|  |  |
| --- | --- |
|  | **2020** |
|  | By:  S.KAARTHIK HARIHARAN.T HARISH.A.J |

|  |
| --- |
| **[AIRLINE RESERVATION SYTEM]** |
| [Type the abstract of the document here. The abstract is typically a short summary of the contents of the document. Type the abstract of the document here. The abstract is typically a short summary of the contents of the document.] |

**AIRLINE RESERVATION SYSTEM**

**Objective**

The objective is to design a software system that provides airline ticket booking, cancellation and viewing flight information.

**1 Analysis Phase**

**1.1 Problem Statement**

Airline reservation System is a computerized system used to store and retrieve information and conduct transactions related to air travel. The project is aimed at exposing the relevance and importance of Airline Reservation Systems. It is projected towards enhancing the relationship between customers and airline agencies through the use of ARSs, and thereby making it convenient for the customers to book the flights as when they require such that they can utilize this software to make reservations.

**1.2 IEEE Standard SRS Document**

**1.2.1 Introduction**

Airline reservation System is a computerized system used to store and retrieve information and conduct transactions related to air travel.To book a flight the system asks the customer to enter his details such as name, address, city, state, and credit card number and contact number. Then it checks the validity of card and book the flight and update the airline database and user database. The system also allows the customer to cancel his/her reservation, if any problem occurs.

**1.2.1.1 Purpose**

The purpose of this SRS (Software Requirements Specification) is to list out the requirements needed for an airline reservation system. This SRS is based on any person who needs to book an airline e-ticket. The system provides a solution to allow the user to search for flights satisfying the user criteria, to reserve seats, to manage the user account, and to book a flight.

This software is used to reduce the manual errors involved in the airline reservation process and make it convenient for the customers to book the flights as when they require such that they can utilize this software to make reservations, modify reservations or cancel a particular reservation.

**1.2.1.2 Scope Of The Project**

The product allows the user to check the availability of an airline e-ticket for a particular destination, time and date.

In case the requirements of the user are satisfied, then the user can book the tickets through a facility being provided by the product.

In case the user wants to opt for cancellation of the e-tickets, he/she can do the same.

Also, the user can view any information about flights and their status.

**1.2.2 Overall Description**

**1.2.2.1 Product Perspective**

This project represents Airline Reservation system. All requirements listed herein describe a self-contained system. At a high level, this project will allow a user to book tickets, check flights, cancel tickets, and view flight information. The goal is to allow customers greater and easier access to the airline’s booking system, twenty-four hours a day.

**1.2.2.2 Operating Environment**

The operating environment in this system is client machines, and database.

**1.2.2.3 User Characteristics**

Two types of user interact with the system:

Administrator

The administrator is allowed to access all the services in the system. The administrator can update flight information/status.

User

The user is allowed to access the services such as booking tickets, cancellation and viewing information.

**1.2.2.4 Assumption and Dependencies**

End user is the person having enough knowledge for the project operation.

Basic knowledge of the computers should be known by the users.

**1.2.3 External Interface Recruitment**

**1.2.3.1 Software Interface**

The software is developed with C++, creating the user friendly environment for various services.

The data inserted will be stored in SQL.

**1.2.3.2 Hardware Interface**

Keyboard and mouse are the input for all the systems designed forms.

Keyboard and mouse will provide user friendly interaction between the user and the system and the forms can be viewed through a monitor in GUI.

**1.2.3.3 User Interface**

GUI is provided by the software which is self-explanatory.

User friendly forms, menus and options is contained by the software.

Designing of the product have to be studied by the developer.

Software gives warnings with necessary assistance to the clients.

**1.2.4 System Features**

**1.2.4.1 Hardware**

⦁ Processor : Intel core processor 2 GHz

⦁ RAM : 2 GB RAM

⦁ Hard Disk : 80 GB HDD

⦁ Monitor : Compatible Resolution

⦁ Keyboard : Any Keyboard

**1.2.4.2 Software**

⦁ Operating System : Microsoft Windows 10

⦁ Package : C++ extension in visual studio

⦁ Database : MY SQL/ORACLE

⦁ Diagram & Design : ArgoUML

⦁ Browser : IE (Version 6 or higher) , Mozilla Firefox or Google Chrome

**1.2.4.3 Constraints**

**1.2.4.3.1 Hardware Limitations**

The limitation of dream viewer is that it requires RAM that cannot be less than 520 MB and the processor cannot be less than 2 GHz speed as recommended in the hardware requirements.

**1.2.5 Functional And Non Functional Requirements**

**1.2.5.1 Functional Requirements**

Search for flights

Using this function a customer is able to search for one-way, round-trip and multiple destination flights by choosing specific dates and destinations.

Book flights

This function allows customer to book flights by choosing ticket types and processing online payment.

Request cancellation

This function indicates that customer can request the cancellation of the reserved ticket.

Add new flights

The function grants administrator the privilege of adding new flights to the system. Modify flight details

Using this functionality administrator can modify the details of the existing flights. Remove flights

With the help of this function administrator removes the flights from the system that are cancelled for whatever reason.

**1.2.5.2 Non Functional Requirements**

⦁ This application is secure for every kind of its users, because here is facility of session management.

⦁ The database used here is robust, reliable & fast.

⦁ So users will have to wait for the output very short time.

⦁ This application can be accessed from any type of platform.

⦁ There is no case of redundancy in the database so it will not take extra memory space.

**1.2.5.3 Reliability Requirements**

The main reliability requirement is the validation used. Without proper validation the system does not allow to enter that value into database. All the required validation controls are kept controls are kept to keep the system secure. The following are the some of the reliability requirements:

⦁ In the email ID the user cannot enter any dummy value, the validation checks that whether there is a '@' or '\_' symbol in that.

⦁ Any null value is not allowed in place of compulsory fields.

⦁ In numeric field user cannot enter any character value.

⦁ In date of birth, user cannot enter date and time other than given format.

⦁ Entered password and confirm password must match to each other.

⦁ User can not re –register an account on his primary email.

**1.2.5.4 Safety and Security Considerations Safety**

The source of this software will be kept at more than one place with user ID, password and also in CD ROM in case of server failure.

⦁ Security

Security in this software provide to different user in different ways by giving different user id. If user is admin, he has all the privileges and constraints. He can access the entire database. He can change or delete database from other user’s accounts. HR department have limited access according to their role. Because of limited privileges one of them cannot update other details of the candidate.

**1.3 Project Plan**

**1.3.1 Project Planning**

Project planning includes description of the project task, activity and function, dependencies, resource requirements and detail schedules. Project planning involves estimating how much time, efforts, money and resources will be required build a specific software system.

**1.3.2 Gantt Chart**

**2 Design Phase**

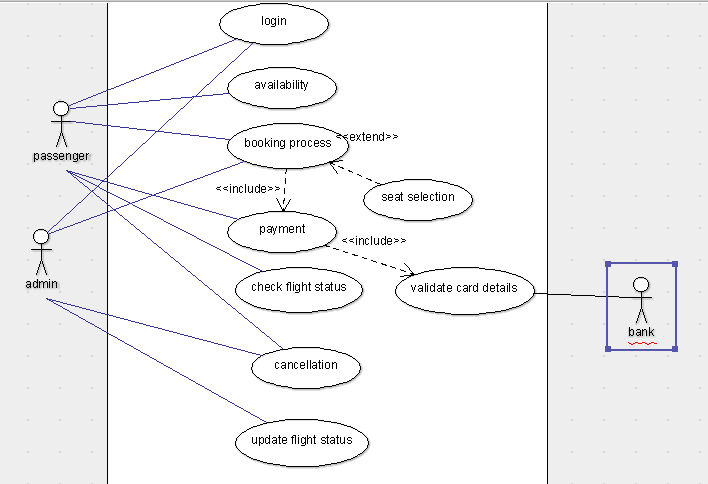
**2.1 UML Diagrams**

The Unified Modelling Language is a language for specifying, constructing, visualizing, and documenting the artifacts of a software-intensive system. Analogous to the use of architectural blueprints in the construction industry, UML provides a common language for describing software models, and it can be used in conjunction with a wide range of software lifecycles and development processes.

**2.1.1 Use Case Diagram**

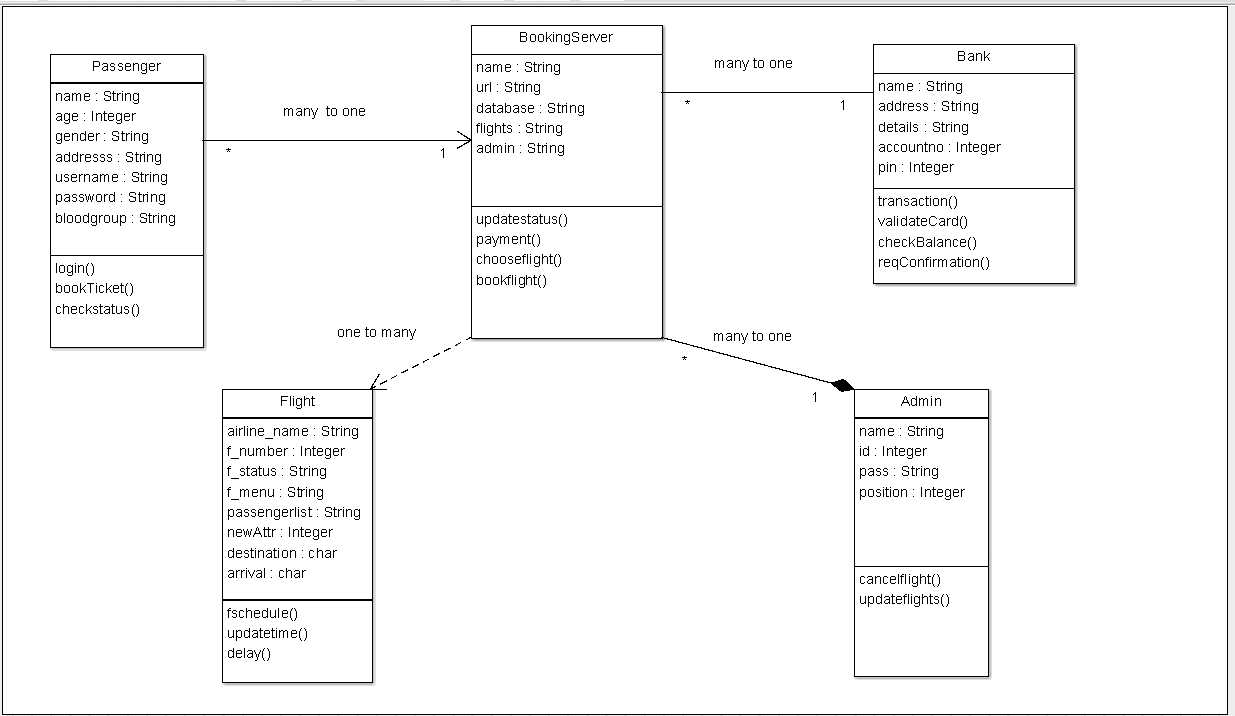
Use case diagrams are behaviour diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

The actors in this use case diagram are administrator, passenger and bank. The use cases are the activities performed by actors.



**2.1.2 Class Diagram**

The class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application. The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. It is also known as a structural diagram.

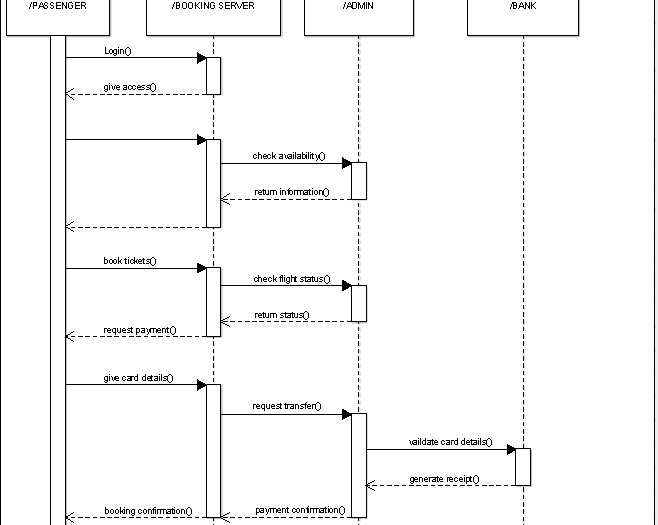


**2.1.3 Interaction Diagram**

We have two types of interaction diagrams in UML. One is sequence diagram and the other is a collaboration diagram. The sequence diagram captures the time sequence of message flow from one object to another and the collaboration diagram describes the organization of objects in a system taking part in the message flow.

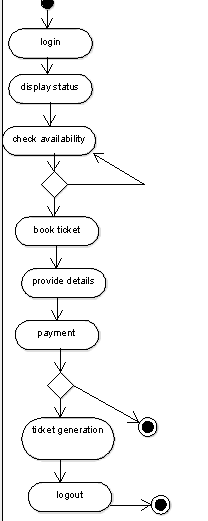
**2.1.3.1 Sequence Diagram**

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart.



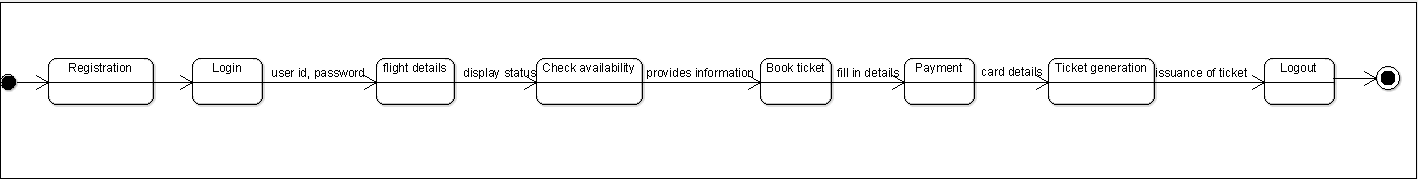
**2.1.4 Activity Diagram**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control. An activity is shown as an rounded box containing the name of the operation.



**2.1.5 State Chart Diagram**

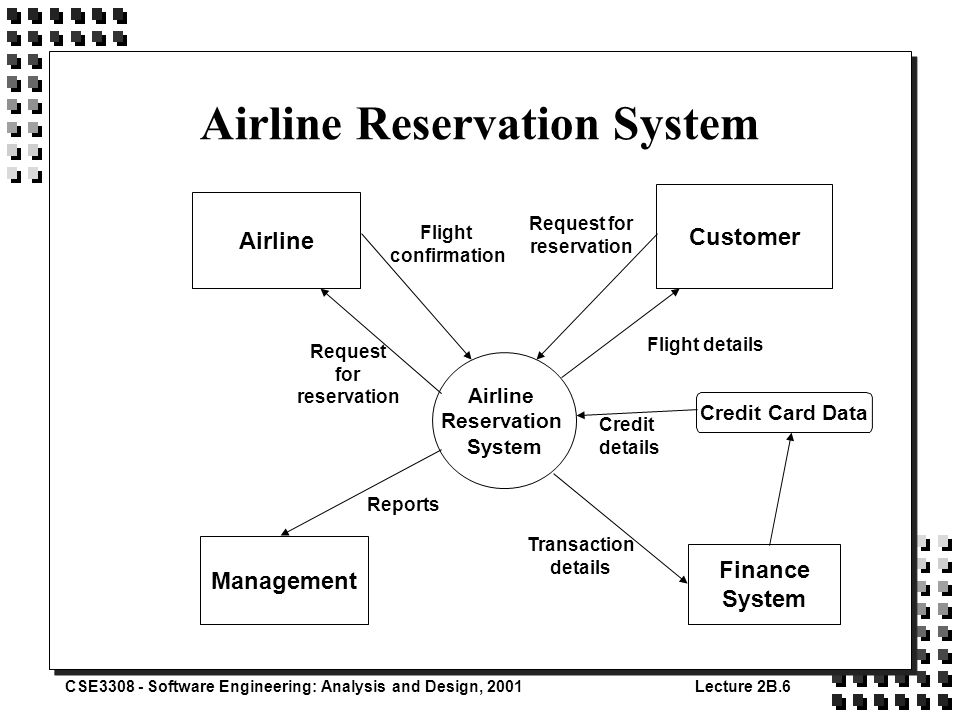
The purpose of state chart diagram is to understand the algorithm involved in performing a method. It is also called as state diagram. A state is represented as a round box, which may contain one or more compartments. An initial state is represented as small dot. A final state is represented as circle surrounding a small dot.



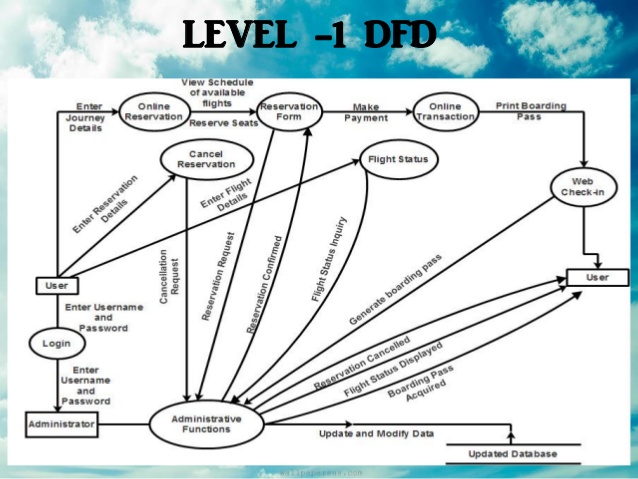
**2.1.6 Data Flow Diagram**

Data flow diagram represents one of the most ingenious tools for structured analysis. A DFD has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It is the major starting point in the design phase that functionally decomposes the requirement specifications down to the lowest level of detail.

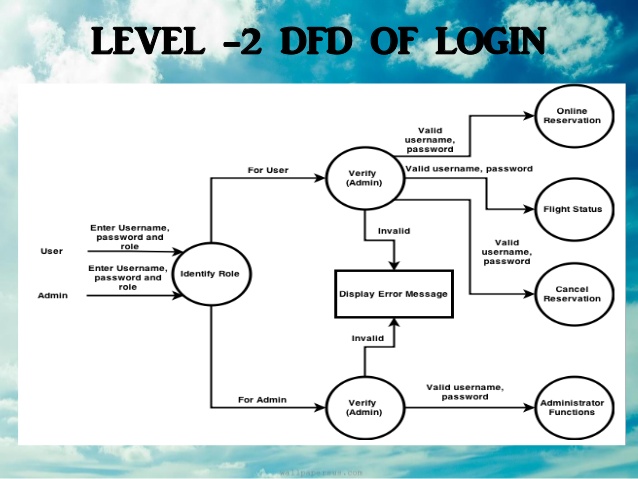
**LEVEL-0**



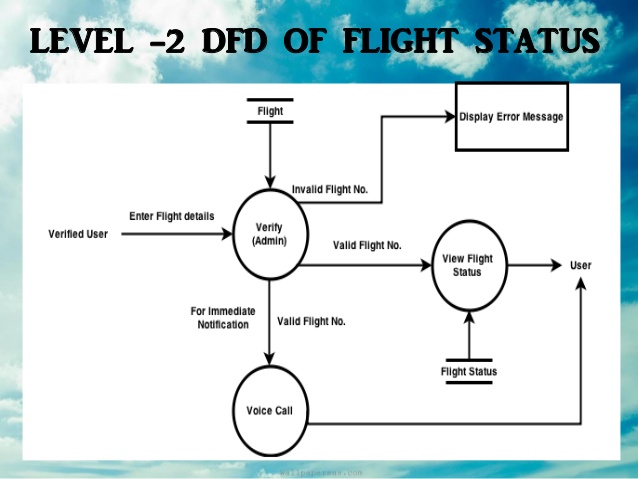
**LEVEL-1**



**LEVEL-2[login]**



**LEVEL-2[flight status]**



**2.2 Source code**

#include <iostream>

#include<fstream>

#include<string.h>

using namespace std;

int glob=0; //global variables

int global=10;

class d\_booking //class for domestic booking

{

protected:

int pnr;

char f\_d[10],toja[7],tojd[7]; //protected members

long int doj;

int choice,src,dest;

public://public member functions

void d\_pnr()

{

glob++; // increment variable

pnr=glob;

}

int j\_detail() // function declaration and definition for domestic journey

{

cout << "\nEnter DateOfJourney(DDMMYY)." << "Please enter a valid date." << endl;

cin >> doj;

cout << "\1.Brisbane(1) \t\2.Sydney(2) \t\3.Melbourne(3) \t\4.Canberra(4)" << endl << endl;

cout << "\tEnter Source" << endl;

cin >> src;

cout << "\tEnter destination" << endl;

cin >> dest;

if((src==1 && dest==2) || (src==2 && dest==1))//condition

{

cout << "\t \t \tFlights Found" << endl << endl;

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Qantas(1)\t08:00\t\t11:05\t\tRs.5000\t\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t17:05\t\tRs.5500\t\tRefundable\n";

cout << "\3.Go Air(3)\t19:00\t\t22:05\t\tRs.6000\t\tRefundable\n";

}

else if((src==1 && dest==3) || (src==3 && dest==1))//condition

{

cout << "\t \t \tFlights Found" << endl << endl;

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Qantas(1)\t08:00\t\t11:05\t\tRs.5000\t\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t17:05\t\tRs.5500\t\tRefundable\n";

cout << "\3.Go Air(3)\t19:00\t\t22:05\t\tRs.6000\t\tRefundable\n";

}

else if((src==1 && dest==4) || (src==4 && dest==1))//condition

{

cout << "\t \t \tFlights Found" << endl << endl;

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Qantas(1)\t08:00\t\t11:05\t\tRs.4000\t\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t17:05\t\tRs.4250\t\tRefundable\n";

cout << "\3.Go Air(3)\t19:00\t\t22:05\t\tRs.6100\t\tRefundable\n";

}

else if((src==2 && dest==3) || (src==3 && dest==2))//condition

{

cout << "\t \t \tFlights Found" << endl << endl;

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Qantas(1)\t08:00\t\t11:05\t\tRs.5400\t\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t17:05\t\tRs.2500\t\tRefundable\n";

cout << "\3.Go Air(3)\t19:00\t\t22:05\t\tRs.2890\t\tRefundable\n";

}

else if((src==2 && dest==4) || (src==4 && dest==2))//condition

{

cout << "\t \t \tFlights Found" << endl << endl;

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Qantas(1)\t08:00\t\t11:05\t\tRs.5000\t\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t17:05\t\tRs.4500\t\tRefundable\n";

cout << "\3.Go Air(3)\t19:00\t\t22:05\t\tRs.6000\t\tRefundable\n";

}

else if((src==3 && dest==4) || (src==4 && dest==3))//condition

{

cout << "\t \t \tFlights Found" << endl << endl;

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Qantas(1)\t08:00\t\t11:05\t\tRs.5800\t\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t17:05\t\tRs.5508\t\tRefundable\n";

cout << "\3.Go Air(3)\t19:00\t\t22:05\t\tRs.6050\t\tRefundable\n";

}

else if(src==dest)//condition

{

cout << "\nSource and destination can't be same.\nTry again\n\n\n" << endl;

return j\_detail();

}

else

{

cout <<"\nWrong input entered\nTry again\n\n\n" << endl;

return j\_detail();

}

}

int select\_flight() //function declaration and definition for selecting flight

{ cout << "\nEnter your choice" << endl;

cin >> choice;

switch(choice) // switch case

{

case 1://condition

cout << "\nFlight selected:"<<endl;

cout << "Qantas"<<endl;

strcpy(f\_d,"Qantas");//copy to string

cout << "Departure Time : 08:00"<<endl;

cout<<"Arrival Time: 11:05"<<endl;

strcpy(tojd,"8:00"); //copy to string

strcpy(toja,"11:05");// copy to string

break;

case 2://condition

cout << "\nFlight selected:"<<endl;

cout << "Fly Dubai"<<endl;

strcpy(f\_d,"Fly Dubai");//copy to string

cout << "Departure Time : 14:00"<<endl;

cout<<"Arrival Time: 17:05"<<endl;

strcpy(tojd,"14:00");//copy to string

strcpy(toja,"17:05");//copy to string

break;

case 3://condition

cout << "\nFlight selected:" << endl;

cout << "Go Air" << endl;

strcpy(f\_d,"Go Air");//copy to string

cout << "Departure Time : 19:00" << endl;

cout<<"Arrival Time: 22:05" << endl;

strcpy(tojd,"19:00");//copy to string

strcpy(toja,"22:05");//copy to string

break;

default://condition

cout << "Wrong input entered.\nTry again" << endl;

return select\_flight();

}

}

};

class i\_booking//class for international booking

{

protected://protected members

int pnri;

char f\_i[10],tojai[7],tojdi[7];

long int doji;

int srci,desti,choicei;

public://public member functions

void i\_pnr()

{

global++;//increment variable

pnri=global;

}

//brought to you by code-projects.org

int j\_detaili()// function declaration and definition for journey details

{

cout << "Enter DateOfJourney(DDMMYY)." << "Please enter a valid date." << endl;;

cin >> doji;

cout << "\1.London(1) \2.Dubai(2) \3.Abu Dhabi(3) \4.Singapore(4) \5.NewYork(5) " << endl << endl;

cout << "\tEnter Source" << endl;

cin >> srci;

cout << "\nEnter destination" ;

cin >> desti;

cout << "\t \t \tFlights Found" << endl << endl;

if((srci==1 && desti==3) || (srci==3 && desti==1))//condition

{

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Vistara(1)\t10:00\t\t14:05\t\tRs.25000\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t18:05\t\tRs.21500\tRefundable\n";

cout << "\3.Emirates(3)\t18:00\t\t22:05\t\tRs.24000\tRefundable\n";

}

else if((srci==1 && desti==4) || (srci==4 && desti==1))//condition

{

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Vistara(1)\t10:00\t\t14:05\t\tRs.25500\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t18:05\t\tRs.21300\tRefundable\n";

cout << "\3.Emirates(3)\t18:00\t\t22:05\t\tRs.24650\t\tRefundable\n";

}

else if((srci==1 && desti==5) || (srci==5 || desti==1))//condition

{

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Vistara(1)\t10:00\t\t14:05\t\tRs.52500\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t18:05\t\tRs.59420\tRefundable\n";

cout << "\3.Emirates(3)\t18:00\t\t22:05\t\tRs.64892\tRefundable\n";

}

else if((srci==2 && desti==3) || (srci==3 && desti==2))//condition

{

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Vistara(1)\t10:00\t\t14:05\t\tRs.17800\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t18:05\t\tRs.14900\tRefundable\n";

cout << "\3.Emirates(3)\t18:00\t\t22:05\t\tRs.18700\tRefundable\n";

}

else if((srci==2 && desti==4) || (srci==4 && desti==2))//condition

{

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Vistara(1)\t10:00\t\t14:05\t\tRs.32000\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t18:05\t\tRs.38500\tRefundable\n";

cout << "\3.Emirates(3)\t18:00\t\t22:05\t\tRs41259\tRefundable\n"; }

else if(srci==2 && desti==5 || (srci==5 && desti==2))//condition

{

cout << "Airline:\tDeparture:\tArrival:\tPrice:\t\tCategory:\n";

cout << "\1.Vistara(1)\t10:00\t\t14:05\t\tRs.82500\tRefundable\n";

cout << "\2.Fly Dubai(2)\t14:00\t\t18:05\t\tRs.87550\tRefundable\n";

cout << "\3.Emirates(3)\t18:00\t\t22:05\t\tRs81478\tRefundable\n";

}

else if(srci==desti)//condition

{

cout << "wrong input entered.\nTry again\n\n\n"<< endl;

return j\_detaili();

}

else//condition

{

cout << "Wrong input entered.\nTry again\n\n\n";

return j\_detaili();

}

}

int select\_flighti()//function declaration and definition for selecting flight

{

cout << "\nEnter your choice" << endl;

cin >> choicei;

switch(choicei)//switch case

{

case 1://condition

cout << "\nFlight selected:" <<endl;

cout << "Vistara" << endl;

strcpy(f\_i,"Vistara");//copy to string

cout << "Departure Time: 10:00" << endl;

cout << "Arrival Time: 14:05" << endl;

strcpy(tojdi,"10:00");//copy to string

strcpy(tojai,"14:05");//copy to string

break;

case 2://condition

cout << "\nFlight selected:" << endl;

cout << "Fly Dubai" << endl;

strcpy(f\_i,"Fly Dubai");//copy to string

cout << "Departure Time: 14:00" << endl;

cout << "Arrival Time: 18:05" << endl;

strcpy(tojdi,"14:00");//copy to string

strcpy(tojai,"18:05");//copy to string

break;

case 3://condition

cout << "\nFlight selected:" << endl;

cout << "Emirates" << endl;

strcpy(f\_i,"Emirates");//copy to string

cout << "Departure Time : 18:00" << endl;

cout << "Arrival Time: 22:05" << endl;

strcpy(tojdi,"18:00");//copy to string

strcpy(tojai,"22:05");//copy to string

break;

default://condition

cout << "Wrong input entered" << endl;

return select\_flighti();

}

}

};

class passenger: public d\_booking,public i\_booking//class passenger publicly inherited from class d\_booking and i\_booking

{

protected://protected members

char f\_name[20],l\_name[20],email[50];

int age,gender;

long int c\_no;

public://public member functions

void p\_detail(int x)//function declaration and definition

{ if(x==1)//if else for domestic and international booking selection

{ j\_detail();//function call

select\_flight();//function call

}

else

{ j\_detaili();//function call1353

select\_flighti();//function call

}

cout << "\n\n\nEnter passenger details";

cout << "\nFirst Name:";

cin >> f\_name;

cout << "Last Name:";

cin >> l\_name;

}

int gender\_check()//to check gender input as valid

{

cout << "\nGender:\nMale-press:1::\nFemale-press:2::";

cin >> gender;

if(gender>2)//condition

{

cout << "\n\nWrong input entered.\nTry again\n\n" << endl;

return gender\_check();//function call

}

}

void more\_details()//to take more details of the passenger

{

cout << "Age:";

cin >> age;

cout << "Email Id:";

cin >> email;

cout << "Contact no.(6 digits):";

cin >> c\_no;

cout << "\n\nDetails Entered:\n";

cout << "Name:" << f\_name << " " << l\_name << endl;

cout << "Gender:" << gender << endl; //displaying details

cout << "Age:" << age << endl;

cout << "Email id:" << email << endl;

cout << "Contact No.:" << c\_no << endl;

}

int getpnr()//function to get pnr for domestic booking

{

return pnr;

}

int getpnri()//function to get pnr for international booking

{

return pnri;

}

void disp()//function to display details for domestic booking

{

cout<<"PNR:" << pnr << endl;

cout<<"Flight:" << f\_d << endl;

cout<<"Name:" << f\_name << " " << l\_name << endl;

cout<<"DOJ:" << doj << endl;

cout<<"Departure Time:" << tojd << endl;

cout<<"Arrival Time:" << toja;

}

void dispi()//function to display details for international booking

{

cout<<"PNR:" << pnri << endl;

cout<<"Flight:" << f\_i << endl;

cout<<"Name:" << f\_name << " " << l\_name << endl;

cout<<"DOJ:" << doji << endl;

cout<<"Departure Time:" << tojdi << endl;

cout<<"Arrival Time:" << tojai;

}

};

class payment//class for payment

{

protected://protected members

long

int choice1,bank,card,date,cvv,user\_id;

char password[10];

public://public members functions

void pay\_detail()//function declaration and definition for payment method

{ cout << "\n\n\nHow would you like to pay?:\n";

cout << "\n\1.Debit Card(1) \n\2.Credit Card(2) \n\3.Net Banking(3)";

cout << "\n\nEnter your choice";

cin >> choice1;

switch(choice1)//switch case

{

case 1://condition

cout << "\nEnter card no.:";

cin >> card;

cout << "\nEnter expiry date:";

cin >> date;

cout << "\nEnter CVV no.:";

cin >> cvv;

cout << "\nTransaction Successful\n";

break;

case 2://condition

cout << "\nEnter card no.:";

cin >> card;

cout << "\nEnter expiry date:";

cin >> date;

cout << "\nEnter password:";

cin >> password;

cout << "\nTransaction Successful\n";

break;

case 3://condition

cout << "Banks Available: \1.West Pac Bank(1) \2.Nabil Bank(2) \3.Standard Chartered Bank(3) \4.AMP Bank(4) \5.Others(5)";

cout << "\nSelect your bank:";

cin >> bank;

cout << "\nYou have selected:" << bank;

cout << "\nEnter user id:";

cin >> user\_id;

cout << "\nEnter password:";

cin >> password;

cout << "\nTransaction Successful\n";

break;

default://condition

cout << "\nWrong input entered.\nTry again\n\n";

return pay\_detail();

}

}

};

void createfile(passenger p)//file creation for domestic booking

{ ofstream fin("domestic.txt",ios::binary|ios::app);

fin.write((char\*)&p,sizeof(p));//writing to file

fin.close();//closing file

}

void cancelticket(int x)//function to cancel ticket

{ passenger p;

int f=0;

ifstream fout("domestic.txt",ios::binary|ios::app);//for reading file

ofstream fin("domestic1.txt",ios::binary|ios::app);//for writing to a new file

fout.read((char \*)&p,sizeof(p));//reading file

while(fout)

{

if(p.getpnr()!=x)//checking pnr

fin.write((char \*)&p,sizeof(p));//writing to file

else

{

p.disp();//display details

cout<<"\nYour Above ticket is being canceled:\n" << "Amount refunded: Rs 1000\n";

f++;//incrementing f if pnr found

}

fout.read((char \*)&p,sizeof(p));//reading another record from file

}

if(f==0)//if f==0,pnr not found

cout<<"Ticket not found\n";

fout.close();//closing file

fin.close();//closing file

remove("domestic.txt");//deleting old file

rename("domestic1.txt","domestic.txt");//renaming new file

}

void checkticket(int x)//function to check pnr or ticket

{ passenger p;

int f=0;

ifstream fout("domestic.txt",ios::binary);//opening file

fout.read((char \*)&p,sizeof(p));//reading file

while(fout)

{

if(p.getpnr()==x)//checking pnr

{p.disp();//display details

cout<<"\nYour ticket"<<endl;

f++;//incrementing f if onr found

break;

}

fout.read((char \*)&p,sizeof(p));//reading another record from the same file

}

fout.close();//closing file

if(f==0)//if f==0, pnr not found

cout<<"Ticket not found"<<endl;

}

void createfilei(passenger p)//opening a file for international booking

{ ofstream fin("international.txt",ios::binary|ios::app);

fin.write((char\*)&p,sizeof(p));//writing to file

fin.close();//closing file

}

void cancelticketi(int x)//function to cancel ticket

{ passenger p;

int f=0;

ifstream fout("international.txt",ios::binary|ios::app);//opening file

ofstream fin("international1.txt",ios::binary|ios::app);//writing to a new file

fout.read((char \*)&p,sizeof(p));//reading old file

while(fout)

{

if(p.getpnri()!=x)//checking pnr

fin.write((char \*)&p,sizeof(p));//writing to new file;

else

{

p.dispi();//display details

cout<<"Your Above ticket is being deleted:\n"<<"Amount refunded: Rs 1000\n";

f++;//incrementing f if pnr found

}

fout.read((char \*)&p,sizeof(p));//reading another record from old file

}

if(f==0)//if f==0,pnr not found

cout<<"\nTicket not found\n";

fout.close();//closing file

fin.close();//closing file

remove("international.txt");//deleting old file

rename("international1.txt","international.txt");//renaming new file

}

void checkticketi(int x)//function to check pnr or ticket

{ passenger p;

int f=0;

ifstream fout("international.txt",ios::binary);//opening file

fout.read((char \*)&p,sizeof(p));//reading file

while(fout)

{

if(p.getpnri()==x)//checking pnr

{p.dispi();//display details

cout<<"\nYour ticket"<<endl;

f++;//incrementing f if pnr found

break;

}

fout.read((char \*)&p,sizeof(p));//reading another record from the file

}

fout.close();//closing file

if(f==0)//if f==0, pnr not found

cout<<"Ticket not found"<<endl;

}

int main()//main function

{

class d\_booking d1;//object for class d\_booking

class i\_booking i1;//object for class i\_booking

class passenger p1;//object for class passenger

class payment p2;//object for class payment

int ch,ch1,n;//integer variables

char input;//character variables

do//do while loop

{

system("CLS");

cout << "\n\n \t\tWelcome To Flight Reservation System" << endl << endl;

cout <<"\t <><><><><><><><><><><><><><><><><><><><><><><>\n";

cout << "\t Book your Flight tickets at affordable prices!" << endl;

cout <<"\t <><><><><><><><><><><><><><><><><><><><><><><>";

cout << "\n\n\t\t\t\1.Book Flight(1) \n\t\t\t\2.Cancel Fight(2) \n\t\t\t\3.Check Ticket(3) \n\t\t\t\4.Exit(4)" << endl;

cout << "\n\t\t Please enter your choice:";

cin >> ch;

switch(ch)//witch case

{

case 1://condition

system("CLS");

cout << "\n\n\1.Domestic Fights(1) \n\2.International Flights(2)" << endl;

cout << "\nPlease enter your option" << endl;

cin >> ch1;

switch(ch1)//inner switch case

{

case 1://for booking domestic ticket

p1.d\_pnr();

p1.p\_detail(1);//function calls

p1.gender\_check();

p1.more\_details();

p2.pay\_detail();

p1.disp();

createfile(p1);//call to create file

break;

case 2: //for booking international ticket

p1.p\_detail(2);//function calls

p1.i\_pnr();

p1.gender\_check();

p1.more\_details();

p2.pay\_detail();

p1.dispi();

createfilei(p1);//call to create file

break;

default://wrong input

cout << "Wrong input entered\nTry again\n\n\n" << endl;

return main();

}

break;

case 2:

//for canceling ticket

system("CLS");

cout << "\1.Domestic Fights(1) \n\2.International Flights(2)" << endl;

cout << "\nPlease enter your option" << endl;

cin >> ch1;

if(ch1==1)

{

cout << "Please enter your PNR no.:" << endl;

cin>>n;

cancelticket(n);//function call for domestic booking cancellation

}

else if(ch1==2)

{ cout << "Please enter your PNR no.:" << endl;

cin>>n;

cancelticketi(n);//function call for international cancellation

}

else

{

cout << "Wrong input entered\nTry again\n\n\n";

return main();

}

break;

case 3://for displaying booked ticket details

system("CLS");

cout << "\1.Domestic Fights(1) \n\2.International Flights(2)" << endl;

cout << "\nPlease enter your option" << endl;

cin >> ch1;

if(ch1==1)

{cout << "Please enter your PNR no.:" << endl;

cin>>n;

checkticket(n);}//function call to display domestic ticket details

else if(ch1==2)

{ cout << "Please enter your PNR no.:" << endl;

cin>>n;

checkticketi(n);//function call to display domestic ticket details

}

else

{

cout << "Wrong input entered.\nTry again\n\n\n";

return main();

}

break;

case 4:

system("CLS");

cout<<"\n\n\t\t\t\tBrought to you by code-projects.org";

return 0;

default://for wrong input

cout << "Wrong input entered\nTry again.\n\n\n\n" << endl;

return main();

}

cout<<"\n\n\nDo you wish to continue:(y/Y)" << endl;

cin >> input;

}while(input=='Y' || input=='y');//condition for do while loop

return 0;

}

**2.3. Testing**

Tool Used: Selenium

The testing of software means measuring or accessing the software to determine the quality. Testing is a measuring instrument for software quality with the unit of measurement being the number of defects found during testing.

Testing activities also help to achieve software quality. Testing is essential in the development of any system software. Testing is essential in the development of any software system. Testing is in order to access what the system actually does and how well expected to spend approximately40% of development cost and time in testing in order to achieve reasonable quality levels.

**2.3.1 Test cases:**

## **2.3.2 Testing Reports**

* No bug blocks the execution of tasks
* System states are visible
* All factors affecting the output are visible
* Functional simplicity
* Distinct output is generated for each input

|  |  |
| --- | --- |
| Test Case no | 1 |
| Test case action | Checks system behavior when credentials provided by admin are correct |
| Input | choose/click on check flights |
| Expected output | Result page will contain information about flights |
| pass/fail | pass |

|  |  |
| --- | --- |
| Test Case no . | 2 |
| Test case action | Checks system behavior when credentials provided by admin are correct |
| Input | Update information on flights |
| Expected output | Result page contains the updated information |
| pass/fail | pass |

|  |  |
| --- | --- |
| Test case | 3 |
| Test case action | Checks system behavior when credentials provided by admin are correct |
| Input | The user uses one among the given options |
| Output | Displays the corresponding option |
| pass/fail | Pass |

|  |  |
| --- | --- |
| Test case no | 4 |
| Test case action | Checks system behavior when credentials provided by user are correct |
| Input | In book ticket page user enters required information |
| Expected output | The result message displays the user/passenger details |
| pass/fail | pass |

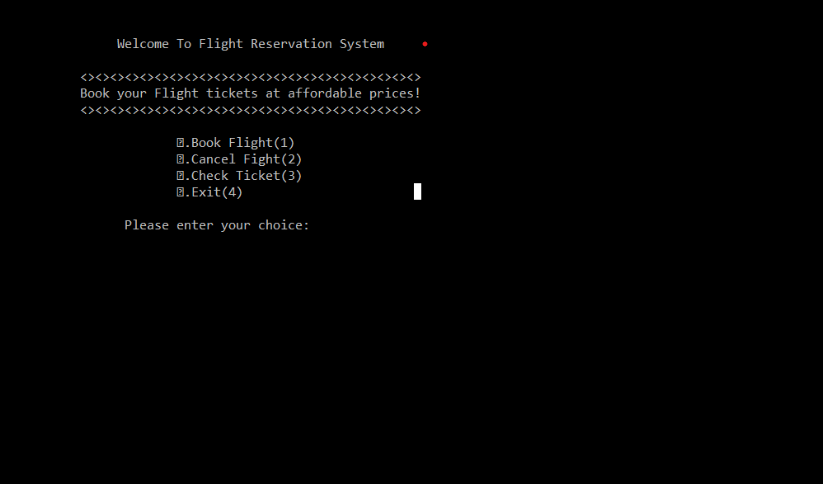
|  |  |
| --- | --- |
| Text case no | 5 |
| Test case action | Checks system behavior when credentials provided by user are correct |
| Input | In the payment section the user must enter correct credentials |
| Output | Resultant message showing the booking confirmation |
| pass/fail | Pass |

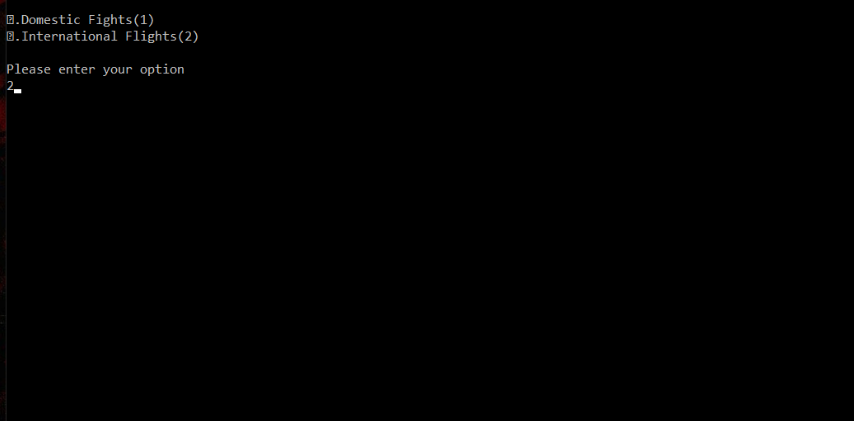
|  |  |
| --- | --- |
| Test case | 6 |
| Test case action | Checks system behavior when credentials provided by user are correct |
| Input | In the payment section the user must enter correct credentials |
| Output | If entered correctly the resultant message of cancellation of message is displayed |
| pass/fail | Pass |

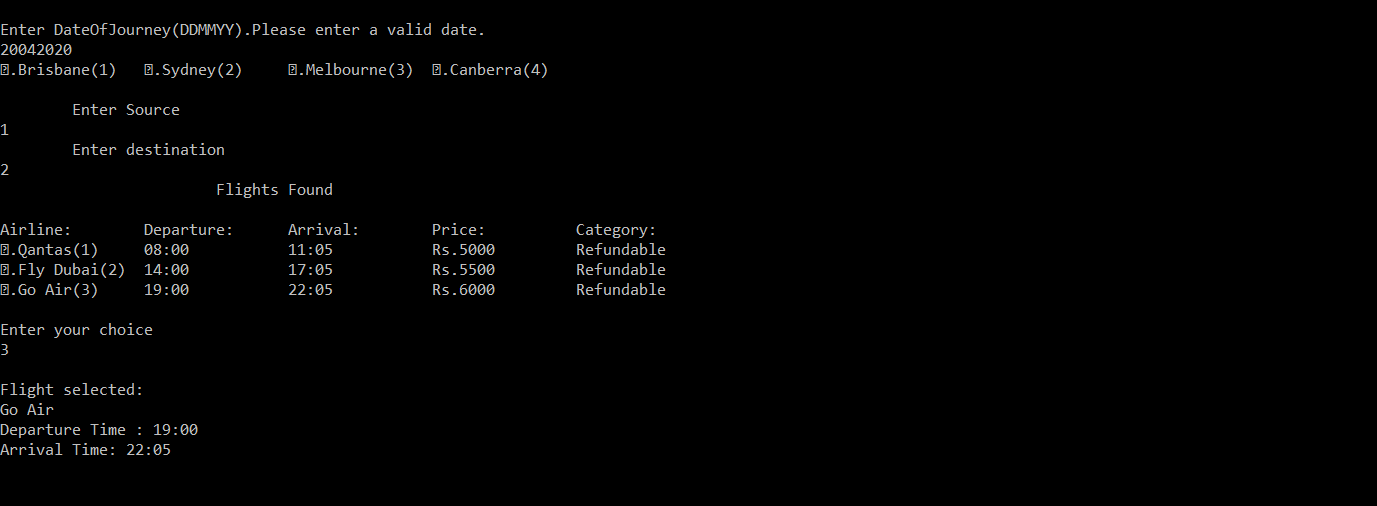
|  |  |
| --- | --- |
| Test case | 7 |
| Test case action | Checks system behavior when credentials provided by user are correct |
| Input | In the check ticket section the user must enter correct credentials |
| Output | Details of the corresponding flight |
| pass/fail | Pass |

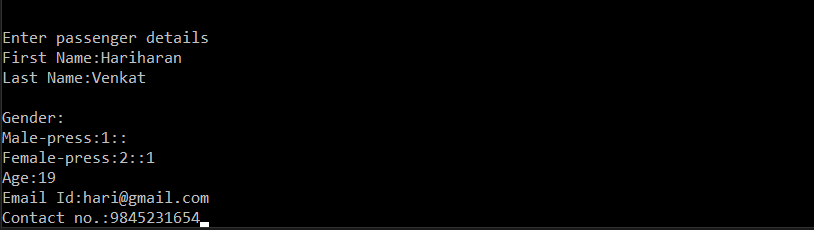
|  |  |
| --- | --- |
| Test case | 7 |
| Test case action |  |
|  |  |
|  |  |
|  |  |

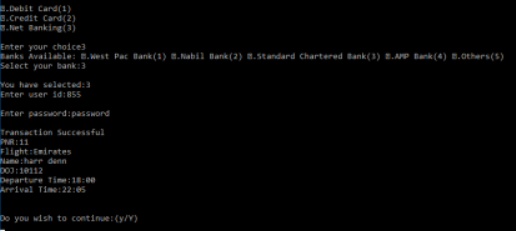
**2.4 Screenshots**

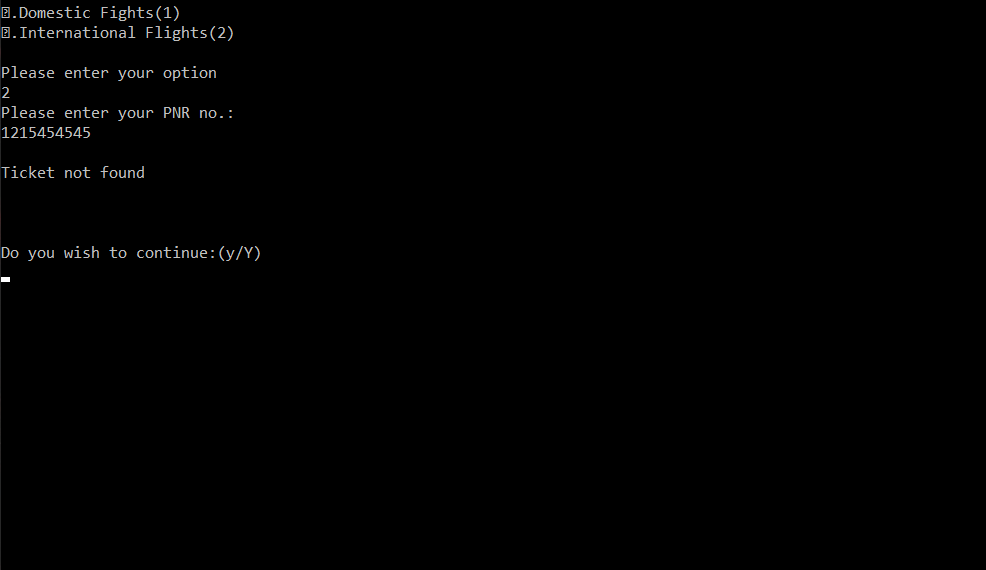












**Result**

Thus the experiment regarding Online Recruitment system is studied, analysed and the corresponding models are generated.