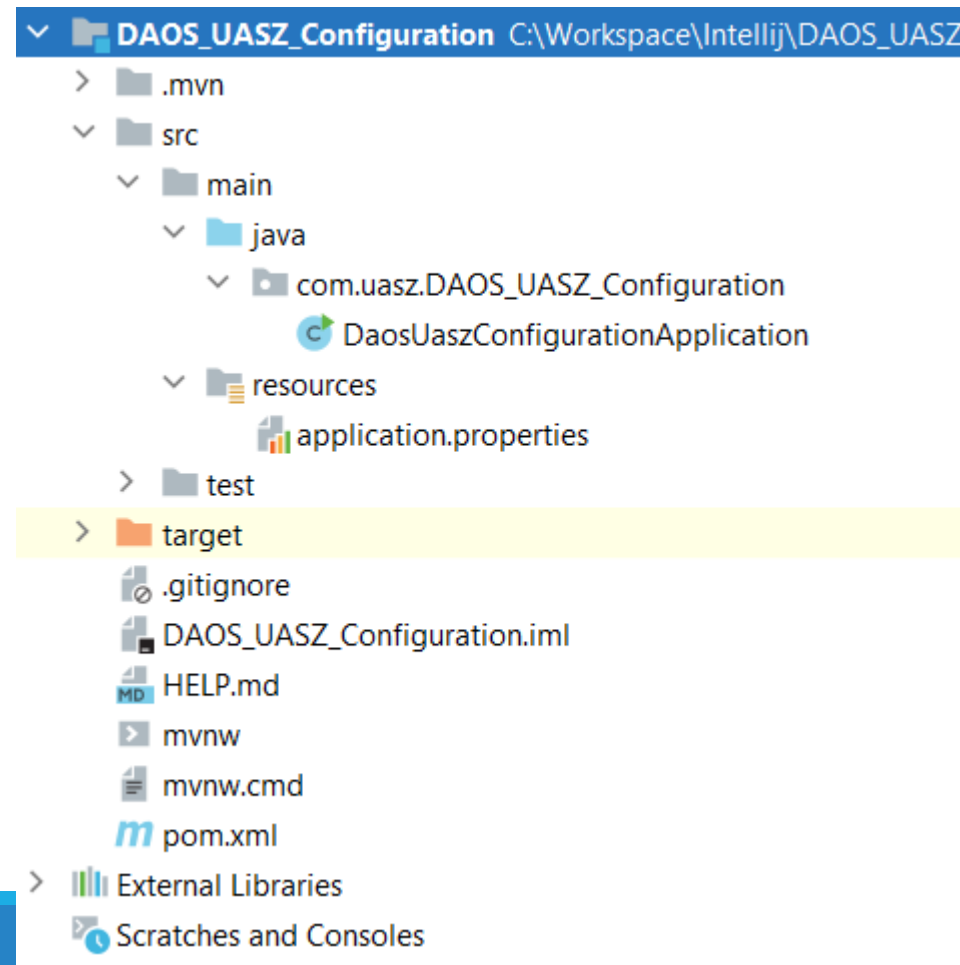


# DAOS

---

# Micro service Configuration

---



# Pom.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

---

```
@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

The screenshot shows a web browser window with the address bar displaying `localhost:8888/maquette-service/master`. The page content is a JSON configuration for a service named "maquette-service". The JSON is displayed in a structured, collapsible format. The configuration includes a "profiles" section with a "master" profile, and a "propertySources" section with a single source pointing to a file. The file contains various database and server configuration properties.

```
{
  "name": "maquette-service",
  "profiles": {
    "0": {
      "label": null,
      "version": "32c89280825a2fef42579f50468b5801e0da5553",
      "state": null
    }
  },
  "propertySources": {
    "0": {
      "name": "file://C:\\Users\\lno/cloud-config/maquette-service.properties",
      "source": {
        "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"
      }
    }
  }
}
```

# Micro Service Maquette

---

```
<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

---

```
<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

---

```
<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/
```



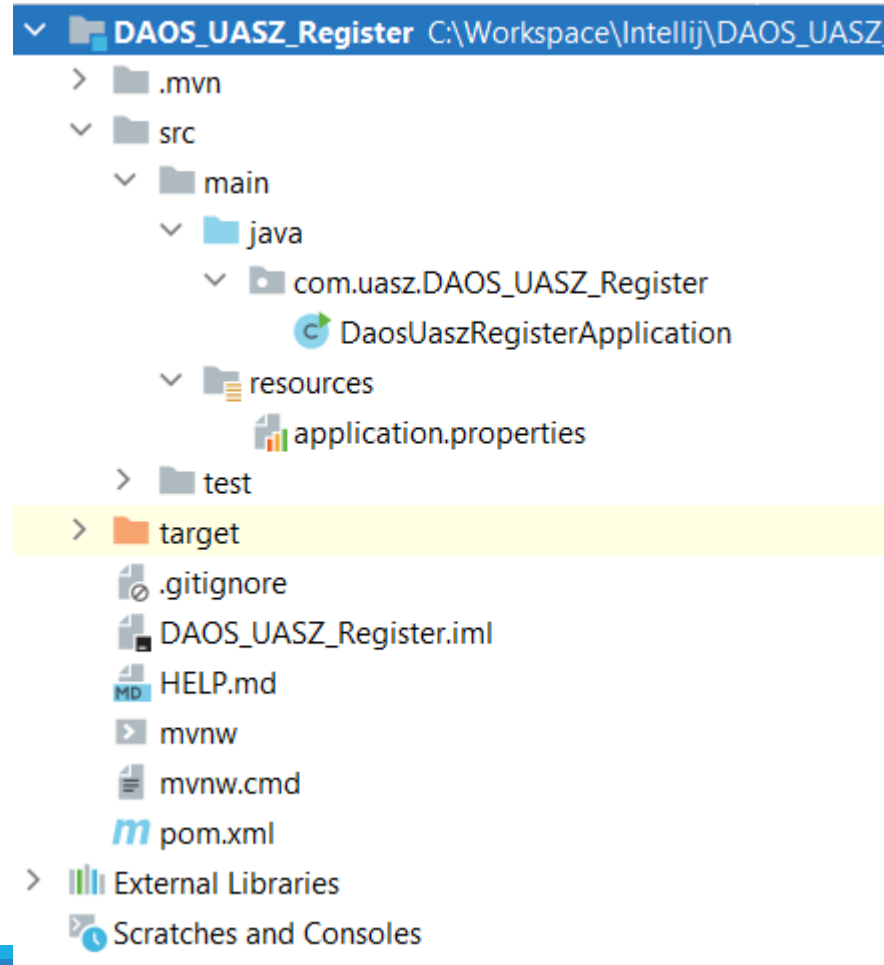
# Actuator

---

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

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

# Micro service Register



# Pom.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

---

```
@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

---

```
<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 {
```

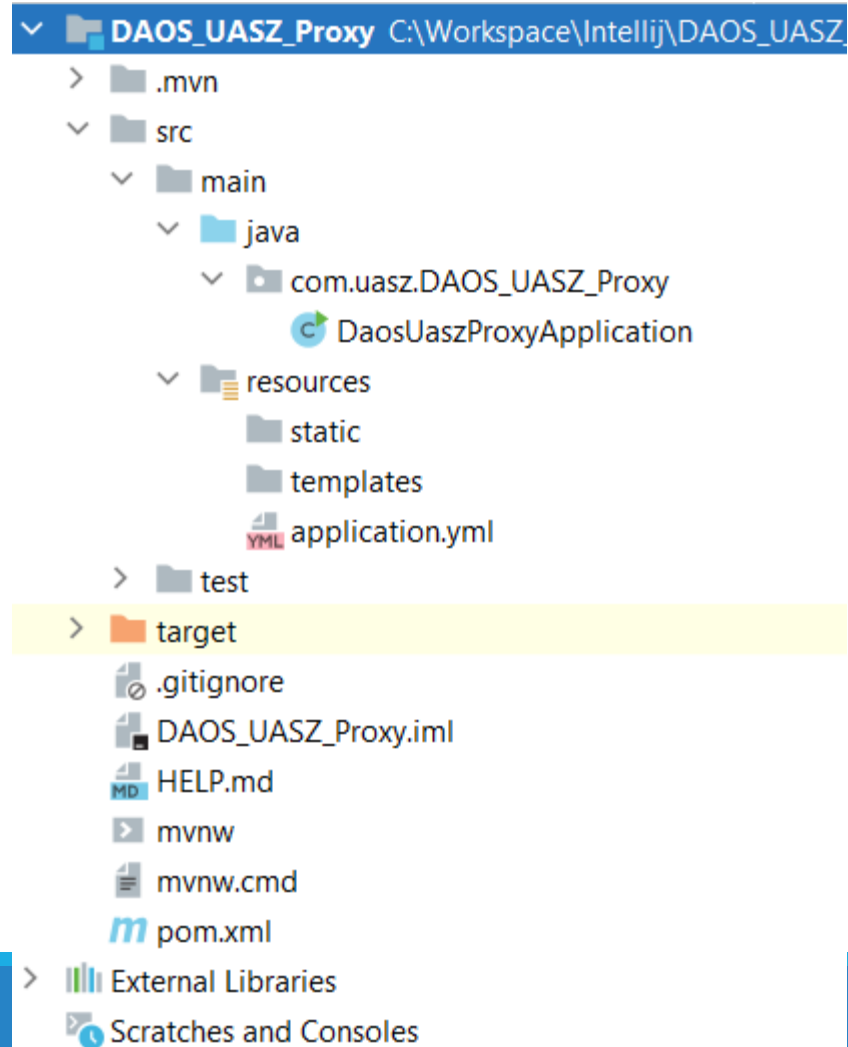
```
@EnableDiscoveryClient
@SpringBootApplication
public class DaosUaszRepartitionApplication implements CommandLineRunner {
```

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

# Tester le MS Register

← → ↻ localhost:8761 120 % ☆			
Instances currently registered with Eureka			
Application	AMIs	Availability Zones	Status
EMPLOIDUTEMPS-SERVICE	n/a (1)	(1)	UP (1) - <a href="localhost:emploiDuTemps-service:8083">localhost:emploiDuTemps-service:8083</a>
MAQUETTE-SERVICE	n/a (1)	(1)	UP (1) - <a href="localhost:maquette-service:8081">localhost:maquette-service:8081</a>
REPARTITION-SERVICE	n/a (1)	(1)	UP (1) - <a href="localhost:repartition-service:8082">localhost:repartition-service:8082</a>
General Info			
Name	Value		
total-avail-memory	68mb		
num-of-cpus	4		
current-memory-usage	40mb (58%)		
server-uptime	00:05		

# Micro service Proxy



# Pom.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

---

```
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

---

```
@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

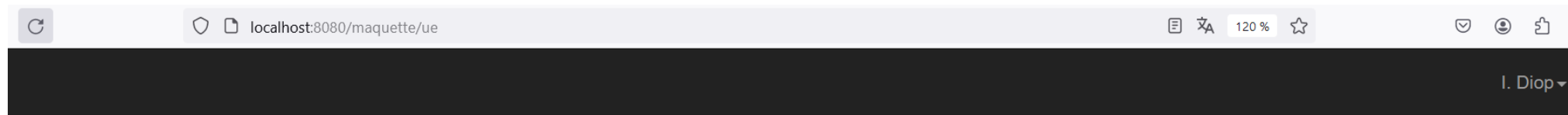
# proxy-service.properties

---

```
server.port=8080
```



# Tester le MS Proxy



## La liste des UE

Ajouter UE

Show 10 entries

Search:

Code	Libelle	Credits	Coefficient	Operations	Details
INF351	Reseaux et Telecoms	8	4	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>
INF352	Genie Logiciel 1	8	4	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>
INF353	Technologies embarques et Mobiles (1)	8	4	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>
INF354	Gestions de donnees structurees	8	4	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>
INF355	Humanites et Entreprise	6	3	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>

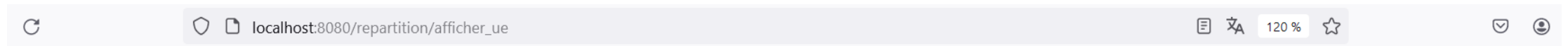
Showing 1 to 5 of 5 entries

Previous

1

Next

# Tester le MS Proxy



## La liste des UE

Ajouter UE

Show 10 entries

Search:

Code	Libelle	Credits	Coefficient	Operations	Details
INF351	Reseaux et Telecoms	8	4	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>
INF352	Genie Logiciel 1	8	4	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>
INF353	Technologies embarques et Mobiles (1)	8	4	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>
INF354	Gestions de donnees structurees	8	4	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>
INF355	Humanites et Entreprise	6	3	<a href="#">Modifier</a> <a href="#">Supprimer</a>	<a href="#">Voir</a>

Showing 1 to 5 of 5 entries

Previous

1

Next