Fialoux mateo

Compte rendu symfony API REST

1- Mise en place des données

1.1 Sur la base des connaissances acquises dans les TPs précédents, créer un projet de test web Symfony nommé garage. Détailler les commandes exécutées.

Cherche le chemin de symfony pour avoir accès à la commande symfony export PATH="$HOME/.symfony5/bin:$PATH"

Crée le projet : symfony new mon_projet --webapp

1.2 Définition des entités

Définir l'entité Doctrine correspondante via la console de Symfony. Détailler la commande utilisée.

php bin/console make:entity Owner

New property name (press <return> to stop adding fields): name

Field type (enter ? to see all types) [string]: string

Field length [255]: 255

Can this field be null in the database (nullable) (yes/no) [no]: no

1.2.2 Entité Car (voiture)

Définir l'entité Doctrine correspondante via la console de Symfony. Pourquoi est-il préférable de créer l'entité Owner avant Car?

Crée l'entité avec Doctrine php bin/console make:entity Car

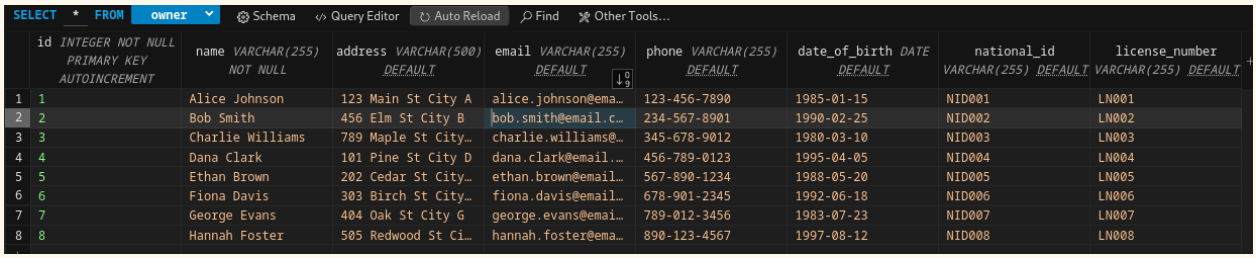
L'entité Car contient une relation ManyToOne vers Owner.
Doctrine doit donc connaître la classe Owner pour générer la propriété, les annotations/attributs ORM et les relations inverses (OneToMany).

Si tu essaies de créer Car avant Owner, la commande make:entity ne pourra pas proposer la relation.

1.2.3 Alimentation des entités avec une fixture

Quel fichier a été créé? Après exécution, vérifier que les tables Car et Owner contiennent des données

src/DataFixtures/GarageFixture.php



SELECT * FROM owner — Schema — Query Editor — Auto Reload — Find — Other Tools...

	id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT	name VARCHAR(255) NOT NULL	address VARCHAR(500) DEFAULT	email VARCHAR(255) DEFAULT	phone VARCHAR(255) DEFAULT	date_of_birth DATE DEFAULT	national_id VARCHAR(255) DEFAULT	license_number VARCHAR(255) DEFAULT
1	1	Alice Johnson	123 Main St City A	alice.johnson@ema…	123-456-7890	1985-01-15	NID001	LN001
2	2	Bob Smith	456 Elm St City B	bob.smith@email.c…	234-567-8901	1990-02-25	NID002	LN002
3	3	Charlie Williams	789 Maple St City…	charlie.williams@…	345-678-9012	1980-03-10	NID003	LN003
4	4	Dana Clark	101 Pine St City D	dana.clark@email.…	456-789-0123	1995-04-05	NID004	LN004
5	5	Ethan Brown	202 Cedar St City…	ethan.brown@email…	567-890-1234	1988-05-20	NID005	LN005
6	6	Fiona Davis	303 Birch St City…	fiona.davis@email…	678-901-2345	1992-06-18	NID006	LN006
7	7	George Evans	404 Oak St City G	george.evans@emai…	789-012-3456	1983-07-23	NID007	LN007
8	8	Hannah Foster	505 Redwood St Ci…	hannah.foster@ema…	890-123-4567	1997-08-12	NID008	LN008

1.2.4 Analyse du chargement de la fixture

Quel est l'intérêt fonctionnel de la variable $ownerRecords?

$ownerRecords est un tableau qui sert à stocker en mémoire les objets Owner créés lors de la lecture du fichier Owners.csv.

Les ids stockés dans le fichier CSV sont-ils exploités pour alimenter la base et pourquoi?

Les IDs du CSV servent uniquement à faire le lien entre les données dans le code (relier voitures et propriétaires) mais ne sont pas utilisés comme identifiants en base, car Doctrine génère ses propres ID auto-incrémentés.

Ouvrir la base de données. Que se passe-t-il au niveau des IDs des tables Car et Owner si l'on relance le chargement de la fixture?

À chaque relance, la base est vidée et les auto-incréments réinitialisés, donc les IDs en base repartent à 1 et peuvent changer, mais les relations restent cohérentes grâce au traitement en mémoire pendant l'import.

2- Déploiement d'une API REST avec API Platform

2.1- Maintenant, démarrer le serveur et accéder aux urls suivantes:

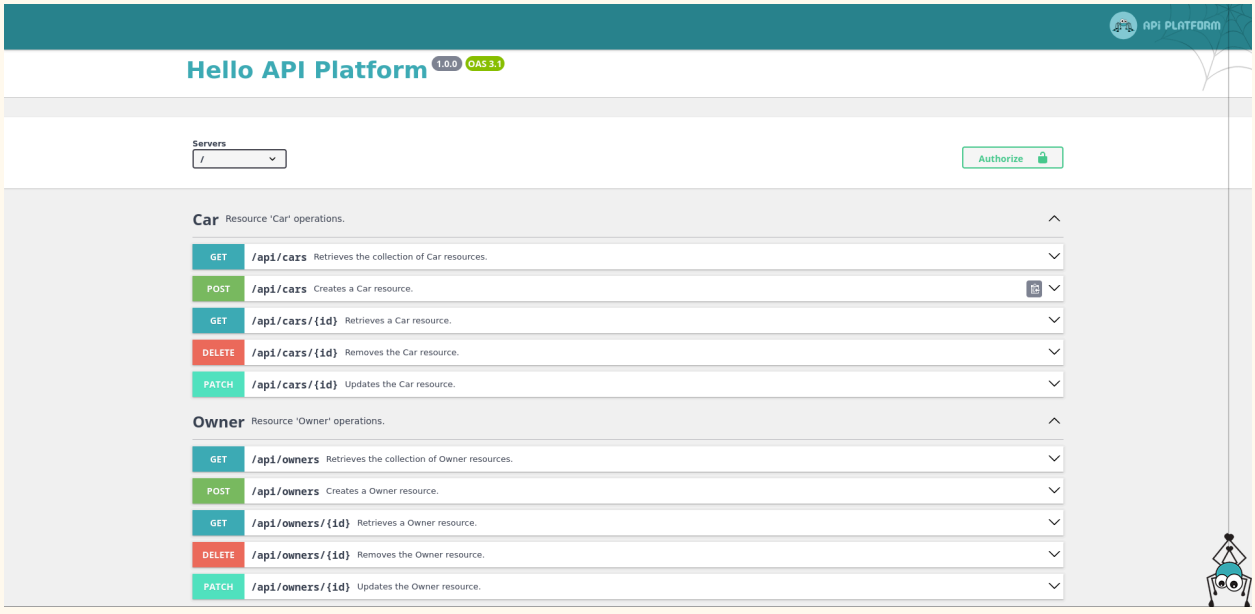
- http://localhost:8000/api/cars
- http://localhost:8000/api/owners

Que découvre-t-on?

On retrouve les informations de la base de données sous format json

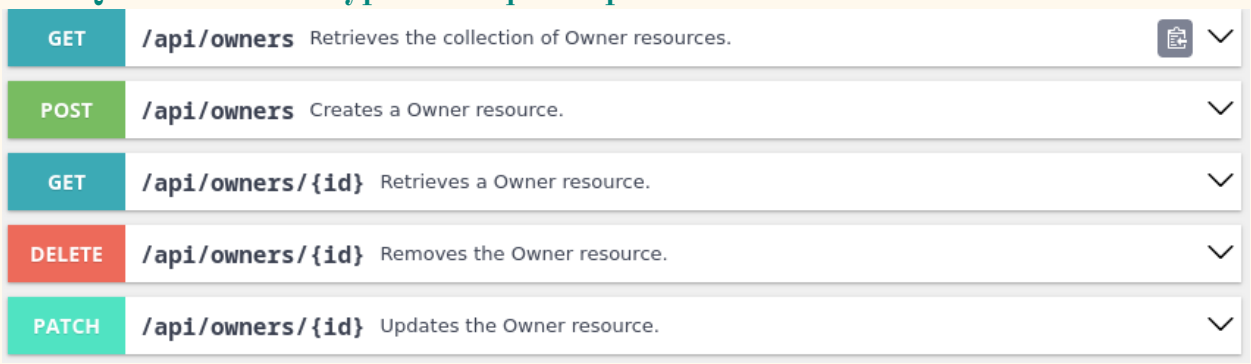
2.2- Accéder à l'url: http://localhost:8000/api

Que découvre-t-on?



La plateforme API

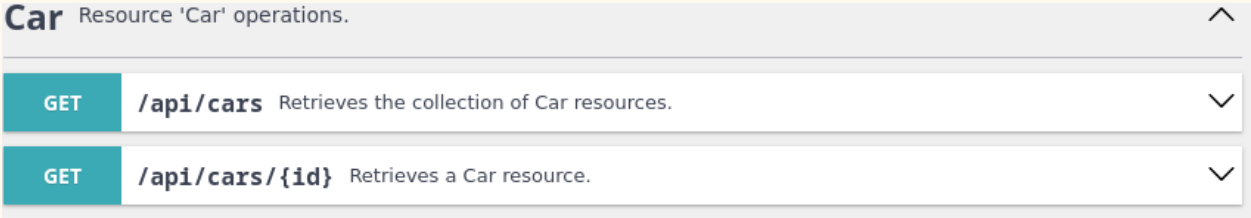
2.3 - Quels sont les 5 types de requêtes possibles via l'API REST?



2.4 - Utiliser l'API pour supprimer la voiture Toyota de la base. Quelle requête/url doit-on exécuter/charger?

2.5 - Modifier les annotations comme ci-après.

Recharger la page d'accès à l'api. Que se passe-t-il?



Il ne nous reste plus que les requêtes get .

2.6 - Modifier le fichier de configuration d'API Platform pour y inclure les formats JSON et XML comme ci-après. Que se passe-t-il?

On a en plus de la page JSON avec les informations une page avec les informations en XML.

3- Aller plus loin

🚗 Liste des voitures

Marque	Modèle	Année	Prix
Honda	Civic	2019	18 000 €
BMW	X5	2021	50 000 €
Ford	Focus	2018	15 000 €
Tesla	Model 3	2022	45 000 €
Audi	A4	2019	28 000 €
Nissan	Altima	2021	23 000 €
Kia	Optima	2020	19 000 €
Mazda	CX-5	2021	25 000 €
Hyundai	Elantra	2019	17 000 €
Chevrolet	Malibu	2018	16 000 €
Jeep	Cherokee	2022	32 000 €
Subaru	Impreza	2020	21 000 €
Dodge	Charger	2021	29 000 €
Volkswagen	Jetta	2019	19 000 €
Mercedes	C-Class	2022	55 000 €
Lexus	ES 350	2018	33 000 €
Acura	TLX	2021	36 000 €
Infiniti	Q50	2019	31 000 €

```php
    #[Route(path: '/', name: 'app_voiture')]
    4 references
    public function index(): Response
    {
        $url = 'http://localhost:8000/api/cars.json';

        try {
            $response = $this->client->request(method: 'GET', url: $url);
            $voitures = $response->toArray(); // Décodage automatique JSON => tableau PHP
        } catch (\Exception $e) {
            $voitures = [];
        }

        return $this->render(view: 'voiture/index.html.twig', parameters: [
            'voitures' => $voitures,
        ]);
    }
}
```