PROJET SYMFONY

FOODTRUCK – Diawara Alpha Malick

ESGI

SYMFONY

**Sommaire**

1. **Introduction**.……………………………………………………..
2. **Fonctionnalités du Projet…………………………………..**
3. **Techniques………………………………………………………**
4. **Base de Données……………………………………………..**
5. **Ressources……………………………………………………..**
6. **Test………………………………………………………………..**
7. **Annexes………………………………………………………….**
8. **Introduction**

Dans le cadre de mon projet Symfony, j’ai décidé de réaliser une application de foodtruck nommée **« La Burgererie »**. Cette application permet aux utilisateurs de réserver de la nourriture et aux administrateurs de gérer l’ensemble des activités sur le site. L’objectif principal est de faciliter l’écoulement des stocks de nourriture pour les foodtrucks.

1. **Fonctionnalités du projet**

Pour ce projet , il fallait que je réalise une application Symfony avec une interface plutôt épurée qui répondent aux contraintes techniques et à mon cahier de charge.

User :

* L’utilisateur doit pouvoir s’inscrire , se connecter , réinitialiser son mot de passe
* L’utilisateur doit pouvoir contacter les administrateurs du site via un formulaire de contact
* L’utilisateur doit pouvoir consulter un menu de produit
* L’utilisateur doit pouvoir être capable d’ajouter des produits qui l’intéressent à son panier puis passer commande
* Une fois la commande passé l’utilisateur doit être notifié par mail avec les informations de sa commande (petit reçu)
* L’utilisateur doit pouvoir laisser un avis sur un produit qu’il a déjà commander , avis qui pourras être ensuite consulter par les autres utilisateurs

Admin :

* L’administrateur doit pouvoir accéder à un panneau d’administration ou il pourra gérer le site/
* Une fois dans le panneau d’administration , l’administrateur doit pouvoir être habilité à modérer les comptes utilisateurs , gérer les catégories de produits , modérer les produits et les menus associés
* L’administrateur doit pouvoir marquer les commandes comme terminé / les supprimer
* L’administrateur doit pouvoir voir les demandes de contact et y répondre / supprimer la demande
* L’administrateur doit pouvoir modérer les avis laisser sur les produits

1. **Technique**

Les technologies utilisées pour ce projet sont :

* **Symfony** : Framework PHP permettant de développer des applications web
* **MySQL** : Gestion de bases de données relationnelles utilisé pour stocker et organiser les données
* **Twig** : Un moteur de templating pour PHP, utilisé dans Symfony pour structurer les vues
* **Tailwind** : Framework CSS utilitaire qui facilite la conception d'interfaces modernes et responsives
* **Git** : système de gestion de versions permettant de suivre les modifications du code

Pour développer cette application, j’ai utilisé **Visual Studio Code** comme environnement de développement.

J’ai mis en place un cadre de travail structuré en configurant les fichiers essentiels comme **.env** (gestion des variables d’environnement), **.gitignore** (exclusion des fichiers sensibles du suivi Git) et le gestionnaire de paquets **Composer** pour la gestion des dépendances.

**Sécurité et Gestion des Accès**

La sécurité étant une priorité, plusieurs contraintes ont été mises en place :

* Un **système d’authentification sécurisé** garantissant la protection des comptes utilisateurs.
* Un **gestionnaire de permissions** permettant de définir des accès selon le rôle de chaque utilisateur :
  + **ROLE\_USER** → Utilisateur classique : peut passer des commandes et contacter les administrateurs.
  + **ROLE\_ADMIN** → Administrateur : dispose d’un accès au panneau d’administration pour gérer le site et les commandes.
  + **ROLE\_BANNED** → Utilisateur banni : n’a aucun accès au site.

Grâce à cette architecture , l’application garantit une expérience sécurisée pour tous ses utilisateurs.

1. **Base de données**

La base de données est conçue pour gérer un système de commande en ligne pour un foodtruck. Elle structure et organise les informations essentielles concernant les utilisateurs, les produits, les commandes, les menus et d'autres entités du projet.

**Gestion des utilisateurs et adresses**

* **user** : Stocke les informations des utilisateurs, notamment leur email, mot de passe, nom, prénom, numéro de téléphone et rôles. Chaque utilisateur peut avoir un rôle (ROLE\_USER, ROLE\_ADMIN, ROLE\_BANNED).
* **address** : Permet d'enregistrer les adresses associées aux utilisateurs. Chaque adresse est liée à un utilisateur via la clé étrangère user\_id.

**Gestion des catégories et des produits**

* **category** : Définit les différentes catégories d’aliments proposées, comme les burgers, boissons, desserts, etc.
* **product** : Contient les informations des produits alimentaires, comme leur titre, description, prix et image. Chaque produit appartient à une catégorie grâce à la clé étrangère category\_id.

**Gestion des menus**

* **menu** : Représente les différents menus proposés par le foodtruck. Chaque menu appartient à une catégorie (category\_id).
* **menu\_product** : Gère la relation entre les menus et les produits. Un menu peut contenir plusieurs produits et un produit peut appartenir à plusieurs menus.

**Gestion des commandes**

* **order** : Stocke les commandes passées par les utilisateurs. Une commande est associée à un utilisateur (user\_id) et une adresse de livraison (address\_id). Elle inclut des informations comme le numéro de commande, le prix total, le mode de paiement, le statut, et les dates de création et de mise à jour.
* **order\_product** : Permet de gérer les produits contenus dans une commande. Elle enregistre la quantité et le prix de chaque produit commandé.
* **shipping** : Définit le jour de retrait associé à une commande (order\_id).

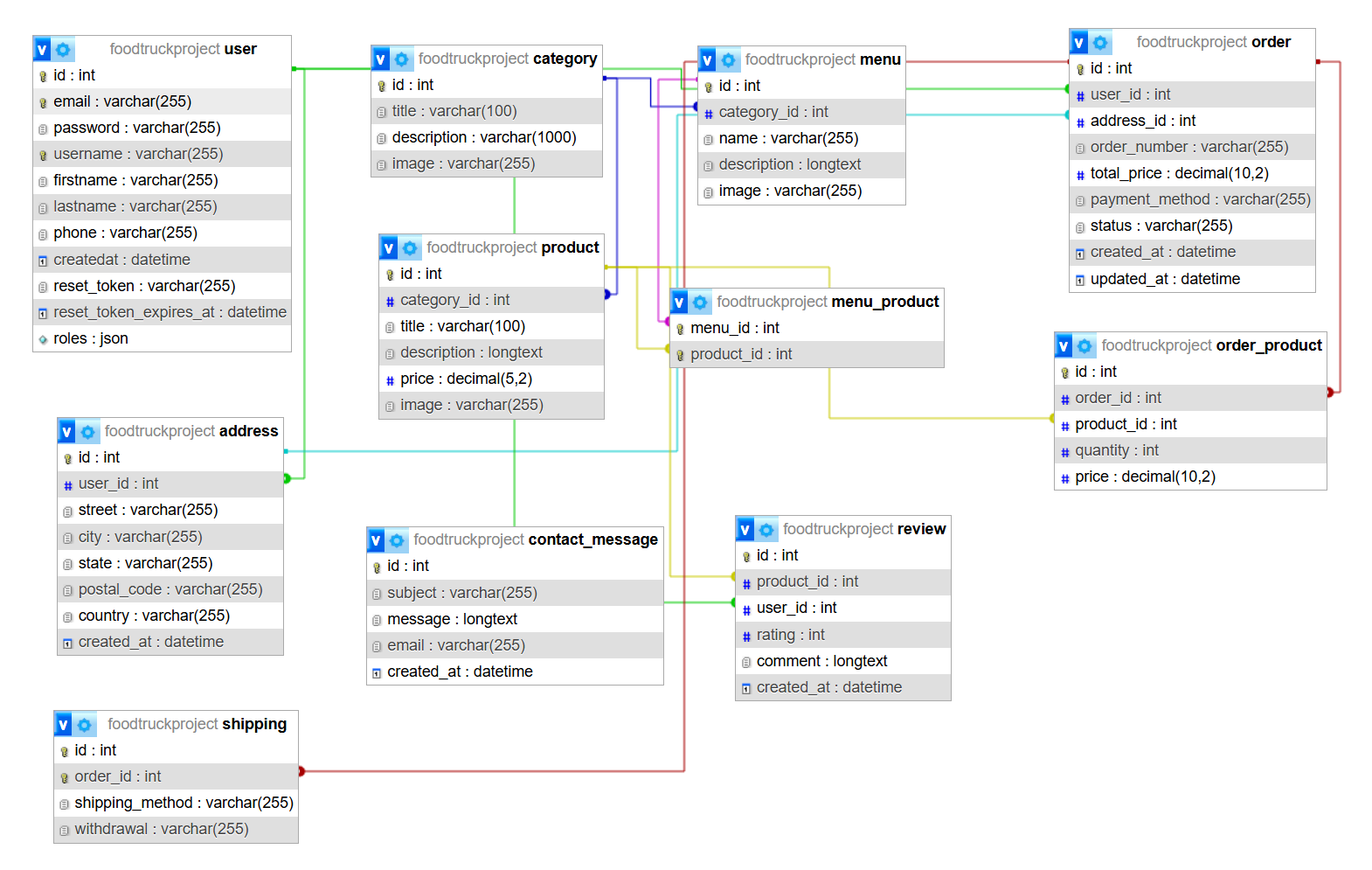
**Gestion des avis et des messages de contact**

* **review** : Permet aux utilisateurs de laisser des avis et des notes sur les produits (product\_id). Chaque avis est lié à un utilisateur (user\_id).
* **contact\_message** : Stocke les messages envoyés via le formulaire de contact, incluant l’email de l’expéditeur, le sujet et le message.

**Sécurisation et intégrité des données**

* Chaque table est reliée par des **clés étrangères** pour assurer la cohérence des données.
* **Index** utilisés pour optimiser les recherches.
* Les id sont des clés primaires auto-incrémenté.

Cette structure permet d'assurer une gestion optimisée du foodtruck en ligne.



1. **Ressources**

J’ai développé seul le projet **Foodtruck**, en m’appuyant sur un ensemble d’outils pour optimiser le processus de développement.

J’ai principalement utilisé **Visual Studio Code** pour l’édition du code, **GitHub Desktop** pour la gestion des versions, ainsi que **GitHub Copilot** et **Stack Overflow** pour accélérer le développement et garantir la fiabilité de l’application.

1. **Tests**

Pour les tests, je me suis basée sur les **User Stories**.

J’ai créé un tableau Excel détaillant chaque User Story, avec les résultats attendus et les résultats obtenus. Cette méthode m’a permis de tester de manière exhaustive l’application, en vérifiant chaque fonctionnalité et en m’assurant qu’aucun bug n’était présent.



1. **Annexe**

**Documentation**

* [Symfony Documentation](https://symfony.com/doc/current/index.html)
* [Twig Documentation](https://twig.symfony.com/doc/3.x/index.html)
* [Tailwind CSS Documentation](https://tailwindcss.com/docs)
* [MySQL Documentation](https://dev.mysql.com/doc/)
* [Git Documentation](https://git-scm.com/doc)