**Les dépendances Spring Boot**

* **Spring Web** : Cette dépendance est utilisée pour créer des applications web, notamment des API REST. Elle inclut des fonctionnalités comme la gestion des requêtes HTTP, les contrôleurs, etc.
* **Spring Data JPA** : Cette dépendance simplifie l'accès aux bases de données en utilisant l'ORM (Object-Relational Mapping). Elle permet de mapper les objets Java aux tables de la base de données et de gérer les opérations CRUD (Create, Read, Update, Delete) de manière simple.
* **PostgreSQL Driver** : C'est le pilote JDBC pour PostgreSQL. Il permet à Spring Boot de se connecter à une base de données PostgreSQL.
* **Lombok** : Lombok est une bibliothèque qui réduit le code répétitif en générant automatiquement des méthodes comme les getters, setters, constructeurs, etc., via des annotations.
* **Thymeleaf** : C'est un moteur de template pour les applications web. Il permet de créer des vues HTML dynamiques en intégrant des données provenant du backend (contrôleurs).
* **Spring Boot DevTools** : Cette dépendance améliore l'expérience de développement en permettant le rechargement automatique de l'application lors des modifications du code (sans redémarrer manuellement).

**Les concepts de base d'un projet Spring Boot**

**application.properties**

C'est un fichier de configuration où tu définis les propriétés de ton application, comme :

* La configuration de la base de données.
* Le port du serveur.
* Les paramètres spécifiques à Thymeleaf, JPA, etc.

**Model (ou Entity)**

Le modèle représente les objets métier de ton application. En Spring Boot, une classe modèle est souvent annotée avec **@Entity** pour indiquer qu'elle correspond à une table dans la base de données.

**Repository**

Le repository est une interface qui étend **JpaRepository** (ou d'autres interfaces Spring Data). Il fournit des méthodes pour interagir avec la base de données (CRUD).

**Service**

Le service contient la logique métier de ton application. Il utilise le repository pour interagir avec la base de données.

**Controller**

Le contrôleur gère les requêtes HTTP et renvoie les réponses appropriées. Il utilise le service pour traiter les données.

**Packages**

Dans un projet Spring Boot, les packages sont utilisés pour organiser le code. Voici une structure typique :

* com.example.demo : Package racine.
  + model : Contient les classes modèles (entités).
  + repository : Contient les interfaces de repository.
  + service : Contient les classes de service.
  + controller : Contient les contrôleurs.

**Architecture MVC**

**Architecture MVC**

MVC (Modèle-Vue-Contrôleur) est un modèle de conception qui sépare l'application en trois parties :

* **Modèle** : Représente les données et la logique métier (les entités et les services).
* **Vue** : Affiche les données (les templates Thymeleaf).
* **Contrôleur** : Gère les requêtes HTTP et fait le lien entre le modèle et la vue.