Louvels.dev Symfony assessment

In this assignment you will be showing off your skills with Symfony.

The assignment below should take around 2 to 4 hours to complete. Please feel free to modify any prewritten code or install libraries if you feel like this improves the final result.

1. Requirements

- Docker (or Docker Desktop)

2. Getting started

- Start docker
- Run: docker-compose up --build -d
- Visit the application at http://localhost:8084 to see if it works.

3. Exercise

We're going to build a service that stores country data. It allows it's users to do CRUD operations on the data and gives us (administrators) a way to reset the data to the original state. To get you started, we've already created some classes and namespaces within this project. However, feel free to create or delete classes and namespaces to make the application match your needs.

3.1 Building a model

Let's start off with an entity. Please create the required model using Doctrine, generate a diff and run the migrations. We've already created a class for you at `App\Entity\Country`.

```
{
  "uuid":"string",
  "name":"string",
  "region":"string",
  "subRegion":"string",
  "demonym":"string",
  "population":0,
  "independant":true,
  "flag":"string",
  "currency":{
    "name":"string",
    "symbol":"string"
  }
}
```

```
php bin/console doctrine:migrations:diff
```

```
php bin/console doctrine:migrations:migrate
```

If you want to, you can validate the database scheme through a MySQL connection in a separate client, or by using:

```
php bin/console doctrine:query:sql "QUERY"
```

3.2 Fetching the data

Now that we have a database up and running, it is time to write some actual logic. We're going to fetch our country data from REST Countries. Please write a service that fetches the required data from their API and syncs it into our database.

3.3 Keeping it all in sync

Now we need to figure out a way to actually run our code and fetch the data. Luckily Symfony has us covered here. Please implement `App\Command\CountrySyncCommand`, to give us a way to fetch the data.

If you want to, you can again validate the synchronized data through a MySQL connection in a separate client or by using:

```
php bin/console doctrine:query:sql "QUERY"
```

3.4 CRUD

We have a database filled with country data, great! However we have no way to interact with that data at this point in time. Please implement the controller at `App\V1\Controller\CountryController` and make sure that users can use our ApiDoc to try out our API.

3.5 Security

An API that is not secured like ours will most likely receive a lot of nonsensical POST, PATCH and DELETE calls. Secure these 3 calls with basic auth and make sure that our command at `App\Command\CountrySyncCommand` removes countries that do not exist according to REST Countries and resets modified countries to their original data (again based on REST Countries). Just so we're able to reset our data when someone figures out our basic auth password.

Hint: You do not have a lot of time, so spend it wisely and use a Memory User Provider. Also, make sure that the GET call is still available to the general public.