**Exercise 1: User and Order Management System**

**Problem:**  
Build two microservices:

* **User Service:** Manage users (CRUD)
* **Order Service:** Manage orders linked to users

**Requirements:**

* Use **REST APIs**
* Communication between services via **WebClient (Spring WebFlux)** or **OpenFeign**
* Use **MySQL** or **PostgreSQL** for data persistence

Example:

* UserService: /users, /users/{id}
* OrderService: /orders, /orders/user/{userId}
* FeignClient or WebClient inside OrderService to get user details.

**Exercise 2: Inventory Management with Service Discovery**

**Problem:**  
Build:

* **Product Service:** CRUD for product items
* **Inventory Service:** Maintain stock per product

**Requirements:**

* Register services using **Spring Cloud Netflix Eureka**
* Use **Spring Cloud Config Server** for centralized configuration

Example:

* ProductService: /products, /products/{id}
* InventoryService: /inventory/{productId}
* Use @EnableEurekaClient, and connect to Config Server

**Exercise 3: API Gateway with Routing and Filters**

**Problem:**  
Create an API Gateway for:

* **Customer Service**
* **Billing Service**

**Requirements:**

* Use **Spring Cloud Gateway**
* Implement:
  + **Rate limiting**
  + **Caching**
  + **Path rewriting**

Example application.yml for routes:

spring:

cloud:

gateway:

routes:

- id: customer\_route

uri: lb://customer-service

predicates:

- Path=/customer/\*\*

filters:

- RewritePath=/customer/(?<segment>.\*), /${segment}

- RequestRateLimiter=...

- id: billing\_route

uri: lb://billing-service

predicates:

- Path=/billing/\*\*

- name: CircuitBreaker

args:

name: billingCB

fallbackUri: forward:/fallback

**Exercise 4: Resilient Payment Service with Circuit Breaker**

**Problem:**  
Build a **Payment Service** that calls a **slow 3rd-party API**.

**Requirements:**

* Use **Resilience4j** for:
  + **Circuit Breaker**
  + **Fallback logic**
* Log and monitor fallback events

Example:

@CircuitBreaker(name = "paymentService", fallbackMethod = "fallback")

public String callThirdPartyApi() {

return restTemplate.getForObject("https://slow.api.com", String.class);

}

public String fallback(Throwable t) {

log.warn("Fallback executed due to: {}", t.getMessage());

return "Fallback response";

}