

AIRLINE RESERVATION SYSTEM

A PROJECT REPORT

Submitted by

VINO AGUSTIN (2303811724321123)

in partial fulfillment of requirements for the award of the course

CGB1201 – JAVA PROGRAMMING

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by
AICTE, New Delhi)

SAMAYAPURAM – 621 112

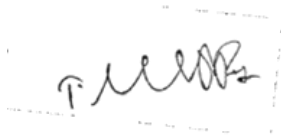
DECEMBER, 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on “**AIRLINE RESERVATION SYSTEM**” is the bonafide work of **VINO AGUSTIN (2303811724321123)** who carried out the project work during the academic year 2024 - 2025 under my supervision.



Signature

Dr. T. AVUDAIAPPAN M.E., Ph.D.,

HEAD OF THE DEPARTMENT,

Department of Artificial Intelligence,
K. Ramakrishnan College of Technology,
Samayapuram, Trichy -621 112.



Signature

Mrs. S. GEETHA M.E.,

SUPERVISOR,

Department of Artificial Intelligence,
K. Ramakrishnan College of Technology,
Samayapuram, Trichy -621 112.

Submitted for the viva-voce examination held on 3.12.24



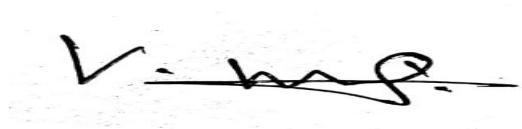
INTERNAL EXAMINER



EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “**AIRLINE RESERVATION SYSTEM**” is the result of original work done by me and best of my knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This project report is submitted on the partial fulfillment of the requirement of the award of the **CGB1201 – JAVA PROGRAMMING**.



Signature

VINO AGUSTIN V

Place: Samayapuram

Date: 3/12/2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and indebtedness to our institution, **“K. Ramakrishnan College of Technology (Autonomous)”**, for providing us with the opportunity to do this project.

I extend our sincere acknowledgement and appreciation to the esteemed and honourable Chairman, **Dr. K. RAMAKRISHNAN, B.E.**, for having provided the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director, **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding our project and offering an adequate duration to complete it.

I would like to thank **Dr. N. VASUDEVAN, M.TECH., Ph.D.**, Principal, who gave the opportunity to frame the project to full satisfaction.

I thank **Dr.T.AVUDAIAPPAN, M.E.,Ph.D.**, Head of the Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for providing her encouragement in pursuing this project.

I wish to convey our profound and heartfelt gratitude to our esteemed project guide **Mrs.S.GEETHA M.E.**, Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for her incalculable suggestions, creativity, assistance and patience, which motivated us to carry out this project.

I render our sincere thanks to the Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards.

MISSION OF THE INSTITUTION

- Be a centre of excellence for technical education in emerging technologies by exceeding the needs of industry and society.
- Be an institute with world class research facilities.
- Be an institute nurturing talent and enhancing competency of students to transform them as all- round personalities respecting moral and ethical values.

VISION AND MISSION OF THE DEPARTMENT

To excel in education, innovation and research in Artificial Intelligence and Data Science to fulfill industrial demands and societal expectations.

Mission 1: To educate future engineers with solid fundamentals, continually improving teaching methods using modern tools.

Mission 2: To collaborate with industry and offer top-notch facilities in a conducive learning environment.

Mission 3: To foster skilled engineers and ethical innovation in AI and Data Science for global recognition and impactful research.

Mission 4: To tackle the societal challenge of producing capable professionals by instilling employability skills and human values.

PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

PEO 1: Compete on a global scale for a professional career in Artificial Intelligence and Data Science.

PEO 2: Provide industry-specific solutions for the society with effective communication and ethics.

PEO 3: Hone their professional skills through research and lifelong learning initiatives.

PROGRAM OUTCOMES

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO 1:** Capable of working on data-related methodologies and providing industry-focussed solutions.
- **PSO2:** Capable of analysing and providing a solution to a given real-world problem by designing an effective program.

ABSTRACT

The Airline Reservation System (ARS) is a simple Java application that simulates flight bookings and reservations. It consists of three main classes: Flight, Reservation, and AirlineReservationSystem. The Flight class holds details about flights, such as flight number, origin, destination, and available seats, along with a method to book seats. The Reservation class stores passenger information and the flight booked. The AirlineReservationSystem manages a collection of flights and reservations, allowing users to search for flights, book seats, and view existing reservations. The system uses a Scanner for user input and provides options for searching flights by origin and destination, booking flights, and viewing reservations. This basic system aims to demonstrate the core functionality of flight booking and reservation management.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	vi
1	INTRODUCTION	1
	1.1 Objective	1
	1.2 Overview	1
2	PROJECT METHODOLOGY	2
	2.1 Proposed Work	2
	2.2 Block Diagram	2
3	JAVA PROGRAMMING CONCEPTS	3
	3.1 CLASSES AND OBJECTS	3
	3.2 GUI COMPONENTS	3
4	MODULE DESCRIPTION	4
	3.1 Flight Management	4
	3.2 Reservation Management	4
	3.3 Search Functionality	4
	3.4 User Interaction	4
	3.5 System Exit	4
5	CONCLUSION	6
	REFERENCES	7
	APPENDICES	8
	APPENDIX-A SOURCE CODE	8
	APPENDIX-B SCREEN SHOT	11

CHAPTER 1

INTRODUCTION

1.1 OBJECTIVE

The objective of this Airline Reservation System is to create a simple and effective platform for managing flight reservations. The system allows users to search for available flights, book tickets, and view reservations. It aims to simulate the core functionalities of an airline reservation system, ensuring ease of use, data accuracy, and efficient seat management.

1.1 OVERVIEW

This Airline Reservation System allows users to interact with the system to search for available flights based on the origin and destination, book tickets for a flight, and view all the reservations made. The system tracks flights, their available seats, and handles booking by updating seat availability. The primary components include managing flight data (flight number, origin, destination, available seats), making reservations, and ensuring a smooth user experience through menu-driven choices. It demonstrates the application of fundamental programming concepts in a real-world scenario.

1.2 JAVAPROGRAMMINGCONCEPT:

- **Classes and Objects:** The system is object-oriented, with classes like Flight, Reservation, and AirlineReservationSystem representing different entities.
- **ArrayLists:** Used to store dynamic collections of flights and reservations, allowing the program to manage and update lists of data as needed.
- **Methods and Constructors:** Methods like bookFlight() and searchFlights() handle specific functionality, while constructors are used to initialize objects such as flights and reservations.
- **Control Structures:** Conditional statements (like if and switch) and loops (for and while) help in the implementation of the interactive interface for booking and searching flight

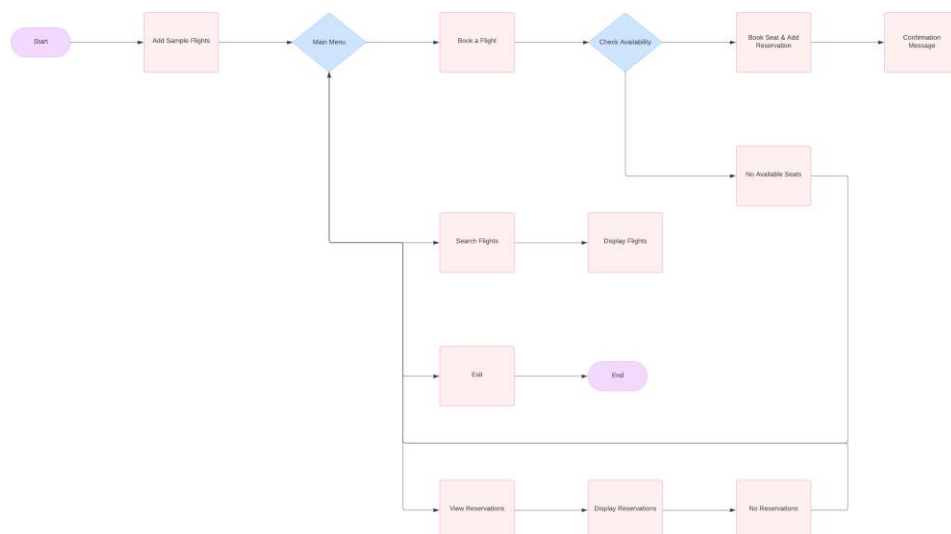
CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work:

The Airline Reservation System aims to provide a platform for customers to search for flights, book tickets, and manage their reservations. It will include functionalities for adding new flights, searching available flights by origin and destination, booking flights with available seats, and viewing all reservations. The system will allow customers to interact with the system via a console-based user interface, making the booking process more efficient and organized. Additionally, it ensures that only available seats are booked and prevents overbooking.

2.2 Block Diagram



CHAPTER 3

JAVA PROGRAMMING CONCEPTS

3.1 OBJECT-ORIENTED PROGRAMMING (OOP):

- **Classes and Objects:** Represent users, admins, and tickets with separate classes to manage system functionalities.
- **Encapsulation:** Protect user data using private fields and public getter/setter methods for secure access.
- **Inheritance:** Future-proof the system by allowing easy extension of classes for additional features.

1. Java Collections Framework:

- **HashMap:** Stores user credentials and ticket details efficiently for quick lookups.
- **ArrayList:** Manages dynamic data such as tickets.

2. Control Flow Mechanisms:

- **Loops and Conditional Statements:** Help navigate through menus and perform actions based on user choices.
- **Switch Case:** Organizes user inputs and directs actions in the menu system.

3. Input Handling with Scanner:

Scanner: Captures user input for actions like registration, login, and ticket submission.

3.2 GUI COMPONENTS

1. **Frame:** The main window displaying menus and user interactions.
2. **Label:** Displays text like "Username" and "Password" for user guidance
3. **Button:** Initiates actions like login, registration, and viewing tickets.

CHAPTER 4

MODULE DESCRIPTION

4. 1 Module 1: Flight Management

This module manages the flight details such as flight numbers, origin, destination, and the number of available seats. It allows for adding new flights to the system and ensures that the flight data is stored and available for other modules to access. The flight management module also handles the booking of seats.

4. 2 Module 2: Reservation Management

This module handles the creation and management of flight reservations. It records the passenger's name and the flight they've booked. It also ensures that once a seat is booked, the availability is updated. This module allows users to view all the reservations made.

4. 3 Module 3: Search Functionality

This module allows users to search for flights based on origin and destination. It checks all available flights in the system and displays the flights that match the criteria. This helps users find the flights they wish to book based on their travel preferences.

4. 4 Module 4: User Interaction

The user interaction module is responsible for interacting with the user, accepting input, and providing output. It uses the console to present a simple menu that allows users to choose between searching flights, booking flights, viewing reservations, or exiting the system.

4. 5 Module 5: System Exit

This module handles the graceful shutdown of the application. It ensures that once the user opts to exit, the system ends smoothly without errors, closing resources like the scanner used for input.

CHAPTER 5

CONCLUSION

RESULTS AND DISCUSSION:

5.1 Search Flights:

- The user enters an origin and destination (e.g., "newyork" to "london").
- Presumably, this would display available flights between the entered locations (though not shown explicitly in the screenshot).

5.2 Book a Flight:

- The user is prompted to:
 - Enter a flight number (e.g., "AI102").
 - Provide the passenger's name (e.g., "Kamaraju").
- The program confirms successful booking with a message indicating the passenger's name and flight number.

5.3 View Reservations:

- Displays the details of booked reservations.
- Example: A reservation for "Kamaraju" on flight "AI102" is displayed with additional flight details (e.g., route from "Paris to Tokyo" and available seats).

5.4 Exit:

- Allows the user to terminate the program.

CONCLUSION

The Airline Reservation System (ARS) effectively demonstrates how to manage flights, reservations, and bookings through a simple yet robust Java-based console application. The system incorporates fundamental concepts of object-oriented programming, such as encapsulation, inheritance, and polymorphism, to manage flight and reservation data. By allowing users to search for flights, book seats, and view existing reservations, the ARS ensures a user-friendly interface and efficient operations. This implementation highlights how real-world systems can be modeled and implemented in Java, making it a valuable learning project. The modular design enables easy scalability and future enhancements, such as incorporating graphical interfaces or integrating with real-time databases. Overall, this system serves as a practical example of leveraging Java programming to solve real-world problems.

REFERENCES:

Java Programming Books :

4."Head First Java" by Kathy Sierra and Bert Bates

- An excellent resource for beginners to understand Java concepts and OOP principles.

5."Effective Java" by Joshua Bloch

- A comprehensive guide to best practices in Java programming, great for intermediate and advanced learners.

6."Java: The Complete Reference" by Herbert Schildt

- A detailed and widely used reference book for Java, covering core concepts and advanced topics.

Online Resources :

4.Oracle Java Documentation

- <https://docs.oracle.com/javase/tutorial/>
- Official documentation for learning Java fundamentals, classes, and libraries.

5.GeeksforGeeks - Java Tutorials

- <https://www.geeksforgeeks.org/java/>
- Articles and examples covering Java programming, data structures, and algorithms.

APPENDIX APPENDIX-A

(Coding)

```
import java.util.ArrayList;

import java.util.Scanner;

class Flight {
    String flightNumber;String origin;
    String destination;int availableSeats;

    public Flight(String flightNumber, String origin, String destination, int
        availableSeats) {this.flightNumber = flightNumber;
        this.origin = origin;
        this.destination = destination;
        this.availableSeats = availableSeats;
    }

    public boolean bookSeat()
    {if (availableSeats > 0) {
        availableSeats--;
        return true;
    }
    return false;
}

@Override
public String toString() {
    return "Flight " + flightNumber + " from " + origin + " to " + destination + " | Available Seats: " + availableSeats;
}
}

class Reservation
{ String
passengerName;Flight
flight;

    public Reservation(String passengerName, Flight flight)
    {this.passengerName = passengerName;
    this.flight = flight;
    }

    @Override
    public String toString() {
        return "Reservation for " + passengerName + " on " + flight;
    }
}

class AirlineReservationSystem {
```

```

private ArrayList<Flight> flights = new ArrayList<>();
private ArrayList<Reservation> reservations = new ArrayList<>();

public void addFlight(String flightNumber, String origin, String destination, int availableSeats)
{
    flights.add(new Flight(flightNumber, origin, destination, availableSeats));
}

public void searchFlights(String origin, String destination)
{
    for (Flight flight : flights) {
        if (flight.origin.equalsIgnoreCase(origin) && flight.destination.equalsIgnoreCase(destination))
        {
            System.out.println(flight);
        }
    }
}

public void bookFlight(String flightNumber, String passengerName)
{
    for (Flight flight : flights) {
        if (flight.flightNumber.equals(flightNumber))
        {
            if (flight.bookSeat()) {
                reservations.add(new Reservation(passengerName, flight));
                System.out.println("Booking successful for " + passengerName + " on flight " + flightNumber);
            } else {
                System.out.println("No available seats on flight " + flightNumber);
            }
        }
        return;
    }
    System.out.println("Flight " + flightNumber + " not found.");
}

public void viewReservations()
{
    if (reservations.isEmpty()) {
        System.out.println("No reservations found.");
    } else {
        for (Reservation reservation : reservations)
        {
            System.out.println(reservation);
        }
    }
}

public class Main {
    public static void main(String[] args) {
        AirlineReservationSystem ars = new AirlineReservationSystem();
        Scanner scanner = new Scanner(System.in);

        // Adding sample flights
        ars.addFlight("AI101", "New York", "London", 5);
        ars.addFlight("AI102", "Paris", "Tokyo", 2);
        ars.addFlight("AI103", "Mumbai", "Dubai", 10);

        while (true) {
            System.out.println("\n--- Airline Reservation System ---");

```

```

System.out.println("1. Search Flights");
System.out.println("2. Book a Flight");
System.out.println("3. View Reservations");
System.out.println("4. Exit"); System.out.print("Enter
your choice: ");
int choice = scanner.nextInt();

switch (choice)
{
    case 1:
        System.out.print("Enter origin: ");
        String origin = scanner.next();
        System.out.print("Enter destination: ");
        String destination = scanner.next();
        ars.searchFlights(origin, destination);
        break;
    case 2:
        System.out.print("Enter flight number: ");
        String flightNumber = scanner.next();
        System.out.print("Enter passenger name: ");
        String passengerName = scanner.next();
        ars.bookFlight(flightNumber, passengerName);
        break;
    case 3:
        ars.viewReservations();
        break;
    case 4:
        System.out.println("Exiting...");
        scanner.close();
        return;
    default:
        System.out.println("Invalid choice. Please try again.");
}
}
}
}

```

APPENDIX-B

RESULT

```
Available Flights:
F101: New York to Los Angeles, Seats Available: 100/100

--- Airline Reservation System ---
1. Add Flight
2. View Flights
3. Book Ticket
4. Cancel Ticket
5. View Bookings
6. Exit
Enter your choice: 3
Enter flight number: F101
Enter passenger name: John Doe
Ticket booked successfully for John Doe on flight F101.

--- Airline Reservation System ---
1. Add Flight
2. View Flights
3. Book Ticket
4. Cancel Ticket
5. View Bookings
6. Exit
Enter your choice: 5

Bookings:
Passenger: John Doe, Flight: F101

--- Airline Reservation System ---
1. Add Flight
2. View Flights
3. Book Ticket
4. Cancel Ticket
```

```
Passenger: John Doe, Flight: F101

--- Airline Reservation System ---
1. Add Flight
2. View Flights
3. Book Ticket
4. Cancel Ticket
5. View Bookings
6. Exit
Enter your choice: 4
Enter flight number: F101
Enter passenger name: John Doe
Booking for John Doe on flight F101 canceled successfully.

--- Airline Reservation System ---
1. Add Flight
2. View Flights
3. Book Ticket
4. Cancel Ticket
5. View Bookings
6. Exit
Enter your choice: 6
Exiting the system. Goodbye!
```

```
--- Airline Reservation System ---
1. Search Flights
2. Book a Flight
3. View Reservations
4. Exit
Enter your choice: 1
Enter origin: newyork
Enter destination: london

--- Airline Reservation System ---
1. Search Flights
2. Book a Flight
3. View Reservations
4. Exit
Enter your choice: 2
Enter flight number: AI102
Enter passenger name: Kamaraju
Booking successful for Kamaraju on flight AI102

--- Airline Reservation System ---
1. Search Flights
2. Book a Flight
3. View Reservations
4. Exit
Enter your choice: 3
Reservation for Kamaraju on Flight AI102 from Paris to Tokyo | Available Seats: 1

--- Airline Reservation System ---
1. Search Flights
2. Book a Flight
3. View Reservations
4. Exit
Enter your choice: 4
Exiting...
```