# “‘--

| CODESECURE  Digicheese  **Cahier des charges**  *Rédigé le 13/01/2025 par Christophe Germain*  *Dernière mise à jour : 17/01/2025* |
| --- |

## Sommaire

[**“‘-- 1**](#)

[Sommaire 2](#_1fob9te)

[I- Descriptif du projet 4](#_1t3h5sf)

[II- Equipe du projet 5](#_4d34og8)

[III- Contexte du projet 6](#_26in1rg)

[1- Exposé de la situation 6](#_lnxbz9)

[2- Nos objectifs : 7](#_35nkun2)

[IV- Architecture Technique 9](#_pjn436fluwpc)

[1- Framework de développement : 9](#_zhdnzj6m94kp)

[2- Architecture Logicielle 9](#_cz9yqkmri6pm)

[3- Architecture Système 10](#_aefpw463hw9e)

[IV- Description fonctionnelle des besoins 12](#_1ksv4uv)

[1- Accueil et connexion 12](#_44sinio)

[2- Fonctionnalités 12](#_2jxsxqh)

[❖ Fonctionnalité [1] : Gestion de la base de donnée 12](#_ssi5m8q3qwle)

[Sous fonctionnalité : CRUD Utilisateurs 12](#_k04gx01ddn7d)

[Sous fonctionnalité : CRUD Communes 14](#_iemfzt2eiukj)

[Sous fonctionnalité : CRUD Objets 14](#_3sp523maulsb)

[Sous fonctionnalité : CRUD Conditionnements 15](#_2ximn9wa9xw)

[❖ Fonctionnalité [2] : Création de commandes 16](#_b8a7aoloibdc)

[❖ Fonctionnalité [3] : Communication Client 17](#_pkkzobichqz0)

[Sous fonctionnalité : Mailing Client 17](#_2g4osymq5l1t)

[Sous fonctionnalité : Envoi de Newsletter 18](#_mta90xazd8cc)

[❖ Fonctionnalité [4] : Gestion des stocks 18](#_vvai1uop2oki)

[Sous fonctionnalité : Mise à jour automatique des stocks 18](#_4e9pha15thth)

[Sous fonctionnalité : Mise à jour de l’inventaire 18](#_xs5eug4sbkx2)

[Sous fonctionnalité : Imprimer rapport stocks 19](#_6n3plgsjls)

[❖ Fonctionnalité [5] : Extraction de statistiques 20](#_s2nvan5sadq9)

[V- Sécurité 21](#_1y810tw)

[1- Sécurité des Données 21](#_1kz12q3y312i)

[2- Non-conformité à la Réglementation 21](#_6tssl1jxgx57)

[3- Gestion des Accès et des Rôles ( à proposer au client) 21](#_y89fjhtednax)

[VI - Budget 23](#_1ci93xb)

[Ressources Humaines : 23](#_xnza291pzo6s)

[Macro-planning 23](#_sjg3trenhxcc)

[Tableau des coûts 24](#_ay61dwb2bj2d)

[VII - Calendrier 25](#_2bn6wsx)

[Tâches supplémentaires possibles et à proposer 25](#_qa2qolmrjm5u)

## I- Descriptif du projet

La **Fromagerie DIGICHEESE**, une entreprise de taille petite à moyenne (TPE), cherche à refondre son application de gestion des cadeaux fidélité. L’application actuelle, créée il y a plus de 20 ans, est obsolète, instable et difficile à maintenir. Elle ne répond plus aux besoins croissants de l’entreprise. L’objectif est de développer une nouvelle solution efficace, plus robuste et plus évolutive.

Clients :

### 

| **Nom – Prénom** | **Rôle projet** | **Société** | **E-mail de contact** |
| --- | --- | --- | --- |
| **Fromagerie DIGICHEESE** | Client principal | Digicheese.fr | contact@DIGICHEESE.com |

## II- Equipe du projet

### 

| **Nom – Prénom** | **Rôle projet** | **Société** | **E-mail de contact** |
| --- | --- | --- | --- |
| **KOENIG Olivier** | développeur concepteur | CodeSecure | okoenig@diginamic-formation.fr |
| **SACKS Alexandre** | développeur concepteur | CodeSecure | alexandre.sacks@gmail.com |
| **GODINEAU Maxime** | développeur concepteur | CodeSecure | godineau.pro@gmail.com |
| **KELONKEU Sylvie** | développeur concepteur | CodeSecure | kelonkeu.tchoumi@gmail.com |
| **Christophe Germain** | Chef de Projet | Tecken | cgermain@diginamic.fr |

## III- Contexte du projet

#### 1- Exposé de la situation

La Fromagerie DIGICHEESE, entreprise familiale centenaire, gère un programme de fidélité destiné à ses clients particuliers et distributeurs. Ce programme repose sur un système manuel de collecte de points et l'échange de ces derniers contre des "goodies" (cadeaux). Le traitement des commandes de fidélité est assuré par une application développée sous Access Office 2000, un système devenu obsolète.

Aujourd'hui, nous rencontrons les problématiques suivantes :

L’application actuelle est basée sur Microsoft Access et utilise VBA pour le traitement des données. Elle présente les problèmes suivants :

* Instabilité : des bugs fréquents perturbent son fonctionnement.
* Maintenance : difficile à maintenir et à mettre à jour.
* Évolutivité : peu de possibilités d’ajout de nouvelles fonctionnalités.
* Accessibilité : manque de fluidité et de visibilité pour les utilisateurs.

L'application Access 2000 présente plusieurs limites qui compromettent son efficacité et sa pertinence dans un environnement technologique moderne. Tout d'abord, son obsolescence technologique est notable. Elle se traduit par une instabilité fréquente, une incompatibilité avec les systèmes d'exploitation récents et une maintenance complexe, rendue difficile par un code vieillissant et peu adaptable.

Par ailleurs, l'ergonomie de l'application est dépassée. Son interface utilisateur rigide et peu intuitive freine les utilisateurs. Les fonctionnalités, souvent ajoutées sans cohérence, alourdissent encore davantage la navigation et nuisent à l'expérience globale.

Sur le plan des performances, l'application montre ses limites. Les temps de traitement des commandes sont longs, et elle peine à gérer de gros volumes de données. De plus, les interactions avec d'autres systèmes sont souvent problématiques, ce qui entrave la fluidité des opérations.

Enfin, l'application engendre des risques organisationnels majeurs. Une forte dépendance aux opérateurs pour le traitement manuel des commandes expose l'entreprise à des erreurs humaines. De même, la gestion des stocks et des colis s'avère complexe, augmentant les risques d'inefficacité et de dysfonctionnements.

Les principaux axes de développements pour répondre à ces problématiques sont :

* **Authentification** : Page de connexion, accessible à tous les utilisateurs.
* **Administration (admin)** : Accès pour les administrateurs (gestion des utilisateurs et des paramètres).
* **Gestion des colis (OP-colis)** : Espace pour les gestionnaires des colis, incluant la gestion des clients, commandes, et emballages.
* **Gestion des stocks (OP-stocks)** : Espace pour les gestionnaires de stock, incluant la gestion des inventaires.

#### 2- Nos objectifs :

**Objectif 1 : Améliorer l'interface utilisateur**Le premier objectif consiste à offrir une interface utilisateur plus ergonomique et efficace pour le client. Cela passe par une mise à jour complète de la partie front-end de l'application, permettant ainsi une meilleure expérience utilisateur et une navigation simplifiée.

**Objectif 2 : Renforcer les performances et la fiabilité**Le deuxième objectif vise à augmenter la performance et la fiabilité de l'application. Cela sera réalisé en modernisant l'infrastructure back-end, notamment par l'optimisation des interactions entre l'application et la base de données, garantissant ainsi des opérations plus rapides et stables.

**Objectif 3 : Transition et optimisation technologiques**L'objectif est de migrer vers une solution web moderne en utilisant des technologies récentes, telles que des frameworks JavaScript et des bases de données relationnelles robustes, tout en privilégiant un hébergement intranet ou cloud. Cette transition permettra de moderniser l'application et de la rendre plus flexible.

L'automatisation des processus de traitement des commandes et de gestion des stocks permettra de réduire les erreurs humaines et d'accélérer les délais de traitement. En parallèle, une interface ergonomique et intuitive sera développée pour simplifier les tâches des opérateurs et améliorer l'expérience utilisateur.

Enfin, l'environnement sera conçu pour être facilement évolutif, afin de répondre aux besoins futurs tout en réduisant les coûts de maintenance.

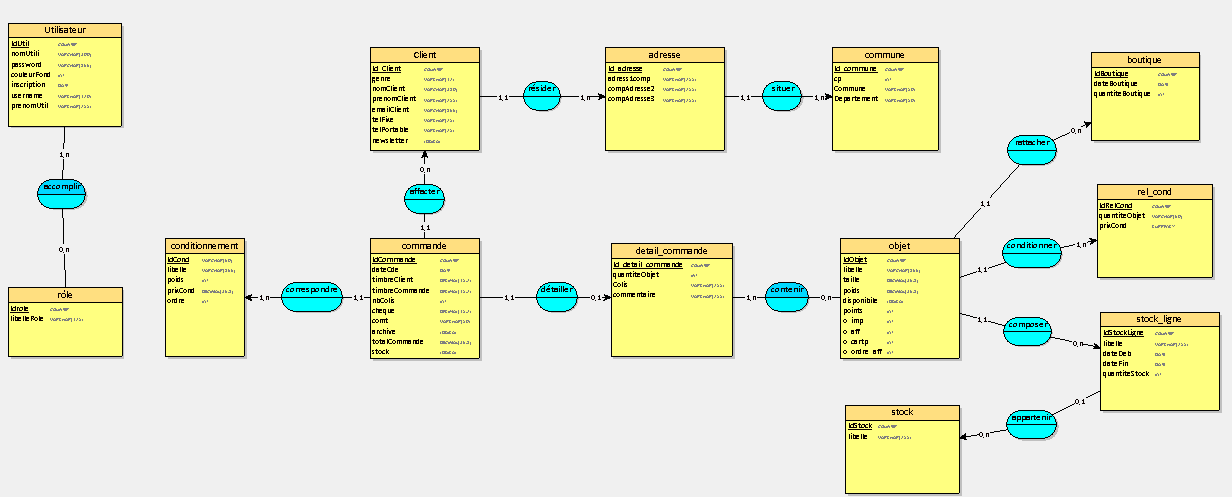
## IV- Architecture Technique

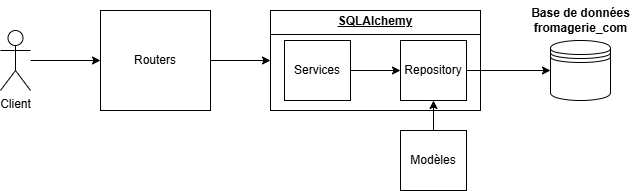
#### 1- Framework de développement :

Le projet sera développé en utilisant les technologies et outils suivants :

* **Langage principal :** Python (3.11+).
* **Framework backend :** FastAPI, pour sa légèreté et sa compatibilité avec les normes modernes (OpenAPI).
* **ORM :** SQLAlchemy, permettant une gestion efficace et flexible des bases de données.
* **Base de données :** MariaDB, choisie pour ses performances et sa compatibilité avec les applications critiques.
* **Serveur d’application :** Uvicorn, assurant une exécution rapide et asynchrone de l’API.
* **Tests :** Pytest, pour une validation rapide et exhaustive des fonctionnalités.

#### 2- Architecture Logicielle





### 

#### 3- Architecture Système

| **Composant** | **OS** | **@IP** | **Hostname** | **Statut** |
| --- | --- | --- | --- | --- |
| Hyperviseur | Proxmox VE 8.3 | 192.168.198.3 | digicheese.com | A construire |
| Serveur applicatif | Debian 12.9 LTS | 10.0.1.2 | digicheeseapp.com | A construire |
| Serveur DB | Debian 12.9 LTS | 10.0.1.3 | digicheesedb.com | A construire |

**digicheese.com**

Type: Hyperviseur

CPU: 3.0Ghz 8 Coeurs 16 Threads

RAM: 128GB

Stockage: 10TB HDD + 1TB SSD

**digicheeseapp.com**

Type : VM

UC : 2.5 GHz

Mémoire : 2 GB

Stockage : 500 GB

Middleware : SQLAlchemy

Paramètres système :

Néant Port d’écoute : 443

**digicheesedb.com**

Type: VM

UC : 2.5 GHz

Mémoire : 4 GB

Stockage : 1 TB

Middleware : MariaDB

Néant Port d’écoute : 5342

## IV- Description fonctionnelle des besoins

#### 1- Accueil et connexion

**Il est possible d'accéder à la plateforme selon les rôles suivants :**

* **Rôle 1** : **Administrateur** : Gestion des utilisateurs, communes, objets, conditionnements, etc.
* **Rôle 2** : **Opérateur colis (OP-colis)** : Gestion des clients, commandes, calculs, envoi de mailing, génération de statistiques.
* **Rôle 3** : **Opérateur stock (OP-stock)** : Gestion des stocks et inventaires, génération de rapports papier.

#### 2- Fonctionnalités

##### **Fonctionnalité [1] : Gestion de la base de donnée**

###### 

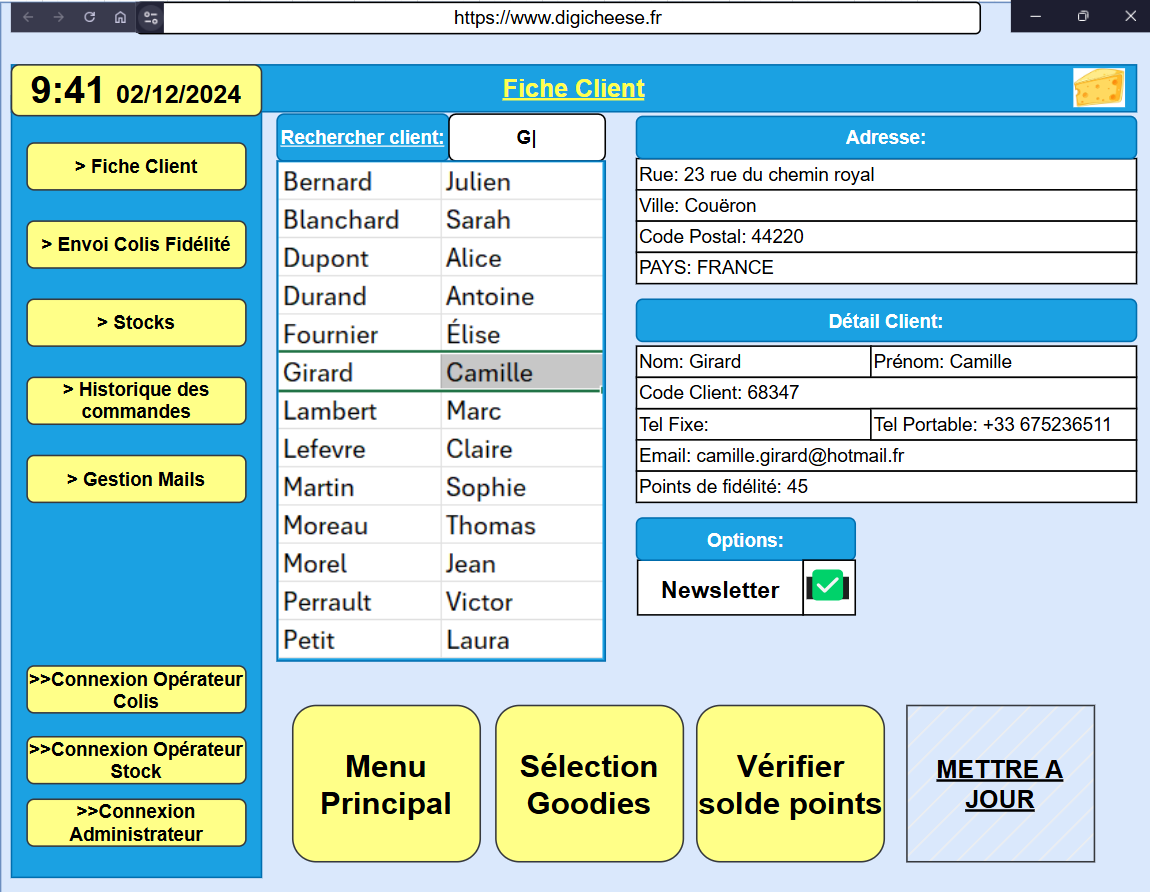
###### Sous fonctionnalité : CRUD Utilisateurs

**Objectif :** Assurer une gestion complète et précise des informations clients dans la base de données, afin de garantir un suivi efficace et personnalisé des relations clients.

**Description :** En tant qu’**Administrateur**, je peux gérer les informations des utilisateurs dans le système à travers les opérations CRUD (Créer, Lire, Mettre à jour, Supprimer). Cela inclut la création de nouveaux utilisateurs pour assurer l'enregistrement des clients dans la base de données, ainsi que la possibilité de mettre à jour ou de supprimer leurs informations afin de maintenir l'exactitude des données clients. Cette gestion permet d'optimiser le suivi des interactions, la personnalisation du service et la gestion des commandes, tout en assurant une expérience client fluide et efficace.

Routes des Endpoint API pour la gestion des Clients

| **Méthode** | **Endpoint** | **Description** | **CRUD** |
| --- | --- | --- | --- |
| GET | /clients | Afficher la liste de tous les clients | READ |
| POST | /clients | Ajouter un client | CREATE |
| GET | /clients/{id} | Afficher les données d’un client | READ |
| PUT | /clients/{id} | Modifier les données d’un client | UPDATE |
| DELETE | /clients/{id} | Supprimer un client | DELETE |

**

*Maquette illustrative*

###### Sous fonctionnalité : CRUD Communes

**Objectif :** Assurer une gestion optimale des communes dans la base de données, afin de faciliter la distribution des colis et la gestion des zones géographiques.

**Description :** En tant qu'Administrateur, je peux gérer les communes dans le système à travers les opérations CRUD (Créer, Lire, Mettre à jour, Supprimer). Cela inclut la création de communes pour garantir que toutes les informations géographiques sont à jour et précises, ainsi que la possibilité de mettre à jour ou de supprimer des communes pour maintenir l’exactitude des données. Cette gestion permet d'optimiser les processus logistiques, en assurant une distribution efficace des colis selon les zones géographiques, et garantit un service rapide et fiable pour le client.

Routes des Endpoint API pour la gestion des Communes

| **Méthode** | **Endpoint** | **Description** | **CRUD** |
| --- | --- | --- | --- |
| GET | /communes | Afficher la liste de tous les communes | READ |
| POST | /communes | Ajouter une commune | CREATE |
| GET | /communes/{id} | Afficher les données d’une commune | READ |
| PUT | /communes/{id} | Modifier les données d’une commune | UPDATE |
| DELETE | /communes/{id} | Supprimer une commune | DELETE |

###### Sous fonctionnalité : CRUD Objets

**Objectif :** Assurer une gestion optimale des objets dans la base de données, afin de faciliter la gestion des stocks, des commandes et des envois de colis.

**Description :** En tant qu'**Administrateur**, je peux gérer les objets dans le système à travers les opérations CRUD (Créer, Lire, Mettre à jour, Supprimer). Cela inclut la création d'objets pour garantir que toutes les informations relatives aux produits sont à jour et correctes, ainsi que la possibilité de mettre à jour ou de supprimer des objets pour maintenir la précision des données. Cette gestion permet d'optimiser les processus de gestion des stocks, des commandes et des expéditions, assurant ainsi un service rapide et fiable pour le client.

Routes des Endpoint API pour la gestion des Objets

| **Méthode** | **Endpoint** | **Description** | **CRUD** |
| --- | --- | --- | --- |
| GET | /objets | Afficher la liste de tous les objets | READ |
| POST | /objets | Ajouter une objet | CREATE |
| GET | /objets/{id} | Afficher les données d’un objet | READ |
| PUT | /objets/{id} | Modifier les données d’un objet | UPDATE |
| DELETE | /objets/{id} | Supprimer un objet | DELETE |

###### Sous fonctionnalité : CRUD Conditionnements

**Objectif :** Optimiser le choix et la gestion des emballages en fonction des besoins logistiques et des spécifications des commandes, afin d'assurer une expédition efficace.

**Description :**

En tant qu'**Administrateur**, il est possible de gérer diverses tâches liées aux conditions d'emballage, notamment pour une gestion complète et précise des emballages disponibles dans le système. En effet, avec ces trois opérations CRUD (Create, Read, Update et Delete), l'Admin peut contrôler les informations relatives à chaque type de conditionnement, telles que la taille, le poids et les spécifications des produits, permettant ainsi une gestion optimale des emballages disponibles dans le système. Cela lui permet également d'assurer la disponibilité des types d'emballages nécessaires et de gérer efficacement les stocks d'emballages pour garantir la satisfaction client.

Routes des Endpoint API pour la gestion des Conditionnements

| **Méthode** | **Endpoint** | **Description** | **CRUD** |
| --- | --- | --- | --- |
| GET | /conditionnement | Afficher la liste de tous les conditionnements | READ |
| POST | /conditionnement | Ajouter un conditionnement | CREATE |
| GET | /conditionnement/{id} | Afficher les données d’un conditionnement | READ |
| PUT | /conditionnement/{id} | Modifier les données d’un conditionnement | UPDATE |
| DELETE | /conditionnement/{id} | Supprimer un conditionnement | DELETE |

##### **Fonctionnalité [2] : Création de commandes**

**Objectif:**  Permettre de créer, mettre à jour et supprimer des commandes en enregistrant et ajustant tous les détails requis tels que les articles, les quantités et les informations de livraison, afin de garantir un traitement rapide, précis et flexible des commandes, faciliter leur suivi et optimiser la satisfaction client.

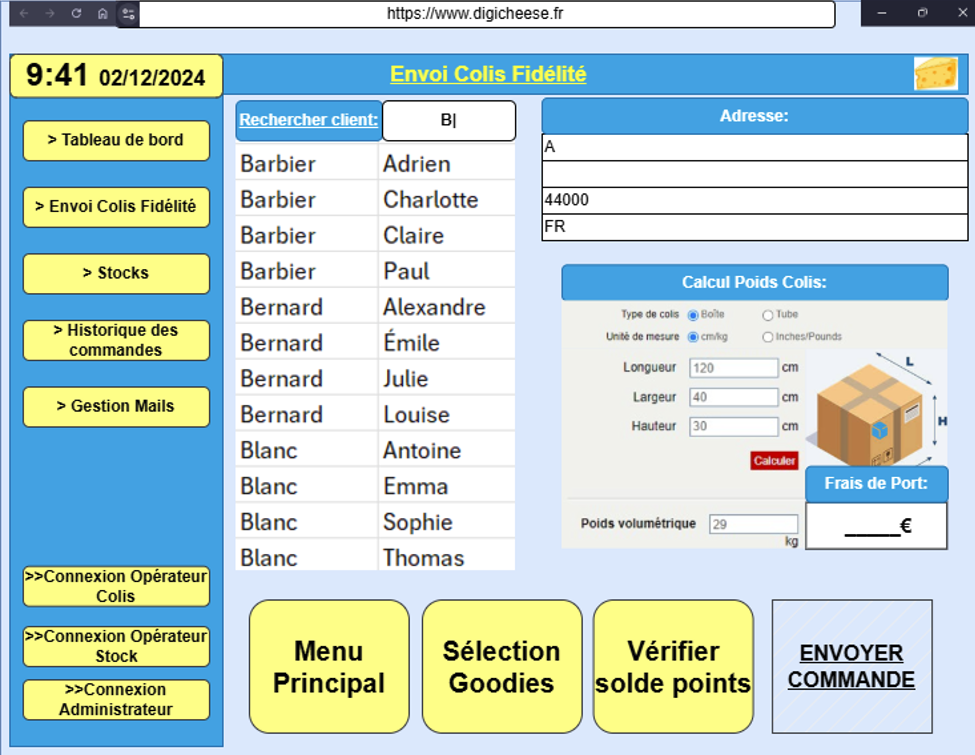
**Description :**

En tant qu’**Opérateur colis (OP-colis)**, je peux créer des commandes en renseignant les informations nécessaires, telles que les articles, les quantités, et les informations de livraison, pour initier le traitement de manière fluide.

Je peux également consulter les informations des commandes existantes pour suivre leur état et progression, permettant ainsi une gestion proactive.

Si des ajustements sont nécessaires, je peux modifier les informations des commandes, que ce soit pour corriger des erreurs ou ajouter des détails importants.

De plus, je peux supprimer des commandes lorsque cela est requis, tout en garantissant une gestion rigoureuse et une traçabilité complète du processus. Cette fonctionnalité assure une gestion flexible et réactive des commandes, contribuant à une expérience client optimisée et à une gestion des stocks fluide.



*Maquette illustrative*

##### **Fonctionnalité [3] : Communication Client**

###### Sous fonctionnalité : Mailing Client

**Objectif :** Permet de faciliter la communication avec les clients en avec l’envoi des mails de confirmation de commande. Cette fonctionnalité vise à informer les clients en temps réel de l'état de leur commande, renforçant ainsi leur confiance et leur satisfaction.

**Description :**

En tant qu’**Opérateur colis (OP-colis)**, je peux envoyer un mail de confirmation de commande aux clients ayant commandé des goodies, afin de les informer du traitement de leur commande, de leur fournir les informations nécessaires concernant la livraison, et ainsi contribuer à la fidélisation du client en offrant une expérience transparente et rassurante.

###### Sous fonctionnalité : Envoi de Newsletter

**Objectif :** Permettre d'envoyer des newsletters ciblées aux clients pour les tenir informés des nouveautés, promotions, et actualités de l’entreprise. Cette fonctionnalité vise à renforcer l'engagement client et à stimuler les interactions commerciales.

**Description :**

En tant qu’**Opérateur colis (OP-colis)**, je peux créer et envoyer des newsletters personnalisées aux clients en fonction de leurs préférences ou de leurs historiques d'achat, permettant ainsi de maintenir une communication pertinente et de renforcer la relation client.

##### **Fonctionnalité [4] : Gestion des stocks**

###### Sous fonctionnalité : Mise à jour automatique des stocks

**Objectif:** Automatiser le processus de mise à jour des stocks pour garantir une gestion en temps réel des inventaires sans intervention manuelle.

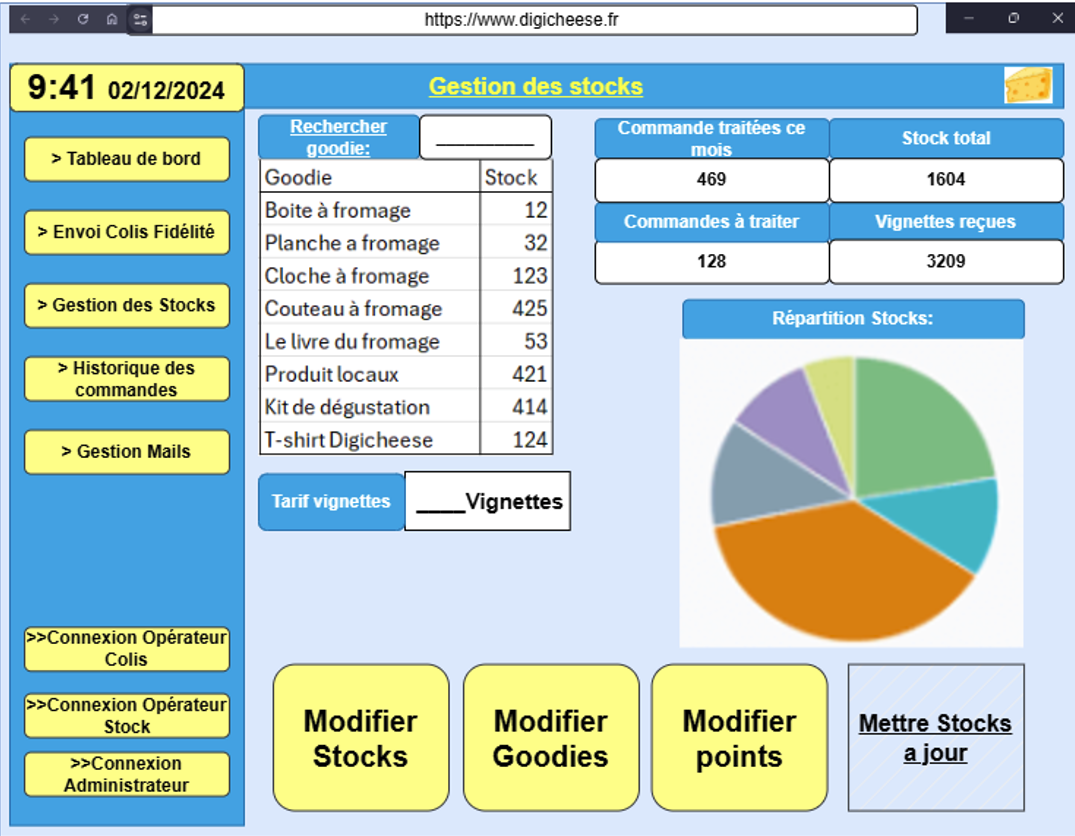
**Description:**

En tant qu'**Opérateur stock (OP-stock)**, je peux configurer l’application pour que les stocks se mettent à jour automatiquement après chaque opération de réception, expédition, ajustement. Cette automatisation permet de réduire les erreurs humaines, d'assurer la précision des données et de garantir une disponibilité constante des produits.

###### Sous fonctionnalité : Mise à jour de l’inventaire

**Objectif:** Permettre de mettre à jour manuellement l’inventaire pour refléter les mouvements de stock, les ajustements ou les réceptions non planifiées, afin d’assurer une gestion précise des ressources.

**Description:** En tant qu'**Opérateur stock (OP-stock)**, je peux mettre à jour l’inventaire de l’entrepôt en ajoutant ou modifiant les quantités des produits selon les opérations physiques réalisées ou les corrections nécessaires, pour garantir l’exactitude du stock disponible. Cette fonctionnalité permet ainsi de maintenir un suivi rigoureux et fiable des stocks en temps réel.

**

###### Sous fonctionnalité : Imprimer rapport stocks

**Objectif** : Optimiser la gestion et l’utilisation des emballages en fournissant des rapports clairs et détaillés sur les stocks disponibles, alignés sur les besoins logistiques et les spécifications des commandes, pour assurer une expédition fluide et efficace.

**Description :** En tant qu’**Opérateur stock (OP-stock)**, je peux générer et imprimer des rapports détaillés et synthétiques sur les stocks d’emballages, incluant des informations telles que les quantités disponibles, les seuils critiques, et l’adéquation avec les types de commandes. Ces rapports sont personnalisables selon des critères spécifiques (type d’emballage, période, seuil critique, état des stocks) pour faciliter la planification, la coordination logistique et le suivi des ressources. L'objectif est d’offrir aux équipes une vision claire et précise des stocks, afin d’anticiper les besoins, éviter les ruptures et garantir une gestion optimale des ressources.

##### **Fonctionnalité [5] : Extraction de statistiques**

**Objectif** : Permettre aux utilisateurs d'accéder, de visualiser et de générer des rapports statistiques sur le service de colisage de fidélité, afin de les partager efficacement avec d'autres collaborateurs ou parties prenantes.

**Description :** En tant qu’**Opérateur colis (OP-colis)**, je peux utiliser la fonctionnalité d'extraction de statistiques pour collecter, analyser et présenter les données du service de colisage de fidélité de manière claire et exploitable. Elle offre des outils pour générer des tableaux, des graphiques interactifs et des rapports personnalisés, avec la possibilité d'appliquer des filtres avancés (par période, région, client, etc.). Les utilisateurs peuvent exporter les rapports dans divers formats (PDF, Excel, PowerPoint) afin de les partager avec d'autres collaborateurs. Cette fonctionnalité, mise à jour en temps réel, vise à améliorer la prise de décision et la collaboration interne en fournissant des informations précises et faciles à comprendre.

## V- Sécurité

#### 1- Sécurité des Données

**Objectif :** Protéger les informations sensibles des utilisateurs, clients, commandes et stocks.

**Mesures :**

Chiffrement des données sensibles (mot de passe, coordonnées clients, etc.):

* Mots de Passe: Hachage non réversible avec Argon2
* Chiffrement des Coordonnées clients et autres données Sensibles via un chiffrement symétrique AES (Advanced Encryption Standard). L’AES sera implémenté à l'aide de la librairie Python Cryptography.fernet.
* Utilisation de gestionnaire de secrets.
* mise en place d’un fichier .env pour des secrets indépendants par client

Mise en place de sauvegardes régulières pour éviter la perte de données.

#### 2- Non-conformité à la Réglementation

**Objectif :** Garantir la conformité du système aux réglementations applicables pour éviter des sanctions, protéger les données des utilisateurs et maintenir la crédibilité de l’entreprise.

**Mesures :**

Audits réguliers

Formation des équipes aux bonnes pratiques liées à la protection des données et aux réglementations pertinentes.

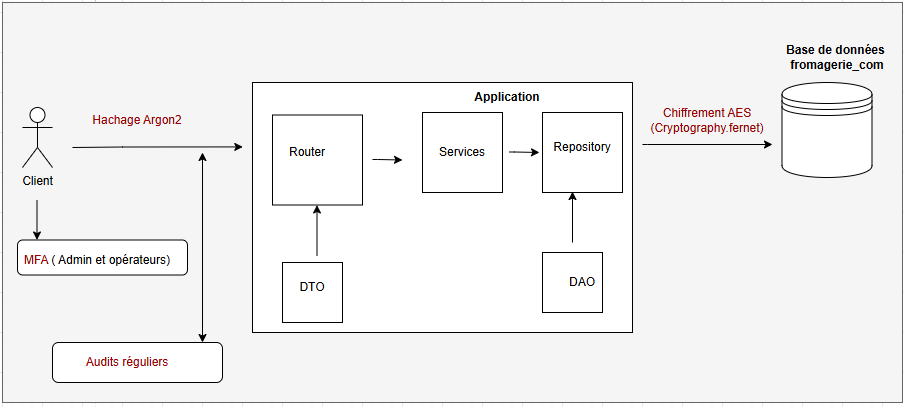
#### 3- Gestion des Accès et des Rôles ( à proposer au client)

**Objectif:** Assurer un accès contrôlé et sécurisé aux fonctionnalités du système.

**Mesures:**

Authentification multi-facteurs (MFA) pour l'administrateur, Opérateur colis, Opérateur de stock.

Gestion des rôles et permissions (Admin, Opérateur colis, Opérateur de stock).



## VI - Budget

#### Ressources **Humaines :**

* 1 Chef de projet (Christophe Germain)
* 4 Développeurs full-stack

**Ressources Matérielles :**

* Serveur Dell PowerEdge T550

**Ressources Financières :**

Budget estimé : 53 774 €/HT charge 15 % comprise.

#### Macro-planning

| **Tâche** | **Durée** | **Responsable** | **coût** |
| --- | --- | --- | --- |
| Analyses des besoins | * 5jh | Chef de projet (Christophe) | 700€ / jour ht |
| Développement Front-end | * 56jh | Développeur full-stack junior | 350€ / jour ht |
| Intégration API | * 28jh | Développeur Full-stack junior | 350€ / jour ht |
| Tests et validation | * 28jh | Développeur Full-stack junior | 350€ / jour ht |

#### Tableau des coûts

| **Element** | **Coût** | **Détails** |
| --- | --- | --- |
| Salaires | Développeurs:  350€ x 112jh = 39 200 €  Chef de projet = 700€x5jh 3500 €  Total = 42 700 € | Salaires équipe de projet |
| Maintenance logicielle 2 ans | 500 € | Permet une maintenance du code |
| Maintenance corrective de l’infrastructure 2 ans | 500 € | Permet une résolution d’incident de l’infrastructure sans frais supplémentaires |
| Serveur Dell PowerEdge T550 | 2000 € | Infrastructure |
| Licence Proxmox VE Standard 2 ans | 1060 € | Outils, licences |
| Total budget | 46 760€ | Sans marge |
| Marge 15% | 7014 € | Pour frais imprévus |
| Total budget | 53 774 € | Marge comprise |

## VII - Calendrier

| **Semaine 1** | **Semaine 2** | **Semaine 3** | **Semaine 4** | **Semaine 5+** |
| --- | --- | --- | --- | --- |
| **Analyse des besoins** | **Intégration API** | **Tests et validation API** | **Développement Front-end** | **Maintenance** |
| *Étude de la documentation technique.* | *Choix des API à intégrer et configuration des environnements.* | *Mise en place d’un plan de test complet (tests de performance, de sécurité, de compatibilité).* | *Conception des maquettes (wireframes et prototypes).* | *Surveillance des performances et correction des anomalies.* |
| *Réunion avec les parties prenantes pour clarifier les objectifs.* | *Développement de scripts pour consommer les API.* | *Exécution des tests sur les différentes fonctionnalités de l’API.* | *Développement des interfaces utilisateur (UI).* | *Mise à jour régulière de l’API et des dépendances.* |
| *Identification des contraintes et des priorités.* | *Mise en place de tests unitaires pour vérifier la connectivité.* | *Résolution des bugs identifiés lors des tests.* | *Intégration des appels API dans les composants front-end.* | *Documentation pour faciliter la maintenance future.* |
| *Création d’un cahier des charges détaillé.* | *Documentation des processus d’intégration.* | *Validation finale avec les parties prenantes.* | *Tests d’interaction utilisateur (UX).* | *Formation ou transfert de connaissances à l’équipe de support.* |
| *Validation du plan de travail.* |  |  |  |  |

### Tâches supplémentaires possibles et à proposer

Ajoutez les tâches spécifiques à votre projet dans les créneaux libres :

* Amélioration des performances.
* Configuration du déploiement continu (CI/CD).
* Sécurisation des données (chiffrement, contrôle d'accès).
* Gestion des retours des utilisateurs.
* Création de rapports d’utilisation.
* Automatisation des processus répétitifs.