1. Présentation du projet

Symfony Stubborn est une petite application e-commerce développée avec le framework **Symfony**. Elle propose :

- Une page d'accueil (produits, inscription, connexion…)
- Un panier (gestion des articles, suppression, validation)
- Un processus de paiement (Stripe)
- Un espace d'administration (CRUD sur les produits)

L'objectif est d'offrir une base fonctionnelle pour apprendre Symfony et créer un projet e-commerce simple.

---

2. Installation

2.1. Récupérer le projet

Cloner le dépôt Git :

```
git clone https://github.com/FloDevs/Symfony_stubborn.git
cd Symfony_stubborn
```

Installer les dépendances :

```
composer install
npm install
```

Créer le fichier `.env` ou `.env.local`. Par exemple :

```
APP_ENV=dev
APP_SECRET= # Mettre une valeur secrète ici, par ex. générée par
https://randomkeygen.com/
DATABASE_URL= l'url de votre base de données SQL avec le nom que vous
voulez lui attribuer à la fin exemple :
"mysql://root:@127.0.0.1:3306/mon_nom_de_bdd"
STRIPE_PUBLIC_KEY=pk_test_xxx
STRIPE_SECRET_KEY=sk_test_xxx
```

Créer la base de données et exécuter les migrations :

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

J’ai créé un fichier pour créé un utilisateur admin et donc avoir accès à toutes les fonctionnalité il suffit juste de mettre :

```
php bin/console app:create-admin admin@example.com admin123
```

Lancer l’application en mode développement :

```
symfony serve
```

(Ou, à défaut, `php -S 127.0.0.1:8000 -t public`)

Accéder à l’application via :
http://127.0.0.1:8000

---

3. Configuration

3.1. Variables d’environnement

- **APP_ENV=dev**
  Indique que l’application tourne en environnement de développement.

- **APP_SECRET=**
  Clé secrète utilisée par Symfony pour les jetons CSRF et d’autres mécanismes de sécurité.

- **DATABASE_URL=**
  URL de connexion à ta base de données, par ex.
  `mysql://username:password@127.0.0.1:3306/nom_de_bdd`

- **STRIPE_PUBLIC_KEY=**
  Clé publique Stripe pour le paiement.

- **STRIPE_SECRET_KEY=**
  Clé secrète Stripe pour valider les paiements côté serveur.

Assure-toi de mettre tes vraies clés Stripe, sinon les paiements ne fonctionneront pas.

---

4. Routes disponibles

Nom	Méthodes	URI	Description
_preview_error	ANY	`/_error/{code}.{_format}`	Page d'erreur interne (gérée par Symfony)
admin_product_index	GET | POST	`/admin/`	Page d'administration des produits
cart_show	ANY	`/cart`	Afficher le contenu du panier
cart_remove	ANY	`/cart/remove/{productId}/{size}`	Supprimer un produit (et sa taille) du panier
cart_checkout	ANY	`/cart/checkout`	Passer la commande (paiement)
cart_success	ANY	`/cart/success`	Confirmation de commande après paiement
cart_clear	ANY	`/cart/clear`	Vider complètement le panier
app_home	ANY	`/`	Page d'accueil
app_products	GET	`/products/`	Voir la liste des produits
app_product_show	GET	`/products/{id}`	Détails d'un produit

cart_add	POST	`/products/{id}/add-to-cart`	Ajouter un produit au panier
app_register	ANY	`/register`	Page d'inscription utilisateur
app_login	ANY	`/login`	Page de connexion
app_logout	ANY	`/logout`	Déconnexion

---

5. Utilisation de l'application

1. **Page d'accueil** : accessible à l'URL `/`.
    - Présentation sommaire du site.
2. **Liste des produits** : via `/products/`.
    - Parcourir les produits disponibles.
3. **Détails d'un produit** : via `/products/{id}`.
    - Ajouter au panier avec la route POST `/products/{id}/add-to-cart`.
4. **Panier** : via `/cart`.
    - Affiche les articles sélectionnés, leur quantité et leur taille (si géré).
    - Possibilité de supprimer un article : `/cart/remove/{productId}/{size}`.
    - Possibilité de vider le panier : `/cart/clear`.
5. **Passer la commande** :
    - La route `/cart/checkout` redirige vers un système de paiement (Stripe).
    - Une fois le paiement validé, l'utilisateur est redirigé vers `/cart/success`.
6. **Administration** : via `/admin/`
    - Gestion des produits (CRUD).
    - Nécessite un compte administrateur (selon la configuration de sécurité).
7. **Inscription / Connexion** :
    - `/register` pour créer un nouveau compte.
    - `/login` pour se connecter.
    - `/logout` pour se déconnecter.

---

6. Paiement avec Stripe

- Les **clés d'API** Stripe sont à configurer dans ton `.env` ou `.env.local`.
- La route `/cart/checkout` initie le paiement via la clé secrète Stripe.
- Après paiement validé, l'utilisateur est redirigé vers `/cart/success`.

Pour tester, tu peux utiliser les **clés de test** Stripe :

- Clé publique : `pk_test_...`
- Clé secrète : `sk_test_...`
- Carte de test : `4242 4242 4242 4242` (avec une date d'expiration future et n'importe quel CVC)