**Project Report**

**Train Booking System**

**Index**

|  |  |  |
| --- | --- | --- |
| **Sr.no** | **Topic** | **Page no** |
| 1 | Title of Research | 5 |
| 2 | Certificate | 2 |
| 3 | Abstract | 5 |
| 4 | Introduction | 6-9 |
| 5 | Objective | 9- |
| 6 | Programming Language | 9-18 |
| 7 | Data gathering | 19 |
| 8 | System design /UML Diagrams | 20-22 |
| 9 | Proposed system features | 23-25 |
| 10 | Evalution | 26-27 |
| 11 | Future scope | 28 |
| 12 | conclusion | 29 |
| 13 | Biography | 30 |
| 14 | References | 31 |

**Acknowledgement**

I take this opportunity to express my profound gratitude and deep regards to my teachers Prof.Rajashri Thete for their exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by them time to time shall carry me a long way in the journey of life on which I am about to embark.

Special gratitude I give to my respected head of the division Rajashri Thete , for allowing me to use the facilities available and also help me to coordinate my project

Furthermore, I would also like to acknowledge with much appreciation the crucial role of faculty members on this occasion.

Lastly, I thank almighty, my parents and friends for their constant encouragement without which this project would not be possible.

**Abstract**

The ***Train Booking System*** is a web-based platform designed to simplify the ticket reservation process for both passengers and administrators. It provides a user-friendly interface that allows users to search for trains based on source and destination stations, view train schedules, select travel classes, and book tickets efficiently. The system supports secure user registration, login, and booking history tracking, enhancing the overall travel experience.

For administrators, the Train Booking System offers robust tools to manage train routes, schedules, seat availability, and passenger data through a centralized dashboard. Features such as booking management, cancellation handling, and reporting functions enable efficient decision-making and streamlined operations. Integration with a secure payment gateway ensures that transactions are processed safely and smoothly.

The platform is designed to be fully responsive and accessible on desktops, tablets, and smartphones, providing a consistent user experience across devices. By automating essential processes such as ticket reservations, seat allocation, and schedule updates, the Train Booking System helps reduce manual workload, minimize errors, and improve service delivery. Its scalable design makes it ideal for use in both regional railway networks and national transportation systems.

Introduction

In today’s age of Information Communication and Technology, we are surrounded by technology at every moment. From the moment we wake up to the time we go to bed, technology plays a crucial role in our lives. One of the most impactful developments in this digital era is the advancement of online transportation services. These systems allow people to plan and book their travel conveniently, saving both time and effort while offering flexible and accessible solutions from anywhere.

Our project aims to develop an online **Train Ticket Booking System** that allows users to search for trains and reserve tickets quickly, securely, and efficiently. This platform provides a better alternative to traditional ticket booking methods by offering a seamless and user-friendly digital experience. Users can search for available trains based on routes and dates, check seat availability, and complete bookings through a secure and streamlined process.

The system also features an administrative panel that allows the management of train schedules, routes, seat inventory, and passenger information. With a responsive design and easy navigation, the Train Ticket Booking System ensures accessibility on various devices, making the booking process more flexible and convenient for users.

### **1.1 Objective of the Present Work**

The objectives of this project are as follows:

* To develop a web application for a Train Ticket Booking System that allows users to book tickets online conveniently.
* To provide a platform where users can search trains, check availability, and make secure reservations.
* To enable real-time updates of train schedules, seat availability, and fare details.
* To deliver a booking experience that is easy-to-use, visually appealing, and interactive.
* To ensure fast, efficient, and secure transactions through an integrated payment and ticketing system.
* To make it possible for anyone, anywhere, at any time to access the booking system via the internet at a low cost.

## **System Analysis**

### **3.1 Problem Definition**

Many passengers still rely on traditional methods to book train tickets, such as standing in long queues or dealing with limited offline booking options. These methods are time-consuming, inconvenient, and prone to human error. Existing online booking platforms may be costly, complex, or lack essential features for smaller railway services. Users also face challenges like poor system navigation, unavailability of real-time train data, and insecure payment gateways.

The Train Ticket Booking System addresses these issues by offering a secure, user-friendly, and scalable platform for passengers to book tickets online. It allows real-time train search, booking, cancellation, and payment processing. Admins can efficiently manage train schedules, routes, seat availability, and passenger data from a centralized dashboard.

### **3.2 Preliminary Investigation**

**Purpose**  
The purpose of this project is to develop a responsive and efficient web-based train ticket booking system that simplifies ticket reservations and streamlines railway operations. It replaces manual systems with a digital platform that supports real-time updates and secure bookings.

**Benefits**

* **Quick Ticket Booking:** Passengers can search trains and book tickets easily.
* **Efficient Admin Control:** Admins manage schedules, trains, and seat availability with ease.
* **User Accessibility:** The system is available online 24/7, allowing booking from anywhere.
* **Secure Transactions:** Integrated payment support ensures secure, fast transactions.

### **3.3 Feasibility Study**

**Technical Feasibility**

* Built using PHP and MySQL with HTML/CSS/JavaScript for the frontend.
* Compatible with modern web browsers and supports mobile responsiveness.

**Economic Feasibility**

* Uses open-source tools to minimize cost.
* Improves operational efficiency and reduces overhead.

**Operational Feasibility**

* Simple user interface with minimal training required.
* Supports smooth workflows for booking and administration.

**Schedule Feasibility**

* Follows a phased development cycle with realistic milestones.

**Social Feasibility**

* Encourages digital ticketing habits and reduces in-person congestion.
* Accessible to a wide range of users with basic internet skills**.**

### **3.4 Project Planning**

**Purpose of Project Planning**  
Project planning ensures that the development of the Train Ticket Booking System follows a systematic and strategic approach. It helps outline major activities, allocate resources efficiently, identify potential risks, and set realistic timelines to ensure the project is completed successfully and on schedule.

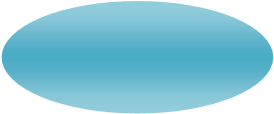
**Phases Covered in the Plan**  
The project plan includes the following phases:

1. **Preliminary Investigation** – Understanding the goals, scope, and objectives of the train booking system.
2. **System Analysis** – Identifying current challenges in manual or outdated booking systems and gathering functional requirements.
3. **System Design** – Designing the system architecture, database schema, and user interface for passengers and administrators.
4. **Coding** – Developing backend logic in PHP and creating user-friendly frontends using HTML, CSS, and JavaScript.
5. **Security** – Implementing secure login, data protection, and safe online payment handling.
6. **Testing** – Performing unit testing, integration testing, and final acceptance testing to ensure the system functions correctly.
7. **Implementation** – Launching the system on a web server and supporting ongoing maintenance and enhancements.

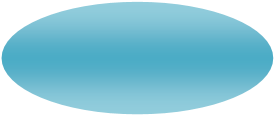
**3.5 Project Scheduling**

Activity diagram

Grant chart diagram



**Stop**



**Start**



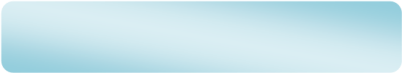
Preliminary



System Analysis



System Design



Coding



Testing

Security



Implementation

### **Software & Hardware Requirements**

#### **Software Requirements**

* **Frontend:**
  + HTML, CSS, JavaScript, Bootstrap – for designing a responsive and interactive user interface.
* **Backend:**
  + PHP – for handling server-side logic and managing application workflows.
* **Database:**
  + MySQL – to store user data, train details, routes, booking history, and payment records.
* **Payment Integration:**
  + Razorpay or PayPal – for secure and reliable online ticket payments.
* **Web Server:**
  + Apache – to deploy and host the application efficiently.
* **Authentication & Security:**
  + Secure session handling and password encryption for safe login and user data protection.

### **Hardware Requirements**

* **Processor:** Intel Core i5 or higher
* **RAM:** Minimum 8GB
* **Storage:** At least 100GB for storing user data, travel history, and logs
* **Connectivity:** High-speed internet for real-time booking and admin operations
* **Display:** Minimum 1366x768 resolution for proper layout and navigation

## **Functional Requirements**

### **1.** **User Module**

**Users can:**

* **Search Trains** – View available trains based on selected source, destination, and travel date.
* **Check Seat Availability** – View real-time seat availability for different classes (e.g., Sleeper, AC).
* **User Registration/Login** – Create a user account, log in securely, and manage personal details.
* **Book Tickets** – Select train, class, number of passengers, and proceed with booking via secure payment.
* **Cancel Tickets** – Cancel booked tickets with applicable refund policies.
* **View Booking History** – Access past bookings with ticket details and status.
* **Download E-Ticket** – Generate and download a digital copy of the ticket.
* **Track Booking Status** – Check the real-time status of upcoming or waitlisted tickets.

### **2. Admin Module**

Admins have full control over the Train Ticket Booking System and can:

* **Secure Login System** – Access the admin dashboard using authenticated login credentials.
* **Train Management** – Add, edit, delete, and restore train details including train name, number, classes, and capacity.
* **Route & Schedule Management** – Define train routes, stations, and timing schedules for each train.
* **Booking Management** – View, update, cancel user bookings, and handle refund processes.
* **User Management** – View, manage, or deactivate registered user accounts and monitor user activity.
* **Sub-Admin Management** – Create sub-admin accounts and assign specific permissions for limited access tasks.
* **Reports and Analytics** – Generate reports on ticket sales, user statistics, train occupancy, and cancellation trends.
* **Page Management** – Manage content of static pages such as ‘Privacy Policy’, ‘Help & Support’, and ‘Contact Us’.

### **3. Sub-Admin Module**

Sub-Admins can:

* **Manage Trains & Routes** – Add, update, or delete train schedules and route details within assigned regions or categories.
* **Moderate Bookings** – Handle booking approvals, cancellations, and refunds for designated routes.
* **Limited Access** – Work within permission levels defined by the admin, with no access to sensitive system-wide configurations.
* **Customer Support Handling** – Respond to passenger queries and assist users with booking or train-related issues.

## **3.8 Software Engineering Paradigm**

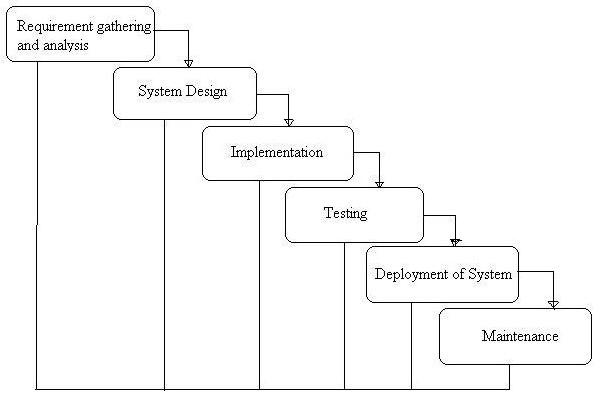
The development of the **Train Ticket Booking System** adopts a systematic approach to ensure the solution is functional, secure, and maintainable. The selected paradigm guides each phase from analysis to implementation while allowing improvements through feedback and testing.

### **Development Model: Adapted Waterfall Model**

The Waterfall Model, a sequential software development lifecycle, is used with certain enhancements to improve flexibility. Feedback loops are incorporated to refine stages such as coding and testing based on practical insights.

**Key Adaptations in the Waterfall Model:**

1. **Structured Phase Progression** – Each stage follows a clear order, making the project easier to track and manage.
2. **Iterative Refinements** – Testing outcomes inform code improvements, enabling early detection and resolution of issues.
3. **Defined Milestones** – Deliverables for each stage are specified and reviewed before proceeding to the next phase.
4. **Flexible Adjustments** – Certain phases may overlap when beneficial, allowing dynamic improvements without disrupting overall flow.



**Phases of Development**

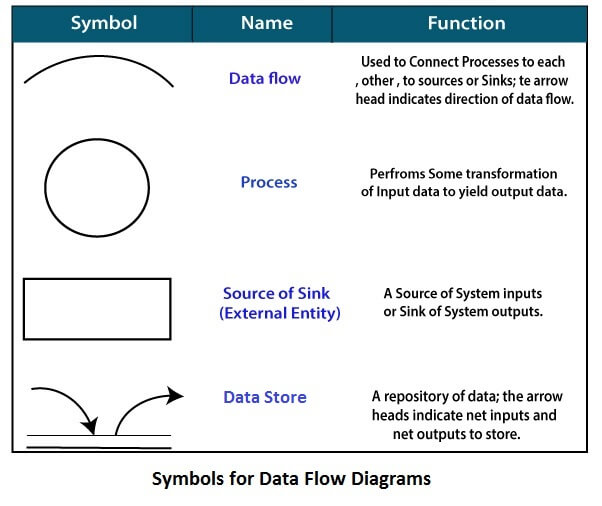
1. **Requirement Analysis & System Study**
   * Identify project goals, challenges, and functional requirements for the e-commerce system.
   * Gather stakeholder inputs and define core features such as product management, user accounts, and payment processing.
2. **System Design**
   * Design database schemas, system architecture, and module interactions.
   * Develop user interface layouts focused on usability and responsiveness for multiple devices.
3. **Implementation (Coding)**
   * Backend development using Python (Django) for business logic and data management.
   * Frontend development with HTML, CSS, JavaScript, and Bootstrap to create an engaging UI.
   * Database integration with MySQL for secure storage of product, order, and user data.
   * Integration of payment gateways and security measures.
4. **Testing & Debugging**
   * Conduct unit tests, integration tests, and user acceptance tests to validate system performance.
   * Debug identified issues to improve reliability and user experience.
5. **Deployment & Maintenance**
   * Deploy the application on a scalable, secure web server.
   * Provide ongoing maintenance, including bug fixes, updates, and feature enhancements based on user feedback.

**3.9 Data Flow Diagram (DFD)**

A Data Flow Diagram (DFD) is a visual representation that illustrates how data moves through the Easy Shop e-commerce system. It provides a clear graphical view of how information enters, is processed, stored, and exits the system.

The main purpose of the DFD is to outline the scope and boundaries of the Easy Shop system and to serve as an effective communication tool between system analysts, developers, and stakeholders. It can also be used as a foundation for system redesign or enhancement.

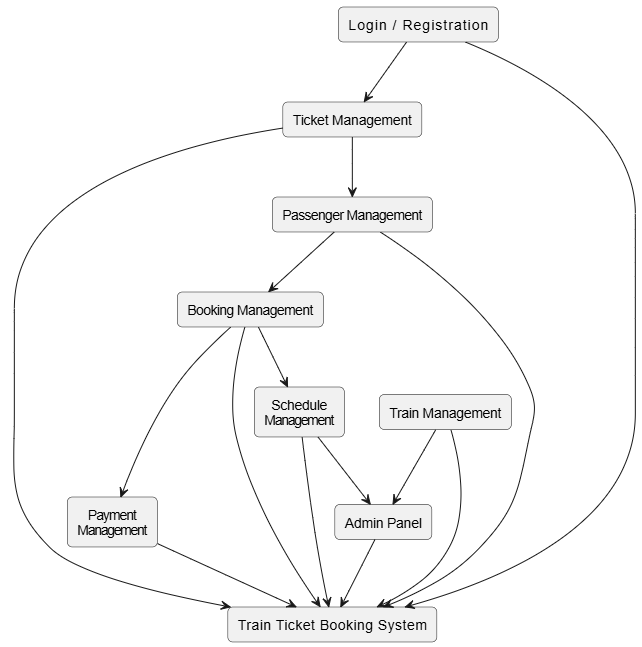
**Important Points about DFDs:**

1. **Unique Naming:** Every element in the DFD (process, data store, external entity) should have a unique name for easy reference.
2. **Data Flow, Not Process Flow:** Unlike flowcharts, arrows in DFDs represent data flow, not the sequence of events or control flow.
3. **No Logical Decisions:** Avoid including decision points (diamonds) in a DFD since it does not represent control logic or conditional flows.
4. **Simplicity:** Avoid overloading the diagram with details. Defer error handling and exception flows until later stages of analysis. ******

**Standard Symbols Used in DFDs:**

* **Process (Circle/Bubble):** Represents a process that transforms input data into output data.
* **Data Flow (Arrow):** Shows the movement of data between processes, data stores, and external entities.
* **Data Store (Parallel Lines):** Indicates storage locations where data is held for later use.
* **External Entity (Square/Rectangle):** Represents sources or destinations of data outside the system, such as customers, payment gateways, or suppliers.

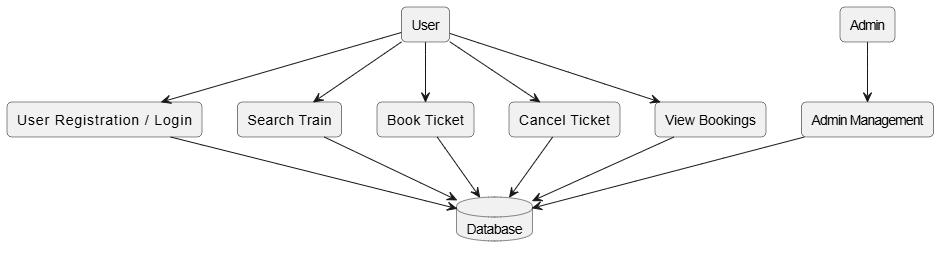
**Zero Level DFD :**

****

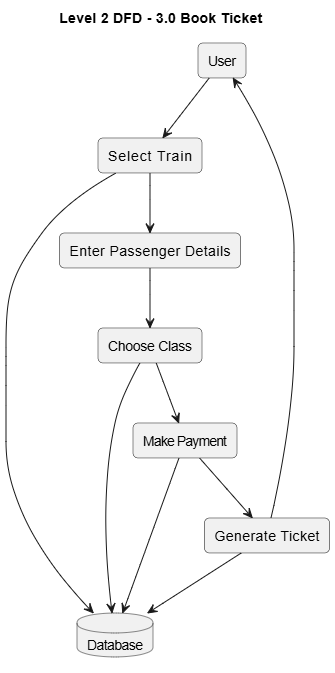
The Level 0 DFD (Context Diagram) of the Easy Shop system illustrates the key external entities interacting with the system and the major data flows. These include:

* **Users (Customers):** Browsing products, placing orders, making payments, and writing reviews.
* **Admin:** Managing products, orders, user accounts, and generating reports.
* **Payment Gateway:** Processing secure payment transactions.
* **Warehouse:** Managing inventory updates based on orders.
* **Shipping Service:** Handling delivery information and tracking.

The diagram highlights how data such as product information, user details, orders, payment confirmation, and shipment status flows between these entities and the core Easy Shop system, showcasing the overall system workflow.



The **First-Level DFD** shows how the system handles key functions like login, admin, sub-admin, authorization, and password management, and connects them to content-related modules such as news, category, subcategory, webpage, and user comments management for smooth portal operations



The **Second-Level DFD** shows the internal working of the admin in the ONPS system. It includes processes like login, credential checking, role-based access, and module management. The admin can manage subadmins, categories, subcategories, news, webpages, and user comments, along with profile updates, password changes, and performing sentiment analysis.

**Entity Relationship Diagram(ERD)**

The **ER Diagram** of the **Easy shop Ecommerce System** shows key relationships between **Users, Admins, News Articles, Categories, Comments, and Sentiment Analysis**, ensuring structured data management and insightful moderation

**4. System Design**

### **4.1 Modules**

#### **1. User Module**

Users (Passengers) can:

* **Search Trains** by source, destination, travel date, and class.
* **View Train Details** including train name, number, timings, seat availability, and fare.
* **Register and Log In** to create and manage their account.
* **Book Tickets** by selecting train, class, number of seats, and entering passenger details.
* **Make Secure Payments** using multiple payment options (credit/debit cards, UPI, wallets).
* **Download/Print E-Tickets** or receive via email/SMS.
* **View Booking History** including past and upcoming journeys.
* **Cancel Tickets** and view refund status as per cancellation policies.
* **Access Support** for inquiries, complaints, or ticketing help.

#### **2. Admin Module**

Admins have full access to system management and can:

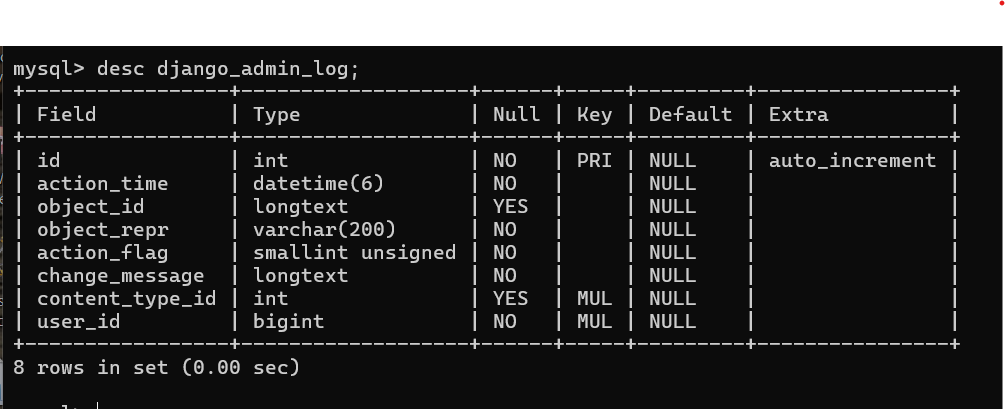
* **Secure Login** with role-based access to the admin panel.
* **Train Management** – Add, update, or delete train details, schedules, and seat classes.
* **Route Management** – Define train routes, stops, and timings.
* **Booking Management** – View, monitor, and manage ticket bookings and cancellations.
* **User Management** – View and manage registered user profiles and activity.
* **Reports & Analytics** – Generate reports on bookings, revenue, occupancy rates, and user activity.
* **Support Management** – Monitor and respond to customer support requests.
* **Fare Management** – Set and modify fare structures and dynamic pricing rules.
* **Promotion Management** – Create promotional codes or discounts for special occasions.

**4.2 DATA STRUCTURE OF ALL MODULES:**

We have organized one database “**Train Booking System”** for system design. It can be accessed directly or sequentially by registered. The database determines files, record, fields, and characters. It can be easily controlled and updated. **“Train Booking System” contains 15 MySQL tables(In this MySQL 6 table is customized and 9 table made by default in django) :**

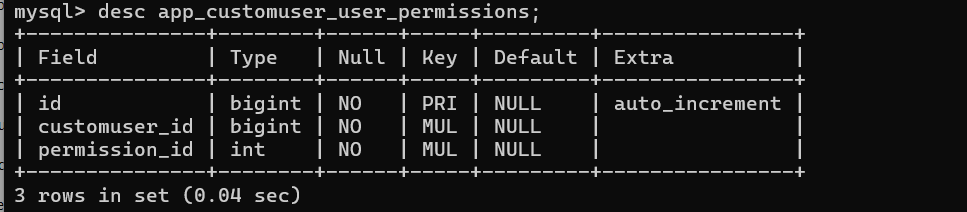
**Customized Tables Details**

**Admin Table(django\_admin) :**

This store admin personal and login details.

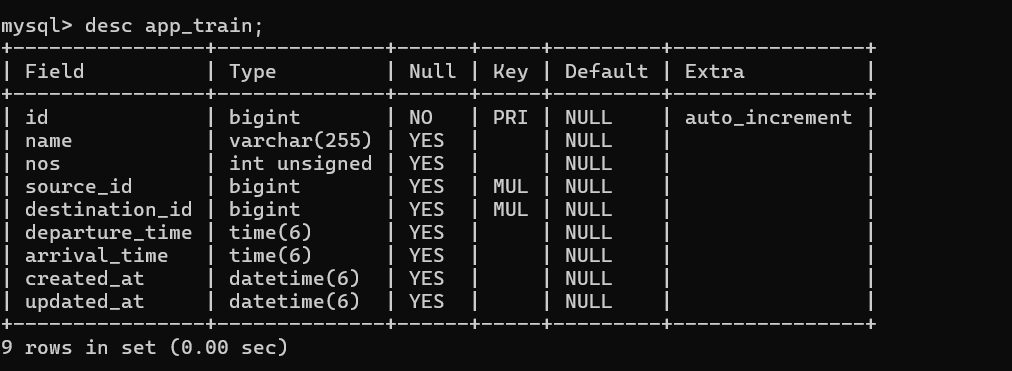
**Table name(**app\_customeuser\_user\_permissions)

This table store the details of user



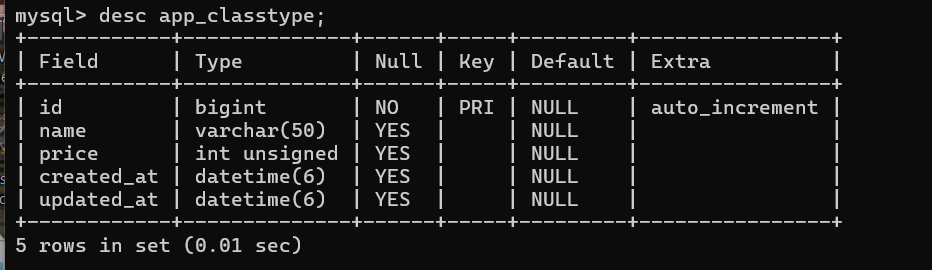
**Table name(**app\_train)

This table store the details of trains



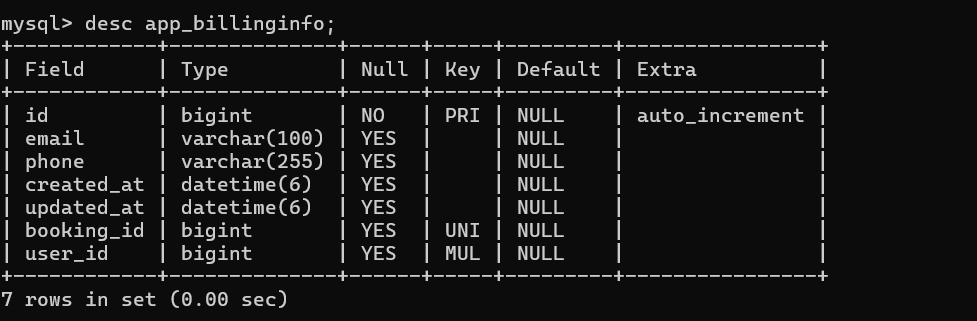
**Table name(app\_train\_class\_type**)

This table store the details of train class



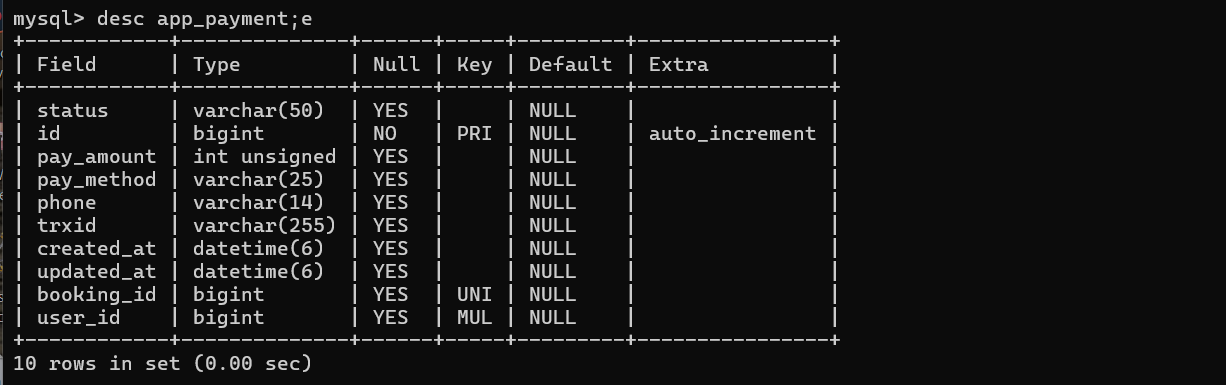
**Table name(**app\_billinginfo)

This table store the details of information of billing



**Table name(**app\_payment)

This table store the details of payment



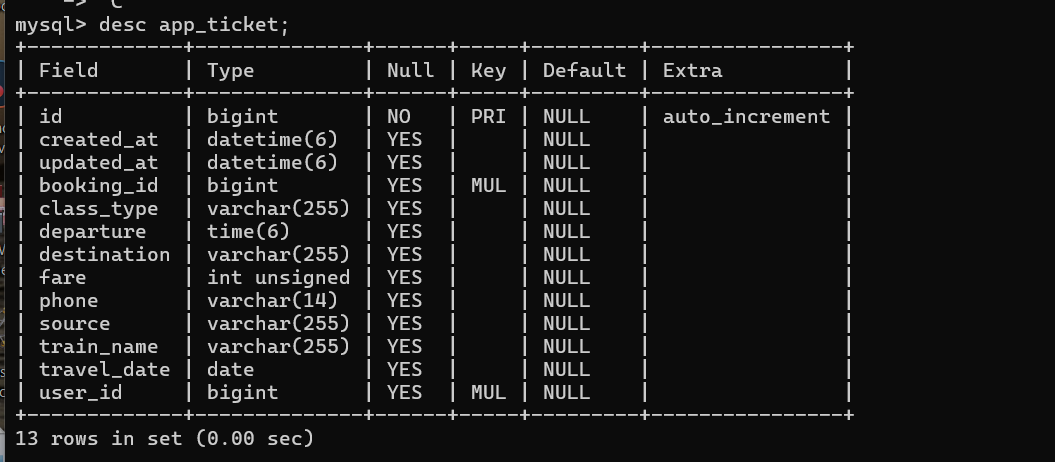
**Table name**(auth\_group)

This table store the details of auth group

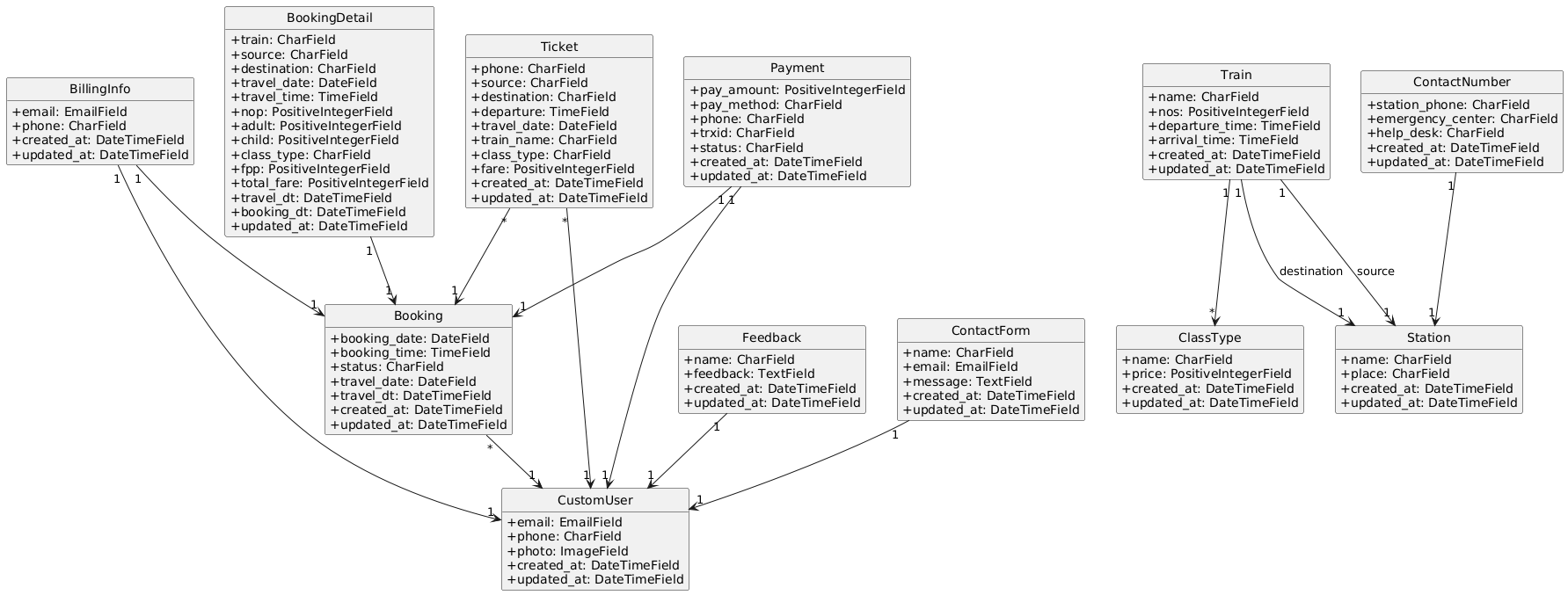


**Table name(app**\_**ticket)**

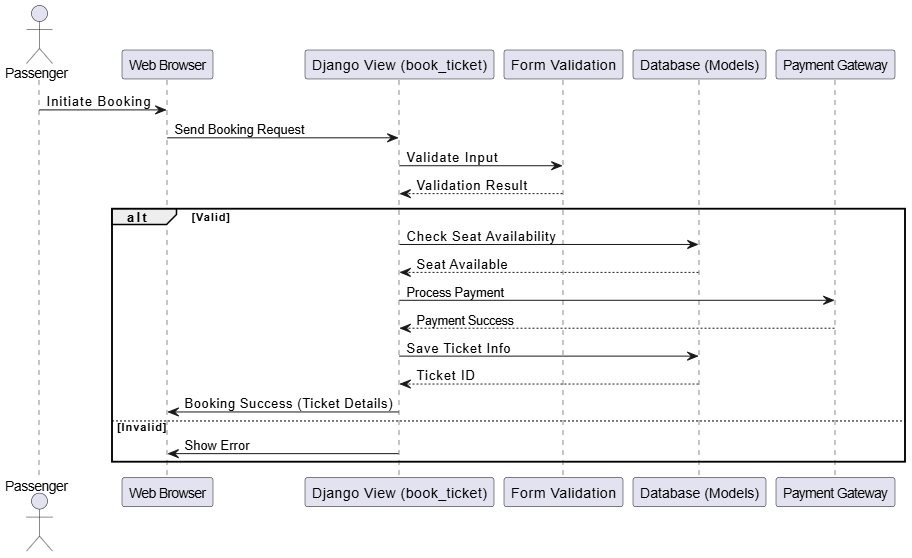
This table store the details of ticket



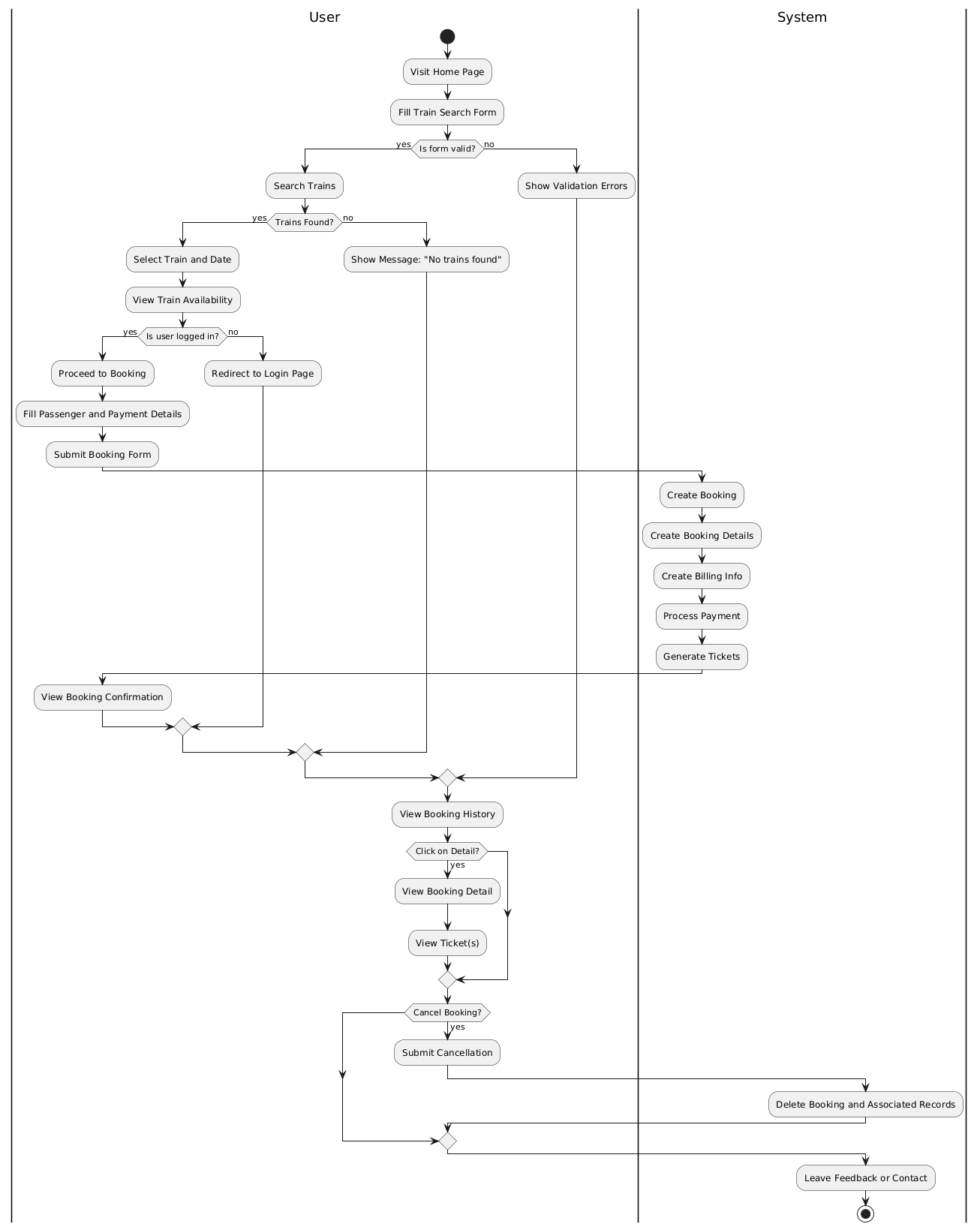
**Class Diagram:**

****

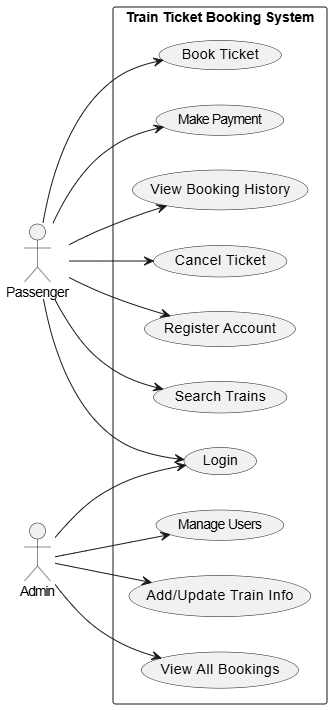
## **Sequence Diagram:**



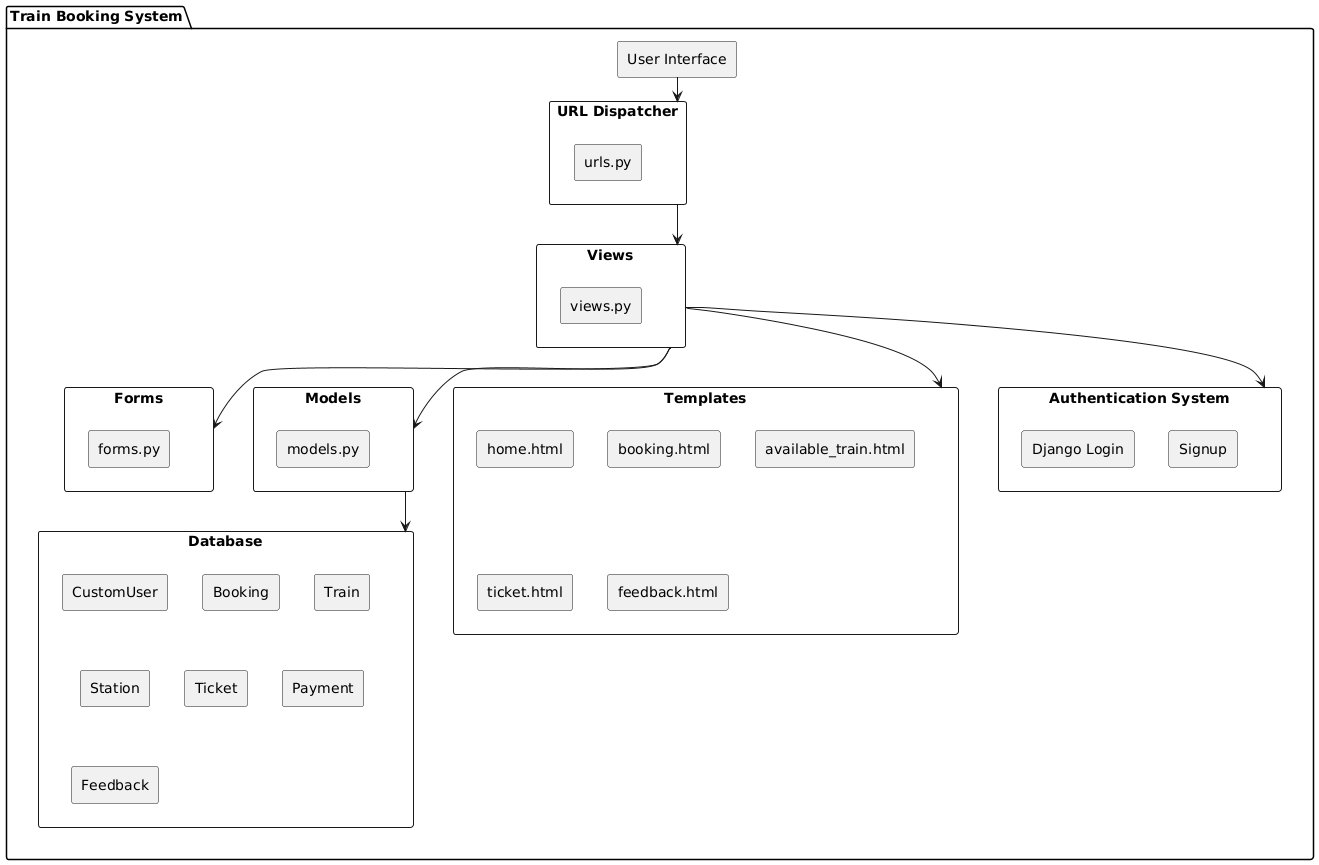
## **Activity Diagram:**

****

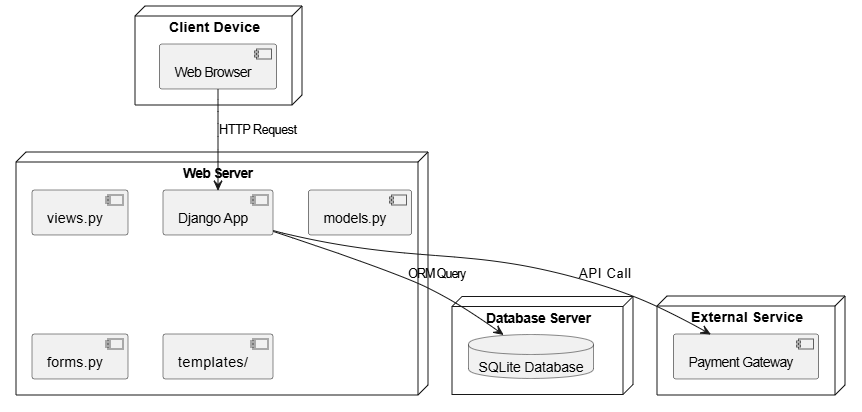
## **Use Case Diagram :**



## **Component Diagram:**



## **Deployment Diagram:**



## **4.3 Procedural Design**

### **4.3.1 User Panel Design**

The **User Panel** in Train Booking System is designed to allow passengers to search for trains, book tickets, and manage bookings effortlessly. The interface is intuitive and mobile-responsive for accessibility.

* Users land on the homepage, where they can search for trains by entering source, destination, journey date, and class.
* Available trains matching the criteria are listed along with seat availability, fare, and timings.
* Users select a train and proceed to enter passenger details.
* After reviewing the summary, users proceed to the secure payment gateway.
* Upon successful payment, an e-ticket is generated, and booking confirmation is displayed.
* Users can then:
  + View or download their ticket.
  + Track booking history and status.
  + Cancel bookings (if eligible) and request refunds.
  + Contact support for help.

**The flow of user actions from homepage to ticket confirmation is visually represented in the following flowchart:**  
(Add flowchart showing steps: Search → View Trains → Book → Payment → Confirmation)

### **4.3.2 Admin Panel Design**

The **Admin Panel** is protected by secure login and provides access to all core system management features.

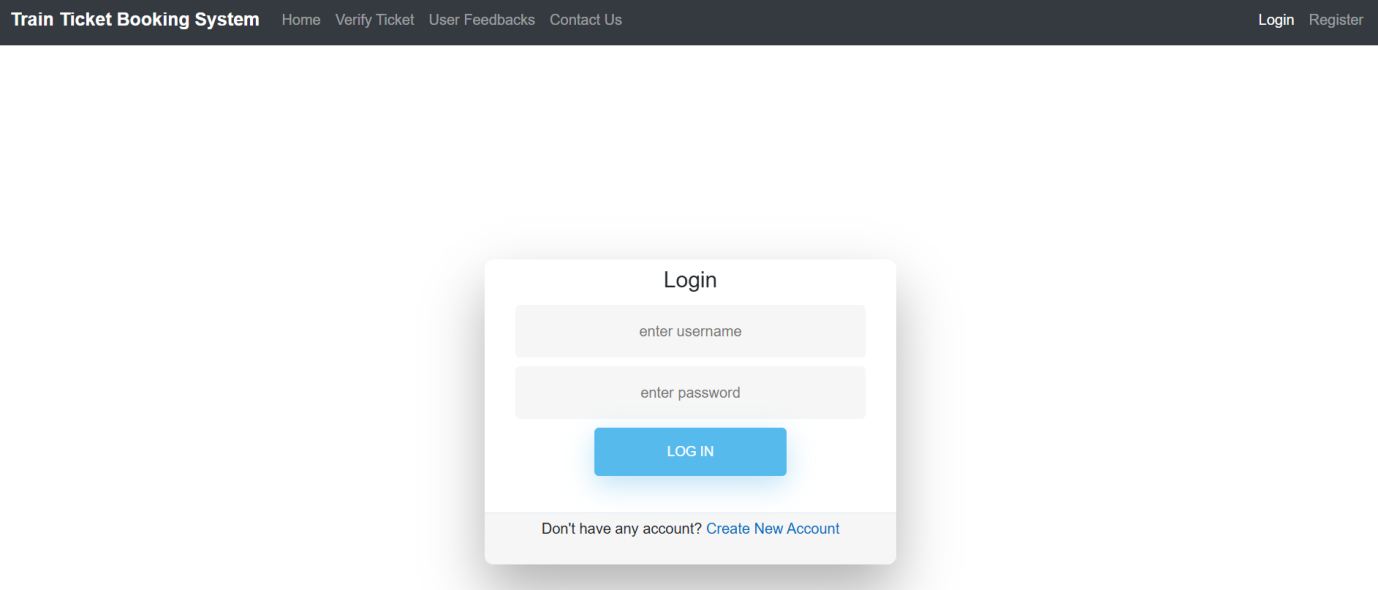
* The admin visits the login page and enters their username and password.
* Upon clicking the login button, the system validates credentials against the database.
* If the login is successful:
  + A session is initiated, and the admin is redirected to the dashboard.
  + From the dashboard, the admin can:
    - Manage train details and schedules.
    - Monitor ticket bookings and cancellations.
    - View user accounts and activity.
    - Generate reports and view system analytics.
    - Respond to customer inquiries or issues.
* If the login fails:
  + An error message is displayed prompting the admin to retry.
* Unauthorized users cannot access the admin panel without authentication.

**Screenshots**

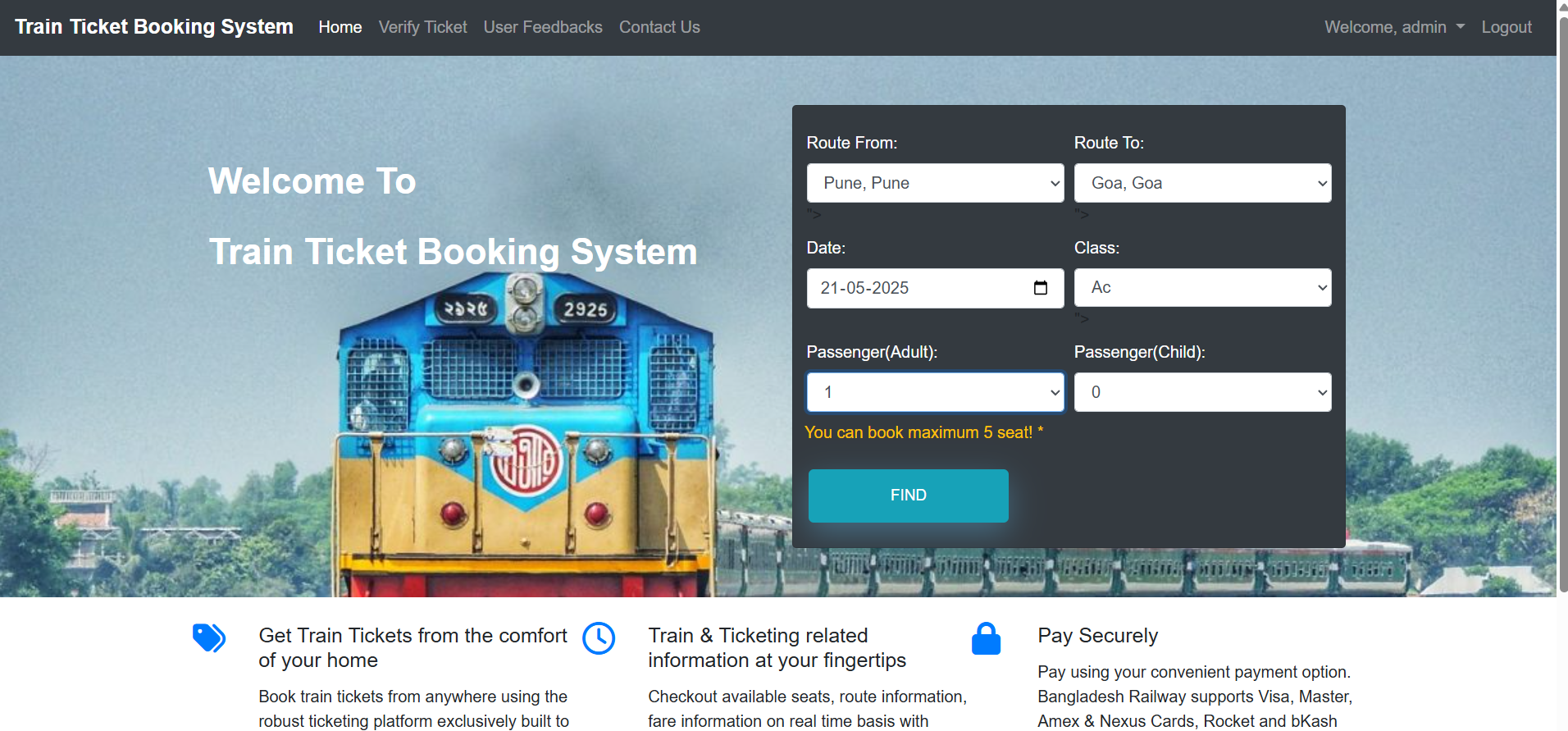
**User Registration:**



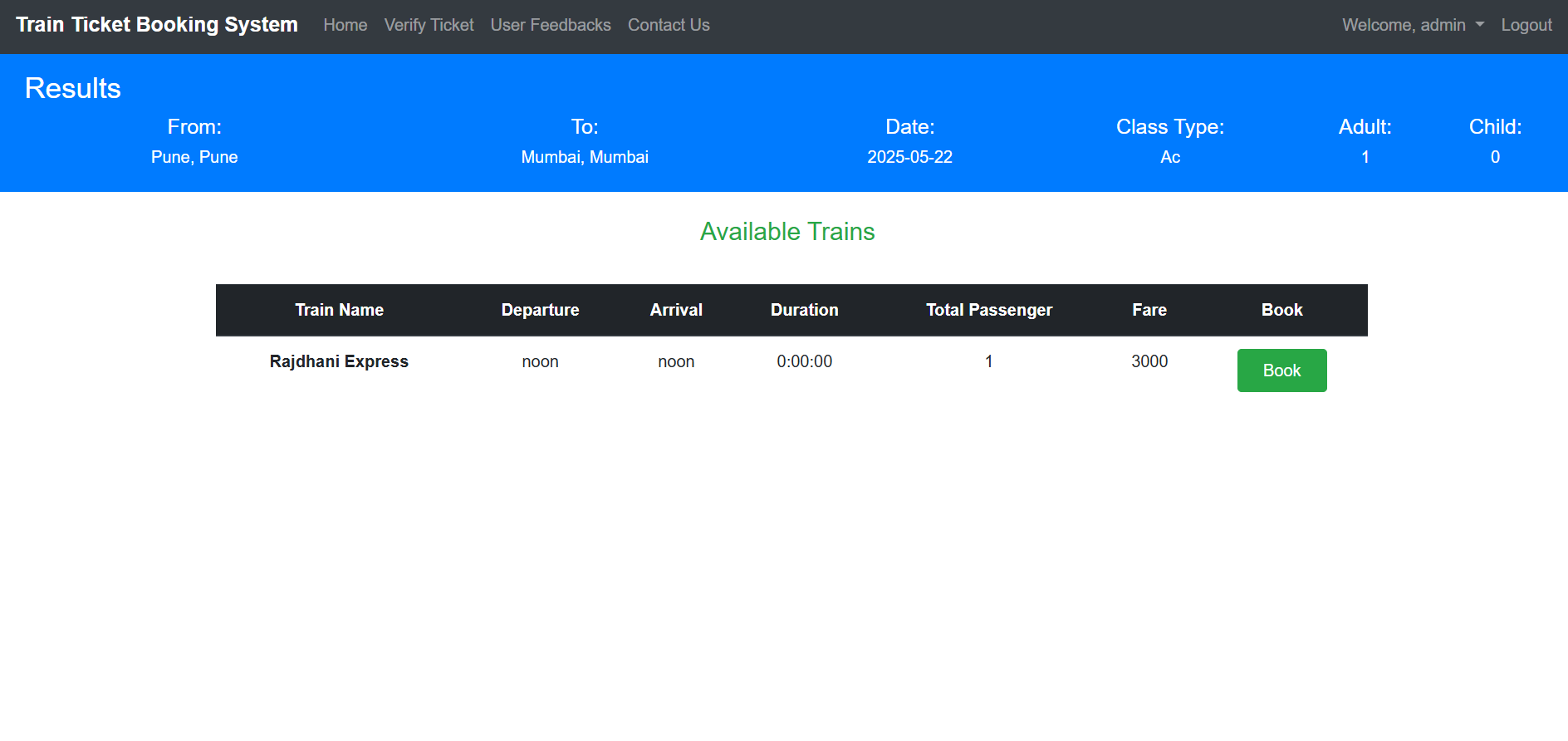
**User Login:**



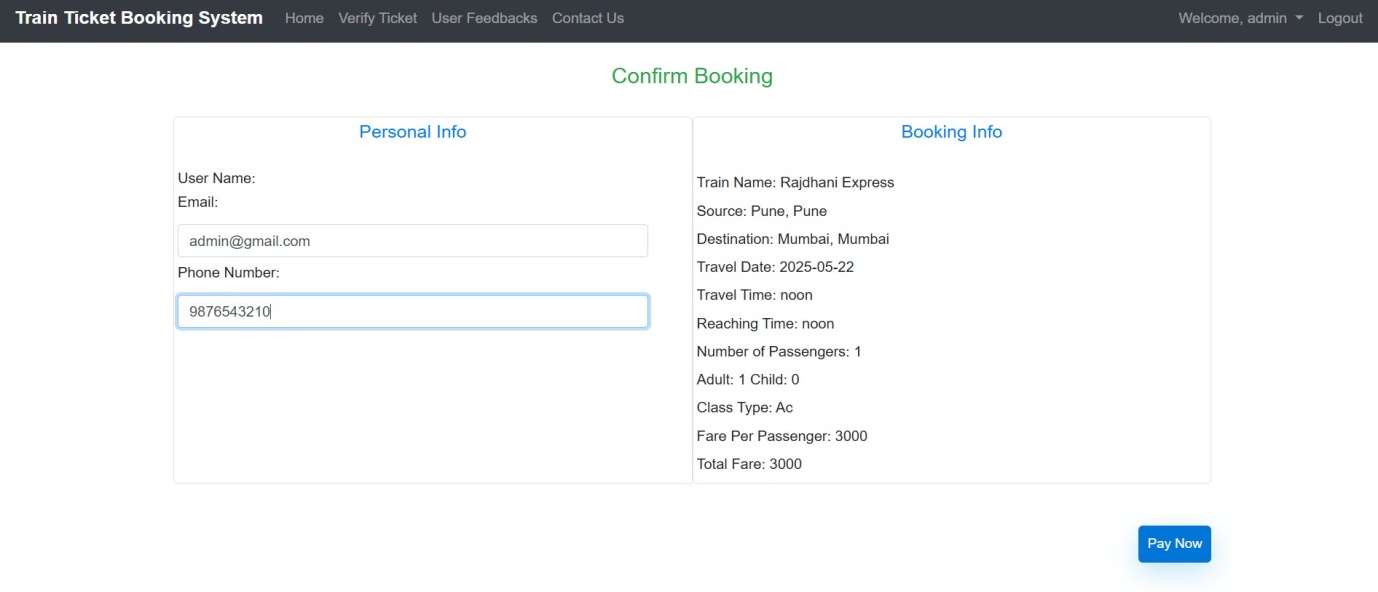
**Home Page:**

****

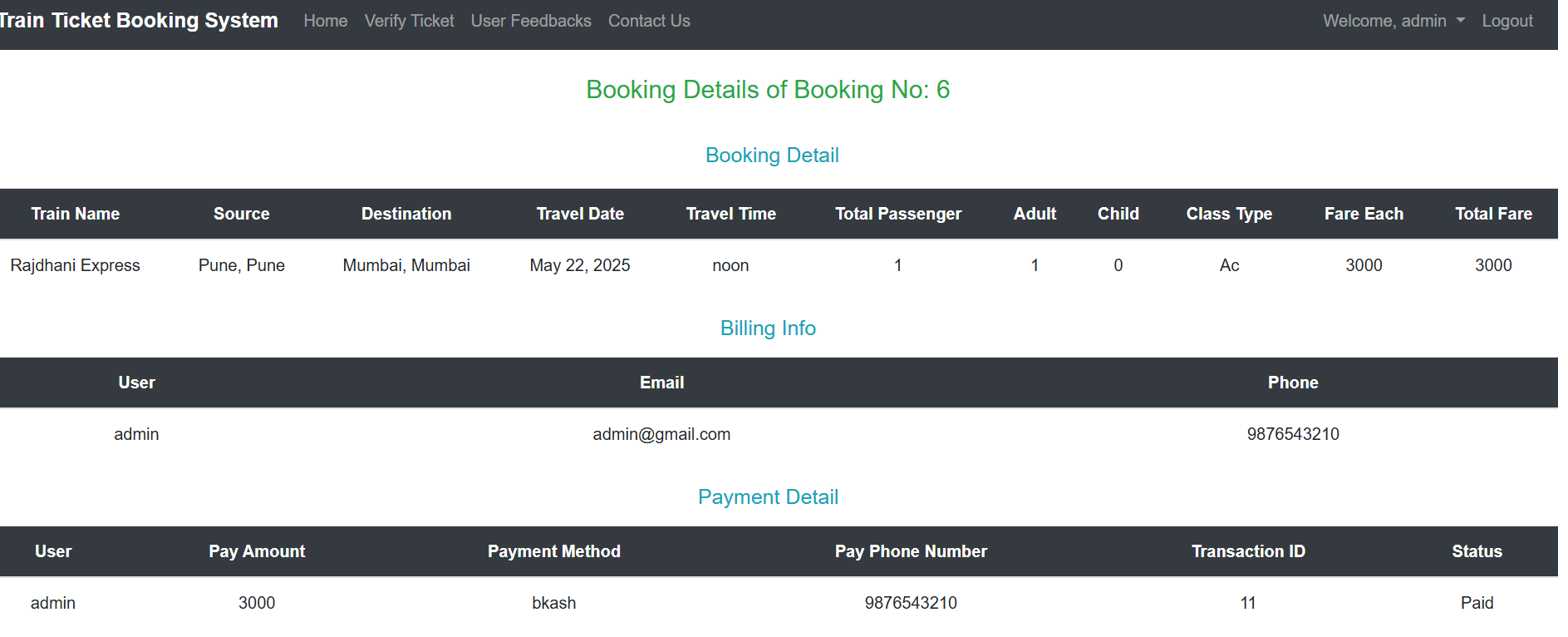
**Find Trains:**

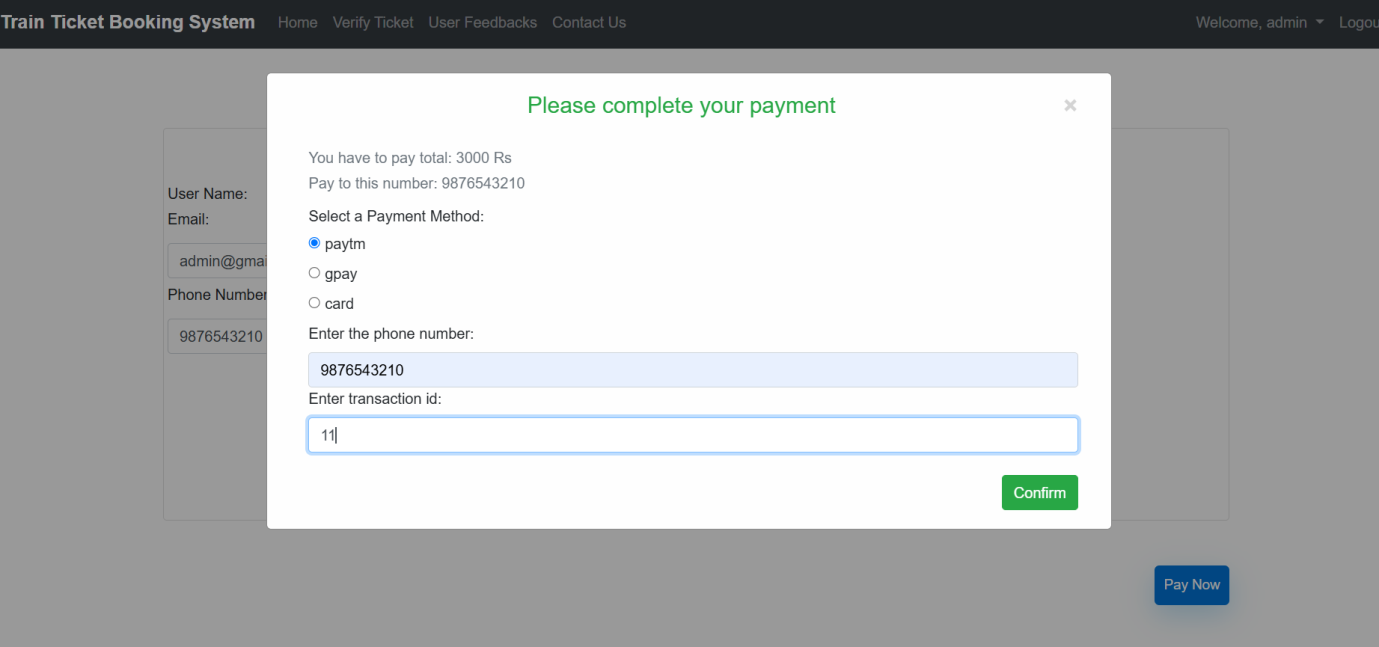
****

**Confirm Bookings:**

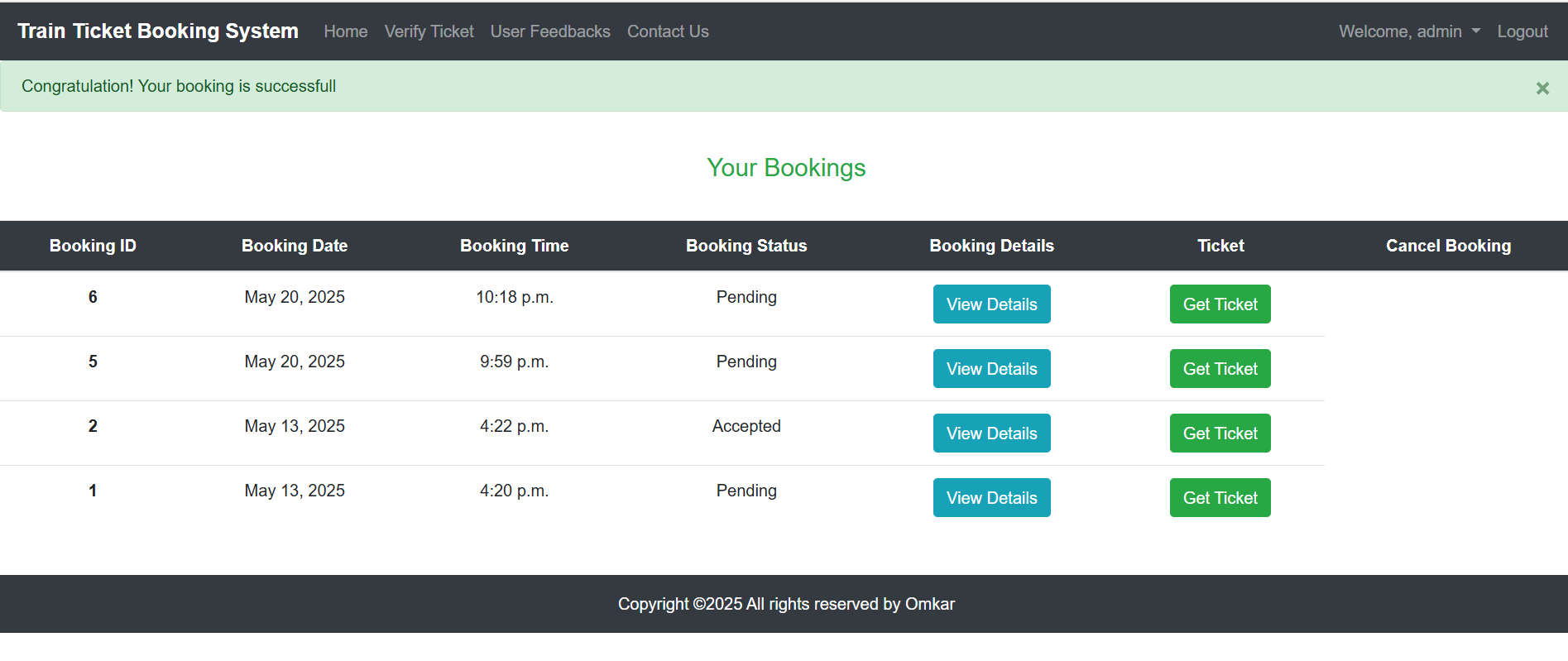


**Booking Details :**

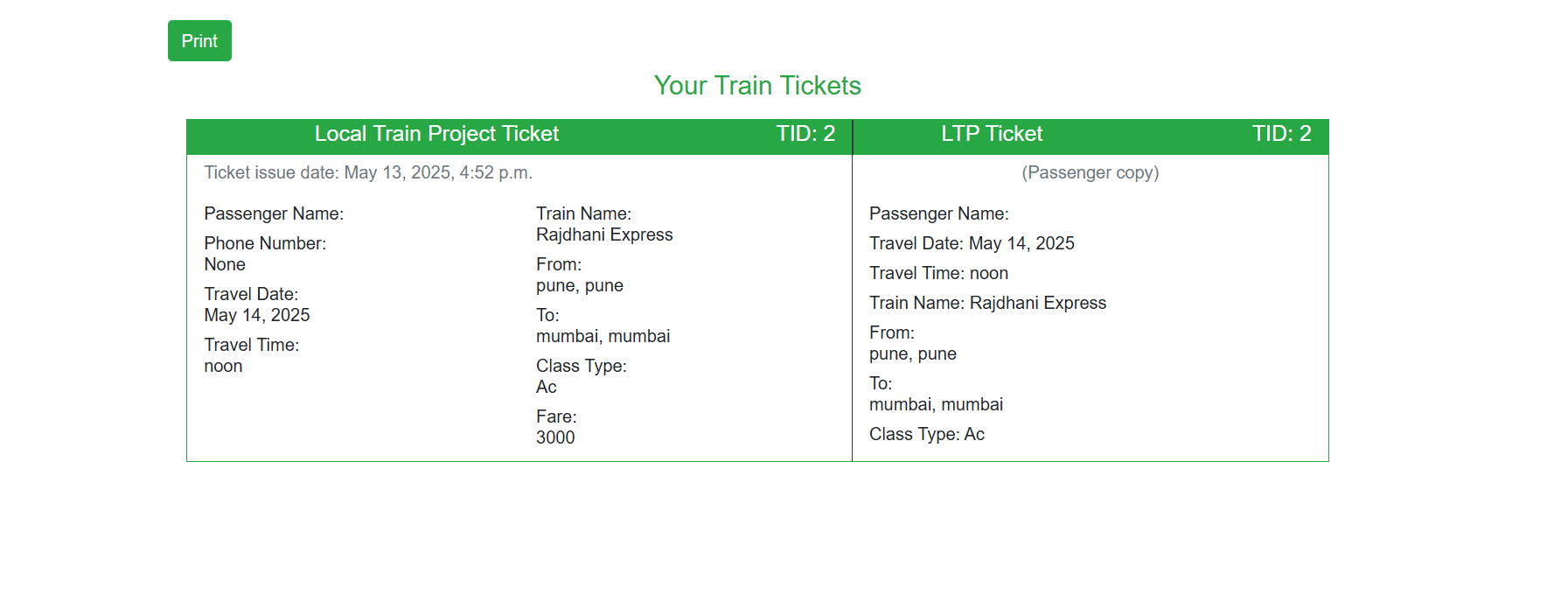
****

**Payments :**

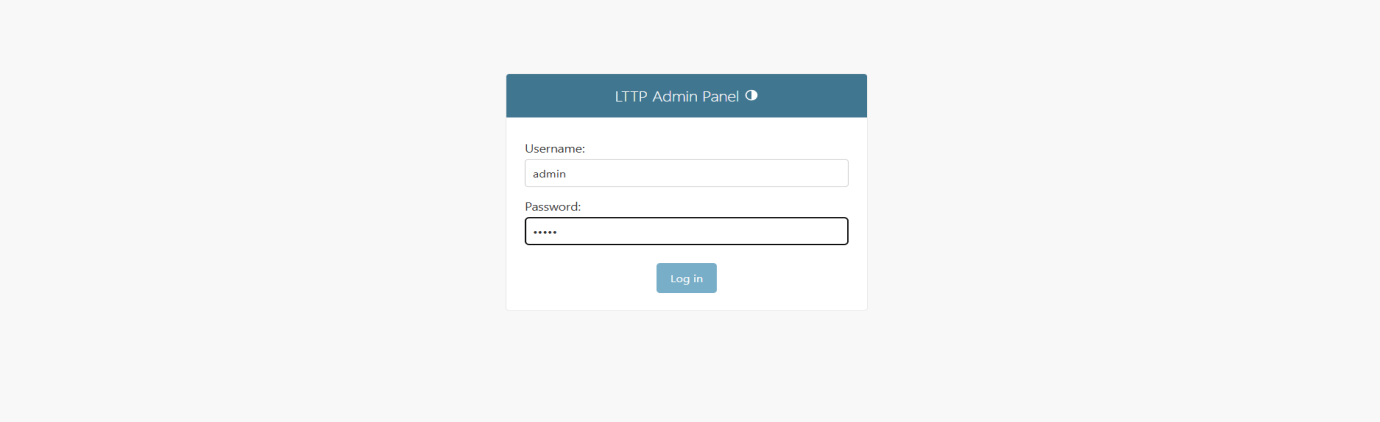
**Your Bookings :**

****

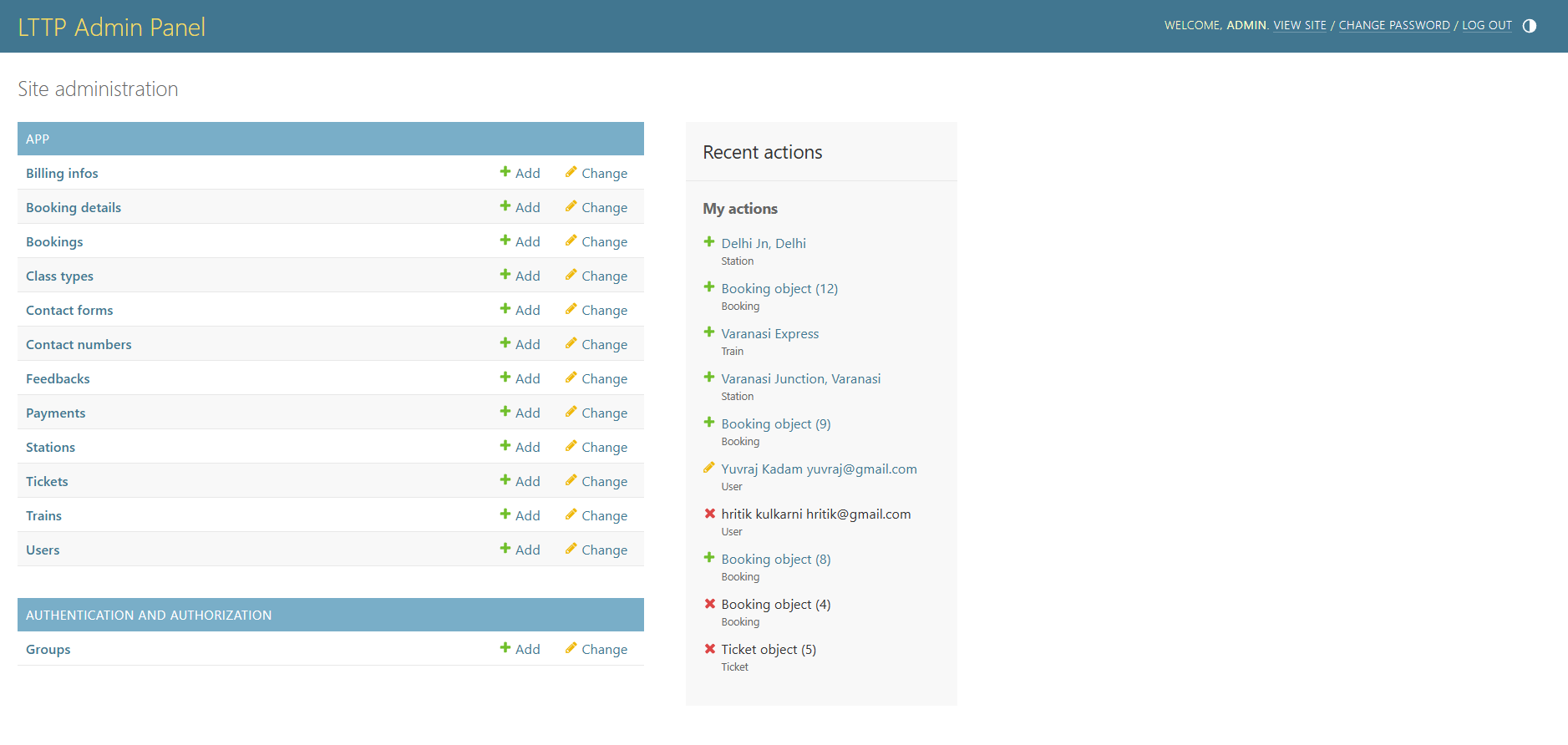
**Tickets :**



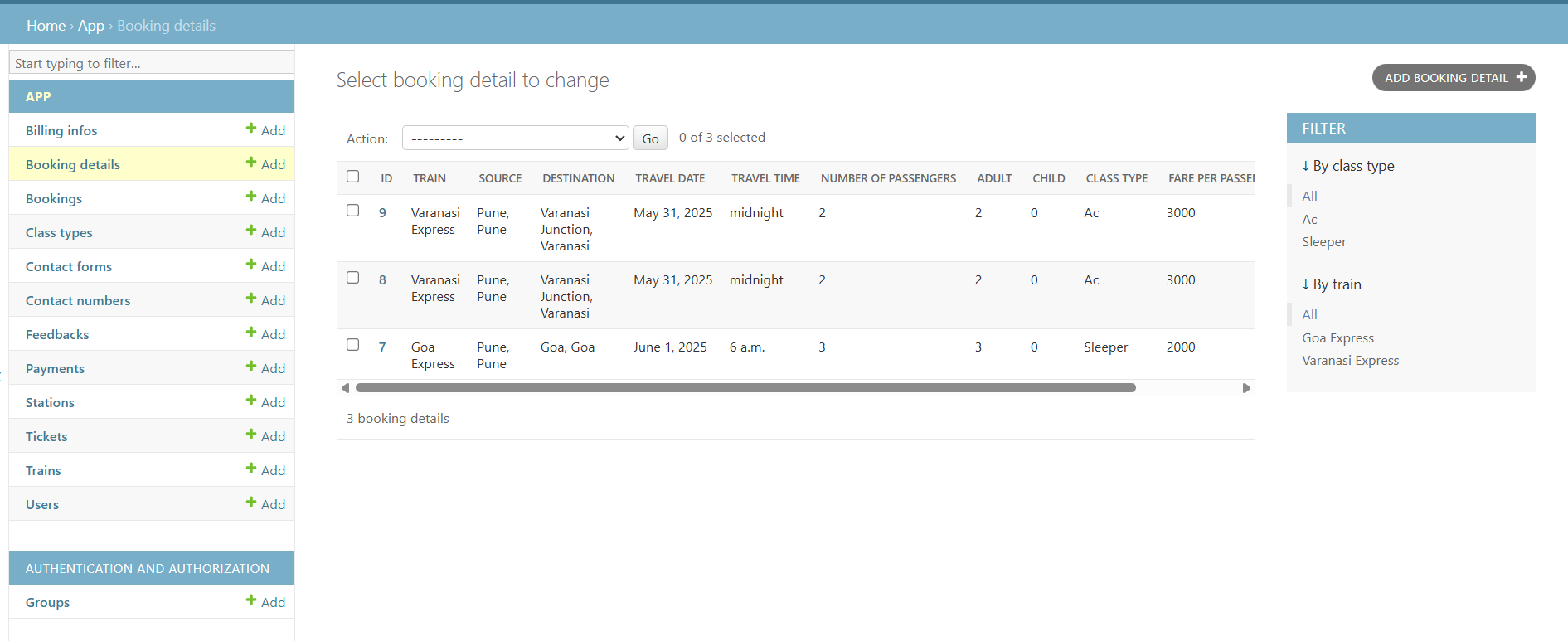
**Admin Login :**

****

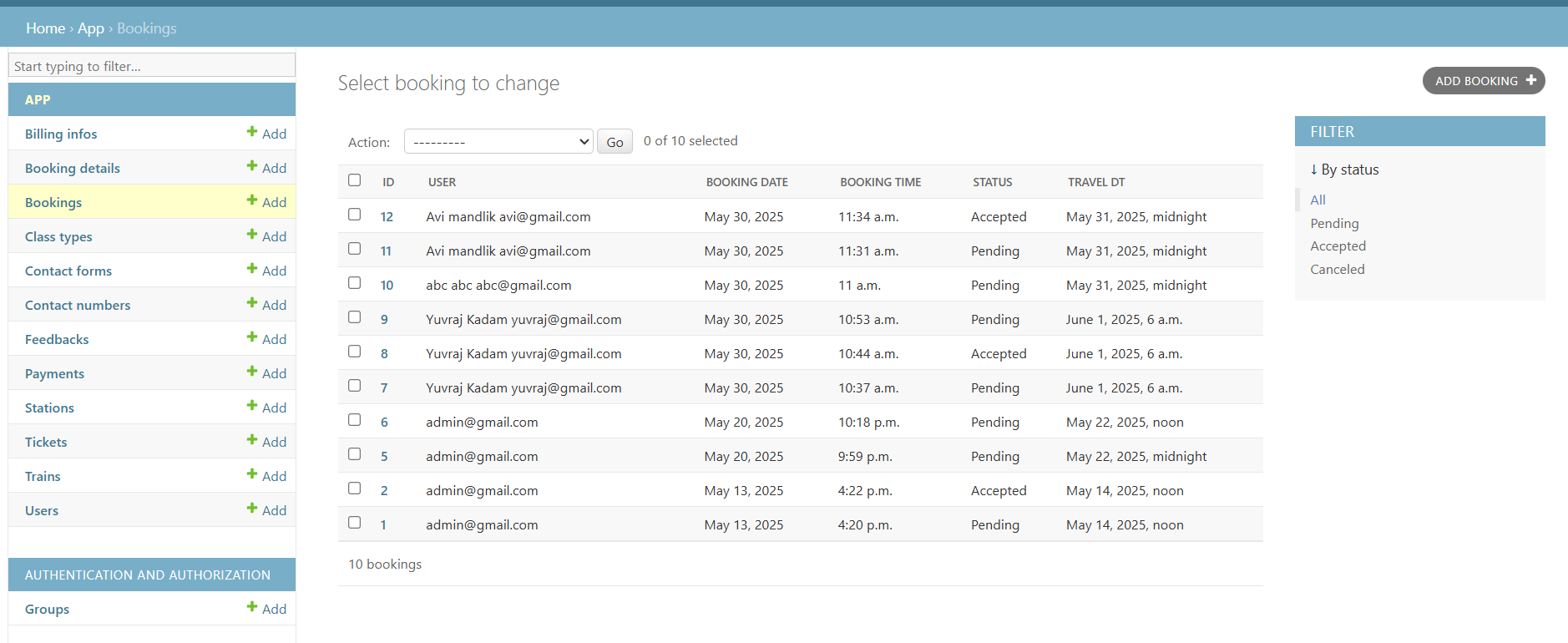
**Admin Panel :**

****

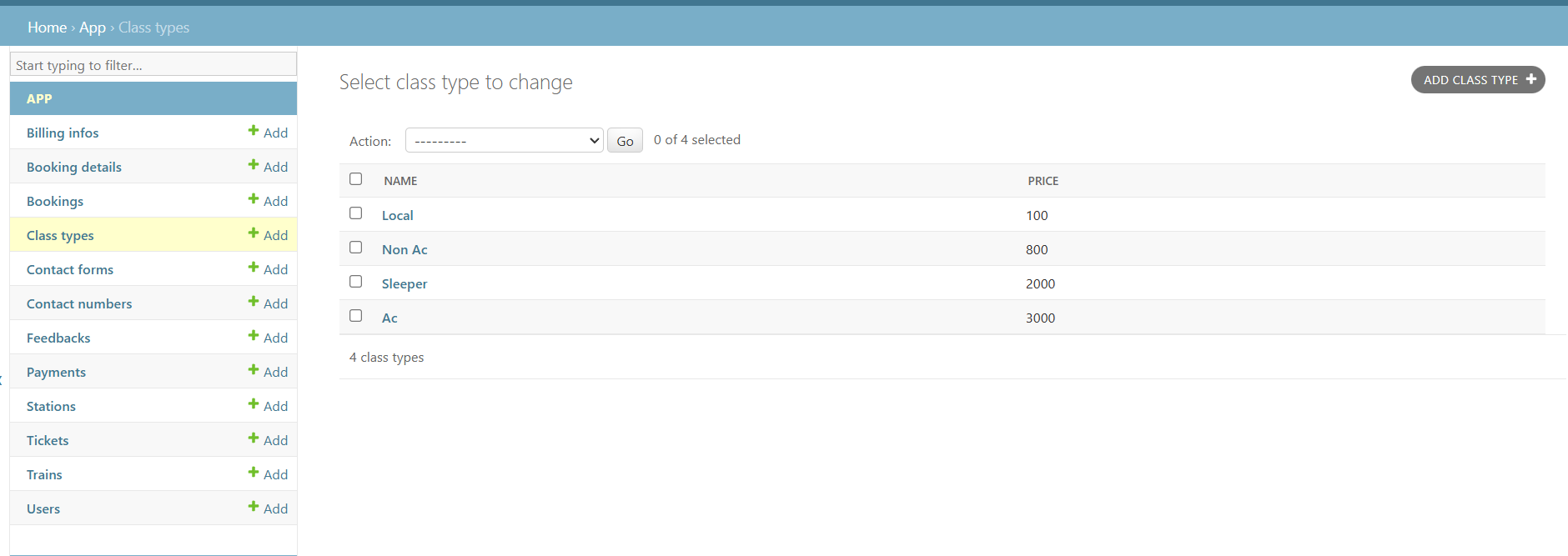
**Booking Details :**

****

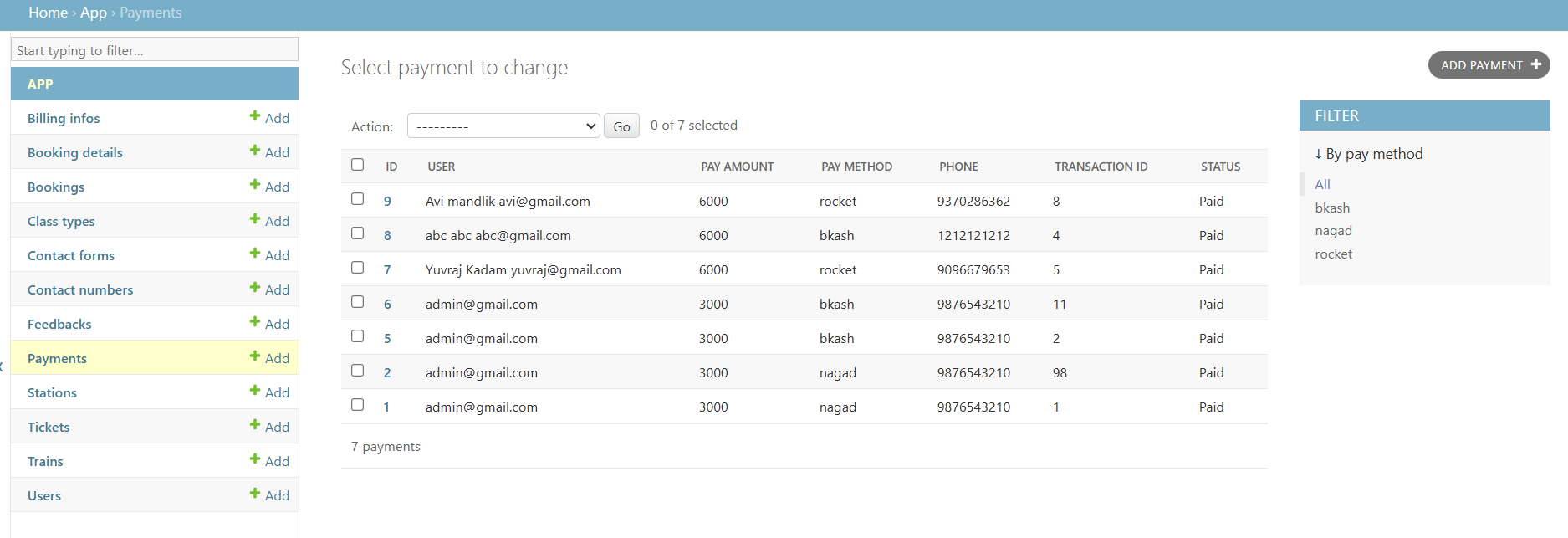
**Bookings :**



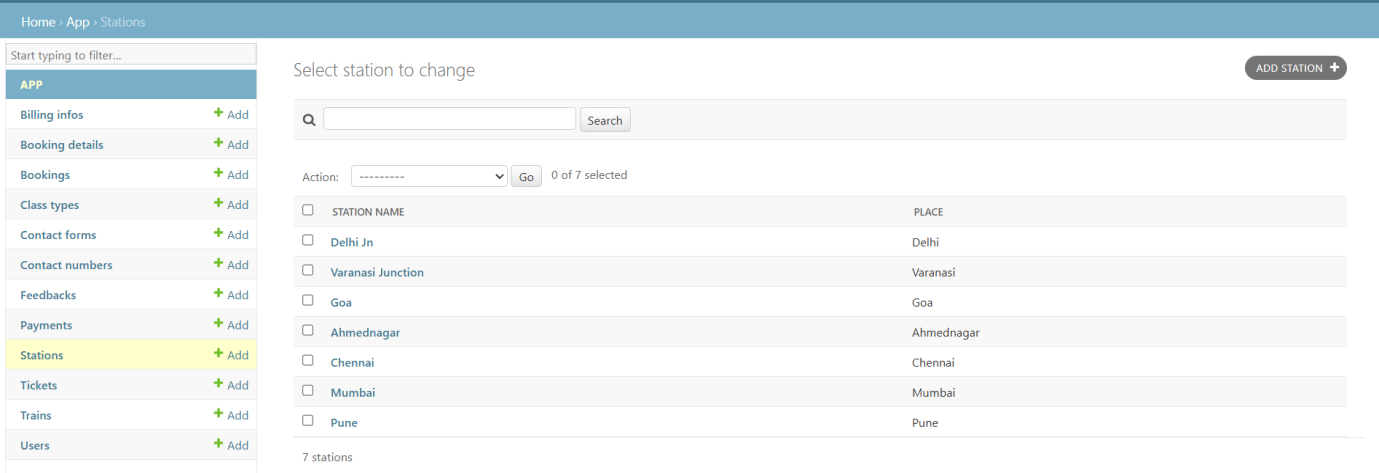
**Class Type :**



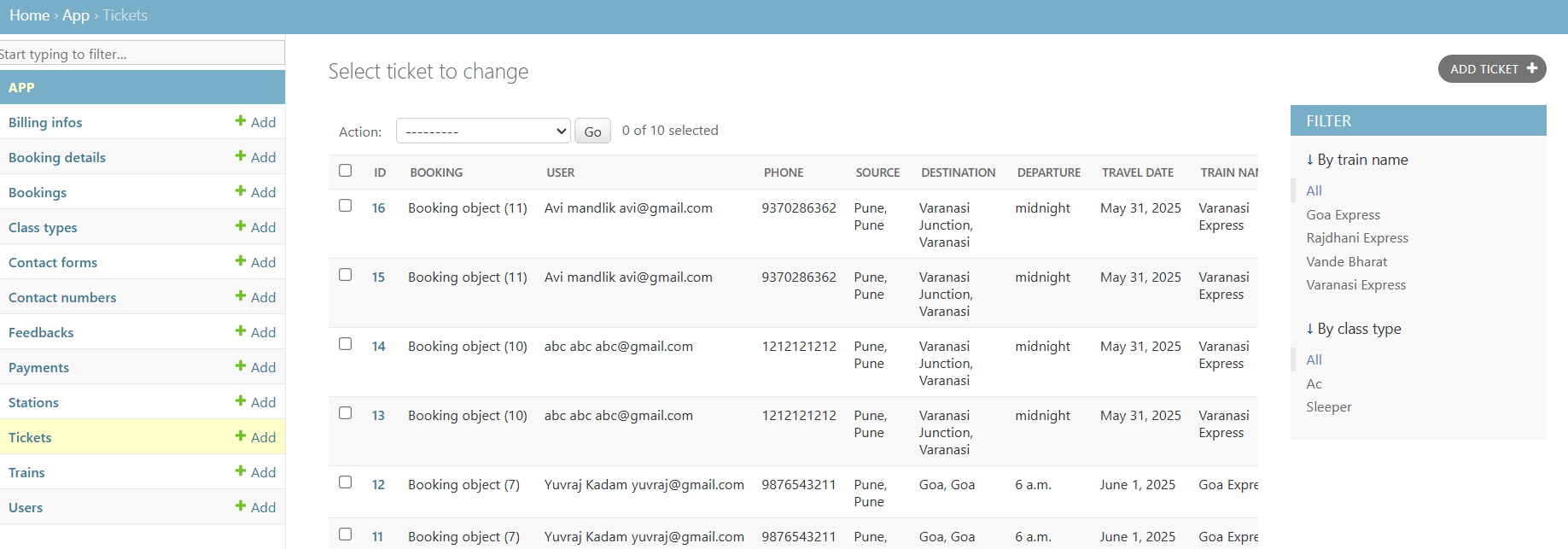
**Payments Information :**



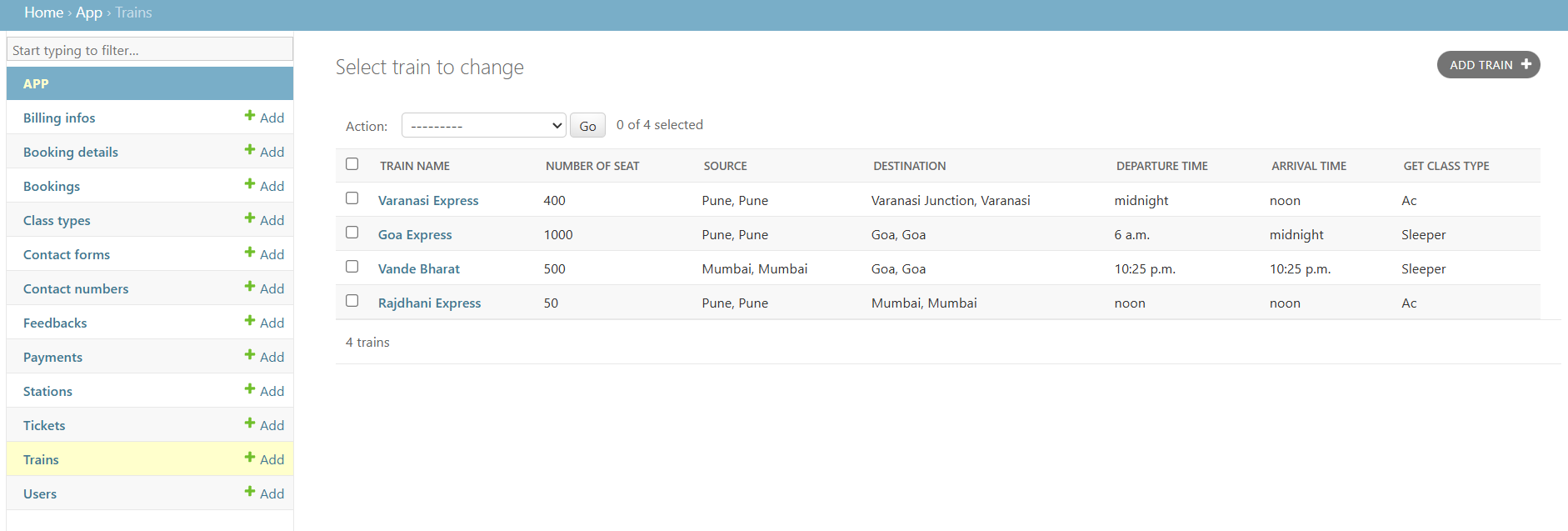
**Stations :**

****

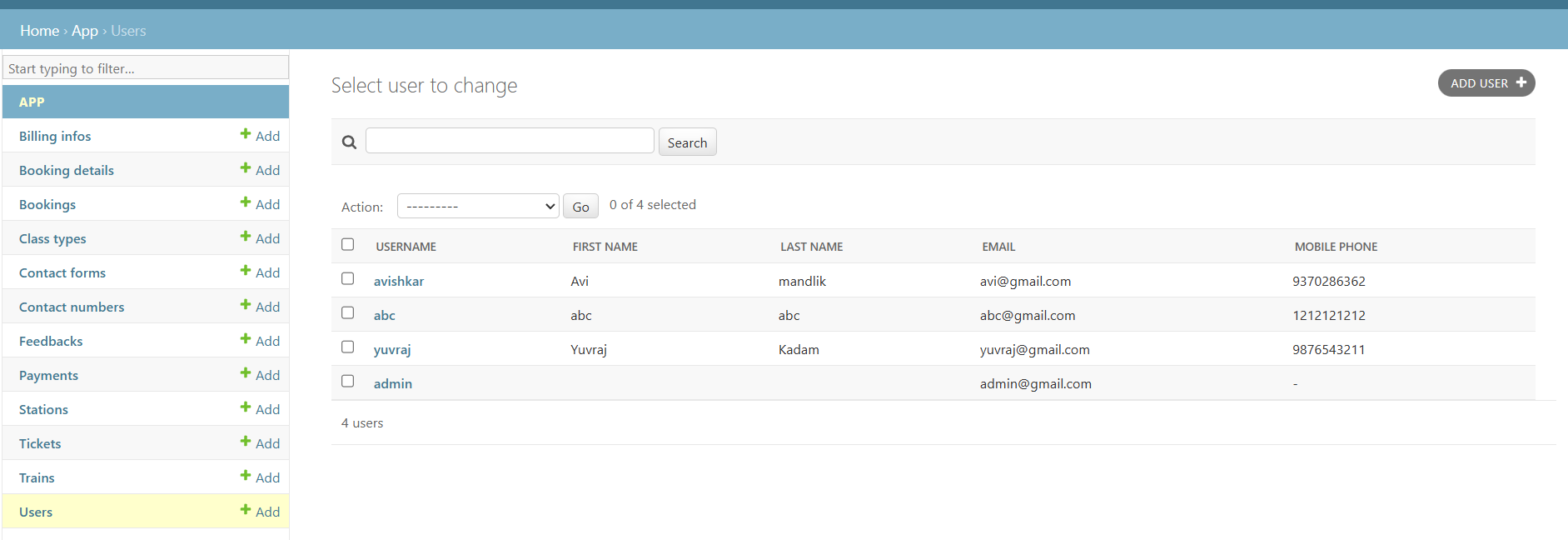
**Tickets :**



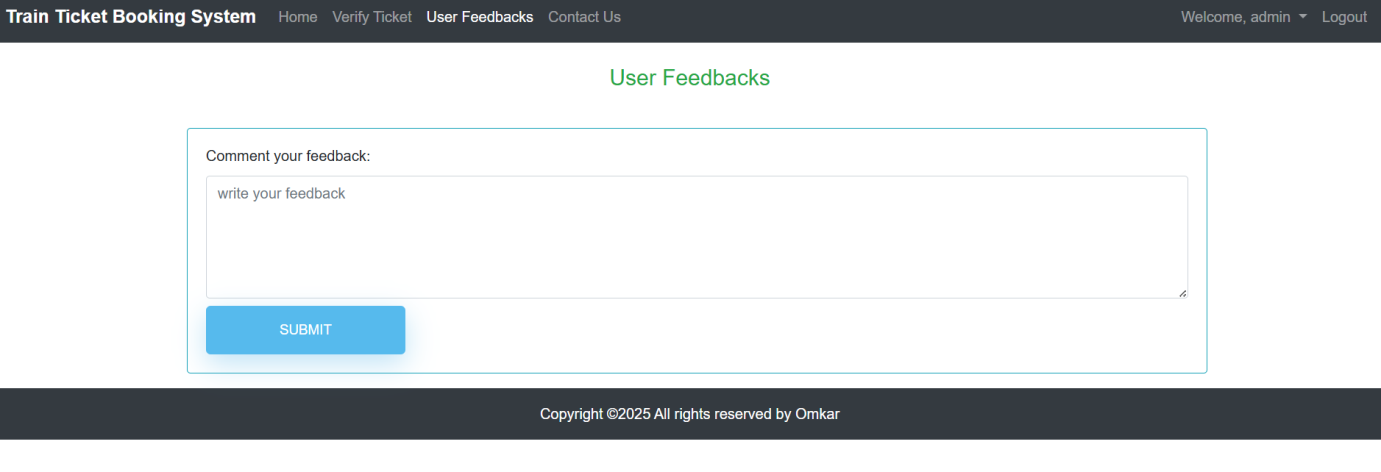
**Trains :**

****

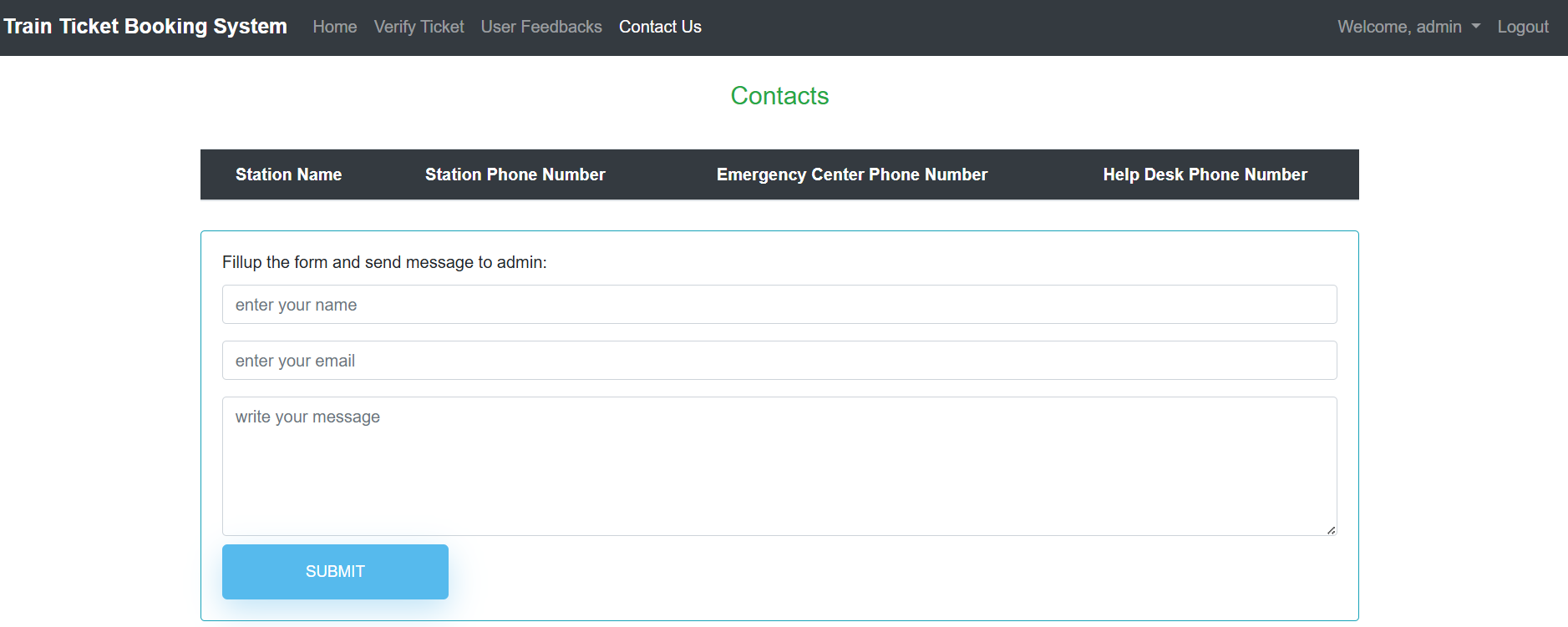
**Users :**

****

**User Feedback :**

****

**Contact Us :**

****

## **CONCLUSION**

The **Train Ticket Booking System** is a reliable and efficient solution aimed at simplifying the railway reservation process through modern web technologies. Built using **Spring Boot** for the backend and **Angular 17+** for the frontend, the platform offers a responsive and user-friendly interface that supports key features such as **user registration, login, train listing, ticket booking, cancellation, and admin management**.

The system ensures **role-based access** for users and admins, promoting secure and controlled interaction. It addresses real-world needs by making the ticketing process digital, accessible, and less error-prone, especially in regions where manual booking systems are still prevalent.

Technically, the project follows **MVC architecture**, uses **RESTful APIs**, and was rigorously tested using **JUnit, Postman, and Angular testing tools** to ensure functionality, performance, and security. **Spring Security** helps protect user data, while frontend validations improve the overall user experience.

In summary, this project demonstrates a strong application of **full-stack development principles** and showcases practical problem-solving in the transport domain. With future upgrades such as **online payment integration, SMS/email notifications, live train tracking**, and **AI-based seat suggestions**, the system holds great potential for real-world deployment and scalability. The Train Ticket Booking System not only streamlines the booking process but also promotes **digital transformation and accessibility** in public transportation.

## **BIBLIOGRAPHY**

1. **Chaudhari, S., & Patil, R.** (2021). Online Train Reservation System. International Journal of Engineering Research & Technology (IJERT), Vol. 10, Issue 04. Retrieved from https://www.ijert.org/research/online-train-reservation-system-IJERTV10IS040189.pdf
2. **International Journal of Computer Applications.** (2018). Design and Implementation of a Train Ticket Booking System Using Web Technology. DOI: https://doi.org/10.5120/ijca2018916890
3. **Open Source Project**. (2022). Spring Boot Train Ticket Booking System [GitHub Repository]. Retrieved from <https://github.com/developer1105/spring-boot-train-booking>
4. **Kumar, A., & Singh, N.** (2020). Digital Train Reservation Portal Using Full Stack Development. International Research Journal of Engineering and Technology (IRJET), Vol. 7, Issue 6. Retrieved from https://www.irjet.net/archives/V7/i6/IRJET-V7I6431.pdf
5. **Nayak, P.** (2023). Train Ticket Reservation System Using Angular and Spring Boot [GitHub Repository]. Retrieved from <https://github.com/pnayak-dev/train-reservation-system>

## **REFERENCES**

1. **GeeksforGeeks.** (2023). Spring Boot CRUD Example with JPA. Retrieved from https://www.geeksforgeeks.org/spring-boot-crud-operations-with-mysql/
2. **TutorialsPoint.** (2024). Angular Components and Services Overview. Retrieved from <https://www.tutorialspoint.com/angular8/>
3. **Medium.com.** (2023). Building Full Stack Applications with Angular and Spring Boot. Retrieved from <https://medium.com/>
4. **Spring Framework Documentation.** (2024). Spring Boot Reference Documentation. Retrieved from <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>
5. **Angular Documentation.** (2024). Angular – Developer Guide. Retrieved from <https://angular.io/docs>
6. **MySQL Documentation Team.** (2024). MySQL 8.0 Reference Manual. Oracle Corporation. Retrieved from <https://dev.mysql.com/doc/>
7. **Postman Team.** (2024). Postman Learning Center – Getting Started. Retrieved from https://learning.postman.com/docs/getting-started/introduction/
8. **Git Documentation.** (2024). Git Basics. Retrieved from <https://git-scm.com/docs/gittutorial>
9. **Tailwind CSS Documentation.** (2024). Tailwind CSS – A Utility-First CSS Framework. Retrieved from <https://tailwindcss.com/docs>
10. **Font Awesome.** (2024). Font Awesome – Icon Toolkit Documentation. Retrieved from <https://fontawesome.com/docs>