



Project on

“Railway Reservation System”

Submitted in partial fulfillment of the requirements for the award of degree of

**Bachelor of Technology
in
Computer Science & Engineering**

UE21CS352B – Object Oriented Analysis & Design using Java

Submitted by:

Srikrishna Sripati Nayak	PES1UG21CS620
Srikrishna B	PES1UG21CS618
Sudeep Dhotre	PES1UG21CS631
Srijan Badhya	PES1UG21CS616

Under the guidance of

Prof. Bhargavi Mokashi
Designation

January - May 2024

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013)

This is a draft report, will make edits to this within 10 hours

Problem Statement

This is a simple Railway Reservation System built using Java Spring Boot Framework and MySQL database. The system allows users to book train tickets and view their bookings.

Features:

User registration and authentication

Book train tickets

View booked tickets

Analysis and Design Models

1. Use Case Diagram

Software Design Principles Followed with screenshots of code

Single Responsibility Principle (SRP):

Each class and component in our system has a single responsibility. For example, controllers handle the user's requests, services contain business logic, repositories handle database operations, etc. This principle helps in making the codebase more modular, maintainable, and testable.


Dependency Injection (DI):

Spring Framework heavily utilizes Dependency Injection, allowing you to inject dependencies into classes rather than creating them within the class itself. This promotes loose coupling between components and makes our code more flexible and easier to test and maintain.

Model-View-Controller (MVC) Architecture:

Our project follows the MVC architectural pattern, where models represent data, controllers handle user input and interaction, and views display the user interface. This separation of concerns makes the codebase more organized and easier to maintain.

Open/Closed Principle (OCP):



Our codebase is designed to be open for extension but closed for modification. This means that we can add new features or make changes to existing ones without modifying the existing codebase extensively.

Software Design Patterns Used with screenshots of code

Builder Pattern:

The Builder pattern is implemented with the ReservationBuilder class. This pattern is used to construct a complex object step by step. In the context of our railway ticket booking system, the ReservationBuilder is used to create reservation objects by setting various attributes of the reservation in a step-by-step manner.

Facade Pattern:

Our services (ReservationService, TrainService, UserService) act as facades. The Facade pattern provides a simplified interface to a complex system of classes, making it easier to interact with. Each service encapsulates a part of the functionality related to reservations, trains, and users respectively, providing a simplified interface for the controllers to interact with the underlying business logic.

Chain of Responsibility Pattern:

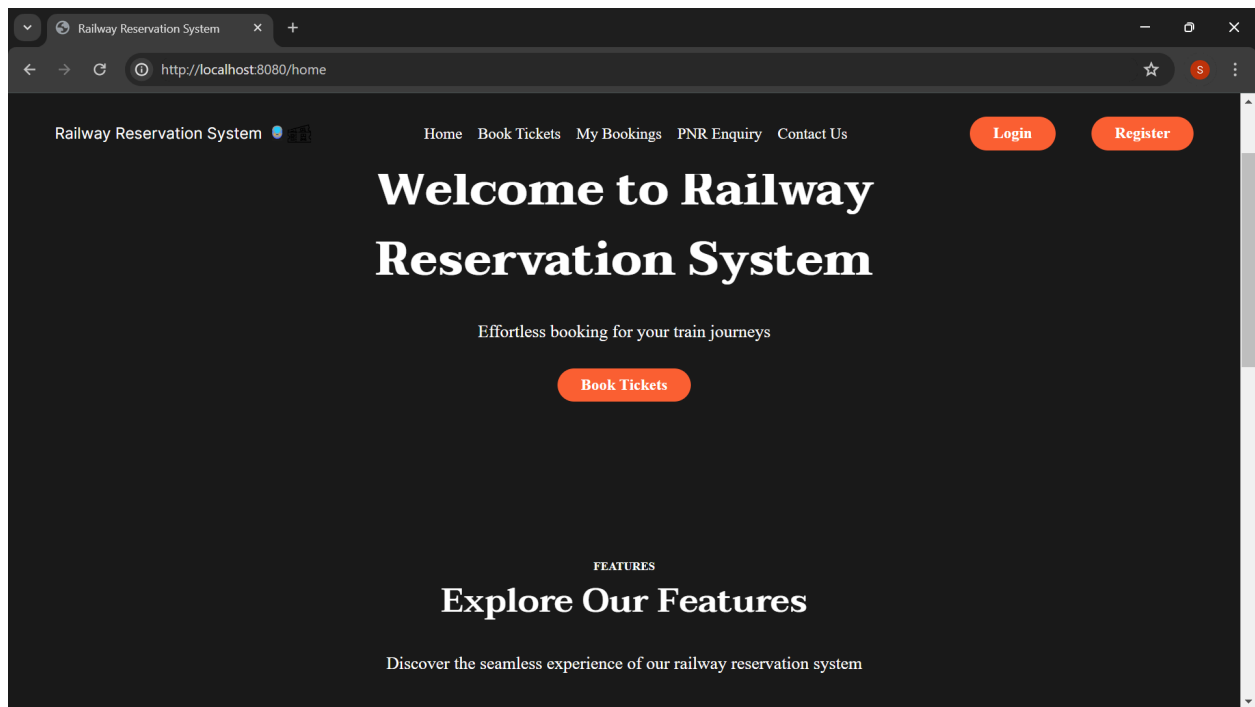
Spring Security is configured to enforce authentication for accessing all pages, which resembles the Chain of Responsibility pattern. In this pattern, a request is passed through a chain of handlers until it is handled by the appropriate handler. Similarly, in our application, each page request goes through Spring Security's filter chain, and if the user is not authenticated, they are redirected to the login page. Once authenticated, they gain access to the requested page.

Repository Pattern:

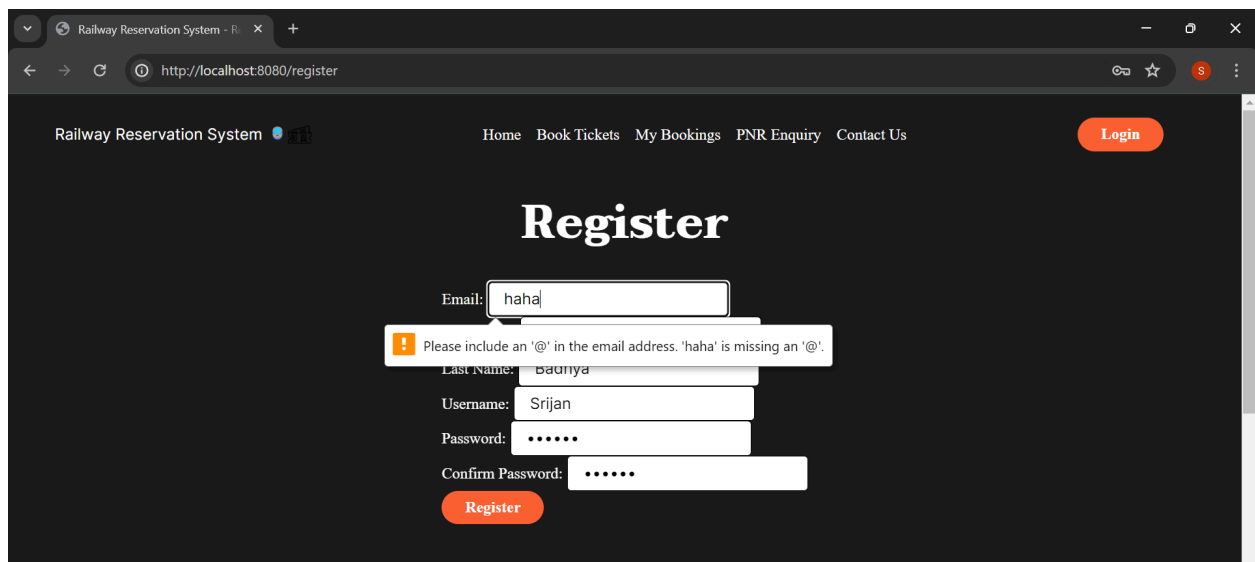
The Repository pattern is used with the repository classes (ReservationRepository, TrainRepository, UserRepository). This pattern provides a way to encapsulate data access logic, allowing the rest of the application to interact with the database without exposing the underlying database implementation details. Each repository is responsible for CRUD (Create, Read, Update, Delete) operations related to its corresponding entity (reservation, train, user).

User Interface Screenshots

Home Page, loaded when you access localhost8080



After clicking the register button, you are taken to the register page. The screenshot is an example of the register page not proceeding when you enter invalid information.



When you go to the login page and you enter information not in the database, you remain at the login page.

Railway Reservation System

Home Book Tickets My Bookings PNR Enquiry Contact Us

Register

Login

Username: Srija

Password:

Login

Invalid username or password.

Once you successfully log in, your username appears in the top right corner of the page.

Home Book Tickets My Bookings PNR Enquiry Contact Us

Srija

Logout

These are the options when you visit the booking page after logging in.

Railway Reservation System

Home Book Tickets My Bookings PNR Enquiry Contact Us

Book Tickets

Source: MANGALURU CNTL

Destination: KSR BENGALURU

Travel Date: 09-05-2024

Train: Select a train

Number: Select a train

UDYAN EXPRESS (11301)

UDYAN EXPRESS (11302)

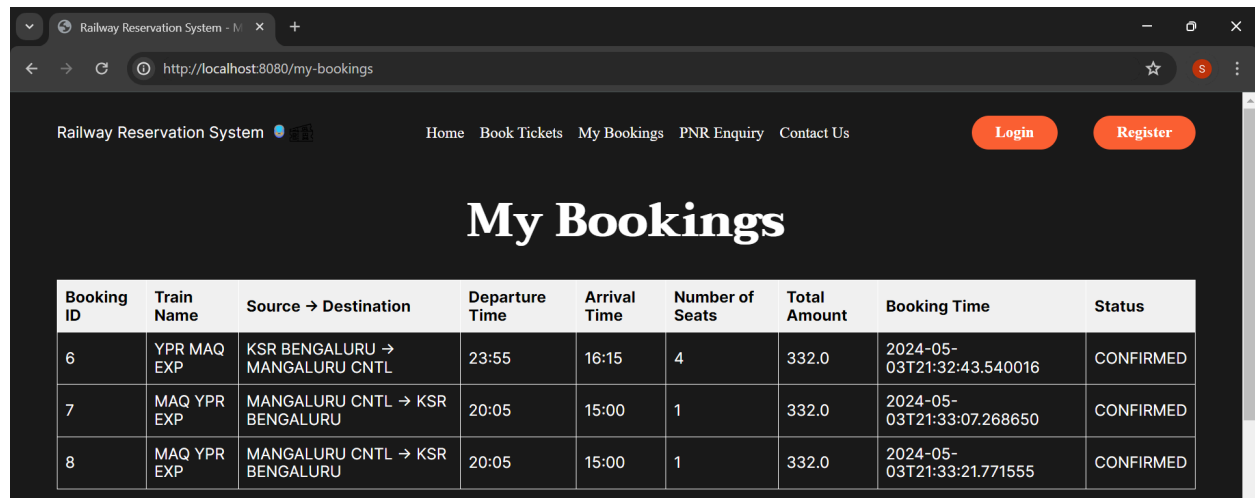
YPR MAQ EXP (16565)

MAQ YPR EXP (16566)

PANCHAGANGA EXP (16595)

PANCHAGANGA EXP (16596)

After you are done booking a ticket, you are taken to a page which displays the details of all your bookings.



The screenshot shows a web browser window with the URL `http://localhost:8080/my-bookings`. The page title is "Railway Reservation System". The navigation bar includes links for Home, Book Tickets, My Bookings, PNR Enquiry, and Contact Us. There are also "Login" and "Register" buttons. The main heading is "My Bookings". Below it is a table with the following data:

Booking ID	Train Name	Source → Destination	Departure Time	Arrival Time	Number of Seats	Total Amount	Booking Time	Status
6	YPR MAQ EXP	KSR BENGALURU → MANGALURU CNTL	23:55	16:15	4	332.0	2024-05-03T21:32:43.540016	CONFIRMED
7	MAQ YPR EXP	MANGALURU CNTL → KSR BENGALURU	20:05	15:00	1	332.0	2024-05-03T21:33:07.268650	CONFIRMED
8	MAQ YPR EXP	MANGALURU CNTL → KSR BENGALURU	20:05	15:00	1	332.0	2024-05-03T21:33:21.771555	CONFIRMED