IDV-WEB4 / API REST / Etape 2 (Java)

Modalités

Type	Description
Dépôt	https://rendu-git.etna-alternance.net/module-9438/activity-51077/group-1010680
Dossier	Etape_2/
Correction	Moulinette
Durée	3 semaines
Taille de groupe	Groupe de 2

Objectifs

Notion	Description
SpringBoot	Mettre en place une entité dans un projet SpringBoot

Consignes

Avant de commencer, vous devez copier votre projet de l'étape 1 dans un dossier nommé `Etape_2`.

1) Ajout de la dépendance MySQL

Vous devez modifier votre fichier pom.xml pour ajouter les dépendances suivantes :

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<dependency>
  <groupId>com.mysql</groupId>
  <artifactId>mysql-connector-j</artifactId>
```

```
    <scope>runtime</scope>
  </dependency>
```

2) Configuration de votre datasource

Voici les paramètres pour configurer votre datasource :

- URL : `jdbc:mysql://localhost:3306/quest_web?useUnicode=true&characterEncoding=utf8&useSSL=false&serverTimezone=UTC`
- Nom d'utilisateur : `application`
- Mot de passe : `password`

Il faut donc créer une base de donnée nommée `quest_web` à laquelle l'utilisateur `application` a accès grâce à son mot de passe `password`.

3) Création de l'entité `User`

Toutes les entités que vous allez créer pour ce quest seront rangées dans le package : `com.quest.etna.model`. Une fois le package ajouté, vous devez créer une classe nommée `User`.

Pour que l'entité soit détectée par Hibernate, il est important d'utiliser les bonnes annotations. Ensuite vous devez ajouter les propriétés suivantes à la classe :

```
id : auto-généré et unique
username : chaîne de caractères (non-vide, maximum 255 caractères)
password : chaîne de caractères (non-vide, maximum 255 caractères)
role : UserRole (avec `ROLE_USER` comme valeur par défaut)
creationDate : date
updatedDate : date
```

`UserRole` est une énumération que vous devez créer, avec les valeurs `ROLE_USER` et `ROLE_ADMIN`. En base de données, vous sauvegarderez le rôle sous forme de `VARCHAR(255)`.
La date de création doit être définie au moment de la création. La date de modification doit être modifiée à chaque modification de l'entité. En base de données, vous sauvegarderez les dates sous forme de `DATETIME`.

Voici le describe de la table SQL `user` que vous devez avoir :

```
mysql> DESCRIBE user;
+---------------+--------------+------+-----+---------+-------+
| Field         | Type         | Null | Key | Default | Extra |
+---------------+--------------+------+-----+---------+-------+
| id            | int(11)      | NO   | PRI | NULL    |       |
| username      | varchar(255) | NO   | UNI | NULL    |       |
| password      | varchar(255) | NO   |     | NULL    |       |
| role          | varchar(255) | YES  |     | NULL    |       |
| creation_date | datetime     | YES  |     | NULL    |       |
| updated_date  | datetime     | YES  |     | NULL    |       |
+---------------+--------------+------+-----+---------+-------+
6 rows in set (0,00 sec)
```

Pour que votre entité soit valide, il faut toutes les annotations au bon endroit, toutes les propriétés listées, leurs getters et setters ainsi qu'un constructeur. Vous devez également implémenter les méthodes `toString`, `hashCode` et `equals`.

4) Création du repository pour `User`

Tous les *repositories* que vous allez créer pour ce quest seront rangés dans le package `com.quest.etna.repositories`. Une fois le package ajouté, vous devez créer une interface nommée `UserRepository` qui doit étendre l'interface `CrudRepository`.

Vous devez ajouter une fonction `findByUsername` qui retourne une entité `User` en se basant sur le Username.

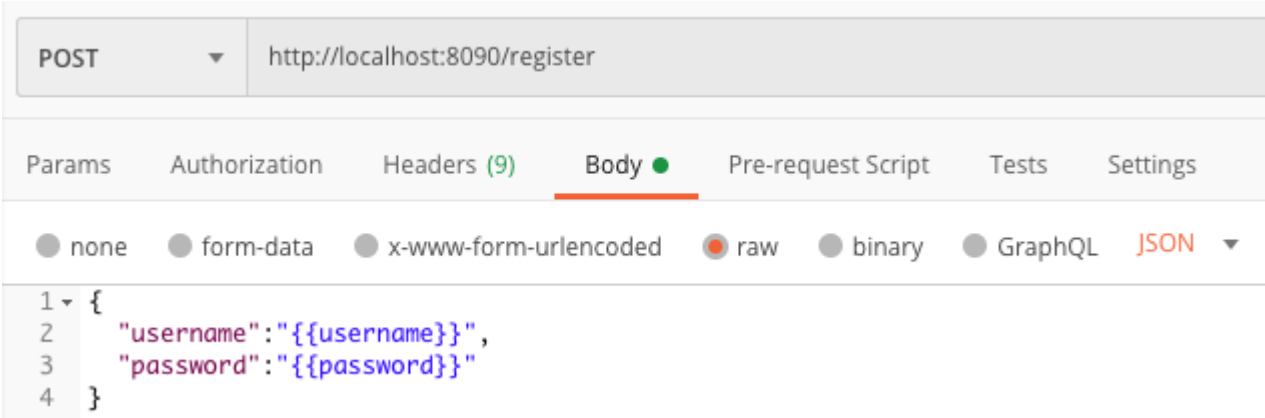
5) Création d'un modèle UserDetails

Vous devez créer un nouveau modèle `UserDetails`, qui contiendra les propriétés `username` (de type `String`) et `role` (de type `UserRole`).

6) Premier controller pour s'enregistrer

Vous devez créer un nouveau controller nommé `AuthenticationController` qui contiendra le `UserRepository` comme propriété statique.
Il faut ensuite ajouter une route POST `/register` qui permettra d'enregistrer un utilisateur en envoyant son username et son mot de passe.

En entrée, vous recevrez les paramètres `username` et `password` dans le corps de la requête (puisqu'il s'agit d'un POST) :



En sortie, voici les différents cas auxquels vous devez répondre :

Cas	Code HTTP	Réponse (toujours JSON)
Success	201	Modèle `UserDetails`
Duplicata	409	JSON avec un message d'erreur ou l'exception
Error	400	JSON avec un message d'erreur ou l'exception

Attention, il ne faut pas ajouter un utilisateur si son username se trouve déjà en base de données et il faut penser à gérer la casse.

Attention

Si vous ne configurez pas l'accès à votre base de données comme demandé, vous ne pourrez pas être corrigé.

Vous devez suivre les conventions de nommage de Java pour ce projet. Cependant, dans les tables MySQL, les propriétés seront stockés en **snake_case**.