**MICROSERVIC WITH SPRING CLOUD**

[**Microservice**](#_Service_Discovery)[**Architecture**](#_Microservice_Architecture)**……………………………………... 1**

[**Service Discovery**](#_Service_Discovery)**……………………………………………… 2**

[**API Gateway**](#_API_Gateway)**………………………………………………….. 3**

[**Cloud Config Server**](#_Cloud_Config_Server)**…………………………………………… 4**

[**Microservices**](#_Microservices) **…………………………………………………. 5**

[**Distributed Tracing Tool**](#_Distributed_Tracing_Tool) **………………………………………. 6**

##### Microservice Architecture

API Gateway

Customer

Service

Department

Service

Config Server

Service Registry

Git

##### 

##### 

##### 

##### Service Discovery

service-registry:

* This is application is service discovery or registry. It manages all microservices port detail, uri information and so on.

🡺 open <https://start.spring.io/>

🡺 Add Eureka Server (Spring cloud discovery) and generate the war file and import it to IDE

🡺 Create application.yml in resources 🡺 add bellow configuration

Graphical user interface, text, application

Description automatically generated

🡺 To register services to Eureka Server, Open <https://start.spring.io/> 🡺 add Eureka Discovery Client (Spring

cloud discovery) dependency not Eureka Server and copy the dependency 🡺 add it to all microservices such

as customer-services and department-services.

1, **<spring-cloud.version>**2021.0.4**</spring-cloud.version> inside <properties>**

**2, <dependency>**

**<groupId>**org.springframework.cloud**</groupId>**

**<artifactId>**spring-cloud-starter-netflix-eureka-server**</artifactId>**

**</dependency>**

3, **<dependencyManagement>**

**<dependencies>**

**<dependency>**

**<groupId>**org.springframework.cloud**</groupId>**

**<artifactId>**spring-cloud-dependencies**</artifactId>**

**<version>**${spring-cloud.version}**</version>**

**<type>**pom**</type>**

**<scope>**import**</scope>**

**</dependency>**

**</dependencies>**

**</dependencyManagement>**

**<build>**

**<plugins>**

**<plugin>**

**<groupId>**org.springframework.boot**</groupId>**

**<artifactId>**spring-boot-maven-plugin**</artifactId>**

**</plugin>**

**</plugins>**

**</build>**

🡺 We also supposed to add bellow configuration to each microservices and gateway server. But, we will add

this configuration into cloud config to avoid repetitive configuration.

Text

Description automatically generated with medium confidence

🡺 We add bellow snippet of code into microservice main class who wants to contact other microservices

// Automatically do the load balancing whenever we use this template to communicate between

microservices

@Bean

@LoadBalanced

public RestTemplate restTemplate(){

return new RestTemplate();

}

🡺 Later we will open <http://localhost:8761/> and verify the services if added successfully.

##### API Gateway

**cloud-gateway:**

* This is API gateway it routs call to customer-service and department-service microservices by using only one URL endpoint. E.x <http://localhost:9191/customer/1> we only change name to make different call

🡺 open <https://start.spring.io/>

🡺 Add Eureka Discovery Client (Spring cloud discovery) dependency not Eureka Server

🡺 Add Gateway (Spring cloud routing) dependency

🡺 Add Actuator (Spring spring boot ops ) dependency

🡺 Generate jar file and open it with IDE and create application.yml and configure it bellows snippet

A picture containing graphical user interface

Description automatically generated

🡺 To point application to config-server application add bellow dependency and create bootstrap.yml

(external cloud configuration yml) in resources and add bellow configuration

<dependency>

<groupId> org.springframework.cloud </groupId>

<artifactId>spring-cloud-starter-config</artifactId>

</dependency>

Graphical user interface, text, application

Description automatically generated

##### Cloud Config Server

**config-server:**

* This application a centralized config server which help as to config all microservices, gateway, & config server in remoter repository GitHub.

🡺 open <https://start.spring.io/>

🡺 Add Eureka Discovery Client (Spring cloud discovery) dependency not Eureka Server

🡺 Add Config Server (Spring Cloud Config) dependency

🡺 Enable @EnableConfigServer @EnableEurekaClient

🡺 Create application.yml file and add below config to connect to git repository

Graphical user interface, text, application, email

Description automatically generated

**department-service:**

* This application a micro service rest Api. It adds department.

🡺 open <https://start.spring.io/>

🡺 Add Spring Web, Spring Data JP, H2 Database SQL(for demo purpose h2 used), and Lombok. Open

application with IDE.

🡺 Post method to add department (this uri is after gateway is connected to microservices)

<http://localhost:9191/department/>

{

"departmentName": "IT",

"depatmentAddress": "1212 A st, California",

"departmentCode": "IT-123"

}

🡺 Get method to get added department <http://localhost:9191/department/1>

##### Microservices

**customer-service:**

* This application is a micro service rest Api. It adds customer.

🡺 open <https://start.spring.io/>

🡺 Add Spring Web, Spring Data JP, H2 Database SQL(for demo purpose h2 used), and Lombok. Open

application with IDE.

🡺 Post method to add department (this uri is after gateway is connected to microservices)

<http://localhost:9191/customer/>

{

"firstName": "Peter",

"lastName": "Wang",

"email": "peter@gmail.com",

"departmentId": "2"

}

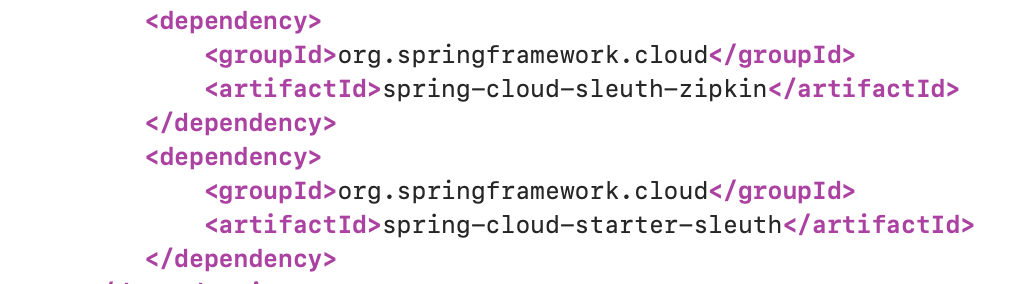
🡺 Get method to get added department <http://localhost:9191/customer/1>

##### Distributed Tracing Tool

**zipkin:**

* This is a distributed tracing system.

🡺 Add bellow dependency to all microservices so that Zipkin can trace the activity



🡺 download Zipkins and run java -jar zipkin-server-2.23.18-exec.jar command

🡺 Open <http://127.0.0.1:9411/zipkin/> and check if services are register by sending queries