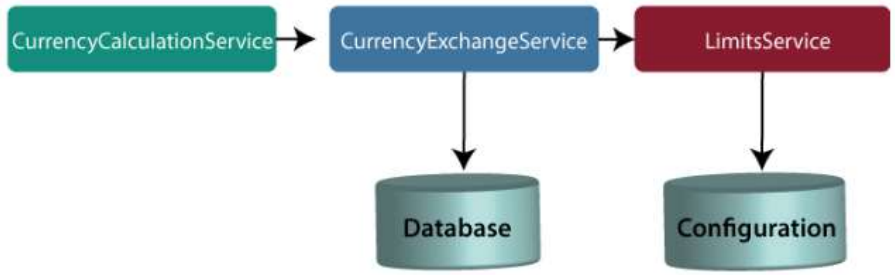
MasterMicroservice_CourseSelf Notes

21 March 2024 00:45

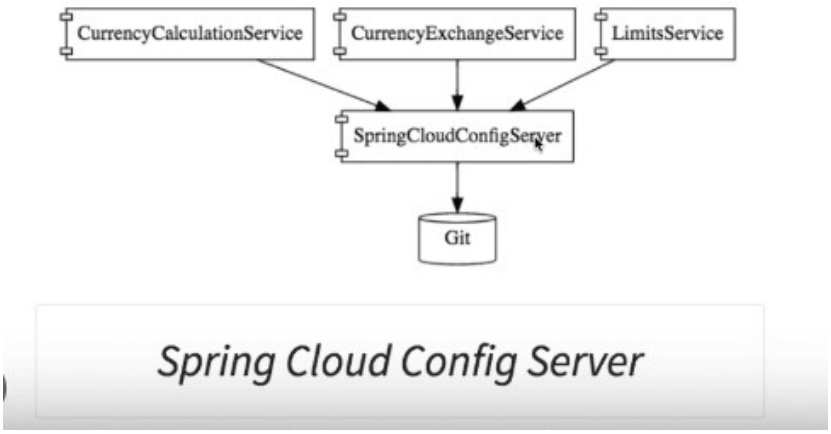
Ports Standarization

- spring cloud config server = 8888
- limits-service = 8080
- currency exchange service = 8000, 8001, 8002,....
- currency conversion service = 8100, 8101, 8102,....
- Eureka naming-server = 8761
- API-Gateway = 8765

Microservice Base layout of project

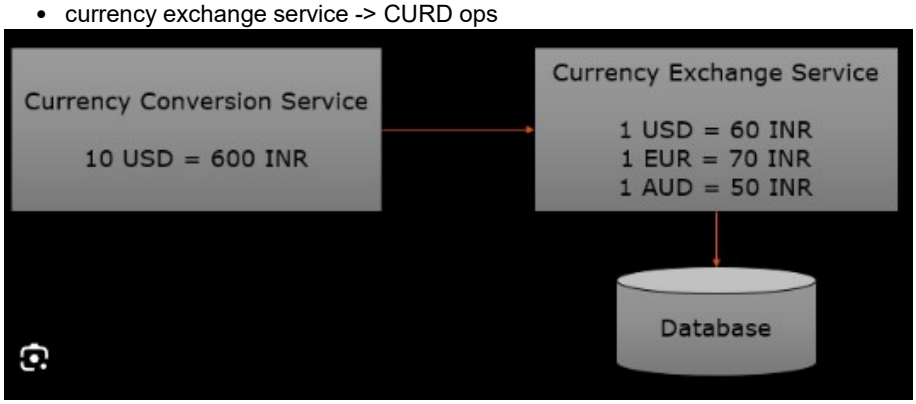


Step 0 - Initialize the local-git-repo for Spring-Cloud-Config-Server

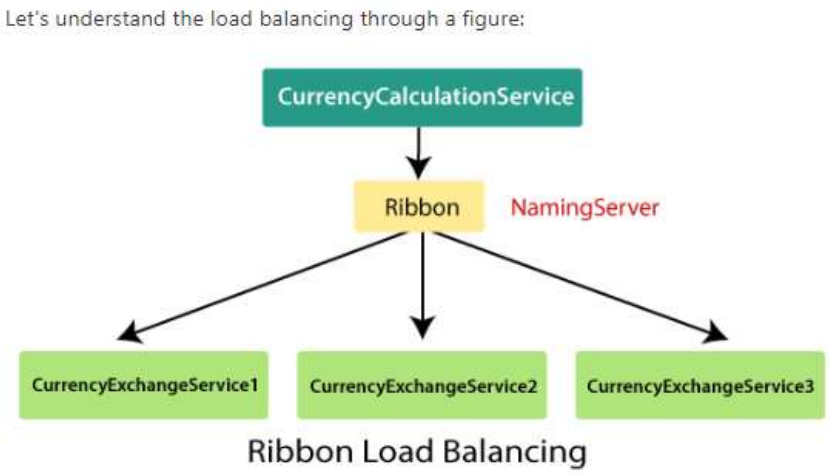


Step - 1 - Base Layout of two microservices

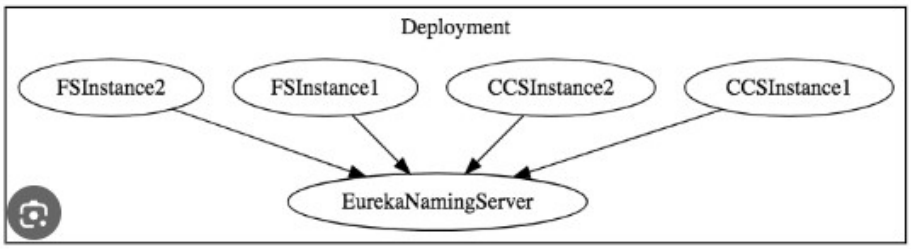
- currency conversion service
- currency exchange service -> CURD ops



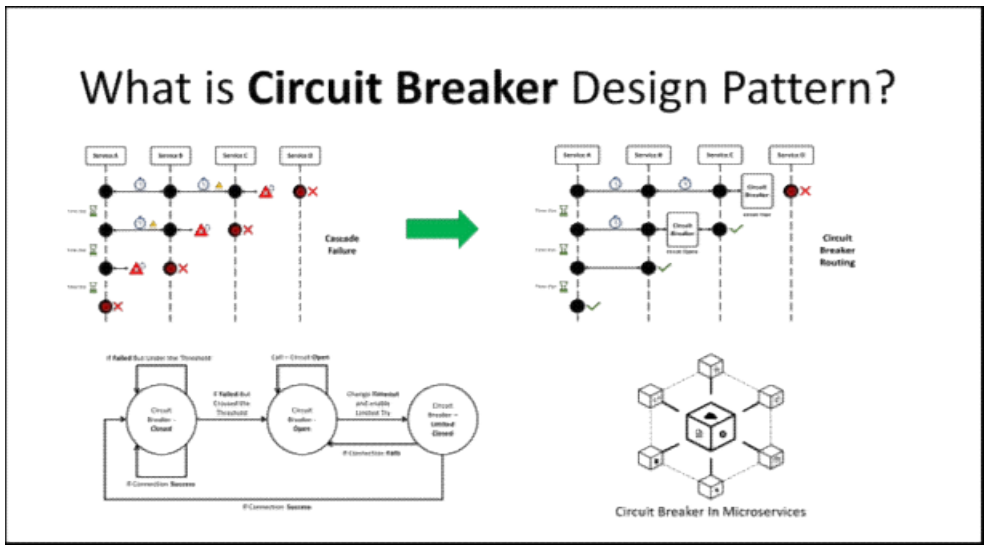
Step - 2 - Load Balancing



Step - 3 - Naming Server to get upInstance details



Step - 4 - Circuit Breaker - Resilience



Steps I have follower -

- Centralized Configuration
  a. Created Limit-service

Desc - It has configuration values that we are fetching from Spring cloud config server
Basic min and max value

- After perform **step-2,** Check github code for same module. Now it will fetch the values from git repo files and not from application.properties.

application.properties	##Specifyingtheportwhereconfigserverwillrun spring.config.import=optional:configserver:http://localhost:8888 spring.application.name=limits-service #limits-service.minimum=2 #limits-service.maximum=998 spring.profiles.active=dev

b. Spring cloud Config server -

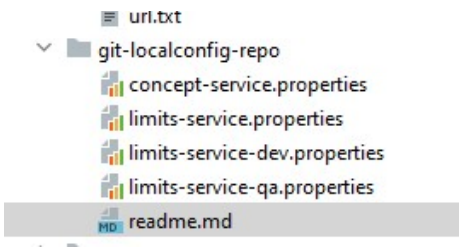
application.properties	spring.application.server=spring-cloud-config-server server.port=8888 **spring.cloud.config.server.git.uri=file:///E:/Project/Java/springBoot/SpringBoot_Microservices/git-localconfig-repo**

```
Gauty
@EnableConfigServer
@SpringBootApplication
public class SpringCloudConfigServerApplication {

    Gauty
    public static void main(String[] args) {
        SpringApplication.run(SpringCloudConfigServerApplication.class, args);
    }
}
```

Desc -

Intialize the git-loacl-repo to access the configuration present inside git repo

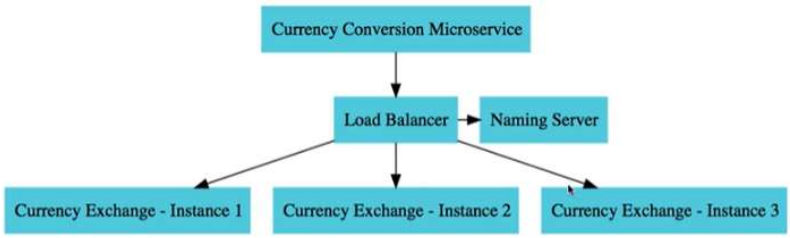


We have two service

- currency conversion service
- currency exchange service -> CURD ops

- (Eureka Naming server) Make new springBoot project of naming server - On this server all service will register themselves
  - @EnableEurekaServer (annotation used in main class - @SpringBootApplicaion)

- (Register the Service on naming server)Add the client dependency in above two service
  - | Application.properties | eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka |



Load Balancing

- (API - Gateway) - It will also register itself on Eureka Naming Server like in above steps
  - application.properties-

```
spring.application.name=api-gateway
server.port=8765
eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka
spring.config.import=optional:configserver:

#Todiscoverytheserviceovertheserver
spring.cloud.gateway.discovery.locator.enabled=true
spring.cloud.gateway.discovery.locator.lowerCaseServiceId=true
```