AIRPORT MANAGEMENT SYSTEM

Java project

INTRODUCTION

- Airports are complex hubs of activity, where efficiency and organization are paramount to ensure smooth operations and a positive experience for passengers. Managing various facets of an airport, such as flight scheduling, passenger services, baggage handling, and security checks, can be challenging when done manually. To address these challenges, we have developed an Airport Management System using Java. This system aims to automate and streamline airport operations, thereby enhancing efficiency and improving the overall passenger experience.

OBJECTIVE

- The primary objective of our Airport Management System is to create a robust, scalable, and user-friendly platform that can handle the multifaceted needs of an airport. The system is designed to:
-
- Automate flight scheduling and management to minimize delays and optimize resource allocation.
- Provide real-time updates and information to passengers, improving their travel experience.

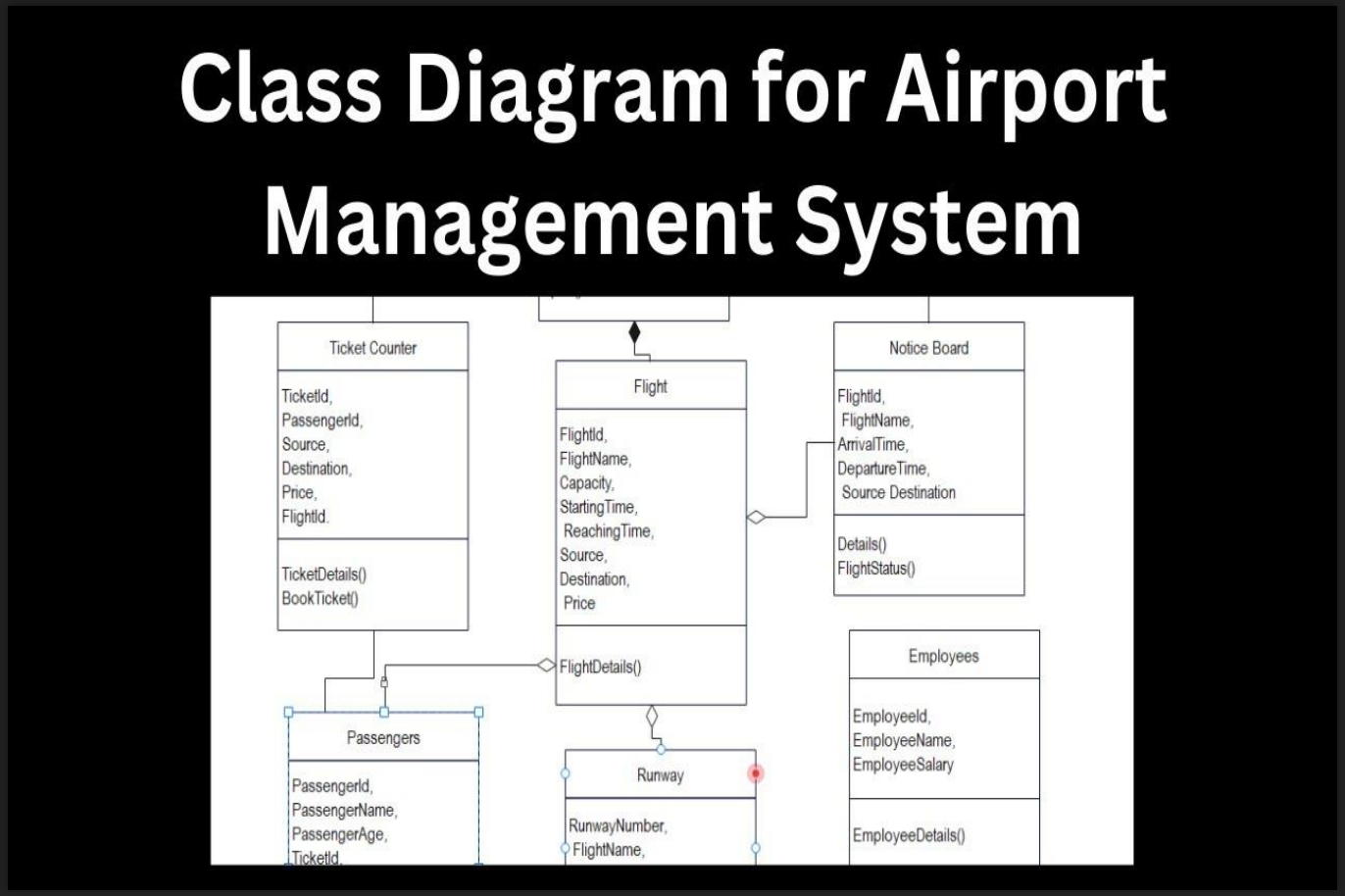
FEATURES

- Flight Scheduling and Management
- Automated Flight Scheduling: Generate and manage flight schedules based on factors like aircraft availability, crew schedules, and weather conditions.
- Real-time Updates: Provide real-time information on flight arrivals, departures, delays, and cancellations.
- Conflict Resolution: Detect and resolve scheduling conflicts to ensure optimal resource utilization.

SECURITY

- Security Management
- Integrated Security Checks: Implement integrated security protocols for passenger screening, baggage scanning, and access control.
- Incident Reporting: Provide tools for reporting and managing security incidents and breaches

CLASS DIAGRAM



EXAMPLE CODE

```
Import java.util.ArrayList;
import java.util.List;

public class AirportManagementSystem {
    private List<Flight> flights;
    private List<Booking> bookings;

    public AirportManagementSystem() {
        flights = new ArrayList<>();
        bookings = new ArrayList<>();
    }

    public void addFlight(String flightNumber, String destination, String departureTime, int capacity) {
        flights.add(new Flight(flightNumber, destination, departureTime, capacity));
    }
```

```
Public Flight getFlight(String flightNumber) {
        for (Flight flight : flights) {
            if (flight.getFlightNumber().equals(flightNumber)) {
                return flight;
            }
        }
        return null;
    }
```

```
Public boolean bookFlight(String flightNumber, String passengerName, String
passportNumber) {
    Flight flight = getFlight(flightNumber);
    if (flight != null && flight.bookSeat()) {
        Passenger passenger = new Passenger(passengerName, passportNumber);
        Booking booking = new Booking(flight, passenger);
        bookings.add(booking);
        return true;
    }
```

```
Return false;
    }

    public List<Booking> getBookings() {
        return bookings;
    }

    public static void main(String[] args) {
        AirportManagementSystem system = new AirportManagementSystem();
```

```
    system.addFlight("FL123", "New York", "10:00 AM", 100);
        system.addFlight("FL456", "Los Angeles", "12:00 PM", 200);

        system.bookFlight("FL123", "John Doe", "P12345678");
        system.bookFlight("FL123", "Jane Smith", "P87654321");

        for (Booking booking : system.getBookings()) {
            System.out.println(booking);
        }
    }
}
```

CONCLUSION

- The Airport Management System project demonstrates the power and flexibility of Java in building real-world applications. By focusing on modular design and core functionalities, the project serves as a strong starting point for developing a comprehensive airport management solution. With future enhancements and continuous improvements, this system has the potential to significantly streamline airport operations, improve passenger experiences, and ensure efficient management of airport resources.

TEAM MEMBERS

PREETHI

SHARVESHINI

DHUSHITHA

DEEKSHITHA

POOJAA

SHIBITHA

THANK YOU