AIRLINE RESERVATION SYSTEM

A PROJECT REPORT

Submitted by

PUSHBAJA K-2303811710422124

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

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

SAMAYAPURAM – 621 112

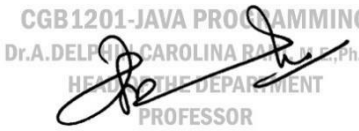
NOVEMBER- 2024

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 **PUSHBAJA K (2303811710422124)** who carried out the project work during the academic year 2024 - 2025 under my supervision.



SIGNATURE	**SIGNATURE**
Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,	Mrs.K.Valli Priyadharshini, M.E.,(Ph.D.,),
HEAD OF THE DEPARTMENT	**SUPERVISOR**
PROFESSOR	ASSISTANT PROFESSOR
Department of CSE	Department of CSE
K.Ramakrishnan College of Technology (Autonomous)	K.Ramakrishnan College of Technology (Autonomous)
Samayapuram–621112.	Samayapuram–621112.

Submitted for the viva-voce examination held on 03/12/2024

INTERNAL EXAMINER	EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **"AIRLINE RESERVATION SYSTEM"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING.**

Signature

PUSHBAJA K

Place: Samayapuram

Date: 03/12/2024

ACKNOWLEDGEMENT

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

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for 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 to our project and offering adequate duration in completing our project.

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

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mrs. K. VALLI PRIYADHARSHINI, M.E., (Ph.D.,),** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

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

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

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 manaGHJKL;'gement 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.

ABSTRACT

The Airline Reservation System is a desktop-based application developed using Java and the Abstract Window Toolkit (AWT). The system aims to simplify the process of managing flight bookings by providing an intuitive graphical user interface. It is designed as a standalone application that allows users to perform key tasks such as viewing available flights, booking tickets by entering passenger details, canceling existing bookings, and exiting the application seamlessly.

The project leverages object-oriented programming principles and event- driven design to ensure modularity and responsiveness. A lightweight, simulated data layer is used to store flight information, with options for future integration of dynamic data handling via databases. The graphical interface features clear navigation options and validation mechanisms to enhance user experience and reduce errors.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The Airline Reservation System is a software application developed to manage flight bookings, cancellations, and display available flights. It provides a user-friendly interface using Java's AWT for the graphical user interface, allowing users to interact with the system by booking tickets or canceling existing bookings. The system is built with a modular approach using Object-Oriented Programming (OOP) principles like encapsulation, inheritance, and polymorphism to organize the code efficiently and make it easy to maintain and extend. The project aims to offer a simple, interactive solution for managing airline reservations and improve the user experience.	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -2** **PO6 -3** **PO7 -3** **PO8 -3** **PO9 -3** **PO10 -3**	**PSO1 -3** **PSO2 -3** **PSO3 -2** **PSO4-2**

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Objective

The Airline Reservation System aims to simplify the ticket booking process by providing users with a functional and user-friendly graphical interface. It allows users to view available flights, book tickets, cancel bookings, and exit the application seamlessly. The system is designed using Java's AWT framework, focusing on modularity, scalability, and effective application of object-oriented programming (OOP) concepts.

1.2 Overview

The Airline Reservation System is a desktop-based application designed to manage basic airline operations. The system provides four core functionalities:

1. Show Flights:

   Displays a list of available flights with details such as flight ID, destination, and ticket price.

2. Book Ticket:

   Facilitates ticket booking by collecting passenger details and flight information.

3. Cancel Booking:

   Enables users to cancel a booking by entering a valid ticket ID.

4. Exit:

   Allows users to close the application safely.

1.3 Java Programming Concepts

1. Object-Oriented Programming (OOP):

The program employs core OOP principles like encapsulation, inheritance, and polymorphism to create a well-structured system.

- ❖ Encapsulation: All application logic is encapsulated within the AirlineReservationSystem class, ensuring that the implementation details are hidden from other parts of the program.
- ❖ Inheritance: The class extends Frame from the java.awt package, inheriting properties and methods for creating GUI components.
- ❖ Polymorphism: The program overrides the action Performed() method from the ActionListener interface to handle user interactions dynamically, enabling modular event handling.

2. Event-Driven Programming: Event-driven programming is integral to the application's interactivity.

3. AWT (Abstract Window Toolkit): AWT forms the backbone of the GUI design for this application.

4. Exception Handling: The program integrates basic error-handling techniques to ensure a smooth user experience.

5. Control Structures: Conditional statements (if-else) drive the application’s decision-making processes.

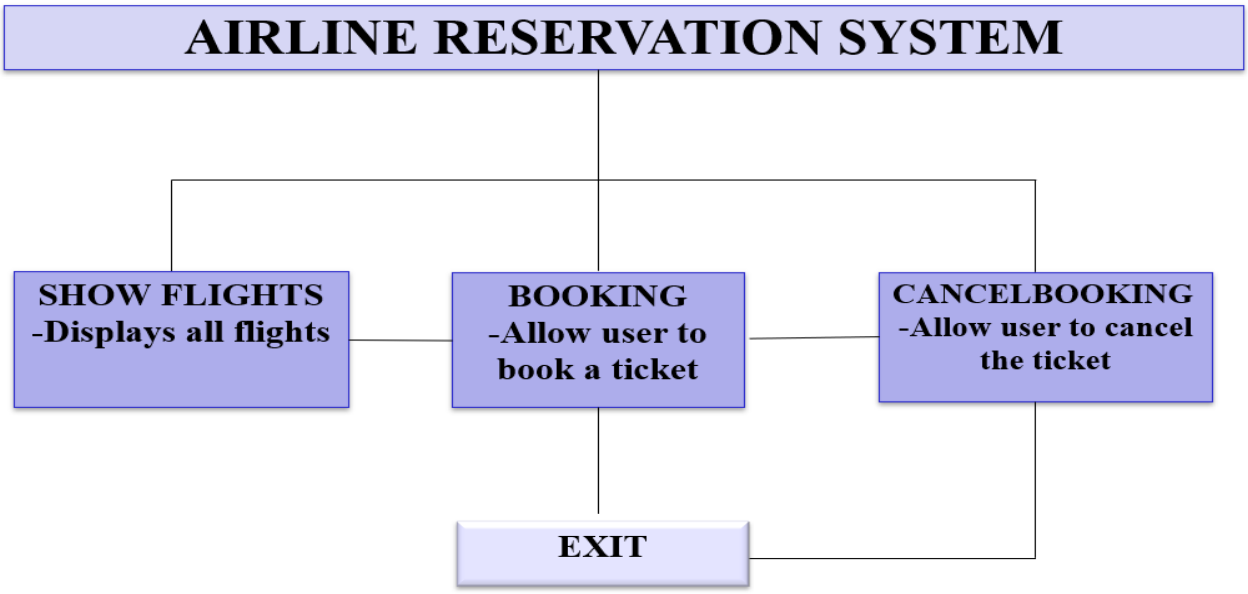
6. Static Methods: The main() method is static, serving as the program's entry point. It ensures that the program initializes and runs without requiring an instance of the class, simplifying execution.

CHAPTER 2
PROJECT METHODOLOGY

2.1 Proposed Work

The proposed system is structured into modules to ensure modularity and ease of maintenance. Each module addresses a specific functionality, ensuring the system is easy to navigate and use. By leveraging AWT for the graphical interface, the system becomes accessible to users with minimal technical knowledge. The event-handling mechanisms provide a seamless experience by responding to user inputs in real-time.

2.2 Block Diagram



CHAPTER 3
MODULE DESCRIPTION

3.1 Home Page

The home page serves as the central interface of the Airline Reservation System. It is the first screen the user interacts with and provides easy navigation to the main functionalities of the system. The home page displays a welcome message and four clearly labeled buttons: Show Flights, Book Ticket, Cancel Booking, and Exit. These buttons allow users to access various modules of the application. Designed with simplicity in mind, the layout ensures that users can navigate the system with minimal effort.

3.2 Show Flights

The Show Flights module is designed to provide users with detailed information about available flights. When this option is selected, a new window opens, displaying a list of flights with details such as flight ID, destination, and ticket price. The information is presented in a user-friendly format using labels, ensuring that users can easily browse the available options. The module includes a Back button, allowing users to return to the home page after reviewing the flight details.

3.3 Book Ticket

The Book Ticket module is the core functionality of the system, enabling users to book a flight ticket. In this module, a form is presented to collect user details, such as the flight ID and passenger name. These inputs are validated to ensure that no fields are left empty, and upon successful submission, a success message is displayed to confirm the booking. If any field is incomplete, an error message prompts the user to enter all required details.

3.4 Cancel Booking

The Cancel Booking module allows users to cancel their previously booked tickets. Users are required to enter a valid ticket ID in a simple input form. Upon submission, the system checks for validity and, if successful, displays a confirmation message to inform the user that the cancellation was successful. If the input field is left blank or the ticket ID is invalid, an error message is shown, guiding the user to provide the correct information. This module is crucial for ensuring flexibility in managing bookings.

3.5 Exit

The Exit module is designed to provide users with a convenient way to close the application. It can be accessed via the Exit button on the home page or by closing the application window. The system uses event listeners to detect the user's intent and safely terminates the application using the System.exit(0) method. This ensures that all resources are properly released, and the application closes gracefully.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The Airline Reservation System developed using Java successfully demonstrates the application of Object-Oriented Programming (OOP), event-driven programming, and AWT for creating an interactive, user-friendly application. The project provides users with functionalities such as booking, canceling, and viewing available flights. By utilizing OOP concepts like encapsulation, inheritance, and polymorphism, the system is modular, making it easier to maintain and extend. Event-driven programming ensures that the system responds dynamically to user inputs, enhancing user interaction. The use of AWT components contributes to the creation of an intuitive interface, while exception handling validates user input, ensuring data integrity. The system is designed with a modular architecture, allowing each functionality to be independently managed and improved.

In conclusion, this project effectively demonstrates how Java's features can be applied to build functional, scalable, and maintainable desktop applications, providing a strong foundation for future enhancements.

4.2 FUTURE SCOPE

The Airline Reservation System has significant potential for further development and enhancement. One of the primary improvements would be the integration of a database management system ,allowing the storage and retrieval of dynamic flight data, bookings, and customer information. This would facilitate real-time updates and improve the system's ability to handle large datasets. Another potential enhancement is the addition of user authentication, enabling customers to create accounts, track their booking history, and manage personal details securely. Expanding the system to mobile platforms through Android or iOS applications would increase accessibility and allow users to book flights on-the-go.

APPENDIX A

(SOURCE CODE)

```
import java.awt.*;
import java.awt. event.*;
public class AirlineReservationSystem extends Frame implements ActionListener {
   private Button btnFlights, btnBooking, btnCancel, btnExit;
   public AirlineReservationSystem() {
      setTitle("Airline Reservation System");
      setSize(400, 300);
      setLayout(new FlowLayout());
      setLocationRelativeTo(null); // Center the window
      setResizable(false);

      Label lblWelcome = new Label("Welcome to Airline Reservation System");
      lblWelcome.setFont(new Font("Arial", Font.BOLD, 16));
      add(lblWelcome);

      btnFlights = new Button("Show Flights");
      btnBooking = new Button("Book Ticket");
      btnCancel = new Button("Cancel Booking");
      btnExit = new Button("Exit");
      add(btnFlights);
      add(btnBooking);
      add(btnCancel);
      add(btnExit);

     btnFlights.addActionListener(this);
     btnBooking.addActionListener(this);
```

```
  btnCancel.addActionListener(this);
      btnExit.addActionListener(this);
      addWindowListener (new WindowAdapter () {
          public void window Closing(WindowEvent e) {
            System.exit(0);
          }
      });
 }
public void actionPerformed(ActionEvent e) {
   if (e.getSource() == btnFlights) {
       showFlights();
} else if (e.getSource() == btnBooking) {
       bookTicket();
} else if (e.getSource() == btnCancel) {
       cancelBooking();
} else if (e.getSource() == btnExit) {
       System.exit(0);
   }
}
private void showFlights() {
   Frame flightsFrame = new Frame("Available Flights");
   flightsFrame.setSize(400, 300);
   flightsFrame.setLayout(new FlowLayout ());
   flightsFrame.setLocationRelativeTo(null);

   Label lblFlights = new Label("Available Flights:");
   flightsFrame.add(lblFlights);
   String[] flights = {
```

```
 "Flight ID: 101 | Destination: New York | Price: $500",
        "Flight ID: 102 | Destination: London | Price: $700",
        "Flight ID: 103 | Destination: Paris | Price: $650"
    };

    for (String flight : flights) {
        flightsFrame.add(new Label(flight));
    }
    Button btnBack = new Button("Back");
    btnBack.addActionListener(e -> flightsFrame. Dispose());
    flightsFrame.add(btnBack);

    flightsFrame.addWindowListener(new WindowAdapter () {
    public void windowClosing(WindowEvent e) {
        flightsFrame.dispose();
    }
});
flightsFrame.setVisible(true);
}
private void bookTicket() {
        Frame bookFrame = new Frame("Book a Ticket");
        bookFrame.setSize(400, 300);
        bookFrame.setLayout(new FlowLayout());
        bookFrame.setLocationRelativeTo(null);

        Label lblFlightId = new Label("Flight ID:");
        TextField txtFlightId = new TextField(20);
        Label lblPassengerName = new Label("Passenger Name:");
```

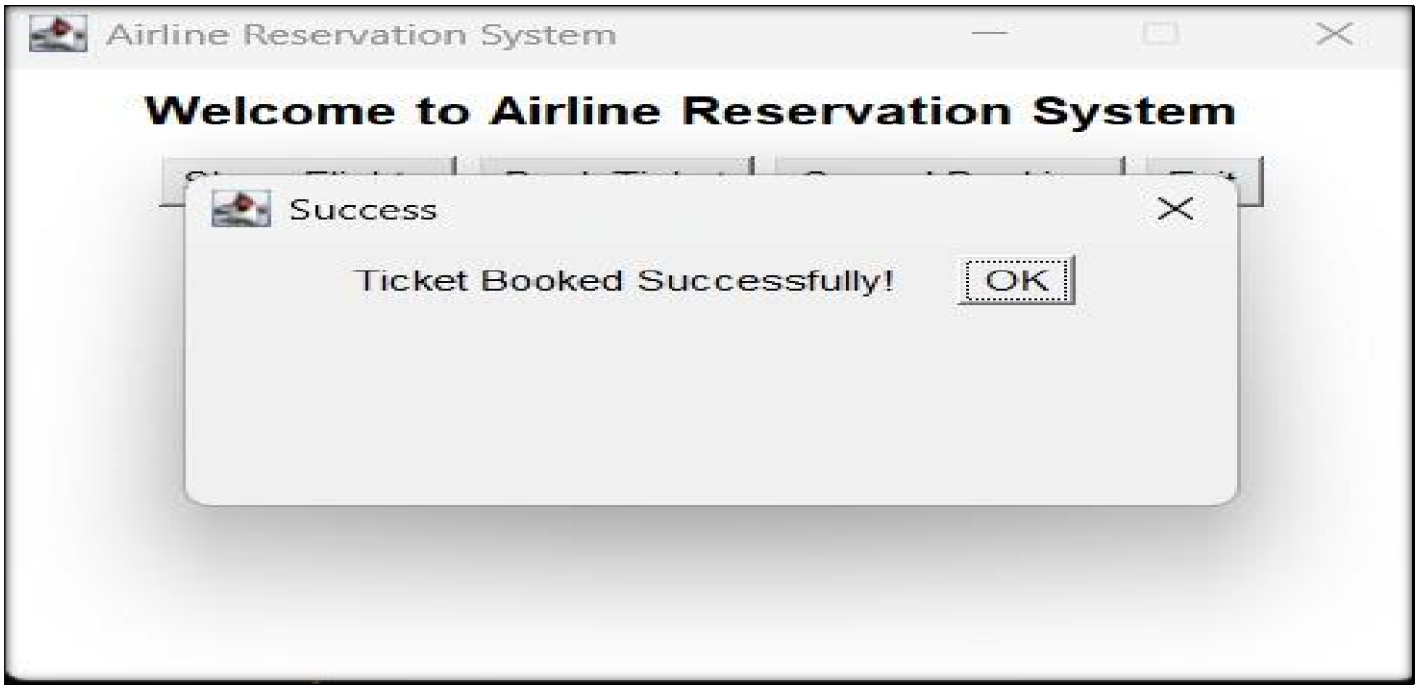
```
        TextField txtPassengerName = new TextField(20);

        Button btnSubmit = new Button("Submit");
        btnSubmit.addActionListener(e -> {
            String flightId = txtFlightId.getText();
            String passengerName = txtPassengerName.getText();
            if (flightId.isEmpty() || passengerName.isEmpty()) {
                showMessage("Error", "Please fill all fields!");
            } else {
                showMessage("Success", "Ticket Booked Successfully!");
                bookFrame.dispose();
            }
        });
    Button btnBack = new Button("Back");
    btnBack.addActionListener(e -> bookFrame.dispose());

    bookFrame.add(lblFlightId);
    bookFrame.add(txtFlightId);
    bookFrame.add(lblPassengerName);
    bookFrame.add(txtPassengerName);
    bookFrame.add(btnSubmit);
    bookFrame.add(btnBack);
    bookFrame.addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent e) {
            bookFrame.dispose();
        }
    });
    bookFrame.setVisible(true);
    }
```

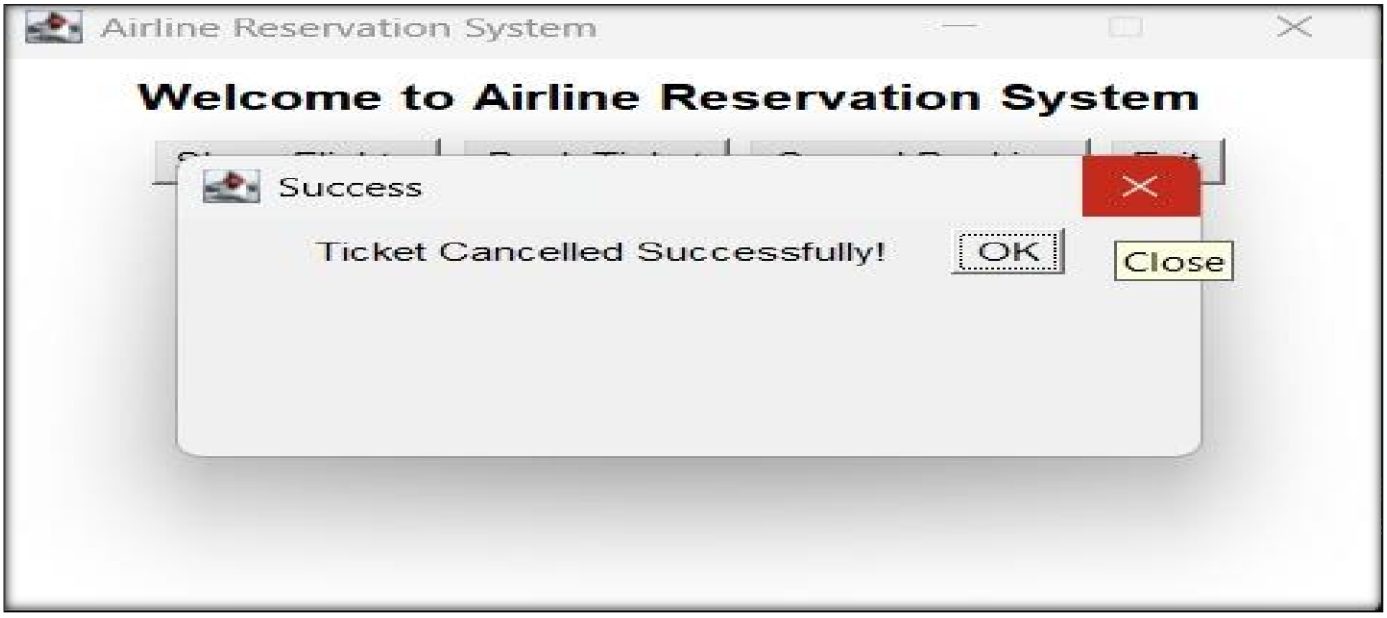
```
private void cancelBooking() {
    Frame cancelFrame = new Frame("Cancel Booking");
    cancelFrame.setSize(400, 200);
    cancelFrame.setLayout(new FlowLayout());
    cancelFrame.setLocationRelativeTo(null);
    Label lblTicketId = new Label("Ticket ID:");
    TextField txtTicketId = new TextField(20);
    Button btnCancel = new Button("Cancel Ticket");
    btnCancel.addActionListener(e -> {
        String ticketId = txtTicketId.getText();
    if (ticketId.isEmpty()) {
        showMessage("Error", "Please enter a valid Ticket ID!");
    } else {
        showMessage("Success", "Ticket Cancelled Successfully!");
        cancelFrame.dispose();
    }
});
Button btnBack = new Button("Back");
btnBack.addActionListener(e -> cancelFrame.dispose());

cancelFrame.add(lblTicketId);
cancelFrame.add(txtTicketId);
cancelFrame.add(btnCancel);
cancelFrame.add(btnBack);

cancelFrame.addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent e) {
       cancelFrame.dispose();
    }
```

```
});
cancelFrame.setVisible(true);
}
private void showMessage(String title, String message) {
    Dialog dialog = new Dialog(this, title, true);
    dialog.setSize(300, 150);
    dialog.setLayout(new FlowLayout());
    dialog.add(new Label(message));
    Button btnOK = new Button("OK");
    btnOK.addActionListener(e -> dialog.dispose());
    dialog.add(btnOK);
    dialog.setLocationRelativeTo(this);
    dialog.setVisible(true);
}

public static void main(String[] args) {
      new AirlineReservationSystem().setVisible(true);
}
}
```

APPENDIX B

(SCREENSHOTS)


Airline Reservation System
Welcome to Airline Reservation System
Success
Ticket Booked Successfully!
OK


Airline Reservation System
Welcome to Airline Reservation System
Success
Ticket Cancelled Successfully!
OK
Close

REFERENCES

1. Books

Herbert Schildt, Java: The Complete Reference, 10th Edition, McGraw-Hill Education, 2018.

This book covers Java programming concepts and object-oriented programming, which are essential for building the airline reservation system.

R. S. Pressman, Software Engineering: A Practitioner's Approach, 9th Edition, McGraw-Hill, 2014.

2. Online Resources

Oracle Java AWT Documentation
https://docs.oracle.com/javase/7/docs/api/java/awt/package-summary.html.

GeeksforGeeks, Java AWT Tutorial
https://www.geeksforgeeks.org/java-awt-abstract-window-toolkit/.

3. Websites

Stack Overflow, “To create a GUI in Java using AWT”
https://stackoverflow.com/questions/13131419/how-to-create-a-gui-in-java-using-awt.