CAPSTON PROJECT

MAKETRIP - UNIFIED TRAVEL BOOKING SYSTEM

SANJAY . N

JAVA FULL SRACK

BE - CSE - AI

43731114

1. INTRODUCTION

The Project aims to automate the process of Travel and Hospitality Management and streamline the booking experience for various travel services. It provides a platform where users can view Hotels, Transport options (Flights, Trains, Buses), and Food menus that are currently available. Additionally, it offers detailed information on pricing, schedules, and availability so that users may assess their travel plans and book instantly. The user experience is further enhanced by a modern Glassmorphism interface and advanced search filters, allowing travelers to tailor results by city, budget, and dietary preferences.

On the administrative side, the system helps service providers and administrators find eligible bookings and manage inventory according to their criteria. This includes real-time inventory tracking that instantly updates availability to prevent overbooking. This Unified Travel System consists of several key components, including an Admin Module, User Module, Hotel Module, Transport Module, Food Module, and Booking Module. Underpinning these features is a robust security framework using Spring Security and BCrypt, ensuring that user data and administrative controls remain secure within a scalable MVC architecture.

2. ABSTRACT

MakeTrip is a Unified Travel Booking System designed to develop software that manages travel booking activities. It creates an intensive platform where the system can manage the details of all bookings, hotels, transportation, and food orders on a central console. Anyone can access the platform to search for services and make reservations using advanced filters for cities, transport modes, and accommodation types. The reason behind creating this project is to help travelers organize their entire trip in one place without visiting multiple websites, while providing a modern, responsive user interface designed with a Glassmorphism aesthetic.

The system enables Administrators to efficiently manage bookings and real-time inventory to prevent overbooking, secured by role-based access and BCrypt encryption. It is built on a scalable MVC architecture using Spring Boot, MySQL, and Thymeleaf

3. TECHNICAL SPECIFICATIONS

What we are going to build:

- Authentication: Proper Login and Register API using Spring Security.
- API Features: Services API includes Pagination and Sorting (e.g., Sort Hotels by Price).
- Validation: Proper user input validation handling (e.g., Date checks).
- Exception Handling: Proper exception handling (e.g., "Seats not available").
- Security: Role-based authentication (ADMIN vs USER) and BCrypt password encryption.
- Documentation: Document all REST APIs for consumer understanding.
- Deployment: Deploy the backend application on a cloud platform.

4. TECHNOLOGIES AND TOOLS

Framework: Spring Boot Java Framework.

- Language: Java 21 (LTS)
- Build Tool: Maven
- IDE: Spring Tool Suite (STS) / IntelliJ IDEA
- Server: Apache Tomcat (Embedded)
- Backend: Spring Core, Spring Security 6, Spring Data JPA (Hibernate)
- Database: MySQL Database 8.0
- Testing: Postman REST Client
- Frontend: HTML5, CSS3 (Glassmorphism), Thymeleaf

5. SYSTEM REQUIREMENTS

Software Requirements:

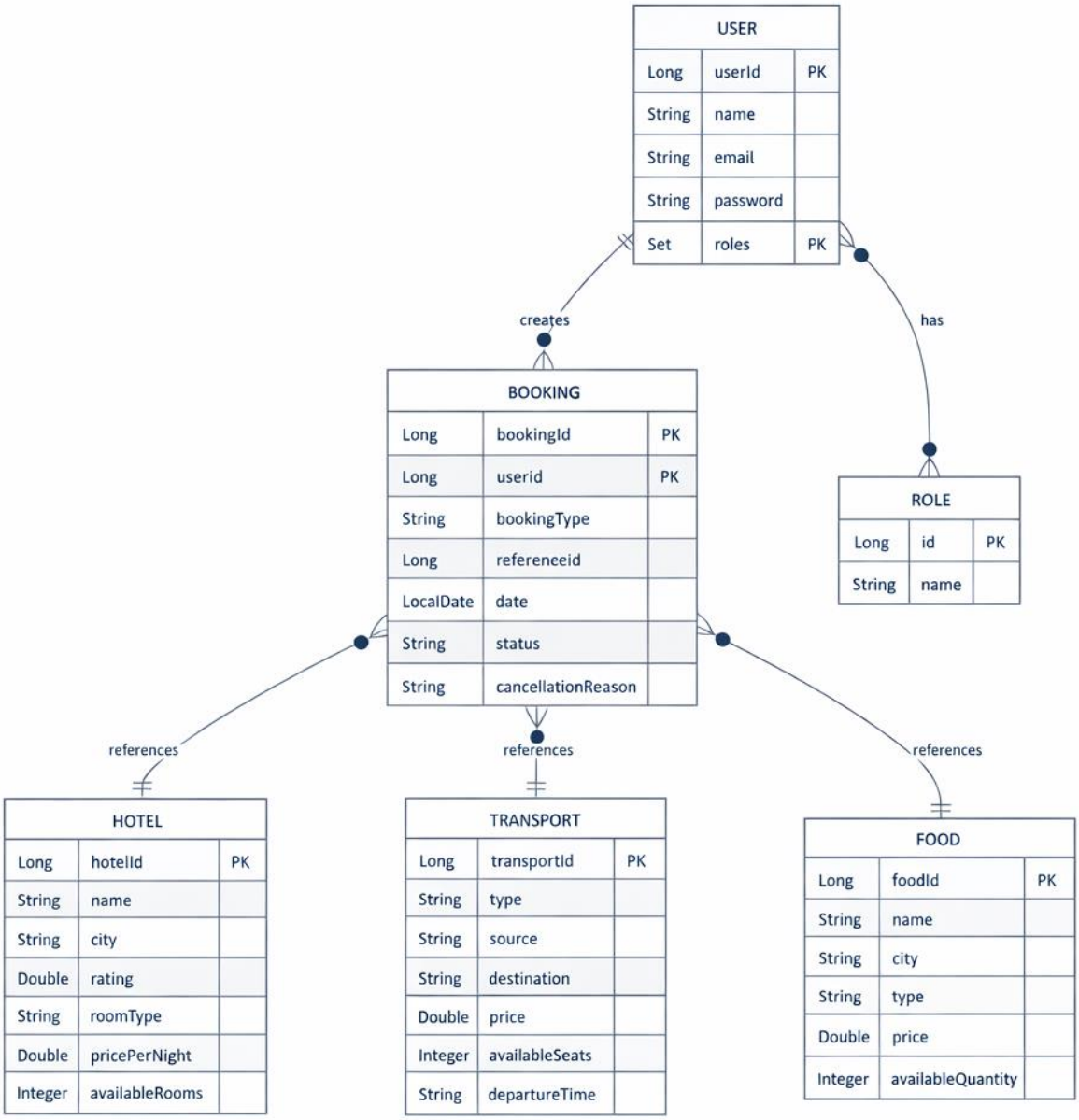
- Operating System: Windows 10/11 or Linux
- Back End: Spring Boot, JPA Hibernate
- Database: MySQL Server 8.0
- Web Server: Apache Tomcat
- Browser: Google Chrome / Edge

6. PROJECT MODULES

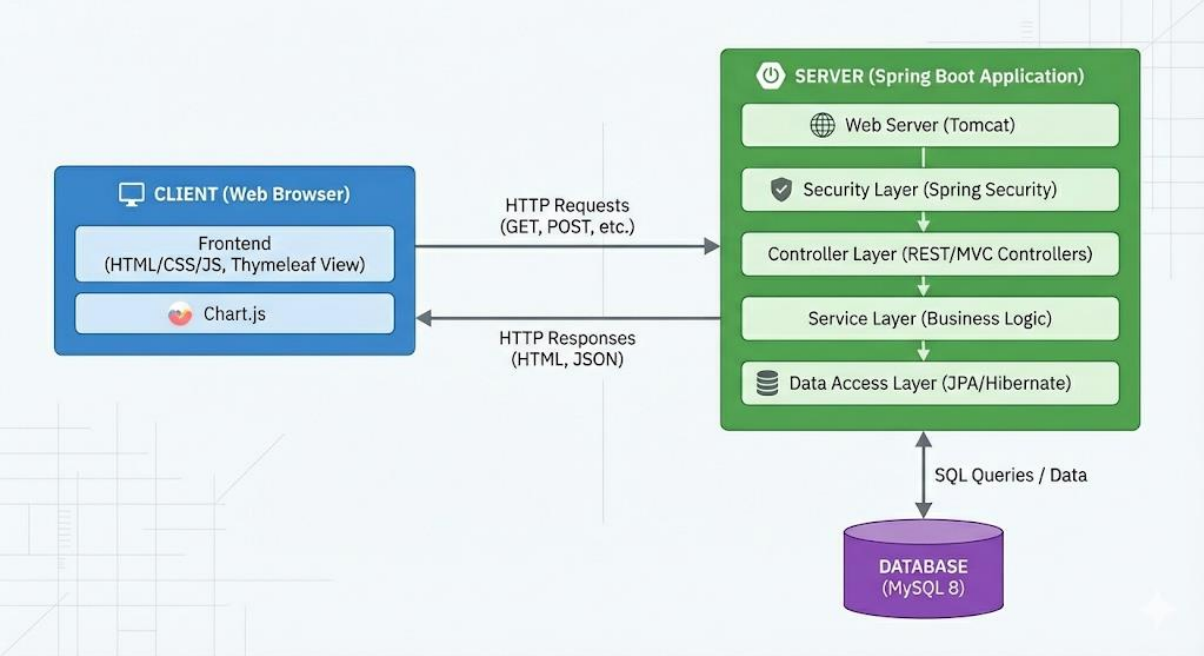
The system is divided into the following functional modules:

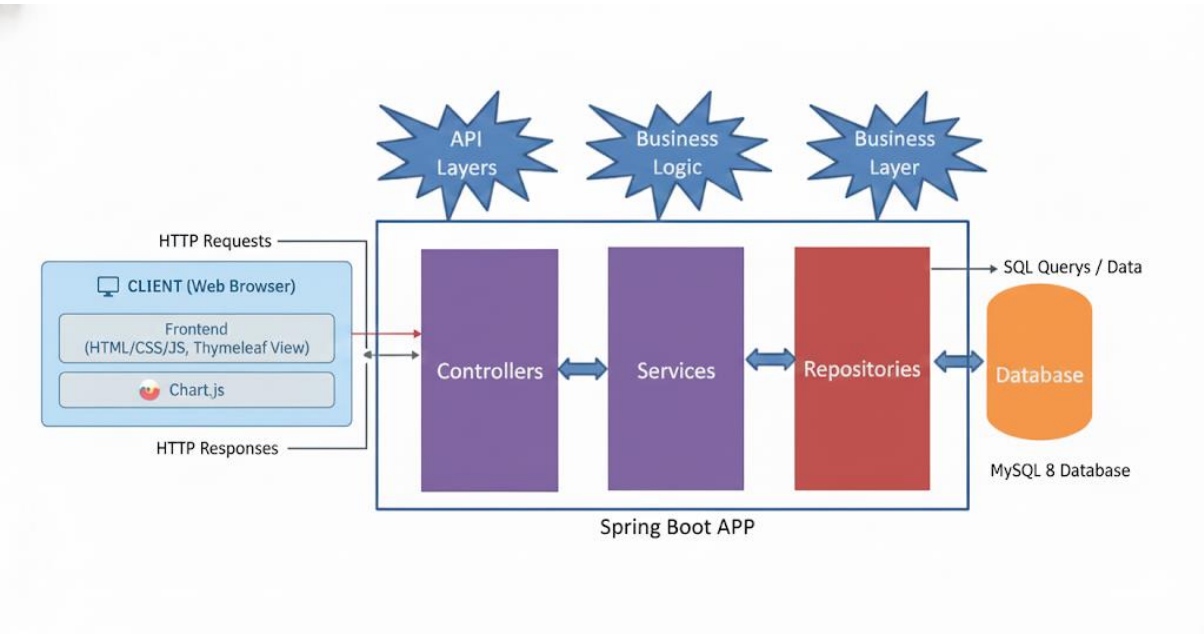
1. Admin Module
2. User Module
3. Hotel Module
4. Transport Module
5. Food Module
6. Booking Module

7. ER DIAGRAM (Conceptual)



8. CLIENT - SERVER ARCHITECTURE





9. MODULE DESCRIPTIONS

A. Admin Module

In the Admin Module, the user can move after logging in successfully. Admin can save their details by using the application interface or Postman Tool.

- Save details: Using Post Mapping.
- Get details: Using Get Mapping.
- Update details: Using Put Mapping.
- Delete details: Using Delete Mapping.

B. User Module

After completing the Admin Module, we move into the User Module.

- We can save the user data (Registration) by using Post Mapping.
- Users can log in using Spring Security authentication.
- Users can view their own profile and booking history using Get Mapping.

C. Hotel Module

- Admin can add hotel details by using Post Mapping.
- Users can search for hotels by City using Get Mapping.
- Users can book rooms, which updates the availableRooms count in the database.

D. Transport Module

- Admin can add transport routes (Source, Destination, Type, Price).
- Users can search for transport to collect a list of available routes.
- Users can book tickets, updating the availableSeats.

E. Food Module

- Admin can add food items to the menu.
- Users can search for food by City and Type (Veg/Non-Veg).
- Users can order food, which links to the Booking Module.

F. Booking Module

This module acts as the transaction layer.

- It saves the referenceId of the service and links it to the userId.
- It handles the logic for confirming availability.
- It generates a status (CONFIRMED / CANCELLED).

10. HTTP REQUEST METHODS

HTTP defines a set of request methods to indicate the desired action to be performed for a given resource.

Method	URI Endpoint	Description
GET	http://localhost:8080/hotels	Return the list of hotels
GET	http://localhost:8080/transport	Return the list of transport options
POST	http://localhost:8080/auth/register	Create a new user
POST	http://localhost:8080/hotels/book	Create a new hotel booking
DELETE	http://localhost:8080/admin/delete/1	Delete a service item

11. PROJECT OUTPUTS

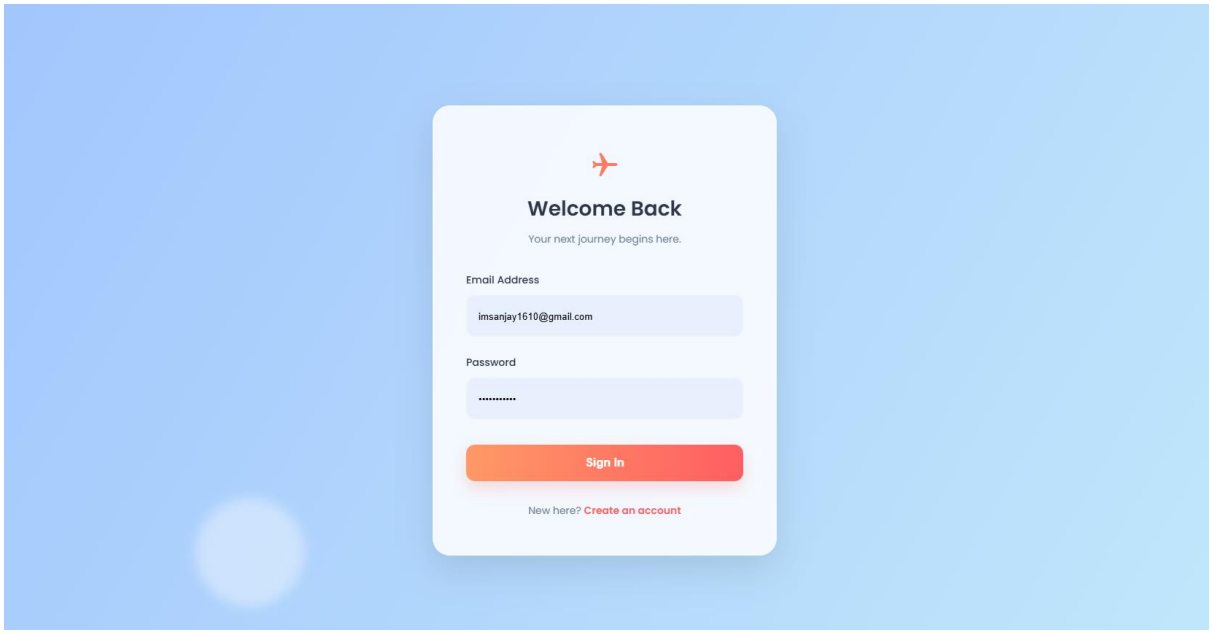


Figure: Login Page

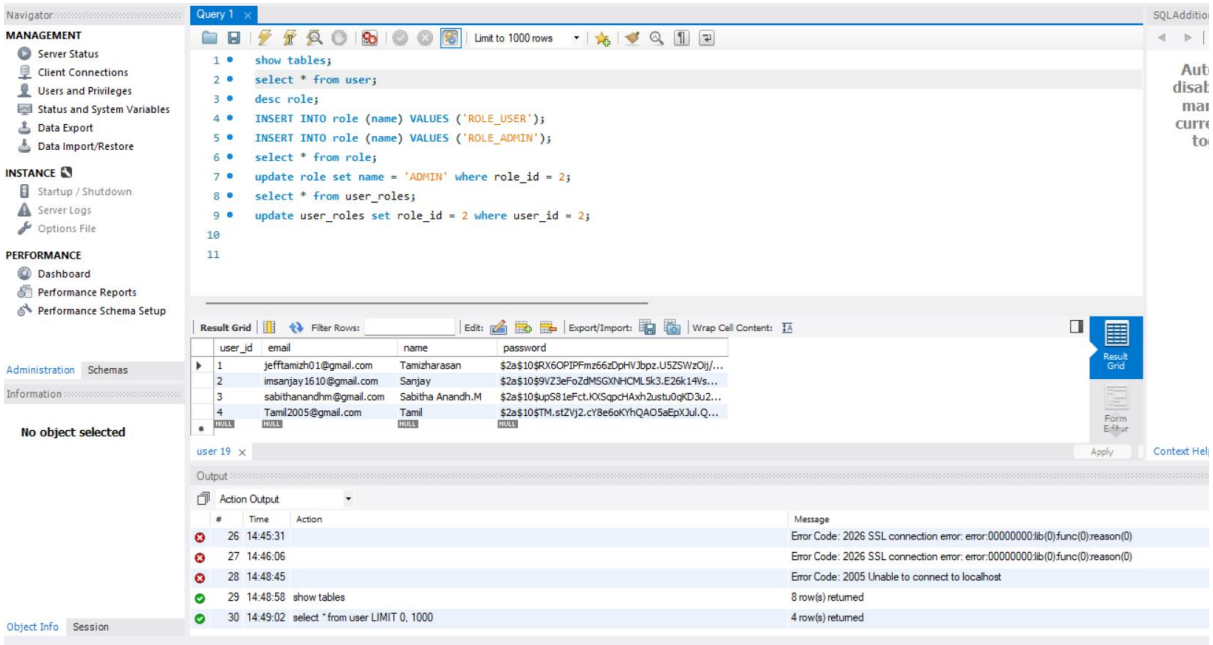


Figure: User Data

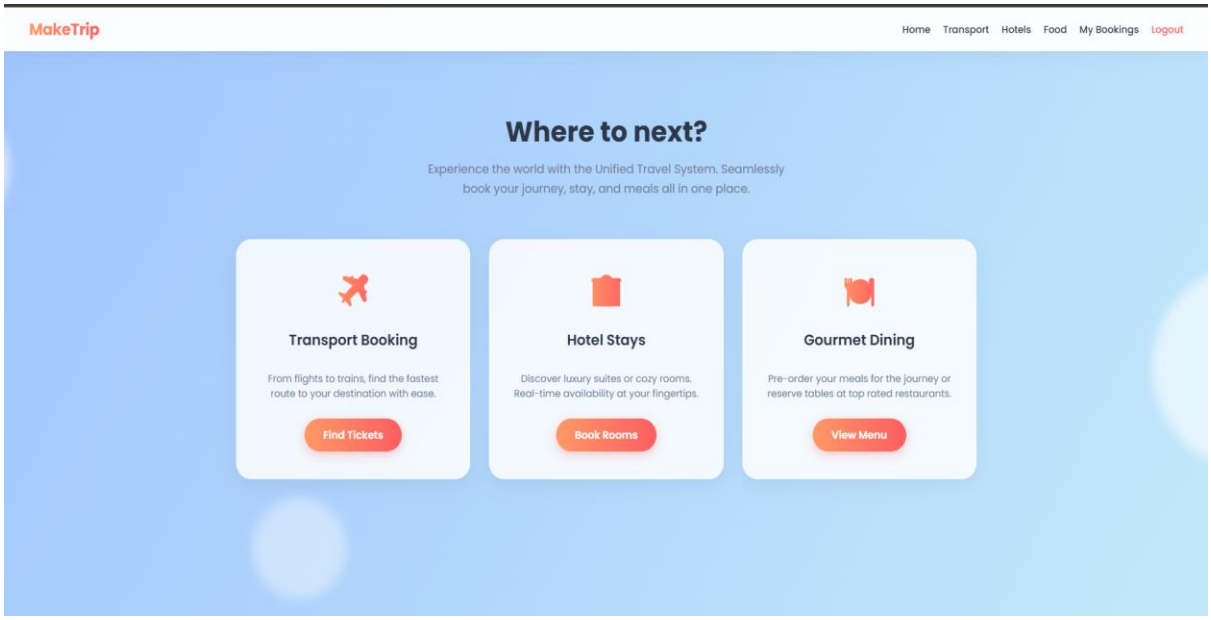


Figure: Home Page

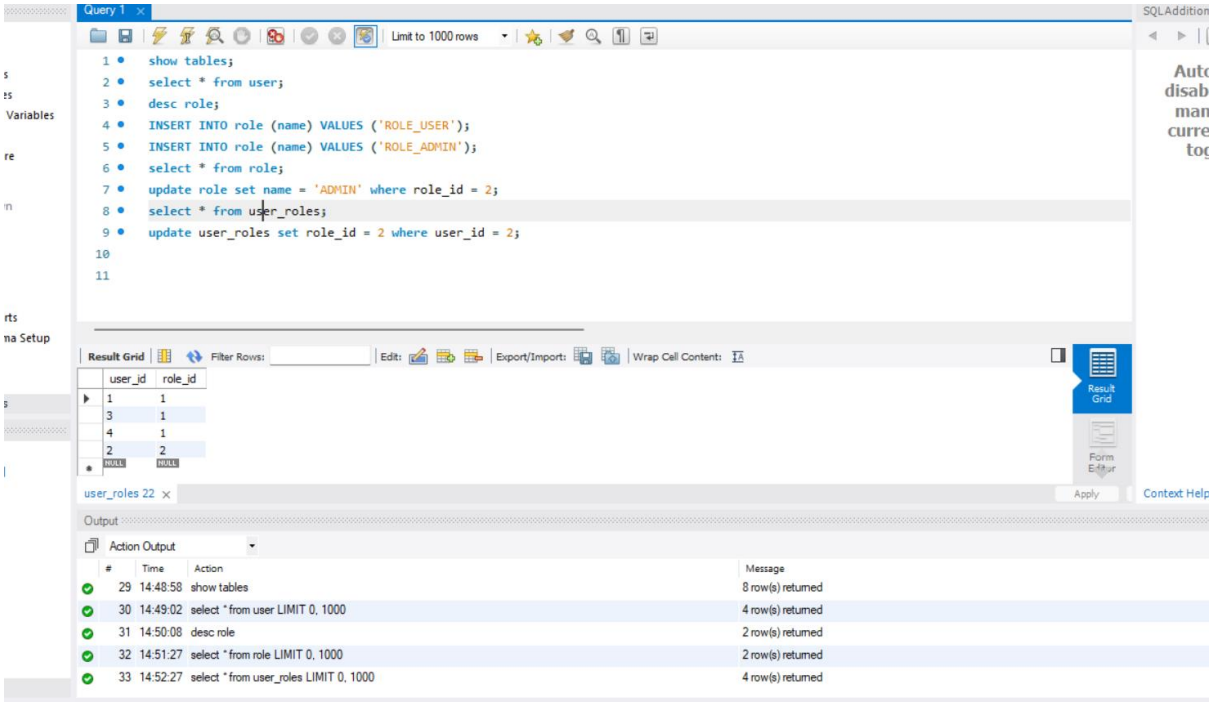


Figure: Roles Data

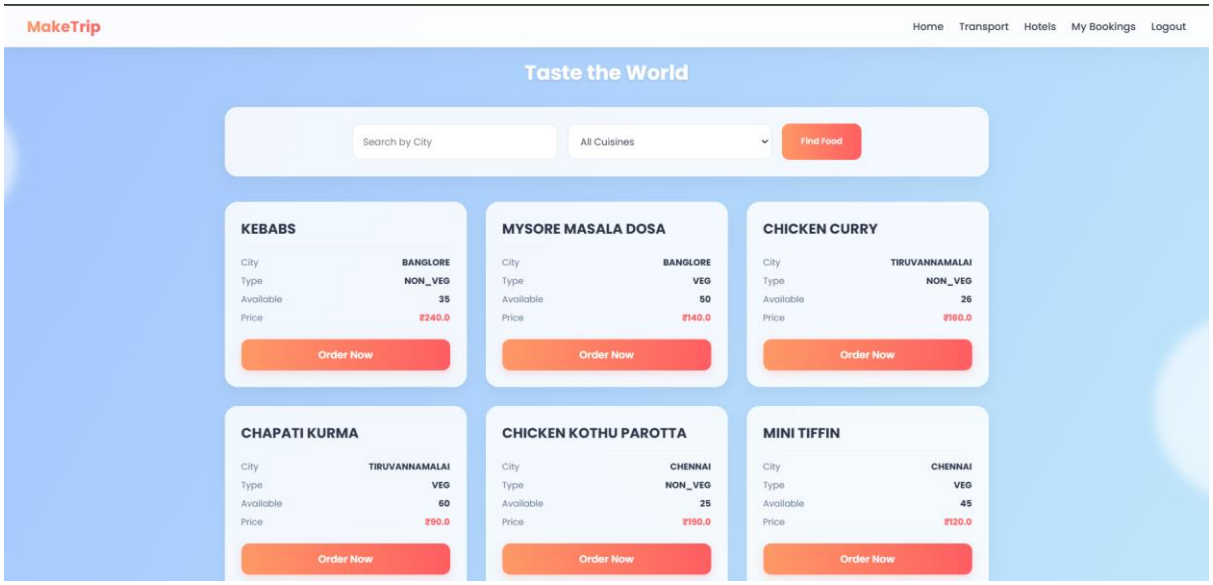


Figure: Food Page

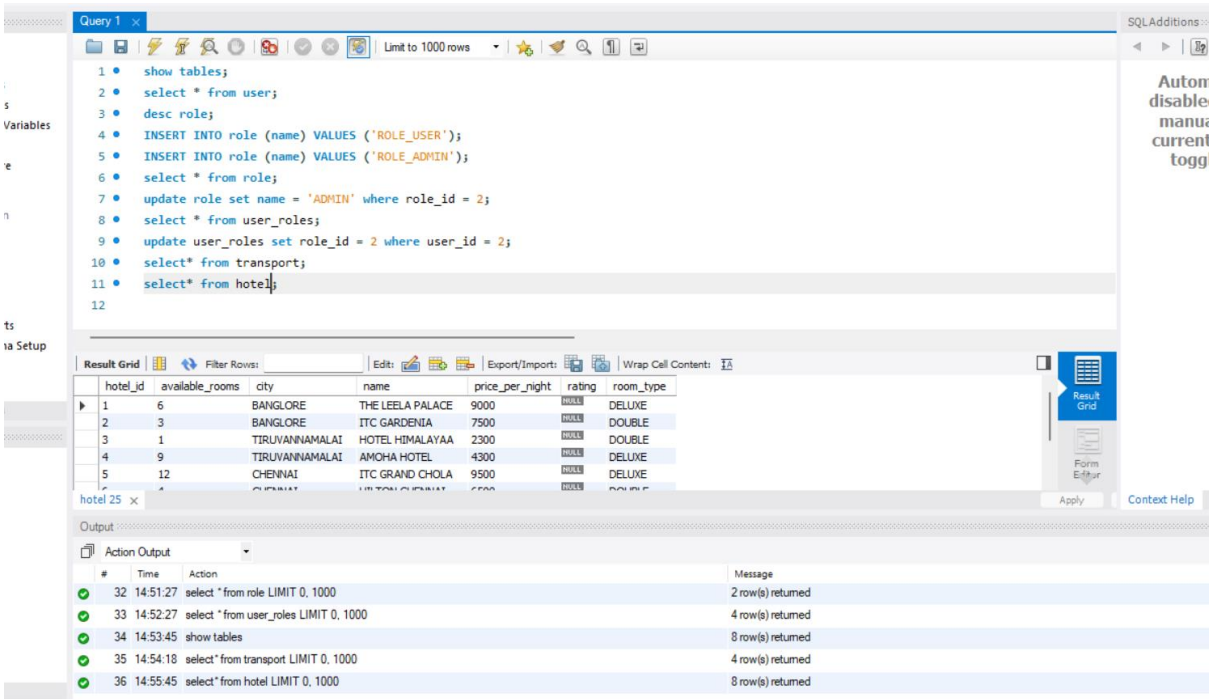


Figure: Hotel Data

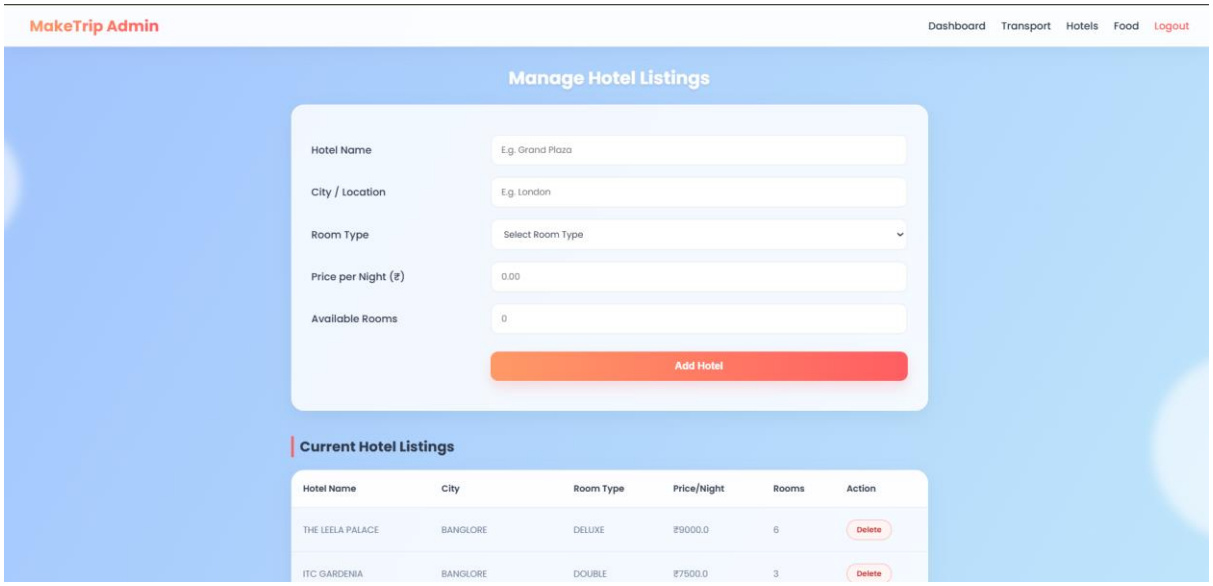


Figure: Hotel Manage Page

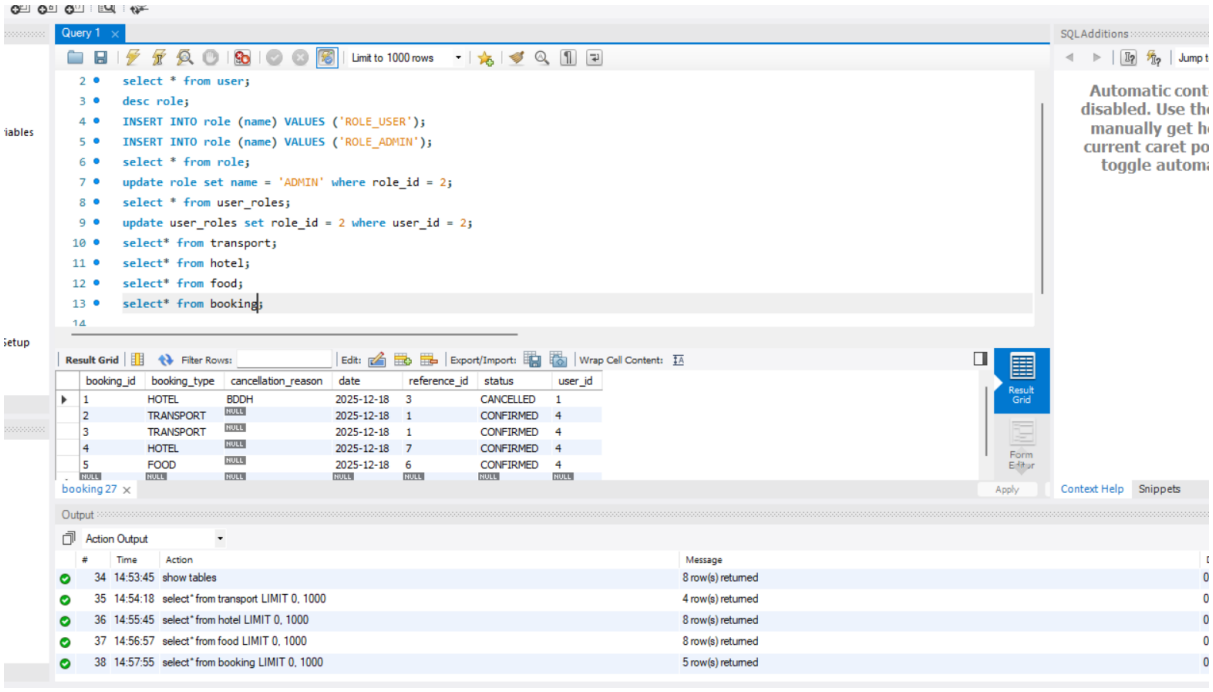


Figure: Booking Data

CONCLUSION

The **MakeTrip** project successfully automates the travel booking process. It integrates multiple services into a single architecture using Spring Boot. The system ensures data integrity, secure access control, and a user-friendly interface for both Customers and Administrators.

DRIVE LINK : https://drive.google.com/file/d/15OdzJcnIgkgp2YVtjcL4yim1-KYNfrs5/view?usp=drive_link