type of UML (Unified Modelling Language) diagram that visually represents the flow of interactions between objects or components in a system over time. It focuses on how and in what order the components interact to perform a particular function or achieve a use case. Main Components of a Sequence Diagram:

- Actors - Represent external entities that interact with the system.
- Objects - Represent system components or instances (e.g., database, server, or a class).
- Lifelines - Represent the existence of an actor or object over time during the interaction.
- Messages - Arrows between lifelines that depict communication or interaction between actors and objects.
- Activation Bars - Represent the period an object is active and processing a message.



## 4.8 Data Flow Diagram

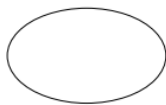
A Data Flow Diagram (DFD) is a graphical representation of the flow of data within a system. It illustrates how data is processed, transformed, and transferred between different components, including external entities, processes, data stores, and data flows.

### Main Symbols Used in a DFD:

- External Entity - Sources or destinations of data outside the system (e.g., a user, organization, or another system). It is represented as



- Process - A function or activity that transforms input data into output data. It is represented as

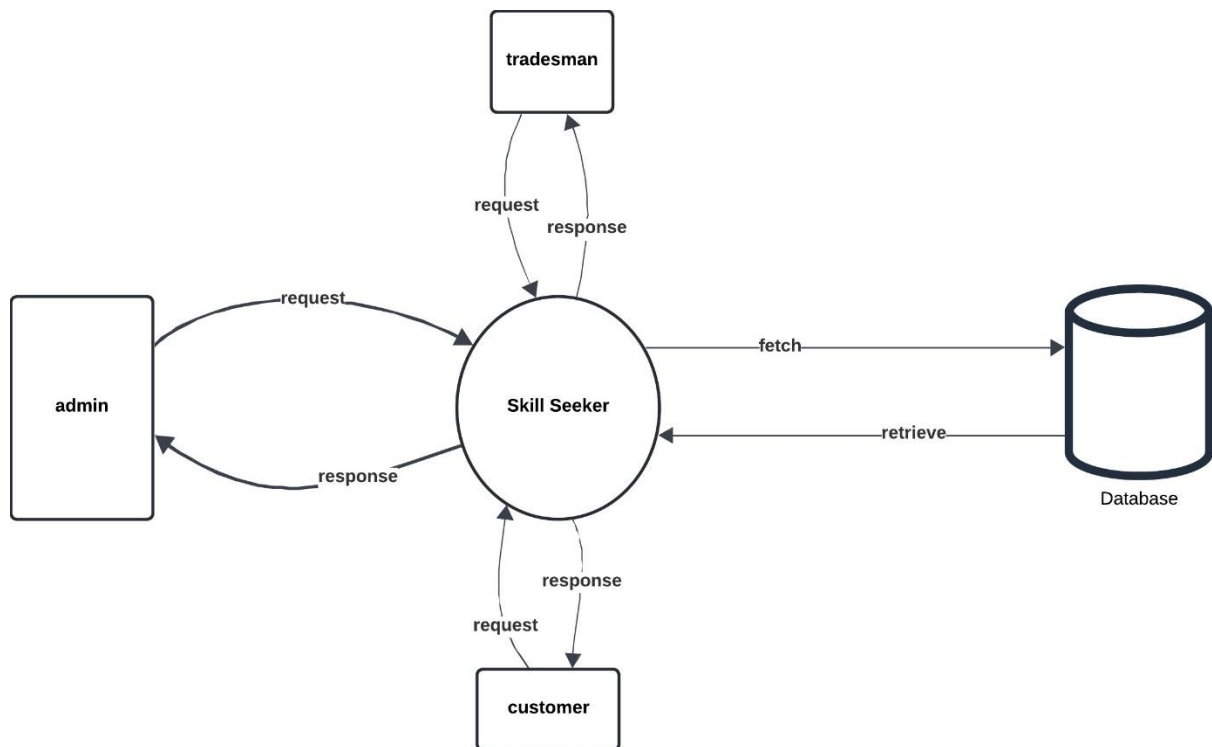


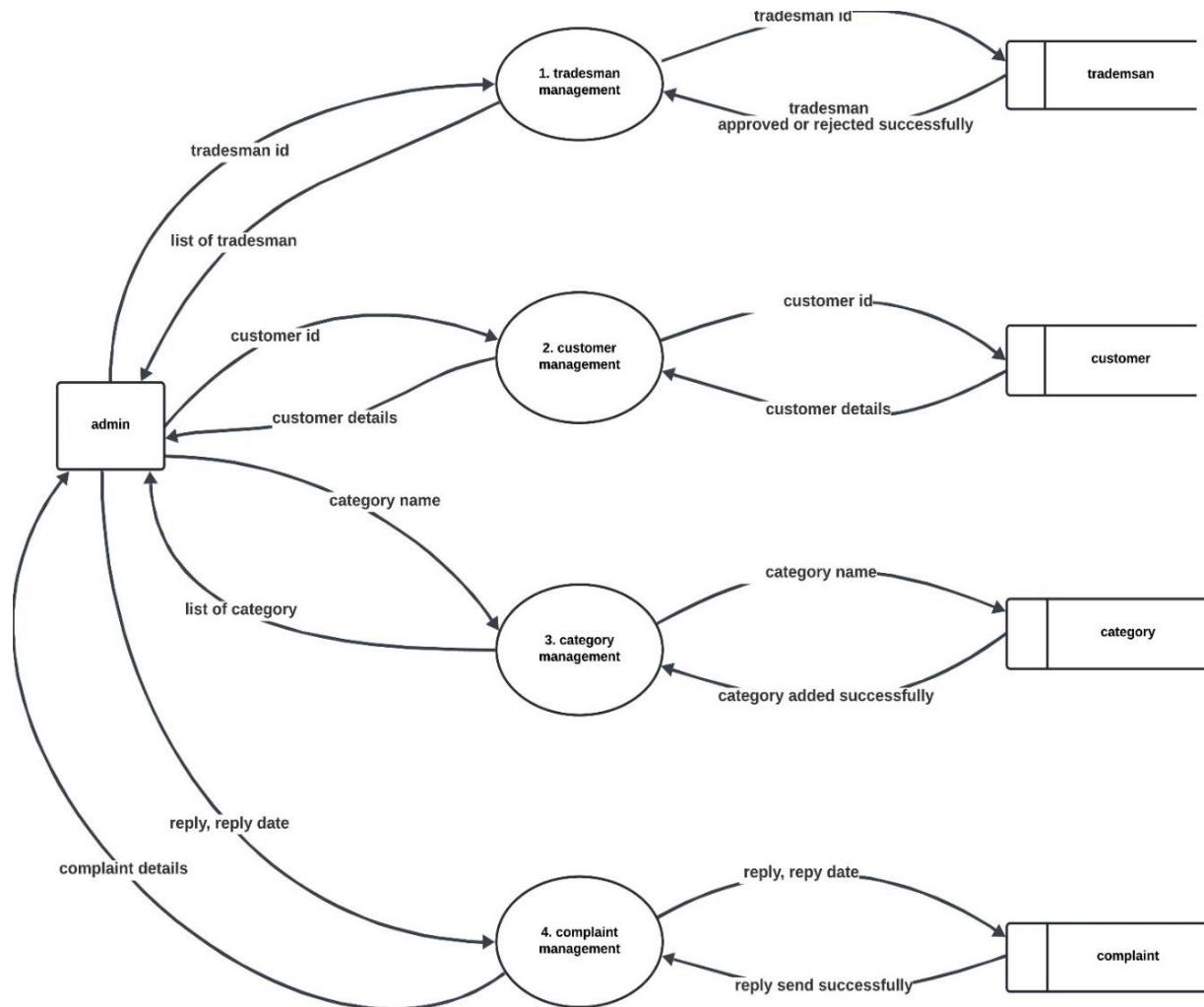
- Data Store – It is a repository where data is stored (e.g., database, file, or storage unit). It is represented as

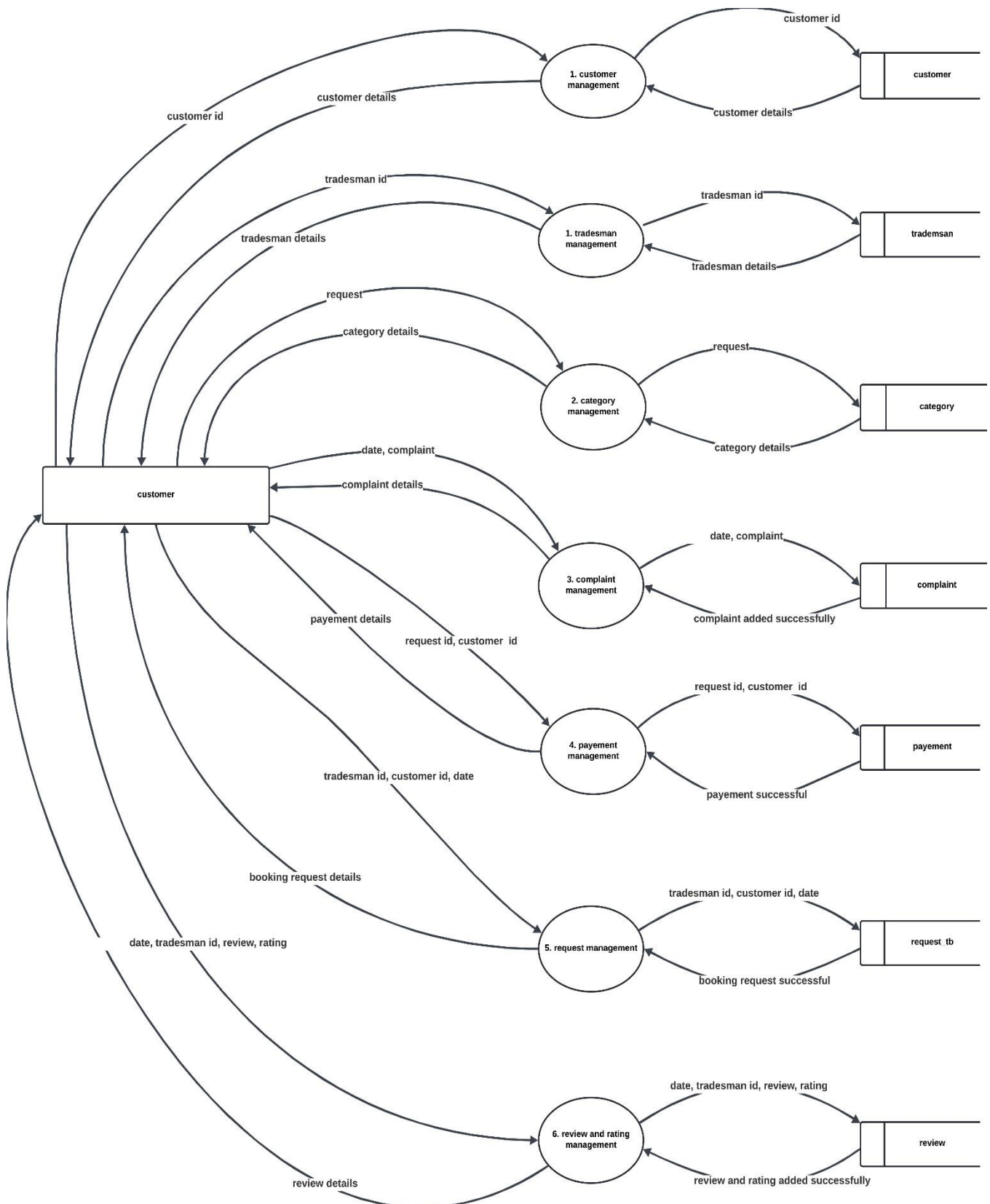


- Data Flow – It denotes the movement of data between entities, processes, and data stores. It is represented as



**Level 0 DFD:**

**Admin Level 1 DFD:**

**Customer Level 1 DFD:**



**Tradesman Level 1 DFD:**

## 4.9 Interface design

Interface design is the process of creating the visual and interactive elements of a system that enable users to interact with it effectively. It focuses on designing user interfaces (UIs) that are intuitive, efficient, and aesthetically pleasing, ensuring a seamless interaction between the user and the system.

Interface design encompasses:

### 1. Visual Design:

- Involves the selection of colours, fonts, layouts, and visual elements to create an appealing and cohesive look.
- Ensures that the interface aligns with branding and enhances usability.

### 2. Interaction Design:

- Defines how users interact with the system, including buttons, menus, input fields, and other interactive components.
- Focuses on creating predictable and user-friendly interaction patterns.

### 3. Usability:

- Prioritizes ease of use, ensuring that users can achieve their goals with minimal effort and confusion.

- Includes considerations for accessibility to accommodate users with disabilities.

#### 4. Consistency:

- Ensures uniformity in design elements across all pages and features, improving user familiarity and efficiency.

### 4.9.1 Input design

Input design is the process of creating efficient, user-friendly methods for capturing and processing data within a system. It focuses on how data is entered into the system, ensuring accuracy, consistency, and minimal effort for users. Proper input design minimizes errors, speeds up data entry, and ensures the system operates effectively.

In my project, input design was meticulously planned to ensure ease of use, accuracy, and security for users. Input forms were created for various functionalities, such as user registration, tradesman registration, customer registration and booking requests. Each input field was designed with proper validation mechanisms to prevent errors—for example, email fields ensure correct formatting, and numeric fields restrict non-numeric entries. Drop-down menus were used for predefined choices, such as selecting tradesman categories minimizing input errors and improving efficiency. Additionally, all sensitive inputs, like passwords, were securely handled by validation using pattern. This thoughtful input design ensures a seamless user experience while maintaining data integrity and reliability.

Some examples of input forms present in this project are:

- Add category
- Send booking request
- Send review and rating
- Send complaint

- Send reply

Registration Form

awd

kannur 676767 kannur

Google  
This page can't load Google Maps correctly.  
Do you own this website?

CHOOSE CATEGORY

gardener  
plumber

Password

Confirm Password

Submit

Figure 1: Drop down list in tradesman registration

Registration Form

testuser

8957412480 test@gmail.com

676767 kannur kannur

Google  
This page can't load Google Maps correctly.  
Do you own this website?

...

Please match the requested format.  
Password must contain 8 characters and at least one lowercase letter, one capital letter and one number

Submit

Figure 2: Password validation in customer registration

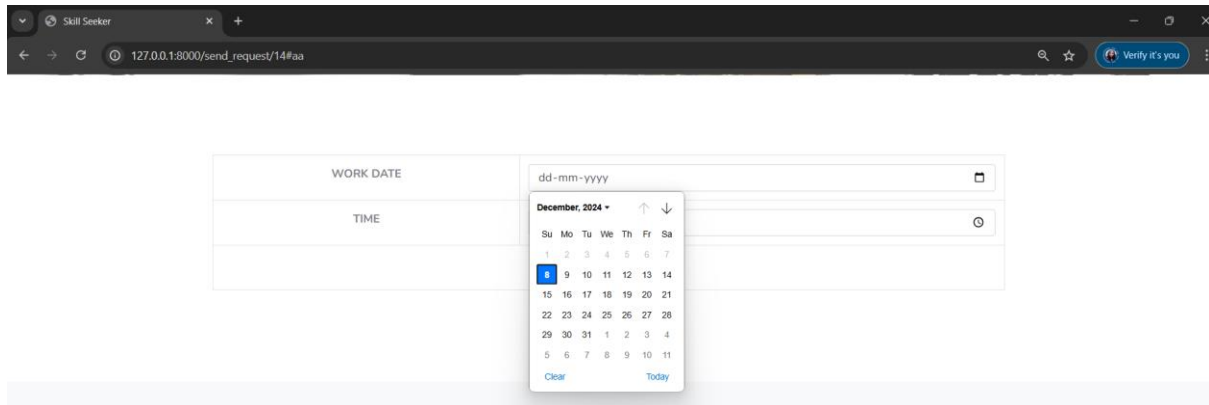


Figure 3 : date input while entering dates

## 4.9.2 Output Design

Output design focuses on presenting processed data in a meaningful, clear, and user-friendly manner. It ensures that the information generated by the system meets the needs of its users and supports decision-making. Effective output design emphasizes readability, relevance, and timeliness.

In my project, output design focuses on presenting information in a clear and visually appealing manner to enhance user experience. Key outputs include booking confirmations, tradesman profiles, request statuses, and payment bill, which are displayed in an organized and user-friendly format. For instance, dashboards were designed to show upcoming, ongoing, and completed bookings in separate sections for easy tracking. The request status page in my project is designed to display the statuses of all requests sent so far in a clear and organized manner. The layout ensures that users can quickly access and understand the status of their requests at a glance. Review and rating summaries are displayed using star ratings

and feedback comments to assist users in decision-making. The view tradesman page in my project is designed to showcase a list of all available tradesmen along with their essential details such as their name, location, address.

Some output forms in my project are:

- View category
- View tradesman
- View booking requests
- View review and reply
- View complaint
- View user profile

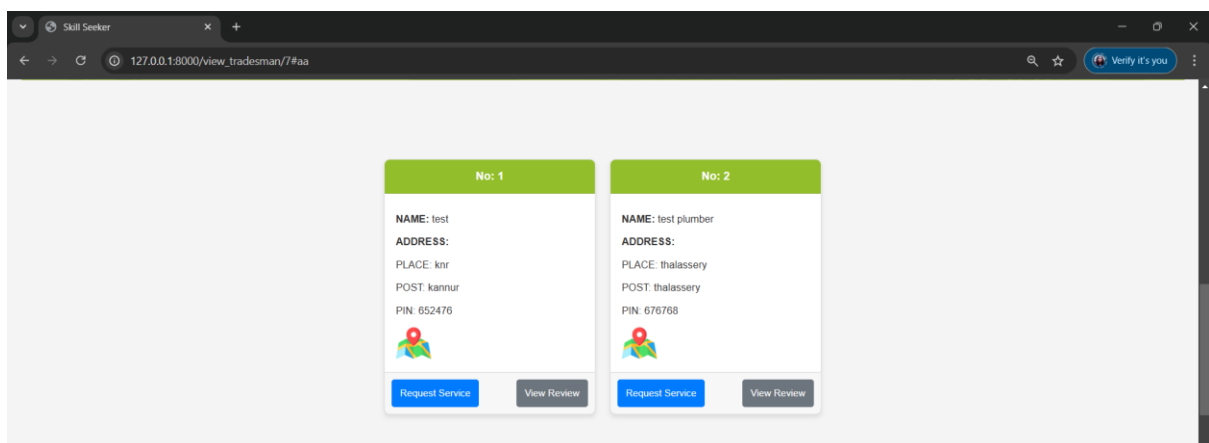


Figure 4: view tradesman page

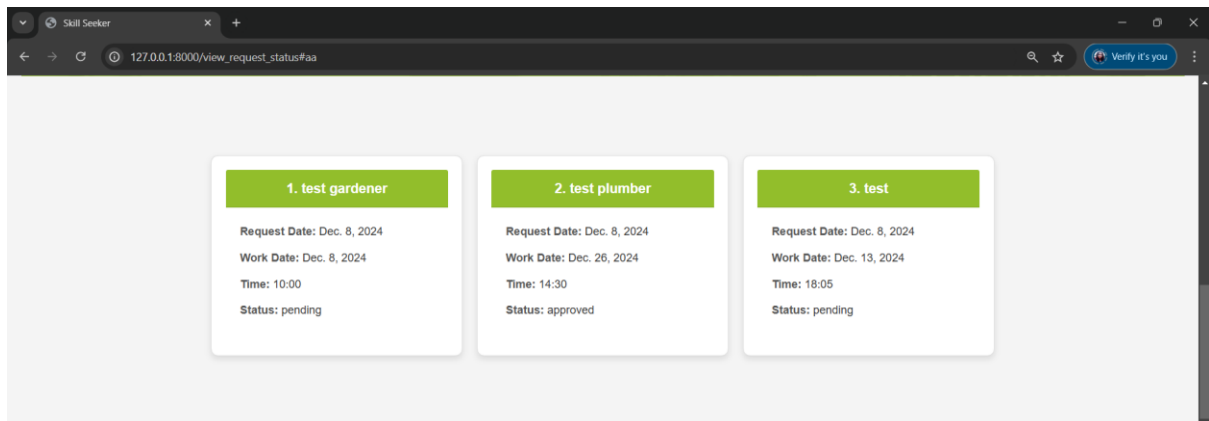


Figure 5: view request status page

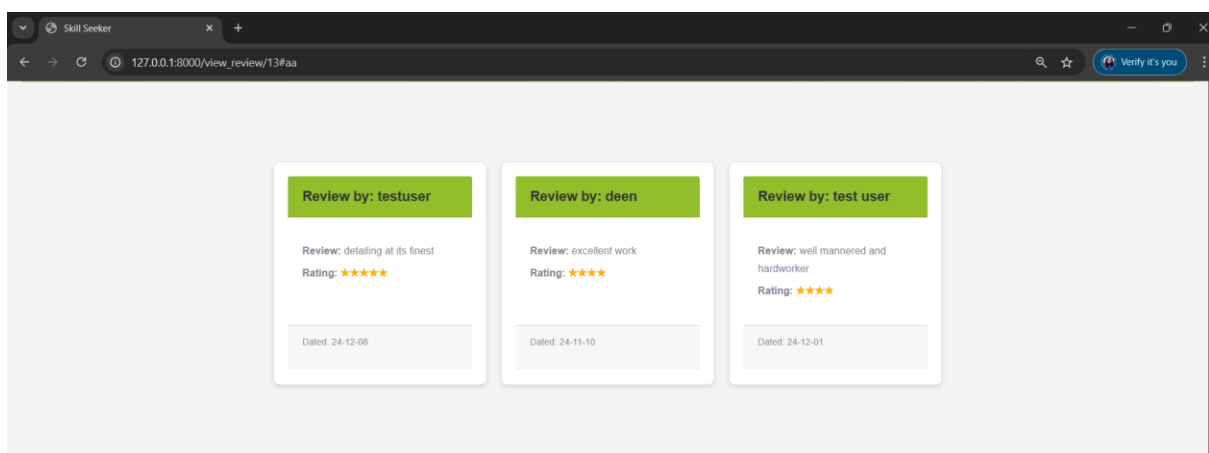


Figure 6: view review page

### 4.9.3 MENU DESIGN

Menu design refers to the process of organizing and presenting navigation options in a way that enhances user interaction and experience. It involves creating a visually appealing, intuitive, and functional structure for menus that guide users to desired content or actions.

In this project, menu design plays a vital role in helping users navigate between features like searching for tradesmen, managing profiles, viewing bookings, and accessing reviews. For example, the dropdown menu on the tradesman home page, built with Django and styled using CSS, provides a structured way for tradesman to access essential functions effortlessly.

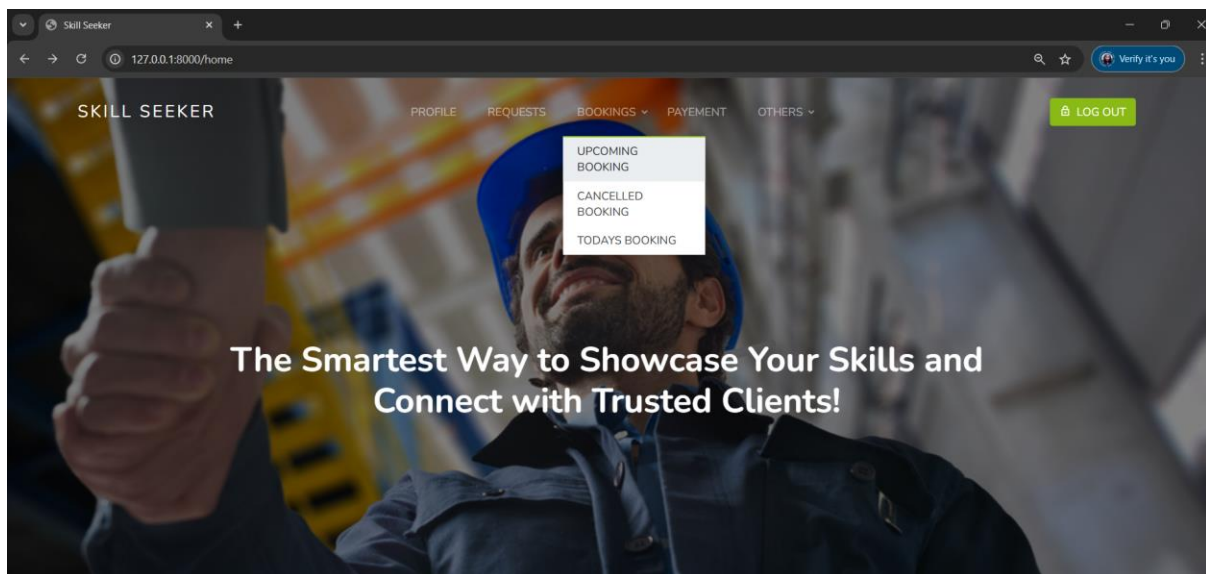


Figure 7: menu design in tradesman home page

## 4.10 CODE DESIGN

Code design refers to the process of planning and structuring the codebase of a software application to ensure it is efficient, maintainable, scalable, and aligned with the project's requirements. It involves making decisions about how the system's functionality will be implemented in code, including the choice of algorithms, data structures, and design patterns.

### High-Level Design

High-Level Design (HLD) provides a comprehensive overview of a system's architecture by defining its major modules and their interactions. It emphasizes the relationships between



components, the data flow, and the technologies used, without diving into implementation details. HLD serves as a blueprint that helps stakeholders and developers understand the structure of the system, ensuring alignment with requirements.

In the Skill Seeker project, the HLD is structured around core modules, each addressing specific functionalities of the platform. The modules include:

1. Customer Management - This module handles user registration, authentication, and profile management. It allows customers to create accounts, log in, view their profiles, change their passwords and manage their activity within the platform.
2. Tradesman Management - Designed for tradesmen, this module facilitates tradesman registration, authentication, and profile management. Tradesmen can update their profile and manage their activity within the platform. Admin can approve or reject registered tradesman and can also block or unblock approved tradesman.
3. Category Management - This module organizes services into well-defined categories, making it easier for users filter tradesmen based on specific job types or skills. It enables the admin to add new category when needed.
4. Request Management – It oversees the entire service booking process, acting as the central hub for coordination between customers and tradesmen. For customers, this module allows them to initiate a service request by specifying preferred date, and time. Customers can also view the status of their requests, whether pending, accepted, or rejected, and receive notifications when tradesmen respond to their requests.

For tradesmen, the module provides tools to view the details of each request, date, time and customer information. Based on their availability, they can choose to accept or reject the request. If a request is rejected, the module notifies the customer and allows them to search for other available tradesmen.

5. Review Management - Customers can leave reviews and star ratings for tradesmen after services are completed. This module displays reviews to help future users make informed decisions while selecting tradesmen.

6. Complaint Management - This module addresses user concerns. It allows users to raise complaints and receive resolutions, ensuring user satisfaction and platform credibility.
7. Payment Management – It enable the tradesman to update bill when the work is done. Customer can pay the bill through online transactions. It enables customers to pay tradesmen directly while ensuring transaction records are maintained.

The modules interact through a centralized backend (Django), with a HTML, CSS frontend handling user interactions and MySQL managing persistent data. This modular architecture ensures seamless functionality, scalability, and maintainability for the system.

### **Low-Level Design**

Low-Level Design (LLD) provides a detailed, technical blueprint of the system's components and their internal logic. It translates the high-level design into implementable code specifications by defining class structures, algorithms and database schemas.

For the Skill Seeker project, LLD focuses on the internal workings of each module and how they interact with the system's architecture. LLD for the key modules are:

#### **1. Customer Management**

Function: register\_customer(request)

Input: name, email, phone, pin, post, place, latitude, longitude, login\_id

Output: Success message or error message

Logic:

- Validate inputs (e.g., check if the email and phone number is unique).
- Store customer data in the customer table.
- Return success or error based on the outcome.

#### **2. Tradesman Management**

Function: view\_tradesman(request,id)

Input: category\_id

Output: Display all tradesman under the specified category

Logic:

- Fetch all tradesmen from the database who belong to the specified category\_id.
- If tradesmen are found, return the list of their details (e.g., name, rating, location).

### 3. Request Management

Function: send\_request(request,id)

Input: customer\_id, tradesman\_id, date, time, work\_date

Output: success message after successful booking

Logic:

- If a request for the same tradesman on the same day doesn't exist, a new entry in the request\_tb table is created with status "pending".
- If a request already exists, an error message is returned.

### 4. Category Management

Function: category\_add(request)

Input: category\_name

Output: Success or error message

Logic:

- Check if the category already exists.
- If it already exists, return an error message.
- Else the category is added.

### 5. Review Management

Function: send\_review\_rating(request,id)

Input: customer\_id, tradesman\_id, review, rating, date

Output: Success or error message

Logic:

- Insert the review and rating into the review table.

#### 6. Complaint Management

Function: send\_complaint (request, id)

Input: date, complaint, tradesman\_id, customer\_id

Output: Success or error message

Logic:

- Store date and complaint to the complaint table.
- Set the reply and reply to “pending” status.

#### 7. Payment Management

Function: update\_bill (request, id)

Input: date, amount, request\_id

Output: Success or error message

Logic:

- Store date and amount to the payment table.
- Set the method and status to “pending”.

## 4.11 Database design

Database design is the process of organizing data into a structured format that allows for efficient storage, retrieval, and management of information in a database system. A well-

designed database ensures data consistency, reduces redundancy, and facilitates scalability and usability.

### Key Components of a Database

1. Tables (Entities) - Tables are the core storage structures in a database. Columns define the attributes or properties of the entity (e.g., name, email,) and rows (Records) represent individual data entries in the table.
2. Fields - Fields are the specific columns in a table that represent the attributes of the entity. Example: In a User's table, fields might include User\_ID, Name, Email, and Password.
3. Relationships – They are the connections between tables based on common fields, enabling relational databases to model real-world associations. Types of relationships in databases are:
  - a) One-to-One (1:1): Each record in Table A corresponds to one record in Table B and vice versa. Example: A User table and a User\_profile table where each user has one profile.
  - b) One-to-Many (1:N): A single record in Table A can be associated with multiple records in Table B. Example: A Customer table and an Orders table where one customer can place multiple orders.
  - c) Many-to-Many (M:N): Multiple records in Table A can be associated with multiple records in Table B. Example: A Students table and a Courses table, connected via an Enrolments table.

### Normalization


Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity. It involves dividing large tables into smaller, related tables and defining relationships between them, often using keys. This ensures that each piece of data is

stored only once, minimizing inconsistencies and simplifying updates. Normalization is carried out through a series of steps called "normal forms". They are:

- 1NF (First Normal Form): Ensures that each column contains atomic values and every row is unique.
- 2NF (Second Normal Form): It eliminates partial dependency. It ensures all non-key attributes are fully dependent on the entire primary key.
- 3NF (Third Normal Form): It eliminates transitive dependency and ensures that non-key attributes depend only on the primary key.
- 


The database schema for this project is designed using MySQL to support functionalities such as user registration, tradesman profiles, job searching, bookings, reviews, and payments.

#### Tradesman table:

	Field	Type	Comment
	id	int(11) NOT NULL	
	name	varchar(200) NOT NULL	
	email	varchar(200) NOT NULL	
	phone	varchar(200) NOT NULL	
	pin	varchar(200) NOT NULL	
	post	varchar(200) NOT NULL	
	place	varchar(200) NOT NULL	
	latitude	varchar(200) NOT NULL	
	longitude	varchar(200) NOT NULL	
	CATEGORY_id	int(11) NOT NULL	
	LOGIN_id	int(11) NOT NULL	
	status	varchar(200) NOT NULL	


Purpose: Stores information of all tradesman.

#### Customer table:

	Field	Type	Comment
	id	int(11) NOT NULL	
	name	varchar(200) NOT NULL	
	email	varchar(200) NOT NULL	
	phone	varchar(200) NOT NULL	
	pin	varchar(200) NOT NULL	
	post	varchar(200) NOT NULL	
	place	varchar(200) NOT NULL	
	latitude	varchar(200) NOT NULL	
	longitude	varchar(200) NOT NULL	
	LOGIN_id	int(11) NOT NULL	


Purpose: Stores information for all customer.

#### Login table:

	Field	Type	Comment
	id	int(11) NOT NULL	
	username	varchar(200) NOT NULL	
	password	varchar(200) NOT NULL	
	usertype	varchar(200) NOT NULL	


Purpose: Manage and authenticate user's access to the system

#### Category table:

	Field	Type	Comment
	id	int(11) NOT NULL	
	category_name	varchar(200) NOT NULL	


Purpose: Stores all available categories

#### Complaint table:

	Field	Type	Comment
	id	int(11) NOT NULL	
	date	varchar(200) NOT NULL	
	complaint	varchar(200) NOT NULL	
	CUSTOMER_id	int(11) NOT NULL	
	TRADESMAN_id	int(11) NOT NULL	
	reply	varchar(200) NOT NULL	
	reply_date	varchar(200) NOT NULL	


Purpose: Stores all complaints and its replies

### Payment table

	Field	Type	Comment
	id	int(11) NOT NULL	
	date	varchar(200) NOT NULL	
	amount	varchar(200) NOT NULL	
	status	varchar(200) NOT NULL	
	REQUEST_id	int(11) NOT NULL	
	method	varchar(200) NOT NULL	

Purpose: Stores all payment details


### Request\_tb table:

	Field	Type	Comment
	id	int(11) NOT NULL	
	date	date NOT NULL	
	time	varchar(200) NOT NULL	
	status	varchar(200) NOT NULL	
	CUSTOMER_id	int(11) NOT NULL	
	TRADESMAN_id	int(11) NOT NULL	
	work_date	date NOT NULL	
	updated_date	date NOT NULL	

Purpose: Stores all booking request details



**Review table:**

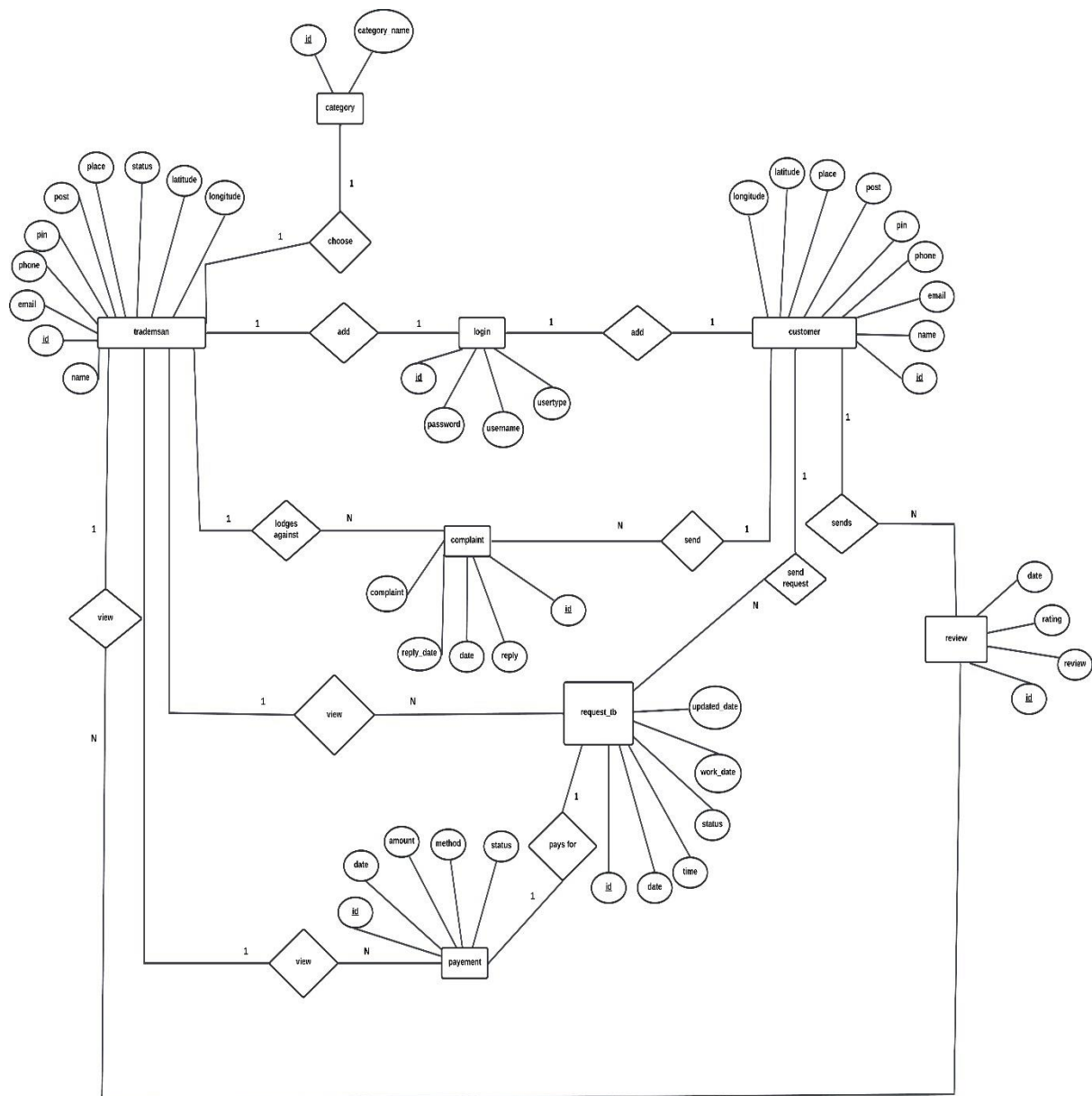
	Field	Type	Comment
	id	int(11) NOT NULL	
	date	varchar(200) NOT NULL	
	review	varchar(200) NOT NULL	
	CUSTOMER_id	int(11) NOT NULL	
	TRADESMAN_id	int(11) NOT NULL	
	rating	varchar(200) NOT NULL	

Purpose: Stores all review details

## 4.12 ER diagram

An Entity-Relationship (ER) Diagram is a visual representation of the structure of a database. It illustrates the entities (tables), their attributes (columns), and the relationships between these entities.

ER diagram for the Skill Seeker project is as follows:



### 4.13 Validation and checks

#### Admin:

SL No	PROCESS	INPUT	EXPECTED OUTPUT	OBTAINED OUTPUT	REMARKS
1.	Login	Username, password	Login Successful / Error in Login	Log in Successful	Success
2.	Add category	Category name	Added successfully / Failed to add	Added successfully	Success
3.	Approve tradesman	Tradesman id	Approved successfully / Failed to approve	Approved successfully	Success
4.	Reject tradesman	Tradesman id	Rejected successfully / Failed To reject	Rejected successfully	Success
5.	Reply to complaint	Complaint id	Reply send successfully / Failed To send reply	Reply send successfully	Success

**Customer:**

SL No	PROCESS	INPUT	EXPECTED OUTPUT	OBTAINED OUTPUT	REMARKS
1.	Send booking request	Tradesman id, work date, time	Request send successfully/ failed to send request	Request send successfully	Success
2.	Cancel booking	Booking request id	Cancelled successfully/ failed to cancel	Cancelled successfully	Success
3.	Send review and rating	Tradesman id, review, rating	Send successfully/ Failed to send	Review and rating send successfully	Success
4.	Send complaint	Tradesman id, complaint	Send successfully / Failed to send	Complaint send successfully	Success
5.	View tradesman	Category id	Tradesmen viewed successfully / Failed to retrieve tradesman	Tradesmen viewed successfully	Success

**Tradesman:**

SL No	PROCESS	INPUT	EXPECTED OUTPUT	OBTAINED OUTPUT	REMARKS
1.	View booking requests	Tradesman id	Viewed Successfully / Failed To retrieve requests	Booking requests viewed successfully	Success
2.	Approve booking requests	Booking request id	Approved successfully / Failed to approve	Approved successfully	Success
3.	Reject booking requests	Booking request id	Rejected successfully / Failed to reject	Rejected successfully	Success
4.	Cancel booking requests	Booking request id	Cancelled successfully / Failed to cancel	Cancelled successfully	Success
5.	Update bill	Customer id, bill amount	Bill updated successfully / Failed To update bill	Bill updated successfully	Success

## 5.0 IMPLEMENTATION OF SYSTEM AND TESTING

The implementation of the Skill Seeker project involves integrating its frontend, backend, and database to create a seamless platform. The various activities involved are:

### 1. Coding

The actual code for the Skill Seeker platform is written, implementing business logic, designing interfaces, and setting up databases to ensure the system functions as per the defined specifications. The development strictly adheres to the design documentation created in the earlier phase, ensuring accuracy and consistency. Key tasks include coding backend logic in Django, building front end and integrating payment system for seamless transactions.

### 2. Database Setup

The MySQL database is created and configured according to the design specifications. This involves creating tables for users, tradesmen, bookings, reviews, and payment records, along with defining relationships between them. The database schema is optimized for efficient data retrieval, storage, and processing, ensuring it meets the functional requirements of the platform. Proper indexing and normalization are implemented to maintain data integrity and performance.

### 3. Interface Development

The frontend of the Skill Seeker platform is developed using HTML, CSS and javascript. This step includes creating responsive web pages, forms, dropdown menus, and navigation features to ensure a user-friendly and intuitive interface. CSS ensures the design is visually appealing and consistent. The interface aligns with the earlier design to provide a cohesive and engaging user experience.

### 4. Configuration

The deployment environment is configured to ensure the platform runs smoothly. This includes setting up the WAMP Server for local hosting during development, configuring network settings, security protocols, and user access permissions. The system is tested in a controlled environment to identify and resolve any configuration-related issues, ensuring readiness for deployment.

## 5. Deployment Preparation

Once development and testing are complete, the system is prepared for deployment. This includes finalizing deployment scripts, conducting pre-deployment testing, and ensuring a smooth transition to the production environment. Necessary checks are performed to validate the system's stability, scalability, and usability, ensuring Skill Seeker is fully operational and ready for users.

## Software testing

Software testing is the process of evaluating a software application to ensure that it meets the required specifications and works as intended. Testing helps identify defects or issues in the software, ensuring reliability, performance, and quality before deployment. It ensures that the software meets the desired quality standards and performs efficiently under various conditions. Testing ensures that the software meets user expectations and requirements, increasing customer satisfaction and trust in the product.

Different types of testing are:

### Unit Testing

Unit testing focuses on verifying individual components of a software application in isolation to ensure that each unit behaves as expected. These units can be functions, methods, or classes. This testing phase is crucial for identifying bugs early in the development cycle, reducing the cost and complexity of fixing defects. In the Skill Seeker project, unit testing will involve testing individual components such as the customer management, booking module, or the review module in isolation. For example, when

testing the customer management module, we ensure that attributes like phone number, and email id are validated and stored correctly.

### **Integration Testing**

Integration testing checks how different modules or components of an application interact with each other. The goal is to identify issues in the interaction between units that may not surface during unit testing. This could involve testing database interactions, or inter-module communications. Integration testing often follows unit testing in the software development lifecycle.

For this project, integration testing validates interactions between modules. For example, after a user registers, the system should correctly update the database and allow the user to log in. Another example is testing the interaction between the category management module, booking module, and database during the booking of a tradesman by the customer.

### **Validation testing**

Validation testing ensures that the developed software meets the specified requirements and fulfils the intended purpose. It is conducted after integration testing and often involves stakeholder feedback. This type of testing answers the question, Are we building the right product?

In Skill Seeker project, validation testing will confirm that users can search for tradesmen by location and category, book appointments, and process payments as expected. Another example is Testing the user registration system to ensure that users can register successfully with valid data (e.g., name, email, password, place, pin code, post, phone number) and receive appropriate error messages for invalid inputs. For instance, entering valid details should register the user, while missing or incorrect inputs should display error messages.

### **Black Box Testing**

Black box testing focuses on the functionality of an application without considering its internal code or structure. Testers interact with the application's user interface and check outputs against expected outcomes. It is ideal for ensuring that the system behaves as a user

expects, regardless of how it is implemented.

For Skill Seeker project, black box testing would involve checking the booking functionalities, ensuring that a customer searching for tradesmen gets accurate results and can book appointments seamlessly. For example, submitting a form with a valid trade category should return the correct tradesman profiles without errors.

### **White Box Testing**

White box testing involves examining an application's internal structure, logic, and code. Testers use their knowledge of the codebase to design test cases that ensure thorough coverage, such as testing loops, conditions, and data flows. It is essential for ensuring the internal code adheres to design principles and performs efficiently.

In this project, white box testing would focus on critical backend code, such as the logic for validating user input during registration. Another example is testing the tradesman search system to ensure it accurately retrieves tradesmen based on the trade category. In this, the function that processes user inputs is tested to ensure it generates the correct SQL query and retrieves only matching tradesmen.



## 6.0 CONCLUSION

The Skill Seeker project successfully fulfils its objective of connecting property owners with skilled tradesmen in an efficient and user-friendly manner. By using modern technologies such as Django, MySQL, and WAMP Server, the platform provides an intuitive interface for users to search for tradesmen, book services, and make secure payments.

The system offers flexibility to both customers and tradesmen in the booking process. Customers can select tradesmen, dates, and times based on their preferences, while tradesmen have the option to accept or reject booking requests. This mutual control ensures a seamless and adaptable experience, catering to the needs of both parties effectively.

The system also allows tradesmen to manage their profiles and handle bookings while incorporating a feedback mechanism to ensure quality service through user reviews and ratings. The inclusion of a complaint management system and a payment system ensures the platform is comprehensive and addresses key user needs.

The successful implementation of Skill Seeker demonstrates how technology can bridge the gap between service providers and property owners, creating a solution that enhances convenience and trust. Future enhancements could include extending the system to mobile platforms and integrating additional features, ensuring the platform continues to evolve and cater to user needs effectively.

## 7.0 SUGGESTION FOR FUTURE WORK

By providing a flexible and user-friendly platform, the Skill Seeker system has been designed to address the challenges of connecting property owners with skilled tradesmen. While the system successfully achieves its primary goals, I recognize that there is always room for improvement and growth. The current version of the system serves as a strong foundation, but there is significant potential to enhance functionality and expand its scope to better serve users.

One major improvement could be the inclusion of a real-time chat feature. This would allow customers and tradesmen to communicate directly, making it easier to discuss job details, share updates, and address specific requirements before finalizing bookings.

Another valuable addition would be enabling tradesmen to upload images and descriptions of their previous work. This feature would allow tradesmen to showcase their skills and build trust with potential customers, who can evaluate the quality of work before making a selection. It would also serve as a portfolio for tradesmen to attract more clients.

Developing a dedicated mobile application for the platform could significantly increase accessibility and user engagement. A mobile app would make it more convenient for users to search for tradesmen, manage bookings, and communicate on the go, providing a seamless experience across devices.

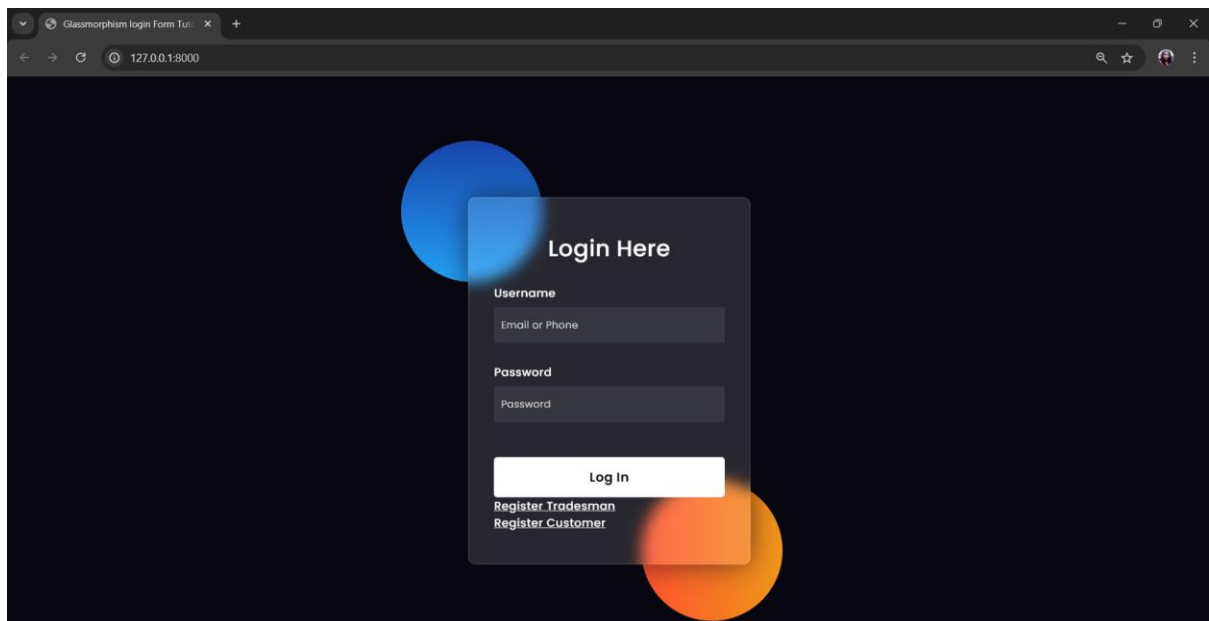
These features would further improve the platform's utility and reach, making it a more comprehensive and user-friendly solution.

## 8.0 BIBLIOGRAPHY

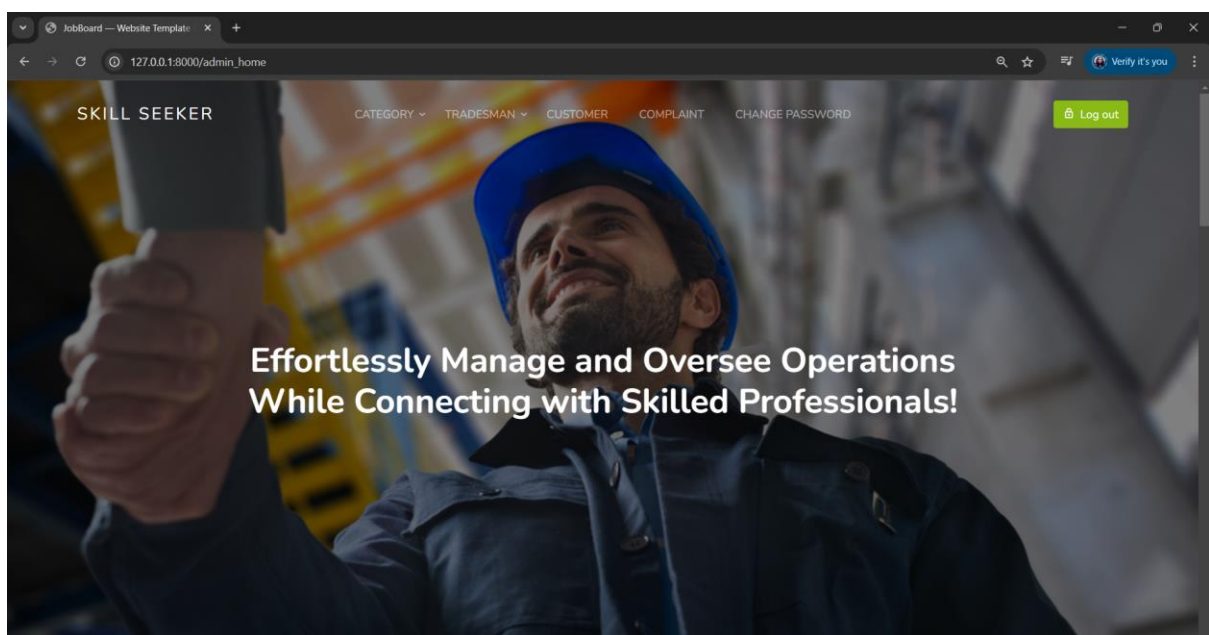
- Django Software Foundation. (2024). *Django documentation*. Django.  
<https://www.djangoproject.com/>
- MySQL. (2024). *MySQL Documentation*. MySQL. <https://dev.mysql.com/doc/>
- Pressman, R., & Maxim, B. (2023). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill.
- Stack Overflow. (2022). *How would you create a 'manual' Django migration?*. Stack Overflow.  
<https://stackoverflow.com/questions/50320989/how-would-you-create-a-manual-django-migration>
- WampServer. (2024). *WampServer*. StackOverflow.  
<https://www.wampserver.com/en/>
- Niederst Robbins, J. (2006). *Web design in a nutshell*. O'Reilly Media.
- Checkatrade. (2024). *Find a guaranteed tradesperson*. Checkatrade. Retrieved from <https://www.checkatrade.com/>
- Urban Company. (2024). *Home services at your doorstep*. Urban Company. Retrieved from <https://www.urbancompany.com/kochi>

## 9.0 APPENDICES

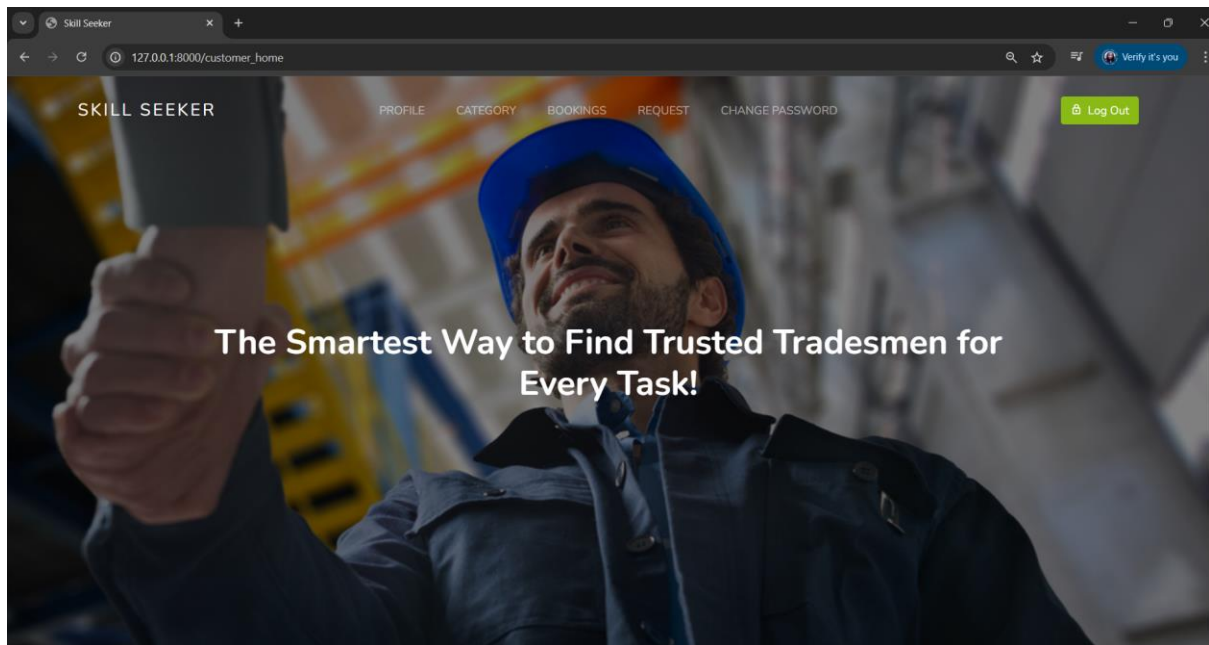
### Login page:



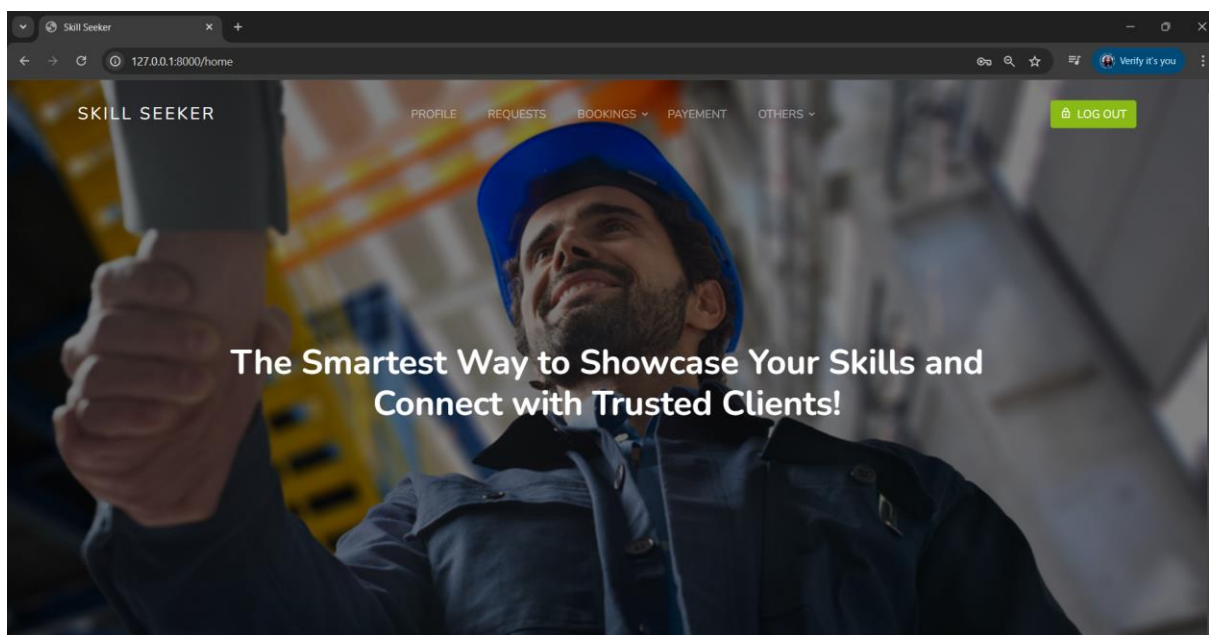
### Admin home page:

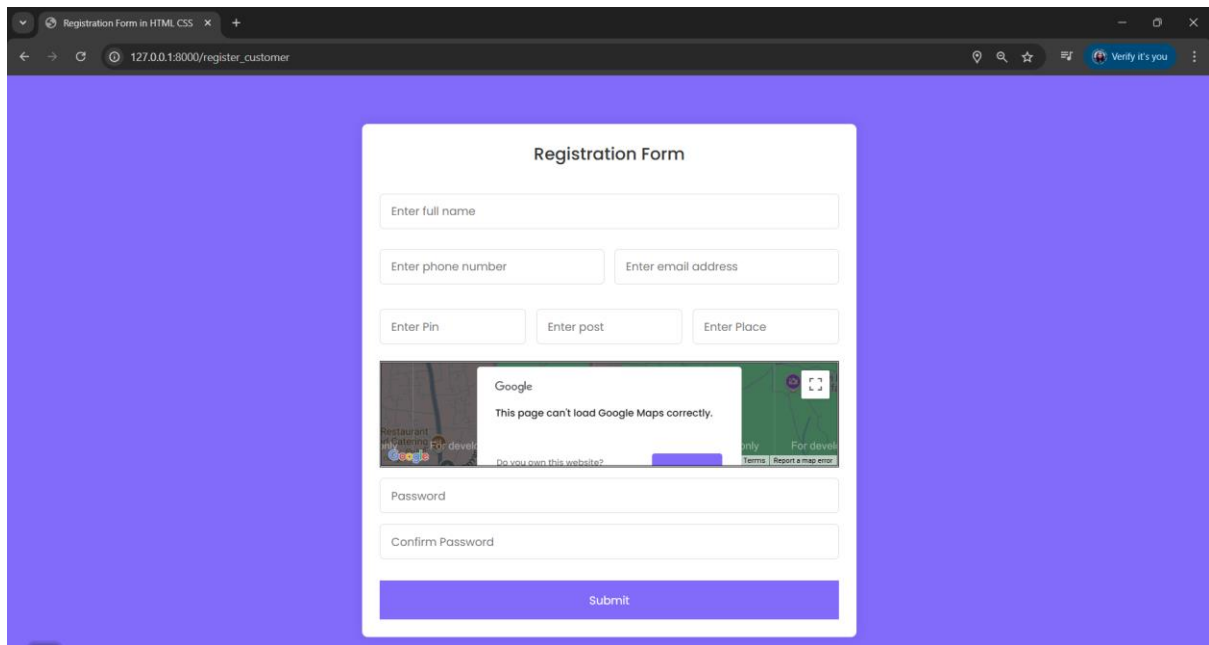


### Customer home page:

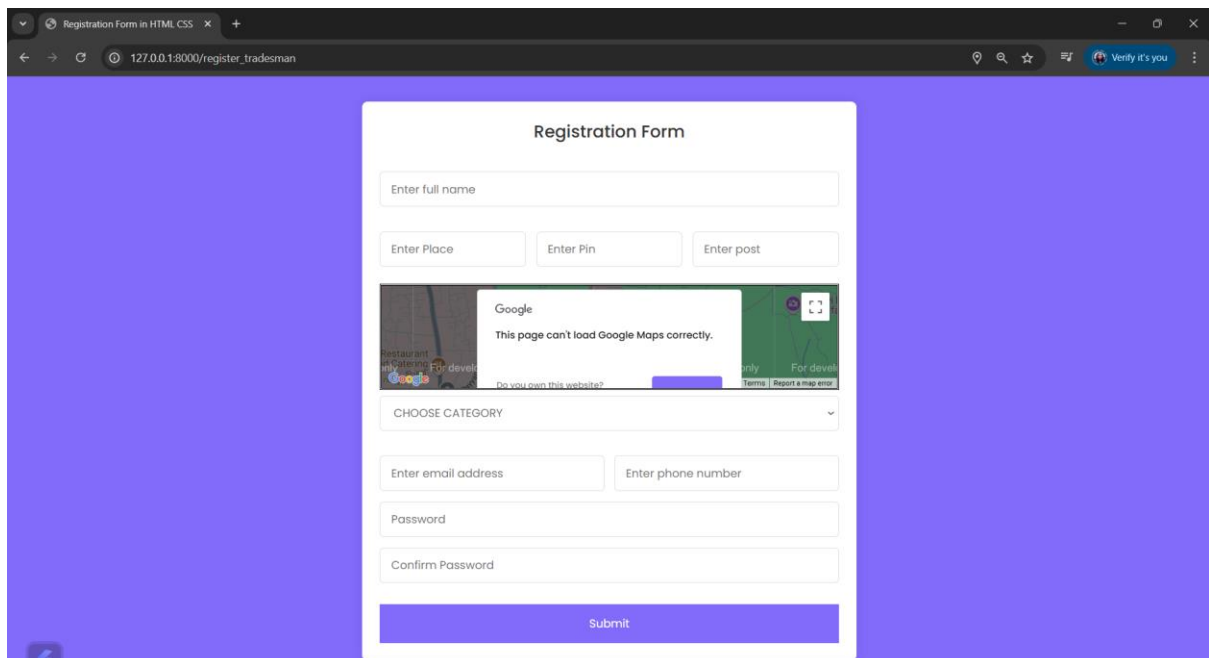


### Tradesman home page:

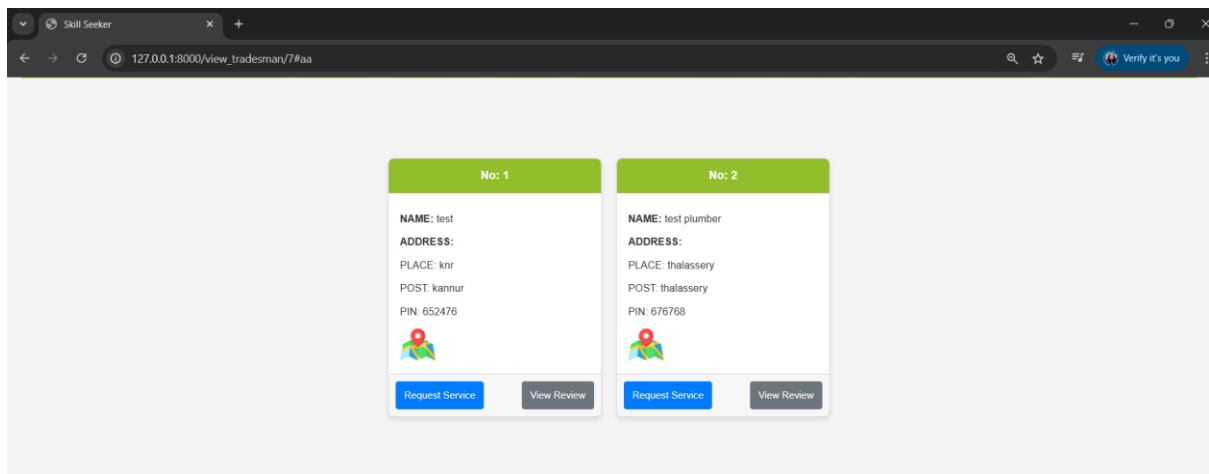
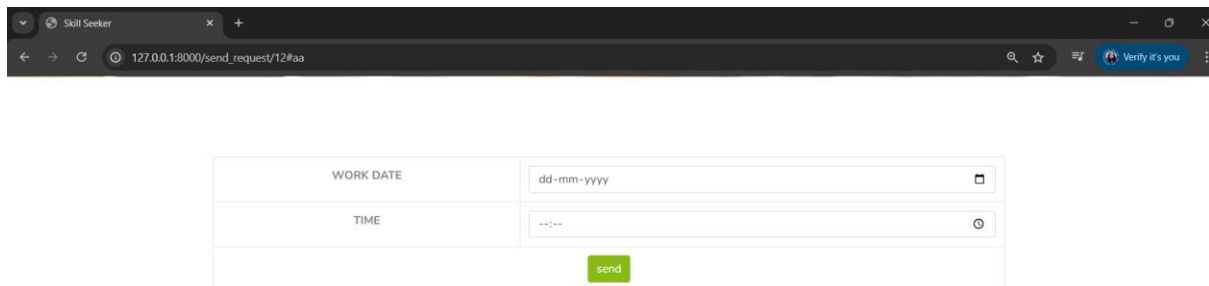
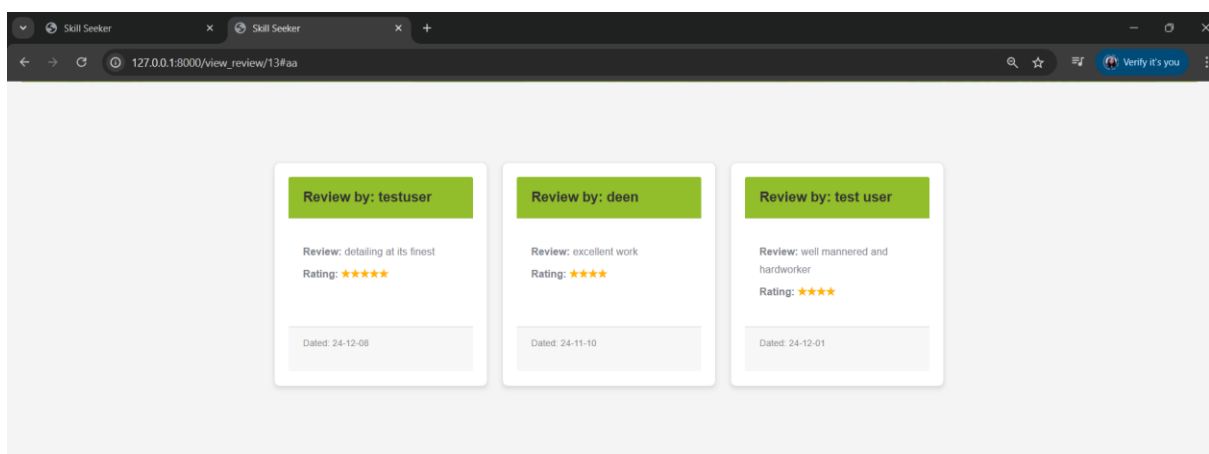


**Customer registration page:**

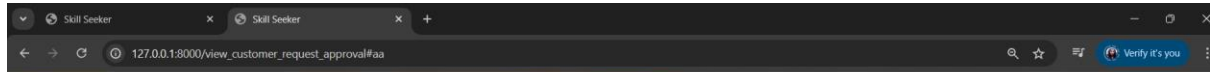
The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/register\_customer". The page has a purple background. In the center is a white "Registration Form" box. The form contains the following fields: "Enter full name", "Enter phone number", "Enter email address", "Enter Pin", "Enter post", "Enter Place", a Google Maps placeholder with an error message "This page can't load Google Maps correctly.", "Password", "Confirm Password", and a blue "Submit" button at the bottom.

**Tradesman registration page:**

The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/register\_tradesman". The page has a purple background. In the center is a white "Registration Form" box. The form contains the following fields: "Enter full name", "Enter Place", "Enter Pin", "Enter post", a Google Maps placeholder with an error message "This page can't load Google Maps correctly.", a "CHOOSE CATEGORY" dropdown menu, "Enter email address", "Enter phone number", "Password", "Confirm Password", and a blue "Submit" button at the bottom.

**View tradesman:****Send booking request page:****View review:**

### Customer request approval or reject:



**Request #1**

Date: Dec. 8, 2024

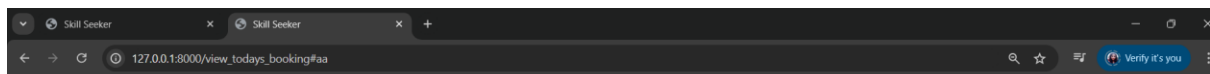
Time: 10:00

Work Date: Dec. 8, 2024

Customer: testuser

ApproveReject

### View todays booking:



**Booking #1**

Time: 10:00

Customer: arya

Status: completed

Amount: 2000

Work Completed,  
Payment Pending