**Microservices with Spring Boot 3 and Spring Cloud**

2. Microservices with API gateway

**Creating Microservices for account and loan**

**Account Microservice**

**Group**: com.cognizant

**Artifact**: account

**Dependencies:** Spring Boot DevTools**,** Spring Web

**CODE:**

Controller Code for Account Service (AccountController.java)

package com.cognizant.account.controller;

import org.springframework.web.bind.annotation.\*;

@RestController

@RequestMapping("/accounts")

public class AccountController {

@GetMapping("/{number}")

public Account getAccountDetails(@PathVariable String number) {

// Dummy response

return new Account(number, "savings", 234343);

}

}

**Account.java**

package com.cognizant.account.model;

public class Account {

private String number;

private String type;

private double balance;

public Account(String number, String type, double balance) {

this.number = number;

this.type = type;

this.balance = balance;

}

// Getters and Setters

public String getNumber() { return number; }

public String getType() { return type; }

public double getBalance() { return balance; }

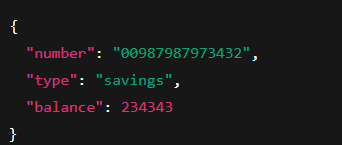
public void setNumber(String number) { this.number = number; }

public void setType(String type) { this.type = type; }

public void setBalance(double balance) { this.balance = balance; }

}

**RESPONSE:**



**Loan Microservice**

**Group**: com.cognizant

**Artifact**: loan

**CODE:**

**Controller Code for Loan Service (LoanController.java)**

package com.cognizant.loan.controller;

import org.springframework.web.bind.annotation.\*;

@RestController

@RequestMapping("/loans")

public class LoanController {

@GetMapping("/{number}")

public Loan getLoanDetails(@PathVariable String number) {

// Dummy response

return new Loan(number, "car", 400000, 3258, 18);

}

}

**Loan.java**

package com.cognizant.loan.model;

public class Loan {

private String number;

private String type;

private double loan;

private double emi;

private int tenure;

public Loan(String number, String type, double loan, double emi, int tenure) {

this.number = number;

this.type = type;

this.loan = loan;

this.emi = emi;

this.tenure = tenure;

}

// Getters and Setters

public String getNumber() { return number; }

public String getType() { return type; }

public double getLoan() { return loan; }

public double getEmi() { return emi; }

public int getTenure() { return tenure; }

public void setNumber(String number) { this.number = number; }

public void setType(String type) { this.type = type; }

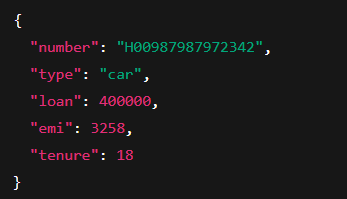
public void setLoan(double loan) { this.loan = loan; }

public void setEmi(double emi) { this.emi = emi; }

public void setTenure(int tenure) { this.tenure = tenure; }

}

**RESPONSE:**



**Create Eureka Discovery Server and register microservices**

Eureka Discovery Server holds a registry of all the services that are available for immediate consumption. Anybody whom wants to consume a RESTful Web Service can come to the discovery server and find out what is available and ready for consumption. Eureka Discovery Server is part of spring cloud module.

**1.Creating Eureka Discovery Server**

**Group**: com.cognizant

**Artifact**: eureka-discovery-server

**Spring Cloud Discovery > Eureka Server**

**EurekaDiscoveryServerApplication.java**

package com.cognizant.eurekaserver;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.cloud.netflix.eureka.server.EnableEurekaServer;

@SpringBootApplication

@EnableEurekaServer

public class EurekaDiscoveryServerApplication {

public static void main(String[] args) {

SpringApplication.run(EurekaDiscoveryServerApplication.class, args);

}

}

**Configure application.properties:**

server.port=8761

eureka.client.register-with-eureka=false

eureka.client.fetch-registry=false

logging.level.com.netflix.eureka=OFF

logging.level.com.netflix.discovery=OFF

**2.Registering Account Microservice to Eureka Server**

**Added Discovery Client to Main Class (AccountApplication.java)**

package com.cognizant.account;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

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

@SpringBootApplication

@EnableDiscoveryClient

public class AccountApplication {

public static void main(String[] args) {

SpringApplication.run(AccountApplication.class, args);

}

}

**Properties:**

server.port=8080

spring.application.name=account-service

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

**3.Registering Loan Microservice to Eureka Server**

**LoanApplication.java**

package com.cognizant.loan;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

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

@SpringBootApplication

@EnableDiscoveryClient

public class LoanApplication {

public static void main(String[] args) {

SpringApplication.run(LoanApplication.class, args);

}

}

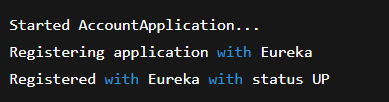
**Propertires:**

server.port=8081

spring.application.name=loan-service

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

OUTPUT:



**Final Confirmation Summary:**

**Component Port Status**

Eureka Server 8761 UP

Account Microservice 8080 UP

Loan Microservice 8081 UP