Jouen Matthias

**Documentation test technique back**

Kiss The Bride

ligne horizontale

# Introduction

J’ai décidé de faire ce projet en utilisant le Framework Symfony, couplé avec API-Platform.   
Le tout sera sur une image docker. Je vais expliquer ma réflexion ainsi que les points importants. Il y aura également un petit tuto d’installation pour faire fonctionner le projet sur n’importe quel poste.

## Docker

La première étape a été de créer des conteneurs Docker avec MySQL, PhpMyAdmin et Apache/PHP. Une fois le tout configuré, en accédant au terminal de l’environnement PHP, j’ai pu créer un “skeleton project” Symfony. Pour ce qui est des versions on est ici sur PHP 8.2.9 et Symfony 6.3. Pour Apache c’est du 2.4.

## Symfony

Après avoir obtenu le projet Symfony vierge, j’ai configuré le .env pour relier mon projet à la base de données.

DATABASE\_URL="mysql://root:@db\_docker\_symfony:3306/db\_test\_frais"

Ici on définit donc qu’on est sur du MySQL, le nom d’utilisateur est root, le mot de passe est vide, le nom de la base est “db\_test\_frais”. Bien évidemment en conditions réelles on ne laissera pas des noms d’utilisateurs et mots de passe comme ceci et on pourrait utiliser des variables d’environnement serveur pour les récupérer.

Afin de tester cette liaison et le bon fonctionnement du projet, j’ai utilisé le maker Bundle pour créer ma première entité. J’ai commencé par l’entité User.

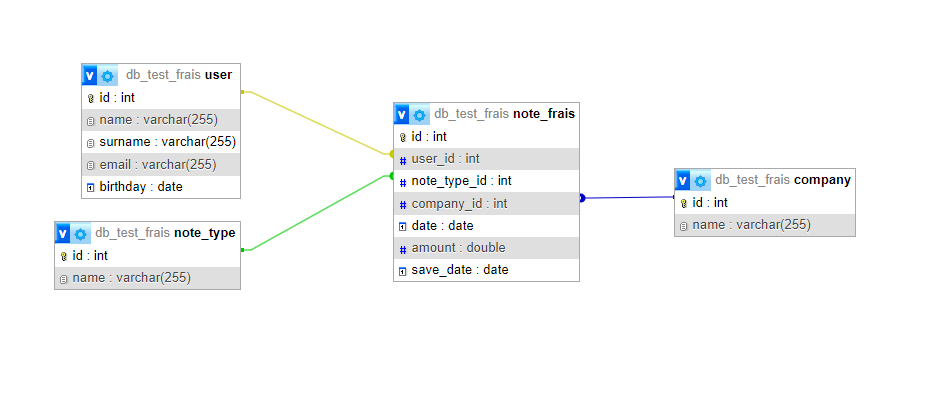
Après la création de base de données : php bin/console doctrine:database:create

La création de la migration : php bin/console make:migration

Et la migration en elle-même : php bin/console doctrine:migrations:migrate

On se retrouve bien avec notre table dans la base de données sur PhpMyAdmin.

Etant donné le bon fonctionnement, je me suis attaqué aux autres entités et voici le résultat des tables que j’ai conçu :



Pour résumer, voici ce qu’on a textuellement. Des notes de frais sont associées à un utilisateur. Cet utilisateur à un nom, un prénom, une date de naissance et un email. Ces notes de frais sont également d’un certain type et sont liées à une entreprise. Elles ont une date de saisie, une date à laquelle elle a eu lieu et une somme.

## Api Platform

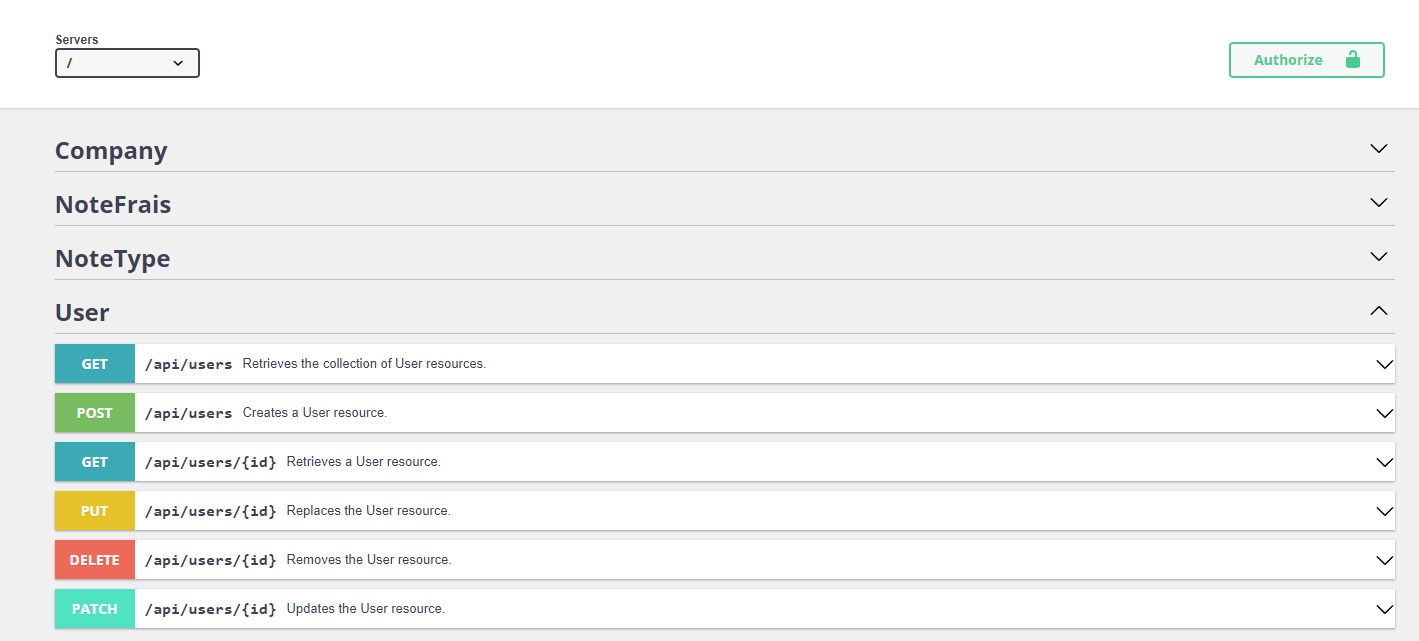
Dans mon projet Symfony, j’ai intégré API-Platform, qui me fera gagner énormément de temps.

composer require api

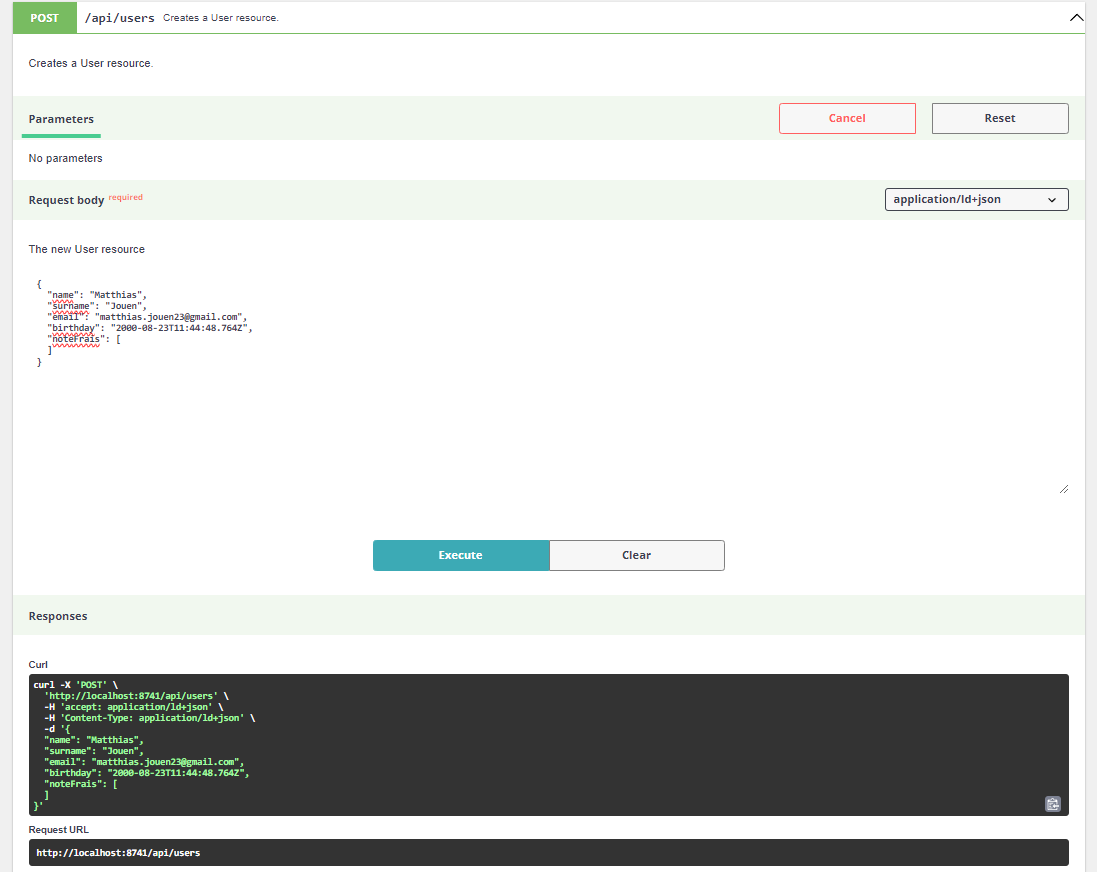
Le gros avantage d’api platform c’est sa rapidité d’installation et son efficacité. Dans le cas de ce projet, elle peut couvrir 100% des requêtes dont on a besoin.

La seule chose à faire est de définir les entités comme ressources de l’api :

#[ApiResource()]



Cela permet ensuite de tester les requêtes directement depuis l’interface avant de passer à Postman. Voici ci-dessous un exemple avec une requête POST pour un User.

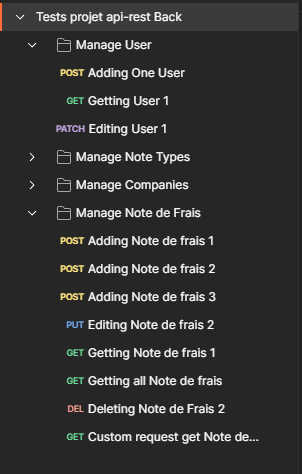


On peut ensuite constater le résultat sur notre base de données :



## Postman

Dans postman j'ai créé une collection de requêtes pour ajouter, modifier, lire et supprimer des données. Cette collection permettra de certifier du bon fonctionnement de l’application, grâce aux codes de retour des requêtes (200 ou 201 par exemple).



Pour finir j'ai créé une route custom qui n'est pas de base prévu avec api-platform et qui permet de récupérer la liste des notes de frais d'un utilisateur en particulier.

http://localhost:8741/api/user/1/note\_frais

Cette requête est également testée dans la collection Postman.

### 

### Installation

Afin de tester le projet, il faut tout d’abord s’assurer de ces prérequis :

* Posséder docker
* Posséder Postman

Il faut tout d’abord accéder au projet Git qui se trouve sous ce lien et le télecharger : <https://github.com/MatthiasJouen/SymfonyNoteDeFrais>

Une fois que le dossier est prêt sur votre poste. Vous pouvez lancer cette commande depuis le dossier:

docker-compose up -d

Puis celle-ci pour exécuter des commandes à l’intérieur du conteneur directement.

docker exec -it www\_docker\_symfony bash

Pour finir enchaîner ces trois commandes pour faire successivement l’installation des packages, la création de la base de données et la migration de la structure de la BDD.

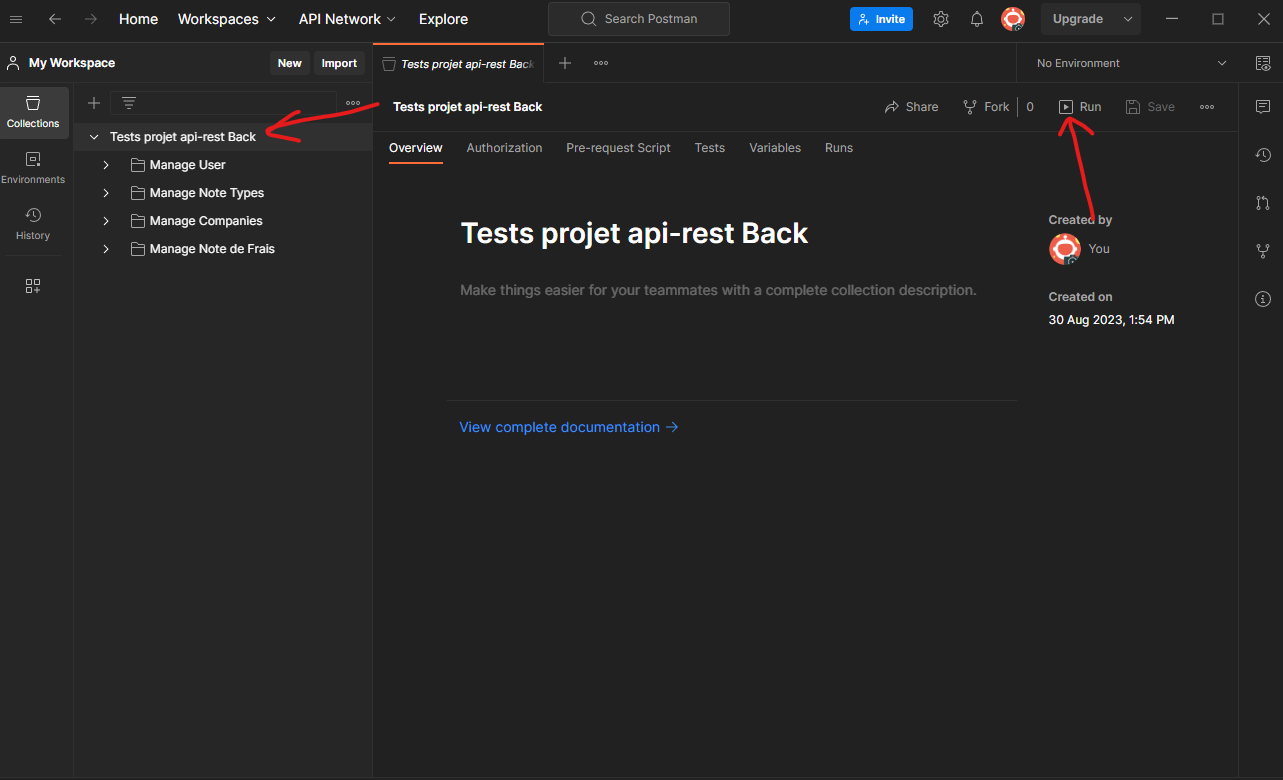
composer install

cd symfony-back-api-rest

php bin/console doctrine:database:create

php bin/console doctrine:migrations:migrate --no-interaction

Ensuite aller sur Postman et importer la collection qui se trouve dans Docs/



Valider l'exécution et constater le fonctionnement des requêtes.