SZABIST

Course Name: CSCL1208 Lab - OOP
Course Instructor: Syed Muhammad Hassan

Course Name: CSC1208 - OOP
Course Instructor: Ali Mobin Memon

PROJECT REPORT

Airline Reservation System

Group Member	Registration Number
Mustan Ali	2112121
Umer Amir	2112241
Rohail Rathore	2012362

Table of Content

1. Introduction & Problem Statement

This project is about the Airline Reservation System. The program will allow the user to choose from the menu to perform any relevant task required by the user. Some of the main features are adding and removing the customers. The user can also add and cancel reservations accordingly. It also allows the user to view the list of customers and reservations. Typically, if someone wishes to reserve a ticket, they must contact the closest travel agent. The Airline Reservation System provides an interface to schedule flights and reservations for an airline. It is responsible for managing customers, flight data, and flight scheduling.

2. Features:

- Menu Display
- Add Customer
- Remover Customer
- Display Customer List
- Add Reservation
- Cancel Reservation
- Display Reservations
- Add Flight Description
- Remove Flight Description
- Display Flight Description
- Schedule New Flight
- Cancel Scheduled Flight
- Display Scheduled Flight
- Display Scheduled Flight Passengers

3.0. Program Code:

3.1. Main Class

```
package AirlineReservationSystem;

import java.util.InputMismatchException;
import java.util.Scanner;

public class Main {
   static ConsoleColors cc = new ConsoleColors();

   public static void main(String[] args) {

       Person person1 = new Person("Ali", "123 Street");
       ProjectDB.add(person1);

       Person person2 = new Person("Jeff", "123 Street");
       ProjectDB.add(person2);

       Passenger passenger1 = new Passenger(person1, 1);
       ProjectDB.add(passenger1);

       FlightDescription flightDescription1 = new FlightDescription("Karachi", "Lahore", "01:00",
"02:45", 10);
       ProjectDB.add(flightDescription1);

       ScheduledFlight scheduledFlight1 = new ScheduledFlight(flightDescription1, "25/06/2022");
       ProjectDB.add(scheduledFlight1);


       print_header();
       main_menu();
   }

   //To exit the program
   private static void exitMessage(){
       System.out.println("Thank you for using airline reservation system");
   }

   private static void print_header() {
       System.out.println(cc.GREY_BACKGROUND + "<><><><><><><><><><><><><><><><><><><><>" +
cc.RESET);
       System.out.println(cc.RED_BACKGROUND_BRIGHT + cc.BLACK_BOLD + "       Airline Reservation
System       " + cc.RESET);
       System.out.println(cc.GREY_BACKGROUND + "<><><><><><><><><><><><><><><><><><><><>" +
cc.RESET);
       System.out.print("\n");
   }


   private static void main_menu() {
       System.out.println(cc.RED_BACKGROUND + cc.BLACK_BOLD + "---------->  Main Menu
<-----------" + cc.RESET);
       System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "1- Passengers Menu
" + cc.RESET);
       System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "2- Flight Management Menu
" + cc.RESET);
       System.out.println(cc.GREY_BACKGROUND + cc.RED_BOLD + "3- Exit System
" + cc.RESET);
       System.out.println(cc.RED_BACKGROUND + cc.BLACK_BOLD +
"------------------------------------" + cc.RESET);
       short choice=4;
```

```
        Scanner input = new Scanner(System.in);
        do {
            System.out.print("Choice: ");
            //choice = input.nextShort();
            try{
                choice = input.nextShort();
                input.nextLine();
            }catch (InputMismatchException e){
                System.out.println();
            }
            switch (choice) {
                case 1:
                    System.out.println();
                    passengers_menu();
                    break;
                case 2:
                    System.out.println();
                    flights_menu();
                    break;
                case 3:
                    exitMessage();
                    break;
                case 4:
                    main_menu();
                    break;
                default:
                    System.out.println("ERROR: Choice not valid!");
            }
        } while (choice < 1 || choice > 4);
    }

    private static void passengers_menu() {
        System.out.println(cc.RED_BACKGROUND + cc.BLACK_BOLD + "------->  Passengers Menu
<--------" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "1- Add Customer
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "2- View All Customers
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "3- Remove Customer
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "4- New Reservation
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "5- view All Reservations
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "6- Cancel Reservation
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.RED_BOLD + "7- Main Menu
" + cc.RESET);
        System.out.println(cc.RED_BACKGROUND + cc.BLACK_BOLD +
"-------------------------------------" + cc.RESET);
        short choice=8;
        int index;
        Scanner input = new Scanner(System.in);
        do {
            System.out.print("Choice: ");
            try{
                choice = input.nextShort();
                input.nextLine();
            }catch (InputMismatchException e){
                System.out.println("Invalid Choice");
            }

            switch (choice) {
                case 1:
```

```
            System.out.println("---->  NEW CUSTOMERS  <----");
            input = new Scanner(System.in); // refresh scanner to avoid errors
            System.out.print("Full Name: ");
            String name = input.nextLine();
            System.out.print("Address: ");
            String address = input.nextLine();
        {
            try {
                ProjectDB.add(new Person(name, address));
            } catch (Exception ex) {
                System.out.println("ERROR: File not Found!");
            }
        }
        System.out.println("Added successfully : " + name + "\n");
        passengers_menu();
        break;

        case 2:
            System.out.println("=> CUSTOMERS TABLE  <----");
            Person.show_all();
            passengers_menu();
            break;

        case 3:
            System.out.println("---->  CUSTOMERS TABLE  <----");
            Person.show_all();
            if (ProjectDB.person_list.size() == 0) {
                passengers_menu();
            }
            else {
                do {
                    System.out.print("Customer Index to remove : ");
                    index = input.nextInt();
                } while (index < 1 || index > ProjectDB.person_list.size());
                ProjectDB.person_list.remove(ProjectDB.person_list.get(index - 1));
                System.out.println("Removed Successfully!\n");
                passengers_menu();
            }
            break;
        case 4:
            System.out.println("---->  NEW RESERVATION   <----");
            //Choose person
            Person.show_all();
            if (ProjectDB.person_list.size() == 0) {
                passengers_menu();
            }
            else {
                do {
                    System.out.print("Customer Index : ");
                    index = input.nextInt();
                } while (index < 1 || index > ProjectDB.person_list.size());
                Person p = ProjectDB.person_list.get(index - 1);
                //Choose flight
                ScheduledFlight scf;

                ScheduledFlight.show_all();
                if (ProjectDB.scheduled_flight_list.size() == 0) {
                    passengers_menu();
                }
                else {
                    do {
                        System.out.print("Flight Index : ");
                        index = input.nextInt();
                    } while (index < 1 || index > ProjectDB.scheduled_flight_list.size());
```

```
                            scf = ProjectDB.scheduled_flight_list.get(index - 1);
                            if (scf.capacity ==
Passenger.getSCFlightPassengersCount(scf.flight_number) || ProjectDB.passenger_list.size() == 0)
{
                                System.out.println("This flight is at maximum capacity.");
                            }
                            else {
                                int prevLen = ProjectDB.passenger_list.size();
                                {
                                    try {
                                        ProjectDB.add(new Passenger(p, scf.flight_number));
                                    } catch (Exception ex) {
                                        System.out.println("ERROR : FILE NOT FOUND !");
                                    }
                                }
                                int afterLen = ProjectDB.passenger_list.size();
                                if (prevLen != afterLen) {
                                    System.out.println("Reservation completed : " + p.name + " ("
+ scf.from + " -> " + scf.to + ")\n");
                                }
                            }
                            passengers_menu();
                        }
                    }
                    //passengers_menu();
                    break;
                case 5:
                    System.out.println("---->  RESERVATIONS TABLE  <----");
                    Passenger.show_all();
                    passengers_menu();
                    break;
                case 6:
                    System.out.println("---->  RESERVATIONS TABLE  <----");
                    Passenger.show_all();
                    if (ProjectDB.passenger_list.size() == 0) {
                        passengers_menu();
                    }
                    else {
                        do {
                            System.out.print("Passenger Index to Cancel trip for : ");
                            index = input.nextInt();
                        } while (index < 1 || index > ProjectDB.passenger_list.size());
                        ProjectDB.passenger_list.remove(ProjectDB.passenger_list.get(index - 1));
                        System.out.println("Reservation Canceled Successfully!\n");
                        passengers_menu();
                    }
                    break;

                case 7:
                    System.out.println();
                    main_menu();
                    break;
                case 8:
                    passengers_menu();
                    break;
                default:
                    System.out.println("ERROR: Choice not valid");
            }
        } while (choice < 1 || choice > 8);

    }

    private static void flights_menu() {
        System.out.println(cc.RED_BACKGROUND + cc.BLACK_BOLD + "---->  Flight Management Menu 
<----" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "1- Add New Flight Description 
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "2- View All Flight Description 
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "3- Remove Flight Description 
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "4- Schedule New Flight 
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "5- view All Scheduled Flights 
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "6- Cancel Scheduled Flight 
" + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.BLACK_BOLD + "7- View Scheduled Flight 
Passengers " + cc.RESET);
        System.out.println(cc.GREY_BACKGROUND + cc.RED_BOLD + "8- Main Menu 
" + cc.RESET);
        System.out.println(cc.RED_BACKGROUND + cc.BLACK_BOLD + 
"------------------------------------" + cc.RESET);
        short choice=9;
        int index;
        Scanner input = new Scanner(System.in);
        do {
            System.out.print("Choice: ");
            //choice = input.nextShort();
            try{
                choice = input.nextShort();
                input.nextLine();
            }catch (InputMismatchException e){
                System.out.println("Invalid Choice");
            }

            switch (choice) {
                case 1:
                    System.out.println("---->  NEW FLIGHT DESCRIPTION  <----");
                    input = new Scanner(System.in); // refresh scanner to avoid errors
                    System.out.print("From : ");
                    String from = input.nextLine();
                    System.out.print("To   : ");
                    String to = input.nextLine();
                    String depTime, arrTime;

                    System.out.print("Departure time (HH:MM): ");
                    depTime = input.nextLine();

                    System.out.print("Arrival   time (HH:MM): ");
                    arrTime = input.nextLine();


                    System.out.print("Capacity : ");
                    input = new Scanner(System.in);
                    int cap = input.nextInt();
                    int prevSize = ProjectDB.flight_desc_list.size();
                {
                    try {
                        ProjectDB.add(new FlightDescription(from, to, depTime, arrTime, cap));
                    } catch (Exception ex) {
                        System.out.println("ERROR: File not Found!");
                    }
                }
                int afterSize = ProjectDB.flight_desc_list.size();
                if (prevSize != afterSize) {
```

```
                System.out.println("Flight Description added successfully : " + from + " -> " 
+ to + "\n");
            }
            flights_menu();
            break;

            case 2:
                System.out.println("---->  FLIGHT DESCRIPTION TABLE  <----");
                FlightDescription.show_all();
                flights_menu();
                break;
            case 3:
                System.out.println("---->  FLIGHT DESCRIPTION TABLE  <----");
                FlightDescription.show_all();
                if (ProjectDB.flight_desc_list.size() == 0) {
                    flights_menu();
                }
                else {
                    do {
                        System.out.print("Flight description index to remove : ");
                        index = input.nextInt();
                    } while (index < 1 || index > ProjectDB.flight_desc_list.size());
                    ProjectDB.flight_desc_list.remove(ProjectDB.flight_desc_list.get(index - 
1));

                    System.out.println("Flight description removed Successfully!\n");
                    flights_menu();
                }
                break;
            case 4:
                System.out.println("---->  FLIGHT DESCRIPTION TABLE  <----");
                FlightDescription.show_all();
                if (ProjectDB.flight_desc_list.size() == 0) {
                    flights_menu();
                }
                else {
                    do {
                        System.out.print("Flight description index to schedule : ");
                        index = input.nextInt();
                    } while (index < 1 || index > ProjectDB.flight_desc_list.size());
                    FlightDescription fd = ProjectDB.flight_desc_list.get(index - 1);
                    input = new Scanner(System.in); // refresh scanner to avoid errors
                    String date;

                    System.out.print("Date (YYYY/MM/DD) : ");
                    date = input.nextLine();

                    int prevLen = ProjectDB.scheduled_flight_list.size();
                    {
                        try {
                            ProjectDB.add(new ScheduledFlight(fd, date));
                        } catch (Exception ex) {
                            System.out.println("ERROR : FILE NOT FOUND !");
                        }
                    }
                    int afterLen = ProjectDB.scheduled_flight_list.size();
                    if (prevLen != afterLen) {
                        System.out.println("Scheduled " + date + " for flight : " + fd.from + 
" -> " + fd.to + "\n");
                    }
                    flights_menu();
                }
                //flights_menu();
                break;
                case 5:
                    System.out.println("---->  SCHEDULED FLIGHTS TABLE  <----");
                    ScheduledFlight.show_all();
                    flights_menu();
                    break;

                case 6:
                    System.out.println("---->  SCHEDULED FLIGHT TABLE  <----");
                    ScheduledFlight.show_all();
                    if (ProjectDB.scheduled_flight_list.size() == 0) {
                        flights_menu();
                    }
                    else {
                        do {
                            System.out.print("Scheduled Flight index to canceled : ");
                            index = input.nextInt();
                        } while (index < 1 || index > ProjectDB.scheduled_flight_list.size());

ProjectDB.scheduled_flight_list.remove(ProjectDB.scheduled_flight_list.get(index - 1));
                        System.out.println("Scheduled Flight & Reservations canceled
Successfully!\n");
                        flights_menu();
                    }
                    break;

                case 7:
                    System.out.println("---->  SCHEDULED FLIGHT TABLE  <----");
                    ScheduledFlight.show_all();
                    do {
                        System.out.print("Flight Index : ");
                        index = input.nextInt();
                    } while (index < 1 || index > ProjectDB.scheduled_flight_list.size());
                    int flight_num = ProjectDB.scheduled_flight_list.get(index - 1).flight_number;
                    Passenger.show_only_flight_no(flight_num);
                    flights_menu();
                    break;
                case 8:
                    System.out.println();
                    main_menu();
                    break;
                case 9:
                    flights_menu();
                    break;
                default:
                    System.out.println("ERROR: Choice not valid");
            }
        } while (choice < 1 || choice > 9);
    }
}
```

3.2. Person Class

```
package AirlineReservationSystem;

public class Person {
   public String name;
   public String address;

   public Person(String name, String address) {
```

```java
        this.name = name;
        this.address = address;
    }

    public static void show_all() {
        int counter = 0;
        for (int i = 0; i < 93; i++)
            System.out.print("-");
        System.out.println();
        System.out.printf("%5s | %-30s | %-50s |\n", "Index", "Full Name", "Address");
        for (int i = 0; i < 93; i++)
            if (i == 6 || i == 39 || i == 92)
                System.out.print("|");
            else
                System.out.print("-");
        System.out.println();

        if (ProjectDB.person_list.isEmpty()) {
            System.out.println("\t==> No Customers added yet <==");
        }
        for (Person p : ProjectDB.person_list) {
            System.out.printf("%5d | %-30s | %-50s |\n", ++counter, p.name, p.address);
        }
        for (int i = 0; i < 93; i++)
            System.out.print("-");
        System.out.println();
    }

}
```

3.3. FlightDescription

```java
package AirlineReservationSystem;

public class FlightDescription {
    public String from;
    public String to;
    public String departure_time;
    public String arrival_time;
    public int capacity;

    public FlightDescription(String from, String to, String departureTime, String arrivalTime, int
capacity) {
        this.from = from;
        this.to = to;
        this.departure_time = departureTime;
        this.arrival_time = arrivalTime;
        this.capacity = capacity;
    }


    public static void show_all() {
        int counter = 0;
        for (int i = 0; i < 90; i++)
            System.out.print("-");
        System.out.println();
        System.out.printf("%5s | %-20s | %-20s | %-10s | %-10s | %-8s |\n", "Index", "FROM", "To",
"Dep Time", "Arr Time", "Capacity");
        for (int i = 0; i < 90; i++)
            if (i == 6 || i == 29 || i == 52 || i == 65 || i == 78 || i == 89)
                System.out.print("|");
            else
```

```
                System.out.print("-");
            System.out.println();

            if (ProjectDB.flight_desc_list.isEmpty()) {
                System.out.println("\t==> No Flight descriptions added yet <==");
            }
            for (FlightDescription fd : ProjectDB.flight_desc_list) {
                System.out.printf("%5d | %-20s | %-20s | %-10s | %-10s | %8d |\n",
                        ++counter, fd.from, fd.to, fd.departure_time, fd.arrival_time, fd.capacity);
            }
            for (int i = 0; i < 90; i++)
                System.out.print("-");
            System.out.println();
        }
    }
```

3.4. PassengerClass

```
package AirlineReservationSystem;

import java.util.ArrayList;

public class Passenger extends Person {
   public int flight_number;

   public Passenger(Person person, int flight_number) {
        super(person.name, person.address);
        this.flight_number = flight_number;
   }

   public static int getSCFlightPassengersCount(int flight_num) {
        int counter = 0;
        for (Passenger pa : ProjectDB.passenger_list) {
            if (pa.flight_number == flight_num)
                counter++;
        }
        return counter;

   }

   public static void show_all() {
        int counter = 0;
        for (int i = 0; i < 48; i++)
            System.out.print("-");
        System.out.println();
        System.out.printf("%5s | %-5s | %-30s |\n", "Index", "FN", "Full Name");
        for (int i = 0; i < 48; i++)
            if (i == 6 || i == 14 || i == 47)
                System.out.print("|");
            else
                System.out.print("-");
        System.out.println();

        if (ProjectDB.passenger_list.isEmpty()) {
            System.out.println("\t==> No Reservations added yet <==");
        }

        for (Passenger p : ProjectDB.passenger_list) {
            System.out.printf("%5d | %5d | %-30s |\n", ++counter, p.flight_number, p.name);
        }
        for (int i = 0; i < 48; i++)
            System.out.print("-");
        System.out.println();
    }

    public static void show_only_flight_no(int flight_num) {
        ArrayList<Passenger> output = new ArrayList<>();
        for (Passenger pa : ProjectDB.passenger_list) {
            if (pa.flight_number == flight_num)
                output.add(pa);
        }

        int counter = 0;
        for (int i = 0; i < 40; i++)
            System.out.print("-");
        System.out.println();

        System.out.printf("%5s | %-30s |\n", "Index", "Full Name");
        for (int i = 0; i < 40; i++)
            if (i == 6 || i == 39)
                System.out.print("|");
            else
                System.out.print("-");
        System.out.println();

        if (output.isEmpty()) {
            System.out.println("\t=> No Reservations added yet <=");
        }

        for (Passenger p : output) {
            System.out.printf("%5d | %-30s |\n", ++counter, p.name);
        }
        for (int i = 0; i < 40; i++)
            System.out.print("-");
        System.out.println();
    }

}
```

3.5. ScheduledFlight Class

```
package AirlineReservationSystem;

public class ScheduledFlight extends FlightDescription {
    public String date;
    public int flight_number;

    public ScheduledFlight(FlightDescription f_desc, String date) {
        super(f_desc.from, f_desc.to, f_desc.departure_time, f_desc.arrival_time,
f_desc.capacity);
        this.date = date;
        this.flight_number = generate_flight_num();
    }

    private static int generate_flight_num() {
        int max = 0;
        for (ScheduledFlight scf : ProjectDB.scheduled_flight_list) {
            if (max < scf.flight_number)
                max = scf.flight_number;
        }
        return max + 1;
    }
    public static void show_all() {
        int counter = 0;
        for (int i = 0; i < 113; i++)
            System.out.print("-");
        System.out.println();
        System.out.printf("%5s | %-5s | %-10s | %-20s | %-20s | %-10s | %-10s | %-10s |\n",
"Index", "FN", "Date", "FROM", "To", "Dep Time", "Arr Time", "Passengers");
        for (int i = 0; i < 113; i++)
            if (i == 6 || i == 14 || i == 27 || i == 50 || i == 73 || i == 86 || i == 99 || i ==
112)
                System.out.print("|");
            else
                System.out.print("-");
        System.out.println();

        if (ProjectDB.scheduled_flight_list.isEmpty()) {
            System.out.println("\t==> No Scheduled flights added yet <==");
        }

        for (ScheduledFlight scf : ProjectDB.scheduled_flight_list) {
            int pNumber = Passenger.getSCFlightPassengersCount(scf.flight_number);
            String pCount = (pNumber == scf.capacity) ? "Full(" + pNumber + ")" :
Integer.toString(pNumber);
            System.out.printf("%5d | %5d | %-10s | %-20s | %-20s | %-10s | %-10s | %10s |\n",
                    ++counter, scf.flight_number, scf.date, scf.from, scf.to, scf.departure_time,
scf.arrival_time, pCount);
        }
        for (int i = 0; i < 113; i++)
            System.out.print("-");
        System.out.println();
    }

}
```

3.6. ProjectDB Class

```
package AirlineReservationSystem;

import java.util.ArrayList;

public class ProjectDB {

    public static ArrayList<Person> person_list = new ArrayList<>();
    public static ArrayList<Passenger> passenger_list = new ArrayList<>();
    public static ArrayList<FlightDescription> flight_desc_list = new ArrayList<>();
    public static ArrayList<ScheduledFlight> scheduled_flight_list = new ArrayList<>();


    public static void add(Person person) {
        for (Person p : person_list) {
            if (p.name.equals(person.name)) {
                System.out.println("Can't save this data!");
                System.out.println(person.name + " : Already saved!");
                return;
            }
        }
        person_list.add(person);
    }

    public static void add(Passenger passenger) {
        for (Passenger p : passenger_list) {
            if (p.flight_number == passenger.flight_number && p.name.equals(passenger.name)) {
```

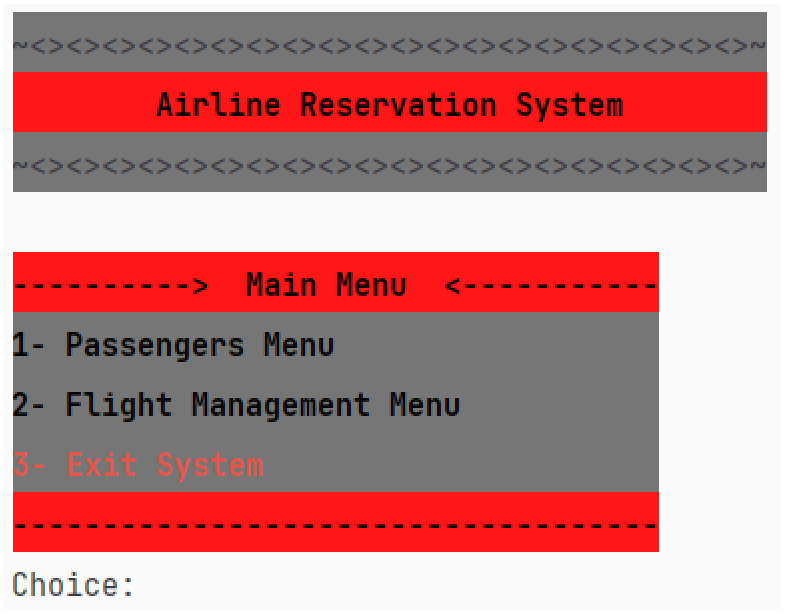
```java
                System.out.println("Can't save this data!");
                System.out.println(passenger.name + " : Already reserved this flight!");
                return;
            }
        }
        passenger_list.add(passenger);
    }

    public static void add(FlightDescription flight_desc) {
        for (FlightDescription flight : flight_desc_list) {
            if (flight.arrival_time.equals(flight_desc.arrival_time) &&
                    flight.departure_time.equals(flight_desc.departure_time) &&
                    flight.from.equals(flight_desc.from) &&
                    flight.to.equals(flight_desc.to) &&
                    flight.capacity == flight_desc.capacity) {
                System.out.println("Can't save this data!");
                System.out.println("This Flight description Already exists!");
                return;
            }
        }
        flight_desc_list.add(flight_desc);
    }

    public static void add(ScheduledFlight sc_flight) {
        for (ScheduledFlight flight : scheduled_flight_list) {
            if (flight.arrival_time.equals(sc_flight.arrival_time) &&
                    flight.departure_time.equals(sc_flight.departure_time) &&
                    flight.from.equals(sc_flight.from) &&
                    flight.to.equals(sc_flight.to) &&
                    flight.capacity == sc_flight.capacity &&
                    flight.date.equals(sc_flight.date)) {
                System.out.println("Can't save this data!");
                System.out.println("This Flight Already scheduled!");
                return;
            }
        }
        scheduled_flight_list.add(sc_flight);
    }
}
```

3.6. ConsoleColors Class

```java
package AirlineReservationSystem;

public class ConsoleColors {
    // Reset
    public final String RESET = "\033[0m"; // Text Reset

    // Bold
    public final String BLACK_BOLD = "\033[1;30m"; // BLACK
    public final String RED_BOLD = "\033[1;31m"; // RED

    // Background
    public final String RED_BACKGROUND = "\033[48;5;9m"; // RED
    public final String GREY_BACKGROUND = "\033[48;5;243m"; // GRAY

    // Bright backgrounds
    public final String RED_BACKGROUND_BRIGHT = "\033[0;101m";// RED

}
```

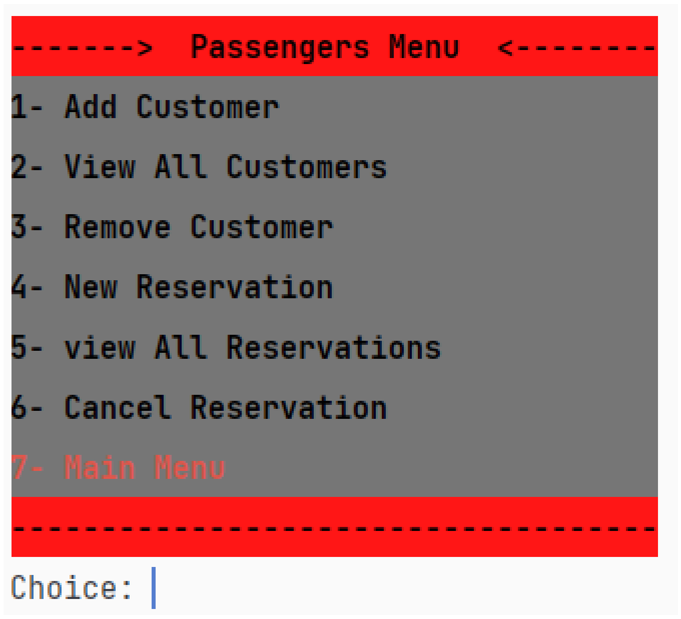
4.0. Output

4.1. Main Menu



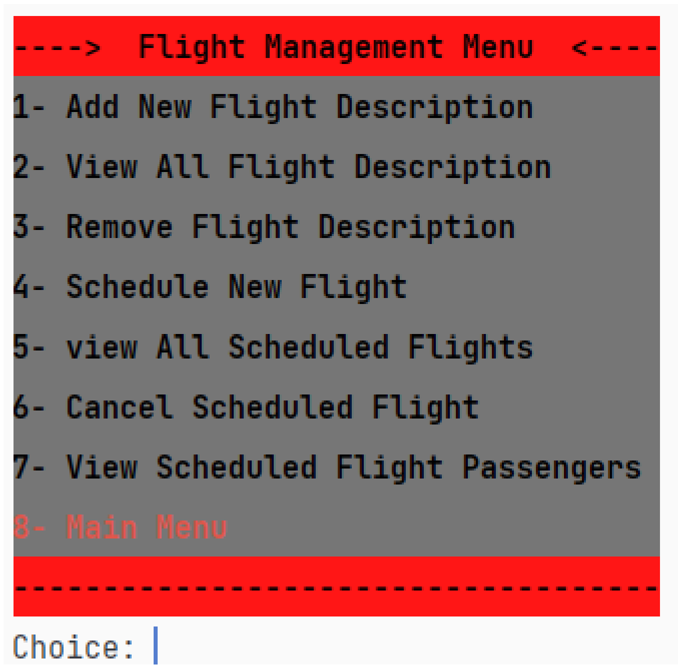
```
~<><><><><><><><><><><><><><><><><><><><><>~
        Airline Reservation System
~<><><><><><><><><><><><><><><><><><><><><>~

---------->  Main Menu  <-----------
1- Passengers Menu
2- Flight Management Menu
3- Exit System
------------------------------------
Choice:
```

4.2. Passengers Menu



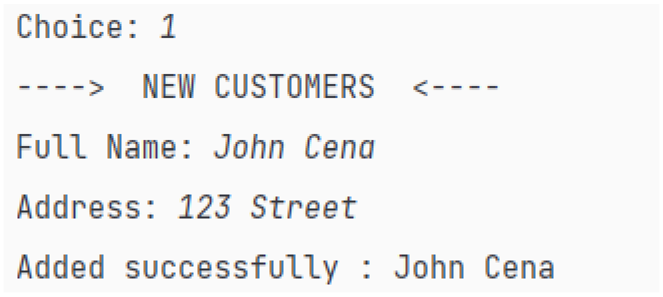
```
------->  Passengers Menu  <--------
1- Add Customer
2- View All Customers
3- Remove Customer
4- New Reservation
5- view All Reservations
6- Cancel Reservation
7- Main Menu
------------------------------------
Choice:
```

4.3. Flight Management Menu



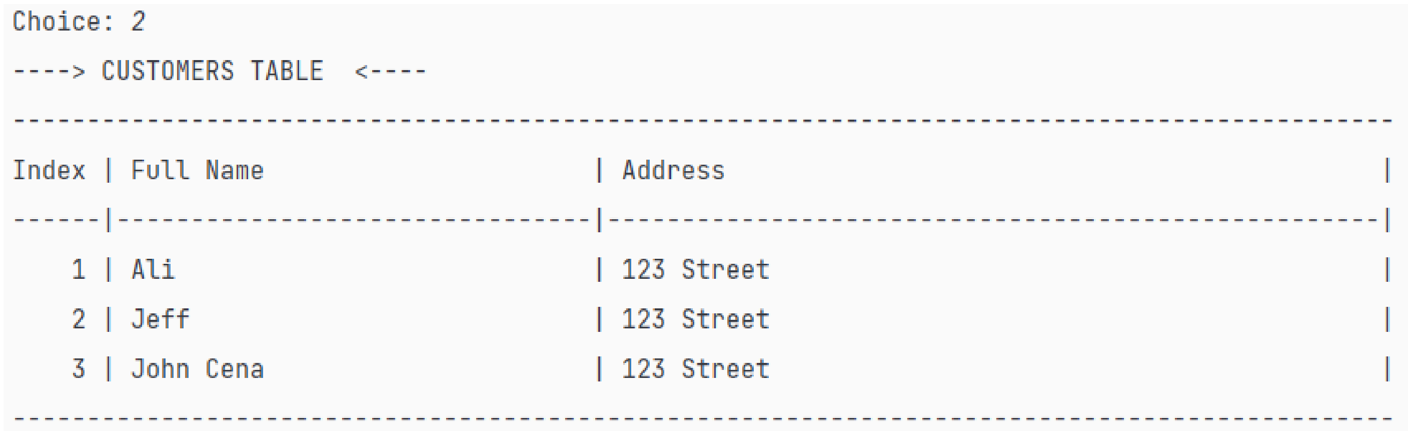
```
---->  Flight Management Menu  <----
1- Add New Flight Description
2- View All Flight Description
3- Remove Flight Description
4- Schedule New Flight
5- view All Scheduled Flights
6- Cancel Scheduled Flight
7- View Scheduled Flight Passengers
8- Main Menu
------------------------------------
Choice: 
```

4.4. Add Customer



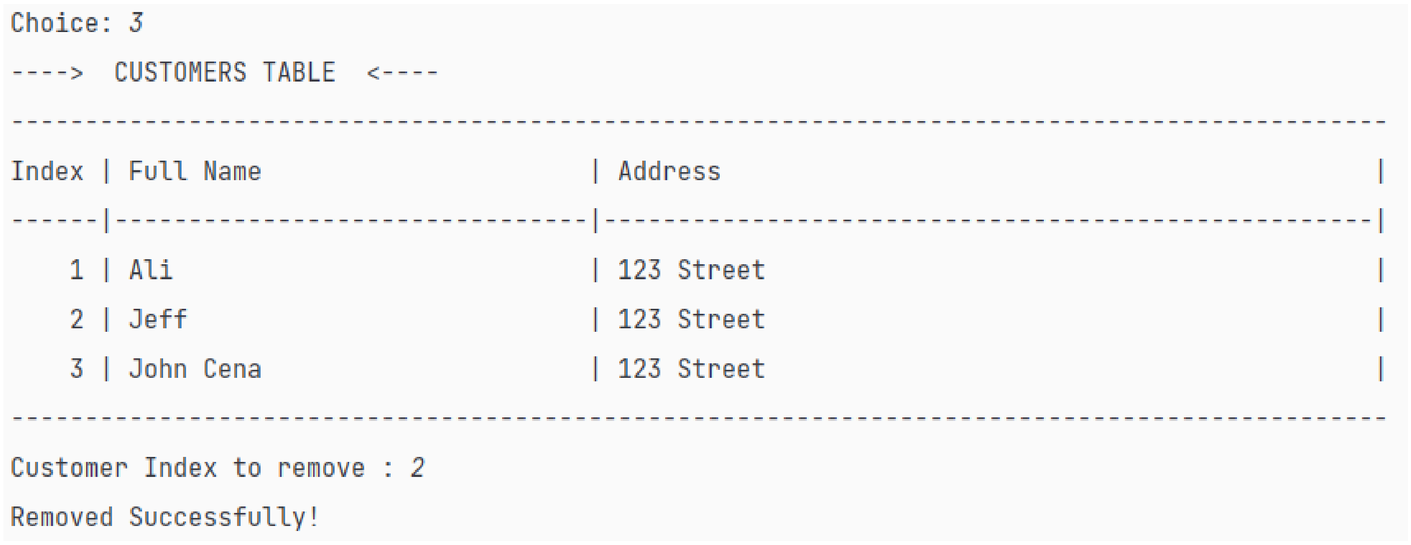
```
Choice: 1
---->  NEW CUSTOMERS  <----
Full Name: John Cena
Address: 123 Street
Added successfully : John Cena
```

4.5. View ALL Customers



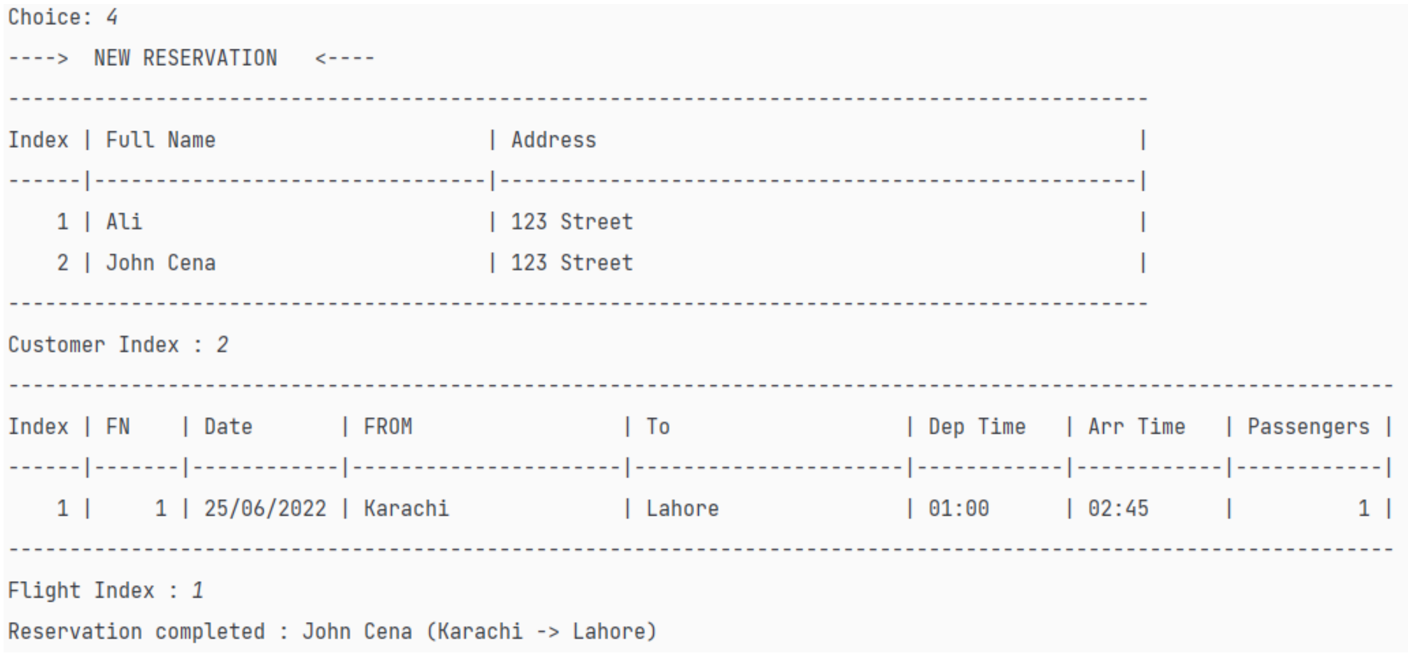
```
Choice: 2
----> CUSTOMERS TABLE  <----
----------------------------------------------------------------------------
Index | Full Name                       | Address                                          |
------|---------------------------------|--------------------------------------------------|
    1 | Ali                             | 123 Street                                       |
    2 | Jeff                            | 123 Street                                       |
    3 | John Cena                       | 123 Street                                       |
----------------------------------------------------------------------------
```

4.6. Remove Customers



```
Choice: 3
---->  CUSTOMERS TABLE  <----
---------------------------------------------------------------------------------------------
Index | Full Name                      | Address                                            |
------|--------------------------------|----------------------------------------------------|
    1 | Ali                            | 123 Street                                         |
    2 | Jeff                           | 123 Street                                         |
    3 | John Cena                      | 123 Street                                         |
---------------------------------------------------------------------------------------------
Customer Index to remove : 2
Removed Successfully!
```

4.7. New Reservation



```
Choice: 4
---->  NEW RESERVATION   <----
---------------------------------------------------------------------------------------------
Index | Full Name                      | Address                                            |
------|--------------------------------|----------------------------------------------------|
    1 | Ali                            | 123 Street                                         |
    2 | John Cena                      | 123 Street                                         |
---------------------------------------------------------------------------------------------
Customer Index : 2
-----------------------------------------------------------------------------------------------------------------
Index | FN    | Date       | FROM                 | To                   | Dep Time   | Arr Time   | Passengers |
------|-------|------------|----------------------|----------------------|------------|------------|------------|
    1 |     1 | 25/06/2022 | Karachi              | Lahore               | 01:00      | 02:45      |          1 |
-----------------------------------------------------------------------------------------------------------------
Flight Index : 1
Reservation completed : John Cena (Karachi -> Lahore)
```

4.8. View ALL Reservations

```
Choice: 5
---->  RESERVATIONS TABLE  <----
-----------------------------------------------
Index | FN     | Full Name                      |
------|-------|--------------------------------|
    1 |      1 | Ali                            |
    2 |      1 | John Cena                      |
-----------------------------------------------
```

4.9. Add New Flight Description

```
Choice: 1
---->  NEW FLIGHT DESCRIPTION  <----
From : Lahore
To   : Karachi
Departure time (HH:MM): 01:30
Arrival   time (HH:MM): 2:45
Capacity : 250
Flight Description added successfully : Lahore -> Karachi
```

4.10. View All Flight Description

```
Choice: 2
---->  FLIGHT DESCRIPTION TABLE  <----
-----------------------------------------------------------------------------------------
Index | FROM                 | To                   | Dep Time    | Arr Time    | Capacity |
------|---------------------|---------------------|------------|------------|----------|
    1 | Karachi              | Lahore               | 01:00       | 02:45       |       10 |
    2 | Lahore               | Karachi              | 01:30       | 2:45        |      250 |
-----------------------------------------------------------------------------------------
```

4.11. Remove Flight Description

```
Choice: 3
---->  FLIGHT DESCRIPTION TABLE  <----
------------------------------------------------------------------------------------------
Index | FROM                 | To                   | Dep Time   | Arr Time   | Capacity |
------|----------------------|----------------------|------------|------------|----------|
    1 | Karachi              | Lahore               | 01:00      | 02:45      |       10 |
    2 | Lahore               | Karachi              | 01:30      | 2:45       |      250 |
------------------------------------------------------------------------------------------
Flight description index to remove : 2
Flight description removed Successfully!
```

4.12. Schedule New Flight

```
Choice: 4
---->  FLIGHT DESCRIPTION TABLE  <----
------------------------------------------------------------------------------------------
Index | FROM                 | To                   | Dep Time   | Arr Time   | Capacity |
------|----------------------|----------------------|------------|------------|----------|
    1 | Karachi              | Lahore               | 01:00      | 02:45      |       10 |
------------------------------------------------------------------------------------------
Flight description index to schedule : 1
Date (YYYY/MM/DD) : 2022/06/15
Scheduled 2022/06/15 for flight : Karachi -> Lahore
```

4.13. view All Scheduled Flights

```
Choice: 5
---->  SCHEDULED FLIGHTS TABLE  <----
------------------------------------------------------------------------------------------------------------------
Index | FN    | Date       | FROM                 | To                   | Dep Time   | Arr Time   | Passengers |
------|-------|------------|----------------------|----------------------|------------|------------|------------|
    1 |     1 | 25/06/2022 | Karachi              | Lahore               | 01:00      | 02:45      |          2 |
    2 |     2 | 2022/06/15 | Karachi              | Lahore               | 01:00      | 02:45      |          0 |
------------------------------------------------------------------------------------------------------------------
```

4.14. Cancel Scheduled Flight

```
Choice: 6
---->  SCHEDULED FLIGHT TABLE  <----
---------------------------------------------------------------------------------------------------------------------
Index | FN    | Date       | FROM                 | To                   | Dep Time   | Arr Time   | Passengers |
------|-------|------------|----------------------|----------------------|------------|------------|------------|
    1 |     1 | 25/06/2022 | Karachi              | Lahore               | 01:00      | 02:45      |          2 |
    2 |     2 | 2022/06/15 | Karachi              | Lahore               | 01:00      | 02:45      |          0 |
---------------------------------------------------------------------------------------------------------------------
Scheduled Flight index to canceled : 2
Scheduled Flight & Reservations canceled Successfully!
```

4.15. View Scheduled Flight Passengers

```
Choice: 7
---->  SCHEDULED FLIGHT TABLE  <----
---------------------------------------------------------------------------------------------------------------------
Index | FN    | Date       | FROM                 | To                   | Dep Time   | Arr Time   | Passengers |
------|-------|------------|----------------------|----------------------|------------|------------|------------|
    1 |     1 | 25/06/2022 | Karachi              | Lahore               | 01:00      | 02:45      |          2 |
---------------------------------------------------------------------------------------------------------------------
Flight Index : 1
----------------------------------------
Index | Full Name                       |
------|---------------------------------|
    1 | Ali                             |
    2 | John Cena                       |
----------------------------------------
```

5. Future Scope

Airline companies now play a significant part in transportation, and in order to make reservations reliable, they require a system that will make bookings simpler, quicker, and safer. This project was made to meet the airline reservation system requirements. By using this application, the company can provide reservation services and information to their customers without the limitation of office hours or manpower. Moreover, data is managed efficiently and accurately which would help customers and airline companies to fetch everything easily.

6. Conclusion

In general, the project ’s objectives have been fulfilled. This enables both passengers and admin the greatest services. If we look at this project over the long term, we will see that the world is evolving and everything is now digital. As a result, this project will be productive and improve everyone's workload by reducing human efforts and replacing manual paper work.