

# Microservices

Spring Cloud Load Balancer

# Client Side Load Balancer

Alternate to Netflix Ribbon

# Spring Cloud Load Balancer

- Spring Cloud Load Balancer is a client-side load balancing library provided by the Spring Cloud framework.
- It is designed to distribute incoming requests among multiple instances of a service, enhancing the availability, scalability, and performance of microservice architectures
- Supports various load balancing algorithms like round-robin, random selection, least connections, and more
- It serves as a lightweight and modern alternative to Netflix Ribbon

# In our project

- Student service calls Course Service
- If we have multiple instances of course service
- Then we need cloud load balancer to send traffic to all available instance using round robin algorithm from student service
- For that
  - We need to include load balancer dependency in student service.
  - Create a class to load balancer with openfeign.

# Add dependency in student service

```
StudentService2/pom.xml X
34<dependencies>    Add Spring Boot Starters...
35    <dependency>
36        <groupId>org.springframework.cloud</groupId>
37        <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
38    </dependency>
39    <dependency>
40        <groupId>org.springframework.cloud</groupId>
41        <artifactId>spring-cloud-starter-loadbalancer</artifactId>
42    </dependency>
```

# Class to load balance feign client request



```
Package Explorer X [Icons] 8 - □ | CourseServiceLoadBalancerConfig.java X | CourseFeignClient.java
└─ com.marlabs.controller
  └─ StudentController.java
└─ com.marlabs.entity
└─ com.marlabs.feignclients
  └─ CourseFeignClient.java
  └─ CourseServiceLoadBalancerConfig.java
└─ com.marlabs.repository
└─ com.marlabs.request
└─ com.marlabs.response
└─ com.marlabs.service
  └─ StudentService.java
src/main/resources
└─ static
└─ templates

8
9 //name must be same as CourseFeignClient
10 //@FeignClient(name="courservice".....
11 @LoadBalancerClient(name="courservice")
12 public class CourseServiceLoadBalancerConfig {
13     @LoadBalanced
14     @Bean
15     public Feign.Builder feignBuilder(){
16         return Feign.builder();
17     }
18 }
19
```

# Load Balancer in action

- Start the Eureka Server
- Start the Course Service
- Now change the port no to 8083 in properties file and start another instance
- Start the Student Service
- Confirm multiple instances in eureka server
- When you call from postman for 4 times, we will get log message of Course service 2 in both the instances