**Helping Hands**

***Project submitted in partial fulfilment of the requirements  
 for the award of the degree of***

Master of Computer Applications

***Semester 4***

**January- May, 2025**

|  |  |
| --- | --- |
| **Submitted by:**  **Krishnpal Singh Bhadoriya**  **2011114**  **Rishit Chouhan**  **2011125** | **Under the guidance of:**  **Mr. Mohit Kumar Varma** |

******

**School of Computer Science & IT   
Devi Ahilya Vishwavidyalaya, Indore, (MP) - 452017, India**

**School of Computer Science & IT   
Devi Ahilya Vishwavidyalaya, Indore, (MP) - 452017, India**

**DECLARATION**

I hereby declare that the project titled “**Helping Hands**” submitted by me for the partial fulfilment of the requirement for the award of Master of Computer Applications to School of Computer Science & IT, Devi Ahilya Vishwavidyalaya, Indore, comprises my own work and due acknowledgement has been made in text to all other material used.

Signature of Student:

Date:

Place:

**School of Computer Science & IT   
Devi Ahilya Vishwavidyalaya, Indore, (MP) - 452017, India**

**CERTIFICATE FROM GUIDE**

It is to certify that dissertation on **“Helping Hands”,** submitted by Mr. Krishnpal Singh Bhadoriya : 2011114 and Mr. Rishit Chouhan: 2011125 to the School of Computer Science & IT, Devi Ahilya Vishwavidyalaya, Indore has been completed under my supervision and the work is carried out and presented in a manner required for its acceptance in partial fulfilment for the award of the degree of Master of Computer Applications.

Project Guide

Signature:

Name:

Date:

**School of Computer Science & IT   
Devi Ahilya Vishwavidyalaya, Indore, (MP) - 452017, India**

**APPROVAL CERTIFICATE FROM EXAMINER(S)**

It is to certify that we have examined the dissertation on **“Helping Hands”**, submitted by Mr. Krishnpal Singh Bhadoriya : 2011114 and Mr. Rishit Chouhan: 2011125 to the School of Computer Science & IT, Devi Ahilya Vishwavidyalaya, Indore and hereby accord our approval of it as a study carried out and presented in a manner required for its acceptance in partial fulfilment for the award of the degree of Master of Computer Applications.

|  |  |
| --- | --- |
| **Internal Examiner** | **External Examiner** |
| Signature:  Name:  Date: | Signature:  Name:  Date: |

**School of Computer Science & IT   
Devi Ahilya Vishwavidyalaya, Indore, (MP) - 452017, India**

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to all those who supported and guided us throughout the development of our final year project, **Helping Hands**.

First and foremost, we extend our heartfelt thanks to our respected mentor, **Mr. Mohit Varma**, for his continuous guidance, encouragement, and valuable insights, which played a vital role in the successful completion of this project. His expertise and constant motivation kept us focused and inspired.

We are also grateful to our college faculty and the Department of Computer Science for providing the necessary resources, infrastructure, and a platform to carry out this project.

A special thanks to our teammates:

* **Rishit Chauhan** – for leading the frontend development using React.js, ensuring a responsive and user-friendly interface.
* **Krishnpal Singh Bhadoriya** – for handling the backend development using Spring Boot and integrating the database and APIs effectively.

Last but not least, we thank our families and friends for their unwavering support, patience, and encouragement throughout this journey.

This project has been a great learning experience and would not have been possible without the combined efforts of everyone mentioned above.

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Chapter Name** | **Page No.** |
|  | **List of Figures** | **1** |
|  | **List of Tables** | **2** |
| **1.** | **INTRODUCTION** | **4** |
|  | 1.1 Background  1.2 Purpose of the Project  1.3 Project Scope  1.4 Project Objectives |  |
| **2.** | **PROJECT PLANNING AND SCHEDULING** |  |
|  | 2.1 Project Plan  2.2 Work Breakdown Structure  2.3 Gantt Chart  2.4 PERT Chart/ CPM  2.5 Team Structure and Responsibilities  2.6 Project Development Methodology  2.7 Hardware and Software Requirements |  |
| **3.** | **SYSTEM ANALYSIS** |  |
|  | 3.1 Problem Description  3.1.1 Problem Definition  3.1.2 Proposed Solution  3.2 Requirements  3.2.1 Functional Requirements  3.2.2 Non-Functional Requirements  3.3 Problem Analysis Diagrams  3.3.1 Data Flow Diagram/ Process Flow Diagram  3.3.2 Use Case Diagram and Sequence Diagram  3.4 Database Schema |  |
| **4.** | **SYSTEM DESIGN** |  |
|  | 4.1 System Architecture  4.2 Physical Design  4.2.1 Structure Chart  4.2.2 ER Diagram  4.2.3 Class Diagram and Object Diagram  4.3 Input and Output Design  4.4 Algorithmic Design |  |
| **5.** | **IMPLEMENTATION** |  |
|  | 5.1 Integration of Modules/ Files |  |
| **6.** | **SYSTEM TESTING** |  |
|  | 6.1Test Case Design  6.1.1Unit Testing  6.1.2Integration Testing  6.1.3System Testing  6.1.4Acceptance Testing  6.2Specific System Testing  6.3Test Reports |  |
| **7.** | **CONCLUSION OF THE PROJECT** |  |
|  | 7.1Results  7.2Conclusion  7.3Limitations of the project  7.4Future Work  7.5Lessons Learned |  |
|  | **REFERENCES** |  |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Table Name** | **Page No.** |
| 1. | Gantt Chart |  |
| 2. | Pert Chart |  |
| 3. | Context Level DFD |  |
| 4. | First Level DFD |  |
| 5. | Use Case Diagram |  |
| 6. | Sequence Diagram |  |
| 7. | System Architecture |  |
| 8. | Structure Chart |  |
| 9. | ER Diagram |  |
| 10. | Class Diagram |  |
| 11. | Object Diagram |  |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Table Name** | **Page No.** |
| 1. | Team Structure and Responsibilities |  |
| 2. | Hardware Requirements |  |
| 3. | Software Requirements |  |
| 4. | Input Design |  |
| 5. | Output Design |  |
| 6. | Algorithmic Design |  |
| 7. | Integration of Modules |  |
| 8. | Test Report |  |

**ABSTRACT**

In today's fast-paced urban lifestyle, the demand for reliable, on-demand home services such as plumbing, electrical repairs, beauty treatments, and appliance servicing is growing rapidly. However, users often struggle to find trusted professionals due to the unorganized nature of the service industry.”**Helping Hands”** is a web-based platform designed to address this issue by connecting customers with verified and skilled service providers efficiently and transparently.

The project aims to build a scalable and user-friendly application using React.js for the frontend and Spring Boot for the backend, with MySQL as the database. The system will support secure user registration, service browsing, real-time booking, online payment, and review features. Service providers will be able to manage their profiles, accept jobs, and track payments through a dedicated dashboard. An admin panel will oversee all activities to ensure service quality and resolve disputes.

By bridging the gap between demand and supply of local services, Helping Hands seeks to enhance convenience for users while providing income opportunities for service professionals, ultimately transforming the traditional home services market into a digitized and efficient ecosystem.

**CHAPTER 1**

**INTRODUCTION**

* 1. **Background**

In today's fast-paced world, finding reliable and skilled professionals for home services can be a challenge***.* Helping Hands** is a comprehensive home services platform designed to bridge this gap by connecting customers with trained and verified professionals for a wide range of services. These include beauty and wellness, home repairs, cleaning, and appliance servicing.

The platform offers a convenient and efficient way for users to book on-demand services through its website. With a strong emphasis on quality and customer satisfaction, **Helping Hands** ensures that all professionals undergo thorough background verification and training. Additionally, they are equipped with high-quality tools and products to deliver superior service. By leveraging technology and a customer-centric approach, Helping Hands aims to redefine the home services industry, making professional assistance more accessible, trustworthy, and hassle-free.

* 1. **Purpose of the Project**

Helping Hands is designed to bridge the gap between customers and skilled professionals by offering a convenient, reliable, and efficient home services platform. It leverages technology to streamline service booking, ensuring that users receive high-quality assistance for various household needs.

**Key Objectives:**

1. **Simplified Access to Home Services**
   * Helping Hands serves as a one-stop solution for multiple home services, including beauty and wellness, cleaning, home repairs, and appliance servicing.
   * Instead of searching for individual service providers, customers can find and book professionals in one place, saving time and effort.
2. **Enhanced Customer Convenience**
   * The platform provides an easy-to-use website where users can browse available services, check pricing, and schedule appointments at their convenience.
   * With a few clicks, customers can book services without the hassle of making multiple calls or negotiating prices.
3. **Reliable and High-Quality Service**
   * All professionals are carefully vetted, background-verified, and trained to ensure they deliver top-notch service.
   * The use of high-quality tools and products ensures that customers receive premium service every time.
4. **Transparency and Trust**
   * Customers have access to clear and upfront pricing, eliminating hidden costs and surprises.
   * Verified customer reviews and ratings help users make informed decisions when booking services.
5. **Time and Cost Efficiency**
   * Helping Hands eliminates the need for customers to search for independent service providers, negotiate prices, or worry about quality.
   * By optimizing service delivery, it helps professionals maximize their earnings while providing cost-effective solutions to customers.
   1. **Project Scope**

 **Service Categories** – The platform will offer multiple service categories, including:

* Beauty and wellness (haircuts, spa, salon services)
* Home cleaning (deep cleaning, pest control)
* Home repairs (plumbing, electrical work, carpentry)
* Appliance servicing (AC repair, washing machine service)
* Other household services as required

 **User Management**

* Customer registration and profile management
* Professional registration, verification, and profile setup
* Admin panel for managing services, pricing, and customer queries

 **Service Booking System**

* Search and filter options for different services
* Booking and scheduling services based on availability
* Order tracking and service status updates

 **Reviews and Ratings**

* Customers can rate and review professionals.
* Feedback system to ensure quality improvement.

**1.4 Project Objective**

The primary objective of the Helping Hands project is to develop a user-friendly, reliable, and scalable home service platform that connects customers with skilled service professionals. The system is designed to simplify the process of booking household services while ensuring transparency, convenience, and satisfaction for all users. The key objectives of this project are as follows:

1. **Provide Seamless Service Booking Experience:**

Enable users to easily browse, select, and schedule a wide range of home services such as salon at home, appliance repair, pest control, and house cleaning.

1. **Implement Real-time Availability and Scheduling:**

Allow users to choose available dates and time slots for services, enhancing scheduling efficiency and minimizing booking conflicts.

1. **Ensure Secure User Authentication and Authorization:**

Integrate user login and role-based access controls to protect user data and ensure that service providers and customers have appropriate permissions.

1. **Facilitate Reliable Payment Processing:**

Integrate a secure payment gateway (Stripe) to handle online transactions smoothly and safely.

1. **Maintain a Scalable and Maintainable System Architecture:**

Use a modular architecture with React.js for frontend, Spring Boot for backend, and MySQL for database to ensure scalability and ease of maintenance.

1. **Enable Feedback and Review Mechanism:**

Allow users to rate services and provide feedback to improve the quality of service delivery and customer satisfaction.

1. **Promote Admin-Level Control and Monitoring:**

Provide admin functionalities to manage users, services, and bookings, and to monitor platform performance and service quality.

1. **Enhance Accessibility and Usability:**

Design a clean, responsive, and intuitive user interface using Tailwind CSS and Material UI for enhanced user experience across devices.

**CHAPTER 2**

**PROJECT PLANNING AND SCHEDULING**

**2.1 Project Plan**

The **Helping Hands** project was planned across the following phases to ensure timely and structured execution:

* **Requirement Analysis**: Understanding user needs and defining system scope.
* **System Design**: Designing the user interface, backend architecture, and database schema.
* **Development**: Building the frontend using React.js and backend with Spring Boot.
* **Testing**: Conducting unit, integration, and user acceptance testing.
* **Deployment**: Hosting the application and database, configuring services.
* **Documentation and Closure**: Preparing reports and presenting the final project.

**2.2 Work Breakdown Structure**

**(i) Project Initiation**

1. Requirement Gathering
2. Feasibility Analysis
3. Project Proposal & Approval

**(ii) Project Planning**

1. Project Scheduling
2. Resource Allocation
3. Risk Analysis
4. Team Roles Assignment

**(iii) System Design**

1. UI/UX Design
2. Database Design (ER Diagram, Schema)
3. Architecture Design (React.js + Spring Boot)
4. Flow Diagrams (DFD, Use Case)

**(iv) Frontend Development (React.js)**

1. Page Layout & Navigation
2. Service Display Module
3. Booking Functionality
4. Order Tracking UI
5. Responsive Design

**(v) Backend Development (Spring Boot)**

1. Server Setup
2. REST API Development
3. Authentication & Authorization
4. Business Logic & Order Handling

**(vi) Database Integration (MySQL)**

1. Schema Design
2. Data Storage & Retrieval
3. Admin CRUD Operations

**(vii) Testing**

1. Unit Testing
2. Integration Testing
3. Bug Fixing
4. Acceptance Testing

**(viii) Deployment**

1. Hosting on Server
2. Database Deployment
3. Final Configuration

**(ix) Documentation**

1. Technical Documentation
2. User Manual
3. Final Project Report

**(x) Review & Closure**

1. Final Presentation
2. Project Handover
3. Lessons Learned

#### 

**2.3 Gantt Chart**

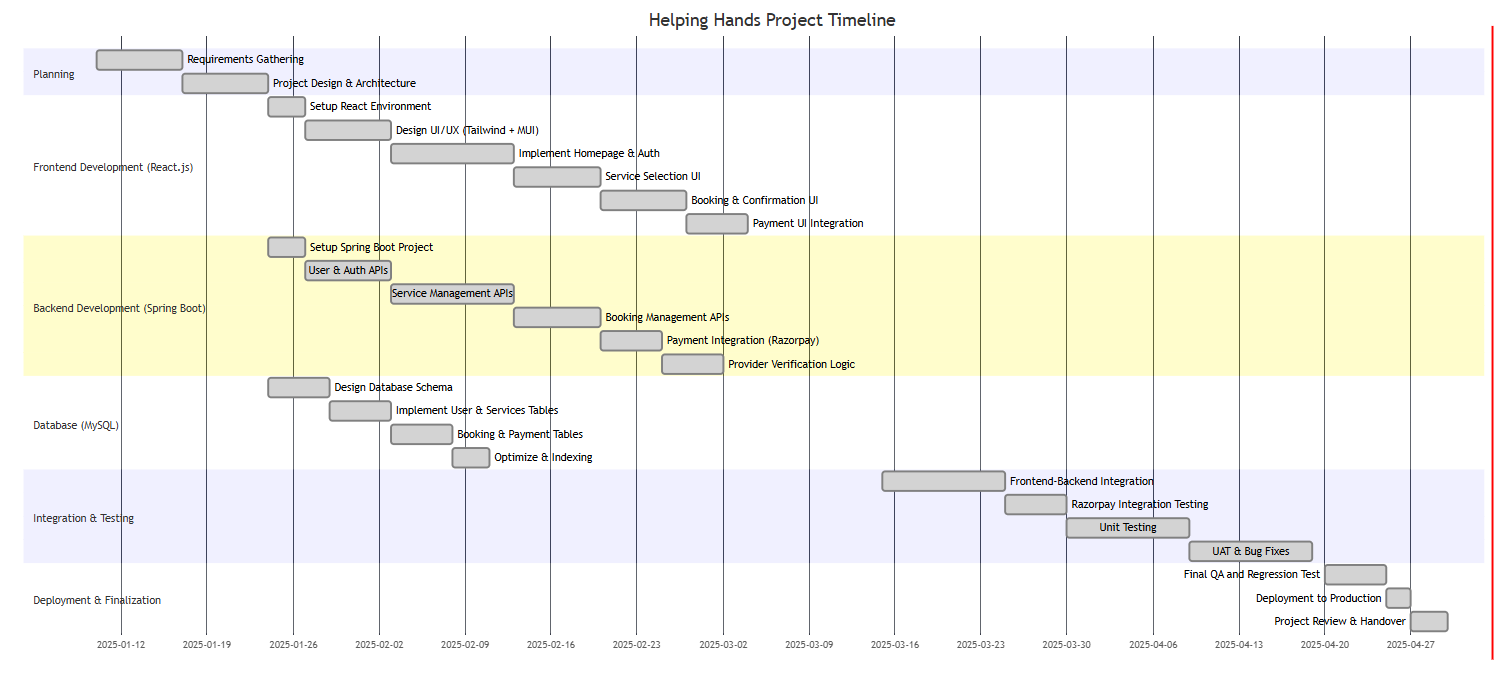


Fig. 2.3.1 Gantt Chart

**2.4 PERT Chart/ CPM**

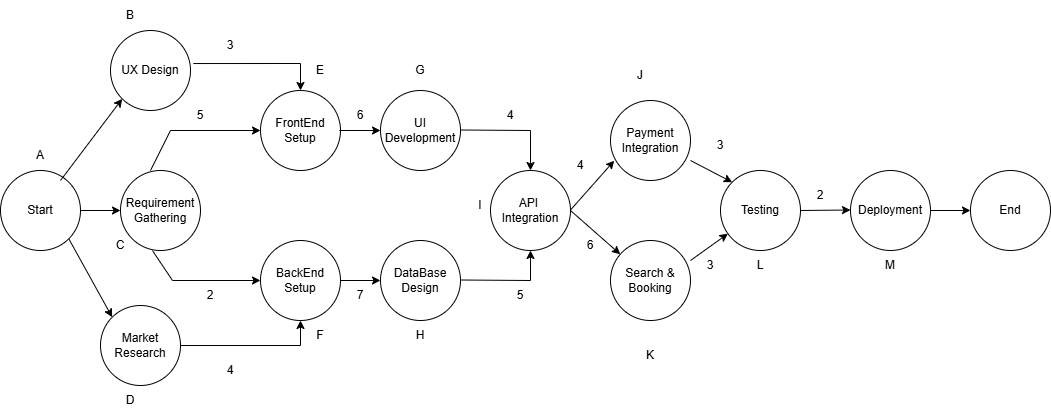
****

Fig. 2.4.1 Pert Chart

**Critical Path Identification**: The critical path (the sequence of tasks that determines the minimum project duration) : Start → Requirement Gathering → Backend Setup → Database Design → API Integration → Search & Booking → Testing → Deployment .

**Duration Estimates**: Task durations in weeks are shown on each connection.

**2.5 Team Structure and Responsibilities**

|  |  |  |
| --- | --- | --- |
| **Team Member** | Role | Responsibilities |
| Rishit Chauhan | Frontend Developer | - Develop UI using React.js, Tailwind CSS, and Material UI - Design responsive service pages and booking forms - Integrate Stripe payment gateway (frontend side) - Ensure smooth navigation and user experience |
| Krishnpal Singh Bhadoriya | Backend Developer | - Build REST APIs using Spring Boot - Handle user authentication, service management, and booking logic - Manage MySQL database schemas and queries |
| Mr. Mohit Varma | |  | | --- | |  |  |  | | --- | | Project Mentor | | |  | | --- | |  |  |  | | --- | | - Guide project architecture and best practices - Provide technical mentorship and code reviews - Monitor project progress and ensure timely completion - Help resolve blockers and improve project quality | |

Table. 2.5.1.Team Structure and Responsibilities

**2.6 Project Development Methodology**

**1. Requirement Gathering**  
We began by understanding the concept of an on-demand service platform similar to Urban Company. Key features were identified, including user authentication, service listings, service booking, and payment integration.

**2.Task Division**

The project was divided into major segments: frontend, backend, database, and integrations. Responsibilities were assigned as follows:

* **Rishit Chauhan**: Frontend Development
* **Krishnpal Singh Bhadoriya**: Backend Development
* **Mr. Mohit Varma (Mentor)**: Technical guidance, code review, and project monitoring

**3.Design**   
UI wireframes were created using Figma, and the backend database schema was designed. Frontend components and backend API endpoints were planned in alignment with the features.

**4. Iterative Development (Agile Sprints)**  
We followed Agile methodology with weekly sprints for modular development:

* **Sprint 1**: UI layout, backend structure.
* **Sprint 2**: Service list, booking feature, database integration.
* **Sprint 3**: Stripe integration and full-stack connectivity.
* **Sprint 4**: Testing, optimization, and UI refinements.

**5. Continuous Integration**  
Code was pushed and reviewed regularly via Git. Each new feature was tested locally before merging into the main branch to ensure stability.

**6. Testing Phase:-**  
Thorough **unit testing** and **integration testing** were conducted.End-to-endworkflows such as service selection, booking, and payment were validated.

**7.Final Review and Deployment:**   
After implementing final feedback from our mentor, the project was documented and finalized for deployment. The application was then prepared for presentation and future scalability.

**2.7 Hardware and Software Requirements**

**1. Hardware Requirements**

|  |  |
| --- | --- |
| **Component** | **Minimum Requirement** |
| Processor | Intel i3 or equivalent |
| RAM | |  | | --- | |  |  |  | | --- | | 4 GB or higher | |
| Hard Disk | 100 GB (SSD preferred) |
| Display | |  | | --- | |  |  |  | | --- | | 1366x768 resolution or higher | |
| Internet Connection | |  | | --- | |  |  |  | | --- | | Required for API integration and testing | |

Table2.7.1. Hardware Requirements

**2. Software Requirements**

|  |  |
| --- | --- |
| **Software** | **Description** |
| Frontend | React.js, Tailwind CSS, Material UI |
| Backend | Spring Boot (Java) |
| Database | MySQL |
| Payment Gateway | Stripe API |
| Development Tools | Visual Studio Code, IntelliJ IDEA |
| Testing Tools | Postman, Browser Developer Tools |
| Version Control | Git, GitHub |
| Design Tool | Draw.io (for wireframes and UI design) |
| Build Tools | Maven (for Spring Boot) |
| Browser | Google Chrome / Firefox |

Table2.7.2. Software Requirements

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 Problem Description**

In today’s fast-paced urban lifestyle, finding reliable home service professionals—such as electricians, plumbers, cleaners, and beauticians—can be inefficient and unreliable. Traditional methods like personal referrals, classified ads, or random online searches often result in inconsistent service quality, non-transparent pricing, and a lack of verified providers.

This leads to significant inconvenience for users and limits income opportunities for skilled workers due to the absence of a unified platform.

There is a clear need for a digital solution that offers a centralized, trustworthy, and efficient way to book local services. Helping Hands aims to fill this gap by providing a seamless online platform that connects users with verified service professionals in their area, offering features like scheduling, payments, and real-time service tracking.

**3.1.1 Problem Definition**

In urban areas, the demand for reliable and professional home services—such as plumbing, electrical repairs, salon services, and home cleaning—is rapidly increasing. However, the process of finding skilled and trustworthy service providers remains largely unorganized and inefficient. Customers often rely on personal contacts, local classifieds, or unverified online listings, leading to problems such as:

Uncertainty about service quality and professionalism.

Lack of price transparency and standardization.

Difficulty in scheduling and managing appointments.

No platform to track service history or leave reviews.

Limited digital visibility for skilled workers.

This results in inconvenience for customers and missed employment opportunities for local service professionals.

Helping Hands is designed to address this problem by developing a digital platform that connects users with verified service providers across various categories. The system will offer features such as user-friendly service discovery, online booking and scheduling, secure payments, service tracking, and customer reviews. It aims to streamline the entire service experience while empowering professionals with tools to grow and manage their business online.

**3.1.2 Proposed Solution**

To address the issues in the current unstructured home services market, Helping Hands proposes the development of a comprehensive online service marketplace platform that bridges the gap between customers and verified service professionals. The system will streamline service discovery, booking, payments, and feedback, thereby improving user experience and creating better job opportunities for professionals.

**🔧 Key Features of the Proposed Solution:-**

**User-Friendly Web & Mobile Application:**

Intuitive interface for customers to browse, filter, and book services by category, location, or rating.

**Service Listing and Categorization:**

Services grouped into categories like Home Cleaning, Beauty & Wellness, Appliance Repair, etc., with descriptions and pricing.

**Booking and Scheduling System:**

Real-time availability management with options to choose preferred time slots and dates.

**Online Payment Integration:**

Secure payment gateway for cashless transactions with optional wallet or COD support.

**Ratings and Reviews System:**

Post-service feedback mechanism to build trust and ensure service quality.

**Admin Dashboard:**

Central control panel to manage users, services, bookings, payments, reports, and resolve disputes.

**Benefits of the Proposed Solution**

**For Users:**

Easy access to trusted services.

Transparent pricing and secure payments.

Time-saving booking process.

**For Service Providers:**

Greater customer reach.

Improved job scheduling and management.

Increased credibility through reviews and ratings.

**For Admins:**

Centralized control over platform activities.

Analytics and reporting for business decisions.

Dispute resolution and quality assurance.

**3.2 Requirements**

The system must allow users and service providers to register, log in, and manage their profiles securely.Users should be able to browse, filter, and book services based on category, location, and availability.Integrated payment functionality must support secure online transactions and receipts.An admin dashboard is required for managing users, services, bookings, and resolving issues.The platform should be responsive, scalable, and secure, ensuring smooth performance on web and mobile.

**3.2.1 Functional Requirements**

These define the core features and functionalities that the system must support:

**1.1** **User Registration and Authentication**

Customers and service providers must be able to register and log in securely.

Support for password recovery and OTP/email verification.

**1.2** **User Profile Management**

Customers can edit personal details, view booking history, and manage addresses.

Service providers can update skills, availability, pricing, and view past jobs.

**1.3 Service Listing and Categorization**

Services should be listed under categories such as Cleaning, Electrical, Plumbing, Beauty, etc.

Each service includes description, price, duration, and availability.

**1.4** **Service Search and Filtering**

Users can search services by name, category, location, rating, and price range.

**1.5 Booking System**

Customers can select a service, choose a date/time, and confirm the booking.

Service providers are notified of new bookings and can accept or reject them.

**1.6** **Payment Integration**

Integration with payment gateways (e.g., Razorpay, Stripe).

Support for online payments, wallets, and optional COD.

**1.7** **Ratings and Reviews**

Customers can rate and review services after completion.

Service providers can view their ratings and respond if needed.

**1.8** **Admin Dashboard**

Admin can manage users, services, bookings, payments, reviews, and handle disputes

Ability to generate reports and monitor platform metrics.

**3.2.2 Non-Functional Requirements**

These define the quality attributes and operational constraints of the system:

**2.1** **Performance**

System should load pages within 3 seconds under normal load.

Support for concurrent users without performance degradation.

**2.2** **Scalability**

System must handle increasing number of users and service providers as the platform grows.

**2.3** **Security**

Data encryption for passwords and transactions.

Role-based access control (Admin, Customer, Service Provider).

Secure APIs and protection against common attacks (SQL injection, XSS, etc.).

**2.4 Usability**

Clean and responsive UI for mobile and web.

Easy navigation and booking flow for users of all technical skill levels.

**2.5** **Availability**

The platform should maintain 99.9% uptime.

Automatic backups and failover mechanisms.

**2.6** **Maintainability**

Modular codebase with proper documentation.

Easy to update features or fix bugs without affecting the entire system.

**3.3 Problem Analysis Diagrams**

**Problem Analysis Diagram** for the "Helping Hands" platform, which connects users with professional service providers. This diagram identifies the core challenges faced by each actor and outlines the system modules designed to address them.

**3.3.1 Data Flow Diagram/ Process Flow Diagram**

****

Fig. 3.3.1.1 Context Level DFD

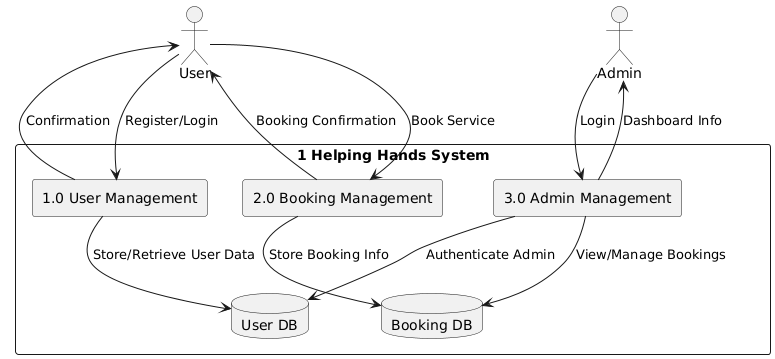
****

Fig. 3.3.1.2 First Level DFD

**3.3.2 Use Case Diagram and Sequence Diagram**

**1. Use Case Diagram**

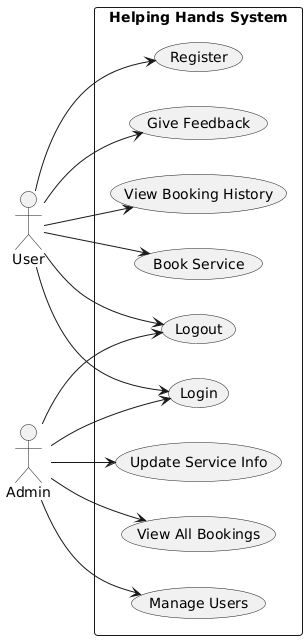
****

Fig. 3.3.2.1. Use Case

**2. Sequence Diagram**

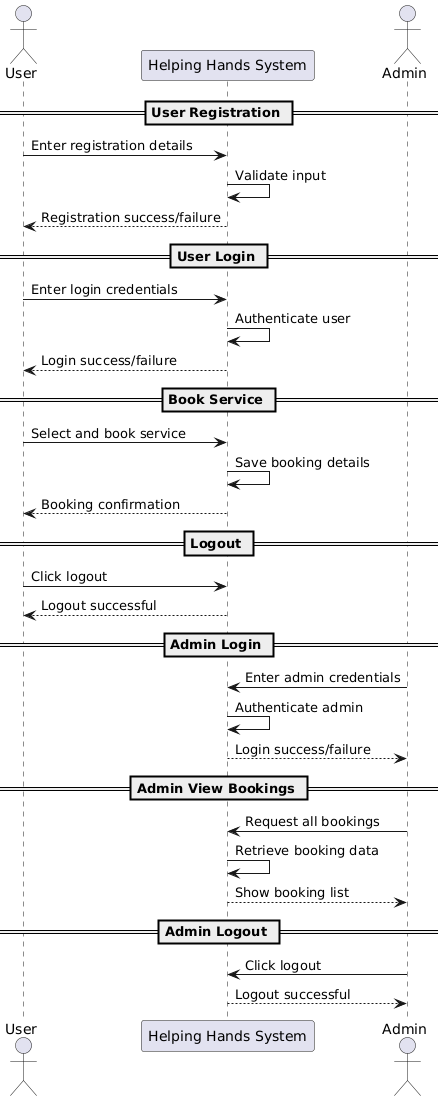


Fig. 3.3.2.2.Sequence Diagram

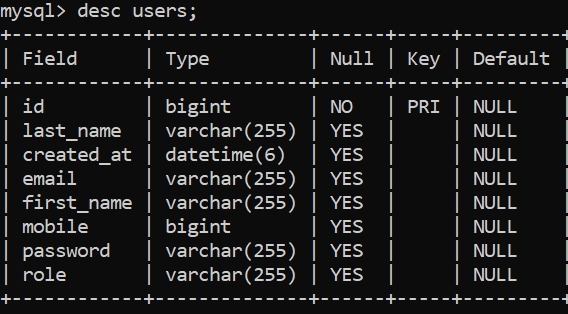
**3.4 Database Schema**

**3.4.1. Create the database schema**

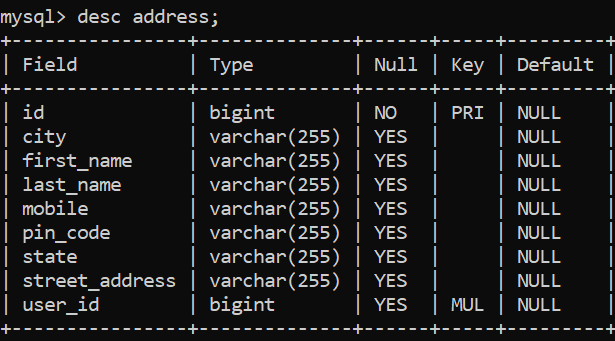
CREATE DATABASE IF NOT EXISTS helpinghands\_db;

USE helpinghands\_db;

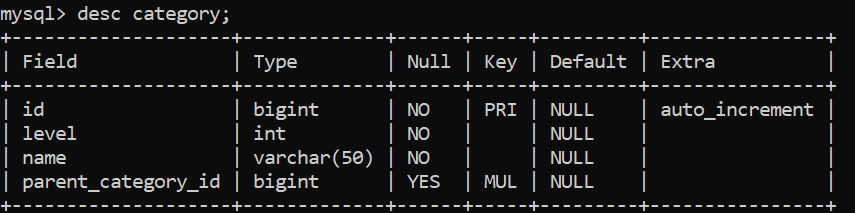
**3.4.2. Users table**



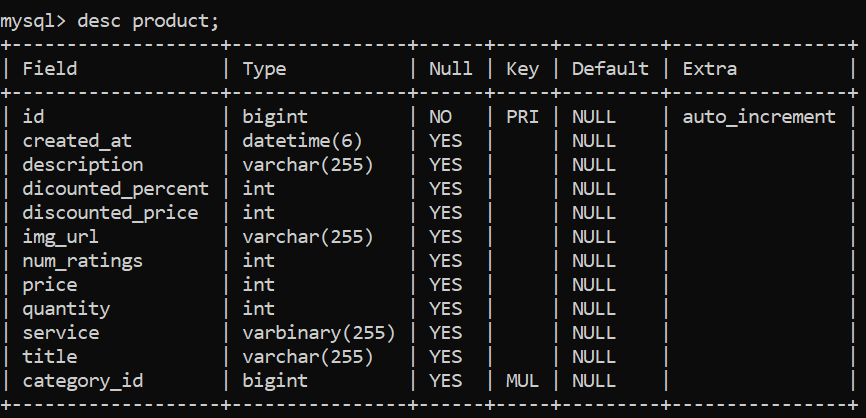
**3.4.3. Address table**



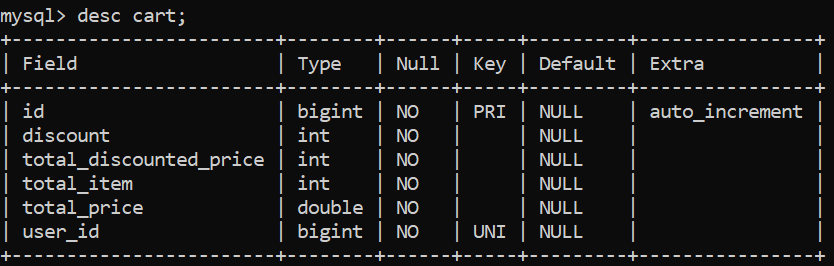
**3.4.4. Category table**



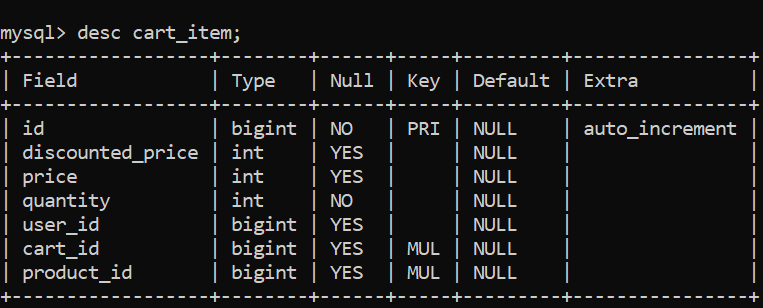
**3.4.5. Product table**

.

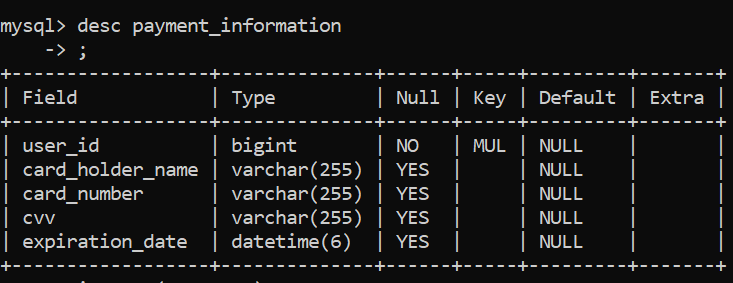
**3.4.6. Cart table**



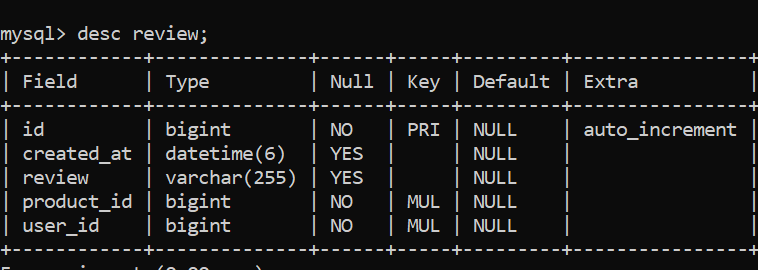
**3.4.7. Cart Item table**



**3.4.8. Payment Information table**



**3.4.9. Review table**



**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 System Architecture**

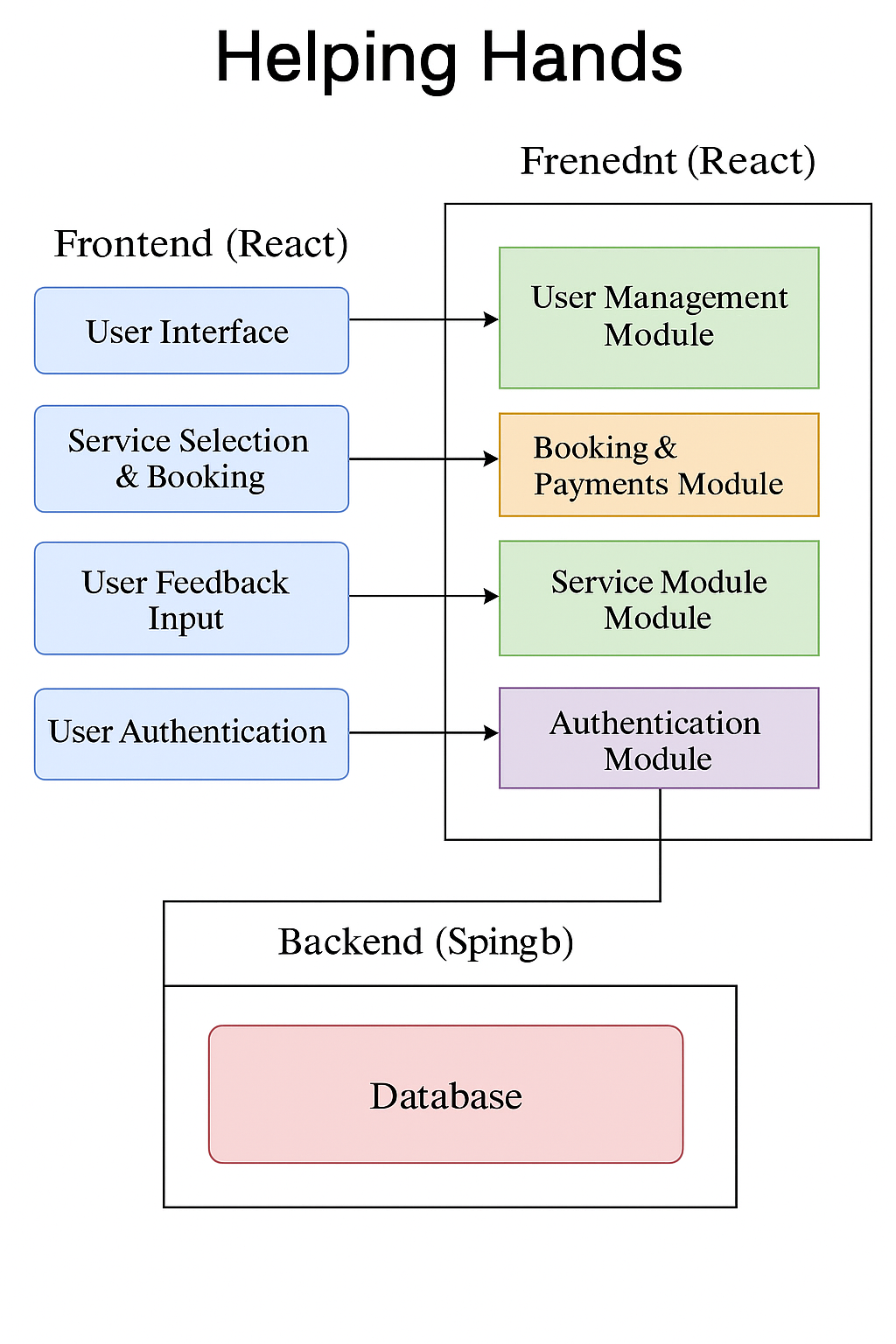


Fig. 4.1.1. System Architecture

**4.2 Physical Design**

The physical design of the **Helping Hands** system defines how data and software components are physically stored and deployed in the system. It specifies the hardware configuration, server setup, network topology, and actual database tables used during implementation.

**4.2.1 Structure Chart**

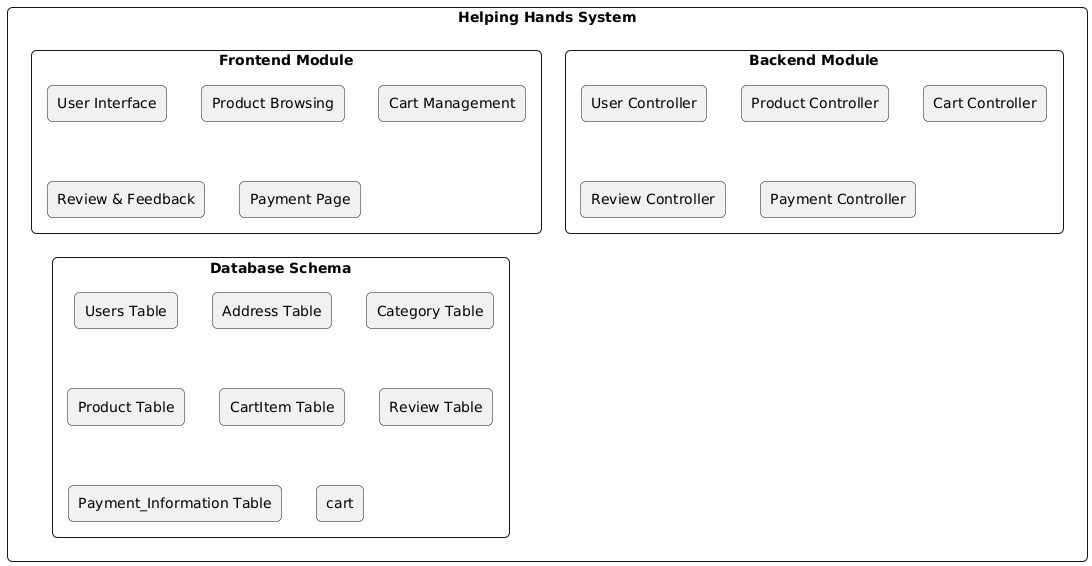


Fig. 4.2.1.1.Structure Chart

### ****Frontend (React.js)****

Developed by **Rishit Chauhan**, this is the client-side interface where users interact with the application.

* **Home Page**: Entry point displaying featured services and promotions.
* **Service Page**: Lists services by category (e.g., cleaning, repair).
* **Login/Register Page**: Allows user sign-up and login.
  + with Booking and Payment logic via backend APIs.

### ****Backend (Spring Boot)****

Built by **Krishnpal Singh Bhadoriya**, it manages data operations, authentication, and logic.

* **User Controller**: CRUD operations for user profiles.
* **Booking Controller**: Handles creating, updating, and retrieving bookings.
* **Auth Controller**: Manages secure login/logout using tokens.
* **Admin Controller**: Admin-exclusive features like viewing all users, bookings, and modifying services.

**4.2.2 ER Diagram**

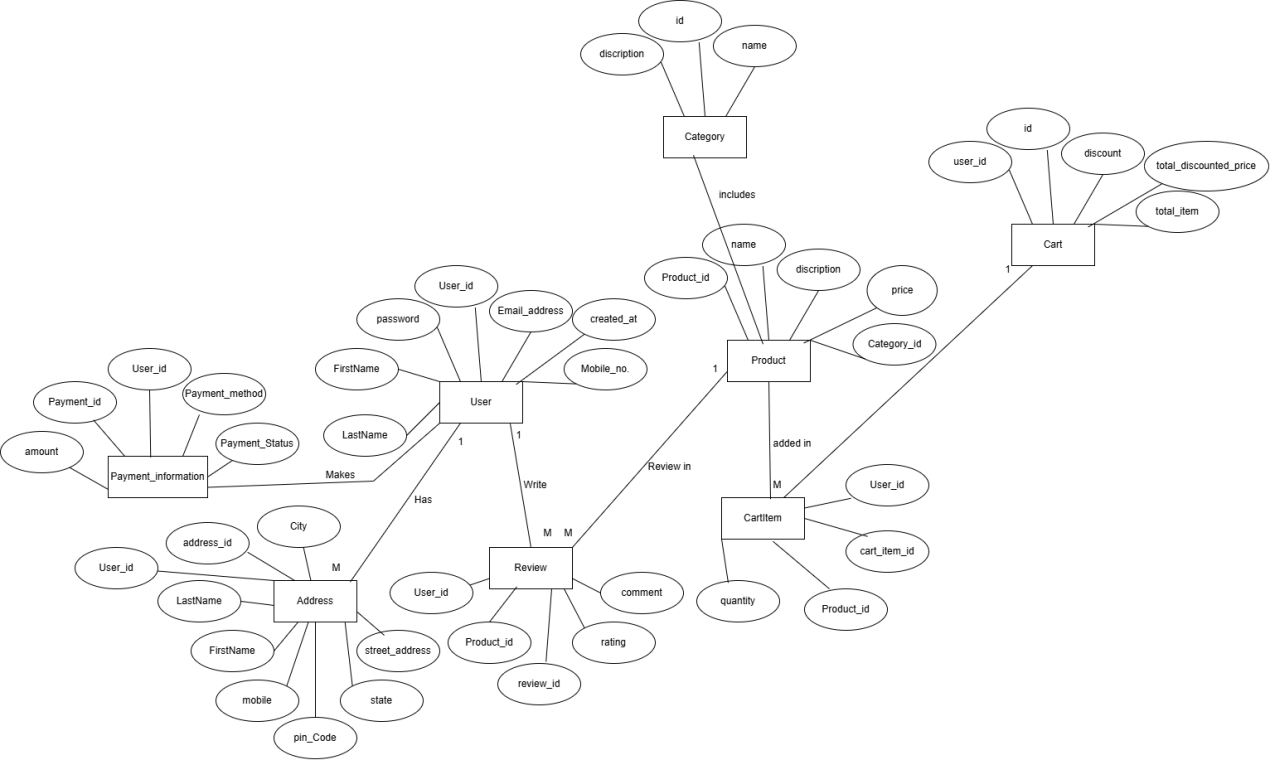


Fig. 4.2.2.1.ER Diagram

**4.2.3 Class Diagram and Object Diagram**

**1. Class Diagram:-**

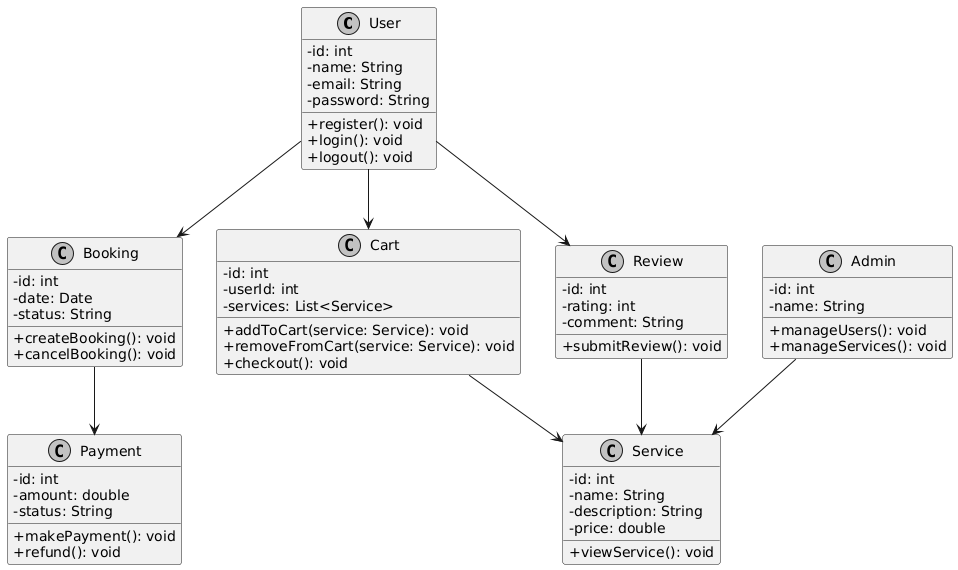


Fig. 4.2.3.1 Class Diagram

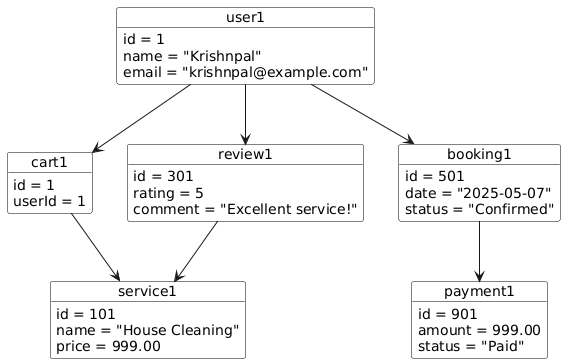
**2. Object Diagram:-**

Fig. 4.2.3.2.Object Diagram

**4.3 Input and Output Design**

**1. Input Design:-**

| **Input Name** | **Source** | **Format** | **Validation Rules** |
| --- | --- | --- | --- |
|  |  |  |  |
| Registration Details | User | Name, Email, Password | Valid email format, password length ≥ 6 characters |
| Login Credentials | User/Admin | Email, Password | Match with stored credentials |
| Service Booking | User | Service, Date, Address | Non-empty, date must be valid and not in the past |
| Payment Details | User | Card/UPI Info | Valid payment info required |
| Review and Rating | User | Stars (1–5), Comment | Rating must be 1–5, optional comment max 200 chars |
| Service Management | Admin | Name, Desc, Price | Mandatory fields, price ≥ 0 |

Fig. 4.3.1.Input Design

**2. Output Design:-**

| **Output Name** | **Target User** | **Format** | **Purpose** |
| --- | --- | --- | --- |
| Login Confirmation | User/Admin | Message/Dialog Box | Indicates successful login |
| Booking Confirmation | User | Email & Dashboard View | Confirms service booking |
| Payment Receipt | User | On-screen & Email | Displays transaction ID and status |
| Admin Dashboard Report | Admin | Tabular and Graphical | Shows user activity and service requests |
| Review Display | All Users | Star Ratings & Comments | Helps users evaluate services |
|  |  |  |  |

Fig. 4.3.1.Output Design

**4.4 Algorithmic Design**

| **S.No** | **Algorithm Name** | **Input** | **Output** | **Steps** |
| --- | --- | --- | --- | --- |
| 1 | **User Registration** | Name, Email, Password | Success or Error Message | 1. Start 2. Accept inputs 3. Validate inputs 4. Check if email exists 5. Hash password 6. Save to DB 7. Return response 8. End |
| 2 | **User Login** | Email, Password | User Dashboard or Error Message | 1. Start 2. Accept login input 3. Fetch user by email 4. Compare hashed password 5. Create session/token 6. Redirect or show error 7. End |
| 3 | **Book a Service** | Service ID, Date, Time, Address | Booking Confirmation | 1. Start 2. Authenticate user 3. Validate inputs 4. Save booking to DB 5. Return confirmation 6. End |
| 4 | **Payment Processing** | Booking ID, Payment Details | Payment Status | 1. Start 2. Validate session 3. Collect payment info 4. Call Stripe API 5. Update status on success 6. Return status 7. End |

Fig. 4.4.1.Algorithmic Design

**CHAPTER 5**

**IMPLEMENTATION**

**5.1 Integration of Modules/ Files**

| **Module** |  | **Technology Used** | **Integrated With** | **Purpose** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Frontend UI |  | React.js | Backend REST APIs | To send/receive data for services, booking, login, etc. |
| User Authentication |  | Spring Boot (Java) | MySQL (Users Table), React UI | Handles user login, registration, and logout processes |
| Product Management |  | Spring Boot | React UI, MySQL (Product, Category tables) | Enables browsing and viewing available services/products |
| Cart Module |  | React.js | MySQL (CartItem), Spring Boot (Cart Controller) | Allows users to add/remove items before booking or payment |
| Payment Integration |  | Stripe, Spring Boot | Frontend (React), MySQL (Payment\_Information) | Handles secure payment processing and transaction recording |
| Review & Rating Module |  | React.js + Spring | MySQL (Review Table) | Allows users to submit and view service feedback |
| Address Management |  | Spring Boot | MySQL (Address Table), React UI | Collects and manages user addresses during booking |

Fig. 5.1.1 .Integration of Modules

**CHAPTER 6**

**SYSTEM TESTING**

**6.1 Test Case Design**

Test case design involves creating a set of inputs, execution conditions, and expected outcomes that ensure the system works as intended. The purpose is to validate that each module performs correctly under various scenarios. Testing includes both functional and non-functional aspects of the system.

**6.1.1 Unit Testing**

Unit testing involves verifying individual components of the system, such as functions, classes, or modules. In the Helping Hands project, unit tests are conducted for components like user login, service listing, and booking modules. Each test checks for expected input/output behavior and handles edge cases or errors.

**6.1.2 Integration Testing**

Integration testing checks the interaction between modules. For example, after logging in, a user should be able to access the dashboard and book a service. Testing also verifies interactions between frontend (React.js) and backend (Spring Boot), ensuring data flow is smooth and error-free.

**6.1.3 System Testing**

System testing validates the entire application as a whole. The system is tested for compliance with requirements. This includes testing all functionalities like registration, login, service search, booking, payment, and logout to ensure end-to-end performance across user scenarios.

**6.1.4 Acceptance Testing**

Acceptance testing is performed to ensure the system meets client expectations and real-world usability. Stakeholders validate whether the project delivers its promised value. In this case, testing is conducted under the supervision of the mentor and includes real-use workflows like booking services and making payments.

**6.2 Specific System Testing**

Specific testing includes performance testing, load testing, and security testing. For instance, the booking and payment modules are stress-tested with multiple simultaneous users. Also, login and logout functionalities are tested for session management and access control.

**6.3 Test Reports**

| **Test Case ID** | **Module Name** | **Test Scenario** | **Expected Result** | **Actual Result** | **Status** |
| --- | --- | --- | --- | --- | --- |
| TC001 | Login | User enters valid credentials | User is redirected to the dashboard | User successfully redirected | Pass |
| TC002 | Login | User enters invalid credentials | Error message is shown | Error message displayed | Pass |
| TC003 | Registration | New user fills valid data | Account is created and redirected to login page | Account created successfully | Pass |
| TC004 | Booking | User books a service | Booking confirmation is shown | Confirmation displayed | Pass |
| TC005 | Payment | User completes Stripe payment | Payment success message is shown | Payment success message displayed | Pass |
| TC006 | Logout | User clicks logout | Session ends and redirected to home | User logged out and redirected | Pass |
| TC007 | Review & Rating | User submits rating and review | Review is added and visible | Review visible in service page | Pass |
| TC008 | Admin Dashboard | Admin views all bookings | List of bookings displayed | Bookings list loaded correctly | Pass |
| TC009 | Cart Functionality | User adds service to cart | Service appears in cart | Cart updated successfully | Pass |
| TC010 | Data Validation | User leaves form fields empty | Validation messages shown | Proper validation messages displayed | Pass |

Fig. 6.3.1.Test Report

**CHAPTER 7**

**CONCLUSION OF THE PROJECT**

**7.1 Results**

The project "Helping Hands" successfully achieved its core objectives, offering a smooth and intuitive platform for users to book home services online. Users were able to register/login, browse categorized services, add services to the cart, book services, and complete payments securely. Admin functionalities allowed oversight and management of service categories, users, and bookings efficiently.

**7.2 Conclusion**

This project serves as a comprehensive home service booking system where users and admins interact through well-defined interfaces. It demonstrates effective integration between frontend (React.js), backend (Spring Boot), and database (MySQL), providing a scalable and user-friendly solution. The use of technologies like Stripe for payment and structured database design ensured robust performance and real-world applicability.

**7.3 Limitations of the project**

 The system does not currently support real-time communication between user and admin.

 There is no support for service provider-side login or scheduling.

 Location-based service filtering is limited and not dynamically integrated.

 The admin panel lacks advanced analytics for performance monitoring.

**7.4 Future Work**

 Integration of live chat between users and service providers.

 Inclusion of a dedicated mobile app version.

 Addition of AI-driven service recommendations based on user behavior.

 Advanced admin dashboard with service trends, revenue reports, and feedback analysis.

**7.5 Lessons Learned**

 Learned how to build and manage a full-stack application integrating multiple technologies.

 Understood the importance of clear database design and API communication.

 Gained experience in team collaboration and version control through Git.

 Improved understanding of user experience and front-end design principles.

**REFERENCES**

 **React.js Documentation**  
https://reactjs.org/docs/getting-started.html

 **Spring Boot Documentation**  
<https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>

 **MySQL Documentation**  
<https://dev.mysql.com/doc/>

 **PlantUML Documentation**  
<https://plantuml.com/>

 **Mermaid.js Documentation**  
https://mermaid.js.org/

 **Draw.io (diagrams.net)**  
<https://www.diagrams.net/>

 **MDN Web Docs** (for HTML/CSS/JavaScript Reference)  
<https://developer.mozilla.org/>

 **Postman Tool**  
<https://www.postman.com/>

 **Tailwind CSS Documentation**  
<https://tailwindcss.com/docs>