**AIRPORT MANAGEMENT SYSTEM**

****

**Session 2022 – 2026**

**Submitted by:**

Raveeha Mohsin 2022-CS-149

**Supervised by:**

Dr. Awai s Hassan

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

# Table of Contents

[1 Short Description of Project: 3](#_Toc128720818)

[2 Users of Application 3](#_Toc128720819)

[2.1 Admin 3](#_Toc128720820)

[2.2 Passenger 3](#_Toc128720821)

[3 Functional Requirements 4](#_Toc128720822)

[3.1 Admin 4](#_Toc128720823)

[3.2 Passenger 5](#_Toc128720824)

[4 Wireframes 6](#_Toc128720825)

[4.1.1 Sign up & Sign in options 6](#_Toc128720826)

[4.1.2 Sign up for admin 6](#_Toc128720827)

[4.1.3 Sign in Admin 7](#_Toc128720828)

[4.1.4 Admin Interface 7](#_Toc128720829)

[4.1.5 Admin option 1 8](#_Toc128720830)

[4.1.6 Option 1 further options 8](#_Toc128720831)

[4.1.7 Admin option 2 8](#_Toc128720832)

[4.1.8 Option 3 9](#_Toc128720833)

[4.1.9 Option 3 further menu 9](#_Toc128720834)

[4.1.10 Option 4 9](#_Toc128720835)

[10](#_Toc128720836)

[4.1.11 Option 5(New record of passenger will be shown here) 10](#_Toc128720837)

[4.1.12 Option 6 (new record will be shown here) 10](#_Toc128720838)

[4.1.13 Option 7 10](#_Toc128720839)

[4.1.14 Option 7 B opt 11](#_Toc128720840)

[4.1.15 Option 8 11](#_Toc128720841)

[4.1.16 Option 9 11](#_Toc128720842)

[4.1.17 Option 9 B 12](#_Toc128720843)

[4.1.18 Passenger Sign Up 12](#_Toc128720844)

[4.1.19 Passenger Sign in 12](#_Toc128720845)

[4.1.20 Passenger Interface 13](#_Toc128720846)

[4.1.21 Option 1 13](#_Toc128720847)

[4.1.22 Option 1 further menu 14](#_Toc128720848)

[4.1.23 Option 2 14](#_Toc128720849)

[4.1.24 Option 3 15](#_Toc128720850)

[4.1.25 Option 3 further menu 15](#_Toc128720851)

[4.1.26 Option 4 16](#_Toc128720852)

[4.1.27 Option 4 further submenu 16](#_Toc128720853)

[4.1.28 Option 5 17](#_Toc128720854)

[4.1.29 Option 6 17](#_Toc128720855)

[4.1.30 Option 7 18](#_Toc128720856)

[4.1.31 Option 8 18](#_Toc128720857)

[4.1.32 Option 9 19](#_Toc128720858)

[4.1.33 Option 10 19](#_Toc128720859)

[4.1.34 Admin updated option 5 20](#_Toc128720860)

[4.1.35 Admin updated option 6 20](#_Toc128720861)

[4.1.36 Admin updated option 8 20](#_Toc128720862)

[5 Data Structure (Parallel Arrays) 21](#_Toc128720863)

[6 Function Prototypes 22](#_Toc128720864)

[7 Functions Working Flow 23](#_Toc128720865)

[8 Complete Code of the Business Application 24](#_Toc128720866)

[9 Weaknesses in the Business Application 24](#_Toc128720867)

[10 Future Directions 24](#_Toc128720868)

# Short Description of Project:

In my project, the passenger is going to book his ticket and all the record will be given to the admin. The objective is to efficiently manage and coordinate all aspects of airport management system to ensure safety. Airport management systems are essential tools used in the aviation industry to manage and control various operations within an airport. These systems are highly computerized and use advanced software to streamline airport operations, improve efficiency, and enhance security. The use of airport management systems has significant benefits in the field of computer science, including: data Management, real - time information, security, efficiency. Airport management systems rely on computerized databases to store and manage data related to airport operations. This includes information on flights, passenger details, baggage tracking, and other critical data. The efficient handling of this data is a crucial aspect of airport management, and computer science plays a vital role in developing and maintaining these databases. The output given by management system at the end will be that it will print boarding passes for the passengers and for admin all the data of all passengers will be shown.

# Users of Application

An admin and number of passengers can access my management system.

## Admin

An admin can add, update, view and delete the flight schedules of different airlines. He can also view all the reviews and information of all the passengers added to the system. An admin can add recommendation and passenger trafficking reports.

## Passenger

A passenger is my main user who will give his personal information and then select particular airline and book his ticket after giving some info regarding his ticket. He can review the system in the end.

# Functional Requirements

## Admin

***As a I want to perform so that I can***

|  |  |  |
| --- | --- | --- |
| Admin | Add new flights | Give passenger the options |
| Admin | Check all the schedules | Ensure the system is working properly |
| Admin | Update the flights | Update my system |
| Admin | Delete a flight | Update my system |
| Admin | Task of checking tickets sold in a day | Maintain a record |
| Admin | Task of checking the passenger’s info | Maintain records in my system |
| Admin | I want to generate reports on passenger trafficking | For the management of system |
| Admin | I want to check the customer service | Handle the feedback and requests of passengers |
| Admin | I want to recommend the passenger the most sold ticket | Offer some recommendation to my passenger |
| Admin | Exit | Logout from the system. |

## Passenger

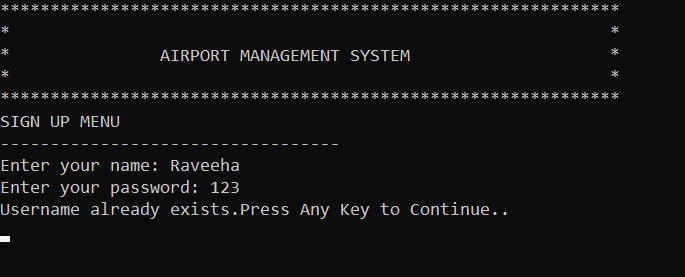
***As a I want to perform so that I can***

|  |  |  |
| --- | --- | --- |
| Passenger | Check which airlines are serving in the system | Book a flight |
| Passenger | Add my personal information | Give a record |
| Passenger | Enter the flight details | Tell my flight schedule |
| Passenger | Select Business or Economy class | Book my flight |
| Passenger | Meal selection | Select my meal during flight |
| Passenger | Add luggage details | Pay more in case of extra luggage |
| Passenger | Check discounts on my system | See my final price |
| Passenger | Confirmation of my booking | Confirm my booking |
| Passenger | Print the boarding pass | Get my boarding pass |
| Passenger | Give feedback of the service | Review the management system |
| Passenger | Exit | Logout. |

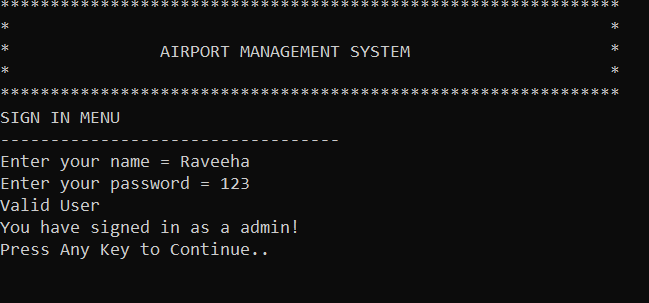
# Wireframes



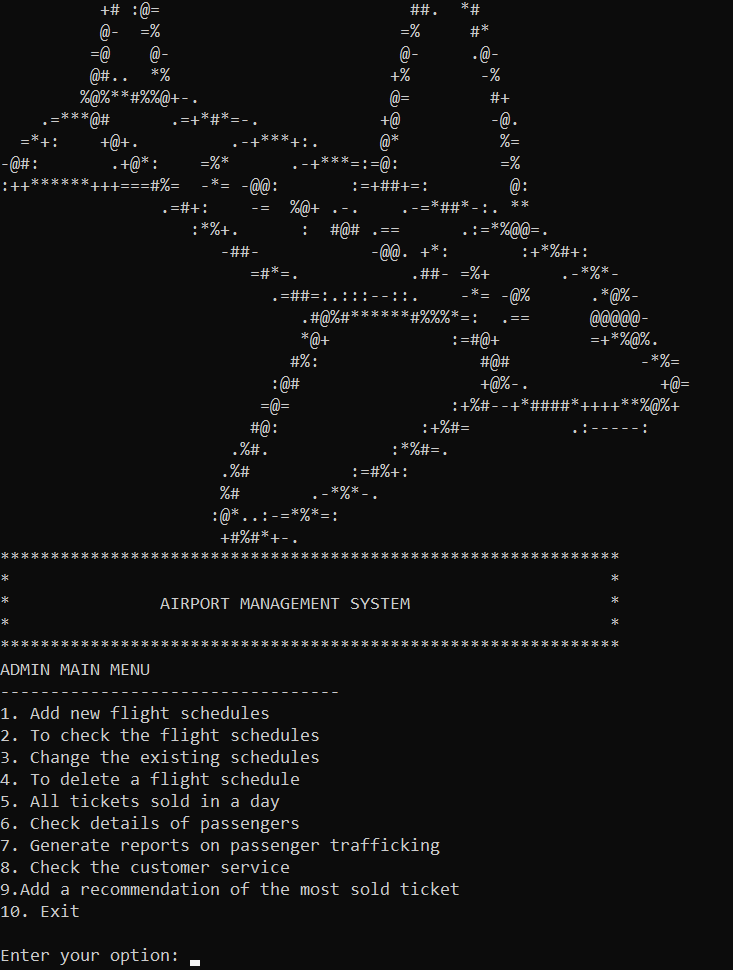
### Sign up & Sign in options



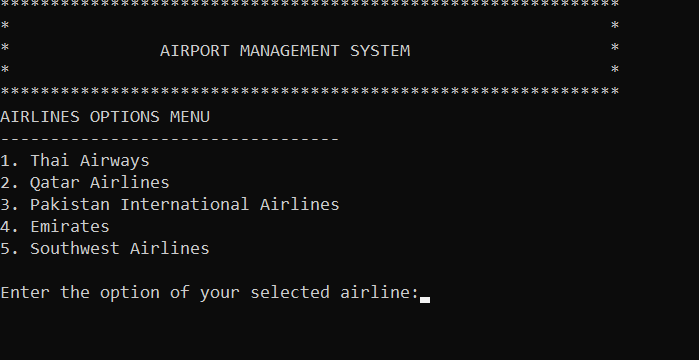
### Sign up for admin



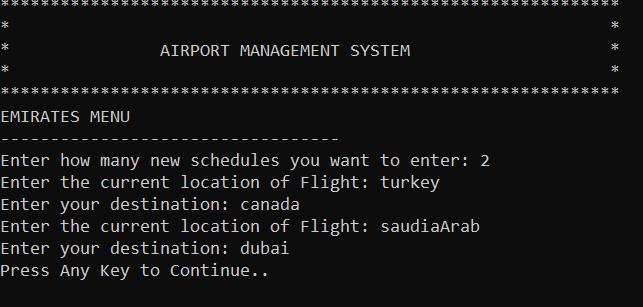
### Sign in Admin



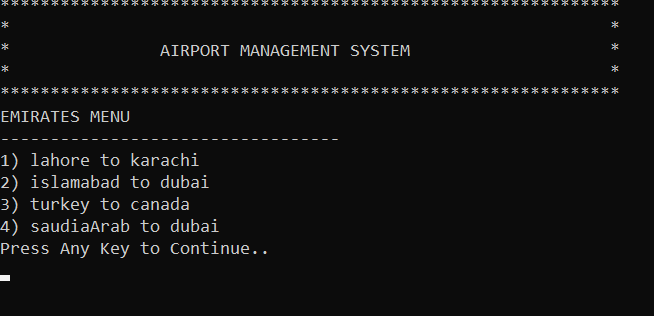
### Admin Interface



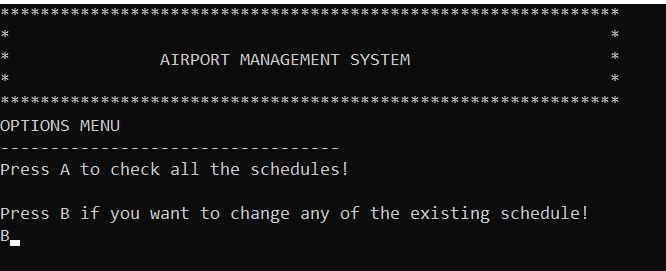
### Admin option 1



### Option 1 further options



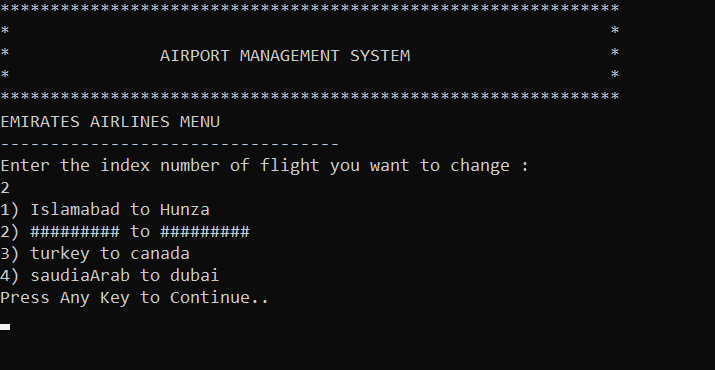
### Admin option 2



### Option 3



### Option 3 further menu

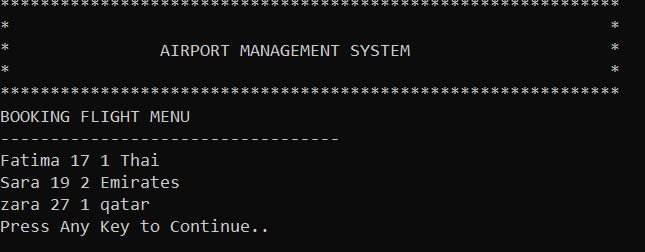


### 

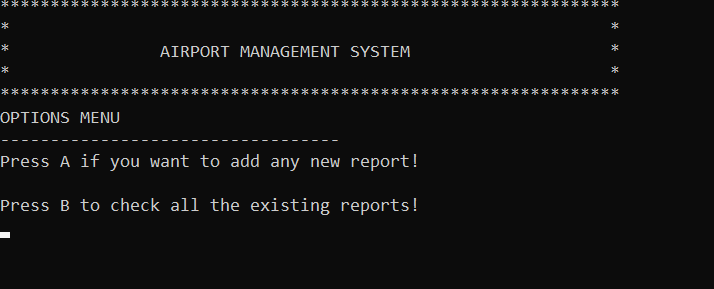
### Option 4

### 

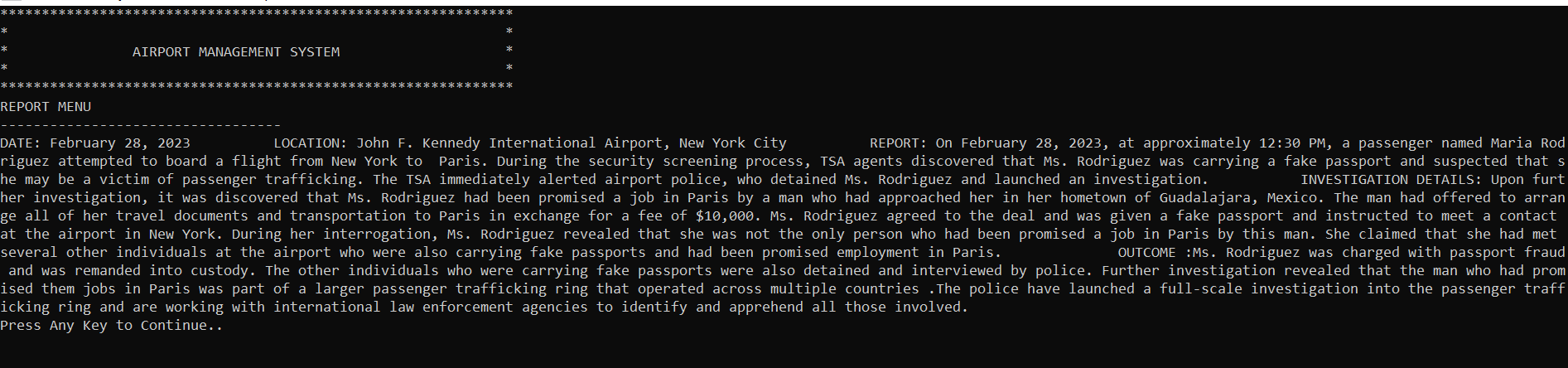
### Option 5(New record of passenger will be shown here)



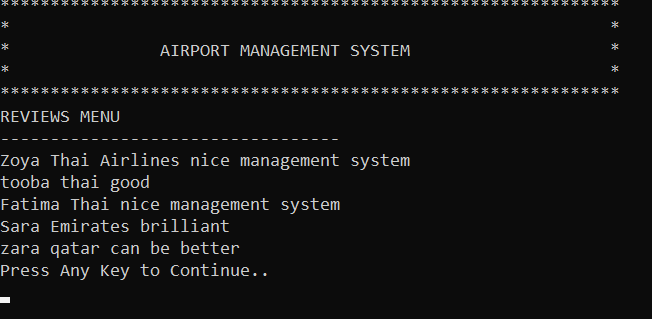
### Option 6 (new record will be shown here)



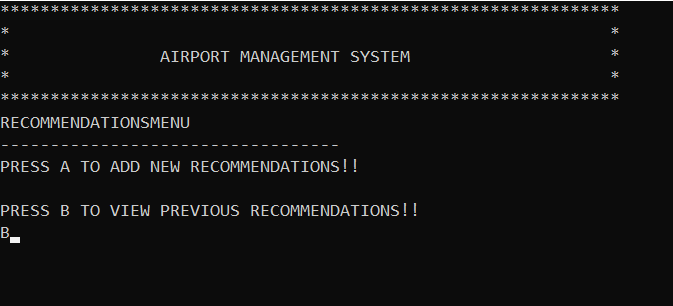
### Option 7



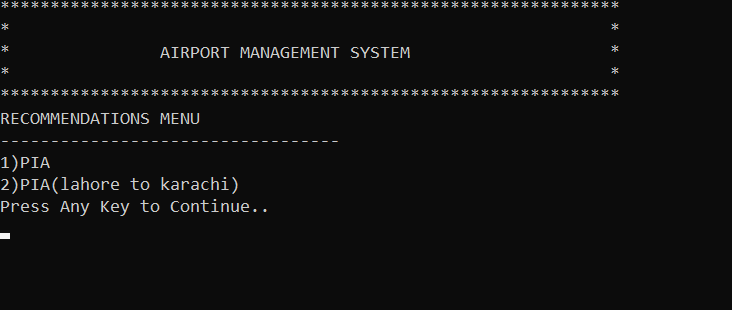
### Option 7 B opt



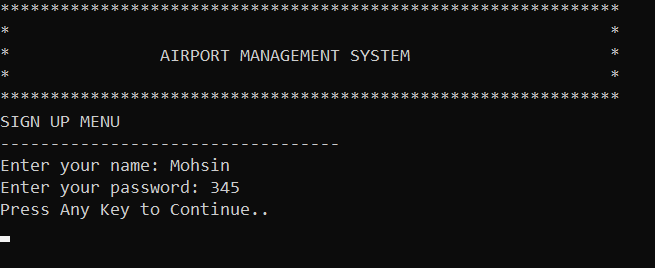
### Option 8



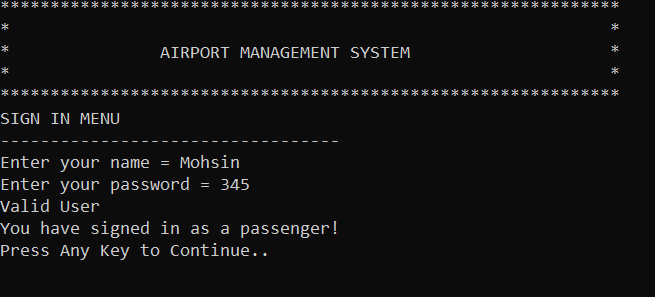
### Option 9



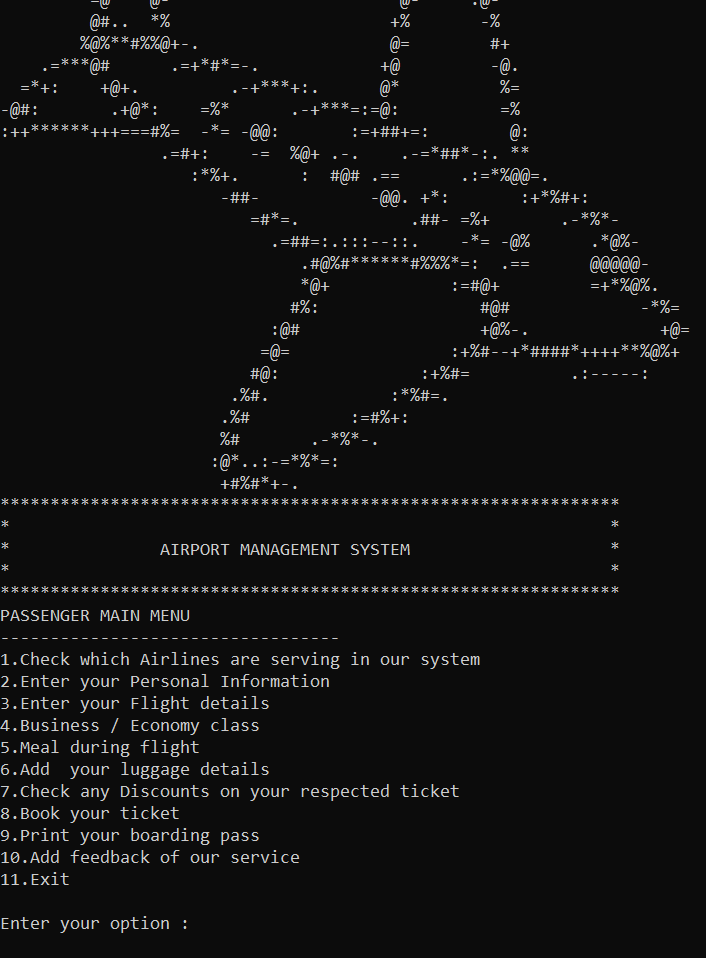
### Option 9 B



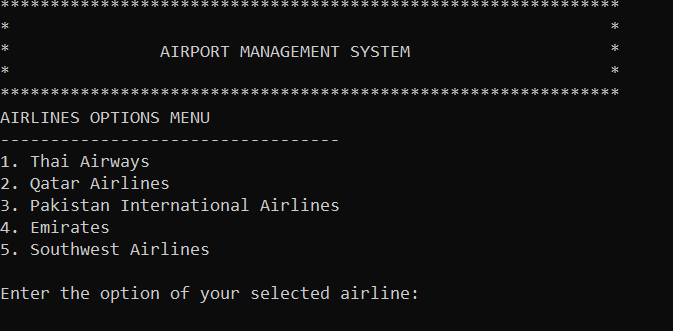
### Passenger Sign Up



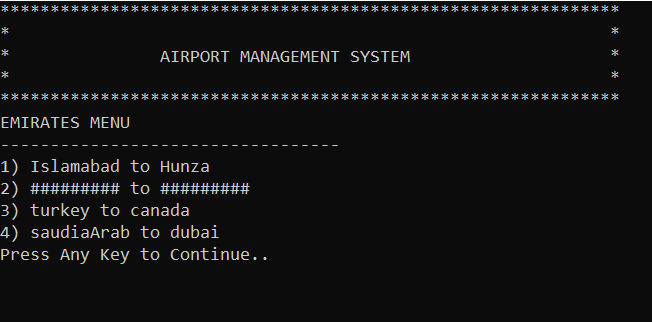
### Passenger Sign in



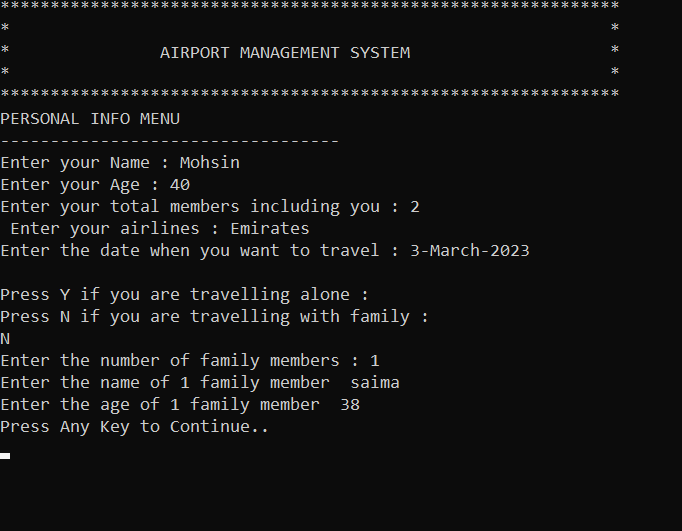
### Passenger Interface



### Option 1

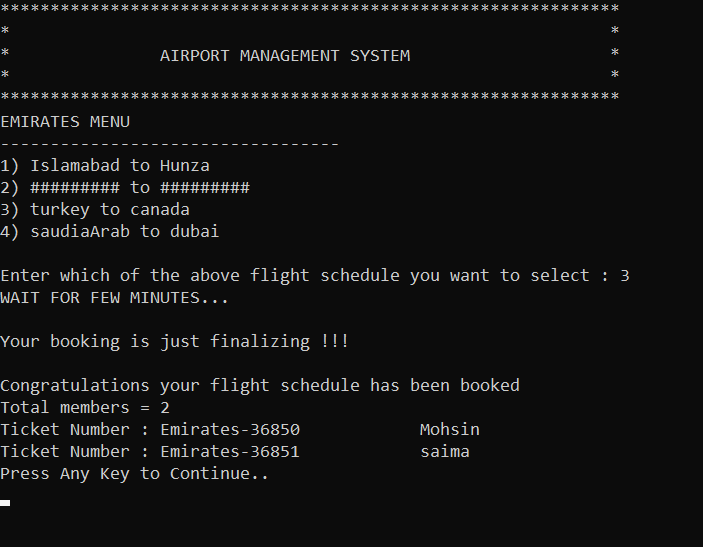


### Option 1 further menu

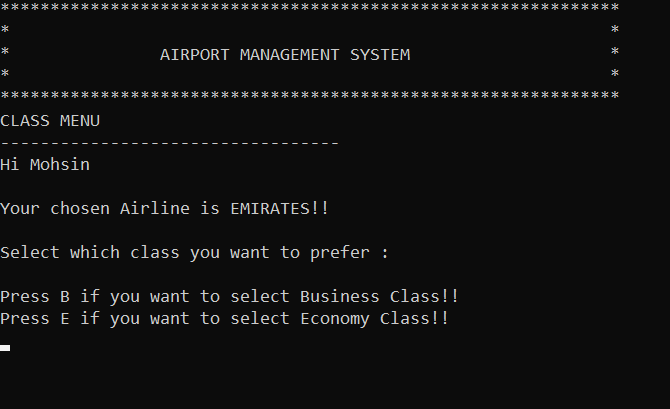


### Option 2

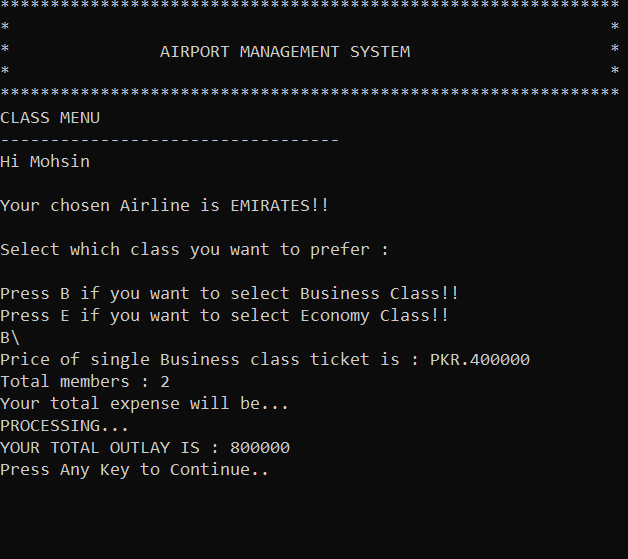
### Option 3



### Option 3 further menu

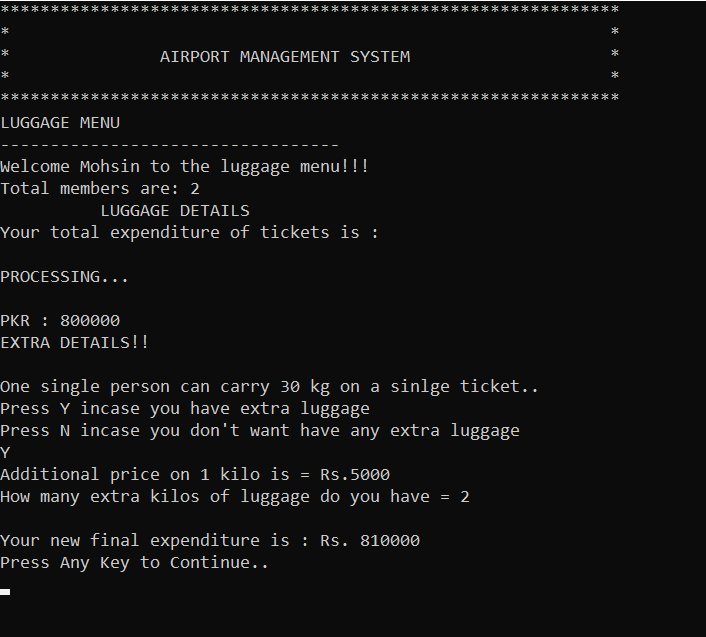


### Option 4

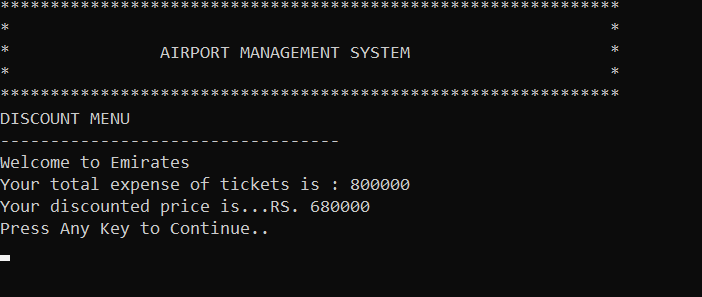


### Option 4 further submenu

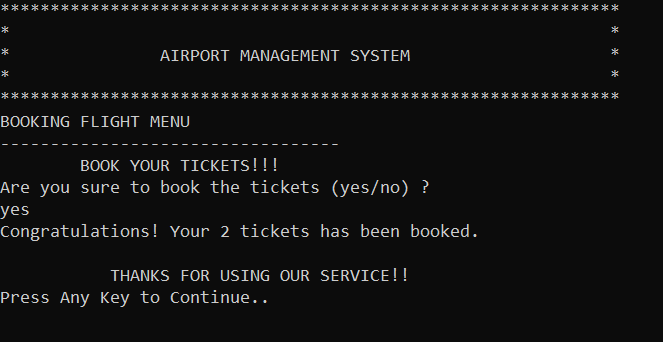
### Option 5



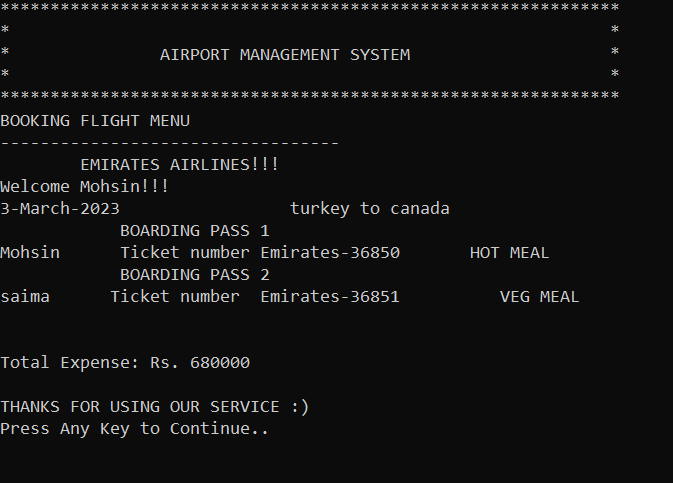
### Option 6



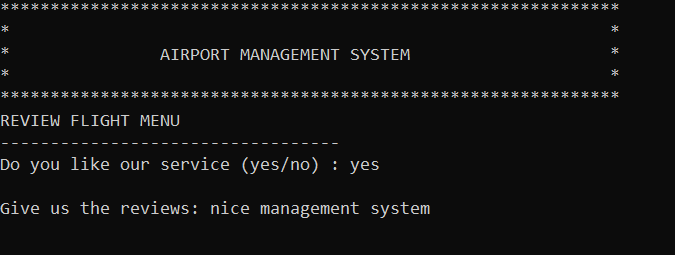
### Option 7



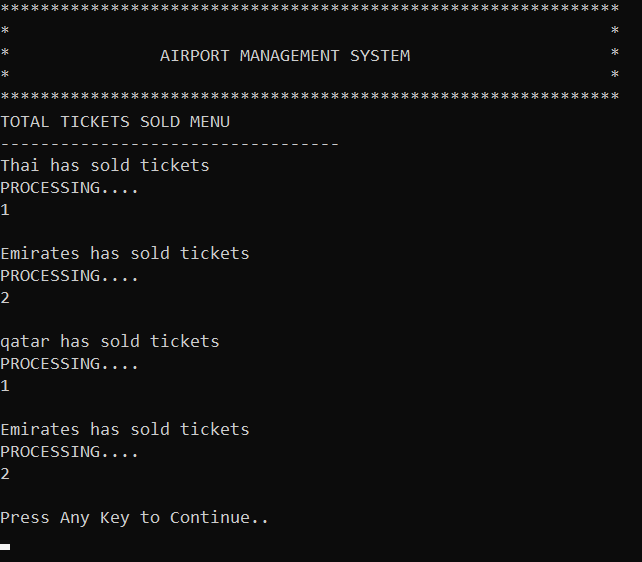
### Option 8



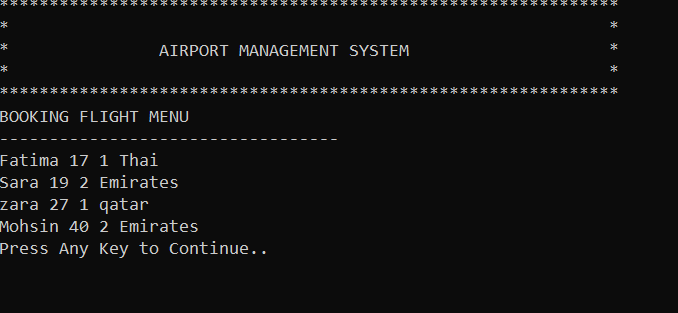
### Option 9



### Option 10



### Admin updated option 5



### Admin updated option 6

### Admin updated option 8

# Data Structure (Parallel Arrays)

***Arrays Arrays Variable Counter***

string users[arraysize] string passwords[arraysize] int usercount

string currentlocation1[1000] string desiredlocation1[1000] int input\_count1

string currentlocation2[1000] string desiredlocation2[1000] int input\_count2

string currentlocation3[1000] string desiredlocation3[1000] int input\_count3

string currentlocation4[1000] string desiredlocation4[1000] int input\_count4

string currentlocation5[1000] string desiredlocation5[1000] int input\_count5

string nameF[1000] int ageF[1000] int numberF

int menuoption[1000] int numberF

string report[1000] int optiontraffic

string recommendation2[1000] int idxrecommendation;

# Function Prototypes

|  |  |  |  |
| --- | --- | --- | --- |
| void printheader()  void submenuaftermenu(string submenu)  int loginmenu()  void signup(string name, string password)  bool signin(string name, string password);  int admininterface();  int passengerinterface();  void newflightschedule1();  void checkallschedules1();  void clearScreen();  void changeflights1();  void newflightschedule2();  void checkallschedules2();  void changeflights2  void bookschedule4();  void bookschedule5(); | void newflightschedule3()  void checkallschedules3();  void changeflights3();  void newflightschedule4()  void checkallschedules4();  void changeflights4();  void newflightschedule5()  void checkallschedules5();  void changeflights5();  char personalinfo();  void bookschedule1();  void bookschedule2();  void bookschedule3(); | void busEco();  void mealmenu1();  void mealmenu2();  void mealmenu3();  void mealmenu4();  void mealmenu5();  void luggagemenu();  void dicsountmenu();  void bookingticket();  void boardingpass1();  void boardingpass2();  void boardingpass3();  void boardingpass4();  void boardingpass5();  void ticketssold();  void passengertrafficking()  void storeUserInfo1();  void loaddata();  void storeUserInfo2();  void loaddata2();  void storeUserInfo3();  void loaddata3(); | void storeUserInfo4();  void loaddata4();  void storeUserInfo5();  void loaddata5();  void storeRecommendation();  void loadRecommendation();  void storePassengertrafficking();  void loadPassengerTrafficking()  void storecustomerservice();  void loadcustomerservice();  void loadtotalTickets();  void loadcheck();  bool isvalid(string name);  void loaduser();  void viewUsers();  void storeUserInfo(string name, string password);  string getfield(string line, int field); |

# Functions Working Flow

# Complete Code of the Business Application

#include <iostream>

#include <windows.h>

#include <conio.h>

#include <fstream>

#include <limits>

using namespace std;

const int arraysize = 1000;

string users[arraysize]; // to store users

string passwords[arraysize]; // to store passwords

string currentlocation1[1000]; // currentlocationforairline1

string desiredlocation1[1000]; // desiredlocationforairline1

string currentlocation2[1000]; // currentlocationforairline2

string desiredlocation2[1000]; // desiredlocationforairline2

string currentlocation3[1000]; // currentlocationforairline3

string desiredlocation3[1000]; // desiredlocationforairline3

string currentlocation4[1000]; // currentlocationforairline4

string desiredlocation4[1000]; // desiredlocationforairline4

string currentlocation5[1000]; // currentlocationforairline5

string desiredlocation5[1000]; // desiredlocationforairline5

int usercount = 0;

int schedule1 = 0;

int input\_count1 = 0;

int schedule2 = 0;

int input\_count2 = 0;

int schedule3 = 0;

int input\_count3 = 0;

int schedule4 = 0;

int input\_count4 = 0;

int schedule5 = 0;

int input\_count5 = 0;

string nameM; // name of the passenger

string page; // age of passenger

string pm; // total members

string pa; // airline name

string nameF[1000]; // names of family members

int ageF[1000]; // age of family members

int numberF = 0; // number of family members

int optionairline; // option of choosen airline

int menuoption[1000]; // menuoption for family members

int finalprice = 0; // ticket price stored here

string date; // date on which he is going to travel

int menuoption1; // menuoption for passenger

int flightnumber;

string booking;

string report[1000];

int optiontraffic = 0;

int idxtraffic = 0;

int recommendation1;

string recommendation2[1000];

int idxrecommendation;

int sumzero = 0;

bool flag;

void printheader(); // to print header

void submenuaftermenu(string submenu); // submenu

int loginmenu(); // login

void signup(string name, string password); // sign up

bool signin(string name, string password); // to check correct password & username

int admininterface(); // admininterface

int passengerinterface(); // passengerinterface

void newflightschedule1(); // add flights of thai airline

void checkallschedules1(); // check flights of thai airline

void clearScreen(); // if press any key then cls function will happen

void changeflights1(); // change flights of thai airline

void newflightschedule2(); // add flights of qatar

void checkallschedules2(); // check flights of qatar

void changeflights2(); // change flights of qatar

void newflightschedule3(); // add flights of pia

void checkallschedules3(); // check flights of pia

void changeflights3(); // change flights of PIA

void newflightschedule4(); // add flights of emirates

void checkallschedules4(); // check flights of qatar

void changeflights4(); // change flights of qatar

void newflightschedule5(); // add flights of southwest

void checkallschedules5(); // check flights of southwest

void changeflights5(); // change flights of southwest

char personalinfo(); // passenger info

void bookschedule1(); // book flight airline 1

void bookschedule2(); // book flight airline 2

void bookschedule3(); // book flight airline 3

void bookschedule4(); // book flight airline 4

void bookschedule5(); // book flight airline 5

void busEco(); // flight class type

void mealmenu1(); // meal airline 1

void mealmenu2(); // meal airline 2

void mealmenu3(); // meal airline 3

void mealmenu4(); // meal airline 4

void mealmenu5(); // meal airline 5

void luggagemenu(); // luggage menu

void dicsountmenu(); // discount

void bookingticket(); // final response of booking tickets

void boardingpass1(); // boarding pass for flight 1

void boardingpass2(); // boarding pass for flight 2

void boardingpass3(); // boarding pass for flight 3

void boardingpass4(); // boarding pass for flight 4

void boardingpass5(); // boarding pass for flight 5

string feedback(); // feedback of passenger

void ticketssold(); // total tickets sold

string review; // reviews

void passengertrafficking(); // reports

void deleteflights1(); // delete schedule of flight airline 1

void deleteflights2(); // delete schedule of flight airline 2

void deleteflights3(); // delete schedule of flight airline 3

void deleteflights4(); // delete schedule of flight airline 4

void deleteflights5(); // delete schedule of flight airline 5

void mainpic(); // plane pic

void passengertrafficking2(); // report

void recommendation(); // recommendations

void printrecommendation(); // recommendations

void storeUserInfo1(); // to store user info

void loaddata(); // to load

void storeUserInfo2(); // to store user info

void loaddata2(); // to load

void storeUserInfo3(); // to store user info

void loaddata3(); // to load

void storeUserInfo4(); // to store user info

void loaddata4(); // to load

void storeUserInfo5(); // to store user info

void loaddata5(); // to load

void storeRecommendation(); // store recommendations

void loadRecommendation(); // load recommendations

void storePassengertrafficking(); // store report

void loadPassengerTrafficking(); // load report

void storecustomerservice(); // to store customer service

void loadcustomerservice(); // load customer service

void loadtotalTickets(); // load the total tickets sold

void loadcheck(); // to load the passenger info

bool isvalid(string name); // username used only once

void loaduser(); // load users

void viewUsers(); // to view all users

void storeUserInfo(string name, string password); // to store usernames & passwords

string getfield(string line, int field); // to get the field

main()

{

loaduser();

loaddata();

loaddata2();

loaddata3();

loaddata4();

loaddata5();

loadRecommendation();

loadPassengerTrafficking();

while (true)

{

int option = 0;

while (option <= 4)

{

mainpic();

printheader();

submenuaftermenu("LOGIN ");

int option = loginmenu();

if (sumzero + option == 1)

{

bool decision;

system("cls");

string name;

string password;

printheader();

submenuaftermenu("SIGN UP ");

cout << "Enter your name: ";

getline(cin >> ws, name);

bool flag = true;

while (flag)

{

for (int idx = 0; idx < name.length(); idx++)

{

if (!((name[idx] >= 97 && name[idx] <= 122) || (name[idx] >= 65 && name[idx] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, name);

break;

}

else

{

flag = false;

}

}

}

cout << "Enter your password: ";

getline(cin >> ws, password);

bool flag1 = true;

while (flag1)

{

for (int idx = 0; idx < password.length(); idx++)

{

if (!(password[idx] >= 48 && password[idx] <= 57))

{

cout << "Invalid!! Enter Again" << endl;

cin >> password;

break;

}

else

{

flag1 = false;

}

}

}

decision = isvalid(name);

if (decision == true)

{

signup(name, password);

storeUserInfo(name, password);

}

else

{

cout << "Username already exists.";

}

clearScreen();

}

else if (sumzero + option == 3)

{

viewUsers();

clearScreen();

}

else if (sumzero + option == 4)

{

return 0;

}

else if (option == 2)

{

string name;

string password;

system("cls");

printheader();

submenuaftermenu("SIGN IN ");

cout << "Enter your name = ";

getline(cin >> ws, name);

bool flag2 = true;

while (flag2)

{

for (int idx = 0; idx < name.length(); idx++)

{

if (!((name[idx] >= 97 && name[idx] <= 122) || (name[idx] >= 65 && name[idx] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

cin >> name;

break;

}

else

{

flag2 = false;

}

}

}

cout << "Enter your password = ";

getline(cin >> ws, password);

bool flag3 = true;

while (flag3)

{

for (int idx = 0; idx < password.length(); idx++)

{

if (!(password[idx] >= 48 && password[idx] <= 57))

{

cout << "Invalid!! Enter Again" << endl;

cin >> password;

break;

}

else

{

flag3 = false;

}

}

}

flag = signin(name, password);

if (flag == false)

{

cout << "Invalid User" << endl;

}

else if (flag == true)

{

cout << "Valid User" << endl;

if (name == "Raveeha" && password == "123")

{

Sleep(90);

cout << "You have signed in as a admin!" << endl;

}

else

{

Sleep(90);

cout << "You have signed in as a passenger!" << endl;

}

if (name == "Raveeha" && password == "123")

{

clearScreen();

while (1)

{

mainpic();

printheader();

submenuaftermenu("ADMIN MAIN ");

int option = admininterface();

if (option == 1)

{

int optionairline;

system("cls");

printheader();

submenuaftermenu("AIRLINES OPTIONS ");

cout << "1. Thai Airways" << endl;

cout << "2. Qatar Airlines" << endl;

cout << "3. Pakistan International Airlines" << endl;

cout << "4. Emirates" << endl;

cout << "5. Southwest Airlines" << endl;

cout << endl;

cout << "Enter the option of your selected airline:";

while (!(cin >> optionairline))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

if (optionairline == 1)

{

system("cls");

printheader();

submenuaftermenu("THAI AIRLINES ");

newflightschedule1();

storeUserInfo1();

clearScreen();

}

else if (optionairline == 2)

{

system("cls");

printheader();

submenuaftermenu("QATAR AIRLINES ");

newflightschedule2();

storeUserInfo2();

clearScreen();

}

else if (optionairline == 3)

{

system("cls");

printheader();

submenuaftermenu("PIA ");

newflightschedule3();

storeUserInfo3();

clearScreen();

}

else if (optionairline == 4)

{

system("cls");

printheader();

submenuaftermenu("EMIRATES ");

newflightschedule4();

storeUserInfo4();

clearScreen();

}

else if (optionairline == 5)

{

system("cls");

printheader();

submenuaftermenu("SOUTHWEST AIRLINES ");

newflightschedule5();

storeUserInfo5();

clearScreen();

}

}

if (option == 2)

{

int optionairline;

system("cls");

printheader();

submenuaftermenu("AIRLINES OPTIONS ");

cout << "1. Thai Airways" << endl;

cout << "2. Qatar Airlines" << endl;

cout << "3. Pakistan International Airlines" << endl;

cout << "4. Emirates" << endl;

cout << "5. Southwest Airlines" << endl;

cout << endl;

cout << "Enter the option of your selected airline:";

while (!(cin >> optionairline))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

if (optionairline == 1)

{

system("cls");

printheader();

submenuaftermenu("THAI AIRLINES ");

checkallschedules1();

clearScreen();

}

else if (optionairline == 2)

{

system("cls");

printheader();

submenuaftermenu("QATAR AIRLINES ");

checkallschedules2();

clearScreen();

}

else if (optionairline == 3)

{

system("cls");

printheader();

submenuaftermenu("PIA ");

checkallschedules3();

clearScreen();

}

else if (optionairline == 4)

{

system("cls");

printheader();

submenuaftermenu("EMIRATES ");

checkallschedules4();

clearScreen();

}

else if (optionairline == 5)

{

system("cls");

printheader();

submenuaftermenu("SOUTHWEST AIRLINES ");

checkallschedules5();

clearScreen();

}

}

if (option == 3)

{

system("cls");

printheader();

submenuaftermenu("OPTIONS ");

char option;

cout << "Press A to check all the schedules!";

cout << endl;

cout << endl;

cout << "Press B if you want to change any of the existing schedule!";

cout << endl;

cin >> option;

while (!(option == 65 || option == 66))

{

cout << "Enter Again";

cin >> option;

}

if (option == 'A')

{

int optionairline;

system("cls");

printheader();

submenuaftermenu("AIRLINES OPTIONS ");

cout << "1. Thai Airways" << endl;

cout << "2. Qatar Airlines" << endl;

cout << "3. Pakistan International Airlines" << endl;

cout << "4. Emirates" << endl;

cout << "5. Southwest Airlines" << endl;

cout << endl;

cout << "Enter the option of your selected airline:";

while (!(cin >> optionairline))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

if (optionairline == 1)

{

system("cls");

printheader();

submenuaftermenu("THAI AIRLINES ");

checkallschedules1();

clearScreen();

}

else if (optionairline == 2)

{

system("cls");

printheader();

submenuaftermenu("QATAR AIRLINES ");

checkallschedules2();

clearScreen();

}

else if (optionairline == 3)

{

system("cls");

printheader();

submenuaftermenu("PIA ");

checkallschedules3();

clearScreen();

}

else if (optionairline == 4)

{

system("cls");

printheader();

submenuaftermenu("EMIRATES ");

checkallschedules4();

clearScreen();

}

else if (optionairline == 5)

{

system("cls");

printheader();

submenuaftermenu("SOUTHWEST AIRLINES ");

checkallschedules5();

clearScreen();

}

}

if (option == 'B')

{

system("cls");

printheader();

submenuaftermenu("FLIGHTS CHANGING ");

int optionairline;

system("cls");

printheader();

submenuaftermenu("AIRLINES OPTIONS ");

cout << "1. Thai Airways" << endl;

cout << "2. Qatar Airlines" << endl;

cout << "3. Pakistan International Airlines" << endl;

cout << "4. Emirates" << endl;

cout << "5. Southwest Airlines" << endl;

cout << endl;

cout << "Enter the option of your selected airline:";

while (!(cin >> optionairline))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

if (optionairline == 1)

{

system("cls");

printheader();

submenuaftermenu("THAI AIRLINES ");

changeflights1();

storeUserInfo1();

clearScreen();

}

else if (optionairline == 2)

{

system("cls");

printheader();

submenuaftermenu("QATAR AIRLINES ");

changeflights2();

storeUserInfo2();

clearScreen();

}

else if (optionairline == 3)

{

system("cls");

printheader();

submenuaftermenu("PIA ");

changeflights3();

storeUserInfo3();

clearScreen();

}

else if (optionairline == 4)

{

system("cls");

printheader();

submenuaftermenu("EMIRATES ");

changeflights4();

storeUserInfo4();

clearScreen();

}

else if (optionairline == 5)

{

system("cls");

printheader();

submenuaftermenu("SOUTHWEST AIRLINES ");

changeflights5();

storeUserInfo5();

clearScreen();

}

}

}

if (option == 4)

{

int optionairline;

system("cls");

printheader();

submenuaftermenu("AIRLINES OPTIONS ");

cout << "1. Thai Airways" << endl;

cout << "2. Qatar Airlines" << endl;

cout << "3. Pakistan International Airlines" << endl;

cout << "4. Emirates" << endl;

cout << "5. Southwest Airlines" << endl;

cout << endl;

cout << "Enter the option of your selected airline:";

while (!(cin >> optionairline))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

if (optionairline == 1)

{

system("cls");

printheader();

submenuaftermenu("THAI AIRLINES ");

deleteflights1();

storeUserInfo1();

clearScreen();

}

if (optionairline == 2)

{

system("cls");

printheader();

submenuaftermenu("QATAR AIRLINES ");

deleteflights2();

storeUserInfo2();

clearScreen();

}

if (optionairline == 3)

{

system("cls");

printheader();

submenuaftermenu("PAKISTAN INTERNATIONAL AIRLINES ");

deleteflights3();

storeUserInfo3();

clearScreen();

}

if (optionairline == 4)

{

system("cls");

printheader();

submenuaftermenu("EMIRATES AIRLINES ");

deleteflights4();

storeUserInfo4();

clearScreen();

}

if (optionairline == 5)

{

system("cls");

printheader();

submenuaftermenu("SOUTHWEST AIRLINES ");

deleteflights5();

storeUserInfo5();

clearScreen();

}

}

if (option == 5)

{

system("cls");

printheader();

submenuaftermenu("TOTAL TICKETS SOLD ");

loadtotalTickets();

clearScreen();

}

if (option == 6)

{

system("cls");

printheader();

submenuaftermenu("BOOKING FLIGHT ");

loadcheck();

clearScreen();

}

if (option == 7)

{

system("cls");

printheader();

submenuaftermenu("OPTIONS ");

char option;

cout << "Press A if you want to add any new report!";

cout << endl;

cout << endl;

cout << "Press B to check all the existing reports!";

cout << endl;

cin >> option;

while (!(option == 65 || option == 66))

{

cout << "Enter again" << endl;

cin >> option;

}

if (option == 'A')

{

system("cls");

printheader();

submenuaftermenu("REPORT ");

passengertrafficking();

storePassengertrafficking();

clearScreen();

}

if (option == 'B')

{

system("cls");

printheader();

submenuaftermenu("REPORT ");

passengertrafficking2();

clearScreen();

}

}

if (option == 8)

{

system("cls");

printheader();

submenuaftermenu("REVIEWS ");

loadcustomerservice();

clearScreen();

}

if (option == 9)

{

char option;

system("cls");

printheader();

submenuaftermenu("RECOMMENDATIONS");

cout << "PRESS A TO ADD NEW RECOMMENDATIONS!!" << endl

<< endl;

cout << "PRESS B TO VIEW PREVIOUS RECOMMENDATIONS!! " << endl;

cin >> option;

while (!(option == 65 || option == 66))

{

cout << "Try Again" << endl;

cin >> option;

}

if (option == 'A')

{

system("cls");

printheader();

submenuaftermenu("RECOMMENDATIONS ");

recommendation();

storeRecommendation();

clearScreen();

}

if (option == 'B')

{

system("cls");

printheader();

submenuaftermenu("RECOMMENDATIONS ");

printrecommendation();

clearScreen();

}

}

if (option == 10)

{

break;

}

}

}

else

{

clearScreen();

while (1)

{

mainpic();

printheader();

submenuaftermenu("PASSENGER MAIN ");

int optionA = passengerinterface();

if (optionA == 1)

{

system("cls");

printheader();

submenuaftermenu("FLIGHT SCHEDULE ");

int optionairline;

system("cls");

printheader();

submenuaftermenu("AIRLINES OPTIONS ");

cout << "1. Thai Airways" << endl;

cout << "2. Qatar Airlines" << endl;

cout << "3. Pakistan International Airlines" << endl;

cout << "4. Emirates" << endl;

cout << "5. Southwest Airlines" << endl;

cout << endl;

cout << "Enter the option of your selected airline:";

while (!(cin >> optionairline))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

if (optionairline == 1)

{

system("cls");

printheader();

submenuaftermenu("THAI AIRLINES ");

checkallschedules1();

clearScreen();

}

else if (optionairline == 2)

{

system("cls");

printheader();

submenuaftermenu("QATAR AIRLINES ");

checkallschedules2();

clearScreen();

}

else if (optionairline == 3)

{

system("cls");

printheader();

submenuaftermenu("PIA ");

checkallschedules3();

clearScreen();

}

else if (optionairline == 4)

{

system("cls");

printheader();

submenuaftermenu("EMIRATES ");

checkallschedules4();

clearScreen();

}

else if (optionairline == 5)

{

system("cls");

printheader();

submenuaftermenu("SOUTHWEST AIRLINES ");

checkallschedules5();

clearScreen();

}

}

if (optionA == 2)

{

system("cls");

printheader();

submenuaftermenu("PERSONAL INFO ");

char option = personalinfo();

if (option == 'Y')

{

clearScreen();

}

else if (option == 'N')

{

cout << "Enter the number of family members : ";

while (!(cin >> numberF))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < numberF; idx++)

{

cout << "Enter the name of " << idx + 1 << " family member ";

getline(cin >> ws, nameF[idx]);

bool flagfamname = true;

while (flagfamname)

{

for (int i = 0; i < nameF[idx].length(); i++)

{

if (!((nameF[idx][i] >= 97 && nameF[idx][i] <= 122) || (nameF[idx][i] >= 65 && nameF[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, nameF[idx]);

break;

}

else

{

flagfamname = false;

}

}

}

cout << "Enter the age of " << idx + 1 << " family member ";

while (!(cin >> ageF[idx]))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

}

clearScreen();

}

}

if (optionA == 3)

{

system("cls");

printheader();

submenuaftermenu("FLIGHT SCHEDULE ");

cout << "Dear " << nameM << "!!!" << endl

<< endl;

cout << " WELCOME TO THE FLIGHT MANAGEMENT SYSTEM " << endl

<< endl;

cout << "Which Airline you want to select ? " << endl;

cout << "1. Thai Airways" << endl;

cout << "2. Qatar Airlines" << endl;

cout << "3. Pakistan International Airlines" << endl;

cout << "4. Emirates" << endl;

cout << "5. Southwest Airlines" << endl;

cout << endl;

cout << "Enter the option of your selected airline:";

while (!(cin >> optionairline))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

if (optionairline == 1)

{

system("cls");

printheader();

submenuaftermenu("THAI AIRLINES ");

checkallschedules1();

bookschedule1();

clearScreen();

}

else if (optionairline == 2)

{

system("cls");

printheader();

submenuaftermenu("QATAR AIRLINES ");

checkallschedules2();

bookschedule2();

clearScreen();

}

else if (optionairline == 3)

{

system("cls");

printheader();

submenuaftermenu("PIA ");

checkallschedules3();

bookschedule3();

clearScreen();

}

else if (optionairline == 4)

{

system("cls");

printheader();

submenuaftermenu("EMIRATES ");

checkallschedules4();

bookschedule4();

clearScreen();

}

else if (optionairline == 5)

{

system("cls");

printheader();

submenuaftermenu("SOUTHWEST AIRLINES ");

checkallschedules5();

bookschedule5();

clearScreen();

}

}

if (optionA == 4)

{

system("cls");

printheader();

submenuaftermenu("CLASS ");

busEco();

cout << endl;

clearScreen();

}

if (optionA == 5)

{

system("cls");

printheader();

submenuaftermenu("FLIGHT MEAL ");

if (optionairline == 1)

{

mealmenu1();

clearScreen();

}

if (optionairline == 2)

{

mealmenu2();

clearScreen();

}

if (optionairline == 3)

{

mealmenu3();

clearScreen();

}

if (optionairline == 4)

{

mealmenu4();

clearScreen();

}

if (optionairline == 5)

{

mealmenu5();

clearScreen();

}

}

if (optionA == 6)

{

system("cls");

printheader();

submenuaftermenu("LUGGAGE ");

luggagemenu();

clearScreen();

}

if (optionA == 7)

{

system("cls");

printheader();

submenuaftermenu("DISCOUNT ");

dicsountmenu();

cout << endl;

clearScreen();

}

if (optionA == 8)

{

system("cls");

printheader();

submenuaftermenu("BOOKING FLIGHT ");

bookingticket();

cout << endl;

clearScreen();

}

if (optionA == 9)

{

if (booking == "yes" || booking == "YES" || booking == "Yes")

{

system("cls");

printheader();

submenuaftermenu("BOOKING FLIGHT ");

if (optionairline == 1)

{

boardingpass1();

cout << endl;

clearScreen();

}

if (optionairline == 2)

{

boardingpass2();

cout << endl;

clearScreen();

}

if (optionairline == 3)

{

boardingpass3();

cout << endl;

clearScreen();

}

if (optionairline == 4)

{

boardingpass4();

cout << endl;

clearScreen();

}

if (optionairline == 5)

{

boardingpass5();

cout << endl;

clearScreen();

}

}

else

{

cout << "SORRY...you have not yet booked your tickets!!" << endl;

clearScreen();

}

}

if (optionA == 10)

{

system("cls");

printheader();

submenuaftermenu("REVIEW FLIGHT ");

feedback();

storecustomerservice();

clearScreen();

}

if (optionA == 11)

{

break;

}

}

}

}

clearScreen();

}

}

}

}

void printheader()

{

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\* \*" << endl;

cout << "\* AIRPORT MANAGEMENT SYSTEM \*" << endl;

cout << "\* \*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

void submenuaftermenu(string submenu)

{

string menu = submenu + "MENU";

cout << menu << endl;

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

}

void clearScreen()

{

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

getch();

system("cls");

}

int loginmenu()

{

int option;

cout << "1. Sign up to get your credentials " << endl;

cout << "2. Sign in with your credentials " << endl;

cout << "3. View all the users " << endl;

cout << "4. Exit your application " << endl;

cout << "Enter your option: ";

while (!(cin >> option))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

return option;

}

void signup(string name, string password)

{

users[usercount] = name;

passwords[usercount] = password;

usercount++;

}

bool signin(string name, string password)

{

bool flag = false;

for (int idx = 0; idx < usercount; idx++)

{

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

{

flag = true;

break;

}

}

return flag;

}

int admininterface()

{

int option;

cout << "1. Add new flight schedules" << endl;

cout << "2. To check the flight schedules " << endl;

cout << "3. Change the existing schedules " << endl;

cout << "4. To delete a flight schedule " << endl;

cout << "5. All tickets sold in a day" << endl;

cout << "6. Check details of passengers " << endl;

cout << "7. Generate reports on passenger trafficking" << endl;

cout << "8. Check the customer service" << endl;

cout << "9.Add a recommendation of the most sold ticket" << endl;

cout << "10. Exit" << endl;

cout << endl;

cout << "Enter your option: ";

while (!(cin >> option))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

return option;

}

void newflightschedule1()

{

cout << "Enter how many new schedules you want to enter: ";

cin >> schedule1;

int idx = 0;

while (idx != schedule1)

{

cout << "Enter the current location of Flight: ";

getline(cin >> ws, currentlocation1[input\_count1]);

bool flag4 = true;

while (flag4)

{

for (int i = 0; i < currentlocation1[input\_count1].length(); i++)

{

if (!((currentlocation1[input\_count1][i] >= 97 && currentlocation1[input\_count1][i] <= 122) || (currentlocation1[input\_count1][i] >= 65 && currentlocation1[input\_count1][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, currentlocation1[input\_count1]);

break;

}

else

{

flag4 = false;

}

}

}

cout << "Enter your destination: ";

getline(cin >> ws, desiredlocation1[input\_count1]);

bool flag5 = true;

while (flag5)

{

for (int i = 0; i < desiredlocation1[input\_count1].length(); i++)

{

if (!((desiredlocation1[input\_count1][i] >= 97 && desiredlocation1[input\_count1][i] <= 122) || (desiredlocation1[input\_count1][i] >= 65 && desiredlocation1[input\_count1][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, desiredlocation1[input\_count1]);

break;

}

else

{

flag5 = false;

}

}

}

input\_count1++;

idx++;

}

}

void storeUserInfo1()

{

fstream file1;

file1.open("InformationOfFlights.txt", ios ::out);

int idx = 0;

while (idx < input\_count1)

{

file1 << currentlocation1[idx] << ",";

file1 << desiredlocation1[idx] << endl;

idx++;

}

file1.close();

}

void loaddata()

{

string line;

fstream file1;

file1.open("InformationOfFlights.txt", ios ::in);

while (getline(file1, line))

{

currentlocation1[input\_count1] = getfield(line, 1);

desiredlocation1[input\_count1] = getfield(line, 2);

input\_count1++;

}

file1.close();

}

void checkallschedules1()

{

for (int idx = 0; idx < input\_count1; idx++)

{

cout << idx + 1 << ") " << currentlocation1[idx] + " " + "to" + " " + desiredlocation1[idx] << endl;

}

}

void changeflights1()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

cin >> number;

for (int idx = 0; idx < input\_count1; idx++)

{

if (number == idx + 1)

{ cout << idx + 1 << ") " << currentlocation1[idx] + " " + "to" + " " + desiredlocation1[idx] << endl

<< endl;

cout << "Enter new current location: ";

getline(cin >> ws, currentlocation1[idx]);

bool flag6 = true;

while (flag6)

{

for (int i = 0; i < currentlocation1[idx].length(); i++)

{

if (!((currentlocation1[idx][i] >= 97 && currentlocation1[idx][i] <= 122) || (currentlocation1[idx][i] >= 65 && currentlocation1[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

cin >> currentlocation1[idx];

break;

}

else

{

flag6 = false;

}

}

}

cout << "Enter new desired location: ";

getline(cin >> ws, desiredlocation1[idx]);

bool flag7 = true;

while (flag7)

{

for (int i = 0; i < desiredlocation1[idx].length(); i++)

{

if (!((desiredlocation1[idx][i] >= 97 && desiredlocation1[idx][i] <= 122) || (desiredlocation1[idx][i] >= 65 && desiredlocation1[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

cin >> desiredlocation1[idx];

break;

}

else

{

flag7 = false;

}

}

}

cout << "Your new updated flight schedule is as follows: " << endl;

cout << idx + 1 << ") " << currentlocation1[idx] + " " + "to" + " " + desiredlocation1[idx] << endl;

}

}

}

void newflightschedule2()

{ cout << "Enter how many new schedules you want to enter: ";

cin >> schedule2;

int idx = 0;

while (idx != schedule2)

{

cout << "Enter the current location of Flight: ";

getline(cin >> ws, currentlocation2[input\_count2]);

bool flag8 = true;

while (flag8)

{

for (int i = 0; i < currentlocation2[input\_count2].length(); i++)

{

if (!((currentlocation2[input\_count2][i] >= 97 && currentlocation2[input\_count2][i] <= 122) || (currentlocation2[input\_count2][i] >= 65 && currentlocation2[input\_count2][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, currentlocation2[input\_count2]);

break;

}

else

{

flag8 = false;

}

}

}

cout << "Enter your destination: ";

getline(cin >> ws, desiredlocation2[input\_count2]);

bool flag9 = true;

while (flag9)

{

for (int i = 0; i < desiredlocation2[input\_count2].length(); i++)

{

if (!((desiredlocation2[input\_count2][i] >= 97 && desiredlocation2[input\_count2][i] <= 122) || (desiredlocation2[input\_count2][i] >= 65 && desiredlocation2[input\_count2][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, desiredlocation2[input\_count2]);

break;

}

else

{

flag9 = false;

}

}

}

input\_count2++;

idx++;

}

}

void storeUserInfo2()

{

fstream file1;

file1.open("InformationOfFlightsQATAR.txt", ios ::out);

int idx = 0;

while (idx < input\_count2)

{

file1 << currentlocation2[idx] << ",";

file1 << desiredlocation2[idx] << endl;

idx++;

}

file1.close();

}

void loaddata2()

{

string line;

fstream file1;

file1.open("InformationOfFlightsQATAR.txt", ios ::in);

while (getline(file1, line))

{

currentlocation2[input\_count2] = getfield(line, 1);

desiredlocation2[input\_count2] = getfield(line, 2);

input\_count2++;

}

file1.close();

}

void checkallschedules2()

{

for (int idx = 0; idx < input\_count2; idx++)

{

cout << idx + 1 << ") " << currentlocation2[idx] + " " + "to" + " " + desiredlocation2[idx] << endl;

}

void changeflights2()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

cin >> number;

for (int idx = 0; idx < input\_count2; idx++)

{

if (number == idx + 1)

{ cout << idx + 1 << ") " << currentlocation2[idx] + " " + "to" + " " + desiredlocation2[idx] << endl

<< endl;

cout << "Enter new current location: ";

getline(cin >> ws, currentlocation2[idx]);

bool flag10 = true;

while (flag10)

{

for (int i = 0; i < currentlocation2[idx].length(); i++)

{

if (!((currentlocation2[idx][i] >= 97 && currentlocation2[idx][i] <= 122) || (currentlocation2[idx][i] >= 65 && currentlocation2[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, currentlocation2[idx]);

break;

}

else

{

flag10 = false;

}

}

}

cout << "Enter your destination: ";

getline(cin >> ws, desiredlocation2[idx]);

bool flag11 = true;

while (flag11)

{

for (int i = 0; i < desiredlocation2[idx].length(); i++)

{

if (!((desiredlocation2[idx][i] >= 97 && desiredlocation2[idx][i] <= 122) || (desiredlocation2[idx][i] >= 65 && desiredlocation2[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, desiredlocation2[idx]);

break;

}

else

{

flag11 = false;

}

}

}

cout << "Your new updated flight schedule is as follows: " << endl;

cout << idx + 1 << ") " << currentlocation2[idx] + " " + "to" + " " + desiredlocation2[idx] << endl;

}

}

}

void newflightschedule3()

{ cout << "Enter how many new schedules you want to enter: ";

cin >> schedule3;

int idx = 0;

while (idx != schedule3)

{ cout << "Enter the current location of Flight: ";

getline(cin >> ws, currentlocation3[input\_count3]);

bool flag12 = true;

while (flag12)

{

for (int i = 0; i < currentlocation3[input\_count3].length(); i++)

{

if (!((currentlocation3[input\_count3][i] >= 97 && currentlocation3[input\_count3][i] <= 122) || (currentlocation3[input\_count3][i] >= 65 && currentlocation3[input\_count3][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, currentlocation3[input\_count3]);

break;

}

else

{

flag12 = false;

}

}

}

cout << "Enter your destination: ";

getline(cin >> ws, desiredlocation3[input\_count3]);

bool flag13 = true;

while (flag13)

{

for (int i = 0; i < desiredlocation3[input\_count3].length(); i++)

{

if (!((desiredlocation3[input\_count3][i] >= 97 && desiredlocation3[input\_count3][i] <= 122) || (desiredlocation3[input\_count3][i] >= 65 && desiredlocation3[input\_count3][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, desiredlocation3[input\_count3]);

break;

}

else

{

flag13 = false;

}

}

}

input\_count3++;

idx++;

}

}

void storeUserInfo3()

{ fstream file1;

file1.open("InformationOfFlightsPIA.txt", ios ::out);

int idx = 0;

while (idx < input\_count3)

{ file1 << currentlocation3[idx] << ",";

file1 << desiredlocation3[idx] << endl;

idx++;

}

file1.close();

}

void loaddata3()

{

string line;

fstream file1;

file1.open("InformationOfFlightsPIA.txt", ios ::in);

while (getline(file1, line))

{ currentlocation3[input\_count3] = getfield(line, 1);

desiredlocation3[input\_count3] = getfield(line, 2);

input\_count3++;

}

file1.close();

}

void checkallschedules3()

{

for (int idx = 0; idx < input\_count3; idx++)

{

cout << idx + 1 << ") " << currentlocation3[idx] + " " + "to" + " " + desiredlocation3[idx] << endl;

}

}

void changeflights3()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

cin >> number;

for (int idx = 0; idx < input\_count3; idx++)

{

if (number == idx + 1)

{

cout << idx + 1 << ") " << currentlocation3[idx] + " " + "to" + " " + desiredlocation3[idx] << endl

<< endl;

cout << "Enter new current location: ";

getline(cin >> ws, currentlocation3[idx]);

bool flag14 = true;

while (flag14)

{

for (int i = 0; i < currentlocation3[idx].length(); i++)

{

if (!((currentlocation3[idx][i] >= 97 && currentlocation3[idx][i] <= 122) || (currentlocation3[idx][i] >= 65 && currentlocation3[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, currentlocation3[idx]);

break;

}

else

{

flag14 = false;

}

}

}

cout << "Enter your destination: ";

getline(cin >> ws, desiredlocation3[idx]);

bool flag15 = true;

while (flag15)

{

for (int i = 0; i < desiredlocation3[idx].length(); i++)

{

if (!((desiredlocation3[idx][i] >= 97 && desiredlocation3[idx][i] <= 122) || (desiredlocation3[idx][i] >= 65 && desiredlocation3[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, desiredlocation3[idx]);

break;

}

else

{

flag15 = false;

}

}

}

cout << "Your new updated flight schedule is as follows: " << endl;

cout << idx + 1 << ") " << currentlocation3[idx] + " " + "to" + " " + desiredlocation3[idx] << endl;

}

}

}

void newflightschedule4()

{ cout << "Enter how many new schedules you want to enter: ";

cin >> schedule4;

int idx = 0;

while (idx != schedule4)

{

cout << "Enter the current location of Flight: ";

getline(cin >> ws, currentlocation4[input\_count4]);

bool flag16 = true;

while (flag16)

{

for (int i = 0; i < currentlocation4[input\_count4].length(); i++)

{

if (!((currentlocation4[input\_count4][i] >= 97 && currentlocation4[input\_count4][i] <= 122) || (currentlocation4[input\_count4][i] >= 65 && currentlocation4[input\_count4][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, currentlocation4[input\_count4]);

break;

}

else

{

flag16 = false;

}

}

}

cout << "Enter your destination: ";

getline(cin >> ws, desiredlocation4[input\_count4]);

bool flag17 = true;

while (flag17)

{

for (int i = 0; i < desiredlocation4[input\_count4].length(); i++)

{

if (!((desiredlocation4[input\_count4][i] >= 97 && desiredlocation4[input\_count4][i] <= 122) || (desiredlocation4[input\_count4][i] >= 65 && desiredlocation4[input\_count4][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, desiredlocation4[input\_count4]);

break;

}

else

{

flag17 = false;

}

}

}

input\_count4++;

idx++;

}

}

void storeUserInfo4()

{

fstream file1;

file1.open("InformationOfFlightsEMIRATES.txt", ios ::out);

int idx = 0;

while (idx < input\_count4)

{

file1 << currentlocation4[idx] << ",";

file1 << desiredlocation4[idx] << endl;

idx++;

}

file1.close();

}

void loaddata4()

{

string line;

fstream file1;

file1.open("InformationOfFlightsEMIRATES.txt", ios ::in);

while (getline(file1, line))

{ currentlocation4[input\_count4] = getfield(line, 1);

desiredlocation4[input\_count4] = getfield(line, 2);

input\_count4++;

}

file1.close();

}

void checkallschedules4()

{

for (int idx = 0; idx < input\_count4; idx++)

{

cout << idx + 1 << ") " << currentlocation4[idx] + " " + "to" + " " + desiredlocation4[idx] << endl;

}

}

void changeflights4()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

cin >> number;

for (int idx = 0; idx < input\_count4; idx++)

{

if (number == idx + 1)

{

cout << idx + 1 << ") " << currentlocation4[idx] + " " + "to" + " " + desiredlocation4[idx] << endl

<< endl;

cout << "Enter new current location: ";

getline(cin >> ws, currentlocation4[idx]);

bool flag18 = true;

while (flag18)

{

for (int i = 0; i < currentlocation4[idx].length(); i++)

{

if (!((currentlocation4[idx][i] >= 97 && currentlocation4[idx][i] <= 122) || (currentlocation4[idx][i] >= 65 && currentlocation4[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, currentlocation4[idx]);

break;

}

else

{

flag18 = false;

}

}

}

cout << "Enter your destination: ";

getline(cin >> ws, desiredlocation4[idx]);

bool flag19 = true;

while (flag19)

{

for (int i = 0; i < desiredlocation4[idx].length(); i++)

{

if (!((desiredlocation4[idx][i] >= 97 && desiredlocation4[idx][i] <= 122) || (desiredlocation4[idx][i] >= 65 && desiredlocation4[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, desiredlocation4[idx]);

break;

}

else

{

flag19 = false;

}

}

}

cout << "Your new updated flight schedule is as follows: " << endl;

cout << idx + 1 << ") " << currentlocation4[idx] + " " + "to" + " " + desiredlocation4[idx] << endl;

}

}

}

void newflightschedule5()

{

cout << "Enter how many new schedules you want to enter: ";

cin >> schedule5;

int idx = 0;

while (idx != schedule5)

{

cout << "Enter the current location of Flight: ";

getline(cin >> ws, currentlocation5[input\_count5]);

bool flag20 = true;

while (flag20)

{

for (int i = 0; i < currentlocation5[input\_count5].length(); i++)

{

if (!((currentlocation5[input\_count5][i] >= 97 && currentlocation5[input\_count5][i] <= 122) || (currentlocation5[input\_count5][i] >= 65 && currentlocation5[input\_count5][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, currentlocation5[input\_count5]);

break;

}

else

{

flag20 = false;

}

}

}

cout << "Enter your destination: ";

getline(cin >> ws, desiredlocation5[input\_count5]);

bool flag21 = true;

while (flag21)

{

for (int i = 0; i < desiredlocation5[input\_count5].length(); i++)

{

if (!((desiredlocation5[input\_count5][i] >= 97 && desiredlocation5[input\_count5][i] <= 122) || (desiredlocation5[input\_count5][i] >= 65 && desiredlocation5[input\_count5][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, desiredlocation5[input\_count5]);

break;

}

else

{

flag21 = false;

}

}

}

input\_count5++;

idx++;

}

}

void storeUserInfo5()

{

fstream file1;

file1.open("InformationOfFlightsSOUTHWESTAIRLINE.txt", ios ::out);

int idx = 0;

while (idx < input\_count5)

{

file1 << currentlocation5[idx] << ",";

file1 << desiredlocation5[idx] << endl;

idx++;

}

file1.close();

}

void loaddata5()

{

string line;

fstream file1;

file1.open("InformationOfFlightsSOUTHWESTAIRLINE.txt", ios ::in);

while (getline(file1, line))

{

currentlocation5[input\_count5] = getfield(line, 1);

desiredlocation5[input\_count5] = getfield(line, 2);

input\_count5++;

}

file1.close();

}

void checkallschedules5()

{

for (int idx = 0; idx < input\_count5; idx++)

{

cout << idx + 1 << ") " << currentlocation5[idx] + " " + "to" + " " + desiredlocation5[idx] << endl;

}

}

void changeflights5()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

cin >> number;

for (int idx = 0; idx < input\_count5; idx++)

{

if (number == idx + 1)

{

cout << idx + 1 << ") " << currentlocation5[idx] + " " + "to" + " " + desiredlocation5[idx] << endl

<< endl;

cout << "Enter new current location: ";

getline(cin >> ws, currentlocation5[idx]);

bool flag22 = true;

while (flag22)

{

for (int i = 0; i < currentlocation5[idx].length(); i++)

{

if (!((currentlocation5[idx][i] >= 97 && currentlocation5[idx][i] <= 122) || (currentlocation5[idx][i] >= 65 && currentlocation5[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, currentlocation5[idx]);

break;

}

else

{

flag22 = false;

}

}

}

cout << "Enter your destination: ";

getline(cin >> ws, desiredlocation5[idx]);

bool flag23 = true;

while (flag23)

{

for (int i = 0; i < desiredlocation5[idx].length(); i++)

{

if (!((desiredlocation5[idx][i] >= 97 && desiredlocation5[idx][i] <= 122) || (desiredlocation5[idx][i] >= 65 && desiredlocation5[idx][i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

getline(cin >> ws, desiredlocation5[idx]);

break;

}

else

{

flag23 = false;

}

}

}

cout << "Your new updated flight schedule is as follows: " << endl;

cout << idx + 1 << ") " << currentlocation5[idx] + " " + "to" + " " + desiredlocation5[idx] << endl;

}

}

}

int passengerinterface()

{

int optionP;

cout << "1.Check which Airlines are serving in our system" << endl;

cout << "2.Enter your Personal Information" << endl;

cout << "3.Enter your Flight details" << endl;

cout << "4.Business / Economy class" << endl;

cout << "5.Meal during flight" << endl;

cout << "6.Add your luggage details" << endl;

cout << "7.Check any Discounts on your respected ticket" << endl;

cout << "8.Book your ticket " << endl;

cout << "9.Print your boarding pass" << endl;

cout << "10.Add feedback of our service" << endl;

cout << "11.Exit" << endl;

cout << endl;

cout << "Enter your option : ";

while (!(cin >> optionP))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

return optionP;

}

void loadcheck()

{

string line;

fstream file1;

file1.open("check info.txt", ios ::in);

while (getline(file1, line))

{

cout << getfield(line, 1) << " ";

cout << getfield(line, 2) << " ";

cout << getfield(line, 3) << " ";

cout << getfield(line, 4) << endl;

}

file1.close();

}

char personalinfo()

{

char option;

cout << "Enter your Name : ";

cin >> ws;

getline(cin, nameM);

bool flagname = true;

while (flagname)

{

for (int i = 0; i < nameM.length(); i++)

{

if (!((nameM[i] >= 97 && nameM[i] <= 122) || (nameM[i] >= 65 && nameM[i] <= 90)))

{

cout << "Invalid!! Enter Again" << endl;

cin >> nameM;

break;

}

else

{

flagname = false;

}

}

}

cout << "Enter your Age : ";

while (!(cin >> page))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

cout << "Enter your total members including you : ";

while (!(cin >> pm))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

cout << " Enter your airlines : ";

getline(cin >> ws, pa);

cout << "Enter the date when you want to travel : ";

getline(cin >> ws, date);

cout << endl;

fstream file1;

file1.open("check info.txt", ios ::app);

file1 << nameM << "," << page << "," << pm << "," << pa << endl;

file1.close();

ticketssold();

cout << "Press Y if you are travelling alone : " << endl;

cout << "Press N if you are travelling with family : " << endl;

cin >> option;

while (!(option == 89 || option == 78))

{

cout << "Try Again" << endl;

cin >> option;

}

return option;

}

void bookschedule1()

{

cout << endl;

cout << "Enter which of the above flight schedule you want to select : ";

while (!(cin >> flightnumber))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count1; idx++)

{

if (flightnumber == idx + 1)

{

cout << "WAIT FOR FEW MINUTES..." << endl

<< endl;

Sleep(1000);

cout << "Your booking is just finalizing !!! " << endl

<< endl;

Sleep(1000);

cout << "Congratulations your flight schedule has been booked" << endl;

cout << "Total members = " << numberF + 1 << endl;

cout << "Ticket Number : ThaiAir-1370"

<< " " << nameM << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket Number : ThaiAir-137" << idx + 1 << " " << nameF[idx] << endl;

}

}

}

}

void bookschedule2()

{

cout << endl;

cout << "Enter which of the above flight schedule you want to select : ";

while (!(cin >> flightnumber))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count2; idx++)

{

if (flightnumber == idx + 1)

{

cout << "WAIT FOR FEW MINUTES... " << endl

<< endl;

Sleep(1000);

cout << "Your booking is just finalizing !!! " << endl

<< endl;

Sleep(1000);

cout << "Congratulations your flight schedule has been booked" << endl;

cout << "Total members = " << numberF + 1 << endl;

cout << "Ticket Number : Qatar-AYX18960"

<< " " << nameM << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket Number : Qatar-AYX1896" << idx + 1 << " " << nameF[idx] << endl;

}

}

}

}

void bookschedule3()

{

cout << endl;

cout << "Enter which of the above flight schedule you want to select : ";

while (!(cin >> flightnumber))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count3; idx++)

{

if (flightnumber == idx + 1)

{

cout << "WAIT FOR FEW MINUTES..." << endl

<< endl;

Sleep(1000);

cout << "Your booking is just finalizing !!! " << endl

<< endl;

Sleep(1000);

cout << "Congratulations your flight schedule has been booked" << endl;

cout << "Total members = " << numberF + 1 << endl;

cout << "Ticket Number : PIA-125690"

<< " " << nameM << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket Number : PIA-12569" << idx + 1 << " " << nameF[idx] << endl;

}

}

}

}

void bookschedule4()

{

cout << endl;

cout << "Enter which of the above flight schedule you want to select : ";

while (!(cin >> flightnumber))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count4; idx++)

{

if (flightnumber == idx + 1)

{

cout << "WAIT FOR FEW MINUTES..." << endl

<< endl;

Sleep(1000);

cout << "Your booking is just finalizing !!! " << endl

<< endl;

Sleep(1000);

cout << "Congratulations your flight schedule has been booked" << endl;

cout << "Total members = " << numberF + 1 << endl;

cout << "Ticket Number : Emirates-36850"

<< " " << nameM << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket Number : Emirates-3685" << idx + 1 << " " << nameF[idx] << endl;

}

}

}

}

void bookschedule5()

{

cout << endl;

cout << "Enter which of the above flight schedule you want to select : ";

while (!(cin >> flightnumber))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count5; idx++)

{

if (flightnumber == idx + 1)

{

cout << "WAIT FOR FEW MINUTES..." << endl

<< endl;

Sleep(1000);

cout << "Your booking is just finalizing !!! " << endl

<< endl;

Sleep(1000);

cout << "Congratulations your flight schedule has been booked" << endl;

cout << "Total members = " << numberF + 1 << endl;

cout << "Ticket Number : SouthWest-25970"

<< " " << nameM << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket Number : SouthWest-2597" << idx + 1 << " " << nameF[idx] << endl;

}

}

}

}

void busEco()

{

cout << "Hi " << nameM << endl

<< endl;

if (optionairline == 1)

{

char class1;

cout << "Your chosen airline is THAI AIRWAYS" << endl;

cout << endl;

cout << "Select which class you want to prefer : " << endl;

cout << endl;

cout << "Press B if you want to select Business Class!!" << endl;

cout << "Press E if you want to select Economy Class!!" << endl;

cin >> class1;

while (!(class1 == 66 || class1 == 69))

{

cout << "Try Again" << endl;

cin >> class1;

}

if (class1 == 'B')

{

cout << "Price of single Business class ticket is : PKR.200000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000); finalprice = ((numberF + 1) \* 200000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

if (class1 == 'E')

{

cout << "Price of single Economy class ticket is : PKR.100000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000);

finalprice = ((numberF + 1) \* 100000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

}

if (optionairline == 2)

{

char class2;

cout << "Your chosen airline is QATAR AIRLINES" << endl;

cout << endl;

cout << "Select which class you want to prefer : " << endl;

cout << endl;

cout << "Press B if you want to select Business Class!!" << endl;

cout << "Press E if you want to select Economy Class!!" << endl;

cin >> class2;

while (!(class2 == 66 || class2 == 69))

{

cout << "Try Again" << endl;

cin >> class2;

}

if (class2 == 'B')

{

cout << "Price of single Business class ticket is : PKR.200000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000);

finalprice = ((numberF + 1) \* 200000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

if (class2 == 'E')

{

cout << "Price of single Economy class ticket is : PKR.150000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000);

finalprice = ((numberF + 1) \* 150000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

}

if (optionairline == 3)

{

char class3;

cout << "Your chosen airline is PIA" << endl;

cout << endl;

cout << "Select which class you want to prefer : " << endl;

cout << endl;

cout << "Press B if you want to select Business Class!!" << endl;

cout << "Press E if you want to select Economy Class!!" << endl;

cin >> class3;

while (!(class3 == 66 || class3 == 69))

{

cout << "Try Again" << endl;

cin >> class3;

}

if (class3 == 'B')

{

cout << "Price of single Business class ticket is : PKR.60000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000);

finalprice = ((numberF + 1) \* 60000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

if (class3 == 'E')

{

cout << "Price of single Economy class ticket is : PKR.30000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000);

finalprice = ((numberF + 1) \* 30000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

}

if (optionairline == 4)

{

char class4;

cout << "Your chosen Airline is EMIRATES!!" << endl;

cout << endl;

cout << "Select which class you want to prefer : " << endl;

cout << endl;

cout << "Press B if you want to select Business Class!!" << endl;

cout << "Press E if you want to select Economy Class!!" << endl;

cin >> class4;

while (!(class4 == 66 || class4 == 69))

{

cout << "Try Again" << endl;

cin >> class4;

}

if (class4 == 'B')

{

cout << "Price of single Business class ticket is : PKR.400000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000);

finalprice = ((numberF + 1) \* 400000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

if (class4 == 'E')

{

cout << "Price of single Economy class ticket is : PKR.200000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000);

finalprice = ((numberF + 1) \* 200000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

}

if (optionairline == 5)

{

char class5;

cout << "Your chosen Airline is SouthWest Airlines" << endl;

cout << endl;

cout << "Select which class you want to prefer : " << endl;

cout << endl;

cout << "Press B if you want to select Business Class!!" << endl;

cout << "Press E if you want to select Economy Class!!" << endl;

cin >> class5;

while (!(class5 == 66 || class5 == 69))

{

cout << "Try Again" << endl;

cin >> class5;

}

if (class5 == 'B')

{

cout << "Price of single Business class ticket is : PKR.250000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000);

finalprice = ((numberF + 1) \* 250000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

if (class5 == 'E')

{

cout << "Price of single Economy class ticket is : PKR.125000" << endl;

cout << "Total members : " << numberF + 1 << endl;

cout << "Your total expense will be... " << endl;

Sleep(1000);

cout << "PROCESSING..." << endl;

Sleep(1000);

finalprice = ((numberF + 1) \* 125000);

cout << "YOUR TOTAL OUTLAY IS : " << finalprice;

}

}

}

void mealmenu1()

{ cout << "Welcome again...." << nameM << endl;

cout << "Total members are : " << numberF + 1 << endl

<< endl;

if (optionairline == 1)

{

cout << " THAI AIRLINES FLIGHT MEAL MENU!!! " << endl;

cout << "1. Hot Meal" << endl;

cout << "2. Muslim Meal" << endl;

cout << "3. Indian Meal" << endl;

cout << "4. Veg Meal" << endl;

cout << "5. Non-Veg Meal" << endl;

cout << "6. Child Meal" << endl;

cout << "7.Infant Meal" << endl;

cout << "Select the meal for ticket number ThaiAir-1370 ";

cin >> menuoption1;

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{ cout << "Select the meal for ticket number ThaiAir-137" << idx + 1 << " ";

cin >> menuoption[idx];

cout << endl;

}

Sleep(1000);

cout << "PROCESSING.." << endl;

Sleep(1000);

cout << "WAIT FOR FEW MINS..." << endl;

Sleep(1000);

cout << "YOUR ORDER OF MEAL IS FINALIZED!!!" << endl;

cout << "Ticket number ThaiAir-1370"

<< " " << nameM << " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL ";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket number ThaiAir-137" << idx + 1 << " " << nameF[idx] << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL ";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

}

}

}

void mealmenu2()

{

cout << "Welcome again...." << nameM << endl;

cout << "Total members are : " << numberF + 1 << endl

<< endl;

if (optionairline == 2)

{

cout << " QATAR AIRWAYS FLIGHT MEAL MENU!!! " << endl;

cout << "1. Hot Meal" << endl;

cout << "2. Muslim Meal" << endl;

cout << "3. Indian Meal" << endl;

cout << "4. Veg Meal" << endl;

cout << "5. Non-Veg Meal" << endl;

cout << "6. Child Meal" << endl;

cout << "7.Infant Meal" << endl;

cout << "Select the meal for ticket number Qatar-AYX18960 ";

cin >> menuoption1;

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Select the meal for ticket number Qatar-AYX1896" << idx + 1 << " ";

cin >> menuoption[idx];

cout << endl;

}

Sleep(1000);

cout << "PROCESSING.." << endl;

Sleep(1000);

cout << "WAIT FOR FEW MINS..." << endl;

Sleep(1000);

cout << "YOUR ORDER OF MEAL IS FINALIZED!!!" << endl;

cout << "Ticket number Qatar-AYX18960"

<< " " << nameM << " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket number Qatar-AYX1896" << idx + 1 << " " << nameF[idx] << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

}

}

}

void mealmenu3()

{

cout << "Welcome again...." << nameM << endl;

cout << "Total members are : " << numberF + 1 << endl

<< endl;

if (optionairline == 3)

{

cout << " PIA FLIGHT MEAL MENU!!! " << endl;

cout << "1. Hot Meal" << endl;

cout << "2. Muslim Meal" << endl;

cout << "3. Indian Meal" << endl;

cout << "4. Veg Meal" << endl;

cout << "5. Non-Veg Meal" << endl;

cout << "6. Child Meal" << endl;

cout << "7.Infant Meal" << endl;

cout << "Select the meal for ticket number PIA-125690 ";

cin >> menuoption1;

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Select the meal for ticket number PIA-12569" << idx + 1 << " ";

cin >> menuoption[idx];

cout << endl;

}

Sleep(1000);

cout << "PROCESSING.." << endl;

Sleep(1000);

cout << "WAIT FOR FEW MINS..." << endl;

Sleep(1000);

cout << "YOUR ORDER OF MEAL IS FINALIZED!!!" << endl;

cout << "Ticket number PIA-125690"

<< " " << nameM << " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket number PIA-12569" << idx + 1 << " " << nameF[idx] << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

}

}

}

void mealmenu4()

{

cout << "Welcome again...." << nameM << endl;

cout << "Total members are : " << numberF + 1 << endl

<< endl;

if (optionairline == 4)

{

cout << " EMIRATES FLIGHT MEAL MENU!!! " << endl;

cout << "1. Hot Meal" << endl;

cout << "2. Muslim Meal" << endl;

cout << "3. Indian Meal" << endl;

cout << "4. Veg Meal" << endl;

cout << "5. Non-Veg Meal" << endl;

cout << "6. Child Meal" << endl;

cout << "7.Infant Meal" << endl;

cout << "Select the meal for ticket number Emirates-36850 ";

cin >> menuoption1;

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Select the meal for ticket number Emirates-3685" << idx + 1 << " ";

cin >> menuoption[idx];

cout << endl;

}

Sleep(1000);

cout << "PROCESSING.." << endl;

Sleep(1000);

cout << "WAIT FOR FEW MINS..." << endl;

Sleep(1000);

cout << "YOUR ORDER OF MEAL IS FINALIZED!!!" << endl;

cout << "Ticket number Emirates-36850"

<< " " << nameM << " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL ";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket number Emirates-3685" << idx + 1 << " " << nameF[idx] << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL ";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

}

}

}

void mealmenu5()

{

cout << "Welcome again...." << nameM << endl;

cout << "Total members are : " << numberF + 1 << endl

<< endl;

if (optionairline == 5)

{

cout << " SOUTHWEST AIRLINES FLIGHT MEAL MENU!!! " << endl;

cout << "1. Hot Meal" << endl;

cout << "2. Muslim Meal" << endl;

cout << "3. Indian Meal" << endl;

cout << "4. Veg Meal" << endl;

cout << "5. Non-Veg Meal" << endl;

cout << "6. Child Meal" << endl;

cout << "7.Infant Meal" << endl;

cout << "Select the meal for ticket number SouthWest-25970 ";

cin >> menuoption1;

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Select the meal for ticket number SouthWest-2597" << idx + 1 << " ";

cin >> menuoption[idx];

cout << endl;

}

Sleep(1000);

cout << "PROCESSING.." << endl;

Sleep(1000);

cout << "WAIT FOR FEW MINS..." << endl;

Sleep(1000);

cout << "YOUR ORDER OF MEAL IS FINALIZED!!!" << endl;

cout << "Ticket number SouthWest-25970"

<< " " << nameM << " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << "Ticket number SouthWest-2597" << idx + 1 << " " << nameF[idx] << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

}

}

}

void luggagemenu()

{

char luggage;

int amount;

cout << "Welcome " << nameM << " to the luggage menu!!!";

cout << endl;

cout << "Total members are: " << numberF + 1 << endl;

cout << " LUGGAGE DETAILS " << endl;

cout << "Your total expenditure of tickets is : " << endl

<< endl;

Sleep(1000);

cout << "PROCESSING..." << endl

<< endl;

Sleep(500);

cout << "PKR : " << finalprice << endl;

cout << "EXTRA DETAILS!!" << endl

<< endl;

cout << "One single person can carry 30 kg on a sinlge ticket.." << endl;

Sleep(500);

cout << "Press Y incase you have extra luggage" << endl;

cout << "Press N incase you don't want have any extra luggage " << endl;

cin >> luggage;

while (!(luggage == 89 || luggage == 78))

{

cout << "Try Again" << endl;

cin >> luggage;

}

if (luggage == 'Y')

{

cout << "Additional price on 1 kilo is = Rs.5000" << endl;

cout << "How many extra kilos of luggage do you have = ";

while (!(cin >> amount))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

cout << endl;

finalprice = ((amount \* 5000) + finalprice);

Sleep(500);

cout << "Your new final expenditure is : Rs. " << finalprice << endl;

}

else if (luggage == 'N')

{

cout << "No extra luggage...Your final price is " << finalprice << endl;

}

}

void dicsountmenu()

{

if (optionairline == 1)

{

cout << "Welcome to Thai Airlines " << endl;

cout << "Your total expense of tickets is : " << finalprice << endl;

Sleep(500);

if (finalprice >= 500000)

{

finalprice = (finalprice - ((finalprice \* 15) / 100));

cout << "Your discounted price is...";

Sleep(1000);

cout << "RS. " << finalprice;

}

else

{

cout << "SORRY!! NO DISCOUNT IS THERE ON YOUR TICKET.";

}

}

if (optionairline == 2)

{

cout << "Welcome to Qatar Airlines " << endl;

cout << "Your total expense of tickets is : " << finalprice << endl;

Sleep(500);

if (finalprice >= 300000)

{

finalprice = (finalprice - ((finalprice \* 15) / 100));

cout << "Your discounted price is...";

Sleep(1000);

cout << "RS. " << finalprice;

}

else

{

cout << "SORRY!! NO DISCOUNT IS THERE ON YOUR TICKET.";

}

}

if (optionairline == 3)

{

cout << "Welcome to Pakistan Interational Airlines " << endl;

cout << "Your total expense of tickets is : " << finalprice << endl;

Sleep(500);

if (finalprice >= 200000)

{

finalprice = (finalprice - ((finalprice \* 10) / 100));

cout << "Your discounted price is...";

Sleep(1000);

cout << "RS. " << finalprice;

}

else

{

cout << "SORRY!! NO DISCOUNT IS THERE ON YOUR TICKET.";

}

}

if (optionairline == 4)

{

cout << "Welcome to Emirates " << endl;

cout << "Your total expense of tickets is : " << finalprice << endl;

Sleep(500);

if (finalprice >= 600000)

{

finalprice = (finalprice - ((finalprice \* 15) / 100));

cout << "Your discounted price is...";

Sleep(1000);

cout << "RS. " << finalprice;

}

else

{

cout << "SORRY!! NO DISCOUNT IS THERE ON YOUR TICKET.";

}

}

if (optionairline == 5)

{

cout << "Welcome to SouthWest Airlines " << endl;

cout << "Your total expense of tickets is : " << finalprice << endl;

Sleep(500);

if (finalprice >= 400000)

{

finalprice = (finalprice - ((finalprice \* 20) / 100));

cout << "Your discounted price is...";

Sleep(1000);

cout << "RS. " << finalprice;

}

else

{

cout << "SORRY!! NO DISCOUNT IS THERE ON YOUR TICKET.";

}

}

}

void bookingticket()

{

cout << " BOOK YOUR TICKETS!!! " << endl;

cout << "Are you sure to book the tickets (yes/no) ? " << endl;

cin >> booking;

if (booking == "yes" || booking == "YES" || booking == "Yes")

{

cout << "Congratulations! Your " << numberF + 1 << " tickets has been booked." << endl

<< endl;

cout << " THANKS FOR USING OUR SERVICE!! ";

}

else if (booking == "NO" || booking == "no" || booking == "No")

{

}

else

{

}

}

void boardingpass1()

{

cout << " THAI AIRLINES!!! " << endl;

cout << "Welcome " << nameM << "!!!" << endl;

cout << date << " ";

for (int idx = 0; idx < flightnumber; idx++)

{

if (flightnumber == idx + 1)

{

cout << " " << currentlocation1[idx] + " " + "to" + " " + desiredlocation1[idx] << endl;

}

}

cout << " BOARDING PASS 1 " << endl;

cout << nameM << " "

<< "Ticket number ThaiAir-1370"

<< " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL ";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl

<< endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << " BOARDING PASS " << idx + 2 << endl;

cout << nameF[idx] << " "

<< "Ticket number ThaiAir-137" << idx + 1 << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl

<< endl;

}

cout << endl;

cout << "Total Expense: Rs. " << finalprice << endl

<< endl;

cout << "THANKS FOR USING OUR SERVICE :) ";

}

void boardingpass2()

{

cout << " QATAR AIRLINES!!! " << endl;

cout << "Welcome " << nameM << "!!!" << endl;

cout << date << " ";

for (int idx = 0; idx < flightnumber; idx++)

{

if (flightnumber == idx + 1)

{

cout << " " << currentlocation2[idx] + " " + "to" + " " + desiredlocation2[idx] << endl;

}

}

cout << " BOARDING PASS 1 " << endl;

cout << nameM << " "

<< "Ticket number Qatar-AYX18960 "

<< " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL ";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl

<< endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << " BOARDING PASS " << idx + 2 << endl;

cout << nameF[idx] << " "

<< "Ticket number Qatar-AYX1896" << idx + 1 << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl

<< endl;

}

cout << endl;

cout << "Total Expense: Rs. " << finalprice << endl

<< endl;

cout << "THANKS FOR USING OUR SERVICE :) ";

}

void boardingpass3()

{

cout << " PAKISTAN INTERNATIONAL AIRLINES!!! " << endl;

cout << "Welcome " << nameM << "!!!" << endl;

cout << date << " ";

for (int idx = 0; idx < flightnumber; idx++)

{

if (flightnumber == idx + 1)

{

cout << " " << currentlocation3[idx] + " " + "to" + " " + desiredlocation3[idx] << endl;

}

}

cout << " BOARDING PASS 1 " << endl;

cout << nameM << " "

<< "Ticket number PIA-125690 "

<< " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL ";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl

<< endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << " BOARDING PASS " << idx + 2 << endl;

cout << nameF[idx] << " "

<< "Ticket number PIA-12569" << idx + 1 << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl

<< endl;

}

cout << endl;

cout << "Total Expense: Rs. " << finalprice << endl

<< endl;

cout << "THANKS FOR USING OUR SERVICE :) ";

}

void boardingpass4()

{

cout << " EMIRATES AIRLINES!!! " << endl;

cout << "Welcome " << nameM << "!!!" << endl;

cout << date << " ";

for (int idx = 0; idx < flightnumber; idx++)

{

if (flightnumber == idx + 1)

{

cout << " " << currentlocation4[idx] + " " + "to" + " " + desiredlocation4[idx] << endl;

}

}

cout << " BOARDING PASS 1 " << endl;

cout << nameM << " "

<< "Ticket number Emirates-36850"

<< " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL ";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << " BOARDING PASS " << idx + 2 << endl;

cout << nameF[idx] << " "

<< "Ticket number Emirates-3685" << idx + 1 << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl

<< endl;

}

cout << endl;

cout << "Total Expense: Rs. " << finalprice << endl

<< endl;

cout << "THANKS FOR USING OUR SERVICE :) ";

}

void boardingpass5()

{

cout << " SOUTHWEST AIRLINES!!! " << endl;

cout << "Welcome " << nameM << "!!!" << endl;

cout << date << " ";

for (int idx = 0; idx < flightnumber; idx++)

{

if (flightnumber == idx + 1)

{

cout << " " << currentlocation5[idx] + " " + "to" + " " + desiredlocation5[idx] << endl;

}

}

cout << " BOARDING PASS 1 " << endl;

cout << nameM << " "

<< "Ticket number SouthWest-25970"

<< " ";

if (menuoption1 == 1)

{

cout << "HOT MEAL ";

}

if (menuoption1 == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption1 == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption1 == 4)

{

cout << "VEG MEAL ";

}

if (menuoption1 == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption1 == 6)

{

cout << "CHILD MEAL ";

}

if (menuoption1 == 7)

{

cout << "INFANT MEAL";

}

cout << endl;

for (int idx = 0; idx < numberF; idx++)

{

cout << " BOARDING PASS " << idx + 2 << endl;

cout << nameF[idx] << " "

<< "Ticket number SouthWest-2597" << idx + 1 << " ";

if (menuoption[idx] == 1)

{

cout << "HOT MEAL ";

}

if (menuoption[idx] == 2)

{

cout << "MUSLIM MEAL ";

}

if (menuoption[idx] == 3)

{

cout << "INDIAN MEAL ";

}

if (menuoption[idx] == 4)

{

cout << "VEG MEAL ";

}

if (menuoption[idx] == 5)

{

cout << "NON-VEG MEAL";

}

if (menuoption[idx] == 6)

{

cout << "CHILD MEAL";

}

if (menuoption[idx] == 7)

{

cout << "INFANT MEAL";

}

cout << endl

<< endl;

}

cout << endl;

cout << "Total Expense: Rs. " << finalprice << endl

<< endl;

cout << "THANKS FOR USING OUR SERVICE :) ";

}

string feedback()

{

string option;

cout << "Do you like our service (yes/no) : ";

cin >> option;

cout << endl;

if (option == "yes" || option == "Yes" || option == "YES")

{

cout << "Give us the reviews: ";

cin.ignore();

getline(cin, review);

}

else if (option == "no" || option == "No" || option == "NO")

{

cout << "Tell us some suggestions how we can better our service: ";

cin.ignore();

getline(cin, review);

}

return review;

}

void ticketssold()

{

fstream file1;

file1.open("Total tickets sold.txt", ios ::app);

file1 << pa << "," << pm << endl;

file1.close();

}

void loadtotalTickets()

{

string line;

fstream file1;

file1.open("Total tickets sold.txt", ios ::in);

while (getline(file1, line))

{

cout << getfield(line, 1) << " has sold tickets" << endl;

Sleep(100);

cout << "PROCESSING...." << endl;

Sleep(100);

cout << getfield(line, 2) << endl

<< endl;

}

file1.close();

}

void passengertrafficking()

{

cout << "How many reports you want to add ?";

while (!(cin >> optiontraffic))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

int idx = 0;

while (idx != optiontraffic)

{

cout << "Enter your report:";

cin.ignore();

getline(cin, report[idxtraffic]);

idxtraffic++;

idx++;

}

}

void passengertrafficking2()

{

for (int idx = 0; idx < idxtraffic; idx++)

{

cout << report[idx];

cout << endl;

}

}

void storePassengertrafficking()

{

fstream file1;

file1.open("InformationOfPassengerTrafficking.txt", ios ::out);

int idx = 0;

while (idx < idxtraffic)

{

file1 << report[idx] << endl;

idx++;

}

file1.close();

}

void loadPassengerTrafficking()

{

string line;

fstream file1;

file1.open("InformationOfPassengerTrafficking.txt", ios ::in);

while (getline(file1, line))

{

report[idxtraffic] = line;

idxtraffic++;

}

file1.close();

}

void recommendation()

{

cout << "How many recommendations you want to add ?";

while (!(cin >> recommendation1))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

int idx = 0;

while (idx != recommendation1)

{

cout << "Enter your recommendation:";

cin.ignore();

getline(cin, recommendation2[idxrecommendation]);

idxrecommendation++;

idx++;

}

}

void printrecommendation()

{

for (int idx = 0; idx < idxrecommendation; idx++)

{

cout << idx + 1 << ")" << recommendation2[idx];

cout << endl;

}

}

void storeRecommendation()

{

fstream file1;

file1.open("InformationOfRecommendations.txt", ios ::out);

int idx = 0;

while (idx < idxrecommendation)

{

file1 << recommendation2[idx] << endl;

idx++;

}

file1.close();

}

void loadRecommendation()

{

string line;

fstream file1;

file1.open("InformationOfRecommendations.txt", ios ::in);

while (getline(file1, line))

{

recommendation2[idxrecommendation] = line;

idxrecommendation++;

}

file1.close();

}

void deleteflights1()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

while (!(cin >> number))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count1; idx++)

{

if (number == idx + 1)

{

currentlocation1[idx] = "#########";

desiredlocation1[idx] = "#########";

checkallschedules1();

}

}

}

void deleteflights2()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

while (!(cin >> number))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count2; idx++)

{

if (number == idx + 1)

{

currentlocation2[idx] = "#########";

desiredlocation2[idx] = "#########";

checkallschedules2();

}

}

}

void deleteflights3()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

while (!(cin >> number))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count3; idx++)

{

if (number == idx + 1)

{

currentlocation3[idx] = "#########";

desiredlocation3[idx] = "#########";

checkallschedules3();

}

}

}

void deleteflights4()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

while (!(cin >> number))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count4; idx++)

{

if (number == idx + 1)

{

currentlocation4[idx] = "#########";

desiredlocation4[idx] = "#########";

checkallschedules4();

}

}

}

void deleteflights5()

{

int number;

cout << "Enter the index number of flight you want to change : " << endl;

while (!(cin >> number))

{

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cout << "Invalid! Enter again : ";

}

for (int idx = 0; idx < input\_count5; idx++)

{

if (number == idx + 1)

{

currentlocation5[idx] = "#########";

desiredlocation5[idx] = "#########";

checkallschedules5();

}

}

}

void mainpic()

{

cout << " . " << endl;

cout << " #@@+ \*@@@+ " << endl;

cout << " +# :@= ##. \*# " << endl;

cout << " @- =% =% #\* " << endl;

cout << " =@ @- @- .@- " << endl;

cout << " @#.. \*% +% -% " << endl;

cout << " %@%\*\*#%%@+-. @= #+ " << endl;

cout << " .=\*\*\*@# .=+\*#\*=-. +@ -@. " << endl;

cout << " =\*+: +@+. .-+\*\*\*+:. @\* %= " << endl;

cout << "-@#: .+@\*: =%\* .-+\*\*\*=:=@: =% " << endl;

cout << ":++\*\*\*\*\*\*+++===#%= -\*= -@@: :=+##+=: @: " << endl;

cout << " .=#+: -= %@+ .-. .-=\*##\*-:. \*\* " << endl;

cout << " :\*%+. : #@# .== .:=\*%@@=. " << endl;

cout << " -##- -@@. +\*: :+\*%#+: " << endl;

cout << " =#\*=. .##- =%+ .-\*%\*- " << endl;

cout << " .=##=:.:::--::. -\*= -@% .\*@%- " << endl;

cout << " .#@%#\*\*\*\*\*\*#%%%\*=: .== @@@@@- " << endl;

cout << " \*@+ :=#@+ =+\*%@%. " << endl;

cout << " #%: #@# -\*%= " << endl;

cout << " :@# +@%-. +@= " << endl;

cout << " =@= :+%#--+\*####\*++++\*\*%@%+ " << endl;

cout << " #@: :+%#= .:-----: " << endl;

cout << " .%#. :\*%#=. " << endl;

cout << " .%# :=#%+: " << endl;

cout << " %# .-\*%\*-. " << endl;

cout << " :@\*..:-=\*%\*=: " << endl;

cout << " +#%#\*+-. " << endl;

}

bool isvalid(string name)

{

bool flag = true;

for (int idx = 0; idx < usercount; idx++)

{

if (users[idx] == name)

{

flag = false;

break;

}

}

return flag;

}

void loaduser()

{

string name;

string password;

fstream file;

file.open("InformationOfPassengers.txt", ios ::in);

while (getline(file, name) && getline(file, password))

{

users[usercount] = name;

passwords[usercount] = password;

cout << endl;

usercount++;

}

file.close();

}

void viewUsers()

{

cout << "USERNAMES "

<< "\t"

<< "\t"

<< "PASSWORDS" << endl

<< endl;

for (int idx = 0; idx < usercount; idx++)

{

cout << users[idx] << "\t"

<< "\t" << passwords[idx] << endl;

}

}

void storeUserInfo(string name, string password)

{

fstream file;

file.open("InformationOfPassengers.txt", ios ::app);

file << name << endl;

file << password << endl;

file.close();

}

string getfield(string line, int field)

{

int commacount = 1;

string item = "";

for (int i = 0; i < line.length(); i++)

{

if (line[i] == ',')

{

commacount++;

}

else if (commacount == field)

{

item = item + line[i];

}

}

return item;

}

void storecustomerservice()

{

fstream file1;

file1.open("customer service.txt", ios ::app);

file1 << nameM << "," << pa << "," << review << endl;

file1.close();

}

void loadcustomerservice()

{

string line;

fstream file1;

file1.open("customer service.txt", ios ::in);

while (getline(file1, line))

{ cout << getfield(line, 1) << " ";

cout << getfield(line, 2) << " ";

cout << getfield(line, 3) << endl;

}

file1.close();

}

# Weaknesses in the Business Application

The weakness in the business application is that if there would be more admins then it would be more better approach.

# Future Directions

In the future, I’d like to add more admins and then their relative passengers will only be shown to them.

**Student Reg. No. : Student Name.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **A-Extensive Evidence** | **B-Convincing Evidence** | **C-Limited Evidence** | **D-No Evidence** |
| Documentation Formatting **Grade:** | All the documentation meets all the criteria. | Documentation is well formatted but some of the criteria is not fulfilled. | Documentation is required a lot of improvement. | Documentation is not Available |
| **Documentation Formatting Criteria:** In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given  **guidelines**. Project **Poster** is professionally design and well presented | | | | |
| Documentation Contents **Grade:** | Documentation includes all of the criteria. | Documentation meet more than 80% of the criteria given. | Documentation meet more than 50% of the criteria. | When the documentation meet less than 50% of the criteria. |
| **Documentation Contents Criteria: Title** Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow**  Diagram-**Data** Structure (Arrays)-**Function** Headers and Description -Project **Code.** - **Weakness** in the Project and **Future** Directions. - **Conclusion** and What your **Learn** from the Project and Course and What is your **Future** Planning. | | | | |
| Project  Complexity  **Grade:** | Project has at least 2 user’s types and each user  has at least 5 functionalities. | Project complexity meet 80% criteria given in extensive  evidence | Project complexity meet 50% criteria given in extensive  evidence | Project complexity meet less than 50% criteria given in  extensive evidence |
| Code Style  **Grade:** | All Code style criteria is followed | All code style criteria followed but some  improvements required | lot of improvements required in coding style. | **Did not follow** code style, |
| **Code Style Criteria:** Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added. | | | | |
| Code  Documentation Mapping **Grade:** | Code and documentation is synchronized. | Code and documentation does not synchronized at **some** places | Code and documentation does not synchronized at **many** places | Code and documentation **does not** synchronized. |
| Data Structure (Arrays)  **Grade:** | Data structure is sufficient for the project requirements | Data Structure is sufficient but require improvement to meet project requirements. | Data structure is not sufficient and need a lot of improvement | Data Structure is not properly identified and declared. |
| Modularity  **Grade:** | Meet all Modularity criteria | Meet all Modularity criteria but at some places it is missing | Do not sufficiently meet the modularity criteria. | No modularity or very minimum modularity. |
| **Modularity criteria:** Functions are defined for each major feature. Functions are independent (identify from parameter list and return types). | | | | |
| Validations  **Grade:** | Validations on all number type inputs are applied | Validations are applied but at some places it is missing. | Validations are missing at lot of places | No Validations are used |
| File Handling  **Grade:** | Separate files for separate data. Data in csv format | File handing require some improvements | File handing require a lot of improvements | Not implemented |
| Aesthetics of the User Interface **Grade:** | UI is presentable. Proper coloring, Headers and  clear screen is done | UI require some improvements | UI require a lot of improvements | Not implemented |
| Presentation and Demo  **Grade:** | Presentation and Demo was 100% working | Presentation and Demo require some improvements | Presentation and Demo require a lot of improvements | Presentation was not ok and Demo was not working |
| Student  Understanding with the Code. **Grade:** | Student has complete understanding how the code is working and knows the concept. | Student has good understand but some place he does not know the concepts | Student has a very little understand and lack the major concepts. | Student does not have any level of understanding of the code. |

|  |  |
| --- | --- |
| **Checked by:** |  |
| **Comments:** |  |

**Student Reg. No. : Student Name.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **A-Extensive Evidence** | **B-Convincing Evidence** | **C-Limited Evidence** | **D-No Evidence** |
| Documentation Formatting **Grade:** | All the documentation meets all the criteria. | Documentation is well formatted but some of the criteria is not fulfilled. | Documentation is required a lot of improvement. | Documentation is not Available |
| **Documentation Formatting Criteria:** In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given  **guidelines**. Project **Poster** is professionally design and well presented | | | | |
| Documentation Contents **Grade:** | Documentation includes all of the criteria. | Documentation meet more than 80% of the criteria given. | Documentation meet more than 50% of the criteria. | When the documentation meet less than 50% of the criteria. |
| **Documentation Contents Criteria: Title** Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow**  Diagram-**Data** Structure (Arrays)-**Function** Headers and Description -Project **Code.** - **Weakness** in the Project and **Future** Directions. - **Conclusion** and What your **Learn** from the Project and Course and What is your **Future** Planning. | | | | |
| Project  Complexity  **Grade:** | Project has at least 2 user’s types and each user  has at least 5 functionalities. | Project complexity meet 80% criteria given in extensive  evidence | Project complexity meet 50% criteria given in extensive  evidence | Project complexity meet less than 50% criteria given in  extensive evidence |
| Code Style  **Grade:** | All Code style criteria is followed | All code style criteria followed but some  improvements required | lot of improvements required in coding style. | **Did not follow** code style, |
| **Code Style Criteria:** Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added. | | | | |
| Code  Documentation Mapping **Grade:** | Code and documentation is synchronized. | Code and documentation does not synchronized at **some** places | Code and documentation does not synchronized at **many** places | Code and documentation **does not** synchronized. |
| Data Structure (Arrays)  **Grade:** | Data structure is sufficient for the project requirements | Data Structure is sufficient but require improvement to meet project requirements. | Data structure is not sufficient and need a lot of improvement | Data Structure is not properly identified and declared. |
| Modularity  **Grade:** | Meet all Modularity criteria | Meet all Modularity criteria but at some places it is missing | Do not sufficiently meet the modularity criteria. | No modularity or very minimum modularity. |
| **Modularity criteria:** Functions are defined for each major feature. Functions are independent (identify from parameter list and return types). | | | | |
| Validations  **Grade:** | Validations on all number type inputs are applied | Validations are applied but at some places it is missing. | Validations are missing at lot of places | No Validations are used |
| File Handling  **Grade:** | Separate files for separate data. Data in csv format | File handing require some improvements | File handing require a lot of improvements | Not implemented |
| Aesthetics of the User Interface **Grade:** | UI is presentable. Proper coloring, Headers and  clear screen is done | UI require some improvements | UI require a lot of improvements | Not implemented |
| Presentation and Demo  **Grade:** | Presentation and Demo was 100% working | Presentation and Demo require some improvements | Presentation and Demo require a lot of improvements | Presentation was not ok and Demo was not working |
| Student  Understanding with the Code. **Grade:** | Student has complete understanding how the code is working and knows the concept. | Student has good understand but some place he does not know the concepts | Student has a very little understand and lack the major concepts. | Student does not have any level of understanding of the code. |

|  |  |
| --- | --- |
| **Checked by:** |  |
| **Comments:** |  |