Microservices Assignment - Code and Output (Excluding Account & Loan)

# Exercise 1: Customer Microservice

Code (Customer.java):

public class Customer {  
 private int id;  
 private String name;  
 private String email;  
 // constructors, getters, setters  
}

Code (CustomerController.java):

@RestController  
@RequestMapping("/customers")  
public class CustomerController {  
 List<Customer> customerList = new ArrayList<>();  
  
 @GetMapping("/{id}")  
 public Customer getCustomer(@PathVariable int id) {  
 return customerList.stream().filter(c -> c.getId() == id).findFirst().orElse(null);  
 }  
  
 @PostMapping  
 public String addCustomer(@RequestBody Customer customer) {  
 customerList.add(customer);  
 return "Customer added";  
 }  
  
 @DeleteMapping("/{id}")  
 public String deleteCustomer(@PathVariable int id) {  
 customerList.removeIf(c -> c.getId() == id);  
 return "Customer deleted";  
 }  
}

Sample Output:

POST /customers with JSON: {"id":1,"name":"Alice","email":"alice@email.com"} → Customer added  
GET /customers/1 → {"id":1,"name":"Alice","email":"alice@email.com"}

# Exercise 2: Integrate with Loan Service

Code (CustomerWithLoans.java):

public class CustomerWithLoans {  
 private Customer customer;  
 private List<Loan> loans;  
}

Code (Integration):

@GetMapping("/customer-loans/{id}")  
public CustomerWithLoans getLoans(@PathVariable int id) {  
 Customer customer = ... // fetch from list or DB  
 List<Loan> loans = restTemplate.getForObject("http://localhost:8082/loan/customer/" + id, List.class);  
 return new CustomerWithLoans(customer, loans);  
}

Sample Output:

GET /customer-loans/1 → {"customer":{"id":1,"name":"Alice"}, "loans":[{"loanId":101,"amount":5000}]}

# Exercise 3: Eureka Discovery

Eureka Server Main Class:

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

application.properties (Server):

server.port=8761  
eureka.client.register-with-eureka=false  
eureka.client.fetch-registry=false

application.properties (Clients):

spring.application.name=customer-service  
eureka.client.service-url.defaultZone=http://localhost:8761/eureka

Output: Services appear on http://localhost:8761

# Exercise 4: API Gateway

application.yml:

spring:  
 cloud:  
 gateway:  
 routes:  
 - id: account  
 uri: http://localhost:8081  
 predicates:  
 - Path=/account/\*\*  
 - id: loan  
 uri: http://localhost:8082  
 predicates:  
 - Path=/loan/\*\*

Output: /account/\*\* and /loan/\*\* routes pass through gateway.

# Exercise 5: Resilience4j Circuit Breaker

Code:

@CircuitBreaker(name = "loanService", fallbackMethod = "fallbackLoan")  
public List<Loan> getLoans(@PathVariable int id) {  
 return restTemplate.getForObject("http://loan-service/loan/customer/" + id, List.class);  
}  
  
public List<Loan> fallbackLoan(int id, Throwable t) {  
 return List.of(new Loan(0, "Fallback Loan"));  
}

Output: If loan service fails, fallback method returns a default loan.