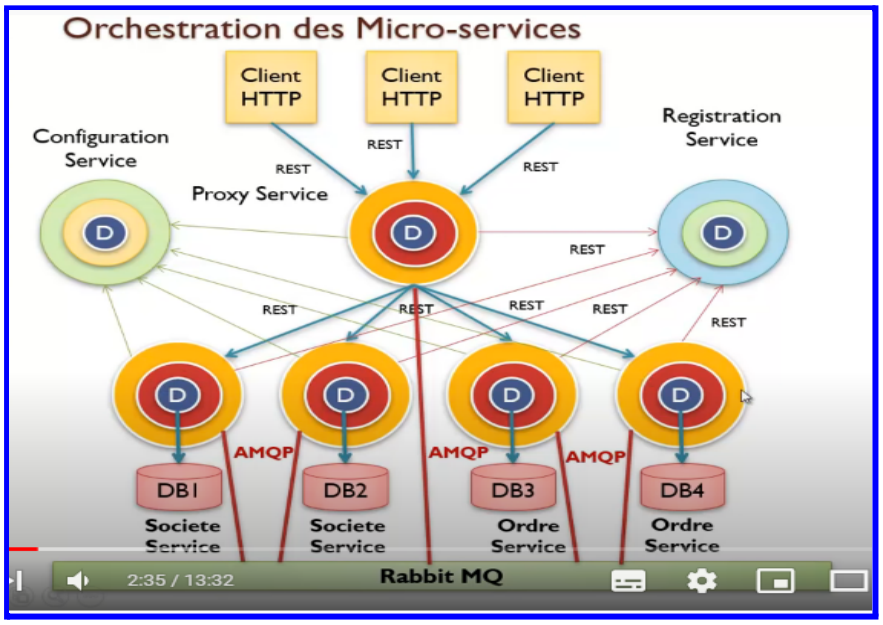
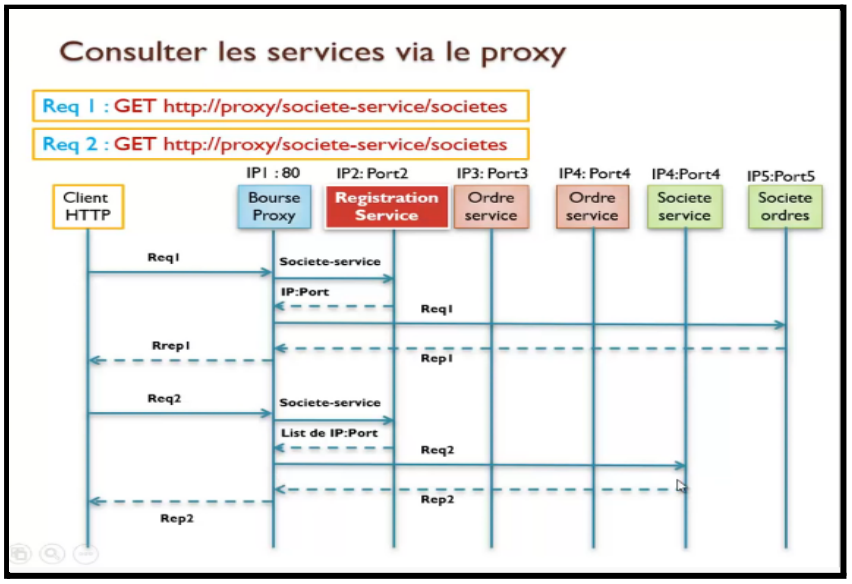
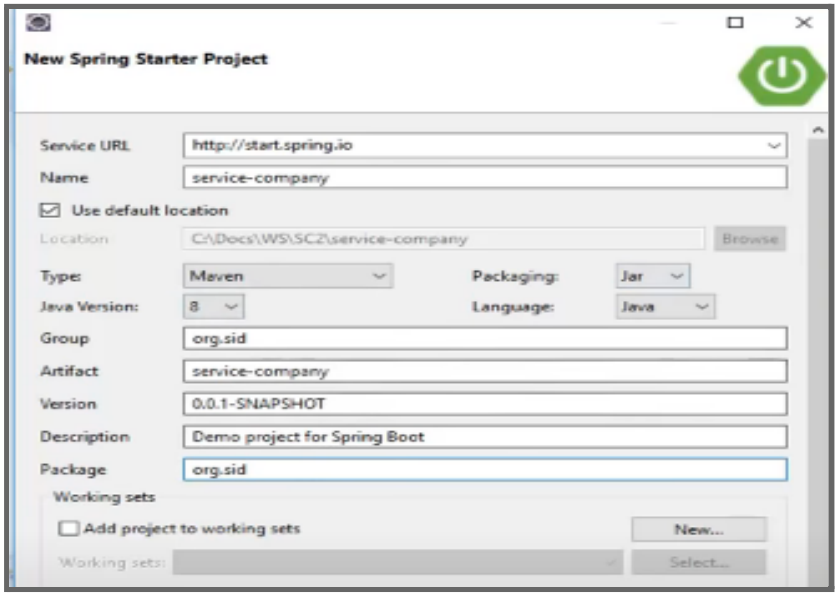
# Approche Microservices

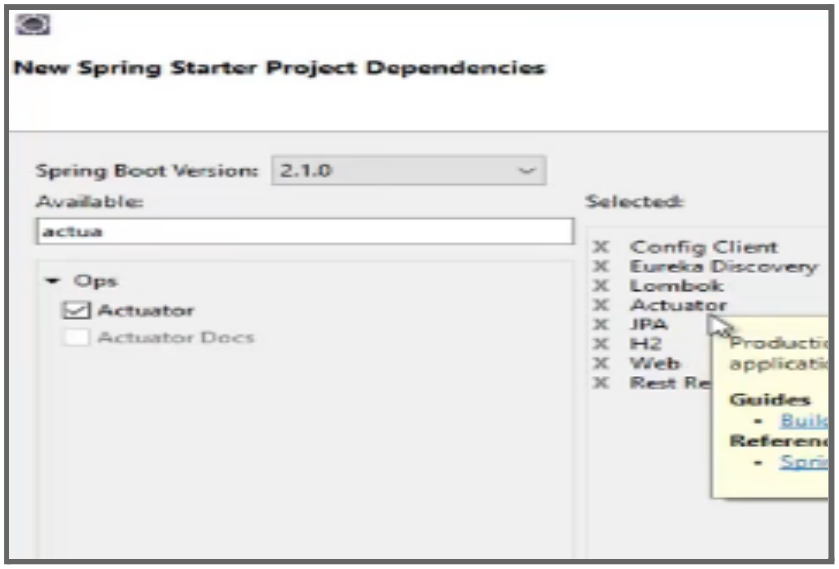



Comment orchestrer les microservices :
On commence en général par développer les microservices un à un.

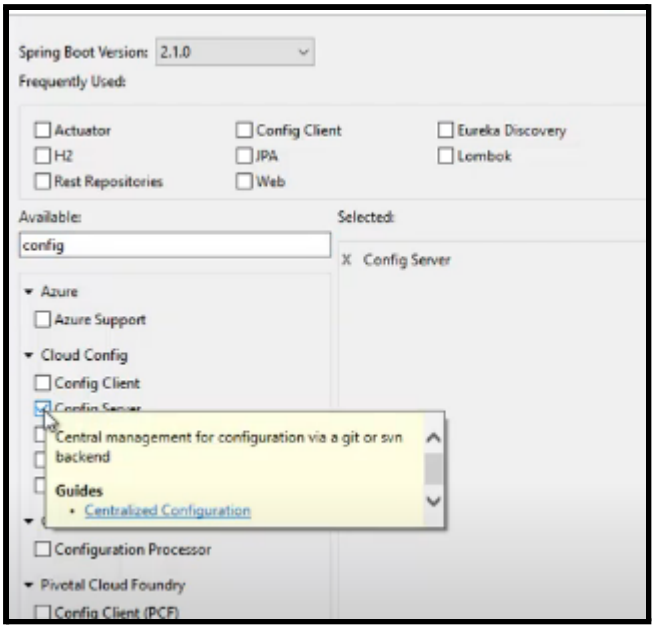
Centraliser la configuration (non pas distribuer des microservices )
un service de configuration




New Spring Starter Project
Service URL
http://start.spring.io
Name
service-company
Use default location
Location
C:\Docs\WS\SC2\service-company
Browse
Type:
Maven
Packaging:
Jar
Java Version:
8
Language:
Java
Group
org.sid
Artifact
service-company
Version
0.0.1-SNAPSHOT
Description
Demo project for Spring Boot
Package
org.sid
Working sets
Add project to working sets
New...
Working sets:
Select...


New Spring Starter Project Dependencies
Spring Boot Version:
2.1.0
Available:
actua
Ops
Actuator
Actuator Docs
Selected:
Config Client
Eureka Discovery
Lombok
Actuator
JPA
H2
Web
Guides

# Service de configuration




using spring cloud config

```
@EnableConfigServer
@SpringBootApplication
public class ServiceConfigApplication {
```

---

```
server.port=8888
spring.cloud.config.server.git.uri=file://${user.home}/cloud-config
```

---

Pour test
n oublier pas de commit abec git par branch master

http://localhost:8888/application/master

http://localhost:8888/{{NomServeMicrose}}/master

# Appel de Service de configuraton

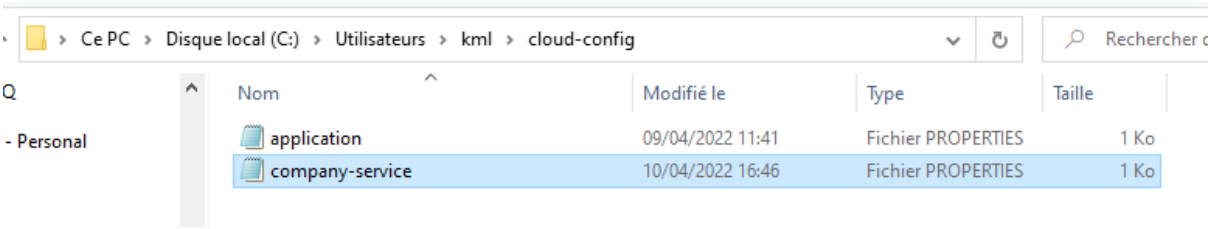
On va demander aux company lorsqu' il démarrer d' appeler le microservice de configuration pour récupérer sa configuration externaliser.

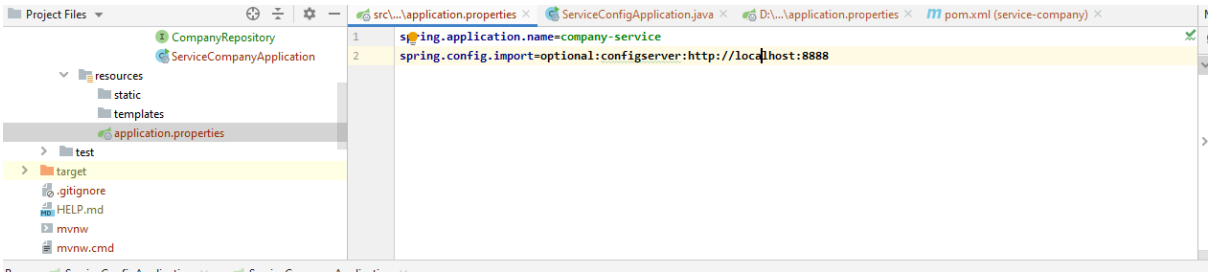
On vers pom.xml de company Server et decommenter le code

```
<!--		<dependency>-->
<!--			<groupId>org.springframework.cloud</groupId>-->
<!--			<artifactId>spring-cloud-starter-config</artifactId>-->
<!--		</dependency>-->
```

ensuite changer le fichier application.properties

- Chaque microservice il faut l attribuer un nom  le nom doit correspndre au no m de fichoer externaliser
- Renommer le fichier d application.properties to bootstrap.properties sinon tu peux faire
-




appeler la config
Nommer le service
http://localhost:8081/companies
Pour tester RestResource
‘

Faculataif Actuator

http://localhost:8081/actuator

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

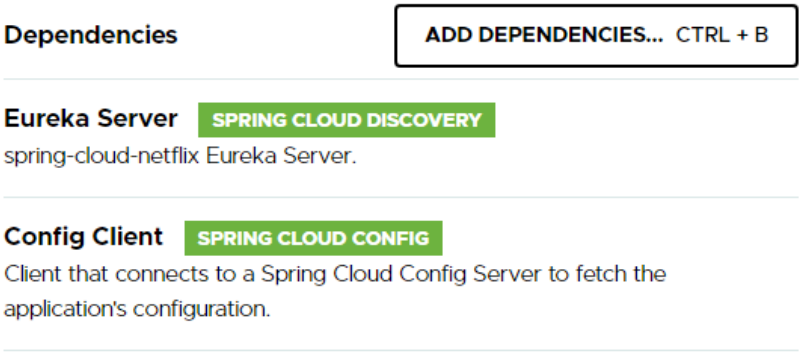
```
management.endpoints.web.exposure.include=*
```

—----------

# Registration SERVICE

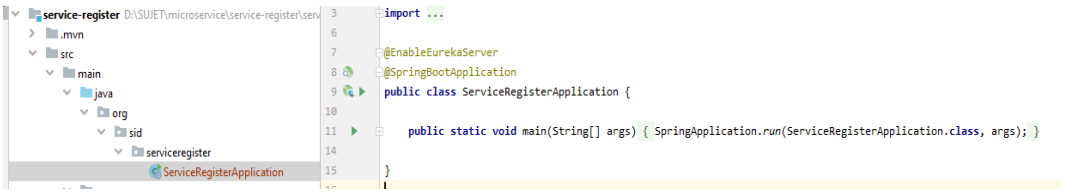
Stat.spring.io creer une nouveau projet nommer service-Register

- ☐ Eureka Server
- ☐ Config client pour centraliser sa config aussi



1- Activer Eureka Server

Ajouter l annotation @EnableEurekaServer a entrer de l application

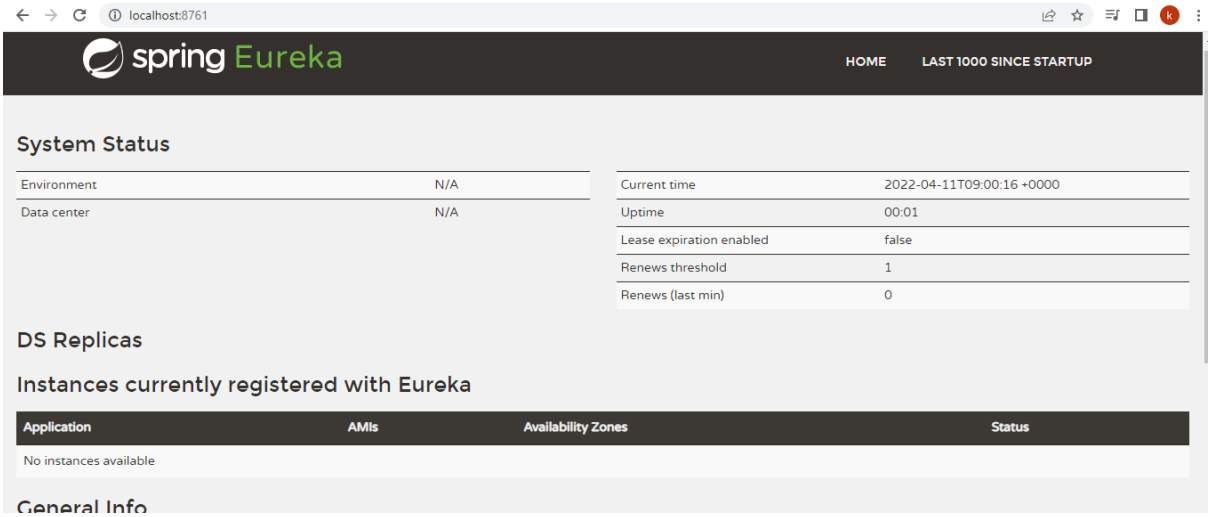



dans un ficheir applicaiton.propeties ajouter ces inforamtion

**spring.application.name=register-service**
**spring.config.import=optional:configserver:http://localhost:8888**

sinon mieux externaliser ces inform

**eureka.client.registerWithEureka = false**
**eureka.client.fetchRegistry = false**
**server.port = 8761**



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

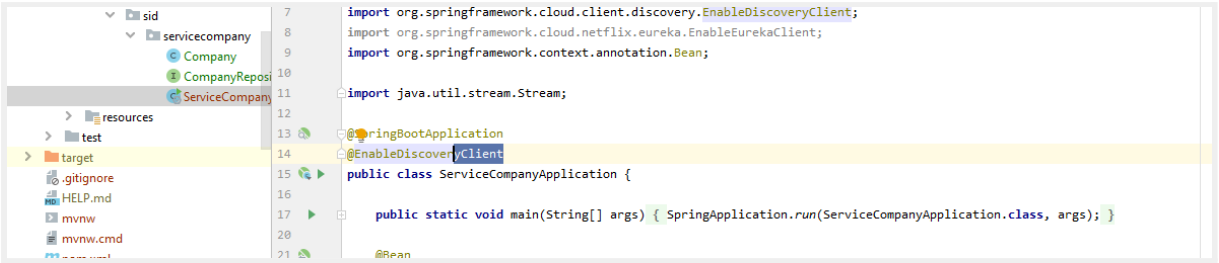
ajouer ce code pour tous les client d eureka pour les enregister et ajouter lannotation a lentree de lapplicaiton

@ENABLEdISC

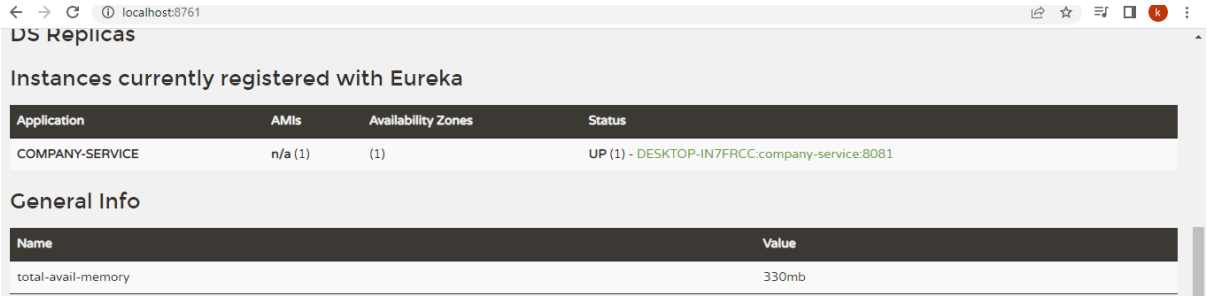
either we use **@EnableEurekaCient** otherwise we use **@EnableDiscovery , differnece is**

@EnableEurekaCient will initilize the service discovery using eureka server, its like explicitly specifying to use EUREKA

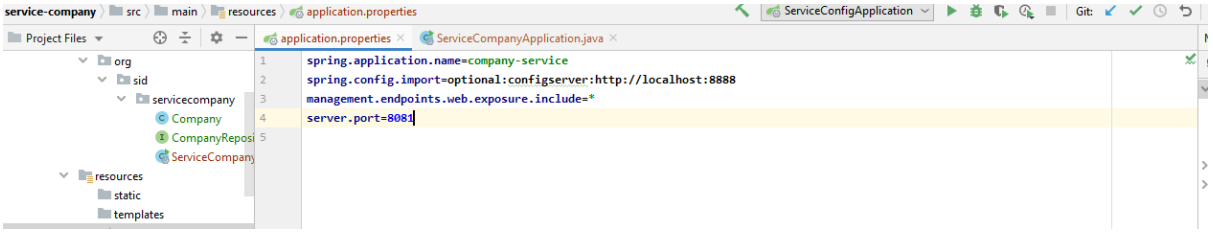
While @EnableDiscovery Client will register dicovery service using the jar available in classpath like consul, Eureka, Kubernetes.

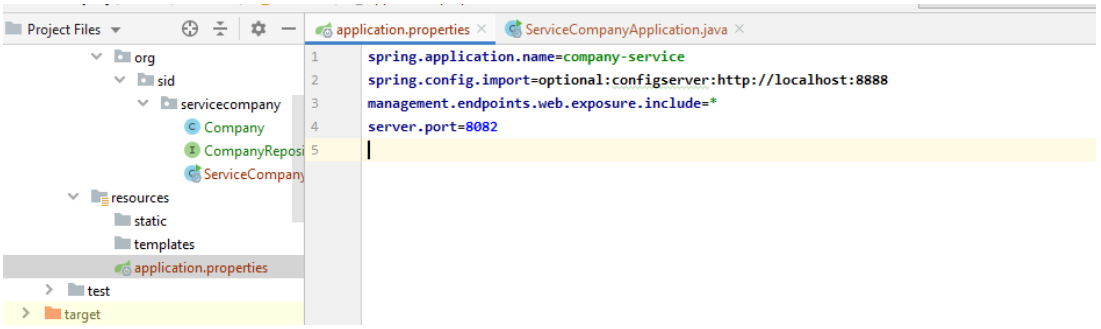



Run project and check in eureka interface
the are other centrer registration like kubernetes +

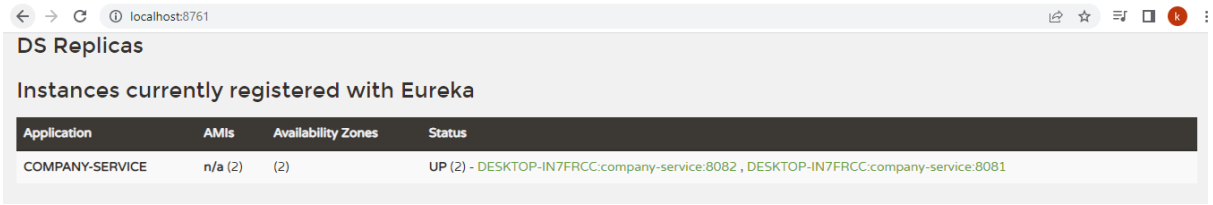


On met le port dans le fichier de properties pour le changer en cas de création de plusieur instances



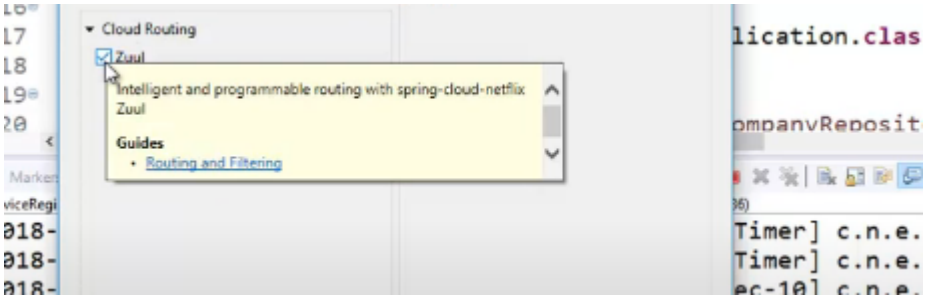




two ports of two different instances of the same service, we notice in eureka we have two instances of the same service



# Sevice PROXY  ZUUL

CREER NEW PROJECT name service-proxy with ONE DEPENENCIY

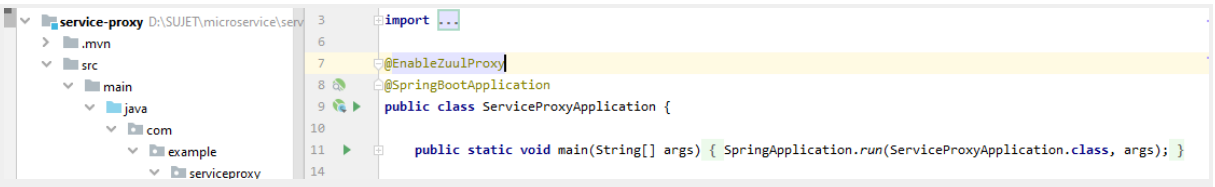



```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-zuul</artifactId>
    <version>1.0.0.RELEASE</version>
</dependency>
```

vilnerable depenendcy

or add jar manually in
.m2

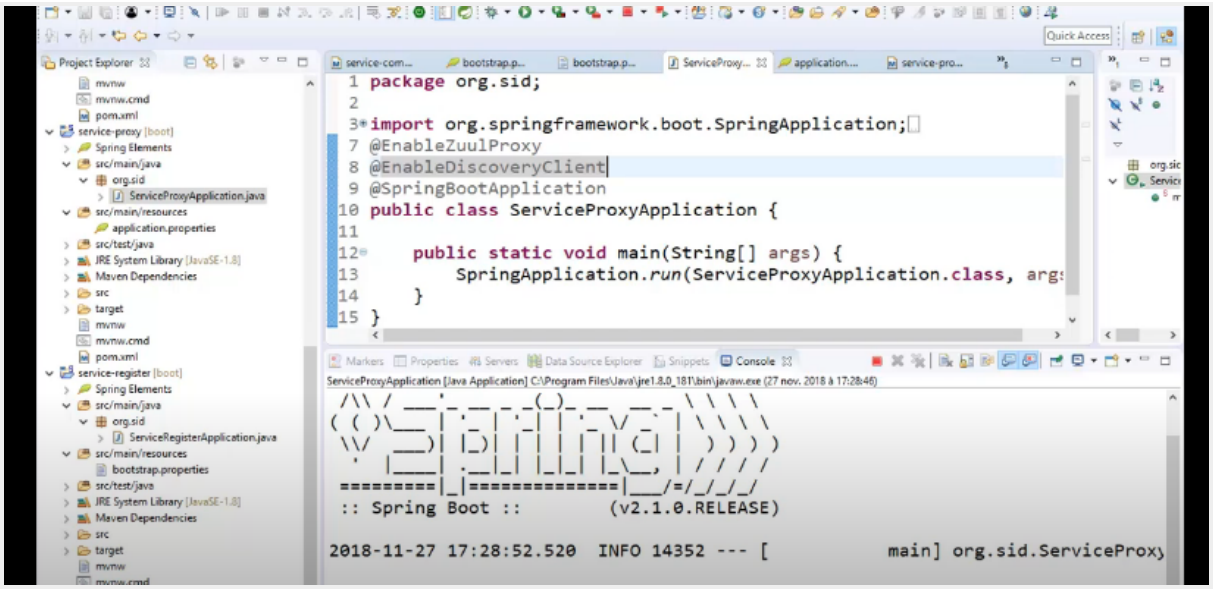
.m2\repository\org\springframework\cloud\spring-cloud-starter-zuul
\1.0.0.RELEASE




```
server.port=8089
```

**Test :**

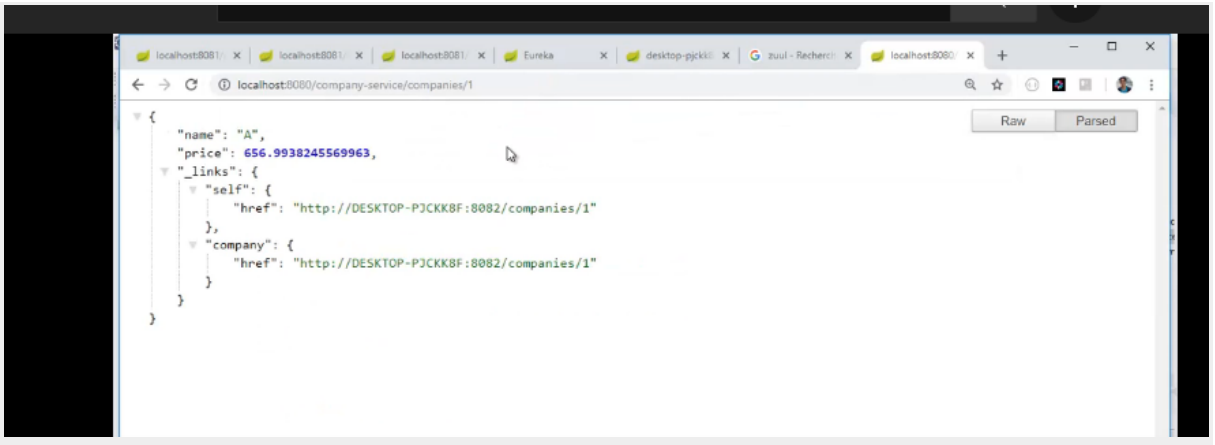
min 14:50 video 5




IL FAUT L ENTEGISTRER DANS EUREKA AUSSI


```
spring.application.name=proxy
spring.config.import=optional:configserver:http://localhost:8888
```

cONSULTER VIA PROXY




aPPELER LES SERVICE POUR LE NOM DE SERVICE NON PAR LEUR IP /: port

on aura plusieur instance avec meme nom et port different

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.5.12</version>
    <relativePath/> <!-- lookup parent from repository -->
  </parent>
  <groupId>org.sid</groupId>
  <artifactId>service-company</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>service-company</name>
  <description>Service company </description>
  <properties>
    <java.version>8</java.version>
    <spring-cloud.version>2020.0.5</spring-cloud.version>
  </properties>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-actuator</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-rest</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springframework.cloud</groupId>
      <artifactId>spring-cloud-starter-config</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springframework.cloud</groupId>
      <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
    </dependency>

    <dependency>
      <groupId>mysql</groupId>
      <artifactId>mysql-connector-java</artifactId>
```

```
      <scope>runtime</scope>
    </dependency>
    <dependency>
      <groupId>org.projectlombok</groupId>
      <artifactId>lombok</artifactId>
      <optional>true</optional>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-test</artifactId>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
  </dependencies>
  <dependencyManagement>
    <dependencies>
      <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-dependencies</artifactId>
        <version>${spring-cloud.version}</version>
        <type>pom</type>
        <scope>import</scope>
      </dependency>
    </dependencies>
  </dependencyManagement>

  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
        <configuration>
          <excludes>
            <exclude>
              <groupId>org.projectlombok</groupId>
              <artifactId>lombok</artifactId>
            </exclude>
          </excludes>
        </configuration>
      </plugin>
    </plugins>
  </build>

</project>
```