KONGU ENGINEERING COLLEGE

(Autonomous)

Perundurai, Erode – 638 060

DEPARTMENT OF ELECTRONICS AND INSTRUMENTATION ENGINEERING

AIRLINE RESERVATION SYSTEM

AN MICROPROJECT REPORT

for

JAVA PROGRAMMING (22ITC31)

Submitted by

ABISHA CARIN B 23EIR004

ANIRUDDHA PRANAV M 23EIR005

ANNA SHRUTHI V 23EIR006

KONGU ENGINEERING COLLEGE

(Autonomous)

Estd : 1984

Transform Yourself

Perundurai, Erode – 638 060

DEPARTMENT OF ELECTRONICS AND INSTRUMENTATION ENGINEERING

BONAFIDE CERTIFICATE

Name & Roll No. **ABISHA CARIN(23EIR004)**

ANIRUADDHA PRANAV(23EIR005)

ANNA SHRUTHI V(23EIR006)

Course Code : **22ITC31**

Course Name : **JAVA PROGRAMMING**

Semester : **III**

Certified that this is a bonafide record of work for application project done by the above students for **22ITC31-JAVA PROGRAMMING** during the academic year **2024-2025.**

Submitted for the Viva Voce Examination held on _____

Faculty In-Charge **Year In-charge**

TABLE OF CONTENTS

AIRLINE RESERVATION SYSTEM

Abstract

An Airline Reservation System (ARS) is a software application that facilitates the management of flight reservations, ticket booking, and customer information. This system plays a critical role in automating and streamlining the process of flight bookings for passengers, travel agents, and airline operators. The purpose of this project is to design and implement a simple Airline Reservation System using Java, enabling users to search for flights, book tickets, view available flights, and manage reservations efficiently.

Problem Statement

In modern weather applications, managing temperature records for various cities dynamically and efficiently is critical. There is a need for a lightweight system that enables the user to interactively add, update, or delete temperature records and display them in different formats. This project addresses this need by creating a simple console-based weather forecasting system to manage city-wise temperatures.

Methodology

User Registration: Users can register in the system to create a profile and store booking information.

Flight Search: Passengers can search for flights based on parameters like source, destination, and date.

Booking Process: Once a user selects a flight, they can book a ticket, which includes providing their name, contact information, and payment details.

Ticket Cancellation: Passengers can cancel their tickets and receive a refund or cancel the booking completely.

Flight Availability: The system displays available flights based on user input and ensures seat availability is updated after booking or cancellation.

Scalability: The system should be scalable to accommodate more passengers, flights, and bookings.

Reliability: The system should consistently handle booking and flight search operations.

Performance: The system must respond quickly to user queries for flight searches, bookings, and cancellations.

Implementation

Contains flight details such as flight number, source, destination, date, time, available seats, and ticket price.

Includes methods for displaying flight details and updating the available seat count after a booking or cancellation.

Manages flight reservations, including ticket booking, passenger details, and flight assignments.

Updates seat availability and tracks the reservation status.

The main class that integrates all functionalities, including flight searching, booking, and cancellation.

Provides methods to display available flights, book tickets, cancel reservations, and display booking history.

Results and Discussion

1.Ticket Cancellation

Test Case: The user cancels a previously booked ticket for a given flight.

Expected Result: The system should increase the available seat count and remove the reservation details from the system.

Actual Result: The system correctly handled the cancellation by updating the seat availability and showing the confirmation message that the booking was canceled.

Conclusion: The cancellation feature is working as intended, updating both the seat count and reservation list.

2.Flight Availability After Booking and Cancellation

Test Case: After booking and canceling tickets, the system should reflect the changes in seat availability.

Expected Result: The available seat count should accurately reflect bookings and cancellations.

Actual Result: After each booking and cancellation, the available seat count for the respective flights was updated correctly.

Conclusion: The system effectively tracks seat availability and updates it in real time.

Conclusion

The **Airline Reservation System** developed in Java is a functional application designed to streamline the booking and management of flight reservations. Through the use of object-oriented programming principles, the system enables passengers to search for flights, book tickets, and manage their reservations efficiently. The key features of the system, such as flight availability checks, ticket booking, and cancellation, were successfully implemented and tested, providing a seamless user experience in a command-line interface (CLI).

Sample coding

```
import java.util.Scanner;
public class AirlineReservationSystem {
private static String[] passengerNames = new String[5];
private static String[] passengerPhones = new String[5];
private static String[] passengerCities = new String[5];
private static String[] passengerDOBs = new String[5];
private static String[][] airlines = {
{"A001", "Airline A", "New York", "Los Angeles", "100"},
{"A002", "Airline B", "San Francisco", "Chicago", "120"},
{"A003", "Airline C", "Miami", "Dallas", "150"},
{"A004", "Airline D", "Boston", "Seattle", "180"},
{"A005", "Airline E", "Houston", "Washington DC", "200"}
```

```
};
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Enter your phone number or Date of Birth (DD/MM/YYYY): ");
String input = scanner.nextLine();
int passengerIndex = findPassenger(input);
if (passengerIndex == -1) {
System.out.println("You are not registered. Please register now.");
System.out.print("Enter your name: ");
String name = scanner.nextLine();
System.out.print("Enter your phone number: ");
String phone = scanner.nextLine();
System.out.print("Enter your city: ");
String city = scanner.nextLine();
System.out.print("Enter your Date of Birth (DD/MM/YYYY): ");
String dob = scanner.nextLine();
addPassenger(name, phone, city, dob);
} else {
System.out.println("Welcome back, " + passengerNames[passengerIndex]);
}
System.out.println("\nAvailable airlines:");
for (String[] airline : airlines) {
System.out.println("Code: " + airline[0] + ", Airline: " + airline[1] + ", From: " +
airline[2] + ", To: " + airline[3]);
}
System.out.print("\nEnter airline code to book a ticket: ");
String airlineCode = scanner.nextLine();
bookTicket(airlineCode);
}
```

```
private static int findPassenger(String input) {
for (int i = 0; i < passengerPhones.length; i++) {
if (passengerPhones[i] != null && (passengerPhones[i].equals(input) ||
passengerDOBs[i].equals(input))) {
return i; // Return the index if the passenger is found
}
}
return -1; // Return -1 if not found
}
private static void addPassenger(String name, String phone, String city, String dob) {
for (int i = 0; i < passengerNames.length; i++) {
if (passengerNames[i] == null) { passengerNames[i] = name;
passengerPhones[i] = phone;
passengerCities[i] = city;
passengerDOBs[i] = dob;
System.out.println("Registration successful!");
return;
}
}
System.out.println("Passenger list is full, unable to register.");
}
private static void bookTicket(String airlineCode) {
Scanner scanner = new Scanner(System.in);
boolean airlineFound = false;
for (String[] airline : airlines) {
if (airline[0].equals(airlineCode)) {
airlineFound = true;
System.out.println("You selected: " + airline[1] + " from " + airline[2] + " to " +
```

```
airline[3]);
System.out.println("Ticket price: $" + airline[4]);
System.out.print("Do you want a single trip (1) or round trip (2)? ");
int tripType = scanner.nextInt();
double totalAmount = Double.parseDouble(airline[4]);
if (tripType == 2) {
totalAmount *= 2; // Round trip is double the price
}
System.out.println("Total amount: $" + totalAmount);
System.out.print("Choose payment method (1. Cash, 2. Internet): ");
int paymentMethod = scanner.nextInt();
scanner.nextLine

PaymentMode paymentMode = (paymentMethod == 1) ? PaymentMode.CASH :
PaymentMode.ONLINE;
processPayment(totalAmount, paymentMode);
return
}
}
// If the airline code is invalid
if (!airlineFound) {
System.out.println("Invalid airline code. Please try again.");
}
}
private static void processPayment(double amount, PaymentMode mode) {
Scanner scanner = new Scanner(System.in);
if (mode == PaymentMode.CASH) {
System.out.println("Payment method: Cash");
System.out.println("Booking completed successfully.");
```

```
} else if (mode == PaymentMode.ONLINE) {
System.out.print("Proceed with the transaction? (yes/no): ");
String proceed = scanner.nextLine();
if ("yes".equalsIgnoreCase(proceed)) {
System.out.println("Checking bank balance...");
Bank bank = new Bank();
if (bank.checkBalance(amount)) {
System.out.println("Transaction successful. Booking completed.");
} else {
System.out.println("Insufficient funds. Transaction failed.");
}
} else {
System.out.println("Transaction cancelled.");
}
}
}
public enum PaymentMode {
CASH, ONLINE
}
// Bank class to simulate checking bank balance
static class Bank {
public boolean checkBalance(double amount) {
double bankBalance = 500.0; // Simulating a fixed bank balance
return bankBalance >= amount;
}
}
}
```

OUTPUT:-

Enter your phone number or Date of Birth (DD/MM/YYYY): 9876543210

You are not registered. Please register now.

Enter your name: Alice Smith

Enter your phone number: 9876543210

Enter your city: Los Angeles

Enter your Date of Birth (DD/MM/YYYY): 20/03/1992

Registration successful!

Available airlines:

Code: A001, Airline: Airline A, From: New York, To: Los Angeles

Code: A002, Airline: Airline B, From: San Francisco, To: Chicago

Code: A003, Airline: Airline C, From: Miami, To: Dallas

Code: A004, Airline: Airline D, From: Boston, To: Seattle

Code: A005, Airline: Airline E, From: Houston, To: Washington DC

Enter airline code to book a ticket: A001

You selected: Airline A from New York to Los Angeles
Ticket price: $100.0

Do you want a single trip (1) or round trip (2)? 1
Total amount: $100.0
Choose payment method (1. Cash, 2. Internet): 1
Payment method: Cash
Booking completed successfully.
=== Code Execution Successful ===\

Faculty Incharge **Academic Coordinator**

HOD