

Cahier des charges techniques

Logiciel de gestion commerciale
des *Jardins de Cocagne*



DE RYCKE Leanne - COLIN Gabin

15/10/2025

Jardins de Cocagne

- Présentation de l'équipe

DE RYCKE Leanne & COLIN Gabin

Développeurs chez
Software Architectures Évolutions



Sommaire

1. *Objectifs techniques*
2. *Architecture logicielle*
3. *Technologies et langages*
4. *Modules et fonctionnalités techniques*
5. *Contraintes techniques*
6. *Environnement de collaboration*
7. *Solution d'hébergement*
8. *Solution pour les tests*



1. Objectifs techniques

Développer un logiciel web centralisé de gestion commerciale pour les Jardins de Cocagne.

- **Back office** pour gérer : clients, adhérents, produits, abonnements, points de dépôt et livraisons.
- **Front office** pour les clients : gestion autonome de l'abonnement, consultation et paiement en ligne.
- **Modularité** et **paramétrage** élevé pour s'adapter aux spécificités de chaque Jardin.
- Logiciel **Open Source**, évolutif, sécurisé et simple d'utilisation.

2. Architecture logicielle

Le système reposera sur une architecture web modulaire, hébergée de façon centralisée.

- **Back-end** : API REST en PHP, gérant la logique, les données et la communication entre les différentes parties du logiciel.
- **Front-end** : interface web responsive et intuitive pour les utilisateurs.
- **Base de données** : base commune partagée, avec des données séparées pour chaque Jardin.
- **Conteneurisation** (*Docker*) : déploiement simplifié, évolutif et maintenable.
- **Système de paiement** : intégration de Stripe pour la gestion sécurisée des abonnements et règlements en ligne.
- **Hébergement** (*Google Cloud*) : Solution cloud centralisée, sécurisée et évolutive avec gestion automatique des ressources
- **Tests** (*Postman*) : validation des routes API et vérification du bon fonctionnement des échanges entre le front-end et le back-end.



3. Technologies et langages ⁽¹⁾

Couches	Langage	Rôle
Back-end	PHP	Développement du web service REST
Front-end	JavaScript	Interface utilisateur et interactions
API (RESTful, intégrée au back-end PHP)	JSON	Communication entre front et back
Base de données	SQL	Stockage des données



3. Technologies et langages ⁽²⁾

Couches	Langage	Rôle
Conteneurisation	Docker	Déploiement et environnement isolé
Méthode de paiement	Stripe	Gestion des paiements en ligne sécurisés et abonnements
Hébergement	Google Cloud	Hébergement centralisé, sécurisé et évolutif du logiciel et des données.
Tests	Postman	Tests des différentes routes REST

Gestion des clients / adhérents

- Création, modification, suppression des profils
- Gestion des rôles : client, adhérent, administrateur

Produits et abonnements

- Création et paramétrage des paniers et abonnements
- Tarifs, remises, promotions

Points de dépôt et livraisons

- Paramétrage des points de dépôt
- Suivi des livraisons et notifications

Cartes, itinéraires et calendrier

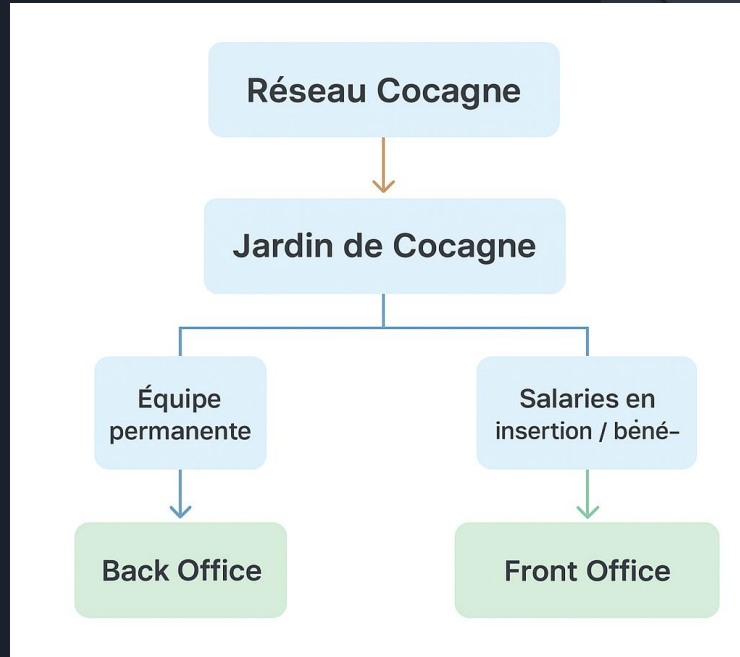
- Intégration Google Maps / OpenStreetMap
- Calcul automatique des itinéraires
- Affichage du calendrier de livraisons

Front office

- Consultation et modification des abonnements
- Paiement en ligne sécurisé
- Historique et suivi des livraisons

4. Modules et fonctionnalités techniques

Vue d'ensemble du système



5. Contraintes techniques

Contrainte	Description
Interface intuitive	L'application doit être simple à utiliser pour tout profil d'utilisateur.
Documentation complète	Permet la maintenance et l'évolution du projet par d'autres développeurs.
Maintenance centralisée	Mises à jour et corrections déployées à distance.
Open source & modulaire	Le code doit pouvoir être adapté et enrichi facilement.

6. Environnement de collaboration

Outil	Description
Versionning	Utilisation de Git pour faciliter la collaboration et la gestion du projet.
CI/CD	Permet le déploiement et la distribution automatique et continue du code.
Gestion de projet	Utilisation de Trello pour centraliser les idées, tâches et suivi de l'avancement.
Tests automatisés	Mise en place de tests unitaires et d'intégration tout au long du projet.

7. Solution d'hébergement

Google Cloud :

- **Scalabilité automatique** : Cloud Run adapte automatiquement les ressources nécessaires en fonction de la charge (pics de paiements ou de commandes).
- **Maintenance simplifiée** : les bases de données et serveurs sont gérés par Google (mises à jour et sécurité automatiques).
- **Coûts optimisés** : facturation à l'usage, avec arrêt automatique des services non utilisés.
- **Sécurité renforcée** : données hébergées en France (Paris), certificats SSL intégrés et pare-feu (*Cloud Armor*).
- **Déploiement rapide** : intégration continue (*CI/CD*) et possibilité de revenir facilement à une version antérieure.
- **Évolutif** : possibilité de migrer plus tard vers Kubernetes pour une architecture plus complexe si nécessaire.

8. Solution pour les tests

Postman :

- **Tests d'API automatisés** : vérifie le bon fonctionnement des routes REST (création, lecture, mise à jour, suppression).
- **Scénarios de test** : simulation complète des échanges entre le front-end et le back-end.
- **Validation des données** : contrôle des réponses JSON, statuts HTTP et temps de réponse.
- **Débogage simplifié** : détection rapide des erreurs d'intégration et de communication.
- **Documentation dynamique** : génération automatique de la documentation des endpoints pour les développeurs.
- **Collaboration facilitée** : partage des tests au sein de l'équipe.