Step 1. Create all sprbs to which you want to apply the microservices architecture.

Point1

Implementing service discovery server/service registry

St1. Create sprb application

St2.add dependency cloud boostrap , eureka server

St3. Add annotation @enableEurekaServer

St4. In application.properties

eureka.instance.hostname = localhost  
eureka.client.register-with-eureka = false  
eureka.client.serviceUrl.defaultZone= http://localhost:8761/eureka/

implementing service discovery client

st1. add dependency cloud boostrap and eureka client and along with the proper pom.xml part

st2.add in application.properties

eureka.client.fetch-registry= true  
eureka.instance.prefer-ip-address= true  
eureka.client.serviceUrl.defaultZone= http://localhost:8761/eureka/

to make a communication in between microservices , we need to use rest template

1. Create a bean of rest template in configuration file or in main class
2. In which class /place we want to use that , autowired it , use its instance.

ResponseEntity<List<Rating>> response = rt.exchange(  
 url,  
 HttpMethod.*GET*,  
 null,  
 new ParameterizedTypeReference<List<Rating>>() {}  
);

Exchange = method like getForEntity or getForObject

url= to hit at particular api

.GET =for mentioning http method

Null = here we can pass the request body or header body .

Feign client implementation.

Add one dependency where we want to use feign client  
  
implementation 'org.springframework.cloud:spring-cloud-starter-openfeign'

Openfeign dependency

Then create interface and inside that mention method like below

@FeignClient(name = "hotel")  
public interface HotelService {  
  
 @GetMapping("/hotels/")  
 List<Hotel> getAllHotels();  
  
}

Here hotel:-- it’s a service name mentioned in the eureka server

In getmapping : -- we have to mentioned the thing like full url means from service to here (requestMapping + getmapping)

To implement loadBalancer

We just need to add annotation in bean of rest template

And wherever , we use url along with the port , what ever we keep application.name of respective service , we can keep inplace of url along with port.

15--------------------

Cofig server :

Used for externalize the configuration , and directly loads the configuration form the application

18 min.

Here create one spring boot application

Use dependencies config server

In main class , below springBootappilcation add EnableConfigServer

Then in application.properties

Add spring.clood.config.server.git.url and spring.clood.config.server.git.clone-on-start = true

It will loads all the files /properties on endpoint .  
  
and in another microservice , where we want to use or in which microservices , we want to import configuration from config server ,

In it , add dependencies config client

In application.properties and spring.config.import add this properties and value like name:url

Likr configServer:http://localhost:8085

And using spring.profile.active , take particular configuration loads to project from config server.