Faire un CRUD pour les livres

1. Créer l'entité et le repository ! (Créer le model)

Grâce aux commandes, créer une entité : « Book » avec les champs suivants :

nom	type	taille	null
title	string	255	no
description	text	-	yes
genre	string	255	yes
createdAt	DateTime	-	no
updatedAt	DateTime	-	no

> **INDICE**
> Il y a 2 commandes, une pour générer l'entitél'autre pour mettre à jour la base de données ...

2. Créer le « AdminBookController » !

Grâce aux commandes, générer un controller nommé : « AdminBookController »

3. Créer la méthode « create » qui permet de créer un Livre

Ajouter une méthode `create` à ce controller, avec la route suivante :

uri	name	methods
/admin/livres/creation	app_nomDuController_nomDeLaMethod	GET, POST

Ce controller à 2 cas d'utilisation :

1. La méthode HTTP GET

Si le controller est appelé avec la méthode alors afficher un template twig (en suivant les convention de nommage). Ce template doit contenir une formulaire HTML en méthode POST :

```
<form method="POST">
  <div>
    <input type="text" name="title" placeholder="Titre du livre" />
  </div>
  <div>
    <textarea name="description" placeholder="Description du livre"></textarea>
  </div>
  <div>
    <input type="text" name="genre" placeholder="Genre du livre" />
```

```
  </div>
  <div>
    <button type="submit">Envoyer</button>
  </div>
</form>
```

2. La méthode HTTP POST

Si le controller est en méthode POST, alors il faudra créer un nouveau livre avec à l'intérieur les données du formulaire !

Ensuite, en utilisant le Repository, enregistré le livre.

Pour terminer, rediriger sur la route de l'exercice suivant

4. Créer la méthode « list »

Dans le même controller, ajouter un méthode `list` avec la route suivante :

uri	name	methods
/admin/livres	app_nomDuController_nomDeLaMethod	GET

Ce controller doit récupérer tout les livres et afficher un template twig qui liste ces derniers.

4. Créer la méthode « update »

Dans le même controller, ajouter un méthode `update` avec la route suivante :

uri	name	methods
/admin/livres/{id}	app_nomDuController_nomDeLaMethod	GET,POST

Ce controller doit tout d'abord récupérer le livre avec l'identifiant spécifier dans la route.

1. La méthode HTTP GET

Si le controller est appelé avec la méthode http GET alors afficher un template twig (en suivant les convention de nommage). Ce template doit contenir une formulaire HTML en méthode POST :

```
<form method="POST">
  <div>
    <input
      type="text"
      name="title"
      placeholder="Titre du livre"
      value="{{ book.title }}"
    />
  </div>
  <div>
    <textarea name="description" placeholder="Description du livre">
{{ book.description }}</textarea
```

```
    >
  </div>
  <div>
    <input
      type="text"
      name="genre"
      placeholder="Genre du livre"
      value="{{ book.genre }}"
    />
  </div>
  <div>
    <button type="submit">Envoyer</button>
  </div>
</form>
```

2. La méthode HTTP POST

Si le controller est en méthode POST, alors il faudra mettre à jour le livre avec à l'intérieur les données du formulaire !

Ensuite, en utilisant le Repository, enregistré le livre.

Pour terminer, rediriger sur la route de de la liste !

5. Créer la méthode « remove »

Toujours dans le même controller, ajouter une méthode `remove` avec la route suivante :

uri	name	methods
/admin/livres/{id}/supprimer	app_nomDuController_nomDeLaMethod	GET

Dans ce controller, récupérer le livre avec l'identifiant passé en paramètre de route, puis en utilisant le repository, supprimer le livre.

Un fois terminé, rediriger vers la route de la liste !