TP Maven

Objectif du TP

Développer une application Java en ligne de commande permettant de gérer un stock de produits (ajout, suppression, mise à jour, consultation), en utilisant **Maven** pour la gestion du projet

---

🛠 Technologies et outils requis

- **Java 11 ou supérieur**
- **Maven**
- **IDE** : IntelliJ IDEA
- Utilisation de la **console** pour l'interface utilisateur

---

1. Créer un nouveau projet Maven

1. Ouvrez IntelliJ IDEA.
2. Cliquez sur **File > New > Project**.
3. Sélectionnez **Maven** dans la liste des types de projets.
4. Renseignez les champs :
    - **GroupId** : par exemple, com.exemple
    - **ArtifactId** : par exemple, gestion-stock
5. Cliquez sur **Next**, puis sur **Finish**.

2. Structure du projet

Le projet suivra l'architecture standard d'un projet Maven :

3. Définition de la classe Produit

- Créer la classe Produit avec les attributs suivants :
  - id (entier)
  - nom (chaîne de caractères)
  - quantite (entier)
  - prix (double)
- Inclure les constructeurs, les accesseurs (getters/setters) et la méthode toString().

4. Implémentation du service de gestion de stock

- Créer la classe GestionStockService contenant :
  - Une liste de produits en mémoire (List<Produit>).
  - Des méthodes pour :
    - Ajouter un produit.
    - Supprimer un produit par ID.
    - Mettre à jour un produit.
    - Afficher tous les produits.
    - Rechercher un produit par nom ou ID.

5. Création de l'interface utilisateur en console

- Dans la classe Main, implémenter un menu interactif permettant à l'utilisateur de :
  - Ajouter un nouveau produit.
  - Supprimer un produit existant.
  - Mettre à jour les informations d'un produit.
  - Afficher la liste des produits.
  - Rechercher un produit.
  - Quitter l'application.

6. Tests et validation

- Tester chaque fonctionnalité pour s'assurer de leur bon fonctionnement.
- Gérer les cas d'erreurs et les entrées invalides.

7. Configurer Maven dans les paramètres de l'IDE

1. Allez dans **File > Settings** (ou **IntelliJ IDEA > Preferences** sur macOS).
2. Naviguez jusqu'à **Build, Execution, Deployment > Build Tools > Maven**.
3. Dans **Maven home directory**, sélectionnez :
   - **Bundled (Maven 3)** pour utiliser la version intégrée.
   - Ou spécifiez le chemin vers une installation Maven personnalisée.
4. Cliquez sur **OK** pour enregistrer les modifications.

8. Ajouter une configuration d'exécution Maven

1. Cliquer sur ‘Current file’.

2. Cliquez sur ‘edit configuration ….’
3. Dans le champ **Name**, donnez un nom à votre configuration, par exemple, Run GestionStock.
4. Dans le champ **Command line**, entrez les objectifs Maven que vous souhaitez exécuter, tels que clean install.
5. Cliquez sur **OK** pour enregistrer la configuration.

9. Commandes Maven essentielles

Voici quelques commandes Maven de base que vous utiliserez fréquemment :

- **mvn clean** : Supprime le répertoire target pour nettoyer les fichiers compilés précédemment.
- **mvn compile** : Compile le code source du projet.
- **mvn test** : Exécute les tests unitaires du projet.
- **mvn package** : Compile le code et empaquette le projet dans un fichier JAR ou WAR.
- **mvn install** : Installe le fichier JAR ou WAR dans le référentiel local Maven.
- **mvn clean install** : Nettoie le projet, compile, teste et installe le paquet dans le référentiel local.

10. **Ajouter la dépendance JDBC MySQL dans pom.xml**

```
<dependencies>
  <!-- JDBC Driver pour MySQL -->
  <dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.33</version>
  </dependency>
</dependencies>
```

11. Créer la base de données et la table produits depuis phpmyadmin

```
CREATE DATABASE gestion_stock;
USE gestion_stock;

CREATE TABLE produits (
    id INT PRIMARY KEY AUTO_INCREMENT,
    nom VARCHAR(100) NOT NULL,
    quantite INT NOT NULL,
    prix DOUBLE NOT NULL
);
```

12. Créer une classe ConnexionBD pour se connecter à MySQL

13. Créer GestionStockServiceJdbc.java permettant d’ajouter un produit

14. Modifier la classe main

15. Lancer la commande clean install