PROJECT REPORT

Data Structures And Algorithms (CSL 221)

BS(CS)-3A

BAHRIA UNIVERSITY KARACHI CAMPUS

Department of Computer Science

Group Members

Name	Enrollment
Shalal Bakhtiar	02-134212-021
Arham Tanveer Mallick	02-134212-028
Mania Imam	02-134212-013

Submitted to:

Ma’am Saba Imtiaz

TABLE OF CONTENT

1. Introduction:

1.1) Project Description:

In this project of Railway Booking Management System, we will be implementing the ability to reserve and modify railway tickets. We will use Linked List Operations, Searching & Sorting Algorithms to modify and print the data according to user's needs. This project is simple to use and very user-friendly. It has minimal hardware and software requirements, and no training is needed for using it. We have 2 modules in this scenario, where a user or the passenger, has their own options while admin has to login separately before having their own options. It works from the entering of user details, to payment, to printing of tickets and other operations are available as well.

1.2) Project Scope:

The scope of this project is to help a user reserve and modify their tickets without any hassle. This program will allow a user to reserve a ticket, edit their personal details, ask for a refund or cancel booking and view the reserved ticket. An Admin Panel will allow admins to View all the booked tickets, Add or Remove scheduled trains and Modify/Cancel a Ticket on User Request.

1.3) Modules in Project:

This program has 2 Modules:

User Module:

This module is for user or the passenger to book their ticket, make payments for ticket directly, modify their ticket, ask for a refund, and print/generate their ticket.

Admin Module:

This module allows admins to View all the booked tickets, Add or Remove Trains and Modify a ticket or remove it on a passenger's request.

1.4) Project Features:

User Module:

- Reserve a Ticket: Allows user to Reserve Tickets for available trains on desirable dates.
  We will be using linked list operations to insert ticket details.
- Choose Payment Methods & Pay: Allows user to pay for the ticket using Credit Card Details.
  We will use simply hard-coded Input/output for this feature as there is no other way to check Credit Card details i.e. No Databases used.
- Cancel/Refund Booked Tickets: Allows user to Cancel and Refund your tickets.
- We will be using linked list to delete ticket details. We will use searching algorithms to search and delete.
- Edit Booked Tickets Personal Details: Allows user to Edit Personal Details from your Booked Ticket. We will use searching algorithms to search and edit.
  We will be using linked list operations to edit ticket details.
- Print the Tickets: Prints on screen your ticket details for viewing and printing.
- We will be using linked list operations to display ticket details. We will use searching algorithms to search and print.

Admin Module:

- View All Booked Tickets: Allows Admin to View All Booked Tickets saved.
  We will be using linked list operations to display all tickets details.
- Add or Remove Trains: Add or Remove Available Trains from Database.
  We will be using linked list operations to display all trains details.
- Modify Booked Tickets (based on User Requests): Tickets booked can be modified.
  We will be using linked list operations to edit ticket details.

2. Requirements Specification:

2.1) Hardware Specifications:

- Dual Core x86 Processor or above.
- 2GB RAM minimum or above.
- Minimum 20GB Hardware required.

2.1) Software Specification:

- Visual Studio 2015
- Microsoft Windows 7 or above required.

3. SYSTEM IMPLEMENTATION:

3.1) INTRODUCTION:

In this project of Railway Booking Management System, we implemented the ability to reserve and modify railway tickets. We used Linked List Operations, Searching & Sorting Algorithms to modify and display as needed. This project is simple to use and very user-friendly. It has minimal hardware and software requirements, and no training is needed for using it.

3.2) CODE:

```
#include <iostream>
#include <string>
#include <sstream>
#include <ctime>
#include <fstream>
#include <Windows.h>
#include <conio.h>
using namespace std;
struct node
{
    int ticketID;
    string Name, Age, NIC, Contact, BookedClass, BookedTName, BookedTID,
BookedDest, BookedSource, BookedDate, DeptTime, ArrTime, BookedSeatNum;
    float Price;
    node *next;
};
struct trains
{
    string TName, TID, TDest, TSource, TDate, arriveTime, departTime, TClasses,
TxtFName;
    int TSeats;
};
struct SeatRecord
{
    string RowA;
    string RowB;
    string RowC;
};
node *head = NULL;
node *tail = NULL;
void admin();
int main();
string Name, Age, NIC, Contact, Dest, Source, SeatNum, Class, TID, Date, ATime,
DTime,
    TName;
bool trainCheck = false;
float Price;
string BookedID;
int currentID, Tis;
trains availTrains[10];
int noOfTrains = 5;
void copyTemplate(string fileName)
{
```

```
    string line;
    // For writing text file
    // Creating ofstream & ifstream class object
    ifstream ini_file{"template.txt"}; // This is the original file
    ofstream out_file{fileName};
    if (ini_file && out_file)
    {

        while (getline(ini_file, line))
        {
            out_file << line << "\n";
        }
    }
    else
    {
        // Something went wrong
        printf("Cannot read File");
    }
    // Closing file
    ini_file.close();
    out_file.close();
}
void Payment()
{
    string cc, cvv, dob, payChoice;
    string accNum, accHolderFName, accHolderLName;
    cout << endl
         << endl;
    cout << "\n\n\t\t\t\t\t==============PAYMENT DETIALS==============" <<
endl;
    cout << "\t\t\t\t\tChoose Payment Method. \n";
    cout << "\t\t\t\t\t1. JazzCash. \n";
    cout << "\t\t\t\t\t2. EasyPaisa. \n";
    cout << "\t\t\t\t\t3. Credit/Debit Card. \n";
    cout << "\n\n\t\t\t\t\tEnter Choice (1-3): ";
flag:
    cin >> payChoice;
    if (payChoice == "1")
    {
    phoneCheck:
        cout << "\n\n\t\t\t\t\tEnter JazzCash Mobile Account Number: ";
        cin >> accNum;
        if (accNum.length() != 11 || accNum.length() > 11)
        {
            cout << endl;
```

```
            cout << "\n\n\t\t\t\t\tEnter a Valid 11 Digit Mobile Phone Number!"
<< endl
                 << endl;
            goto phoneCheck;
        }
        cout << "\t\t\t\t\tEnter Account Holder's First Name: ";
        cin >> accHolderFName;
        cout << "\t\t\t\t\tEnter Account Holder's Last Name: ";
        cin >> accHolderLName;
    }
    else if (payChoice == "2")
    {
        cout << "\n\n\t\t\t\t\tEnter EasyPaisa Mobile Account Number: ";
        cin >> accNum;
        cout << "\n\n\t\t\t\t\tEnter Account Holder's First Name: ";
        cin >> accHolderFName;
        cout << "\n\n\t\t\t\t\tEnter Account Holder's Last Name: ";
        cin >> accHolderLName;
    }
    else if (payChoice == "3")
    {
    check:
        cout << "\n\n\t\t\t\t\tEnter Credit/Debit Card Number (16 digits): ";
        cin >> cc;
        if (cc.length() != 16)
        {
            cout << "\n\n\t\t\t\t\tCC number must have a length of 16 only! \n";
            goto check;
        }
        else
        check2:
            cout << "\n\n\t\t\t\t\tEnter CVV (3 digits): ";
        for (int i = 0; i > -1; i++)
        {
            char temp;
            temp = _getch();
            if (temp != 13 && temp != 8)
            {
                _putch('*');
            }
            if (temp == 13)
            {
                break;
            }
            if (temp == 8 || temp == 127 && !cvv.empty())
```

```
            {
                cout << "\b \b";
                cvv.erase(cvv.size() - 1);
            }
            else
                cvv += temp;
        }
        if (cvv.length() != 3)
        {
            cout << "\n\n\t\t\t\t\tCVV number must have a length of 3 only! \n";
            cvv = "";
            goto check2;
        }
        else
        {
            cout << "\n\n\t\t\t\t\tEnter Date Of Expiry(DD/MM/YYYY): ";
            cin >> dob;
        }
    }
    else
    {
        cout << "\t\t\t\t\tInvalid Choice. Please try again. \n";
        goto flag;
    }
    cout << "\n\n\t\t\t\t\t";
}
void trainSort()
{
    string temp, temp2, temp3, temp4, temp5, temp6, temp7, temp9, temp10;
    int temp8, j;
    for (int i = 0; i < noOfTrains; i++)
    {
        temp = availTrains[i].arriveTime;
        temp2 = availTrains[i].departTime;
        temp3 = availTrains[i].TClasses;
        temp4 = availTrains[i].TDate;
        temp5 = availTrains[i].TDest;
        temp6 = availTrains[i].TID;
        temp7 = availTrains[i].TName;
        temp8 = availTrains[i].TSeats;
        temp9 = availTrains[i].TSource;
        temp10 = availTrains[i].TxtFName;
        j = i - 1;
        while (j >= 0 && availTrains[j].TName > temp7)
        {
```

```
            availTrains[j + 1].arriveTime = availTrains[j].arriveTime;
            availTrains[j + 1].departTime = availTrains[j].departTime;
            availTrains[j + 1].TClasses = availTrains[j].TClasses;
            availTrains[j + 1].TDate = availTrains[j].TDate;
            availTrains[j + 1].TDest = availTrains[j].TDest;
            availTrains[j + 1].TID = availTrains[j].TID;
            availTrains[j + 1].TName = availTrains[j].TName;
            availTrains[j + 1].TSeats = availTrains[j].TSeats;
            availTrains[j + 1].TSource = availTrains[j].TSource;
            availTrains[j + 1].TxtFName = availTrains[j].TxtFName;
            j--;
        }
        availTrains[j + 1].arriveTime = temp;
        availTrains[j + 1].departTime = temp2;
        availTrains[j + 1].TClasses = temp3;
        availTrains[j + 1].TDate = temp4;
        availTrains[j + 1].TDest = temp5;
        availTrains[j + 1].TID = temp6;
        availTrains[j + 1].TName = temp7;
        availTrains[j + 1].TSeats = temp8;
        availTrains[j + 1].TSource = temp9;
        availTrains[j + 1].TxtFName = temp10;
    }
}
string SeatChoose()
{
    fstream Seat;
    int count = 0, delimit = 0;
    string line;
    SeatRecord Seats[10];
    Seat.open(availTrains[Tis].TxtFName);
    while (getline(Seat, line))
    {
        count++;
    }
    Seat.close();
    Seat.open(availTrains[Tis].TxtFName);
    for (int j = 0; j < count; j++)
    {
        string line1;
        getline(Seat, line1);
        for (int i = 0; i > -1; i++)
        {
            char temp;
            temp = line1[i];
```

```
            if (temp == '-')
            {
                delimit = i;
                break;
            }
            Seats[j].RowA += temp;
        }
        for (int i = delimit + 1; i > -1; i++)
        {
            char temp;
            temp = line1[i];
            if (temp == '-')
            {
                delimit = i;
                break;
            }
            Seats[j].RowB += temp;
        }
        for (int i = delimit + 1; i > -1; i++)
        {
            char temp;
            temp = line1[i];
            if (temp == '-')
            {
                delimit = i;
                break;
            }
            Seats[j].RowC += temp;
        }
    }
    Seat.close();
    string RowName, SeatNum, FinalSeatNum;
    bool Flag = false;
    cout << "\n\n\t\t\t\t\t\t\tA"
         << "\t"
         << " B"
         << "\t"
         << " C"
         << "\t\n";
    for (int i = 0; i < 10; i++)
    {
        cout << "\n\n\t\t\t\t\t\t\t" << Seats[i].RowA << "\t" << Seats[i].RowB <<
"\t" << Seats[i].RowC << "\t\n";
    }
    cout << "\n\n\t\t\t\t\t\t ====================================\n";
```

```
    for (int x = 0; x > -1; x++)
    {
        cout << "\n\n\t\t\t\t\t\tEnter the row name that you chose: ";
        cin >> RowName;
        if (RowName != "A" && RowName != "B" && RowName != "C")
        {
            cout << "\n\n\t\t\t\t\t\tInvalid row, please try again. ";
        }
        else
            break;
    }
    for (int x = 0; x > -1; x++)
    {
    check7:
        cout << "\n\n\t\t\t\t\t\tEnter the seat number that you chose: ";
        cin >> SeatNum;
        if (SeatNum == "XX")
        {
            cout << "\n\n\t\t\t\t\t\tInvalid seat, please try again! \n";
            goto check7;
        }
        for (int i = 0; i < 10; i++)
        {
            if (RowName == "A")
            {
                if (SeatNum == Seats[i].RowA)
                {
                    Seats[i].RowA = "XX";
                    Flag = true;
                    break;
                }
            }
            else if (RowName == "B")
            {
                if (SeatNum == Seats[i].RowB)
                {
                    Seats[i].RowB = "XX";
                    Flag = true;
                    break;
                }
            }
            else if (RowName == "C")
            {
                if (SeatNum == Seats[i].RowC)
                {
```

```
                    Seats[i].RowC = "XX";
                    Flag = true;
                    break;
                }
            }
        }
        if (Flag == false)
        {
            cout << "\n\n\t\t\t\t\tSeat number not found, please try again. \n";
        }
        else
            break;
    }
    FinalSeatNum = RowName + SeatNum;
    ofstream Del;
    Del.open(availTrains[Tis].TxtFName, ios::trunc);
    Del.close();
    Del.open(availTrains[Tis].TxtFName, ios::trunc);
    for (int i = 0; i < 10; i++)
    {
        Del << Seats[i].RowA << "-" << Seats[i].RowB << "-" << Seats[i].RowC <<
"-\n";
    }
    Del.close();
    return FinalSeatNum;
}
void iniTrain()
{
    availTrains[0].TName = "Shalimar Express";
    availTrains[0].TID = "SHE753";
    availTrains[0].TSource = "Karachi";
    availTrains[0].TDest = "Lahore";
    availTrains[0].TDate = "24-Jan-2023";
    availTrains[0].arriveTime = "20:00 (8:00 PM)";
    availTrains[0].departTime = "16:40 (4:40 PM)";
    availTrains[0].TClasses = "Economy, AC Lower, AC Business";
    availTrains[0].TxtFName = availTrains[0].TName;
    availTrains[0].TxtFName.append(".txt");
    availTrains[0].TSeats = 30;
    copyTemplate(availTrains[0].TxtFName);
    availTrains[1].TName = "Karakoram Express";
    availTrains[1].TID = "KKE694";
    availTrains[1].TSource = "Karachi";
    availTrains[1].TDest = "Faisalabad";
    availTrains[1].TDate = "26-Jan-2023";
```

```
    availTrains[1].arriveTime = "08:00 (8:00 AM)";
    availTrains[1].departTime = "04:40 (4:40 AM)";
    availTrains[1].TClasses = "Economy, AC Lower, AC Business";
    availTrains[1].TxtFName = availTrains[1].TName;
    availTrains[1].TxtFName.append(".txt");
    availTrains[1].TSeats = 30;
    copyTemplate(availTrains[1].TxtFName);
    availTrains[2].TName = "Green Line Express";
    availTrains[2].TID = "GLE400";
    availTrains[2].TSource = "Karachi";
    availTrains[2].TDest = "Islamabad";
    availTrains[2].TDate = "28-Jan-2023";
    availTrains[2].arriveTime = "12:00 (12:00 PM)";
    availTrains[2].departTime = "10:00 (10:00 AM)";
    availTrains[2].TClasses = "Economy, AC Lower, AC Business";
    availTrains[2].TxtFName = availTrains[2].TName;
    availTrains[2].TxtFName.append(".txt");
    availTrains[2].TSeats = 30;
    copyTemplate(availTrains[2].TxtFName);
    availTrains[3].TName = "Tezgam Express";
    availTrains[3].TID = "TEZ123";
    availTrains[3].TSource = "Karachi";
    availTrains[3].TDest = "Faisalabad";
    availTrains[3].TDate = "29-Jan-2021";
    availTrains[3].arriveTime = "15:00 (03:00 PM)";
    availTrains[3].departTime = "9:00 (09:00 AM)";
    availTrains[3].TClasses = "Economy, AC Lower, AC Business";
    availTrains[3].TxtFName = availTrains[3].TName;
    availTrains[3].TxtFName.append(".txt");
    availTrains[3].TSeats = 30;
    copyTemplate(availTrains[3].TxtFName);
    availTrains[4].TName = "Karachi Express";
    availTrains[4].TID = "KHE123";
    availTrains[4].TSource = "Karachi";
    availTrains[4].TDest = "Lahore";
    availTrains[4].TDate = "19-Jan-2023";
    availTrains[4].arriveTime = "12:00 (12:00 AM)";
    availTrains[4].departTime = "12:00 (09:00 PM)";
    availTrains[4].TClasses = "Economy, AC Lower, AC Business";
    availTrains[4].TxtFName = availTrains[4].TName;
    availTrains[4].TxtFName.append(".txt");
    availTrains[4].TSeats = 30;
    copyTemplate(availTrains[4].TxtFName);
}
string trainFill()
```

```
{
    bool check = false;
    bool check2 = false;
    trainCheck = true;
flag:
    cout << "\t\t\t\t\tEnter Your Departure City: ";
    cin >> Dest;
    cin.ignore();
    cout << "\t\t\t\t\tEnter Your Arrival City: ";
    cin >> Source;
    cin.ignore();
    trainSort();
    cout << "\n\n\t\t\t\t\tAVAILABLE TRAINS ON THAT DAY IN THIS ROUTE ARE AS
FOLLOWS (IFANY): \n\n"
         << endl;
    for (int i = 0; i < noOfTrains; i++)
    {
        if (availTrains[i].TSource == Dest && availTrains[i].TDest == Source)
        {
            cout << "\t\t\t\t\tTrain Name: " << availTrains[i].TName << endl;
            cout << "\t\t\t\t\tTrain ID: " << availTrains[i].TID << endl;
            cout << "\t\t\t\t\tTrain Source: " << availTrains[i].TSource << endl;
            cout << "\t\t\t\t\tTrain Destination: " << availTrains[i].TDest <<
endl;
            cout << "\t\t\t\t\tTrain Departure Date: " << availTrains[i].TDate <<
endl;
            cout << "\t\t\t\t\tTrain Arrival Time: " << availTrains[i].arriveTime
<< endl;
            cout << "\t\t\t\t\tTrain Departure Time: " <<
availTrains[i].departTime << endl;
            cout << "\t\t\t\t\tAvailable Train Classes: " <<
availTrains[i].TClasses << endl;
            cout << "\t\t\t\t\tSeats Available: " << availTrains[i].TSeats <<
endl;
            cout << endl
                 << endl;
            check = true;
        }
    }
    if (check == false)
    {
        cout << "\t\t\t\t\tNo trains available on this route. Re-enter your
details. \n";
        goto flag;
    }
```

```
    else
    {
    redo:
        cout << "\t\t\t\t\tInput Train ID that you wish to book: ";
        cin >> BookedID;
        for (int i = 0; i < noOfTrains; i++)
        {
            if (BookedID == availTrains[i].TID)
            {
                Tis = i;
                availTrains[i].TSeats--;
                return BookedID;
                check2 = true;
            }
        }
        if (check2 == false)
        {
            cout << "\t\t\t\t\tInput correct train ID! \n";
            goto redo;
        }
    }
}
void trainPrice()
{
    int choice = 0;
    cout << "\t\t\t\t\tPrices based on classes are as follows: \n";
    cout << "\t\t\t\t\t1. Economy \t\t 2500 PKR \n";
    cout << "\t\t\t\t\t2. AC Lower \t\t 4500 PKR \n";
    cout << "\t\t\t\t\t3. AC Business \t\t 6500 PKR \n\n";
flag:
    cout << "\t\t\t\t\tInput Choice (1-3): ";
    cin >> choice;
    if (choice == 1)
    {
        Price = 2500;
        Class = "Economy [No Berth]";
    }
    else if (choice == 2)
    {
        Price = 4500;
        Class = "AC Lower [Berth Included]";
    }
    else if (choice == 3)
    {
        Price = 6500;
```

```
        Class = "AC Business [Berth Included]";
    }
    else
    {
        cout << "\t\t\t\t\tInvalid Input. Try again. \n\n";
        goto flag;
    }
}
void reserveTicket()
{
    string fName, lName;
    node *obj = new node();
    node *temp = tail;
    if (head == NULL && tail == NULL)
    {
        obj->next = NULL;
        head = obj;
        tail = obj;
    }
    else
    {
        temp->next = obj;
        tail = obj;
    }
    cout << endl
         << endl
         << endl
         << endl
         << endl;
    cout << "\t\t\t\t\tEnter First Name of Passenger: ";
    cin >> fName;
    cout << "\t\t\t\t\tEnter Last Name of Passenger: ";
    cin >> lName;
    Name = fName + " " + lName;
    cout << "\t\t\t\t\tEnter Age: ";
    cin >> Age;
nicCheck:
    cout << "\t\t\t\t\tEnter CNIC Number: ";
    cin >> NIC;
    if (NIC.length() != 13 || NIC.length() > 13)
    {
        cout << endl;
        cout << "\t\t\t\t\tEnter a Valid 13 digit CNIC Number!" << endl
             << endl;
        goto nicCheck;
```

```
    }
phoneCheck:
    cout << "\t\t\t\t\tEnter Contact Number: ";
    cin >> Contact;
    if (Contact.length() != 11 || Contact.length() > 11)
    {
        cout << endl;
        cout << "\t\t\t\t\tEnter a Valid 11 digit Mobile Phone Number!" << endl
             << endl;
        goto phoneCheck;
    }
    cin.clear();
    trainFill();
    int seatCheck;
    for (int i = 0; i < noOfTrains; i++)
    {
        if (availTrains[i].TID == BookedID)
        {
            TName = availTrains[i].TName;
            TID = BookedID;
            Date = availTrains[i].TDate;
            ATime = availTrains[i].arriveTime;
            DTime = availTrains[i].departTime;
            Tis = i;
        }
    }
    trainPrice();
    SeatNum = SeatChoose();
    Payment();
    system("cls");
    cout << "\n\n\t\t\t\t\t Your payment is successfully processed!" << endl;
    cout << "\t\t\t\t\t PKR " << Price << " has been deducted from your Account."
<< endl;
    srand(time(0));
    // Randomly Generated Ticket ID:
    obj->ticketID = (rand());
    cout << endl
         << endl;
    cout << "\t\t\t\t\tTicket Generated! Your ID is: " << obj->ticketID << endl;
    currentID = obj->ticketID;
    obj->Name = Name;
    obj->NIC = NIC;
    obj->Age = Age;
    obj->Contact = Contact;
    obj->BookedTName = TName;
```

```
    obj->BookedDest = Dest;
    obj->BookedSource = Source;
    obj->BookedSeatNum = SeatNum;
    obj->Price = Price;
    obj->BookedClass = Class;
    obj->BookedTID = TID;
    obj->BookedDate = Date;
    obj->DeptTime = DTime;
    obj->ArrTime = ATime;
    cout << endl
         << endl;
}
void viewTicket()
{
    int search;
    string choice;
    bool check = false;
    node *temp = head;
    cin.clear();
    cout << endl
         << endl
         << endl
         << endl
         << endl;
jump:
    cout << "\t\t\t\t\tDo You Want to View Most Recent Booking? If so, Press
1\n";
    cout << "\t\t\t\t\tDo You Want to Search on Older Booking? If so, Press 2\n";
    cout << "\t\t\t\t\tEnter Choice (1-2): ";
    cin >> choice;
    cin.ignore();
    if (choice == "1")
    {
        search = currentID;
        check = true;
    }
    else if (choice == "2")
    {
        cout << "\t\t\t\t\tEnter Your Booking ID: ";
        cin >> search;
        cout << endl
             << endl;
        check = true;
    }
    else
```

```
    {
        cout << "\t\t\t\t\tInvalid input. Retry! \n";
        goto jump;
    }
    do
    {
        if (temp != NULL)
        {
            if (temp->ticketID == search && check == true)
            {
                cout << "\t\t\t\t\tBooking ID: " << temp->ticketID << endl;
                cout << "\t\t\t\t\tName: " << temp->Name << endl;
                cout << "\t\t\t\t\tAge: " << temp->Age << endl;
                cout << "\t\t\t\t\tNIC: " << temp->NIC << endl;
                cout << "\t\t\t\t\tContact: " << temp->Contact << endl;
                cout << "\t\t\t\t\tTrain ID: " << temp->BookedTID << endl;
                cout << "\t\t\t\t\tTrain Name: " << temp->BookedTName << endl;
                cout << "\t\t\t\t\tTrain Class: " << temp->BookedClass << endl;
                cout << "\t\t\t\t\tSeat Number: " << temp->BookedSeatNum << endl;
                cout << "\t\t\t\t\tDate of Booking: " << temp->BookedDate <<
endl;
                cout << "\t\t\t\t\tSource: " << temp->BookedDest << endl;
                cout << "\t\t\t\t\tDestination: " << temp->BookedSource << endl;
                cout << "\t\t\t\t\tDeparture TIme: " << temp->DeptTime << endl;
                cout << "\t\t\t\t\tArrival Time: " << temp->ArrTime << endl;
                cout << endl
                     << endl;
                break;
                main();
            }
            else
            {
                temp = temp->next;
            }
        }
        else
            cout << "\t\t\t\t\tNo Bookings Exist. \n";
    } while (temp);
}
void display()
{
    cout << endl
         << endl
         << endl
         << endl
```

```
         << endl;
    if (head != NULL && tail != NULL)
    {
        cout << "\t\t\t\t\tList is as follows: \n";
        node *temp = head;
        do
        {
            cout << "\t\t\t\t\tTicket ID: " << temp->ticketID << endl;
            cout << "\t\t\t\t\tName: " << temp->Name << endl;
            cout << "\t\t\t\t\tAge: " << temp->Age << endl;
            cout << "\t\t\t\t\tNIC: " << temp->NIC << endl;
            cout << "\t\t\t\t\tContact: " << temp->Contact << endl;
            cout << "\t\t\t\t\tTrain ID: " << temp->BookedTID << endl;
            cout << "\t\t\t\t\tTrain Name: " << temp->BookedTName << endl;
            cout << "\t\t\t\t\tTrain Class: " << temp->BookedClass << endl;
            cout << "\t\t\t\t\tSeat Number: " << temp->BookedSeatNum << endl;
            cout << "\t\t\t\t\tDate of Booking: " << temp->BookedDate << endl;
            cout << "\t\t\t\t\tSource: " << temp->BookedSource << endl;
            cout << "\t\t\t\t\tDestination: " << temp->BookedDest << endl;
            cout << "\t\t\t\t\tDeparture TIme: " << temp->DeptTime << endl;
            cout << "\t\t\t\t\tArrival Time: " << temp->ArrTime << endl;
            cout << endl;
            temp = temp->next;
        } while (temp);
    }
    else
        cout << "List is empty, nothing to display. \n";
}
void removeTicket(int remTicket)
{
    node *prev = head;
    node *delNode = head;
    while (delNode != NULL)
    {
        if (delNode->ticketID == remTicket)
        {
            break;
        }
        else
        {
            prev = delNode;
            delNode = delNode->next;
        }
    }
    if (delNode == NULL)
```

```
    {
        cout << "Ticket ID Not Found!" << endl;
    }
    else
    {
        cout << "Deleted Ticket ID: " << delNode->ticketID << "\n";
        prev->next = delNode->next; // unlink the node you remove
        delete delNode;             // delete the node
    }
}
void editDetais()
{
    string fName, lName;
    cout << "\t\t\t\t\tEDIT DETAILS:" << endl
         << endl;
    int search;
    node *temp = head;
    cin.clear();
    cout << endl
         << endl
         << endl
         << endl
         << endl;
    cout << "\t\t\t\t\tEnter ticket ID: ";
    cin >> search;
    while (temp != NULL)
    {
        if (temp->ticketID == search)
        {
            cout << "\t\t\t\t\tEnter First Name of Passenger: ";
            cin >> fName;
            cout << "\t\t\t\t\tEnter Last Name of Passenger: ";
            cin >> lName;
            Name = fName + " " + lName;
            cout << "\t\t\t\t\tEnter CNIC Number: ";
            cin >> NIC;
            cin.ignore();
            cout << "\t\t\t\t\tEnter Contact Number: ";
            cin >> Contact;
            cin.ignore();
            temp->Name = Name;
            temp->NIC = NIC;
            temp->Age = Age;
            temp->Contact = Contact;
        }
```

```
        temp = temp->next;
    }
}
void addTrains()
{
    string choice;
    do
    {
        cin.ignore();
        cout << endl
             << endl
             << endl
             << endl
             << endl;
        cout << "\t\t\t\t\tEnter Train Name: ";
        getline(cin, availTrains[noOfTrains].TName);
        cout << "\t\t\t\t\tEnter Train ID: ";
        getline(cin, availTrains[noOfTrains].TID);
        cout << "\t\t\t\t\tEnter Train Source: ";
        getline(cin, availTrains[noOfTrains].TSource);
        cout << "\t\t\t\t\tEnter Train Destination: ";
        getline(cin, availTrains[noOfTrains].TDest);
        cout << "\t\t\t\t\tEnter Train Departure Date [HH:MM (MM:HH AM/PM)]: ";
        getline(cin, availTrains[noOfTrains].TDate);
        cout << "\t\t\t\t\tEnter Train Departure Time: ";
        getline(cin, availTrains[noOfTrains].departTime);
        cout << "\t\t\t\t\tEnter Train Arrival Time: [HH:MM (MM:HH AM/PM)]: ";
        getline(cin, availTrains[noOfTrains].arriveTime);
        availTrains[noOfTrains].TClasses = "Economy, AC Lower, AC Business";
        availTrains[noOfTrains].TSeats = 30;
        availTrains[noOfTrains].TxtFName = availTrains[noOfTrains].TName;
        availTrains[noOfTrains].TxtFName.append(".txt");
        copyTemplate(availTrains[noOfTrains].TxtFName);
        noOfTrains++;
        cin.clear();
        cout << "\t\t\t\t\tDo you want to add more trains? (Y/N): ";
        cin >> choice;
        if (choice != "Y" && choice != "N")
            cout << "\t\t\t\t\tInvalid input. Enter choice again. ";
        else if (choice == "N")
        {
            system("cls");
            cout << "\t\t\t\t\tTrain added! \n";
            trainSort();
            cout << "\t\t\t\t\tSession logged out. Please login again. \n";
```

```
            main();
        }
    } while (choice != "N");
}
void admin()
{
    string choice, user, password;
    system("cls");
    cin.clear();
    cout << endl
         << endl
         << endl
         << endl
         << endl;
    cout << "\t\t\t\t\tWELCOME TO ADMIN PORTAL!\n\n";
    cout << "\t\t\t\t\tEnter your login credentials below! \n";
login:
    cout << "\t\t\t\t\tEnter your Username: ";
    cin >> user;
    cout << "\t\t\t\t\tEnter your Password: ";
    for (int i = 0; i > -1; i++)
    {
        char temp;
        temp = _getch();
        if (temp != 13 && temp != 8)
        {
            _putch('*');
        }
        if (temp == 13)
        {
            break;
        }
        if (temp == 8 || temp == 127 && !password.empty())
        {
            cout << "\b \b";
            password.erase(password.size() - 1);
        }
        else
            password += temp;
    }
    // cin >> password;
    if (user == "Shalal" && password == "shalal123" || user == "Arham" &&
password == "arham123" || user == "Mania" && password == "mania123")
    {
        cout << "\t\t\t\t\tLogged in Successfully!\n";
```

```
        cin.clear();
        system("cls");
        cout << endl
             << endl
             << endl
             << endl
             << endl;
        cout << "\t\t\t\t\tSelect 1 to View all Booked Tickets. \n";
        cout << "\t\t\t\t\tSelect 2 to add Trains. \n";
        cout << "\t\t\t\t\tSelect 3 to Exit\n";
        do
        {
            cout << "\t\t\t\t\tEnter Choice To Proceed (1-2): ";
            cin >> choice;
            if (choice == "1")
                display;
            else if (choice == "2")
            {
                cin.clear();
                addTrains();
                if (trainCheck == false)
                    trainFill();
            }
            else if (choice == "3")
            {
                system("cls");
                main();
            }
            else
                cout << "\t\t\t\t\tEnter correct choice. Try again. \n";
        } while (choice != "3");
    }
    else
    {
        cout << "\t\t\t\t\t\nInvalid credentials. Please login again. \n";
        goto login;
    }
}

int main()
{
    system("color F0");
    iniTrain();
relog:
    cin.clear();
```

```
    string check;
    cout << endl
         << endl
         << endl
         << endl
         << endl;
    cout << "\t\t\t\t\tWELCOME TO RAILWAY BOOKING SYSTEM!\n\n\n";
    Sleep(300);
    cout << "\n\n\t\t\t\t\t\t\tProject Prepared by:";
    Sleep(400);
    cout << "\n\n\t\t\t\t\t\t\t--------------------";
    Sleep(500);
    cout << "\n\n\t\t\t\t Shalal Bakhtiar                    02-134212-021";
    Sleep(1000);
    cout << "\n\n\t\t\t\t Arham Tanveer Mallick             02-134212-028";
    Sleep(2000);
    cout << "\n\n\t\t\t\t Mania Imam                        02-134212-013";
    Sleep(2200);
    cout << endl;
    system("cls");
log:
    cout << "\n\n\n\n\n\t\t\t\tIf you are a user looking to book a ticket, type 1
\n";
    cout << "\t\t\tIf you are an admin, type 2 \n";
    cout << "\n\n\t\t\t\t\t\tChoice: ";
    cin >> check;
    if (check == "1")
    {
        int tickID;
        string choice;
        system("cls");
        cout << endl
             << endl
             << endl
             << endl
             << endl;
        do
        {
            cout << "\t\t\t\t ----------------------------------------\n";
            cout << "\t\t\t\t\tSelect 1 to Reserve a Ticket. \n";
            cout << "\t\t\t\t\tSelect 2 to View your Reservation. \n";
            cout << "\t\t\t\t\tSelect 3 to Display all tickets. \n";
            cout << "\t\t\t\t\tSelect 4 to Delete a ticket. \n";
            cout << "\t\t\t\t\tSelect 5 to Edit Details. \n";
            cout << "\t\t\t\t\tSelect 6 to Log Out of Session. \n";
```

```
            cout << "\t\t\t\t\tSelect 7 to Exit\n";
            cout << endl
                 << endl;
            cout << "\t\t\t\t\tEnter Choice To Proceed (1-7): ";
            cin >> choice;
            if (choice == "1")
            {
                system("cls");
                cin.clear();
                reserveTicket();
            }
            else if (choice == "2")
            {
                system("cls");
                viewTicket();
            }
            else if (choice == "3")
            {
                system("cls");
                display();
            }
            else if (choice == "4")
            {
                cout << "\t\t\t\t\tEnter your Ticket ID: ";
                cin >> tickID;
                removeTicket(tickID);
            }
            else if (choice == "5")
            {
                system("cls");
                editDetais();
            }
            else if (choice == "6")
            {
                goto relog;
            }
            else if (choice == "7")
            {
                Sleep(300);
                cout << "\t\t\t\t\tTHANKYOU FOR USING OUR SOFTWARE! \n";
                Sleep(300);
                exit(0);
            }
            else
                cout << "\t\t\t\t\tEnter correct choice. Try again. \n";
```

```
        } while (choice != "7");
    }
    else if (check == "2")
        admin();
    else
    {
        cout << "\t\t\t\t\tInvalid input. Try again. \n";
        goto log;
    }
}
```

4. SYSTEM IMPLEMENTATION:

4.1 Testing Methods

Black Box Testing: Black box testing involves testing a system with no prior knowledge of its internal workings. A tester provides an input, and observes the output generated by the system under test.

White Box Testing: is software testing technique in which internal structure, design and coding of software are tested to verify flow of input-output and to improve design, usability and security. In white box testing, code is visible to testers so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing and Glass box testing.

5. SAMPLE SCREENSHOTS OF SYSTEM

```
                WELCOME TO RAILWAY BOOKING SYSTEM!


                        Project Prepared by:

                        --------------------

        Shalal Bakhtiar                 02-134212-021

        Arham Tanveer Mallick           02-134212-028

        Mania Imam                      02-134212-013
```

```
If you are a user looking to book a ticket, type 1
If you are an admin, type 2

                        Choice:
```

```
----------------------------------------
        Select 1 to Reserve a Ticket.
        Select 2 to View your Reservation.
        Select 3 to Display all tickets.
        Select 4 to Delete a ticket.
        Select 5 to Edit Details.
        Select 6 to Log Out of Session.
        Select 7 to Exit


        Enter Choice To Proceed (1-7):
```

```
Enter First Name of Passenger: Butcher
Enter Last Name of Passenger: Williams
Enter Age: 22
Enter CNIC Number: 4220131247586
Enter Contact Number: 03331145933
Enter Your Departure City: Karachi
Enter Your Arrival City: Lahore


AVAILABLE TRAINS ON THAT DAY IN THIS ROUTE ARE AS FOLLOWS (IFANY):


Train Name: Karachi Express
Train ID: KHE123
Train Source: Karachi
Train Destination: Lahore
Train Departure Date: 19-Jan-2023
Train Arrival Time: 12:00 (12:00 AM)
Train Departure Time: 12:00 (09:00 PM)
Available Train Classes: Economy, AC Lower, AC Business
Seats Available: 30


Train Name: Shalimar Express
Train ID: SHE753
Train Source: Karachi
Train Destination: Lahore
Train Departure Date: 24-Jan-2023
Train Arrival Time: 20:00 (8:00 PM)
Train Departure Time: 16:40 (4:40 PM)
Available Train Classes: Economy, AC Lower, AC Business
Seats Available: 30


Input Train ID that you wish to book:
```

```
Input Train ID that you wish to book: KHE123
Prices based on classes are as follows:
1. Economy                2500 PKR
2. AC Lower               4500 PKR
3. AC Business            6500 PKR

Input Choice (1-3): 3


                A        B       C


                XX       11      21


                02       12      22


                03       13      23


                04       14      24


                05       15      25


                06       16      26


                07       17      27


                08       18      28


                09       19      29


                10       20      30


          ======================================
```

```
Enter the row name that you chose: A


Enter the seat number that you chose: 09



===============PAYMENT DETIALS===============
Choose Payment Method.
1. JazzCash.
2. EasyPaisa.
3. Credit/Debit Card.


Enter Choice (1-3):
```

```
     Your payment is successfully processed!
     PKR 6500 has been deducted from your Account.


    Ticket Generated! Your ID is: 5867
```

```
      List is as follows:
      Ticket ID: 5867
      Name: Butcher Williams
      Age: 22
      NIC: 4220131247586
      Contact: 03331145933
      Train ID: KHE123
      Train Name: Karachi Express
      Train Class: AC Business [Berth Included]
      Seat Number: A09
      Date of Booking: 19-Jan-2023
      Source: Lahore
      Destination: Karachi
      Departure TIme: 12:00 (09:00 PM)
      Arrival Time: 12:00 (12:00 AM)
```

```
WELCOME TO ADMIN PORTAL!

Enter your login credentials below!
Enter your Username: Shalal
Enter your Password: *********
```

```
Select 1 to View all Booked Tickets.
Select 2 to add Trains.
Select 3 to Exit
Enter Choice To Proceed (1-2):
```