Documentation du projet e-commerce Stubborn

1. Introduction

Nom du projet : Boutique en ligne Stubborn
Technologies utilisées : Symfony, PHP, MySQL, Stripe
Objectifs : Développer une application e-commerce permettant : - La gestion des utilisateurs et des rôles (client et administrateur). - La gestion des produits (CRUD). - La gestion d'un panier avec simulation de paiement via Stripe.

Fonctionnalités principales : - Authentification et inscription des utilisateurs. - Consultation des produits avec filtrage par prix. - Gestion d'un panier d'achat dynamique. - Interface administrateur pour la gestion des produits. - Paiement sécurisé en mode développement avec Stripe.

---

2. Fonctionnalités

2.1 Utilisateur Client

- S'inscrire et recevoir un email de confirmation.
- Se connecter et accéder à la boutique.
- Ajouter des produits au panier avec sélection de la taille.
- Valider une commande et simuler un paiement.

2.2 Utilisateur Administrateur

- Accéder au back-office sécurisé.
- Ajouter, modifier ou supprimer des produits.
- Gérer les stocks et les mises en avant des produits.

---

3. Architecture du projet

3.1 Entités principales

3.1.1 `User`

- **Attributs :**
  - `id`: Identifiant unique.
  - `name`: Nom de l'utilisateur.
  - `email`: Adresse email de l'utilisateur.
  - `password`: Mot de passe haché.
  - `delivery_address`: Adresse de livraison.
- **Relations :**
  - `OneToMany` avec `Cart`.

3.1.2 `Product`

- **Attributs :**
  - `id`: Identifiant unique.
  - `name`: Nom du produit.
  - `price`: Prix du produit.
  - `highlighted`: Produit mis en avant (booléen).
- **Relations :**
  - `OneToMany` avec `ProductSize`.

3.1.3 `ProductSize`

- **Attributs :**
  - `id`: Identifiant unique.
  - `size`: Taille du produit (XS, S, M, L, XL).
  - `stock`: Quantité disponible.
- **Relations :**
  - `ManyToOne` avec `Product`.

3.1.4 `Size`

- **Attributs :**
  - `id`: Identifiant unique.
  - `name`: Nom de la taille (XS, S, M, L, XL).
- **Relations :**
  - `OneToMany` avec `ProductSize`.

3.1.5 `Cart`

- **Attributs :**
  - `id`: Identifiant unique.
  - `total_price`: Prix total de la commande.
- **Relations :**
  - `ManyToOne` avec `User`.
  - `ManyToMany` avec `ProductSize`.

---

4. Contrôleurs

4.1 `CartController`

- Gère les opérations sur le panier.
- **Actions principales :**
  - Ajouter un produit au panier.
  - Supprimer un produit du panier.
  - Valider la commande.

4.2 `HomeController`

- Gère les pages publiques (accueil, connexion, inscription).

4.3 `ProductController`

- Gère les produits.
- **Actions principales :**
  - Afficher tous les produits.
  - Filtrer les produits par prix.
  - Afficher les détails d'un produit.

4.4 `RegistrationController`

- Gère les inscriptions des utilisateurs.
- **Actions principales :**
  - Afficher le formulaire d'inscription.
  - Envoyer un email de confirmation.

4.5 `SecurityController`

- Gère l'authentification et la sécurité des utilisateurs.
- **Actions principales :**
  - Afficher le formulaire de connexion.
  - Déconnecter un utilisateur.

4.6 `StripeController`

- Gère les paiements via Stripe.
- **Actions principales :**
  - Initialiser le paiement.
  - Traiter les réponses de Stripe.

---

5. Instructions d'installation

1. **Cloner le dépôt GitHub :**

   ```
   git clone https://github.com/Emmatremlet/stubborn-ecommerce.git
   ```

2. **Installer les dépendances :**

   ```
   composer install
   npm install
   ```

3. **Configurer la base de données :**

   - Modifier le fichier `.env` pour indiquer les paramètres de connexion MySQL.

```
DATABASE_URL="mysql://user:password@127.0.0.1:3306/stubborn_db"
```

- Exécuter les migrations :

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

4. **Configurer le service de messagerie :** Ajoutez la configuration suivante dans le fichier `.env` :

```
MAILER_DSN=smtp://user:password@smtp.example.com:587
```

5. **Alimenter la base de données :**

```
php bin/console doctrine:fixtures:load
```

6. **Lancer le serveur Symfony :**

```
symfony server:start
```

---

6. Tests

6.1 Lancer les tests

1. Exécuter les tests PHPUnit :

```
php bin/phpunit
```

6.2 Tests disponibles

- Tests pour l'ajout d'un produit au panier.
- Tests pour la validation d'une commande.

---

7. Gestion des paiements avec Stripe

1. **Créer un compte Stripe.**
2. **Configurer les clés API dans le fichier `.env` :**

```
STRIPE_SECRET_KEY=sk_test_...
STRIPE_PUBLIC_KEY=pk_test_...
```

3. **Simuler un paiement avec le mode bac à sable.**
   - 4242 4242 4242 4242 : Carte valide pour un paiement réussi.
   - 4000 0000 0000 9995 : Carte invalide pour tester les erreurs de paiement.

---

8. Hébergement local

- Assurez-vous que l'application est accessible à l'adresse suivante :

  `http://127.0.0.1:8000`

- Testez le processus d'achat avec les fonctionnalités Stripe.

---

9. Glossaire

- **Symfony :** Framework PHP utilisé pour développer des applications web.
- **CRUD :** Opérations Create, Read, Update, Delete.
- **Stripe :** Solution de paiement en ligne intégrée au projet.

---

10. Annexes

- Diagramme UML des entités disponible dans le dossier : `diagramme_UML/Ecommerce_UML.png`.
- Documentation format PDF : `documentation.pdf`.