**Project Report: AI Flight Ticket Reservation System**

**1. Project Overview**

The **AI Flight Ticket Reservation System** is a Python-based application designed to assist users in finding, booking, and recommending flight tickets. It integrates interactive user registration and login functionalities, an AI-driven chatbot for flight inquiries, and personalized flight recommendations using **cosine similarity**.

**2. Objectives**

* **Primary Objective**: Simplify the process of finding and booking flights using AI.
* **Secondary Objective**: Enhance user experience through personalization based on user preferences.

**3. Features**

**3.1 User Authentication**

* **User Registration**: New users can create an account with a unique username and password.
* **Login System**: Registered users can securely log in using their credentials.

**3.2 AI Flight Chat Assistant**

* Users can query the system using natural language commands like:
  + *"Find flights from New York to Los Angeles"*
* Filters flights based on the source and destination cities provided.
* Provides real-time flight data with details such as price, availability, and departure time.

**3.3 Flight Recommendations**

* Recommends top 3 flights based on **user preferences** using a similarity score.
* Employs **cosine similarity** to align flight offerings with the user’s city ratings.

**3.4 Flight Booking System**

* Allows users to book flights using a unique FlightID.
* Automatically updates the number of available seats after booking.
* Ensures error handling for invalid IDs and fully booked flights.

**4. Technical Implementation**

**4.1 Tools and Libraries Used**

* **Python**: Core programming language.
* **Pandas**: For managing tabular flight and user data.
* **Scikit-learn**: For implementing cosine similarity in recommendations.
* **Datetime**: For handling flight date and time information.

**4.2 Code Structure**

**a) Data Management**

* Flight information is stored in a **Pandas DataFrame**, containing details such as:
  + FlightID, Source, Destination, Price, SeatsAvailable, and Date.
* User preferences are stored in another DataFrame, with ratings for cities.

**b) User Authentication**

* User credentials are stored in a Python dictionary:
  + users = {}
* Registration ensures unique usernames. Login validates credentials.

**c) Recommendation System**

* **Cosine Similarity** compares user preferences with flight offerings:
  + One-hot encoding is applied to flight data for comparison.
  + The top 3 flights with the highest similarity scores are recommended.

**d) Interactive Menu**

The main program loop provides the following options:

1. Registering a new user.
2. Logging in.
3. Using the chatbot.
4. Viewing recommendations.
5. Booking flights.

**5. Data Examples**

**5.1 Sample Flight Data**

| **FlightID** | **Source** | **Destination** | **Price** | **SeatsAvailable** | **Date** |
| --- | --- | --- | --- | --- | --- |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **1** | New York | Los Angeles | 300 | 20 | 2024-12-15 | 10:30:00 |
| **2** | Lahore | Karachi | 500 | 30 | 2024-12-19 | 20:00:00 |

|  |
| --- |
| **5.2 Sample User Preferences** |
| | **UserID** | **New York** | **Los Angeles** | **Chicago** | **Karachi** | **Lahore** | **Islamabad** |  | | --- | --- | --- | --- | --- | --- | --- | --- | | 1 | 5 | 3 | 2 | 4 | 3 | 2 |  | |

**6. User Interaction Flow**

**Step 1: Registration and Login**

* **New User**: Registers using a unique username and password.
* **Returning User**: Logs in with existing credentials.

**Step 2: Selecting Features**

* **Chatbot**: Inquire about specific flights.
* **Recommendations**: Get personalized flight suggestions.
* **Booking**: Reserve a seat on a chosen flight.

**Step 3: Booking Confirmation**

* Updates seat availability and confirms the booking with details.

**7. Testing and Validation**

**7.1 Test Cases**

| **Feature** | **Test Case** | **Expected Outcome** |
| --- | --- | --- |
| Registration | Register with a new username. | User is successfully registered. |
| Registration | Register with an existing username. | Error: *"Username already taken."* |
| Login | Log in with valid credentials. | Success message: *"Login successful!"* |
| Login | Log in with invalid credentials. | Error: *"Invalid username or password."* |
| Flight Chatbot | Query flights: *"Find flights from Karachi to Lahore."* | Displays flight details for the route. |
| Flight Chatbot | Query unavailable flights: *"Find flights from Lahore to Dubai."* | Message: *"Sorry, no flights found."* |
| Recommendations | View recommendations for user preferences. | Displays top 3 recommended flights. |
| Booking | Book a flight with a valid FlightID. | Updates seat availability and confirms the booking. |
| Booking | Book a flight with an invalid FlightID. | Error: *"Invalid FlightID!"* |

**7.2 Edge Cases**

* Booking a flight with no available seats.
* Querying flights without specifying source and destination.

**8. Challenges Faced**

1. **Input Parsing**: Handling various user input formats for the chatbot.
   * **Solution**: Normalize and validate user inputs.
2. **Recommendation Alignment**: Ensuring matrix dimensions match during cosine similarity computation.
   * **Solution**: Reindex columns and fill missing values with zeros.

**9. Conclusion**

The AI Flight Ticket Reservation System successfully integrates authentication, a chatbot, and AI-based recommendations to improve user experience. By leveraging cosine similarity and user preferences, it ensures personalized service.

**10. Future Enhancements**

* **Persistent Data Storage**: Replace in-memory structures with a database for user and flight data.
* **Dynamic Pricing**: Incorporate real-time pricing updates based on demand.
* **Multi-user Support**: Expand support for simultaneous users in a networked environment.

**11. References**

* Pandas Documentation
* [Scikit-learn Documentation](https://scikit-learn.org/)
* [Python Datetime Module](https://docs.python.org/3/library/datetime.html)