24-Développement Spring-Data

24-01-Gestion d'utilisateurs-Spring-Data

Sommaire

(01.) Introduction

(02.) Architecture de la future application

(03.) Création du squelette de projet

(04.) Configuration du module « parent »

(05.) Configuration des modules « enfants »

(06.) Configuration du contexte de l'application

(07.) Configuration des loggers de l'application

(08.) Modèle de données de l'application

(09.) Architecture de test de l'application

(10.) Compilation de l'application

(11.) Lancement des tests de l'application

(01.) Introduction :

L'objectif de ce document est de créer et de configurer le squelette d'un projet multi-modules « GestionDUtilisateurs-02 ».

Pour réaliser ce squelette d'application, nous utiliserons les éléments suivants :

- le gestionnaire de dépendances « APACHE-MAVEN »
- la bibliothèque logicielle « Spring-Boot »
- la bibliothèque logicielle « Spring-Data »
- la bibliothèque logicielle « Spring-Core »
- la bibliothèque logicielle « Spring-Web »

Ce squelette de projet sera utilisé pour réaliser une application multi-plateformes, de type «Web-Services», avec une architecture « N-Tiers ».

On remarque qu'une architecture N-Tiers satisfait à des normes et à des règles de fonctionnement (définition des responsabilités des différentes couches logicielles).

Par conséquent, le squelette de projet que nous allons réaliser devra être conforme à ces normes et à ces règles.

Bon courage à tous !! Et bonne rentrée ! ...

(02.) Architecture de la future application :

L'architecture de la future application sera de type « N-Tiers ». Cela signifie qu'elle sera constituée des composants applicatifs suivants :

(01.) Un Serveur de Gestion de Base de Données Relationnelle :

Responsable du stockage des données.
Pour le réaliser, nous choisissons le SGBDR « mariaDB ».

(02.) Un composant « model » (orthographe anglaise) **:**

Responsable de la création des objets « métier » (définis dans le Modèle Conceptuel de Données).
Pour le réaliser, nous choisissons la bibliothèque logicielle « Spring-Data ».

(02.) Un composant « persistence » (orthographe anglaise):

Responsable des opérations de persistance des objets « métier ».
Pour le réaliser, nous choisissons la bibliothèque logicielle « Spring-Data ».

(03.) Un composant « business » (orthographe anglaise):

Responsable des traitements « métier ».
Pour le réaliser, nous choisissons la bibliothèque logicielle « Spring-Core ».

(05.) Un composant « user interface » (orthographe anglaise) :

Responsable du dialogue Homme-Machine.
Pour le réaliser, nous choisissons la bibliothèque logicielle « Spring-Web ».

L'interdépendance entre ces composants applicatifs :

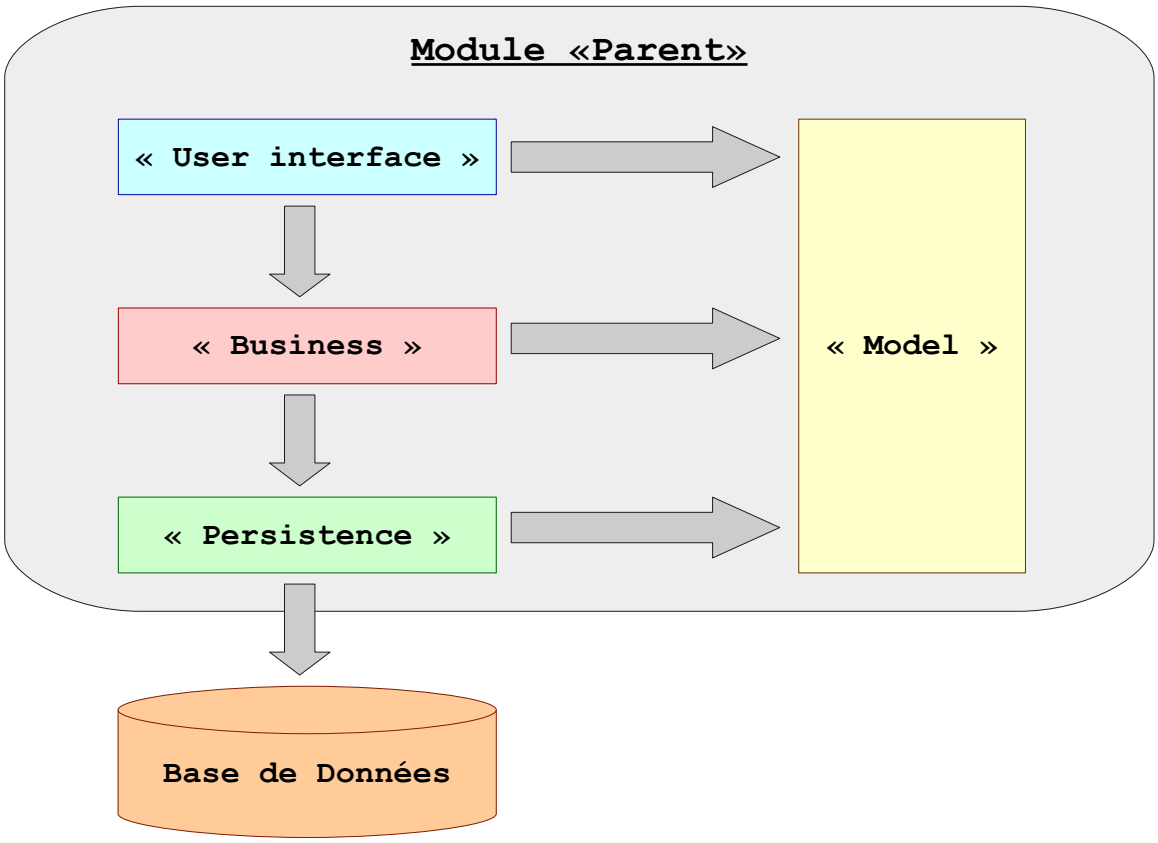
Elle est illustrée par le schéma ci-dessous :

(Toute ressemblance avec le tutoriel « maven » d'« openclassrooms » est un pur hasard !).

Remarque :

Les flèches entre 2 modules :

■ Chaque flèche représente l'ensemble de tous les appels entre les 2 modules concernés.
■ Chaque flèche va du module appelant vers le module appelé.



(03.) Création du squelette de projet :

La création de ce squelette de projet sera constituée des étapes suivantes :

- **La création d'un module « parent » .**
- **La configuration du module « parent » .**
- **La création du module «model».**
- **La création du module «persistence ».**
- **La création du module «business».**
- **La création du module «ui».**

Ces tâches sont détaillées dans les paragraphes ci-dessous.

(03.01.) Création du module «parent»

Pour créer le module « parent », on effectue les tâches ci-dessous :

(03.01.A.) Démarrer une invite de commande :

Lancer le programme « cmd.exe ».

(03.01.B.) Pointer sur le répertoire du workspace :

Lancer la commande :

```
cd 23-01-GestionDUtilisateurs-SpringBoot
```

(03.01.C.) Créer un répertoire pour les logs de génération:

Lancer la commande :

```
mkdir logGenerate
```

(03.01.D.) Lancer la commande de création du module « parent » :

Commande de création du module « parent » :

```
REM ********************************************************************************
REM * (01.)CREER LE MODULE 'PARENT' DE L'APPLICATION :
REM *
REM *      PROCEDURE : CREER UN MODULE 'PARENT' AVEC LES CARACTERISTIQUES SUIVANTES :
REM *
REM *      ->(01.01.)UN FICHIER 'POM.XML'.
REM *      ->(01.02.)UN ARCHETYPE 'QUICKSTART'.
REM ********************************************************************************
mvn archetype:generate ^
 -DgroupId=fr.afpa ^
 -DartifactId=GestionDUtilisateurs-02 ^
 -DarchetypeGroupId=org.apache.maven.archetypes ^
 -DarchetypeArtifactId=maven-archetype-quickstart ^
 -DarchetypeVersion=1.1 ^
 -Dversion=1.0-SNAPSHOT ^
 -DinteractiveMode=false ^
 -e -X ^
 > .\logGenerate\generate--GestionDUtilisateurs-02.log
```

(03.01.E.) Vérifier les logs de génération:

```
type .\logGenerate\generate--GestionDUtilisateurs-02.log
```

A la fin du fichier de log, on trouve la mention ci-dessous (en cas de succès) :

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

(03.02.) Configuration du module «parent»

Pour configurer le module « parent », on effectue les tâches ci-dessous :

(03.02.A.) Démarrer une invite de commande :

Lancer le programme « cmd.exe ».

(03.02.B.) Pointer sur le répertoire du module « parent » :

Lancer la commande :

```
REM ************************************************************************************
REM * (02.)CONFIGURER LE MODULE 'PARENT' CREE PRECEDEMMENT :
REM *
REM *      PROCEDURE : CONFIGURER LE MODULE 'PARENT' EN EFFECTUANT LES TACHES
SUIVANTES :
REM *
REM *      ->(02.01.)POINTER SUR LE REPERTOIRE DU MODULE PARENT CREE PRECEDEMMENT.
REM *      ->(02.02.)SUPPRIMER LE REPERTOIRE 'SRC' :
REM *      ->(02.03.)DANS LE FICHIER 'POM.XML' :
REM *                ->LA PROPRIETE 'PACKAGING' : DEFINIR SA VALEUR A 'POM' (ET NON
'JAR' !! A CORRIGER !!)
REM ************************************************************************************
cd GestionDUtilisateurs-02
```

(03.02.C.) Ouvrir le fichier « pom.xml » du module « parent »:

Le fichier « pom.xml » du module « parent » :

```
<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>


  <groupId>fr.afpa</groupId>
  <artifactId>GestionDUtilisateurs-02</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>


  <name>GestionDUtilisateurs-02</name>
  <url>http://maven.apache.org</url>


  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>


  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

Mise en forme du fichier « pom.xml » :

Effectuer la mise en forme du fichier comme indiqué ci-dessous (veuillez être attentifs aux sauts de lignes et aux indentations).

Modification du fichier « pom.xml » :

Dans la propriété « packaging » : définir la valeur « **pom** » (remplacer la valeur « **jar** »).

(03.02.D.) Créer un répertoire pour les logs de génération:

Lancer la commande :

```
mkdir logGenerate
```

(03.03.) Création du module « model »

Pour configurer le module «model», on effectue les tâches ci-dessous :

(03.03.A.) Démarrer une invite de commande :

Lancer le programme « cmd.exe ».

(03.03.B.) Pointer sur le répertoire du module «parent» :

Lancer la commande :

```
cd GestionDUtilisateurs-02
```

(03.03.C.) Lancer la commande de création du module «model» :

Commande de création du module «model» :

```
REM ************************************************************************************
REM * (03.)CREER LE MODULE 'MODEL'
REM *
REM *      PROCEDURE : CREER UN PROJET DE TYPE 'ENFANT' AVEC LES CARACTERISTIQUES
SUIVANTES :
REM *
REM *      ->(03.01.)POINTER SUR LE REPERTOIRE DU MODULE PARENT CREE PRECEDEMMENT.
REM *      ->(03.02.)UN FICHIER 'POM.XML' DE TYPE 'POM-ENFANT' (RENSEIGNER LA PROPRIETE
'PARENT').
REM *      ->(03.03.)UN ARCHETYPE 'QUICKSTART'.
REM *      ->(03.04.)DANS LE FICHIER 'POM.XML' :
REM *                ->LA PROPRIETE 'PACKAGING' : DEFINIR SA VALEUR A 'JAR'.
REM ************************************************************************************
mvn -B archetype:generate ^
 -DgroupId=fr.afpa ^
 -DartifactId=GestionDUtilisateurs-02-model ^
 -Dpackage=fr.afpa.model ^
 -DarchetypeGroupId=org.apache.maven.archetypes ^
 -DarchetypeArtifactId=maven-archetype-quickstart ^
 -DarchetypeVersion=1.1 ^
 -DinteractiveMode=false ^
 -e -X ^
 > .\logGenerate\generate--GestionDUtilisateurs-02-model.log
```

Remarque :

Le répertoire pour les logs de génération « logGenerate » : il a été créé au paragraphe (03.02.D).

(03.03.D.) Vérifier les logs de génération:

Lancer la commande :

```
type .\logGenerate\generate--GestionDUtilisateurs-02-model.log
```

A la fin du fichier de log, on trouve la mention ci-dessous (en cas de succès) :

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

(03.03.E.) Vérifier le fichier « pom.xml » du module « parent »:

La rubrique « modules » a été rajoutée et renseignée : elle est détaillée ci-dessous.

```
  <modules>
    <module>GestionDUtilisateurs-02-model</module>
  </modules>
```

(03.04.) Création du module « persistence »

Pour configurer le module «persistence», on effectue les tâches ci-dessous :

(03.04.A.) Démarrer une invite de commande :

Lancer le programme « cmd.exe ».

(03.04.B.) Pointer sur le répertoire du module «parent» :

Lancer la commande :

```
cd GestionDUtilisateurs-02
```

(03.04.C.) Lancer la commande de création du module «persistence» :

Commande de création du module «persistence» :

```
REM ************************************************************************************
REM * (04.)CREER LE MODULE 'PERSISTENCE'
REM *
REM *      PROCEDURE : CREER UN PROJET DE TYPE 'ENFANT' AVEC LES CARACTERISTIQUES
SUIVANTES :
REM *
REM *      ->(04.01.)POINTER SUR LE REPERTOIRE DU MODULE PARENT CREE PRECEDEMMENT.
REM *      ->(04.02.)UN FICHIER 'POM.XML' DE TYPE 'POM-ENFANT' (RENSEIGNER LA PROPRIETE
'PARENT').
REM *      ->(04.03.)UN ARCHETYPE 'QUICKSTART'.
REM *      ->(04.04.)DANS LE FICHIER 'POM.XML' :
REM *                ->LA PROPRIETE 'PACKAGING' : DEFINIR SA VALEUR A 'JAR'.
REM ************************************************************************************
mvn -B archetype:generate ^
 -DgroupId=fr.afpa ^
 -DartifactId=GestionDUtilisateurs-02-persistence ^
 -Dpackage=fr.afpa.persistence ^
 -DarchetypeGroupId=org.apache.maven.archetypes ^
 -DarchetypeArtifactId=maven-archetype-quickstart ^
 -DarchetypeVersion=1.1 ^
 -DinteractiveMode=false ^
 -e -X ^
 > .\logGenerate\generate--GestionDUtilisateurs-02-persistence.log
```

(03.04.D.) Vérifier les logs de génération:

Lancer la commande :

```
type .\logGenerate\generate--GestionDUtilisateurs-02-persistence.log
```

A la fin du fichier de log, on trouve la mention ci-dessous (en cas de succès) :

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

(03.04.E.) Vérifier le fichier « pom.xml » du module « parent »:

La rubrique « modules » a été modifiée : elle est détaillée ci-dessous.

```
  <modules>
    <module>GestionDUtilisateurs-02-model</module>
    <module>GestionDUtilisateurs-02-persistence</module>
  </modules>
```

(03.05.) Création du module « business »

Pour configurer le module «business», on effectue les tâches ci-dessous :

(03.05.A.) Démarrer une invite de commande :

Lancer le programme « cmd.exe ».

(03.05.B.) Pointer sur le répertoire du module «parent» :

Lancer la commande :

```
cd GestionDUtilisateurs-02
```

(03.05.C.) Lancer la commande de création du module «business» :

Commande de création du module «business» :

```
REM ************************************************************************************
REM * (05.)CREER LE MODULE 'BUSINESS'
REM *
REM *      PROCEDURE : CREER UN PROJET DE TYPE 'ENFANT' AVEC LES CARACTERISTIQUES
SUIVANTES :
REM *
REM *      ->(05.01.)POINTER SUR LE REPERTOIRE DU MODULE PARENT CREE PRECEDEMMENT.
REM *      ->(05.02.)UN FICHIER 'POM.XML' DE TYPE 'POM-ENFANT' (RENSEIGNER LA PROPRIETE
'PARENT').
REM *      ->(05.03.)UN ARCHETYPE 'QUICKSTART'.
REM *      ->(05.04.)DANS LE FICHIER 'POM.XML' :
REM *                ->LA PROPRIETE 'PACKAGING' : DEFINIR SA VALEUR A 'JAR'.
REM ************************************************************************************
mvn -B archetype:generate ^
 -DgroupId=fr.afpa ^
 -DartifactId=GestionDUtilisateurs-02-business ^
 -Dpackage=fr.afpa.business ^
 -DarchetypeGroupId=org.apache.maven.archetypes ^
 -DarchetypeArtifactId=maven-archetype-quickstart ^
 -DarchetypeVersion=1.1 ^
 -DinteractiveMode=false ^
 -e -X ^
 > .\logGenerate\generate--GestionDUtilisateurs-02-business.log
```

(03.05.D.) Vérifier les logs de génération:

Lancer la commande :

```
type .\logGenerate\generate--GestionDUtilisateurs-02-business.log
```

A la fin du fichier de log, on trouve la mention ci-dessous (en cas de succès) :

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

(03.05.E.) Vérifier le fichier « pom.xml » du module « parent »:

Le bloc « modules » a été modifié : il est détaillé ci-dessous.

```
  <modules>
    <module>GestionDUtilisateurs-02-model</module>
    <module>GestionDUtilisateurs-02-persistence</module>
    <module>GestionDUtilisateurs-02-business</module>
  </modules>
```

(03.06.) Création du module « ui »

Pour configurer le module «ui», on effectue les tâches ci-dessous :

(03.06.A.) Démarrer une invite de commande :

Lancer le programme « cmd.exe ».

(03.06.B.) Pointer sur le répertoire du module «parent» :

Lancer la commande :

```
cd GestionDUtilisateurs-02
```

(03.06.C.) Lancer la commande de création du module «ui» :

Commande de création du module «ui» :

```
REM ************************************************************************************
REM * (06.)CREER LE MODULE 'UI'
REM *
REM *      PROCEDURE : CREER UN PROJET DE TYPE 'ENFANT' AVEC LES CARACTERISTIQUES
SUIVANTES :
REM *
REM *      ->(06.01.)POINTER SUR LE REPERTOIRE DU MODULE PARENT CREE PRECEDEMMENT.
REM *      ->(06.02.)UN FICHIER 'POM.XML' DE TYPE 'POM-ENFANT' (RENSEIGNER LA PROPRIETE
'PARENT').
REM *      ->(06.03.)UN ARCHETYPE 'WEBAPP'.
REM *      ->(06.04.)DANS LE FICHIER 'POM.XML' :
REM *                ->LA PROPRIETE 'PACKAGING' : DEFINIR SA VALEUR A 'WAR'.
REM ************************************************************************************
mvn -B archetype:generate ^
 -DgroupId=fr.afpa ^
 -DartifactId=GestionDUtilisateurs-02-ui ^
 -Dpackage=fr.afpa.ui ^
 -DarchetypeGroupId=org.apache.maven.archetypes ^
 -DarchetypeArtifactId=maven-archetype-webapp ^
 -DarchetypeVersion=RELEASE ^
 -DinteractiveMode=false ^
 -e -X ^
 > .\logGenerate\generate--GestionDUtilisateurs-02-ui.log
```

(03.06.D.) Vérifier les logs de génération:

Lancer la commande :

```
type .\logGenerate\generate--GestionDUtilisateurs-02-ui.log
```

A la fin du fichier de log, on trouve la mention ci-dessous (en cas de succès) :

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

(03.06.E.) Vérifier le fichier « pom.xml » du module « parent »:

Le bloc « modules » a été modifié : il est détaillé ci-dessous.

```
  <modules>
    <module>GestionDUtilisateurs-02-model</module>
    <module>GestionDUtilisateurs-02-persistence</module>
    <module>GestionDUtilisateurs-02-business</module>
    <module>GestionDUtilisateurs-02-ui</module>
  </modules>
```

(04.) Configuration du module « parent » :

La configuration du module « parent » de ce projet consistera à mettre en forme, puis nettoyer et enrichir le fichier de configuration de ce module (le fichier « pom.xml »).

Cela sera effectué en mettant en forme, en supprimant et en rajoutant des blocs de code xml. Ces blocs sont listés ci-dessous (en tant que rubrique) :

- **(04.00.) La rubrique « Informations sur le projet parent »**
- **(04.01.) La rubrique « Informations sur le projet »**
- **(04.02.) La rubrique « Propriétés du projet »**
- **(04.03.) La rubrique « Modules du projet »**
- **(04.04.) La rubrique « Gestion des dépendances du projet »**
- **(04.05.) La rubrique « Construction de l'exécutable du projet »**

Ces rubriques sont détaillées dans les paragraphes ci-dessous.

(04.00.) La rubrique « Informations sur le projet parent»

La rubrique « Informations sur le projet » est déjà présente dans votre fichier de configuration du module « parent ». Elle est détaillée ci-dessous :

```
<!-- ============================================================ -->
<!-- (00.)Informations sur le projet parent -->
<!-- ============================================================ -->

<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.0.3.RELEASE</version>
  <relativePath /> <!-- lookup parent from repository -->
</parent>
```

Veuillez la récupérer intégralement.

(04.01.) La rubrique « Informations sur le projet »

La rubrique « Informations sur le projet » est déjà présente dans votre fichier de configuration du module « parent ». Elle est détaillée ci-dessous :

```
<!-- ========================================================== -->
<!-- (01.)Informations sur le projet -->
<!-- ========================================================== -->

<!-- ===== (01.01.)Informations d'identification du projet Maven ===== -->

<groupId>fr.afpa</groupId>
<artifactId>GestionDUtilisateurs-02</artifactId>
<version>1.0-SNAPSHOT</version>
<packaging>pom</packaging>

<!-- ===== (01.02.)Informations generales ===== -->

<name>GestionDUtilisateurs-02</name>
<url>http://maven.apache.org</url>
```

Vous pouvez vous en inspirer pour effectuer les rajouts de commentaires et la mise en forme dans votre fichier. Ou sinon, vous pouvez la récupérer intégralement.

(04.02.) La rubrique « Propriétés du projet »

La rubrique « Informations sur le projet » est déjà présente dans votre fichier de configuration du module « parent ». Elle est détaillée ci-dessous :

```
<!-- ============================================================ -->
<!-- (02.)Proprietes du projet -->
<!-- ============================================================ -->

<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
  <java.version>1.8</java.version>
</properties>
```

Vous devez l'enrichir, afin qu'elle soit comme ci-dessus.

(04.03.) La rubrique « Modules du projet»

La rubrique « Modules du projet » est déjà présente dans votre fichier de configuration du module « parent ». Elle est détaillée ci-dessous :

```
<!-- ============================================================ -->
<!-- (03.)Modules du projet -->
<!-- ============================================================ -->

<modules>
  <module>GestionDUtilisateurs-02-model</module>
  <module>GestionDUtilisateurs-02-persistence</module>
  <module>GestionDUtilisateurs-02-business</module>
  <module>GestionDUtilisateurs-02-ui</module>
</modules>
```

Elle est déjà renseignée comme ci-dessous (rien à changer) :

(04.04.) La rubrique « Gestion des dépendances du projet»

Concernant la rubrique « Gestion des dépendances du projet » :

■ Elle est absente dans votre fichier de configuration du module « parent » (balise « dependencyManagement »).
■ Elle se présente sous la forme ci-dessous.
■ Veuillez la récupérer intégralement.

```
<!-- ============================================================ -->
<!-- (04.)Gestion des dependances du projet -->
<!-- ============================================================ -->

<dependencyManagement>
  <dependencies>

    <!-- ===== (04.01.)Modules du projet ===== -->
    <!-- FIXME : A RENSEIGNER -->

    <!-- ===== (04.02.)Bibliothèques tierces ===== -->
    <!-- FIXME : A RENSEIGNER -->

  </dependencies>
<dependencyManagement>
```

Concernant ce bloc « dependencies» : Il est subdivisé en 7 parties listées ci-dessous :

■ (04.04.01.) La partie « Modules du projet »
■ (04.04.02.) La partie « Connecteurs de BDD »
■ (04.04.03.) La partie « Spring-Boot »
■ (04.04.04.) La partie « Thymeleaf-Spring-Security »
■ (04.04.05.) La partie « Apache-LOG4J-SLF4J-Logger »
■ (04.04.06.) La partie « Apache-Logger »
■ (04.04.07.) La partie « Tests Unitaires »

Ces parties sont détaillées dans les paragraphes ci-dessous.

(04.04.01.) La partie « Modules du projet » :

Dans cette partie « Modules du projet », vous devez rajouter une multitude de blocs « dependency ». Chacun de ces blocs « dependency » définit un module « enfant » qui devra être incorporé dans le module « parent ».

Voici les blocs « dependency » que vous devez rajouter. Veuillez les récupérer ci-dessous et les insérer à l'intérieur du bloc « dependencies ». :

```
        <!-- ==================================== -->
        <!-- ===== (04.01.)Modules du projet ===== -->
        <!-- ==================================== -->

        <dependency>
          <groupId>${project.groupId}</groupId>
          <artifactId>GestionDUtilisateurs-02-model</artifactId>
          <version>1.0-SNAPSHOT</version>
        </dependency>

        <dependency>
          <groupId>${project.groupId}</groupId>
          <artifactId>GestionDUtilisateurs-02-persistence</artifactId>
          <version>1.0-SNAPSHOT</version>
        </dependency>

        <dependency>
          <groupId>${project.groupId}</groupId>
          <artifactId>GestionDUtilisateurs-02-business</artifactId>
          <version>1.0-SNAPSHOT</version>
        </dependency>

        <dependency>
          <groupId>${project.groupId}</groupId>
          <artifactId>GestionDUtilisateurs-02-ui</artifactId>
          <version>1.0-SNAPSHOT</version>
        </dependency>
```

(04.04.02.) La partie « Connecteurs de BDD » :

Cette partie « Connecteurs de BDD » comporte le bloc « dependency » détaillé ci-dessous. Veuillez le récupérer et l'insérer à l'intérieur du bloc « dependencies ». :

```
<!-- ============================================================ -->
<!-- ====== (04.02.)Connecteurs de BDD ========================== -->
<!-- ============================================================ -->

<!-- Connecteur JDBC pour MYSQL -->
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>8.0.11</version>
</dependency>
```

(04.04.03.) La partie « Spring-Boot » :

Cette partie « Spring-Boot » comporte les blocs « dependency » détaillés ci-dessous. Veuillez les récupérer et les insérer à l'intérieur du bloc « dependencies » :

Le contenu de cette partie est présenté sur les 2 pages suivantes. (Veuillez le récupérer !!).

```
<!-- ============================================================ -->
<!-- ====== (04.03.)Spring-BOOT ================================= -->
<!-- ============================================================ -->

<!-- (04.03.01.)Spring-BOOT-Starter -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter</artifactId>
  <version>2.0.3.RELEASE</version>
  <exclusions>
    <exclusion>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-logging</artifactId>
    </exclusion>
  </exclusions>
</dependency>

<!-- (04.03.02.)Spring-BOOT-starter-Data-JPA -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
  <version>2.0.3.RELEASE</version>
</dependency>

<!-- (04.03.03.)Spring-BOOT-starter-data-REST -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-rest</artifactId>
  <version>2.0.3.RELEASE</version>
</dependency>

<!-- (04.03.04.)Spring-BOOT-starter-TOMCAT -->
<dependency>
```

```
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-tomcat</artifactId>
        <version>2.0.3.RELEASE</version>
        <scope>provided</scope>
    </dependency>

    <!-- (04.03.05.)Spring-BOOT-starter-test -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <version>2.0.3.RELEASE</version>
        <scope>test</scope>
    </dependency>

    <!-- (04.03.06.)Spring-BOOT-starter-Security : A DESACTIVER !!!-->
    <!-- dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-security</artifactId>
        <scope>runtime</scope>
        <version><FIXME : A RENSEIGNER !!></version>
    </dependency -->

    <!-- (04.03.07.)Spring-BOOT-starter-Thymeleaf -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-thymeleaf</artifactId>
        <version>2.0.3.RELEASE</version>
    </dependency>

    <!-- (04.03.08.)Spring-BOOT-starter-LOG4J2 -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-log4j2</artifactId>
        <version>2.0.3.RELEASE</version>
    </dependency>

    <!-- (04.03.09.)Spring-BOOT-DEVTOOLS -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-devtools</artifactId>
        <version>2.0.3.RELEASE</version>
        <scope>runtime</scope>
    </dependency>
```

Remarque :

Dans cette rubrique, tous les blocs relatifs à la sécurité sont provisoirement désactivés (= mis en commentaires).

(04.04.04.) La partie « Thymeleaf-Spring-Security » :

Cette partie « Thymeleaf-Spring-Security » comporte les blocs « dependency » détaillés ci-dessous. Veuillez les récupérer et les insérer à l'intérieur du bloc « dependencies » :

```
    <!-- ============================================================ -->
    <!-- ====== (04.04.)Thymeleaf-Spring-Security ================== -->
    <!-- ============================================================ -->
```

```
<!-- (04.04.01.)Thymeleaf-Spring-Security : A DESACTIVER !!! -->
<!-- dependency>
  <groupId>org.thymeleaf.extras</groupId>
  <artifactId>thymeleaf-extras-springsecurity4</artifactId>
  <scope>runtime</scope>
  <version><FIXME : A RENSEIGNER !!></version>
</dependency -->
```

Remarque :

Dans cette rubrique, tous les blocs relatifs à la sécurité sont provisoirement désactivés (= mis en commentaires).

(04.04.05.) La partie « Apache-LOG4J-SLF4J-Logger » :

Cette partie « Apache-LOG4J-SLF4J-Logger » comporte les blocs « dependency » détaillés ci-dessous. Veuillez les récupérer et les insérer à l'intérieur du bloc « dependencies » :

```
<!-- ============================================================ -->
<!-- ====== (04.05.)Apache-LOG4J-SLF4J-Logger ================== -->
<!-- ============================================================ -->

<!-- (04.05.01.)SLF4J-API -->
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>1.7.25</version>
</dependency>

<!-- (04.05.02.)Apache-LOG4J-SLF4J-IMPL -->
<dependency>
  <groupId>org.apache.logging.log4j</groupId>
  <artifactId>log4j-slf4j-impl</artifactId>
  <version>2.11.0</version>
</dependency>

<!-- (04.05.03.)SLF4J-LOG4J12 : A DESACTIVER !!! -->
<!-- dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-log4j12</artifactId>
  <scope><FIXME AJOUTER SI NECESSAIRE></scope>
  <version><FIXME : A RENSEIGNER !!></version>
</dependency -->
```

(04.04.06.) La partie « Apache-Logger » :

Cette partie « Apache-Logger » comporte les blocs « dependency » détaillés ci-dessous. Veuillez les récupérer et les insérer à l'intérieur du bloc « dependencies » :

```
<!-- ============================================================ -->
<!-- ====== (04.06.)Apache-Logger ================================ -->
<!-- ============================================================ -->

<!-- (04.06.01.)Apache-LOG4J-CORE -->
<dependency>
  <groupId>org.apache.logging.log4j</groupId>
  <artifactId>log4j-core</artifactId>
  <version>2.11.0</version>
</dependency>

<!-- (04.06.02.)Apache-LOG4J-API -->
<dependency>
  <groupId>org.apache.logging.log4j</groupId>
  <artifactId>log4j-api</artifactId>
  <version>2.11.0</version>
</dependency>

<!-- (04.06.03.)Apache-LOG4J-JCL -->
<dependency>
  <groupId>org.apache.logging.log4j</groupId>
  <artifactId>log4j-jcl</artifactId>
  <version>2.11.0</version>
</dependency>
```

(04.04.07.) La partie « Tests Unitaires » :

Cette partie « Tests-Unitaires» comporte le bloc « dependency » détaillé ci-dessous. Veuillez le récupérer et l'insérer à l'intérieur du bloc « dependencies » :

```
<!-- ============================================================ -->
<!-- ====== (04.07.)Tests unitaires =============================== -->
<!-- ============================================================ -->

<!-- JUnit -->
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.12</version>
  <scope>test</scope>
</dependency>
```

(04.05.) La rubrique « Construction de l'exécutable du projet»

Concernant la rubrique « Construction de l'exécutable du projet » :

- Elle est absente dans votre fichier de configuration du module « parent » (balise «build»).
- Elle se présente sous la forme ci-dessous.
- Veuillez la récupérer intégralement.

```
<!-- ========================================================== -->
<!-- (05.)Construction de l'exécutable du projet -->
<!-- ========================================================== -->

<build>
  <!-- ========================================================== -->
  <!-- (05.01.)Nom du fichier executable -->
  <!-- ========================================================== -->

  <!-- ========================================================== -->
  <!-- (05.02.)Gestionnaire de plugins -->
  <!-- ========================================================== -->

  <!-- ========================================================== -->
  <!-- (05.03.)Plugins -->
  <!-- ========================================================== -->

</build>
```

Cette rubrique comporte les 3 sous-rubriques suivantes (c'est le 2ème niveau d'imbrication !).

- (05.01.) Sous-rubrique « Nom du fichier executable »
- (05.02.) Sous-rubrique « Gestion des plugins »
- (05.03.) Sous-rubrique « Plugins »

Ces sous-rubriques sont détaillées dans les paragraphes ci-dessous :

(04.05.01.) La sous-rubrique « Nom du fichier executable »

Elle est présentée sous forme du bloc de données ci-dessous. (Veuillez le récupérer !).

```
    <!-- ========================================================== -->
    <!-- (05.01.)Nom du fichier executable -->
    <!-- ========================================================== -->

    <finalName>GestionDUtilisateurs-02</finalName>
```

(04.05.02.) La sous-rubrique « Gestion des plugins »

Elle est présentée sous forme du bloc de données ci-dessous. (Veuillez le récupérer !).

```
    <!-- ========================================================== -->
    <!-- (05.02.)Gestionnaire de plugins -->
    <!-- ========================================================== -->

    <pluginManagement>
      <!-- FIXME : A RENSEIGNER -->
    </pluginManagement>
```

Cette sous-rubrique comporte également une sous-sous-rubrique (c'est le 3ème niveau !) «plugins ». Celle-ci sera rajoutée sous la forme du bloc de données ci-dessous :

Veuillez le récupérer ci-dessous et l'insérer à l'intérieur du bloc « pluginManagement ».

```
      <plugins>
        <!-- FIXME : A RENSEIGNER -->
      </plugins>
```

Le contenu de ce bloc est présenté sur les 2 pages suivantes. (Veuillez le récupérer !!).

```
        <!-- (05.02.01.)Plug-in de nettoyage -->
        <plugin>
          <groupId>org.apache.maven.plugins</groupId>
          <artifactId>maven-clean-plugin</artifactId>
          <version>3.0.0</version>
        </plugin>

        <!-- (05.02.02.)Plugin de prise en compte des resources -->
        <plugin>
          <groupId>org.apache.maven.plugins</groupId>
          <artifactId>maven-resources-plugin</artifactId>
          <version>3.0.2</version>
        </plugin>

        <!-- (05.02.03.)Plug-in de compilation -->
        <plugin>
          <groupId>org.apache.maven.plugins</groupId>
          <artifactId>maven-compiler-plugin</artifactId>
```

```
            <version>3.7.0</version>
        </plugin>

        <!-- (05.02.04.)Plug-in infaillible -->
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-surefire-plugin</artifactId>
            <version>2.20.1</version>
        </plugin>

        <!-- (05.02.05.)Plugin de génération du war -->
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-war-plugin</artifactId>
            <version>3.2.0</version>
        </plugin>

        <!-- (05.02.06.)Plugin de generation du jar -->
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-jar-plugin</artifactId>
            <version>3.1.0</version>
        </plugin>

        <!-- (05.02.07.)Plug-in d'installation -->
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-install-plugin</artifactId>
            <version>2.5.2</version>
        </plugin>

        <!-- (05.02.08.)Plug-in de deploiement -->
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-deploy-plugin</artifactId>
            <version>2.8.2</version>
        </plugin>

        <!-- (05.02.09.)Plug-in d'interfacage entre Spring-Boot et Maven -->
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
            <version>2.0.3.RELEASE</version>
        </plugin>
```

(04.05.03.) La sous-rubrique « plugins »

Elle est présentée sous forme du bloc de données ci-dessous. (Veuillez le récupérer !).

```
    <!-- ========================================================== -->
    <!-- (05.03.)Plugins -->
    <!-- ========================================================== -->

    <plugins>
      <!-- Plug-in de compilation -->
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <configuration>
          <source>${java.version}</source>
          <target>${java.version}</target>
        </configuration>
      </plugin>
    </plugins>
```

(05.) Configuration des modules « enfants » :

Concernant la configuration des modules « enfants » de ce projet :

Pour chacun de ces modules, cela consistera à mettre en forme, puis compléter le fichier de configuration du module (le fichier « pom.xml »).

Les fichiers de configuration de tous les modules sont subdivisés en rubriques listées ci-dessous :

- **(05.01.) La rubrique « Information sur le parent du projet »**
- **(05.02.) La rubrique « Information sur le projet »**
- **(05.03.) La rubrique «Propriétés du projet »**
- **(05.04.) La rubrique «Dépendances du projet »**
- **(05.05.) La rubrique «Construction de l'exécutable du projet »**

Pour chacune de ces rubriques, les tâches de mise en forme et de complétion seront détaillées dans les paragraphes suivants.

(05.01.) La rubrique « Informations sur le parent du projet »

Concernant la rubrique « Informations sur le parent du projet » :

■ Elle doit être rajoutée manuellement dans le fichier de configuration de chacun des 4 modules « enfants ».
■ Elle est identique pour chacun des 4 modules « enfants ».
■ Veuillez la récupérer ci-dessous :

```
<!-- ============================================================ -->
<!-- (05.01.)Informations sur le parent du projet -->
<!-- ============================================================ -->

<parent>
  <groupId>fr.afpa</groupId>
  <artifactId>GestionDUtilisateurs-02</artifactId>
  <version>1.0-SNAPSHOT</version>
<parent>
```

(05.02.) La rubrique « Informations sur le projet »

Concernant la rubrique « Informations sur le projet » :

■ Elle est spécifique à chaque module « enfant ».
■ Elle est subdivisée en 2 parties : La partie « Informations d'identification du projet », et la partie « Informations générales ».
■ Elle est détaillée ci-dessous (pour chaque module « enfant»).
■ Veuillez remarquer ce qui les différencie (les éléments « artifactId », « packaging » et « name »).

(05.02.A.)La rubrique « Informations sur le projet » du module « model » :

```
<!-- ========================================================== -->
<!-- (02.)Informations sur le projet -->
<!-- ========================================================== -->

<!-- ===== (02.01.)Informations d'identification du projet ===== -->

<groupId>fr.afpa</groupId>
<artifactId>GestionDUtilisateurs-02-model</artifactId>
<version>1.0-SNAPSHOT</version>
<packaging>jar</packaging>

<!-- ===== (02.02.)Informations generales ===== -->

<name>GestionDUtilisateurs-02-model</name>
<url>http://maven.apache.org</url>
```

(05.02.B.) La rubrique « Informations sur le projet » du module « persistence » :

```
<!-- ========================================================== -->
<!-- (02.)Informations sur le projet -->
<!-- ========================================================== -->

<!-- ===== (02.01.)Informations d'identification du projet ===== -->

<groupId>fr.afpa</groupId>
<artifactId>GestionDUtilisateurs-02-persistence</artifactId>
<version>1.0-SNAPSHOT</version>
<packaging>jar</packaging>

<!-- ===== (02.02.)Informations generales ===== -->

<name>GestionDUtilisateurs-02-persistence</name>
<url>http://maven.apache.org</url>
```

(05.02.C.) La rubrique « Informations sur le projet » du module « business » :

```
<!-- ========================================================== -->
<!-- (02.)Informations sur le projet -->
<!-- ========================================================== -->

<!-- ===== (02.01.)Informations d'identification du projet ===== -->

<groupId>fr.afpa</groupId>
<artifactId>GestionDUtilisateurs-02-business</artifactId>
<version>1.0-SNAPSHOT</version>
<packaging>jar</packaging>

<!-- ===== (02.02.)Informations generales ===== -->

<name>GestionDUtilisateurs-02-business</name>
<url>http://maven.apache.org</url>
```

(05.02.D.) La rubrique « Informations sur le projet » du module «ui» :

```
<!-- ========================================================== -->
<!-- (02.)Informations sur le projet -->
<!-- ========================================================== -->

<!-- ===== (02.01.)Informations d'identification du projet ===== -->

<groupId>fr.afpa</groupId>
<artifactId>GestionDUtilisateurs-02-ui</artifactId>
<version>1.0-SNAPSHOT</version>
<packaging>war</packaging>

<!-- ===== (02.02.)Informations generales ===== -->

<name>GestionDUtilisateurs-02-ui</name>
<url>http://maven.apache.org</url>
```

(05.03.) La rubrique « Propriétés du projet »

Concernant la rubrique « Propriétés du projet » :

■ Elle est déjà présente dans le fichier de configuration de chacun des modules « enfant ». (balise « properties »)
■ Vous devez la vider dans chacun des modules « enfant ».
■ Veuillez insérer uniquement la rubrique vide ci-dessous.

```
<!-- ============================================================ -->
<!-- (03.)Proprietes du projet -->
<!-- ============================================================ -->
```

Remarque :

La rubrique « propriétés du projet » du module « parent » est transmise par héritage à tous les modules « enfant ».

(05.04.) La rubrique « Dépendances du projet»

Concernant la rubrique « Dépendances du projet » :

■ Elle est déjà présente dans le fichier de configuration de chacun des modules « enfant » (balise « dependencies »).
■ Elle est spécifique à chaque module « enfant ».
■ Elle se présente sous la forme ci-dessous.

```
<!-- ============================================================ -->
<!-- (04.)Dependances du projet -->
<!-- ============================================================ -->

<dependencies>
  <!-- FIXME : A RENSEIGNER -->
</dependencies>
```

Concernant ce bloc « dependencies» : Il est subdivisé en 7 parties listées ci-dessous (les mêmes que dans le « pom-parent »):

■ (05.04.01.) La partie « Modules du projet »
■ (05.04.02.) La partie « Connecteurs de BDD »
■ (05.04.03.) La partie « Spring-Boot »
■ (05.04.04.) La partie « Thymeleaf-Spring-Security »
■ (05.04.05.) La partie « Apache-LOG4J-SLF4J-Logger »
■ (05.04.06.) La partie « Apache-Logger »
■ (05.04.07.) La partie « Tests Unitaires »

Chacune de ces parties est détaillée dans les paragraphes suivants.

(05.04.01.) La partie « Modules du projet » :

Dans cette partie « Modules du projet », vous devez rajouter un bloc « dependency » spécifique, pour chaque module « enfant » appelé.

Pour le module « model » :

La partie « module du projet » est vide : Vous n'avez rien à rajouter !

Pour le module « persistence » :

Veuillez rajouter le bloc ci-dessous :

```
<!-- ===== (04.01.)Modules du projet ===== -->

<dependency>
   <groupId>${project.groupId}</groupId>
   <artifactId>GestionDUtilisateurs-02-model</artifactId>
</dependency>
```

Pour le module «business» :

Veuillez rajouter le bloc ci-dessous :

```
<!-- ===== (04.01.)Modules du projet ===== -->

<dependency>
   <groupId>${project.groupId}</groupId>
   <artifactId>GestionDUtilisateurs-02-persistence</artifactId>
</dependency>

<dependency>
   <groupId>${project.groupId}</groupId>
   <artifactId>GestionDUtilisateurs-02-model</artifactId>
</dependency>
```

Pour le module «ui» :

Veuillez rajouter le bloc ci-dessous :

```
    <!-- ===== (04.01.)Modules du projet ===== -->

    <dependency>
       <groupId>${project.groupId}</groupId>
       <artifactId>GestionDUtilisateurs-02-business</artifactId>
    </dependency>

    <dependency>
       <groupId>${project.groupId}</groupId>
       <artifactId>GestionDUtilisateurs-02-model</artifactId>
    </dependency>
```

(05.04.02.) La partie « Connecteurs de BDD » :

Cette partie « Connecteurs de BDD » est présente uniquement dans le module «persistence».

Vous devez le récupérer dans le pom-parent, en retenant uniquement les éléments suivants :

■ « group-id »
■ « artifact-id ».

Il en résulte le bloc ci-dessous. Veuillez le récupérer.

```
    <!-- ============================================================ -->
    <!-- ====== (04.02.)Connecteurs de BDD ========================== -->
    <!-- ============================================================ -->

    <!-- Connecteur JDBC pour MYSQL -->
    <dependency>
       <groupId>mysql</groupId>
       <artifactId>mysql-connector-java</artifactId>
    </dependency>
```

(05.04.03.) La partie « Spring-Boot » :

Cette partie « Spring-Boot » est présente identiquement dans tous les modules enfants.

Vous devez la récupérer dans le pom-parent, en retenant pour chaque bloc « dependency », uniquement les éléments suivants :

■ « group-id »
■ « artifact-id ».

Il en résulte les blocs ci-dessous. Veuillez les récupérer.

```
    <!-- ============================================================ -->
    <!-- ====== (04.03.)Spring-BOOT ================================= -->
    <!-- ============================================================ -->

    <!-- (04.03.01.)Spring-BOOT-Starter -->
    <dependency>
       <groupId>org.springframework.boot</groupId>
       <artifactId>spring-boot-starter</artifactId>
    </dependency>

    <!-- (04.03.02.)Spring-BOOT-starter-Data-JPA -->
    <dependency>
       <groupId>org.springframework.boot</groupId>
       <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>

    <!-- (04.03.03.)Spring-BOOT-starter-data-REST -->
    <dependency>
       <groupId>org.springframework.boot</groupId>
```

```
      <artifactId>spring-boot-starter-data-rest</artifactId>
    </dependency>

    <!-- (04.03.04.)Spring-BOOT-starter-TOMCAT -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-tomcat</artifactId>
    </dependency>

    <!-- (04.03.05.)Spring-BOOT-starter-test -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-test</artifactId>
    </dependency>

    <!-- (04.03.06.)Spring-BOOT-starter-Security : A IGNORER !!! -->

    <!-- (04.03.07.)Spring-BOOT-starter-Thymeleaf : A IGNORER !!! -->

    <!-- (04.03.08.)Spring-BOOT-starter-LOG4J2 -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-log4j2</artifactId>
    </dependency>

    <!-- (04.03.09.)Spring-BOOT-DEVTOOLS -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-devtools</artifactId>
    </dependency>
```

(05.04.04.) La partie « Thymeleaf-Spring-Security » :

Cette partie « Thymeleaf-Spring-Security » est présente identiquement, mais désactivée, dans tous les modules enfants.

Vous devez la récupérer dans le pom-parent, en retenant pour chaque bloc « dependency », uniquement le commentaire d'en-tête :

Il en résulte le bloc ci-dessous. Veuillez le récupérer.

```
    <!-- ============================================================ -->
    <!-- ====== (04.04.)Thymeleaf-Spring-Security ================== -->
    <!-- ============================================================ -->

    <!-- (04.04.01.)Thymeleaf-Spring-Security : A DESACTIVER !!! -->
```

(05.04.05.) La partie « Apache-LOG4J-SLF4J-Logger » :

Cette partie « Apache-LOG4J-SLF4J-Logger » est présente identiquement dans tous les modules enfants.

Vous devez la récupérer dans le pom-parent, en retenant pour chaque bloc « dependency », uniquement les éléments suivants :

■ « group-id »
■ « artifact-id ».

Il en résulte les blocs ci-dessous. Veuillez les récupérer.

```
    <!-- ============================================================ -->
    <!-- ====== (04.05.)Apache-LOG4J-SLF4J-Logger ================== -->
    <!-- ============================================================ -->

    <!-- (04.05.01.)SLF4J-API -->
    <dependency>
       <groupId>org.slf4j</groupId>
       <artifactId>slf4j-api</artifactId>
    </dependency>

    <!-- (04.05.02.)Apache-LOG4J-SLF4J-IMPL -->
    <dependency>
       <groupId>org.apache.logging.log4j</groupId>
       <artifactId>log4j-slf4j-impl</artifactId>
    </dependency>

    <!-- (04.05.03.)SLF4J-LOG4J12 -->
    <!-- dependency>
       <groupId>org.slf4j</groupId>
       <artifactId>slf4j-log4j12</artifactId>
    </dependency -->
```

(05.04.06.) La partie « Apache-Logger » :

Cette partie « Apache-Logger » est présente identiquement dans tous les modules enfants.

Vous devez la récupérer dans le pom-parent, en retenant pour chaque bloc « dependency », uniquement les éléments suivants :

■ « group-id »
■ « artifact-id ».

Il en résulte les blocs ci-dessous. Veuillez les récupérer.

```
    <!-- ============================================================ -->
    <!-- ====== (04.06.)Apache Logger ============================== -->
    <!-- ============================================================ -->
```

```
    <!-- (04.06.01.)Apache-LOG4J-CORE -->
    <dependency>
      <groupId>org.apache.logging.log4j</groupId>
      <artifactId>log4j-core</artifactId>
    </dependency>

    <!-- (04.06.02.)Apache-LOG4J-API -->
    <dependency>
      <groupId>org.apache.logging.log4j</groupId>
      <artifactId>log4j-api</artifactId>
    </dependency>

    <!-- (04.06.03.)Apache-LOG4J-JCL -->
    <dependency>
      <groupId>org.apache.logging.log4j</groupId>
      <artifactId>log4j-jcl</artifactId>
    </dependency>
```

(05.04.07.) La partie «Tests Unitaires» :

Cette partie « Apache-Logger » est présente identiquement dans tous les modules enfants.

Vous devez la récupérer dans le pom-parent, en retenant pour chaque bloc « dependency », uniquement les éléments suivants :

- « group-id »
- « artifact-id ».

Il en résulte le bloc ci-dessous. Veuillez le récupérer.

```
    <!-- ============================================================ -->
    <!-- ====== (04.07.)Tests unitaires ============================= -->
    <!-- ============================================================ -->

    <!-- JUnit -->
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
    </dependency>
```

(05.05.) La rubrique « Construction de l'exécutable du projet»

La rubrique « Construction de l'exécutable du projet » :

■ Elle est absente dans le fichier de configuration de tous les modules « enfant » (balise « build »).
■ Veuillez la récupérer ci-dessous.

```
<!-- ========================================================== -->
<!-- (05.)Construction de l'exécutable du projet -->
<!-- ========================================================== -->
<build>
  <!-- FIXME : A RENSEIGNER -->
</build>
```

Cette rubrique comporte les 3 sous-rubriques suivantes (c'est le 2ème niveau d'imbrication !).

■ (05.05.01.) Sous-rubrique « Nom du fichier exécutable »
■ (05.05.02.) Sous-rubrique « Plugins »
■ (05.05.03.) Sous-rubrique «Ressources»

Ces sous-rubriques sont détaillées dans les paragraphes ci-dessous :

(05.05.01.) La sous-rubrique « Nom du fichier exécutable »

■ Elle est présente dans le fichier de configuration de chaque module enfant.
■ Son contenu est spécifique à chaque module enfant.
■ Veuillez le récupérer ci-dessous !

```
    <!-- ============================================================ -->
    <!-- (05.01.)Nom du fichier executable -->
    <!-- ============================================================ -->

    <finalName>GestionDUtilisateurs-02-[module]</finalName>
```

Ici, vous devez remplacer le terme [module] par le nom du module « enfant » dans lequel vous vous trouvez.

(05.05.02.) La sous-rubrique « Plugins »

■ Elle est présente dans le fichier de configuration de chaque module enfant.
■ Son contenu est spécifique à chaque module enfant.
■ Veuillez le récupérer ci-dessous !

```
    <!-- ============================================================ -->
    <!-- (05.02.)Plugins -->
    <!-- ============================================================ -->

    <plugins>
      <!-- FIXME : A RENSEIGNER -->
    </plugins>
```

Cette sous-rubrique est spécifique à chaque module « enfant ».
Pour chaque module « enfant », son contenu est fourni séparément ci-dessous :

Sous-rubrique « plugins » pour tous les modules enfants :

Dans tous les modules enfants, la rubrique « Plugins » est vide.
Vous devez y rajouter cette rubrique vide, conformément à ci-dessous. (Veuillez la récupérer !).

```
    <!-- ============================================================ -->
    <!-- (05.02.)Plugins -->
    <!-- ============================================================ -->
```

(05.05.03.) La sous-rubrique « resources »

Concernant cette sous-rubrique :

■ Elle est présente identiquement dans le fichier de configuration de tous les modules « enfant ».
■ Veuillez la récupérer ci-dessous :

```
    <!-- ========================================================== -->
    <!-- (05.03.)Resources -->
    <!-- ========================================================== -->

    <resources>
      <resource>
        <directory>src/main/resources</directory>
        <filtering>true</filtering>
      </resource>
    </resources>
```

(06.) Configuration du contexte de l'application:

Toute application basée sur « Spring-Boot » possède les éléments suivants :

- (06.01.) Un « Fichier de propriétés de l'application ».
- (06.02.) Un « Contexte de l'application ».
- (06.03.) Un « Chargeur du contexte de l'application ».

Les paragraphes suivants indiquent :

- La manière d'intégrer ces éléments dans l'application
- La manière dont l'application utilise ces éléments.

(06.01.) Le « Fichier de propriétés de l'application » :

Concernant le « Fichier de propriétés de l'application » :

■ Il en existe un pour chaque module « enfant » de l'application.
■ Il se trouve dans l'exécutable du module considéré.

Nous allons procéder aux 2 tâches suivantes :

■ (06.01.01.) Créer le fichier de propriétés de l'application
■ (06.01.02.) Remplir le fichier de propriétés de l'application

(06.01.01.) Création du fichier de propriétés de l'application

Je vous propose de créer ce fichier conformément aux indications suivantes :

Module		GestionDUtilisateur-02-persistence
Répertoire		src\main\resources
Fichier		application.properties

Remarque :
Nous avons ainsi doté un seul module enfant d'un fichier de propriétés.

Question :
Qu'adviendra-t-il des autres modules enfant de notre application ?

Réponse :
Chacun des autres modules enfant de notre application sera doté ultérieurement d'un fichier de propriétés.

(06.01.02.) Remplissage du fichier de propriétés de l'application

Je vous propose de remplir ce fichier avec les informations suivantes :

```
spring.datasource.url=jdbc:mysql://localhost:3306/utilisateurdb?
createDatabaseIfNotExist=true

spring.datasource.username=root
spring.datasource.password=tcharou
spring.datasource.driverClassName=com.mysql.cj.jdbc.Driver

spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

logging.config=classpath:log4j2-spring.xml
```

Remarques :

■ Ce fichier contient une liste de couples « clés-valeurs ».
■ Chacun de ces couples « clés-valeurs » contient une donnée liée à l'environnement d'exécution du module considéré.
■ Veuillez adapter les valeurs à **VOTRE ENVIRONNEMENT D'EXECUTION !!!!**

(06.02.) Le « Contexte de l'application » :

Concernant le « Contexte de l'application » :

■ Il en existe un pour chaque module « enfant » de l'application.
■ C'est un objet de la classe « ApplicationContext »
■ Nous l'appellerons « applicationContext »
■ Il est créé dès le début de l'exécution du module considéré.
■ La durée de vie de cet objet est la durée de l'exécution du module considéré.

Remarques :

■ Cet objet contient une collection de type «Map » (avec des couples « clé-valeur »).
■ Cette collection servira à stocker les couples « clé-valeur » issus du fichier de propriétés de l'application.

(06.03.) Le « Chargeur de contexte de l'application » :

Concernant le « Chargeur du contexte de l'application »:

■ C'est un objet de la classe « ContextLoader »
■ Nous l'appellerons « contextLoader ».
■ Il est créé dès le début de l'exécution de l'application.
■ Il récupère le contenu du fichier de propriétés « application.properties ».
■ Il charge le contenu de ce fichier dans l'objet « applicationContext ».

Remarques :

■ Nous ne créerons pas ce « Chargeur de contexte de l'application ».
■ Nous créerons une classe principale qui sera chargée de créer ce « Chargeur de contexte » automatiquement.

Nous allons procéder aux 2 tâches suivantes :

■ (06.03.01.) Créer la classe principale de l'application
■ (06.03.02.) Remplir la classe principale de l'application

(06.03.01.) Création de la classe principale de l'application

Je vous propose de créer cette classe conformément aux indications suivantes :

Module		GestionDUtilisateur-02-persistence
Répertoire		src\main\java
Package		fr.afpa.persistence
Fichier		Application.java

Remarque :

Nous avons ainsi doté un seul module enfant d'une classe principale.

Question :

Qu'adviendra-t-il des autres modules enfant de notre application ?

Réponse :

Chacun des autres modules enfant de notre application sera doté ultérieurement d'une classe principale.

(06.03.02.) Remplissage de la classe principale de l'application

Je vous propose de remplir ce fichier avec le code ci-dessous:

```
package fr.afpa.persistence;


import org.springframework.boot.autoconfigure.SpringBootApplication;


/**
 * <b>CLASSE QUI IMPLEMENTE LA METHODE D'ENTREE DE L'APPLICATION</b>
 *
 * @author chat_roux
 */
@SpringBootApplication
public class Application {

}
```

Remarque :

Cette classe principale semble – terriblement – vide !!!

Question :

Est-ce grave, docteur ?

Réponse :

Ne vous inquiétez pas, le chargement du contexte de l'application réussira malgré cela !
Et vous aurez l'occasion d'enrichir cette classe ultérieurement !

(07.) Configuration des loggers de l'application :

Pour la plupart des applications d'entreprises, il est indispensable qu'elles fournissent des informations sur leur déroulement (correct ou incorrect).

Pour atteindre cet objectif, on fait en sorte que l'application produise elle-même ces informations durant son exécution : cela correspond aux activités de traçage et d'historisation, (« logging »).

On utilise pour cela un composant logiciel (« logger »), dont la fonctionnalité principale est d'écrire des messages dans un fichier d'historisation («log-file»).

Par conséquent, ce composant logiciel (« logger ») devra donc être raccordé à l'application dont il doit tracer et historiser le déroulement.

La bibliothèque logicielle « LOG4J2» offre un procédé simple pour raccorder une application « Spring-Boot » à un « logger ». Ce procédé utilise un fichier de configuration des « Loggers ».

Concernant ce « Fichier de configuration des Loggers de l'application » :

- Il en existe un pour chaque module « enfant » de l'application.
- Il se trouve dans l'exécutable du module considéré.

Je vous propose d'effectuer les 3 tâches suivantes :

- (07.01.) Créer un fichier de configuration des Loggers
- (07.02.) Remplir ce fichier de configuration
- (07.03.) Enregistrer ce fichier de configuration dans le fichier des propriétés.

(07.01.) Création du « Fichier de configuration des Loggers » :

Je vous propose de créer ce fichier conformément aux indications suivantes :

Module		GestionDUtilisateur-02-persistence
Répertoire		src\main\resources
Fichier		log4j2-spring.xml

Remarque :

Nous avons ainsi doté un seul module enfant d'un fichier de configuration des Loggers.

Question :

Qu'adviendra-t-il des autres modules enfant de notre application ?

Réponse :

Chacun des autres modules enfant de notre application sera doté ultérieurement d'un fichier de configuration des Loggers.

(07.02.) Remplissage du fichier de configuration des Loggers

Je vous propose de remplir ce fichier avec les informations suivantes :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Configuration>


<Configuration monitorInterval="60">

  <!-- ============================================== -->
  <!-- (00.) Proprietes generales -->
  <!-- ============================================== -->

  <!-- FIXME : A RENSEIGNER !!! -->


  <!-- ================================================================== -->
  <!-- (01.) Appenders :                                                  -->
  <!--   - (01.A.) Chaque Appender pointe sur le fichier qui lui est associe  -->
  <!--   - (01.B.) Chaque Appender ecrit dans ce fichier                  -->
  <!-- ================================================================== -->

  <!-- FIXME : A RENSEIGNER !!! -->


  <!-- ================================================================== -->
  <!-- (02.) Loggers :                                                    -->
  <!--   - (01.A.) Chaque Logger scanne le package qui lui est associe    -->
  <!--   - (01.B.) Chaque Logger récupère les logs qu'il trouve dans les classes -->
  <!-- ================================================================== -->

  <!-- FIXME : A RENSEIGNER !!! -->

</Configuration>
```

Dans ce fichier, nous remarquons un bloc « configuration ». Ce bloc est subdivisé en 3 rubriques :

- Rubrique (00.) Propriétés générales
- Rubrique (01.) Les « Appenders »
- Rubrique (02.) Les « Loggers »

Ces 3 rubriques sont détaillées dans les paragraphes ci-dessous.

(07.02.00.) La rubrique « Propriétés générales »

Concernant cette rubrique, vous devez :

■ Récupérer cette rubrique intégralement
■ Insérer cette rubrique dans le bloc « configuration ».

Elle détaillée ci-dessous.

```
<!-- ============================================== -->
<!-- (00.) Proprietes generales -->
<!-- ============================================== -->

<Properties>
  <Property name="log-path">./logExecute</Property>
</Properties>
```

(07.02.01.) La rubrique « Appenders »

Concernant cette rubrique, vous devez :

■ Récupérer cette rubrique intégralement
■ Insérer cette rubrique dans le bloc « configuration ».

Elle détaillée ci-dessous.

```
<!-- ============================================================================ -->
<!-- (01.) Appenders :                                                            -->
<!--   - (01.A.) Chaque Appender pointe sur le fichier qui lui est associe  -->
<!--   - (01.B.) Chaque Appender ecrit dans ce fichier                      -->
<!-- ============================================================================ -->

<Appenders>
  <Console name="Console-Appender" target="SYSTEM_OUT">
    <PatternLayout>
      <pattern>[%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n</pattern>
    </PatternLayout>
  </Console>

  <File name="Application-File-Appender"
    fileName="${log-path}/GestionDUtilisateurs-02-application.log">
    <PatternLayout>
      <pattern>[%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n</pattern>
    </PatternLayout>
  </File>

  <File name="SpringBoot-File-Appender"
    fileName="${log-path}/GestionDUtilisateurs-02-springBoot.log">
    <PatternLayout>
      <pattern>[%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n</pattern>
    </PatternLayout>
  </File>

  <File name="SpringContext-File-Appender"
    fileName="${log-path}/GestionDUtilisateurs-02-springContext.log">
    <PatternLayout>
      <pattern>[%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n</pattern>
    </PatternLayout>
  </File>
</Appenders>
```

(07.02.02.) La rubrique « Loggers »

Concernant cette rubrique, vous devez :

■ Récupérer cette rubrique intégralement
■ Insérer cette rubrique dans le bloc « configuration ».

Elle détaillée ci-dessous.

```
<!-- ================================================================================ -->
<!-- (02.) Loggers :                                                                  -->
<!--   - (02.A.) Chaque Logger scanne le package qui lui est associe                  -->
<!--   - (02.B.) Chaque Logger récupère les logs qu'il trouve dans les classes        -->
<!-- ================================================================================ -->

<Loggers>
  <Logger name="fr.afpa.persistence" level="all" additivity="true">
    <AppenderRef ref="Application-File-Appender" level="all" />
  </Logger>

  <Logger name="org.springframework.boot" level="info" additivity="true">
    <AppenderRef ref="SpringBoot-File-Appender" level="all" />
  </Logger>

  <Logger name="org.springframework.context" level="info" additivity="true">
    <AppenderRef ref="SpringContext-File-Appender" level="all" />
  </Logger>

  <Root>
    <AppenderRef ref="Console-Appender" level="all" />
  </Root>
</Loggers>
```

(08.) Modèle de données de l'application :

L'objectif de ce chapitre : c'est de créer le modèle de données de l'application.

Pour cela, vous allez effectuer les tâches ci-dessous :

- (08.01.) Créer des entités de l'application.
- (08.02.) Enrichir ces entités avec du code.

Ces 2 tâches sont détaillées dans les paragraphes ci-dessous.

(08.01.) Créer les entités de l'application

Vous devez créer les entités à l'emplacement défini ci-dessous :

Module		GestionDUtilisateur-02-model
Répertoire		src\main\java
Package		fr.afpa.model
Fichiers		Droit.java Personne.java Utilisateur.java

Pour que les entités soient reconnues comme telles par « JPA », vous les exigences ci-dessous doivent être remplies :

- **(08.01.01.) L'entité doit implémenter l'interface « Serializable »**
- **(08.01.02.) La déclaration de la classe doit porter l'annotation « @Entity »**

Illustration :

```
@Entity
public class Droit implements Serializable {
   ...
}
```

```
@Entity
public class Personne implements Serializable {
   ...
}
```

```
@Entity
public class Utilisateur implements Serializable {
   ...
}
```

(08.02.) Enrichir les entités de l'application

Nous allons fournir les détails de chacune des entités, dans les paragraphes ci-dessous :

(08.02.01.) Entité « Droit »

Veuillez rajouter les **attributs** ci-dessous dans l'entité « Droit » :

```
  /**
   * <b>OBJET QUI POSSEDE LES FONCTIONNALITES SUIVANTES : </b> <br/>
   * LES FONCTIONNALITES D'ECRITURE DE MESSAGES DE LOG DANS LA CONSOLE.
   */
  @Transient
  private static final Logger LOGGER = LoggerFactory.getLogger(Droit.class);

  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)// strategie de generation de la clé
primaire
  private long id;

  @Column(nullable = false, unique = true)
  private String libelle;

  @OneToMany(mappedBy = "droit")
  private Collection<Utilisateur> utilisateurs;
```

Veuillez rajouter les **constructeurs** ci-dessous dans **l'entité « Droit »** :

```
  /**
   * <b>CONSTRUCTEUR SANS ARGUMENTS</b>
   */
  public Droit() {

     super();
     Droit.LOGGER.info("CLASS : Droit -- METHOD : Droit -- ARGUMENTS : None -- BEGIN");
     Droit.LOGGER.info("CLASS : Droit -- METHOD : Droit -- ARGUMENTS : None - END");
}
```

```
  /**
   * <b>CONSTRUCTEUR DEFINI DE LA MANIERE SUIVANTE :<b><br/>
   * ->1 ARGUMENT POUR CHAQUE ATTRIBUT<br/>
   * ->CLE PRIMAIRE     : INCLUES<br/>
   * ->ASSOCIATIONS     : EXCLUES<br/>
   * ->TIMESTAMP        : EXCLUS<br/>
   *
   * @param pId
   * @param pLibelle
   */
  public Droit(final Long pId, final String pLibelle) {

    Droit.LOGGER.info(
        "CLASS : Droit -- METHOD : Droit -- ARGUMENTS : 1 PAR ATTRIBUT (CLE PRIMAIRE :
INCLUE / ASSOCIATIONS : EXCLUES) -- BEGIN");

    this.id = pId;
    this.libelle = pLibelle;

    Droit.LOGGER.info(
        "CLASS : Droit -- METHOD : Droit -- ARGUMENTS : 1 PAR ATTRIBUT (CLE PRIMAIRE :
INCLUE / ASSOCIATIONS : EXCLUES) -- END");
  }
```

```
  /**
   * <b>CONSTRUCTEUR DEFINI DE LA MANIERE SUIVANTE :<b><br/>
   * ->1 ARGUMENT POUR CHAQUE ATTRIBUT<br/>
   * ->CLE PRIMAIRE     : EXCLUE<br/>
   * ->ASSOCIATIONS     : EXCLUES<br/>
   * ->TIMESTAMP        : EXCLUS<br/>
   *
   * @param pLibelle
   */
  public Droit(final String pLibelle) {

    Droit.LOGGER.info(
        "CLASS : Droit -- METHOD : Droit -- ARGUMENTS : 1 PAR ATTRIBUT (CLE PRIMAIRE :
EXCLUE / ASSOCIATIONS : EXCLUES) -- BEGIN");

    this.libelle = pLibelle;

    Droit.LOGGER.info(
        "CLASS : Droit -- METHOD : Droit -- ARGUMENTS : 1 PAR ATTRIBUT (CLE PRIMAIRE :
EXCLUE / ASSOCIATIONS : EXCLUES) -- END");
  }
```

(08.02.02.) Entité «Personne»

Veuillez rajouter les **attributs** ci-dessous dans l'entité «Personne» :

```
  /**
   * <b>OBJET QUI POSSEDE LES FONCTIONNALITES SUIVANTES : </b> <br/>
   * LES FONCTIONNALITES d'ecriture de messages de log dans la console.
   */
  @Transient
  private static final Logger LOGGER = LoggerFactory.getLogger(Personne.class);

  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY) // strategie de generation de la clé primaire
  private long id;

  private String nom;
  private String prenom;

  @Column(nullable = false, unique = true)
  private String mail;

  @OneToMany(mappedBy="personne")
  private Collection<Utilisateur> utilisateurs;
```

Veuillez rajouter les **constructeurs** ci-dessous dans **l'entité «Personne»** :

```
  /**
   * <b>CONSTRUCTEUR SANS ARGUMENTS</b>
   */
  public Personne() {
     super();

     Personne.LOGGER.info("CLASS : Personne -- METHOD : Personne -- ARGUMENTS : None -- BEGIN");
     Personne.LOGGER.info("CLASS : Personne -- METHOD : Personne -- ARGUMENTS : None -- END");
  }
```

```
  /**
   * <b>CONSTRUCTEUR DEFINI DE LA MANIERE SUIVANTE :<b><br/>
   * ->1 ARGUMENT POUR CHAQUE ATTRIBUT<br/>
   * ->CLE PRIMAIRE     : INCLUES<br/>
   * ->ASSOCIATIONS     : EXCLUES<br/>
   * ->TIMESTAMP        : EXCLUS<br/>
   *
   * @param pId
   * @param pNom
   * @param pPrenom
   * @param pMail
   */
  public Personne(final Long pId,
            final String pNom, final String pPrenom, final String pMail) {

    Personne.LOGGER.info(
        "CLASS : Personne -- METHOD : Personne -- ARGUMENTS : 1 PAR ATTRIBUT (CLE
PRIMAIRE : INCLUE / ASSOCIATIONS : EXCLUES) -- BEGIN");

    this.id = pId;
    this.nom = pNom;
    this.prenom = pPrenom;
    this.mail = pMail;

    Personne.LOGGER.info(
        "CLASS : Personne -- METHOD : Personne -- ARGUMENTS : 1 PAR ATTRIBUT (CLE
PRIMAIRE : INCLUE / ASSOCIATIONS : EXCLUES) -- END");
  }
```

```
  /**
   * <b>CONSTRUCTEUR DEFINI DE LA MANIERE SUIVANTE :<b><br/>
   * ->1 ARGUMENT POUR CHAQUE ATTRIBUT<br/>
   * ->CLE PRIMAIRE     : EXCLUE<br/>
   * ->ASSOCIATIONS     : EXCLUES<br/>
   * ->TIMESTAMP        : EXCLUS<br/>
   *
   * @param pNom
   * @param pPrenom
   * @param pMail
   */
  public Personne(final String pNom,
            final String pPrenom, final String pMail) {

    Personne.LOGGER.info(
        "CLASS : Personne -- METHOD : Personne -- ARGUMENTS : 1 PAR ATTRIBUT (CLE
PRIMAIRE : EXCLUE / ASSOCIATIONS : EXCLUES) -- BEGIN");

    this.nom = pNom;
    this.prenom = pPrenom;
    this.mail = pMail;

    Personne.LOGGER.info(
        "CLASS : Personne -- METHOD : Personne -- ARGUMENTS : 1 PAR ATTRIBUT (CLE
PRIMAIRE : EXCLUE / ASSOCIATIONS : EXCLUES) -- END");
  }
```

(08.02.03.) Entité «Utilisateur»

Veuillez rajouter les **attributs** ci-dessous dans l'entité «Personne» :

```
  /**
   * <b>OBJET QUI POSSEDE LES FONCTIONNALITES SUIVANTES : </b> <br/>
   * LES FONCTIONNALITES d'ecriture de messages de log dans la console.
   */
  @Transient
  private static final Logger LOGGER = LoggerFactory.getLogger(Utilisateur.class);

  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY) // strategie de generation de la
clé primaire
  private Long id;

  @Column(nullable = false, unique = true)
  private String identifiant;

  @Column(nullable = false)
  private String motDePasse;

  private Date dateInscription;

  @ManyToOne
  private Personne personne;

  @ManyToOne
  private Droit droit;
```

Veuillez rajouter les **constructeurs** ci-dessous dans **l'entité «Utilisateur»** :

```
  /**
   * <b>CONSTRUCTEUR SANS ARGUMENTS</b>
   */
  public Utilisateur() {
     super();

     Utilisateur.LOGGER.info("CLASS : Utilisateur -- METHOD : Utilisateur -- ARGUMENTS :
None -- BEGIN");
     Utilisateur.LOGGER.info("CLASS : Utilisateur -- METHOD : Utilisateur -- ARGUMENTS :
None -- END");
  }
```

```
  /**
   * <b>CONSTRUCTEUR DEFINI DE LA MANIERE SUIVANTE :<b><br/>
   * ->1 ARGUMENT POUR CHAQUE ATTRIBUT<br/>
   * ->CLE PRIMAIRE     : INCLUES<br/>
   * ->ASSOCIATIONS     : INCLUES<br/>
   * ->TIMESTAMP        : EXCLUS<br/>
   *
   * @param pId
   * @param pIdentifiant
   * @param pMotDePasse
   * @param pPersonne
   * @param pDroit
   */
  public Utilisateur(final Long pId, final String pIdentifiant, final String pMotDePasse,
final Personne pPersonne,
       final Droit pDroit) {

    Utilisateur.LOGGER.info(
         "CLASS : Utilisateur -- METHOD : Utilisateur -- ARGUMENTS : 1 PAR ATTRIBUT (CLE
PRIMAIRE : INCLUE / ASSOCIATIONS : INCLUES) -- BEGIN");

    this.id = pId;

    this.identifiant = pIdentifiant;
    this.motDePasse = pMotDePasse;
    this.dateInscription = new Date();

    this.personne = pPersonne;
    this.droit = pDroit;

    Utilisateur.LOGGER.info(
         "CLASS : Utilisateur -- METHOD : Utilisateur -- ARGUMENTS : 1 PAR ATTRIBUT (CLE
PRIMAIRE : INCLUE / ASSOCIATIONS : INCLUES) -- END");
  }
```

```
  /**
   * <b>CONSTRUCTEUR DEFINI DE LA MANIERE SUIVANTE :<b><br/>
   * ->1 ARGUMENT POUR CHAQUE ATTRIBUT<br/>
   * ->CLE PRIMAIRE     : EXCLUE<br/>
   * ->ASSOCIATIONS     : EXCLUES<br/>
   * ->TIMESTAMP        : EXCLUS<br/>
   *
   * @param pIdentifiant
   * @param pMotDePasse
   */
  public Utilisateur(final String pIdentifiant, final String pMotDePasse) {

    Utilisateur.LOGGER.info(
         "CLASS : Utilisateur -- METHOD : Utilisateur -- ARGUMENTS : 1 PAR ATTRIBUT (CLE
PRIMAIRE : EXCLUE / ASSOCIATIONS : EXCLUES) -- BEGIN");

    this.identifiant = pIdentifiant;
    this.motDePasse = pMotDePasse;
    this.dateInscription = new Date();

    Utilisateur.LOGGER.info(
         "CLASS : Utilisateur -- METHOD : Utilisateur -- ARGUMENTS : 1 PAR ATTRIBUT (CLE
PRIMAIRE : EXCLUE / ASSOCIATIONS : EXCLUES) -- END");
  }
```

(09.) Architecture de test de l'application :

Toute application basée sur la bibliothèque logicielle « Spring-Boot » doit être pourvue d'une classe de test principale.

Concernant cette « Classe de test principale de l'application »:

- Elle est lancée (par « Maven ») à la fin de la compilation de l'application.
- Elle charge la « Classe principale de l'application » (que vous avez créée au paragraphe 06.03).
- Cette dernière crée le « Chargeur de contexte de l'application » (décrit au paragraphe 06.03.).

Nous allons procéder aux 2 tâches suivantes :

- (09.01) Créer la classe de test principale de l'application
- (09.02.) Remplir la classe de test principale de l'application
- (09.03) Créer la classe de test des dao de l'application
- (09.04.) Remplir la classe de test des dao de l'application

(09.01.) Création de la classe de test principale :

Je vous propose de créer cette classe conformément aux indications suivantes :

Module		GestionDUtilisateur-02-persistence
Répertoire		src\test\java
Package		fr.afpa.persistence
Fichier		ApplicationTest.java

Remarque :

Nous avons ainsi doté un seul module enfant d'une classe de test principale.

Question :

Qu'adviendra-t-il des autres modules enfant de notre application ?

Réponse :

Chacun des autres modules enfant de notre application sera doté ultérieurement d'une classe de test principale.

(09.02.) Remplissage de la classe de test principale

Concernant cette classe de test principale :

La déclaration de la classe de test principale doit porter les annotations ci-dessous. Veuillez les rajouter !!!

```
/**
 * <b>CLASSE QUI IMPLEMENTE LES FONCTIONNALITES SUIVANTES : </b><br/>
 * ->LE LANCEMENT DES TESTS DU MODULE " PERSISTENCE ".<br/>
 *
 * @author chat_roux
 */
@ComponentScan(basePackages={"fr.afpa.persistence"})
@RunWith(SpringRunner.class)
@SpringBootTest(classes=ApplicationTest.class)
public class ApplicationTest implements CommandLineRunner {
  ...
}
```

Cette classe de test principale est constituée des éléments suivants :

■ (09.02.01.) Un attribut « Logger »
■ (09.02.02.) Un attribut « daoTest »
■ (09.02.03.) Une méthode « run »
■ (09.02.04.) Une méthode « allTests »

Ces éléments sont détaillés dans les paragraphes ci-dessous.

(09.02.01.) L'attribut « Logger »

Cet attribut est détaillé dans le code ci-dessous. Veuillez récupérer ce code et l'insérer dans la classe de test principale.

```
/**
 * <b>OBJET QUI POSSEDE LES FONCTIONNALITES SUIVANTES : </b> <br/>
 * Les fonctionnalites d'ecriture de messages de log dans la console.
 */

private static final Logger LOGGER = LoggerFactory.getLogger(ApplicationTest.class);
```

(09.02.02.) L'attribut « daoTest »

Cet attribut est détaillé dans le code ci-dessous. Veuillez récupérer ce code et l'insérer dans la classe de test principale.

```
/**
 * <b>UN OBJET DE TYPE 'DAO-TEST'.</b>
 */
@Autowired
private DaoTest daoTest;
```

(09.02.03.) La méthode « run»

Cette méthode est détaillée dans le code ci-dessous. Veuillez récupérer ce code et l'insérer dans la classe de test principale.

```
/**
 * <b>METHODE PRINCIPALE DE LA CLASSE DE TEST PRINCIPALE</b>
 *
 * @param args Une liste d'arguments.
 */
@Override
public void run(final String... arg0) throws Exception {

   ApplicationTest.LOGGER.info( "+----------+--------------------------------------+");
   ApplicationTest.LOGGER.info( "| MODULE   | GestionDUtilisateurs-02-persistence |");
   ApplicationTest.LOGGER.info( "| CLASSE   | ApplicationTest                      |");
   ApplicationTest.LOGGER.info( "| METHODE  | run                                  |");
   ApplicationTest.LOGGER.info( "| POSITION | DEBUT                                |");
   ApplicationTest.LOGGER.info( "+----------+--------------------------------------+");

   ApplicationTest.LOGGER.info( "+----------+--------------------------------------+");
   ApplicationTest.LOGGER.info( "| MODULE   | GestionDUtilisateurs-02-persistence |");
   ApplicationTest.LOGGER.info( "| CLASSE   | ApplicationTest                      |");
   ApplicationTest.LOGGER.info( "| METHODE  | run                                  |");
   ApplicationTest.LOGGER.info( "| POSITION | FIN                                  |");
   ApplicationTest.LOGGER.info( "+----------+--------------------------------------+");
}
```

(09.02.04.) La méthode « allTests»

Cette méthode est détaillée dans le code ci-dessous. Veuillez récupérer ce code et l'insérer dans la classe de test principale.

```
/**
 * <b>METHODE QUI LANCE L'INTEGRALITE DES TESTS SUR LE MODULE.</b>
 */
@Test
public void allTests() {

   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");
   ApplicationTest.LOGGER.info( "| MODULE   | GestionDUtilisateurs-02-persistence |");
   ApplicationTest.LOGGER.info( "| CLASSE   | ApplicationTest                     |");
   ApplicationTest.LOGGER.info( "| METHODE  | allTests                            |");
   ApplicationTest.LOGGER.info( "| POSITION | DEBUT                               |");
   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");

   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");
   ApplicationTest.LOGGER.info( "| MODULE   | GestionDUtilisateurs-02-persistence |");
   ApplicationTest.LOGGER.info( "| CLASSE   | ApplicationTest                     |");
   ApplicationTest.LOGGER.info( "| METHODE  | allTests                            |");
   ApplicationTest.LOGGER.info( "| POSITION | FIN                                 |");
   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");
}
```

(09.03.) Création de la classe de test des dao :

Je vous propose de créer cette classe conformément aux indications suivantes :

Module		GestionDUtilisateur-02-persistence
Répertoire		src\test\java
Package		fr.afpa.persistence.dao
Fichier		DaoTest.java

(09.04.) Remplissage de la classe de test des dao :

Concernant cette classe de test des dao:

La déclaration de la classe de test des dao doit porter les annotations ci-dessous. Veuillez les rajouter !!!

```
/**
 * <b>CLASSE QUI IMPLEMENTE LES FONCTIONNALITES SUIVANTES : <b><br/>
 * ->L'ENSEMBLE DES TESTS SUR LES DAO DE L'APPLICATION.<br/>
 *
 * @author chat_roux
 */
@EntityScan(basePackages = { "fr.afpa.model" })
@Component
public class DaoTest implements CommandLineRunner {
  ...
}
```

Cette classe de test principale est constituée des éléments suivants :

- (09.04.01.) Un attribut « Logger »
- (09.04.02.) Une méthode « run »
- (09.04.03.) Une méthode « allTests »

Ces éléments sont détaillés dans les paragraphes ci-dessous.

(09.04.01.) L'attribut « Logger »

Cet attribut est détaillé dans le code ci-dessous. Veuillez récupérer ce code et l'insérer dans la classe de test principale.

```
/**
 * <b>OBJET QUI POSSEDE LES FONCTIONNALITES SUIVANTES : </b> <br/>
 * Les fonctionnalites d'ecriture de messages de log dans la console.
 */

private static final Logger LOGGER = LoggerFactory.getLogger(DaoTest.class);
```

(09.04.02.) La méthode « run»

Cette méthode est détaillée dans le code ci-dessous. Veuillez récupérer ce code et l'insérer dans la classe de test principale.

```
/**
 * <b>METHODE PRINCIPALE DE LA CLASSE DE TEST PRINCIPALE</b>
 *
 * @param args Une liste d'arguments.
 */
@Override
public void run(final String... arg0) throws Exception {

   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");
   ApplicationTest.LOGGER.info( "| MODULE   | GestionDUtilisateurs-02-persistence |");
   ApplicationTest.LOGGER.info( "| CLASSE   | DaoTest                             |");
   ApplicationTest.LOGGER.info( "| METHODE  | run                                 |");
   ApplicationTest.LOGGER.info( "| POSITION | DEBUT                               |");
   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");

   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");
   ApplicationTest.LOGGER.info( "| MODULE   | GestionDUtilisateurs-02-persistence |");
   ApplicationTest.LOGGER.info( "| CLASSE   | DaoTest                             |");
   ApplicationTest.LOGGER.info( "| METHODE  | run                                 |");
   ApplicationTest.LOGGER.info( "| POSITION | FIN                                 |");
   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");
}
```

(09.04.03.) La méthode « allTests»

Cette méthode est détaillée dans le code ci-dessous. Veuillez récupérer ce code et l'insérer dans la classe de test principale.

```
/**
 * <b>METHODE QUI LANCE L'INTEGRALITE DES TESTS SUR LE MODULE.</b>
 */
@Test
public void allTests() {

   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");
   ApplicationTest.LOGGER.info( "| MODULE   | GestionDUtilisateurs-02-persistence |");
   ApplicationTest.LOGGER.info( "| CLASSE   | DaoTest                             |");
   ApplicationTest.LOGGER.info( "| METHODE  | allTests                            |");
   ApplicationTest.LOGGER.info( "| POSITION | DEBUT                               |");
   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");

   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");
   ApplicationTest.LOGGER.info( "| MODULE   | GestionDUtilisateurs-02-persistence |");
   ApplicationTest.LOGGER.info( "| CLASSE   | DaoTest                             |");
   ApplicationTest.LOGGER.info( "| METHODE  | allTests                            |");
   ApplicationTest.LOGGER.info( "| POSITION | FIN                                 |");
   ApplicationTest.LOGGER.info( "+----------+-------------------------------------+");
}
```

(10.) Compilation de l'application:

La compilation de ce squelette de projet sera effectuée en suivant la procédure suivante :

■ (10.01.) Créer un sous-répertoire pour les logs de compilation

■ (10.02.) Créer les sous-répertoires pour les logs d'exécution

■ (10.03.) Démarrer le serveur de base de données.

■ (10.04.) Lancer la compilation du projet.

■ (10.05.) Vérifier le contenu des logs de compilation

■ (10.06.) Vérifier le contenu des logs d'exécution

■ (10.07.) Vérifier la présence de la base de données

(10.01.) Créer un sous-répertoire pour les logs de compilation

Ce sous-répertoire pour les logs d'exécution doit être créé dans le module parent.

Veuillez créer ce sous-répertoire conformément aux indications suivantes :

Module		GestionDUtilisateur-02
Chemin		[A la racine]
Sous-répertoire		logCompile

(10.02.) Créer les sous-répertoires pour les logs d'exécution

Dans chaque module enfant, un sous-répertoire pour les logs d'exécution doit être créé.

Dans chaque module enfant, veuillez créer ce sous-répertoire conformément aux indications suivantes :

Module		GestionDUtilisateur-02-[module enfant]
Chemin		[A la racine]
Sous-répertoire		logExecute

(10.03.) Démarrer le serveur de base de données

Effectuer les actions listées ci-dessous :

■ (10.03.01.) Lancer une invite de commande (« cmd.exe »).
■ (10.03.01.) Pointer sur le sous-répertoire « bin » de « mariadb ».
■ (10.03.01.) Lancer la commande de démarrage du serveur « mariadb ».

(10.04.) Lancer la compilation du projet

Effectuer les actions listées ci-dessous :

- (10.04.01.) Lancer une invite de commande (« cmd.exe »).
- (10.04.02.) Pointer sur le répertoire du module parent.
- (10.04.02.) Lancer la commande « Maven » de compilation du projet.

Choisir l'une des 3 commandes suivantes (en fonction de l'heure de la journée) :

```
REM ************************************************************************************
REM * (01.)EFFECTUER LES OPERATIONS SUIVANTES :
REM *
REM *       ->NETTOYER LE PROJET
REM *       ->COMPILER LE PROJET (GENERER LES CLASSES EN BYTE-CODE).
REM *       ->GENERER LE FICHIER EXECUTABLE DU PROJET.
REM *       ->COPIER LE FICHIER EXECUTABLE DANS LE DEPOT MAVEN LOCAL.
REM ************************************************************************************
mvn clean install ^
 -e -X ^
 > .\logCompile\compile--GestionDUtilisateurs-02--%date:~6,4%.%date:~3,2%.%date:~0,2%--
0%time:~1,1%.%time:~3,2%.%time:~6,2%.log

REM ************************************************************************************
mvn clean install ^
 -e -X ^
 > .\logCompile\compile--GestionDUtilisateurs-02--%date:~6,4%.%date:~3,2%.%date:~0,2%--
%time:~0,2%.%time:~3,2%.%time:~6,2%.log

REM ************************************************************************************
mvn clean install ^
 -e -X ^
 > .\logCompile\compile—GestionDUtilisateurs-02.log

REM ************************************************************************************
```

(10.05.) Vérifier le contenu des logs de compilation

```
type .\logCompile\compile-GestionDUtilisateurs-02.log
```

A la fin du fichier de log, on trouve la mention ci-dessous (en cas de succès) :

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

(10.06.) Vérifier le contenu des logs d'exécution

Dans le cas d'une compilation terminée avec succès :

Dans chaque module « enfant » :

■ **(10.06.01.) Ouvrir le sous-répertoire « logExecute».**

■ **(10.06.02.) Dans ce sous-répertoire « logExecute» :**
Vérifier la présence d'un fichier de log (fichier «.log»).

■ **(10.06.03.) Dans ce fichier de log :**
Vérifier le contenu.

Remarque :
La vérification ci-dessus sera effectuée **uniquement dans le module « persistence » !!!**

(10.07.) Vérifier la présence de la base de données

■ (10.07.01.) Lancer une invite de commande (« cmd.exe »).
■ (10.07.02.) Pointer sur le sous-répertoire « bin » de « mariadb ».
■ (10.07.03.) Lancer la commande de connexion avec le serveur « mariadb ».
■ (10.07.04.) Lancer l'instruction SQL d'affichage de la liste des BDD.

(11.) Lancement des tests de l'application :

L'objectif des tests de l'application :

C'est de répertorier, décrire et reproduire tous les cas d'erreurs qui pourraient survenir lors de la compilation du projet.

J'ai dressé ci-dessous une liste des cas d'erreur possibles (liste non exhaustive !) :

- Désactivation de l'annotation « @SpringBootApplication »
- Désactivation des annotations « @RunWith » et « @SpringBootTest »
- Désactivation de l'annotation « @SpringBootTest »
- Désactivation de la dépendance « mysql-connector-java »

Ces cas d'erreur sont détaillés dans les paragraphes ci-dessous.

(11.01.) Désactivation de l'annotation « @SpringBootApplication »

- **(11.01.01.) Dans la classe « Application » :**
  Désactiver l'annotation « @SpringBootApplication ».

- **(11.01.02.) Relancer la compilation du projet.**

- **(11.01.03.) Vérifier les points suivants :**
  La base de données n'a pas été créée.

(11.02.) Désactivation des annotations « @RunWith» et « @SpringBootTest »

■ **(11.02.01.) Dans la classe « ApplicationTest » :**
Désactiver individuellement les annotations « @RunWith » et « @SpringBootTest »

■ **(11.02.02.) Relancer la compilation du projet.**

■ **(11.02.03.) Vérifier les points suivants :**
La base de données n'a pas été créée.

(11.03.) Désactivation de l'annotation « @Test »

■ **(11.03.01.) Dans la classe « ApplicationTest » :**
Sur la méthode « AllTests » : Désactiver l'annotation « @Test »

■ **(11.03.02.) Relancer la compilation du projet.**

■ **(11.03.03.) Vérifier les points suivants :**
La base de données n'a pas été créée.
La compilation du projet a échoué.

(11.04.) Désactivation de la dépendance « mysql-connector-java »

■ **(11.04.01.) Dans le module « persistence » :**
Dans la rubrique « Dependencies » : Désactiver le bloc « mysql-connector-java »

■ **(11.04.02.) Relancer la compilation du projet.**

■ **(11.04.03.) Vérifier les points suivants :**
La base de données n'a pas été créée.
La compilation du projet a échoué.