**Hotel Management Microservices**

Welcome to the Hotel Management Microservices project. This application is designed to manage various aspects of a hotel, including booking rooms, adding hotels.

**Table of Contents**

* [Features](https://github.com/Amanastel/Hotel-Management-Microservices#features)
* [Tech Stack](https://github.com/Amanastel/Hotel-Management-Microservices#tech-stack)
* [Getting Started](https://github.com/Amanastel/Hotel-Management-Microservices#getting-started)
* [Installation](https://github.com/Amanastel/Hotel-Management-Microservices#installation)
* [Configuration](https://github.com/Amanastel/Hotel-Management-Microservices#configuration)
* [Usage](https://github.com/Amanastel/Hotel-Management-Microservices#usage)
* [Checking Service Status](https://github.com/Amanastel/Hotel-Management-Microservices#checking-service-status)
* [API Endpoints](https://github.com/Amanastel/Hotel-Management-Microservices#api-endpoints)
* [Contributing](https://github.com/Amanastel/Hotel-Management-Microservices#contributing)
* [License](https://github.com/Amanastel/Hotel-Management-Microservices#license)

**Features**

This application provides the following features:

**User Service**

* User registration and authentication.
* User can book hotel rooms.
* User can cancel bookings.

**Hotel Service**

* Add hotels with details.
* Add rooms to hotels.
* Get hotel details by ID or name.
* Get a list of all available rooms in a hotel.
* Get a list of all booked rooms in a hotel.

**Booking Service**

* Book rooms in hotels.
* Get booking details by booking ID.
* Get a list of all bookings.

**Service Registry**

* Register and discover microservices.

**Configuration Server**

* Manage centralized configurations for microservices.

**API Gateway**

* Gateway for accessing microservices.

**Tech Stack**

* Java
* Spring Boot
* Spring Cloud
* Spring Cloud Eureka
* Spring Cloud Config
* Spring Security
* OAuth
* Spring Data MongoDB
* Spring Data REST
* Spring Cloud Gateway
* Netflix Eureka

**Configuration**

* Configure the properties of each microservice according to your requirements.

**Usage**

* Run each microservice individually to start the Hotel Management system.

**Checking Service Status**

* **Eureka Server**:
  + Eureka Dashboard: http://localhost:8761
    - You can check the status of all registered microservices here. It will show which services are up and running and their corresponding instances.

Instances currently registered with Eureka:

* **API-GATEWAY**:
  + Availability Zones: UP (1) - 192.168.1.4:API-GATEWAY:8086
* **CONFIG-SERVER**:
  + Availability Zones: UP (1) - 192.168.1.4:CONFIG-SERVER:8085
* **HOTELS-SERVICE**:
  + Availability Zones: UP (1) - 192.168.1.4:hotels-service:8082
  + : UP (1) - 192.168.1.4:rating-service:8083
* **USERS-SERVICE**:
  + Availability Zones: UP (1) - 192.168.1.4:users-service:8081
* **API Gateway Default URL**: http://localhost:8086

**User Service Routes**

* User login: http://localhost:8086/auth/login
  + Login on this URL to obtain an access token. You can use this access token to make authenticated requests to other service endpoints.
* Fetch all users: http://localhost:8086/users/all
* User registration: http://localhost:8086/users/register
* User login: http://localhost:8086/users/login
* Book a hotel room: http://localhost:8086/users/addBooking
* Complete a Booking hotel: http://localhost:8086/users/completeBooking/{bookingId}
* Cancel a booking: http://localhost:8086/users/cancelBooking/{bookingId}
* : **Hotel Service Routes**
* Fetch all hotels: http://localhost:8086/hotels/all
* Fetch hotel by ID: http://localhost:8086/hotels/{hotelId}
* Add a hotel: http://localhost:8086/hotels/add
* Add a room to a hotel: http://localhost:8086/hotels/{hotelId}/rooms/add
* Book a room in a hotel: http://localhost:8086/hotels/{hotelId}/bookings/add
* Get all bookings in a hotel: http://localhost:8086/hotels/{hotelId}/bookings/all