**Railway Management System**



**Session 2023 – 2027**

**Submitted by:**

Mobeen Butt 2023-CS-28

**Supervised by:**

Prof. Awais Hassan

**Course:**

CSC-102 Programming Fundamentals

**Department of Computer Science**

**University of Engineering and Technology Lahore**

**Contents**

[1.1 Introduction 4](#_Toc151849177)

[1.2 Project Scope 4](#_Toc151849178)

[1.3 Users of Application 4](#_Toc151849179)

[1.4 Admin 5](#_Toc151849180)

[1.4.1 Login 5](#_Toc151849181)

[1.4.2 Sign Up 5](#_Toc151849182)

[1.4.3 Sign In 5](#_Toc151849183)

[1.5 Note for Login: 5](#_Toc151849184)

[1.5.1 Add New Train 5](#_Toc151849185)

[1.5.2 View Added Train 5](#_Toc151849186)

[1.5.3 Remove Added Trains 5](#_Toc151849187)

[1.5.4 Available Trains 5](#_Toc151849188)

[1.5.5 Information 6](#_Toc151849189)

[1.5.6 Ticket Information 6](#_Toc151849190)

[1.5.7 Helpline 6](#_Toc151849191)

[1.5.8 E-Ticket 6](#_Toc151849192)

[1.5.9 Logout 6](#_Toc151849193)

[1.6 User 6](#_Toc151849194)

[1.6.1 Login 7](#_Toc151849195)

[1.6.2 Sign Up 7](#_Toc151849196)

[1.6.3 Sign In 7](#_Toc151849197)

[1.7 Note for Login 7](#_Toc151849198)

[1.7.1 Home 7](#_Toc151849199)

[1.7.2 Train Timings 7](#_Toc151849200)

[1.7.3 Rates 7](#_Toc151849201)

[1.7.4 Information 7](#_Toc151849202)

[1.7.5 E-Booking 7](#_Toc151849203)

[1.7.6 Feedback and Ratings 7](#_Toc151849204)

[1.7.7 Ticket Information 8](#_Toc151849205)

[1.7.8 Helpline 8](#_Toc151849206)

[1.7.9 Logout 8](#_Toc151849207)

[1.8 Wireframes 8](#_Toc151849208)

[1.9 Functions Prototypes 9](#_Toc151849209)

[2.1 Functions Working Flow 10](#_Toc151849210)

[2.2 Complete Code of Railway Management Application 10](#_Toc151849211)

[2.3 Weakness in the Application 10](#_Toc151849212)

[2.3.1 Login Weakness 10](#_Toc151849213)

[2.3.2 Lack of Persistence 11](#_Toc151849214)

[2.3.3 Account Security 11](#_Toc151849215)

[2.3.4 Single Responsibility Function 11](#_Toc151849216)

[2.3.5 Mis-Behaviour of application 11](#_Toc151849217)

[2.3.6 Limited Access 11](#_Toc151849218)

[2.3.7 Not Access of Berth 11](#_Toc151849219)

[2.3.8 Class of Passenger 11](#_Toc151849220)

[2.4 Future Directions 11](#_Toc151849221)

# 1.1 Introduction

I am pleased to submit this proposal for the development of a Railway Management System (RMS) application. The proposed application aims to modernize and streamline Railway operations, improving efficiency, safety, and customer experience. This application will encompass various modules, including ticket booking, train scheduling, passenger management, feedback, ratings and more, to meet the diverse needs of a comprehensive Railway Management System.

# 1.2 Project Scope

* Passenger Information and Service
* Reporting and Analytics
* Customer Support and Feedback
* Mobile Application
* Online Ticket Booking
* Train Scheduling and Management

# 1.3 Users of Application

|  |  |  |
| --- | --- | --- |
| **Admin** | Add Train | Add new train. |
| Remove Train | Remove added train. |
| View Added Train | View added or remove trains. |
| Helpline | Get 24/7 help for any query. |
| Available Trains | View the available trains. |
| Information | Information about the application. |
| E-Ticket | Buy online tickets. |
| Ticket Information | Get tickets information. |
| Exit | Exit completely from application. |
| Log out | Log out from the Application. |
| **User** | Home | View the Services of our Train. |
| Train Timings | View Train timings. |
| Rates | View Rates or Fares of Trains. |
| Information | Information about the application. |
| E-Booking | Online Booking of Trains Ticket. |
| Feedback | Give Feedback to our application. |
| Ratings | Give Ratings to our application and services. |
| Ticket Information | Get tickets information. |
| Helpline | Get 24/7 help for any query. |
| Log Out | Log out from the application. |

# 1.4 Admin

Admin as administrator can add, remove, view trains also buy e-ticket.

## 1.4.1 Login

I add the login functionality. It improve the system by adding a strong login feature. This will allow users to sign in or sign up securely, giving them access to personalized features and information.

## 1.4.2 Sign Up

I add the Sign up function and introduce a user-friendly sign-up process where users can effortlessly create accounts. They will be prompted to provide essential information and preferences, ensuring a personalized experience according to their needs.

## 1.4.3 Sign In

I add the Sign In function and implement a secure sign-in process to authenticate users and grant access to their accounts. This feature enhances the security of user data.

# 1.5 Note for Login:

Login function is for Admin and Passenger. A user may be Admin or Passenger so, for Admin user must enter **Admin** and Passenger must enter **User** in the role choice.

## 1.5.1 Add New Train

Facilitate the addition of new trains to the system through a user-friendly and straightforward process. Administrators can effortlessly input essential train details such as the train name, destination, and departure time. The system ensures a smooth handling of this information, guaranteeing a hassle-free experience when integrating new trains into the Railway Management System.

## 1.5.2 View Added Train

Facilitate administrators in easily viewing added trains through a user-friendly process. This feature provides access to a comprehensive list of all system trains, highlighting essential details like train name, destination, and departure time. The interface ensures smooth monitoring and management, contributing to the system's overall efficiency by offering a clear overview of available trains. This enhancement streamlines train-related tasks for administrators.

## 1.5.3 Remove Added Trains

Admin can also remove added trains by Entering the Name of the train he/she wants to remove according to desired needs.

## 1.5.4 Available Trains

Here you check Available trains and track the train. For this purpose, we have trains name as follows:

* Pak-Train
* Yaadgar Express
* Jaffer Express
* Green Train

You can also check train for specific locations like:

* Lahore
* Karachi
* Peshawar
* Multan

## 1.5.5 Information

This function provides users with information about the railway system, historical data, and upcoming projects.

## 1.5.6 Ticket Information

It displays fare breakdowns for different routes and locations. User can get information about trains and ticket rates according to their desired journey.

## 1.5.7 Helpline

A 24/7 helpline assists users with queries, providing auto responses and connecting users with live agents for more complex issues. Users can get any kind of help about desired trains and any issue related to Railway Management System.

## 1.5.8 E-Ticket

Users can book e-tickets, selecting seats and providing preferences. The system calculates ticket prices based on factors like destination. Also display the information of passenger according to booked ticket.

## 1.5.9 Logout

User also logout from their accounts either Admin or Passenger (User) and change, create and add more accounts according to desired needs.

# 1.6 User

User may be Passenger, who get many facilities like creating account, check Train Timings, give Feedback, give Ratings, gets Information and also buy number of online tickets according to their needs.

# 1.6.1 Login

I add the login functionality. It improve the system by adding a strong login feature. This will allow users to sign in or sign up securely, giving them access to personalized features and information.

## 1.6.2 Sign Up

I add the Sign up function and introduce a user-friendly sign-up process where users can effortlessly create accounts. They will be prompted to provide essential information and preferences, ensuring a personalized experience according to their needs.

## 1.6.3 Sign In

I add the Sign In function and implement a secure sign-in process to authenticate users and grant access to their accounts. This feature enhances the security of user data.

# 1.7 Note for Login

Login function is for Admin and Passenger. A user may be Admin or Passenger so, for Admin user must enter **Admin** and Passenger must enter **User** in the role choice.

## 1.7.1 Home

Here Passenger, can check our services which we provide to our respected passengers. We provide best facility to our users.

## 1.7.2 Train Timings

Here our Passengers, check the trains timings by entering the train names like as follows:

* Pak Train
* Yaad Gar Express
* Jaffer Express
* Green Train

## 1.7.3 Rates

Passengers can check the fares and rates of trains according to Train name, location and destination. We provide reasonable fares to our users, that’s our priority.

## 1.7.4 Information

This function provides users with information about the railway system, historical data, and upcoming projects.

## 1.7.5 E-Booking

Passengers can book e-tickets, selecting seats and providing preferences. The system calculates ticket prices based on factors like destination. Also display the information of passenger according to booked ticket.

## 1.7.6 Feedback and Ratings

Passengers can give feedback to our facility and Railway Management System. It includes a ratings feature, allowing users to express their satisfaction levels. So that we can improve our facilities.

## 1.7.7 Ticket Information

Here Passenger add information of its ticket and verify ticket is available or not. Passengers provide their name, cnic, location and some more items for verification.

## 1.7.8 Helpline

A 24/7 helpline assists users with queries, providing auto responses and connecting users with live agents for more complex issues. Users can get any kind of help about desired trains and any issue related to Railway Management System.

## 1.7.9 Logout

User also logout from their accounts either Admin or Passenger (User) and change, create and add more accounts according to desired needs.

# 1.8 Wireframes

My login screen looks like that and its command line interface is displayed below:

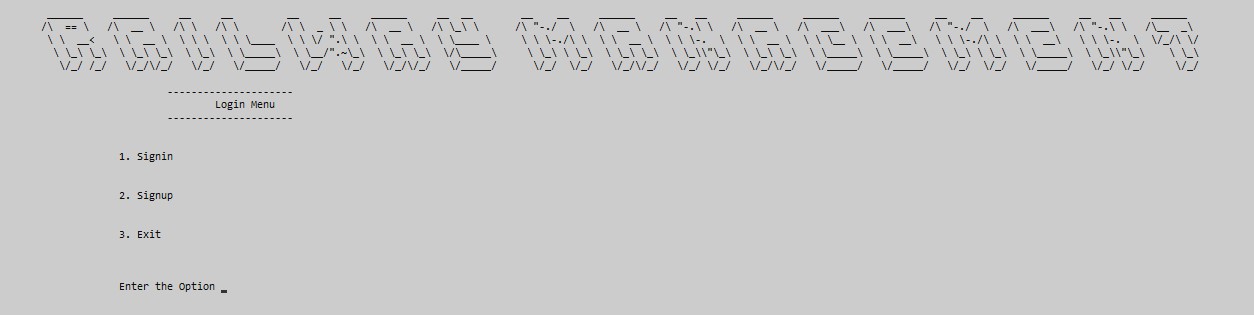
****

Figure 1: Login Screen

My Main Menu Screen is displayed as follows:

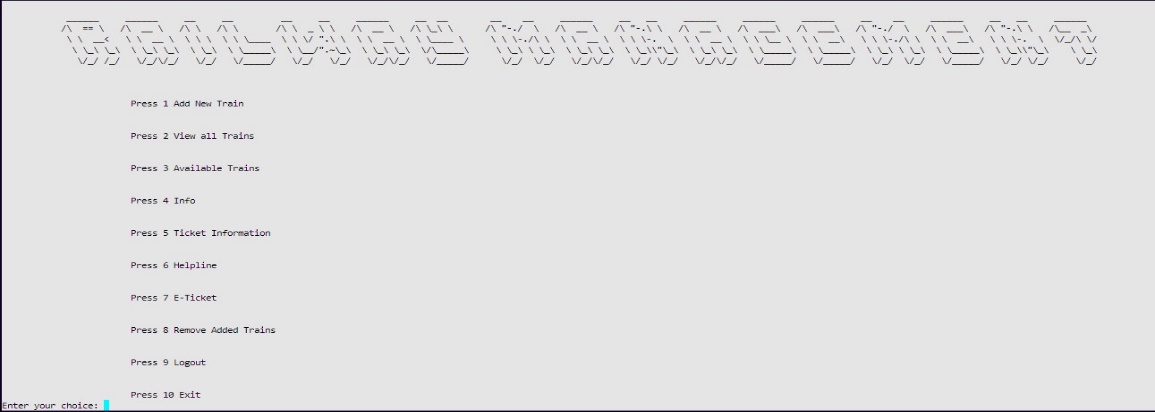


Figure 2: Admin Menu Screen

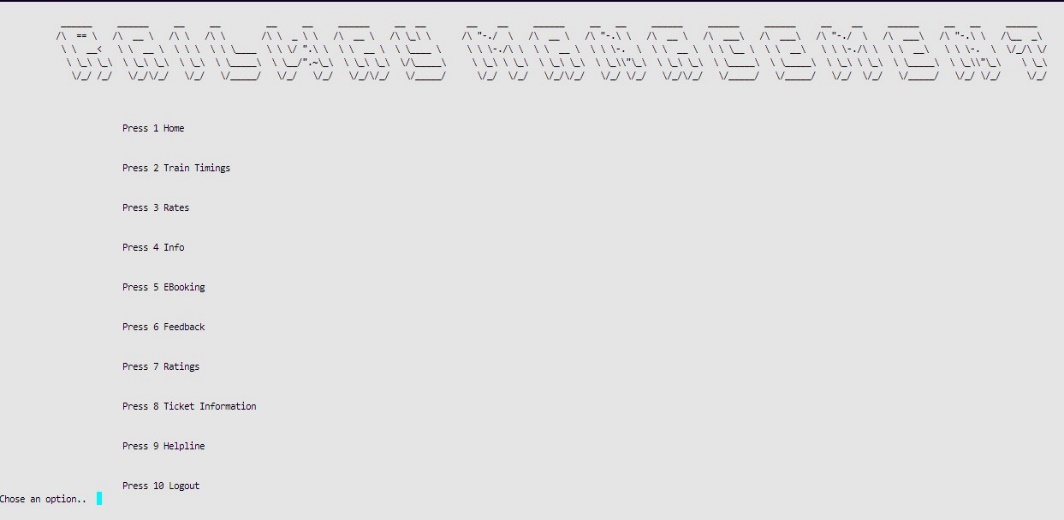
****

Figure 3: User Menu Screen

# 1.9 Functions Prototypes

My functions prototypes are displayed below:

1. void User();
2. void services();
3. void rate();
4. void dashboard();
5. void header();
6. void eticket();
7. void feedback();
8. void rating();
9. void helpline();
10. void Menu(string submenu);
11. void subMenu(string submenu);
12. void clearScreen();
13. int loginMenu();
14. void Exit();
15. void displayMenu();
16. void displayInformation(string names[], int cnics[], string locations[], int nums[], int totalAmounts[], int numTickets);
17. string Signin(string name, string password, string users[], string passwords[], string roles[], int count);
18. bool Signup(string name, string password, string role, string users[], string passwords[], string roles[], int &usersCount, int userArrSize);
19. void manageTrains(string names[], string destinations[], int departureTimes[], int &numTrains, const int MAX\_TRAINS);
20. void removeTrain(string names[], string destinations[], int departureTimes[], int &numTrains);
21. void addTrain(string names[], string destinations[], int departureTimes[], int &numTrains, const int MAX\_TRAINS);
22. void viewTrains(const string names[], const string destinations[], const int departureTimes[], int numTrains);
23. void addTicketInformation();

# 2.1 Functions Working Flow

admin

user

main ()

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Addnew** |  | **View add** |  | **Available** |  | **Info()** |  | **Ticket** |  | **Helpline()** | **E-ticket()** |  | **Remove added** |  | **Exit()** |  | **log out()** |
| **trains()** |  | **trains()** |  | **Trains()** |  |  |  | **information()** |  |  |  |  | **trains()** |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Home()** |  | **Train** |  |  |  | **Info()** |  | **e-booking()** |  | **Feedback()** |  | **Ratings()** |
|  |  | **timings()** |  | **rates()** |  |  |  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **Ticketinformation()** | **Helpline()** | **log out()** |

# 2.2 Complete Code of Railway Management Application

#include <iostream>

#include <conio.h>

#include <windows.h>

#include <iomanip>

#include <string>

using namespace std;

void User();

void services();

void rate();

void dashboard();

void header();

void eticket();

void feedback();

void rating();

void helpline();

void Menu(string submenu);

void subMenu(string submenu);

void addTicketInformation();

void clearScreen();

int loginMenu();

void Exit();

void displayMenu();

void displayInformation(string names[], int cnics[], string locations[], int nums[], int totalAmounts[], int numTickets);

string Signin(string name, string password, string users[], string passwords[], string roles[], int count);

bool Signup(string name, string password, string role, string users[], string passwords[], string roles[], int &usersCount, int userArrSize);

void manageTrains(string names[], string destinations[], int departureTimes[], int &numTrains, const int MAX\_TRAINS);

void removeTrain(string names[], string destinations[], int departureTimes[], int &numTrains);

void addTrain(string names[], string destinations[], int departureTimes[], int &numTrains, const int MAX\_TRAINS);

void viewTrains(const string names[], const string destinations[], const int departureTimes[], int numTrains);

int main()

{

const int userArrSize = 100;

string users[userArrSize];

string passwords[userArrSize];

string roles[userArrSize];

int count = 0;

int option = 0;

while (option != 3)

{

header();

Menu("Login");

option = loginMenu();

if (option == 1)

{

system("cls");

string name;

string password;

string role;

header();

Menu("Signin");

cout << "Enter your Name: ";

cin>>name;

cout << "Enter your Password: ";

cin >> password;

role = Signin(name, password, users, passwords, roles, count);

system("cls");

if (role == "Admin")

{

clearScreen();

const int MAX\_TRAINS = 10; // Maximum number of trains

header();

string names[MAX\_TRAINS];

string destinations[MAX\_TRAINS];

int departureTimes[MAX\_TRAINS];

int numTrains = 0;

int choice = 0;

while (choice != 9)

{

system("cls");

header();

cout << "\n\n\t\t\tPress 1 Add New Train" << endl;

cout << "\n\n\t\t\tPress 2 View added Trains " << endl;

cout << "\n\n\t\t\tPress 3 Available Trains" << endl;

cout << "\n\n\t\t\tPress 4 Info" << endl;

cout << "\n\n\t\t\tPress 5 Ticket Information" << endl;

cout << "\n\n\t\t\tPress 6 Helpline" << endl;

cout << "\n\n\t\t\tPress 7 E-Ticket" << endl;

cout << "\n\n\t\t\tPress 8 Remove Added Trains" << endl;

cout << "\n\n\t\t\tPress 9 Logout" << endl;

cout << "\n\n\t\t\tPress 10 Exit" << endl;

cout << "Enter your choice: ";

cin >> choice;

if (choice == 1)

{

system("cls");

manageTrains(names, destinations, departureTimes, numTrains, MAX\_TRAINS);

system("cls");

}

if (choice == 3)

{

system("cls");

dashboard();

system("cls");

}

if (choice == 4)

{

system("cls");

Sleep(800);

header();

cout << "Pakistan Railways is the national, state-owned railway company of Pakistan in Lahore. Founded in 1861 as the North Western State Railway and headquartered in Lahore, it owns 7,490 kilometres of operational track across Pakistan, stretching from Torkham to Karachi, offering both freight and passenger services." << endl;

clearScreen();

}

if (choice == 5)

{

system("cls");

rate();

system("cls");

}

if (choice == 6)

{

system("cls");

helpline();

system("cls");

}

if (choice == 7)

{

system("cls");

eticket();

system("cls");

}

if (choice == 8)

{

system("cls");

removeTrain(names, destinations, departureTimes, numTrains);

system("cls");

}

if (choice == 2)

{

system("cls");

viewTrains(names, destinations, departureTimes, numTrains);

clearScreen();

}

if (choice == 9)

{

system("cls");

clearScreen();

system("cls");

Menu("Login");

system("cls");

}

if (choice == 10)

{

system("cls");

Exit();

}

else

{

system("cls");

cout << "Invalid choice. Please try again.\n";

system("cls");

}

}

}

else if (role == "User")

{

clearScreen();

User();

}

else if (role == "undefined")

{

system("cls");

cout << "UnIdentifed! " << endl;

}

}

else if (option == 2)

{

system("cls");

string name;

string password;

string role;

header();

Menu("Signup");

cout << "\t\t\t\t\tEnter your Name (Spaces Not Allowed): ";

cin>>name;

cout << "\t\t\t\t\tEnter your Password: ";

cin >> password;

cout << "\t\t\t\t\tEnter your Role (Admin or User): ";

cin >> role;

bool isValid = Signup(name, password, role, users, passwords, roles, count, userArrSize);

if (isValid)

{

cout << "\t\t\tCongratulation Signup Succeessfully as " << role << endl;

}

if (!isValid)

{

cout << "\t\t\tInvalid ! Username or Password" << endl;

}

}

clearScreen();

}

return 0;

}

void header()

{

system("Color 0A");

cout << R"(

\_\_\_\_\_\_ \_\_\_\_\_\_ \_\_ \_\_ \_\_ \_\_ \_\_\_\_\_\_ \_\_ \_\_ \_\_ \_\_ \_\_\_\_\_\_ \_\_ \_\_ \_\_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_\_ \_\_ \_\_ \_\_\_\_\_\_ \_\_ \_\_ \_\_\_\_\_\_

/\ == \ /\ \_\_ \ /\ \ /\ \ /\ \ \_ \ \ /\ \_\_ \ /\ \\_\ \ /\ "-./ \ /\ \_\_ \ /\ "-.\ \ /\ \_\_ \ /\ \_\_\_\ /\ \_\_\_\ /\ "-./ \ /\ \_\_\_\ /\ "-.\ \ /\\_\_ \_\

\ \ \_\_< \ \ \_\_ \ \ \ \ \ \ \\_\_\_\_ \ \ \/ ".\ \ \ \ \_\_ \ \ \\_\_\_\_ \ \ \ \-./\ \ \ \ \_\_ \ \ \ \-. \ \ \ \_\_ \ \ \ \\_\_ \ \ \ \_\_\ \ \ \-./\ \ \ \ \_\_\ \ \ \-. \ \/\_/\ \/

\ \\_\ \\_\ \ \\_\ \\_\ \ \\_\ \ \\_\_\_\_\_\ \ \\_\_/".~\\_\ \ \\_\ \\_\ \/\\_\_\_\_\_\ \ \\_\ \ \\_\ \ \\_\ \\_\ \ \\_\\"\\_\ \ \\_\ \\_\ \ \\_\_\_\_\_\ \ \\_\_\_\_\_\ \ \\_\ \ \\_\ \ \\_\_\_\_\_\ \ \\_\\"\\_\ \ \\_\

\/\_/ /\_/ \/\_/\/\_/ \/\_/ \/\_\_\_\_\_/ \/\_/ \/\_/ \/\_/\/\_/ \/\_\_\_\_\_/ \/\_/ \/\_/ \/\_/\/\_/ \/\_/ \/\_/ \/\_/\/\_/ \/\_\_\_\_\_/ \/\_\_\_\_\_/ \/\_/ \/\_/ \/\_\_\_\_\_/ \/\_/ \/\_/ \/\_/

)";

}

void addTrain(string names[], string destinations[], int departureTimes[], int &numTrains, const int MAX\_TRAINS)

{

header();

if (numTrains < MAX\_TRAINS)

{

cout << "\nEnter Train Details:\n";

cout << "Train Name: ";

cin.ignore(); // Ignore newline character in the input buffer

getline(cin, names[numTrains]);

cout << "Destination: ";

getline(cin, destinations[numTrains]);

cout << "Departure Time (24-hour format, e.g., 1400): ";

cin >> departureTimes[numTrains];

cout << "Train added successfully!\n";

numTrains++;

}

else

{

cout << "Maximum number of trains reached. Cannot add more trains.\n";

}

}

void viewTrains(const string names[], const string destinations[], const int departureTimes[], int numTrains)

{

header();

if (numTrains > 0)

{

cout << "\n===== List of Added Trains =====\n";

cout << setw(15) << "Train Name" << setw(15) << "Destination" << setw(20) << "Departure Time\n";

for (int i = 0; i < numTrains; ++i)

{

cout << setw(15) << names[i] << setw(15) << destinations[i] << setw(20) << departureTimes[i] << "\n";

}

getch();

}

else

{

cout << "No trains added yet.\n";

}

}

void eticket()

{

system("cls");

const int MAX\_PASSENGERS = 10;

header();

int numTickets;

cout << "Enter number of tickets: ";

cin >> numTickets;

string names[MAX\_PASSENGERS];

int cnics[MAX\_PASSENGERS];

string locations[MAX\_PASSENGERS];

int nums[MAX\_PASSENGERS];

int totalAmounts[MAX\_PASSENGERS];

cout << "\t\t\t\t\t\tPASSENGERS DETAILS" << endl;

for (int i = 0; i < numTickets; ++i)

{

cout << "Enter Name for Passenger " << i + 1 << ": ";

cin >> names[i];

cout << "Enter CNIC for Passenger " << i + 1 << ": ";

cin >> cnics[i];

bool validLocation = false;

while (!validLocation)

{

cout << "Enter Location for Passenger " << i + 1 << ": ";

cin >> locations[i];

if (locations[i] != "Lahore" && locations[i] != "Gujranwala" && locations[i] != "Peshawar" && locations[i] != "Karachi" && locations[i] != "Multan")

{

cout << "Invalid Location. Please enter a valid location (Lahore, Gujranwala, Peshawar, Karachi, Multan).\n";

}

else

{

validLocation = true;

}

}

nums[i] = 1;

if (locations[i] == "Lahore")

{

totalAmounts[i] = nums[i] \* 4500;

}

else if (locations[i] == "Karachi")

{

totalAmounts[i] = nums[i] \* 4200;

}

else if (locations[i] == "Peshawar")

{

totalAmounts[i] = nums[i] \* 4500;

}

else if (locations[i] == "Multan")

{

totalAmounts[i] = nums[i] \* 4000;

}

else if (locations[i] == "Gujranwala")

{

totalAmounts[i] = nums[i] \* 4500;

}

}

displayInformation(names, cnics, locations, nums, totalAmounts, numTickets);

clearScreen();

}

void displayInformation(string names[], int cnics[], string locations[], int nums[], int totalAmounts[], int numTickets)

{

clearScreen();

header();

cout << "\n\n\n\n\t\t\t\t\t\tTicket Details\n\n\n\n"

<< endl;

for (int i = 0; i < numTickets; ++i)

{

cout << "Passenger " << i + 1 << " Name: " << names[i] << endl;

cout << "Passenger " << i + 1 << " CNIC: " << cnics[i] << endl;

cout << "Passenger " << i + 1 << " Location: " << locations[i] << endl;

cout << "Passenger " << i + 1 << " Number of Tickets: " << nums[i] << endl;

cout << "Passenger " << i + 1 << " Total Amount: " << totalAmounts[i] << endl;

cout << "-----------------------------------------" << endl;

}

}

void User()

{

header();

int option;

cout << "\n\n\t\t\tPress 1 Home" << endl;

cout << "\n\n\t\t\tPress 2 Train Timings" << endl;

cout << "\n\n\t\t\tPress 3 Rates" << endl;

cout << "\n\n\t\t\tPress 4 Info" << endl;

cout << "\n\n\t\t\tPress 5 EBooking" << endl;

cout << "\n\n\t\t\tPress 6 Feedback" << endl;

cout << "\n\n\t\t\tPress 7 Ratings" << endl;

cout << "\n\n\t\t\tPress 8 Ticket Information" << endl;

cout << "\n\n\t\t\tPress 9 Helpline" << endl;

cout << "\n\n\t\t\tPress 10 Logout" << endl;

cout << "Chose an option.. ";

cin >> option;

system("cls");

if (option == 4)

{

header();

cout << "Pakistan Railways is the national, state-owned railway company of Pakistan in Lahore. Founded in 1861 as the North Western State Railway and headquartered in Lahore, it owns 7,490 kilometres of operational track across Pakistan, stretching from Torkham to Karachi, offering both freight and passenger services." << endl;

cout << endl;

clearScreen();

User();

}

if (option == 10)

{

system("cls");

}

if (option == 3)

{

rate();

User();

}

if (option == 1)

{

services();

User();

}

if (option == 2)

{

dashboard();

User();

}

if (option == 5)

{

eticket();

User();

}

if (option == 7)

{

rating();

User();

}

if (option == 8)

{

system("cls");

addTicketInformation();

clearScreen();

User();

}

if (option == 6)

{

feedback();

User();

}

if (option == 9)

{

helpline();

User();

}

}

void helpline()

{

clearScreen();

header();

cout << "Call us at following numbers: " << endl;

cout << "xxxx-xxx-xxxx" << endl;

cout << "0000-000-0000\n";

clearScreen();

}

void feedback()

{

cin.ignore();

string res;

cout << "Enter Your Feedback about our Services: ";

getline(cin, res);

cout << "Thanks for your feedback!" << endl;

Sleep(300);

clearScreen();

}

void rating()

{

header();

cout << "\t\t\t\tGive Your Response" << endl;

cout << "1. Good \n";

cout << "2. Average\n";

cout << "3. Worst\n";

int op;

cout << "Chose any option.. ";

cin >> op;

if (op > 3)

{

cout << "Invalid choice.Chose other option!";

}

else

cout << "Thanks for your opinion! " << endl;

Sleep(300);

clearScreen();

}

void dashboard()

{

int op;

cout << "\n\n\t\t\tPress 1 Enter Train:" << endl;

cout << "\n\n\t\t\tPress 2 Location" << endl;

cout << "\n\n\t\t\tPress 3 Back" << endl;

cin >> op;

if (op == 1)

{

string name;

cout << "\n\n\t\t\tName: ";

cin >> name;

if (name == "JAFFAR EXPRESS")

{

cout << "\n\n\t\t\tTrain Arrive At 10AM";

}

if (name == "YAAD-GAR-EXPRESS")

{

cout << "\n\n\t\t\tTrain is on time. ";

}

if (name == "PAK-TRAIN")

{

cout << "\n\n\t\t\tTrain is late due to Rain!";

}

if (name == "GREEN TRAIN")

{

cout << "\n\n\t\t\tTrain arrives at 2AM";

}

else

{

cout << "Invalid Name";

}

}

if (op == 2)

{

string location;

cout << "\n\n\t\t\tEnter Location: ";

cin >> location;

if (location == "Lahore")

{

cout << "\n\n\t\t\tJAFFAR EXPRESS is coming soon! \n";

Sleep(800);

system("cls");

dashboard();

}

if (location == "Karachi")

{

cout << "\n\n\t\t\tYAAD-GAR-EXPRESS is on its Way!\n";

Sleep(800);

system("cls");

dashboard();

}

if (location == "Peshawar")

{

cout << "\n\n\t\t\tGREEN TRAIN arrives soon! \n";

Sleep(800);

system("cls");

dashboard();

}

if (location == "Multan")

{

cout << "\n\n\t\t\tTrain is OUT OF ORDER! \n";

Sleep(800);

system("cls");

dashboard();

}

}

cout << endl;

clearScreen();

}

void services()

{

cout << "\n\n\t\t\tOur Services:";

cout << "\n\n\t\t\tJAFFAR EXPRESS";

cout << "\n\n\t\t\tYAAD-GAR-EXPRESS";

cout << "\n\n\t\t\tGREEN TRAIN";

cout << "\n\n\t\t\tPAK-TRAIN \n\n\n\n";

clearScreen();

}

void rate()

{

cout << setw(70) << setfill('-') << "" << setfill(' ') << endl;

cout << left << setw(30) << "City" << setw(30) << "Train"

<< "Ticket Rate" << endl;

cout << setw(30) << "Lahore-to-Karachi" << setw(30) << "JAFFAR EXPRESS"

<< "Rs 4500" << endl;

cout << setw(30) << "Lahore-to-Multan" << setw(30) << "PAK-TRAIN"

<< "Rs 4500" << endl;

cout << setw(30) << "Karachi-to-Multan" << setw(30) << "YAAD-GAR-EXPRESS"

<< "Rs 4200" << endl;

cout << setw(30) << "Karachi-to-Lahore" << setw(30) << "JAFFAR EXPRESS"

<< "Rs 4200" << endl;

cout << setw(30) << "Multan-to-Lahore" << setw(30) << "GREEN TRAIN"

<< "Rs 4000" << endl;

cout << setw(30) << "Peshawar-to-Karachi" << setw(30) << "JAFFAR EXPRESS"

<< "Rs 4500" << endl;

cout << setw(30) << "Peshawar-to-Lahore" << setw(30) << "YAAD-GAR-EXPRESS"

<< "Rs 4500" << endl;

cout << setw(30) << "Gujranwala-to-Lahore" << setw(30) << "PAK-TRAIN"

<< "Rs 4500" << endl;

clearScreen();

}

int loginMenu()

{

int option;

cout << "\n\n\t\t\t1. Signin" << endl;

cout << "\n\n\t\t\t2. Signup" << endl;

cout << "\n\n\t\t\t3. Exit" << endl;

cout << endl;

cout << "\n\n\t\t\tEnter the Option ";

cin >> option;

if (option > 3)

{

cout << "Invaild Option \n";

}

return option;

}

string Signin(string name, string password, string users[], string passwords[], string roles[], int count)

{

for (int i = 0; i < count; i++)

{

if (users[i] == name && passwords[i] == password)

{

return roles[i];

}

}

return "undefined";

}

bool Signup(string name, string password, string role, string users[], string passwords[], string roles[], int &count, int userArrSize)

{

bool isPresent = false;

for (int i = 0; i < count; i++)

{

if (users[i] == name && passwords[i] == password)

{

isPresent = true;

break;

}

}

if (isPresent == true)

{

return 0;

}

else if (count < userArrSize)

{

users[count] = name;

passwords[count] = password;

roles[count] = role;

count++;

return true;

}

else

{

return false;

}

}

void Menu(string submenu)

{

string message = submenu + " Menu";

cout << "\t\t\t\t---------------------" << endl;

cout << "\t\t\t\t\t" << message << endl;

cout << "\t\t\t\t---------------------" << endl;

}

void subMenu(string submenu)

{

string message = "Main Menu > " + submenu;

cout << message << endl;

cout << "---------------------" << endl;

}

void clearScreen()

{

cout << "Press Any Key to Continue.. " << endl;

getch();

system("cls");

}

void Exit()

{

system("cls");

while (true)

{

break;

}

exit(0);

}

void displayMenu()

{

header();

cout << "1. Add a Train\n";

cout << "2. View Added Trains\n";

}

void manageTrains(string names[], string destinations[], int departureTimes[], int &numTrains, const int MAX\_TRAINS)

{

addTrain(names, destinations, departureTimes, numTrains, MAX\_TRAINS);

}

void removeTrain(string names[], string destinations[], int departureTimes[], int &numTrains)

{

header();

if (numTrains > 0)

{

viewTrains(names, destinations, departureTimes, numTrains); // Display the list of trains for reference

string trainName;

cout << "\nEnter the name of the train to remove: ";

cin.ignore(); // Ignore newline character in the input buffer

getline(cin, trainName);

bool found = false;

for (int i = 0; i < numTrains; ++i)

{

if (names[i] == trainName)

{

// Shift elements to remove the train

for (int j = i; j < numTrains - 1; ++j)

{

names[j] = names[j + 1];

destinations[j] = destinations[j + 1];

departureTimes[j] = departureTimes[j + 1];

}

// Decrement the number of trains

--numTrains;

cout << "Train '" << trainName << "' removed successfully!\n";

found = true;

break;

}

}

if (!found)

{

cout << "Train '" << trainName << "' not found. No changes made.\n";

}

}

else

{

cout << "No trains to remove.\n";

}

clearScreen();

}

void addTicketInformation()

{

const int MAX\_PASSENGERS = 10;

header();

int numTickets;

cout << "Enter number of tickets: ";

cin >> numTickets;

string names[MAX\_PASSENGERS];

int cnics[MAX\_PASSENGERS];

string locations[MAX\_PASSENGERS];

int nums[MAX\_PASSENGERS];

int totalAmounts[MAX\_PASSENGERS];

cout << "\t\t\t\t\t\tPASSENGERS DETAILS" << endl;

for (int i = 0; i < numTickets; ++i)

{

cout << "Enter Name for Passenger " << i + 1 << ": ";

cin >> names[i];

cout << "Enter CNIC for Passenger " << i + 1 << ": ";

cin >> cnics[i];

cout << "Enter Location for Passenger " << i + 1 << ": ";

cin >> locations[i];

nums[i] = 1; // Assuming each passenger gets one ticket by default

if (locations[i] == "Lahore")

{

totalAmounts[i] = nums[i] \* 4500;

}

else if (locations[i] == "Karachi")

{

totalAmounts[i] = nums[i] \* 4200;

}

else if (locations[i] == "Peshawar")

{

totalAmounts[i] = nums[i] \* 4500;

}

else if (locations[i] == "Multan")

{

totalAmounts[i] = nums[i] \* 4000;

}

else if (locations[i] == "Gujranwala")

{

totalAmounts[i] = nums[i] \* 4500;

}

}

displayInformation(names, cnics, locations, nums, totalAmounts, numTickets);

}

# 2.3 Weakness in the Application

There exist some major weakness in my application explain below:

## 2.3.1 Login Weakness

There is Login weakness as anyone can get access to Admin as administration and change many operations and leads to failure of Railway Management System (RMS).

## 2.3.2 Lack of Persistence

User and train data are not persisted after the program terminates. There exist no mechanism in application to save data to a file or a database so that it can be retrieved when the program restarts.

## 2.3.3 Account Security

No secure practices for handling sensitive information, such as passwords. Consider encrypting passwords and use secure methods for user authentication.

## 2.3.4 Single Responsibility Function

In my application there exist some functions which are not following the Principle of Single Responsibility Function. It can lead to various issues and make the code harder to understand, maintain and extend.

## 2.3.5 Mis-Behaviour of application

Sometimes my application misbehaves and don’t know what is it doing.

## 2.3.6 Limited Access

Access is very limited to users. They only have limited access to some functions.

## 2.3.7 Not Access of Berth

Access of Berth is not available, this leads passengers not buy a single seat.

## 2.3.8 Class of Passenger

Class or Type of train is not specified. Elite, A.C, or other classes of passengers were not specified.

# 2.4 Future Directions

Make application using a graphical user interface (GUI) library to create a more visually appealing and user-friendly interface. This can enhance the overall user experience

Provide an offline mode for users to view their booked tickets, train schedules, and other relevant information even when they are not connected to the internet.

Make Single Responsible functions to easily understand the code.

Also persist the data in application to save data to a file or a database so that it can be retrieved when the program restarts.

Class and Berth is added in future.

Security of account is increased and ensured in future. This has a major role in application.