République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abderrahmane Mira - Béjaïa

Faculté des Sciences Exactes

Département d'Informatique

Mémoire de fin de cycle

En vue de l'obtention du Master professionel en Génie Logiciel

Thème

Conception et réalisation d'une application web pour la gestion de pharmacie

Réalisé par :
Mr. BOUHAFNA Moncef
Mlle. BEKHOUS Anais

Encadré par :
Dr GADOUCHE Hania

Examiné par :

Présidente :	Mme KHOULALENE Nadjet	MCB	Université A/Mira - Béjaïa
Examinatrice :	Mme GASMI Badrina	MCB	Université A/Mira - Béjaïa
Examinateur :	Mr BEDJOU Khaled	MCB	Université A/Mira - Béjaïa
Examinateur :	Mr SIDER Abderrahmane	MCA	Université A/Mira - Béjaïa

Année universitaire : 2024/2025

- Remerciements -

Avant toute chose, nous souhaitons exprimer notre profonde gratitude à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce travail.

Nous tenons à remercier tout particulièrement **Dr Gadouche Hania**, notre encadrante, pour sa disponibilité, ses conseils avisés, sa rigueur scientifique ainsi que pour sa bienveillance tout au long de ce projet. Son accompagnement constant a grandement contribué à la qualité de ce mémoire.

Nous remercions également les membres du jury pour l'intérêt qu'ils portent à notre travail, pour le temps qu'ils nous consacrent, ainsi que pour leurs remarques constructives qui permettront de faire évoluer notre réflexion.

Nos remerciements s'adressent également aux enseignants du département d'Informatique pour la qualité de l'enseignement dispensé durant notre parcours, ainsi qu'aux responsables pédagogiques pour leur soutien tout au long de notre formation.

Nos remerciements s'adressent également au Dr Boualouche de la CNAS d'Akbou qui nous a fourni toutes les informations techniques nécessaires à la construction de ce système .

- Dédicaces -

Derrière chaque page de ce mémoire, il y a des sourires, du stress, des fous rires et surtout beaucoup d'amour. Cette dédicace est pour vous.

À mes parents,
Surtout à ma chère maman,
Qui m'a porté dans la vie et dans le cœur. Merci pour ton amour inconditionnel, tes prières silencieuses, ta patience et ta force. Maman, merci d'avoir toujours cru en moi, même dans le silence.

À Tata Samira Zaghib et Tonton Raouf Zaghib,
Deux piliers discrets, merci pour votre présence, votre affection et vos conseils pleins de sagesse.

À mes sœurs, Dalia et Pipo et ma grand-mère Mimi,
Pour votre soutien tendre et vos encouragements dans les moments de doute. Ce mémoire est aussi le vôtre.

À toute la famille,
Merci pour votre amour et votre confiance.

À Anais, ma binôme,
Une collègue devenue une amie précieuse. Merci pour ton sérieux, ton humour et ta loyauté tout au long de cette aventure.

À mes amies et amis,
Merci pour vos éclats de rire au cœur du chaos, pour ces instants de légèreté qui ont fait respirer mes journées et l'amitié qui rendait tout ça moins lourds. Vous avez été essentiels.

Merci aussi à ***Lana Del Rey***, dont la voix a été la bande-son de mes nuits de travail, et à moi-même pour être resté debout jusqu'au bout.

- BOUHAFNA Moncef -

- Dédicaces -

En hommage à notre cher camarade Tigrine Youcef, parti trop tôt. Ton art, tes écrits, ta gentillesse et ton sourire constant persisteront à faire vivre ta mémoire dans nos vies.

Un grand merci à **ma maman**, symbole de courage, de résilience et d'amour, qu'elle n'a cessé de me partager tout au long de ma vie.

À ma sœur Elena, ma première fille et la prunelle de mes yeux.

À mon père, qui m'a poussée à poursuivre ce parcours universitaire.

À mes tantes Samia, Sabiha et Kahina, mes meilleures supportrices et mes deuxièmes mamans.

Merci à mes chers amis **Aya, Maya, Ines, Yassou, Idir, Cerine, Nedjima et Kamel,** merci pour votre loyauté et votre soutien.

Merci à mon amie depuis toujours, **Chanez**, d'être mon pilier et ma sœur de cœur.

Et enfin, merci à mon cher ami et binôme **Moncef**. Je n'aurais pas pu choisir une meilleure personne avec qui faire ce projet.

Je vous aime.

- BEKHOUS Anais -

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

API : Application Programming Interface

BDD : Base De Données

CASNOS : Caisse Nationale des Assurances Sociales des Non-Salariés

CNAS : Caisse Nationale des Assurances Sociales

CSS : Cascading Style Sheets

HTML : HyperText Markup Language

IDE : Integrated Development Environment

MVC : Model-View-Controller

OMG : Object Management Group

PDF : Portable Document Format

SGBD : Système de Gestion de Base de Données

UML : Unified Modeling Language

Chapitre 1
Contexte général et étude de l'existant

Chapitre 1 :Contexte général et étude de l'existant

1. Introduction

Le secteur pharmaceutique constitue un pilier fondamental de tout système de soins moderne. En Algérie, les officines jouent un rôle crucial non seulement dans la dispensation des traitements, mais également dans l'accompagnement global des patients. Ce rôle s'intensifie dans un contexte de croissance démographique, d'allongement de l'espérance de vie et de diversification croissante des traitements disponibles.

Les officines connaissent une transformation profonde avec l'intégration progressive des outils numériques, rendant l'informatisation des processus de gestion incontournable. Cette évolution permet une organisation plus rigoureuse, une meilleure traçabilité des traitements, une gestion automatisée des stocks et une consolidation du lien entre le pharmacien et le patient, tout en répondant aux exigences de sécurité, de qualité et de conformité réglementaire. [1]

Malgré les avancées notables dans le domaine de la digitalisation des pharmacies, plusieurs officines en Algérie continuent de fonctionner avec des outils rudimentaires, voire manuellement. Cette situation engendre de nombreux inconvénients : erreurs de gestion de stock, perte d'informations concernant les patients et des difficultés à produire des documents réglementaires rapidement.

Les solutions logicielles actuellement disponibles sur le marché sont souvent coûteuses, peu flexibles et parfois inadaptées aux besoins spécifiques des petites structures locales. De plus, elles ne prennent pas toujours en charge certains aspects cruciaux comme le suivi de l'historique médical des patients ou la gestion fine des ventes et des remboursements.

Face à ces constats, une question centrale se pose : *Comment concevoir une application web adaptée aux pharmacies algériennes permettant une gestion efficace et sécurisée des produits, des ventes, des stocks et de l'historique pharmaceutique des patients ?*

Pour y répondre, plusieurs objectifs ont été définis : automatiser les tâches quotidiennes liées à la gestion des produits, assurer un suivi personnalisé des patients à travers leur historique pharmaceutique, réduire les erreurs humaines grâce à des processus numériques fiables, générer automatiquement les documents nécessaires aux démarches administratives et aux remboursements, améliorer l'organisation interne via une interface adaptée aux rôles du personnel et prendre en compte les spécificités locales telles que les cartes CNAS ou CASNOS.

2. Présentation du domaine pharmaceutique

Le domaine pharmaceutique constitue un maillon essentiel du système de santé, en assurant non seulement la dispensation des médicaments, mais également la prévention des maladies et le conseil thérapeutique aux patients. Il occupe ainsi une position stratégique dans la chaîne de soins, en garantissant l'accès aux traitements, la sécurité d'utilisation des produits pharmaceutiques et la promotion de la santé publique. La pharmacie, et plus particulièrement la pharmacie d'officine, se trouve au cœur de ce dispositif en tant qu'interface directe entre le système médical et la population.

Une pharmacie d'officine ne se limite pas à la simple délivrance de médicaments ; elle joue un rôle multidimensionnel. Elle assure la vente de médicaments avec ou sans ordonnance, la préparation de certaines formules magistrales, la gestion rigoureuse des stocks, et le suivi des dossiers des clients afin d'assurer une traçabilité optimale. Le pharmacien, en tant que professionnel de santé, veille à la vérification des prescriptions médicales, à la prévention des interactions médicamenteuses et à la sensibilisation du patient sur les bonnes pratiques thérapeutiques.

En parallèle, la pharmacie d'officine remplit également une fonction administrative et économique importante. Elle gère les relations avec les organismes sociaux tels que la Caisse Nationale des Assurances Sociales (CNAS) et la Caisse Nationale des Assurances Sociales des Non-Salariés (CASNOS), notamment à travers les procédures de remboursement et de facturation. Cette dimension administrative exige une organisation rigoureuse et une gestion précise des données, tant pour les patients que pour les produits.

Ainsi, le rôle du pharmacien d'officine dépasse largement la simple transaction commerciale : il s'inscrit dans une démarche de santé publique globale, où la qualité du service, la sécurité du patient et l'efficacité de la gestion constituent des enjeux majeurs.

3. Étude de l'existant

Historiquement, les pharmacies étaient gérées de manière manuelle, ce qui engendrait de nombreuses limitations : erreurs humaines, problèmes pour suivre les stocks et perte de temps dans les tâches répétitives. Pour surmonter ces obstacles, l'informatisation du secteur apparaît comme une solution essentielle. Cela permet d'améliorer les opérations quotidiennes.

En Algérie, diverses initiatives provenant du secteur public et privé cherchent à moderniser le fonctionnement des pharmacies à travers des logiciels spécialisés. Des outils tels que PharmaX ou Pharmacium ont été développés pour répondre à ces enjeux. Néanmoins, la majorité de ces systèmes est conçue pour un usage par des particuliers, souvent onéreux et ne s'adaptent pas toujours aux besoins spécifiques de toutes les pharmacies, spécifiquement celles de petite taille situées dans des zones moins densément peuplées, engendrant plusieurs problèmes comme :

- **Manque de traçabilité des traitements :** absence d'un historique détaillé des médicaments délivrés à chaque patient.
- **Difficultés à identifier les interactions médicamenteuses :** faible suivi des prescriptions pouvant entraîner des risques pour la santé des patients.
- **Impossibilité de vérifier la dernière date de dispensation :** sans historique, le pharmacien ne sait pas si un patient a déjà reçu un médicament récemment, ce qui peut entraîner un surdosage ou un traitement inapproprié.
- **Difficulté à analyser les habitudes de consommation :** sans données sur les achats passés, le pharmacien ne peut pas proposer des conseils personnalisés ou détecter une consommation excessive.
- **Manque de preuves pour la CNAS ou les assurances :** en cas de contrôle, il est plus compliqué de justifier la délivrance de certains médicaments sans historique détaillé.
- **Rupture de stock fréquent.**
- **Péremption des produits.**
- **Manque de visibilité sur les stocks en temps réel.**

3.1 Cas d'étude : PharmaX

PharmaX est une application de gestion pharmaceutique qui automatise et optimise les opérations des pharmacies en assurant la gestion des stocks, des ventes, de l'inventaire, des ordonnances et des clients, tout en intégrant des fonctionnalités avancées comme la lecture des codes-barres, l'intégration avec Chifa et un système de fidélisation pour les produits de parapharmacie, afin d'améliorer l'efficacité et la performance des officines, qu'elles soient petites ou grandes. [2] Ci-dessous le logo de PharmaX :

Figure 1 – Logo de PharmaX

Ici le tableau comparatif des points forts et limitations de PharmaX :

Points forts de PharmaX	**Limitations de PharmaX**
Détection automatique des interactions médicamenteuses : — Contrôle instantané pour prévenir les erreurs et assurer la sécurité des patients.	**Absence d'un historique détaillé des traitements :** — Historique des traitements peu détaillé. — Suivi des dossiers patients insuffisant. — Suivi des dates de péremption limité.
Gestion efficace des stocks : — Suivi des entrées et sorties des médicaments avec alertes en cas de stock faible.	**Ergonomie parfois complexe :** — Absence de modernité dans l'expérience utilisateur. — Adaptabilité limitée aux écrans et supports.
Interface adaptée aux pharmacies de toutes tailles : — Ça convient aussi bien aux petites officines qu'aux grandes structures.	**Absence de prise de rendez-vous pour la vaccination :** — L'application ne permet pas aux patients de planifier un créneau de vaccination.
Support du code-barres — Scan rapide et fiable des médicaments.	

Tableau 1 – Comparatif des points forts et limitations de PharmaX

3.2 Cas d'étude : Profficine

Profficine est un logiciel algérien de gestion d'officine pharmaceutique développé pour répondre aux besoins spécifiques des pharmacies locales, notamment la prise en charge des ventes CHIFA, l'inventaire par code-barres, et la gestion des commandes automatiques. Ci-dessous le logo de Profficine :

Figure 2 – Logo de Profficine

Ici le tableau comparatif des points forts et limitations de Profficine :

Points forts de Profficine	Limitations de Profficine
Saisie automatique des ventes CHIFA : — Automatise l'envoi des produits remboursables vers Chifa. — Calcule automatiquement le montant dû selon le type de vente.	**Dépendance à l'infrastructure locale :** — Besoin d'un bon réseau local ou internet pour certaines fonctionnalités. — Performance variable selon le matériel utilisé.
Inventaire facilité par code-barres : — Permet un inventaire rapide, avec codes-barres pour chaque produit.	**Fonctionnalités avancées payantes ou modulaires :** — Certaines options ne sont pas incluses par défaut. — Coût lié aux modules supplémentaires (alertes, rapports, etc.).
Commandes automatisées : — Génère automatiquement les commandes selon les ruptures de stock ou la rotation des produits.	**Adaptabilité limitée dans certaines officines :** — Pour les très petites officines, certaines fonctions sont trop complètes ou peu utilisées. — L'interface peut ne pas convenir à toutes les préférences utilisateurs.
Gestion de la trésorerie et historique des ventes : — Suivi des comptes, caisses et paiements. — Historique des ventes permettant d'analyser l'activité.	**Manque possible de modules spécialisés :** — Peu d'informations sur la présence de modules comme la fidélisation client ou les notifications mobiles. — Certaines fonctions personnalisées peuvent nécessiter un développement spécifique.

Tableau 2 – Comparatif des points forts et limitations de Profficine

3.3 Cas d'étude : SmartPharma

SmartPharma est une solution cloud moderne permettant aux pharmacies de gérer leurs activités depuis n'importe quel appareil (ordinateur, tablette ou smartphone). Elle intègre un système de sauvegarde en ligne, un tableau de bord analytique en temps réel et une connexion avec des API externes (Chifa, CNAS, fournisseurs). L'objectif principal est d'offrir une interface fluide et intelligente pour une gestion centralisée et sécurisée.Ci-dessous le logo de SmartPharma :

Figure 3 – Logo de SmartPharma

Ici le tableau comparatif des points forts et limitations de SmartPharma :

Points forts de SmartPharma	**Limitations de SmartPharma**
Basé sur le cloud : — Accès sécurisé depuis n'importe où. — Sauvegarde automatique des données.	**Dépendance à Internet :** — Nécessite une connexion stable.
Interface moderne et intuitive : — Compatible mobile et tablette.	**Coût d'abonnement mensuel :** — Modèle SaaS avec frais récurrents.
Tableau de bord analytique : — Statistiques sur les ventes, les stocks et les tendances.	**Complexité d'intégration :** — Certaines pharmacies traditionnelles peinent à migrer vers le cloud.

Tableau 3 – Comparatif des points forts et limitations de SmartPharma

3.4 Synthèse comparative des applications étudiées

Le tableau ci-dessous présente une comparaison des trois applications de gestion pharmaceutique étudiées : **PharmaX**, **Profficine** et **SmartPharma**.

Critères	**PharmaX**	**Profficine**	**SmartPharma**
Type d'application	Logiciel local de gestion de pharmacie	Application de bureau connectée à Chifa	Application web et mobile basée sur le cloud
Objectif principal	Automatiser la gestion des ventes et stocks	Gérer l'activité globale d'une officine	Centraliser les données et offrir une gestion à distance
Public cible	Petites et moyennes pharmacies	Pharmacies d'officine locales et régionales	Pharmacies souhaitant une solution moderne et mobile
Fonctionnalités principales	— Gestion des stocks et ventes — Suivi des ordonnances — Lecture code-barres	— Gestion des ventes CHIFA — Trésorerie et historique des ventes — Commandes fournisseurs	— Gestion multi-appareils (web/mobile) — Sauvegarde cloud — Tableau de bord analytique
Forces principales	— Détection d'interactions médicamenteuses — Alertes de stock	— Intégration avec Chifa — Gestion comptable et commandes	— Accessibilité depuis n'importe où — Interface moderne et ergonomique
Limitations	— Interface peu intuitive — Historique patient limité	— Modules avancés payants — Interface classique	— Dépendance à Internet — Abonnement mensuel
Technologie	Logiciel local Windows	Client/serveur local (avec accès réseau)	Cloud SaaS (Web + Mobile)
Orientation future	Modernisation de l'interface et ajout de fonctionnalités en ligne	Simplification et adaptation aux petites officines	Extension vers les API santé et IA prédictive

Tableau 4 – Tableau comparatif des trois applications étudiées

4. Objectifs d'une application de gestion pour pharmacie

Le développement de cette application repose sur une série d'objectifs visant à structurer et optimiser les principales fonctionnalités réalisées au niveau d'une pharmacie, citant :

- Fournir une solution intégrée permettant la gestion complète des stocks de produits, incluant le suivi des entrées, des sorties, des dates de péremption ainsi que des alertes automatiques en cas de niveau de stock critique.
- Assurer la sécurité des patients grâce à la détection automatique des interactions médicamenteuses et et au contrôle rigoureux des ordonnances, afin de minimiser les erreurs de prescription et de délivrance.
- L'application doit permettre aux pharmaciens de conserver et consulter un historique complet et détaillé des médicaments délivrés à chaque client.
- Assurer que l'application respecte les normes et réglementations en vigueur dans le secteur pharmaceutique, notamment en matière de traçabilité et de confidentialité des données.
- Proposer des modules de formation intégrés ainsi qu'un support technique efficace pour faciliter l'adoption de l'application et garantir une prise en main fluide.
- Suivre les ventes liées aux différentes caisses (CNAS et CASNOS), automatiser les calculs de prise en charge et générer les documents nécessaires à la déclaration et au suivi des remboursements.
- Fournir des tableaux de bord et des rapports statistiques détaillés, gestion du personnel et le clients du système.
- Intégrer un module de prise de rendez-vous pour la vaccination, permettant aux patients de réserver un créneau horaire et au personnel d'optimiser l'organisation et la disponibilité des vaccins.

5. Conclusion

Ce premier chapitre nous a permis de poser les bases de notre projet en présentant le contexte général du domaine pharmaceutique, en identifiant les limites des méthodes de gestion traditionnelles, ainsi qu'en étudiant les fonctionnalités et insuffisances d'un système existant tel que *PharmaX*. L'analyse menée a mis en évidence des besoins cruciaux tels que le suivi détaillé des traitements , la gestion optimisée des stocks et l'amélioration de l'expérience utilisateur.

Ces constats nous ont conduit à définir les objectifs fonctionnels et non fonctionnels de notre propre solution. Celle-ci devra répondre aux exigences spécifiques des pharmaciens tout en restant accessible, modulable et conforme aux normes en vigueur.

Chapitre 2 :
Spécifications et conception de l'application

Chapitre 2 : Spécifications et conception de l'application

1. Introduction

La réussite du développement d'une application dépend en grande partie de la clarté et de la précision des spécifications établies. Ce chapitre vise à formaliser les besoins du système, tant sur le plan fonctionnel que non fonctionnel, en se basant sur une analyse rigoureuse du domaine d'application. Ces spécifications servent de fondement à la conception technique de l'application.

Ce chapitre présente l'architecture générale de l'application, en détaillant les composants principaux, leur organisation ainsi que les choix technologiques retenus pour répondre efficacement aux objectifs du projet.

2. Expression des besoins

Cette section présente les besoins fonctionnels et non fonctionnels identifiés pour le développement de l'application, afin d'en guider la conception de manière claire et structurée.

2.1 Besoins fonctionnels

Les besoins identifiés lors de l'analyse ont été regroupés selon les grandes fonctionnalités du système pour une meilleure organisation incluant notamment la gestion des clients, des ventes, de stock, de facturation, de vaccination, des utilisateurs et Dashboard :

Gestion des clients et de leur historique médicamenteux

- Création, modification, suppression et consultation des fichiers clients.
- Enregistrement automatique des médicaments délivrés lors de l'achat.
- Recherche rapide d'un client et consultation de son historique.
- Gestion du dossier pharmaceutique regroupant les informations liées à l'identité du client ainsi que les médicaments délivrés.

Gestion des ventes

- Réalisation rapide des ventes avec un panier dynamique permettant l'ajout ou la suppression de produits en temps réel.
- Alerter le pharmacien en cas de problème détecté :
  - Nombre maximal d'unités du médicament atteint. (Alerte de surdosage)
  - Médicaments incompatibles achetés ensemble (Interactions médicamenteuses)
  - Le patient présente une allergie à une ou plusieurs familles de médicaments délivrés.
  - Médicament servi comme traitement mais pas d'ordonnance enregistrée.

  (L'application informe le vendeur des risques, sans interdire la vente, sauf en cas d'alerte liée à une prescription obligatoire.)

Gestion du stock

- Ajout, modification, suppression et consultation des produits.
- Alerte en cas de stock faible ou de péremption des produits.
- Gestion des entrées et des sorties de produits.
- Affichage du stock en temps réel.
- Génération d'alertes automatiques en cas de stock faible ou de péremption imminente.

Facturation

— Calcul automatique du montant total de la commande avec prise en compte des remises et des couvertures d'assurance.
— Génération automatique de reçus de vente et mise à jour simultanée du stock.

Gestion des utilisateurs

— Création, modification, suppression et consultation des utilisateur.
— Gestion des rôles et permissions afin de limiter l'accès aux données sensibles.

Gestion des rendez-vous de vaccination

— Prise de rendez-vous en ligne pour se faire vacciner.
— Consultation par le pharmacien de l'agenda des rendez-vous confirmés.
— Notification automatique par e-mail pour rappel du rendez-vous.

Tableau de bord analytique

— Visualisation des statistiques liées aux ventes et stock à travers des graphiques interactifs.
— Génération des rapports périodiques exportables au format PDF.

2.2 Besoins non fonctionnels

Performance et réactivité

— L'application doit garantir un temps de réponse rapide, même lors de l'accès simultané de plusieurs utilisateurs.
— Les opérations courantes (vente, recherche d'un client, mise à jour du stock) doivent s'exécuter en moins de 2 secondes.

Sécurité et confidentialité

— L'accès à l'application nécessite une authentification par identifiant et mot de passe.
— Les données sensibles, comme l'historique médical des clients, doivent être protégées contre tout accès non autorisé.
— Les sessions sont gérées de manière sécurisée avec expiration automatique après inactivité.

Fiabilité

— L'application doit fonctionner de manière stable sans erreurs critiques.
— Elle doit être capable de gérer les pannes partielles sans perte de données, avec un système de sauvegarde régulier.
— Elle doit tolérer les pannes et permettre un redémarrage sans perte d'informations.

Ergonomie et facilité d'utilisation

— L'interface utilisateur doit être intuitive et conviviale pour une prise en main rapide.
— Les fonctionnalités doivent être accessibles en quelque clics.

Évolutivité et maintenabilité

— Le système doit être conçu de manière modulaire afin de permettre l'ajout de nouvelles fonctionnalités sans refonte complète.
— Le code doit être documenté et respecter les bonnes pratiques de développement pour faciliter la maintenance.

Portabilité et compatibilité

— L'application doit être adaptable à différents types d'écrans (ordinateurs, tablettes).
— Elle doit être compatible avec les systèmes d'exploitation les plus utilisés, notamment Windows.

3. Identification des acteurs

Pharmacien (Administrateur du système)

— Gestion des comptes des utilisateurs (création, modification et suppression).
— Supervision de la gestion des stocks (ajout, modification et suppression des lots).
— Gestion des produits de la pharmacie (ajout, modification et suppression).
— Gestion des clients du système (ajout, modification et suppression).
— Consultation et mise à jour des données des clients (historique des achats et allergies).
— Accès aux rapports et analyses des ventes et du stock.
— Accès global du pharmacien avec privilèges de consultation des remboursements.
— Et cela en plus des fonctionnalités accordées au vendeur.

Vendeur (Utilisateur classique)

— Réalisation des ventes et enregistrement des achats des clients.
— Consultation de l'historique médicamenteux d'un client pour vérifier d'éventuelles interactions.
— Réception des alertes sur les stocks faibles et la péremption des produits.
— Accès limité du vendeur à la consultation des produits.
— Consultation des rendez-vous de vaccination.

Client

— Consulte son propre historique d'achats et de traitements médicamenteux.
— Accède à ses données de manière sécurisée afin de préserver la confidentialité.

Visiteur

— Peut prendre rendez-vous en ligne pour se faire vacciner.
— Dispose uniquement d'une interface simplifiée orientée vers la réservation.
— Peut consulter la disponibilité des produits (stock).

Voici le diagramme de contexte de l'application spécifiant les différents acteurs du système :

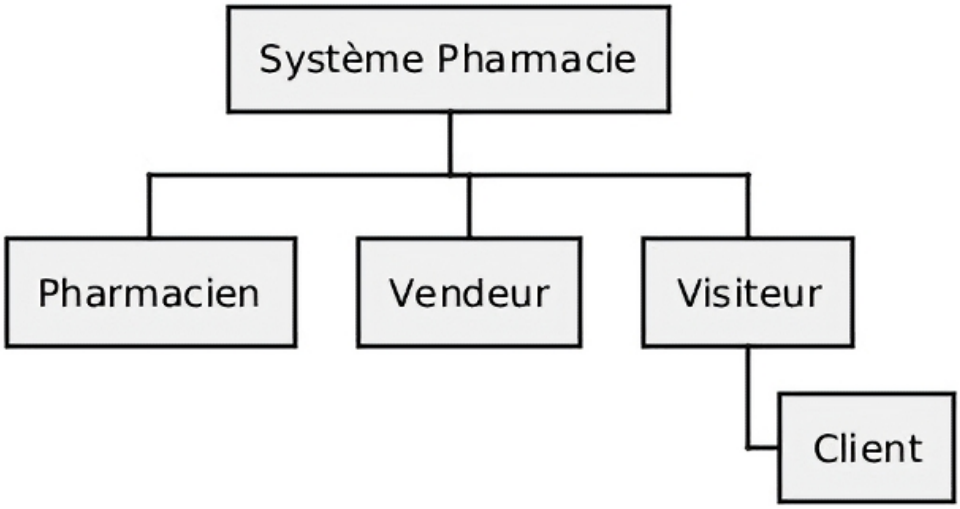

Figure 4 – Organigramme hiérarchique des acteurs

Chapitre 2 : Spécifications et conception de l'application

4. Méthodologie de développement

4.1 Méthodes agiles

Les méthodes agiles ont été conçues en réponse aux limites des approches classiques de développement (cycle en V, cascade). Ces dernières imposaient une planification rigide et laissant peu de place aux changements en cours de projet, ce qui entraînait souvent des dépassements de délais et des écarts par rapport aux besoins réels du client.

Les méthodes agiles, au contraire, reposent sur des cycles courts et adaptatifs, intégrant en contenu les retours des utilisateurs et favorisant la communication entre les parties prenantes.

Le *Manifeste Agile (2001)* définit quatre valeurs fondamentales :
— Privilégier les individus et leurs interactions plutôt que les processus et outils.
— Produire un logiciel opérationnel plutôt qu'une documentation exhaustive.
— Collaborer avec le client plutôt que négocier un contrat.
— S'adapter au changement plutôt que suivre un plan rigide. [3]

4.2 Gestion de projet avec Scrum

Parmi les différentes méthodes agiles, nous avons choisi **Scrum**, un cadre de travail léger mais structuré, permettant de gérer efficacement un projet logiciel grâce à des cycles courts appelés **sprints**. Chaque sprint dure généralement entre deux et quatre semaines et se conclut par un incrément fonctionnel du produit.

Le développement de notre application de gestion pharmaceutique s'est appuyé sur la méthode **Scrum**. Chaque sprint a été structuré autour de cycles courts, intégrant des phases de planification, réalisation, revue et rétrospective. Cette approche a permis une meilleure gestion des priorités, une adaptation continue aux besoins exprimés et une amélioration progressive du produit.

Nous avons défini un **Product Backlog** global regroupant l'ensemble des fonctionnalités attendues du projet, classées par priorité et servant de référence. À partir de ce dernier, le travail a été organisé en **quatre sprints** successifs, chacun disposant de son **Sprint Backlog** propre, constitué d'un sous-ensemble du Product Backlog et représentant les éléments à réaliser durant l'itération correspondante.

Voici le Product Backlog global de l'application :

ID	User Story / Fonctionnalité	Priorité	Valeur métier	Critères d'acceptation
PB1	En tant que pharmacien, je veux **gérer les produits** afin d'ajouter, modifier et supprimer les références disponibles.	Haute	Essentielle pour l'activité quotidienne	— Ajout d'un produit — Modification d'un produit existant — Suppression d'un produit — Consultation d'un produit
PB2	En tant que pharmacien ou vendeur, je veux **gérer le stock** afin de suivre les quantités et éviter les ruptures.	Haute	Essentielle pour la continuité du service	— Visualisation du stock en temps réel — Alertes automatiques de stock bas — Historique des sorties
PB3	En tant que pharmacien ou vendeur, je veux **enregistrer une vente** afin de générer une facture.	Haute	Génération de revenus	— Facture générée automatiquement — Application des remises — Suivre la logique métier
PB4	En tant que pharmacien, je veux **gérer les utilisateurs** afin de contrôler l'accès à l'application.	Haute	Sécurité et organisation interne	— Authentification avec mot de passe chiffré — Attribution de rôles (pharmacien, vendeur, client) — Permissions différenciées
PB5	En tant que pharmacien ou vendeur, je veux **gérer les informations des clients** afin d'assurer un meilleur suivi.	Moyenne	Fidélisation et suivi médical	— Informations personnelles enregistrées — Cartes CNAS/CASNOS — Historique des achats lié au client
PB6	En tant que pharmacien ou vendeur, je veux **consulter l'historique médical** afin de mieux adapter les traitements.	Moyenne	Amélioration du suivi patient	— Historique lié aux prescriptions — Accès rapide depuis la fiche client
PB7	En tant que pharmacien ou vendeur, je veux **consulter les rendez-vous de vaccination** afin de mieux organiser mon travail.	Moyenne	Services clients	— Affichage de la liste des rendez-vous programmés — Consultation des informations liées (client, date, type de vaccin)

Tableau 5 – Product Backlog global de l'application

ID	User Story / Fonctionnalité	Priorité	Valeur métier	Critères d'acceptation
PB8	En tant qu'utilisateur, je veux **bénéficier d'une sécurité renforcée** afin de protéger mes données personnelles.	Haute	Conformité et protection des données	— Chiffrement des mots de passe (bcrypt) — reCaptcha lors de la connexion — Sessions sécurisées
PB9	En tant que pharmacien, je veux **consulter les remboursements** afin de vérifier les dossiers existants.	Haute	Suivi interne	— Affichage de la liste des remboursements — Consultation du statut (en attente, validé, rejeté)
PB10	En tant que pharmacien, je veux **générer des rapports et statistiques** afin d'analyser l'activité de la pharmacie.	Faible	Pilotage et décision	— Top ventes du mois — Ventes par vendeur — Revenu quotidien / mensuel
PB11	En tant que visiteur, je veux **prendre un rendez-vous de vaccination** afin de planifier une intervention.	Moyenne	Accessibilité	— Saisie des informations personnelles (nom, email, téléphone) — Choix de la date et de l'heure disponibles — Confirmation par email
PB12	En tant que visiteur, je veux **consulter le stock** afin de vérifier la disponibilité d'un produit.	Moyenne	Transparence	— Recherche d'un produit par nom — Affichage uniquement des produits disponibles
PB13	En tant que client, je veux **consulter mes informations et mes traitements** afin de suivre mon historique.	Haute	Suivi patient	— Affichage des informations personnelles enregistrées — Accès à l'historique des traitements prescrits
PB14	En tant qu'utilisateur, je veux **réinitialiser mon mot de passe en cas d'oubli** afin de pouvoir accéder de nouveau à mon compte.	Haute	Sécurité	— Envoi d'un code de réinitialisation — Mise à jour du mot de passe
PB15	En tant qu'utilisateur, je veux **m'authentifier avec mon email et mon mot de passe** afin d'accéder à l'application de manière sécurisée.	Haute	Sécurité	— L'utilisateur saisit son email et son mot de passe — Vérification avec le mot de passe chiffré (bcrypt) — Refus d'accès si les informations sont incorrectes

Tableau 6 – Product Backlog global de l'application - Suite

Voici l'organisation des fonctionnalités par sprint :

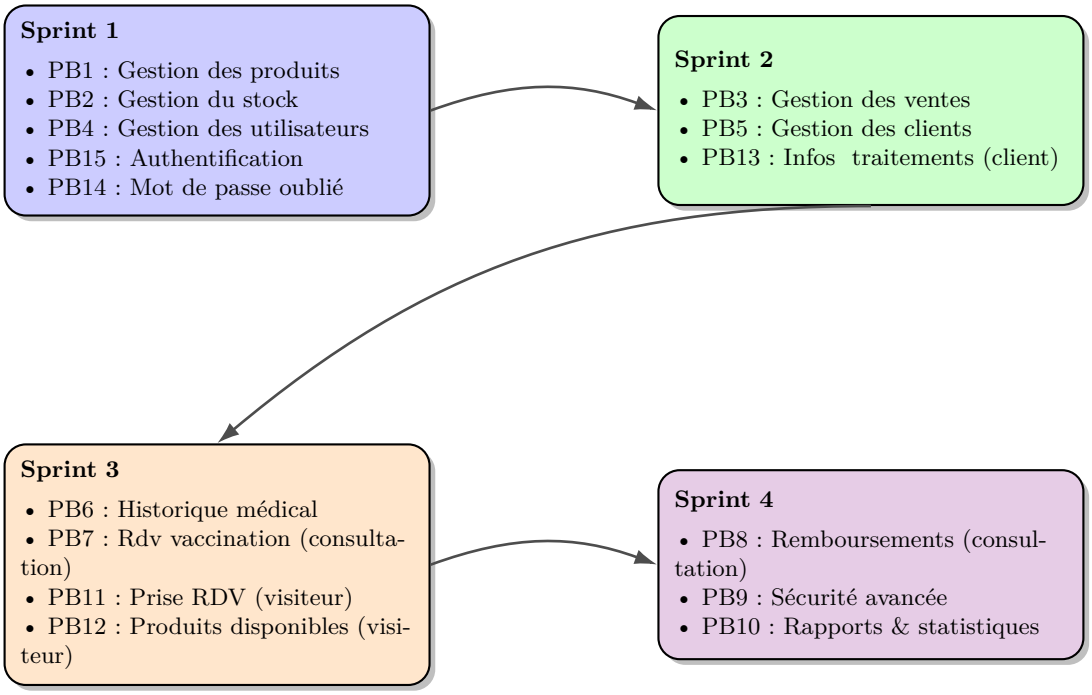

FIGURE 5 – Organisation des fonctionnalités par sprint — flux d'itération

5. Présentation de l'UML (Unified Modeling Language)

Le langage UML (Unified Modeling Language) est un langage de modélisation visuelle normalisé pour représenter les aspects structurels et comportementaux d'un système logiciel. Proposé par l'Object Management Group (OMG), il est devenu un standard largement adopté dans le domaine de l'ingénierie logicielle. [5]

UML n'est pas une notation fermée : elle est générique, extensible et configurable par l'utilisateur, ces concepteurs ont recherché avant tout la simplicité ; UML est intuitive, homogène et cohérente. [6]

Selon un article, UML offre une vérifiabilité et une traçabilité accrues tout au long du cycle de vie logiciel. Il améliore la communication entre les intervenants (analystes, développeurs, clients et testeurs) en fournissant un vocabulaire visuel commun. Par exemple, les diagrammes de cas d'utilisation permettent d'exprimer les besoins fonctionnels d'un système du point de vue des utilisateurs, tandis que les diagrammes de séquence ou d'activités aident à décrire le déroulement des traitements. [7] Ci-dessous le logo de l'UML :

FIGURE 6 – Logo de l'UML

6. Diagrammes UML utilisés dans le projet

6.1 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation (CU) permet de représenter les interactions entre les utilisateurs (acteurs) et le système. Il met en évidence les fonctionnalités principales offertes par le système du point de vue des utilisateurs finaux. Ce diagramme est particulièrement utile pour définir les exigences fonctionnelles et clarifier les attentes des parties prenantes. [6]

6.2 Diagramme de classes

Le diagramme de classes décrit la structure statique du système en représentant les classes, leurs attributs, méthodes ainsi que les relations (associations, héritages et agrégations) entre elles. Il sert à modéliser la conception objet du logiciel et constitue la base pour l'implémentation. Ce diagramme facilite la compréhension de l'architecture interne et des liens entre les différents composants du système. [6]

6.3 Diagramme de séquence

Le diagramme de séquence est un diagramme comportemental qui illustre l'enchaînement temporel des interactions entre objets lors d'un scénario particulier. Il montre comment les objets communiquent via des messages pour accomplir une fonctionnalité. Ce diagramme permet de visualiser le déroulement dynamique des opérations et d'identifier les collaborations nécessaires pour répondre aux besoins fonctionnels. [6]

Dans le cadre de notre étude, nous avons modélisé plusieurs fonctionnalités clés à l'aide de diagramme de séquence, notamment :

— L'authentification des utilisateurs.
— L'ajout d'un lot de médicament.
— La consultation de l'historique d'un client.
— La modification des informations d'un utilisateur.
— La réinitialisation du mot de passe.
— La gestion d'une vente.
— La prise d'un rendez-vous pour vaccination.

Ces diagrammes ont permis de décrire précisément les échanges entre les différents objets et acteurs impliqués, facilitant ainsi la compréhension dynamique du système.

7. Choix et justification des outils de modélisation

Pour la réalisation des diagrammes UML dans ce projet, deux outils principaux ont été utilisés : PlantUML et Draw.io.

PlantUML a été privilégié pour la modélisation des diagrammes de classes grâce à sa capacité à générer automatiquement des diagrammes à partir d'une syntaxe textuelle simple et efficace. Cet outil facilite la maintenance et la modification rapide des diagrammes, tout en assurant une bonne lisibilité et une standardisation conforme aux notations UML. De plus, PlantUML s'intègre aisément dans des environnements de développement et des plateformes collaboratives, ce qui améliore la traçabilité des versions et le travail en équipe.

FIGURE 7 – Logo de PlantUML

Pour les autre diagrammes tels que les diagrammes de cas d'utilisation et de séquence, Draw.io a été choisi. Cet outil en ligne, intuitif et riche en fonctionnalités, permet de créer facilement des diagrammes visuels avec une interface graphique conviviale. Draw.io offre également une grande flexibilité dans la personnalisation des éléments graphiques, ce qui permet de représenter précisément les spécificités fonctionnelles et comportementales du système modélisé.

FIGURE 8 – Logo de Draw.io

Le choix combiné de ces deux outils a permis de bénéficier à la fois de la rapidité et de la robustesse de PlantUML pour les diagrammes structurels complexes, et de la facilité d'utilisation de Draw.io pour les diagrammes interactifs et dynamiques. Cette approche mixte optimise la qualité des modèles tout en s'adaptant aux besoins variés du projet.

8. Conclusion

Ce chapitre a permis de définir de manière rigoureuse les besoins fonctionnels et non fonctionnels du système, tout en identifiant les différents acteurs impliqués dans son utilisation. Ces éléments constituent la base de la conception de l'application et assurent une compréhension commune entre les parties prenantes du projet.

L'adoption de la méthode agile **Scrum** a offert un cadre de développement itératif et évolutif, favorisant une meilleure réactivité face aux changements de besoins et garantissant une amélioration continue du produit au fil des sprints. Le découpage du projet en incréments successifs, chacun apportant une valeur ajoutée concrète, a permis d'assurer une progression structurée et contrôlée.

Enfin, la présentation du **Product Backlog** et la répartition des fonctionnalités par sprint ont fourni une vision claire de la planification du développement. Ces spécifications détaillées, combinées à une approche de conception basée sur le langage **UML**, offrent une modélisation précise du système et servent de fondement solide à la phase d'implémentation qui sera développée dans le chapitre suivant.

Chapitre 3 :
Étude des sprints 1,2,3 et 4

Chapitre 3 : Étude des sprints 1,2,3 et 4

1. Introduction

La conception d'une application de gestion pour pharmacie nécessite une approche rigoureuse et structurée afin de garantir la qualité, la maintenabilité et l'évolutivité du système. Pour atteindre ces objectifs, il est essentiel d'adopter des méthodes de modélisation éprouvées qui facilitent la compréhension, la communication et la documentation tout au long du cycle de développement.

Dans le cadre de ce projet, nous avons retenu une démarche combinant à la fois un langage de modélisation normalisé et une méthodologie de gestion de projet agile. Cette combinaison permet de structurer la conception tout en restant flexible face aux évolutions et aux besoins exprimés par les utilisateurs.

Ainsi, ce chapitre présente les bases théoriques et conceptuelles sur lesquelles repose notre travail, notamment les principes de modélisation, la méthodologie de développement adoptée, ainsi que les outils retenus pour la réalisation. Ces éléments constituent le socle nécessaire à la suite de notre démarche et préparent l'introduction des diagrammes et modèles qui seront exploités dans le projet.

2. Étude du premier sprint

2.1 Décomposition du premier sprint

Le premier sprint, d'une durée de **deux semaines**, a été consacré à **l'analyse des besoins** et au **cadrage fonctionnel** de l'application. L'objectif principal était de comprendre les attentes des différents acteurs, de définir les fonctionnalités essentielles et de poser les bases de la modélisation initiale.

Sprint Planning
- Rencontre avec les pharmaciens et la sous-direction informatique de la CNAS pour identifier les attentes principales.
- Élaboration du Product Backlog initial regroupant les fonctionnalités majeurs.
- Définition du Sprint Backlog 1

ID	Tâche associée	Responsable	Livrable attendu
PB1	Analyser les besoins liés à la **gestion des produits** (ajout, modification, suppression)	Équipe projet	Cahier des charges fonctionnel + maquette interface gestion produits
PB2	Définir les besoins pour la **gestion du stock** (alertes, suivi des entrées/sorties) et gestion des lots (Ajout, modification et suppression)	Équipe projet	Documentation des règles de gestion du stock + diagramme de cas d'utilisation + Diagramme de séquence (Ajout d'un lot)
PB4	Identifier les rôles et permissions pour la **gestion des utilisateurs**	Équipe projet	Liste des rôles (pharmacien, vendeur, client) + Diagramme de séquence (Modification d'un vendeur)
PB15	Définir le processus d'**authentification** (connexion sécurisée avec bcrypt)	Équipe projet	Maquette de l'écran de connexion + diagramme de séquence d'authentification
PB14	Spécifier le cas **mot de passe oublié**	Équipe projet	Scénario fonctionnel + maquette du formulaire de réinitialisation + Diagramme de séquence (Réinitialiser le mot de passe)

Tableau 7 – Sprint Backlog 1 — Analyse des besoins et cadrage fonctionnel

Daily Scrum

Réunions quotidiennes de 15 minutes entre les membres de l'équipe pour :
- Suivre l'avancement de la collecte des besoins.
- Répartir les tâches d'analyse (produits, stock, utilisateurs et authentification).
- Identifier rapidement les obstacles (incohérences dans les besoins exprimés).

Sprint Review

Présentation des premiers livrables :
- Maquettes d'interface utilisateur (login, gestion des produits et gestion des lots/stock)
- Diagrammes UML initiaux (Cas d'utilisation, diagramme de classes préliminaire et diagrammes de séquence)
- Recueil des retours de l'encadrante et ajustement des exigences.

Sprint Retrospective

Points positifs :
- Bonne interaction avec les pharmaciens et la CNAS
- Clarification des modules critiques

Points à améliorer :
- Formalisation des besoins parfois incomplète
- Manque de priorisation claire sur certaines fonctionnalités secondaires

Actions décidées :
- Mettre en place un suivi plus rigoureux des exigences fonctionnelles
- Préparer une documentation plus détaillée avant d'attaquer la conception technique **(sprint 2)**

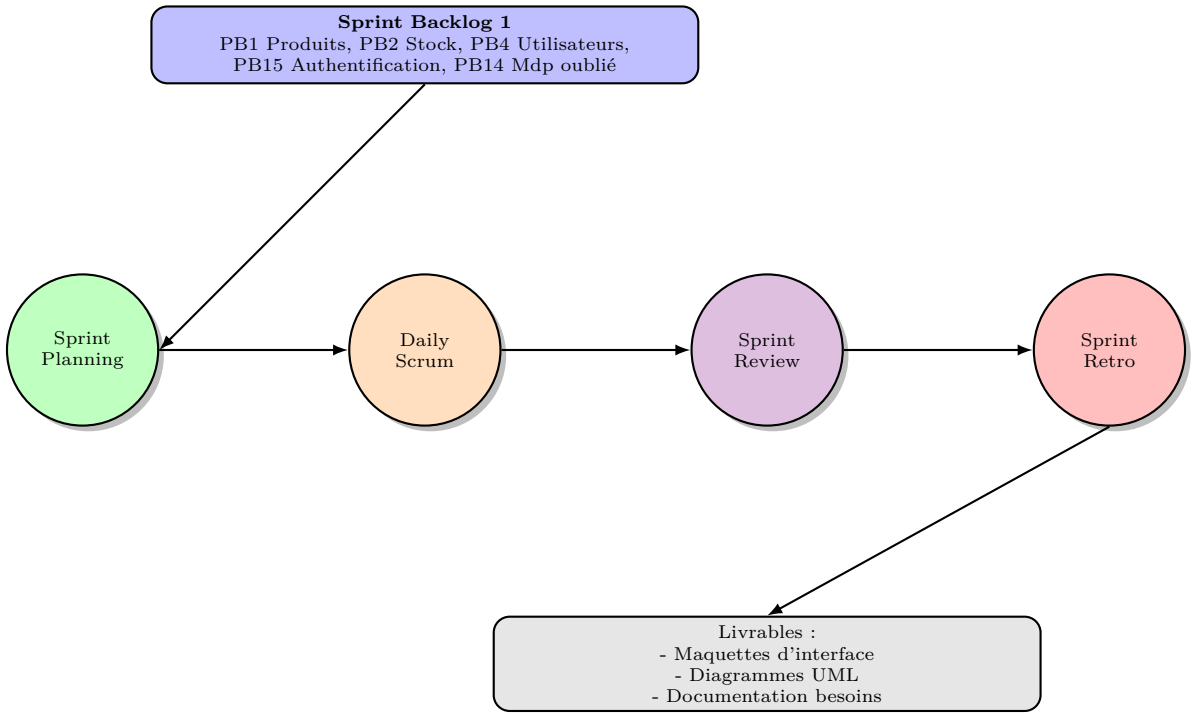

FIGURE 9 – Synthèse du Sprint 1

Chapitre 3 : Étude des sprints 1,2,3 et 4

2.2 Diagrammes de cas d'utilisation

La figure suivante montre les diagramme de cas d'utilisation globaux du premier sprint :

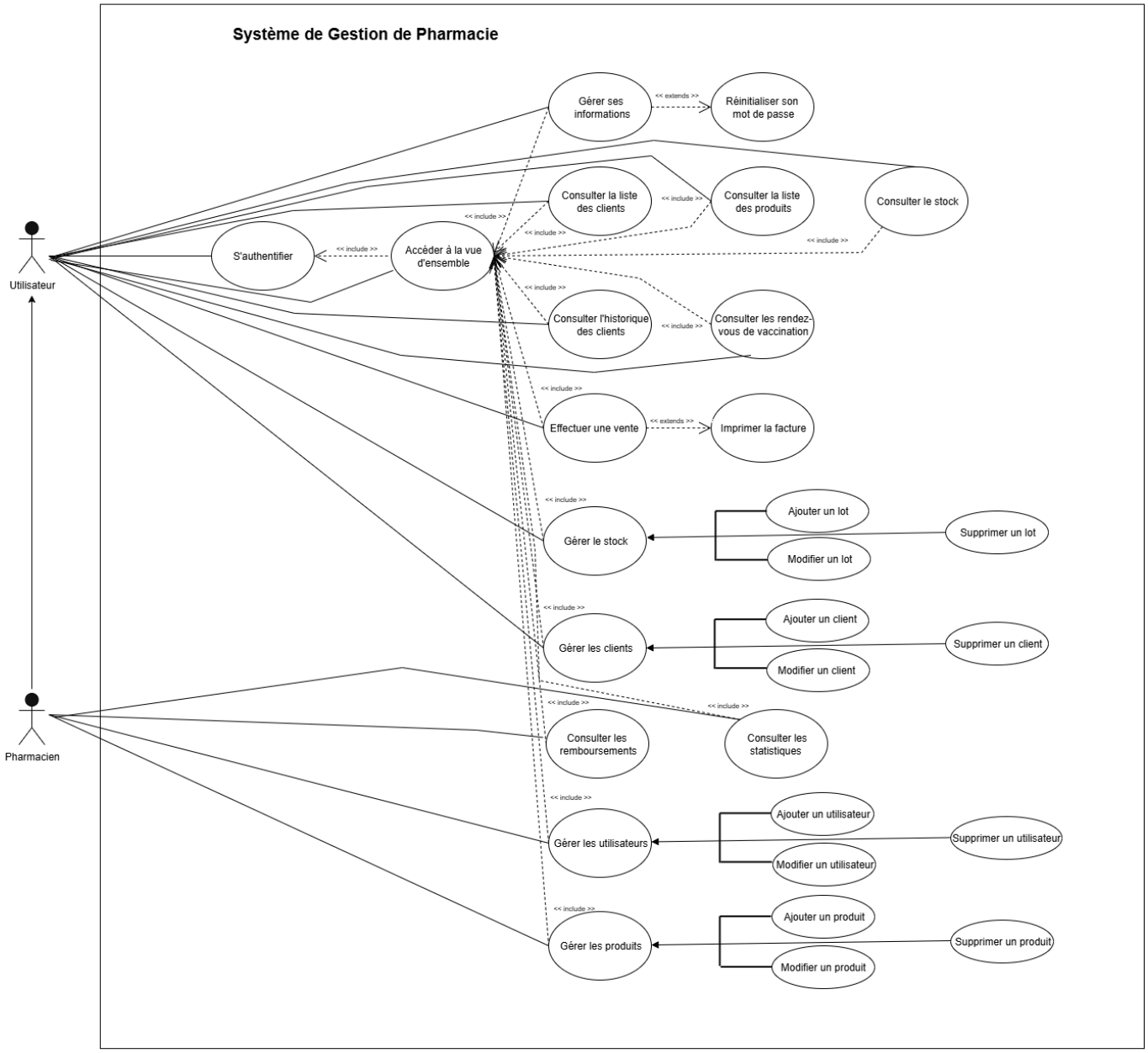

FIGURE 10 – Diagramme de cas d'utilisation n°1

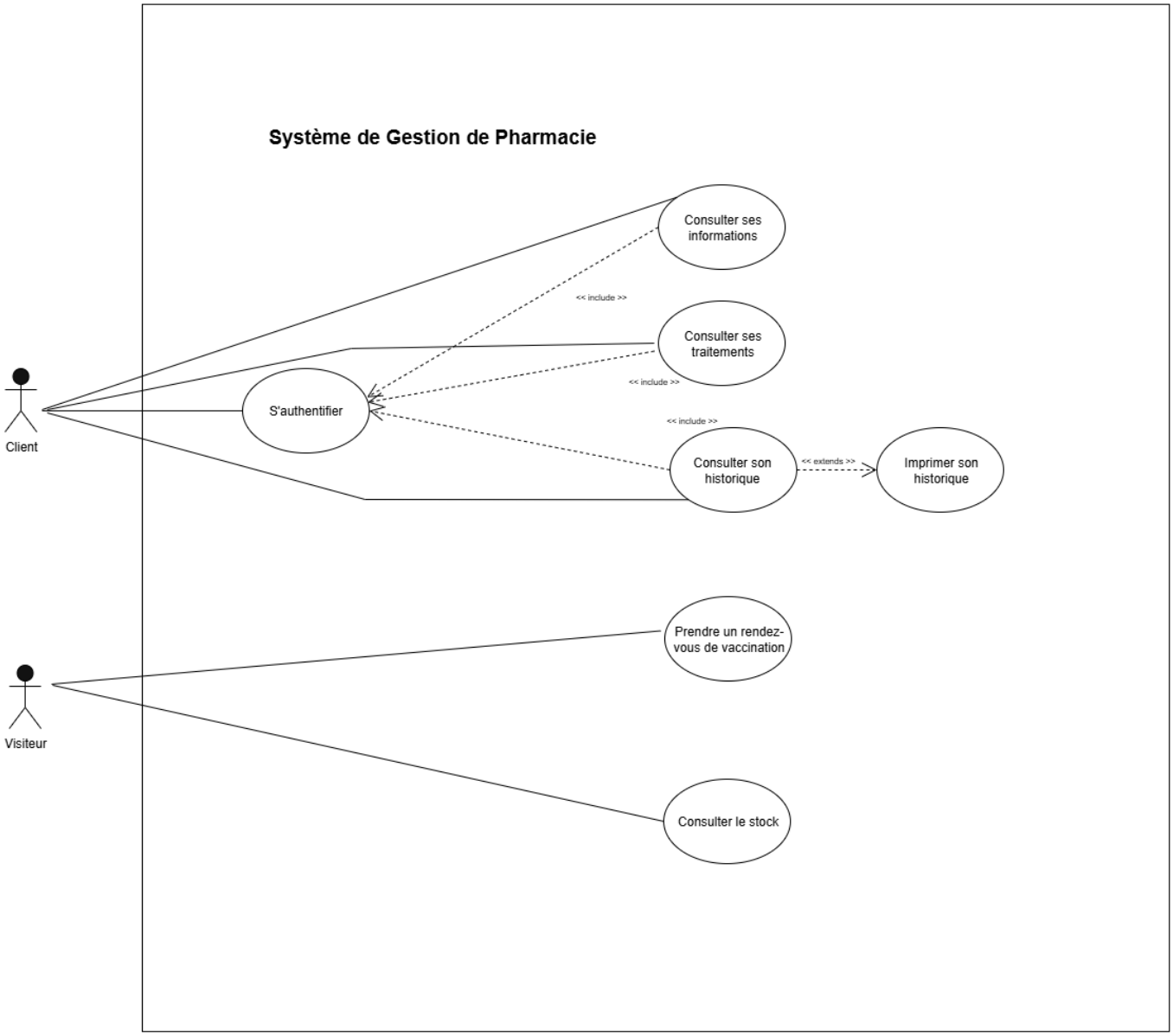

FIGURE 11 – Diagramme de cas d'utilisation n°2

2.3 Description textuelle des cas d'utilisation

Les tableaux ci-dessous montrent les descriptions textuelles des cas d'utilisation :

Cas d'utilisation	S'authentifier
Acteurs	Utilisateur (Pharmacien, vendeur ou client)
Préconditions	— L'utilisateur possède un compte valide (identifiant et mot de passe) — L'application est accessible
Déroulement normal	1. L'utilisateur accède à la page de connexion 2. Il saisit son identifiant et son mot de passe 3. Le système vérifie les informations saisies 4. Si elles sont valides, l'utilisateur est redirigé vers son interface selon son rôle 5. Le système affiche un message de bienvenue
Scénarios alternatifs	**Informations incorrectes** : — Le système affiche un message d'erreur : "Identifiant ou mot de passe incorrect" — L'utilisateur peut réessayer
Postconditions	L'utilisateur est connecté et peut accéder aux fonctionnalités liées à son rôle

Tableau 8 – Description textuelle du cas d'utilisation : S'authentifier

Le tableau ci-dessous montre la description textuelle du cas d'utilisation : Accéder à la vue d'ensemble :

Cas d'utilisation	Accéder à la vue d'ensemble
Acteurs	Utilisateur (Pharmacien ou vendeur)
Préconditions	— L'utilisateur est authentifié avec succès
Déroulement normal	1. Le système identifie le rôle de l'utilisateur (pharmacien, vendeur ou client) 2. Il redirige automatiquement vers l'interface correspondant à ce rôle 3. L'interface personnalisée s'affiche avec les fonctionnalités autorisées
Scénarios alternatifs	Aucun
Postconditions	L'utilisateur accède à une interface adaptée à son rôle avec les fonctionnalités disponibles

Tableau 9 – Description textuelle du cas d'utilisation : Accéder à la vue d'ensemble

Cas d'utilisation	Réinitialiser le mot de passe
Acteurs	Utilisateur (Pharmacien ou vendeur)
Préconditions	— L'utilisateur a oublié son mot de passe **ou** souhaite le modifier — L'utilisateur possède un compte valide dans le système — L'application est accessible
Déroulement normal	**Cas 1 : mot de passe oublié** — L'utilisateur clique sur "Mot de passe oublié" depuis la page de connexion — Il saisit son e-mail — Le système vérifie l'existence du compte — Un code de réinitialisation est envoyé par e-mail — Il saisit le code puis un nouveau mot de passe et le confirme — Le système enregistre le nouveau mot de passe et affiche une confirmation **Cas 2 : modification dans le profil** — L'utilisateur accède à son profil — Il sélectionne "Modifier mot de passe" dans les réglages — Il saisit son mot de passe actuel, puis le nouveau mot de passe — Le système vérifie l'ancien mot de passe — Si la vérification est réussie, le mot de passe est mis à jour — Un message de confirmation est affiché
Scénarios alternatifs	**Cas 1** : utilisateur non reconnu — Le système affiche : "Utilisateur introuvable" — L'utilisateur peut réessayer **Cas 2** : ancien mot de passe incorrect — Le système affiche : "Ancien mot de passe incorrect" — L'utilisateur peut réessayer
Postconditions	Le mot de passe de l'utilisateur est mis à jour avec succès. Il peut se connecter avec ses nouvelles informations.

Tableau 10 – Description textuelle du cas d'utilisation : Réinitialiser ou modifier le mot de passe

Cas d'utilisation	Consulter les rendez-vous de vaccination
Acteurs	Utilisateur (Pharmacien ou vendeur)
Préconditions	— L'utilisateur est connecté à l'application — Des rendez-vous existent dans le système
Déroulement normal	1. L'utilisateur accède à l'interface des rendez-vous de vaccination
Scénarios alternatifs	Aucun
Postconditions	La liste est affichée correctement

Tableau 11 – Description textuelle du cas d'utilisation : Consulter les rendez-vous de vaccination

Cas d'utilisation	Consulter l'historique des achats d'un client
Acteurs	Utilisateur (Pharmacien ou vendeur)
Préconditions	— L'utilisateur est connecté à l'application — Le client est enregistré dans le système
Déroulement normal	1. L'utilisateur accède à la section "Clients" ou "Historique" 2. Il recherche le client à l'aide de son nom 3. Le système affiche les informations du client 4. L'utilisateur sélectionne le client pour consulter son historique 5. Le système affiche la liste détaillée des achats effectués par ce client
Scénarios alternatifs	**Client introuvable** : — Le système affiche un message "Client non trouvé" — L'utilisateur peut relancer la recherche ou vérifier les informations saisies
Postconditions	L'utilisateur a consulté l'historique des achats du client et peut utiliser ces informations pour un nouveau traitement, une impression ou un suivi

Tableau 12 – Description textuelle du cas d'utilisation : Consulter l'historique des achats d'un client

Cas d'utilisation	Gérer ses informations
Acteurs	Utilisateur (Pharmacien ou vendeur)
Préconditions	— L'utilisateur est connecté à l'application
Déroulement normal	1. L'utilisateur accède à son espace personnel 2. Le système récupère et affiche ses informations 3. L'utilisateur peut modifier une de ses informations 4. Le système enregistre les modifications et met à jour la Bdd
Scénarios alternatifs	**Erreur de chargement des données ou données invalides** : — Le système affiche un message d'erreur.
Postconditions	Les données sont affichées et bien enregistrées si une modification a été effectuée

Tableau 13 – Description textuelle du cas d'utilisation : Gérer ses informations

Cas d'utilisation	Gérer les clients
Acteurs	Utilisateur (Pharmacien ou vendeur)
Préconditions	— L'utilisateur est connecté à l'application — L'utilisateur a les droits d'accès à la gestion des clients
Déroulement normal	1. L'utilisateur accède à la section "Gestion des clients" 2. Trois actions sont possibles : — **Ajouter un client** : l'utilisateur saisit les informations du client pour valider l'ajout — **Modifier un client** : l'utilisateur sélectionne un client existant, modifie les informations souhaitées, puis valide la modification — **Supprimer un client** : l'utilisateur sélectionne un client et confirme sa suppression 3. Le système effectue l'action demandée et affiche un message de confirmation ou d'erreur
Scénarios alternatifs	**Ajout/Modification invalide** : — Le système détecte un champ vide ou une erreur de format — Il affiche un message d'erreur et empêche l'enregistrement
Postconditions	Le client est ajouté, modifié ou supprimé du système selon l'action effectuée, avec mise à jour des données

Tableau 14 – Description textuelle du cas d'utilisation : Gérer les clients

Cas d'utilisation	Consulter la liste des clients
Acteurs	Utilisateur (vendeur ou pharmacien)
Préconditions	— L'utilisateur est connecté à l'application.
Déroulement normal	1. L'utilisateur accède à la section "Clients". 2. Le système affiche la liste des clients enregistrés dans le système. 3. L'utilisateur peut effectuer une recherche.
Scénarios alternatifs	Aucun
Postconditions	L'utilisateur visualise les informations sur les clients.

Tableau 15 – Description textuelle du cas d'utilisation : Consulter la liste des clients

Cas d'utilisation	Gérer le stock
Acteurs	Utilisateur (Pharmacien ou vendeur)
Préconditions	— L'utilisateur est connecté à l'application — L'utilisateur a les droits d'accès à la gestion des lots
Déroulement normal	1. L'utilisateur accède à la section "Gestion des lots" 2. Trois actions sont possibles : — **Ajouter un lot** : l'utilisateur saisit les informations d'un lot pour valider l'ajout — **Modifier un lot** : l'utilisateur sélectionne un lot existant, modifie les informations souhaitées, puis valide la modification — **Supprimer un lot** : l'utilisateur sélectionne un lot et confirme sa suppression 3. Le système effectue l'action demandée et affiche un message de confirmation ou d'erreur
Scénarios alternatifs	**Ajout/Modification invalide** : — Le système détecte un champ vide ou une erreur de format — Il affiche un message d'erreur et empêche l'enregistrement
Postconditions	Le lot est ajouté, modifié ou supprimé du système selon l'action effectuée, avec mise à jour des données

Tableau 16 – Description textuelle du cas d'utilisation : Gérer le stock

Cas d'utilisation	Consulter le stock
Acteurs	Utilisateur (vendeur ou pharmacien)
Préconditions	— L'utilisateur est connecté à l'application.
Déroulement normal	1. L'utilisateur accède à la section "Stock". 2. Le système affiche la liste des lots enregistrés dans le système. 3. L'utilisateur peut effectuer une recherche.
Scénarios alternatifs	Aucun
Postconditions	L'utilisateur visualise les informations sur les lots du stock.

Tableau 17 – Description textuelle du cas d'utilisation : Consulter le stock

Cas d'utilisation	Gérer les produits
Acteurs	Pharmacien
Préconditions	— Le pharmacien est connecté à l'application — Le pharmacien a les droits d'accès à la gestion des produits
Déroulement normal	1. Le pharmacien accède à la section "Gestion des produits" 2. Trois actions sont possibles : — **Ajouter un produit** : le pharmacien saisit les informations d'un produit pour valider l'ajout — **Modifier un produit** : le pharmacien sélectionne un produit existant, modifie les informations souhaitées, puis valide la modification — **Supprimer un produit** : le pharmacien sélectionne un produit et confirme sa suppression 3. Le système effectue l'action demandée et affiche un message de confirmation ou d'erreur
Scénarios alternatifs	**Ajout/Modification invalide** : — Le système détecte un champ vide ou une erreur de format — Il affiche un message d'erreur et empêche l'enregistrement
Postconditions	Le produit est ajouté, modifié ou supprimé du système selon l'action effectuée, avec mise à jour des données

Tableau 18 – Description textuelle du cas d'utilisation : Gérer les produits

Cas d'utilisation	Consulter la liste des produits
Acteurs	Utilisateur (vendeur ou pharmacien)
Préconditions	— L'utilisateur est connecté à l'application.
Déroulement normal	1. L'utilisateur accède à la section "Produits". 2. Le système affiche la liste des produits enregistrés dans le système. 3. L'utilisateur peut effectuer une recherche.
Scénarios alternatifs	Aucun
Postconditions	L'utilisateur visualise les informations sur les produits.

Tableau 19 – Description textuelle du cas d'utilisation : Consulter la liste des produits

Cas d'utilisation	Gérer les utilisateurs
Acteurs	Pharmacien
Préconditions	— Le pharmacien est connecté à l'application — Le pharmacien a les droits d'accès à la gestion des utilisateurs
Déroulement normal	1. Le pharmacien accède à la section "Gestion des utilisateurs" 2. Trois actions sont possibles : — **Ajouter un utilisateur** : le pharmacien saisit les informations d'un utilisateur pour valider l'ajout — **Modifier un utilisateur** : le pharmacien sélectionne un utilisateur existant, modifie les informations souhaitées, puis valide la modification — **Supprimer un utilisateur** : le pharmacien sélectionne un utilisateur et confirme sa suppression 3. Le système effectue l'action demandée et affiche un message de confirmation ou d'erreur
Scénarios alternatifs	**Ajout/Modification invalide** : — Le système détecte un champ vide ou une erreur de format — Il affiche un message d'erreur et empêche l'enregistrement
Postconditions	L'utilisateur est ajouté, modifié ou supprimé du système selon l'action effectuée, avec mise à jour des données

Tableau 20 – Description textuelle du cas d'utilisation : Gérer les utilisateurs

Cas d'utilisation	Consulter ses informations
Acteurs	Client
Préconditions	— Le client est connecté à l'application
Déroulement normal	1. Le client accède à son espace personnel 2. Le système récupère et affiche ses informations
Scénarios alternatifs	**Erreur de chargement des données** : — Le système affiche un message d'erreur.
Postconditions	La vente est enregistrée avec tous les produits ajoutés et le stock est mis à jour.

Tableau 21 – Description textuelle du cas d'utilisation : Consulter ses informations

Cas d'utilisation	Effectuer une vente
Acteurs	Utilisateur (Vendeur ou pharmacien)
Préconditions	— L'utilisateur est connecté à l'application — Le stock est disponible et à jour
Déroulement normal	1. L'utilisateur accède à l'interface des ventes 2. Il remplit un formulaire avec les informations nécessaires 3. Il ajoute le produit au panier 4. Le système ajoute le produit à la liste temporaire de la vente 5. L'utilisateur peut répéter les étapes 2 à 4 pour plusieurs produits 6. Une fois la liste complète, il clique sur valider 7. Le système calcule le total, enregistre la vente, met à jour le stock et affiche une confirmation.
Scénarios alternatifs	**Produit non disponible ou quantité insuffisante** : — Le système affiche un message d'erreur. — L'utilisateur peut corriger les informations ou supprimer le produit de la liste.
Postconditions	La vente est enregistrée avec tous les produits ajoutés et le stock est mis à jour.

Tableau 22 – Description textuelle du cas d'utilisation : Effectuer une vente

Cas d'utilisation	Consulter les remboursements
Acteurs	Pharmacien
Préconditions	— Le pharmacien est connecté à l'application — Des informations de remboursements existent dans le système
Déroulement normal	1. L'utilisateur accède à l'interface des remboursements
Scénarios alternatifs	Aucun
Postconditions	La liste des remboursements est affichée correctement

Tableau 23 – Description textuelle du cas d'utilisation : Consulter les remboursements

Cas d'utilisation	Consulter les statistiques
Acteurs	Pharmacien
Préconditions	— Le pharmacien est connecté à l'application — Des statistiques existent dans le système
Déroulement normal	1. Le pharmacien accède à l'interface des statistiques (Vue d'ensemble)
Scénarios alternatifs	Aucun
Postconditions	Les statistiques sont affichées correctement

Tableau 24 – Description textuelle du cas d'utilisation : Consulter les statistiques

Cas d'utilisation	Imprimer la facture
Acteurs	Utilisateur (Pharmacien ou vendeur)
Préconditions	— L'utilisateur est connecté à l'application — Une vente a été effectuée et enregistrée
Déroulement normal	1. L'utilisateur accède à l'interface des ventes 2. L'utilisateur effectue une vente et choisit l'option - Imprimer 3. Le système génère la facture et envoie la facture vers l'imprimante configurée
Scénarios alternatifs	**Erreur d'impression** : — Le système affiche un message d'erreur. — L'utilisateur peut sauvegarder la facture en format PDF pour une impression ultérieure
Postconditions	La facture est imprimée et remise au client

Tableau 25 – Description textuelle du cas d'utilisation : Imprimer la facture

Cas d'utilisation	Consulter ses traitements ou son historique
Acteurs	Client
Préconditions	— Le client est connecté à l'application — Des achats ont déjà été enregistrés dans le système
Déroulement normal	1. Le client accède à son profil 2. Le système récupère et affiche ses données
Scénarios alternatifs	**Erreur de chargement des données** : — Le système affiche un message d'erreur.
Postconditions	Le client peut visualiser son historique ou ses traitements

Tableau 26 – Description textuelle du cas d'utilisation : Consulter ses traitements ou son historique

Cas d'utilisation	Imprimer son historique
Acteurs	Client
Préconditions	— Le client est connecté à l'application — Des achats ont déjà été enregistrés dans le système
Déroulement normal	1. Le client accède à son profil et choisit l'option d'impression 2. Le système envoie l'historique vers l'imprimante configurée
Scénarios alternatifs	**Erreur d'impression** : — Le système affiche un message d'erreur.
Postconditions	L'historique du client imprimé correctement

Tableau 27 – Description textuelle du cas d'utilisation : Imprimer son historique

Cas d'utilisation	Consulter la liste des produits disponibles
Acteurs	Visiteur
Préconditions	— Le visiteur accède à l'application et le stock mis à jour
Déroulement normal	1. Le visiteur accède à l'interface de consultation du stock 2. Le système affiche la liste des produits disponibles 3. Le visiteur peut rechercher un produit
Scénarios alternatifs	**Stock vide ou produit non trouvé**
Postconditions	Le visiteur a consulté l'état du stock

Tableau 28 – Description textuelle du cas d'utilisation : Consulter le stock

Cas d'utilisation	Prendre un rendez-vous de vaccination
Acteurs	Visiteur
Préconditions	— Le visiteur accède à l'application — Le système dispose de créneaux de vaccinations et vaccins disponibles
Déroulement normal	1. Le visiteur accède à l'interface de prise de rendez-vous 2. Il sélectionne le vaccin désiré 3. Il saisit ses informations personnelles 4. Il sélectionne un créneau disponible pour la vaccination 5. Le système enregistre le rendez-vous et envoie un e-mail au visiteur pour confirmer le rendez-vous
Scénarios alternatifs	**Aucun créneau disponible ou informations invalides** — Le système affiche un message d'erreur et demande de corriger les informations du visiteur. — Le système affiche un message d'erreur indiquant qu'aucune plage horaire n'est disponible
Postconditions	Le visiteur a la confirmation de son rendez-vous de vaccination et enregistré dans le système

Tableau 29 – Description textuelle du cas d'utilisation : Consulter le stock

2.4 Diagramme de séquence : Authentification

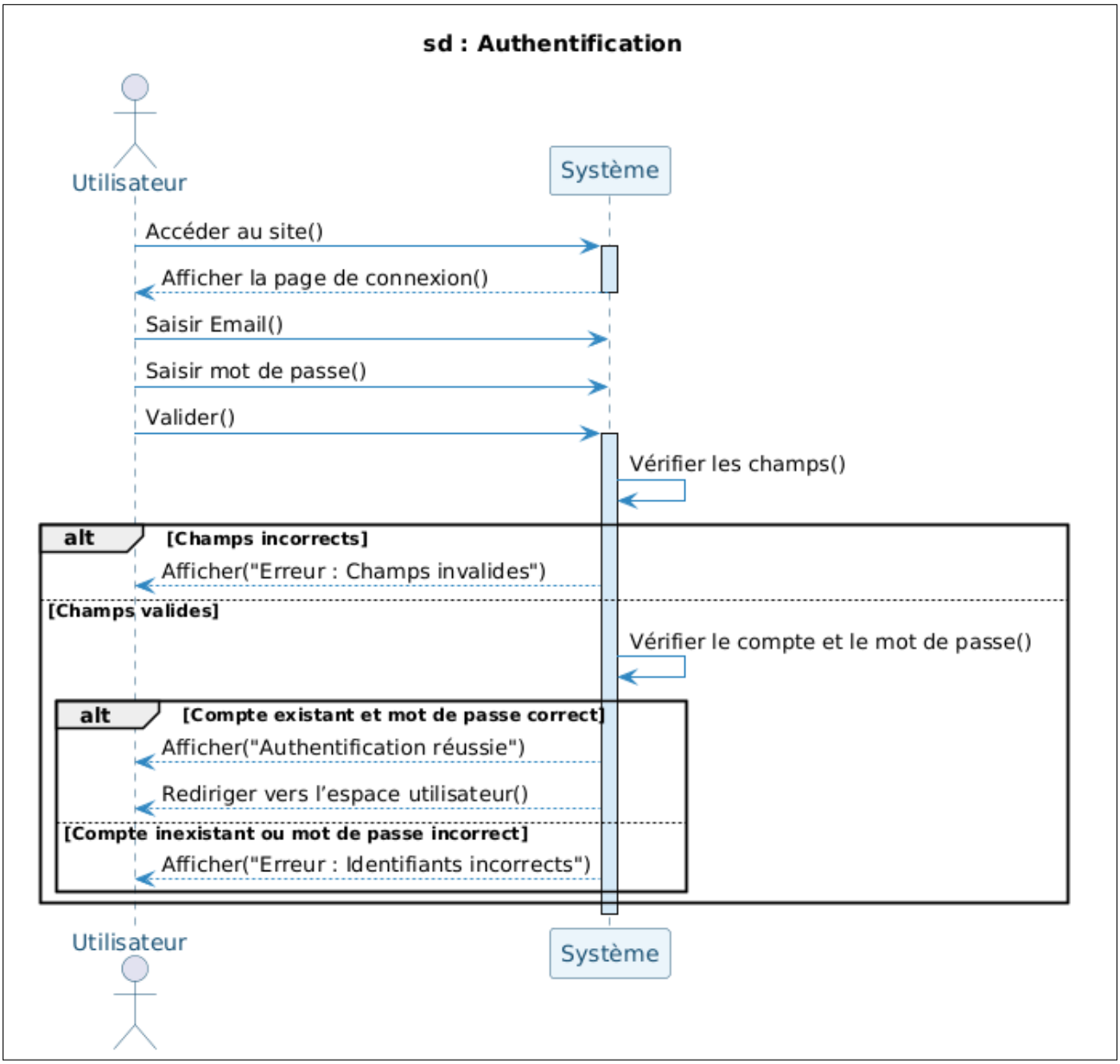

Figure 12 – Diagramme de séquence : Authentification

2.5 Diagramme de séquence : Ajout d'un lot

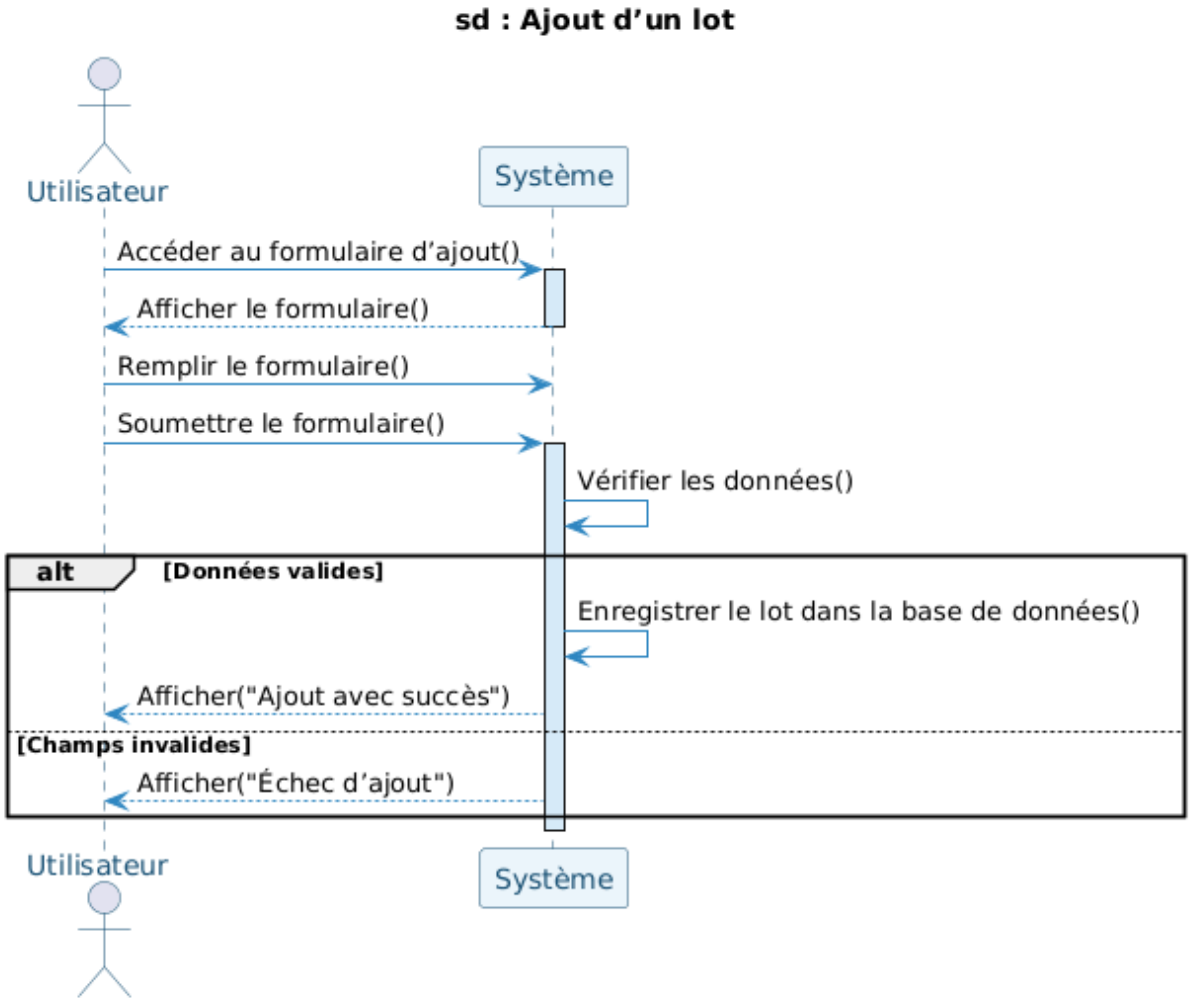

FIGURE 13 – Diagramme de séquence : Ajout d'un lot

2.6 Diagramme de séquence : Modification d'un vendeur

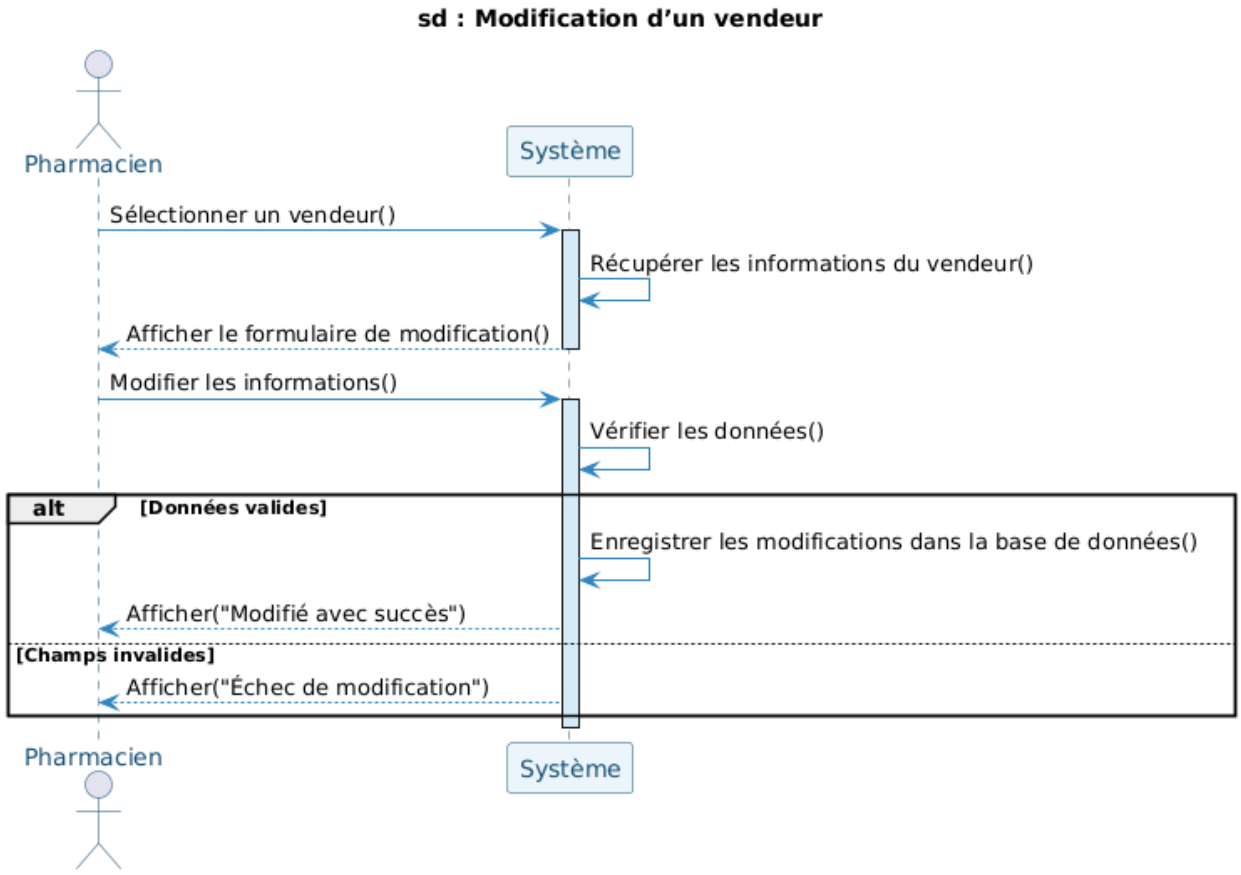

FIGURE 14 – Diagramme de séquence : Modification des informations d'un vendeur

2.7 Diagramme de séquence : Réinitialisation du mot de passe

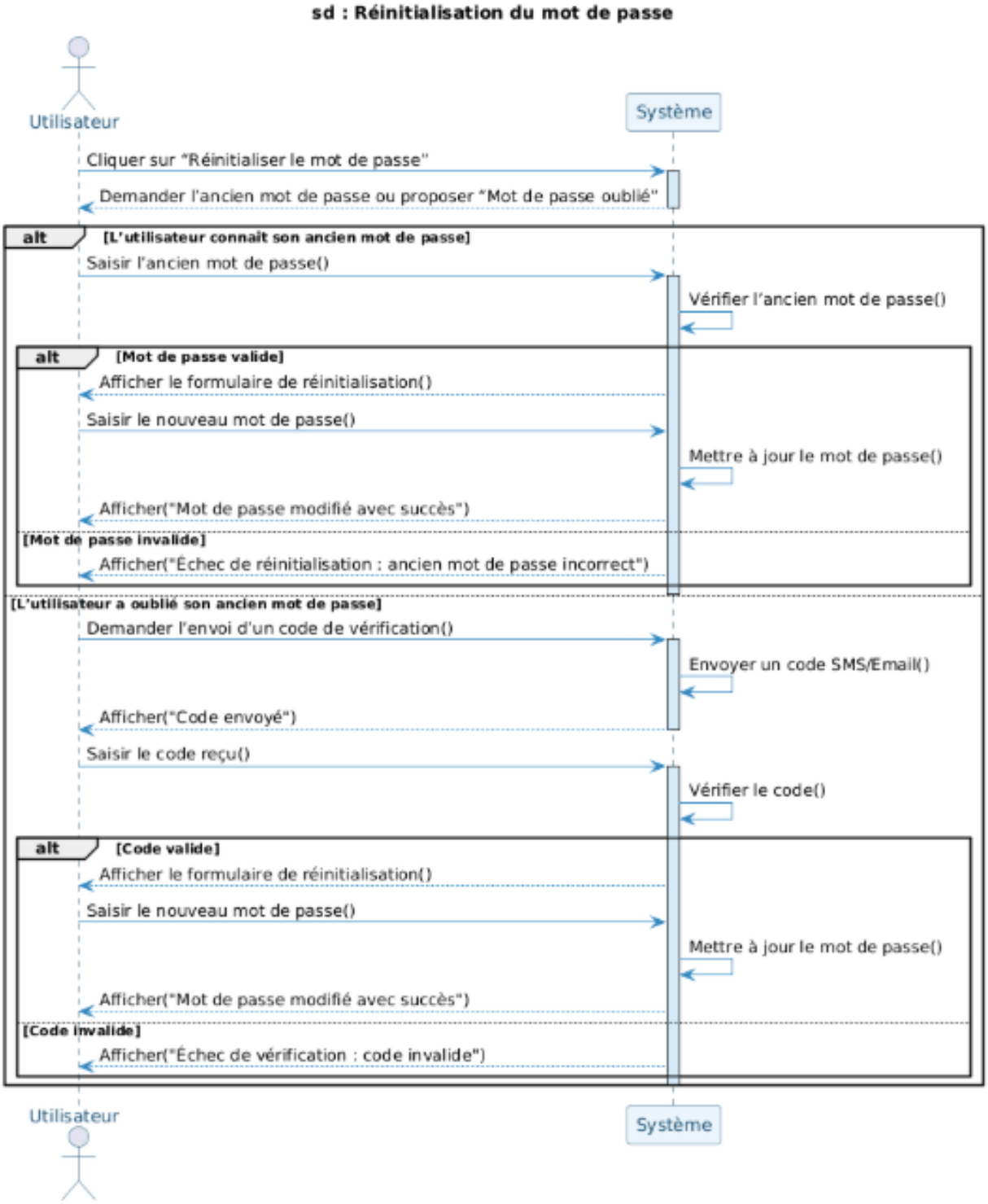

FIGURE 15 – Diagramme de séquence : Réinitialisation du mot de passe

Chapitre 3 : Étude des sprints 1,2,3 et 4

3. Étude du deuxième sprint

3.1 Décomposition du deuxième sprint

Le deuxième sprint, d'une durée de **trois semaines**, a été consacré à la **conception détaillée** et à la **modélisation** de l'application. L'objectif principal était de traduire les besoins identifiés lors du premier sprint en modèles concrets, en précisant la structure de la base de données, en affinant les diagrammes UML et en assurant la cohérence des différents modules prioritaires (gestion des ventes, gestion des clients et consultation des informations du client).

Sprint Planning

— Priorisation des modules critiques :

1. Gestion des ventes
2. Gestion des clients
3. Consultation des informations et traitements d'un client

— Définition des outils de développement :

1. **Front-end :** HTML, CSS, Javascript et Bootstrap
2. **Back-end :** Node.js
3. **Base de données :** MongoDB

— Planification des tâches de conception UML complète et modélisation de la base de données.
— Définition du Sprint Backlog 2

ID	Tâche associée	Responsable	Livrable attendu
PB3	Concevoir le module de **gestion des ventes** (facturation, application des remboursements, les alertes)	Binôme	Diagramme de séquence "enregistrer une vente" + modèle de données associé
PB5	Concevoir le module de **gestion des clients** (fiches, cartes CNAS/CASNOS)	Binôme	modèle conceptuel client
PB13	Définir le processus de **consultation des infos et traitements (client)**	Binôme	schéma de données liés
–	Conception globale du système	Binôme	Diagramme de classes détaillé + modèle conceptuel, logique et physique de la BD (MongoDB)

Tableau 30 – Sprint Backlog 2 — Conception détaillée et modélisation

Daily Scrum

Suivi quotidien de la progression sur :
— Diagrammes UML : cas d'utilisation, classes et séquence.
— Modèle conceptuel, logique et physique de la base de données.
Discussions régulières sur la structure des collections MongoDB et la cohérence entre modules.

Sprint Review

Présentation des livrables intermédiaires :
— Diagrammes UML détaillés (cas d'utilisation, classes et séquence).
— Modèle conceptuel, logique et physique de la base de données.
Retour de l'encadrante, notamment sur la gestion des remboursements (PB8, prévu Sprint 4).

Sprint Retrospective

Points positifs :
- Conception UML avancée et validée.
- Bonne cohérence entre les modules choisis.

Difficultés :
- Risques de surcharge d'interface utilisateur.
- Questions de sécurité des données sensibles.

Actions décidées :
- Cloisonnement strict des rôles (pharmacien, vendeur et client).
- Application de bonnes pratiques de conception et sécurité.

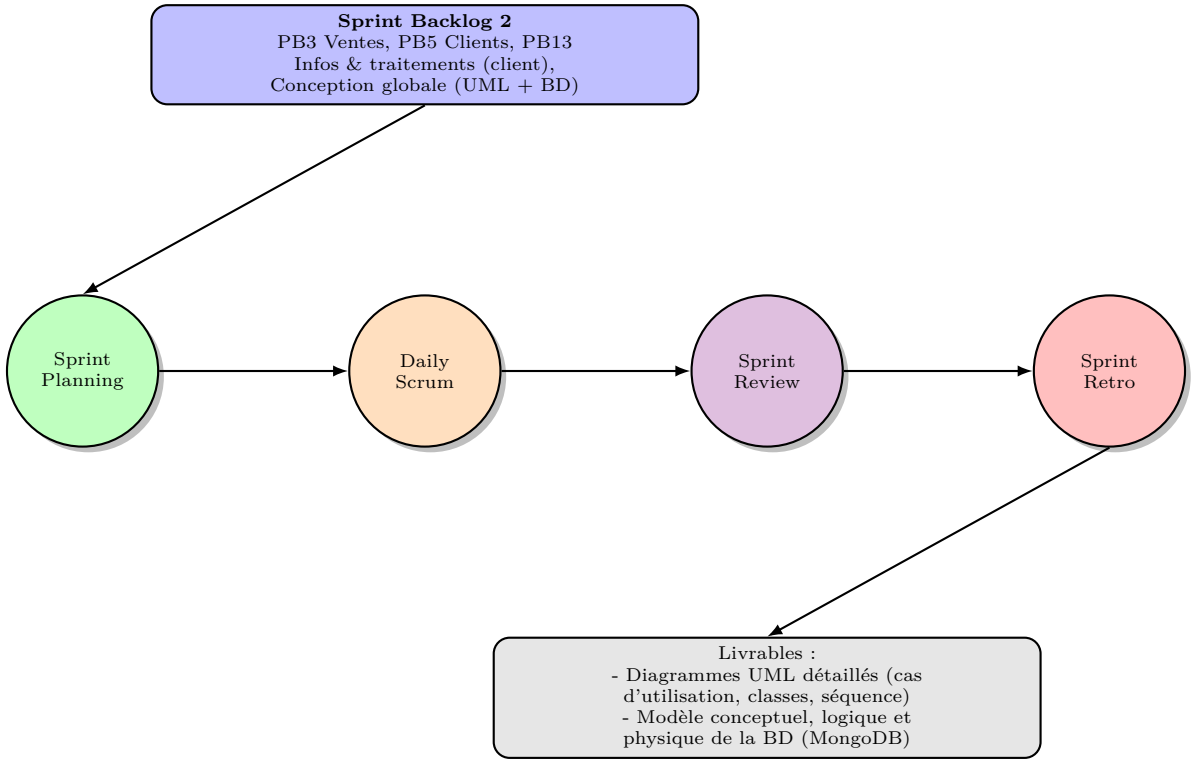

FIGURE 16 – Synthèse du Sprint 2

3.2 Diagramme de séquence : Gestion d'une vente

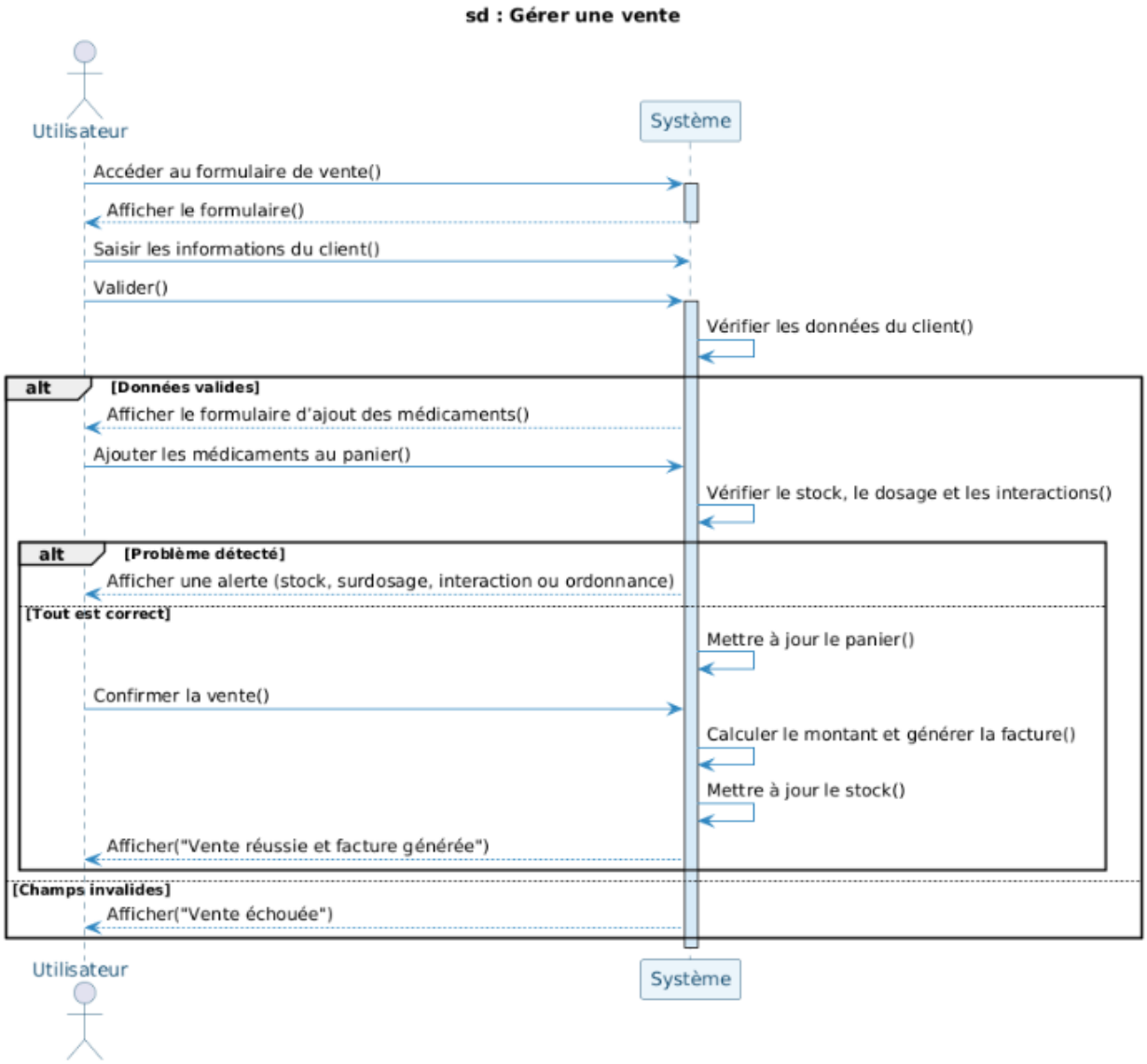

FIGURE 17 – Diagramme de séquence : Gestion d'une vente

Chapitre 3 : Étude des sprints 1,2,3 et 4

3.3 Diagramme de classes

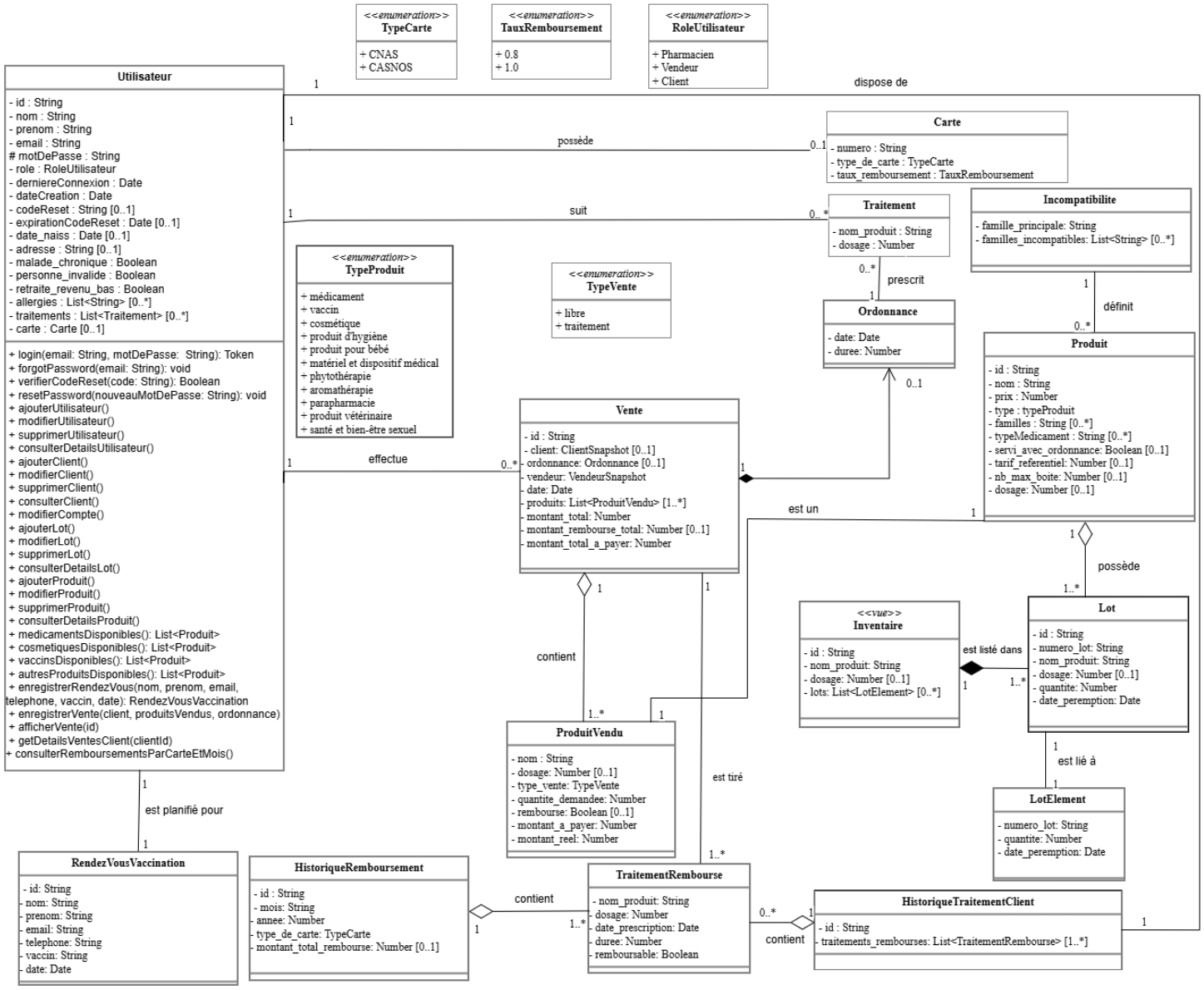

FIGURE 18 – Diagramme de classes

3.4 Structure de la base de données

3.4.1 Règles de passage

- **Transformation des classes :**
  - Chaque classe principale du modèle conceptuel devient une **collection MongoDB**.
  - Chaque attribut devient un **champ** dans le document, avec un type correspondant (`String`, `Number`, `Boolean`, `Date`, etc.).
  - Une clé primaire est représentée par `_id`, générée automatiquement sous forme d'`ObjectId` ou bien définie manuellement si un identifiant métier est nécessaire.
  - Les méthodes définies dans le modèle conceptuel ne sont pas traduites dans la base de données. Elles sont gérées dans la couche applicative.
  - Exemple : l'entité `Utilisateur` devient une collection où chaque utilisateur est stocké comme un document JSON contenant ses informations personnelles et administratives.

- **Sous-documents :**
  - Les entités dépendantes sont intégrées comme **sous-documents** dans la collection parente. Cela permet de conserver une relation d'imbrication naturelle.
  - Exemple : `Carte` et `Traitement` sont intégrés dans `Utilisateur`. Chaque utilisateur possède donc un objet `carte` et un tableau de `traitements`.
  - Exemple : `LotElement` est intégré dans `Inventaire`, ce qui permet de suivre la liste des lots disponibles pour un produit donné.
  - Exemple : `ProduitVendu`, `ClientSnapshot`, `VendeurSnapshot`, et `Ordonnance` sont intégrés directement dans `Vente`, afin que chaque document de vente contienne toutes les informations nécessaires sans dépendance externe.

- **Listes et collections :**
  - Les attributs multivalués sont représentés par des **tableaux JSON**. Chaque élément du tableau correspond à une valeur ou un objet.
  - Exemple : `allergies` est stocké comme un tableau de chaînes de caractères dans `Utilisateur`.
  - Exemple : `familles` et `typesMedicament` sont des tableaux de chaînes associés à un `Produit`.
  - Exemple : `lots` dans `Inventaire` contient plusieurs sous-documents représentant chaque lot.
  - Exemple : `familles_incompatibles` dans `Incompatibilite` liste les familles qui ne peuvent pas être associées.
  - Exemple : `produits` dans `Vente` regroupe tous les produits vendus dans une transaction.
  - Exemple : `traitements_rembourses` dans `HistoriqueTraitementClient` contient la liste des traitements déjà remboursés.

- **Collections principales :**
  - Les entités indépendantes sont modélisées comme des collections séparées. Elles possèdent une existence propre et peuvent être interrogées directement.
  - Exemple : `Utilisateur`, `Produit`, `Lot`, `Inventaire`, `Incompatibilite`, `Vente`, `HistoriqueRemboursement`, `HistoriqueTraitementClient`, et `RendezVousVaccination`.
  - Cette séparation garantit la flexibilité et permet une meilleure organisation des données tout en exploitant les forces du modèle documentaire de MongoDB.

- **Avantages de cette modélisation :**
  - Réduction des jointures complexes grâce à l'intégration directe des sous-documents.
  - Accès rapide aux informations complètes lors des opérations critiques (exemple : lecture d'une vente avec ses produits et son ordonnance).
  - Possibilité de gérer efficacement les données multivaluées grâce aux tableaux JSON.
  - Adaptation naturelle aux besoins d'une application de gestion pharmaceutique, où certaines informations (ordonnance, carte, traitements) sont fortement liées à leur entité parente.

3.4.2 Schéma de la base de données

- **Carte (sous-document de Utilisateur)** : (numero, type_de_carte, taux_remboursement)
- **Traitement (sous-document de Utilisateur)** : (nom_produit, dosage)
- **Utilisateur** : (id, nom, prenom, email, motDePasse, role, dernierConnexion, dateCreation, codeReset, expirationCodeReset, date_naiss, adresse, malade_chronique, personne_invalide, retraite_revenu_bas, allergies [liste de String], traitements [liste de Traitement], carte)
- **Produit** : (id, nom, prix, type, familles [liste de String], typesMedicament [liste de String], servi_avec_ordonnance, tarif_referentiel, nb_max_boite, dosage)
- **Lot** : (id, numero_lot, nom_produit, dosage, quantite, date_peremption)
- **LotElement (sous-document de Inventaire)** : (numero_lot, quantite, date_peremption)
- **Inventaire** : (id, nom_produit, dosage, lots [liste de LotElement])
- **Incompatibilite** : (famille_principale, familles_incompatibles [liste de String])
- **ProduitVendu (sous-document de Vente)** : (nom, dosage, type_vente, quantite_demandee, rembourse, montant_a_payer, montant_reel)
- **ClientSnapshot (sous-document de Vente)** : (nom, prenom, type_de_carte, numero)
- **VendeurSnapshot (sous-document de Vente)** : (nom, prenom)
- **Ordonnance (sous-document de Vente)** : (date, duree)
- **Vente** : (id, client (ClientSnapshot), ordonnance (Ordonnance), vendeur (VendeurSnapshot), date, produits [liste de ProduitVendu], montant_total, montant_rembourse_total, montant_total_a_payer)
- **HistoriqueRemboursement** : (id, mois, annee, type_de_carte, montant_total_rembourse)
- **TraitementRembourse (sous-document de HistoriqueTraitementClient)** : (nom_produit, dosage, date_prescription, duree, remboursable)
- **HistoriqueTraitementClient** : (id, traitements_rembourses [liste de TraitementRembourse], idUtilisateur)
- **RendezVousVaccination** : (id, nom, prenom, email, telephone, vaccin, date)

3.4.3 Dictionnaire de données

Table	Attribut	Type	Taille	Description
Carte	numero	String	8	Numéro unique de la carte
	type_de_carte	TypeCarte	—	Type de carte (CNAS ou CASNOS)
	taux_remboursement	TauxRemboursement	—	Taux de remboursement associé à la carte
Traitement	nom_produit	String	—	Nom du traitement habituel du client
	dosage	Number	—	Dosage du traitement
Utilisateur	id	String	—	Clé primaire générée automatiquement
	nom	String	—	Nom de l'utilisateur
	prenom	String	—	Prénom de l'utilisateur
	email	String	—	Adresse email
	motDePasse	String	—	Mot de passe hashé
	role	RoleUtilisateur	—	Rôle de l'utilisateur
	derniereConnexion	Date	—	Date de dernière connexion
	dateCreation	Date	—	Date de création du compte
	codeReset	String	6	Code de réinitialisation (optionnel)
	expirationCodeReset	Date	—	Expiration du codeReset (optionnel)
	date_naiss	Date	—	Date de naissance
	adresse	String	—	Adresse de l'utilisateur
	malade_chronique	Boolean	1	Indique si le client a une maladie chronique
	personne_invalide	Boolean	1	Indique si le client est invalide
	retraite_revenu_bas	Boolean	1	Indique si le client retraité a un faible revenu
	allergies	List<String>	—	Liste des allergies (optionnel)
	traitements	List<Traitement>	—	Liste des traitements prescrits
	carte	Carte	—	Carte associée (uniquement si client)
Produit	id	String	—	Clé primaire générée automatiquement
	nom	String	—	Nom du produit
	prix	Number	—	Prix unitaire
	type	TypeProduit	—	Type du produit
	familles	List<String>	—	Familles du produit (si médicament)
	typesMedicament	List<String>	—	Types de médicament (si médicament)
	servi_avec_ordonnance	Boolean	1	Produit nécessitant une ordonnance (si médicament)
	tarif_referentiel	Number	—	Tarif référentiel (si médicament)
	nb_max_boite	Number	—	Nombre max de boîtes (si médicament)
	dosage	Number	—	Dosage (si médicament)
LotElement	numero_lot	String	—	Numéro du lot
	quantite	Number	—	Quantité contenue
	date_peremption	Date	—	Date de péremption

Tableau 31 – Dictionnaire de données – Tables 1

Table	Attribut	Type	Taille	Description
Lot	id	String	—	Clé primaire générée automatiquement
	numero_lot	String	—	Numéro du lot
	nom_produit	String	—	Nom du produit
	dosage	Number	—	Dosage (si médicament)
	quantite	Number	—	Quantité contenue dans le lot
	date_peremption	Date	—	Date de péremption du lot
Inventaire	id	String	—	Clé primaire générée automatiquement
	nom_produit	String	—	Nom du produit
	dosage	Number	—	Dosage (si médicament)
	lots	List<LotElement>	—	Liste des lots associés
Incompatibilite	famille_principale	String	—	Famille principale (clé primaire)
	familles_incompatibles	List<String>	—	Familles incompatibles
ProduitVendu	nom	String	—	Nom du produit vendu
	dosage	Number	—	Dosage (si médicament)
	type_vente	TypeVente	—	Type de vente
	quantite_demandee	Number	—	Quantité demandée
	rembourse	Boolean	—	Indique si remboursé
	montant_a_payer	Number	—	Montant à payer
	montant_reel	Number	—	Montant réel payé
ClientSnapshot	id	String	—	Clé primaire générée automatiquement
	nom	String	—	Nom du client
	prenom	String	—	Prénom du client
	type_de_carte	TypeCarte	—	Type de carte du client
	numero	String	8	Numéro de la carte
VendeurSnapshot	id	String	—	Clé primaire générée automatiquement
	nom	String	—	Nom du vendeur
	prenom	String	—	Prénom du vendeur
Vente	id	String	—	Clé primaire générée automatiquement
	client	ClientSnapshot	—	Snapshot du client
	ordonnance	Ordonnance	—	Ordonnance associée
	vendeur	VendeurSnapshot	—	Snapshot du vendeur
	date	Date	—	Date de la vente
	produits	List<ProduitVendu>	—	Produits vendus
	montant_total	Number	—	Montant total
	montant_rembourse_total	Number	—	Montant remboursé
	montant_total_a_payer	Number	—	Montant à payer

Tableau 32 – Dictionnaire de données – Tables 2

Table	Attribut	Type	Taille	Description
H-Remboursement	id	String	—	Clé primaire générée automatiquement
	mois	String	—	Mois concerné
	annee	Number	—	Année concernée
	type_de_carte	TypeCarte	—	Type de carte associée
	montant_total_rembourse	Number	—	Montant total remboursé
Ordonnance	date	Date	—	Date de prescription
	duree	Number	—	Durée de l'ordonnance
T-Rembourse	nom_produit	String	—	Nom du produit remboursé
	dosage	Number	—	Dosage du produit
	date_prescription	Date	—	Date de prescription
	duree	Number	—	Durée de prescription
	remboursable	Boolean	—	Indique si le traitement est actuellement remboursable
H-TraitementClient	id	String	—	Clé primaire générée automatiquement
	traitements_rembourses	List<T-Rembourse>	—	Liste des traitements remboursés
R-V-Vaccination	id	String	—	Clé primaire générée automatiquement
	nom	String	—	Nom de la personne
	prenom	String	—	Prénom de la personne
	email	String	—	Email de la personne
	telephone	String	—	Téléphone de la personne
	vaccin	String	—	Nom du vaccin
	date	Date	—	Date et heure du rendez-vous

Tableau 33 – Dictionnaire de données – Tables 3

Type	Valeur	Description
TypeProduit	médicament	Produit médicamenteux
	vaccin	Vaccin
	cosmétique	Produit cosmétique
	produit d'hygiène	Produit d'hygiène
	produit pour bébé	Produit pour bébé
	matériel et dispositif médical	Matériel médical
	phytothérapie	Phytothérapie
	aromathérapie	Aromathérapie
	parapharmacie	Parapharmacie
	produit vétérinaire	Produit vétérinaire
	santé et bien-être sexuel	Santé sexuelle
TypeCarte	cnas	Affilié à la cnas
	casnos	Affilié à la casnos
RoleUtilisateur	pharmacien	est un pharmacien
	vendeur	est un vendeur
	client	est un client
TauxRemboursement	0.8	80% de taux de remboursement
	1.0	100% de taux de remboursement
TypeVente	libre	Vente au comptoir ne nécessite pas d'ordonnance
	traitement	Nécessite une ordonnance

Tableau 34 – Dictionnaire de données - Types

Chapitre 3 : Étude des sprints 1,2,3 et 4

4. Étude du troisième sprint

4.1 Décomposition du troisième sprint

Le troisième sprint, d'une durée **quatre semaines**, a été consacré au **développement des fonctionnalités principales** et à l'**intégration des modules**. L'objectif principal était de mettre en place un environnement de développement stable, d'implémenter les fonctionnalités prévues et de réaliser les premiers tests fonctionnels afin de garantir la cohérence et la fiabilité du système.

Sprint Planning

— Mise en place de l'environnement de développement (Node.js, MongoDB et Bootstrap).
— Répartition des tâches de codage :

1. Historique médical.
2. Consultation des rendez-vous de vaccination.
3. Prise d'un rendez-vous (visiteur).
4. Consultation des produits disponibles (visiteur).

— Planification de l'intégration continue et des tests unitaires.
— Définition du Sprint Backlog 3.

ID	Tâche associée	Responsable	Livrable attendu
PB6	Développer le module d'**historique médical**	Binôme	Code implémenté + tests unitaires + interface d'accès à l'historique
PB7	Implémenter la **consultation des rendez-vous de vaccination**	Binôme	Code fonctionnel + tests de navigation + interface de consultation
PB11	Développer la fonctionnalité de **prise de rendez-vous (visiteur)**	Binôme	Code implémenté + formulaire de réservation + test de validation
PB12	Implémenter la **consultation des produits disponibles (visiteur)**	Binôme	Code fonctionnel + interface de recherche produit + tests associés
–	Intégration continue et tests globaux	Binôme	Application intégrée + rapport de bugs corrigés

Tableau 35 – Sprint Backlog 3 — Développement et tests

Daily Scrum

— Synchronisation quotidienne sur l'avancement du codage.
— Suivi de l'intégration front-end/Back-end.
— Résolution rapide des obstacles techniques liés à la base de données et aux interfaces.

Sprint Review

— Démonstration des modules achevés (historique médical, rendez-vous, consultation stock).
— Vérification de la navigation et de la clarté des interfaces.
— Recueil des retours de l'encadrante pour améliorer l'expérience utilisateur.

Sprint Retrospective

Problèmes rencontrés :
— Incohérence d'affichage entre modules.
— Bugs de validation sur les formulaire (ex. prise d'un rendez-vous).

Actions correctives :

— Automatisation de certains tests.
— Mise en place de bonnes pratiques de contrôle qualité.

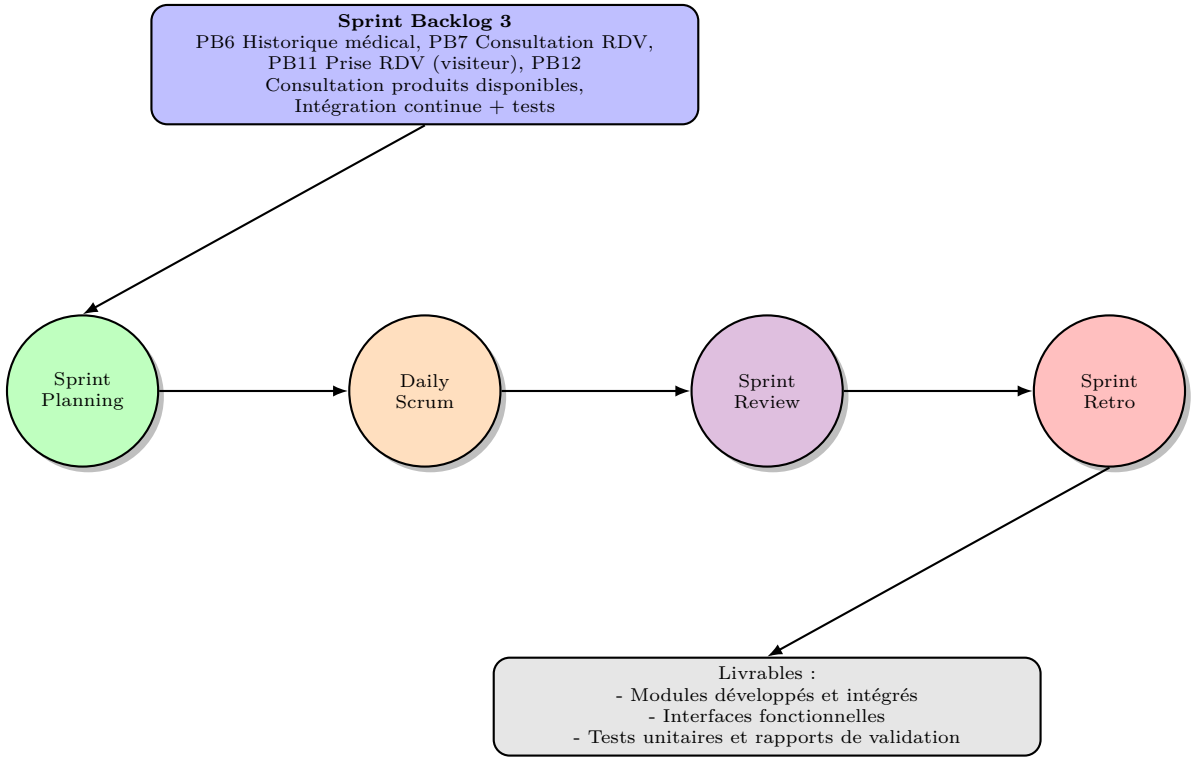

FIGURE 19 – Synthèse du Sprint 3

4.3 Diagramme de séquence : Consultation de l'historique d'un client

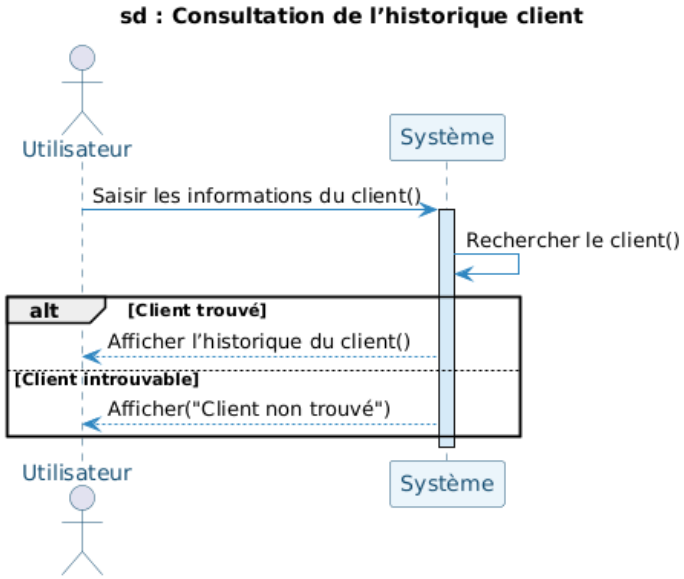

FIGURE 20 – Diagramme de séquence : Consultation de l'historique d'un client

5. Étude du quatrième sprint

5.1 Décomposition du quatrième sprint

Le quatrième sprint, qui a duré **trois semaines**, a été dédié à la **finalisation de l'application**, à **l'intégration des fonctionnalités** plus sophistiquées et à la préparation de la remise finale. La priorité était de terminer les modules restants, d'assurer l'intégration totale entre tous les éléments et de mener à bien les tests finaux afin d'assurer stabilité et conformité aux besoins métier.

Sprint Planning

— Priorisation des tâches restantes : Consultation des remboursements, sécurité avancée, rapports et statistiques
— Intégration complète des modules développés dans les sprints précédents.
— Planification des tests finaux et de l'optimisation de l'interface utilisateur.
— Définition du Sprint Backlog 4.

ID	Tâche associée	Responsable	Livrable attendu
PB8	Développer la fonctionnalité de **consultation des remboursements**	Binôme	Module fonctionnel + interface de consultation + tests unitaires
PB9	Implémenter les **mécanismes de sécurité avancée** (sessions sécurisées, reCaptcha, chiffrement)	Binôme	Code sécurisé + documentation + tests d'accès et permissions
PB10	Générer des **rapports et statistiques**	Binôme	Module de reporting fonctionnel + graphiques + tests de calcul et affichage
–	Intégration finale et tests globaux	Binôme	Application complète, stable et prête à la livraison + rapport de tests finaux

Tableau 36 – Sprint Backlog 4 — Finalisation et fonctionnalités avancées

Daily Scrum

— Supervision de l'implémentation des fonctionnalités restantes.
— Contrôle de la compatibilité et de l'intégration des modules élaborés lors des précédents sprints.
— Détection et correction des bugs identifiés.

Sprint Review

— Démonstration des fonctionnalités de pointe : remboursements, sécurité et rapports.
— Vérification de l'intégration complète de l'application.
— Collecte des commentaires de l'encadrante pour toute modification finale.

Sprint Retrospective

Points positifs :
— Application stable et prête à être livrée.
— Cohérence et intégration réussies entre tous les modules.

Difficultés :
— Ajustements mineurs dans les modules hérités des sprints précédents.
— Optimisation de la performance des rapports et statistiques.

Actions décidées :
— Finalisation de la documentation technique et utilisateur.
— Préparation des livrables pour la soutenance ou mise en production.

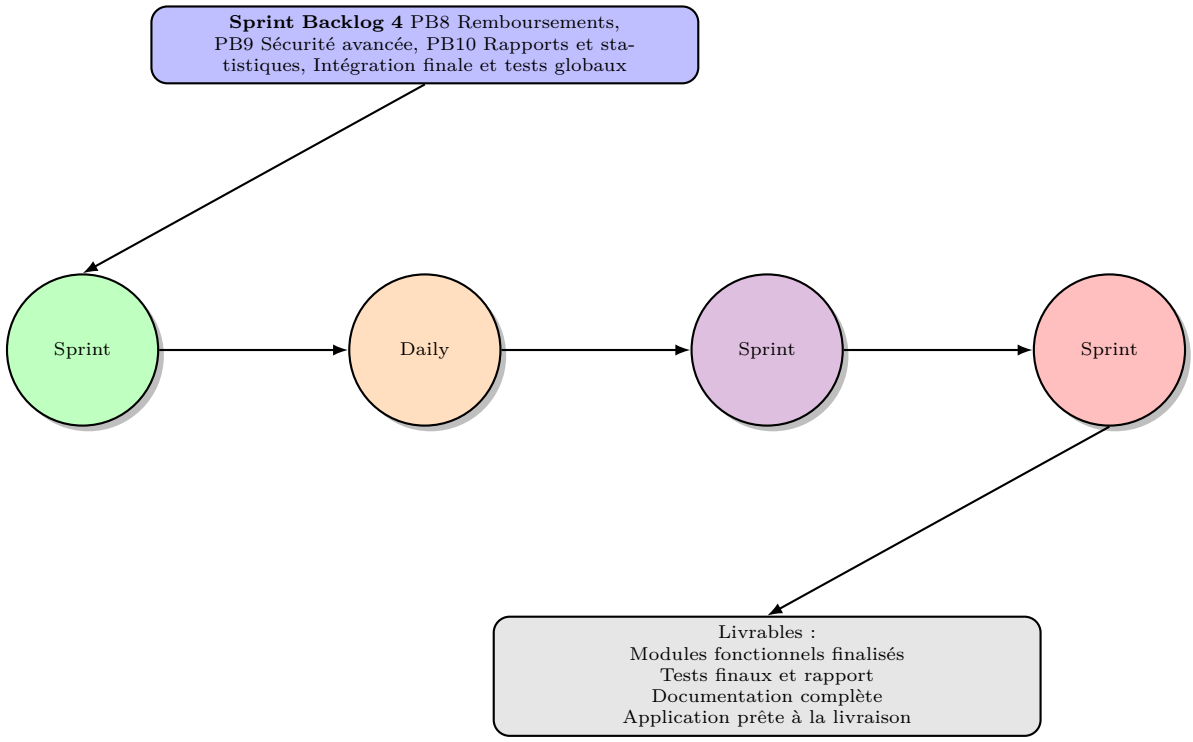

FIGURE 21 – Synthèse du Sprint 4

5.2 Diagramme de séquence : Prise d'un rendez-vous de vaccination

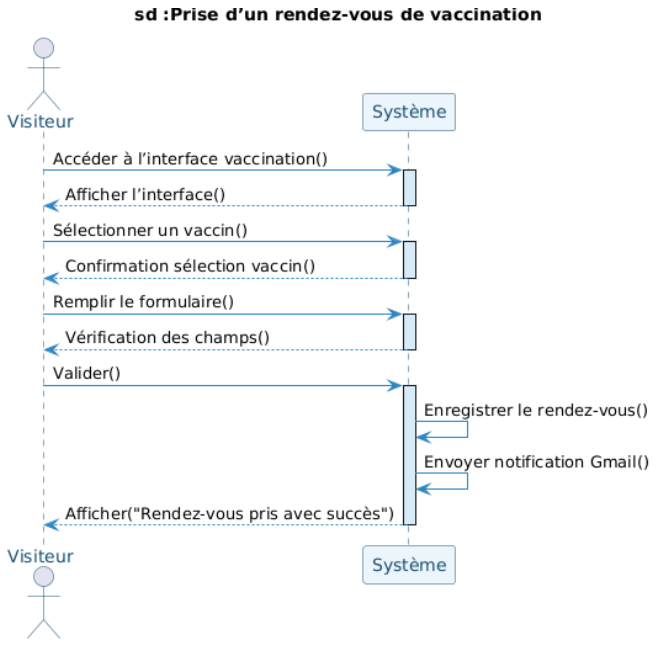

FIGURE 22 – Diagramme de séquence : Prise d'un rendez-vous de vaccination

5.2 Architecture générale de l'application

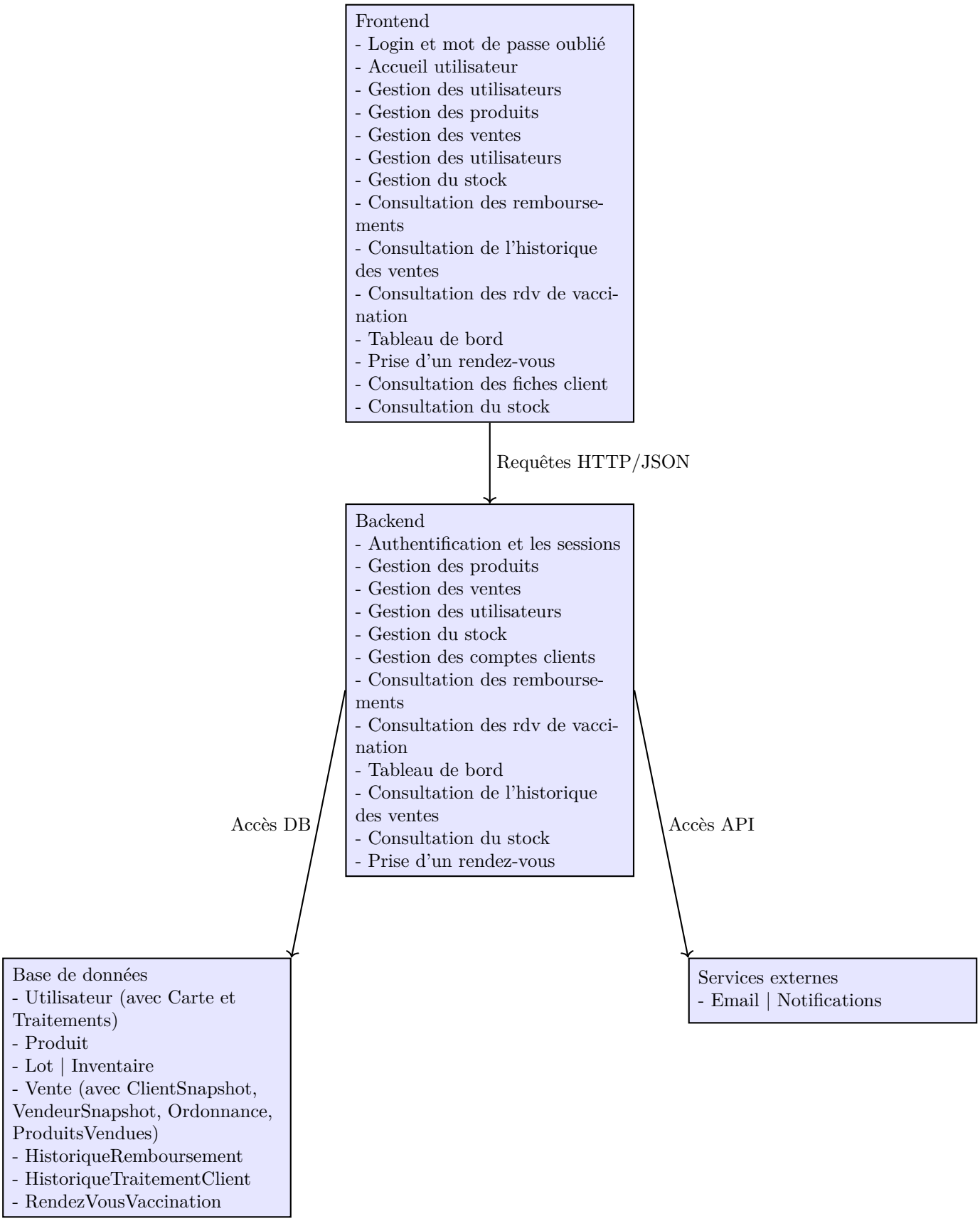

Figure 23 – Architecture globale de l'application

6. Synthèse et conclusions sur l'application de Scrum

À l'issue des quatre sprints de développement, l'application de gestion pharmaceutique est fonctionnelle et intégrée, répondant aux besoins identifiés au départ. La méthode **Scrum** a permis de structurer le travail en cycles courts et itératifs, tout en facilitant l'adaptation aux retours des utilisateurs et de l'encadrante.

Récapitulatif des réalisations par sprint :

— **Sprint 1 :** Analyse des besoins et cadrage fonctionnel, définition du Product Backlog et élaboration des premières maquettes d'interface et diagrammes UML initiaux.

— **Sprint 2 :** Conception détaillée, modélisation complète de la base de données (MongoDB), diagrammes UML détaillés (cas d'utilisation, classes et séquences) et préparation des spécifications pour le développement.

— **Sprint 3 :** Développement des modules prioritaires, notamment l'historique médical, la consultation et la prise de rendez-vous, ainsi que la consultation des produits disponibles. Les tests unitaires et l'intégration continue ont été réalisés pour assurer la cohérence des modules.

— **Sprint 4 :** Finalisation des fonctionnalités avancées telles que la gestion des remboursements, la sécurité renforcée et les rapports statistiques. Intégration complète de l'application et tests finaux pour valider l'ensemble du système.

7. Conclusion

Ce chapitre a posé les fondations essentielles pour la réalisation technique de l'application. Nous avons tout d'abord identifié et formulé de manière précise les besoins fonctionnels et non fonctionnels, assurant ainsi une compréhension claire des attentes du système.

Nous avons pu identifier clairement les différents acteurs du système ainsi que leurs rôles, la modélisation UML a structuré les principales interactions de manière visuelle. Le passage au modèle relationnel de la base de données a permis de traduire les besoins fonctionnels du système en une structure de données facilement exploitable par l'application.

Chapitre 4 : Implémentation de l'application

Chapitre 4 : Implémentation de l'application

1. Introduction

Cette partie détaille la procédure de mise en œuvre de l'application web destinée à la gestion pharmaceutique, s'appuyant sur les modèles et spécifications établis dans les sections antérieures. Il a pour but d'exposer les décisions technologiques, le cadre de développement, ainsi que les phases de programmation des divers modules fonctionnels du système.

Le but est de transformer la conception théorique en une solution logicielle tangible, solide et opérationnelle. Pour ce faire, nous allons aussi traiter les éléments de sécurité, comme l'authentification, l'autorisation et la vérification des données, pour garantir la fiabilité et la sauvegarde du système.

2. Environnement de développement

L'environnement de développement représente l'ensemble des outils techniques et matériels employés pour élaborer, programmer, tester et mettre en production l'application. Pour la réalisation de ce projet, nous avons fait appel aux éléments suivants :

2.1 Matériel utilisé

— **Ordinateur portable** : équipé d'un processeur Intel Core i3, 8 Go de mémoire vive et un disque SSD de 256 Go, ce matériel offre des performances suffisantes pour le développement et les tests.
— **Système d'exploitation** : Windows 11 a été utilisé pour son interface conviviale et sa compatibilité avec la majorité des outils de développement.

2.2 Technologie utilisées

— **Front-end** : HTML, CSS et JavaScript ont été utilisés pour structurer, styliser et rendre les pages interactives. Bootstrap permet une conception responsive et rapide grâce à ses composants préconçus.

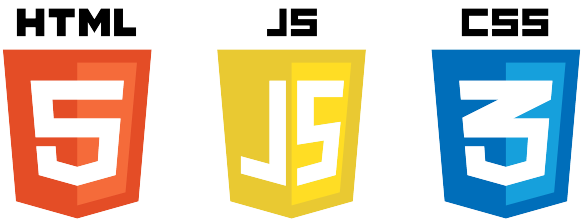

FIGURE 24 – Logos HTML/CSS/JavaScript

FIGURE 25 – Logo de Bootstrap

— **Back-end** : Node.js est un environnement d'exécution JavaScript côté serveur, conçu pour la création d'applications web efficaces et de haute performance.

]

FIGURE 26 – Logo de Node.js

— **Base de données** : MongoDB a été choisi pour sa flexibilité, sa capacité à gérer des données semi-structurées (documents JSON), et sa facilité de mise à l'échelle horizontale. Ce choix permet également une adaptation rapide aux évolutions du modèle de données, ce qui est essentiel pour ce projet.

FIGURE 27 – Logo de MongoDB

2.3 Environnement de développement intégré

— **Visual Studio Code (VS Code)** : un IDE (:Integrated Development Environment) léger et performant, largement utilisé par la communauté, qui offre de nombreuses extensions pour faciliter le développement web.

FIGURE 28 – Logo de Visual Studio Code

2.4 Navigateur pour les tests

— **Google Chrome** : utilisé pour tester l'application grâce à ses outils de développement intégrés qui permettent de déboguer efficacement le code HTML, CSS et JavaScript.

FIGURE 29 – Logo de Google Chrome

2.5 Autres outils

— **Postman** : utilisé pour simuler et tester les requêtes HTTP, particulièrement utile lors de l'intégration avec le back-end.

FIGURE 30 – Logo de Postman

— **Figma** : outil de prototypage et de design d'interface, utilisé pour la modélisation des maquettes de l'application avant implémentation.

FIGURE 31 – Logo de Figma

— **Adobe Photoshop et Adobe Illustrator** : ces deux outils complémentaires ont été utilisés pour la création du logo de l'application. Illustrator a permis de concevoir les formes vectorielles, tandis que Photoshop a été utilisé pour les retouches et les effets.

FIGURE 32 – Logo de Photoshop/Illustrator

3. Architecture technique de l'application

3.1 Architecture globale : Client-Serveur

Le protocole de communication client-serveur fait référence à l'échange d'informations entre deux programmes ou processus, dans lequel le client transmet des requêtes et le serveur y réagit en offrant un service. Il est possible d'exécuter le client et le serveur sur des machines distinctes ou bien sur une seule et même machine. Les serveurs ont généralement des capacités matérielles supérieures pour permettre la gestion simultanée de plusieurs clients de manière efficace. [8]

Dans l'application web Pharmadise, le protocole client-serveur permet la communication entre le navigateur (client) et le serveur. Le client envoie des requêtes pour gérer les produits, les stocks ou les clients. Le serveur reçoit ces requêtes, traite les données via la base de données, puis renvoie les réponses au client. La communication utilise le protocole HTTP/HTTPS, garantissant un échange efficace et sécurisé entre client et serveur. [9]

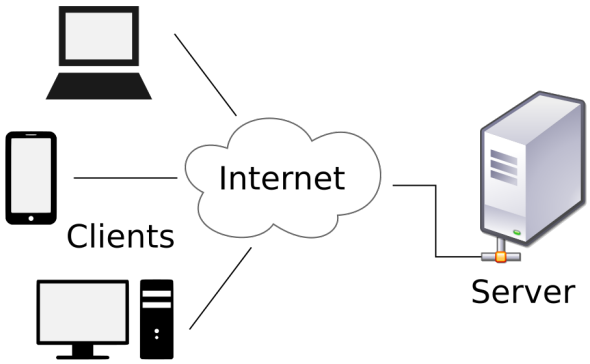

FIGURE 33 – Architecture Client Serveur

3.2 Architecture logique : MVC

Le modèle-Vue-Contrôleur, ou MVC, est un modèle d'architecture logicielle conçu pour les interfaces graphiques, il est particulièrement prisé pour les applications web. Le motif comprend trois catégories de modules, chacune ayant une responsabilité distincte : les modèles, les vues et les contrôleurs. [10]

- **Modèle (Model)** : Renferme les données à présenter. Il contient la logique métier et interagit avec la base de données.
- **Vue (View)** : Englobe la présentation de l'interface utilisateur. Elle est responsable de l'affichage des données fournies par le contrôleur.
- **Contrôleur (Controller)** : Gère la logique relative aux actions réalisées par l'utilisateur. Il agit comme un intermédiaire entre le modèle et la vue, en traitant les requêtes et en définissant les réponses à renvoyer.

4. Logique métier

Dans cette section, nous présentons les principaux algorithmes définissant la logique métier de quelques fonctionnalités complexes de l'application. Chaque algorithme est décrit de manière structurée afin de clarifier les règles de gestion et les contraintes fonctionnelles.

4.1 Processus de vente d'un produit

La vente d'un produit dans une pharmacie s'effectue à travers une série de phases clairement établies qui assurent le respect des normes réglementaires, l'approvisionnement du stock et la satisfaction du client. L'algorithme suivant expose de façon organisée l'ensemble du processus de vente, à partir de la validation du client et du produit jusqu'à l'achèvement et l'enregistrement de la transaction.

Algorithme 1 : Processus de Vente d'un Produit

Entrées : Client, Produit, TypeVente, Ordonnance (optionnelle)
Sorties : Vente validée ou Erreur bloquante

1. Vérification du client
if *client non enregistré* **then**
  Vente possible uniquement en type *libre*;
  Aucun traitement ou remboursement autorisé;
  Aller à l'étape 2 (vente libre uniquement);
else
  Vente possible en type *libre* ou *traitement*;
  Continuer vers l'étape 2;

2. Vérification du produit
Vérifier que le produit existe en base;
Vérifier que le stock est suffisant (lots disponibles avec date > 180 jours);
if *produit inexistant OU stock insuffisant* **then**
  STOP (erreur bloquante);
else
  Continuer vers l'étape 3;

3. Détermination de la nature du produit
if *produit médicament* **then**
  Pas de limite de boîtes, aucun remboursement possible;
  Vente = type *libre* uniquement;
  Aller à l'étape 4.1;
else
  Vérifier nombre maximum de boîtes autorisées;
  if *dépassement* **then**
    STOP (erreur bloquante)
  Vérifier allergie client (alerte si positif, non bloquant);
  Vérifier compatibilité familles (alerte si incompatibilité, non bloquant);
  Continuer vers l'étape 4;

4. Type de vente
if *TypeVente = libre* **then**
  Montant = prix × quantité;
  Aucun remboursement possible;
  Aller à l'étape 6;
else
  if *TypeVente = traitement* **then**
    Vérifier client enregistré et ordonnance valide;
    Vérifier éligibilité ordonnance selon cas (exonéré, malade chronique, 90 jours);
    if *inéligible* **then**
      STOP (erreur bloquante)
    Continuer vers l'étape 5;
  else
    STOP (type inconnu);

5. Remboursement (si traitement)
if *produit non remboursable* **then**
  Montant à payer = prix × quantité;
else
  Montant réel = prix × quantité;
  Calcul du remboursement selon tarif référentiel et taux carte client;
  Application du plafond : si montant total > 5000 DA et client non exonéré → pas de remboursement;

6. Vérification finale de la vente
Si la vente contient un traitement et pas d'ordonnance → **STOP**;
Construire snapshots (Client, Vendeur, Produits);
Calculer totaux (montant total, à payer, remboursé);
Enregistrer la vente (date et heure);
Déstocker produits en suivant dates de péremption croissantes;
Fin

4.2 Gestion des lots dans l'inventaire

Algorithme 2 : Algorithme de la logique métier des Lots

Entrées : numero_lot, nom_produit, quantité, date_péremption, dosage (si médicament)
Sorties : Lots et Inventaire mis à jour
Début
1. Ajout d'un lot
1.1 Vérifier champs obligatoires : numero_lot, nom_produit, quantité, date_péremption.
if *manquant* **then**
  STOP
1.2 Vérifier que le produit existe.
if *inexistant* **then**
  STOP
1.3 **if** *produit = médicament* **then**
  dosage requis
1.4 Enregistrer lot dans Lot.
1.5 Mettre à jour Inventaire :
  if *inventaire existe déjà (produit/dosage)* **then**
    ajouter lot
  else
    créer nouvel inventaire avec ce lot
2. Modification d'un lot
2.1 Vérifier que le produit (nouveau nom) existe.
if *inexistant* **then**
  STOP
2.2 **if** *produit = médicament* **then**
  dosage requis
2.3 Retirer le lot de l'ancien inventaire.
2.4 Mettre à jour champs du lot (nom, dosage, quantité, date_péremption).
2.5 Sauvegarder lot.
2.6 **if** *inventaire n'a plus de lots* **then**
  supprimer inventaire
else
  sauvegarder inventaire mis à jour
3. Suppression d'un lot
3.1 Supprimer lot de la collection Lot.
3.2 Mettre à jour inventaire associé :
  Retirer lot supprimé.
  if *inventaire vide* **then**
    supprimer inventaire
  else
    sauvegarder inventaire mis à jour
4. Consultation des lots
4.1 Calculer état_stock selon date péremption :
  if $\leq$ *aujourd'hui* **then**
    Expiré
  if $\leq$ *180 jours* **then**
    Expirant bientôt
  else
    Non expiré
4.2 Retourner liste avec état.
5. Règles de cohérence Lot / Inventaire
Toute opération sur un Lot met à jour immédiatement son Inventaire.
Disponibilités = somme des lots valides (quantités, dates > 180 jours).
Aucun lot sans inventaire associé.
Aucun inventaire sans lots (supprimé automatiquement).
Fin

4.3 Gestion des remboursements

Algorithme 3 : Algorithme de la logique métier des Remboursements

Entrées : Vente, Client (avec typeCarte CNAS/CASNOS), montant_rembourse_total
Sorties : Historique remboursements mis à jour
Début
1. Vérification initiale
if *montant_rembourse_total = 0* ***OU*** *pas de client lié* **then**
└ **STOP**
Sinon → continuer.
2. Historique remboursement (CNAS / CASNOS)
2.1 Récupérer date vente → {mois, année}.
2.2 Identifier typeCarte client (CNAS / CASNOS).
2.3 Mettre à jour historique remboursement :
Rechercher document (mois, année, typeCarte).
if *document trouvé* **then**
└ incrémenter montant_total_rembourse
else
└ créer nouveau document
3. Historique traitement du client
3.1 **if** *aucun client lié* **then**
└ **STOP**
3.2 Filtrer produits vendus : garder {type_vente = traitement ET rembourse = true}.
3.3 **if** *aucun traitement remboursé* **then**
└ **STOP**
3.4 Dé-dupliquer produits par (nom, dosage).
3.5 Pour chaque produit unique :
Rechercher ou créer historique client.
Ajouter dans traitements_rembourses :
- nom_produit
- dosage
- date_prescription (ordonnance)
- durée (ordonnance)
- remboursable = false (passe à true après expiration durée)

- vente_id
4. Conséquences pour le client
- Peut consulter historique de ses remboursements.
- Utilisé pour :
Vérifier fréquence prescriptions (< 2 en 90 jours).
Identifier traitements habituels.
5. Conséquences pour CNAS / CASNOS
- Historique agrégé permet de suivre :
Montants remboursés mensuels par typeCarte.
Gestion budgétaire et déclarations.
Fin

5. Charte graphique

5.1 Nom de l'application

Le nom **Pharmadise** est un jeu de mots entre *pharmacie* et *paradise* (paradis en anglais), reflétant l'idée d'une application complète et agréable à utiliser pour la gestion pharmaceutique.

5.2 Logo

Le logo de l'application **Pharmadise** est représenté ci-dessous. Il combine la *coupe d'Hygie*, symbole universel de la pharmacie, avec un design moderne et digital, reflétant le côté interactif de l'application.

FIGURE 34 – Logo de l'application Pharmadise

5.3 Palette de couleur

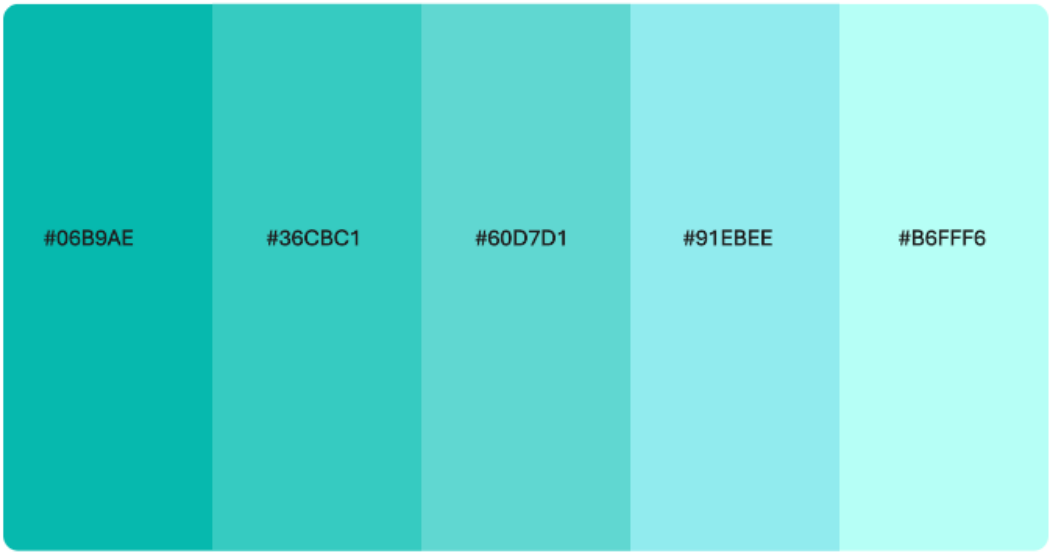

FIGURE 35 – Palette de couleur

6. Sécurité

La protection des données critiques, notamment dans le secteur médical et pharmaceutique, est d'une importance capitale. Pour garantir la confidentialité, l'intégrité et l'accès aux informations, notre application a intégré divers mécanismes de sécurité :

— **Authentification des utilisateurs :** un système d'authentification par nom d'utilisateur et mot de passe a été mis en place. Chaque utilisateur (pharmacien, vendeur ou client) doit se connecter pour accéder à l'interface.

— **Gestion des rôles et des permissions :** l'accès aux différentes parties de l'application est contrôlé selon le rôle de l'utilisateur. Par exemple, bien que le vendeur puisse gérer le stock et les clients ou consulter la liste des produits, il n'a pas accès aux modules critiques tels que la gestion des utilisateurs ou des produits, la consultation des remboursements, les analyses statistiques de la pharmacie.

— **Protection des données sensibles :** les informations critiques telles que les données des patients, les transactions et les mots de passe des utilisateurs sont protégées à plusieurs niveaux. Côté serveur, les données sont sécurisées pour empêcher tout accès non autorisé. Les mots de passe sont hachés à l'aide de la librairie *bcrypt*, afin de prévenir les attaques par dictionnaire ou par tables arc-en-ciel. Dans un environnement de déploiement réel, les échanges entre client et serveur peuvent être protégés par le protocole HTTPS afin de garantir la confidentialité des communications.

— **Gestion des sessions :** l'application maintient la session de l'utilisateur pendant la durée d'utilisation. Une session est automatiquement expirée après une période d'inactivité de 4 heures afin de renforcer la sécurité et limiter les risques d'accès non autorisé.

— **Suivi des connexions utilisateur :** la date et l'heure de la dernière connexion sont affichées dans l'interface, permettant de détecter toute activité suspecte non autorisée.

— **Encodage sécurisé des jetons d'authentification :** les jetons (tokens) utilisé pour la vérification des utilisateurs, notamment dans les processus de réinitialisation de mot de passe ou de session, sont encodés et signés afin de garantir leur intégrité et leur non-altérabilité.

— **Réinitialisation du mot de passe :** lorsqu'un utilisateur demande la réinitialisation de son mot de passe, un lien sécurisé et temporaire lui est envoyé. Ce lien est valable pour une durée limitée de 15 minutes, au-delà de laquelle il expire automatiquement.

— **Utilisation de reCAPTCHA :** un système de reCAPTCHA a été intégré dans les formulaires sensibles (connexion, réinitialisation de mot de passe, etc.) afin de distinguer les utilisateurs humains des robots et de limiter les attaques automatisées.

Figure 36 – Logo de Google reCAPTCHA

7. Interfaces de l'application

7.1 Interface d'accueil - Visiteur

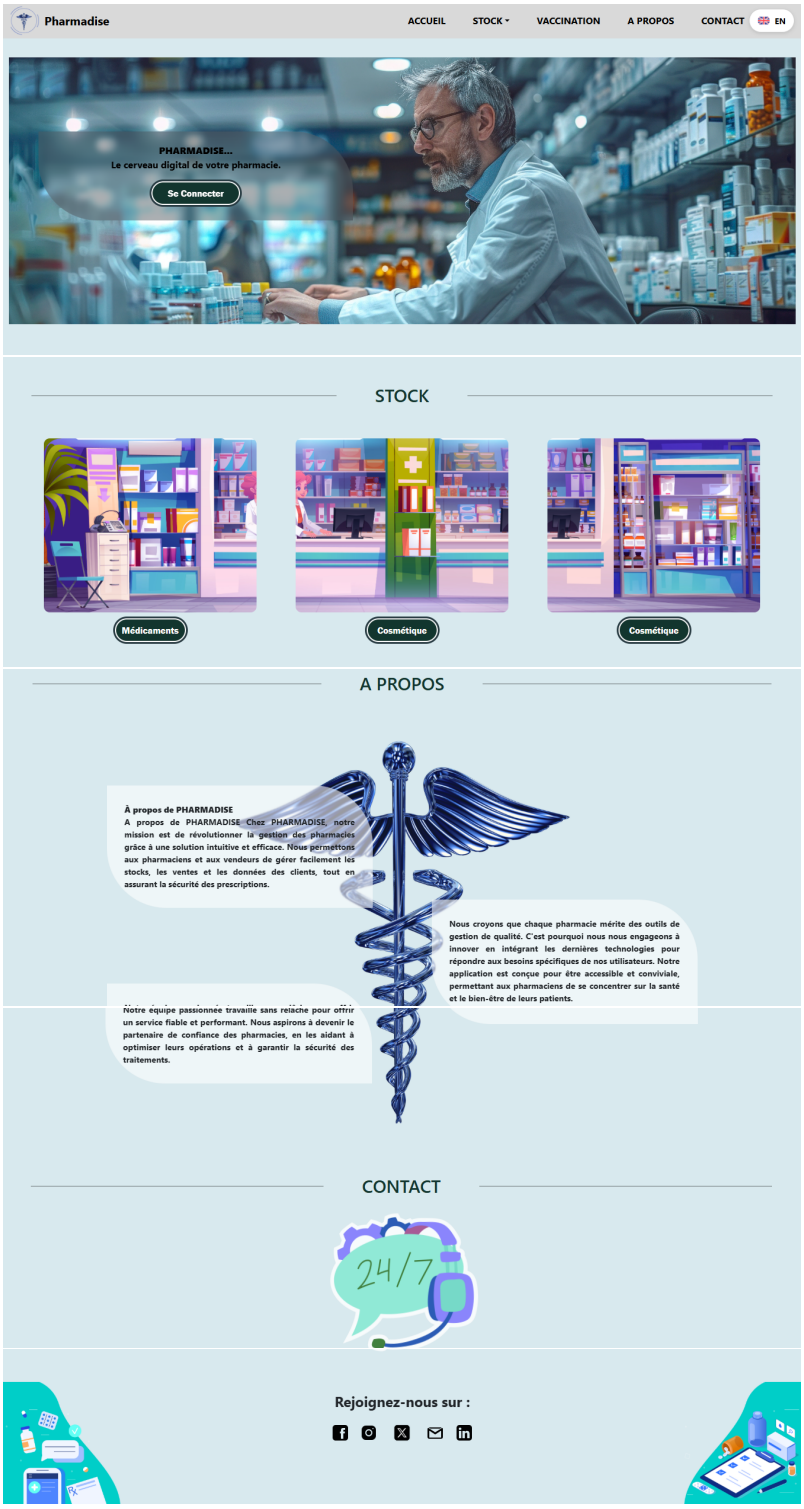

FIGURE 37 – Interface d'accueil pour le visiteur

7.2 Interface de connexion

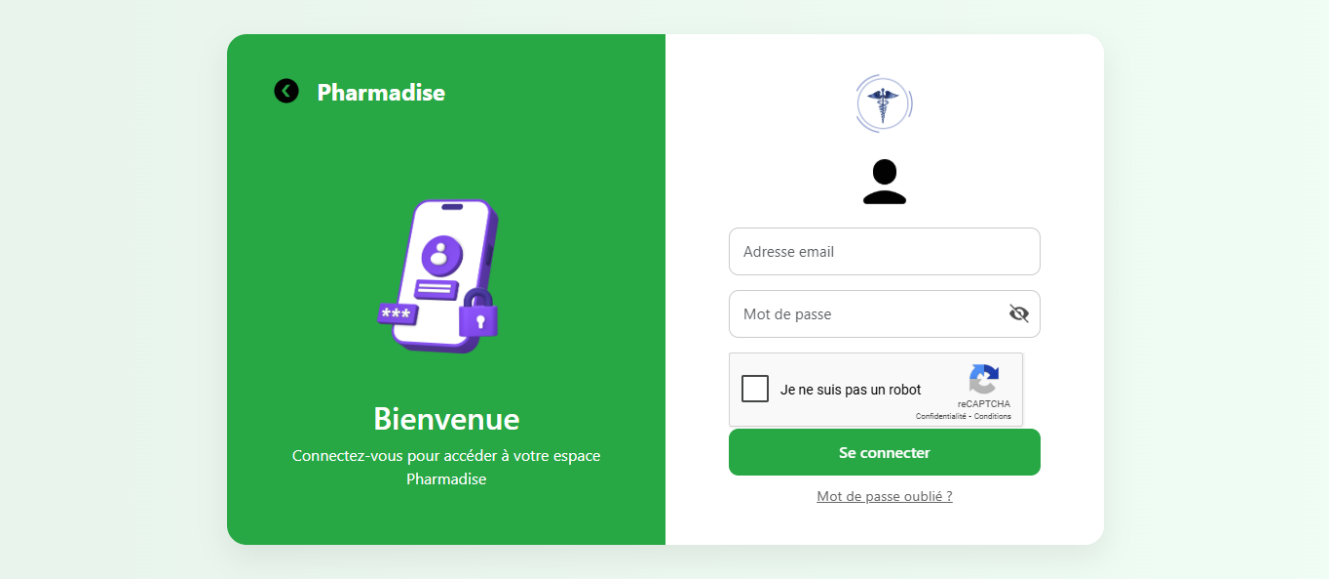

FIGURE 38 – Interface de connexion

7.3 Interface d'accueil - Pharmacien

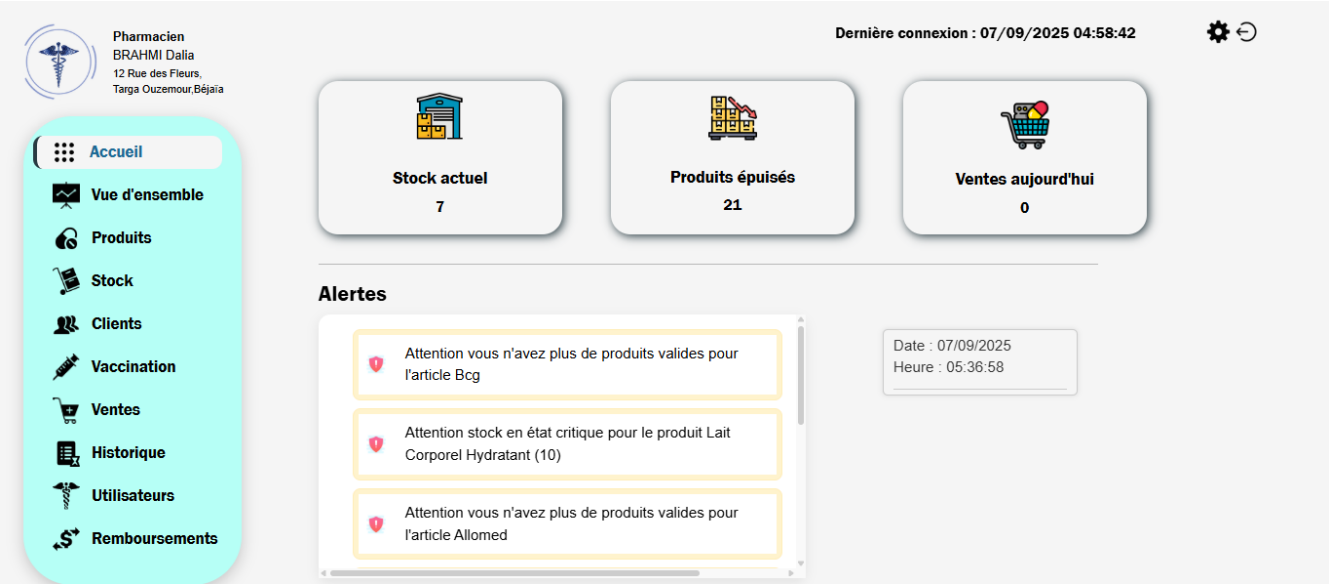

FIGURE 39 – Interface d'accueil - Pharmacien

7.4 Interface d'accueil - Vendeur

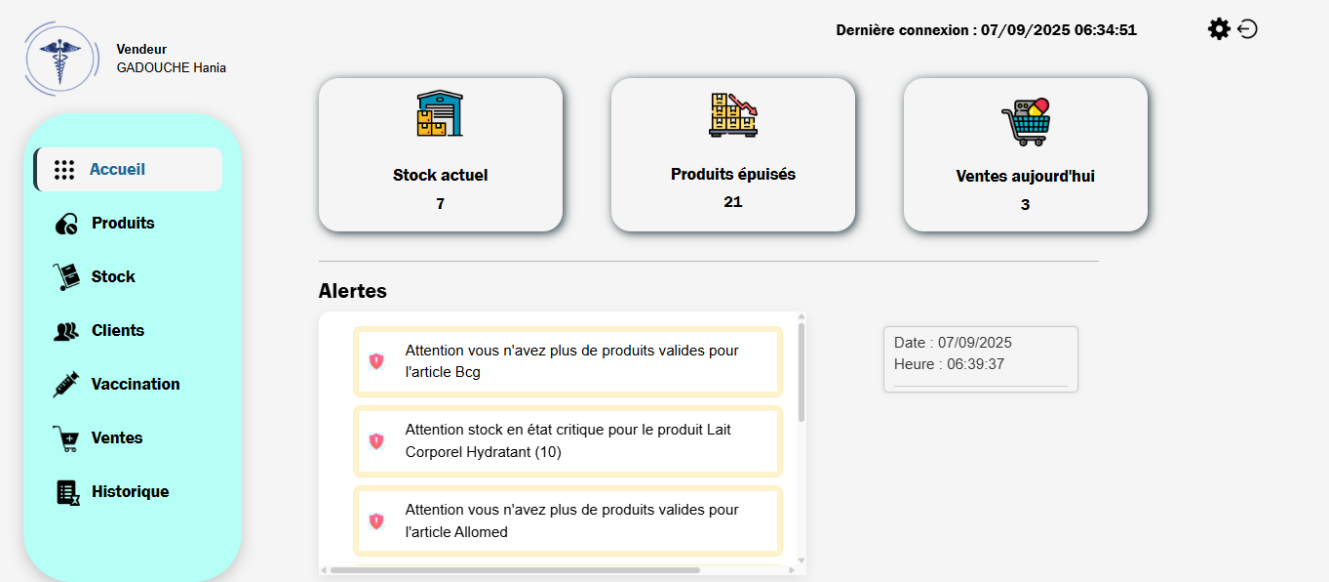

FIGURE 40 – Interface d'accueil - Vendeur

7.5 Interface des statistiques

FIGURE 41 – Interface des statistiques

7.6 Interface des produits

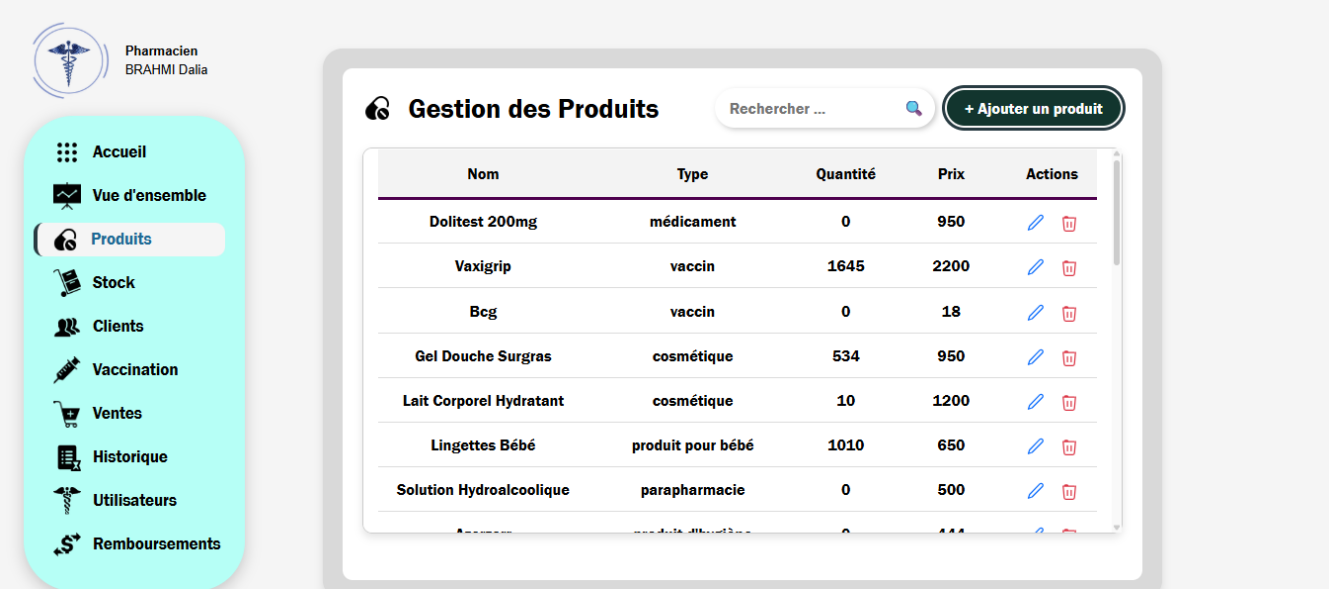

FIGURE 42 – Interface des produits

7.7 Interface de stock

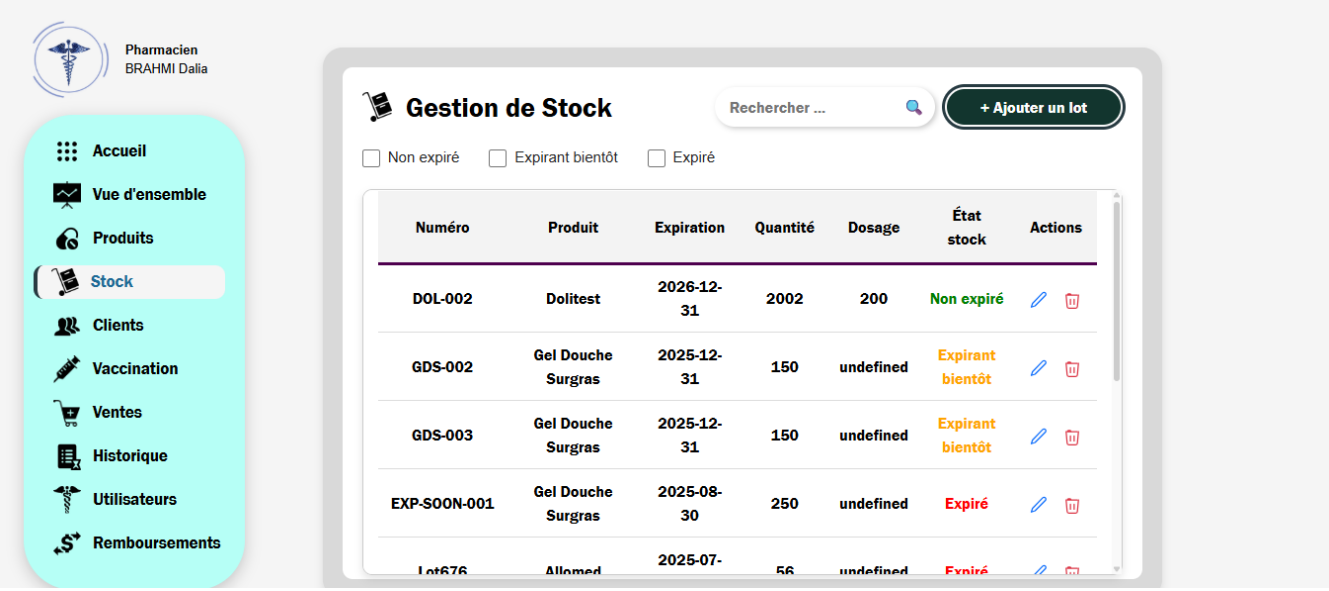

FIGURE 43 – Interface de stock

7.8 Interface des clients

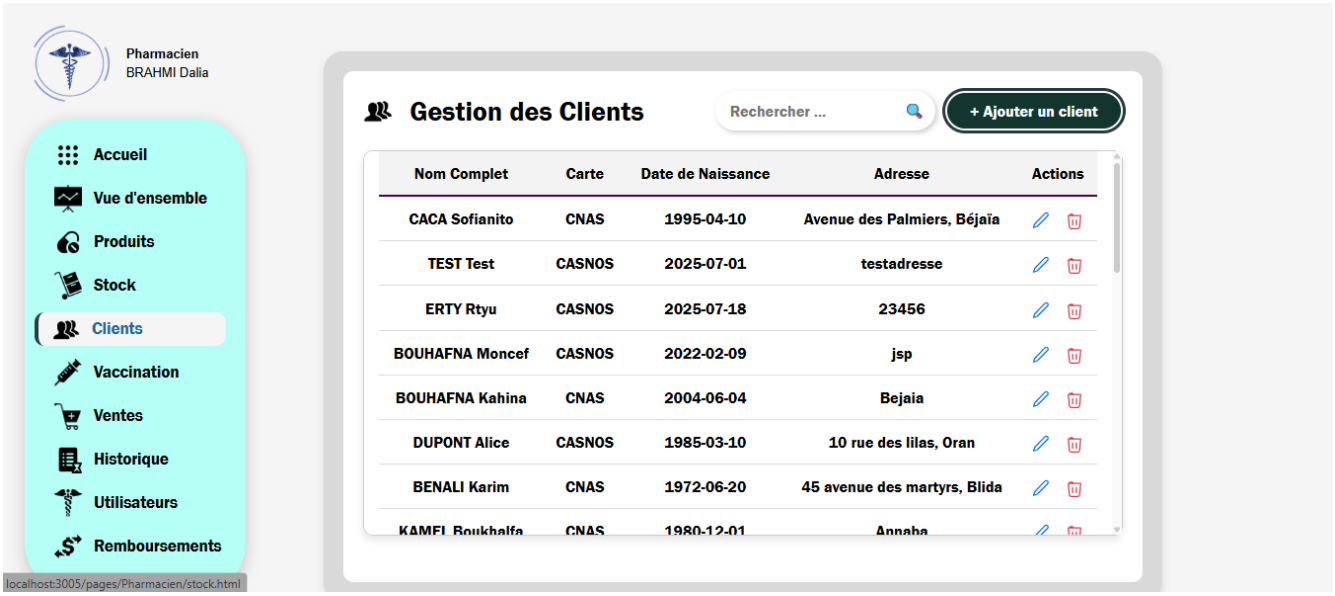

FIGURE 44 – Interface des clients

7.9 Interface des ventes

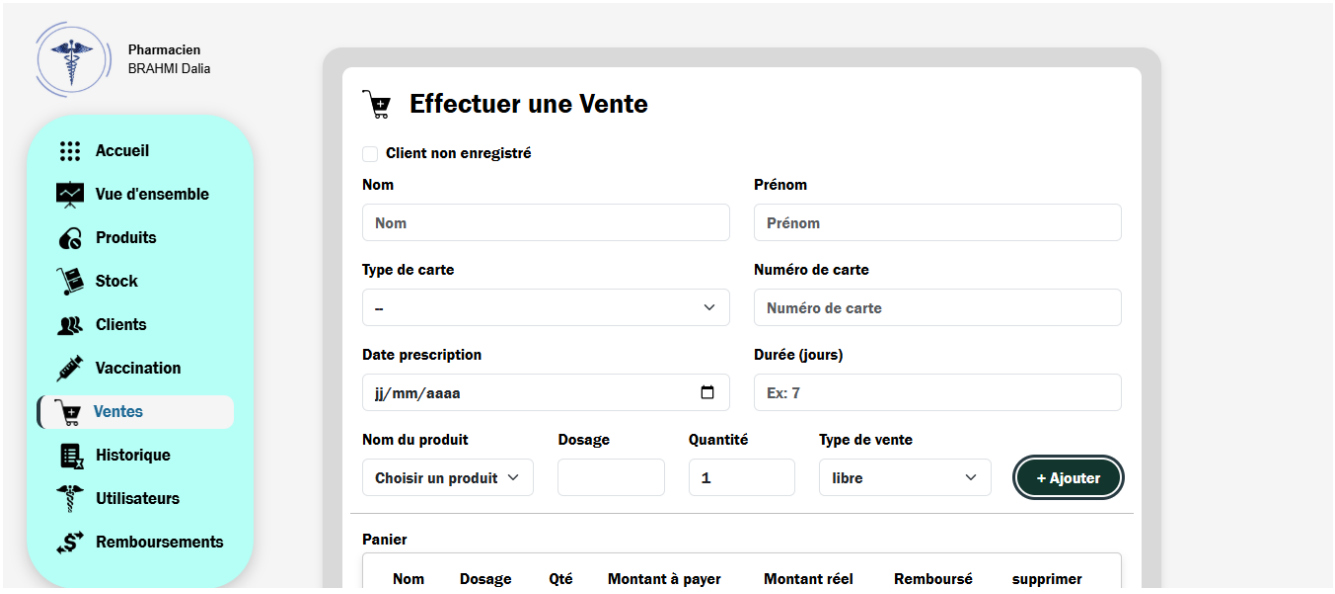

FIGURE 45 – Interface des ventes

7.10 Interface de l'historique

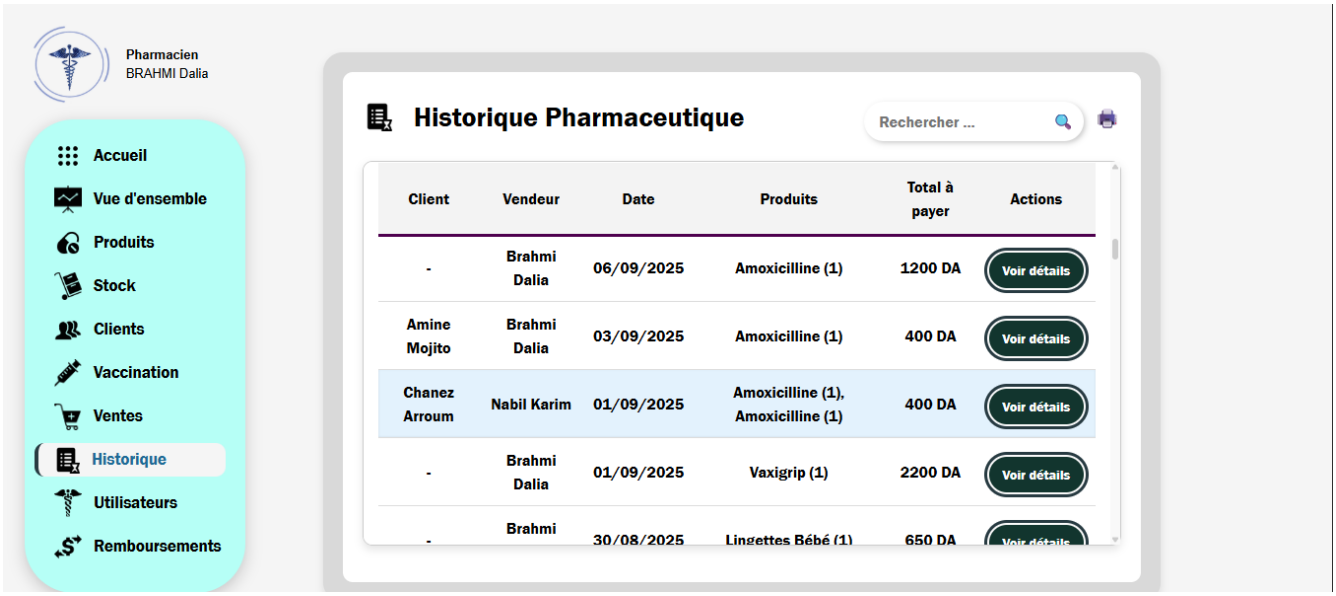

FIGURE 46 – Interface de l'historique

7.11 Interface des utilisateurs

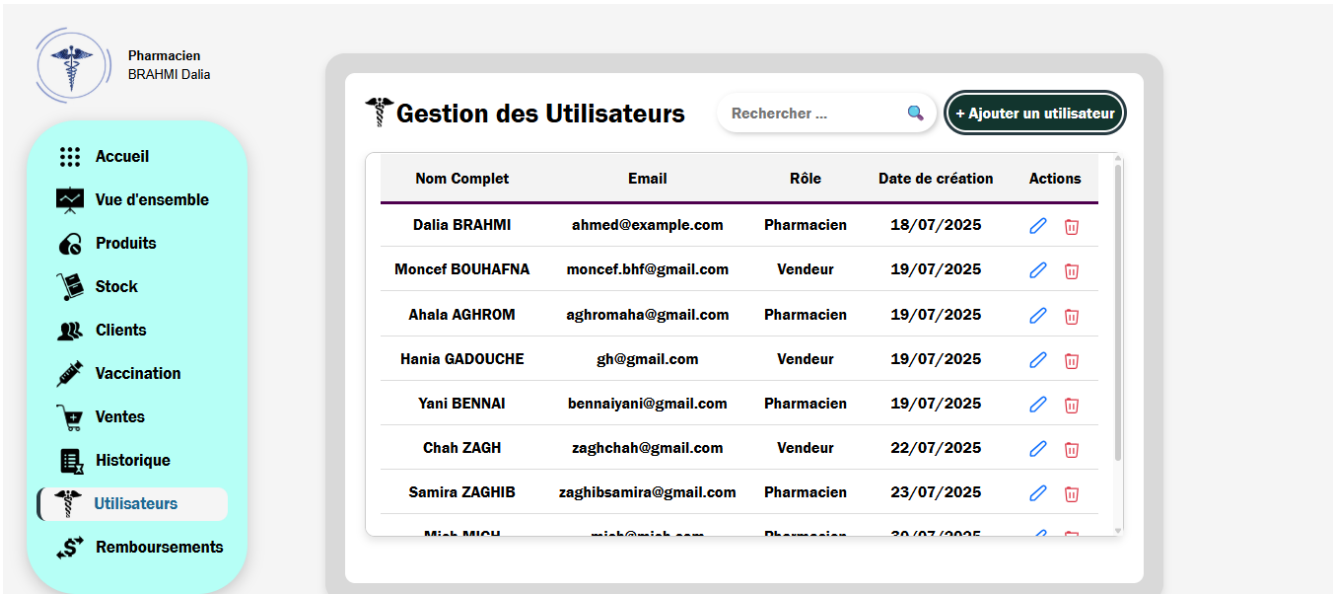

FIGURE 47 – Interface des utilisateurs

7.12 Interface des remboursements

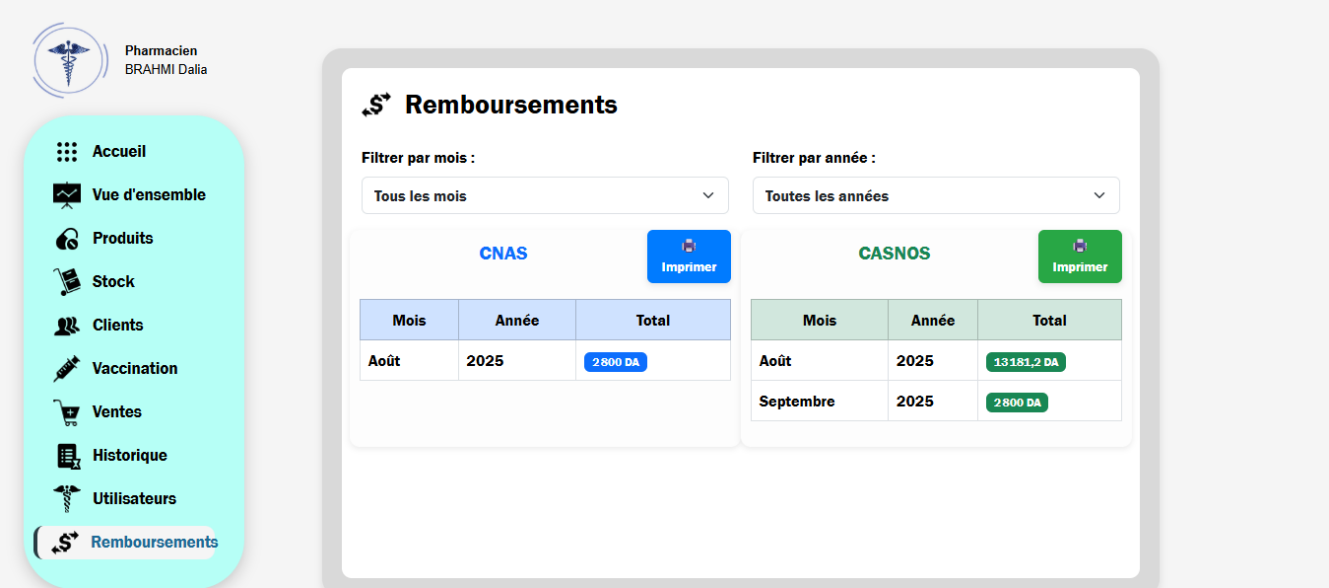

FIGURE 48 – Interface des remboursements

7.13 Interface des rendez-vous de vaccination

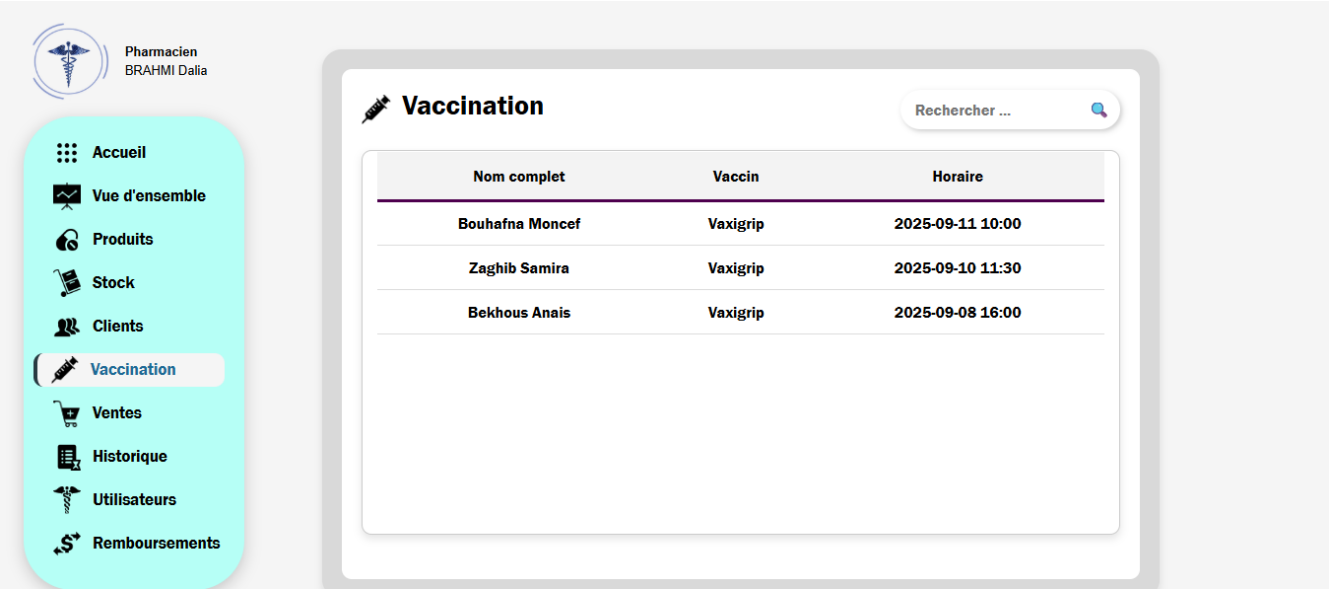

FIGURE 49 – Interface des rendez-vous de vaccination

7.14 Interface de prise des rendez-vous de vaccination - Visiteur

FIGURE 50 – Interface de prise des rendez-vous de vaccination - Visiteur

7.15 Interface d'accueil - Client

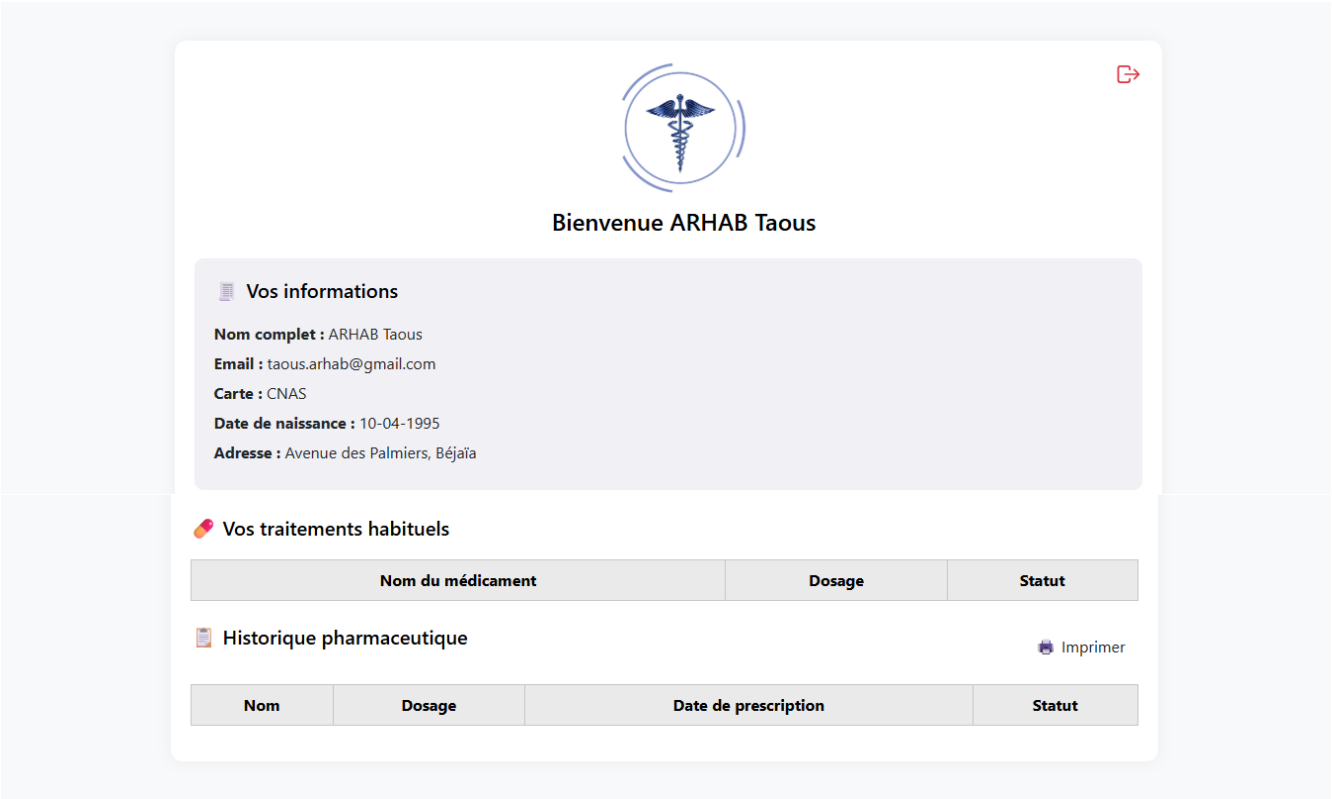

FIGURE 51 – Interface d'accueil - Client

7.16 Interface de réglages d'un compte utilisateur

FIGURE 52 – Interface de réglages d'un compte

8. Conclusion

Ce chapitre a permis de mettre en lumière les différentes étapes pratiques de la concrétisation de l'application web Pharmadise. En s'appuyant sur les choix technologiques définis et l'architecture logicielle retenue, chaque modules fonctionnel a été soigneusement développé afin de répondre aux exigences exprimées lors de la phase de conception.

L'implémentation a été menée dans un environnement de développement moderne et cohérent, intégrant des outils adaptés à la fois au front-end, au back-end, à la base de données et aux tests. La structure e couches, selon le modèle MVC, a favorisé une séparation claire des responsabilités, facilitant ainsi la maintenance et l'évolution future de l'application.

Une attention particulière a été accordée aux aspects de sécurité, essentiels dans un système manipulant des données sensibles. Les mécanismes d'authentification, d'autorisation. de protection des données et de contrôle des sessions contribuent à garantir la fiabilité et la confidentialité des informations traitées.

Ainsi, l'ensemble des fonctionnalités attendues a été traduit en une application fonctionnelle, intuitive et sécurisée, constituant une base solide pour un déploiement réel ou une évolution vers une version plus avancée.

Conclusion générale

Conclusion générale

Ce projet a abouti à la conception et au développement d'une application web dédiée à la gestion des pharmacies, réalisée selon une méthodologie rigoureuse allant de l'analyse des besoins jusqu'au déploiement. L'étude du contexte pharmaceutique algérien nous a permis d'identifier les principaux défis liés à la traçabilité, à la gestion des stocks et au suivi des ventes. Ces constats ont orienté la conception d'une solution adaptée aux besoins réels du terrain.

Grâce à l'utilisation combinée de technologies modernes telles que **Node.js**, **MongoDB**, **HTML**, **CSS**, **JavaScript** et **Bootstrap**, nous avons pu concevoir une application performante, ergonomique et sécurisée. Celle-ci intègre les fonctionnalités essentielles à une gestion efficace : administration des produits et lots, suivi des clients, historique pharmaceutique, ainsi qu'un système d'authentification sécurisé.

Ce travail a constitué une expérience complète, mobilisant nos compétences théoriques et pratiques à travers toutes les étapes du cycle de développement logiciel : analyse, modélisation UML, conception de la base de données, développement et tests. En réponse à la problématique initiale, notre application offre une solution fiable, intuitive et évolutive pour améliorer la productivité des pharmacies et la qualité du service offert aux patients.

Plusieurs pistes d'amélioration peuvent enrichir l'application.

Tout d'abord, l'exploitation des données de vente à des fins **statistiques et épidémiologiques** pourrait permettre d'identifier les maladies les plus courantes et d'appuyer la recherche médicale. Des rapports comparatifs et graphiques offriraient aux autorités de santé des indicateurs pertinents pour anticiper les besoins et suivre les tendances locales.

Une autre évolution envisagée serait l'intégration d'un **module de suggestion automatique de médicaments génériques**, permettant au pharmacien de proposer instantanément des alternatives thérapeutiques équivalentes, favorisant ainsi des choix plus économiques et une meilleure gestion des stocks.

Sur le plan institutionnel, l'ajout d'un **système de communication directe avec les organismes sociaux** (CNAS, CASNOS) permettrait d'automatiser la transmission des ventes remboursables, réduisant les délais de paiement, les fraudes et la charge administrative.

Enfin, la mise en place d'un **module de gestion des commandes fournisseurs** offrirait un meilleur contrôle sur les approvisionnements, les coûts et la rentabilité. Ces évolutions, qu'elles soient fonctionnelles ou techniques, visent à transformer l'application en une plateforme complète, intelligente et interconnectée, au service de la modernisation du secteur pharmaceutique algérien.

Bibliographie

[1] *M.Bochu, création d'un site internet d'une officine de pharmacie : Analyse de faisabilité. Thèse de doctorat en pharmacie, Université Joseph Fourier - Faculté de Pharmacie de Grenoble, 2004.*

[2] *https ://pharmax.dz/* (Consulté le 10/04/2025)

[3] *https ://agilemanifesto.org/* (Consulté le 02/03/2025)

[5] *https ://developer.ibm.com/articles/an-introduction-to-uml/* (Consulté le 17/04/2025)

[6] *Modélisation objet avec UML, Pierre-Alain Muller, 1997.*

[7] *https ://www.labri.fr/perso/domenger/Cours/UML.pdf* (Consulté le 05/03/2025)

[8] *https ://fr.wikipedia.org/wiki/Client-serveur* (Consulté le 04/06/2025)

[9] *https ://www.geonov.fr/architecture-client-serveur/* (Consulté le 04/06/2025)

[10] *https ://fr.wikipedia.org/wiki/Modèle-vue-côntroleur* (Consulté le 09/06/2025)

Résumé

Le mémoire que nous avons présenté porte sur la conception et la réalisation d'une application web pour gérer une pharmacie. Cette application, nommée **Pharmadise**, a pour objectif de faciliter le travail du pharmacien en automatisant les tâches liés à la gestion du stock, des ventes, des clients, des vendeurs et des produits remboursés par les organismes de sécurité sociale tels que la CNAS et la CASNOS.

Pharmadise propose une interface conviviale avec une distinction claire entre les rôles du pharmacien et du vendeur, chacun disposant d'un accès adapté à ses fonctionnalités. L'application permet notamment d'enregistrer une vente et l'archiver dans l'historique du client, identifier les produits en voie de péremption ou en rupture de stock et d'assurer une gestion rigoureuse des lots de produits pharmaceutiques.

Ce projet a été développé en suivant la méthode Scrum, jugé plus adapté à la dynamique de travail adopté. Pour la modélisation, nous avons utilisé le langage **UML** (Unified Modeling Language). L'implémentation de l'application repose sur des technologies modernes telles que **Node.js, HTML, CSS, JavaScript, Bootstrap** et une base de données **MongoDB**.

Mots clés : Application web, pharmacie, gestion de stock, Node.js, HTML, CSS, JavaScript, Bootstrap, UML, MongoDB, Scrum.

Abstract

The dissertation we have presented focuses on the design and implementation of a web application for managing a pharmacy. This application, named **Pharmadise**, aims to facilitate the pharmacist's work by automating tasks related to stock management, sales, clients, vendors and products reimbursed by social security organizations such as CNAS and CASNOS.

Pharmaduse offers a user-friendly interface with a clear distinction between the roles of pharmacist and vendor, each having access to features suited to their responsabilities. The application allows, among other things, to record a sale and archive it in the client's history, identify products nearing expiration or out of stock and ensure management of pharmaceutical product batches.

This project was developed following the Scrum model, considered more suitable for thee adopted work dynamic. For modeling, we used the **UML** (Unified Modeling Language). The application was implented using modern technologies such as **Node.js, HTML, CSS, JavaScript, Bootstrap** and an **MongoDB** database.

Keywords : Web application, Pharmacy, stock management, Node.js, HTML, CSS, JavaScript, Bootstrap, UML, MongoDB, Scrum .