WEEK – 5

JAVA FSE

1) Creating Microservices for account and loan

Tools/Technologies

- Spring Boot 3
- Spring Cloud Netflix Eureka
- Spring Web
- Spring Boot DevTools

1. 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
```

2. 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
```

3. 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

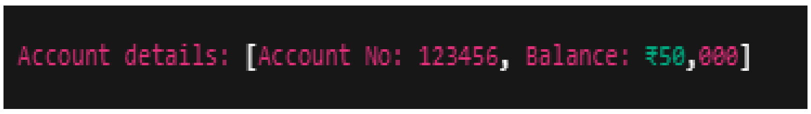
```
server:
  port: 8082

spring:
  application:
    name: loan-service

eureka:
  client:
    service-url:
      defaultZone: http://localhost:8761/eureka
```

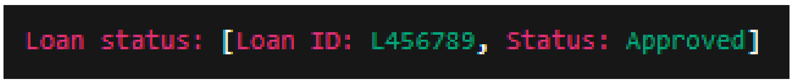
Output

http://localhost:8081/account/details



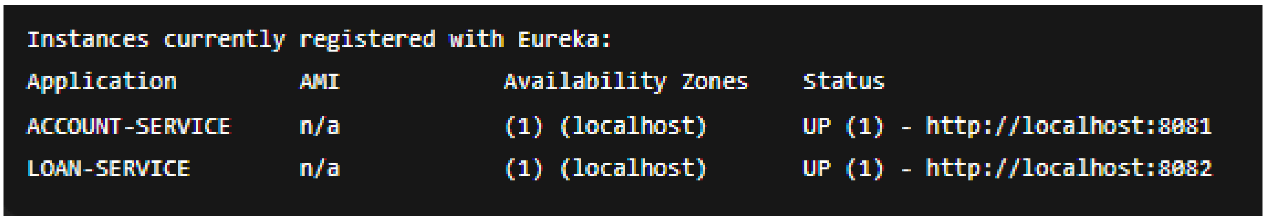
```
Account details: [Account No: 123456, Balance: ₹50,000]
```

http://localhost:8082/loan/status



```
Loan status: [Loan ID: L456789, Status: Approved]
```

http://localhost:8761/



```
Instances currently registered with Eureka:
Application         AMI            Availability Zones    Status
ACCOUNT-SERVICE     n/a            (1) (localhost)       UP (1) - http://localhost:8081
LOAN-SERVICE        n/a            (1) (localhost)       UP (1) - http://localhost:8082
```

2) 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

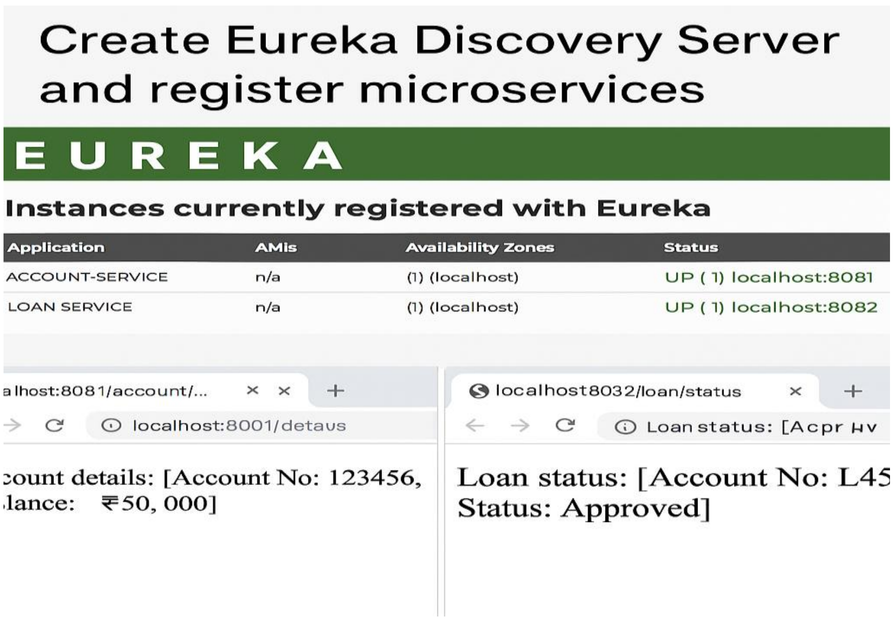
```
@SpringBootApplication
@EnableEurekaClient
public class LoanServiceApplication {
    public static void main(String[] args) {
        SpringApplication.run(LoanServiceApplication.class, args);
    }
}
```

LoanController.java

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

    @GetMapping("/status")
    public String getLoanStatus() {
        return "Loan status: [Loan ID: L456789, Status: Approved]";
    }
}
```

Output



Create Eureka Discovery Server and register microservices

EUREKA

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
ACCOUNT-SERVICE	n/a	(1) (localhost)	UP (1) localhost:8081
LOAN SERVICE	n/a	(1) (localhost)	UP (1) localhost:8082

a lhost:8081/account/...

localhost:8001/detaus

```
count details: [Account No: 123456,
lance:   ₹50, 000]
```

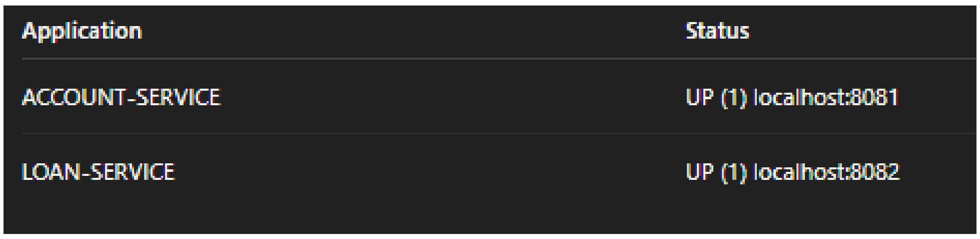
localhost8032/loan/status

Loan status: [Acpr ʜv

```
Loan status: [Account No: L45
Status: Approved]
```

Eureka Server UI

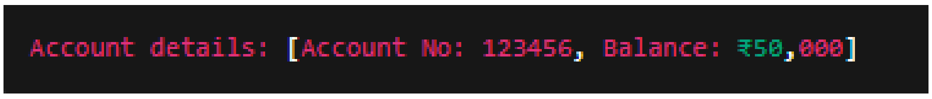
URL: http://localhost:8761



Application	Status
ACCOUNT-SERVICE	UP (1) localhost:8081
LOAN-SERVICE	UP (1) localhost:8082

Account Service API

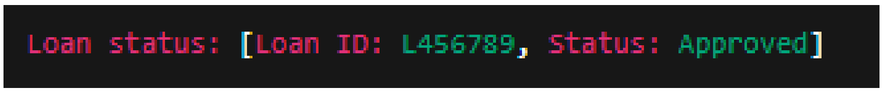
URL: http://localhost:8081/account/details



```
Account details: [Account No: 123456, Balance: ₹50,000]
```

Loan Service API

URL: http://localhost:8082/loan/status



```
Loan status: [Loan ID: L456789, Status: Approved]
```