Documentation technique – Application e-commerce Stubborn

1. Présentation générale

Nom du projet :

- Stubborn – Boutique en ligne de sweat-shirts

Technologies principales :

- Symfony 6, PHP 8.2, MySQL 8, Stripe (paiement), GitHub (versioning), PHPUnit (tests)

Client :

- Nom : Stubborn
- Adresse : Piccadilly Circus, London W1J 0DA, Royaume-Uni
- Contact : stubborn@blabla.com
- Slogan : Don't compromise on your look

Objectif :

Développer une boutique en ligne permettant aux clients d'acheter des sweat-shirts avec un parcours utilisateur complet : navigation, inscription, authentification, panier, paiement en ligne, et gestion back-office pour les administrateurs.

2. Architecture technique

Environnement :

- Framework : Symfony 7.3
- Base de données : MySQL MariaDB 10.4.28 (Distribuer avec XAMPP)
- Gestionnaire de dépendances : Composer
- Contrôle de version : GitHub
- Serveur local : Symfony CLI / Apache avec PHP 8.2
- Paiement : Stripe (mode sandbox)
- Tests unitaires : PHPUnit

Organisation du code :

- **src/Entity** : entités (User, Product, Cart, Order, etc.)
- **src/Repository** : accès aux données
- **src/Controller** : logique métier et routage
- **src/Form** : gestion des formulaires (login, register, ajout produit, etc.)
- **templates/** : vues Twig
- **public/** : assets (images, CSS, JS)
- **tests/** : tests unitaires

3. Base de données

Entités principales :

- **User** : id, name, email, password (hashé), roles, delivery_address
- **Product** : id, name, price, sizes (XS-XL), stock, featured, image
- **Cart** : id, user_id, product_id, size, quantity
- **Order** : id, user_id, total_amount, status, created_at
- **Payment** : id, order_id, stripe_session_id, status

Relations :

- Un utilisateur possède plusieurs commandes (**OneToMany**).

- Un produit peut apparaître dans plusieurs paniers (**ManyToMany**).

- Une commande contient plusieurs produits (**OneToMany via un OrderItem**).

4. Fonctionnalités principales

Accueil (/):
- Présentation de la société Stubborn.
- Affichage des 3 produits mis en avant (Blackbelt, Pokeball, BornInUsa).
- Menu dynamique selon connexion.

Authentification :
- /login : formulaire de connexion (Symfony Security).
- /register : formulaire d'inscription + email de confirmation.

- CONNEXION utilisateur test:
testuser@example.com
password123

Produits :
- /products : affichage + filtres prix.
- /product/{id} : fiche produit avec ajout au panier.

Panier (/cart) :
- Ajout, suppression, calcul total.
- Validation via Stripe (mode test).

Paiement Stripe :
- Intégration via SDK Stripe (sandbox).
- Simulation avec cartes de test.

Back-office (/admin) :
- Ajout, modification, suppression produits.
- Gestion du stock et mise en avant.

- CONNEXION admin test:
test@example.com
password123

5. Tests unitaires

Réalisés avec PHPUnit :
- Ajout produit au panier.
- Suppression produit du panier.
- Calcul du total commande.
- Simulation d'un paiement Stripe.

6. Déploiement local

Installation :
1. **git clone** https://github.com/katoudevb/stubborn_shop
2. **composer install** && **npm install** && **npm run build**
3. Configurer **.env** (DB, MAILER, STRIPE keys)
4. **php bin/console doctrine:database:create**
php bin/console doctrine:migrations:migrate
php bin/console doctrine:fixtures:load
5. **symfony serve –d**

Exécution des tests :
php bin/phpunit

7. Conclusion

Le projet e-commerce Stubborn offre un parcours complet :

- Inscription et authentification sécurisées
- Consultation des produits avec filtres
- Gestion du panier et calcul dynamique
- Paiement simulé avec Stripe
- Back-office pour la gestion du catalogue

L'application respecte les wireframes et permet de démontrer un achat-test en local.