|  |  |
| --- | --- |
| jAVA EE  TP | **tp**  Notre objectif dans ce TP est:  1. L’objectif de ce TP est de savoir comment rendre un projet java ferme à la modification et ouvert à l’extension en utilisant couplage faible.  2. Injection des dépendances en utilisant Framework spring et sans spring.  **Réalise Par**  REDA TAKHDATE |

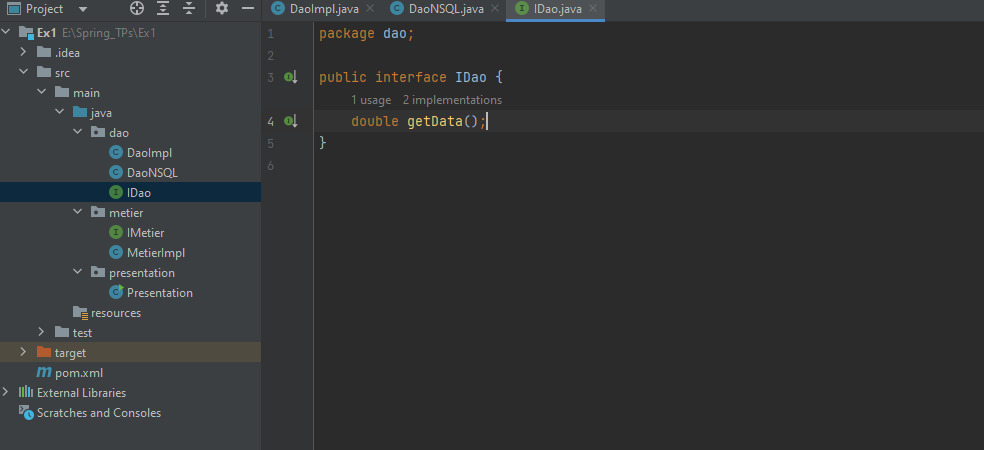
Projet:

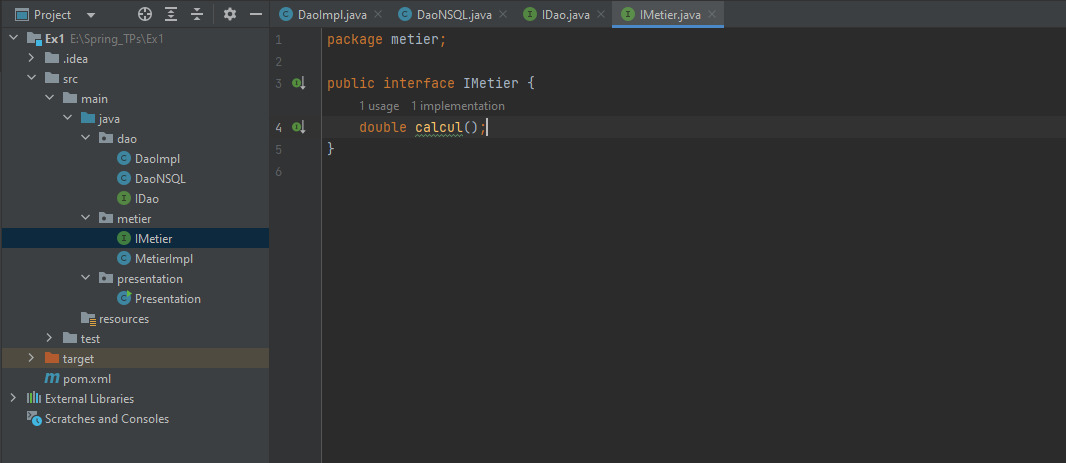
Ce projet contient 3 couches :

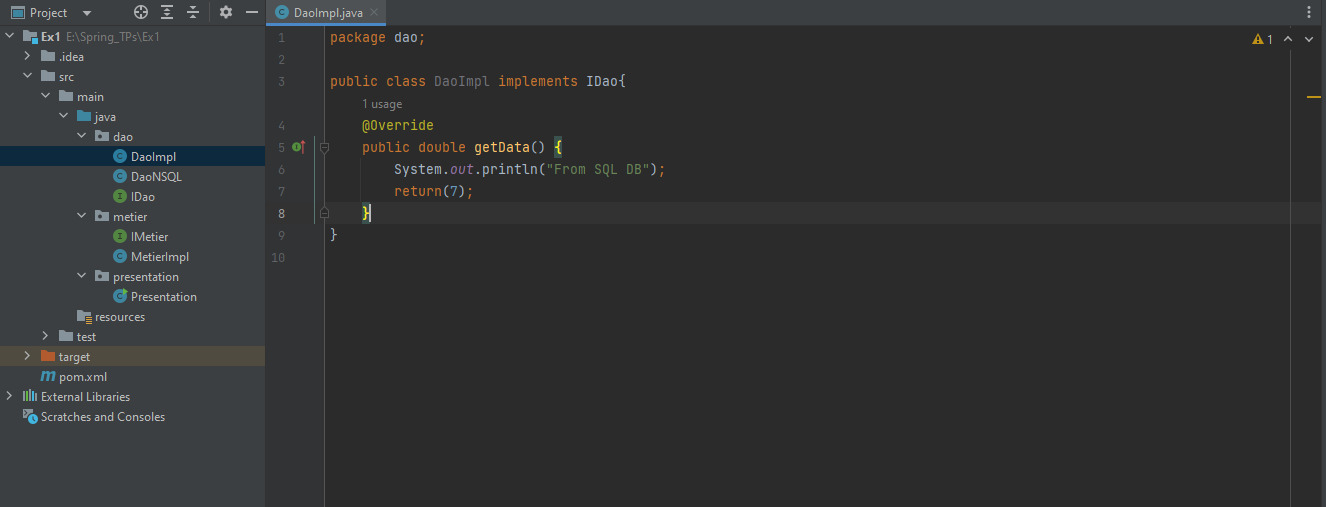
* Couche Présentation : Cette couche est responsable de la présentation de l'application à l'utilisateur final. Elle comprend généralement les interfaces utilisateur, telles que les pages Web, les formulaires et les rapports.
* Couche Métier : Cette couche est responsable de la logique de l'application. Elle comprend les classes Java qui effectuent des opérations sur les données et les transforment en réponse aux requêtes des utilisateurs.
* Couche DAO : Cette couche est responsable de la gestion des données et de l'interaction avec la base de données.

Injection des dépendances (statique) :

Création d’un prj java sur intellij avec 3 packages dao, métier et présentation

Interface IDao contient les méthodes public de la class

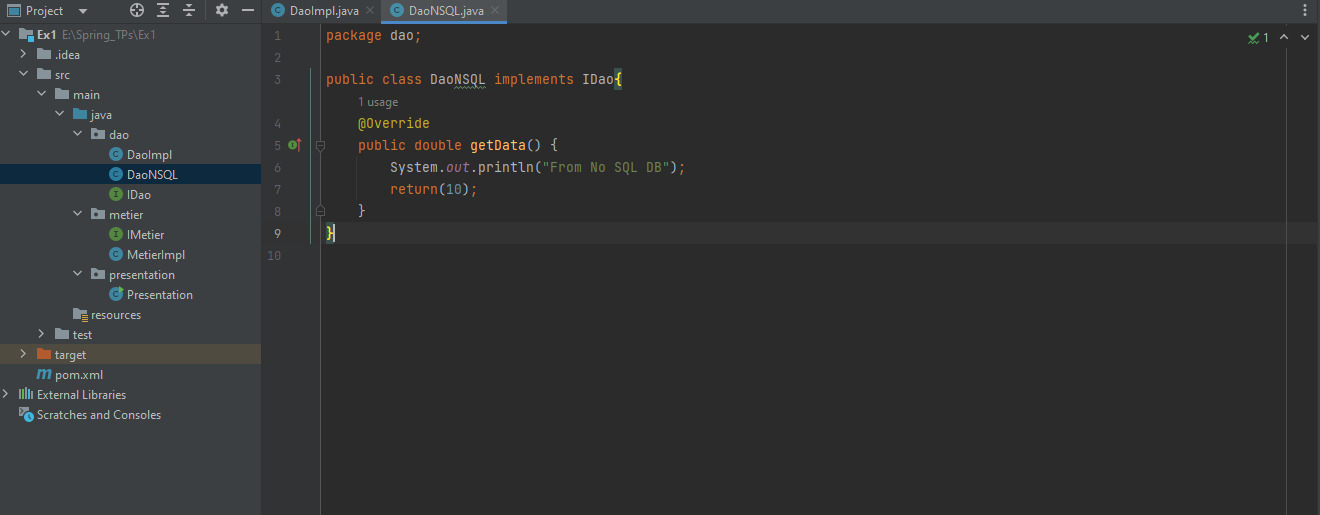
Interface IMetier

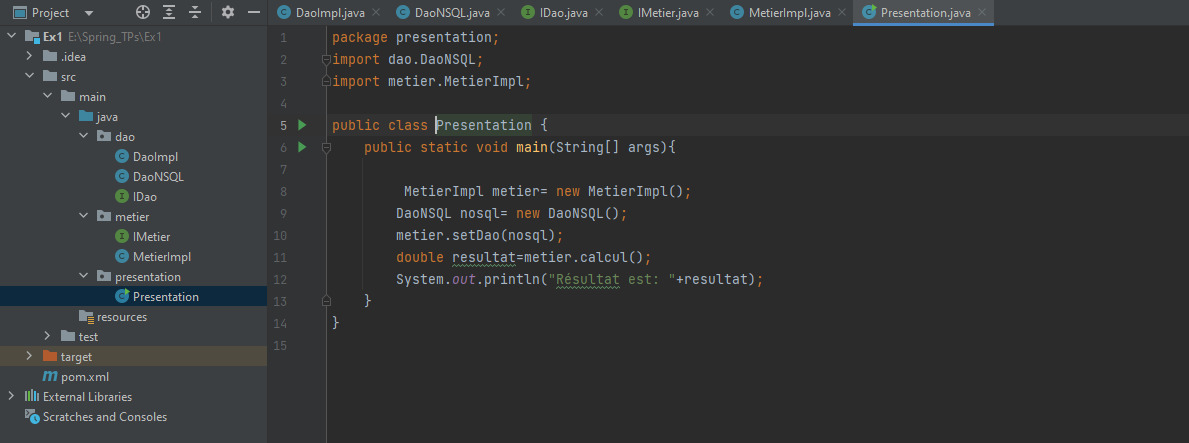
Class Daolmpm

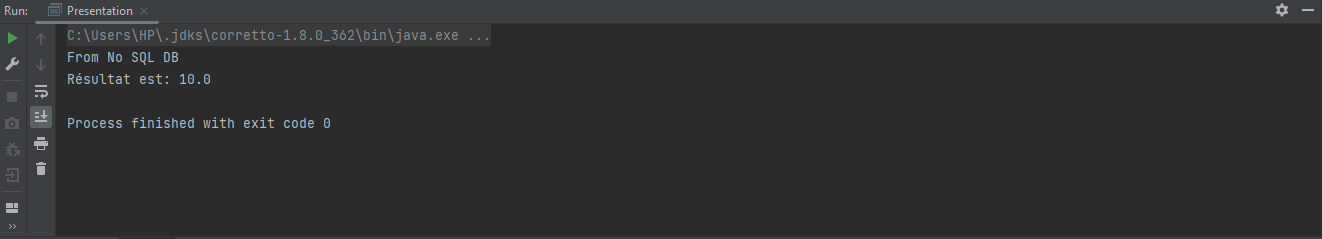
Déclaration d'un objet de type IDao implique le principe du couplage faible.

La méthode calcul permet de récupérer les données depuis la couche DAO puis faire le calcul.

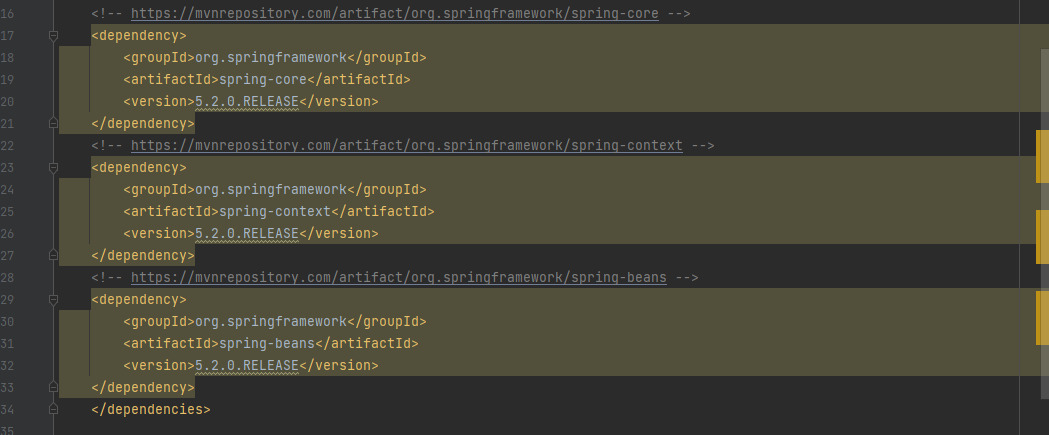
L’ajout d’une extension DaoSQL dans la couche Dao

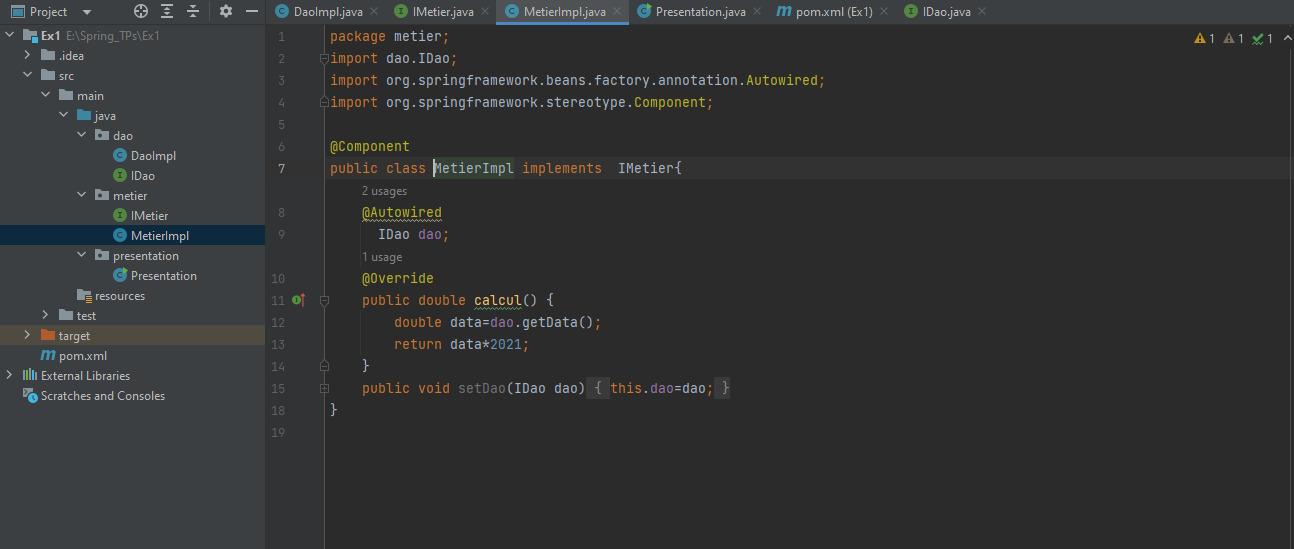


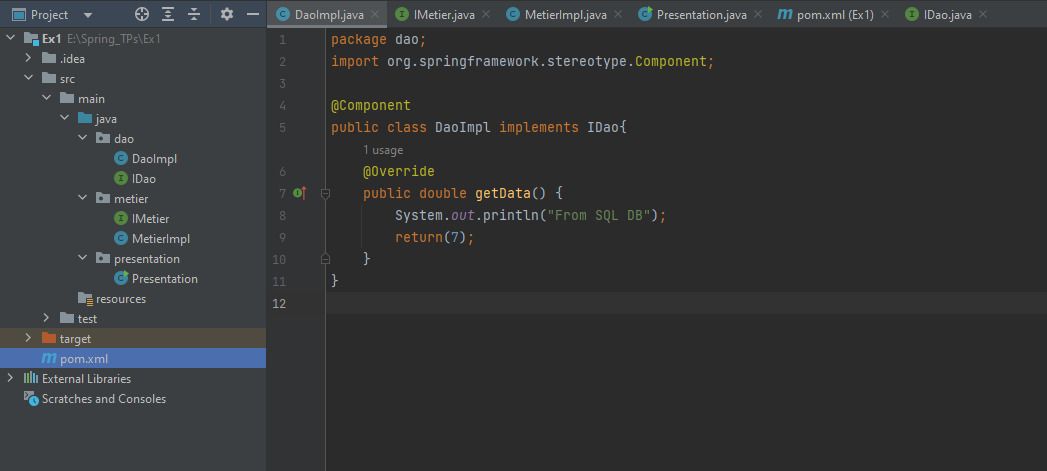
Import la couche métier pour faire le traitement 

Exécution et affichage résultat :

Injection des dépendances AVEC FRAMEWORK SPRING :

Création d’un prj maven et on ajoute les dépendances dans fichier pom.xml :

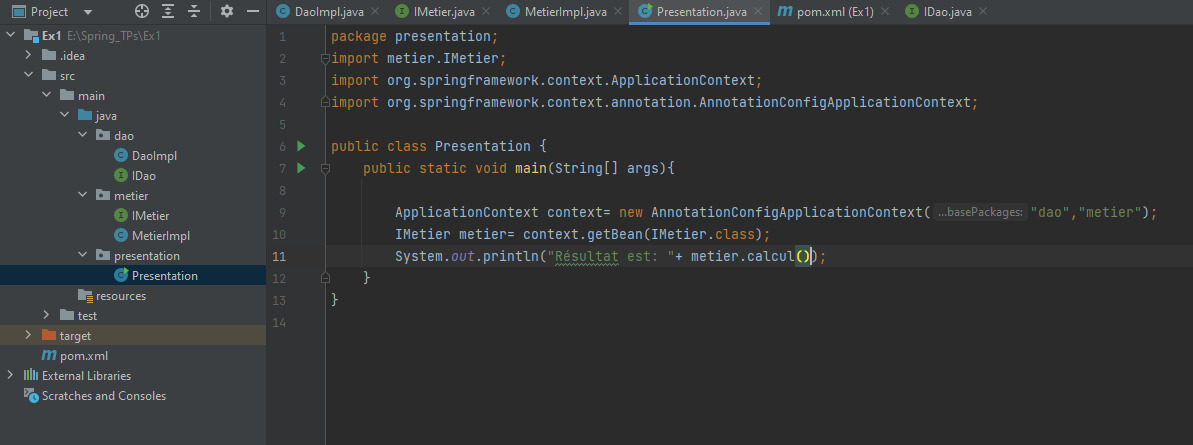
Utilisation de l'annotation @Component qui permet de dire que la classe c'est un composant de Spring et @Autowired qui permet d'activer l'injection des dépendances sur un objet d'une manière automatique.



Au niveau de la couche présentation on doit utiliser objet ApplicationContext

AnnotationConfigApplicationContext permet de scanner toutes les classes des packages dao et métier.

GetBean permet de charger un Bean qui implémente l'interface IMetier.



Exécution et affichage résultat :

