Week 5:

Microservices with API gateway

Creating Microservices for account and loan:

Account:

**AccountController.java:**

package com.cognizant.account;  
  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
@RequestMapping("/accounts")  
public class AccountController {  
  
 @GetMapping("/{number}")  
 public Account getAccount(@PathVariable String number) {  
 return new Account(number, "savings", 2343.0);  
 }  
  
 static 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;  
 }  
  
 public String getNumber() { return number; }  
 public String getType() { return type; }  
 public double getBalance() { return balance; }  
 }  
}

**AccountApplication.java:**

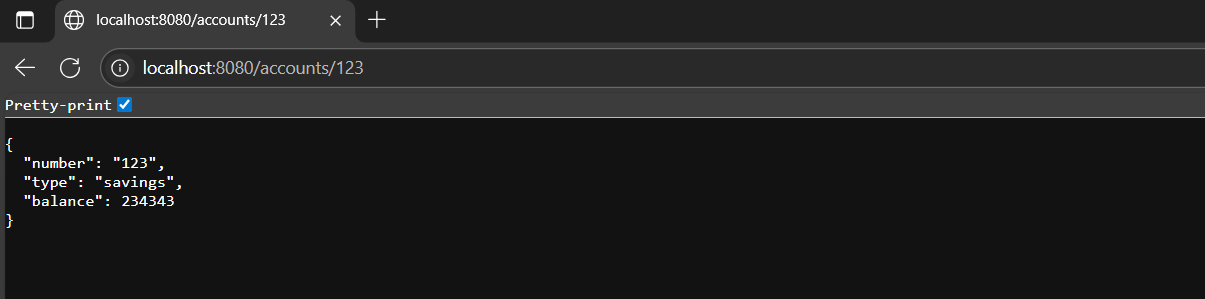
package com.cognizant.account;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;  
  
@SpringBootApplication  
@EnableDiscoveryClient  
public class AccountApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(AccountApplication.class, args);  
 }  
}

**application.properties:**

spring.application.name=account  
server.port=8080  
eureka.client.service-url.defaultZone=http://localhost:8761/eureka/  
spring.cloud.compatibility-verifier.enabled=false

// The extra we add is corresponding maven dependencies into pom.xml file

Output:



Loan:

**LoanController.java:**

package com.cognizant.loan;  
  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
@RequestMapping("/loans")  
public class LoanController {  
  
 @GetMapping("/{number}")  
 public Loan getLoan(@PathVariable String number) {  
 return new Loan(number, "car", 400000.0, 3258, 18);  
 }  
  
 static class Loan {  
 private String number;  
 private String type;  
 private double loan;  
 private int emi;  
 private int tenure;  
  
 public Loan(String number, String type, double loan, int emi, int tenure) {  
 this.number = number;  
 this.type = type;  
 this.loan = loan;  
 this.emi = emi;  
 this.tenure = tenure;  
 }  
  
 public String getNumber() { return number; }  
 public String getType() { return type; }  
 public double getLoan() { return loan; }  
 public int getEmi() { return emi; }  
 public int getTenure() { return tenure; }  
 }  
}

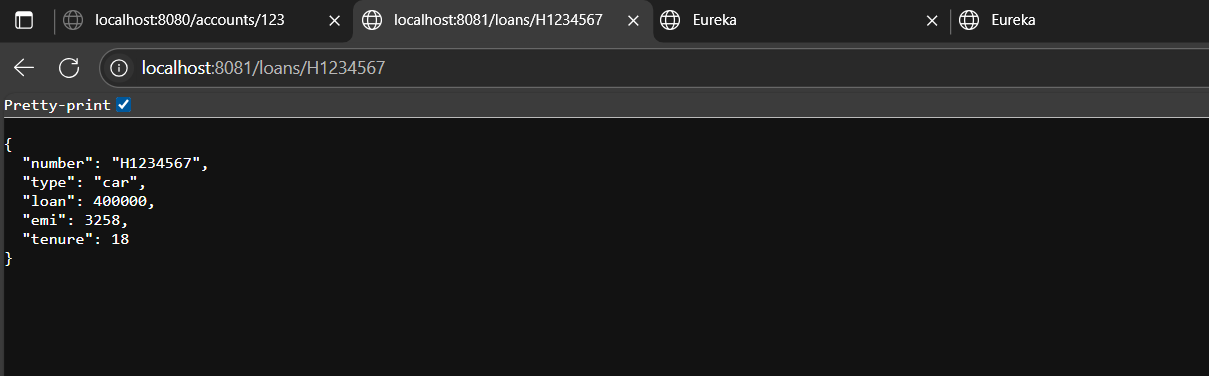
**LoanApplication.java:**

package com.cognizant.loan;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;  
  
@EnableDiscoveryClient  
@SpringBootApplication  
public class LoanApplication {  
  
 public static void main(String[] args) {  
  
 SpringApplication.*run*(LoanApplication.class, args);  
 }  
  
}

**application.properties:**

spring.application.name=loan  
server.port=8081  
eureka.client.service-url.defaultZone=http://localhost:8761/eureka/

Output:



Eureka:

**EurekaDiscoveryServerApplication.java:**

package com.cognizant.eureka;  
  
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);  
 }  
}

**application.properties:**

spring.application.name=eureka-discovery-server  
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  
eureka.server.enable-self-preservation=false  
eureka.server.eviction-interval-timer-in-ms=5000

Output: 