Spring Microservices Exercises

Contents

Exercise 1: Exercise

Time Limit: 60 Minutes

Problem Statement

Convert the below requirements of our InfyTel case study to a microservice

- The call details has a 'single-responsibility' of only dealing with call details of a given customer.
- The plan has a 'single-responsibility' of only dealing with plan related functionality of getting all plans and getting a specific plan.
- The friend family has a 'single-responsibility' of only dealing with adding and retrieving friends.
- The customer has a 'single-responsibility' of only dealing with customer functionality like login, view profile, register

In our session we have implemented InfyTel-Customer and InfyTel-FriendFamily as Microservices.

Now convert the below two as Micro services

1. infytel-calldetails: The **call details** has a 'single-responsibility' of only dealing with call details of a given customer
2. infytel-plans: The **plan** has a 'single-responsibility' of only dealing with plan related functionality of getting all plans and getting a specific plan.

For this exercise, convert the monolithic implementation to a microservices, each talking to an individual schema/database. You can test your app through Postman/SoapUI.

Exercise 2: Config

Time Limit: 20 Minutes

Problem Statement

Incremental exercise on InfyTel:

For this exercise, create config server talking to git and make all the microservices get their properties from the config server..

Exercise 3: Ribbon Exercise

Time Limit: 30 Minutes

Problem Statement

Incremental exercise on InfyGit

Create two instances of " infytel-calldetails " service and invoke the same from the "InfyTel-Customer" service using ribbon load balancer.

Exercise 4: Service Discovery

Time Limit: 30 Minutes

Problem Statement

Incremental exercise on InfyTel:

Register all the services with Eureka. Deploy all the services in random ports and run the application.

Exercise 5: Eureka Exercise

Time Limit: 30 Minutes

Problem Statement

Incremental Exercise on InfyTel:

Modify the ribbon client to use Eureka. Also use RandomRule for load balancing..

Exercise 6: Hystrix

Time Limit: 120 Minutes

Problem Statement

Incremental Exercise on InfyTel:

Add resiliency to your application by using hystrix for all inter-service communications and all the database communications.

Bring down one of the dependent microservices and observe whether it is handled by appropriate fallback. If more than 30% of requests sent in 10 seconds fail, with a minimum of 20 requests sent in that duration, open the circuit and return dummy values. Try closing the circuit again after 1 min.

Exercise 7: Zuul Exercise

Time Limit: 30 Minutes

Problem Statement

Incremental Exercise on InfyTel:

Add a Zuul API Gateway service so that all requests are forwarded through Zuul. Also, all requests to zuul must be prefixed with "infyTel". Handle the prefix using Zuul prefix property.

Exercise 8: Feign Exercise

Time Limit: 30 Minutes

Problem Statement

Incremental Exercise on InfyTel:

Replace all Rest template objects with Feign Clients.Configure the below details in application.yml:

Exercise 9: Security

Time Limit: 60 Minutes

Problem Statement

Incremental Exercise on InfyTel:

Secure all the microservices, the API gateway and the Config server.