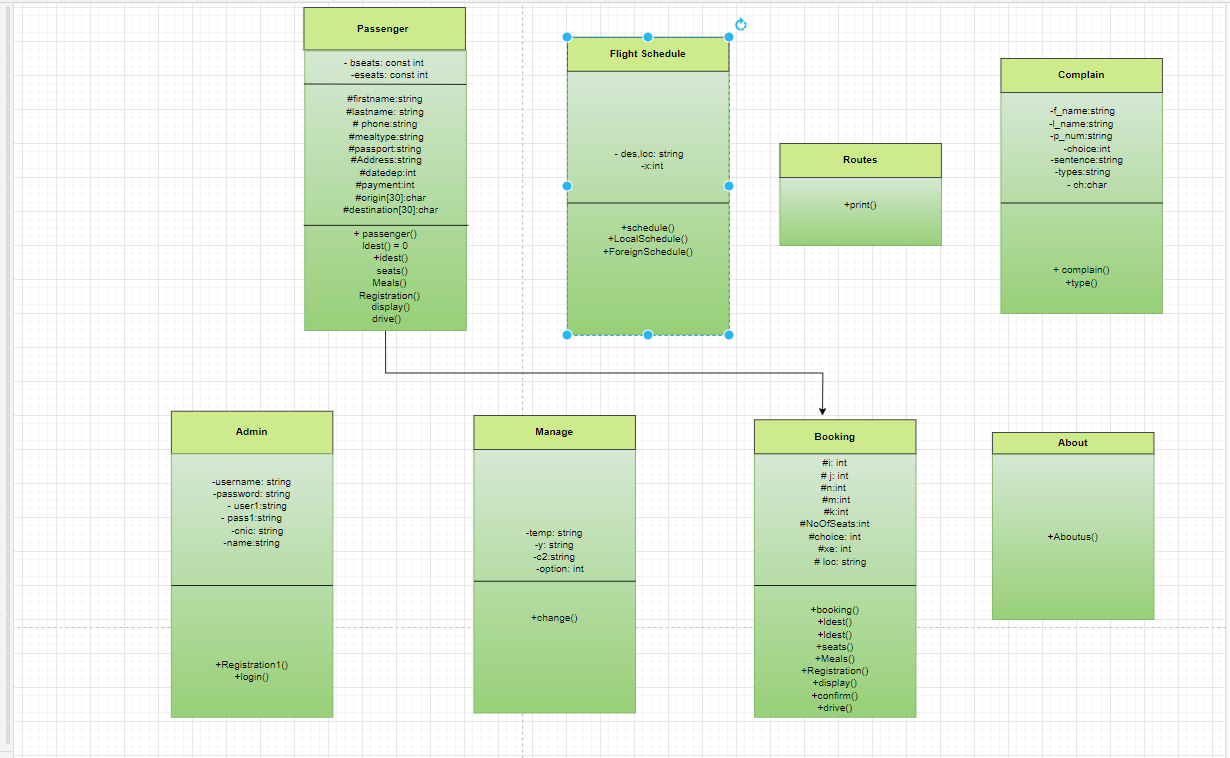
# Project - NUCES Airline Flight System

# **Name: Fawad Humayun**

# **Roll No: 22i-2548**

# **Submitted To: Bilal Khalid**

****

**Report for Flight Management System**

This C++ program that allows passengers to book and cancel seats for flights. The program defines several classes:

1. Flight Schedule
2. Passenger
3. Booking
4. Manage
5. Admin
6. About
7. Complain
8. Routes
9. Main Menu

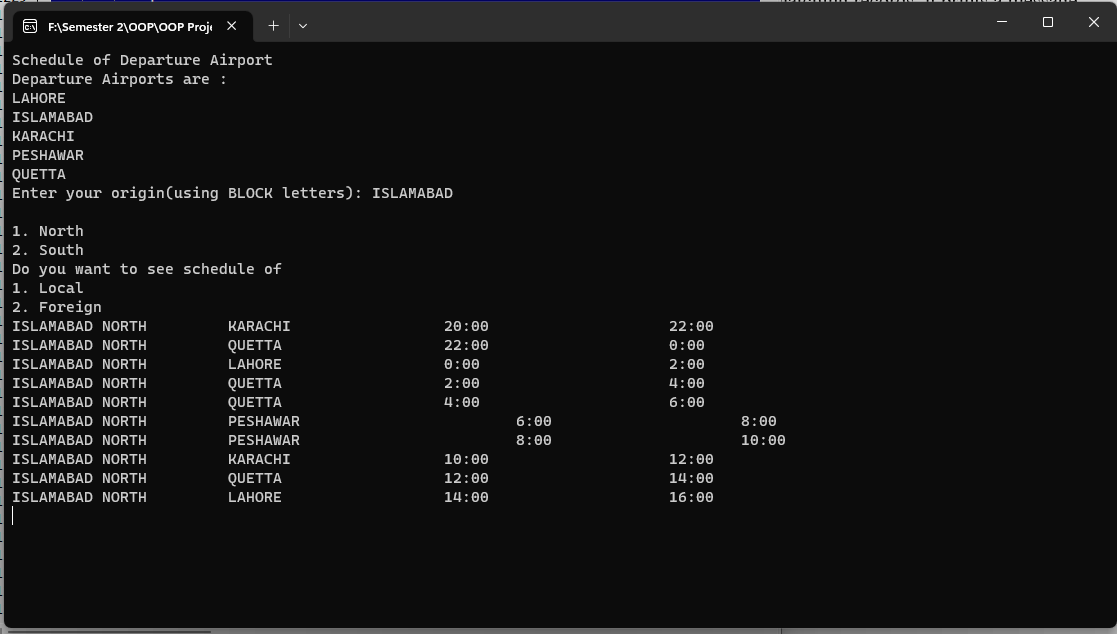
**Flight Schedule:**

This class, named FlightSchedule, represents a flight schedule for an airline. The class has two private data members, 'des' and 'loc', both of type string, and an integer 'x'. The public section of the class has two member functions: schedule () and LocalSchedule (), and ForeignSchedule ().

The **schedule ()** function prompts the user to enter the origin airport (in capital letters) and the direction of the destination (north or south). The user is then asked to choose between local and foreign schedules. If the user selects the local schedule, the function calls the **LocalSchedule ()** function. If the user selects the foreign schedule, the function calls the **ForeignSchedule ()** function.

The **LocalSchedule ()** function generates a schedule of local flights departing from the origin airport. It generates the schedule for ten flights, selecting destinations randomly from a list of five options. If the randomly selected destination is the same as the origin, it is skipped. The function prints the departure time and arrival time for each flight.

The **ForeignSchedule ()** function generates a schedule of international flights departing from the origin airport. It generates the schedule for five flights, selecting destinations randomly from a list of 25 options. The function prints the departure time and arrival time for each flight.



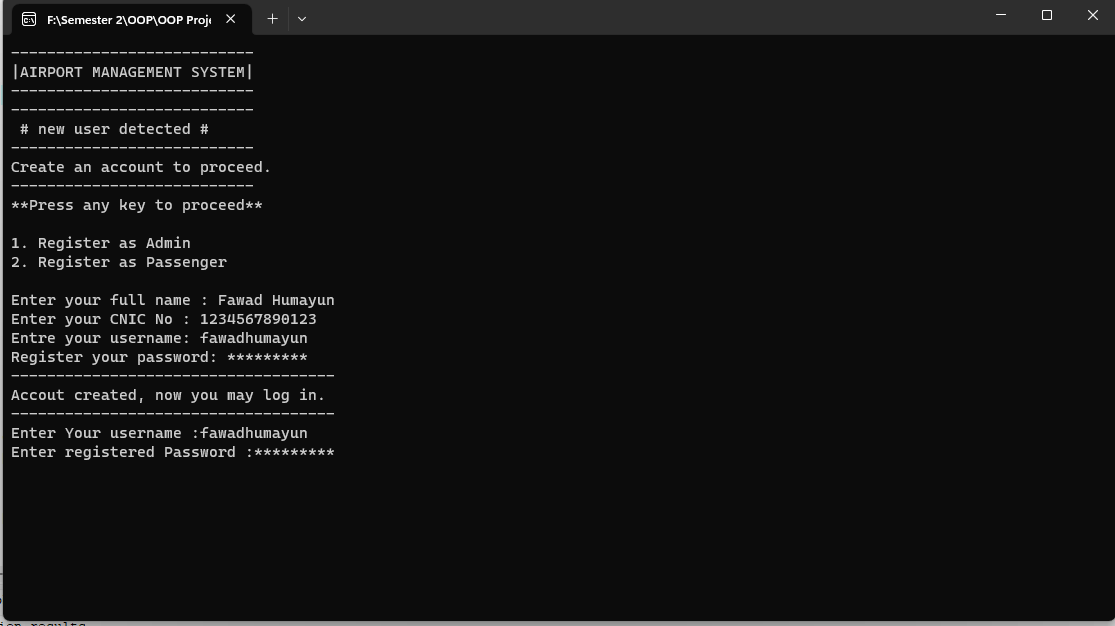
**Passenger:**

The class named passenger, which has private and protected member variables, as well as several virtual member functions that are not implemented in this class.

The private member variables are bseats and eseats, which are of type const int, and cannot be modified once initialized. The protected member variables are firstname, lastname, phone, mealtype, passport, Address, datedep, payment, origin, and destination, which are of various types such as string, int, and char[]. These variables are accessible only by the class and its derived classes.

The constructor of the passenger class takes several parameters such as first name, last name, phone number, address, passport number, and date of departure. It initializes the member variables bseats and eseats to 10 and 50, respectively, and assigns the values of the input parameters to the corresponding member variables.

The passenger class also has several virtual member functions, which are declared but not implemented in this class. These include ldest (), idest (), seats (), Meals (), Registration (), display (), and drive (). These functions can be overridden by derived classes and provide polymorphic behavior to the passenger class.



**Booking:**

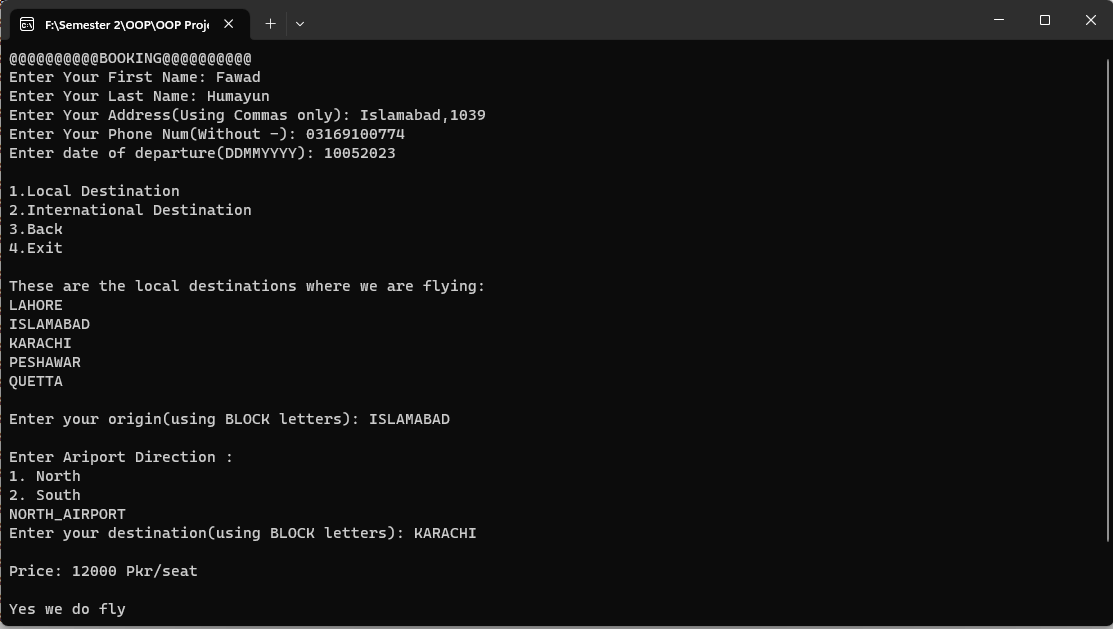
The flight booking system. The code defines a class called "booking" which is derived from a base class called "passenger". The "booking" class contains functions for booking local and international flights, selecting seats in the business or economy class, and displaying the booking details including the price.

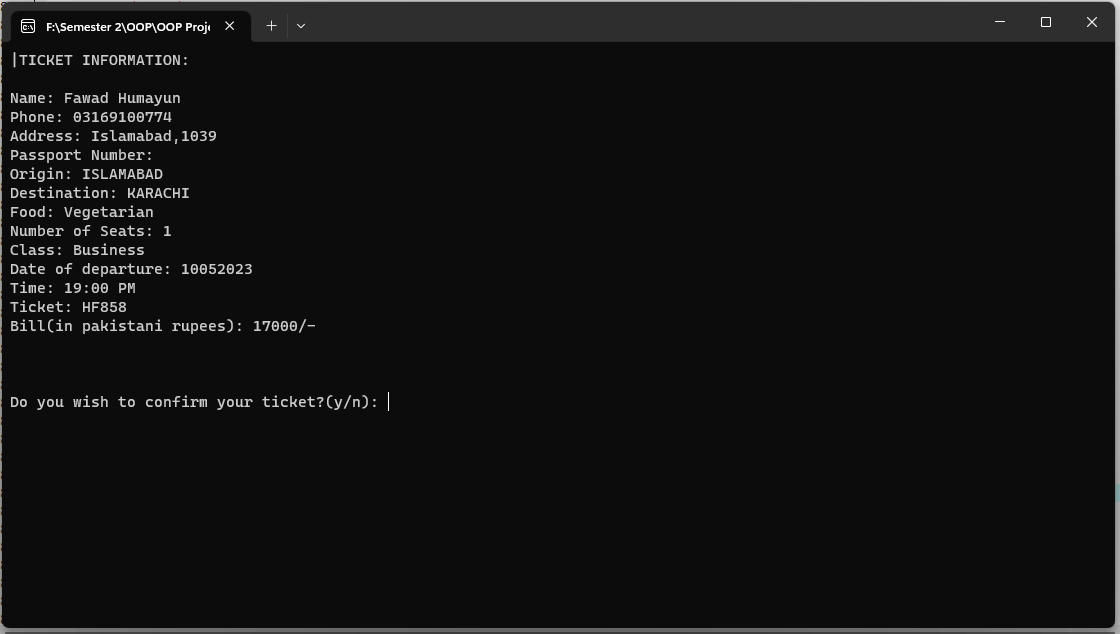
The booking function for local flights is called "ldest" and takes no arguments. It displays a list of local destinations, asks the user for their origin and destination, and checks if both are valid. If they are, it displays the price and returns true; otherwise, it returns false.

The booking function for international flights is called "idest" and takes a string argument "pass" representing the passenger's passport number. It displays a list of international destinations, asks the user for their origin and destination, and checks if both are valid. If they are, it displays the price and returns true; otherwise, it returns false.

The "seats" function allows the user to select seats in the business or economy class. It checks if seats are available and if the user has selected a valid number of seats. If seats are available and the selection is valid, it deducts the number of seats from the available seats and calculates the price.

Overall, the code provides basic functionality for booking flights and selecting seats, but there are some limitations and areas where it could be improved, such as error handling and input validation.



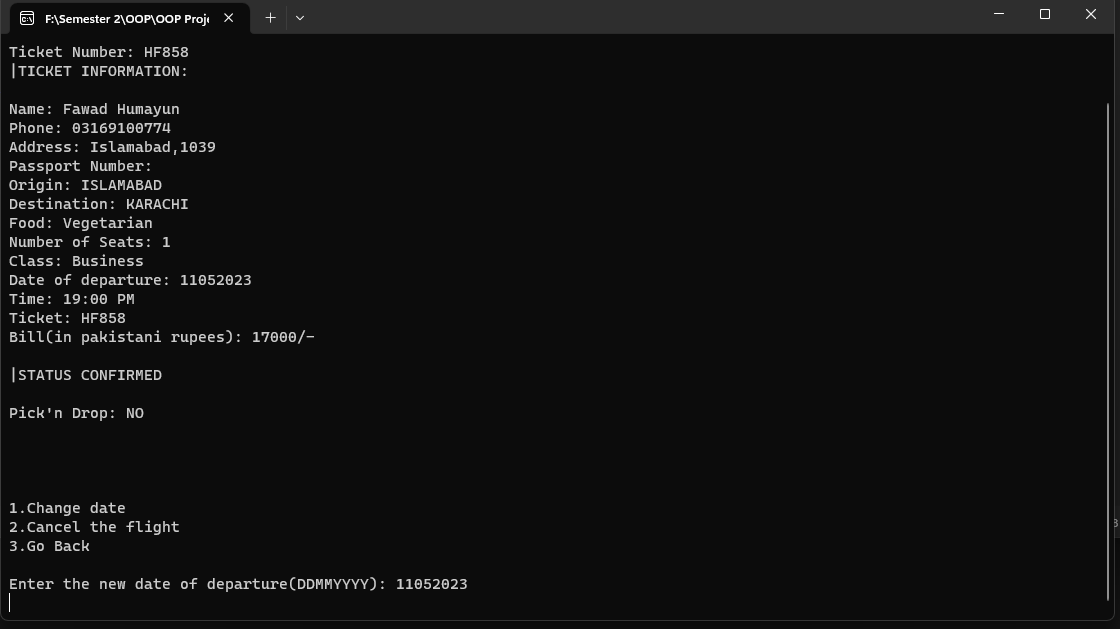
****

**Manage:**

The class is named "Manage" and appears to be a part of a flight booking system. It contains a private string variables temp, y, c2, and an integer variable option. The class has one public method named change () which performs various operations related to flight bookings.

Here's a brief explanation of the change () method:

* It opens a file named "flight.txt" in read mode.
* It asks the user to enter a ticket number or press 0 to exit.
* If the user enters 0, the method returns 1 and exits.
* If the user enters a ticket number, the method reads the file line by line until it finds a line that matches the ticket number entered by the user.
* Once it finds the matching line, it rewinds the file pointer to the beginning of the file.
* It then reads the file line by line and outputs each line to the console until it reaches the end of the file.
* After printing the contents of the file, the method presents the user with three options: change date, cancel the flight, or go back.
* It reads the user's choice from the console and performs the corresponding action.
* If the user chooses to change the date, the method asks the user to enter a new date of departure in the format DDMMYYYY.
* It then creates a new file named "temp.txt" and writes the contents of the original file to the new file, replacing the line that contains the old date of departure with the new date of departure entered by the user.
* It then closes both the original and new files, deletes the original file, renames the new file to "flight.txt", and prints a message indicating that the date has been changed.
* If the user chooses to cancel the flight, the method clears the contents of the "flight.txt" file, deletes the file, prints a message indicating that the booking has been canceled, and returns 1.
* If the user chooses to go back, the method clears the console screen and returns 1.
* If the method is unable to find any matching records, it prints a message indicating that no records were found and returns 1.



**Admin:**

The class called Admin which has three private data members - username, password, user1, and pass1. The class has two public member functions Registration1 and login.

Registration1 function takes input from the user for the username and password, checks if the length of the password is at least 8 characters long, and then saves the username and password to a text file called AdminRegistration.txt.

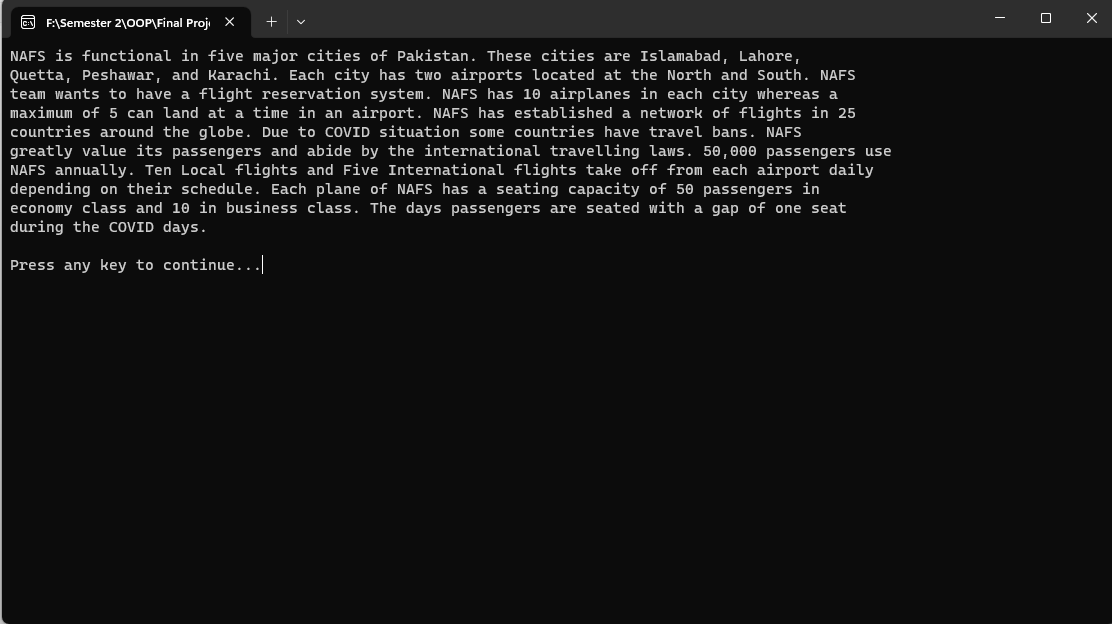
The login function asks the user to enter their username and password, checks if they match the username and password saved in the Admin object, and then displays a message indicating whether the login was successful or not.

Overall, this class seems to be for managing the registration and login of an admin account.

**About:**

This is a simple class called about that contains a single public method called Aboutus (). The purpose of this class is to display some information about NAFS (presumably an airline company).

The Aboutus () method displays some text that describes NAFS's operations and services. It provides information such as the number of cities in which NAFS operates, the number of airplanes it has, its seating capacity, the number of local and international flights it operates, and how it is affected by the COVID-19 pandemic.



**Complain:**

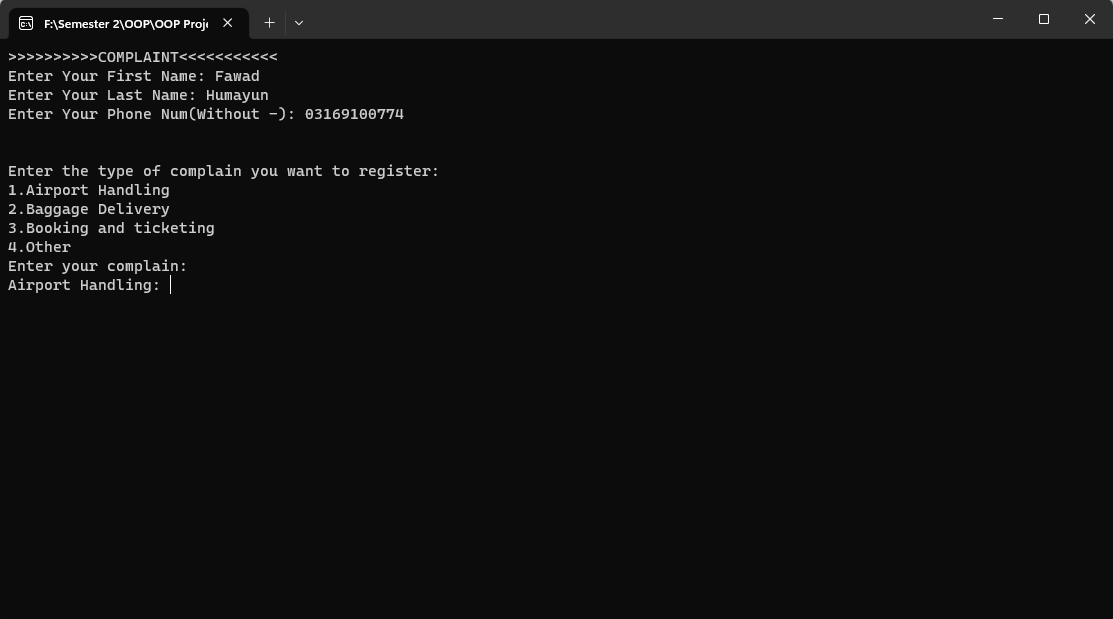
The code defines a class named complain, which is used to register customer complaints.

The class has private data member’s f\_name, l\_name, and p\_num which store the first name, last name, and phone number of the customer respectively.

The class has a public constructor which takes three parameters - the first name, last name, and phone number of the customer.

The class has a public member function type (), which is used to register the complaint type and complaint message entered by the user. The function prompts the user to enter the type of complaint they want to register (Airport Handling, Baggage Delivery, Booking and ticketing, or Other) and then takes the complaint message as input. It then asks the user if they want to edit their complaint or not. If the user chooses to edit their complaint, they are prompted again to enter the complaint type and message. If not, the complaint is saved to a file named "complain.txt" with the customer's name, phone number, complaint type, and complaint message.

Overall, this class is used to register and save customer complaints in a file for further processing by the customer support team.



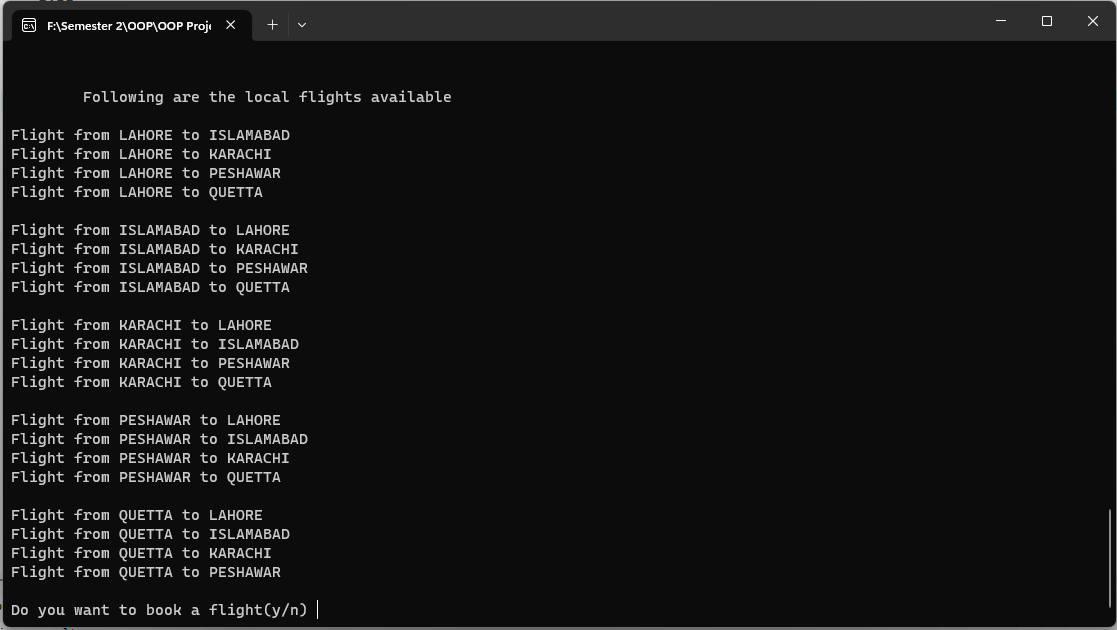
**Routes:**

The class is named "routes" and contains a single public function named "print".

The purpose of this class is to display the available international and local flights to the user and prompt them to book a flight.

The function "print" uses two for loops to iterate over arrays of international and local destinations, skipping any iteration where the source and destination are the same. It then prints out the available flights to the user.

After displaying the available flights, the function prompts the user to book a flight by displaying the message "Do you want to book a flight(y/n)?" and waits for the user's input. If the user chooses to book a flight by entering 'y' or 'Y', the function returns true, otherwise it returns false.



**Main Menu:**

It appears to be a C++ program with an interface for an airport management system. The program allows users to register as either an administrator or a passenger, and then perform various actions depending on their role.

For passengers, the program allows them to book flights, view flight schedules, and manage their bookings. The booking functionality includes inputting personal information such as name, address, and phone number, selecting the departure date, and selecting the flight destination. The program also allows passengers to select their seats and meals during the booking process.

For administrators, the program allows them to manage flights, view flight schedules, and handle complaints. There is also an "About" section for general information about the airport.

The program appears to be incomplete, as several functions are defined but never used. The code could also be improved in terms of formatting, readability, and modularity. Additionally, there are some commented-out lines and debugging statements that could be removed to improve the overall quality of the code.

