# Cahier des Charges – ShopConnect

## 1. Présentation du Projet

Nom du Projet : ShopConnect

Responsable du projet : Vincent Vanhees

Langage principal : PHP (Framework Laravel ou Symfony)

Base de données : MySQL

Front-end : bootstrap

Type de projet : Développement d'une plateforme de gestion pour les petits commerçants locaux.

Objectif : ShopConnect est une plateforme de gestion permettant aux commerçants locaux de créer et gérer leur boutique en ligne. Les clients pourront y naviguer, acheter des produits, et les commerçants pourront gérer leurs produits et leurs commandes.

## 2. Exigences Fonctionnelles

### 2.1 Gestion des Utilisateurs

Inscription : Un utilisateur peut s’inscrire avec son email, Facebook, ou Google.

Authentification : OAuth via Google/Facebook, ou inscription classique avec email/mot de passe.

Récupération de mot de passe : Possibilité de réinitialiser le mot de passe par email.

### 2.2 Gestion des Commerçants

Création de compte commerçant : Un commerçant peut créer un compte et configurer sa boutique.

Gestion des produits : CRUD (Create, Read, Update, Delete) pour les produits (titre, description, prix, image, stock).

Gestion des commandes : Visualiser et traiter les commandes (nouvelle commande, commande en préparation, expédiée, terminée).

### 2.3 Interface Client

Parcourir les boutiques : Les clients peuvent visualiser les boutiques locales.

Recherche de produits : Les clients peuvent chercher des produits par mots-clés.

Panier d'achat : Gestion d'un panier avec calcul des frais de livraison et des taxes.

Paiements : Paiement sécurisé via carte de crédit ou PayPal (API Stripe).

### 2.4 Interface Administrateur

Gestion des utilisateurs : CRUD sur les utilisateurs (ajouter, modifier, supprimer des comptes).

Gestion des commerçants : Suivi et assistance pour les commerçants.

Gestion des commandes : Suivi global des commandes et des paiements sur la plateforme.

Promotions : Le responsable marketing peut encoder des promotions (produit en solde, codes de réduction).

## 3. Exigences Techniques

### 3.1 Architecture Technique

Backend : PHP avec un framework (Certainement Symfony) pour la gestion des API et des processus métiers.

Base de données : MySQL pour le stockage des utilisateurs, produits, commandes, et informations commerciales.

Frontend : Bootstrap pour une interface utilisateur moderne et responsive.

### 3.2 Sécurité

Authentification sécurisée : Utilisation d’OAuth pour Facebook et Google. Gestion des tokens avec JWT (JSON Web Tokens).

Sécurité des API : Toutes les requêtes sont protégées via des tokens JWT. Les données sont chiffrées (HTTPS).

Prévention des attaques : Protection contre les attaques CSRF (Cross-Site Request Forgery) et XSS (Cross-Site Scripting).

### 3.3 Multilingue

Support de plusieurs langues : Français et anglais. Possibilité de traduire dynamiquement les interfaces via Angular i18n.

### 3.4 Paiements

API Stripe/PayPal : Intégration pour gérer les paiements en ligne via carte bancaire ou compte PayPal. Les commerçants peuvent suivre leurs ventes et paiements.

### 3.5 Performance

Cache : Utilisation de cache pour optimiser les temps de chargement des pages les plus consultées (produits, boutiques).

Pagination et lazy loading : Implémentation de la pagination pour les listes longues (produits, commandes) et du lazy loading pour les images.

## 4. Design et Interface Utilisateur

### 4.1 Responsive Design

L'interface doit s’adapter à tous types de supports (ordinateur, tablette, mobile) avec un design moderne et ergonomique.

### 4.2 UX/UI

Utilisation de bonnes pratiques d’UX/UI pour assurer une expérience utilisateur fluide. Les pages de paiement doivent être claires et sécurisées.