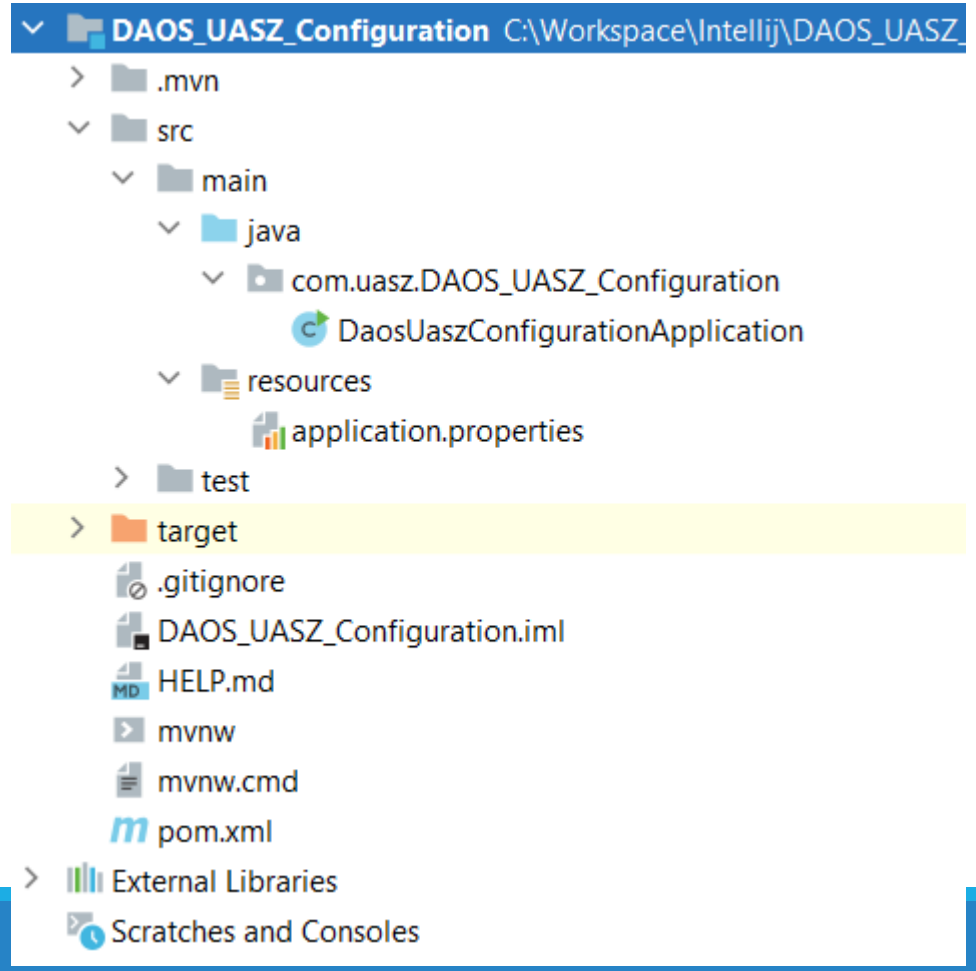
DAOS

Micro service Configuration



Pom.xml

```xml
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-config-server</artifactId>
</dependency>

<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>
```

Fichier Run

```java
@EnableConfigServer
@SpringBootApplication
public class DaosUaszConfigurationApplication {

    public static void main(String[] args) {
        SpringApplication.run(DaosUaszConfigurationApplication.class, args);
    }

}
```

Application.properties

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

Creation du repository git : cloud-config

Lancer git bash :

- cd c:
- cd utilisateurs/nom_user
- mkdir cloud-config
- cd cloud-config
- ls
- git init

Creation des fichiers de configuration : cloud-config

Lancer git bash :

- cd c:
- cd utilisateurs/nom_user
- cd cloud-config
- code application.properties

Le fichier **application.properties** contient tout les parametres de configurations qui sont commun a tout les microservices

Creation des fichiers de configuration : cloud-config

Lancer git bash :

- cd c:
- cd utilisateurs/nom_user
- cd cloud-config
- code ms-service.properties

Le fichier **ms-service.properties** contient tout les parametres de configurations qui sont specifique a un microservice

maquette-service.properties

```
server.port=8081

spring.datasource.driverClassName=org.h2.Driver
spring.datasource.url=jdbc:h2:mem:maquette-db
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.h2.console.enabled=true
spring.h2.console.path=/maquette/h2
```

repartition-service.properties

```
server.port=8082

spring.datasource.driverClassName=org.h2.Driver
spring.datasource.url=jdbc:h2:mem:repartition-db
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.h2.console.enabled=true
spring.h2.console.path=/repartition/h2
```

emploiDuTemps-service.properties

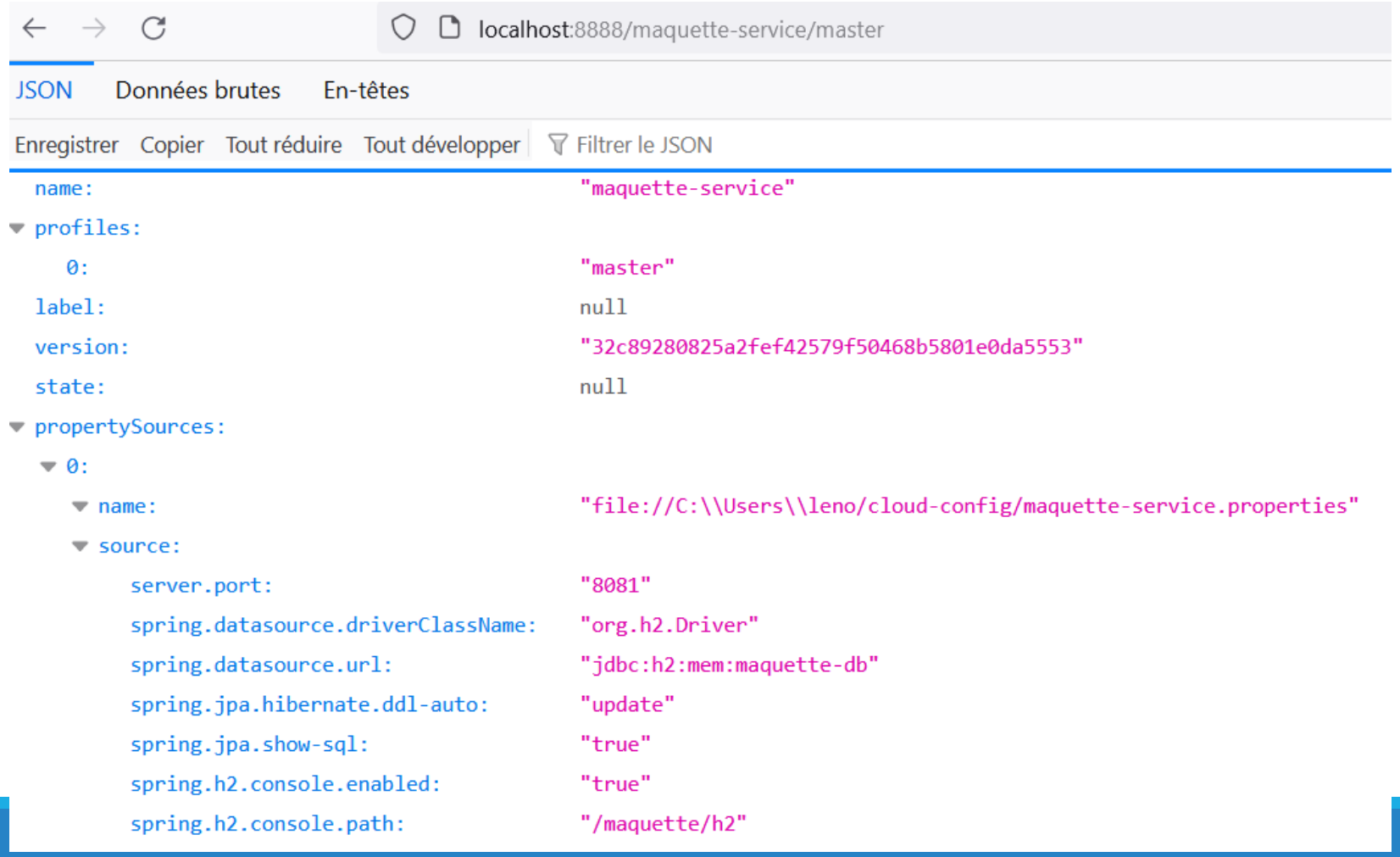
```
server.port=8083

spring.datasource.driverClassName=org.h2.Driver
spring.datasource.url=jdbc:h2:mem:emploiDuTemps-db
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.h2.console.enabled=true
spring.h2.console.path=/emploiDuTemps/h2
```

Creation des fichiers de configuration : cloud-config

Lancer git bash :

- cd c:
- cd utilisateurs/nom_user
- cd cloud-config
- git status
- git add .
- git commit –m " message "

Tester le MS Configuration



Micro Service Maquette

```xml
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
    <version>4.1.0</version>
</dependency>
```

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

Micro Service Repartition

```xml
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
    <version>4.1.0</version>
</dependency>
```

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

Micro Service EmploiDuTemps

```xml
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
    <version>4.1.0</version>
</dependency>
```

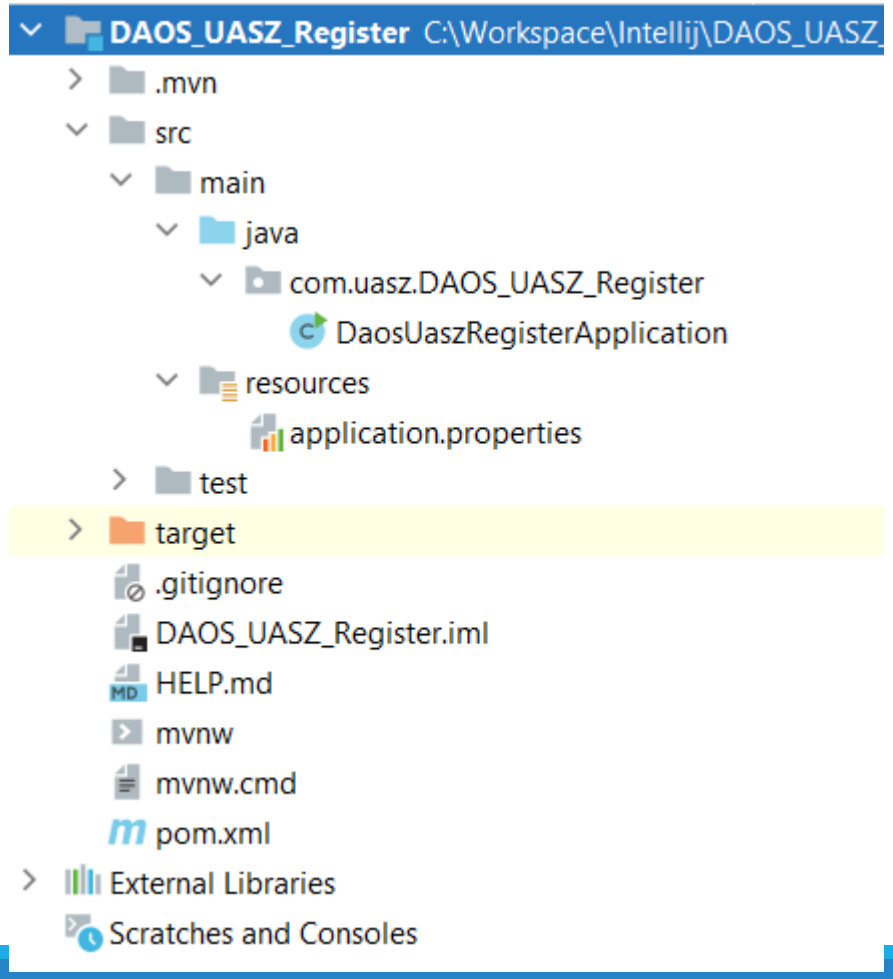
```
spring.application.name=emploiDuTemps-service
spring.config.import=optional:configserver:http://localhost:8888/
```

Actuator

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

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

Micro service Register



Pom.xml

```xml
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
    <version>4.1.0</version>
</dependency>
```

Fichier Run

```java
@EnableEurekaServer
@SpringBootApplication
public class DaosUaszRegisterApplication {

    public static void main(String[] args) {
        SpringApplication.run(DaosUaszRegisterApplication.class, args);
    }

}
```

Application.properties

```
spring.application.name=eureka-service
spring.config.import=optional:configserver:http://localhost:8888/
management.endpoints.web.exposure.include=*
```

Creation des fichiers de configuration : cloud-config

Lancer git bash :

- cd c:
- cd utilisateurs/nom_user
- cd cloud-config
- code eureka-service.properties

Le fichier **eureka-service.properties** contient tout les parametres de configurations qui sont specifique a un microservice register

eureka-service.properties

```
server.port=8761
eureka.client.register-with-eureka=false
eureka.client.fetch-registry=false
eureka.instance.hostname=localhost
```

Pom.xml : MS Maquette – MS Repartition – MS EmploiDuTemps

```xml
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
    <version>4.1.0</version>
</dependency>
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
    <version>4.1.0</version>
</dependency>
```

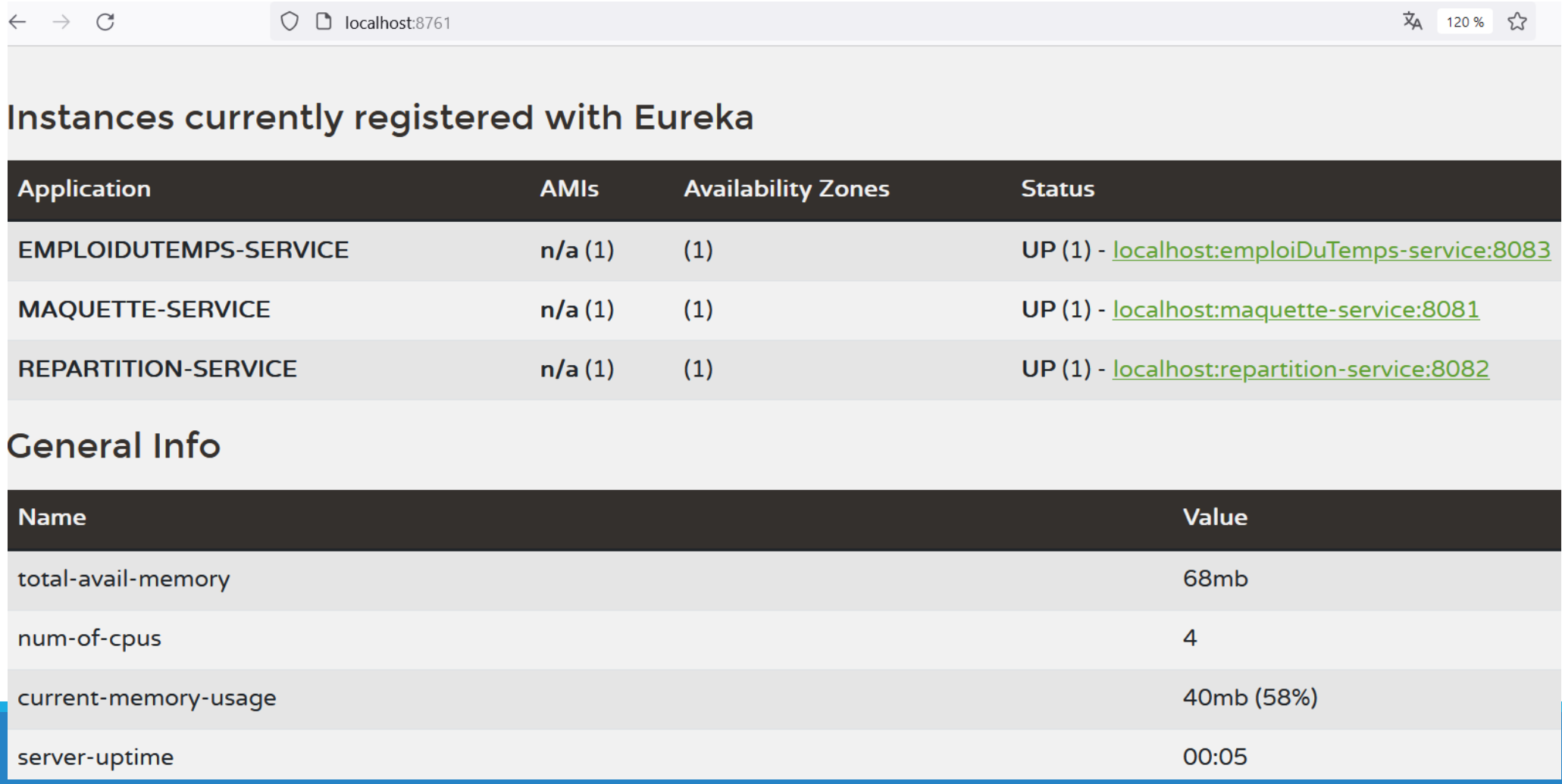
Fichier Run : MS Maquette – MS Repartition – MS EmploiDuTemps

```
@EnableDiscoveryClient
@SpringBootApplication
public class DaosUaszMaquetteApplication implements CommandLineRunner {
```

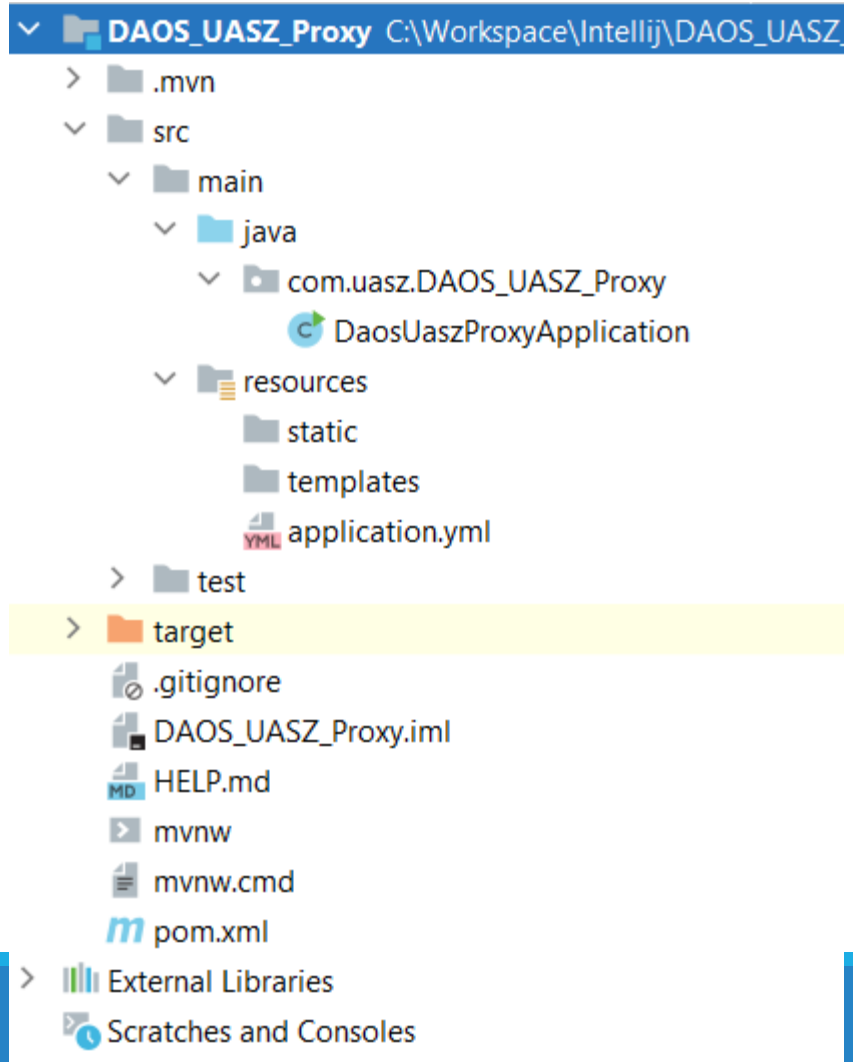
```
@EnableDiscoveryClient
@SpringBootApplication
public class DaosUaszRepartitionApplication implements CommandLineRunner {
```

```
@EnableDiscoveryClient
@SpringBootApplication
public class DaosUaszEmploiDuTempsApplication implements CommandLineRunner {
```

Tester le MS Register



Micro service Proxy



Pom.xml

```xml
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
    <version>4.1.0</version>
</dependency>
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-gateway</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-webflux</artifactId>
</dependency>
<dependency>
    <groupId>io.projectreactor</groupId>
    <artifactId>reactor-test</artifactId>
    <scope>test</scope>
</dependency>
```

Application.yml

```yaml
server:
  port: 8080
spring:
  cloud:
    gateway:
      routes:
        - id: maquette-service
          uri: http://localhost:8081/
          Predicates:
            - Path=/**
        - id: repartition-service
          uri: http://localhost:8082/
          Predicates:
            - Path=/**
        - id: emploiDuTemps-service
          uri: http://localhost:8083/
          Predicates:
            - Path=/**
  application:
    name: proxy-service
eureka:
  client:
    serviceURL:
      defaultZone: http://localhost:8761/eureka
```

Fichier Run

```java
@EnableDiscoveryClient
@SpringBootApplication
public class DaosUaszProxyApplication {

    public static void main(String[] args) {
        SpringApplication.run(DaosUaszProxyApplication.class, args);
    }

    @Bean
    public RouteLocator routerBuilder(RouteLocatorBuilder routeLocatorBuilder){
        return routeLocatorBuilder.routes()
                .route( id: "maquette-service",r->r.path( ...patterns: "/maquette/**")
                        .uri("http://localhost:8081/"))
                .route( id: "repartition-service",r->r.path( ...patterns: "/repartition/**")
                        .uri("http://localhost:8082/"))
                .route( id: "emploiDuTemps-service",r->r.path( ...patterns: "/emploiDuTemps/**")
                        .uri("http://localhost:8083/")).build();
    }

}
```

Creation des fichiers de configuration : cloud-config

Lancer git bash :

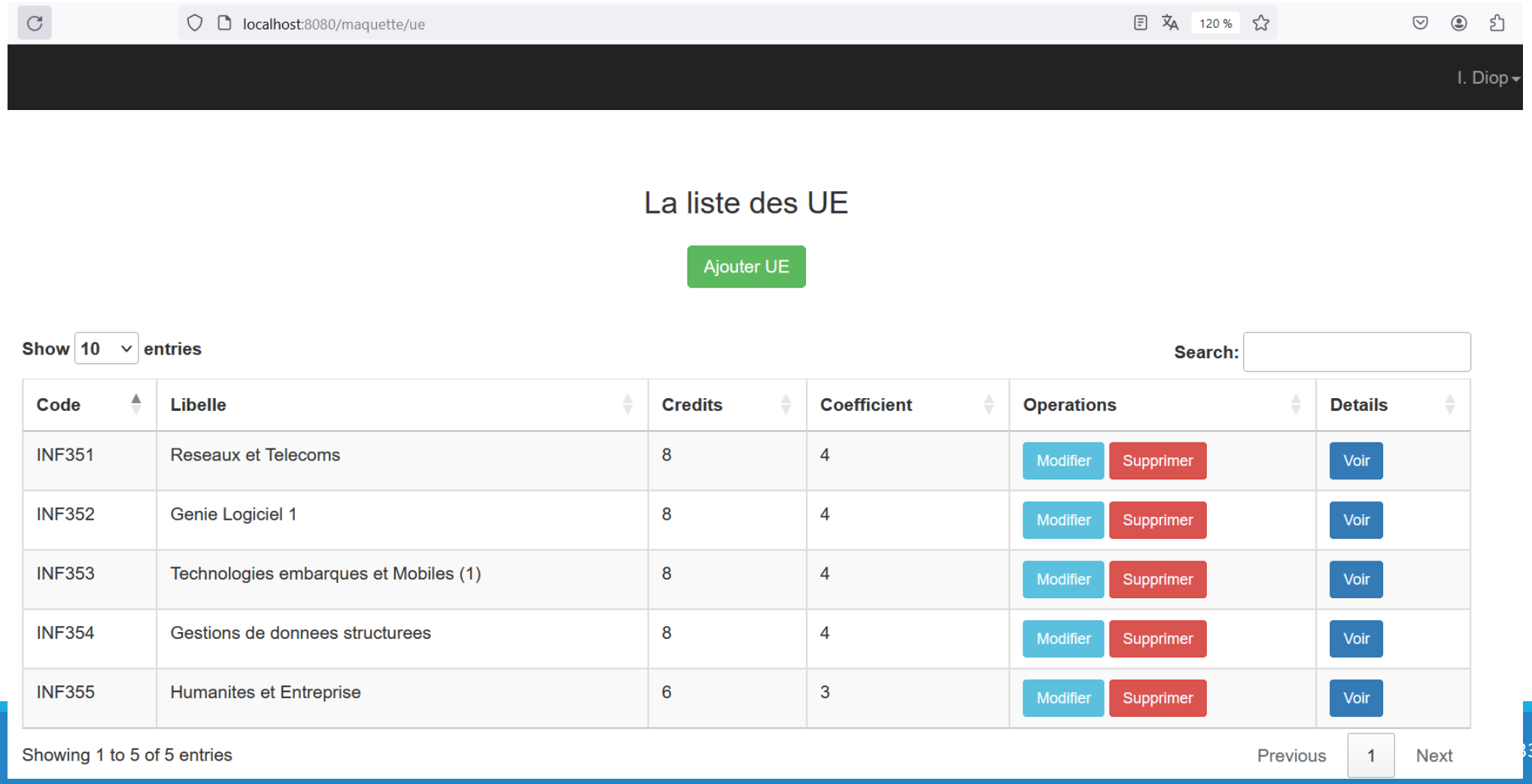
- cd c:
- cd utilisateurs/nom_user
- cd cloud-config
- code eureka-service.properties

Le fichier **proxy-service.properties** contient tout les parametres de configurations qui sont specifique a un microservice proxy

proxy-service.properties

```
server.port=8080
```

Tester le MS Proxy



Tester le MS Proxy