OBJECT ORIENTED PROGRAMMING PROJECT

TEAM MEBER:

1. SUDAIS SHERAZ

2. HOONDRAJ

3. ZEESHAN ALI

ENROLLMENT NO: **01-134212-171, 01-134212-209, 01-134212-197**

CLASS: **BS(CS)-2D**

SUBMITTED TO: **Ma'am Erum Ashraf**

COURSE: **OOP (CSC-210)**

Department of Computer Sciences

BAHRIA UNIVERSITY, ISLAMABAD

Project Name

Airline Management System

Description:

This Project is of "Airline Management System." It allows users to book airline ticket using our program. We As A Team Worked On This Project. The Source Code Is Purely Wriiten By Our Skills Without Any Copy Pasting. We Covered Almost Every Concept We Have Studied So Far In Our Course.

Specially *Exception Handling, Templates, File Handling, Polymorphism, Inheritance etc.*

About Project:

First Of All, This Program Will Show Up A Menu Containing 5 Choices And Will Ask From User To Choose One From Them. Choices Include:

1. Price List
2. Booking
3. About
4. Review
5. Exit.

Comments Has Been Used In Almost After Every Line To enhance the readability of the user And To Show That The Program Written By Us Is Logically Ordered.

Member's Role:

Idea And Template By: *Hoondraj*

Source Code: *Zeeshan*

File Handling, Exceptional Handling and designing: *Sudais*

Source Code:

```
#include<string>
#include<fstream>
using namespace std; // Scope Identifier

ofstream p_det("Passenger Details.txt");  //Creating a text file to store Passenger's Personal Information
ofstream passngr("Booking Details.txt");  //Creating a text file to store Booking details

class Date // Class To Store Date
{
public:
   int day, month, year; // Private Data Members
   void Inp() // Function To Take Values Of Day , Month And Year
   {
      cout << "Enter Day: ";
      cin >> day;
      cout << "Enter Month: ";
      cin >> month;
      cout << "Enter Year: ";
      cin >> year;
   }
   void PrintDate() // Function To Display Date
   {
      if (day < 10 && month < 10) // In Case Day And Month Are Single Digit
         cout << "0" << day << "/0" << month << "/" << year;
      if (day < 10 && month >= 10)  // In Case Day Is Single Digit And Month Is Double Digit
         cout << "0" << day << "/" << month << "/" << year;
      if (day >= 10 && month < 10)  // In Case Day Is Double Digit And Month Is Single Digit
         cout << day << "/0" << month << "/" << year;
      if (day >= 10 && month >= 10) // In Case Both Day And Month Are Double Digit
         cout << day << "/" << month << "/" << year;
   }
```

```
}; // Class Ends

class Time // Class To Store Time
{

   // Private Data Members
public:
   int hours, minutes;
   void setHour(int a) // Set Function For Hour
   {
      hours = a;
   }
   void setMinute(int a) // Set Function For Minute
   {
      minutes = a;
   }
   void Inp() // Function For Taking Values Of Hour And Minute
   {
      cout << "Enter Hour (24 Hour Format): ";
      cin >> hours;
      cout << "Enter Minutes: ";
      cin >> minutes;
   }
   void PrintTime() // Function To Display Time
   {
      if (hours < 10 && minutes < 10) // In Case Both Hour And Minute Are Single Digit
         cout << "0" << hours << ":0" << minutes;
      if (hours < 10 && minutes >= 10) // In Case Hour Is Single Digit And Minute Is Double Digit
         cout << "0" << hours << ":" << minutes;
      if (hours >= 10 && minutes < 10) // In Case Hour Is Double Digit And Minute Is Single Digit
         cout << hours << ":0" << minutes;
      if (hours >= 10 && minutes >= 10) // In Case Both Hour And Minute Are Double Digit
         cout << hours << ":" << minutes;
   }
}; // Class Ends

class Passenger {
   string name;
   string dob;
   string nationality;
   string pasprt;
   string ph;
   string email;
   friend class Flight;
public:
   void input() {  //function to input passenger's personal information

      cout << "\nEnter Name: ";
      getline(cin, name);
      cout << "\nEnter Date of Birth: ";
      getline(cin, dob);
      cout << "\nEnter Your Nationality: ";
      getline(cin, nationality);
      cout << "\nEnter Passport No. : ";
      getline(cin, pasprt);
      cout << "\nEnter Phone No. : ";
      getline(cin, ph);
      cout << "\nEnter Email Address: ";
      getline(cin, email);

      //saving passenger's information in "Booking Details.txt" file
      p_det << "Passenger Name: " << name << endl << "DOB: " << dob << endl << "Nationality: " << nationality <<
endl
```

```
        << "Passport No: " << pasprt << endl << "Phone No: " << ph << endl << "Email: " << email << endl <<
"_________________";
        cout << endl;

    }
    void display() {  //function to display passenger's information

        cout << "\n\n\t Name: " << name;
        cout << "\n\t DOB: " << dob;
        cout << "\n\t Nationality: " << nationality;
        cout << "\n\t Passport No: " << pasprt;
        cout << "\n\t Phone No: " << ph;
        cout << "\n\n*********************************";
    }




};
class Flight // Base Class + Abstract Class
{
protected: // To Give Access In Derived Classes
    string Origin, Destination, A_Name;
    Date DepartureDate, ArrivalDate; // User Defined Type Data Members
    Time DepartureTime, ArrivalTime; // User Defined Type Data Members
    int NoOfTickets, Expenses;
public:
    virtual void Input() // Virtual Function Because It Is Used In Derived Classes
    {

        cin.ignore();
        cout << "\nEnter Agent Name: ";
        getline(cin, A_Name);
        cout << "Enter Origin: ";
        getline(cin, Origin); // Taking Origin In String From User
        cout << "Enter Destination: ";
        getline(cin, Destination); // Taking Destination In String From User
        cout << "Enter Departure Date: \n";
        DepartureDate.Inp(); // Calling InpDeparture Function From Date Class
        cout << "Enter Departure Time: \n";
        DepartureTime.Inp(); // Calling InpDeparture Function From Time Class
        cout << "Enter Arrival Date:\n";
        ArrivalDate.Inp(); // Calling InpArrival Function From Date Class
        cout << "Enter Arrival Time: \n";
        ArrivalTime.Inp(); // Calling InpArrival Function From Time Class
        cout << "How Many Tickets You Want? ";
        cin >> NoOfTickets; // Actually Number Of Passengers


        //File Handling
        passngr << "Agent Name: " << A_Name << endl << "Origin: " << Origin << endl << "Destination: " <<
Destination << endl << "No of Tickets Booked: " << NoOfTickets << endl;
        passngr << "Departure Date: " << DepartureDate.day << " / " << DepartureDate.month << " / " <<
DepartureDate.year;
        passngr << "\nDeparture Time: " << DepartureTime.hours << " : " << DepartureTime.minutes;
        passngr << "\nArrival Date: " << ArrivalDate.day << " / " << ArrivalDate.month << " \ " << ArrivalDate.year;
        passngr << "\nArrival Time: " << ArrivalTime.hours << " : " << ArrivalTime.minutes;

        passngr.close(); //closing file
    }

    // Pure Virtual Function Is Used Because Same Function Is Used In All Derived Classes. To Override Function.
```

```
    // Virtual Function Isn's Necassary Used In All Derived Class But Pure Virtual Must Be Used In Every Derived
Class.
    virtual void display() = 0; // Pure Virtual Function Of display
    virtual void price() = 0; // Pure Virtual Function Of price
    virtual void expenses() = 0; // Pure Virtual Function Of expenses
}; // Base Class Ends

template <typename T> // Template
void Review(T rev, int age) // Two Types Of Data
{
    cout << rev << endl;
    if (age < 18) // Condition For Error
    {
        throw age; // Sending In Case Of ERROR
    }
}

class DomesticFlight : public Flight // 1st Derived Class
{
private:
    int BaggageLimit; // Unique Data Member Of This Derived Class Only.
public:
    void Input() // Function That Is Virtual In Base Class
    {
        cout << "\nDomestic Flight: \n";
        Flight::Input(); // Scope Resolution Operator Because This Input Function Is Written With Virtual In Base Flight
Class.
        cout << "Enter Baggage Limit (in KG): \n";
        cin >> BaggageLimit; // Taking Value Of Unique Data Member Of This Derived Class Only.
        passngr << "\nBaggage Limit: " << BaggageLimit;
    }
    void display() // This Function Is Pure Virtual In Base Class.
    {
        cout << "\nBOOKING DETAILS";
        cout << "\n\n*********************************";
        cout << endl << "\tBooked By Agent: \t" << A_Name;
        cout << endl << "\tOrigin: \t       " << Origin;
        cout << endl << "\tDestination:\t\t" << Destination;
        cout << endl << "\tDeparture Date: \t"; DepartureDate.PrintDate(); // Calling Function Of Class Date
        cout << endl << "\tDeparture Time: \t"; DepartureTime.PrintTime(); // Calling Function Of Class Time
        cout << endl << "\tArrival Date: \t\t"; ArrivalDate.PrintDate(); // Calling Function Of Class Date
        cout << endl << "\tArrival Time: \t\t"; ArrivalTime.PrintTime(); // Calling Function Of Class Time
        cout << endl << "\tBaggage limit: \t\t" << BaggageLimit << "kg";
        cout << endl << "\tTotal Tickets Bought:\t" << NoOfTickets;
        cout << endl << "\tCongratulations !! Ticket Is Booked :) " << endl;
    }
    void expenses() // This Function Is Pure Virtual In Base Class.
    {
        if (NoOfTickets > 0 && NoOfTickets < 3) // Condition According To Price List Of Domestic Flight
        {
            Expenses = 1000 * NoOfTickets;
            if (BaggageLimit > 20) // Condition According To Price List Of Domestic Flight
            {
                Expenses += (Expenses / 10);
                cout << "Your Total Expenses Is = " << Expenses;
            }
            else
                cout << "Your Total Expenses Is = " << Expenses;
        }
        else if (NoOfTickets >= 3 && NoOfTickets < 5) // Condition According To Price List Of Domestic Flight
        {
            Expenses = 850 * NoOfTickets;
            if (BaggageLimit > 20) // Condition According To Price List Of Domestic Flight
            {
```

```
            Expenses += (Expenses / 10);
            cout << "Your Total Expenses Is = " << Expenses;
        }
        else
            cout << "Your Total Expenses Is = " << Expenses;
    }
    else if (NoOfTickets >= 5) // Condition According To Price List Of Domestic Flight
    {
        Expenses = 700 * NoOfTickets;
        if (BaggageLimit > 20) // Condition According To Price List Of Domestic Flight
        {
            Expenses += (Expenses / 10);
            cout << "Your Total Expenses Is = " << Expenses;
        }
        else
            cout << "Your Total Expenses Is = " << Expenses;
    }
}
void price() // This Function Is Pure Virtual In Base Class.
{
    cout << "\t\t PRICE LIST FOR DOMESTIC FLIGHT\n" << endl;
    cout << "\t Normal Price : Per Person 1000 Rs" << endl;
    cout << "\t Special Price ( If More Than 3 Tickets ) : Per Person 850 Rs" << endl;
    cout << "\t Family Pack ( Atleast 5 Persons ) : Per Person 700 Rs" << endl;
    cout << "\t Extra 10% Of Total Price If Baggage Limit Is More Than 20 Kg" << endl;
}
}; // 1st Derived Class Ends

class InternationalFlight : public Flight // 2nd Derived Class
{
private:
    bool Direct; // Unique Data Member Of This Derived Class Only.
public:
    void Input() // Function That Is Virtual In Base Class
    {
        cout << "\nInternational Flight: \n";
        Flight::Input(); // Scope Resolution Operator Because This Input Function Is Written With Virtual In Base Flight
Class.
        cout << "Is Flight Direct (1 For Yes & 0 For No): \n";
        cin >> Direct; // Taking Value Of Unique Data Member Of This Derived Class Only.
    }
    void display() // This Function Is Pure Virtual In Base Class.
    {
        cout << "\nBOOKING DETAILS";
        cout << "\n\n*********************************";
        cout << endl << "\tBooked By Agent: \t" << A_Name;
        cout << endl << "\tOrigin: \t        " << Origin;
        cout << endl << "\tDestination:\t\t" << Destination;
        cout << endl << "\tDeparture Date: \t"; DepartureDate.PrintDate(); // Calling Function Of Class Date
        cout << endl << "\tDeparture Time: \t"; DepartureTime.PrintTime(); // Calling Function Of Class Time
        cout << endl << "\tArrival Date: \t\t"; ArrivalDate.PrintDate(); // Calling Function Of Class Date
        cout << endl << "\tArrival Time: \t\t"; ArrivalTime.PrintTime(); // Calling Function Of Class Time
        cout << endl << "\tDirect Or Connecting:   ";
        (Direct) ? cout << "Direct" : cout << "Conecting";
        cout << endl << "\tTotal Tickets Bought:\t" << NoOfTickets;
        cout << endl << "\tCongratulations !! Ticket Is Booked :) " << endl;
    }
    void expenses() // This Function Is Pure Virtual In Base Class.
    {
        if (NoOfTickets > 0 && NoOfTickets < 3) // Condition According To Price List Of International Flight
        {
            Expenses = 1700 * NoOfTickets;
            if (Direct) // Condition According To Price List Of International Flight
```

```
            {
                Expenses += (Expenses / 5);
                cout << "Your Total Expenses Is = " << Expenses;
            }
            else
                cout << "Your Total Expenses Is = " << Expenses;
        }
        else if (NoOfTickets >= 3 && NoOfTickets < 5) // Condition According To Price List Of International Flight
        {
            Expenses = 1500 * NoOfTickets;
            if (Direct) // Condition According To Price List Of International Flight
            {
                Expenses += (Expenses / 5);
                cout << "Your Total Expenses Is = " << Expenses;
            }
            else
                cout << "Your Total Expenses Is = " << Expenses;
        }
        else if (NoOfTickets >= 5) // Condition According To Price List Of International Flight
        {
            Expenses = 1350 * NoOfTickets;
            if (Direct) // Condition According To Price List Of International Flight
            {
                Expenses += (Expenses / 5);
                cout << "Your Total Expenses Is = " << Expenses;
            }
            else
                cout << "Your Total Expenses Is = " << Expenses;
        }
    }
    void price() // This Function Is Pure Virtual In Base Class.
    {
        cout << "\t\t PRICE LIST FOR INTERNATIONAL FLIGHT\n" << endl;
        cout << "\t Normal Price : Per Person 1700 Rs" << endl;
        cout << "\t Special Price ( If More Than 3 Tickets ) : Per Person 1500 Rs" << endl;
        cout << "\t Family Pack ( Atleast 5 Persons ) : Per Person 1350 Rs" << endl;
        cout << "\t Extra 5% Of Total Price If Flight Is Connecting " << endl;
    }
}; // 2nd Derived Class Ends

class OuterSpaceFlight : public Flight // 3rd Derived Class
{
private:
    int PersonLimit = 4; // According To Condition Given In Price List Of Outer Space Flight
    int NoOfPersons; // Unique Data Member Of This Derived Class Only.
public:
    void Input() // Function That Is Virtual In Base Class
    {
        cout << endl << "Outer Space Exploration Special Flight !! " << endl;
        Flight::Input(); // Scope Resolution Operator Because This Input Function Is Written With Virtual In Base Flight
Class.
    }
    void display() // This Function Is Pure Virtual In Base Class.
    {
        cout << endl << "\tOrigin: \t       " << Origin;
        cout << endl << "\tDestination:\t\t" << Destination;
        cout << endl << "\tDeparture Date: \t"; DepartureDate.PrintDate(); // Calling Function Of Class Date
        cout << endl << "\tDeparture Time: \t"; DepartureTime.PrintTime(); // Calling Function Of Class Time
        cout << endl << "\tArrival Date: \t\t"; ArrivalDate.PrintDate(); // Calling Function Of Class Date
        cout << endl << "\tArrival Time: \t\t"; ArrivalTime.PrintTime(); // Calling Function Of Class Time
        cout << endl << "\tTotal Tickets Bought:\t" << NoOfTickets;
        if (NoOfTickets <= PersonLimit)
        {
            cout << endl << "\tCongratulations !! Ticket Is Booked :) " << endl;
```

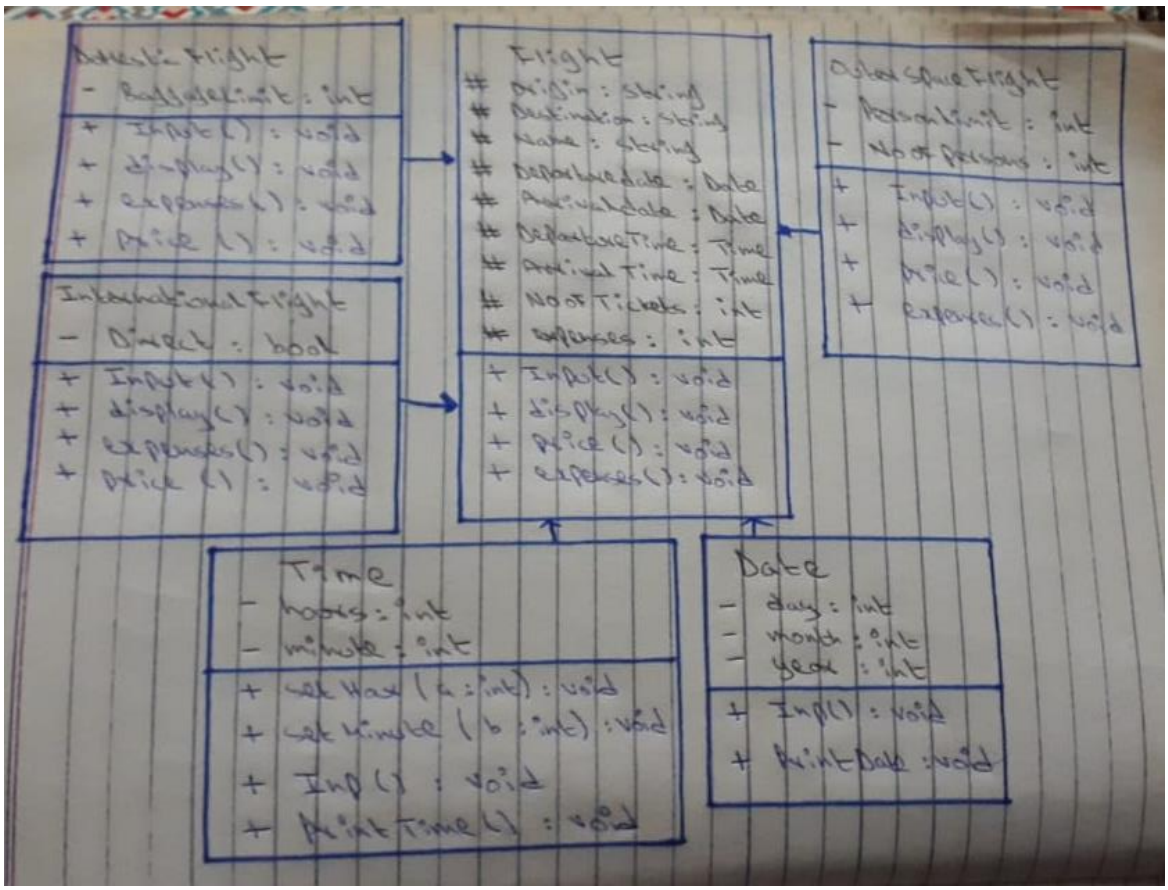
```
        }
        else
            cout << endl << "\tSorry Person Limit Is " << PersonLimit;
    }
    void expenses() // This Function Is Pure Virtual In Base Class.
    {
        if (NoOfTickets == 1) // According To Condition Given In Price List Of Outer Space Flight
        {
            Expenses = 7000;
            cout << "Your Total Expenses Is = " << Expenses;
        }
        else if (NoOfTickets == 2 || NoOfTickets == 3) // According To Condition Given In Price List Of Outer Space
Flight
        {
            Expenses = 6200 * 2;
            cout << "Your Total Expenses Is = " << Expenses;
        }
        else if (NoOfTickets == 4) // According To Condition Given In Price List Of Outer Space Flight
        {
            Expenses = 5900 * 4;
            cout << "Your Total Expenses Is = " << Expenses;
        }
        else // According To Condition Given In Price List Of Outer Space Flight
        {
            cout << "Sorry Only 4 Persons Allowed";
        }
    }
    void price() // This Function Is Pure Virtual In Base Class.
    {
        cout << "\t\t PRICE LIST FOR OUTERSPACE FLIGHT\n" << endl;
        cout << "\t Normal Price : Per Person 7000 Rs" << endl;
        cout << "\t Special Price ( If 2/3 Tickets ) : Per Person 6200 Rs" << endl;
        cout << "\t Max Tax Offer ( 4 Persons ) : Per Person 5900 Rs" << endl;
        cout << "\t Maximum Persons Allowed In Outer Space Flight Is 4 ";
        cout << endl;
    }
}; // 3rd Derived Class Ends

int main() // Main Function
{

    Passenger* p = new Passenger;
    Flight* f{}; // Making Object Of Base Class Flight
    char choice; // Character Type Data Member
    int choose; // Int Type Data Member
    char ans; // Character Type Data Mmeber
    do // Do While Loop To Run Program Again And Again until User Type ' n '
    {
        cout <<
"*****************************************************************************************************
*********" << endl;
        cout << "\n\t\t\t\t WELCOME TO MUSAAFIR AIRLINE !! " << endl;
        cout <<
"\n*****************************************************************************************************
**********" << endl;
        cout << "\n\t\t\t\t\t MAIN MENU " << endl;
        cout <<
"\n*****************************************************************************************************
**********" << endl;
        cout << "\n1) Price List" << endl;
        cout << "2) Flight Booking" << endl;
        cout << "3) About" << endl;
        cout << "4) Review" << endl;
```

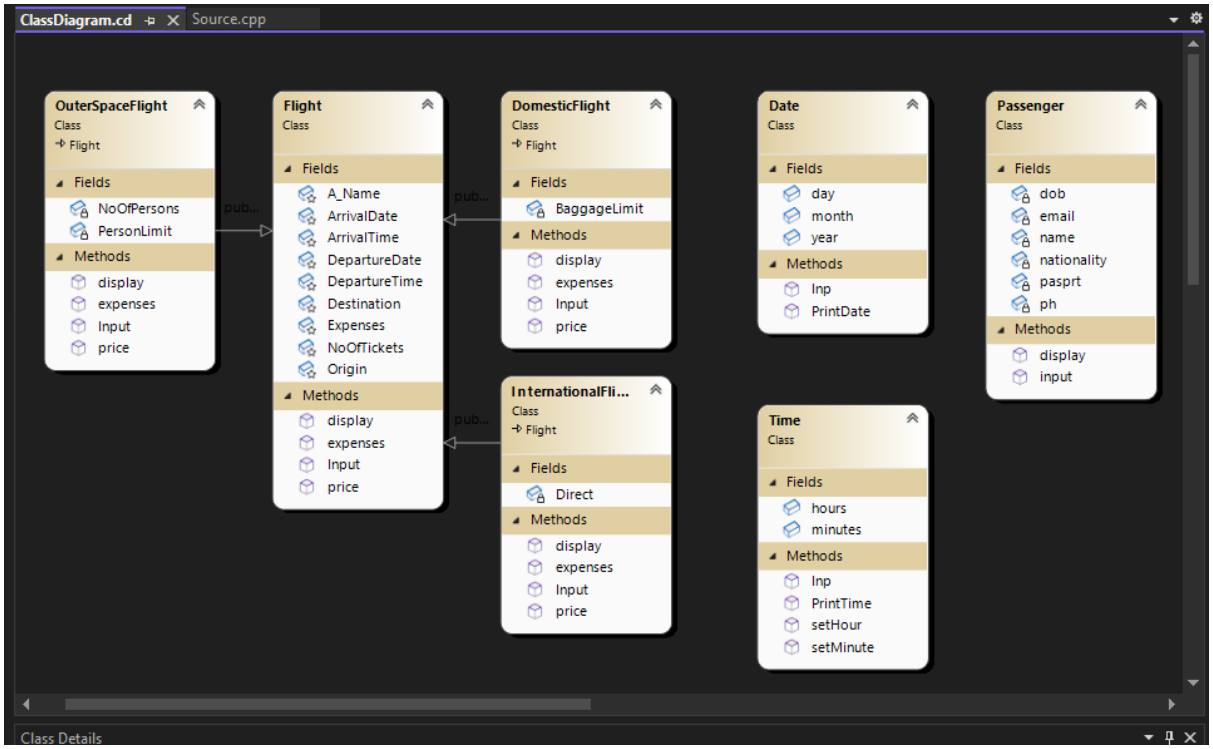
```
cout << "5) Exit" << endl;
cout << "Type Here : ";
cin >> choose; // Taking User Choice In Int

if (choose == 1)
{
   system("cls"); // To Clear Screen
   cout << "Enter D For Domestic Flight" << endl;
   cout << "Enter I For International Flight" << endl;
   cout << "Enter S For Space Exploration" << endl;
   cout << "Type Here: ";
   cin >> choice; // Taking User Choice In Alphabet

   if (choice == 'D' || choice == 'd')
   {
      system("cls"); // To Clear Screen
      f = new DomesticFlight; // Dynamic Allocating Object Of Domestic Flight
      f->price(); // When Using Dynamic Allocation Of Objects, We Use Pointers.
   }

   if (choice == 'I' || choice == 'i')
   {
      system("cls"); // To Clear Screen
      f = new InternationalFlight; // Dynamic Allocating Object Of International Flight
      f->price(); // When Using Dynamic Allocation Of Objects, We Use Pointers.
   }

   if (choice == 'S' || choice == 's')
   {
      system("cls"); // To Clear Screen
      f = new OuterSpaceFlight; // Dynamic Allocating Object Of Outer Space Flight
      f->price(); // When Using Dynamic Allocation Of Objects, We Use Pointers.
   }
}

if (choose == 2)
{
   system("cls"); // To Clear Screen
   cout << "Enter D For Domestic Flight" << endl;
   cout << "Enter I For International Flight" << endl;
   cout << "Enter S For Space Exploration" << endl;
   cout << "Type Here: ";
   cin >> choice; // Taking User Choice In Alphabet

   if (choice == 'D' || choice == 'd')
   {
      system("cls"); // To Clear Screen
      f = new DomesticFlight; // Dynamic Allocating Object Of Domestic Flight
   }
   else if (choice == 'I' || choice == 'i')
   {
      system("cls"); // To Clear Screen
      f = new InternationalFlight; // Dynamic Allocating Object Of International Flight
   }
   else if (choice == 's' || choice == 'S')
   {
      system("cls"); // To Clear Screen
      f = new OuterSpaceFlight; // Dynamic Allocating Object Of Outer Space Flight
   }
   cin.ignore(); // Used To Ignore One Or More Characters From Input
```

```
            cout << "\n****************";
            cout << "\nPassenger Information";
            cout << "\n****************";
            p->input();
            system("cls");
            f->Input(); // When Using Dynamic Allocation Of Objects, We Use Pointers.
            cout << "\nPassenger Details";
            cout << "\n******************************";

            p->display();
            f->display(); // When Using Dynamic Allocation Of Objects, We Use Pointers.
            f->expenses(); // When Using Dynamic Allocation Of Objects, We Use Pointers.
            delete f;
            cout << endl;
        }

        if (choose == 3) // About Airline Service
        {
            system("cls"); // To Clear Screen
            cout << " WELCOME TO MUSAAFIR AIRLINE " << endl;
            cout << endl << " We Are Providing Services Of Flight Since 2003.";
            cout << endl << " Glad To Tell That We Had Been Awarded Best";
            cout << endl << " Airline Servide Provider Previous Year. We";
            cout << endl << " Aim To Provide Cheap Tickets And More Comfort.";
            cout << endl;
            cout << endl << " THANKS FOR CHOOSING OUR AIRLINE , ENJOY :)" << endl;
        }

        if (choose == 4) // Showing Reviews Of All Customers
        {
            system("cls"); // To Clear Screen
            cout << "Reviews Given By Customers In Form Of Text Or Rating Out Of 5 Stars" << endl;
            cout << endl;
            cout << "Ali ( Age 19 ) Gave Rating "; Review(5, 19); // Sending Int Rating And Int Age
            cout << endl;
            cout << "Amad ( Age 24 ) Gave Rating "; Review(3.6, 24); // Sending Float Rating And Int Age
            cout << endl;
            cout << "Sehrish ( Age 23 ) Commented : "; Review("Best Airline Service I Ever Experienced.", 23); //
Sending String Review And Int Age
            cout << endl;
            cout << "Amna ( Age 19 ) Commented : "; Review("You Have To Work On Your Meal Services In
Airplane.", 19); // Sending String Review And Int Age
            cout << endl;
            cout << "Imaan ( Age 19 ) Gave Rating "; Review(4.9, 19); // Sending Float Rating And Int Age

            try // Try Block To Check Wether There Is Error Or Not
            {
                cout << endl;
                cout << "Abeera ( Age 15 ) Gave Rating "; Review(4, 15); // Sending Less Than 18 Age To Cause Erorr
            }
            catch (int a) // Default Catch Block To Recieve Any Data Type In Case Of Error
            {
                cout << "\n\n[Your Age Is : " << a << "]";
                cout << "\n[ Warning ] <<  Age Must Be 18 To Give Review  >>"; // Error Message
                cout << endl;
            }
        }

        if (choose == 5) // To Exit Program
        {
            system("cls"); // To Clear Screen
            cout << "Thank You For Visiting :)" << endl;
            exit(0); // Closing Console
```

```
            system("pause"); // It Is Used To Run Pause Program.
        }

        cout << endl << "Do You Want To Go Back To Main Menu ? " << endl;
        cout << "If Yes Press 'y' Else Press 'n' " << endl;
        cout << "Type Here : ";
        cin >> ans; // Taking User Choice In Alphabet
        system("cls"); // To Clear Screen
    } while (ans != 'n'); // Loop Runs Until User Type 'n'

    system("pause"); // It Is Used To Run Pause Program
}
```

UML Diagram :





Output :

Main Menu

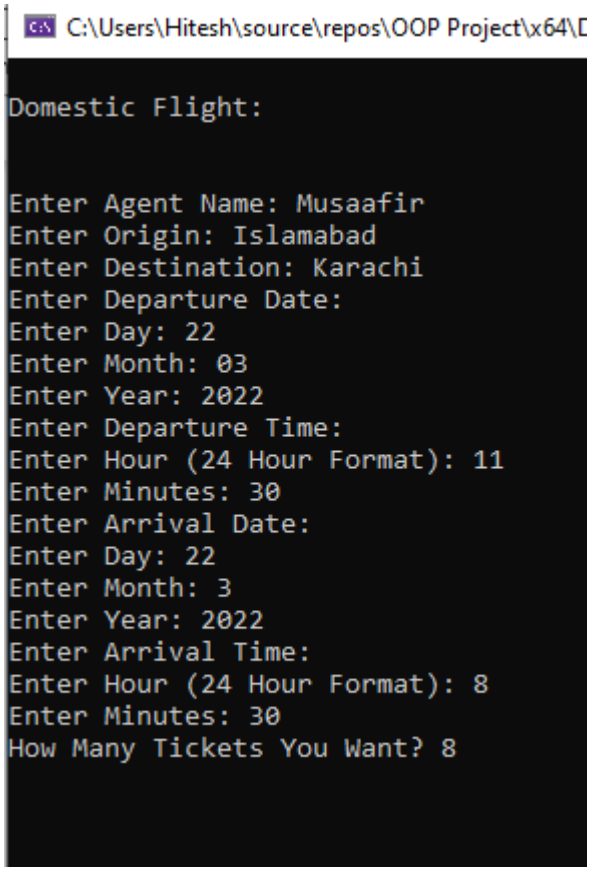
```
C:\Users\Hitesh\source\repos\OOP Project\x64\Debug\OOP Project.exe
**************************************************************************************************************

                                WELCOME TO MUSAAFIR AIRLINE !!

**************************************************************************************************************

                                        MAIN MENU

**************************************************************************************************************

1) Price List
2) Flight Booking
3) About
4) Review
5) Exit
Type Here :
```

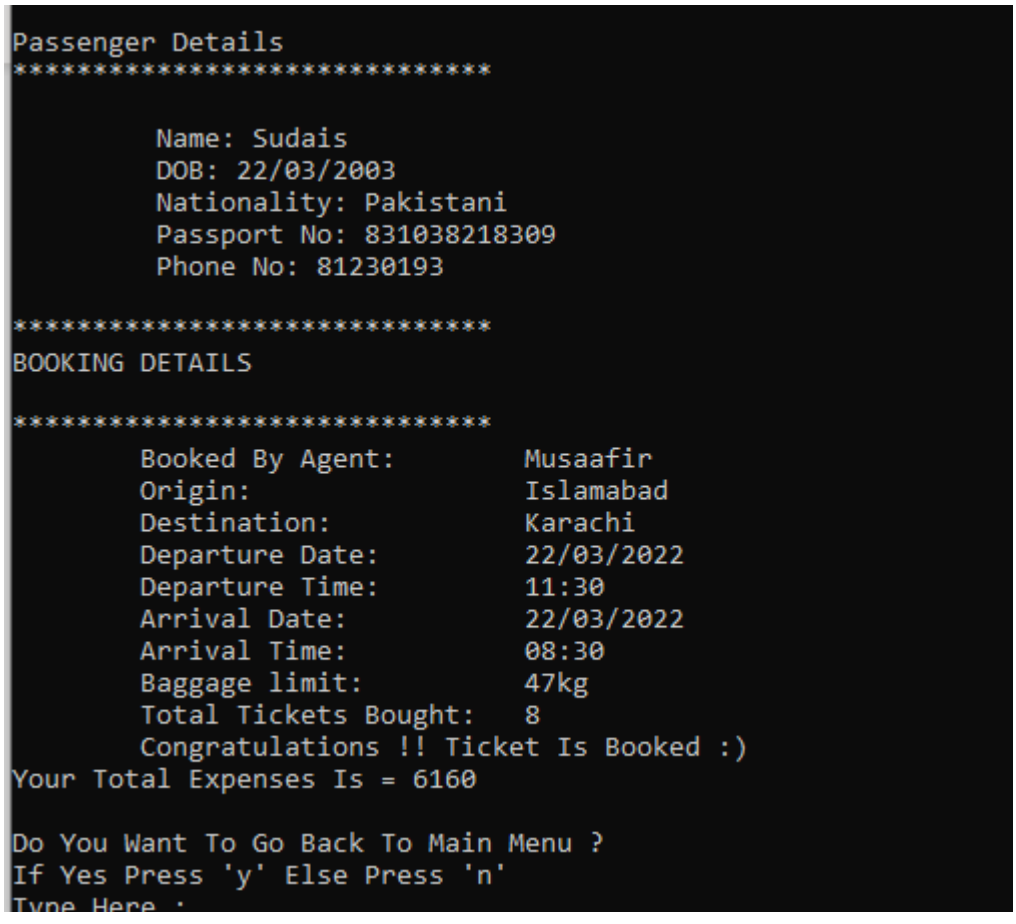
Price List

```
                PRICE LIST FOR DOMESTIC FLIGHT

        Normal Price : Per Person 1000 Rs
        Special Price ( If More Than 3 Tickets ) : Per Person 850 Rs
        Family Pack ( Atleast 5 Persons ) : Per Person 700 Rs
        Extra 10% Of Total Price If Baggage Limit Is More Than 20 Kg

Do You Want To Go Back To Main Menu ?
If Yes Press 'y' Else Press 'n'
Type Here :
```

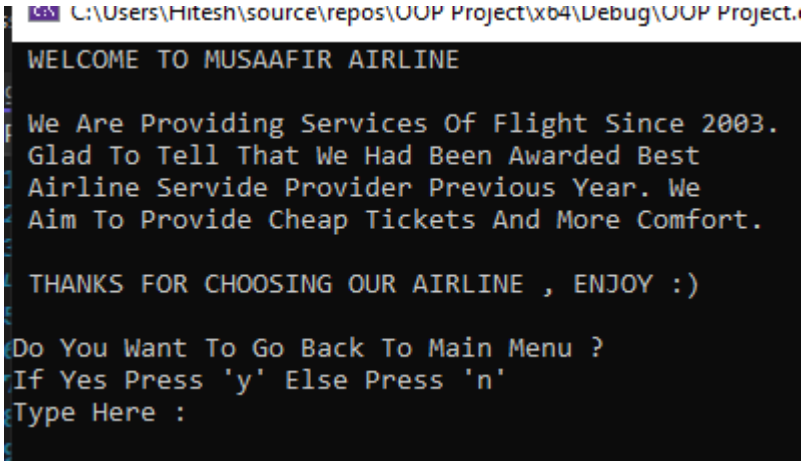
Flight Booking

```
**********
Passenger Information
**********
Enter Name: Sudais

Enter Date of Birth: 22/03/2003

Enter Your Nationality: Pakistani

Enter Passport No. : 831038218309

Enter Phone No. : 81230193

Enter Email Address: sudais@gmail.com
```


C:\Users\Hitesh\source\repos\OOP Project\x64\D
Domestic Flight:
Enter Agent Name: Musaafir
Enter Origin: Islamabad
Enter Destination: Karachi
Enter Departure Date:
Enter Day: 22
Enter Month: 03
Enter Year: 2022
Enter Departure Time:
Enter Hour (24 Hour Format): 11
Enter Minutes: 30
Enter Arrival Date:
Enter Day: 22
Enter Month: 3
Enter Year: 2022
Enter Arrival Time:
Enter Hour (24 Hour Format): 8
Enter Minutes: 30
How Many Tickets You Want? 8


Passenger Details
Name: Sudais
DOB: 22/03/2003
Nationality: Pakistani
Passport No: 831038218309
Phone No: 81230193
BOOKING DETAILS
Booked By Agent: Musaafir
Origin: Islamabad
Destination: Karachi
Departure Date: 22/03/2022
Departure Time: 11:30
Arrival Date: 22/03/2022
Arrival Time: 08:30
Baggage limit: 47kg
Total Tickets Bought: 8
Congratulations !! Ticket Is Booked :)
Your Total Expenses Is = 6160
Do You Want To Go Back To Main Menu ?
If Yes Press 'y' Else Press 'n'
Type Here :

About



Review