Register Order App in Spring Cloud

Description
Objective:

To create microservice and enable discovery client.

Concept Explanation:

1. Eureka Server is a service registry and discovery server designed for microservices architectures.
2. It allows microservices to register themselves and discover other services dynamically.
3. This enables efficient load balancing, failover, and service-to-service communication within distributed systems.
4. Eureka Server is an essential component in building resilient and scalable microservices-based applications.

Concept Implementation:

1. A Maven project for a Spring Boot application called "**OrderApp**" that serves as an order management system. The application includes a REST controller (`OrderController`) with a method (`getOrderDetails`) to fetch order details, and a model class (`Order`) representing an order entity.
2. It utilizes Spring Cloud's Netflix Eureka for service registration and discovery, enabling it to be discovered by other services.
3. The application is configured to run on port 8071, with the name "ordersapp" registered in the Eureka server located at http://webapps.tekstac.com:8760/eureka/. Additionally, it's set to prefer IP address for service registration.

Register Order App in Spring Cloud

This exercise is implemented using Spring Cloud (Eureka) as service registry and Spring Boot for REST Service.

The Order Application is developed and provided as part of the code skeleton. This application is a rest service application. The name of the application is "**ordersapp**". You are required to register the given REST service in the eureka server.

The Eureka server is running in port number 8761

Access Order App Via Router

Description
Objective:

To create microservice, enable discovery client and spring cloud Gateway.

Concept Explanation:

1. A cloud API gateway is a crucial component in modern cloud-based architectures, serving as a centralized entry point for managing and routing requests to backend services and APIs.
2. Its primary function include **Routing**: It directs incoming requests from clients to the appropriate backend services based on predefined rules, paths, or parameters.

Concept Implementation:

1. Create a gateway application that routes incoming requests to the appropriate backend services registered with the Eureka server, providing a centralized entry point for communication within a microservices architecture.
2. The `RouterAppApplication` class is annotated with `@SpringBootApplication`, indicating that it is the main class of a Spring Boot application. Additionally, `@EnableDiscoveryClient` annotation is used to enable service registration and discovery with Eureka.

Access Order App Via Router

This exercise is implemented using Spring Cloud (Eureka) as service registry and Spring Boot as REST Service and the service should be accessed via the router

The Order Application is developed and provided as part of the code skeleton. This application is a rest service application. The name of the application is "**ordersapp**". You are required to register the given REST service in the eureka server.

The Eureka server is running in port number 8761 and the URL for the same is given below:

http://localhost:8761/eureka

The OrderApp application should be accessed via the Router, you need to develop the Router application do to the same.