

01-03-2025

Service Discovery

In a microservices architecture, multiple services communicate with each other. Manually configuring each service's location (host & port) is inefficient, especially in dynamic environments where services scale up or down. Service discovery solves this problem by **allowing services to register themselves and dynamically discover other services.**

What is Eureka?

Netflix Eureka is a **service registry and discovery tool** in Spring Cloud that helps microservices **register and discover each other dynamically**. It consists of:

1. **Eureka Server** – Acts as a service registry where all microservices register.
2. **Eureka Clients** – Microservices that register with Eureka Server and discover other services dynamically.

How Eureka Works

1. **Service Registration**
 - Each microservice registers itself with the Eureka Server.
 - It periodically sends heartbeats to inform the registry that it's alive.
2. **Service Discovery**
 - When a microservice needs to call another service, it queries Eureka Server to get the service's location dynamically.
 - This avoids hardcoding service URLs.
3. **Load Balancing & Fault Tolerance**
 - Eureka provides a list of available service instances.
 - Services can use **Ribbon (deprecated)** or **Spring Cloud Load Balancer** to distribute requests efficiently.

Run the Eureka Server (`@EnableEurekaServer`).

Register microservices (`@EnableDiscoveryClient`).

```
package com.wipro.service;
```

```
import org.springframework.cloud.openfeign.FeignClient;

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

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


import com.wipro.dto.APIResponseDto;

import com.wipro.dto.DepartmentDto;


//@FeignClient(url = "http://localhost:9090",value = "department-service")

@FeignClient(name = "DEPARTMENT-SERVICE")


public interface APIClient {


    @GetMapping("/departments/{departmentCode}")

    public DepartmentDto getDepartmentByCode(@PathVariable("departmentCode") String x);


}
```

Service Registry in Spring Cloud

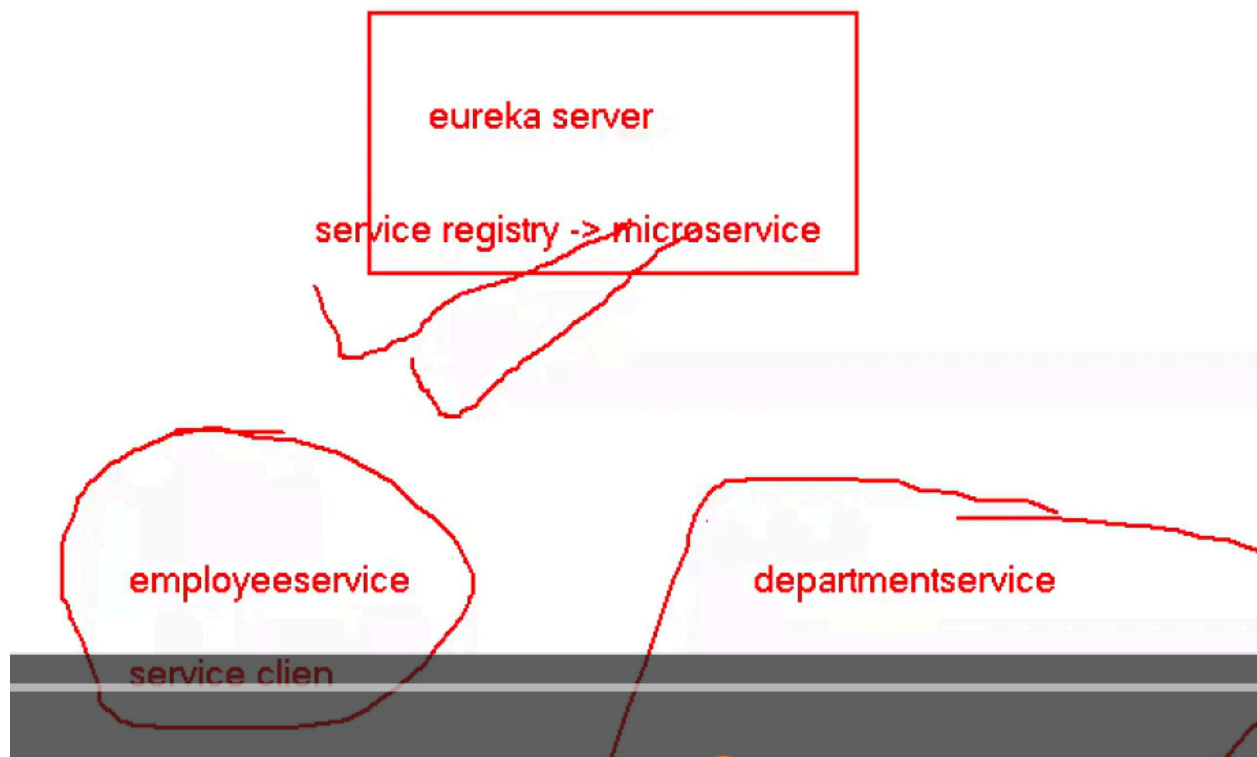
A Service Registry in Spring Cloud is a mechanism that helps in service discovery within a microservice architecture. It allows services to register themselves and discover other services dynamically without needing to hardcode their locations.

Why Service Registry?

1. **Dynamic Scaling** – Microservices may scale up or down, and their IP addresses may change dynamically.
2. **Loose Coupling** – Services do not need to know the exact network location of other services.
3. **Load Balancing** – Works with client-side load balancing (e.g., **Spring Cloud LoadBalancer**).
4. **Fault Tolerance** – Automatically removes unhealthy services from the registry.

How It Works?

1. **Service Registration** – Each microservice registers itself with the Service Registry at startup.
2. **Service Discovery** – Other microservices query the registry to find available instances.
3. **Health Checks** – The registry monitors the health of registered services and removes any failed ones.



To work with this...we need to create the service registry in the start.spring.io

Project

☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ Maven

Language

☒ Java ☐ Kotlin ☐ Groovy

Spring Boot

☐ 3.5.0 (SNAPSHOT) ☐ 3.5.0 (M2) ☐ 3.4.4 (SNAPSHOT) ☒ 3.4.3
☐ 3.3.10 (SNAPSHOT) ☐ 3.3.9

Project Metadata

Group
Artifact
Name
Description
Package name
Packaging ☒ Jar ☐ War
Java ☐ 23 ☐ 21 ☒ 17

Dependencies

ADD ...

Eureka Server

SPRING CLOUD DISCOVERY

spring-cloud-netflix Eureka Server.

To make any application as a service registry we need to add few properties

server.port=8761

eureka.client.register-with-eureka=false

eureka.client.fetch-registry=false

- `server.port=8761` → Runs Eureka Server on port **8761**.
- `eureka.client.register-with-eureka=false` → The Eureka server **does not register itself**.
- `eureka.client.fetch-registry=false` → The server **does not fetch the registry**, as it is the registry itself.

```

1 spring.application.name=service-registry
2
3
4 server.port=8761
5
6 eureka.client.register-with-eureka=false
7 eureka.client.fetch-registry=false
8

```

And this service registry must be enable the `@EnableEurekaServer` in the main application

```

1 package com.wipro;
2
3 import org.springframework.boot.SpringApplication;
4
5
6
7 @SpringBootApplication
8 @EnableEurekaServer
9 public class ServiceRegistryApplication {
10
11     public static void main(String[] args) {
12         SpringApplication.run(ServiceRegistryApplication.class, args);
13     }
14
15 }
16

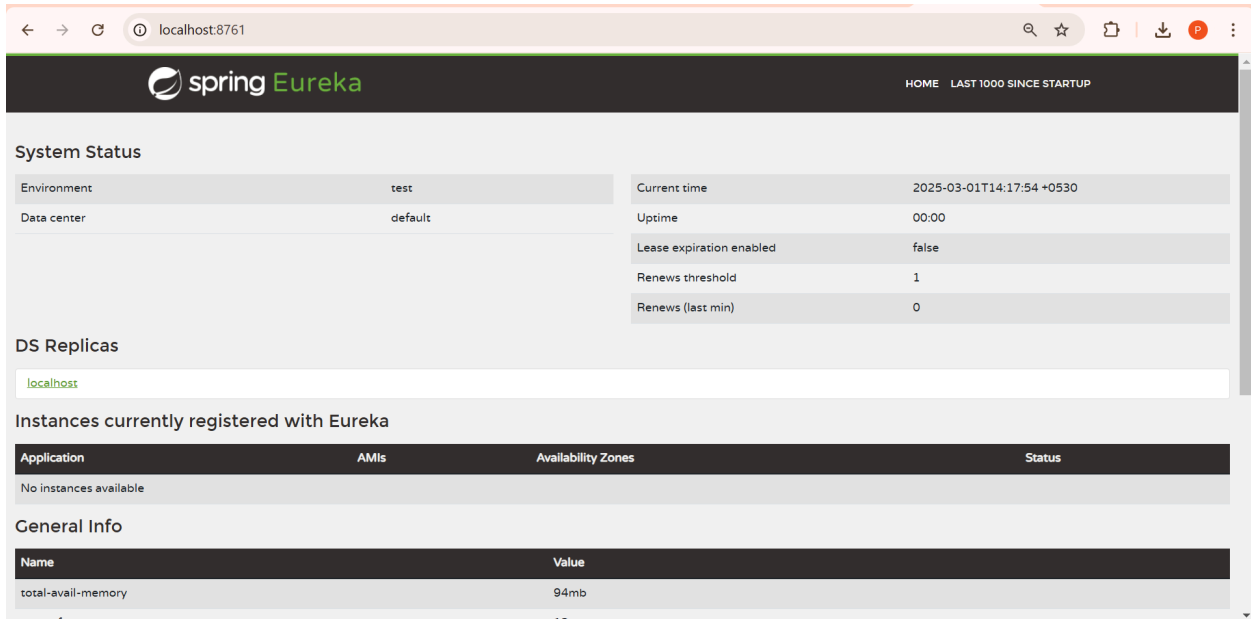
```

`@EnableEurekaServer` Annotation in Spring Cloud

The `@EnableEurekaServer` annotation is used to configure a Spring Boot application as a Eureka Server, which acts as a Service Registry in a microservices architecture.

Now run the server and check in browser

Localhost:8761



As of now we don't have any Microservices registered

Now we need to register them as department and employee service

If we want to make any service as service discovery or eureka client we need to add the dependency into the xml files

`<dependency>`

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

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

`<version>2023.0.1</version>`

`<type>pom</type>`

`<scope>import</scope>`

`</dependency>`

Now open the department service go to the pom.xml file and add it

And also add these properties into the properties in the department service

`eureka.client.register-with-eureka=true`

`eureka.client.fetch-registry=true`

`eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka/`

Then go for department main application and run

<https://drive.google.com/file/d/1Jk9lebPEw-8sh407YLRha706emZAqm7X/view?usp=sharing>

https://drive.google.com/file/d/1Q3AZr_5LvodfenPF9s2WePAKmMQDYsl3/view?usp=sharing

```

service-registry [boot]
├── src/main/java
│   └── com.wipro
│       └── ServiceRegistryApplication.java
├── src/main/resources
│   └── application.properties
├── src/test/java
├── JRE System Library [JavaSE-17]
├── Maven Dependencies
├── src
├── target
├── HELP.md
├── mvnw
├── mvnw.cmd
└── pom.xml

```

```

1 package com.wipro;
2
3 import org.springframework.boot.SpringApplication;
4
5
6 @SpringBootApplication
7 @EnableEurekaServer
8 public class ServiceRegistryApplication {
9
10
11     public static void main(String[] args) {
12         SpringApplication.run(ServiceRegistryApplication.class, args);
13     }
14
15 }
16

```

```

1 spring.application.name=service-registry
2
3 server.port=8761
4 eureka.client.register-with-eureka=false
5 eureka.client.fetch-registry=false
6

```


ServiceRegistryApplication.java application.properties service-registry/pom.xml ^

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>3.4.3</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>com.wipro</groupId>
12  <artifactId>service-registry</artifactId>
13  <version>0.0.1-SNAPSHOT</version>
14  <name>service-registry</name>
15  <description>Demo project for Spring Boot microservice service registry</description>
16  <url/>
17  <licenses>
18    <license/>
19  </licenses>
20  <developers>
21    <developer/>
22  </developers>
23  <scm>
24    <connection/>
25    <developerConnection/>
26    <tag/>
27    <url/>
28  </scm>
29  <properties>
30    <java.version>17</java.version>
31    <spring-cloud.version>2024.0.0</spring-cloud.version>
32  </properties>
33  <dependencies> Add Spring Boot Starters...
34    <dependency>
35      <groupId>org.springframework.cloud</groupId>
36      <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
37    </dependency>
38
39    <dependency>
40      <groupId>org.springframework.boot</groupId>
41      <artifactId>spring-boot-starter-test</artifactId>
42      <scope>test</scope>
43    </dependency>
44  </dependencies>
45  <dependencyManagement>
46    <dependencies>
47      <dependency>
48        <groupId>org.springframework.cloud</groupId>
49        <artifactId>spring-cloud-dependencies</artifactId>
50        <version>${spring-cloud.version}</version>
51        <type>pom</type>
52        <scope>import</scope>
53      </dependency>
54    </dependencies>
55  </dependencyManagement>
56
57  <build>
```

```

44     </dependencies>
45     <dependencyManagement>
46     <dependencies>
47         <dependency>
48             <groupId>org.springframework.cloud</groupId>
49             <artifactId>spring-cloud-dependencies</artifactId>
50             <version>${spring-cloud.version}</version>
51             <type>pom</type>
52             <scope>import</scope>
53         </dependency>
54     </dependencies>
55 </dependencyManagement>
56
57 <build>
58     <plugins>
59         <plugin>
60             <groupId>org.springframework.boot</groupId>
61             <artifactId>spring-boot-maven-plugin</artifactId>
62         </plugin>
63     </plugins>
64 </build>
65
66 </project>
67

```

Above codes are Service registry

Now below codes are Department registry

- ▼ Springboot-Department [boot] [devtools]
 - ▼ src/main/java
 - ▼ com.wipro
 - > SpringbootDepartmentApplication.java
 - > com.wipro.controller
 - > com.wipro.dto
 - > com.wipro.entity
 - > com.wipro.repository
 - > com.wipro.service
 - ▼ src/main/resources
 - static
 - templates
 - application.properties
 - > src/test/java
 - > JRE System Library [JavaSE-17]
 - > Maven Dependencies
 - target/generated-test-sources/test-annotations
 - target/generated-sources/annotations
 - > src
 - > target
 - HELP.md
 - mvnw
 - mvnw.cmd
 - pom.xml

SpringbootDepartmentApplication.java

```
1 package com.wipro;
2
3 import org.modelmapper.ModelMapper;
4
5 @SpringBootApplication
6 public class SpringbootDepartmentApplication {
7
8     public static void main(String[] args) {
9         SpringApplication.run(SpringbootDepartmentApplication.class, args);
10    }
11
12    @Bean
13    public ModelMapper getModelMapper() {
14        return new ModelMapper();
15    }
16 }
17
18
19
20
21
```

```
1 package com.wipro.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4
5 @RestController
6 @RequestMapping("/departments")
7 public class DepartmentController {
8
9     @Autowired
10    DepartmentService service;
11
12
13    @PostMapping
14    public DepartmentDto saveDepartment(@RequestBody DepartmentDto deptdto) {
15        return service.saveDepartment(deptdto);
16    }
17
18    @GetMapping("/{code}")
19    public DepartmentDto getDepartmentByCode(@PathVariable String code) {
20        return service.getDepartmentByCode(code);
21    }
22
23 }
24
25
26
27
28
29
30
31
32
33
34 }
35
36
```

```
1 package com.wipro.dto;
2
3 import lombok.AllArgsConstructor;
4
5
6
7 @AllArgsConstructor
8 @NoArgsConstructor
9 @Data
10 public class DepartmentDto {
11
12     private Long deptId;
13     private String department;
14     public DepartmentDto(Long deptId, String department, String description, String deptcode) {
15         super();
16         this.deptId = deptId;
17         this.department = department;
18         this.description = description;
19         this.deptcode = deptcode;
20     }
21     public Long getDeptId() {
22         return deptId;
23     }
24     public void setDeptId(Long deptId) {
25         this.deptId = deptId;
26     }
27     public String getDepartment() {
28         return department;
29     }
30     public void setDepartment(String department) {
31         this.department = department;
32     }
33     public DepartmentDto() {
34         super();
35     }
36     public String getDescription() {
37         return description;
38     }
39     public void setDescription(String description) {
40         this.description = description;
41     }
42     public String getDeptcode() {
43         return deptcode;
44     }
45     public void setDeptcode(String deptcode) {
46         this.deptcode = deptcode;
47     }
48     private String description;
49     private String deptcode;
50
51 }
52
```

SpringbootDepartmentApplication.java DepartmentController.java Department

```
1 package com.wipro.entity;
2
3 import jakarta.persistence.Entity;
4
11
12 @Entity
13 @Data
14 @NoArgsConstructor
15 @AllArgsConstructor
16 @Table(name = "departments")
17 public class Department {
18
19     @Id
20     @GeneratedValue(strategy = GenerationType.AUTO)
21     private Long deptId;
22     private String department;
23     private String description;
24     private String deptcode;
25     public Long getDeptId() {
26         return deptId;
27     }
28     public void setDeptId(Long deptId) {
29         this.deptId = deptId;
30     }
31     public String getDepartment() {
32         return department;
33     }
34     public void setDepartment(String department) {
35         this.department = department;
36     }
37     public String getDescription() {
38         return description;
39     }
40     public void setDescription(String description) {
41         this.description = description;
42     }
43     public String getDeptcode() {
44         return deptcode;
45     }
46     public void setDeptcode(String deptcode) {
47         this.deptcode = deptcode;
48     }
49     public Department(Long deptId, String department, String description, String deptcode) {
50         super();
51         this.deptId = deptId;
52         this.department = department;
53         this.description = description;
54         this.deptcode = deptcode;
55     }
56     public Department() {
57         super();
58     }
59
60 }
61
62 }
```

```

1 package com.wipro.repository;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5 @Repository
6 public interface DepartmentRepo extends JpaRepository<Department, Long> {
7
8     Department findByDeptcode(String deptcode); // Return Department entity, not DTO
9 }
10
11
12
13
14

```

SpringbootDepartmen... DepartmentController.j... DepartmentDto.java

```

1 package com.wipro.service;
2
3 import org.modelmapper.ModelMapper;
4
5 @Service
6 public class DepartmentService {
7
8     @Autowired
9     private DepartmentRepo repo;
10
11     @Autowired
12     private ModelMapper mapper;
13
14     public DepartmentDto saveDepartment(DepartmentDto deptdto) {
15         Department dept = mapper.map(deptdto, Department.class);
16         Department newdept = repo.save(dept);
17         return mapper.map(newdept, DepartmentDto.class);
18     }
19
20     public DepartmentDto getDepartmentByCode(String code) {
21         Department department = repo.findByDeptcode(code); // Fetch entity
22         return department != null ? mapper.map(department, DepartmentDto.class) : null;
23     }
24 }
25
26
27
28
29
30
31
32

```

```

1 spring.application.name=Springboot-Department
2 # MySQL Configuration
3 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
4 spring.datasource.url=jdbc:mysql://localhost:3306/wipro
5 spring.datasource.username=root
6 spring.datasource.password=#Mahadev7
7
8 server.port:9090
9
10 # JPA & Hibernate
11 spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect
12 spring.jpa.hibernate.ddl-auto=update
13 spring.jpa.show-sql=true
14
15 eureka.client.register-with-eureka=true
16 eureka.client.fetch-registry=true
17 eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka/

```

```

https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>3.4.3</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>com.wipro</groupId>
12  <artifactId>Springboot-Department</artifactId>
13  <version>0.0.1-SNAPSHOT</version>
14  <name>Springboot-Department</name>
15  <description>Demo project for Spring Boot microservice</description>
16  <url/>
17  <licenses>
18    <license/>
19  </licenses>
20  <developers>
21    <developer/>
22  </developers>
23  <scm>
24    <connection/>
25    <developerConnection/>
26    <tag/>
27    <url/>
28  </scm>
29  <properties>
30    <java.version>17</java.version>
31    <spring-cloud.version>2024.0.0</spring-cloud.version>
32  </properties>
33  <dependencies>    Add Spring Boot Starters...
34    <dependency>
35      <groupId>org.springframework.boot</groupId>
36      <artifactId>spring-boot-starter-data-jpa</artifactId>
37    </dependency>
38    <dependency>
39      <groupId>org.springframework.boot</groupId>
40      <artifactId>spring-boot-starter-web</artifactId>
41    </dependency>
42    <dependency>
43      <groupId>org.springframework.cloud</groupId>
44      <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
45    </dependency>
46
47    <dependency>
48      <groupId>org.springframework.boot</groupId>
49      <artifactId>spring-boot-devtools</artifactId>
50      <scope>runtime</scope>
51      <optional>true</optional>
52    </dependency>
53    <dependency>
54      <groupId>org.modelmapper</groupId>
55      <artifactId>modelmapper</artifactId>
56      <version>3.2.0</version>

```



























```

55     <artifactId>modelmapper</artifactId>
56     <version>3.2.0</version>
57 </dependency>
58
59     <dependency>
60         <groupId>com.mysql</groupId>
61         <artifactId>mysql-connector-j</artifactId>
62         <scope>runtime</scope>
63     </dependency>
64     <dependency>
65         <groupId>org.projectlombok</groupId>
66         <artifactId>lombok</artifactId>
67         <optional>true</optional>
68     </dependency>
69     <dependency>
70         <groupId>org.springframework.boot</groupId>
71         <artifactId>spring-boot-starter-test</artifactId>
72         <scope>test</scope>
73     </dependency>
74 </dependencies>
75 <dependencyManagement>
76 <dependencies>
77     <dependency>
78         <groupId>org.springframework.cloud</groupId>
79         <artifactId>spring-cloud-dependencies</artifactId>
80         <version>${spring-cloud.version}</version>
81         <type>pom</type>
82         <scope>import</scope>
83     </dependency>
84 </dependencies>
85 </dependencyManagement>
86
87 <build>
88     <plugins>
89         <plugin>
90             <groupId>org.apache.maven.plugins</groupId>
91             <artifactId>maven-compiler-plugin</artifactId>
92             <configuration>
93                 <annotationProcessorPaths>
94                     <path>
95                         <groupId>org.projectlombok</groupId>
96                         <artifactId>lombok</artifactId>
97                     </path>
98                 </annotationProcessorPaths>
99             </configuration>
100         </plugin>
101         <plugin>
102             <groupId>org.springframework.boot</groupId>
103             <artifactId>spring-boot-maven-plugin</artifactId>
104             <configuration>
105                 <excludes>
106                     <exclude>
107                         <groupId>org.projectlombok</groupId>
108                         <artifactId>lombok</artifactId>
109                     </exclude>
110                 </excludes>
111             </configuration>

```

```
105</configuration>
106<excludes>
107  <exclude>
108    <groupId>org.projectlombok</groupId>
109    <artifactId>lombok</artifactId>
110  </exclude>
111</excludes>
112</configuration>
113</plugin>
114</plugins>
115</build>
116</project>
117
```

Now below codes are employee

- ▼  Springboot-Employees [boot]
- ▼  src/main/java
 - ▼  com.wipro
 - >  SpringbootEmployeesApplication.java
 - >  com.wipro.controller
 - >  com.wipro.dto
 - >  com.wipro.entity
 - >  com.wipro.repository
 - >  com.wipro.service
- ▼  src/main/resources
 -  static
 -  templates
 -  application.properties
- >  src/test/java
- >  JRE System Library [JavaSE-17]
- >  Maven Dependencies
 -  target/generated-sources/annotations
 -  target/generated-test-sources/test-annotations
- >  src
- >  target
 -  HELP.md
 -  mvnw
 -  mvnw.cmd
 -  pom.xml

SpringbootEmployeesApplication.java

```
1 package com.wipro;
2
3 import org.modelmapper.ModelMapper;
4
5 @SpringBootApplication
6 @EnableFeignClients
7 public class SpringbootEmployeesApplication {
8     public static void main(String[] args) {
9         SpringApplication.run(SpringbootEmployeesApplication.class, args);
10    }
11
12    /*
13     * @Bean public ModelMapper getModelMapper() { return new ModelMapper(); }
14     */
15
16    /*
17     * @Bean public WebClient webClient() { return WebClient.builder().build(); }
18     */
19 }
20
21
22
23
24
```

SpringbootEmployeesApplication.java

EmployeeController.java ×

```
1 package com.wipro.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4
5 @RestController
6 @RequestMapping("/employees")
7 public class EmployeeController {
8
9     @Autowired
10     EmployeeService service;
11
12     @PostMapping
13     public EmployeeDto saveEmployee(@RequestBody EmployeeDto empdto) {
14         return service.saveEmployee(empdto);
15     }
16
17     @GetMapping("/{id}")
18     public EmployeeDepartmentDto getEmployeeById(@PathVariable Long id) {
19         return service.getEmployeeById(id);
20     }
21 }
22
23
24
25
26
27
```

SpringbootEmployeesApplication.java EmployeeController.java

```
1 package com.wipro.dto;
2
3 public class EmployeeDepartmentDto {
4
5     private EmployeeDto empDto;
6     private DepartmentDto depDto;
7     public EmployeeDto getEmpDto() {
8         return empDto;
9     }
10    public void setEmpDto(EmployeeDto empDto) {
11        this.empDto = empDto;
12    }
13    public DepartmentDto getDepDto() {
14        return depDto;
15    }
16    public void setDepDto(DepartmentDto depDto) {
17        this.depDto = depDto;
18    }
19    public EmployeeDepartmentDto(EmployeeDto employeeDto, DepartmentDto depDto) {
20        super();
21        this.empDto = employeeDto;
22        this.depDto = depDto;
23    }
24    public EmployeeDepartmentDto() {
25        super();
26    }
27
28
29 }
30
```

```
SpringbootEmployeesApplication.java EmployeeController.java EmployeeE
1 package com.wipro.dto;
2
3 public class EmployeeDto {
4     private Long empId;
5     private String empName;
6     private String email;
7     private String deptCode;
8     public Long getEmpId() {
9         return empId;
10    }
11    public void setEmpId(Long empId) {
12        this.empId = empId;
13    }
14    public String getEmpName() {
15        return empName;
16    }
17    public EmployeeDto() {
18        super();
19    }
20    public EmployeeDto(Long empId, String empName, String email, String deptCode) {
21        super();
22        this.empId = empId;
23        this.empName = empName;
24        this.email = email;
25        this.deptCode = deptCode;
26    }
27    public void setEmpName(String empName) {
28        this.empName = empName;
29    }
30    public String getEmail() {
31        return email;
32    }
33    public void setEmail(String email) {
34        this.email = email;
35    }
36    public String getDeptCode() {
37        return deptCode;
38    }
39    public void setDeptCode(String deptCode) {
40        this.deptCode = deptCode;
41    }
42
43
44 }
45
```

```
1 package com.wipro.entity;
2
3 import jakarta.persistence.Entity;
4
5
6
7
8 @Entity
9 public class Employee {
10
11     @Id
12     @GeneratedValue(strategy=GenerationType.AUTO)
13     private Long empId;
14     private String empName;
15     private String email;
16     private String deptCode;
17
18     public Employee(Long empId, String empName, String email, String deptCode) {
19         super();
20         this.empId = empId;
21         this.empName = empName;
22         this.email = email;
23         this.deptCode = deptCode;
24     }
25     public Long getEmpId() {
26         return empId;
27     }
28     public void setEmpId(Long empId) {
29         this.empId = empId;
30     }
31     public String getEmpName() {
32         return empName;
33     }
34     public void setEmpName(String empName) {
35         this.empName = empName;
36     }
37     public String getEmail() {
38         return email;
39     }
40     public void setEmail(String email) {
41         this.email = email;
42     }
43     public String getDeptCode() {
44         return deptCode;
45     }
46     public void setDeptCode(String deptCode) {
47         this.deptCode = deptCode;
48     }
49     public Employee() {
50         super();
51     }
52
53
54
55
56
57
58 }
59
```

```

1 package com.wipro.repository;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5 @Repository
6 public interface EmployeeRepo extends JpaRepository<Employee, Long> {
7
8 }
9
10
11
12

```

```

SpringbootEmplo... EmployeeControl... EmployeeDepa
1 package com.wipro.service;
2
3 import org.springframework.cloud.openfeign.FeignClient;
4
5 @FeignClient(name="employee-service", url="http://localhost:8080")
6 public interface APIClient {
7     @GetMapping("/departments/{code}")
8     DepartmentDto getDepartmentByCode(@PathVariable String code);
9 }
10
11
12
13
14

```

```

1 package com.wipro.service;
2
3 import com.wipro.dto.EmployeeDepartmentDto;
4
5 public interface EmployeeService {
6
7     public EmployeeDto saveEmployee(EmployeeDto empdto);
8
9     public EmployeeDepartmentDto getEmployeeById(Long id);
10
11 }
12
13

```



```

EmployeeDep... EmployeeDto... Employee.java EmployeeRepo... APIClient.java
1 package com.wipro.service;
2
3 import org.modelmapper.ModelMapper;
12
13 @Service
14 public class EmployeeServiceImpl implements EmployeeService {
15
16     @Autowired
17     EmployeeRepo repo;
18
19     @Autowired
20     ModelMapper mapper;
21
22     /*
23      * @Autowired WebClient client;
24      */
25
26     @Autowired
27     APIClient apiClient;
28
29     @Override
30     public EmployeeDto saveEmployee(EmployeeDto empdto) {
31         Employee emp = mapper.map(empdto, Employee.class);
32         Employee newEmp = repo.save(emp);
33         return mapper.map(newEmp, EmployeeDto.class);
34     }
35
36     @Override
37     public EmployeeDepartmentDto getEmployeeById(Long id) {
38         Employee employee = repo.findById(id)
39             .orElseThrow(() -> new RuntimeException("Employee not found with ID: " + id));
40
41         DepartmentDto departmentDto = apiClient.getDepartmentByCode(employee.getDeptCode());
42         EmployeeDto employeeDto = mapper.map(employee, EmployeeDto.class);
43
44         return new EmployeeDepartmentDto(employeeDto, departmentDto);
45     }
46 }
47

```

```

EmployeeDto... Employee.java EmployeeRepo... APIClient.java
1 spring.application.name=employee-service
2
3 # MySQL Configuration
4 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
5 spring.datasource.url=jdbc:mysql://localhost:3306/wipro
6 spring.datasource.username=root
7 spring.datasource.password=#Mahadev7
8
9 # JPA & Hibernate
10 server.port=9092
11 spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect
12 spring.jpa.hibernate.ddl-auto=update
13 spring.jpa.show-sql=true
14
15 eureka.client.register-with-eureka=true
16 eureka.client.fetch-registry=true
17 eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka/
18
19

```

Employee.java EmployeeRepo... APIClient.java EmployeeServ... EmployeeServ...

```
https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4 <modelVersion>4.0.0</modelVersion>
5 <parent>
6 <groupId>org.springframework.boot</groupId>
7 <artifactId>spring-boot-starter-parent</artifactId>
8 <version>3.4.3</version>
9 <relativePath/> <!-- lookup parent from repository -->
10 </parent>
11 <groupId>com.wipro</groupId>
12 <artifactId>Springboot-Employees</artifactId>
13 <version>0.0.1-SNAPSHOT</version>
14 <name>Springboot-Employees</name>
15 <description>Demo project for Spring Boot microservice</description>
16 <url/>
17 <licenses>
18 <license/>
19 </licenses>
20 <developers>
21 <developer/>
22 </developers>
23 <scm>
24 <connection/>
25 <developerConnection/>
26 <tag/>
27 <url/>
28 </scm>
29 <properties>
30 <java.version>17</java.version>
31 <spring-cloud.version>2024.0.0</spring-cloud.version>
32 </properties>
33 <dependencies> Add Spring Boot Starters...
34 <!-- Spring Boot Web -->
35 <dependency>
36 <groupId>org.springframework.boot</groupId>
37 <artifactId>spring-boot-starter-web</artifactId>
38 </dependency>
39
40 <!-- Spring Cloud Feign Client -->
41 <dependency>
42 <groupId>org.springframework.cloud</groupId>
43 <artifactId>spring-cloud-starter-openfeign</artifactId>
44 </dependency>
45
46 <!-- Spring Boot Data JPA -->
47 <dependency>
48 <groupId>org.springframework.boot</groupId>
49 <artifactId>spring-boot-starter-data-jpa</artifactId>
50 </dependency>
51 <dependency>
52 <groupId>org.springframework.cloud</groupId>
53 <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
54 </dependency>
55
56 <!-- MySQL Driver -->
```

```

53     <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
54 </dependency>
55
56 <!-- MySQL Driver -->
57 <dependency>
58     <groupId>com.mysql</groupId>
59     <artifactId>mysql-connector-j</artifactId>
60     <scope>runtime</scope>
61 </dependency>
62
63 <!-- ModelMapper -->
64 <dependency>
65     <groupId>org.modelmapper</groupId>
66     <artifactId>modelmapper</artifactId>
67     <version>3.2.0</version>
68 </dependency>
69
70 <!-- Lombok -->
71 <dependency>
72     <groupId>org.projectlombok</groupId>
73     <artifactId>lombok</artifactId>
74     <scope>provided</scope>
75 </dependency>
76
77 <!-- Spring Boot Test -->
78 <dependency>
79     <groupId>org.springframework.boot</groupId>
80     <artifactId>spring-boot-starter-test</artifactId>
81     <scope>test</scope>
82 </dependency>
83 <dependency>
84     <groupId>com.wipro</groupId>
85     <artifactId>Springboot-Department</artifactId>
86     <version>0.0.1-SNAPSHOT</version>
87 </dependency>
88 </dependencies>
89
90 <dependencyManagement>
91 <dependencies>
92 <dependency>
93     <groupId>org.springframework.cloud</groupId>
94     <artifactId>spring-cloud-dependencies</artifactId>
95     <version>${spring-cloud.version}</version>
96     <type>pom</type>
97     <scope>import</scope>
98 </dependency>
99 </dependencies>
100 </dependencyManagement>
101
102 <build>
103 <plugins>
104 <plugin>
105     <groupId>org.apache.maven.plugins</groupId>
106     <artifactId>maven-compiler-plugin</artifactId>
107
















```

```

98     </dependency>
99 </dependencies>
100 </dependencyManagement>
101
102 <build>
103   <plugins>
104     <plugin>
105       <groupId>org.apache.maven.plugins</groupId>
106       <artifactId>maven-compiler-plugin</artifactId>
107       <configuration>
108         <annotationProcessorPaths>
109           <path>
110             <groupId>org.projectlombok</groupId>
111             <artifactId>lombok</artifactId>
112           </path>
113         </annotationProcessorPaths>
114       </configuration>
115     </plugin>
116     <plugin>
117       <groupId>org.springframework.boot</groupId>
118       <artifactId>spring-boot-maven-plugin</artifactId>
119       <configuration>
120         <excludes>
121           <exclude>
122             <groupId>org.projectlombok</groupId>
123             <artifactId>lombok</artifactId>
124           </exclude>
125         </excludes>
126       </configuration>
127     </plugin>
128   </plugins>
129 </build>
130
131 </project>
132

```

Now below are the api gateway codes

- ▼  api-gateway [boot]
 - ▼  src/main/java
 - ▼  com.wipro
 - >  ApiGatewayApplication.java
 - ▼  src/main/resources
 -  application.properties
 - >  src/test/java
 - >  JRE System Library [JavaSE-17]
 - >  Maven Dependencies
 - >  src
 -  target
 -  HELP.md
 -  mvnw
 -  mvnw.cmd
 -  pom.xml

```
1 spring.application.name=api-gateway
2
3 server.port=8085
4
5 eureka.client.register-with-eureka=true
6 eureka.client.fetch-registry=true
7 eureka.client.service-url.defaultZone=http://localhost:8761/eureka/
8
9 spring.cloud.gateway.routes[0].id=EMPLOYEE-SERVICE
10 spring.cloud.gateway.routes[0].uri=lb://EMPLOYEE-SERVICE
11 spring.cloud.gateway.routes[0].predicates[0]=Path=/employees/**
12
13 spring.cloud.gateway.routes[1].id=SPRINGBOOT-DEPARTMENT
14 spring.cloud.gateway.routes[1].uri=lb://SPRINGBOOT-DEPARTMENT
15 spring.cloud.gateway.routes[1].predicates[0]=Path=/departments/**
16
17
18 spring.cloud.gateway.discovery.locator.enabled=true
19 spring.cloud.gateway.discovery.locator.lower-case-service-id=true
20
21
```

```

https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>3.4.3</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>com.wipro</groupId>
12  <artifactId>api-gateway</artifactId>
13  <version>0.0.1-SNAPSHOT</version>
14  <name>api-gateway</name>
15  <description>Demo project for Spring Boot microservice api gateway</description>
16  <url/>
17  <licenses>
18    <license/>
19  </licenses>
20  <developers>
21    <developer/>
22  </developers>
23  <scm>
24    <connection/>
25    <developerConnection/>
26    <tag/>
27    <url/>
28  </scm>
29  <properties>
30    <java.version>17</java.version>
31    <spring-cloud.version>2024.0.0</spring-cloud.version>
32  </properties>
33  <dependencies>
34    <!-- Add Spring Boot Starters... -->
35    <dependency>
36      <groupId>org.springframework.cloud</groupId>
37      <artifactId>spring-cloud-starter-gateway</artifactId>
38    </dependency>
39    <dependency>
40      <groupId>com.github.ben-manes.caffeine</groupId>
41      <artifactId>caffeine</artifactId>
42    </dependency>
43    <dependency>
44      <groupId>org.springframework.cloud</groupId>
45      <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
46    </dependency>
47    <dependency>
48      <groupId>org.springframework.boot</groupId>
49      <artifactId>spring-boot-starter-test</artifactId>
50      <scope>test</scope>
51    </dependency>
52  </dependencies>
53  <dependencyManagement>
54    <dependencies>
55      <dependency>

```


```

53     </dependencies>
54     <dependencyManagement>
55         <dependencies>
56             <dependency>
57                 <groupId>org.springframework.cloud</groupId>
58                 <artifactId>spring-cloud-dependencies</artifactId>
59                 <version>${spring-cloud.version}</version>
60                 <type>pom</type>
61                 <scope>import</scope>
62             </dependency>
63         </dependencies>
64     </dependencyManagement>
65
66     <build>
67         <plugins>
68             <plugin>
69                 <groupId>org.springframework.boot</groupId>
70                 <artifactId>spring-boot-maven-plugin</artifactId>
71             </plugin>
72         </plugins>
73     </build>
74
75 </project>
76

```

Now run the main files one by one like register main file then department main file then employee main file then api gateway main file...

Finally our output in the eureka is


HOME LAST 1000 SINCE STARTUP

System Status

Environment	test	Current time	2025-03-03T09:25:53 +0530
Data center	default	Uptime	00:01
		Lease expiration enabled	false
		Renews threshold	6
		Renews (last min)	2

DS Replicas

localhost

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
API-GATEWAY	n/a (1)	(1)	UP (1) - 192.168.1.12:api-gateway:8085
EMPLOYEE-SERVICE	n/a (1)	(1)	UP (1) - 192.168.1.12:employee-service:9092
SPRINGBOOT-DEPARTMENT	n/a (1)	(1)	UP (1) - 192.168.1.12:Springboot-Department:9090

General Info

General Info

Name	Value
total-avail-memory	202mb
num-of-cpus	12
current-memory-usage	112mb (55%)
server-uptime	00:01
registered-replicas	http://localhost:8761/eureka/
unavailable-replicas	http://localhost:8761/eureka/,
available-replicas	

Instance Info

Name	Value
ipAddr	192.168.1.12
status	UP