**Assignment: Business Application**



Session: 2021 – 2025

**Submitted by:**

Muhammad Abdullah 2021-CS-82

**Supervised by:**

DR. Awais Hassan

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

Contents

[**Airlines Reservation System** 2](#_Toc104944472)

[**Step 1: Case Study Scenario** 2](#_Toc104944473)

[**Step 2: Class Diagram** 3](#_Toc104944474)

[**BL Code:Flight.cs** 9](#_Toc104944475)

[**BL Code:MUser.cs** 10](#_Toc104944476)

[**BL Code: Passenger.cs** 11](#_Toc104944477)

# **Airlines Reservation System**

## **Step 1: Case Study Scenario**

Airlines Reservation System is basically about the ticket reservation of the passenger and the Ticket Agent manages its data. Ticket Agent also manages the flights. Passenger can enter data and then reserve the ticket if flight is available.

## **Step 2: Class Diagram**

∞

∞

1

∞

1

Contains

Books

∞

Contains

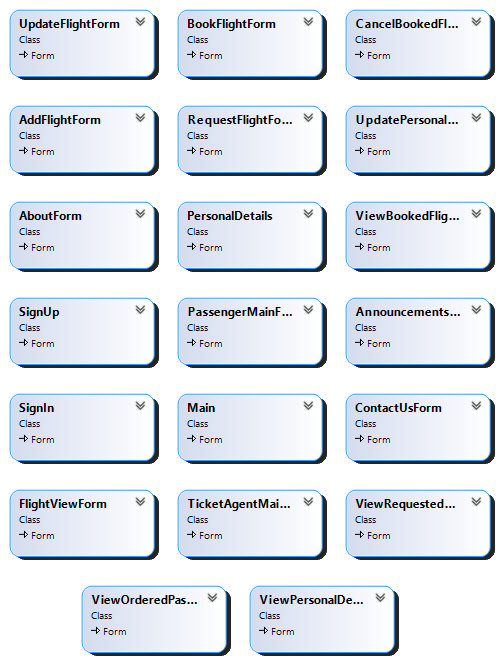
1

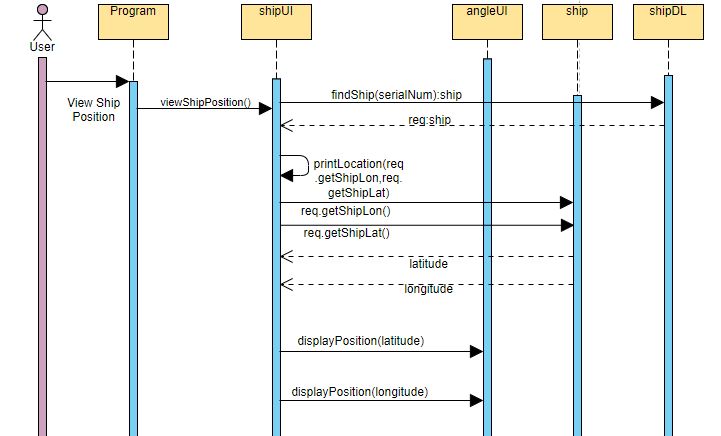
1

is

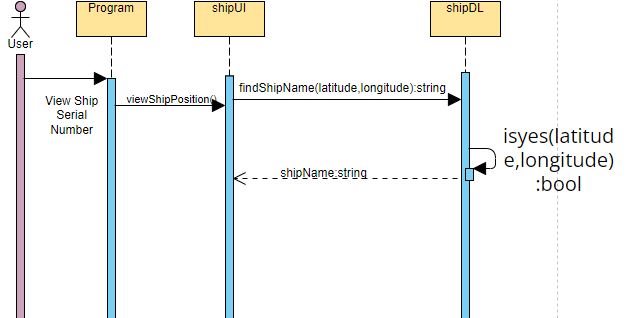
is

Contains

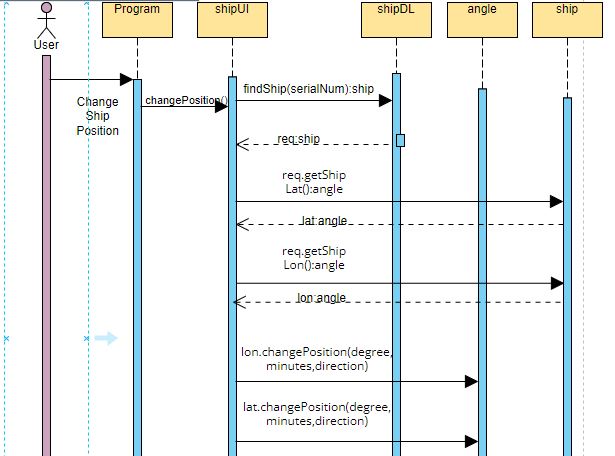


**Option 2: View Ship Position:**

**Option 3: View Ship Serial Number** :



**Option 4: Change Ship Position:**

****

## **BL Code:Flight.cs**

public class Flight

    {

        private string departCity;

        private string arrCity;

        private string tripType;

        private DateTime departDate;

        private string departTime;

        private int seats;

        private double price;

        private string flightClass;

        public Flight (string departCity , string arrCity , string tripType , DateTime departDate , string departTime , int seats , double price , string flightClass)

        {

            this.DepartCity = departCity;

            this.ArrCity = arrCity;

            this.TripType = tripType;

            this.DepartDate = departDate;

            this.DepartTime = departTime;

            this.Seats = seats;

            this.Price = price;

            this.flightClass = flightClass;

        }

        public Flight (string departCity , string arrCity , string tripType , DateTime departDate)

        {

            this.DepartCity = departCity;

            this.ArrCity = arrCity;

            this.TripType = tripType;

            this.DepartDate = departDate;

        }

        public string DepartCity { get => departCity; set => departCity = value; }

        public string ArrCity { get => arrCity; set => arrCity = value; }

        public string TripType { get => tripType; set => tripType = value; }

        public string DepartTime { get => departTime; set => departTime = value; }

        public int Seats { get => seats; set => seats = value; }

        public double Price { get => price; set => price = value; }

        public string FlightClass { get => flightClass; set => flightClass = value; }

        public DateTime DepartDate { get => departDate; set => departDate = value; }

    }

## **BL Code:MUser.cs**

public class MUser

    {

        protected string userName;

        protected string userPassword;

        protected string userRole;

        public MUser () { }

        public MUser (string userName , string userPassword)

        {

            this.UserName = userName;

            this.UserPassword = userPassword;

            this.UserRole = "NA";

        }

        public MUser (string userName , string userPassword , string userRole)

        {

            this.UserName = userName;

            this.UserPassword = userPassword;

            this.UserRole = userRole;

        }

        public string UserName { get => userName; set => userName = value; }

        public string UserPassword { get => userPassword; set { userPassword = value; } }

        public string UserRole { get => userRole; set => userRole = value; }

        public bool isAdmin ()

        {

            if (this.UserRole == "Admin" || this.UserRole == "admin")

            {

                return true;

            }

            return false;

        }

    }

## **BL Code: Passenger.cs**

public class Passenger : MUser

    {

        private string name;

        private string passportNo;

        private string cnic;

        private string eMail;

        private string gender;

        private string contactNum;

        private List<Flight> myFlights = new List<Flight>();

        private double total;

        public Passenger (string userName , string userPassword , string userRole , string name , string passportNo , string cnic , string eMail , string gender , string contactNum) : base(userName , userPassword , userRole)

        {

            this.Name = name;

            this.PassportNo = passportNo;

            this.Cnic = cnic;

            this.EMail = eMail;

            this.Gender = gender;

            this.ContactNum = contactNum;

        }

        public Passenger (string userName , string userPassword , string userRole) : base(userName , userPassword , userRole)

        {

        }

        public Passenger ()

        {

        }

        public string Name { get => name; set => name = value; }

        public string PassportNo { get => passportNo; set => passportNo = value; }

        public string Cnic { get => cnic; set => cnic = value; }

        public string EMail { get => eMail; set => eMail = value; }

        public string Gender { get => gender; set => gender = value; }

        public string ContactNum { get => contactNum; set => contactNum = value; }

        internal List<Flight> MyFlights { get => myFlights; set => myFlights = value; }

        public double Total { get => total; set => total = value; }

        public void cancelBookedFlight (Flight myFlight)

        {

            int index = MyFlights.FindIndex(item => item == myFlight);

            MyFlights.RemoveAt(index);

        }

        public double calculatePrice (Flight item , Flight myFlight , int Adult , int Child , int Infant)

        {

            double x = ((Adult \* item.Price) + (Child \* (item.Price)) / 2 + (Infant \* (item.Price) / 4));

            x += myFlight.Seats \* checkClass(myFlight);

            return x;

        }

        public double checkClass (Flight myFlight)

        {

            if (myFlight.FlightClass == "Business")

            {

                return myFlight.Price \* 10;

            }

            return 1;

        }

        public void calculateTotal ()

        {

            this.total = 0;

            foreach (var item in MyFlights)

            {

                this.total += item.Price;

            }

        }

        public void bookFlight (Flight myFlight)

        {

            MyFlights.Add(myFlight);

        }

    }

**BL Code: TicketAgent.cs**

public class TicketAgent : MUser

    {

    }

**DL Code: FlightDL.cs**

public class FlightDL

    {

        static private List<Flight> flightsList = new List<Flight>();

        static private List<Flight> requestedFlightsList = new List<Flight>();

        internal static List<Flight> FlightsList { get => flightsList; set => flightsList = value; }

        internal static List<Flight> RequestedFlightsList { get => requestedFlightsList; set => requestedFlightsList = value; }

        public static void addFlightIntoList (Flight newFlight)

        {

            FlightsList.Add(newFlight);

        }

        public static void addReqFlightIntoList (Flight reqFlight)

        {

            RequestedFlightsList.Add(reqFlight);

        }

        public static Flight checkFlight (Flight newFlight)

        {

            foreach (Flight item in FlightsList)

            {

                if (item.DepartCity == newFlight.DepartCity && item.ArrCity == newFlight.ArrCity && item.TripType == newFlight.TripType && item.DepartDate == newFlight.DepartDate)

                {

                    return item;

                }

            }

            return null;

        }

        public static void cancelFlight (Flight newFlight)

        {

            int index = FlightsList.FindIndex(item => item == newFlight);

            FlightsList.RemoveAt(index);

        }

        public static void deleteRequestedFlight (Flight newFlight)

        {

            int index = RequestedFlightsList.FindIndex(item => item == newFlight);

            RequestedFlightsList.RemoveAt(index);

        }

        public static void editFlightFromList (Flight previous , Flight updated)

        {

            foreach (Flight item in FlightsList)

            {

                if (item == previous)

                {

                    item.ArrCity = updated.ArrCity;

                    item.DepartCity = updated.DepartCity;

                    item.DepartTime = updated.DepartTime;

                    item.DepartDate = updated.DepartDate;

                    item.Seats = updated.Seats;

                    item.Price = updated.Price;

                    item.TripType = updated.TripType;

                    item.FlightClass = updated.FlightClass;

                }

            }

        }

        public static bool loadFromFile (string path , string pathF)

        {

            StreamReader f = new StreamReader(path);

            string record;

            if (File.Exists(path) && File.Exists(pathF))

            {

                while ((record = f.ReadLine()) != null)

                {

                    string[] splittedRecord = record.Split(',');

                    string departCity = splittedRecord[0];

                    string arrCity = (splittedRecord[1]);

                    string tripType = (splittedRecord[2]);

                    DateTime departDate = DateTime.Parse(splittedRecord[3]);

                    string departTime = (splittedRecord[4]);

                    int seats = int.Parse(splittedRecord[5]);

                    int price = int.Parse(splittedRecord[6]);

                    string flightClass = (splittedRecord[7]);

                    Flight newFlight = new Flight(departCity , arrCity , tripType , departDate , departTime , seats , price , flightClass);

                    addFlightIntoList(newFlight);

                }

                f.Close();

                f = new StreamReader(pathF);

                while ((record = f.ReadLine()) != null)

                {

                    string[] splittedRecord = record.Split(',');

                    string departCity = splittedRecord[0];

                    string arrCity = (splittedRecord[1]);

                    string tripType = (splittedRecord[2]);

                    DateTime departDate = DateTime.Parse(splittedRecord[3]);

                    string departTime = (splittedRecord[4]);

                    int seats = int.Parse(splittedRecord[5]);

                    int price = int.Parse(splittedRecord[6]);

                    string flightClass = (splittedRecord[7]);

                    Flight newFlight = new Flight(departCity , arrCity , tripType , departDate , departTime , seats , price , flightClass);

                    addReqFlightIntoList(newFlight);

                }

                f.Close();

                return true;

            }

            else

            {

                return false;

            }

        }

        public static void storeIntoFile (string path , Flight newFlight)

        {

            StreamWriter f = new StreamWriter(path , true);

            f.WriteLine(newFlight.DepartCity + "," + newFlight.ArrCity + "," + newFlight.TripType + "," + newFlight.DepartDate + "," + newFlight.DepartTime + "," + newFlight.Seats + "," + newFlight.Price + "," + newFlight.FlightClass);

            f.Flush();

            f.Close();

        }

        public static void storeReqFlightsIntoFile (string path , Flight newFlight)

        {

            StreamWriter f = new StreamWriter(path , true);

            f.WriteLine(newFlight.DepartCity + "," + newFlight.ArrCity + "," + newFlight.TripType + "," + newFlight.DepartDate + "," + newFlight.DepartTime + "," + newFlight.Seats + "," + newFlight.Price + "," + newFlight.FlightClass);

            f.Flush();

            f.Close();

        }

        public static void storeAllFlightsIntoFile (string path , string pathF)

        {

            StreamWriter f = new StreamWriter(path);

            foreach (var newFlight in FlightsList)

            {

                f.WriteLine(newFlight.DepartCity + "," + newFlight.ArrCity + "," + newFlight.TripType + "," + newFlight.DepartDate + "," + newFlight.DepartTime + "," + newFlight.Seats + "," + newFlight.Price + "," + newFlight.FlightClass);

            }

            f.Flush();

            f.Close();

            f = new StreamWriter(pathF);

            foreach (var newFlight in RequestedFlightsList)

            {

                f.WriteLine(newFlight.DepartCity + "," + newFlight.ArrCity + "," + newFlight.TripType + "," + newFlight.DepartDate + "," + newFlight.DepartTime + "," + newFlight.Seats + "," + newFlight.Price + "," + newFlight.FlightClass);

            }

            f.Flush();

            f.Close();

        }

    }

**DL Code: MUserDL.cs**

class MUserDL

    {

        private static List<MUser> usersList = new List<MUser>();

        internal static List<MUser> UsersList { get => usersList; set => usersList = value; }

        public static void addUserIntoList (MUser user)

        {

            UsersList.Add(user);

        }

        public static MUser isValid (MUser user)

        {

            foreach (MUser storedUser in UsersList)

            {

                if (storedUser.UserName == user.UserName &&

                storedUser.UserPassword == user.UserPassword)

                {

                    return storedUser;

                }

            }

            return null;

        }

        public static void deleteUserFromList (MUser user)

        {

            for (int i = 0 ; i < UsersList.Count ; i++)

            {

                if (usersList[i].UserName == user.UserName && usersList[i].UserPassword == user.UserPassword)

                {

                    usersList.RemoveAt(i);

                }

            }

        }

        public static void editUserFromList (MUser previous , MUser updated)

        {

            foreach (MUser user in UsersList)

            {

                if (user.UserName == previous.UserName && user.UserPassword == previous.UserPassword)

                {

                    user.UserName = updated.UserName;

                    user.UserPassword = updated.UserPassword;

                    user.UserRole = updated.UserRole;

                }

            }

        }

        public static bool loadFromFile (string path)

        {

            StreamReader f = new StreamReader(path);

            string record;

            if (File.Exists(path))

            {

                while ((record = f.ReadLine()) != null)

                {

                    string[] splittedRecord = record.Split(',');

                    string userName = splittedRecord[0];

                    string userPassword = (splittedRecord[1]);

                    string userRole = (splittedRecord[2]);

                    MUser M = new MUser(userName , userPassword , userRole);

                    addUserIntoList(M);

                }

                f.Close();

                return true;

            }

            else

            {

                return false;

            }

        }

        public static void storeIntoFile (string path , MUser M)

        {

            StreamWriter f = new StreamWriter(path , true);

            f.WriteLine(M.UserName + "," + M.UserPassword + "," + M.UserRole);

            f.Flush();

            f.Close();

        }

        public static void storeAllUsersIntoFile (string path)

        {

            StreamWriter f = new StreamWriter(path);

            foreach (var M in UsersList)

            {

                f.WriteLine(M.UserName + "," + M.UserPassword + "," + M.UserRole);

            }

            f.Flush();

            f.Close();

        }

    }

**DL Code: PassengerDL.cs**

public class PassengerDL

    {

        static private List<Passenger> passengersList = new List<Passenger>();

        static private List<Passenger> sortedPassengersList = new List<Passenger>();

        public static List<Passenger> PassengersList { get => passengersList; set => passengersList = value; }

        public static List<Passenger> SortedPassengersList { get => sortedPassengersList; set => sortedPassengersList = value; }

        public static void addPassengerIntoList (Passenger p)

        {

            PassengersList.Add(p);

        }

        public static void sortPassengersByTotal ()

        {

            foreach (Passenger p in passengersList)

            {

                p.calculateTotal();

            }

            SortedPassengersList = passengersList.OrderByDescending(p => p.Total).ToList();

        }

        public static Passenger isValidPassenger (MUser userX)

        {

            foreach (Passenger item in PassengersList)

            {

                if (userX.UserName == item.UserName && userX.UserPassword == item.UserPassword)

                {

                    return item;

                }

            }

            return null;

        }

        public static void storeIntoFile (string path , Passenger p)

        {

            StreamWriter f = new StreamWriter(path , true);

            string flights = "";

            for (int i = 0 ; i < p.MyFlights.Count ; i++)

            {

                if (i != p.MyFlights.Count - 1)

                {

                    flights += p.MyFlights[i].DepartCity + "^" + p.MyFlights[i].ArrCity + "^" + p.MyFlights[i].TripType + "^" + p.MyFlights[i].DepartDate + ";";

                }

                else

                {

                    flights += p.MyFlights[i].DepartCity + "^" + p.MyFlights[i].ArrCity + "^" + p.MyFlights[i].TripType + "^" + p.MyFlights[i].DepartDate;

                }

            }

            f.WriteLine(p.UserName + "," + p.UserPassword + "," + p.UserRole + ","

          + p.Name + "," + p.PassportNo + "," + p.Cnic + "," +

          p.EMail + "," + p.Gender + "," + p.ContactNum + "," +

           +p.Total + "," +

          flights);

            f.Flush();

            f.Close();

        }

        public static void storeAllIntoFile (string path)

        {

            StreamWriter f = new StreamWriter(path);

            foreach (var p in PassengersList)

            {

                string flights = "";

                for (int i = 0 ; i < p.MyFlights.Count ; i++)

                {

                    if (i != p.MyFlights.Count - 1)

                    {

                        flights += p.MyFlights[i].DepartCity + "^" + p.MyFlights[i].ArrCity + "^" + p.MyFlights[i].TripType + "^" + p.MyFlights[i].DepartDate + ";";

                    }

                    else

                    {

                        flights += p.MyFlights[i].DepartCity + "^" + p.MyFlights[i].ArrCity + "^" + p.MyFlights[i].TripType + "^" + p.MyFlights[i].DepartDate;

                    }

                }

                f.WriteLine(p.UserName + "," + p.UserPassword + "," + p.UserRole + ","

              + p.Name + "," + p.PassportNo + "," + p.Cnic + "," +

              p.EMail + "," + p.Gender + "," + p.ContactNum + "," +

               +p.Total + "," +

              flights);

            }

            f.Flush();

            f.Close();

        }

        public static bool loadFromFile (string path)

        {

            StreamReader f = new StreamReader(path);

            string record;

            if (File.Exists(path))

            {

                while ((record = f.ReadLine()) != null)

                {

                    string[] splittedRecord = record.Split(',');

                    Passenger p = new Passenger();

                    p.UserName = splittedRecord[0];

                    p.UserPassword = (splittedRecord[1]);

                    p.UserRole = (splittedRecord[2]);

                    p.Name = (splittedRecord[3]);

                    p.PassportNo = (splittedRecord[4]);

                    p.Cnic = (splittedRecord[5]);

                    p.EMail = (splittedRecord[6]);

                    p.Gender = (splittedRecord[7]);

                    p.ContactNum = (splittedRecord[8]);

                    p.Total = double.Parse(splittedRecord[9]);

                    string[] splittedRecordFlight = splittedRecord[10].Split(';');

                    if (splittedRecord[10] != "")

                    {

                        for (int i = 0 ; i < splittedRecordFlight.Length ; i++)

                        {

                            string[] splittedRecordx = splittedRecordFlight[i].Split('^');

                            Flight newFlight = new Flight(splittedRecordx[0] , splittedRecordx[1] , splittedRecordx[2] , DateTime.Parse(splittedRecordx[3]));

                            Flight myFlight = FlightDL.checkFlight(newFlight);

                            if (myFlight != null)

                            {

                                p.bookFlight(myFlight);

                            }

                        }

                    }

                    else if (splittedRecord[10] == "")

                    {

                        continue;

                    }

                    addPassengerIntoList(p);

                }

                f.Close();

                return true;

            }

            else

            {

                return false;

            }

        }

    }

**Flight Forms Code: AddFlightForm.cs**

public partial class AddFlightForm : Form

   {

       public AddFlightForm ()

       {

           InitializeComponent();

       }

       private void AddFlightForm\_Load (object sender , EventArgs e)

       {

       }

       private void btnSave\_Click (object sender , EventArgs e)

       {

           double price = Convert.ToDouble(Math.Round(numUpDnPrice.Value , 0));

           int seats = Convert.ToInt32(Math.Round(numUpDnSeats.Value , 0));

           Flight newFlight = new Flight(txtDept.Text , txtArr.Text , cmbxTrip.Text , dateTimePicker1.Value , mskdtxtbxTime.Text , seats , price , cmbxClass.Text);

           FlightDL.addFlightIntoList(newFlight);

           MessageBox.Show("Added Succesfully");

           resetTxt();

       }

       private void resetTxt ()

       {

           foreach (Control C in this.Controls)

           {

               if (C is TextBox || C is MaskedTextBox || C is ComboBox)

               {

                   C.Text = "";

               }

           }

       }

       private void btnBack\_Click (object sender , EventArgs e)

       {

           this.Close();

       }

   }

**Flight Forms Code: FlightViewForm.cs**

public partial class FlightViewForm : Form

    {

        private string path = "Flight.txt";

        private string pathF = "FlightReq.txt";

        public FlightViewForm ()

        {

            InitializeComponent();

        }

        private void FlightViewForm\_Load (object sender , EventArgs e)

        {

            gvFlights.DataSource = FlightDL.FlightsList;//Introspection

        }

        public void dataBind ()

        {

            gvFlights.DataSource = null;

            gvFlights.DataSource = FlightDL.FlightsList;

            gvFlights.Refresh();

        }

        private void btnNew\_Click (object sender , EventArgs e)

        {

            AddFlightForm newForm = new AddFlightForm();

            newForm.ShowDialog();

            FlightDL.storeAllFlightsIntoFile(path , pathF);

            dataBind();

        }

        private void gvFlights\_CellContentClick (object sender , DataGridViewCellEventArgs e)

        {

            Flight newFlight = (Flight)gvFlights.CurrentRow.DataBoundItem;

            if (gvFlights.Columns["Delete"].Index == e.ColumnIndex)

            {

                FlightDL.cancelFlight(newFlight);

                FlightDL.storeAllFlightsIntoFile(path , pathF);

                dataBind();

            }

            else if (gvFlights.Columns["Edit"].Index == e.ColumnIndex)

            {

                UpdateFlightForm newForm = new UpdateFlightForm(newFlight);

                newForm.ShowDialog();

                FlightDL.storeAllFlightsIntoFile(path , pathF);

                dataBind();

            }

        }

    }

**Flight Forms Code: UpdateFlightForm.cs**

public partial class UpdateFlightForm : Form

   {

       private Flight previous;

       public UpdateFlightForm (Flight previous)

       {

           InitializeComponent();

           this.previous = previous;

       }

       private void UpdateFlightForm\_Load (object sender , EventArgs e)

       {

           txtDept.Text = previous.DepartCity;

           txtArr.Text = previous.ArrCity;

           cmbxTrip.Text = previous.TripType;

           dateTimePicker1.Value = previous.DepartDate;

           mskdtxtbxTime.Text = previous.DepartTime;

           numUpDnSeats.Value = (previous.Seats);

           numUpDnPrice.Value = Convert.ToDecimal(previous.Price);

           cmbxClass.Text = previous.FlightClass;

       }

       private void btnSave\_Click (object sender , EventArgs e)

       {

           double price = Convert.ToDouble(Math.Round(numUpDnPrice.Value , 0));

           int seats = Convert.ToInt32(Math.Round(numUpDnSeats.Value , 0));

           Flight updated = new Flight(txtDept.Text , txtArr.Text , cmbxTrip.Text , dateTimePicker1.Value , mskdtxtbxTime.Text , seats , price , cmbxClass.Text);

           FlightDL.editFlightFromList(previous , updated);

           MessageBox.Show("Updated Successfully");

       }

       private void btnBack\_Click (object sender , EventArgs e)

       {

           this.Close();

       }

       private void lblMainHeading\_Click (object sender , EventArgs e)

       {

       }

   }

**MUserForms Code: SignIn.cs**

public partial class SignIn : Form

{

    public SignIn ()

    {

        InitializeComponent();

    }

    private void clearData ()

    {

        txtPassword.Text = "";

        txtUserName.Text = "";

    }

    private void SignIn\_Load (object sender , EventArgs e)

    {

        //this.SetDesktopLocation(10 , 10);

        //ChangeLanguage("es-ES");

    }

    private void maskedTextBox1\_MaskInputRejected (object sender , MaskInputRejectedEventArgs e)

    {

    }

    private void maskedTextBox2\_MaskInputRejected (object sender , MaskInputRejectedEventArgs e)

    {

    }

    private void btnNext\_Click (object sender , EventArgs e)

    {

        string userName = txtUserName.Text;

        string userPassword = txtPassword.Text;

        string path = "MUser.txt";

        MUser user = new MUser(userName , userPassword);

        MUser userX = MUserDL.isValid(user);

        if (userX != null)

        {

            //MessageBox.Show(userX.UserName);

            if (!userX.isAdmin())

            {

                //Passenger p = PassengerDL.isValidPassenger(userX);

                PassengerMainForm pass = new PassengerMainForm(PassengerDL.isValidPassenger(userX));

                pass.Show();

            }

            else if (userX.isAdmin())

            {

                userX = new TicketAgent();

                TicketAgent a;

                a = (TicketAgent)userX;

                TicketAgentMainForm adm = new TicketAgentMainForm(a);

                adm.Show();

            }

            else

            {

                MessageBox.Show("User is invalid");

            }

        }

        clearData();

    }

    private void btnBack\_Click (object sender , EventArgs e)

    {

        this.Close();

    }

    private void signUpToolStripMenuItem\_Click (object sender , EventArgs e)

    {

        this.Close();

        Form signUpForm = new SignUp();

        signUpForm.Show();

    }

    private void menuStrpMain\_ItemClicked (object sender , ToolStripItemClickedEventArgs e)

    {

    }

**MUserForms Code: SignUp.cs**

public partial class SignUp : Form

   {

       public SignUp ()

       {

           InitializeComponent();

       }

       private void clearData ()

       {

           txtPassword.Text = "";

           comboBox1.Text = "";

           txtUserName.Text = "";

       }

       private void SignUp\_Load (object sender , EventArgs e)

       {

       }

       private void btnNext\_Click (object sender , EventArgs e)

       {

           string userName = txtUserName.Text;

           string userPassword = txtPassword.Text;

           string userRole = comboBox1.Text;

           string path = "MUser.txt";

           string path1 = "Passenger.txt";

           MUser user = new MUser(userName , userPassword , userRole);

           MUserDL.addUserIntoList(user);

           MUserDL.storeIntoFile(path , user);

           if (!user.isAdmin())

           {

               Passenger p = new Passenger();

               p.UserName = user.UserName;

               p.UserRole = user.UserRole;

               p.UserPassword = user.UserPassword;

               // MessageBox.Show(p.UserName);

               PassengerDL.addPassengerIntoList(p);

               PassengerDL.storeIntoFile(path1 , p);

           }

           /\*  else

             {

                 userX = new TicketAgent();

                 TicketAgent a;

                 a = (TicketAgent)userX;

             }\*/

           MessageBox.Show("Account Created Successfully. You can now Sign In");

           clearData();

       }

       private void btnBack\_Click (object sender , EventArgs e)

       {

           this.Close();

       }

       private void signInToolStripMenuItem\_Click (object sender , EventArgs e)

       {

           this.Close();

           Form signInForm = new SignIn();

           signInForm.Show();

       }

   }

**PassengerForms Code: BookFlightForm.cs**

public partial class BookFlightForm : Form

   {

       private Passenger p;

       public BookFlightForm (Passenger p)

       {

           InitializeComponent();

           this.p = p;

       }

       private void BookFlightForm\_Load (object sender , EventArgs e)

       {

           cmbxDepartCity.DataSource = FlightDL.FlightsList.Select(c => new { c.DepartCity }).Distinct().ToList();

           cmbxArrCity.DataSource = FlightDL.FlightsList.Select(c => new { c.ArrCity }).Distinct().ToList();

       }

       private void btnSave\_Click (object sender , EventArgs e)

       {

           Flight myFlight = new Flight(cmbxDepartCity.Text , cmbxArrCity.Text , cmbxTripType.Text , dateTimePicker1.Value);

           Flight item = FlightDL.checkFlight(myFlight);

           if (item != null)

           {

               int Adult = Convert.ToInt32(Math.Round(numUpDnAdult.Value , 0));

               int Child = Convert.ToInt32(Math.Round(numUpDnChild.Value , 0));

               int Infant = Convert.ToInt32(Math.Round(numUpDnInfant.Value , 0));

               myFlight.DepartTime = item.DepartTime;

               myFlight.Seats = Adult + Child + Infant;

               myFlight.FlightClass = cmbxClass.Text;

               double x = p.calculatePrice(item , myFlight , Adult , Child , Infant);

               myFlight.Price = x;

               p.bookFlight(myFlight);

           }

           MessageBox.Show("Saved");

       }

       private void btnBack\_Click (object sender , EventArgs e)

       {

           this.Close();

       }

       private void cmbxDepartCity\_SelectedIndexChanged (object sender , EventArgs e)

       {

           cmbxArrCity.DataSource = FlightDL.FlightsList.Where(x => x.DepartCity == cmbxDepartCity.Text).Select(x => x.ArrCity).Distinct().ToList();

           cmbxTripType.DataSource = FlightDL.FlightsList.Where(x => x.DepartCity == cmbxDepartCity.Text && x.ArrCity == cmbxArrCity.Text).Select(x => x.TripType).Distinct().ToList();

       }

       private void cmbxArrCity\_SelectedIndexChanged (object sender , EventArgs e)

       {

           cmbxTripType.DataSource = FlightDL.FlightsList.Where(x => x.DepartCity == cmbxDepartCity.Text && x.ArrCity == cmbxArrCity.Text).Select(x => x.TripType).Distinct().ToList();

       }

       private void cmbxTripType\_SelectedIndexChanged (object sender , EventArgs e)

       {

           dateTimePicker1.Value = FlightDL.FlightsList.Where(x => x.DepartCity == cmbxDepartCity.Text && x.ArrCity == cmbxArrCity.Text).Select(x => x.DepartDate).First();

       }

   }

**PassengerForms Code: CancelBookedFlightForm.cs**

public partial class CancelBookedFlightForm : Form

  {

      private Passenger p;

      public CancelBookedFlightForm (Passenger p)

      {

          InitializeComponent();

          this.p = p;

      }

      private void CancelBookedFlightForm\_Load (object sender , EventArgs e)

      {

          gvFlights.DataSource = p.MyFlights;//Introspection

      }

      public void dataBind ()

      {

          gvFlights.DataSource = null;

          gvFlights.DataSource = FlightDL.FlightsList;

          gvFlights.Refresh();

      }

      private void gvFlights\_CellContentClick (object sender , DataGridViewCellEventArgs e)

      {

          Flight newFlight = (Flight)gvFlights.CurrentRow.DataBoundItem;

          if (gvFlights.Columns["Cancel"].Index == e.ColumnIndex)

          {

              p.cancelBookedFlight(newFlight);

              dataBind();

          }

      }

      private void btnBack\_Click (object sender , EventArgs e)

      {

          this.Close();

      }

  }

**PassengerForms Code: PersonalDetails.cs**

public partial class PersonalDetails : Form

   {

       private Passenger p;

       public PersonalDetails (Passenger p)

       {

           InitializeComponent();

           this.p = p;

       }

       private void btnSave\_Click (object sender , EventArgs e)

       {

           p.Name = txtName.Text;

           p.Cnic = mskdtxtbxCnic.Text;

           p.PassportNo = txtPassport.Text;

           p.ContactNum = mskdtxtbxContact.Text;

           p.Gender = cmbxGender.Text;

           p.EMail = txtEmail.Text;

           MessageBox.Show("Details Saved");

       }

       private void btnBack\_Click (object sender , EventArgs e)

       {

           this.Close();

       }

       private void PersonalDetails\_Load (object sender , EventArgs e)

       {

       }

       private void tableLayoutPanel2\_Paint (object sender , PaintEventArgs e)

       {

       }

   }

**PassengerForms Code:  RequestFlightForm.cs**

public partial class RequestFlightForm : Form

  {

      public RequestFlightForm ()

      {

          InitializeComponent();

      }

      private void RequestFlightForm\_Load (object sender , EventArgs e)

      {

      }

      private void btnRequest\_Click (object sender , EventArgs e)

      {

          Flight reqFlight = new Flight(txtDept.Text , txtArr.Text , cmbxTripType.Text , dateTimePicker1.Value);

          FlightDL.addReqFlightIntoList(reqFlight);

          FlightDL.storeReqFlightsIntoFile("FlightReq.txt" , reqFlight);

          MessageBox.Show("Request Sent");

          resetTxt();

      }

      private void resetTxt ()

      {

          foreach (Control C in this.Controls)

          {

              if (C is TextBox || C is MaskedTextBox || C is ComboBox)

              {

                  C.Text = "";

              }

          }

      }

      private void btnBack\_Click (object sender , EventArgs e)

      {

          this.Close();

      }

  }

**PassengerForms Code:  UpdatePersonalDetailsForm .cs**

public partial class UpdatePersonalDetailsForm : Form

   {

       private Passenger p;

       public UpdatePersonalDetailsForm (Passenger p)

       {

           InitializeComponent();

           this.p = p;

       }

       private void UpdatePersonalDetailsForm\_Load (object sender , EventArgs e)

       {

           txtName.Text = p.Name;

           mskdtxtbxCnic.Text = p.Cnic;

           txtPassport.Text = p.PassportNo;

           mskdtxtbxContact.Text = p.ContactNum;

           cmbxGender.Text = p.Gender;

           txtEmail.Text = p.EMail;

       }

       private void btnSave\_Click (object sender , EventArgs e)

       {

           p.Name = txtName.Text;

           p.Cnic = mskdtxtbxCnic.Text;

           p.PassportNo = txtPassport.Text;

           p.ContactNum = mskdtxtbxContact.Text;

           p.Gender = cmbxGender.Text;

           p.EMail = txtEmail.Text;

           MessageBox.Show("Details Updated");

       }

       private void btnBack\_Click (object sender , EventArgs e)

       {

           this.Close();

       }

   }

**PassengerForms Code:  ViewBookedFlightsForm .cs**

public partial class ViewBookedFlightsForm : Form

   {

       private Passenger p;

       public ViewBookedFlightsForm (Passenger p)

       {

           InitializeComponent();

           this.p = p;

       }

       private void ViewBookedFlightsForm\_Load (object sender , EventArgs e)

       {

           gvFlights.DataSource = p.MyFlights;

       }

       private void gvFlights\_CellContentClick (object sender , DataGridViewCellEventArgs e)

       {

       }

       private void btnBack\_Click (object sender , EventArgs e)

       {

           this.Close();

       }

   }

**TicketAgentForms Code:  ViewOrderedPassengersForm .cs**

public partial class ViewOrderedPassengersForm : Form

    {

        public ViewOrderedPassengersForm ()

        {

            InitializeComponent();

        }

        private void ViewOrderedPassengersForm\_Load (object sender , EventArgs e)

        {

            PassengerDL.sortPassengersByTotal();

            gvPassengers.DataSource = PassengerDL.SortedPassengersList;

            /\*   Flights.DataSource = PassengerDL.SortedPassengersList.Select(c => new

               {

                   c.MyFlights

               }).ToList();\*/

        }

        private void btnBack\_Click (object sender , EventArgs e)

        {

            this.Close();

        }

        private void gvPassengers\_CellContentClick (object sender , DataGridViewCellEventArgs e)

        {

            Passenger p = (Passenger)gvPassengers.CurrentRow.DataBoundItem;

            if (gvPassengers.Columns["Flights"].Index == e.ColumnIndex)

            {

                ViewBookedFlightsForm viewForm = new ViewBookedFlightsForm(p);

                viewForm.Show();

                //dataBind();

            }

        }

**TicketAgentForms Code:  ViewPersonalDetailsForm .cs**

public partial class ViewPersonalDetailsForm : Form

   {

       public ViewPersonalDetailsForm ()

       {

           InitializeComponent();

       }

       private void ViewPersonalDetailsForm\_Load (object sender , EventArgs e)

       {

           gvPassengers.DataSource = PassengerDL.PassengersList;

       }

       private void btnBack\_Click (object sender , EventArgs e)

       {

           this.Close();

       }

       private void gvPassengers\_CellContentClick (object sender , DataGridViewCellEventArgs e)

       {

       }

**TicketAgentForms Code:  ViewRequestedFlightsForm .cs**

public partial class ViewRequestedFlightsForm : Form

    {

        private string path = "Flight.txt";

        private string pathF = "FlightReq.txt";

        public ViewRequestedFlightsForm ()

        {

            InitializeComponent();

            gvFlights.DataSource = FlightDL.RequestedFlightsList;

        }

        private void gvFlights\_CellContentClick (object sender , DataGridViewCellEventArgs e)

        {

            Flight newFlight = (Flight)gvFlights.CurrentRow.DataBoundItem;

            if (gvFlights.Columns["Add"].Index == e.ColumnIndex)

            {

                FlightDL.addFlightIntoList(newFlight);

                UpdateFlightForm newForm = new UpdateFlightForm(newFlight);

                newForm.ShowDialog();

                FlightDL.deleteRequestedFlight(newFlight);

                FlightDL.storeAllFlightsIntoFile(path , pathF);

                dataBind();

            }

        }

        public void dataBind ()

        {

            gvFlights.DataSource = null;

            gvFlights.DataSource = FlightDL.RequestedFlightsList;

            gvFlights.Refresh();

        }

        private void ViewRequestedFlightsForm\_Load (object sender , EventArgs e)

        {

        }

        private void btnBack\_Click (object sender , EventArgs e)

        {

            this.Close();

        }

    }

**Main Forms Code:  Main .cs**

public partial class Main : Form

   {

       public Main ()

       {

           InitializeComponent();

           //this.SetDesktopLocation(10 , 10);

           string path = "MUser.txt";

           string path1 = "Flight.txt";

           string path2 = "Passenger.txt";

           string pathF = "FlightReq.txt";

           if (MUserDL.loadFromFile(path) && FlightDL.loadFromFile(path1 , pathF) && PassengerDL.loadFromFile(path2))

           {

               MessageBox.Show("Data Successfully Loaded");

           }

           else

           {

               MessageBox.Show("Data Loading Failed");

           }

       }

       private void Main\_Load (object sender , EventArgs e)

       {

       }

       private void announcementsToolStripMenuItem\_Click (object sender , EventArgs e)

       {

       }

       private void signInToolStripMenuItem\_Click (object sender , EventArgs e)

       {

           Form signInForm = new SignIn();

           signInForm.Show();

       }

       private void signUpToolStripMenuItem\_Click (object sender , EventArgs e)

       {

           Form signUpForm = new SignUp();

           signUpForm.Show();

       }

       private void aboutToolStripMenuItem\_Click (object sender , EventArgs e)

       {

           AboutForm f = new AboutForm();

           f.Show();

       }

   }

**Main Forms Code:  PassengerMainForm .cs**

public partial class PassengerMainForm : Form

 {

     private Passenger p;

     private string path = "Passenger.txt";

     public PassengerMainForm (Passenger p)

     {

         InitializeComponent();

         this.p = p;

     }

     private void PassengerMainForm\_Load (object sender , EventArgs e)

     {

     }

     private void bookToolStripMenuItem\_Click (object sender , EventArgs e)

     {

         BookFlightForm bookForm = new BookFlightForm(p);

         bookForm.Show();

     }

     private void personalToolStripMenuItem\_Click (object sender , EventArgs e)

     {

         PersonalDetails pDForm = new PersonalDetails(p);

         pDForm.Show();

     }

     private void requestToolStripMenuItem\_Click (object sender , EventArgs e)

     {

         RequestFlightForm reqFlightForm = new RequestFlightForm();

         reqFlightForm.Show();

     }

     private void myFlightsToolStripMenuItem\_Click (object sender , EventArgs e)

     {

         ViewBookedFlightsForm viewFlight = new ViewBookedFlightsForm(p);

         viewFlight.Show();

     }

     private void cancelToolStripMenuItem\_Click (object sender , EventArgs e)

     {

         CancelBookedFlightForm cancelBookedFlight = new CancelBookedFlightForm(p);

         cancelBookedFlight.Show();

     }

     private void updatePersonalDetailsToolStripMenuItem\_Click (object sender , EventArgs e)

     {

         UpdatePersonalDetailsForm updateDetails = new UpdatePersonalDetailsForm(p);

         updateDetails.Show();

     }

     private void btnLogOut\_Click (object sender , EventArgs e)

     {

         PassengerDL.storeAllIntoFile(path);

         this.Close();

     }

 }

**Main Forms Code:  TicketAgentMainForm .cs**

public partial class TicketAgentMainForm : Form

  {

      private TicketAgent a;

      public TicketAgentMainForm (TicketAgent a)

      {

          InitializeComponent();

          this.a = a;

      }

      private void TicketAgentMainForm\_Load (object sender , EventArgs e)

      {

      }

      private void flightStripMenuItem\_Click (object sender , EventArgs e)

      {

          FlightViewForm newForm = new FlightViewForm();

          newForm.Show();

      }

      private void TicketAgentMainForm\_InputLanguageChanged (object sender , InputLanguageChangedEventArgs e)

      {

      }

      private void personalToolStripMenuItem1\_Click (object sender , EventArgs e)

      {

          ViewPersonalDetailsForm persForm = new ViewPersonalDetailsForm();

          persForm.Show();

      }

      private void orderedPassengersToolStripMenuItem\_Click (object sender , EventArgs e)

      {

          ViewOrderedPassengersForm ordForm = new ViewOrderedPassengersForm();

          ordForm.Show();

      }

      private void passengersToolStripMenuItem\_Click (object sender , EventArgs e)

      {

      }

      private void requestedFlightsToolStripMenuItem\_Click (object sender , EventArgs e)

      {

          ViewRequestedFlightsForm reqForm = new ViewRequestedFlightsForm();

          reqForm.Show();

      }

      private void logOutToolStripMenuItem\_Click (object sender , EventArgs e)

      {

          this.Close();

      }

  }