TP3 - Services Web REST avec Spring Boot

Objectif:

- Créer des **web services** de type REST en utilisant le framework java **Spring Boot**
- Tester les web services REST en utilisant Postman.

Outils nécessaires :

❶ **Eclipse IDE for Enterprise Java and Web Developers**

Lien de téléchargement : Eclipse downloads - Select a mirror | The Eclipse Foundation

❷ **Spring Tools Suite pour Eclipse (**voir guide d'installation sur le classroom**)**

❸ **Postman** logiciel de test des web services

Lien de téléchargement : Download Postman | Get Started for Free

Activité 1 : Créer les web services REST avec Spring Boot

1. En se basant sur vos connaissances acquises lors du Workshop 4, créez un nouveau projet Spring Boot de type Maven ayant les dépendances suivantes :
   - ✓ Spring Web
   - ✓ Spring Data JPA
   - ✓ MySQL Driver
2. Créez les 4 couches nécessaires permettent de manipuler le concept OffreStage suivant :

```
@Entity
@Table
public class OffreStage {
	@Id
	@GeneratedValue
	long code;
	String intitule;
	String specialite;
	@ManyToOne
	@JoinColumn(name="idSociete", nullable=false)
	Societe societe;
}
```

```java
@Entity
@Table
public class Societe {
	@Id
	@GeneratedValue
	long idSociete;
	String nomCommercial;
	String activite;
	String pays;
}
```

3. Pour les deux classes entités, ajoutez :
   - ✓ Un constructeur sans paramètres
   - ✓ Un constructeur avec tous les paramètres sauf identifiant
   - ✓ Accesseur et modificateur pour chaque attribut de la classe.
4. Créez une base de donnée MySQL vide nommée SpringTpBase.
5. Modifiez le fichier **application.properties** afin de configurer, entre autres, l'accès à votre base de données SpringTpBase.

```
server.port=8787

spring.datasource.url=jdbc:mysql://localhost:3306/SpringTpBase
spring.datasource.username=root
spring.datasource.password=root
spring.jpa.hibernate.ddl-auto=create-drop
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
spring.jpa.properties.hibernate.format_sql=true
```

6. Exécutez votre projet et vérifiez que les tables Societe et OffreStage ont été créées dans votre base SpringTpBase.
7. Implémentez la liste des services indiquée dans le tableau ci-dessous :

Méthode	URL	Type de paramètre	Méthode de l'interface JpaRepository
getAll() : Retourne la liste de toutes les offres de stage	RestApi/stages	-	findAll();
getStageById(Long id) : Retourne l'offre de stage ayant l'id correspondant	RestApi/stages/{id}	Paramètre de chemin	findById(id).get();

		(@PathVariable)	
deleteStage(Long id) : Supprime une offre de stage ayant l'id correspondant et renvoie un message de notification	RestApi/stages/{id}	Paramètre de chemin (@PathVariable)	deleteById(id);
createStage(OffreStage newStage**) :** Ajoute l'offre de stage à la base de données	RestApi/stages	Paramètre envoyé sous forme d'un objet Json (@RequestBody)	save(newStage);
updateStage(OffreStage updatedStage**) :** Met à jour les données de l'offre de stage et renvoie un message de notification	RestApi/stages/{id}	Paramètre envoyé sous forme d'un objet Json (@RequestBody)	findById(id).get(); (pour récupérer l'offre de stage) save(newStage); (pour l'enregistrement de l'offre après modification)

8. Testez les web services implémentés avec Postman

Activité 2 : JpaRepository et requêtes JPQL personnalisées

9. On se propose d'implémenter les web services suivants :

Méthode	URL	Méthode/Requête JPQL au niveau de JpaRepository
List<OffreStage> getStagesByIntitule(String intitule) : Retourne la liste des offres de stage ayant un intitule donné	RestApi/stages/intitule/{intitule}	`List<OffreStage> findByIntitule(String intitule);`
List<OffreStage> getStagesBySociete(Long idScoiete) : Retourne la liste des offres de stage pour une societe donnée	RestApi/stages/societe/{id}	`@Query("select o from OffreStage o where o.societe.idSociete = ?1")` `List<OffreStage> findBySocieteId(Long idSociete);`
List<OffreStage> getStagesByPays(String pays) :	RestApi/stages/pays/{pays}	

Retourne la liste des offres de stage pour un pays donné		

void deleteStageByActivite(String activite) : Supprime les offres de stages relatives aux sociétés ayant un domaine d'activité donnée	RestApi/stages/specialite/{specialite}	`@Transactional`
		`@Modifying`
		`@Query("delete from OffreStage o where o.specialite = ?1")`
		`void deleteBySocieteSpecialite(String specialite);`

Pour cela, il faut :

- ✓ Commencer par préparer les requêtes appropriées au niveau de l'interface d'accès aux données.
- ✓ Implémenter les services au niveau de la classe service
- ✓ Implémenter les web services au niveau de la classe contrôleur