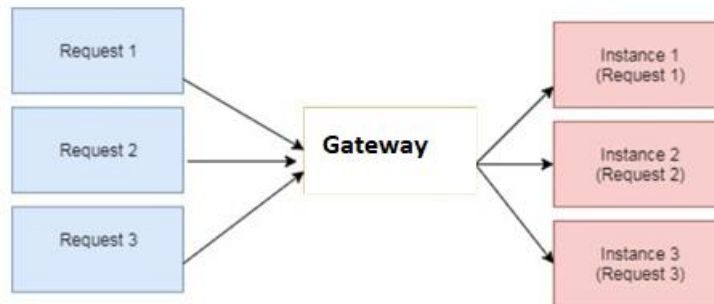


Implémentation « API Gateway »

Objectif

Lorsque le Gateway reçoit une demande, il sélectionne l'un des emplacements physiques disponibles et transmet les demandes à l'instance de service réelle. L'ensemble du processus de mise en cache de l'emplacement des instances de service et du transfert de la demande vers l'emplacement réel est fourni sans aucune configuration supplémentaire.

L'objectif de cet atelier est d'organiser l'appel vers les microservices.



1- Créez un projet Spring Boot en suivant ses étapes :

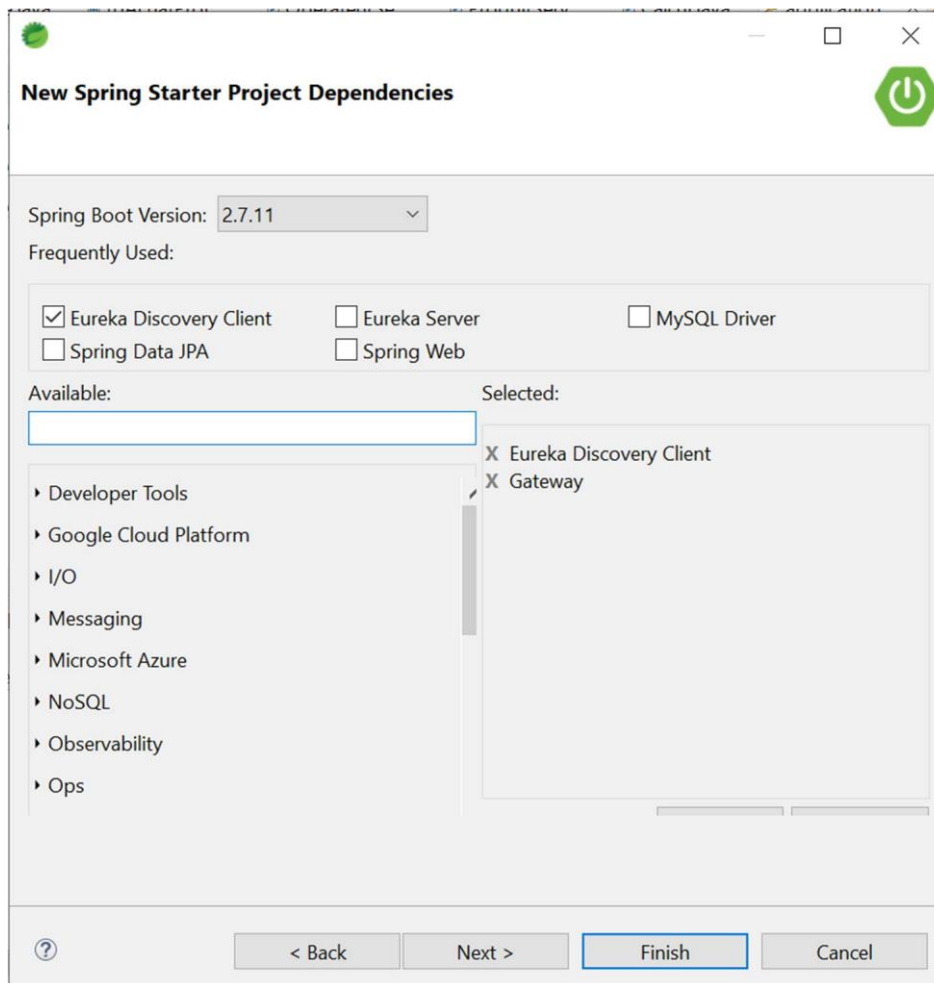
The screenshot shows the 'New Spring Starter Project' dialog box. The fields are filled as follows:

- Service URL: <https://start.spring.io>
- Name: API_Gateway_Server
- ☒ Use default location
- Location: C:\Users\starinfo\Documents\workspace-spring-tool-suite-4-4.15.3.RELE/
- Type: Maven
- Packaging: Jar
- Java Version: 8
- Language: Java
- Group: tn.esprit
- Artifact: API_Gateway_Server
- Version: 1.0
- Description: API_Gateway_Server
- Package: tn.esprit

At the bottom, the 'Next >' button is highlighted with a blue border.

2- Ajoutez ces deux starters :

- Eureka Discovery client
- Gateway



Vous obtenez ces deux dépendances dans votre pom.xml :

```
<dependency>
```

```
    <groupId>org.springframework.cloud</groupId>
```

```
    <artifactId>spring-cloud-starter-gateway</artifactId>
```

```
</dependency>
```

```
<dependency>
```

```
    <groupId>org.springframework.cloud</groupId>
```

```
    <artifactId>spring-cloud-starter-netflix-eureka-  
client</artifactId>
```

```
</dependency>
```

3- Faites un maven  update project et clean install

4- Au niveau de la classe main spring Boot de votre projet, ajouter l'annotation :

`@EnableEurekaClient`

Pour définir une Gateway, il y'a deux méthodes de travail :

- Ajouter la relation entre gateway et MS à travers le fichier application.properties
- Ajouter la relation entre Gateway et MS à travers une classe de configuration

Méthode 1 :

Ajoutez ces lignes dans votre fichier application.properties :

```
server.port=8081
spring.application.name=gateway

eureka.client.serviceUrl.defaultZone = http://localhost:8761/eureka/
eureka.client.register-with-eureka=true

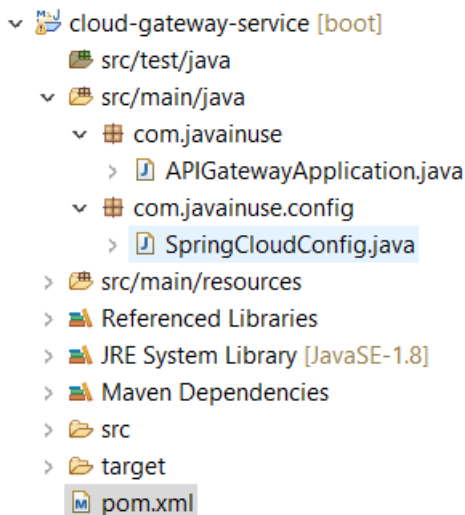
spring.cloud.gateway.discovery.locator.enabled = true

# Candidat Microservice
spring.cloud.gateway.routes[0].id=candidat-service
spring.cloud.gateway.routes[0].uri=http://localhost:8088
spring.cloud.gateway.routes[0].predicates[0]=Path=/candidat/**
```

NB : N'oubliez pas de mettre à jour votre fichier si vous utilisez d'autres ports.

Méthode 2 :

- Ajouter la classe Config comme le montre la figure ci-dessous :



- Le contenu de la classe de configuration **SpringCloudConfig** est le suivant :

```
package com.javainuse.config;

import org.springframework.cloud.gateway.route.RouteLocator;
import org.springframework.cloud.gateway.route.builder.RouteLocatorBuilder;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

@Configuration
public class SpringCloudConfig {

    -
```

```

@Bean
public RouteLocator gatewayRoutes(RouteLocatorBuilder builder) {
    return builder.routes()

        //Micro-service 1
        .route(r -> r.path("/candidat/**")
            .uri("http://localhost:8088/")
            .id("candidat-service"))

        .build();
    }
}

```

Path : c'est le path à utiliser pour accéder au MS-candidat à travers le Gateway

URI : c'est l'emplacement de MS-candidat

Id : c'est l'id de MS-candidat

NB : dans le fichier **application.properties**, vous ajouter juste le port et le nom de l'application

5- Accédez à l'interface de votre Eureka server :

<http://localhost:8761/>

Vous obtenez le résultat suivant :

The screenshot shows the Spring Eureka web interface in a browser. The address bar shows 'localhost:8761'. The page has a dark header with the 'spring Eureka' logo and a 'Toggle navigation' button. Below the header, there's a 'System Status' section with a table showing environment details. To the right, there's a table with system metrics. Below that is the 'DS Replicas' section. At the bottom, there's a section titled 'Instances currently registered with Eureka' containing a table with columns for Application, AMIs, Availability Zones, and Status. Two instances are listed: 'CANDIDAT-SERVICE' and 'GATEWAY', both with a status of 'UP'.

Environment	test	Current time	2023-09-26T10:01:20 +0100
Data center	default	Uptime	00:04
		Lease expiration enabled	true
		Renews threshold	5
		Renews (last min)	12

Application	AMIs	Availability Zones	Status
CANDIDAT-SERVICE	n/a (1)	(1)	UP (1) - host.docker.internal:candidat-service:8088
GATEWAY	n/a (1)	(1)	UP (1) - host.docker.internal:gateway:8081

6- Testez votre application en utilisant l'url (le numéro de port) de votre API-Gateway service :

Exemple :

English ▼

[Preferences](#) [Tools](#) [Help](#)

Login

Saved Settings:

Generic H2 (Embedded) ▼

Setting Name:

Generic H2 (Embedded)

Save

Remove

Driver Class:

org.h2.Driver

JDBC URL:

jdbc:h2:~/test

User Name:

sa

Password:

Connect

Test Connection