CENTER FOR DEVELOPMENT

OF ADVANCED COMPUTING,

CHENNAI

Documentation On

"ONLINE RAILWAY RESERVATION"

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF DEGREE OF

POST GRADUATE DIPLOMA IN ADVANCED COMPUTING

PG-DAC MAR 2023

SUBMITTED BY

Kalpesh Amrutkar PRN NO:- 230360820019

Siddhant Dusane PRN NO:- 230360820053

UNDER THE GUIDANCE OF

Mr. Akash Sir

Dr. Sumithra Mam
Centre Coordinator

Mr. Akash Sir
Project Guide

ACKNOWLEDGEMENT

This is to acknowledge our indebtedness to our Project Guide, **Mr. Akash Sir** C-DAC ACTS, Chennai for her constant guidance and helpful suggestion for preparing this project Online Railway Reservation System. We express our deep gratitude towards her for her inspiration, personal involvement, and constructive criticism that he provided us along with technical guidance during the course of this project.

We take this opportunity to thank the Head of the department **Dr. Sumithra Mam** for providing us with such a great infrastructure and environment for our overall development.

It is our great pleasure in expressing sincere and deep gratitude toward **Mr. Akash Sir** (Course Coordinator, PG-DAC) for their valuable guidance and constant support throughout this work and helps to pursue additional studies. Also, our warm thanks to C-DAC ACTS, Chennai which provides us with this opportunity to carry out this prestigious Project and to enhance our learning in various technical fields.

Your's sincerely,

Mr. Kalpesh D. Amrutkar

Mr. Siddhant V. Dusane

Index

Chapter No.1

Introduction

In this emerging world of computers, almost all-manual system has switched to automated and computerized system. Therefore, we are developing the software for "Railway Reservation System" to model the present system and to remove the drawbacks of the present system. This project explores how computer technology can be used to solve the problem of user.

This being a big step in terms of improvement in the railway system it is widely accepted across the country. Rather than designing manually, we have made use of computer. Use of computer has solved many problems, which are faced during manual calculation. Once data are fed, it can perform accurate functions. Therefore, to reduce the complexity and efficiency a versatile and an outsourcing railway reservation system has been developed. This project introduces railway reservation system. It explains how reservation is being done in Indian Railways. The systematic procedure is explained. This project is developed in java language. All most all the header files have been used in this project. Proper comments have been given at desired locations to make the project user friendly. Various functions and structures are used to make a complete use of this language.

The customers are required to register on the server for getting access to the database and query result retrieval. Upon registration, each user has an account that is essentially the 'view level' for the customer. The account contains comprehensive information of the user entered during registration and permits the customer to get access to his/her past reservations, enquire about travel fare and availability of seats, make fresh reservations, and update his account details. Each passenger is allotted a unique ticket no. through which one can access his/her account.

The railway administrator is another member involved in the transactions. The administrator is required to login using a master password, once authenticated as an administrator, one has access and right of modification to all the information stored in the database. This includes the account information of the customers, attributes and statistics of stations, description of the train stoppages and physical description of coaches, all the reservations that have been made. The railway administrator has the right to modify any information stored at the server database.

This project is dedicated to model the existing railway reservation system that aims at development of Railway Reservation System that facilitates the railway customer to manage

their reservations and the railway administrator to modify the backend database in a user-friendly manner. The customer and the railway administrator are two parties that interact with the database, who have different 'view level schemas' to the database information. The software provides a comprehensive set of features to enhance the operational limits.

Now one can easily plan the journey comfortably as the process is efficient and fast with being easy to access. The efficiency of the railway will increase result of computerization.

1.1 Scope

The purpose of this source is to describe the railway reservation system which provides the train timing details, reservation, billing and cancellation on various types of reservation namely,

- Confirm Reservation for confirm Seat.
- Reservation against Cancellation.
- Online Reservation.
- Freight Revenue enhancement
- Passenger Revenue enhancement
- Improved & optimized service

1.2 Overview

Our website has various kinds of information that helps regarding booking of tickets via railways. Users will be able to search the train availability, the exact fare, the arrival and departure time of the train and they can also book the ticket by using the card and after booking the ticket if the user want to cancel it then they can easily do it also.

1.3 Objective

Our project introduces railway reservation system with an objective to make the reservation system more efficient, easier and fast. This project explores how computer technology can be used to solve the problem of user.

The main objectives provided by this software are as follows:

- We can enquire about availability of trains
- We can reserve and cancel their seats
- We can modify the information related to

a) Trains

- Train Schedule
- Train Name

b) Ticket Fare

This project is dedicated to model existing railway reservation systems that aim at development of Railway Reservation System that facilitates the railway customer to manage their reservations and the railway administrator to modify the backend database in a user-friendly manner.

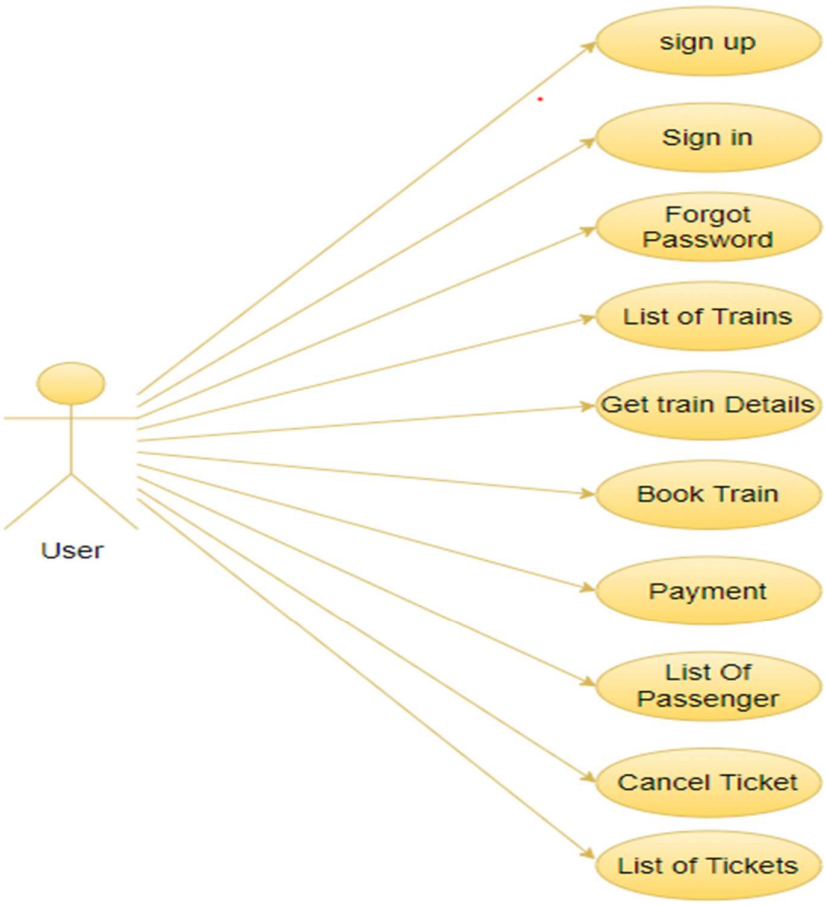
1.4 Purpose of project

The Indian Railways (IR) carries about 5.5 lakhs passengers in reserved accommodation every day. The Computerized Passenger Reservation System (PRS) facilitates the booking and cancellation of tickets from any of the 4000 terminals (i.e. PRS booking window all over the countries). These tickets can be booked or cancelled for journeys commencing in any part of India and ending in any other part, with travel time as long as 72hours and distance up to several thousand kilometres. In the given project we will be developing a website which will help users to find train details, book and cancel tickets and the exact rates of their tickets to the desired destination. With the help of online booking people can book their tickets online through internet, sitting in their home by a single click of mouse. Using their credit cards people can easily get their tickets done within minutes.

Chapter No.2

System Analysis

2.1 Functional Requirements



2.1.1 User Account

The passenger, who will henceforth be called the 'user', will be presented with 3 choices by the reservation system, as the first step in the interaction between them. A user can choose one of these and his choice would be governed by whether he is a guest or a registered user and whether he wants to check the availability of tickets or also buy them. The terms 'registered user' and 'guest' are described below.

A user who has traveled by the railway earlier would have been given a user id and a password. This 'personal information' would be henceforth referred to as 'profile'. Such a user with a profile in DB-user shall be called a 'registered user'. A registered user will be able to check the availability of tickets as well as buy a ticket by logging into the system.

A new user, on the other hand, would either have to

a) register himself with the system by providing personal information or

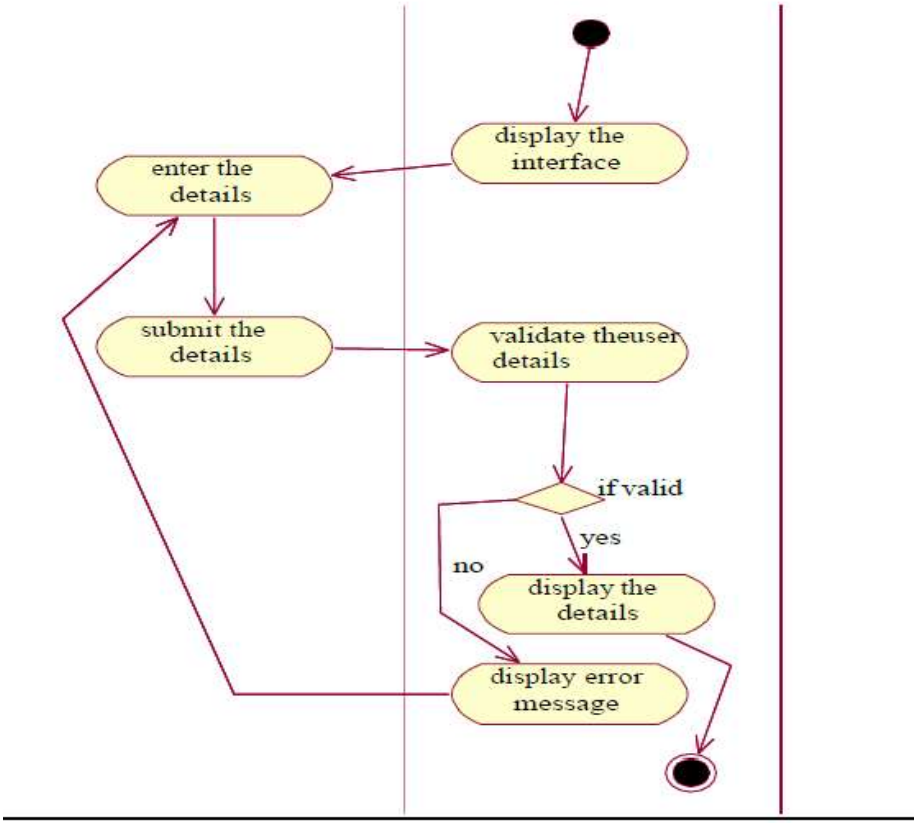
b) log into the system as a user.

In case of 'a', the new user becomes a registered user.

A registered user can also act as a guest if he only wants to check the availability of tickets. 'Availability of tickets' always refers to viewing the train schedule for given days, the price of tickets. The system shall present the user with an option to exit from the system at any time during the following processes.

2.1.2 Registration and creation of user profile

The system shall require a user to register, in order to carry out any transactions. It will ask the user for the following information at the least – a user id, a password, first name, last name, address, phone number, email address, sex, age. The system will automatically create a 'role' field and initialize it to 'user' in the user's profile.



2.1.3 Search Train

Here we provided Search facility for any user to search train schedule with login into account. This will provide user an option for searching train and book for their journey. After logging in a user, the system shall request him to enter the following details – origin city and destination city. "City' is a generic term and refers to a city or town as the case may be.

After the origin and destination cities are ascertained, the system shall now access the train schedule database, referred to as 'train_schedule', and checks if there is a direct operational service between the two cities.

The system shall now ask the user to enter the following details - class departure date and add passengers. 'Class' refers to AC/NON-AC class. This choice shall be made by the user through a dropdown menu indicating all the possible combinations of choices. 'Departure date' refers to a single date, entered through text box.

Having taken all the above input from the user, the system checks for any false entries like the departure date & all. In case of incompatibility, the system will not display any train available.

The system queries the flights database 'train_schedule' to check which of the train on the schedule have seats available. The system displays the results in a suitable form. (a tabular form) with the following information depicted – for each trainId, departure time in origin city, arrival time in destination city, departure city, arrival city
,Ticket price and the number of seats available on that train.

There can be several trains of different types between two cities and from the Origin City. In case, the user has entered a range of dates, the system shall display all the trains for all those dates in the range. There will be a Book button in front of every row displayed n the table of flights searched.

The system will then ask for personal information of all passengers i.e. one registered user can book for multiple users. So all users will be added in the table.

The system shall now display the price of the ticket for the trip. This will be the sum of the prices for all the members of the travel party being represented by the user.

2.1.4 Making Reservations

After having taken the user through the step 2.2, Checking Availability, The system will now ask the user if he wishes to block/buy the ticket. If yes, and

a) if the user has been a guest, he will have to first register and become a registered user and then log onto the system.
b) If the user is already a registered user, and if he has logged on already, he can buy the ticket, but if he has been acting as a guest, he will have to log on.

Having ensured that the user is logged on validly according to 3.4.1, the system compares the departure date with the system date. If the departure date falls within 2 weeks of the system date, the system informs the user that he has no option to block the ticket and asks him if he would like to buy it.

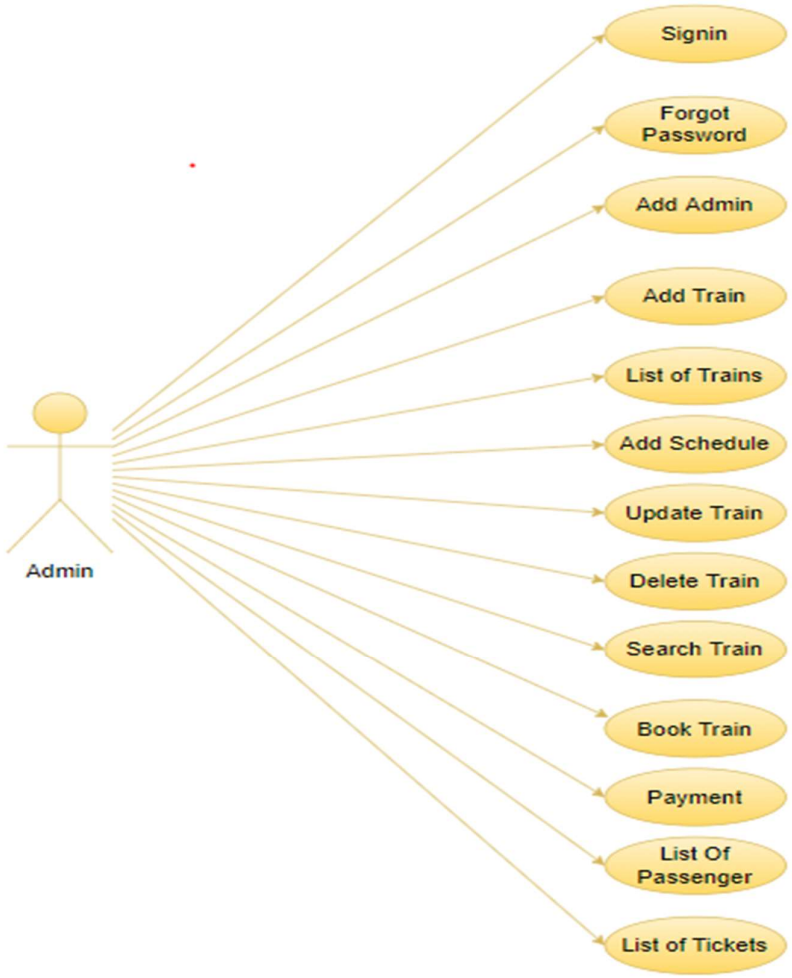
If the difference between the departure date and system date is more than 2 weeks, the system asks the user if he would like to buy the ticket. The system informs the user that he can block the ticket at no cost now. It also informs him that if he chooses to block the ticket, he should make a final decision before 2 weeks of the departure date. The system shall send an email to the user.

Having taken the input from the user in 3.4.2, the system shall now proceed to update the reservation database DB-reservation. It will decrement the number of available seats on the particular train for the particular class by the number of travelers being represented by the user.

In case the user buys the ticket, the system asks for entering his or her bank card information and then charges the price of the ticket to his debit card number.

2.1.5 View Booking History

The system shall allow a user to view all information about his previous bookings. It accesses Passenger table and retrieves the details of the trip and presents them to the user in a tabular format.



Admin should be able to login, add train information, add flight information, Delete train and see train details.

2.2 NON-FUNCTIONAL REQUIREMENTS

- Interface

Go to Appendix B for user interfaces

- Performance

Number of Concurrent Users:

ARS shall be able to handle at least 1000 transactions/inquiries per second

Booking of Tickets:

The system is susceptible to any temporary server failure since it uses the strong feature of Struts 2 and Hibernate. Hence the examination will be continued even if the sever gets disconnected in between the examination.

- Constraint

ARS shall be able to handle at least 1000 transactions/inquiries persecond

- **Other Requirements:**
  - **Hardware Interfaces**

    The SPMS is expected to function on Intel PIII 900 MHz Processor equivalent or above, 128 MB RAM, 20 GB HDD.
  - **Software Interfaces**

The SPMS shall work on MS Windows operating systems family (MS Windows 98, MS Windows NT Workstation, MS Windows 2000, MS Windows XP). It configures to work with Oracle database. This System works on Apache Tomcat server. It uses browser IE 5.0 & above. It uses IIS 5.0 server.

Chapter No.3

System Design

3.1 Database Design

The following table structures depict the database design.

Table1:- User

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
first_name	varchar(100)	YES		NULL	
last_name	varchar(100)	YES		NULL	
age	int	YES		NULL	
gender	varchar(10)	YES		NULL	
email	varchar(100)	YES		NULL	
password	varchar(100)	YES		NULL	
mobile	varchar(20)	YES		NULL	
role	varchar(100)	YES		NULL	
address	varchar(100)	YES		NULL	

Table2:- Train

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
train_name	varchar(100)	YES		NULL	
start_city	varchar(100)	YES		NULL	
dest_city	varchar(100)	YES		NULL	
departure_time	time	YES		NULL	
reach_time	time	YES		NULL	
ac_seating_seat_count	int	YES		NULL	
ac_sleeper_seat_count	int	YES		NULL	
non_ac_seating_seat_count	int	YES		NULL	
non_ac_sleeper_seat_count	int	YES		NULL	
ac_seating_seat_price	int	YES		NULL	
ac_sleeper_seat_price	int	YES		NULL	
non_ac_seating_seat_price	int	YES		NULL	
non_ac_sleeper_seat_price	int	YES		NULL	
total_seat_count	int	YES		NULL	

Table3:- Passenger

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
user_id	int	YES		NULL	
train_id	int	YES		NULL	
ticket_id	int	YES		NULL	
first_name	varchar(100)	YES		NULL	
last_name	varchar(100)	YES		NULL	
age	int	YES		NULL	
gender	varchar(100)	YES		NULL	
seat_class_name	varchar(100)	YES		NULL	
inner_type	varchar(100)	YES		NULL	
date_of_travelling	date	YES		NULL	

Table4:-Train_Schedule

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
train_id	int	YES		NULL	
date_of_travelling	date	YES		NULL	
seat_class_name	varchar(100)	YES		NULL	
seating_seat_count	int	YES		NULL	
sleeper_seat_count	int	YES		NULL	
seating_seat_price	int	YES		NULL	
sleeper_seat_price	int	YES		NULL	
total_seat_count	int	YES		NULL	

Table5:-Ticket

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
no_of_passanger	int	YES		NULL	
start_city	varchar(100)	YES		NULL	
dest_city	varchar(100)	YES		NULL	
departure_time	time	YES		NULL	
reach_time	time	YES		NULL	
ticket_amount	int	YES		NULL	
status	varchar(100)	YES		NULL	

Chapter No.4

Technology Used

Following technology used in this project :

- **Front End :** React Js
- **Back End :** Java Spring Boot API and MySQL

React Js :-

React can be used with a combination of several JS libraries and frameworks like jQuery, Backbone, or even Angular in MVC. The core functionality of React is developed in a way that it makes front end development fast, modular, simple, and scalable.

- Component-Based: A single React app consists of several components. This modular, component-based approach in React JS enables it to integrate and display the design schemes.
- React apps generally have two kinds of components – Stateless Functional Components and Stateful Class Components.
- Declarative: React provides a great developer experience to its users, which in turn gives a good UX. The developers are able to build web apps faster and can more easily debug the screen or components when using React.
- JavaScript Syntax Extension (JSX): JSX is used to develop robust user interfaces. Developers can write the HTML structures and JS codes in the same file, which makes understanding and debugging codes an easy feat since using the complex JS DOM structure can be avoided.
- Virtual DOM: React has provisions of a Virtual DOM (VDOM) which enables easy manipulation. In the event when the state of an object is changed, the VDOM changes that individual object in the real DOM and not all the other existing objects.
- Faster Performance: The VDOM enables the React JS-based web apps to run faster. Plus, react as a platform can be quickly download configuration process.

Java Spring Boot API :-

- Lightweight: Spring is modular lightweight framework which allows you to selectively use any of its modules on the top of Spring Core.
- Inversion of Control (IOC): This is another top feature of Spring framework where application dependencies are satisfied by the framework itself. Framework creates the object in runtime and satisfies application dependencies.

- Aspect Oriented Programming (AOP): Aspect Oriented Programming (AOP) is very popular in programming world and in Spring it is well implemented. Developer can use Aspect Oriented Programming (AOP feature of Spring to develop application in which business logic is separated from system services.
- Container: Spring provides their own container for managing the bean lifecycle.
- HTTP Support: In Spring MVC, a controller can handle the requests for all HTTP methods, which is a backbone of RESTful web services. You can handle a GET, POST, PUT and DELETE methods to remove resources from the server.
- Bypassing View-Based Rendering: By using this, you can directly send a response to a client, as the resource clients want and also in the format they want.
- REST Annotations: The Spring 4.0 release added a dedicated annotation, @RestController, to make the development of RESTful web services even easier.
- Support to extract Data from URL: @RequestParam to get the value of those query parameters but, not to worry, Spring MVC also provides a @PathVariableannotation, which can extract data from a URL.

MySQL :-

MySQL is a free-to-use, open-source database that facilitates effective management of databases by connecting them to the software. It is a stable, reliable and powerful solution with advanced features like the following:

- Data Security: MySQL is globally renowned for being the most secure and reliable database
- management system. The data security and support for transactional processing that accompany the recent version of MySQL, can greatly benefit any business especially if it is an eCommerce business that involves frequent money transfers.
- On-Demand Scalability: MySQL offers unmatched scalability to facilitate the management of deeply embedded apps using a smaller footprint even in massive warehouses that stack terabytes of data. On-demand flexibility is the star feature of MySQL.
- High Performance: MySQL features a distinct storage-engine framework that facilitates system administrators to configure the MySQL database server for a flawless performance.
- Round-The-Clock Uptime: MySQL comes with the assurance of 24*7 uptime and offers a wide range of high availability solutions like specialized cluster servers and master/slave replication configurations.
- Comprehensive Transactional Support: MySQL tops the list of robust transactional database engines available on the market. With features like complete atomic, consistent,

isolated, durable transaction support, multi-version transaction support, and unrestricted row-level locking, it is the go-to solution for full data integrity.

- Complete Workflow Control: With the average download and installation time being less than 30 minutes, MySQL means usability from day one. Whether your platform is Linux, Microsoft, Macintosh or UNIX, MySQL is a comprehensive solution with self-management features that automate everything from space expansion and configuration to data design and database administration

Chapter No.5

System Testing and Implementation

5.1 Test Case/General Testing

SR-NO	TEST CASE	EXPECTED RESULT	ACTUAL RESULT	ERROR MESSAGE
1	User Login Page	Directed to the user functionalities page (book ticket , list of trains ,search train ,cancel ticket)	OK	"please enter valid email and password"
2	Admin Login Page	Directed too page showing admin functionalities (add train ,edit train , delete train ,add admin)	Ok	"Please enter valid email and password again ".
3	Change password	Users or admin password will be reset	Ok	Nothing
4	Quick search Train	Gives start and destination city	Ok	Nothing
5	Available train and dates	Fill date to be travelled on	Ok	"No details available"
6	Booking Ticket	Passenger details and date to be travelled on	Ok	Nothing
7	Add passenger details for tickets	First Name, Last name, age, gender, seat type, Ac or Non-Ac should be filled	Ok	"Please add atleast one passenger"
8	Go to Payment page	Add payment details like card number, cvv number, card holder name, Date of expiry	Ok	"enter details for respective feilds"
9	Add passenger in passenger table in DB	passenger details inserted into passenger table in DB	Ok	Nothing
10	Add Ticket details to ticket table	Ticket details inserted into ticket table in DB	Ok	Nothing

11	Confirmation mail for ticket booking send to user	Mail in the mailbox of user with details train name , start city , destination city , reach time , boarding time , date of travelling , total fare	Ok	Nothing
12	Add new Train to the train table	New train details inserted into table train in DB.	Ok	“enter valid details for respective fields”
13	view all trains	Display all available trains on UI page	OK	Nothing
14	Schedule train	Display all available trains on UI page and select train to be scheduled and added to train_schedule table in DB	Ok	Nothing
15	Edit train	Display available train details and option to edit train details (train name , departure time , arrival time , seats , seat type)	Ok	“enter valid details for respective fields”
16	Delete train	Display available train details and option to delete train. After clicking on delete button train gets deleted from train table and disappear from UI	Ok	Nothing
17	List of Passenger	Display all the details of passengers from passenger table of DB	Ok	Nothing
18	Add new Admin	Admin details inserted into user table of DB	Ok	“enter valid details for respective fields”

5.2 Naming and Capitalization

Below summarizes the naming recommendations for identifiers in Pascal casing is used mainly (i.e. capitalize first letter of each word) with camel casing (capitalize each word except for the first one) being used in certain circumstances.

Identifier	Case	Examples	Additional Notes
Class	Pascal	Person, BankVault, SMSMessage, Dept	Class names should be based on "objects" or "realthings" and should generally be **nouns**. No '_' signs allowed. Do not use type prefixes like 'C' for class.
Method	Camel	getDetails, updateStore	Methods should use **verbs** or verb phrases.
Parameter	Camel	personName, bankCode	Use descriptive parameter names. Parameter names should be descriptive enough that the nameof the parameter and its type can be used to determine its meaning in most scenarios.
Interface	Pascal with "I" prefix	Disposable	Do not use the '_' sign
Property	Pascal	ForeColor, BackColor	Use a noun or noun phrase to name properties.
Associated private member variable	_camelCase	_foreColor, _backColor	Use underscore camel casing for the privatemember variables
Exception Class	"Exception"	Exception,	

Comments

- Comment each type, each non-public type member, and each region declaration.
- Use end-line comments only on variable declaration lines. End-line comments are comments that follow code on a single line.
- Separate comments from comment delimiters (apostrophe) or // with one space.
- Begin the comment text with an uppercase letter.
- End the comment with a period.
- Explain the code; do not repeat it.

Chapter No.6

Conclusion

- This project aid in automating the existing manual system.
- It can be monitored and guarded remotely.
- It cut down the man power required and provides accurate information. All the information can be saved and can be accessed at any time.
- All the customer, admin can get the required information without delay.
- Being able to Book anytime anywhere.

Chapter No.7

Bibliography

7.1 References

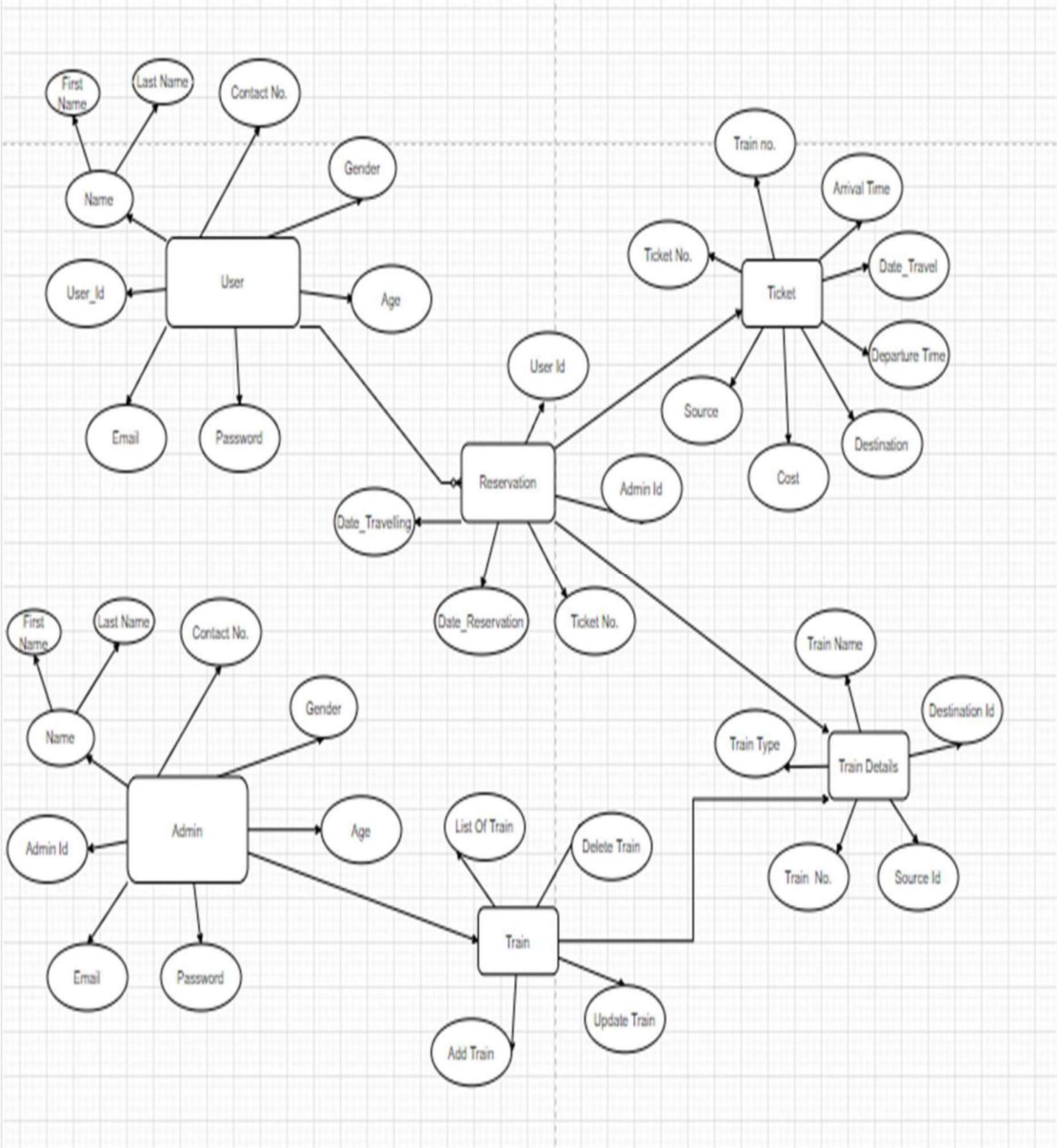
- http://www.google.com
- https://www.irctc.co.in/
- https://docs.oracle.com/en/java/javase/11/docs/api/
- https://docs.spring.io/spring-data/jpa/docs/current/api/
- https://sweetalert2.github.io/
- https://unsplash.com/s/photos/train
- https://reactjs.org/
- https://getbootstrap.com/
- https://www.npmjs.com/package/react-toastify

Chapter No.8

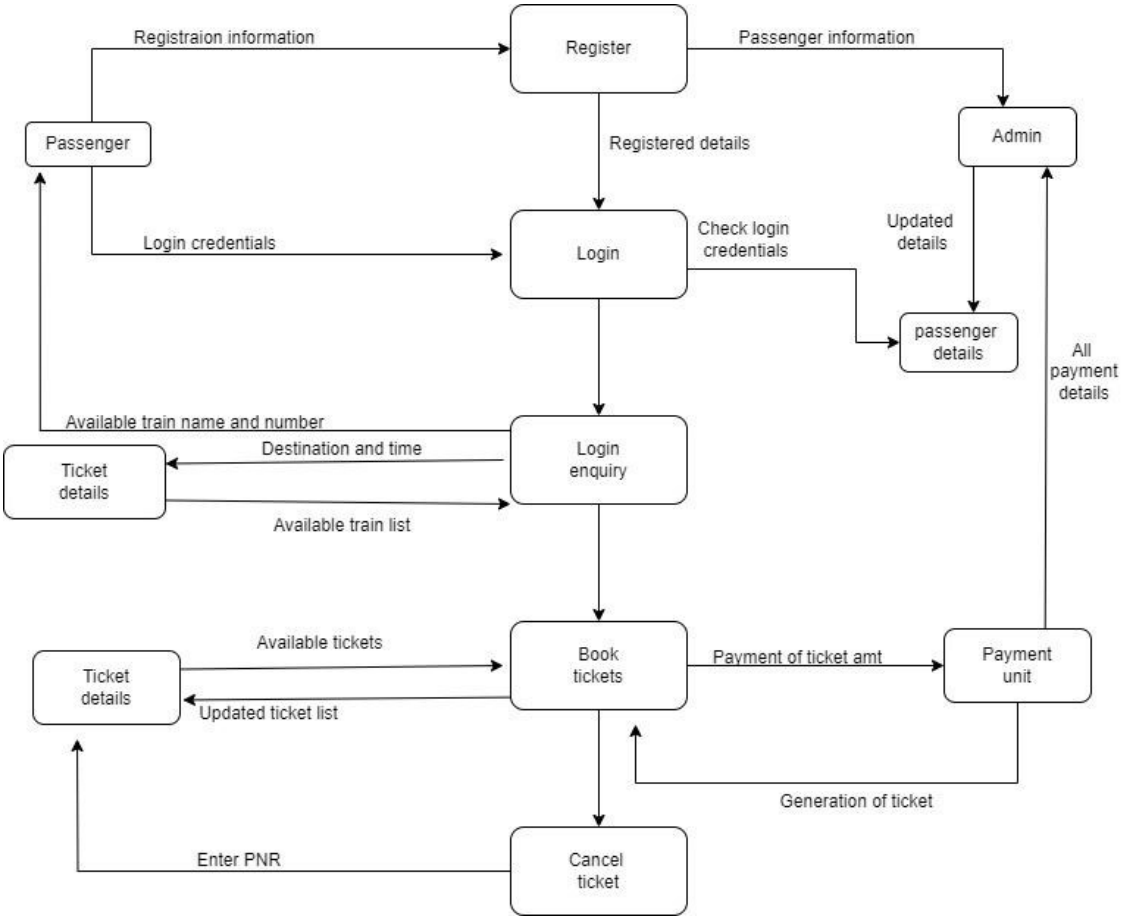
Appendices

Appendix A

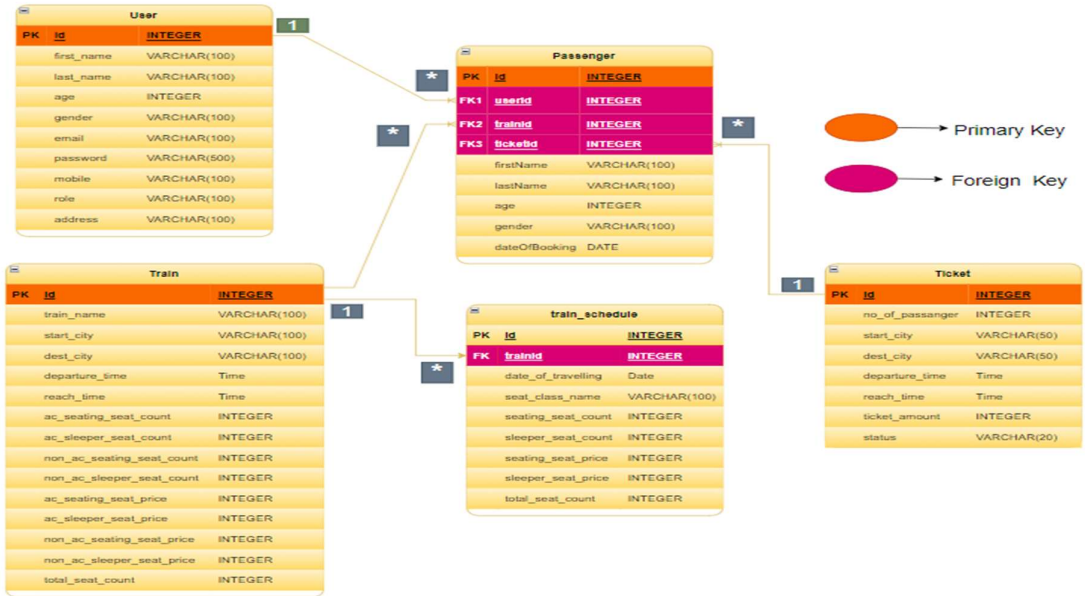
- **E – R Diagram**



- **Data flow Diagram**

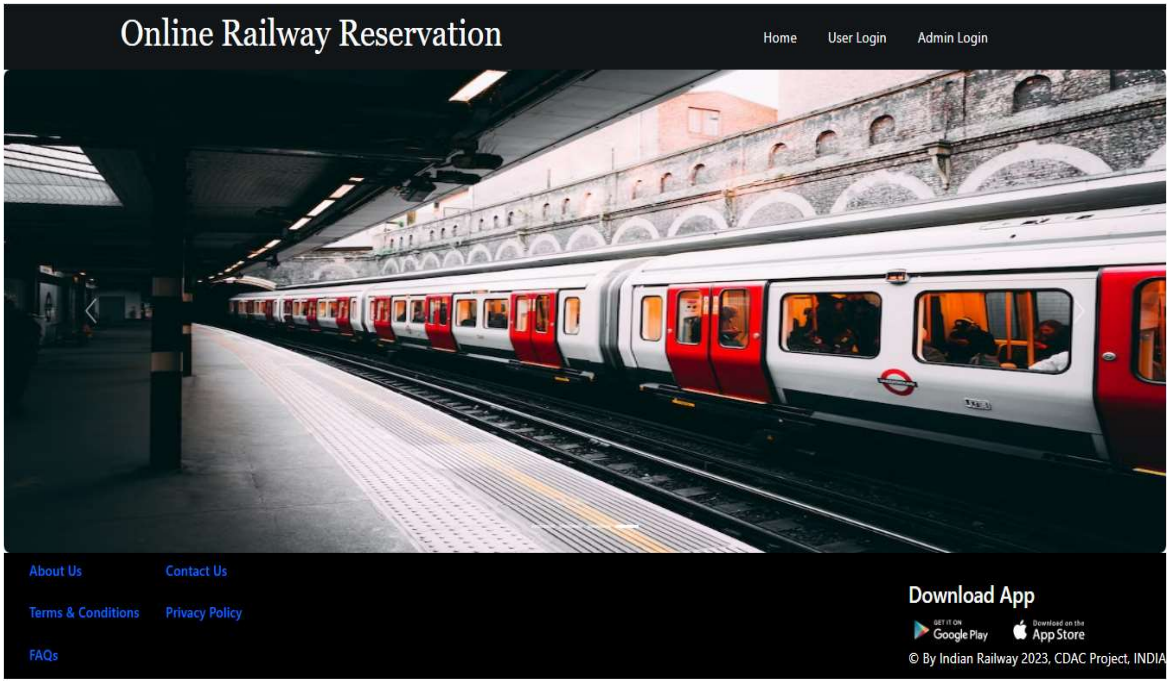


- **Class Diagram**

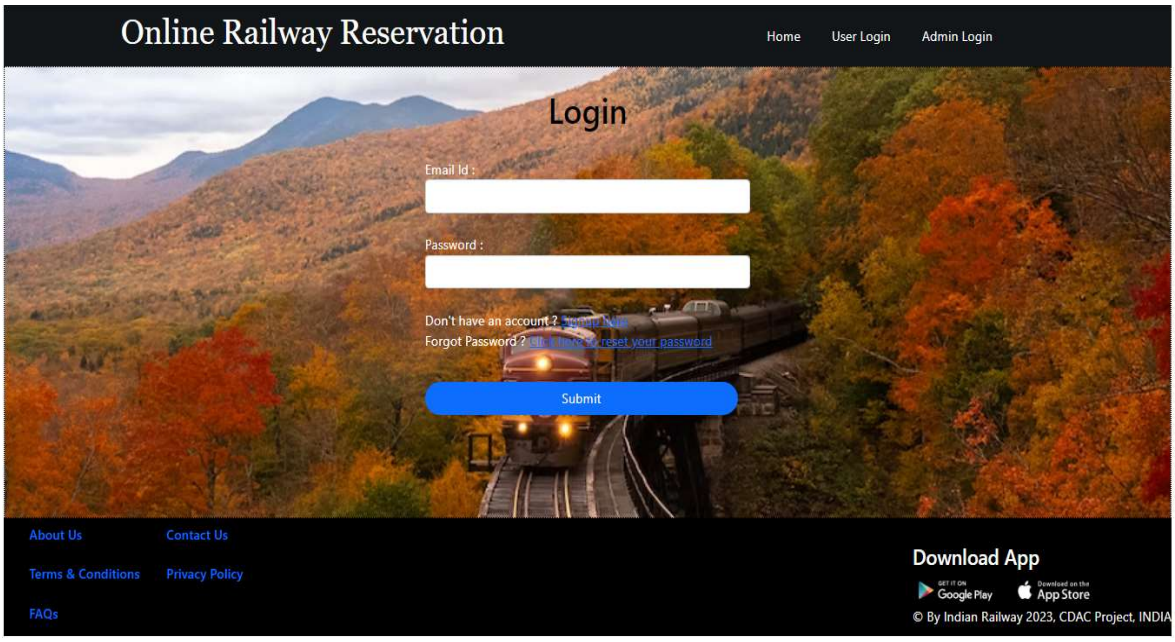


Appendix B

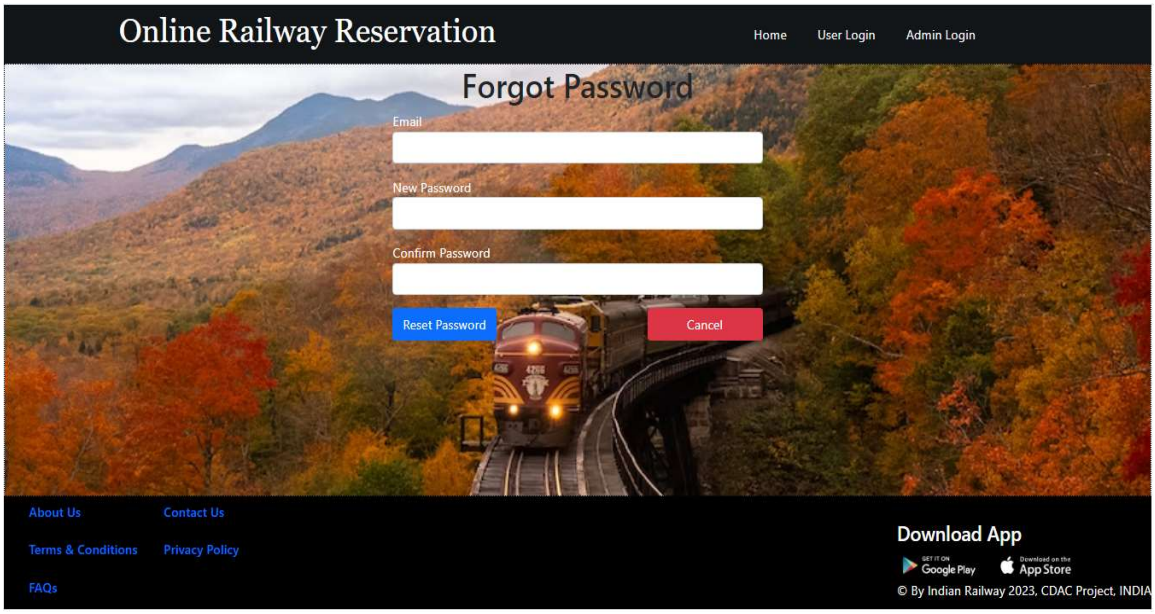
- **Home Page**



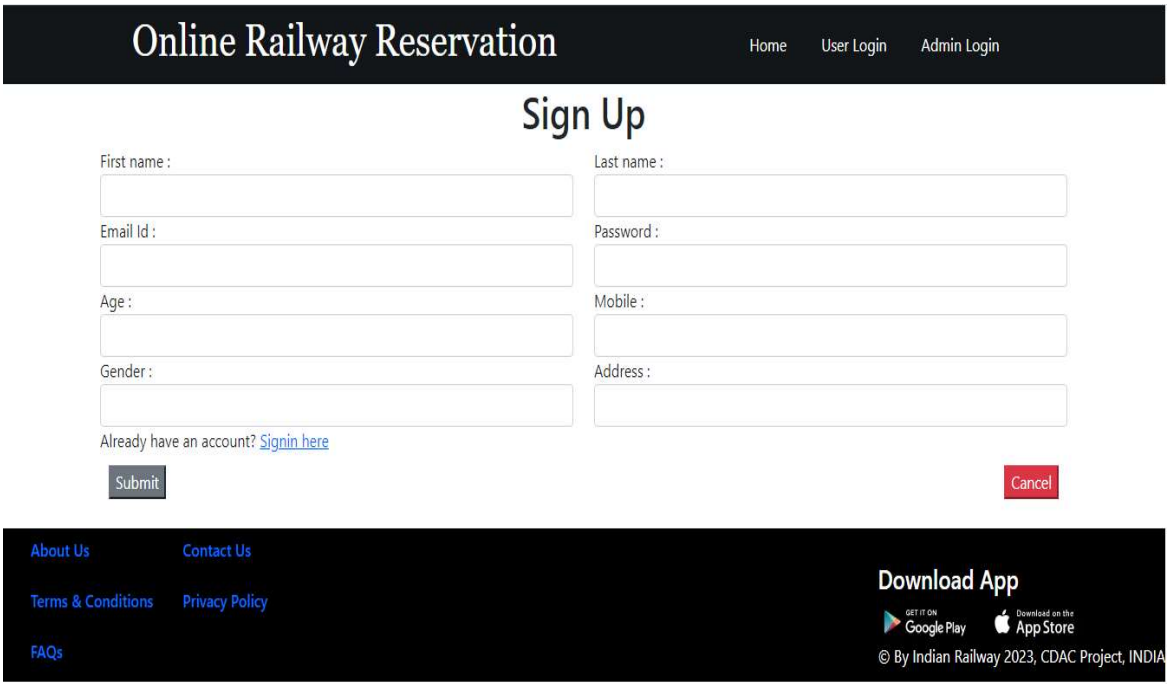
- **Login Page**



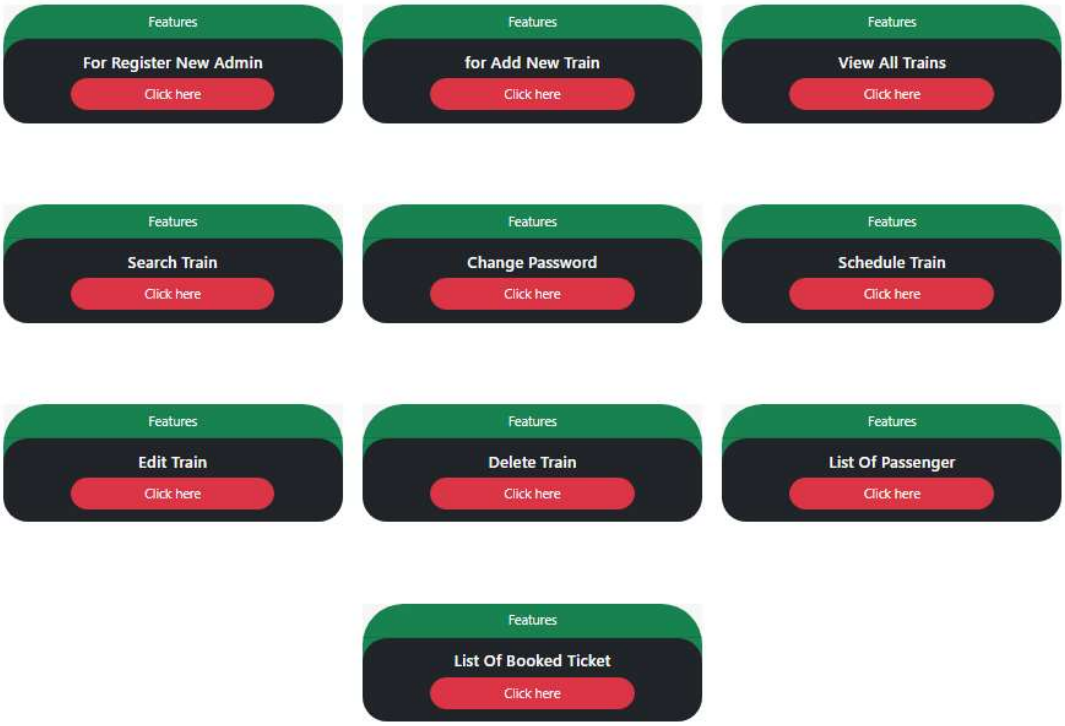
- **Forgot Password**



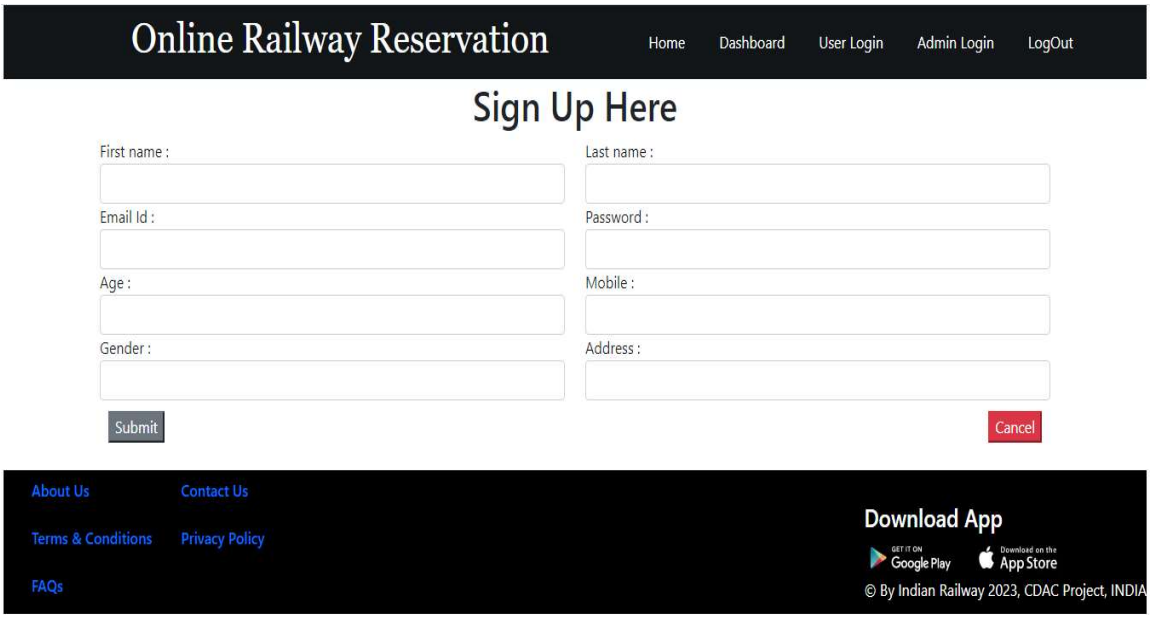
- **User Registration**



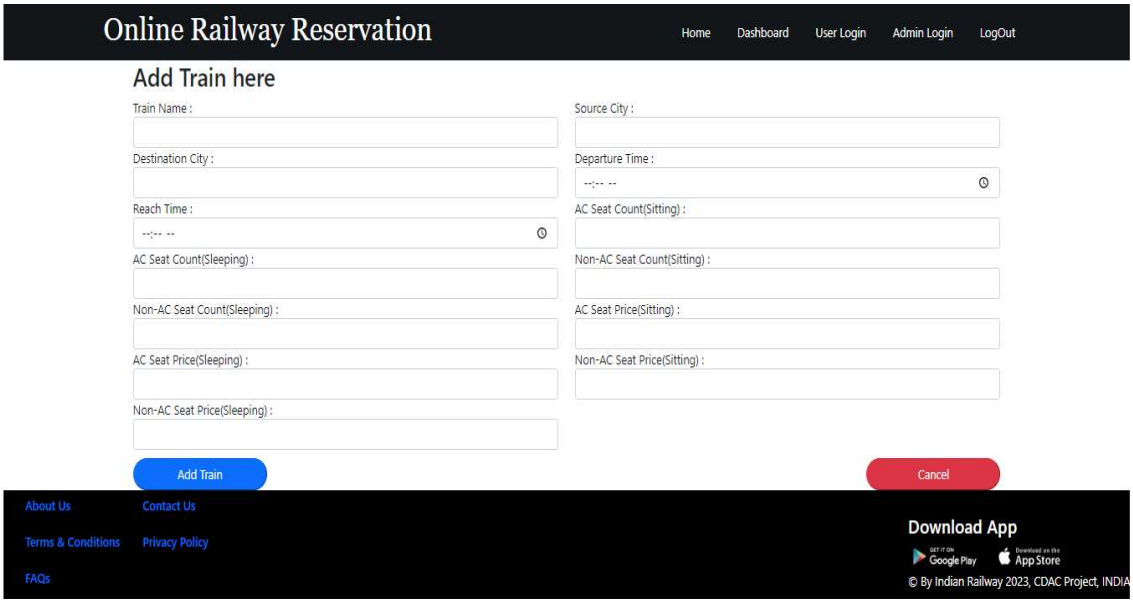
- **Admin Login**



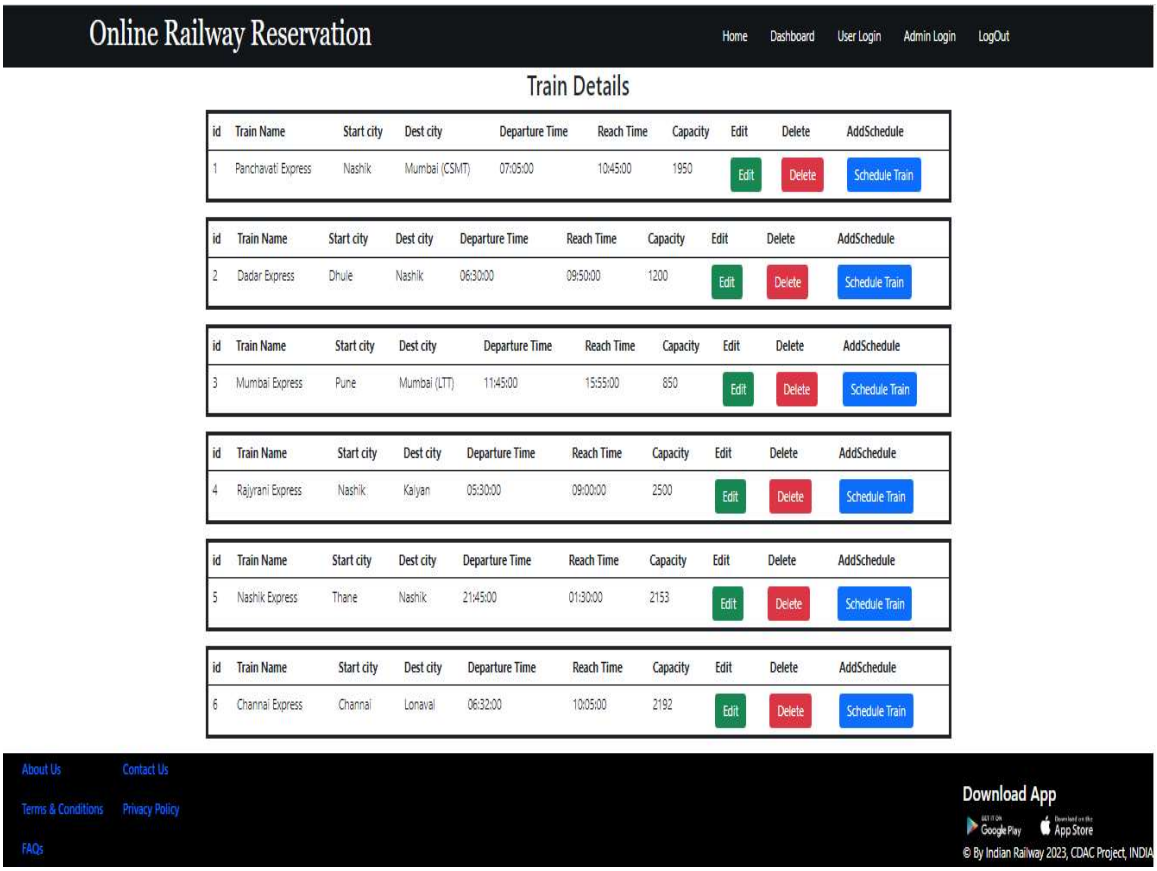
- **Add Admin**



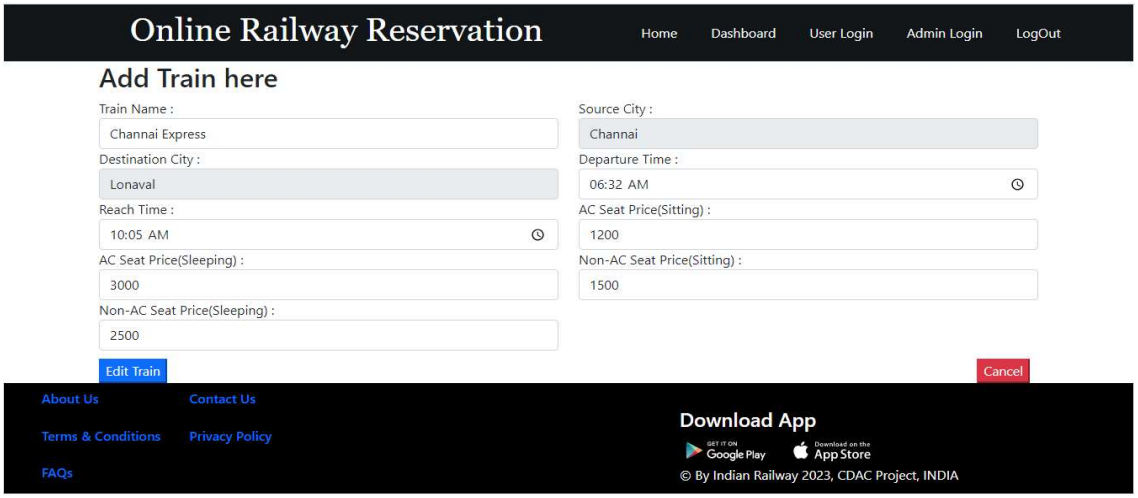
- **Add Train**



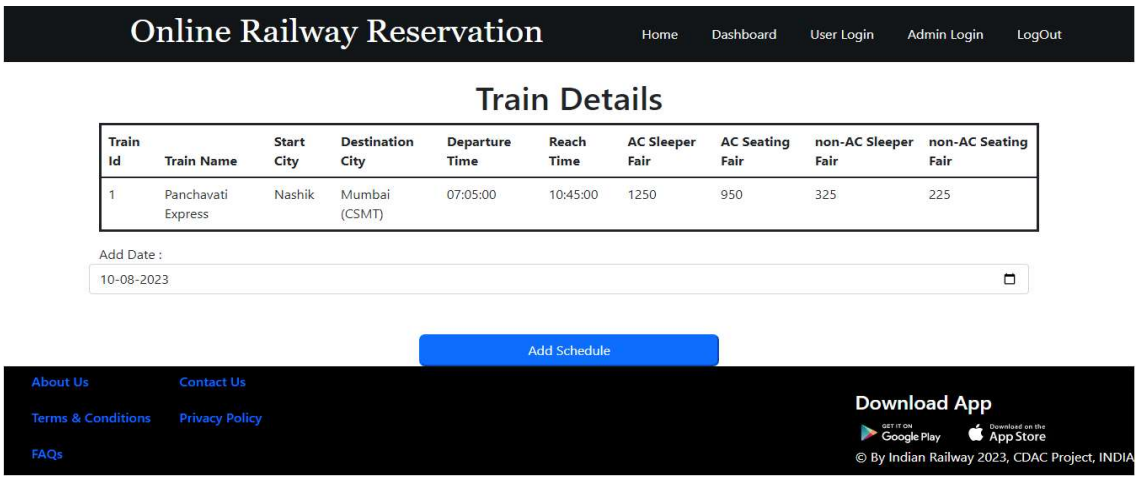
- **Train Details**



- **Update Train**

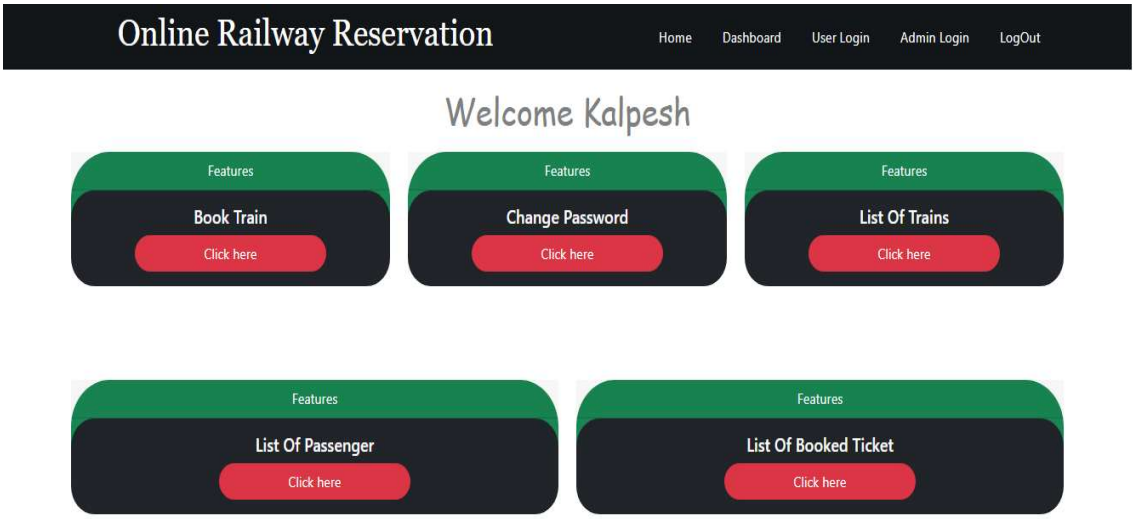


- **Add Schedule**

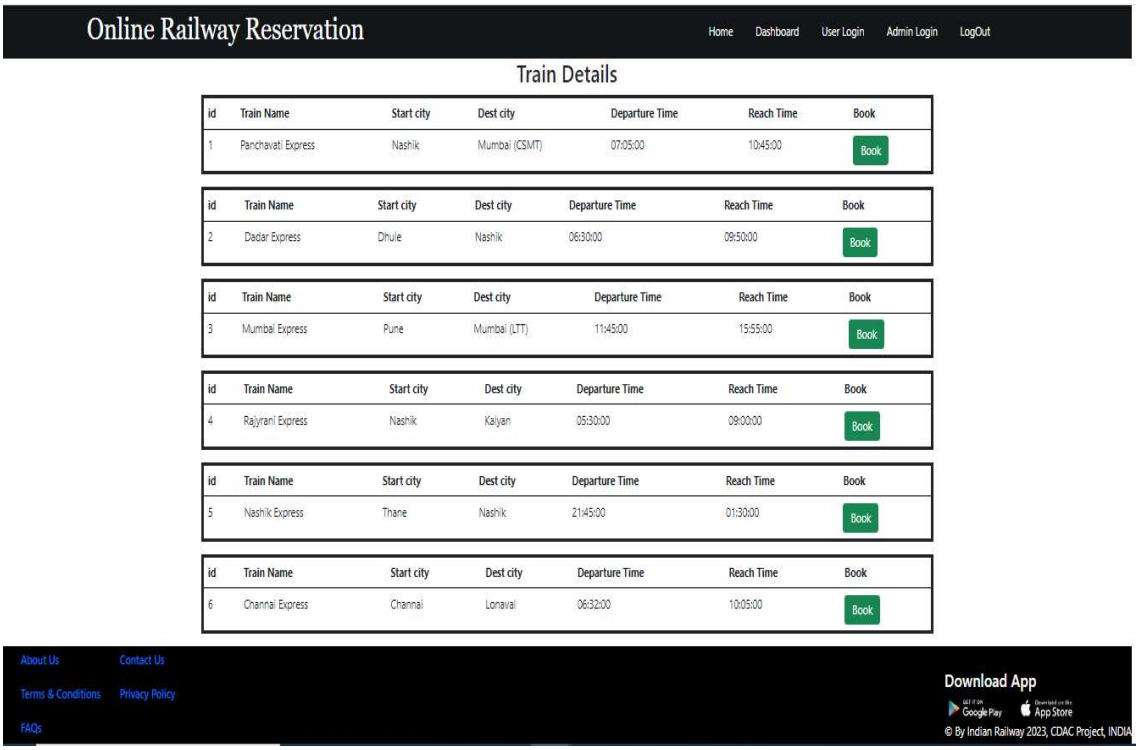


- **Passenger List**

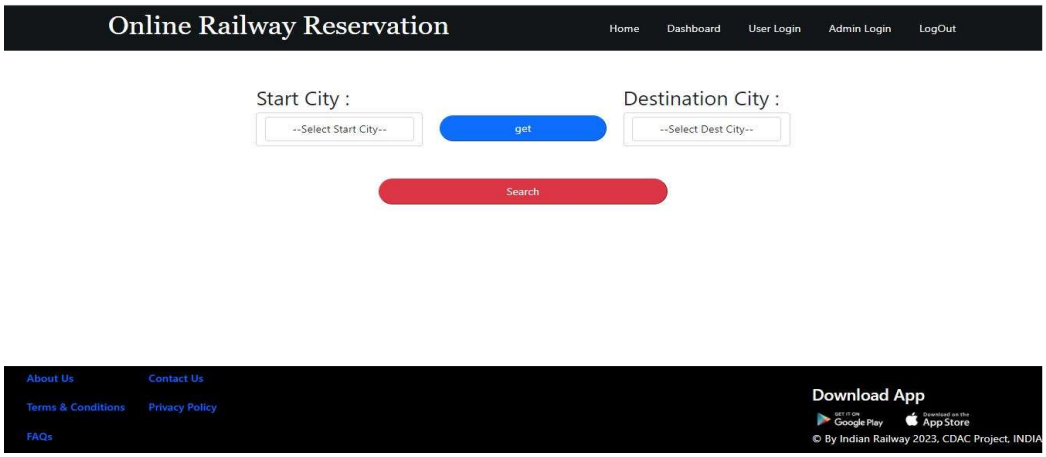
- **User Functionality**



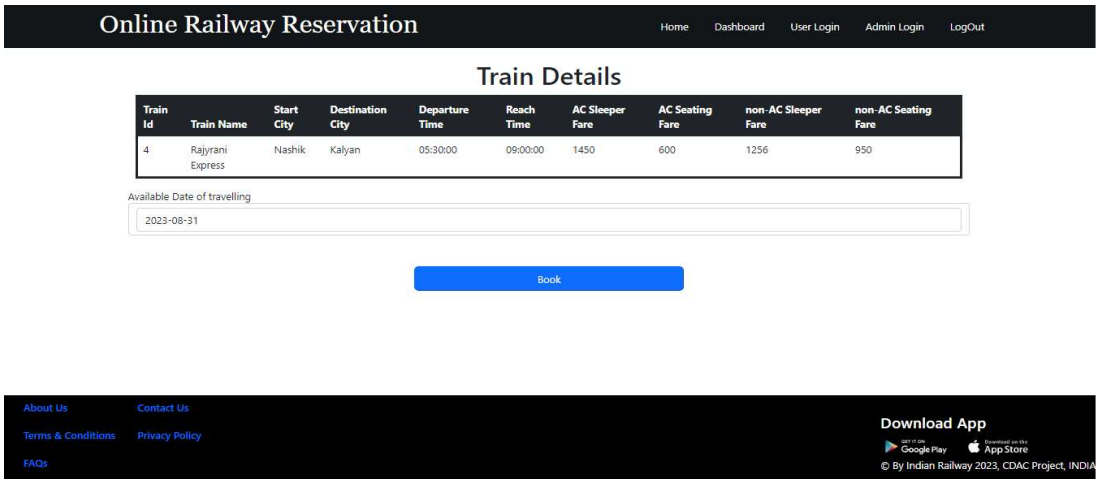
- **List of Trains**



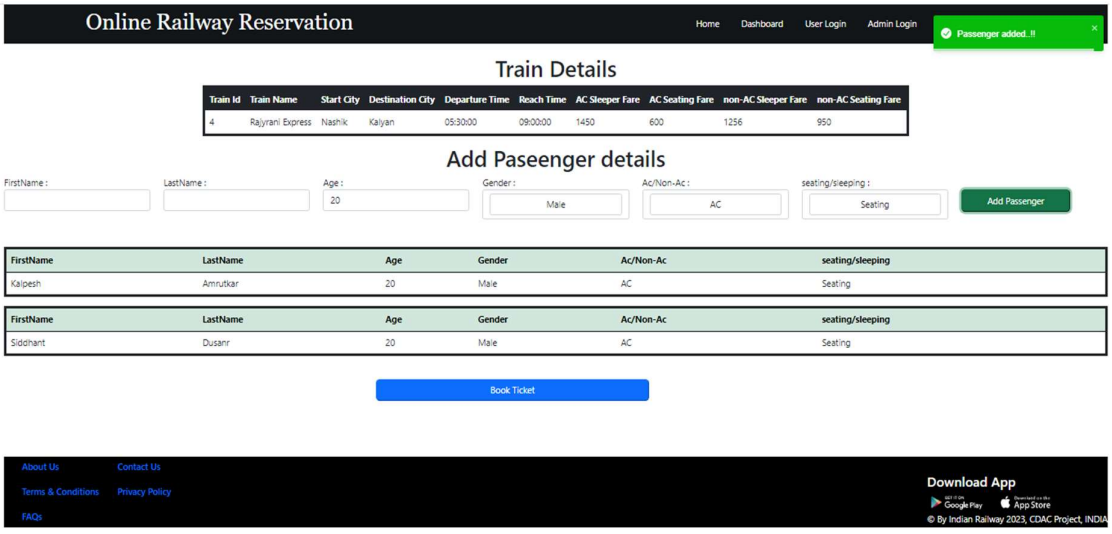
- **Search Train**



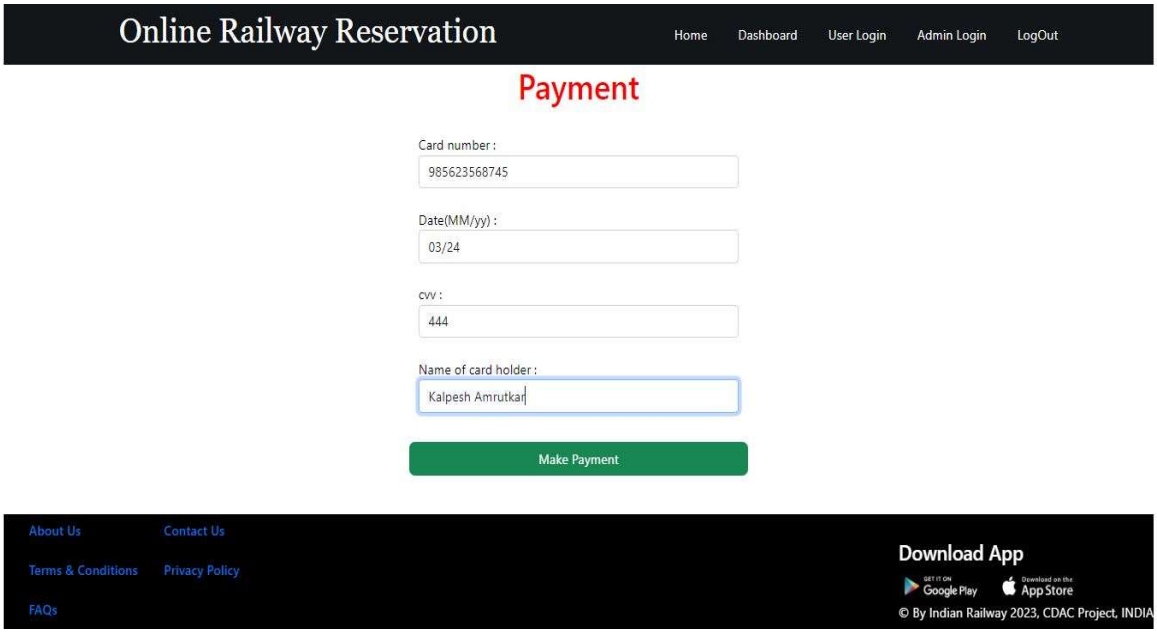
- **Train Details**



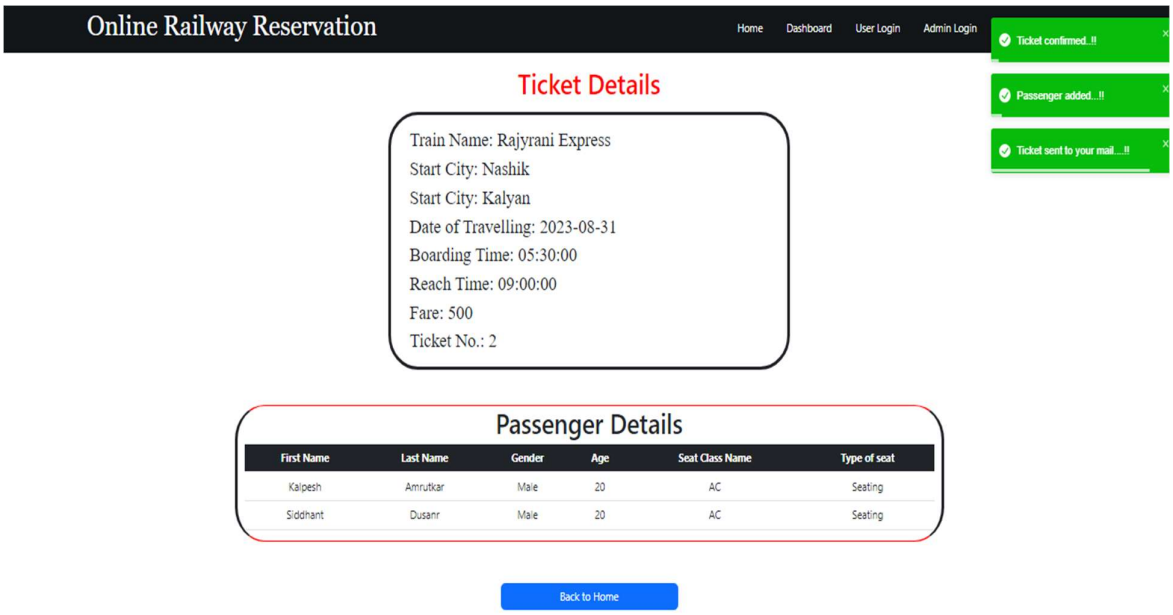
- **Add Passenger**



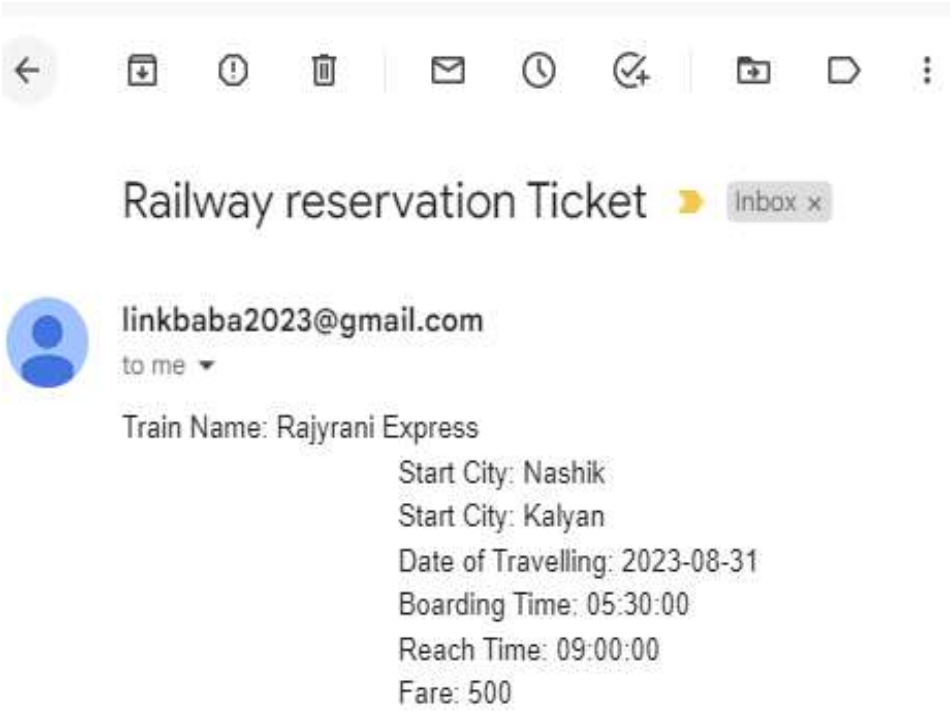
- **Payment Page**



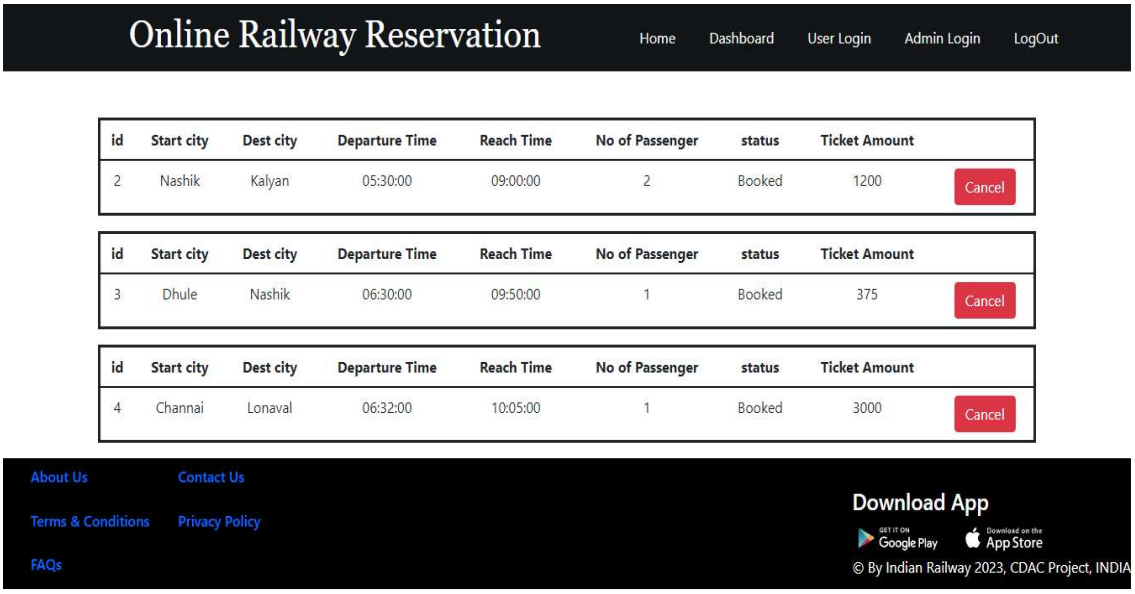
- **Ticket Details**



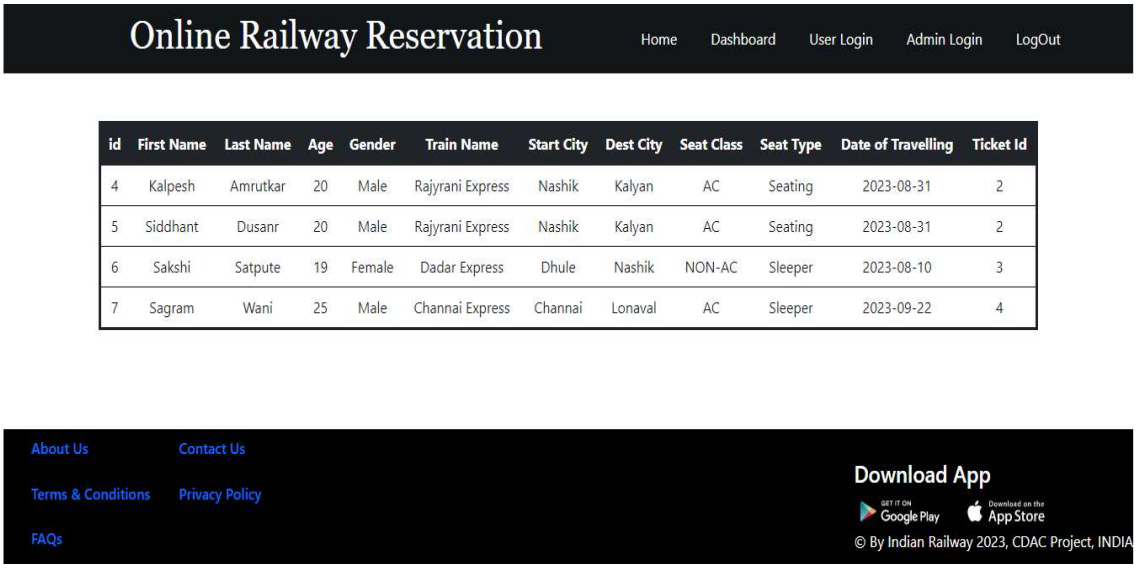
- **Mail To User**



- **Booked Ticket Details**



- **Passenger List**



id	First Name	Last Name	Age	Gender	Train Name	Start City	Dest City	Seat Class	Seat Type	Date of Travelling	Ticket Id
4	Kalpesh	Amrutkar	20	Male	Rajyrani Express	Nashik	Kalyan	AC	Seating	2023-08-31	2
5	Siddhant	Dusanr	20	Male	Rajyrani Express	Nashik	Kalyan	AC	Seating	2023-08-31	2
6	Sakshi	Satpute	19	Female	Dadar Express	Dhule	Nashik	NON-AC	Sleeper	2023-08-10	3
7	Sagram	Wani	25	Male	Channai Express	Channai	Lonaval	AC	Sleeper	2023-09-22	4