07-11-2024

Faculty: Mohammad Shifat-E-Rabbi

Report on Project of CSE 215

Section: 7

Group: 9

Shaira Tahsin Era

Noshin Nawal

Topic: Online Flight Booking System.

1. Abstract

We have made a project that will present the flight booking application that will allow users to search for flights, book tickets and manage reservations. The application utilizes Java swing for the graphical user interface (GUI) and employee's multithreading to handle concurrent booking request efficiently. We have used method, inheritance, loops and array also. We have tried to make it user friendly.

Keywords

Java, Flight Booking System, Flight Booking App, Multithreading, File I/O, Concurrent Requests, Reservation Management, System Designing,

2. Introduction

Menu page will be starting with search flight, see booking and log out options. This project aims to create a flight booking application that facilitates users in searching, booking and managing their reservations seamlessly. The application is designed with a focus on performance, utilising multithreading to handle multiple user requests concurrently. The flight booking industry requires systems that can manage numerous users and provide real-time information about available flights, prices, and seat availability. Traditional booking systems often face challenges such as poor user experience during peak times, long wait times, and errors due to simultaneous access to the same data. This project addresses these issues by implementing a multithreaded flight booking system in Java that can efficiently handle multiple user requests and store flight data using file I/O operations.

The system is designed to:

- Allow users to search for available flights.
- Enable users to book and manage flight reservations.
- Handle multiple concurrent booking requests efficiently using multithreading.
- Store flight details in a file for persistent storage.

3. System Design and Architecture

The system follows a modular architecture with distinct components handling specific functionalities. The system is built around three core modules:

- **Flight Management Module**: Handles flight data such as flight numbers, seat availability, and pricing.
- **Booking Module**: Manages user reservations and ensures that the flight data is updated when a booking is made.
- **Multithreading and Concurrent Handling**: Manages multiple users requesting flight bookings simultaneously, ensuring that the system can handle high loads efficiently.

Flow of the Application

- The user interacts with the interface to search for flights based on the departure and destination cities.
- Once flights are found, the user can select a flight and make a reservation.
- The system then updates the available seat count and writes the booking details to a file.

Use of Multithreading

We used to manage multiple threads ensuring that the booking request do not block by the user interface. Each booking operation is managed in a separate thread. It will go to confirm booking where users will put details individually for number of people they choose.

- The system uses threads to handle multiple user requests simultaneously.
- Each user request is handled by a separate thread, ensuring that booking requests do not block each other.
- Synchronization techniques are used to ensure that updates to the available seats are thread safe.

Technologies and Tools Used

- **Java Programming Language**: The system is developed in Java due to its robustness, platform independence, and support for multithreading.
- **Multithreading**: Java's built-in multithreading capabilities are utilized to handle multiple concurrent booking requests.
- **File I/O**: The flight and booking details are stored in text files, making the system lightweight and easy to implement.

Implementation Details: Class definition

✪ Flight :

- public class Flight {
- private String flightNumber
- private String origin;
- private String destination;
- private String date;
- private String time;
- private boolean[][] seats = new boolean[5][20];
- public Flight(String flightNumber, String origin, String destination, String date, String time) {
- this.flightNumber = flightNumber;
- this.origin = origin;
- this.destination = destination;
- this.date = date;
- this.time = time;}

Output will be like Seat Layout:

[]
[]
[]
[]
[]

With the timing, name and every detail of the flight.

✪ User:

- public User (String firstName, String lastName) {
- this.firstName = firstName;
- this.lastName = lastName;}

Output will be like

First Name: Tahsin

Last Name: Era

Full Name: Tahsin Era.

✪ Reservation:

- ser user = new User ("Tahsin", "Era"); Flight flight = new Flight("AA123", "NYC", "LA", "2024-12-01", "10:00"); Reservation reservation = new Reservation("R123", user, flight, 3); Passenger passenger1 = new Passenger("Jane", "Doe"); Passenger passenger2 = new Passenger("Jim", "Beam"); reservation.addPassenger(passenger1); reservation.addPassenger(passenger2); reservation.printPassengerDetails();

1. Users can search for available flights based on departure and arrival locations, date and time.
2. Users can book tickets and received confirmation.
3. Users can view, modify and cancel their reservations.

✪ Passanger:

- public String toString() {
- return "Name: " + name + ", Age: " + age + ", Gender: " + gender + ", Passport: " + passportNumber;}

So, in this class our system will collect the data of the user.

✪ **FlightBookingSystem:** We have used this class with the use of exception handling, so that if any user gives any input which is incorrect, this class will throw the problem and catch will accept the exception and output will show as UnsupportedInformation.

✪ **FlightBookingApp:** This is the main class of our project. We have inherited all classes in this main class and used multithreading, Exception handling, input of Flight time, date, place, name, info of user and everything.

When any user tries to login, the system will ask for name, mail, password. Then after creating an account. After logging in he can exit, or he can search for a flight. There will be 3 options, then after choosing a place 3 options for timing will be shown. After completing all the process, he can log out or can reserve the seat for

himself, if there is no available seat, it will be shown too. And if he reserves any seat, it will be shown into his account. Then he can log out.

4.Data Storage

Flight data and booking information are stored in text files. There will be 5 X 20 array of seats, which will be next to one another. User will open the program, login and find then will be able to book. Even it will show if whether seats are available or not.

5.Conclusion

The flight booking applications successfully demonstrates the ability to handle multiple concurrent users because providing an easy-to-use interface. Future enhancements could include integration we the database for more robust data management and the addition of online payment functionality. This will save time of user and in near future we will update it more. The Online Flight Booking System provides a robust solution for booking flights efficiently, even under high concurrent loads. The use of multithreading and file-based storage ensures scalability and reliability. Future improvements could include:

- Integration with an actual database for flight data storage.
- A web-based interface for easier user interaction.