ACKNOWLEDGEMENT

The satisfactions that accompany the successful completion of my mini-project on

"{Railway Reservation System}" it would be incomplete without the mention of people who made it possible, whose noble gesture, affection, guidance, encouragement and support crowned my efforts with success. It is my privilege to express my gratitude and respect to all those who inspired me in the completion of my mini-project.

I am extremely grateful to our respective guide **Ms. Tejashwini S.G** for her noble gesture, support co-ordination and valuable suggestions given to me to complete my mini-project.

I am also thankful to **Dr. R. N. Kulkarni,** H.O.D. Department of CSE, for his co-ordination and valuable suggestions given to me to complete my mini-project.

I also thankful to my Principle, Management, Teaching and non-teaching staff for their co-ordination and valuable suggestions given to me in completing the mini-project.

NAMES	USN
Mohammed Fatah	**3BR19CS086**
Mohammed Owais	**3BR19CS087**
Mundergi Sandeep	**3BR19CS091**
P. Vijayalakshmi	**3BR19CS107**
R. Padmapriya	**3BR19CS115**

TABLE OF CONTENTS

ABSTRACT

Travelling is growing business in the world. Railway reservation system gives the details of every passenger in every train. It also contains schedule and information of each train.

To reserve a ticket, it is very easy task, and make our work easier but more than 1000 lines of code will run in a background. we go deep through it and we will understand clearly by going through the code. Sometimes a lot of problem occurs and they facing many disputes with the customers.to solve this problem and further maintaining records of passenger details, seat availability, price per seat, bill generation and other things. we are offering this proposal of computerized reservation system.

In this code we also given the section for the admin too. they can create the data, remove the existing data or modifying the existing data.

By this software we can reserve the tickets from any part of the world, via internet. customers can check the available of seats and they can select it.

The project provides and checks all sorts of constrains so that user gives only useful data and thus validation is done in an effective way.

Chapter 1:

Introduction

The Online Railway Reservation System is a Web-based application. That allows the admin to create a data, delete the data or they can modify the existing data. In these they can also display the train records and also, they can also display the details of every passenger in every train

And visitors can check the availability, they can book the ticket, they can also cancel the ticket and finally we can see the status of passengers.

This system is established for all the users. The use of Railway Reservation system is, there is no need to visit the Railway station to book the ticket or for enquire. We can make these all possible from the place where we are, through the mobile/laptop via internet. From these we can save our time and we can involve in our work easily without any tensions. Users can use these programs directly on their websites and no need to install it and also includes the maintenance of information like schedule and details of each train

This reservation system has 3 modules.

1.Admin access

2.User access

3.Exit

"Admin access" In the admin Module. First it asks the admin password for the security purpose, after entering the password it will give us 7 sections.

The admin access Module has 7 sections.

1. **Create new train file**
2. **New record**
3. **Modify a train record**
4. **Go back**
5. **Delete a train record**
6. **Display all the current records**
7. **Display passenger list**

"User access" In this User access module it helps the user/passenger. To Book the tickets

The user access Module has 4 sections.

1. **To check availability**
2. **To book tickets**
3. **For cancellation**
4. **To go back**

"EXIT" This module is used to exit the program.

1.1 VISION, MISSION AND OBJECTIVES

1.1.1 Vision:

To develop automated system not just fulfill current requirement of reservation system in increasing its efficiency and also provide future support.

1.1.2 Mission:

To develop quality products to empower customers to face challenges.

1.1.3 Objectives:

The main objectives of the proposed "Railway Reservation System" Is to eliminate the manual reservation system.

- Making the reservation system, fast, user friendly avoid the unnecessary delay in reservation
- The new system needs to develop that can handle lots of records efficiency, searching information about reservation data.
- Any report according to the requirements can get easily.
- Response time is very less comparative to manual systems.

1.2 Scope:

Railway Reservation System is one of the modifications that were carried out in the Passengers service system so that working and availability of service area can be broadened

1.3 Problem statement:

Railway Reservation System that Was suggested till now, are not up to the desired level. there is no single system which automates the process. In order to build the system, all the processes in the business should be studied, system study helps us under the problem and needs of the application. System study aims at establishing the requests for the system to be acquired, development and installed.

To design and develop software for Railway Reservation System.

1.4 Existing system:

The existing Railway reservation system is not completely computerized.

- Existing system is totally on book and thus a great amount of manual work has to be done. The amount of manual work increases exponentially which increase in services.
- Needs a lot of working staff and extra attention on all records.
- In existing system, there are various problems like keeping records of items, seat available, prices per seat and soon.
- Finding out details regarding any information is very difficult, as the user has to go through all the books manually.
- Major problem was lack of security, this is major bottle neck.

1.5 Proposed system:

The system is very simple to design and implementation. This system requires very low system resources and the system will work in almost all configuration. It has got following features:

- Ensure data accuracy.
- Any person across the world, having internet can access this service.
- Availability of the seats will be enquired very easily.
- Passengers can also cancel their tickets easily.
- Passengers can check the number of trains from point ‘A’ to ‘B’, price ranges, automated report and bill generation etc.

CHAPTER 2

SYSTEM ANALYSIS AND REQUIREMENTS

2.1 Requirements Specification

1.1.1 Hardware Requirements:

- PC
- Processor: Pentium IV/Dual core/Core duo processors
- RAM: 2 GB/4GB and above
- Hard disk:128GB/256GB/512GB/1TB and above

1.1.2 Software Requirements:

- PC
- Programming language: C++
- Operating System: Windows XP/7/8/10 or Linux

2.2 Functional Requirements:

These are statements of services the system should provide; it defines how the system should behave in particular and how it should react to particular inputs. In some cases, the function requirements may also explicitly state what the system as a whole.

- The system shall incorporate mechanism to authenticate its users
- The system shall verify and validate all user input and should notify in case of error detection and should help the user in error correction
- The system shall allow sharing of files in the system
- The system shall allow quick messages to be exchanged without face-to-face interaction

2.3 Non-Functional Requirements:

Non-functional requirements address aspects of the system other than the specific functions it performs. These aspects include system performance, costs, and such general system characteristics as reliability, security, and portability. The non-functional requirements also address aspects of the system development process and operational personnel. It includes the following:

- The system shall be user friendly and consistent
- The system shall provide attractive graphical interface for the user.
- The system shall allow developer access to installed environment.
- The system shall target customer base.

CHAPTER 3

SYSTEM ARCHITECTURE DESCRIPTION

3.1 SYSTEM ARCHITECTURE

System Architecture is a theoretical blueprint for the construction and performance of a system. It consists of customer requirements, conventions, rules, and standards employed in a framework.

1. Use case modelling.
2. Data flow diagram.

3.1.1 Use Case Diagram for Railway Reservation System

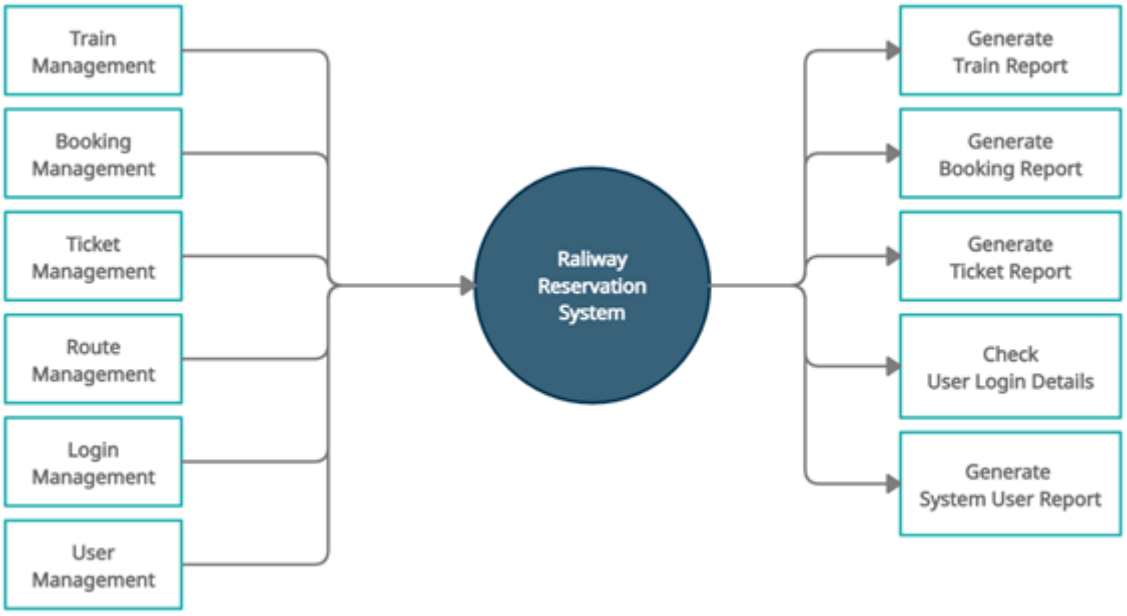


Fig1: Use Case Diagram of Railway Reservation System

3.1.2 Data Flow Diagram for Railway Reservation System

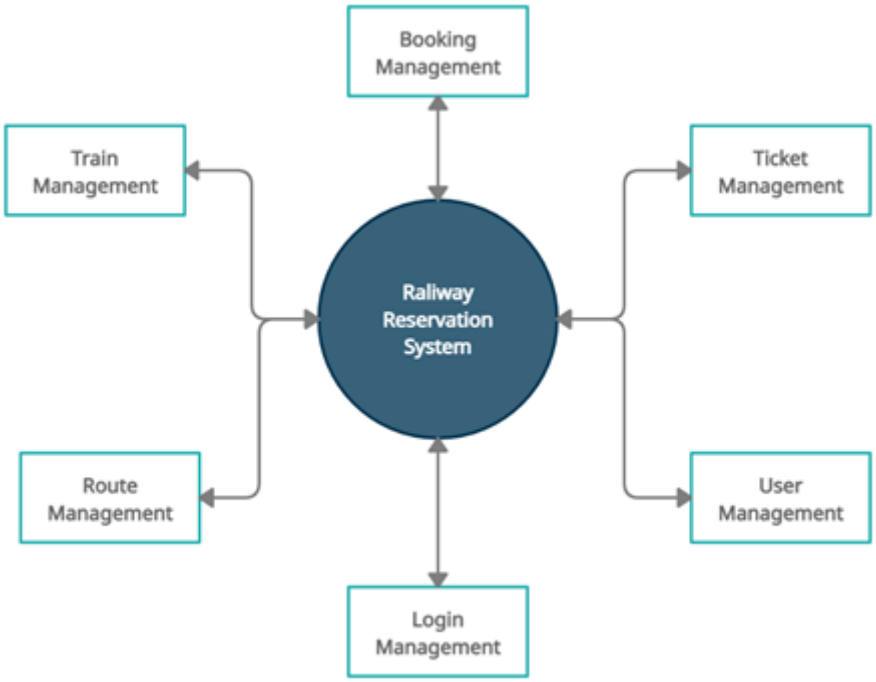
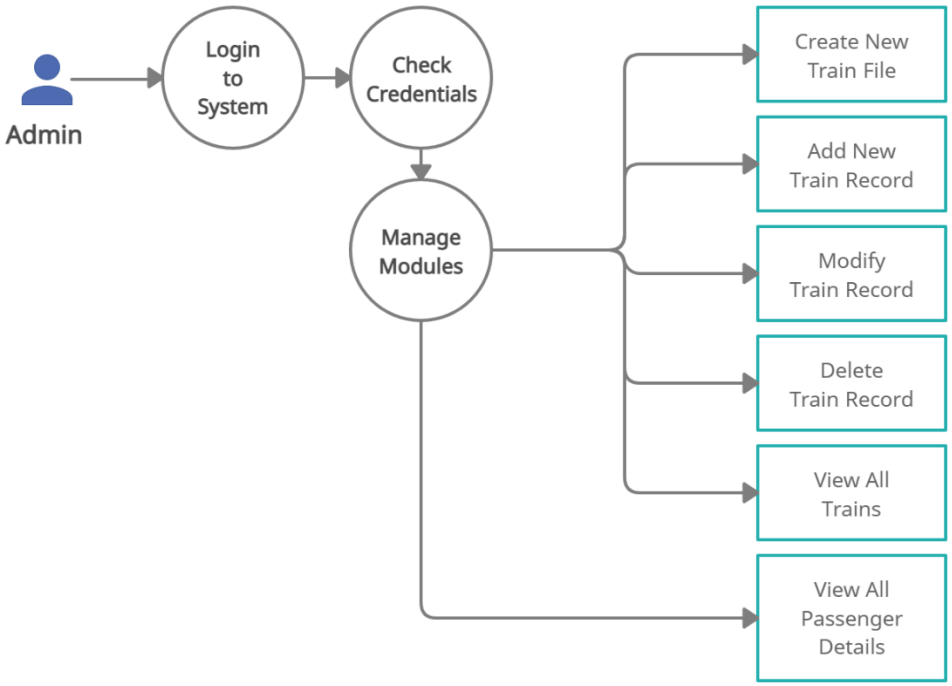


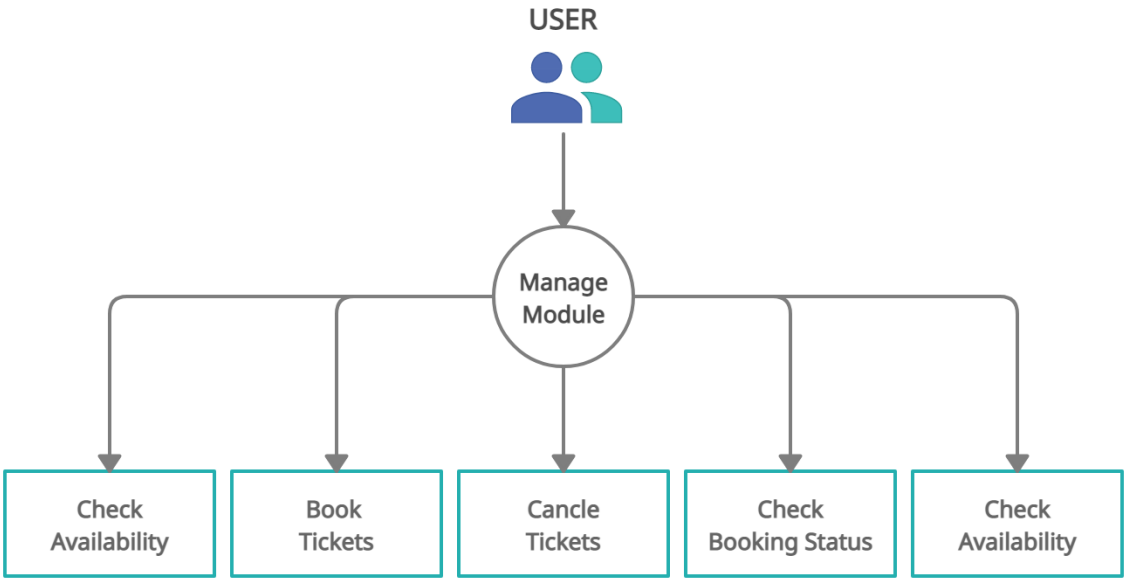
Fig 2: Data Flow Diagram of Railway Reservation system

3.1.3 Overview of Modules

a) Admin Module



b) User Module



Chapter 4:

IMPLEMENTATION

4.1 OVERVIEW OF MODULES / COMPONENTS/DETAIL DESIGN

In This reservation system

Main Menu has 3 modules.

1. **Admin access**
2. **User access**
3. **Exit**

"Admin access" In the admin Module. First it asks the admin password for the security purpose, after entering the password it will give us 7 sections.

The admin access Module has 7 sections.

1. **Create new train file**
2. **New record**
3. **Modify a train record**
4. **Go back**
5. **Delete a train record**
6. **Display all the current records**
7. **Display passenger list**

Section 1:

Create new train file This function is used to create new Train Record file. in this it will ask to give input. train number, train name, starting point, destination point and also it will ask to give number of AC tickets available and it's cost for each seat, number of Sleeper tickets available and it's cost for each seat, number of Second-class tickets available and its cost. Later it will ask the duration and distance in Km. After complete all these it will ask to continue or to stop the new record if we give yes, it will continue otherwise it go back.

Section 2:

New Record This function is used to add new record to the file. And ask to give input to record. train number, train name, starting point, destination point and also it will ask to give AC tickets available and it's cost for each seat, Sleeper tickets available and it's cost for each seat, Second class tickets available and its cost. Later it will ask the duration and distance. After complete these it will ask to continue or to stop the new record if we give yes, it will continue otherwise it go back.

Section 3:

Modify a train record This function is used to modify the data of the train which is already existing in the file or existing record. And ask for input train number, train name, starting point, destination point and also it will ask to give AC tickets available and its cost for each seat, Sleeper tickets available and its cost for each seat, Second class tickets available and its cost. Later it will ask the duration and distance. You should change the data which you need to modify.

Section 4:

Go back This function is used to go back to menu section

Section 5:

Delete a train record This function is used to delete the record. just it will ask to input the train number, which you want to delete if you enter correctly, it will display the details of the train, and it will Confirm once again to delete it, if we are sure then the data will be removed.

Section 6:

Display all current records This function is used to display all the current details of the trains which are existing in the file one by one.

Section 7:

Display passenger list This function is used to display the all the details of the passengers in every train.

"User access" In this User access module it helps the user/passenger. To Book the tickets

The user access Module has 4 sections.

1. **To check availability**
2. **To book tickets**
3. **For cancellation**
4. **Check Status**
5. **To go back**

Section 1:

To check availability This function is used to check the availability of train, it asks the input for starting point and destination point, if we give Correct input, it will show the details of the train and also will show the Train number.

Section 2:

To Book Tickets This function has 2 other sections

1. **To book ticket**
2. **To go back**

1st section: it asks the user input for booking, boarding point and destination, if we give Correct it will give the details of train, later it asks the train number, if we give correct, it will ask the number of tickets to book and it will ask the class. Later it will ask the details of the passengers like name, age, gender. and automatically it will generate the PNR Number and the train details and passenger details.

2nd section: it will go back to the previous section (to book tickets)

Section 3:

For Cancellation This function is used for cancel the, ticket, it asks the user input, PNR Number, after entering it will show the details of the train and passenger later it asks for confirmation of cancel, if we give Yes, the ticket will be cancelled or else it will be stored.

Section 4:

Check Status This function is used for check the booking status , Whether the Ticket has been booked successfully or not, based on the entered valid PNR number if correct it displays the Passenger details and there Booking details.

Section 4:

To go back This function is used to go back to the main menu.

"EXIT" This module is used to exit the program.

4.2 SOURCE CODE

```
#include <iostream>
#include <fstream>
#include <conio.h>
#include <stdio.h>
#include <process.h>
#include <string.h>
#include <iomanip>

using namespace std;
class pn
{
public:
   int pnr;
} x;
void star();       // FUNCTION TO PRINT STARS ON SCREEN
void dash_board(); // FUNCTION USED FOR USER MODULE
class train
{
public:
   int tno, ac, sleeper, sc, dist, fac, fsleeper, fsc;
```

```
    float time;
    char from[100], to[100], name[100];
public:
    void modify();   // FUNCTION TO MODIFY TRAIN RECORDS
    void del();      // FUNCTION TO DELETE TRAIN RECORDS
    void display();  // FUNCTION TO DISPLAY TRAIN RECORDS
    void display1(); // FUNCTION TO DISPLAY TRAIN RECORDS IN ORDER
    void acp();      //      FUNCTION TO ACCEPT TRAIN RECORDS
    void check();    //     FUNCTION TO CHECK AVAILABILITY
    int update(int); //     FUNCTION TO UPDATE NO. OF SEATS AFTER CANCELLATION
    int ret()        // ACCESSOR FUNCTION TO RETURN TRAIN NUMBER
    {
       return tno;
    }
    char *f1() // ACCESSOR FUNCTION TO RETURN STARTING POINT
    {
       return from;
    }
    char *t1() // ACCESSOR FUNCTION TO RETURN DESTINATION
    {
       return to;
    }
} t;        // END OF CLASS
void add();  // FUNCTION TO ADD TRAIN RECORDS
void disp(); // FUNCTION TO DISPLAY TRAIN RECORDS
class book
{
public:
    int no, tickets, pnr;
    char pto[100], pfrom[100], clas[20];
    long int amt;
    int retpnr() // ACCESSOR FUNCTION TO RETURN PNR NUMBER
```

```
    {
        return pnr;
    }
    int rettic() // ACCESSOR FUNCTION TO RETURN TICKETS
    {
        return tickets;
    }
    int retamt() // ACCESSOR FUNCTION TO RETURN AMOUNT
    {
        return amt;
    }
    void get();         // FUNCTION TO GET BOOKING DETAILS
    int assign();       // FUNCTION TO CALCULATE AND ASSIGN BOOKING DETAILS
    void show();        // FUNCTION TO DISPLAY BOOKING DETAILS
    void cancel();      // FUNCTION TO CANCEL BOOKING
    void check_status(); // CHECK THE STATUS OF PASSENGER BOOKING
    void list_pass();   // SEND INDIVIDUAL PNR NUMBER TO DISPLAY FUNCTION
    void list_disp(int); // GENERATE PASSNGER LIST FOR ADMIN SECTION
} b;                    // END OF CLASS
class passenger
{
public:
    int pnr, page;
    char psex, pname[100];
    string date;
    void getp();    // FUNCTION TO GET PASSENGER DETAILS
    void showp(int); //  FUNCTION TO SHOW PASSENGER DETAILS
} p;                // END OF CLASS
void admin();       // ADMIN SECTION TO MANIPULATE TRAIN RECORDS
void menu();        // STARTING OF RAILWAY RESERVATION PROGRAM
int main() // STARTING MAIN FUNCTION
{
```

```
    menu(); // STARTING OF PROGRAM
    return 0;
} // END OF MAIN
void menu()
{
    int n;
    do
    {
        system("cls");
        cout << "\n\n\n\n";
        star();
        cout << "\t\t\t\t\t  WELCOME TO INDIAN RAILWAYS" << endl;
        star();
        cout << "\n\t\t\t\t\t1.ADMIN ACCESS" << endl;
        cout << "\t\t\t\t\t2.USER ACCESS" << endl;
        cout << "\t\t\t\t\t3.EXIT\n" << endl;
        star();
        cout << "\t\t\t\t\t  YOUR CHOICE : ";
        cin >> n;
        switch (n)
        {
        case 1:
            admin();
            break;
        case 2:
            dash_board();
            break;
        case 3:
            exit(0);
        }
    } while (n != 3);
}
```

```
void dash_board()
{
   int n;
   while (1)
   {
      system("cls");
      cout << "\n\n\n";
      star();
      cout << "\t\t\t\t\t  WELCOME TO INDIAN RAILWAYS" << endl;
      star();
      cout << "\n\t\t\t\t\t1.TO CHECK AVAILABILITY" << endl;
      cout << "\t\t\t\t\t2.TO BOOK TICKETS" << endl;
      cout << "\t\t\t\t\t3.FOR CANCELLATION" << endl;
      cout << "\t\t\t\t\t4.CHECK STATUS" << endl;
      cout << "\t\t\t\t\t5.RETURN\n\n";
      star();
      cout << "\t\t\t\t\t  YOUR CHOICE : ";
      cin >> n;
      system("cls");
      switch (n)
      {
      case 1:
         t.check();
         getch();
         break;
      case 2:
         b.get();
         getch();
         break;
      case 3:
         b.cancel();
         getch();
```

```
            break;
        case 4:
            b.check_status();
            break;
        case 5:
            menu();
            break;
        default:
            system("cls");
            star();
            cout << "\t\t\t\tWRONG CHOICE" << endl;
            star();
            getch();
        } // END OF SWITCH
    }     // END OF WHILE
}
void add() // FUNCTION TO WRITE TRAIN RECORDS
{
    system("cls");
    star();
    fstream rr;
    rr.open("train.dat", ios::app | ios::in | ios::binary);
    char ch;
    do
    {
        system("cls");
        star();
        t.acp();
        rr.write((char *)&t, sizeof t);
        cout << "\t\t\t\t\t  MORE RECORDS ? (Y/N) : ";
        cin >> ch;
    } while (ch == 'y' || ch == 'Y');
```

```
    star();
    rr.close();
} // END OF FUNCTION
void disp() // FUNCTION TO DISPLAY TRAIN RECORDS
{
    fstream rr;
    rr.open("train.dat", ios::in | ios::binary);
    system("cls");
    cout << "\n\n\n";
    star();
    cout << "\t\t\t\t\t   TRAIN DETAILS" << endl;
    star();
    cout << "\t\t  TRAIN NO\t\tTRAIN NAME\t     FROM\t    TO\t      TIME" << endl;
    star();
    while (rr.read((char *)&t, sizeof t))
    {
        // system("cls");
        t.display1();
    }
    cout << "\n";
    star();
    rr.close();
} // END OF FUNCTION
void train::display() // FUNCTION TO DISPLAY TRAIN RECORD
{
    cout << "\t\t\t\t\t\tTRAIN DETAILS" << endl;
    star();
    cout << "\n\t\t\t\t\tTRAIN NUMBER : " << tno << endl;
    cout << "\t\t\t\t\tTRAIN NAME : " << name << endl;
    cout << "\t\t\t\t\tSTARTING POINT : " << from << endl;
    cout << "\t\t\t\t\tDESTINATION : " << to << endl;
    cout << "\t\t\t\t\tAC TICKETS AVAILABLE : " << ac << endl;
```

```
    cout << "\t\t\t\t\tPRICE OF EACH AC TICKET Rs. : " << fac << endl;
    cout << "\t\t\t\t\tSLEEPER CLASS TICKETS AVAILABLE : " << sleeper << endl;
    cout << "\t\t\t\t\tPRICE OF EACH SLEEPER CLASS TICKET Rs. : " << fsleeper << endl;
    cout << "\t\t\t\t\tSECOND CLASS TICKETS AVAILABLE : " << sc << endl;
    cout << "\t\t\t\t\tPRICE OF EACH SECOND CLASS TICKET Rs. : " << fsc << endl;
    cout << "\t\t\t\t\tDURATION OF JOURNEY (hrs) : " << time << endl;
    cout << "\t\t\t\t\tTOTAL DISTANCE(km) : " << dist << endl;
    cout << endl;
    star();
    getch();
}
void train::display1() // FUNCTION TO DISPLAY TRAIN RECORD
{
    cout << "\t\t     " << tno << setw(29) << name << setw(15) << from << setw(12) << to <<
setw(10) << time << " hr" << endl;
}
void train::acp() // FUNCTION TO ACCEPT TRAIN RECORDS
{
    system("cls");
    cout << "\n\n\n";
    star();
    cout << "\t\t\t\t\t  ENTER TRAIN DETAILS" << endl;
    star();
    cout << "\n\t\t\t\tTRAIN NUMBER : ";
    cin >> tno;
    cout << "\t\t\t\tTRAIN NAME : ";
    cin >> name;
    cout << "\t\t\t\tSTARTING POINT : ";
    cin >> from;
    cout << "\t\t\t\tDESTINATION : ";
    cin >> to;
invalid_5:
```

```
    cout << "\t\t\t\tNUMBER OF AC TICKETS TO BE MADE AVAILABLE  : ";
    cin >> ac;
    if (ac <= 0)
    {
        star();
        cout << "\t\t\t\tNUMBER OF TICKET SHOULD BE GREATER THAN 0\n";
        star();
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
        goto invalid_5;
    }
invalid_6:
    cout << "\t\t\t\tPRICE OF EACH AC TICKET Rs. : ";
    cin >> fac;
    if (fac <= 0)
    {
        star();
        cout << "\t\t\t\tPRICE SHOULD BE GREATER THAN 0\n";
        star();
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
        goto invalid_6;
    }
invalid_7:
    cout << "\t\t\t\tNUMBER OF SLEEPER TICKETS TO BE MADE AVAILABLE : ";
    cin >> sleeper;
    if (sleeper <= 0)
    {
```

```
        star();
        cout << "\t\t\t\tNUMBER OF TICKET SHOULD BE GREATER THAN 0\n";
        star();
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
        goto invalid_7;
    }
invalid_8:
    cout << "\t\t\t\tPRICE OF EACH SLEEPER TICKET Rs. : ";
    cin >> fsleeper;
    if (fsleeper <= 0)
    {
        star();
        cout << "\t\t\t\tPRICE SHOULD BE GREATER THAN 0\n";
        star();
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
        goto invalid_8;
    }
invalid_9:
    cout << "\t\t\t\tNUMBER OF SECOND CLASS TICKETS TO BE MADE AVAILABLE :
";
    cin >> sc;
    if (sc <= 0)
    {
        star();
        cout << "\t\t\t\tNUMBER OF TICKET SHOULD BE GREATER THAN 0\n";
        star();
```

```
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
        goto invalid_9;
    }
invalid_10:
    cout << "\t\t\t\tPRICE OF EACH SECOND CLASS TICKET Rs. : ";
    cin >> fsc;
    if (fsc <= 0)
    {
        star();
        cout << "\t\t\t\tPRICE SHOULD BE GREATER THAN 0\n";
        star();
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
        goto invalid_10;
    }
invalid_11:
    cout << "\t\t\t\tDURATION OF JOURNEY(hrs)  : ";
    cin >> time;
    if (time <= 0)
    {
        star();
        cout << "\t\t\t\tNUMBER OF HOUR'S SHOULD BE GREATER THAN 0\n";
        star();
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
```

```
        goto invalid_11;
    }
invalid_12:
    cout << "\t\t\t\tTOTAL DISTANCE (km) : ";
    cin >> dist;
    if (dist <= 0)
    {
        star();
        cout << "\t\t\t\tDISTANCE SHOULD BE GREATER THAN 0\n";
        star();
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
        goto invalid_12;
    }
    cout << "\n";
    star();
} // END OF FUNCTION
void train::modify() // FUNCTION TO MODIFY TRAIN RECORD
{
    system("cls");
    cout << "\n\n\n";
    star();
    cout << "\t\t\t\t\t   TRAIN MODIFICATION" << endl;
    star();
    fstream rr;
    rr.open("train.dat", ios::in | ios::out | ios::binary);
    int id, f = 0, n = 0, p;
    cout << "\t\t\t\tTRAIN NUMBER OF TRAIN TO BE MODIFIED : ";
    cin >> id;
    while (rr.read((char *)&t, sizeof t))
```

```
    {
        n++;
        if (id == t.ret())
        {
            acp();
            p = (n - 1) * (sizeof t);
            rr.seekp(p, ios::beg);
            rr.write((char *)&t, sizeof t);
            f = 1;
        }
    }
    if (f == 0)
    {
        star();
        cout << "\t\t\t\t\t   SORRY RECORD NOT FOUND !!" << endl;
        star();
    }
    rr.close();
} //END OF FUNCTION
void train::del() // FUNCTION TO DELETE TRAIN RECORDS
{
    system("cls");
    cout << "\n\nn\n";
    star();
    cout << "\t\t\t\t\t\t DELETE TRAIN" << endl;
    star();
    int id, f = 0;
    fstream rr;
    rr.open("train.dat", ios::in | ios::binary);
    fstream t1;
    t1.open("temp.dat.dat", ios::out | ios::binary);
    cout << "\t\t\t\t\tTRAIN NUMBER TO BE DELETED : ";
```

```
cin >> id;
star();
while (rr.read((char *)&t, sizeof t))
{ // 1
   if (id == t.ret())
   { // 2
      char ch;
      f = 1;
      cout << "\n\n\n";
      t.display();
      cout << "\t\t\t\tARE YOU SURE ? (Y/N) : ";
      cin >> ch;
      if (ch == 'y' || ch == 'Y')
      {
         star();
         cout << "\t\t\t\tRECORD DELETED";
      }
      else
      {
         t1.write((char *)&t, sizeof t);
      }
   } // 2
   else
      t1.write((char *)&t, sizeof t);
} // 1
if (f == 0)
{
   cout << "\t\t\t\t\t   SORRY RECORD NOT FOUND !!" << endl;
   star();
}
rr.close();
t1.close();
```

```
    remove("train.dat");
    rename("temp.dat.dat", "train.dat");
} // END OF FUNCTION
void train::check() // FUNCTION TO CHECK TICKET AVAILABILITY
{
    int count = 0;
    cout << "\n\n\n";
    disp();
    cout << "\t\t\t\t\t  CHECK AVAILABILITY" << endl;
    star();
    char T[100], f[100];
    int flag = 0;
    cout << "\t\t\t\tTHE STARTING POINT : ";
    cin >> f;
    cout << "\t\t\t\tTHE DESTINATION : ";
    cin >> T;
    fstream rr;
    rr.open("train.dat", ios::in | ios::binary);
    while (rr.read((char *)&t, sizeof t))
    {
        if (strcmpi(f, t.f1()) == 0 && strcmpi(T, t.t1()) == 0)
        {
            system("cls");
            cout << "\n\n\n";
            star();
            flag = 1;
            t.display();
            count++;
        }
    } // END OF WHILE
    if (flag == 0)
    {
```

```
        star();
        cout << "\t\t\t\t\t  SORRY NO AVAILABLE TRAINS FOUND" << endl;
        star();
    }
    else
    {
        cout << "\t\t\t\t\t  NUMBER OF TRAINS ARE : " << count << endl;
        star();
    }
} //END OF FUNCTION
int train::update(int q) // FUNCTION TO UPDATE TICKETS AFTER CANCELLATION
{
    fstream rr;
    int n, tic, charge;
    rr.open("booking.dat", ios::in | ios::out | ios::binary);
    while (rr.read((char *)&b, sizeof b))
    {
        if (b.pnr == q)
        {
            n = b.no;
            tic = b.tickets;
            rr.close();
            rr.open("train.dat", ios::in | ios::out | ios::binary);
            rr.seekg(0, ios::beg);
            int f = 0, p;
            while (rr.read((char *)&t, sizeof t))
            {
                f++;
                if (t.tno == n) //1
                {
                    p = (f - 1) * (sizeof(t));
                    rr.seekp(p, ios::beg);
```

```
                if (strcmpi(b.clas, "ac") == 0)
                {
                   t.ac += tic;
                   rr.write((char *)&t, sizeof t);
                   charge = 1;
                }
                else if (strcmpi(b.clas, "sl") == 0)
                {
                   t.sleeper += tic;
                   rr.write((char *)&t, sizeof t);
                   charge = 2;
                }
                else
                {
                   t.sc += tic;
                   rr.write((char *)&t, sizeof t);
                   charge = 3;
                }
              } //1
          }     // END OF WHILE 2
          rr.close();
          break;
       } // END OF IF
   }     // END OF WHILE 1
   return charge;
} // END OF FUNCTION
void book::get() // FUNCTION TO BOOK TICKETS
{
   cout << "\n\n\n";
   int n = 1;
   while (n == 1)
   { // WHILE 1
```

```
system("cls");
cout << "\n\n\n";
star();
cout << "\t\t\t\t\t  WELCOME TO BOOKING MENU" << endl;
star();
cout << "\t\t\t\t\t 1.TO BOOK TICKETS" << endl;
cout << "\t\t\t\t\t 2.TO GO BACK" << endl;
star();
cout << "\t\t\t\t\t  YOUR CHOICE : ";
cin >> n;
if (n == 1)
{ //          IF 1
   system("cls");
   cout << "\n\n\n";
   star();
   cout << "\t\t\t\t\t    BOOKING SECTION" << endl;
   star();
   int flag = 0;
   cout << "\t\t\t\t\tTHE BOARDING POINT : ";
   cin >> pfrom;
   cout << "\t\t\t\t\tTHE DESTINATION : ";
   cin >> pto;
   cout << endl;
   fstream rr;
   rr.open("train.dat", ios::in | ios::binary);
   fstream pas;
   pas.open("passenger.dat", ios::app | ios::binary);
   rr.seekg(0, ios::beg);
   while (rr.read((char *)&t, sizeof t))
   { // WHILE 2
      if (strcmpi(pfrom, t.f1()) == 0 && strcmpi(pto, t.t1()) == 0)
      {
```

```
        system("cls");
        cout << "\n\n\n";
        star();
        flag = 1;
        t.display();
    }
} // END OF WHILE 2
rr.close();
if (flag == 0)
{
    star();
    cout << "\t\t\t\t\tSORRY NO AVAILABLE TRAINS FOUND " << endl;
    star();
    getch();
}
// STARTING THE BOOKING PROCESS
if (flag == 1)
{ //  IF 2
    int f1 = 0;
    system("cls");
    cout << "\n\n\n";
    star();
    cout << "\t\t\t\t\t    BOOKING SECTION" << endl;
    star();
    cout << "\t\t\t\tTHE TRAIN NUMBER TO BOOK TICKETS : ";
    cin >> no;
    rr.open("train.dat", ios::in | ios::binary);
    rr.seekg(0, ios::beg);
    while (rr.read((char *)&t, sizeof t))
    { // WHILE 3
        if (no == t.ret())
        {
```

```
            f1 = 1;
        invalid_1:
            cout << "\t\t\t\tTHE NUMBER OF TICKETS TO BOOK(MAX .6) : ";
            cin >> tickets;
            if (tickets <= 0 || tickets > 6)
            {
                star();
                cout << "\t\t\t\t\tSORRY CANNOT INVALID INPUT\n";
                star();
                getch();
                system("cls");
                cout << "\n\n\n";
                star();
                goto invalid_1;
            }
        invalid_2:
            cout << "\t\t\t\tTHE CLASS (AC : ac / SLEEPER : sl / SECOND CLASS : sc)
: ";
            cin >> clas;
            if (strcmpi(clas, "ac") != 0 && strcmpi(clas, "sl") != 0 && strcmpi(clas, "sc")
!= 0)
            {
                star();
                cout << "\t\t\t\tSORRY YOU HAVE ENTERD INCORRECT CLASS" <<
endl;
                star();
                getch();
                system("cls");
                cout << "\n\n\n";
                star();
                goto invalid_2;
            }
            int z;
```

```
            z = b.assign(); // CALLING THE FUNCTION TO ASSIGN
            // BOOKING DETAILS
            if (z == 0)
            {
               star();
               cout << "\t\t\t\tSORRY CANNOT BOOK " << tickets << " TICKETS ";
               star();
               getch();
               break;
            }
            else
            {
               for (int i = 0; i < tickets; i++)
               {
                  system("cls");
                  star();
                  cout << "\t\t\t\t\t  DETAILS OF PASSENGER " << (i + 1) << endl;
                  p.getp();
                  p.pnr = b.pnr;
                  pas.write((char *)&p, sizeof p);
                  star();
               }
               pas.close();
               rr.close();
               b.show();
               p.showp(b.pnr);
               getch();
               break;
            }
         } // END OF IF 3
      }     // END OF WHILE 3
      if (f1 == 0)
```

```
            {
               star();
               cout << "\t\t\t\tSORRY YOU HAVE ENTERD INCORRECT TRAIN NUMBER" << endl;
               star();
               getch();
            }
         } // END OF IF 2
         pas.close();
         rr.close();
      } // END OF IF 1
   }    // END OF WHILE 1
} // END OF FUNCTION
int book::assign() // FUNCTION TO CALCULATE AND ASSIGN BOOKING DETAILS
{
   int n = 0, p;
   fstream rr;
   rr.open("train.dat", ios::in | ios::out | ios::binary);
   rr.seekg(0, ios::beg);
   while (rr.read((char *)&t, sizeof t))
   {
      n++;
      if (no == t.ret())
      {
         if (strcmpi(clas, "ac") == 0)
         {
            amt = tickets * t.fac;

            p = (n - 1) * sizeof(t);
            rr.seekp(p, ios::beg);

            t.ac = t.ac - tickets;
```

```
            if (t.ac < 0)
            {
                return 0;
            }
            else
                rr.write((char *)&t, sizeof t);
        }
        else if (strcmpi(clas, "sl") == 0)
        {
            amt = tickets * t.fsleeper;
            p = (n - 1) * sizeof(t);
            rr.seekp(p, ios::beg);
            t.sleeper = t.sleeper - tickets;
            if (t.sleeper < 0)
                return 0;
            else
                rr.write((char *)&t, sizeof t);
        }
        else
        {
            amt = tickets * t.fsc;
            p = (n - 1) * sizeof(t);
            rr.seekp(p, ios::beg);
            t.sc = t.sc - tickets;
            if (t.sc < 0)
                return 0;
            else
                rr.write((char *)&t, sizeof t);
        }
        break;
    }
}
```

```
fstream b1;
b1.open("booking.dat", ios::app | ios::binary);
fstream rr2;
rr2.open("pnr.dat", ios::in | ios::binary);
if (!rr2)
{
   rr2.close();
   rr2.open("pnr.dat", ios::in | ios::app | ios::binary);
   rr2.seekp(0, ios::beg);
   x.pnr = 1000;
   rr2.write((char *)&x, sizeof x);
   rr2.close();
}
else
{
   int d;
   rr2.seekg(0, ios::beg);
   while (rr2.read((char *)&x, sizeof x))
   {
      d = x.pnr;
   }
   rr2.close();
   fstream r2;
   r2.open("pnr.dat", ios::app | ios::binary);
   x.pnr = ++d;
   r2.write((char *)&x, sizeof x);
   r2.close();
}
b.pnr = x.pnr;
b1.write((char *)&b, sizeof b);
b1.close();
return 1;
```

```
} //END OF FUNCTION
void book::show() // FUNCTION TO DISPLAY BOOKING DETAILS
{
   system("cls");
   cout << "\n\n\n";
   star();
   cout << "\t\t\t\t\t\tTICKET" << endl;
   star();
   cout << "\t\t\t\t  PNR NUMBER              : " << pnr << endl;
   cout << "\t\t\t\t  TRAIN NUMBER                  : " << no << endl;
   cout << "\t\t\t\t  TOTAL PASSENGERS   : " << tickets << endl;
   cout << "\t\t\t\t  TOTAL AMOUNT                  : " << amt << endl;
   cout << "\t\t\t\t  BOARDING POINT       : " << pfrom << endl;
   cout << "\t\t\t\t  DESTINATION             : " << pto << endl;
   star();
   cout << "\t\t\t\t   YOUR TICKET HAS BEEN RESERVED" << endl;
} // END OF FUNCTION
void book::cancel() // FUNCTION TO CANCEL BOOKING
{
   int C_charge, C_temp;
   cout << "\n\n\n";
   star();
   cout << "\t\t\t\t\t  CANCELLATION" << endl;
   star();
   train tr;
   fstream b1;
   b1.open("booking.dat", ios::in | ios::binary);
   fstream rr;
   rr.open("passenger.dat", ios::in | ios::binary);
   fstream t;
   t.open("temp.dat", ios::out | ios::binary);
   fstream tmp;
```

```
tmp.open("t2.dat", ios::out | ios::binary);
int id;
int flag = 0, f = 0;
char ch;
cout << "\t\t\t\t YOUR PNR NUMBER : ";
cin >> id;
while (b1.read((char *)&b, sizeof b))
{
   if (id == b.retpnr())
   {
      system("cls");
      cout << "\n\n\n";
      star();
      cout << "\t\t\t\t\t  CANCELLATION" << endl;
      star();
      flag = 1;
      f = 1;
      cout << "\t\t\t\t  PNR NUMBER              : " << b.retpnr() << endl;
      cout << "\t\t\t\t  TOTAL TICKETS           : " << b.rettic() << endl;
      cout << "\t\t\t\t  TOTAL AMOUNT            : " << b.retamt() << endl;
      cout << "\t\t\t\t  BOARDING POINT       : " << pfrom << endl;
      cout << "\t\t\t\t  DESTINATION             : " << pto << endl;

      p.showp(id);
      cout << "\t\t\t\t\tARE YOU SURE ? (Y/N) : ";
      cin >> ch;
      if (ch == 'y' || ch == 'Y')
      {
         C_temp = tr.update(id);
         if (C_temp == 1)
         {
            C_charge = amt * 0.2;
```

```
            }
            else if (C_temp == 2)
            {
               C_charge = amt * 0.15;
            }
            else
            {
               C_charge = amt * 0.1;
            }
            cout << "\n\t\t\t\tAFTER CANCELLATION AMOUNT : " << (amt - C_charge) <<
endl;
            star();
            cout << "\t\t\t\t  YOUR BOOKING HAS BEEN CANCELLED" << endl;
            star();
         }
         else
         {
            flag = 0;
            t.write((char *)&b, sizeof b);
         }
      }
      else
      {
         t.write((char *)&b, sizeof b);
      }
   }
   t.close();
   tmp.close();
   b1.close();
   remove("booking.dat");
   rename("temp.dat", "booking.dat");
   if (flag == 1)
```

```
    {
        fstream tmp;
        tmp.open("t2.dat", ios::out | ios::binary);
        rr.seekg(0, ios::beg);
        while (rr.read((char *)&p, sizeof p))
        {
            if (id != b.retpnr())
                tmp.write((char *)&p, sizeof p);
        }
        rr.close();
        tmp.close();
        remove("passenger.dat");
        rename("t2.dat", "passenger.dat");
        flag = 0;
    }
    if (f == 0)
    {
        star();
        cout << "\t\t\t\t\t SORRY INVALID PNR" << endl;
        star();
        remove("temp.dat");
        remove("t2.dat");
    }
    rr.close();
    tmp.close();
    b1.close();
    t.close();
} // END OF FUNCTION
void book::check_status()
{
    cout << "\n\n\n";
    star();
```

```
cout << "\t\t\t\t\t  BOOKING STATUS" << endl;
star();
train tr;
fstream b1;
b1.open("booking.dat", ios::in | ios::binary);
fstream rr;
rr.open("passenger.dat", ios::in | ios::binary);
fstream tt;
tt.open("train.dat", ios::in | ios::binary);
int id;
int flag = 0, f = 0;
char ch;
cout << "\t\t\t\t YOUR PNR NUMBER : ";
cin >> id;
while (b1.read((char *)&b, sizeof b))
{
   if (id == b.retpnr())
   {
      flag = 1;
      while (tt.read((char *)&t, sizeof t))
      { // WHILE 2
         if (strcmpi(pfrom, t.f1()) == 0 && strcmpi(pto, t.t1()) == 0)
         {
            system("cls");
            cout << "\n\n\n";
            star();
            cout << "\t\t\t\t\t  BOOKING STATUS" << endl;
            star();
            flag = 1;
            f = 1;
            cout << "\t\t\t\t  PNR NUMBER        : " << b.retpnr() << endl;
            cout << "\t\t\t\t  TRAIN NUMBER              : " << t.tno << endl;
```

```
                cout << "\t\t\t\t  TRAIN NAME        : " << t.name << endl;
                cout << "\t\t\t\t  TOTAL TICKETS             : " << b.rettic() << endl;
                cout << "\t\t\t\t  TOTAL AMOUNT              : " << b.retamt() << endl;
                cout << "\t\t\t\t  BOARDING POINT       : " << pfrom << endl;
                cout << "\t\t\t\t  DESTINATION               : " << pto << endl;
                p.showp(id);
              }
            } // END OF WHILE 2
        }
    }
    if (flag == 0)
    {
        star();
        cout << "\t\t\t\t\t  SORRY INVALID PNR" << endl;
        star();
    }
    b1.close();
    rr.close();
    tt.close();
    getch();
}
void passenger::getp()
{
    star();
    cout << "\t\t\t\t\tNAME : ";
    cin >> pname;
invalid_3:
    cout << "\t\t\t\t\tAGE : ";
    cin >> page;
    if (page < 10)
    {
        star();
```

```
        cout << "\t\t\t\tINVALID AGE SHOULD BE GREATER THAN 10\n";
        star();
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
        goto invalid_3;
    }
invalid_4:
    cout << "\t\t\t\t\tSEX (M/F) : ";
    cin >> psex;
    if (psex != 'm' && psex != 'M' && psex != 'f' && psex != 'F')
    {
        star();
        cout << "\t\t\t\tINVALID SEX SHOULD BE (M or F)\n";
        star();
        getch();
        system("cls");
        cout << "\n\n\n";
        star();
        goto invalid_4;
    }
} // END OF FUNCTION
void passenger::showp(int q) // FUNCTION TO SHOW PASSENGER DETAILS
{
    int i = 14;
    fstream rr;
    rr.open("passenger.dat", ios::in | ios::binary);
    star();
    cout << "\t\t\t\t\t   PASSENGER LIST" << endl;
    star();
    cout << "\t\t\t   PASSENGER NAME\tPASSENGER AGE\tPASSENGER SEX " << endl;
```

```
   while (rr.read((char *)&p, sizeof p))
   {
      if (p.pnr == q)
      {
         cout << "\t\t" << setw(20) << pname << setw(18) << page << setw(18) << psex << endl;
      }
   }
   star();
   rr.close();
} // END OF FUNCTION
void admin() // FUNCTION TO MANIPULATE TRAIN RECORDS
{
   system("cls");
   cout << "\n\n\n";
   star();
   int m;
   char p[20];
   cout << "\t\t\t\t\t\tADMIN LOGIN" << endl;
   star();
   cout << "\n\t\t\t\tTHE PASSWORD : ";
   for (int i = 0; i < 20; i++)
   {
      p[i] = getch();
      if (p[i] == 13)
      {
         p[i] = '\0';
         break;
      }
      else
         cout << "*";
   }
```

```
if (strcmp(p, "railway123") == 0)
{
   do
   {
      system("cls");
      cout << "\n\n\n";
      star();
      cout << "\t\t\t\t\t\tADMIN SECTION" << endl;
      star();
      cout << "\n\t\t\t\t\t1.CREATE NEW TRAIN FILE" << endl;
      cout << "\t\t\t\t2.NEW RECORD " << endl;
      cout << "\t\t\t\t3.MODIFY A TRAIN RECORD " << endl;
      cout << "\t\t\t\t4.GO BACK " << endl;
      cout << "\t\t\t\t5.DELETE A TRAIN RECORD " << endl;
      cout << "\t\t\t\t6.DISPLAY ALL THE CURRENT RECORDS " << endl;
      cout << "\t\t\t\t7.DISPLAY PASSENGER LIST\n\n";
      star();
      cout << "\t\t\t\tYOUR CHOICE : ";
      cin >> m;
      switch (m)
      {
      case 1:
         remove("train.dat");
         remove("booking.dat");
         remove("passenger.dat");
         remove("temp.dat");
         remove("t2.dat");
         add();
         getch();
         break;
      case 2:
         add();
```

```
            getch();
            break;
        case 3:
            t.modify();
            getch();
            break;
        case 4:
            break;
        case 5:
            t.del();
            getch();
            break;
        case 6:
            disp();
            getch();
            break;
        case 7:
            b.list_pass();
            getch();
            break;
        }
    } while (m != 4);
} // END OF IF
else
{
    system("cls");
    star();
    cout << "\t\t\t\t\t\tWRONG PASSWORD" << endl;
    cout << "\t\t\t\t\t\tACCESS DENIED" << endl;
    star();
    getch();
}
```

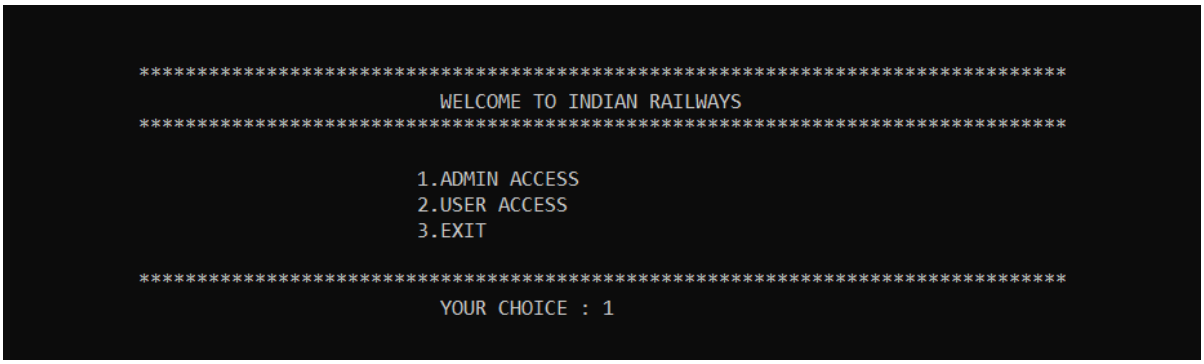
```
} // END OF FUNCTION
void star() // FUNCTION TO PRINT STARS ON SCREEN
{
   cout << "\t\t";
   for (int i = 1; i < 81; i++)
   {
      cout << "*";
   }
   cout << endl;
}
void book::list_pass() // FUNCTION TO GET ALL PASSENGER LIST
{
   system("cls");
   fstream b1;
   b1.open("booking.dat", ios::in | ios::binary);
   cout << "\n\n\n";
   star();
   cout << "\t\t\t\t\t    PASSENGER LIST" << endl;
   star();
   cout << "\t\t PNR_NO TRAIN NO PASSENGER NAME AGE SEX
FROM\tDESTINATION\n\n";
   while (b1.read((char *)&b, sizeof b))
   {
      b.list_disp(b.retpnr());
   }
   star();
   b1.close();
}
void book::list_disp(int q) // FUNCTION TO DISPLAY ALL PASSENGER LIST
{
   int i = 14, train_no;
   fstream rr;
```

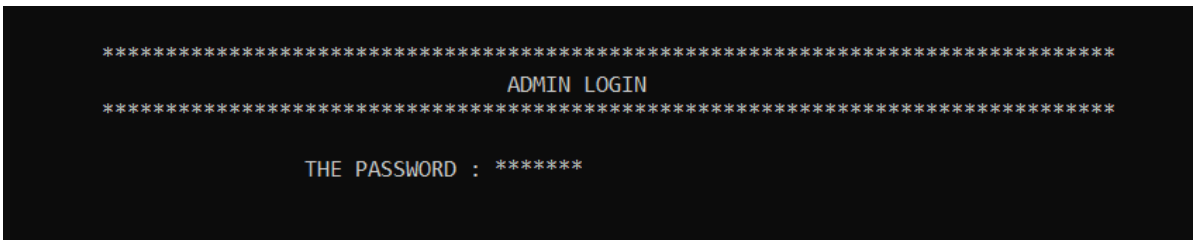
```
    rr.open("passenger.dat", ios::in | ios::binary);
    fstream tt;
    tt.open("train.dat", ios::in | ios::binary);
    while (rr.read((char *)&p, sizeof p))
    {
       if (q == b.retpnr() && q == p.pnr)
       {
          while (tt.read((char *)&t, sizeof t))
          { // WHILE 2
             if (strcmpi(pfrom, t.f1()) == 0 && strcmpi(pto, t.t1()) == 0)
             {
                train_no = t.tno; // GETTING TRAIN NUMBER BASED ON PNR NUMBER
             }
          } // END OF WHILE 2
          cout << "\t\t      " << p.pnr << "    " << train_no << setw(18) << p.pname << setw(8) <<
p.page << setw(6) << p.psex << setw(10) << pfrom << setw(13) << pto << "\n";
       }
    }
    cout << endl;
    rr.close();
    tt.close();
}
```

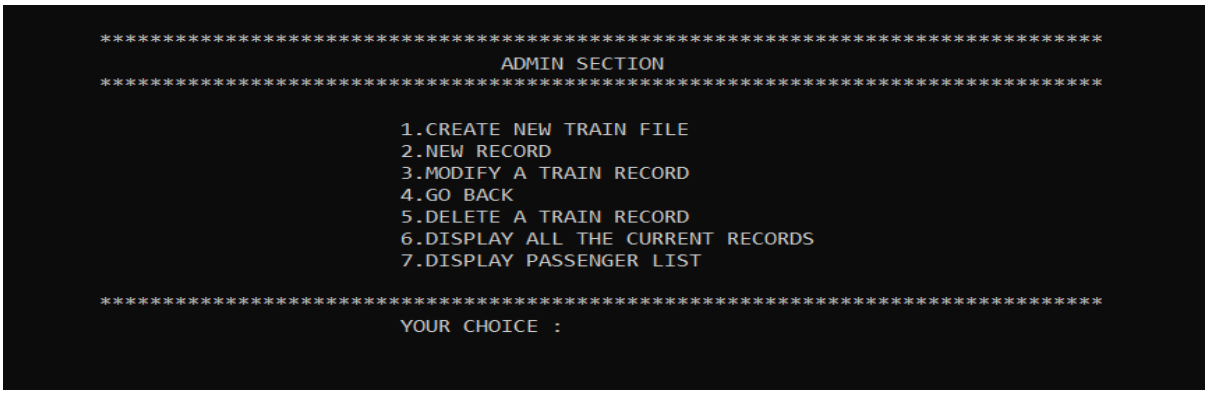
CHAPTER 5:

RESULTS

5.1 Output Screen Shots







```
*******************************************************************************
                        ENTER TRAIN DETAILS
*******************************************************************************

               TRAIN NUMBER : 123
               TRAIN NAME : Hampi_Express
               STARTING POINT : Bellary
               DESTINATION : Hampi
               NUMBER OF AC TICKETS TO BE MADE AVAILABLE  : 10
               PRICE OF EACH AC TICKET Rs. : 500
               NUMBER OF SLEEPER TICKETS TO BE MADE AVAILABLE : 10
               PRICE OF EACH SLEEPER TICKET Rs. : 450
               NUMBER OF SECOND CLASS TICKETS TO BE MADE AVAILABLE : 10
               PRICE OF EACH SECOND CLASS TICKET Rs. : 400
               DURATION OF JOURNEY(hrs)  : 1.5
               TOTAL DISTANCE (km) : 100

*******************************************************************************
                        MORE RECORDS ? (Y/N) : n
```

```
*******************************************************************************
                        ENTER TRAIN DETAILS
*******************************************************************************

               TRAIN NUMBER : 456
               TRAIN NAME : Mumbai_Express
               STARTING POINT : Bellary
               DESTINATION : Mumbai
               NUMBER OF AC TICKETS TO BE MADE AVAILABLE  : 15
               PRICE OF EACH AC TICKET Rs. : 800
               NUMBER OF SLEEPER TICKETS TO BE MADE AVAILABLE : 15
               PRICE OF EACH SLEEPER TICKET Rs. : 750
               NUMBER OF SECOND CLASS TICKETS TO BE MADE AVAILABLE : 15
               PRICE OF EACH SECOND CLASS TICKET Rs. : 700
               DURATION OF JOURNEY(hrs)  : 8
               TOTAL DISTANCE (km) : 3000

*******************************************************************************
                        MORE RECORDS ? (Y/N) : n
```

```
**********************************************************************
                        ADMIN SECTION
**********************************************************************

                    1.CREATE NEW TRAIN FILE
                    2.NEW RECORD
                    3.MODIFY A TRAIN RECORD
                    4.GO BACK
                    5.DELETE A TRAIN RECORD
                    6.DISPLAY ALL THE CURRENT RECORDS
                    7.DISPLAY PASSENGER LIST

**********************************************************************
                    YOUR CHOICE :
```

```
**********************************************************************
                      TRAIN MODIFICATION
**********************************************************************
            TRAIN NUMBER OF TRAIN TO BE MODIFIED : 456
```

```
**********************************************************************
                        TRAIN DETAILS
**********************************************************************

                    TRAIN NUMBER : 456
                    TRAIN NAME : Mumbai_Express
                    STARTING POINT : Bellary
                    DESTINATION : Mumbai
                    AC TICKETS AVAILABLE : 10
                    PRICE OF EACH AC TICKET Rs. : 700
                    SLEEPER CLASS TICKETS AVAILABLE : 10
                    PRICE OF EACH SLEEPER CLASS TICKET Rs. : 650
                    SECOND CLASS TICKETS AVAILABLE : 10
                    PRICE OF EACH SECOND CLASS TICKET Rs. : 600
                    DURATION OF JOURNEY (hrs) : 8
                    TOTAL DISTANCE(km) : 1500

**********************************************************************
```

```
**********************************************************************
                        TRAIN DETAILS
**********************************************************************
  TRAIN NO          TRAIN NAME          FROM          TO        TIME
**********************************************************************
    123            Hampi_Express       Bellary       Hampi      1.5 hr
    456           Mumbai_Express       Bellary      Mumbai        8 hr

**********************************************************************
```

```
**********************************************************************************
                        WELCOME TO INDIAN RAILWAYS
**********************************************************************************

                      1.TO CHECK AVAILABILITY
                      2.TO BOOK TICKETS
                      3.FOR CANCELLATION
                      4.CHECK STATUS
                      5.RETURN

**********************************************************************************
                        YOUR CHOICE :
```

```
**********************************************************************************
                            TRAIN DETAILS
**********************************************************************************
  TRAIN NO            TRAIN NAME          FROM          TO          TIME
**********************************************************************************
    123              Hampi_Express       Bellary       Hampi       1.5 hr
    456             Mumbai_Express       Bellary      Mumbai         8 hr

**********************************************************************************
                          CHECK AVAILABILITY
**********************************************************************************
              THE STARTING POINT : Bellary
              THE DESTINATION : Hampi
```

```
**********************************************************************************
                                TRAIN DETAILS
**********************************************************************************

                     TRAIN NUMBER : 123
                     TRAIN NAME : Hampi_Express
                     STARTING POINT : Bellary
                     DESTINATION : Hampi
                     AC TICKETS AVAILABLE : 10
                     PRICE OF EACH AC TICKET Rs. : 500
                     SLEEPER CLASS TICKETS AVAILABLE : 10
                     PRICE OF EACH SLEEPER CLASS TICKET Rs. : 450
                     SECOND CLASS TICKETS AVAILABLE : 10
                     PRICE OF EACH SECOND CLASS TICKET Rs. : 400
                     DURATION OF JOURNEY (hrs) : 1.5
                     TOTAL DISTANCE(km) : 100

**********************************************************************************
                          NUMBER OF TRAINS ARE : 1
**********************************************************************************
```

```
**********************************************************************************
                        WELCOME TO INDIAN RAILWAYS
**********************************************************************************

                      1.TO CHECK AVAILABILITY
                      2.TO BOOK TICKETS
                      3.FOR CANCELLATION
                      4.CHECK STATUS
                      5.RETURN

**********************************************************************************
                        YOUR CHOICE :
```

```
**********************************************************************************
                        WELCOME TO BOOKING MENU
**********************************************************************************
                        1.TO BOOK TICKETS
                        2.TO GO BACK
**********************************************************************************
                        YOUR CHOICE :
```

```
**********************************************************************************
                          BOOKING SECTION
**********************************************************************************
                      THE BOARDING POINT : Bellary
                      THE DESTINATION : Mumbai
```

```
**********************************************************************************
                              TRAIN DETAILS
**********************************************************************************

                      TRAIN NUMBER : 456
                      TRAIN NAME : Mumbai_Express
                      STARTING POINT : Bellary
                      DESTINATION : Mumbai
                      AC TICKETS AVAILABLE : 10
                      PRICE OF EACH AC TICKET Rs. : 700
                      SLEEPER CLASS TICKETS AVAILABLE : 10
                      PRICE OF EACH SLEEPER CLASS TICKET Rs. : 650
                      SECOND CLASS TICKETS AVAILABLE : 10
                      PRICE OF EACH SECOND CLASS TICKET Rs. : 600
                      DURATION OF JOURNEY (hrs) : 8
                      TOTAL DISTANCE(km) : 1500

**********************************************************************************
```

```
*******************************************************************************
                            BOOKING SECTION
*******************************************************************************
                THE TRAIN NUMBER TO BOOK TICKETS : 456
                THE NUMBER OF TICKETS TO BOOK(MAX .6) : 1
                THE CLASS (AC : ac / SLEEPER : sl / SECOND CLASS : sc) : ac
```

```
*******************************************************************************
                          DETAILS OF PASSENGER 1
*******************************************************************************
                        NAME : Sandeep
                        AGE : 20
                        SEX (M/F) : M
```

```
*******************************************************************************
                                TICKET
*******************************************************************************
                  PNR NUMBER            : 1000
                  TRAIN NUMBER          : 456
                  TOTAL PASSENGERS      : 1
                  TOTAL AMOUNT          : 700
                  BOARDING POINT        : Bellary
                  DESTINATION           : Mumbai
*******************************************************************************
                   YOUR TICKET HAS BEEN RESERVED
*******************************************************************************
                            PASSENGER LIST
*******************************************************************************
           PASSENGER NAME       PASSENGER AGE   PASSENGER SEX
             Sandeep                20                M
*******************************************************************************
```

```
*******************************************************************************
                          WELCOME TO INDIAN RAILWAYS
*******************************************************************************

                        1.TO CHECK AVAILABILITY
                        2.TO BOOK TICKETS
                        3.FOR CANCELLATION
                        4.CHECK STATUS
                        5.RETURN

*******************************************************************************
                          YOUR CHOICE :
```

```
*******************************************************************************
                            BOOKING STATUS
*******************************************************************************
                  YOUR PNR NUMBER : 1000
```

```
**********************************************************************************
                          BOOKING STATUS
**********************************************************************************
                  PNR NUMBER            : 1000
                  TRAIN NUMBER          : 456
                  TRAIN NAME            : Mumbai_Express
                  TOTAL TICKETS         : 1
                  TOTAL AMOUNT          : 700
                  BOARDING POINT        : Bellary
                  DESTINATION           : Mumbai
**********************************************************************************
                          PASSENGER LIST
**********************************************************************************
           PASSENGER NAME       PASSENGER AGE    PASSENGER SEX
             Sandeep                20                 M
**********************************************************************************
```

```
**********************************************************************************
                          PASSENGER LIST
**********************************************************************************
     PNR_NO   TRAIN NO    PASSENGER NAME    AGE    SEX    FROM       DESTINATION

      1000     456           Sandeep       20      M    Bellary       Mumbai

**********************************************************************************
```

```
**********************************************************************************
                       WELCOME TO INDIAN RAILWAYS
**********************************************************************************

                      1.TO CHECK AVAILABILITY
                      2.TO BOOK TICKETS
                      3.FOR CANCELLATION
                      4.CHECK STATUS
                      5.RETURN

**********************************************************************************
                       YOUR CHOICE :
```

```
**********************************************************************************
                          CANCELLATION
**********************************************************************************
                 YOUR PNR NUMBER : 1000
```

```
*********************************************************************************
                          CANCELLATION
*********************************************************************************
                 PNR NUMBER            : 1000
                 TOTAL TICKETS         : 1
                 TOTAL AMOUNT          : 700
                 BOARDING POINT        : Bellary
                 DESTINATION           : Mumbai
*********************************************************************************
                           PASSENGER LIST
*********************************************************************************
          PASSENGER NAME       PASSENGER AGE    PASSENGER SEX
            Sandeep                20                M
*********************************************************************************
                       ARE YOU SURE ? (Y/N) : y

                AFTER CANCELLATION AMOUNT : 560
*********************************************************************************
                  YOUR BOOKING HAS BEEN CANCELLED
*********************************************************************************
```

```
*********************************************************************************
                        WELCOME TO INDIAN RAILWAYS
*********************************************************************************

                     1.TO CHECK AVAILABILITY
                     2.TO BOOK TICKETS
                     3.FOR CANCELLATION
                     4.CHECK STATUS
                     5.RETURN

*********************************************************************************
                       YOUR CHOICE :
```

5.2 Test Cases

```
*********************************************************************************
                        WELCOME TO INDIAN RAILWAYS
*********************************************************************************

                      1.ADMIN ACCESS
                      2.USER ACCESS
                      3.EXIT

*********************************************************************************
                        YOUR CHOICE : 4
```

```
**********************************************************************************
                                WRONG PASSWORD
                                ACCESS DENIED
**********************************************************************************
```

```
**********************************************************************************
                                 ADMIN SECTION
**********************************************************************************

                        1.CREATE NEW TRAIN FILE
                        2.NEW RECORD
                        3.MODIFY A TRAIN RECORD
                        4.GO BACK
                        5.DELETE A TRAIN RECORD
                        6.DISPLAY ALL THE CURRENT RECORDS
                        7.DISPLAY PASSENGER LIST

**********************************************************************************
                        YOUR CHOICE : 8
```

```
**********************************************************************************
                           ENTER TRAIN DETAILS
**********************************************************************************

                TRAIN NUMBER : 789
                TRAIN NAME : Mumbai_Express
                STARTING POINT : Bellary
                DESTINATION : Mumbai
                NUMBER OF AC TICKETS TO BE MADE AVAILABLE  : 0
**********************************************************************************
                NUMBER OF TICKET SHOULD BE GREATER THAN 0
**********************************************************************************
```

```
**********************************************************************************
               NUMBER OF AC TICKETS TO BE MADE AVAILABLE  : 10
               PRICE OF EACH AC TICKET Rs. : 0
**********************************************************************************
               PRICE SHOULD BE GREATER THAN 0
**********************************************************************************
```

```
**********************************************************************************
                PRICE OF EACH AC TICKET Rs. : 100
                NUMBER OF SLEEPER TICKETS TO BE MADE AVAILABLE : 10
                PRICE OF EACH SLEEPER TICKET Rs. : 120
                NUMBER OF SECOND CLASS TICKETS TO BE MADE AVAILABLE : 10
                PRICE OF EACH SECOND CLASS TICKET Rs. : 50
                DURATION OF JOURNEY(hrs)  : 0
**********************************************************************************
                NUMBER OF HOUR'S SHOULD BE GREATER THAN 0
**********************************************************************************
```

```
**********************************************************************************
                          TRAIN MODIFICATION
**********************************************************************************
              TRAIN NUMBER OF TRAIN TO BE MODIFIED : 741
**********************************************************************************
                          SORRY RECORD NOT FOUND !!
**********************************************************************************
```

```
**********************************************************************************
                          DELETE TRAIN
**********************************************************************************
                      TRAIN NUMBER TO BE DELETED : 963
**********************************************************************************
                          SORRY RECORD NOT FOUND !!
**********************************************************************************
```

```
**********************************************************************************
                          TRAIN DETAILS
**********************************************************************************
  TRAIN NO            TRAIN NAME           FROM          TO        TIME
**********************************************************************************
    789             Mumbai_Express        Bellary      Mumbai      1.5 hr

**********************************************************************************
                         CHECK AVAILABILITY
**********************************************************************************
              THE STARTING POINT : hello
              THE DESTINATION : world
**********************************************************************************
                         SORRY NO AVAILABLE TRAINS FOUND
**********************************************************************************
```

```
**********************************************************************************
                              BOOKING SECTION
**********************************************************************************
                  THE TRAIN NUMBER TO BOOK TICKETS : 852
**********************************************************************************
                  SORRY YOU HAVE ENTERD INCORRECT TRAIN NUMBER
**********************************************************************************
```

```
**********************************************************************************
                              BOOKING SECTION
**********************************************************************************
                  THE TRAIN NUMBER TO BOOK TICKETS : 789
                  THE NUMBER OF TICKETS TO BOOK(MAX .6) : -1
**********************************************************************************
                            SORRY CANNOT INVALID INPUT
**********************************************************************************
```

```
**********************************************************************************
                  THE NUMBER OF TICKETS TO BOOK(MAX .6) : 1
                  THE CLASS (AC : ac / SLEEPER : sl / SECOND CLASS : sc) : gh
**********************************************************************************
                  SORRY YOU HAVE ENTERD INCORRECT CLASS
**********************************************************************************
```

```
**********************************************************************************
                            DETAILS OF PASSENGER 1
**********************************************************************************
                           NAME : Buddy
                           AGE : -5
**********************************************************************************
                      INVALID AGE SHOULD BE GREATER THAN 10
**********************************************************************************
```

```
**********************************************************************************
                           AGE : 20
                           SEX (M/F) : G
**********************************************************************************
                      INVALID SEX SHOULD BE (M or F)
**********************************************************************************
```

```
**********************************************************************************
                          BOOKING STATUS
**********************************************************************************
                   YOUR PNR NUMBER : 568
**********************************************************************************
                          SORRY INVALID PNR
**********************************************************************************
```

```
**********************************************************************************
                         CANCELLATION
**********************************************************************************
                  YOUR PNR NUMBER : 485
**********************************************************************************
                         SORRY INVALID PNR
**********************************************************************************
```

```
**********************************************************************************
                        WELCOME TO INDIAN RAILWAYS
**********************************************************************************

                        1.TO CHECK AVAILABILITY
                        2.TO BOOK TICKETS
                        3.FOR CANCELLATION
                        4.CHECK STATUS
                        5.RETURN

**********************************************************************************
                        YOUR CHOICE : -1
```

Conclusion

In this project we have designed and implemented all the requirements of a Railway Reservation System. It has been developed in C++.By using this application, the customer provides reservation services and Passenger details, Seat availability, Price per seat, and other things, we are offering this proposal of computerized reservation system. By using this software, we can Reserve tickets from any part of the world, via internet. Customers can check availability of train and reserve selective seats. The project provides and checks all sorts of constraints so that user does give only useful data and thus validation is done in an effective way.