SERVICE DISCOVERY CONCEPTS AND EUREKA CONFIGURATION

REVIEWER Mr. Thomas Darko

Service Discovery Concepts and Eureka Configuration

Introduction to Service Discovery

In a microservices architecture, multiple services are deployed across different hosts or containers. Service discovery is a mechanism that allows services to find each other without hardcoding IP addresses or hostnames. This is critical in dynamic environments where services might scale up, down, or change locations frequently.

Service discovery can be divided into two main categories:

- Client-side discovery
- Server-side discovery

Client-Side Discovery

In client-side discovery, the client is responsible for determining the location of available service instances and routing the request accordingly. Clients query a service registry, which acts as a database of available services. Examples of service registries include Netflix Eureka and Consul.

Server-Side Discovery

In server-side discovery, the client sends a request to a load balancer, which then queries the service registry and forwards the request to an available instance. The load balancer acts as an intermediary, making this method more centralized compared to client-side discovery.

What is Netflix Eureka?

Netflix Eureka is a service registry for service discovery, providing a REST-based API for managing service instances and their registration. It allows microservices to register themselves and query other services that are available.

Eureka has two main components:

1. Eureka Server: It holds information about all client-service applications. Each microserviceregisters with the Eureka server, and clients query the Eureka server for information.

2. Eureka Client: A microservice that registers itself with the Eureka server and uses it to discoverother services.

Configuring Eureka in a Spring Boot Application

In a Spring Boot application, configuring Eureka is simple thanks to Spring Cloud Netflix. Follow these steps:

1. Add the following dependencies in your `pom.xml` file:

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
</dependency>

<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
</dependency>
```

2. Enable Eureka Server in your main application class:

```
@SpringBootApplication
@EnableEurekaServer public class
EurekaServerApplication {   public static
void main(String[] args) {
```

SpringApplication.run(EurekaServerAppli

cation.class, args);

}

}

3. Configure application.properties or application.yml file:

For the Eureka Server:

```

eureka.client.register-with-eureka=false

eureka.client.fetch-registry=false server.port=8761

```

For Eureka Clients (other microservices):

```

eureka.client.service-url.defaultZone=http://localhost:8761/eureka/ spring.application.name=your-service-

name

```