



# **Daffodil** *International* **University**

## **Project Report**

**Topic Name:** Airline Management System

**Course Code:** CSE222

**Course Name:** Object Oriented Programming II Lab

### **Submitted To:**

Ms. Umme Ayman

**Lecturer**

Department of CSE

Daffodil International University

### **Submitted By:**

Team Name: Hash Hackers

Members Name: Horaira, Saad, Imthehan, Shofiur

Members ID: 1514, 1507, 1510, 1546

SEC: 63-J2

Department: Computer Science and Engineering

Daffodil International University

**Date:** 10/12/2024

# Project Title

Airline Management System: Flight and Booking Management

---

## Project Overview

The Airline Management System is a command-line application designed to streamline the management of airline operations. Using Python's object-oriented programming (OOP) features, the system integrates **Flight Management** and **Booking Management** functionalities into a unified solution. The system supports operations such as adding flights, updating ticket prices, booking and canceling tickets, and generating summary reports.

---

## Project Objectives

1. Provide a system for adding and managing flights with attributes like flight number, destination, seat availability, and ticket price.
  2. Facilitate booking and cancellation of tickets while dynamically updating seat availability.
  3. Enable administrators to search for flights by destination and view booking details.
  4. Generate comprehensive reports summarizing flight and booking data.
  5. Implement a user-friendly menu for seamless interaction.
- 

## System Features

### **Flight Management**

1. **Add Flight:**
  - Add new flights with details such as flight number, destination, total seats, and ticket price.
  - Automatically validate input for duplicate flights or invalid data types.
2. **Update Flight Price:**
  - Modify ticket prices for existing flights.
3. **View Flights:**
  - Display all available flights with details including destination, total seats, available seats, and ticket price.
4. **Search Flights:**
  - Search for flights based on destination, with results showing seat availability and ticket price.

---

## Booking Management

1. **Book Ticket:**
    - Book tickets for flights, automatically reducing the available seat count.
    - Provide a unique ticket ID for each booking.
  2. **Cancel Booking:**
    - Cancel tickets using the ticket ID, restoring seat availability.
  3. **View Booking:**
    - Display booking details for a specific ticket ID, including passenger name, flight details, and ticket price.
  4. **Generate Report:**
    - Provide a summary report of all flights and bookings, showing the current state of the system.
- 

## Technical Implementation

### Technologies Used

- **Programming Language:** Python
- **Paradigm:** Object-Oriented Programming (OOP)

### System Architecture

1. **Class Structure:**
  - **FlightManagement:**
    - Handles all flight-related operations such as adding, updating, and viewing flights.
  - **BookingManagement:**
    - Manages ticket booking, cancellation, and viewing booking details.
  - **AirlineManagementSystem:**
    - Combines functionalities of Flight Management and Booking Management using multiple inheritance.
  - **Menu Interface:**
    - Provides an interactive command-line menu for users to navigate and execute operations.
2. **Key Methods:**
  - `add_flight`: Adds new flights with validation.
  - `update_price`: Updates ticket prices for specific flights.
  - `book_ticket`: Books tickets and updates seat availability.
  - `cancel_booking`: Cancels tickets and restores seat availability.
  - `view_flights`: Displays flight details.

- `generate_report`: Summarizes all flight and booking data.

---

## Results

### Functional Outputs

- Flight Management:**
  - Successfully added flights and updated ticket prices.
  - Displayed flights and their details effectively.
- Booking Management:**
  - Successfully booked tickets, assigned ticket IDs, and adjusted seat availability.
  - Provided clear and accessible booking details.
- Error Handling:**
  - Prevented duplicate flight entries.
  - Handled invalid ticket IDs and flight numbers with user-friendly error messages.
- User Interaction:**
  - Provided a clear and intuitive menu-driven interface.

---

## Testing

### Test Scenarios and Results

Test Case	Expected Outcome	Actual Outcome	Status
Add valid flight	Flight added successfully	Flight added	Passed
Add duplicate flight	Error message displayed	Error message displayed	Passed
Update ticket price with valid data	Ticket price updated successfully	Price updated	Passed
Book ticket with available seats	Ticket booked successfully	Ticket booked	Passed
Book ticket with no available seats	Error message displayed	Error message displayed	Passed
Cancel valid ticket	Booking canceled, seats restored	Booking canceled	Passed
Cancel invalid ticket	Error message displayed	Error message displayed	Passed
View flights and bookings	Data displayed in readable format	Data displayed correctly	Passed

---

## Challenges and Solutions

1. **Challenge:** Preventing duplicate flight entries.
    - **Solution:** Implemented a check for existing flight numbers before adding a new flight.
  2. **Challenge:** Managing seat availability dynamically.
    - **Solution:** Adjusted seat counts in `book_ticket` and `cancel_booking` methods.
  3. **Challenge:** Providing user-friendly error messages.
    - **Solution:** Added exception handling (`try-except`) for invalid inputs and unexpected errors.
- 

## Conclusion

The Airline Management System achieved its objectives by integrating flight and booking management operations into a cohesive solution. It demonstrates the effective use of Python's OOP principles, providing a scalable and extensible foundation for future enhancements.

---

## Future Enhancements

1. **Data Persistence:**
    - Store flight and booking data in files or a database for continuity across sessions.
  2. **Enhanced Search:**
    - Add search filters such as price range and flight availability.
  3. **Web Interface:**
    - Develop a web-based UI for better accessibility.
  4. **Role-Based Access:**
    - Introduce admin and user roles with permissions.
  5. **Reports:**
    - Generate printable reports for flight and booking data.
- 

## Acknowledgments

This project leverages Python's simplicity and OOP features to provide a functional airline management system. Special thanks to the open-source Python community for the tools and libraries enabling this project.