Hotel Booking System

Purpose

The Hotel Booking System is a web-based application that facilitates:

- Managing hotels and rooms.
- Handling reservations for guests.
- Processing payments for bookings.
- Managing guest details and providing reports.
- Sending notifications (e.g., via JMS for reservation confirmation).

Scenario:

- A Customer searches for a hotel.
- The system shows available Hotels.
- The Customer select a hotel and chooses room
- A booking is created with check-in, check-out dates, guests information and payment details
- payment is made
- The hotel room availability is updated.

Features

1. User Management

- User authentication and authorization.
- Role-based access control (Admin and Customer).

2. Hotel Management

- Add, update, and delete hotels.
- View details of hotels, including location and ratings.

3. Room Management

- Add, update, and delete rooms.
- Associate rooms with hotels.
- Manage room availability (booked or free).

4. Reservation Management

- Create, update, and cancel reservations.

- Assign multiple guests to a reservation.
- Generate a unique reservation number for each booking.

- Filter reservations by date and user.

Implemented Technologies and Features

1. **REST API**
    a. Design and development of RESTful APIs for managing hotel bookings, reservations, and related entities.
2. **Dynamic Query, Named Query, and Criteria Query**
    a. Support for flexible and reusable query mechanisms for complex data retrieval.
3. **JMS (ActiveMQ)**
    a. Separate Producer and Consumer projects to implement messaging for reservation confirmation notifications.
4. **Content Negotiation**
    a. Support for both JSON and XML formats using `@Produces` and `@Consumes`.
5. **Spring Security**
    a. Authentication and Authorization mechanisms:
        i. Admin role: Restricted access to administrative APIs.
        ii. User role: Access to user-specific functionalities.
        iii. Public APIs: Accessible by anonymous users without authentication.
6. **Aspect-Oriented Programming (AOP)**
    a. Implementation of logging, auditing, and performance monitoring using AOP.
7. **Environment Profiles**
    a. Configuration-based environment setup:
        i. H2 Database for the development environment.
        ii. MySQL Database for the production environment.
8. **Concurrency Handling**
    a. Optimistic and Pessimistic locking mechanisms implemented for entity consistency during concurrent updates.

POST Signup

http://localhost:5050/api/auth/signup

It is the public api. anyone can be access this api

There can be two type of Role: **ROLE_ADMIN** for Admin and **ROLE_USER** for User

Body raw (json)

json

```
{
    "username": "admin",
    "password": "admin123",
    "email":"mdsahid.hossain@miu.edu",
    "phoneNumber":"9176355640",
    "role":"ROLE_ADMIN"
}
```

POST CreateHotel

http://localhost:5050/api/hotels/create

This API is accessible only by users with the **Admin** role.

AUTHORIZATION Basic Auth

Username	<username>
Password	<password>

Body raw (json)

json

```
{
    "name": "Hotel Paradise",
    "address": "123 Ocean Drive",
    "city": "Fairfield",
    "contactInfo": "info@paradise.com",
    "rating": 5
}
```

GET GetHotels

http://localhost:5050/api/hotels

It is a **public** api. Anyone can be access this api for hotel search

GET GetHotelById

http://localhost:5050/api/hotels/1

This API is accessible only by users with the Admin role.

AUTHORIZATION Basic Auth

Username	<username>
Password	<password>

PUT UpdateHotel

http://localhost:5050/api/hotels/1

This API is accessible only by users with the **Admin** role.

All fileds are not required.

AUTHORIZATION Basic Auth

Username	<username>
Password	<password>

Body raw (json)

json

```
{
    "name": "Al Amin Hotel2",
    "address": "123 Ocean Drive",
    "city": "NY",
    "contactInfo": "info@grandplaza.com",
    "rating": 3
}
```

DELETE DeleteHotel

http://localhost:5050/api/hotels/5

This API is accessible only by users with the **Admin** role.

AUTHORIZATION Basic Auth

Username	<username>
Password	<password>

POST CreateRoom

http://localhost:5050/api/rooms/hotels/2

This API is accessible only by users with the **Admin** role.

AUTHORIZATION Basic Auth

Username	<username>
Password	<password>

Body raw (json)

```json
{
    "roomType": "Delux",
    "bedType": "King",
    "roomNumber": "103",
    "pricePerNight": 200.0,
    "available": true,
    "hotel": {
        "id": 2
    }
}
```

GET GetAllRoom

http://localhost:5050/api/rooms

It is **public api**. Anyone can be access this api.

GET GetRoomById

http://localhost:5050/api/rooms/1

This API is accessible only by users with the **Admin** role.

AUTHORIZATION Basic Auth

Username	<username>
Password	<password>

PARAMS

PUT UpdateRooms

http://localhost:5050/api/rooms/2

This API is accessible only by users with the **Admin** role.
All fields are not required.

AUTHORIZATION Basic Auth

Username	<username>
Password	<password>

Body raw (json)

json

```
{
    "roomType": "Belux",
    "bedType": "King",
    "roomNumber": "103",
    "pricePerNight": 200.0,
    "available": true

}
```

DELETE DeleteRoom

http://localhost:5050/api/rooms/5

This API is accessible only by users with the **Admin** role.

POST MakeReservation

http://localhost:5050/api/reservations/create

This API is accessible only by users with the **User** role.

AUTHORIZATION Basic Auth

Username	<username>
Password	<password>

Body raw (json)

```json
{
    "reservation": {
        "checkInDate": "2024-12-20",
        "checkOutDate": "2024-12-25",
        "numberOfGuests": 2
    },
    "payment": {
        "paymentMethod": "Credit Card"
    },
    "roomIds": [
        1,
        2

    ],
    "guests": [
        {
            "fullName": "John Doe",
            "email": "john.doe@example.com",
            "phone": "123-456-7890",
            "address": "123 Main St, City, Country",
            "dateOfBirth": "1985-05-20"
```

```
        },
        {
            "fullName": "Jane Doe2",
            "email": "jane.doe@example.com",
            "phone": "123-456-7891",
            "address": "123 Main St, City, Country",
            "dateOfBirth": "1980-06-15"
        }
    ]
}
```


GET GetAllReservation

http://localhost:5050/api/reservations?fromDate=2024-12-10&toDate=2024-12-31&userId=2

This API is accessible only by users with the **User** role.

Query is not required.

AUTHORIZATION Basic Auth

Username	<username>
Password	<password>

PARAMS

fromDate	2024-12-10
toDate	2024-12-31
userId	2