CAHIER DES CHARGES TECHNIQUE

SAÉ 5

Alexandre ROBAIL

UNIVERSITÉ DE LORRAINE | IUT Saint-Dié-des-Vosges

SOMMAIRE

1. ORGANISATION

Méthodologie : approche Agile (Scrum)

- Travail en sprints de 2 semaines avec livrables intermédiaires.
- Réunions rapides quotidiennes pour suivre l’avancement.
- Gestion des tâches via GitLab (issues et tableaux de suivi).
- Revues de code croisées avant chaque fusion.

Répartition :

Frontend / UX : intégration Blade + Tailwind.

Backend : logique métier Laravel (tournées, adhérents).

Base de données : conception des modèles et migrations.

Tests / intégration : automatisation et validation fonctionnelle.

2. BESOINS CLIENT

Inscription et suivi des adhérents

Gestion des profils, abonnements et authentification sécurisée.

Tunnel de commande fluide

Sélection d’un abonnement, d’un dépôt et validation simplifiée.

Planification des livraisons

Création de tournées selon la fréquence et les dépôts.

Gestion du stock et des paniers

Répartition des produits selon le type de panier.

Interface claire et adaptative

Navigation simple sur ordinateur, tablette et mobile.

3. ARCHITECTURE

Utilisateur → Interface → Contrôleur → Modèle → Base de données

Structure logicielle

- Modèle MVC (Laravel) : séparation claire entre logique, données et affichage.
- Frontend : Blade + Tailwind pour les vues.
- Backend : Laravel pour les routes, contrôleurs et validation.
- Base de données : SQLite (en dev), exportable vers MySQL en production.
- Environnement : Docker pour uniformiser les configurations.

4. TECHNOLOGIES

Laravel (PHP)

Gestion des routes, logique serveur, sécurité intégrée.

Tailwind CSS

Mise en page rapide sans fichiers CSS lourds.

Docker

Conteneurisation complète pour cohérence dev/prod.

Bootstrap

Eléments visuels préconstruits pour gagner du temps.

JavaScript

Base légère pour développement local.

SQLite

Base légère pour développement local.

5 - OUTILS DE DÉVELOPPEMENT

Documentation

fichiers Markdown + README détaillé.

GitLab

(issues, merge requests, branches)

VS Code

(extensions Laravel, Tailwind, Git).

6. Environnement de test

- **Serveur de test :** Docker local.

- **Base de test :** SQLite séparée.

- **Tests unitaires :** PHPUnit intégré à Laravel.

- **Données fictives :** seeders et factories.

- **Vérifications manuelles :** Postman + navigateur.

- **Tests UX :** retours utilisateurs sur interface responsive.

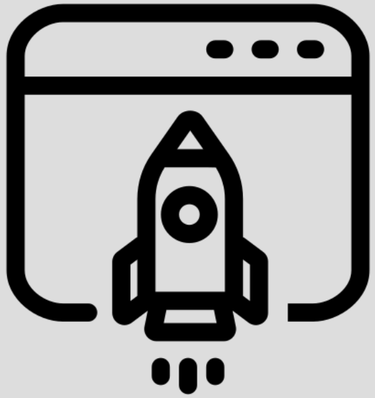
7. HÉBERGEMENT / DÉPLOIEMENT

Infrastructure :

- *Serveur web* : Apache (compatible Laravel).

- *Conteneurs Docker* : garantissent le même environnement en dev et en production.

- *Base de données* : SQLite stockée localement, sauvegardée automatiquement.



8. CARTE, ITINÉRAIRE ET CALENDRIER

Carte et itinéraires

- API : Leaflet + OpenStreetMap (libre et personnalisable).
- Affichage des dépôts et tournées sur carte interactive.
- Trajets calculés via Leaflet Routing Machine.

Calendrier

- Outil : FullCalendar.js.
- Fonctions : affichage des dates de livraison, détails d'un jour, synchronisation avec la base.
- Mise à jour dynamique après chaque commande ou livraison.

9. SÉCURITÉ

- **Authentification par rôles** : administrateur / adhérent.

- **Protection CSRF** et hachage des mots de passe (bcrypt).

- **Validation des formulaires** côté client et serveur.

- **Sauvegarde automatique** de la base (cron Docker).

- HTTPS et .env **sécurisé** pour la production.