Microservices

Microservices are small, autonomous services that work together, and The Microservices Architectural Style is an approach to develop

a Single application as a Suite of small Services, each running in its own process and communicating with light weight mechanism.

Some of the benefits of Microservices application compared to Monolithic applications are:

* Easy to understand​
* No tight coupling between components​
* New components can be easily built and deployed​
* Improve Fault tolerance and its isolation​
* Eliminates long term dependency on a particular tech stack​
* Individual components can be auto-scaled​
* Overall application access unaffected on failure of a component/service

To make an application with the microservices architecture, some of the key components required are

1. **Service Discovery**: For which we have used **Eureka**. There is another service available for this purpose named Apache Zookeeper.
2. **Client side load balancing**: For which we have used Ribbon.
3. **Fault Tolerance**: If a service fails and need to redirect to another service, we have used **Hystrix**.
4. **API Gateway**: As a wrapper for all the client side services, so as to avoid exposing the hosted address and port numbers of the services for which we have used **Zuul**.

**Steps to develop the Microservices using Spring boot**

1. Download and Install Spring Tool Suite
2. Create TOINews spring starter project
3. Add devtools and web dependency
4. Configure the application.properties file
5. Write TOINewsController using @RestController annotation
6. Write a method which will return toi news. Use @GetMapping and add URL
7. TOINews Service is ready.It can be invoked by using postman
8. Now create MiddayNews service by following steps 2,3,4,5.Test the service using postman
9. Create NewsApp service using Spring starter project. Add controller and mapping
10. Write NewsAppService. Use Hytrix annotation to have fallback method. i.e. In case TOIservice is down then MiddayService is invoked by the newsApi call to give news seamlessly to the clients
11. Create Eureka service. Add Eureka server dependency to the project
12. In the main class of Spring boot Eureka application add @EurekaServer annotation
13. Add Eurekaclient dependency in the pom.xml of TOINewsService and MiddayService
14. Add @EnableEurekaClient annotation
15. Start Eurka service and request <http://localhost:8657> on the browser.This will give Eureka server dashboard.This acts as a service repository
16. Name to TOIService and MiddayService in application.properties files by using the parameter spring.application.name= the name of the service
17. Start TOIService and MiddayService
18. Refresh Eureka server repository on the browser
19. Now TOIService and Middayservice can be seen on Eureka Dashboard
20. Make use of @LoadBalanced annotation in main class of NewsApp service above getRestTemplate method for load balancing
21. Create the multiple instances of TOIService or Middayservice using run configuration in STS tool by duplicating the services. Here we can specify the different ports
22. Write Zuul Service using Spring starter project.Add Zuul and Eureka client dependency in the project.
23. Add @EnableZuulProxy annotation in the main class
24. Configure the application.properties.Using Zuul properties

Application is ready with Microservices