Project Report On

Train Ticket Booking System

Submitted to

Chandigarh University

In partial fulfillment of

Master of Computer Application(MCA)

Batch - Jul 2023

Fourth Semester

Submitted by

Rohit Janaba Hongekar

(O23MCA110010)

CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

CERTIFICATE

This is to certify that ,

Rohit Janaba Hongekar

Have successfully completed the project on the topic “Train Ticket Booking System” in satisfactory manner for partial fulfillment of **Master of Computer Application(MCA)** degree for the academic year 2023-25.

To the best of our knowledge and belief, the matter presented here is original and has not been submitted elsewhere for the award of any degree.

Date:- / / 2025

Place:- Pune

Project Guide HOD

Examinar

DECLARATION

I undersigned hereby declare that this report entitled **“TRAIN TICKET BOOKING SYSTEM”**for Chandigarh University,Chandigarh is our original work prepared under guidance of Faculty of Computer Science. The Empirical finding in this report are based on data collected by me. The matter presented is this report is not copied from any source. I understand that such copy is liable for punishment in any way the university Authorities deems to fit. This work has not been submitted to either Chandigarh University or any other University.

This work is humbly submitted to Chandigarh University, Chandigarh for the award of the degree of Master of Computer Application.

Date:- / / 2025
Place:- Pune

Rohit Hongekar

Acknowledgement

It gives great pleasure to remain deeply indebted to our guide Asst.Profssors of Chandigarh University for providing us with the required facilities for the academic achievement under whom we had the privilege to work. The faith and Confidence shown by him in us boosted our moral and motivated us to perform better in preferring this project.

We are thankful to those who have contributed either directly or indirectly to this project.

Thanking You,
Rohit Hongekar

Abstract

This project is a web-based application developed using **Java (J2EE), JDBC, Servlets** and **MySQL**. The primary objective of this system is to automate the train ticket reservation process, offering a seamless, efficient, and user-friendly interface for both passengers and administrators.

Built using **Java J2EE** technologies, the system follows a multi-tier architecture that separates presentation, business logic, and data access layers. The **Servlets** handle HTTP requests and responses, acting as the controller in the application, ensuring smooth navigation and data flow between the user interface and the database.

The system uses **JDBC (Java Database Connectivity)** to establish a secure and efficient connection between the application and the **MySQL database**, where all critical data such as user details, train schedules, bookings, and cancellations are stored and managed. Real-time interaction with the database allows users to search for available trains, check seat availability, book tickets, and receive unique PNR numbers for each booking.

The administrative module provides functionality for managing train details, updating schedules, and monitoring ticket reservations. The use of **MySQL** ensures reliable, structured, and fast access to data, while Java's robust architecture provides platform independence, security, and performance.

Overall, this project demonstrates the integration of core Java technologies in building a dynamic and database-driven web application. It not only enhances the booking experience for users but also simplifies operational management for railway authorities, contributing to a modern and efficient transportation system.

Table of Contents

Introduction

The Train Ticket Booking System is a web-based software solution designed to automate and simplify the process of booking railway tickets. With the growing demand for digital services, traditional methods of manual ticket booking have become inefficient, time-consuming, and prone to human error. This project aims to overcome these limitations by developing a dynamic, secure, and user-friendly online train reservation system.

Developed using Java (J2EE) technology, the application follows a structured and scalable architecture that separates concerns across different layers: presentation, business logic, and data access. Servlets are used to manage the flow of information between the front-end interface and the back-end database, ensuring a smooth and responsive user experience. The system uses JDBC (Java Database Connectivity) to interact with a MySQL database, where all essential data such as user information, train schedules, seat availability, and booking history is stored and maintained securely.

The system offers key functionalities such as:

- User registration and authentication
- Train search based on source, destination, and date
- Real-time seat availability check
- Ticket booking and cancellation
- PNR generation and e-ticket issuance
- Administrative control for managing train details and schedules

From the user's perspective, the system provides a convenient platform to manage train travel from anywhere, reducing dependency on physical counters. For administrators, it offers tools to efficiently manage train data and monitor bookings.

Existing System

The current or existing train ticket booking system primarily includes manual reservation systems or partially digitized systems such as ticket counters at railway stations and legacy web portals maintained by railway authorities (e.g., IRCTC in India). While these systems are functional, they come with several limitations that affect both users and railway management staff.

Limitations of the Existing System:

Manual Process Dependency:

Many traditional systems still rely on physical ticket counters where passengers must stand in long queues to book or cancel tickets.

Time-Consuming:

Booking or cancellation takes significant time, especially during peak travel seasons or in rural areas with limited access to digital systems.

Human Errors:

Manual data entry by staff can lead to errors in booking details, schedules, or fare calculations.

Lack of Real-Time Updates:

The current system often fails to provide real-time seat availability or train schedule updates, leading to outdated or incorrect information being shared with passengers.

Limited Accessibility:

Not all users have easy access to the existing online systems. They may also be too complex or non-responsive on mobile devices.

Low User Engagement:

Existing systems may not provide features like booking history, auto-fill of previous data, personalized suggestions, or user-friendly interfaces.

Limitations of the Existing Manual System

The existing manual train ticket booking system, which involves physical ticket counters and paper-based records, suffers from several drawbacks that hinder efficiency, accuracy, and user satisfaction. Below are the major limitations:

Lack of Real-Time Seat Availability:

Passengers cannot instantly check the availability of seats or get updates on changes in schedules, leading to confusion and planning issues.

Lack of Transparency:

Users have limited visibility into the ticketing process, fares, and seat assignment, which can lead to dissatisfaction.

Inefficient for Staff:

Railway employees spend significant time on repetitiveadministrative tasks, reducing their productivity and increasing operational costs.

Difficult to Scale:

The manual system cannot handle a high volume of transactions effectively, leading to bottlenecks during festival seasons or emergencies.

No Instant Confirmation:

Passengers often have to wait to receive physical tickets, with no immediate confirmation or digital record.

SDLC of the Project

The SDLC for this project follows the Waterfall Model, which ensures structured development and completion of each phase before moving on to the next.

It consists of a detailed plan describing how to develop, maintain, replace, and alter or enhance specific software.

1. Requirement Analysis:

In this phase, the functional and non-functional requirements of the system were gathered through:

- Study of existing manual booking systems
- Interaction with users (passengers and railway staff)
- Analysis of common pain points

  The goal was to identify the scope of the system, user expectations, and system constraints.

2. System Design:

This phase included:

- Defining system architecture (3-tier architecture)
- Designing user interfaces for passengers and admin
- Database design using MySQL

3.Implementation (Coding):

Based on the design:

- The system was developed using Java J2EE technologies.
- Servlets were used for handling requests and controlling application flow.
- JDBC was used to connect the application to a MySQL database for all CRUD (Create, Read, Update, Delete) operations.
- Basic HTML/CSS was used for front-end UI design.

Testing:

Testing was performed to ensure system correctness and reliability:

- Unit Testing: Individual modules like login, booking, cancellation were tested.
- Integration Testing: Ensured smooth interaction between servlets, database, and user interface.
- System Testing: Validated the complete system for real-time scenarios and edge cases.

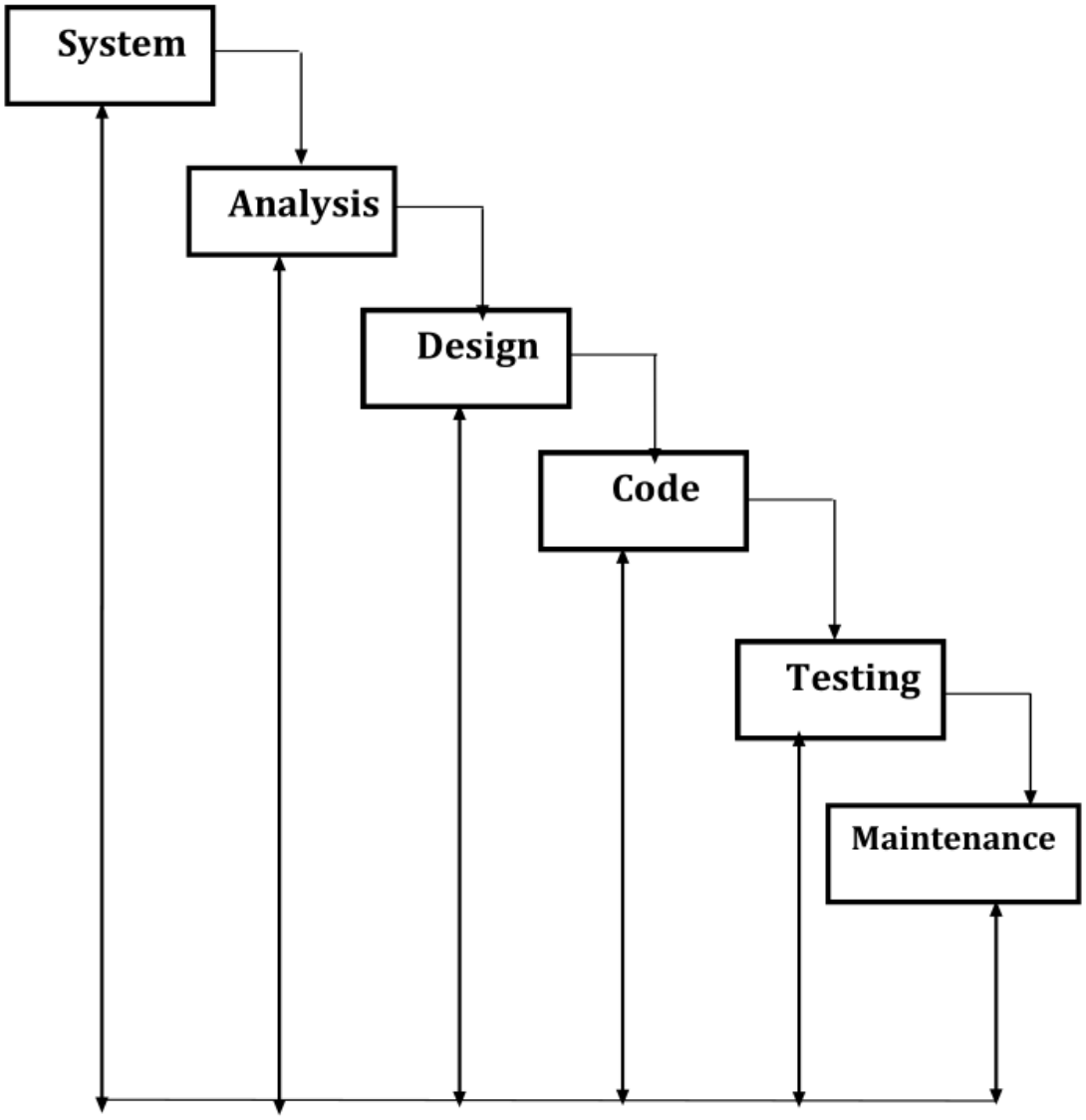
Deployment:

After successful testing, the application was deployed in a server environment (e.g., Apache Tomcat) for demonstration. The system was made accessible to users through a browser.

Maintenance:

Post-deployment, this phase ensures:

- Bug fixes, if any are found
- Performance enhancements
- Updates based on user feedback and future scalability needs



Design

Database Design

Admin Table :

Column Name	Data Type	Description
MAILID	varchar(40)	Mail ID
PWORD	varchar(20)	Password
FNAME	varchar(20)	First Name
LNAME	varchar(20)	Last Name
ADDR	varchar(100)	Address
PHNO	varchar(10)	Phone Number

Customer Table:

Column Name	Data Type	Description
MAILID	varchar(40)	Mail ID
PWORD	varchar(20)	Password
FNAME	varchar(20)	First Name
LNAME	varchar(20)	Last Name
ADDR	varchar(100)	Address
PHNO	varchar(10)	Phone Number

History Table:

Column Name	Data Type	Description
TRANSID	varchar(36)	Transaction ID
MAILID	varchar(40)	Mail ID
TR_NO	int	Train Number
DATE	date	Date
FROM_STN	varchar(20)	From Station
TO_STN	varchar(20)	To Station
SEATS	int	Seats Availble
AMOUNT	decimal(8,2)	Amount

Train Table:

Column Name	Data Type	Description
TR_NO	int	Train Number
TR_NAME	varchar(70)	Train Name
FROM_STN	varchar(20)	From Station
TO_STN	varchar(20)	To Station
SEATS	int	Seats Availble
FARE	double	Fare Amount

Source Code :

Java Source Code:

```
package com.rohit.servlets;
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.rohit.beans.TrainBean;
import com.rohit.beans.TrainException;
import com.rohit.constant.ResponseCode;
import com.rohit.constant.UserRole;
import com.rohit.service.TrainService;
import com.rohit.service.impl.TrainServiceImpl;
import com.rohit.utility.TrainUtil;

@WebServlet("/adminaddtrain")
public class AdminAddTrain extends HttpServlet {

	/**
	 *
```

```
	 */
	private static final long serialVersionUID = 1L;

	private TrainService trainService = new TrainServiceImpl();

	/**
	 *
	 * @param req
	 * @param res
	 * @throws IOException
	 * @throws ServletException
	 */
	protected void doPost(HttpServletRequest req, HttpServletResponse
res) throws IOException, ServletException {
		res.setContentType("text/html");
		PrintWriter pw = res.getWriter();
		TrainUtil.validateUserAuthorization(req, UserRole.ADMIN);
		try {
			TrainBean train = new TrainBean();

	train.setTr_no(Long.parseLong(req.getParameter("trainno")));

	train.setTr_name(req.getParameter("trainname").toUpperCase());

	train.setFrom_stn(req.getParameter("fromstation").toUpperCase());

	train.setTo_stn(req.getParameter("tostation").toUpperCase());
```

```
        train.setSeats(Integer.parseInt(req.getParameter("available")));

        train.setFare(Double.parseDouble(req.getParameter("fare")));
                        String message = trainService.addTrain(train);
                        if
(ResponseCode.SUCCESS.toString().equalsIgnoreCase(message)) {
                                RequestDispatcher rd =
req.getRequestDispatcher("AddTrains.html");
                                rd.include(req, res);
                                pw.println("<div class='tab'><p1
class='menu'>Train Added Successfully!</p1></div>");
                        } else {
                                RequestDispatcher rd =
req.getRequestDispatcher("AddTrains.html");
                                rd.include(req, res);
                                pw.println("<div class='tab'><p1
class='menu'>Error in filling the train Detail</p1></div>");
                        }
                } catch (Exception e) {
                        throw new TrainException(422,
this.getClass().getName() + "_FAILED", e.getMessage());
                }

        }

}
```

JDBC Connection :

username=root

password=

driverName = com.mysql.cj.jdbc.Driver

connectionString=jdbc:mysql://localhost:3306/Train

HTML Code :

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>E-Trains</title>
<link rel="stylesheet" href="UserHome_Css.css">
</head>
<body>
	<header>
		<h1 class="hd">National Ticket Booking System</h1>
		<div class="home">
			<p class="menu">
				<a href="userhome">Home</a>
			</p>
		</div>
		<div class="home">
```

```
                    <p class="menu">
                        <a href="userviewtrainfwd">View Trains</a>
                    </p>
                </div>
                <div class="home">
                    <p class="menu">
                        <a href="trainbwstnfwd">Trains Between
Stations</a>
                    </p>
                </div>
                <div class="home">
                    <p class="menu">
                        <a href="bookingdetails">Ticket Booking
History</a>
                    </p>
                </div>
                <div class="home">
                    <p class="menu">
                        <a href="fareenqfwd">Fare Enquiry</a>
                    </p>
                </div>
                <div class="home">
                    <p class="menu">
                        <a href="useravailfwd">Seat Availability</a>
                    </p>
                </div>
                <div class="home">
```

```
                    <p class="menu">
                        <a href="usersearchtrain">Search By
TrainNo</a>
                    </p>
                </div>
                <div class="home">
                    <p class="menu">
                        <a href="userprofile">Profile</a>
                    </p>
                </div>
                <div class="home">
                    <p class="menu">
                        <a href="userlogout">Logout</a>
                    </p>
                </div>
        </header>
        <div class="tab green">User Login</div>
        <form action="userlogin" class="tab brown" method="post">
                <br /> UserName: <input type="text" name="uname"
                    placeholder="Enter Your EmailId"> <br /> <br />
Password: <input
                    type="password" name="pword"> <br /> <br /> <input
                    type="submit" value=" LOGIN "><br />


        </form>
        <br />
</body>
```

```
</html>
```

Coding & Implementation

Source Code Librery -

https://github.com/rohithongekar/train-ticket-booking

Admin Login -

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.rohit.beans.TrainException;
import com.rohit.constant.UserRole;
import com.rohit.utility.TrainUtil;

@SuppressWarnings("serial")
@WebServlet("/adminlogin")
public class AdminLogin extends HttpServlet {

	/**
	 *
	 * @param req
	 * @param res
```

```
         * @throws IOException
         * @throws ServletException
         */
        protected void doPost(HttpServletRequest req, HttpServletResponse
res) throws IOException, ServletException {
                PrintWriter pw = res.getWriter();
                res.setContentType("text/html");
                String uName = req.getParameter("uname");
                String pWord = req.getParameter("pword");
                try {
                        String message = TrainUtil.login(req, res,
UserRole.ADMIN, uName, pWord);
                        if ("SUCCESS".equalsIgnoreCase(message)) {

                                RequestDispatcher rd =
req.getRequestDispatcher("AdminHome.html");
                                rd.include(req, res);
                                pw.println("<div class='main'><p1
class='menu'>Hello, " + uName + " ! Welcome </p1></div>");
                                pw.println("<div class='tab'>Hi ! Here You can
Manage Train Information as per Your Requirement</div>");

                        } else {
                                RequestDispatcher rd =
req.getRequestDispatcher("AdminLogin.html");
                                rd.include(req, res);
```

```
                    pw.println("<div  class='tab'><p1  class='menu'>"
+ message + "</p1></div>");

                }
            } catch (Exception e) {
                throw new TrainException(422,
this.getClass().getName() + "_FAILED", e.getMessage());
            }
        }
}
```

User Home Page -

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.rohit.constant.UserRole;
import com.rohit.utility.TrainUtil;

@SuppressWarnings("serial")
@WebServlet("/userhome")
```

```
public class UserHome extends HttpServlet {
	protected void doGet(HttpServletRequest req, HttpServletResponse res) throws IOException, ServletException {
		res.setContentType("text/html");
		PrintWriter pw = res.getWriter();
		TrainUtil.validateUserAuthorization(req, UserRole.CUSTOMER);
		RequestDispatcher rd = req.getRequestDispatcher("UserHome.html");
		rd.include(req, res);
		pw.println("<div class='tab'>" + "		<p1 class='menu'>" + "  Hello " + TrainUtil.getCurrentUserName(req)
				+ " ! Welcome to our new NITRTC Website" + "</p1>" + "	</div>");
		pw.println("<div class='main'><p1 class='menu'>User Home</p1></div>");
		pw.println("<div class='tab'>Hello " + TrainUtil.getCurrentUserName(req)
				+ " ! Good to See You here.<br/> Here you can Check up the train "
				+ "details, train schedule, fare Enquiry and many more information.<br/>Just go to the Side Menu Links and "
				+ "Explore the Advantages.<br/><br/>Thanks For Being Connected with us!" + "</div>");

	}
}
```

HTML Admin Home Page -

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>E-Trains</title>
<link rel="stylesheet" href="UserHome_Css.css">
</head>
<body>
	<header>
		<h1 class="hd">National Ticket Booking System</h1>
		<div class="home">
			<p class="menu">
				<a href="AdminHome.html">Home</a>
			</p>
		</div>
		<div class="home">
			<p class="menu">
				<a href="adminviewtrainfwd">View Trains</a>
			</p>
		</div>
		<div class="home">
			<p class="menu">
				<a	href="adminsearchtrainfwd">Search	By
TrainNo</a>
			</p>
```

```
            </div>

            <div class="home">
                <p class="menu">
                    <a href="addtrainfwd">Add Train</a>
                </p>
            </div>
            <div class="home">
                <p class="menu">
                    <a href="cancletrainfwd">Delete Train </a>
                </p>
            </div>
            <div class="home">
                <p class="menu">
                    <a href="updatetrain">Update Train Details</a>
                </p>
            </div>
            <div class="home">
                <p class="menu">
                    <a href="adminlogout">Logout</a>
                </p>
            </div>
        </header>

        <div class='tab'>
            <p class='menu'> Hey, Admin ! Welcome to our new NITRTC
Website
```

```
            </p>
        </div>



</body>
</html>
```

CSS Page -

```
/* ========== Reset & Base Styles ========== */
* {
    box-sizing: border-box;
    margin: 0;
    padding: 0;
}

body {
    font-family: 'Segoe UI', 'Roboto', 'Helvetica Neue', sans-serif;
    background: #121212 url('sandy.jpg') no-repeat center center fixed;
    background-size: cover;
    color: #e0e0e0;
    padding: 20px;
    line-height: 1.6;
}

/* ========== Header ========== */
.hd {
    text-align: center;
    font-size: 48px;
```

```
    color: red;
    font-weight: 800;
    margin-bottom: 40px;
    text-shadow: 2px 2px 10px rgba(0, 255, 255, 0.4);
}

/* ========== Main Content Box ========== */
.main {
    max-width: 650px;
    margin: 0 auto 40px;
    background-color: #1f1f1f;
    padding: 30px;
    border-radius: 15px;
    box-shadow: 0 10px 30px rgba(0, 0, 0, 0.6);
    text-align: center;
    font-size: 18px;
    color: #ff5252;
}

/* ========== Navigation Menu Buttons ========== */
.menu {
    display: inline-block;
    margin: 10px;
    padding: 12px 24px;
    background-color: #212121;
    color: #00e5ff;
    font-weight: bold;
```

```
    border: 2px solid #00e5ff;
    border-radius: 50px;
    font-size: 14px;
    text-transform: uppercase;
    letter-spacing: 1px;
    transition: all 0.3s ease;
    cursor: pointer;
}

.menu:hover {
    background-color: #00e5ff;
    color: #000;
    box-shadow: 0 0 10px #00e5ff;
    transform: scale(1.05);
}

/* ========== Links ========== */
a {
    color: #ff4081;
    text-decoration: none;
    transition: color 0.3s ease;
}

a:hover {
    color: #69f0ae;
}
```

```
/* ========== Table Container ========== */
.tab {
    background-color: #1c1c1c;
    padding: 25px;
    border-radius: 18px;
    max-width: 1000px;
    margin: 40px auto;
    box-shadow: 0 10px 25px rgba(0, 0, 0, 0.7);
    font-weight: 500;
    font-size: 16px;
    color: #ccc;
}

/* ========== Home Button ========== */
.home {
    display: inline-block;
    background-color: #ffca28;
    color: #0d47a1;
    font-style: italic;
    font-weight: bold;
    padding: 10px 25px;
    border: 2px solid #0d47a1;
    border-radius: 10px;
    margin-bottom: 20px;
    transition: all 0.3s ease-in-out;
}
```

```
.home:hover {
    background-color: #ffd740;
    color: #000;
    transform: translateY(-2px);
}

/* ========== Table Styles ========== */
table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 20px;
    background-color: #2a2a2a;
    color: #e0e0e0;
}

th, td {
    border: 1px solid #444;
    padding: 14px 10px;
    text-align: center;
}

th {
    background-color: #263238;
    color: #00bcd4;
    font-size: 18px;
}
```

```
td {
    font-size: 16px;
}

tr:hover {
    background-color: #37474f;
}

/* ========== Utility Colors ========== */
.yel { color: #ffd740; }
.red { color: #ff5252; }
.green { color: #00e676; }
.brown { color: #8d6e63; }

/* ========== Footer ========== */
footer {
    margin-top: 100px;
    text-align: center;
    font-size: 14px;
    color: #888;
}
```

Testing

Create New User -

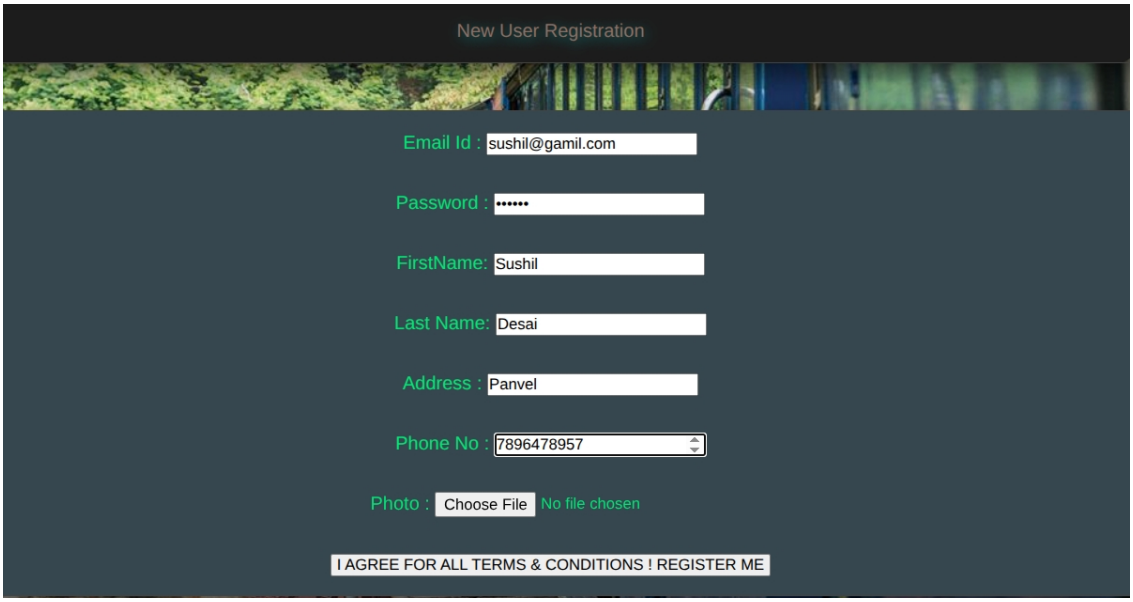
Before Adding New User -

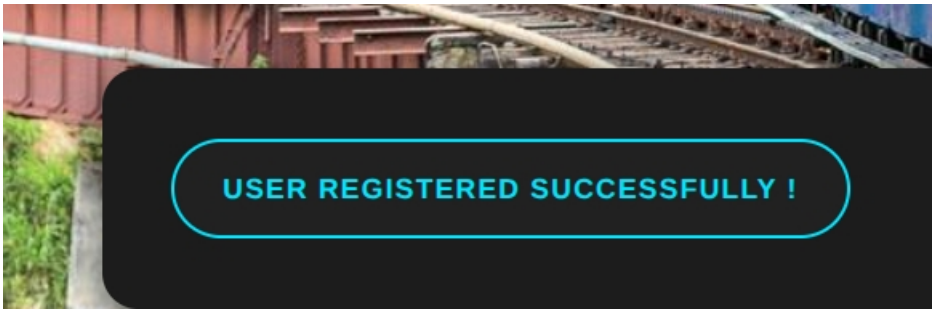
```
mysql> use Train;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from CUSTOMER;
+------------------+--------+-------+----------+--------------------+------------+
| MAILID           | PWORD  | FNAME | LNAME    | ADDR               | PHNO       |
+------------------+--------+-------+----------+--------------------+------------+
| raj@gamil.com    | 211321 | Raj   | Dixit    | koilhapur          | 911365542  |
| Rohit1@gmail.com | Rohit  | Rohit | Patil    | Kolhapur           | 7894579657 |
| Rohit@gmail.com  | rohit  | Rohit | Hongekar | Pune               | 788745145  |
| shashi@gmail.com | raj    | Raj   | Patil1   | Pune, Maharashatra | 954745222  |
+------------------+--------+-------+----------+--------------------+------------+
4 rows in set (0.00 sec)
```

After Adding New User -

In Application -

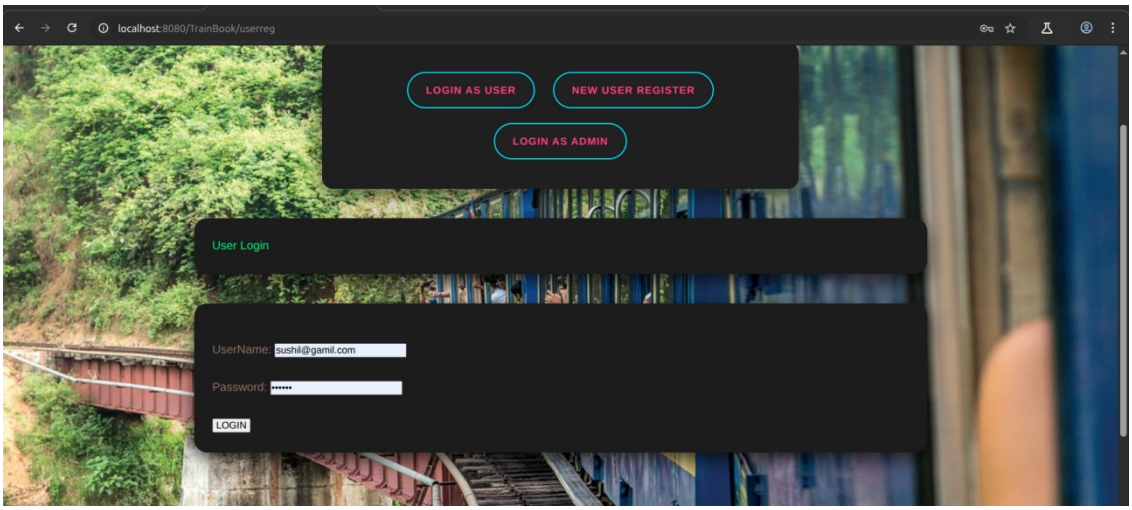




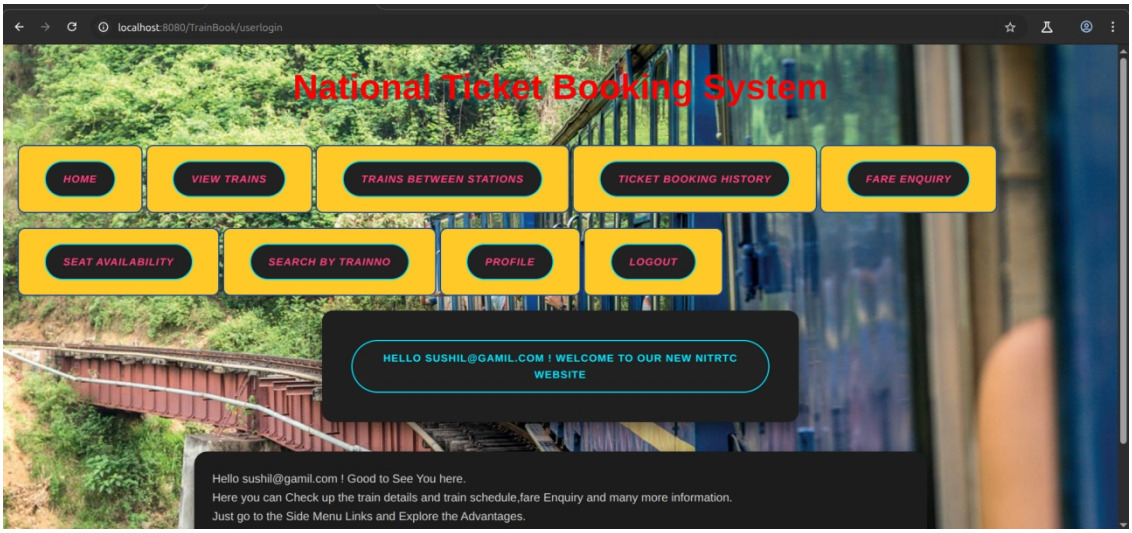
OUTPUT -

```
mysql> select * from CUSTOMER;
+------------------+--------+--------+----------+--------------------+------------+
| MAILID           | PWORD  | FNAME  | LNAME    | ADDR               | PHNO       |
+------------------+--------+--------+----------+--------------------+------------+
| raj@gamil.com    | 211321 | Raj    | Dixit    | koilhapur          | 911365542  |
| Rohit1@gmail.com | Rohit  | Rohit  | Patil    | Kolhapur           | 7894579657 |
| Rohit@gmail.com  | rohit  | Rohit  | Hongekar | Pune               | 788745145  |
| shashi@gmail.com | raj    | Raj    | Patil1   | Pune, Maharashatra | 954745222  |
| sushil@gamil.com | sushil | Sushil | Desai    | Panvel             | 7896478957 |
+------------------+--------+--------+----------+--------------------+------------+
5 rows in set (0.01 sec)
```

Login as User -



User got logged in Successful.

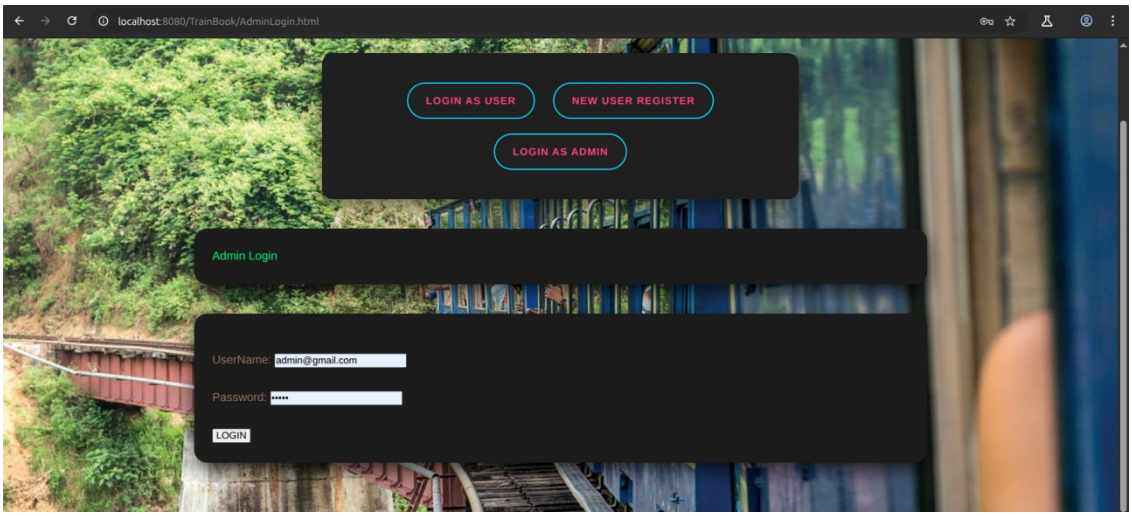


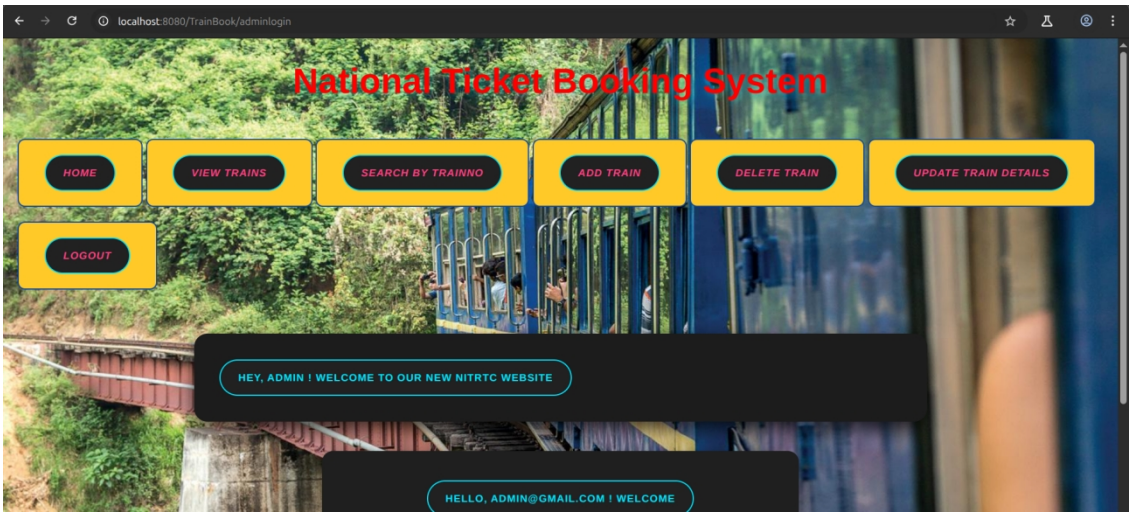
Login As Admin-

Admin Details in Database -

```
mysql> select * from ADMIN;
+-----------------+-------+--------+-------+------------------------+-----------+
| MAILID          | PWORD | FNAME  | LNAME | ADDR                   | PHNO      |
+-----------------+-------+--------+-------+------------------------+-----------+
| admin@gmail.com | admin | System | Admin | Demo Address 123 colony | 974785697 |
+-----------------+-------+--------+-------+------------------------+-----------+
1 row in set (0.01 sec)
```

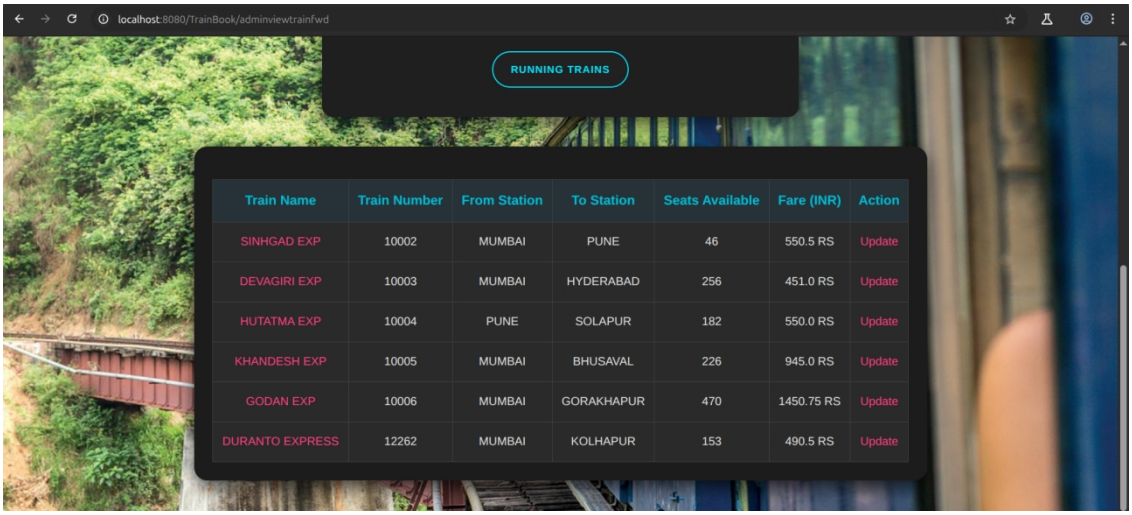
Logging As Admin -



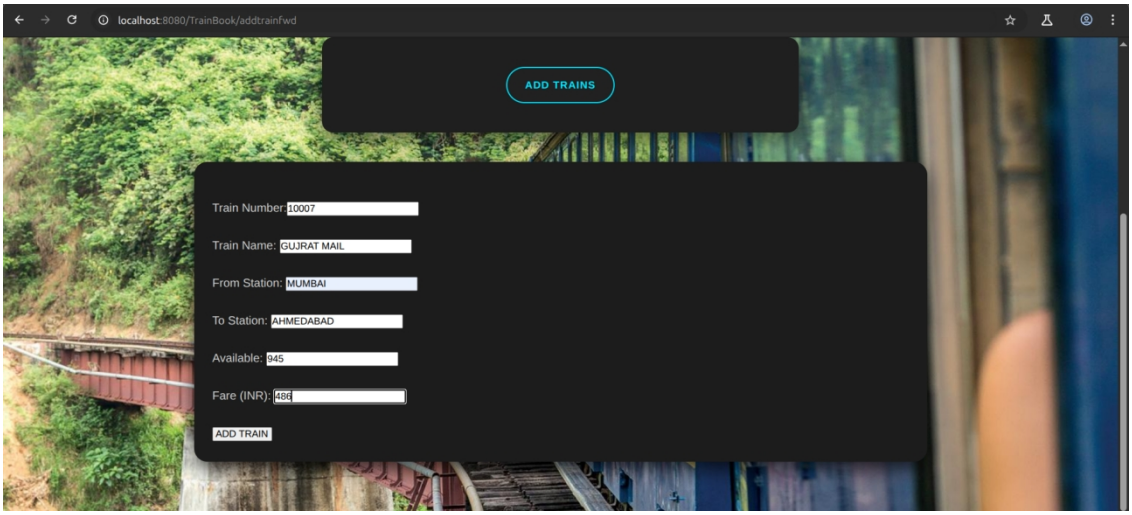


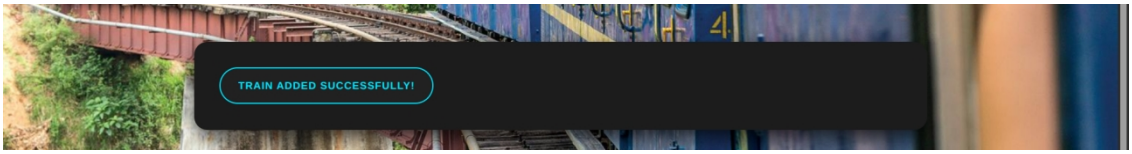
Adding New Train -

Before Adding New Train -

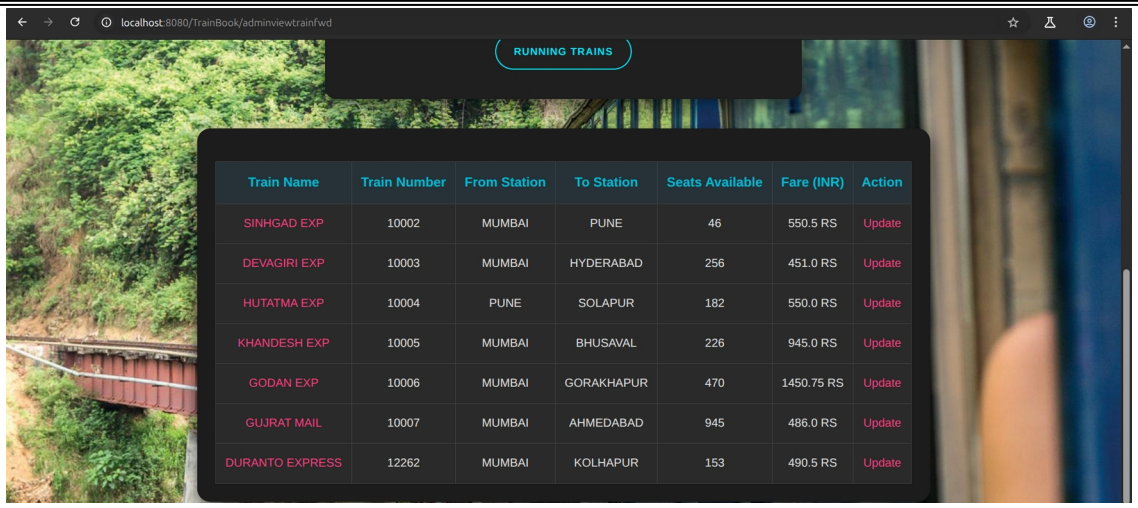


After Adding New Train -



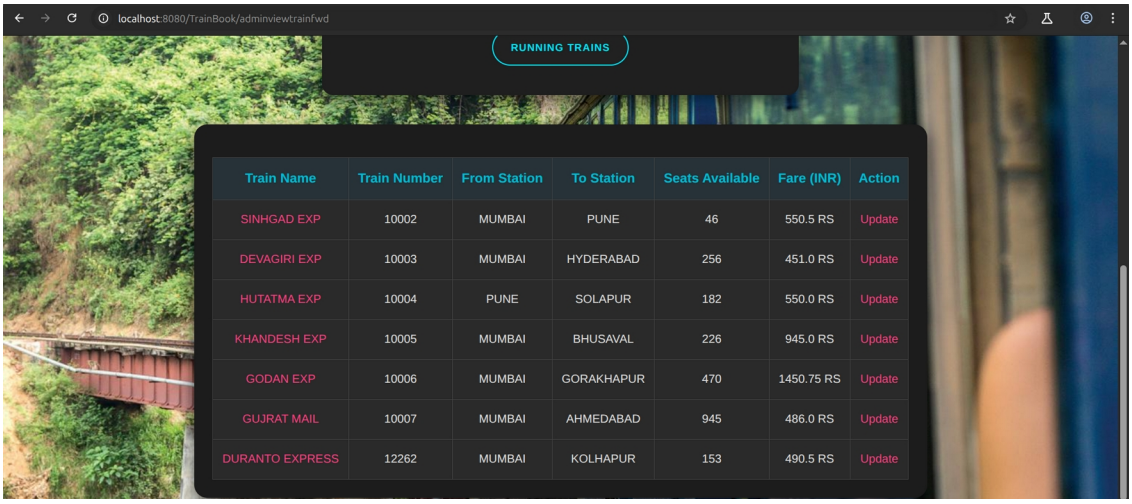


OUTPUT -

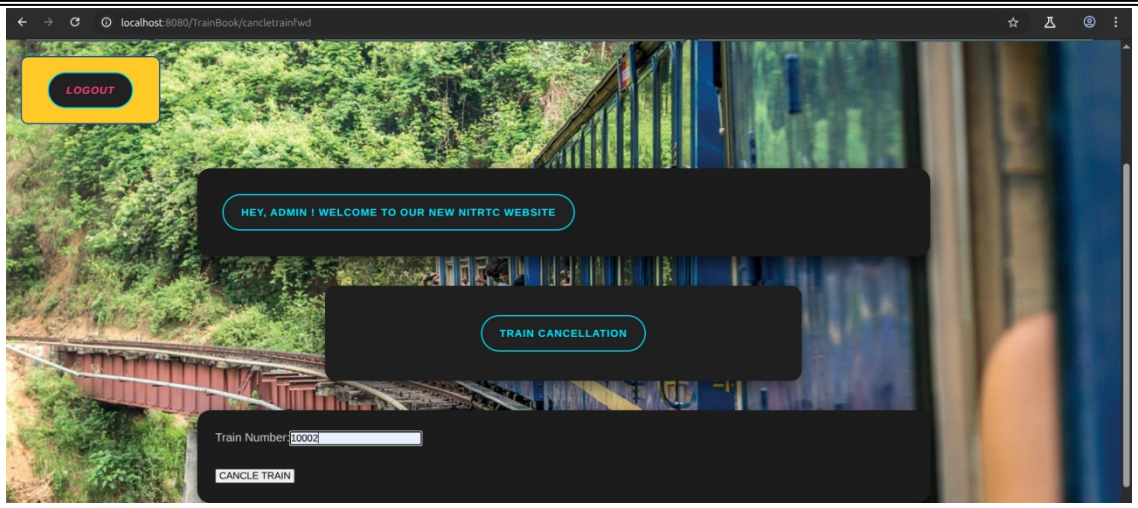


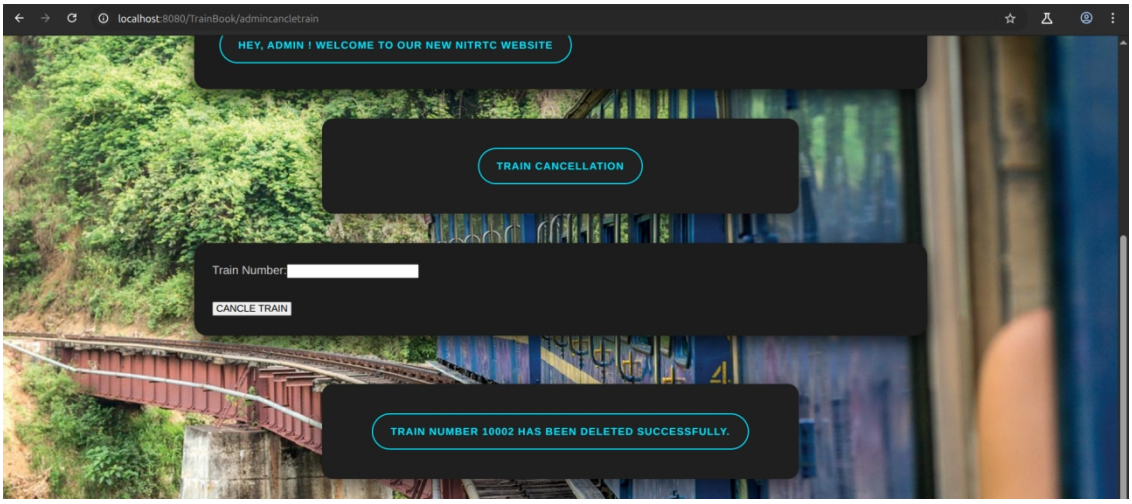
Delete Train -

Before Deleting Train -

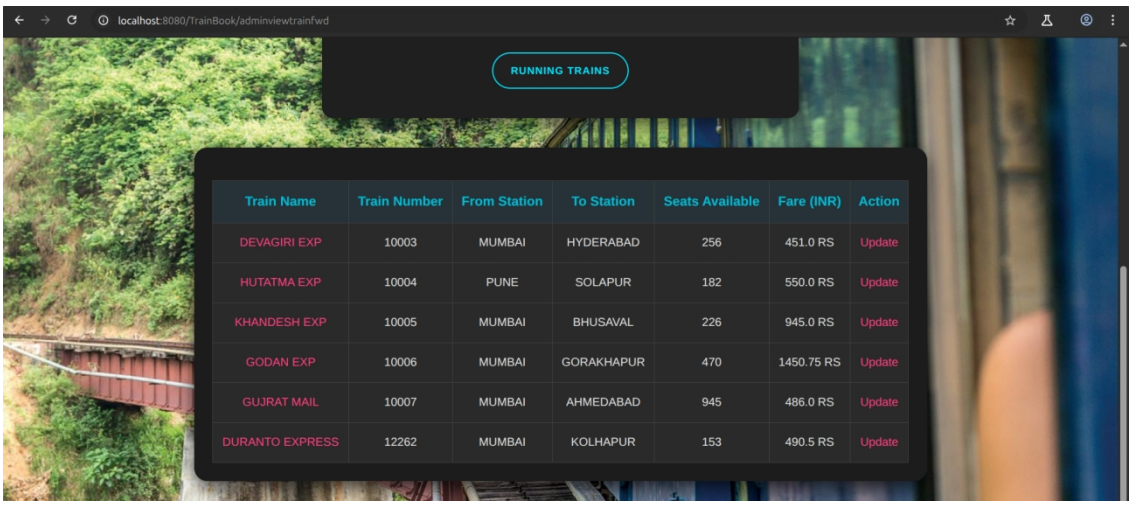


Deleting Train -



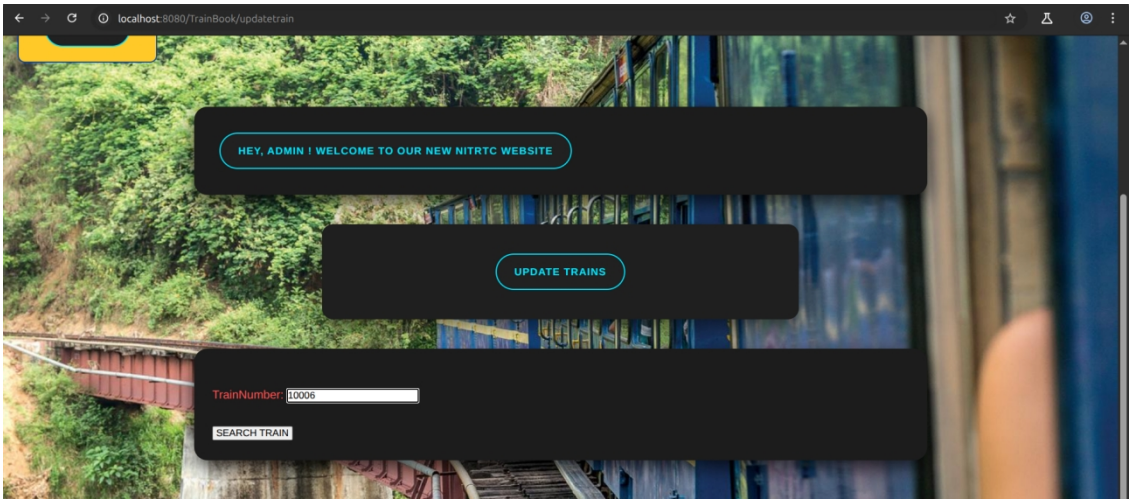


After Deleting Train -

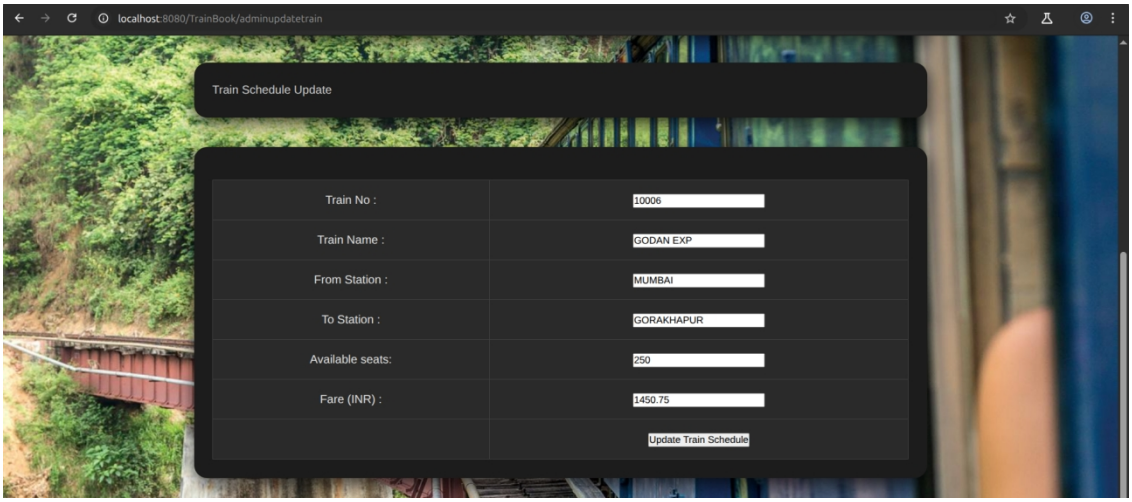


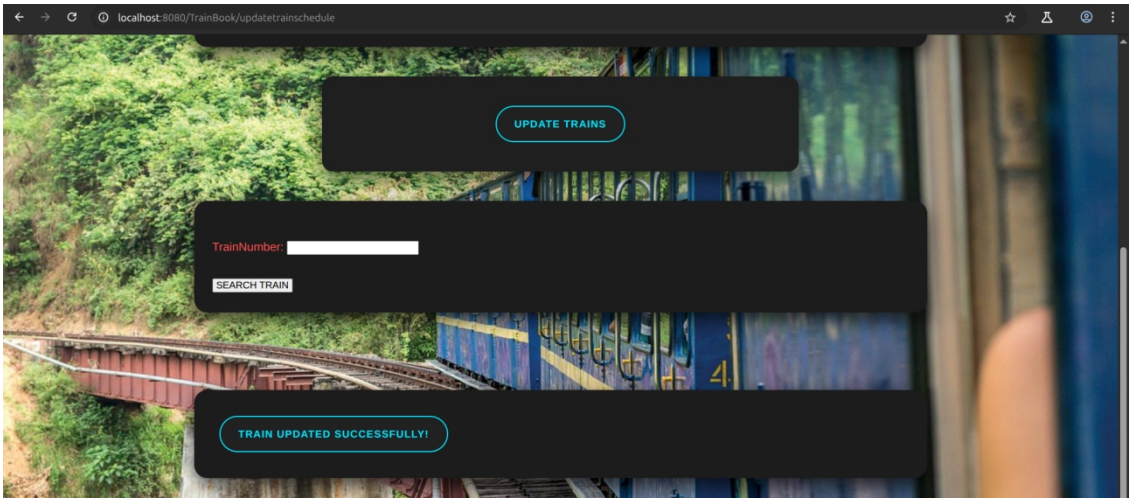
Update Train Details -

Before Train Get Update -

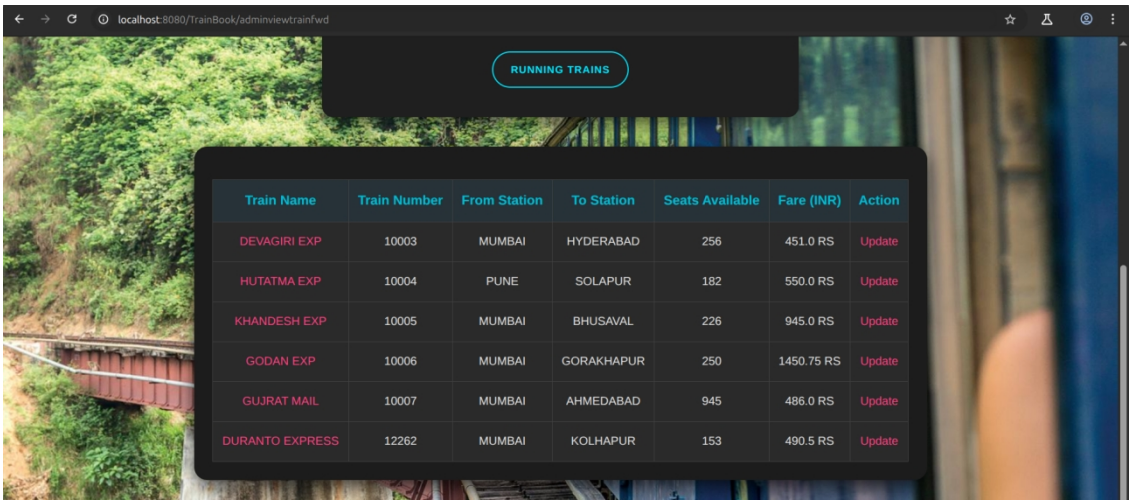


Updating Train Details -

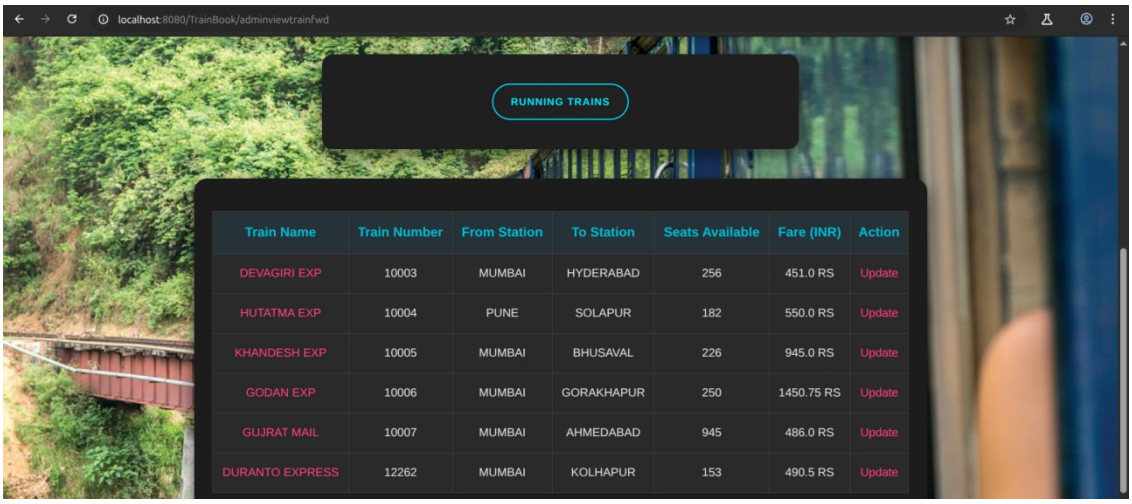




After Train Get Update -

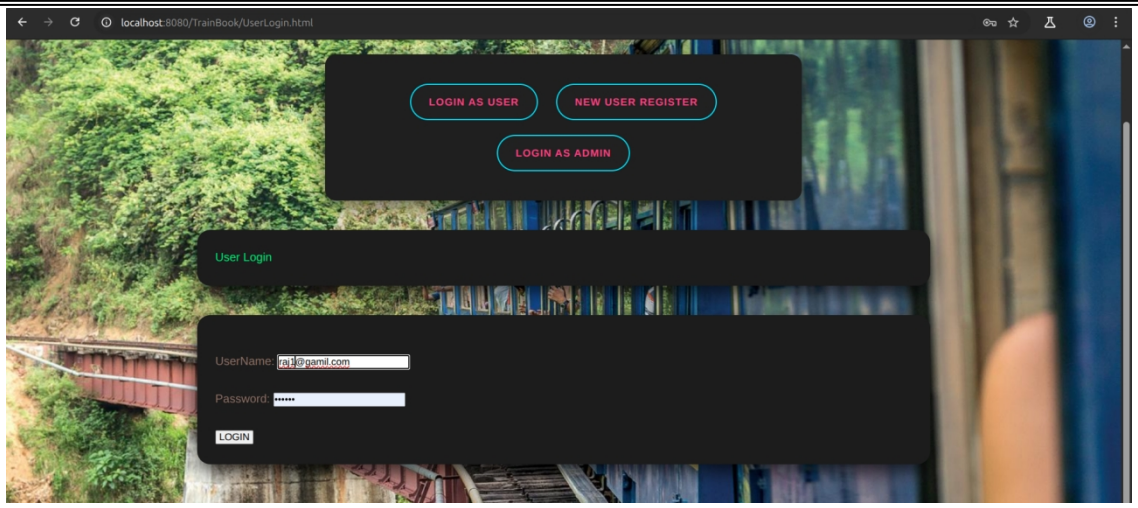


View Train Details -

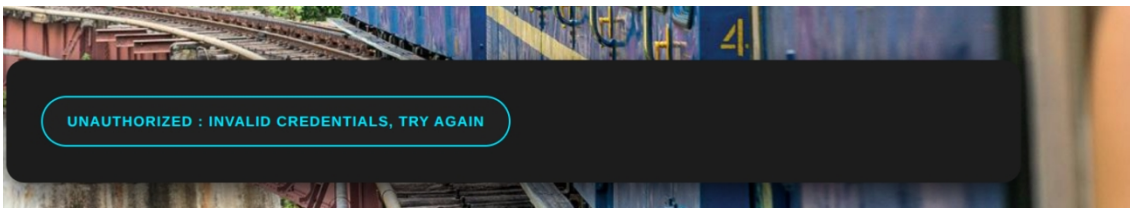


Login User with invalid Details -

If you are trying to login User with incorrect details then it will not be logged in.

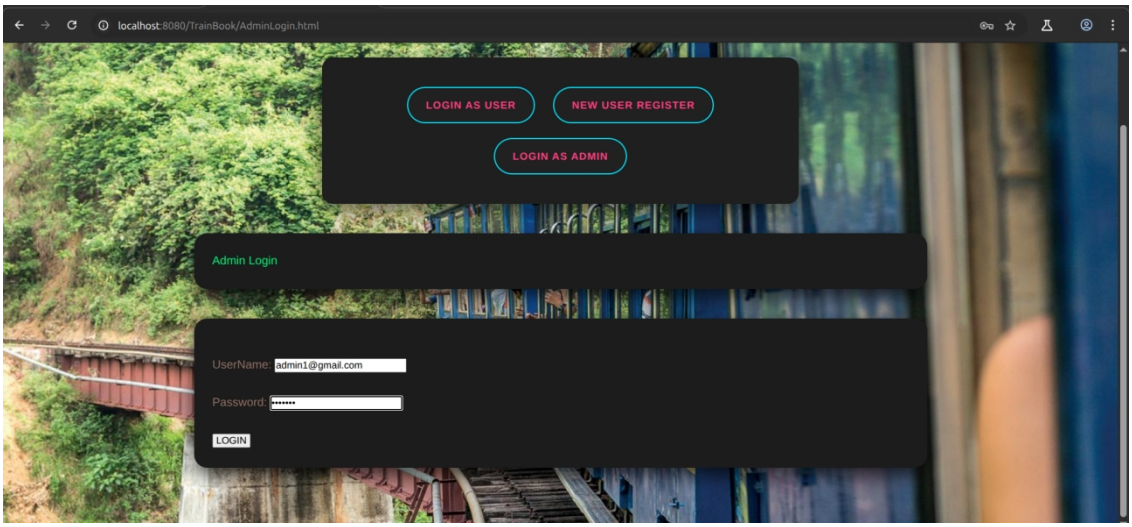


output -

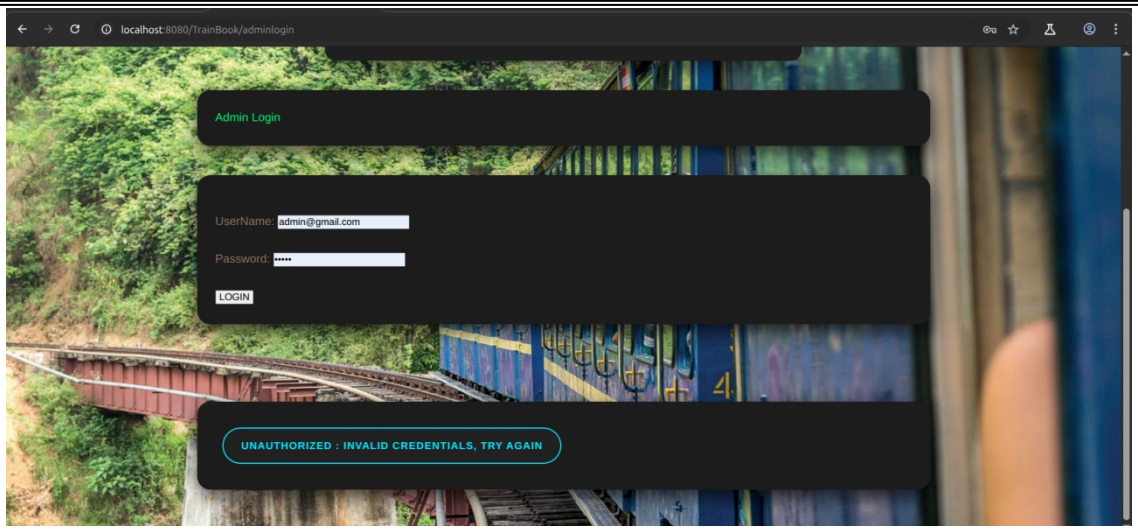


Login Admin with invalid Details -

If you are trying to login Admin with incorrect details then it will not be logged in.

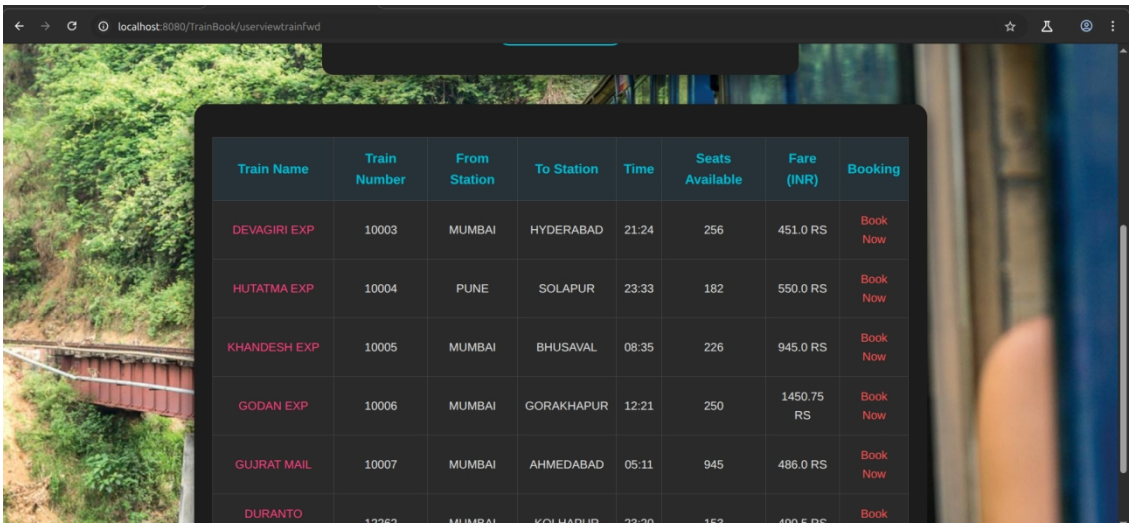


OUTPUT -



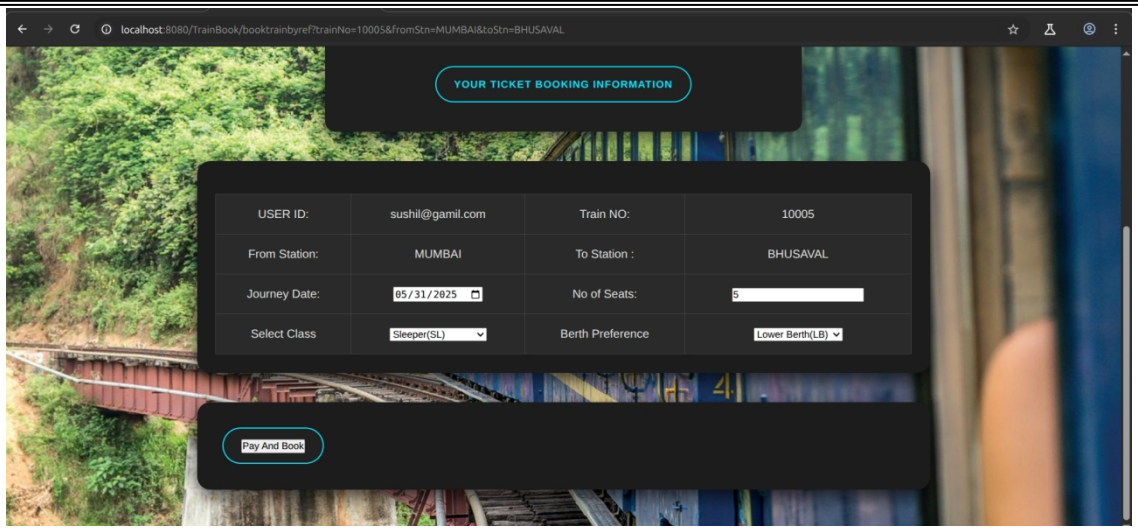
Book Ticket via Customer -

If you are logged in using customer then it will show below details for the train.

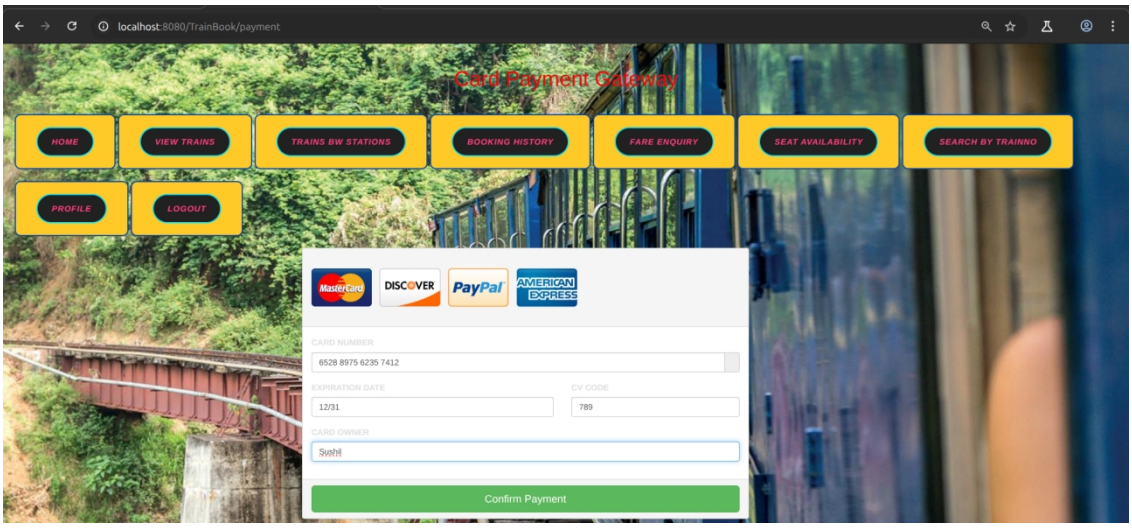


Train Name	Train Number	From Station	To Station	Time	Seats Available	Fare (INR)	Booking
DEVAGIRI EXP	10003	MUMBAI	HYDERABAD	21:24	256	451.0 RS	Book Now
HUTATMA EXP	10004	PUNE	SOLAPUR	23:33	182	550.0 RS	Book Now
KHANDESH EXP	10005	MUMBAI	BHUSAVAL	08:35	226	945.0 RS	Book Now
GODAN EXP	10006	MUMBAI	GORAKHAPUR	12:21	250	1450.75 RS	Book Now
GUJRAT MAIL	10007	MUMBAI	AHMEDABAD	05:11	945	486.0 RS	Book Now
DURANTO							Book

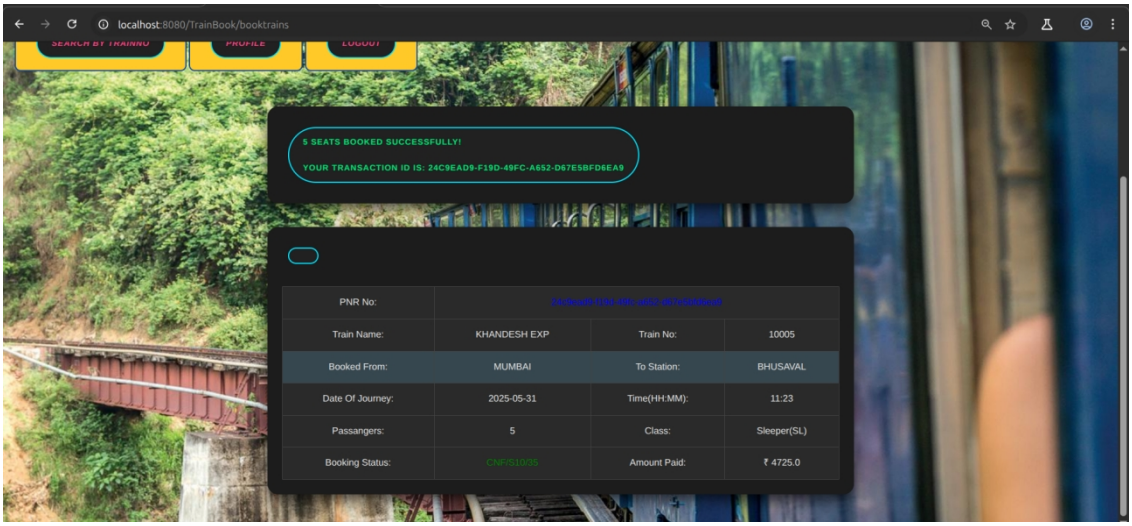
Here we will click on the Book Now for the train .



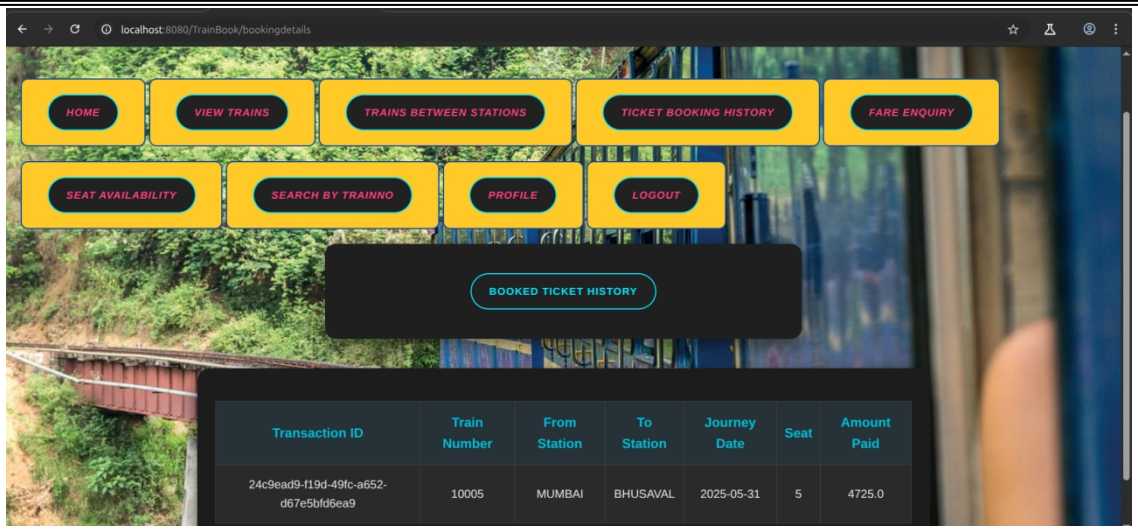
Once we click on Pay and Book then it will redirect to Payment Page.



Once Payment is done then you will redirect to confirmation page.

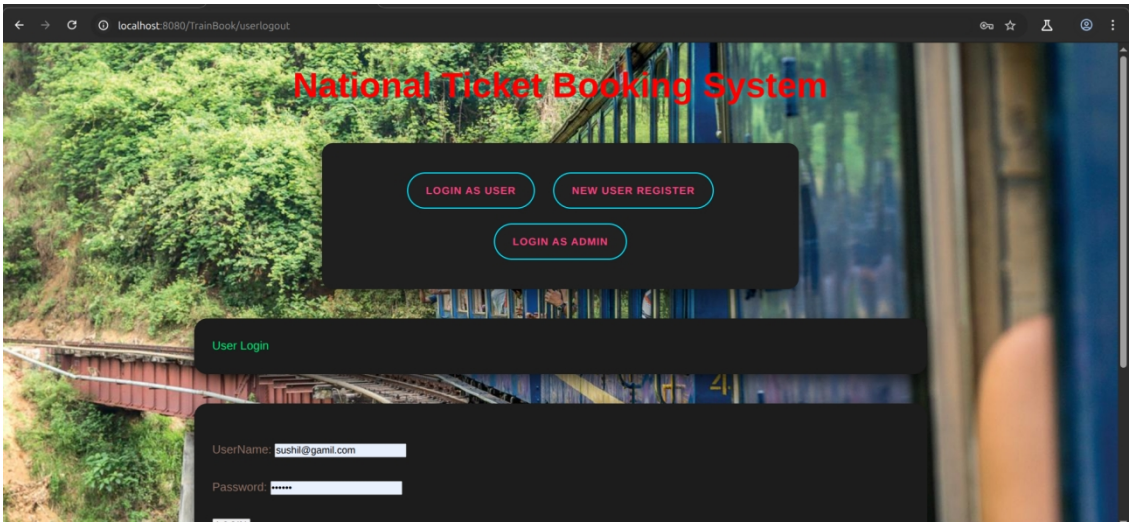


We can check the details on Ticket Booking History Page.

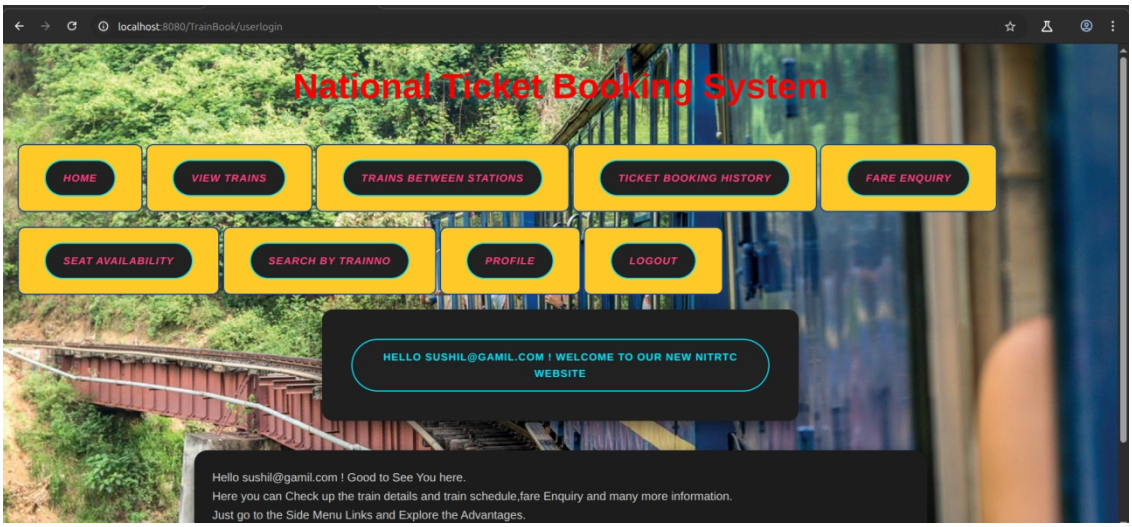

localhost:8080/TrainBook/bookingdetails
HOME
VIEW TRAINS
TRAINS BETWEEN STATIONS
TICKET BOOKING HISTORY
FARE ENQUIRY
SEAT AVAILABILITY
SEARCH BY TRAINNO
PROFILE
LOGOUT
BOOKED TICKET HISTORY
Transaction ID
Train Number
From Station
To Station
Journey Date
Seat
Amount Paid
24c9ead9-f19d-49fc-a652-d67e5bfd6ea9
10005
MUMBAI
BHUSAVAL
2025-05-31
5
4725.0

Application

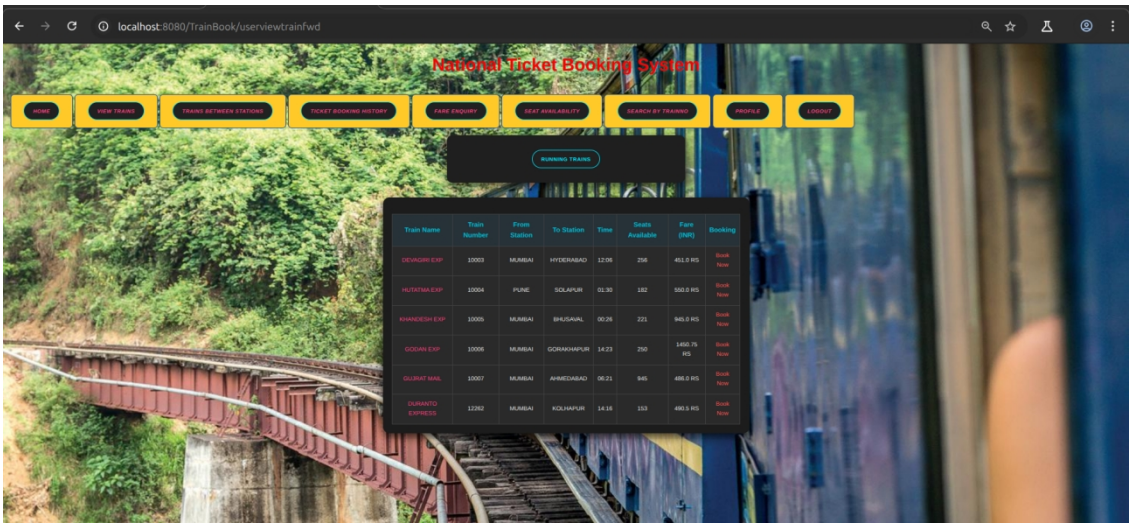
Front Page -



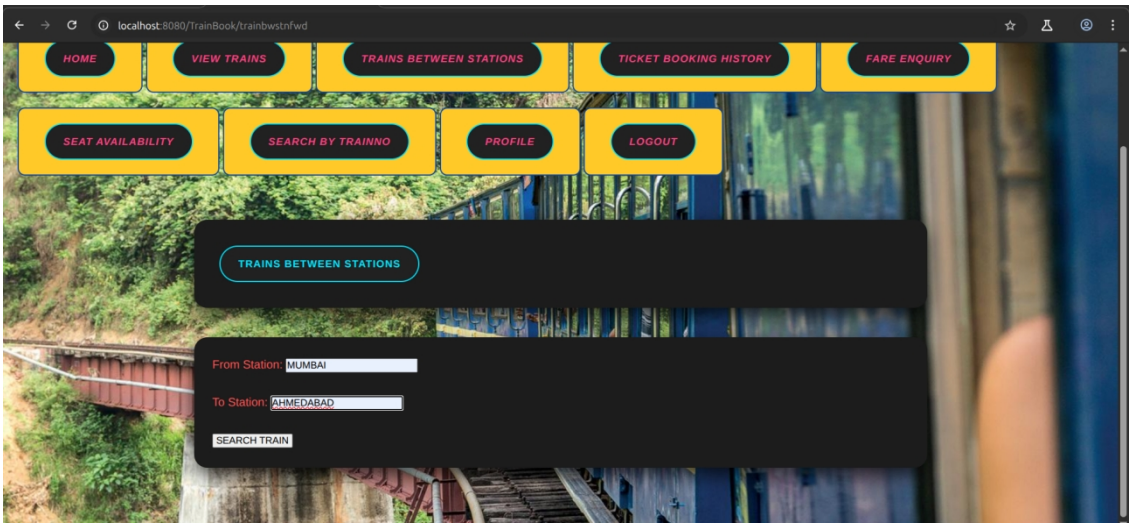
User Page -

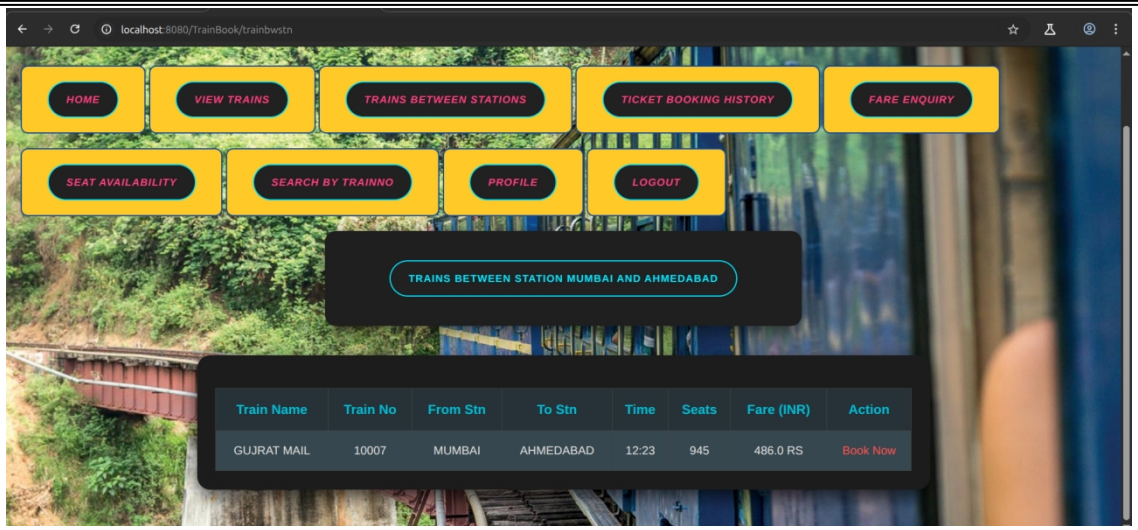


Train Details -

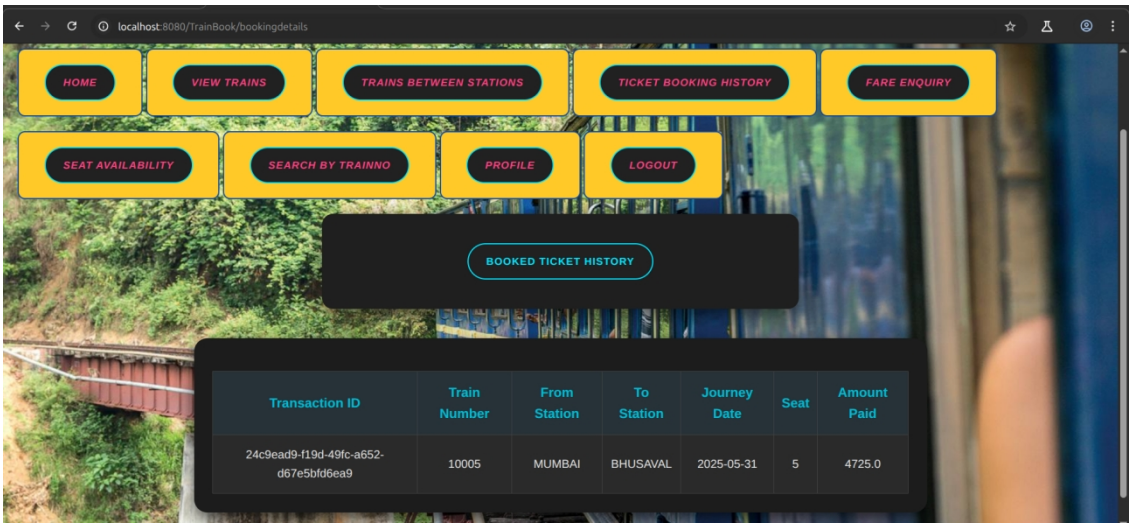


Train Between Station Details -

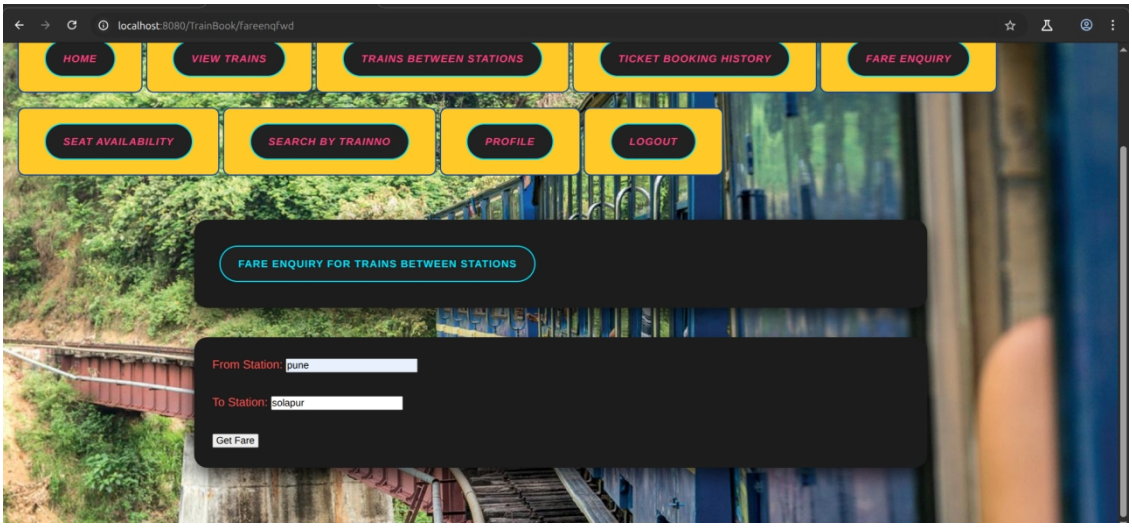


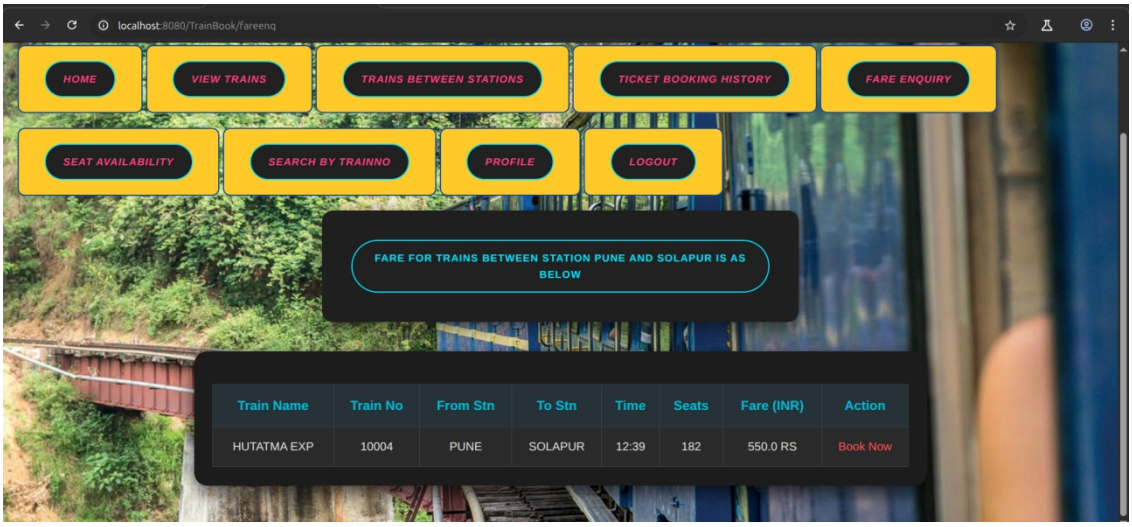


Train Booking History -

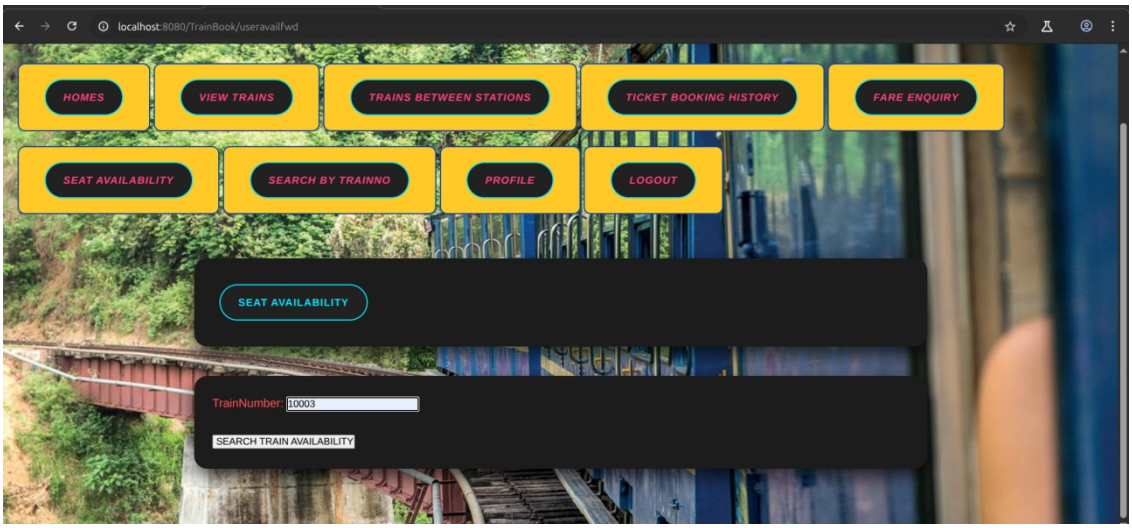


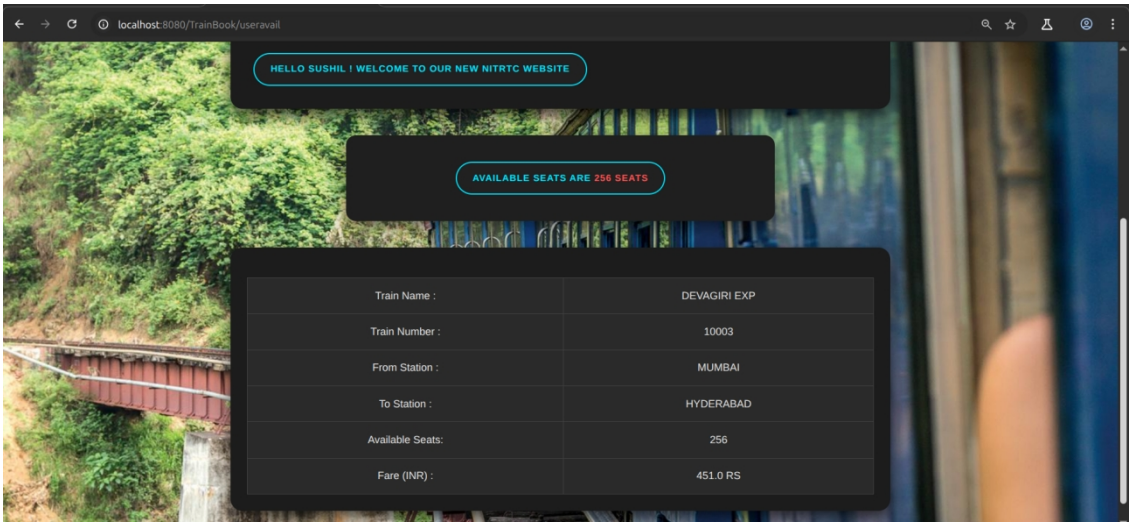
Train Fare Enquiry -



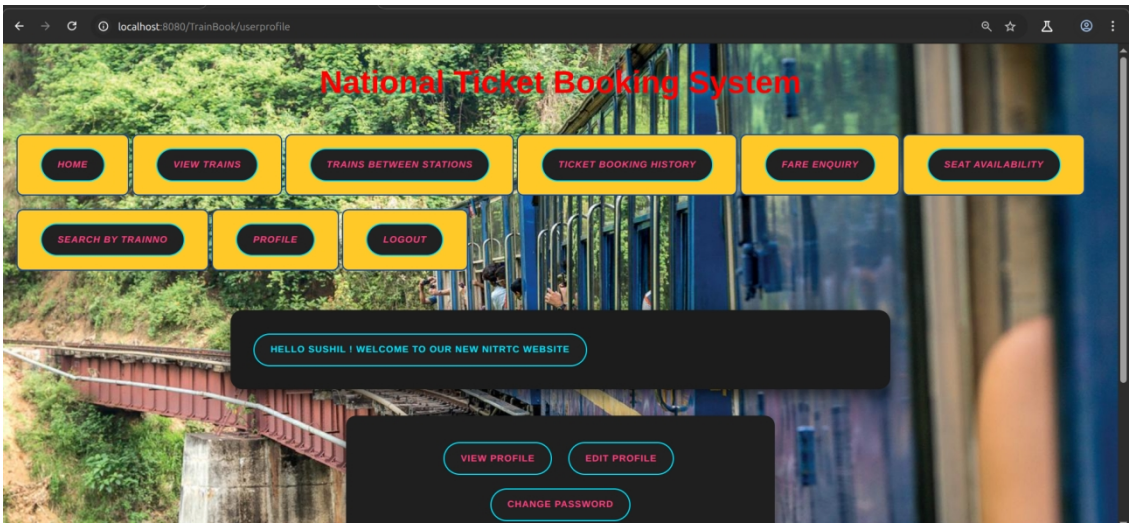


Train Seat Availability -

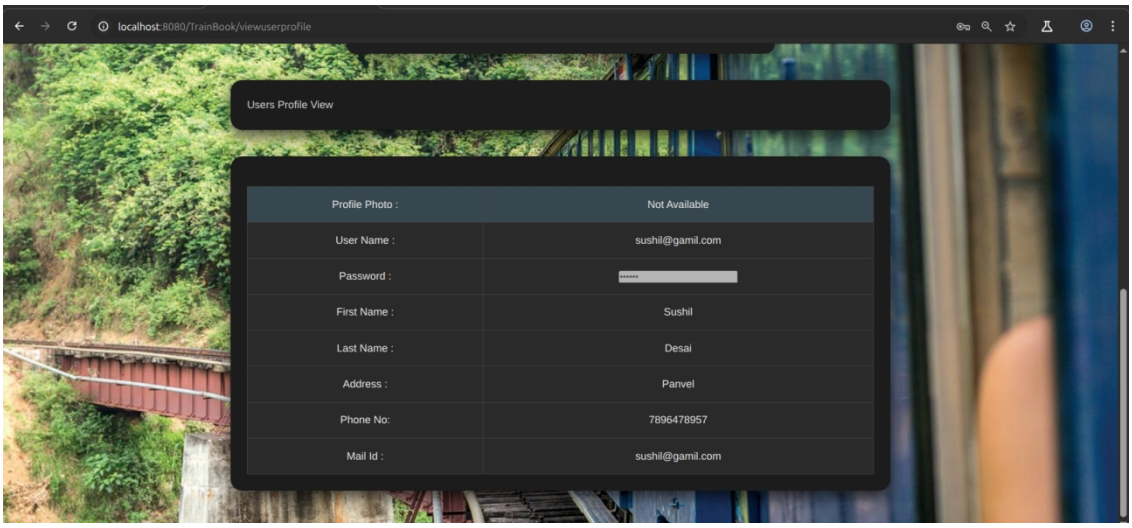




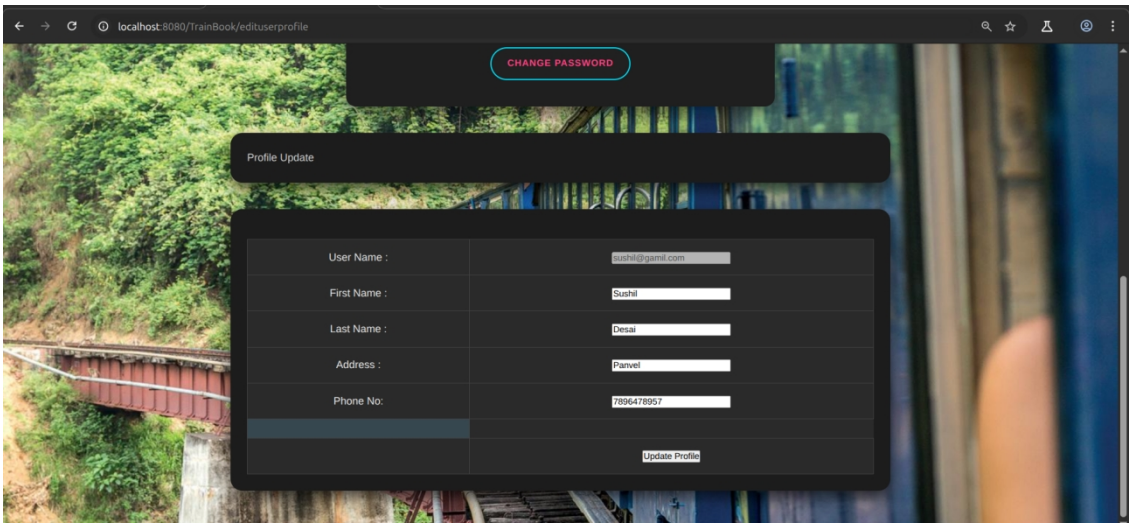
Profile Page-



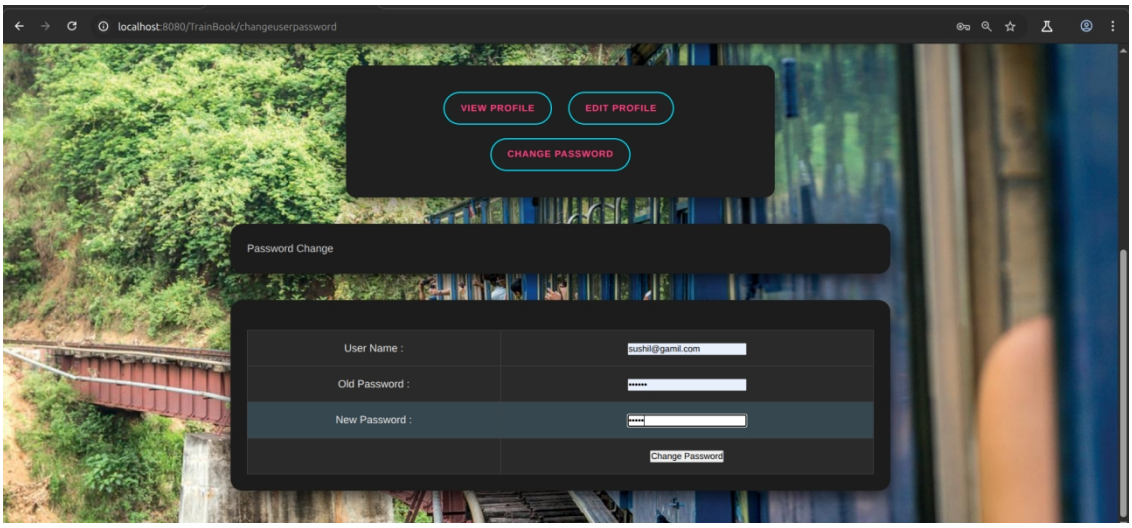
View Profile-



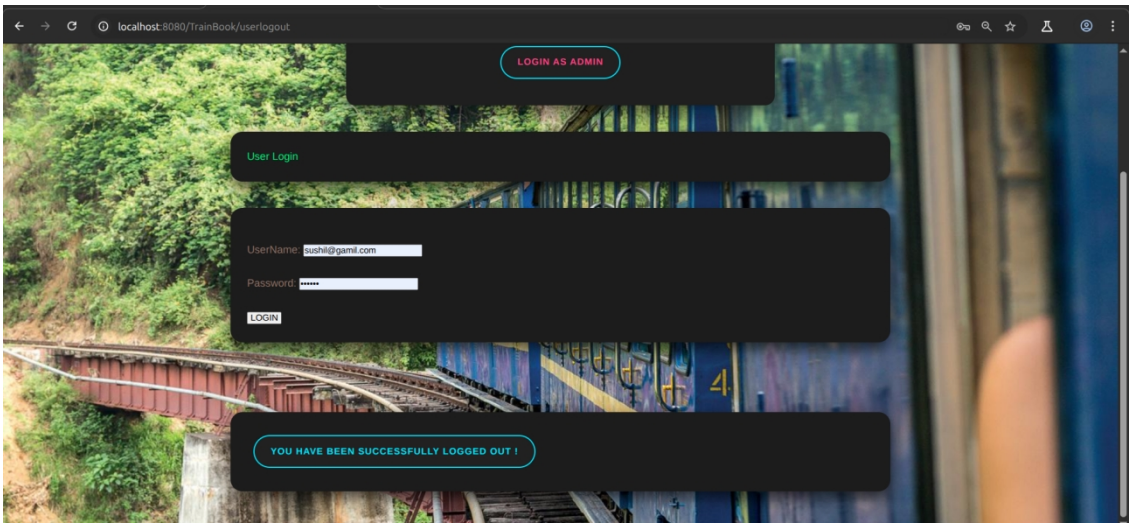
Edit Profile-



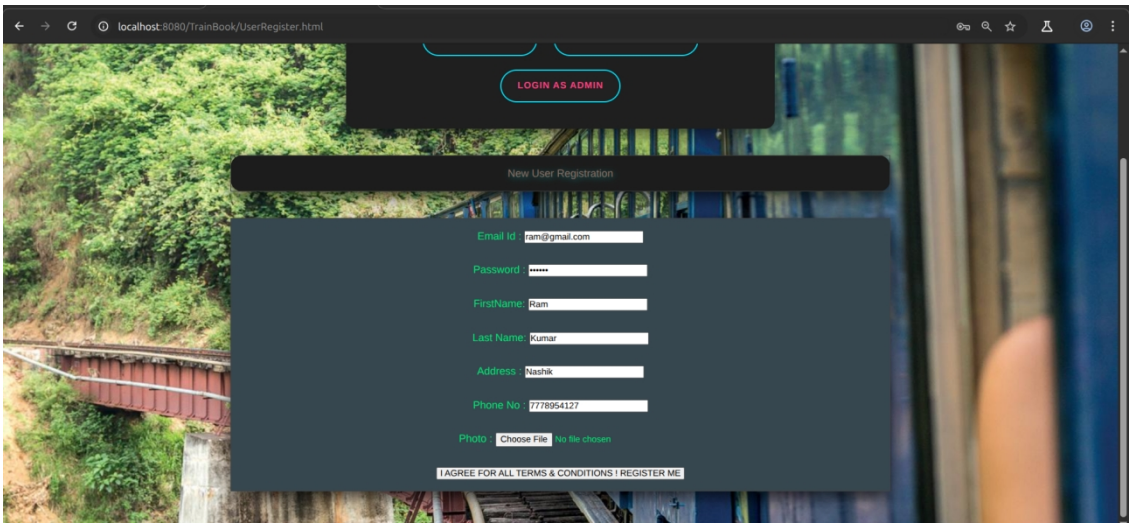
Change Password -



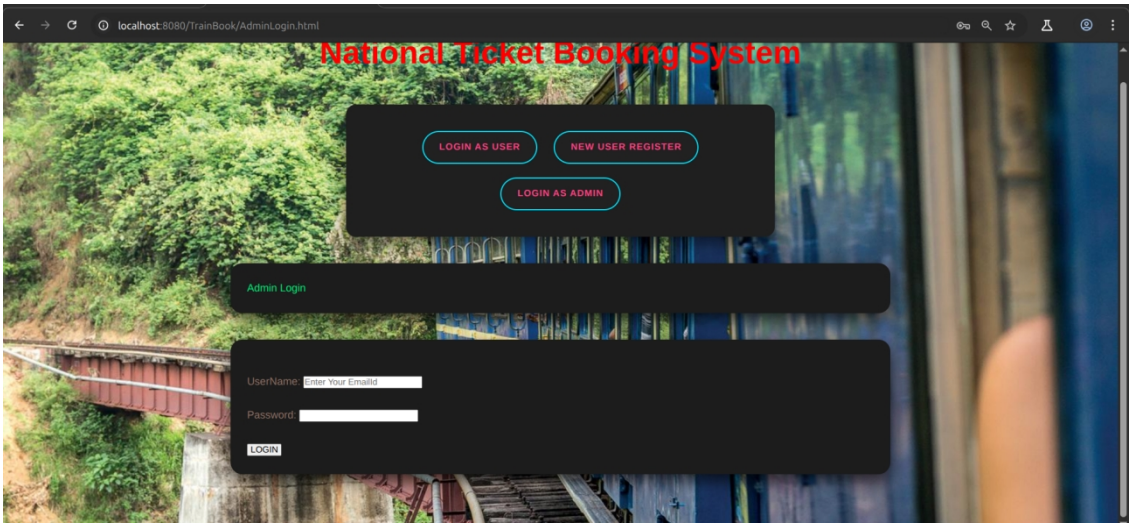
User Logout-



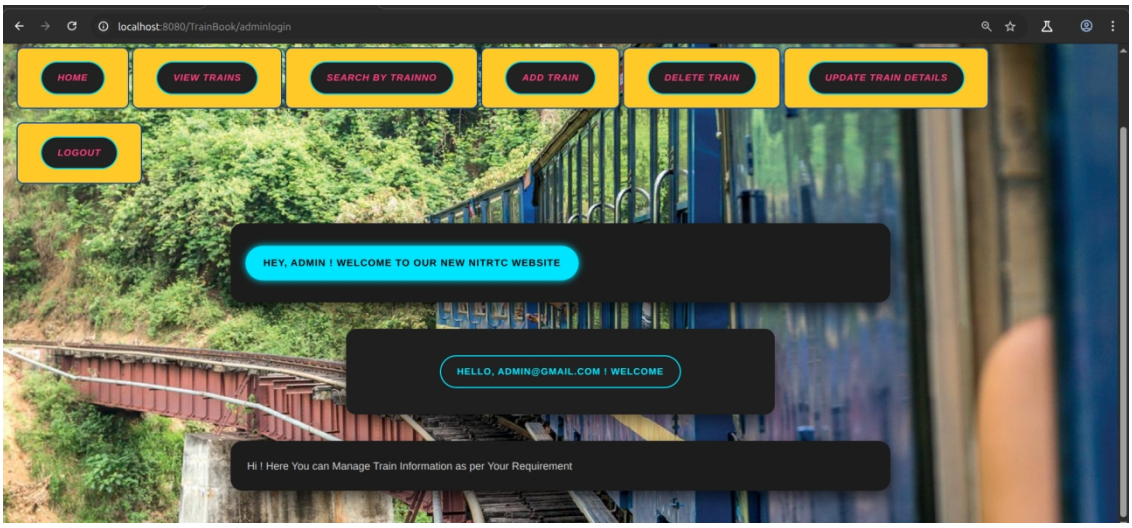
New User Registration -



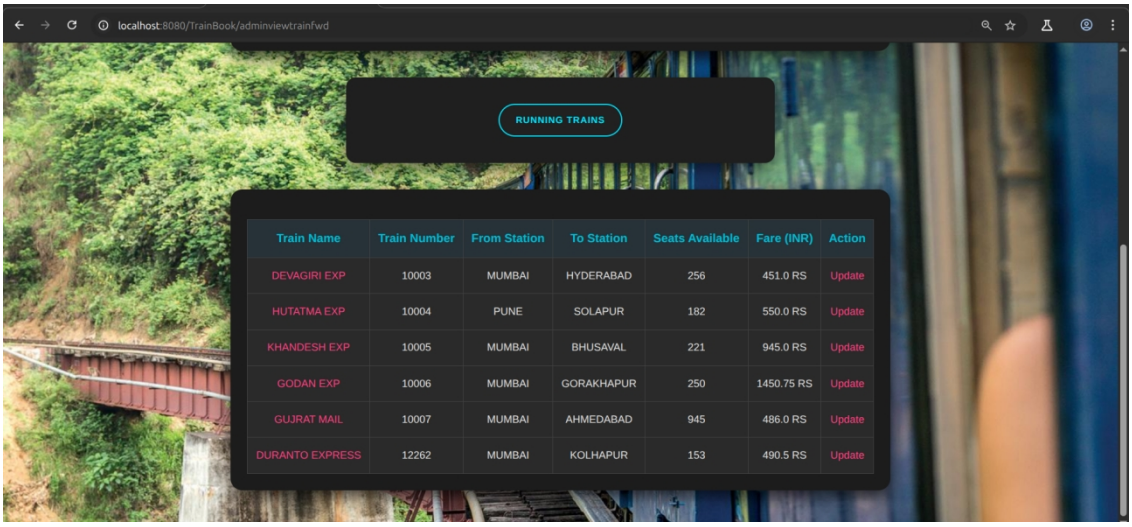
Admin Login-



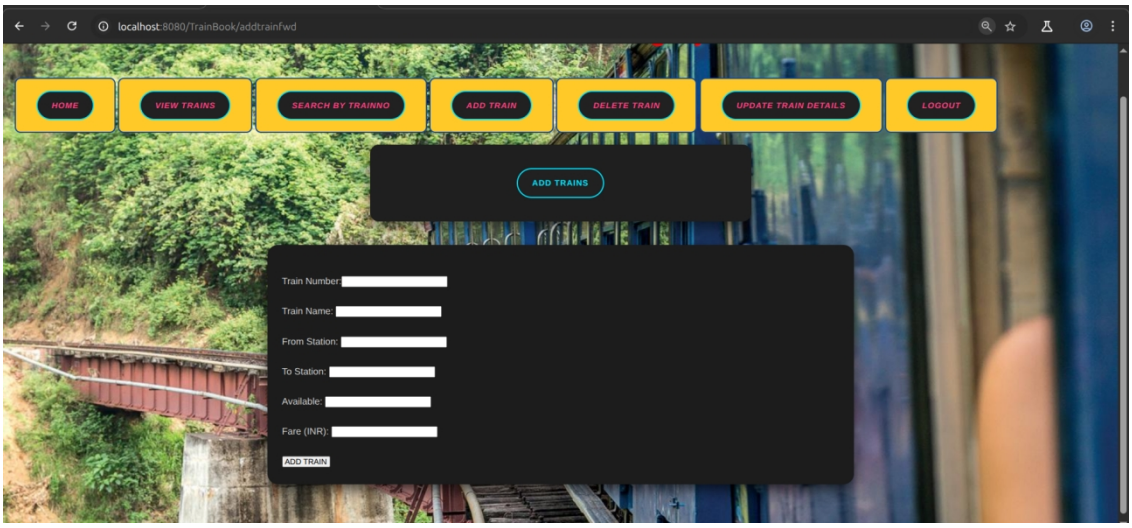
Admin Home Page -



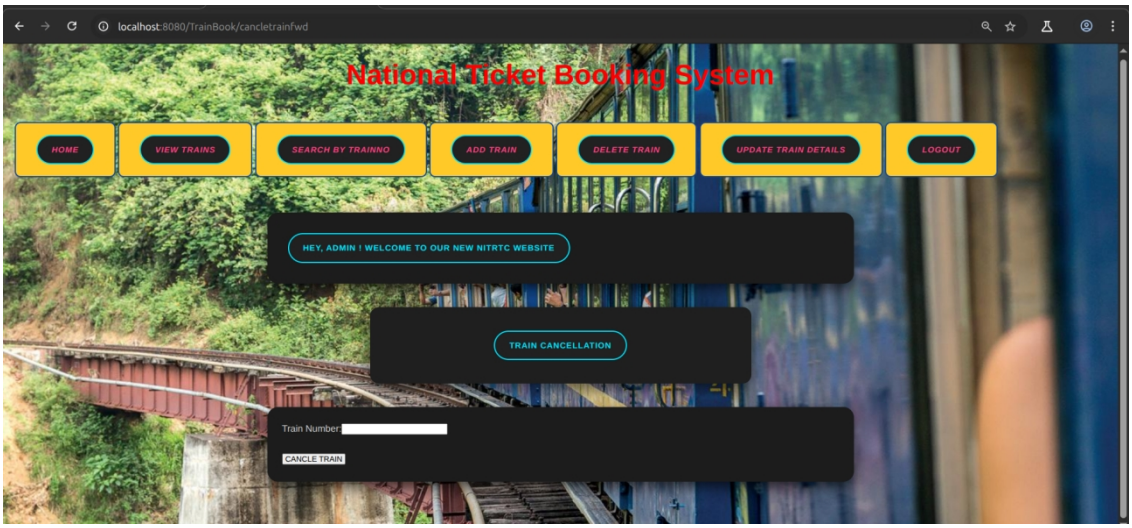
View Running Trains -



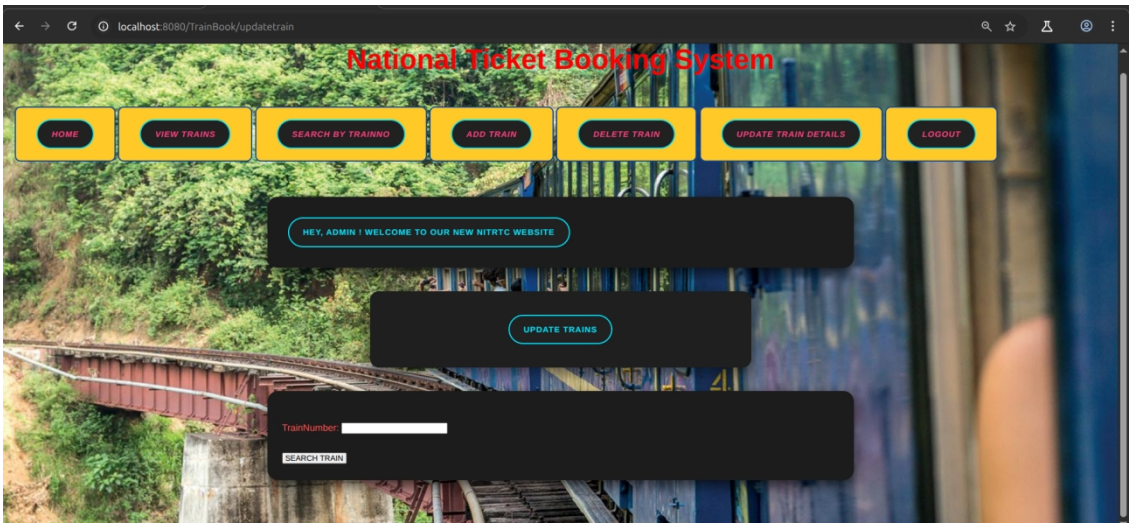
Add New Train -



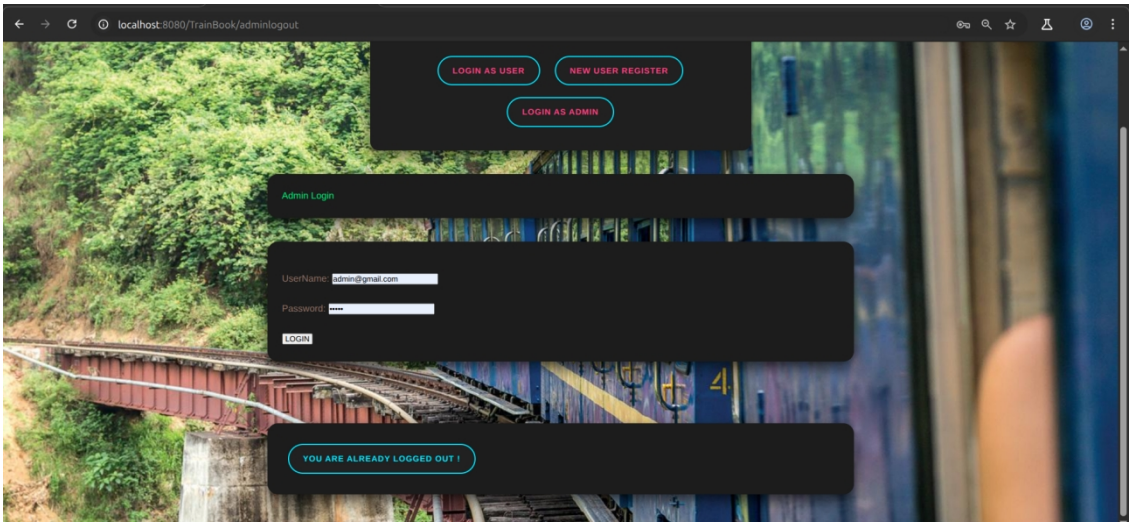
Delete Train -



Update Train -



Admin Logout -



Conclusion

The Train Ticket Booking System project, developed using **Java technologies such as Servlets**, along with **MySQL for database management**, and **HTML/CSS for the front-end interface**, successfully achieves its goal of digitizing and streamlining the railway ticket booking process. By integrating these technologies, the system offers a comprehensive, efficient, and user-friendly platform that improves the experience for both users and administrators.

From the user's perspective, the system simplifies tasks such as viewing train schedules, booking tickets, canceling reservations, and checking booking history. The intuitive HTML and CSS-based interface ensures ease of navigation, while the Java Servlet backend handles dynamic processing, session management, and request handling with reliability and speed. The use of MySQL enables secure and structured data storage, allowing for real-time access and updates to train details, seat availability, and user bookings.

For administrators, the system provides functionality to manage train records, monitor bookings, and update schedules. This reduces the reliance on manual processes, thereby minimizing errors and improving operational efficiency.

Through this project, the practical application of core concepts in **web development, database interaction, and server-side programming** has been effectively demonstrated. It showcases how Java Servlets can be used in conjunction with MySQL to create scalable and secure web applications.

Additionally, the system is designed with future enhancements in mind. Features such as online payment integration, SMS or email notifications, real-time train tracking, and mobile responsiveness could be added to further improve usability and functionality.

In conclusion, this project not only fulfills the key functional requirements of a modern train ticket booking platform but also

provides a strong technical foundation for expanding into a fully integrated transportation management system. It highlights the potential of open-source technologies in solving real-world problems and contributing to the digital transformation of essential services like public transport.

Bibliography

- Eckel, B. (2006). *Thinking in Java* (4th ed.). Prentice Hall.
- Sierra, K., & Bates, B. (2008). *Head First Java* (2nd ed.). O'Reilly Media.
- Horstmann, C. S. (2016). *Core Java Volume I—Fundamentals* (10th ed.). Prentice Hall.
- Nyberg, D. (2002). *Java Servlets and JSP* (2nd ed.). O'Reilly Media.
- Welling, L., & Thomson, L. (2008). *PHP and MySQL Web Development* (4th ed.). Addison-Wesley.
- Duckett, J. (2011). *HTML and CSS: Design and Build Websites*. Wiley.
- Oracle. (n.d.). *The Java™ Tutorials*. Oracle. https://docs.oracle.com/javase/tutorial/
- MySQL Documentation. (n.d.). *MySQL Reference Manual*. Oracle Corporation. https://dev.mysql.com/doc/
- Mozilla Developer Network (MDN). (n.d.). *HTML: HyperText Markup Language*. https://developer.mozilla.org/en-US/docs/Web/HTML
- Mozilla Developer Network (MDN). (n.d.). *CSS: Cascading Style Sheets*. https://developer.mozilla.org/en-US/docs/Web/CSS