# Flight Booking System Documentation

## Modes of Operation

### 1. User Mode

This mode allows a regular user to interact with the system to search for flights, book tickets, and manage their bookings.

#### Features:

- Register: The user creates an account by providing their details such as username, password, and email.

- Log In: The user logs in to access the flight booking system using valid credentials.

- Search Flights: Users can search for available flights by providing criteria such as departure and arrival locations, dates, and times.

- Book Flight: After selecting a flight, users can enter details and confirm their booking.

- Cancel Booking: Users can cancel their existing bookings if necessary.

### 2. Admin Mode

This mode is designed for administrators who manage the flight operations and oversee the system's backend.

#### Features:

- Log In: Admins authenticate themselves using their admin credentials to access the system.

- Create Flights: Admins can add new flights, specifying details such as origin, destination, time, price, and availability.

- Edit Flights: Admins can modify existing flight details, such as schedule changes, prices, or availability.

- Manage Bookings: Admins can view and manage user bookings, including cancellations or changes to reservations.

- View Reports: Admins can generate and view reports summarizing system activity, bookings, or other analytical data.

### 3. Flight Search Mode

This mode handles the retrieval of available flights based on user-provided search criteria. It is a shared functionality between users and the admin interface.

#### Process:

1. User or admin provides search criteria (origin, destination, date, etc.).

2. The system retrieves all flights that match the criteria.

3. Results are displayed for further action, such as booking or editing.

### 4. Flight Booking Mode

This mode allows users to book seats on available flights.

#### Process:

1. User selects a flight from the search results.

2. User enters booking details, including passenger information.

3. The system processes the booking:  
 - Updates the flight's available seats.  
 - Saves the booking details to the database.

4. Confirmation of booking is displayed to the user.

### 5. Flight Management Mode (Admin-Specific)

This mode focuses on creating, updating, and managing flight details.

#### Features:

- Admins can add new flights with complete details (e.g., origin, destination, schedule, seat capacity, price).

- Modify existing flights based on operational changes.

- Remove flights when no longer available.

### 6. Reporting Mode (Admin-Specific)

This mode provides administrators with insights into system usage, booking trends, and operational metrics.

#### Features:

- Generate detailed reports that summarize:  
 - Booking statistics (e.g., total bookings, cancellations).  
 - Revenue data based on bookings.  
 - Flight occupancy rates.

- Download or view these reports within the system.

## Mode Summary Table

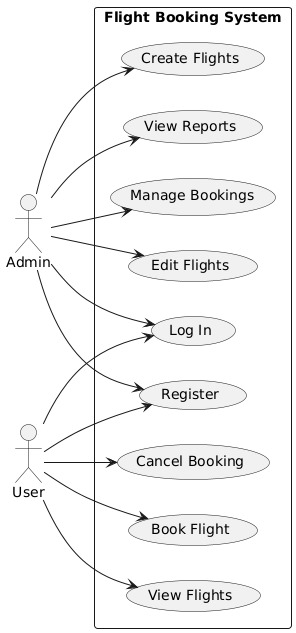
|  |  |  |
| --- | --- | --- |
| Mode | Actor | Key Features |
| User Mode | User | Register, Log In, Search Flights, Book, Cancel Booking |
| Admin Mode | Admin | Log In, Create Flights, Edit Flights, Manage Bookings, View Reports |
| Flight Search | User/Admin | Search for available flights based on criteria |
| Flight Booking | User | Select and book flights, confirm payment and details |
| Flight Management | Admin | Create, update, and manage flight details |
| Reporting Mode | Admin | Generate and view analytical reports |

**Diagram 1: Use Case Diagram**

This diagram represents the high-level functionalities of the Flight Booking System and its interaction with two primary actors: Admin and User.

**Key Details:**

* **Admin Use Cases:**
  + *Create Flights:* Admins can add new flights to the system.
  + *Edit Flights:* Admins can update existing flight details.
  + *Manage Bookings:* Admins can oversee and modify user bookings.
  + *View Reports:* Admins can generate and analyze system reports.
  + *Log In:* Admins must log in to access administrative functionalities.
* **User Use Cases:**
  + *Register:* Users can create an account to use the system.
  + *Log In:* Users log in to access booking and other functionalities.
  + *View Flights:* Users can browse available flight options.
  + *Book Flight:* Users can reserve seats on a flight.
  + *Cancel Booking:* Users can cancel their existing bookings.

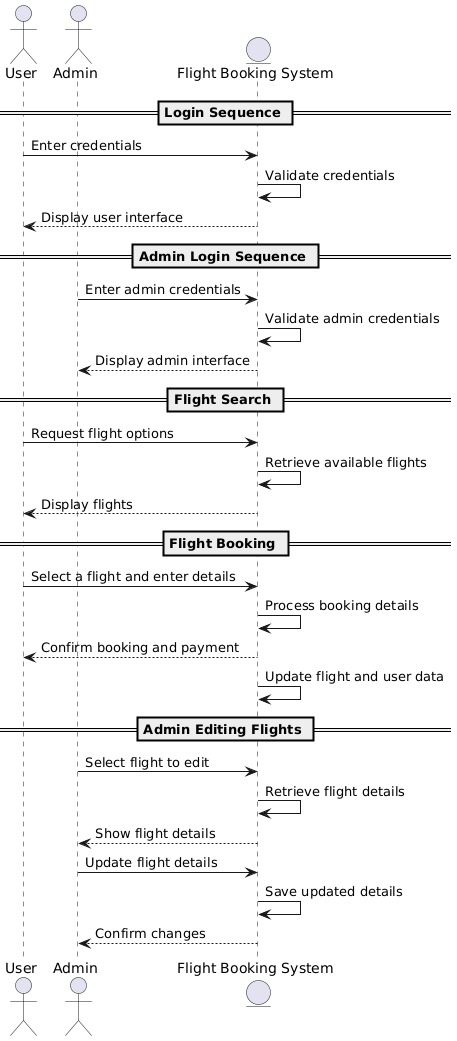


**Diagram 2: Sequence Diagram**

This diagram illustrates the step-by-step interaction between the actors (User/Admin) and the system for various processes.

**Key Sequences:**

1. **Login Sequence:**
   * User/Admin enters credentials.
   * System validates the input.
   * Appropriate interface (user/admin) is displayed based on the credentials.
2. **Flight Search:**
   * User requests flight options.
   * System retrieves and displays the available flights.
3. **Flight Booking:**
   * User selects a flight and provides booking details.
   * System processes the booking and updates relevant flight and user data.
   * Booking confirmation is sent to the user.
4. **Admin Editing Flights:**
   * Admin selects a flight to edit.
   * System retrieves and displays current flight details.
   * Admin modifies and saves updated flight information.

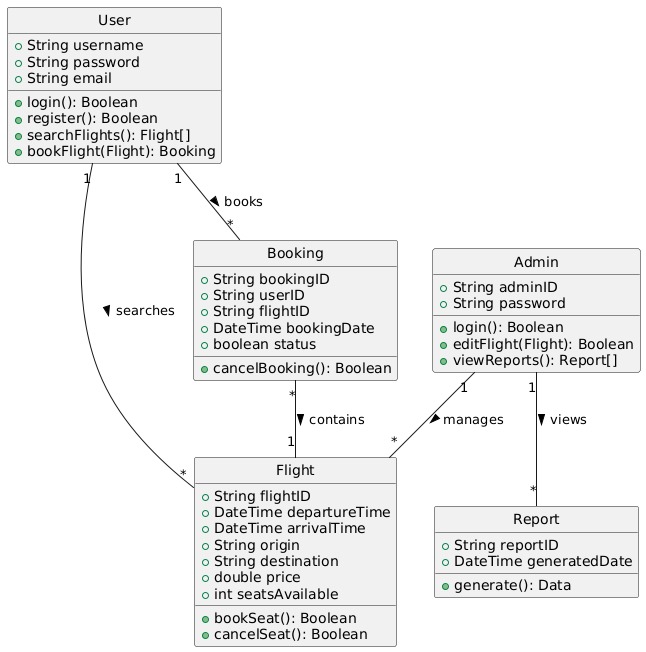


**Diagram 3: Class Diagram**

This diagram represents the structure of the system and relationships between various entities.

**Key Classes:**

* **User:**
  + Attributes: username, password, email
  + Methods: login(), register(), searchFlights(), bookFlight()
  + Description: Represents end-users of the system who can search and book flights.
* **Admin:**
  + Attributes: adminID, password
  + Methods: login(), editFlight(), viewReports()
  + Description: Represents system administrators who manage flights and oversee operations.
* **Flight:**
  + Attributes: flightID, departureTime, arrivalTime, origin, destination, price, seatsAvailable
  + Methods: bookSeat(), cancelSeat()
  + Description: Stores and manages details of available flights.
* **Booking:**
  + Attributes: bookingID, userID, flightID, bookingDate, status
  + Methods: cancelBooking()
  + Description: Represents individual bookings made by users.
* **Report:**
  + Attributes: reportID, generatedDate
  + Methods: generate()
  + Description: Used by admins to analyze system data.



**Diagram 4: Object Diagram**

This diagram provides a specific instance of the class diagram, showing the objects and their relationships during runtime.

**Key Instances:**

1. **User Objects:**
   * Contains details of a specific user, such as username and password.
   * Associated methods allow users to search flights or book a specific flight.
2. **Booking Objects:**
   * Represents a specific booking, linked to a User and a Flight.
3. **Flight Objects:**
   * Stores details for a specific flight, including departure time, destination, and price.
4. **Admin Objects:**
   * Used by administrators to access flight and booking management tools.

