**Microservices – API Gateway Integration**

**Steps:**

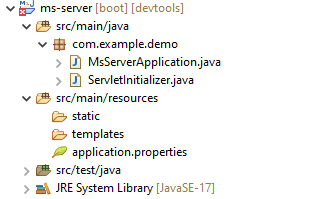
1. Creating a Eureka Server in Springboot

2. Building Microservices in Springboot

3. Usage of API Gateway in Springboot

**4.3.1 Creating a Eureka Server in Springboot**

**Project Structure:**

****

**Dependencies to be add:**

spring-cloud-starter-netflix-eureka-server

spring-boot-starter-web

spring-boot-devtools

**Application.Properties**

**-------------------------------------**

spring.application.name=ms-server

server.port = 8761

eureka.client.register-with-eureka=false

eureka.client.fetch-registry=false

**MsServerApplication.java**

--------------------------------

@SpringBootApplication

@EnableEurekaServer

**public** **class** MsServerApplication {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(MsServerApplication.**class**, args);

System.out.println(“server is running…”);

}

}

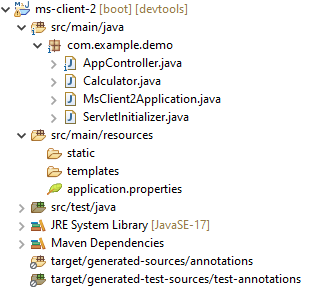
**4.3.2 Building Microservices in Springboot**

Here we are creating two microservices, that are:

Ms-client-2

Ms-client-3

**Project Structure of Ms-client-2**

****

**Dependencies to be add:**

spring-cloud-starter-netflix-eureka-client

spring-boot-starter-web

spring-boot-devtools

**MsClient2Application.java**

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.cloud.client.discovery.EnableDiscoveryClient;

@SpringBootApplication

@EnableDiscoveryClient

public class MsClient2Application {

public static void main(String[] args) {

SpringApplication.run(MsClient2Application.class, args);

System.out.println(“Client-2 is running…”);

}

}

**AppController.java**

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/client2") // Base path for ms-client-2

public class AppController {

@GetMapping("/add/{a}/{b}")

public String add(@PathVariable("a") int a, @PathVariable("b") int b) {

Calculator c1 = new Calculator();

c1.setA(a);

c1.setB(b);

return "From Client 2: " + c1.add();

}

}

**Calculator.java**

public class Calculator {

int a, b, c;

public int getA() {

return a;

}

public void setA(int a) {

this.a = a;

}

public int getB() {

return b;

}

public void setB(int b) {

this.b = b;

}

public int getC() {

return c;

}

public void setC(int c) {

this.c = c;

}

public int add() {

c = a + b;

return c;

}

public int sub() {

c = a - b;

return c;

}

}

**Application.properties**

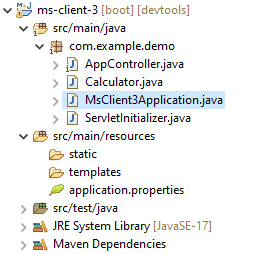
spring.application.name=ms-client-2

server.port = 8082

eureka.client.service-url.default-zone=http://localhost:8761/eureka

**=============================================================**

**Project Structure of Ms-client-3**



**Dependencies to be add:**

spring-cloud-starter-netflix-eureka-client

spring-boot-starter-web

spring-boot-devtools

**MsClient3Application.java**

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.cloud.client.discovery.EnableDiscoveryClient;

@SpringBootApplication

@EnableDiscoveryClient

public class MsClient3Application {

public static void main(String[] args) {

SpringApplication.run(MsClient3Application.class, args);

System.out.println(“Client-3 is running…”);

}

}

**AppController.java**

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/client3") // Base path for ms-client-3

public class AppController {

@GetMapping("/add/{a}/{b}")

public String add(@PathVariable("a") int a, @PathVariable("b") int b) {

Calculator c1 = new Calculator();

c1.setA(a);

c1.setB(b);

return "From Client 3: " + c1.add();

}

}

**Calculator.java**

public class Calculator {

int a, b, c;

public int getA() {

return a;

}

public void setA(int a) {

this.a = a;

}

public int getB() {

return b;

}

public void setB(int b) {

this.b = b;

}

public int getC() {

return c;

}

public void setC(int c) {

this.c = c;

}

public int add() {

c = a + b;

return c;

}

public int sub() {

c = a - b;

return c;

}

}

**Application.properties**

spring.application.name=ms-client-3

server.port = 8083

eureka.client.service-url.default-zone=http://localhost:8761/eureka

==================================================================

**3.3.3 Usage of API Gateway in Springboot**

**Dependencies to be add: (Reactive Gateway, Eureka Client)**

<dependency>

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

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

</dependency>

<dependency>

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

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

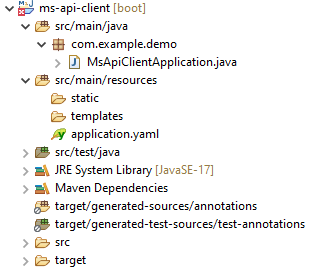
<dependency>

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

<artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>

</dependency>

**Project Structure of ms-api-client**

****

**Application.yaml**

server:

port: 8085

spring:

application:

name: ms-api-client

cloud:

gateway:

routes:

- id: ms-client-2

uri: lb://ms-client-2 # http://localhost:8082/client2/

predicates:

- Path=/client2/\*\*

- id: ms-client-3

uri: lb://ms-client-3 # http://localhost:8083/client3/

predicates:

- Path=/client3/\*\*

eureka:

client:

service-url:

default-zone: http://localhost:8761/eureka

**MsApiClientApplication.java**

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.cloud.client.discovery.EnableDiscoveryClient;

@SpringBootApplication

@EnableDiscoveryClient

public class MsApiClientApplication {

public static void main(String[] args) {

SpringApplication.run(MsApiClientApplication.class, args);

System.out.println(“API-Gateway is running…”);

}

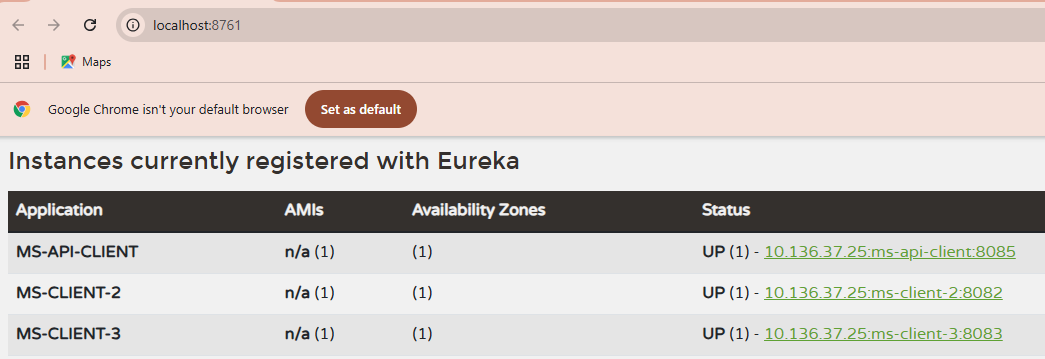
}

**Execution Process**

Restart the ms-server project (by right click on Boot Dashboard and click on Restart)

Run the ms-server project (by right click on Boot Dashboard and click on Open Web Browser)

Initially it will show “No instances are registered”



Restart and Run the ms-client2

Refresh the ms-server browser. It will detect the client-2

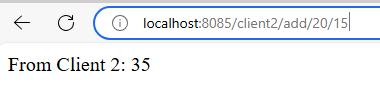
Restart and Run the ms-client3

Refresh the ms-server browser. It will detect the client-3

Restart and Run the ms-api-client (Its port number is 8085)

Refresh the ms-server browser. It will detect the ms-api-client

Executing Client2 with API-Gateway Port number as follows:



Executing Client3 with API-Gateway Port number as follows:

