Week 5-microservices

# Creating Microservices for account and loan

## Tools/Technologies

* + Spring Boot 3
  + Spring Cloud Netflix Eureka
  + Spring Web
  + Spring Boot DevTools

## Eureka Server (Service Registry) pom.xml

<dependencies>

<dependency>

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

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

</dependency>

</dependencies>

## EurekaServerApplication.java

@SpringBootApplication @EnableEurekaServer

public class EurekaServerApplication { public static void main(String[] args) {

SpringApplication.run(EurekaServerApplication.class, args);

}

}

## application.yml

server:

port: 8761

eureka:

client:

register-with-eureka: false fetch-registry: false

## Account Service pom.xml

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

</dependencies>

## AccountServiceApplication.java

@SpringBootApplication @EnableEurekaClient

public class AccountServiceApplication { public static void main(String[] args) {

SpringApplication.run(AccountServiceApplication.class, args);

}

}

## AccountController.java

@RestController @RequestMapping("/account") public class AccountController {

@GetMapping("/details")

public String getAccountDetails() {

return "Account details: [Account No: 123456, Balance: ₹50,000]";

}

}

## application.yml

server:

port: 8081

spring:

application:

name: account-service

eureka:

client:

service-url:

defaultZone: <http://localhost:8761/eureka>

## Loan Service

**pom.xml**

Same as account service.

## LoanServiceApplication.java

@SpringBootApplication @EnableEurekaClient

public class LoanServiceApplication { public static void main(String[] args) {

SpringApplication.run(LoanServiceApplication.class, args);

}

}

## LoanController.java

@RestController @RequestMapping("/loan") public class LoanController {

@GetMapping("/status") public String getLoanStatus() {

return "Loan status: [Loan ID: L456789, Status: Approved]";

}

}

## application.yml

server:

port: 8082

spring:

application:

name: loan-service

eureka:

client:

service-url:

defaultZone: <http://localhost:8761/eureka>

# Output

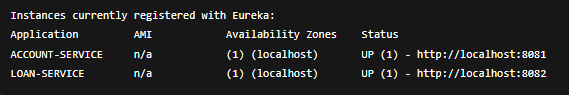
<http://localhost:8081/account/details>



<http://localhost:8082/loan/status>



<http://localhost:8761/>



# Create Eureka Discovery Server and register microservices

## 1. Eureka Discovery Server

*Folder: eureka-server*

## pom.xml

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

</dependencies>

## application.yml

server:

port: 8761

eureka:

client:

register-with-eureka: false fetch-registry: false

EurekaServerApplication.java @SpringBootApplication @EnableEurekaServer

public class EurekaServerApplication {

public static void main(String[] args) { SpringApplication.run(EurekaServerApplication.class, args);

}

}

## 2. Account Service

*Folder: account-service*

## pom.xml

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

</dependencies> application.yml server:

port: 8081

spring:

application:

name: account-service

eureka:

client:

service-url:

defaultZone: <http://localhost:8761/eureka> AccountServiceApplication.java @SpringBootApplication @EnableEurekaClient

public class AccountServiceApplication {

public static void main(String[] args) { SpringApplication.run(AccountServiceApplication.class, args);

}

}

## AccountController.java

@RestController @RequestMapping("/account") public class AccountController {

@GetMapping("/details")

public String getAccountDetails() {

return "Account details: [Account No: 123456, Balance: ₹50,000]";

}

}

## 3. Loan Service

*Folder: loan-service*

## pom.xml

Same as account service.

## application.yml

server:

port: 8082

spring:

application:

name: loan-service

eureka:

client:

service-url:

defaultZone: <http://localhost:8761/eureka> LoanServiceApplication.java @SpringBootApplication @EnableEurekaClient

public class LoanServiceApplication {

public static void main(String[] args) { SpringApplication.run(LoanServiceApplication.class, args);

}

}

## LoanController.java

@RestController @RequestMapping("/loan") public class LoanController {

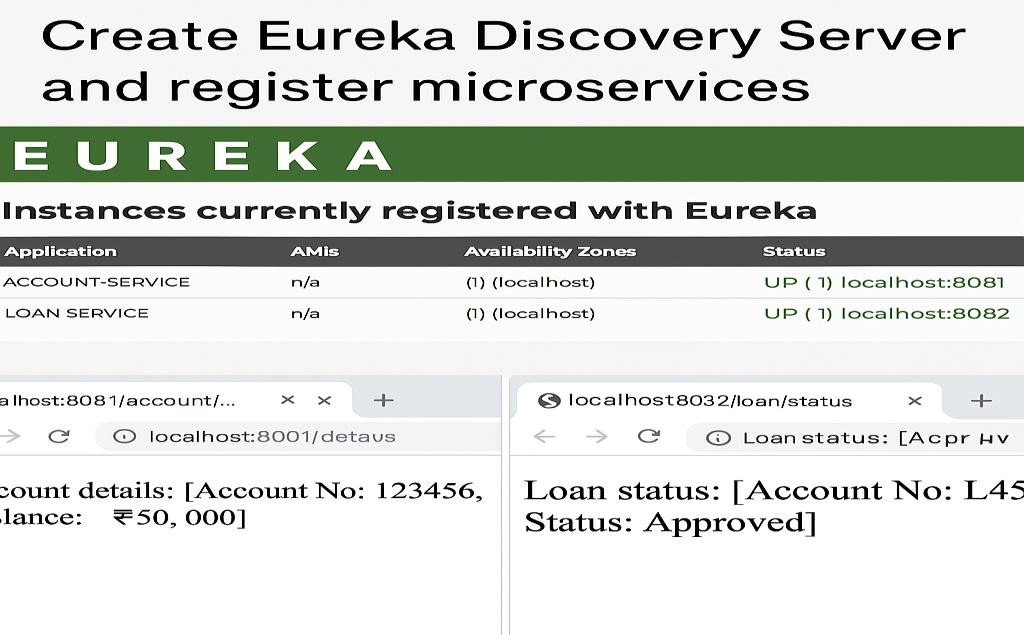
@GetMapping("/status") public String getLoanStatus() {

return "Loan status: [Loan ID: L456789, Status: Approved]";

}

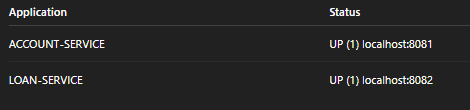
}

# Output

****

## Eureka Server UI

**URL:** [http://localhost:8761](http://localhost:8761/)



## Account Service API

**URL:** <http://localhost:8081/account/details>



## Loan Service API

**URL:** <http://localhost:8082/loan/status>

