Création du micro service "config-server" pour centraliser la configuration sur un git repository.

Code source du Ctrl1/5:

Git clone https://dydossy-admin@bitbucket.org/dydossy/2025.git
Branch : **feat/microservice-tp2-config-server**

Videos à voir:

- Introducing Spring Cloud Config Server - Microservice configuration with Spring Boot [10]
- Set up spring cloud config server from scratch - Microservice configuration with Spring Boot [11]
- Setting up spring cloud config client - Microservice configuration with Spring Boot [12]
- Dynamic config with spring Boot - Microservice configuration with Spring Boot [13]

Support de cours

https://bitbucket.org/dydossy/2025/src/TP_FORMATION_SPRING/

1. Créer un projet Parent Maven qui contiendra plusieurs modules par la suite.

Dans notre cas, le projet principal sera nommé "2025" a changer si vous voulez. Ce projet s'appellera aussi projet parent pour l'ensemble des futurs modules.
Le pom.xml de ce projet définit les versions Spring boot/ Spring cloud qui sera utilisée dans ce cours.

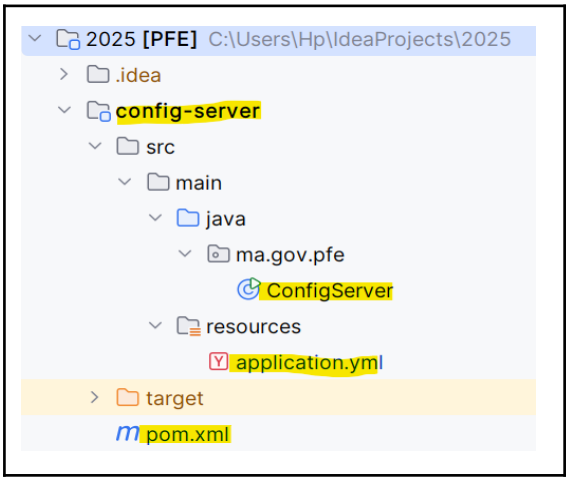
```
<project xmlns="http://maven.apache.org/POM/4.1.0">
    <modelVersion>4.0.0</modelVersion>
    <artifactId>PFE</artifactId>
    <packaging>pom</packaging>
    <modules>
        <module>config-server</module>
        <module>user-service</module>
        <module>eureka-server</module>
    </modules>
    <groupId>ma.gov.pfe</groupId>
    <version>SNAPSHOT1.0</version>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.7.18</version>
    </parent>
    <dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>org.projectlombok</groupId>
```

```
            <artifactId>lombok</artifactId>
            <version>1.18.26</version>
        </dependency>
        <dependency>
            <groupId>org.mapstruct</groupId>
            <artifactId>mapstruct</artifactId>
            <version>1.5.3.Final</version>
        </dependency>
        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-dependencies</artifactId>
            <version>2021.0.9</version>
            <type>pom</type>
            <scope>import</scope>
        </dependency>
    </dependencies>
</dependencyManagement>
<dependencies>

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

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>

    <dependency>
        <groupId>com.h2database</groupId>
        <artifactId>h2</artifactId>
    </dependency>

    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
    </dependency>

    <dependency>
        <groupId>org.mapstruct</groupId>
        <artifactId>mapstruct</artifactId>
    </dependency>

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

```
</project>
```

2. DANS VOTRE PROJET MAVEN, CRÉER LE MODULE "CONFIG-SERVER" POUR CENTRALISER LA CONFIGURATION DE TOUS LES MICRO-SERVICES.

LE MODULE "CONFIG-SERVER" SERA ATTACHÉ À VOTRE PROJET PRINCIPAL. LE BUT EST D'ARRIVER À LA STRUCTURE SUIVANTE:



3. MODIFIER LE POM.XML DU MODULE "CONFIG-SERVER" POUR AJOUTER LA DEPENDENCY **spring-cloud-config-server** COMME SUIVANT:

```
<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
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>ma.gov.pfe</groupId>
        <artifactId>MonPFE</artifactId>
        <version>SNAPSHOT-1.0</version>
    </parent>

    <artifactId>config-server</artifactId>

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

```
    </dependencies>

</project>
```

4. Pour que cette dépendance soit reconnue et importée, il faut ajouter **le Tag dependency management** dans le pom. Xml parent du projet principal.

```
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <version>1.18.26</version>
        </dependency>

        <dependency>
            <groupId>org.mapstruct</groupId>
            <artifactId>mapstruct</artifactId>
            <version>1.5.3.Final</version>
        </dependency>

        <dependency>
            <groupId>org.springdoc</groupId>
            <artifactId>springdoc-openapi-ui</artifactId>
            <version>1.7.0</version>
        </dependency>
        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-dependencies</artifactId>
            <version>2021.0.9</version>
            <scope>import</scope>
            <type>pom</type>
        </dependency>
    </dependencies>
</dependencyManagement>
```

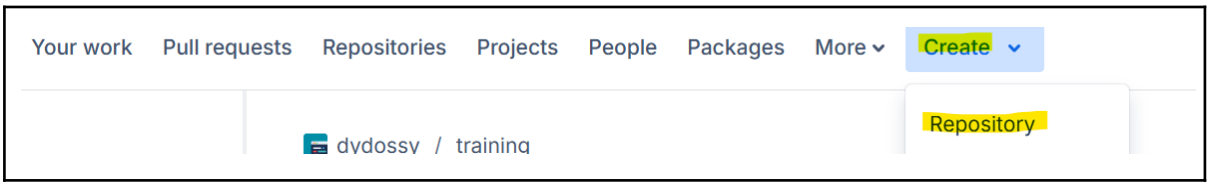
5. Dans le module “config-server”, créer le fichier de configuration “**application.yml**”.
   - Il faut le créer dans **src/main/resources.**
   - Dans ce fichier il faut mettre, le port, le nom de l’application, l’uri du repo git.

Ci-après la configuration à mettre dans ce fichier.

```
server:
```

```
 port: 8888

spring:
 application:
   name: config-server
 cloud:
   config:
     server:
       git:
         uri: https://bitbucket.org/dydossy/2025-config.git
         default-label: master
         clone-on-start: true
```

6. Créer un repository Git (sur BitBucket ou GitHub) pour centraliser les fichiers de configuration de tous les micros services qu'on va créer par la suite.
   - Aller sur https://bitbucket.org/
   - Créer un repository avec le nom "config-server". Laisser l'accessibilité "**public**" et créer la branche principale "**master**" au lieu de la branche par défaut "**main**"



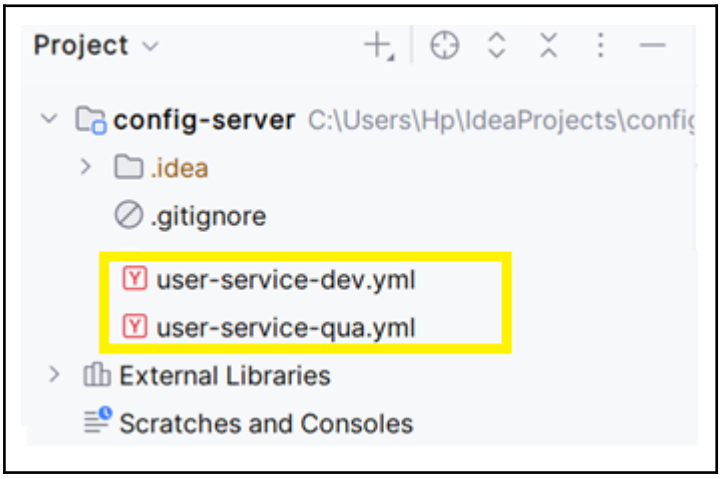
   - Cloner le projet sur un projet intellij

git clone https://dydossy-admin@bitbucket.org/dydossy/config-server.git

```
PS C:\Users\Hp\IdeaProjects> git clone https://dydossy-admin@bitbucket.org/dydossy/config-server.git
```

   - Après l'opération de clone, Ouvrez le projet cloné "config-server" sur intellij.

Noter que ce projet sert seulement à contenir les fichiers Yaml des micros services et il n' a aucune relation avec le module: config-server créé dans l'étape précédente.



7. AJOUTER LES FICHIERS DE CONFIGURATION DU MICRO SERVICE "USER-SERVICE" QUI SERA CRÉÉ PAR LA SUITE:

LE FICHIER DE CONFIGURATION DU MICRO SERVICE "USER-SERVICE" POUR L'ENV DU **DEV:** USER-SERVICE-**DEV**.YAML

```yaml
logging:
 level:
   org.springframework: INFO
   ma.gov.pfe: DEBUG
   reactor.netty: DEBUG
   reactor.core: DEBUG
   org.springframework.web.reactive: DEBUG
spring:
 datasource:
   driver-class-name: org.h2.Driver
   url:
jdbc:h2:file:~/users-${spring.profiles.active}-${server.port}
   serverName : localhost
   username: sa
   password : password

 jpa:
   generate-ddl: true
   show_sql: true
   hibernate:
     ddl_auto : create
 h2:
   console:
     enabled: true
     path: /h2-console
```

Le fichier de configuration du micro service "user-service" pour l'env du Qual: user-service-qua.yaml

```yaml
logging:
 level:
   org.springframework: INFO
   ma.gov.pfe: DEBUG

spring:
 datasource:
   driver-class-name: org.h2.Driver
   url:
jdbc:h2:file:~/users-${spring.profiles.active}-${server.port}
   serverName : localhost
   username: sa
   password : password

 jpa:
   generate-ddl: true
   show_sql: true
   hibernate:
     ddl_auto : create
 h2:
   console:
     enabled: true
     path: /h2-console
```

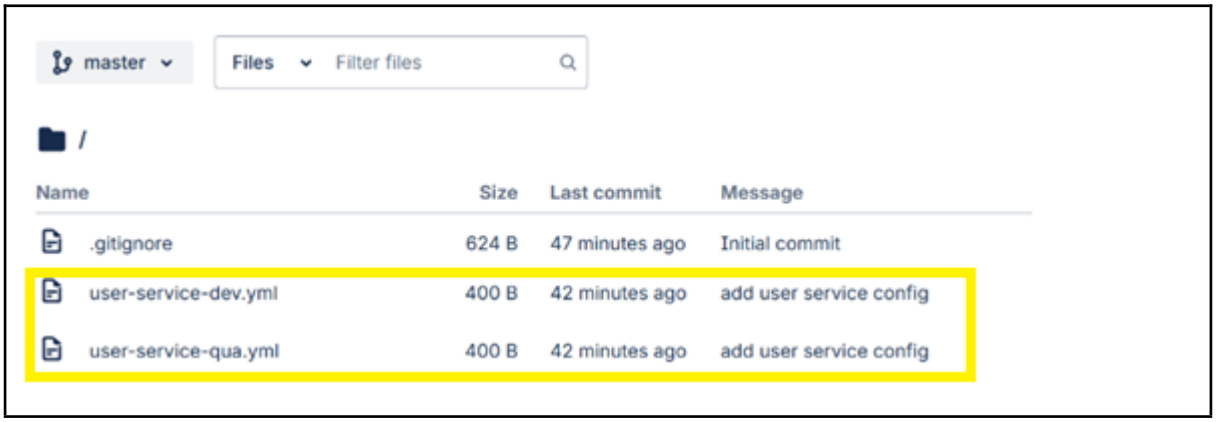
Après chaque modification de configuration, il faut faire un commit et un push vers origin

```
PS C:\Users\Hp\IdeaProjects\config-server> git commit -am "add user service config"
```

Une fois la config est ok, faire un push de la branche main ou master vers le repository remote.

```
PS C:\Users\Hp\IdeaProjects\config-server> git push origin main
```

Vérifier aussi si les fichiers pushes sont aussi sur le bitbucket.



8. Créer la classe de démarrage du micro service "config-service" dans src/main/java
   - La classe de démarrage doit être annotée par `@SpringBootApplication`
   - La classe de démarrage doit être annotée par `@EnableConfigServer`

```
package ma.gov.pfe.configserver;

import lombok.extern.slf4j.Slf4j;
import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplication;
import
org.springframework.cloud.config.server.EnableConfigServer;
import org.springframework.context.ApplicationContext;

@SpringBootApplication
@EnableConfigServer
@Slf4j
public class ConfigServerApplication {
    public static void main(String[] args) {
        SpringApplication.run(ConfigServerApplication.class);
    }
}
```

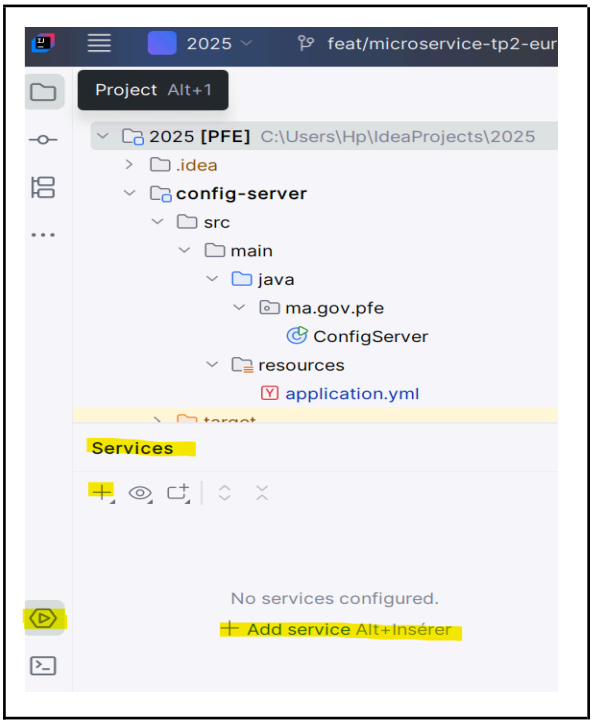
9. Activer les logs spring cloud config server et jgit pour voir le détail de la récupération de la configuration depuis bitbucket.

Ajouter ces ligne au fichier application.yaml du micro service "config-server"

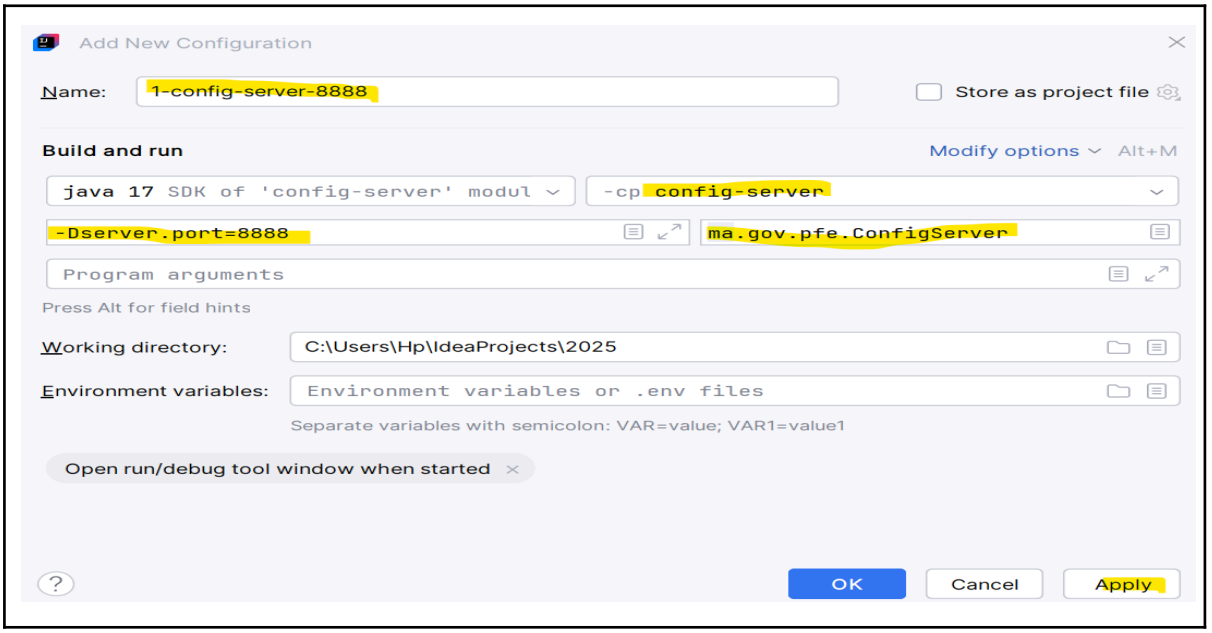
```
LOGGING:
  LEVEL:
    ORG.SPRINGFRAMEWORK.CLOUD.CONFIG.SERVER: DEBUG
    ORG.ECLIPSE.JGIT: DEBUG
```

```
nate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.H2Dialect
.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.pl
ntainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
nfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be per
s.GitCredentialsProviderFactory : No credentials provider required for URI https://dydossy-admin@bitbucket.org/dydossy/
torage.file.FileSnapshot        : file=null, size changed from -1 to 128 bytes
se.jgit.util.SystemReader       : loading config FileBasedConfig[C:\Users\Hp\.config\jgit\config]
torage.file.FileSnapshot        : file=C:\Users\Hp\.config\jgit\config, create new FileSnapshot: lastRead=2025-10-13 15
se.jgit.util.FS                 : readpipe [C:\Program Files\Git\bin\git.exe, --version],C:\Program Files\Git\bin
se.jgit.util.FS                 : readpipe may return 'git version 2.51.0.windows.1'
se.jgit.util.FS                 : remaining output:
```

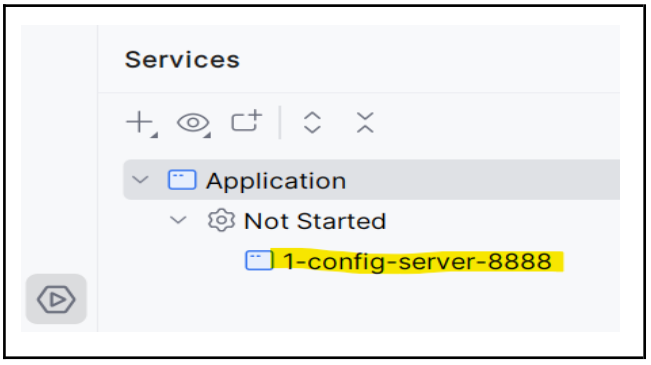
10. DÉMARRER LE MICRO SERVICE "CONFIG-SERVER" ET VÉRIFIER QUE LA CONFIGURATION EST BIEN RÉCUPÉRÉ DEPUIS LE GIT REPO. SUR LA CONSOLE DU MICRO SERVICE, LANCER "CONFIG-SERVER"



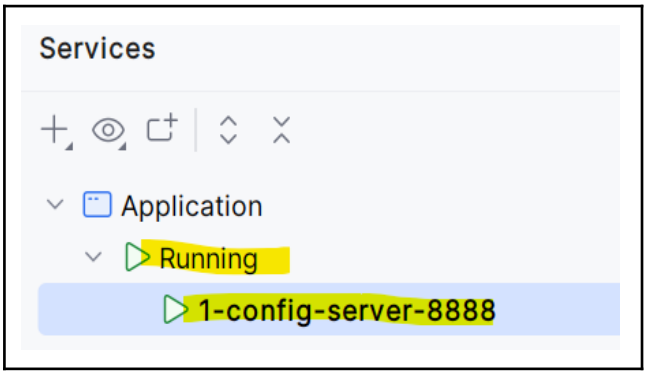
CLIQUER SUR "ADD SERVICE" PUIS "RUN CONFIGURATION" PUIS "APPLICATION"



CLIQUER SUR APPLY.



LANCER LE CONFIG SERVER “1-CONFIG-SERVER-8888”:



CHECK LES LOGS AU NIVEAU DE LA CONSOLE. VÉRIFIER L’ACCÈS AU REPO GIT

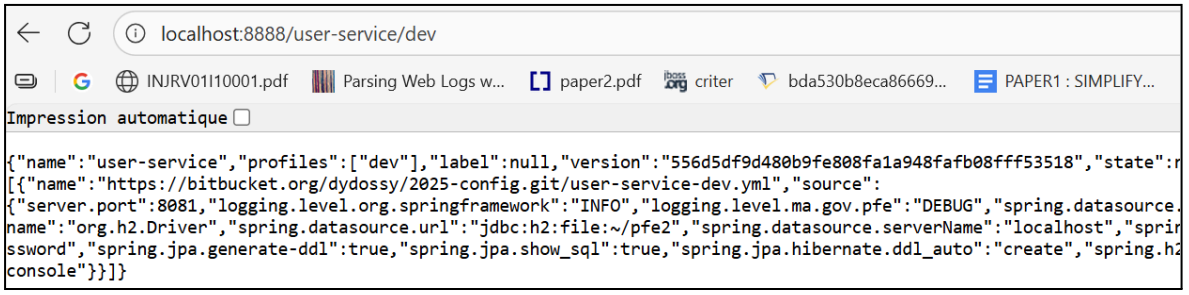
```
HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.No
Initialized JPA EntityManagerFactory for persistence unit 'default'
spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view
No credentials provider required for URI https://dydossy-admin@bitbucket.org/dydossy/config-server.git
file=null, size changed from -1 to 128 bytes
loading config FileBasedConfig[C:\Users\Hp\.config\jgit\config]
file=C:\Users\Hp\.config\jgit\config, create new FileSnapshot: lastRead=2025-11-04 16:24:03.822588100,
readpipe [C:\Program Files\Git\bin\git.exe, --version],C:\Program Files\Git\bin
```

CHECK LE PORT DE DÉMARRAGE DU SERVICE "CONFIG-SERVER", CA DOIT ETRE 8888

```
o.e.j.i.storage.file.FileSnapshot        : file=C:\Users\Hp\AppData\Local\Temp\config-repo-5700754022616
o.e.j.i.storage.file.FileSnapshot        : file=C:\Users\Hp\AppData\Local\Temp\config-repo-5700754022616
o.e.j.i.storage.file.FileSnapshot        : file=C:\Users\Hp\AppData\Local\Temp\config-repo-5700754022616
o.s.b.a.e.web.EndpointLinksResolver      : Exposing 1 endpoint(s) beneath base path '/actuator'
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8888 (http) with context path ''
ma.gov.pfe.ConfigServer                  : Started ConfigServer in 19.042 seconds (JVM running for 21.19
```

ACCÉDER AUX URLS SUIVANTES POUR TESTER LA CONFIGURATION.

- LA CONFIGURATION USER-SERVICE POUR L'ENVIRONNEMENT DE DEV: localhost:8888/user-service/dev



- LA CONFIGURATION USER-SERVICE POUR L'ENVIRONNEMENT DE QUAL: localhost:8888/user-service/qua