Documentation Technique - Stubborn Shop

---

1. Présentation du projet

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

Le projet est un site e-commerce spécialisé dans la vente de sweatshirts tendance.
Développé en PHP avec le framework Symfony 7.2, il repose sur une architecture MVC robuste.

---

2. Environnement technique

- Langage : PHP >= 8.2
- Framework : Symfony 7.2
- Base de données : MariaDB (my_shop_dev)
- ORM : Doctrine
- Moteur de templates : Twig
- Système de mail : Mailtrap (via Symfony Mailer)
- Gestion des paiements : Stripe
- Autres composants : Validator, Security, Notifier, Asset, etc.

---

3. Modèle de données

Entité Product

- id
- nom
- description
- price
- image
- stock total
- stock par taille (XS, S, M, L, XL)
- ManyToMany avec SubCategory

Entité SubCategory

- id
- nom
- ManyToOne vers Category
- ManyToMany avec Product

Entité Category

- id
- name (unique)
- OneToMany vers SubCategory

Entité Order

- id
- prénom
- nom
- téléphone
- adresse
- ville

---

4. Page d’accueil

- **Route** : / (nommée app_home)
- Affiche une sélection de produits populaires (via leurs ID)
- Les produits sont récupérés grâce à ProductRepository
- Vue Twig utilisée : home/index.html.twig

---

5. Fonctionnalité : Panier

Le panier est géré via **Session PHP**.

Routes principales :

- **GET** /cart : voir le panier
- **GET** /cart/add/{id} : ajouter un produit
- **GET** /cart/remove/{id} : retirer un produit spécifique
- **GET** /cart/remove : vider le panier

Les données sont affichées dans cart/index.html.twig.

---

6. Sécurité et Authentification

Le système d'authentification est basé sur SecurityAuthenticator :

- Connexion via email et mot de passe
- Utilisation de Passport, PasswordCredentials, UserBadge
- RememberMeBadge activé pour rester connecté
- Redirection vers app_home après connexion
- Protection CSRF possible (CsrfTokenBadge, commenté dans l'exemple)

Route de connexion : app_login

---

7. Back-office : Gestion des catégories

CRUD complet sur les catégories depuis /admin/category :

- /admin/category (GET) : Liste des catégories
- /admin/category/new (GET|POST) : Ajouter une nouvelle catégorie
- /admin/category/{id}/update (GET|POST) : Modifier une catégorie existante
- /admin/category/{id}/delete (GET) : Supprimer une catégorie

Formulaire utilisé : CategoryFormType.

---

8. Back-office : Gestion des sous-catégories

Gestion CRUD des sous-catégories :

- /sub/category (GET) : Liste des sous-catégories
- /sub/category/new (GET|POST) : Créer une sous-catégorie
- /sub/category/{id} (GET) : Afficher une sous-catégorie
- /sub/category/{id}/edit (GET|POST) : Modifier une sous-catégorie
- /sub/category/{id} (POST) : Supprimer une sous-catégorie

Formulaire utilisé : SubCategoryType.

---

9. Paiement Stripe

L'intégration du paiement Stripe est faite **en full backend**.

Fonctionnement :

- Lors de la validation du formulaire de commande, une **session Stripe Checkout** est créée.
- Les produits du panier sont transformés en **line_items** compatibles Stripe.

- Redirection automatique vers Stripe pour le paiement.
- Retour vers /order/success en cas de paiement réussi.
- Retour vers /order/cancel en cas d'annulation.

Détails techniques :

- Service utilisé : App\Service\StripeService
- Configuration des clés Stripe :
    - Clé secrète injectée via %env(STRIPE_SECRET_KEY)%
    - Configuration dans services.yaml
- Fichiers Twig associés : order/success.html.twig, order/cancel.html.twig