COMPTE RENDU TP1

Cette première partie du TP avait pour but de de développer une API REST simple en java avec le framework Spring Boot.
Cette API permet de :

- afficher des messages,
- retourner un étudiant,
- additionner deux nombres,
- gérer une liste d'étudiants,
- communiquer en JSON avec un client HTTP(Postman, navigateur)

On a appris à utiliser Maven, Spring Initializr, les annotations Spring et les méthodes HTTP pour créer un service web.

Tout d'abord, Maven est un outil de gestion de projet Java. Il sert à gérer les dépendances(Spring Web, JPA,..), compiler le projet, exécuter l'application, structurer le projet. Le fichier pom.xml contient toutes les dépendances nécessaires. Maven télécharge automatiquement les bibliothèques depuis internet lorsqu'on lance le projet.

Spring Initializr est un générateur de projets Spring. Il permet de créer une structure minimale contenant le fichier Maven, le point d'entrée de l'application, les dépendances choisies.

Spring Boot simplifie la création d'applications Java en fournissant un serveur intégré(Tomcat), une auto-configuration, un point d'entrée unique.

Ensuite concernant les classe :

- classe Etudiant :

Elle représente un étudiant avec les informations suivantes : id, nom et moyenne. Elle contient un constructeur par défaut, un constructeur avec paramètres et tous les getters/setters.

Le contrôleur REST : MyApi contient toutes les routes HTTP exposées par l'API. il est annoté avec @RestController qui indique que la classe sert à créer des endpoints REST et que les réponses sont automatiquement converties en JSON.
L'annotation : @GetMapping(value = "/…") sert à récupérer/afficher des données
@PostMapping est utilisé pour créer un nouvel élément avec un JSON envoyé depuis Postman.
@PutMapping représente une mise à jour d'un objet déjà existant.
@DeleteMapping sert à supprimer un objet déjà existant (depuis postman généralement)

Le fichier application.properties dans le package ressources permet de configurer l'application : server.port=9999 signifie que l'application tourne sur le port 9999 et donc les URLS sont de la forme : http://localhost:9999/...

Dans la deuxième partie du TP on crée une application Spring Boot qui utilise Spring Web, Spring Data et la H2 Database. Le but est de gérer une entité Adhérant, de la stocker automatiquement dans la base H2 et d'initialiser la base au démarrage grâce à JPA.

On a tout d’abord le package Entite qui contient le fichier Adherant qui représente un adhérent dans le système et sera transformé en table dans la base H2.
@Entity indique à JPA que cette classe doit devenir une table.
@Id représente la clé primaire.
@GeneratedValue(strategy = GenerationType.AUTO) permet à la base de données de générer automatiquement les Id.

Le package repository qui contient le fichier AdherantRepo et qui est une interface permettant de créer automatiquement toutes les opérations CRUD : save(), delete(), findById(),...

Ensuite, dans le main, on a la méthode CommandLineRunner qui est un composant Spring qui s’exécute automatiquement au démarrage de l’application. Ici, il sert à insérer les 4 adérent dans la base H2, vérifier que le JPA fonctionne et remplir la table avant toutes requêtes.
@Bean dit à Spring : “gère cette méthode et exécute la au démarrage”.

Dans le fichier application.properties on ajoute les informations nécessaires pour pouvoir accéder à la base H2 depuis un port.