# FLIGHT TICKET RESERVATION SYSTEM

Submitted to the

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

by

K. Manoj kumar (Reg No:192110094)

Under the guidance of

**Dr. A. JEGATHEESAN**

Professor

Institute of Computer Science and Engineering

SAVEETHA SCHOOL OF ENGINEERING CHENNAI – 602 105 TAMILNADU, INDIA

FEBRUARY 2024

# BONAFIDE CERTIFICATE

This is to certify that the project report entitled “The Hospital Management System” submitted by “K.MANOJ KUMAR (192110094)” to Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, is a record of bonafide work carried out by him/her under my guidance. The project fulfills the requirements as per the regulations of this institution and in my appraisal meets the required standards for submission.

Dr. A. Jegatheesan
Professor
Department of Knowledge
Engineering,
Saveetha School of Engineering
SIMATS, Chennai – 602 105

Internal examiner External Examiner

## TABLE OF CONTENTS

# ABSTRACT

The Flight Ticket Reservation System (FTRS) is a technological solution aimed at modernizing and streamlining the operations of airlines and travel agencies. Focusing on improving efficiency and enhancing the customer experience, FTRS integrates various administrative, booking, and financial functions into a unified digital platform. Through automation and digital record-keeping, the system aims to optimize resource utilization, reduce manual workload, and facilitate seamless communication among airline staff and travel agents. The primary objective of the Flight Ticket Reservation System is to establish a technologically advanced and traveler-centric environment that enhances overall operational efficiency and ensures the delivery of high-quality travel services.

FTRS encompasses features such as passenger information management, flight scheduling, and seat availability tracking, providing airline professionals and travel agents with real-time access to critical data. The system promotes data accuracy and security by offering a centralized repository for traveler records that can be easily retrieved and updated as necessary.

# INTRODUCTION

Flight Ticket Reservation System introduces a seamless and efficient solution to the intricate process of booking and managing flight journeys. This innovative system encapsulates the essence of modern aviation by leveraging advanced technology to simplify and enhance the entire flight reservation experience.

From swift booking processes to real-time seat allocation, the Flight Ticket Reservation System caters to the diverse needs of travelers and airlines alike.

Its user-friendly interface ensures accessibility for passengers, while its integration with cutting-edge communication tools fosters effective collaboration between travelers and airline professionals.

As we navigate the skies of contemporary travel, the Flight Ticket Reservation System emerges as a vital and transformative tool, providing a streamlined approach to flight reservations that aligns seamlessly with the demands of today's fast-paced world. Welcome to a future of air travel where convenience, innovation, and traveler satisfaction take center stage.

The integration of automation into routine booking tasks reduces the burden of manual processes, allowing travel professionals to allocate more time to personalized traveler assistance. Simultaneously, the system's analytical capabilities provide insights into booking trends, enabling informed decision-making for travel administrators.

With a user-friendly interface, FTRS improves communication between airline staff and travel agents, supporting better collaboration and coordination in managing flight reservations. The integration of booking and travel data aids in the timely and accurate decision-making process, ultimately improving the travel experience for passengers.

# DESCRIPTION

The presented Flight Ticket Reservation System (FTRS) application is a Java-based solution designed to simulate the functionalities of a digital system tailored for managing diverse aspects of flight reservations. This program adheres to object-oriented programming principles, modeling passengers, their information, and relevant airline functionalities.

Key Features of the FTRS Program:

## 1. Passenger Management:

- Users can seamlessly add passengers to the flight records, capturing crucial information such as name, age, and destination.

## 2. Viewing Passengers:

- The system facilitates the viewing of a comprehensive list of all registered passengers, displaying their names, ages, and travel destinations.

## 3. Searching Passengers by Name:

- Users have the capability to search for a specific passenger by entering their name, providing swift access to individual passenger records.

## 4.Checking Available Seats:

- The program provides real-time information about the number of available seats on flights, aiding in seat management.

**Passenger Information and Seat Management:** The program effectively tracks passenger information and dynamically manages the count of available seats. When a new passenger books a seat, the available seat count decreases, and when a passenger cancels, the available seat count increases.

# ADVANTAGES

## 1. Convenience and Accessibility:

- Passengers can conveniently book flights from the comfort of their homes or on-the-go using online platforms or mobile apps, providing 24/7 accessibility.

## 2.Time Savings:

- Automated reservation systems significantly reduce the time required for booking, eliminating the need for physical visits to travel agencies or airline offices.

## 3.Real-time Information:

- Passengers can access real-time information about flight schedules, seat availability, and fares, enabling informed decision-making during the booking process

## 4.Flexible Booking Options:

- Reservation systems offer flexibility in choosing travel dates, times, and seat preferences, allowing passengers to tailor their travel plans to suit their needs.

## 5.Secure Transactions:

- Online reservation systems provide secure payment gateways, ensuring the confidentiality and safety of passengers' financial information during transactions.

## 6.Efficient Resource Management for Airlines:

- Airlines can effectively manage seat availability, optimize flight schedules, and allocate resources efficiently, leading to improved operational efficiency.

## 7.Enhanced Customer Experience:

- Passengers benefit from a user-friendly interface, easy navigation, and personalized services, contributing to an overall positive travel experience.

## 8.Cost Savings:

- Passengers can take advantage of various promotions, discounts, and early booking incentives, leading to potential cost savings on airfare.

# SYSTEM REQUIREMENTS

1. Smartphone with Android OS version 4.4 (KitKat) or higher
2. Minimum 512 MB of RAM
3. A processor with speeds above 1.2 GHz (any make)
4. 16 MB of storage for the app and extra for the data stored, the size of the app increases as the number of entries are increased
5. Android API version 19 .

6.Minimum of 5MB of storage space for application.

6. Permission to install applications over USB and installation from unknown sources from ‘Developer Options’.

7. For processing capabilities, the application is compatible with any processor with speeds above 1.2 GHz, accommodating a diverse range of smartphone makes and models.

8.Storage requirements include a minimum of 16 MB for the application itself, with additional space needed for storing data. As the number of entries increases, the size of the app dynamically adjusts to accommodate the growing dataset.

9.Compatibility with Android API version 19 ensures that the application leverages the functionalities offered by the specified API, contributing to a robust and feature-rich user experience.

10.To install and run the application, a minimum of 5 MB of storage space is required on the device. Additionally, users are required to grant permission to install applications over USB and enable installation from unknown sources within the 'Developer Options' of the Android settings.

# EXISTING WORK

**Open Source Aviation Projects:**

- Platforms like GitHub or SourceForge host open-source projects related to aviation and travel. Explore repositories that focus on flight ticket reservations or airline management systems. Examples might include FlightGear or OpenFlight.

**Industry-Specific Platforms and Forums:**

- Participate in forums and platforms dedicated to aviation technology or software development. Communities like Stack Overflow or specialized aviation forums may discuss existing work, challenges, and innovations in the field of flight ticket reservations.

**Research Journals and Conferences:**

- Explore research journals, conferences, or publications in aviation informatics. Academic research and industry conferences often showcase existing work, advancements, and case studies related to flight ticket reservation systems.

# PROPOSED WORK

**User-Centric Mobile Applications:**

- Design and develop user-friendly mobile applications that provide a seamless and intuitive interface for travelers to search, book, and manage their flight reservations conveniently on their smartphones.

**Enhanced Personalization Features:**

- Implement advanced personalization features in reservation systems to cater to individual preferences, such as seat preferences, meal options, and notification preferences, enhancing the overall travel experience.

**Integration with Emerging Technologies:**

- Explore the integration of emerging technologies like artificial intelligence (AI) and machine learning (ML) to improve predictive analytics, optimize pricing strategies, and provide personalized recommendations for travelers.

# TECHNOLOGY USED

Web Development Technologies:

- HTML, CSS, and JavaScript: These technologies are fundamental for creating interactive and user-friendly web interfaces for online ticket reservation platforms.

Backend Development Technologies:

- Server Frameworks (e.g., Django, Flask for Python; Express.js for JavaScript): Frameworks facilitate the development of server-side applications, handling routing, request handling, and data processing.

APIs (Application Programming Interfaces):

- Payment Gateways (e.g., Stripe, PayPal): APIs integrate secure payment processing into the reservation system.

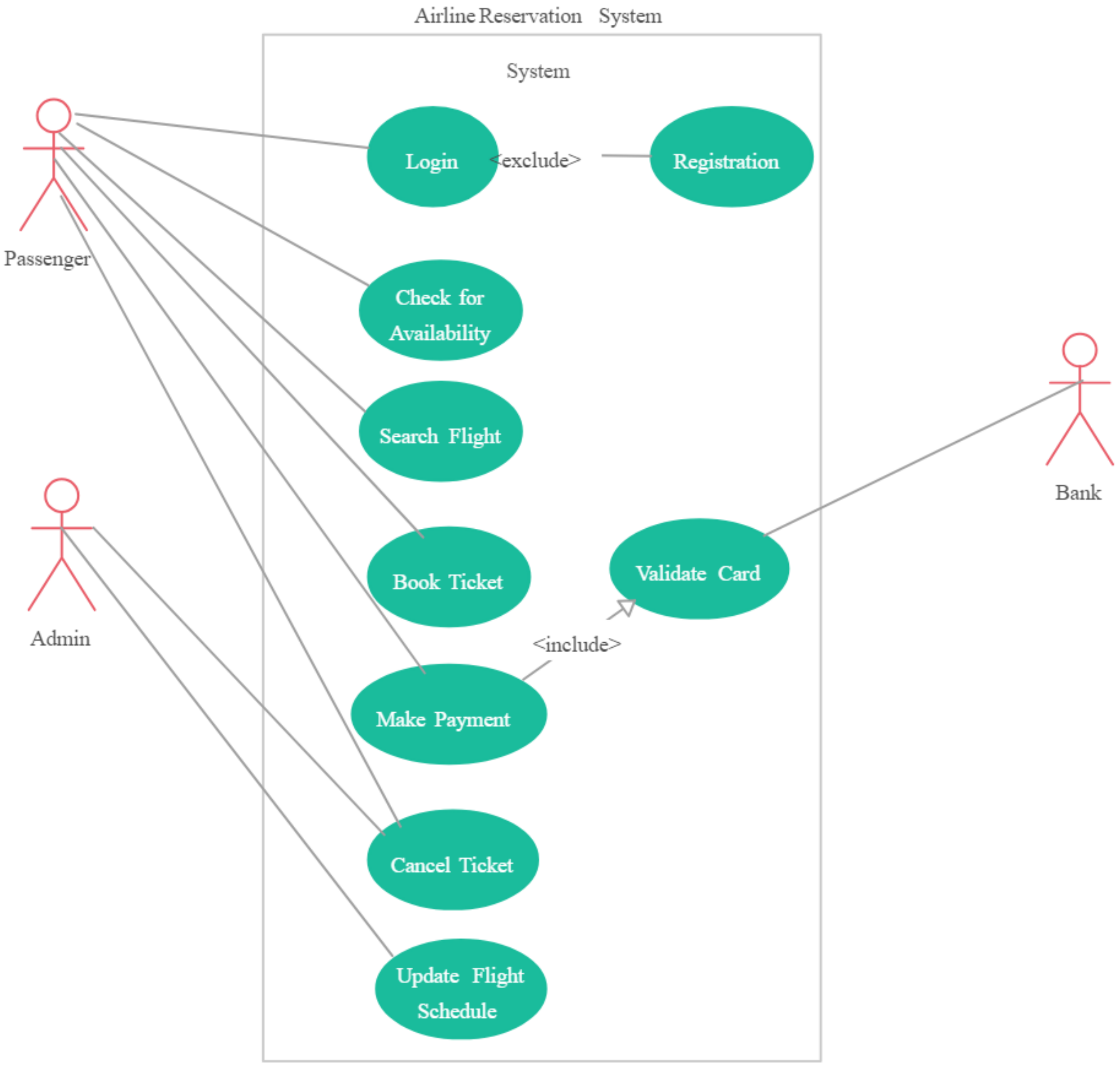
- Flight Information Services (e.g., aviation APIs): APIs provide real-time access to flight schedules, availability, and pricing information.

Mobile App Development Technologies:

- iOS (Swift) and Android (Kotlin/Java): For developing native mobile applications that enable users to make reservations using their smartphones.
- React Native or Flutter: Cross-platform frameworks for developing mobile apps that work on both iOS and Android platforms.

.

# USE CASE DIAGRAM

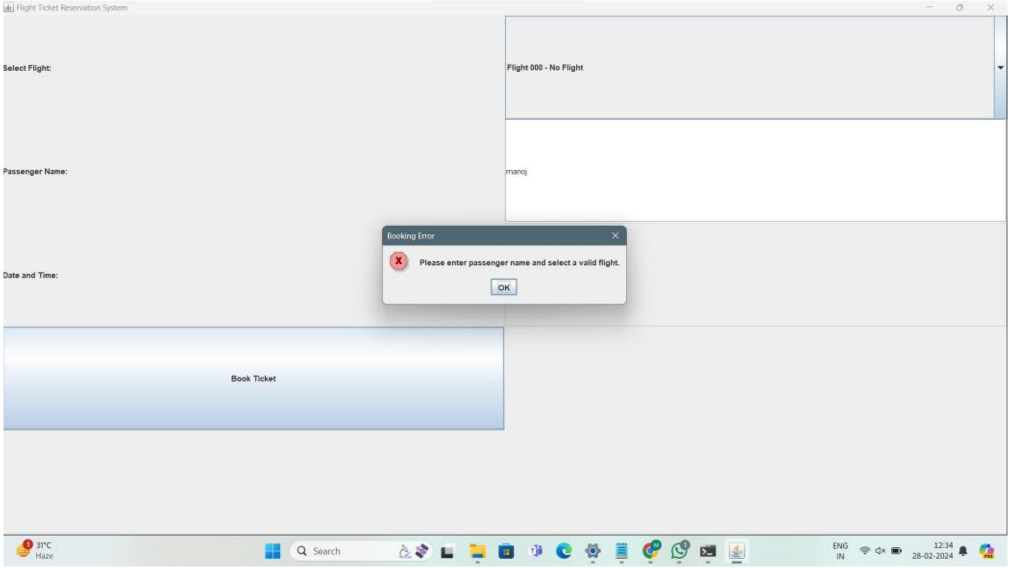


# SOURCE CODE

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class FlightTicketReservationSystem extends JFrame implements ActionListener {
    private JComboBox<String> flightComboBox;
    private JTextField passengerNameField;
    private JTextField dateTimeField;
    private JButton bookTicketButton;
    private Map<String, List<String>> flightPassengerMap;

    public FlightTicketReservationSystem() {
        setTitle("Flight Ticket Reservation System");
        setSize(400, 300);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLayout(new GridLayout(5, 2));
        JLabel flightLabel = new JLabel("Select Flight:");
        String[] flights = {"Flight 000 - No Flight ", "Flight 001 - New York to London", "Flight 002 - Paris to Tokyo", "Flight 003 - Sydney to Dubai"};
        flightComboBox = new JComboBox<>(flights);
        add(flightLabel);
        add(flightComboBox);

        JLabel passengerLabel = new JLabel("Passenger Name:");
        passengerNameField = new JTextField();
        add(passengerLabel);
        add(passengerNameField);

        JLabel dateTimeLabel = new JLabel("Date and Time:");
        dateTimeField = new JTextField();
        dateTimeField.setEditable(false); // Make it read-only
        add(dateTimeLabel);
        add(dateTimeField);

        bookTicketButton = new JButton("Book Ticket");
        bookTicketButton.addActionListener(this);
        add(bookTicketButton);

        flightPassengerMap = new HashMap<>();

        setVisible(true);

        // Set initial date and time
        updateDateTimeField();
    }
    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == bookTicketButton) {
            String selectedFlight = (String) flightComboBox.getSelectedItem();
            String passengerName = passengerNameField.getText();
            String dateTime = dateTimeField.getText(); // You can use this if you want to use the
manually entered date and time

            // Check for missing information
            if (passengerName.isEmpty() || selectedFlight.startsWith("Flight 000 - No Flight")) {
                JOptionPane.showMessageDialog(this, "Please enter passenger name and select a valid
flight.", "Booking Error", JOptionPane.ERROR_MESSAGE);
            } else {
                // Update passenger list for the selected flight
                List<String> passengerList = flightPassengerMap.computeIfAbsent(selectedFlight, k -
> new ArrayList<>());
                passengerList.add(passengerName);

                // Simulate booking process (to be replaced with actual booking logic)
                JOptionPane.showMessageDialog(this, "Ticket booked successfully!\nFlight: " +
selectedFlight +
                        "\nDate and Time: " + dateTime + "\nTotal Passengers in the same flight: " +
passengerList.size());

                // Display the list of passengers for the selected flight
                List<String> passengersForSelectedFlight = getPassengerListForFlight(selectedFlight);
                if (!passengersForSelectedFlight.isEmpty()) {
                    StringBuilder message = new StringBuilder("Passengers in the same flight:\n");
                    for (String passenger : passengersForSelectedFlight) {
                        message.append(passenger).append("\n");
                    }
                    JOptionPane.showMessageDialog(this, message.toString(), "Passengers in the same flight", JOptionPane.INFORMATION_MESSAGE);
                }

                // Clear input fields after booking
                passengerNameField.setText("");
                // Update date and time for the next booking
                updateDateTimeField();
            }
        }
    }

    private void updateDateTimeField() {
        // Get the current date and time
        LocalDateTime now = LocalDateTime.now();
        // Format it as a string
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
        String formattedDateTime = now.format(formatter);
        // Set the formatted date and time to the dateTimeField
        dateTimeField.setText(formattedDateTime);
    }

    public List<String> getPassengerListForFlight(String flight) {
        return flightPassengerMap.getOrDefault(flight, new ArrayList<>());
    }
    public static void main(String[] args) {
        SwingUtilities.invokeLater(FlightTicketReservationSystem::new);
    }
}
```

# SCREENSHOT(OUTPUT)

## FIG1:



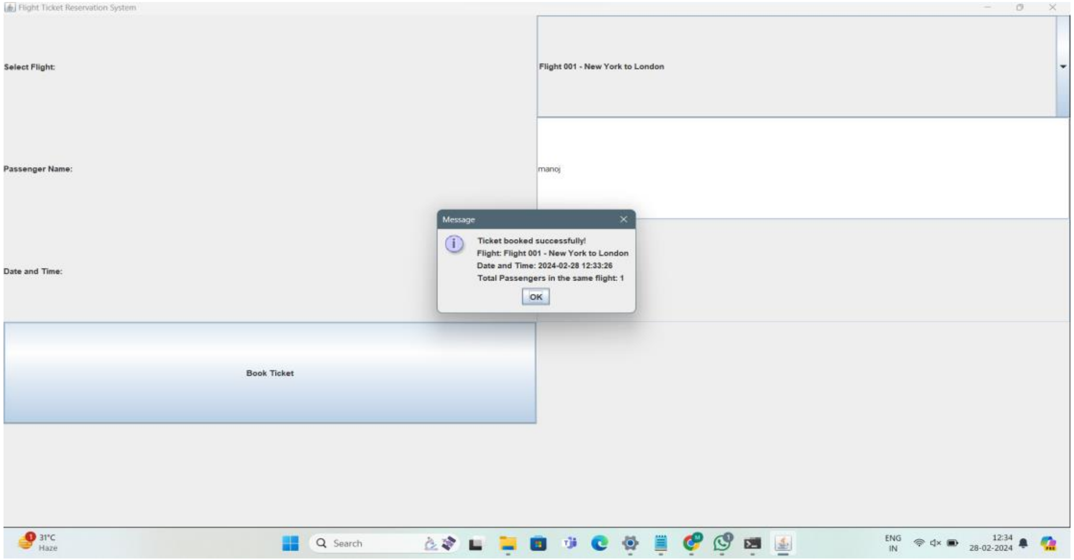
Select Flight: Flight 000 - No Flight

Passenger Name: manoj

Date and Time:

Please enter passenger name and select a valid flight.

## FIG2:



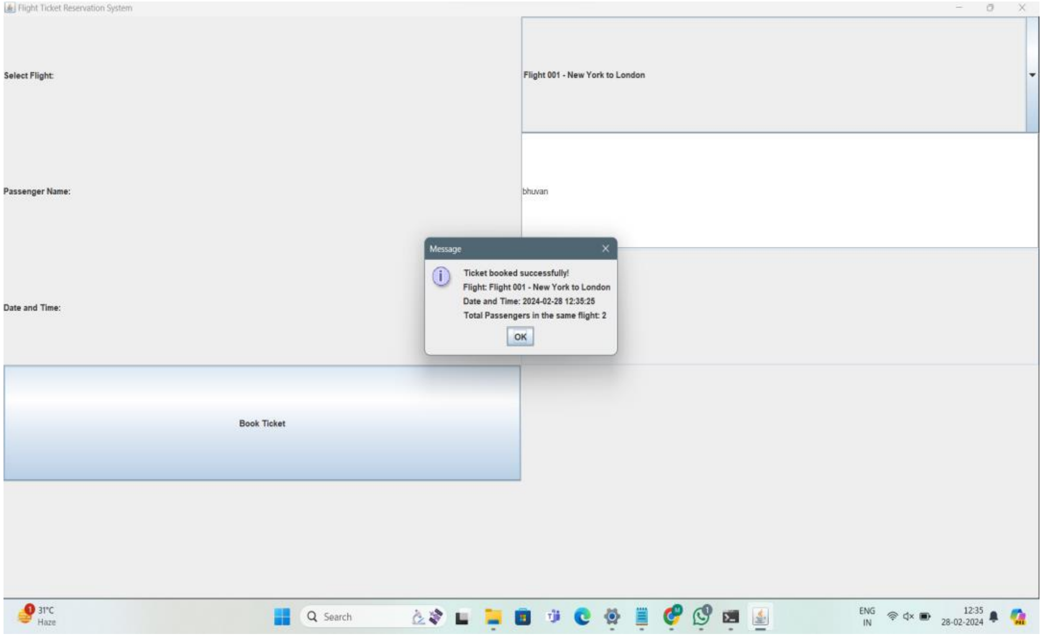
Select Flight: Flight 001 - New York to London

Passenger Name: manoj

Date and Time:

Ticket booked successfully!
Flight: Flight 001 - New York to London
Date and Time: 2024-02-28 12:33:26
Total Passengers in the same flight: 1

## FIG3:



Select Flight: Flight 001 - New York to London

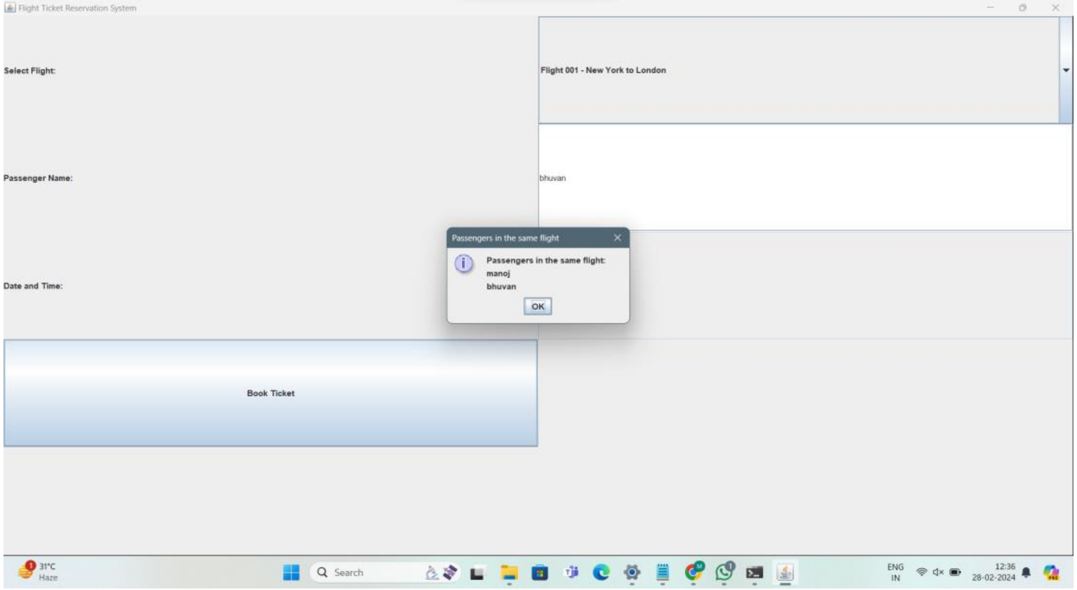
Passenger Name: bhuvan

Date and Time:

Message

Ticket booked successfully!
Flight: Flight 001 - New York to London
Date and Time: 2024-02-28 12:35:25
Total Passengers in the same flight: 2

## FIG4:



Select Flight: Flight 001 - New York to London

Passenger Name: bhuvan

Date and Time:

Passengers in the same flight

Passengers in the same flight:
manoj
bhuvan

# FUTURE ENHANCING

**Web Application:**

- Transition the system from a desktop application to a web-based application using technologies such as Spring Boot, React, or Angular. This allows users to access the reservation system from any device with a web browser.

**Mobile Application:**

- Develop a mobile application for iOS and Android platforms using technologies like React Native or Flutter. This would provide users with the convenience of booking tickets on the go.

**Cloud Integration:**

- Utilize cloud services such as AWS, Azure, or Google Cloud for scalability, reliability, and ease of maintenance. This can help handle varying workloads and improve system availability.

**Microservices Architecture:**

- Implement a microservices architecture for better scalability, maintainability, and flexibility. Each service can handle a specific functionality, such as flight management, booking, and notifications.

**Blockchain for Security:**

- Explore blockchain technology for enhancing security and transparency in transactions. This can be particularly useful for securely managing passenger information, ticket transactions, and loyalty programs.

**AI and Chatbots:**

- Implement artificial intelligence (AI) and chatbots for customer support. Chatbots can assist users in finding flights, answer common queries, and provide real-time support.

**Biometric Authentication:**

- Integrate biometric authentication methods such as fingerprint or facial recognition for secure

# CONCLUSION

The Flight Ticket Reservation System code lays the groundwork for efficiently managing flight reservations and related operations. The program provides a user-friendly interface for tasks such as booking, viewing, and updating flight information, along with features like real-time seat availability tracking and personalized flight suggestions based on destinations.

While the current implementation serves as a starting point, there are several avenues for future improvement. Proposed enhancements include integrating a database for persistent data storage, introducing features like dynamic pricing strategies and loyalty programs, and considering individual passenger preferences in flight suggestions.

The system's simplicity makes it suitable for educational purposes and as a foundation for more advanced airline management solutions. Future developments can be customized to meet the specific requirements and scale of airlines and travel agencies, ensuring streamlined travel experiences, data integrity, and compliance with industry standards. As the aviation landscape evolves, incorporating additional modules and features will contribute to a more comprehensive and effective Flight Ticket Reservation System.

# REFERENCE

1) https://www.tutorialspoint.com/android/android_online_quiz.html

2) https://appsgeyser.com/blog/create-quiz-app-for-android/

3) https://www.geeksforgeeks.org/how-to-create-a-quiz-app-in-android/

4://projectworlds.in/android-projects-with-source-code/androidstudio-quiz-app-source-code-free-download/

5) https://drfone.wondershare.com/screen-recorder/quiz-app-for-androidphone.html

6)https://mobilerecorder24.com/ilmora.wondershare.com/quiz-editor/best-quiz attend-apps-android.html

7) https://www.geeksforgeeks.org/quiz-app-in-android-with-example/