Naming Server :

In the currency exchange proxy we are hard-coding the URL of the currency exchange service.

@FeignClient(name = "currency-exchange", url = "http://localhost:8000/currency\_exchange")

So if I would want to get the currency conversion service to talk to a different instance of currency exchange service, what do I need to do?

I need to go here and change the URL. 8001, 8002 and so on.

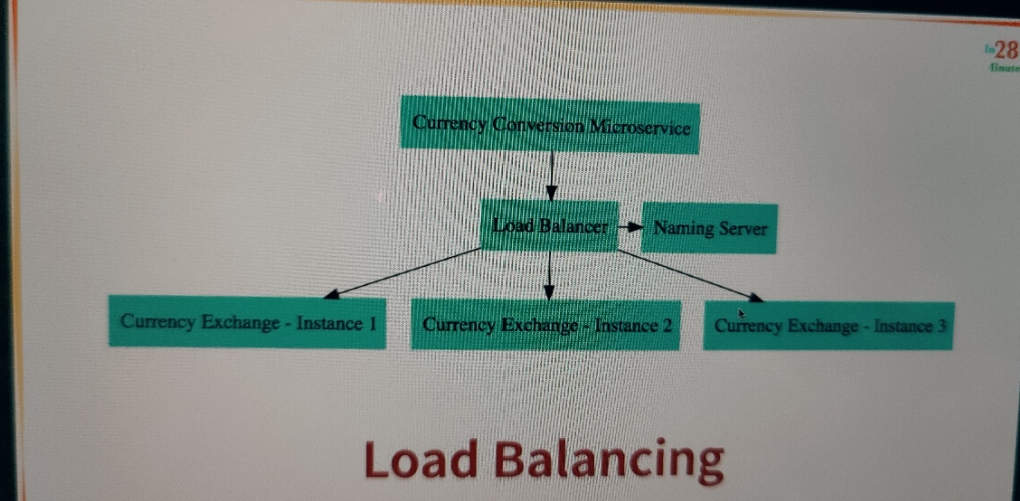
we would want to be able to dynamically launch currency exchange instances and distribute load between them.

let's say 8000 went down and let's say a new instance was brought up on 8002, then you have to change the configuration of this application or the code of this application all the time.

And that's the reason why we go for something called a service registry or a naming server.

As instances come up and go down, we'd want be able to automatically discover them

and load balance between them.



all the instances of all the microservices would register with a service registry.

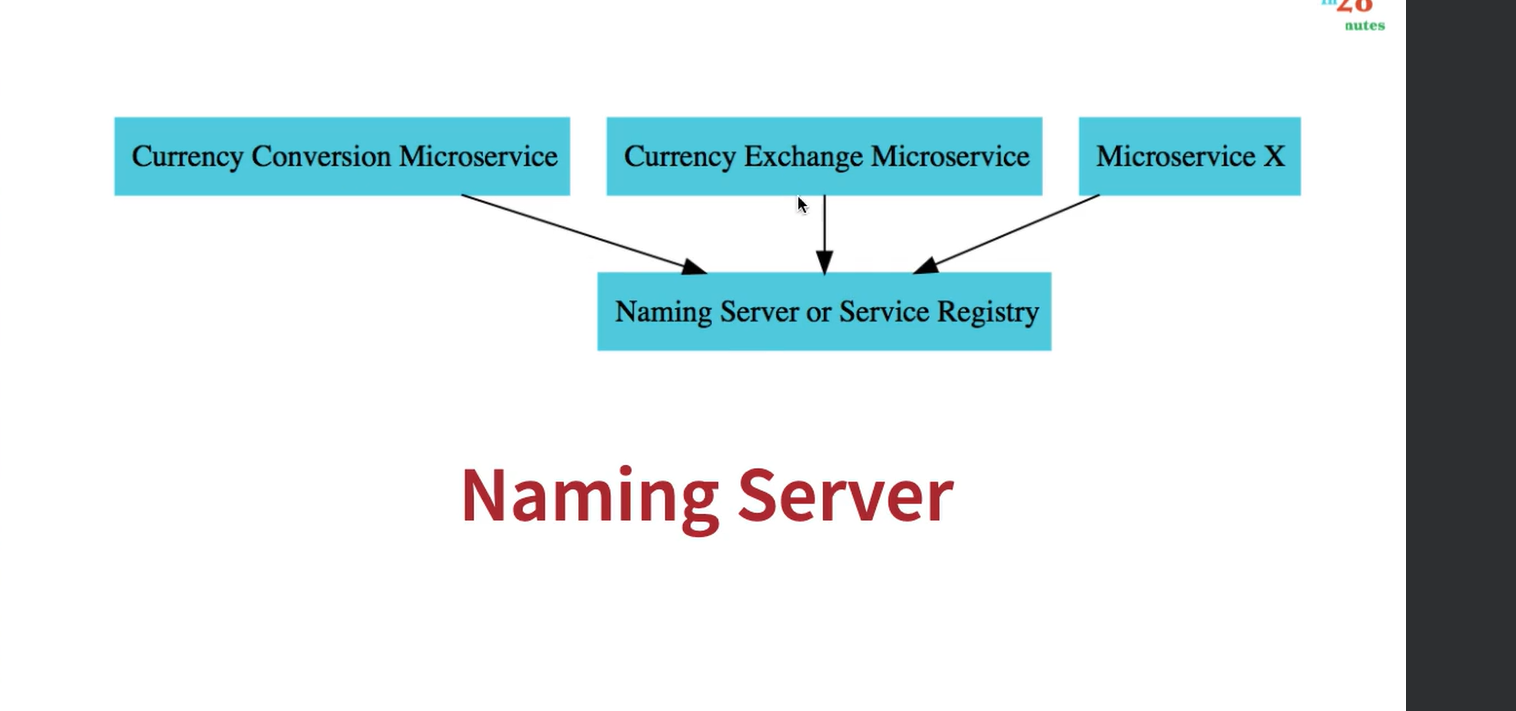
So the currency conversion microservice would register with the service registry, the currency exchange microservice would register with the service registry, and all the other microservices

also would register with the service registry.

And let's say the currency conversion microservice wants to talk to the currency exchange microservice, it would ask the service registry, "Hey, what are the addresses of the currency exchange microservice?"

The service registry would return those back to the currency conversion microservice, and then the currency conversion microservice can send the request out to the currency exchange microservice.

So all the instances would register with the naming server or the service registry.



**Naming server or Service Registiry with euraka (Naming Server / Eureka Server)**

1. @EnableEurekaServer to main class
2. App.properties:  
   #Tells the server not to register itself with Eureka (since it's the server)  
   eureka.client.register-with-eureka=false  
     
   # Instructs the server not to fetch the registry from Eureka, as it's not needed for a server.  
   eureka.client.fetch-registry=false
3. Add dependency

<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>  
</dependency>

**Eureka Client(other microservice need to register with naming server**

So we have to add all the dependency and app.prop to other microservice)

1. Add dependency  
   <dependency>  
    <groupId>org.springframework.cloud</groupId>  
    <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>  
   </dependency>
2. App.prop  
   # URL of the Eureka server where the client should register (http://localhost:8761/eureka/ by default).  
   eureka.client.service-url.defaultZone=http://localhost:8761/eureka/

**Access Eureka dashboard with URL :** [**Eureka**](http://localhost:8761/)