

RÉSUMÉ

Le secteur pharmaceutique évolue rapidement, nécessitant des outils innovants pour améliorer la gestion des soins de santé. Notre plateforme de gestion pharmaceutique vise à simplifier et à sécuriser la gestion des stocks, les réservations des patients, les dossiers médicaux électroniques, et les réclamations. Destinée aux pharmaciens, médecins, et patients, notre application offre une interface conviviale permettant de gérer efficacement les prescriptions électroniques, les rapports médicaux, et les interactions médicamenteuses. Avec un système intégré de vérification des interactions, nous assurons la sécurité des traitements pour les patients. La plateforme propose également un module de magazine interactif, où les utilisateurs peuvent échanger sur divers sujets de santé, commenter des publications et les rechercher par sujet ou utilisateur. Notre solution permet ainsi une gestion complète et sécurisée des soins pharmaceutiques, tout en favorisant la communication et la sensibilisation à la santé.

ABSTRACT

The pharmaceutical sector is evolving rapidly, requiring innovative tools to improve healthcare management. Our pharmaceutical management platform aims to simplify and secure inventory management, patient booking, electronic medical records and claims. Designed for pharmacists, physicians and patients, our application offers a user-friendly interface for efficient management of electronic prescriptions, medical reports and drug interactions. With an integrated interaction verification system, we ensure treatment safety for patients. The platform also features an interactive magazine module, where users can discuss various health topics, comment on publications and search for them by subject or user. Our solution thus enables complete, secure pharmaceutical care management, while promoting communication and health awareness.

Table des matières

1	Introduction générale	1
1	Contexte général de projet	3
1.1	Introduction	4
1.2	Présentation de l'organisme d'accueil	4
1.2.1	Les services	4
1.2.2	Les secteurs d'étude de cas	5
1.3	Présentation du projet	5
1.4	Étude de l'existant	6
1.4.1	AmWell	6
1.4.2	DPM	7
1.5	Problématique	7
1.6	Solution Proposée	8
1.7	Méthodologie adoptée	9
1.7.1	Méthodologies Agiles	9
1.7.2	Pourquoi Opter pour la Méthode SCRUM?	10
1.7.3	Les rôles dans SCRUM	10
1.7.4	Les Artéfacts SCRUM	11
1.7.5	Les événements SCRUM	11
1.8	Conclusion	13
2	Sprint 0 : Préparation et Planification	14
2.1	Introduction	15

2.2	Identification des acteurs	15
2.3	Architecture	15
2.3.1	Architecture physique	15
2.3.2	Architecture logique	16
2.4	Identification des besoins	17
2.4.1	Besoins fonctionnels	17
2.4.2	Besoins non fonctionnels	18
2.5	Méthode SCRUM	19
2.5.1	Product Backlog	19
2.5.2	Planning des Sprints	27
2.6	Le diagramme de cas d'utilisation Globale	28
2.7	Diagramme de classe d'analyse	29
2.8	Environnement de travail	30
2.8.1	Environnement matériel	30
2.8.2	Environnement logiciel	30
2.9	Le choix des technologies	32
2.9.1	Frameworks et technologies utilisées	32
2.9.2	Choix de la base de données	33
2.10	Conclusion	34
3	Sprint 1 : Gestion des utilisateurs	35
3.1	Introduction	36
3.2	Sprint Backlog 1	36
3.3	Partie Analyse	39
3.3.1	Diagramme de cas d'utilisation raffiné "Authentification"	40
3.3.2	Diagramme de cas d'utilisation "gestion des utilisateurs"	40

3.3.3	Diagramme de classe de conception relatif à l'authentification et gestion des utilisateurs :	41
3.4	Partie Conception	41
3.4.1	Diagramme de séquence "authentification"	42
3.4.2	Diagramme de séquence gestion profil	43
3.4.3	Description textuelle de cas d'utilisation "S'authentifier"	43
3.5	Partie Réalisation	44
3.5.1	l'interface d'inscription	44
3.5.2	Détails d'un utilisateur	44
3.5.3	Liste des utilisateurs	45
3.5.4	Assignment des rôles	45
3.5.5	Modifier un utilisateur	46
3.5.6	Ajouter un utilisateur	46
3.6	Conclusion	47
4	Sprint 2 : Gestion du Pharamcie	48
4.1	Introduction	49
4.2	Sprint Backlog 2	49
4.3	Partie Analyse	56
4.3.1	Diagramme de cas d'utilisation globale gestion du pharamcie	57
4.3.2	Diagramme de classe de conception niveau "Gestion Pharmacie"	58
4.3.3	Diagramme de cas d'utilisation raffiné "Gérer les stocks des médicaments"	59
4.3.4	Diagramme de cas d'utilisation raffiné "Gérer les réclamations"	59
4.3.5	Diagramme de cas d'utilisation raffiné "Gérer les réservations"	60
4.3.6	Diagramme de cas d'utilisation raffiné "Gérer ses réclamations"	60
4.3.7	Diagramme de cas d'utilisation raffiné "Réserver médicaments"	61

4.3.8	Diagramme de cas d'utilisation raffiné "Gérer médicaments"	61
4.4	Partie Conception	62
4.4.1	Diagramme de séquence "Traiter status"	62
4.4.2	Diagramme de séquence "Réserver médicaments"	62
4.4.3	Description textuelle de cas d'utilisation "Ajouter médicaments"	63
4.5	Partie Réalisation	63
4.5.1	Affichage de liste des médicaments	63
4.5.2	Traitement des réclamations	64
4.5.3	Suppression des réclamations	65
4.5.4	Traitement d'une réclamation	65
4.5.5	Affichage du liste des laboratoires en Tunisie	66
4.6	Conclusion	66
5	Sprint 3 : Gestion des modèles graphiques	67
5.1	Introduction	68
5.2	Sprint Backlog 3	68
5.3	Partie Analyse	69
5.3.1	Diagramme de cas d'utilisation raffiné "Visualisation des modèles graphiques"	69
5.4	Partie Conception	69
5.4.1	Diagramme de séquence "Visualisation des modèles graphiques"	69
5.4.2	Description textuelle de cas d'utilisation "Visualiser les modèles graphiques"	70
5.5	Partie Réalisation	71
5.5.1	Affichage des modèles graphiques	71
5.6	Conclusion	72

6	Sprint 4 : Gestion des dossiers médicaux	73
6.1	Introduction	74
6.2	Sprint Backlog 4	74
6.3	Partie Analyse	79
6.3.1	Diagramme de cas d'utilisation global : "Gestion dossiers/rapports/Ordonnances"	79
6.3.2	Diagramme de cas d'utilisation raffiné "Gestion des dossiers"	80
6.3.3	Diagramme de cas d'utilisation raffiné "Rédaction du rapport"	80
6.4	Partie Conception	81
6.4.1	Diagramme de classe de conception au niveau "Gestion des dossiers médicaux"	81
6.4.2	Diagramme de séquence "Consulter le dossier médical"	82
6.4.3	Description textuelle de cas d'utilisation "Ajouter pièces jointes"	82
6.5	Partie Réalisation	83
6.5.1	Interface liste des dossiers	83
6.5.2	Gestion du dossier d'un patient	84
6.5.3	Affichage de son ordonnance	85
6.5.4	Téléchargement et impression de l'ordonnance	85
6.6	Conclusion	86
7	Sprint 5 : Gestion du Magazine médicale	87
7.1	Introduction	88
7.2	Sprint Backlog 5	88
7.3	Partie Analyse	91
7.3.1	Diagramme de cas d'utilisation général niveau "Magazine"	91
7.3.2	Diagramme de cas d'utilisation raffiné niveau "Gestion des articles"	92
7.3.3	Diagramme de cas d'utilisation raffiné niveau "Commenter les articles"	92

7.4	Partie Conception	93
7.4.1	Diagramme de classe de conception niveau "Gestion magazine"	93
7.5	Partie Réalisation	93
7.5.1	Partage des publications	93
7.5.2	Publications et Commentaires	94
7.5.3	Gestion des publications	95
7.6	Conclusion	95
8	Sprint 6 : Interaction médicamenteuse	96
8.1	Introduction	97
8.2	Choix du système d'apprentissage	97
8.2.1	Justification du choix	98
8.3	Mise en place et implémentation du code	98
8.4	Implémentation du code	98
8.4.1	Importation des bibliothèques python nécessaires	98
8.4.2	Chargement et échantillonnage du dataset	99
8.4.3	Définition de la classe DrugDataset	99
8.4.4	Initialisation du tokenizer et du modèle DistilBERT	100
8.4.5	Division des données	100
8.4.6	Définition des métriques d'évaluation	100
8.4.7	Paramètres de formation du modèle	101
8.4.8	Initialisation du Trainer et boucle de formation	101
8.5	Analyse comparative des résultats	102
8.6	Résultat final obtenu	102
8.7	Conclusion	103

9	Conclusion générale	104
----------	----------------------------	------------

Table des figures

1.1	Logos de Piterion et Neuron-Universe	4
1.2	Interface AmWell	6
1.3	Interface DPM	7
1.4	Schéma conceptuel global	9
1.5	Diagramme Gantt	12
1.6	Méthode Scrum	13
2.1	Architecture Physique	16
2.2	Architecture Logique	16
2.3	Diagramme complet de cas d'utilisation	28
2.4	Diagramme de classe d'analyse	29
2.5	Logo Visual Studio Code	30
2.6	Logo IntelliJ	30
2.7	Logo Pycharm	31
2.8	Logo Git	31
2.9	Logo Postman	31
2.10	Logo Swagger	32
2.11	Logo Angular	32
2.12	Logo Springboot	33
2.13	Logo Keycloak	33
2.14	Logo PostgreSQL	34
3.1	Diagramme de cas d'utilisation authentification	40

3.2	Diagramme de cas d'utilisation gestion des utilisateurs	40
3.3	Diagramme de classe niveau authentification et gestion utilisateurs	41
3.4	Diagramme de séquence authentification	42
3.5	Diagramme de Séquence Gestion Profil	43
3.6	Réalisation de l'interface d'inscription	44
3.7	détails d'un utilisateur	44
3.8	Liste des utilisateurs	45
3.9	Interface d'assignation des rôles	45
3.10	Interface gestion des utilisateurs	46
3.11	Interface d'ajout d'utilisateur	46
4.1	Diagramme de cas d'utilisation générale "gestion du pharmacie"	57
4.2	Diagramme de classe de conception "Gestion Pharmacie"	58
4.3	Diagramme de cas d'utilisation "Gestion des stocks des médicaments"	59
4.4	Diagramme de cas d'utilisation "Gestion des réclamations"	59
4.5	Diagramme de cas d'utilisation "Gestion des réservations"	60
4.6	Diagramme de cas d'utilisation "Gestion de ses réclamations"	60
4.7	Diagramme de cas d'utilisation "Réservation des médicaments"	61
4.8	Diagramme de cas d'utilisation "Gérer médicaments"	61
4.9	Diagramme de séquence "Traiter staus"	62
4.10	Diagramme de séquence "Réserver médicaments"	62
4.11	Gestion et consultation des médicaments	64
4.12	Traitement du réclamation	64
4.13	Suppression des récalamtions	65
4.14	Succès de traitement d'une réclamation	65
4.15	Les laboratoires en Tunisie	66

5.1	Diagramme de cas d'utilisation "Visualisation des modèles graphiques"	69
5.2	Diagramme de séquence "Visualisation des modèles graphiques"	70
5.3	modèles graphiques pour l'administrateur	71
6.1	Diagramme de cas d'utilisation "Gestion des dossiers/Rapports/Ordonnances" .	79
6.2	Diagramme de cas d'utilisation "Gestion des dossiers médicaux"	80
6.3	Diagramme de cas d'utilisation "Rédaction du rapport"	80
6.4	Diagramme de classe de conception niveau "Gestion des dossiers médicaux" . .	81
6.5	Diagramme de sequence "Consulter le dossier médical"	82
6.6	Liste des dossiers médicaux	83
6.7	Dossier médical d'un patient	84
6.8	Affichage de l'ordonnance	85
6.9	téléchargement et impression de l'ordonnance	85
7.1	Diagramme de cas d'utilisation niveau "Gestion du magazine"	91
7.2	Diagramme de cas d'utilisation raffiné niveau "Gestion des articles"	92
7.3	Diagramme de cas d'utilisation raffiné niveau "Commenter les articles"	92
7.4	Diagramme de classe niveau "Gestion du magazine"	93
7.5	Partage des publications	93
7.6	Publications et Commentaires	94
7.7	Gestion des publications	95
8.1	Comparaison entre "ML" et "DL"	97
8.2	Importation des bibliothèques	99
8.3	Chargement et échantillonnage	99
8.4	Classe DrugDataset	99
8.5	Initialisation du tokenizer et du modèle DistilBERT	100

8.6	Initialisation du tokenizer et du modèle DistilBERT	100
8.7	Métriques d'évaluation	100
8.8	Paramètres de formation du modèle	101
8.9	Initialisation du Trainer	101
8.10	1er exemple du résultat	102
8.11	2ème exemple du résultat	102

Liste des tableaux

2.1	Product Backlog	26
2.2	Sprint Planning	27
2.3	Fiche technique de l'ordinateur portable utilisé	30
3.1	Sprint 1 Backlog	39
3.2	Description textuelle du cas d'utilisation : S'authentifier	43
4.1	Sprint 2 Backlog	56
4.2	Description textuelle du cas d'utilisation : Ajouter médicaments	63
5.1	Sprint 3 Backlog	68
5.2	Description textuelle du cas d'utilisation : Ajouter médicaments	70
6.1	Sprint 4 Backlog	79
6.2	Description textuelle du cas d'utilisation : Ajouter pièce jointe	83
7.1	Sprint 5 Backlog	91

Table des abreviations

IT	: Information Technology
PLM	: Product Life Cycle
SSO	: Single-Sign-On
JWT	: Json Web Token
DME	: Dossiers Médicaux Electroniques
NLP	: Natural Language Processing
AI	: Artificial Intelligence
DL	: Deep Learning
API	: Application Programming Interface
IDE	: Integrated Development Environment/ Environnement de développement intégré
ML	: Machine Learning
PO	: Project owner
PM	: Project Manager
SPA	: Single Page Applications
SQL	: Structured Query Language
UI	: User Interface
UX	: User Experience

Introduction générale

Dans le paysage technologique actuel, en constante évolution, des entreprises comme Neuron-Universe doivent relever le défi de s'adapter à la dynamique changeante du secteur tout en garantissant à leurs clients une prestation de services de la plus haute qualité. Dans le secteur pharmaceutique, où la précision et l'innovation sont primordiales, Neuron-Universe s'est lancé dans un voyage de transformation pour développer une plateforme pharmaceutique personnalisée de pointe. Cette plateforme vise à révolutionner la recherche et le développement pharmaceutiques, en fournissant des solutions sur mesure pour répondre aux besoins uniques de chaque client.

En moins de trois ans, Neuron-Universe a connu un succès remarquable dans le développement d'une gamme variée de solutions logicielles. Ces solutions englobent un éventail de services, notamment des systèmes de gestion des relations avec la clientèle et des systèmes spécialisés de planification des ressources de l'entreprise. Cependant, à mesure que l'entreprise élargissait son portefeuille, elle s'est heurtée à un problème de gestion de l'accès pour les clients ayant acquis plusieurs solutions.

Pour relever ce défi et améliorer l'expérience des utilisateurs, Neuron-Universe a été chargée de créer une solution distincte pour la gestion complète d'un établissement pharmaceutique. L'accent a également été mis sur l'implémentation du machine learning ou du deep learning et sur l'intégration de cette solution de manière transparente, en rationalisant l'accès et en offrant une expérience unifiée aux utilisateurs.

Ce rapport se compose de huit chapitres qui se présentent comme suit :

Dans le premier chapitre, nous décrivons le cadre général de l'organisme d'accueil ainsi que la thématique du sujet et la méthodologie adoptée.

Dans le second chapitre «Sprint 0», nous abordons la phase initiale de préparation et de planification de notre projet. Nous posons les bases de notre plateforme pharmaceutique en préparant l'environnement matériel et logiciel, en choisissant les technologies et la base de données, et en identifiant les acteurs du système. Nous décrivons l'architecture physique et logique, les besoins fonctionnels et non fonctionnels, ainsi que les éléments du Product Backlog et le planning des sprints. À travers des diagrammes d'utilisation et de classes, nous offrons une vue d'ensemble du système, assurant ainsi une préparation complète avant le lancement du développement.

Durant le troisième chapitre «Sprint 1», nous nous concentrons sur la gestion des utilisateurs de la plateforme. Nous commençons par définir le Sprint Backlog, puis analysons les fonctionnalités d'authentification et de gestion des profils à travers des diagrammes d'utilisation et de classes. La partie conception détaille les séquences d'authentification et de gestion de profil, ainsi que la description textuelle du cas d'utilisation "S'authentifier". Nous terminons par la réalisation de l'interface. Ce sprint pose les fondations de la gestion sécurisée des utilisateurs dans le système.

Passant au quatrième chapitre «Sprint 2», centré sur la gestion de la pharmacie, nous analysons les principales fonctionnalités à l'aide des diagrammes de cas d'utilisation. La conception inclut des diagrammes de séquence et une description textuelle, et nous terminons avec la réalisation de l'interface utilisateur. Ce sprint permet de structurer la gestion des ressources et des interactions clients pour la pharmacie.

Dans le cinquième chapitre «Sprint 3», consacré à la gestion et à la visualisation des modèles graphiques dans l'application, nous définissons le Sprint Backlog et analysons le cas d'utilisation pour la visualisation des modèles graphiques à travers des diagrammes détaillés. La conception illustre les séquences d'interactions nécessaires pour afficher ces modèles graphiques, accompagnées d'une description textuelle. La réalisation consiste en l'implémentation de l'interface d'affichage, permettant à l'utilisateur de visualiser efficacement les données.

Passant au sixième chapitre «Sprint 4», nous explorons la gestion des dossiers médicaux, des rapports et des ordonnances. Nous présentons le Sprint Backlog, analysons les cas d'utilisation, et concevons des diagrammes de classes et de séquence, accompagnés d'une description textuelle. La phase de réalisation inclut quelques interfaces implémentées. Ce sprint vise à structurer le module des dossiers médicaux pour une gestion centralisée et efficace des informations.

Pour le septième chapitre «Sprint 5», dédié à la gestion du magazine médical, nous analysons les cas d'utilisation, concevons un diagramme de classes décrivant la structure de ce module, et réalisons une interface utilisateur illustrée par des captures. Ce sprint enrichit la plateforme avec un espace interactif où les utilisateurs peuvent échanger des informations médicales et participer activement aux discussions.

Enfin, le huitième chapitre «Sprint 6» porte sur la prédiction des interactions médicamenteuses à l'aide d'un modèle d'apprentissage automatique. Nous présentons le choix et la justification du système d'apprentissage, avant de décrire l'implémentation du code, de l'importation des bibliothèques nécessaires à l'échantillonnage des données, et à la définition de la classe de gestion des données. Nous terminons par une analyse des résultats obtenus et une discussion sur les performances du modèle. Ce sprint marque une étape cruciale pour renforcer la sécurité des prescriptions grâce à la prédiction des interactions médicamenteuses.

Chapitre 1

Contexte général de projet

1.1	Introduction	4
1.2	Présentation de l'organisme d'accueil	4
1.2.1	Les services	4
1.2.2	Les secteurs d'étude de cas	5
1.3	Présentation du projet	5
1.4	Étude de l'existant	6
1.4.1	AmWell	6
1.4.2	DPM	7
1.5	Problématique	7
1.6	Solution Proposée	8
1.7	Méthodologie adoptée	9
1.7.1	Méthodologies Agiles	9
1.7.2	Pourquoi Opter pour la Méthode SCRUM ?	10
1.7.3	Les rôles dans SCRUM	10
1.7.4	Les Artéfacts SCRUM	11
1.7.5	Les événements SCRUM	11
1.8	Conclusion	13

1.1 Introduction

Ce premier chapitre met l'accent sur la présentation du cadre global du projet, un aspect essentiel pour en comprendre le contexte.

out d'abord, nous présenterons l'organisation hôte, **Neuron-Universe**, filiale du **Piterion Group**, puis nous décrirons le projet. Ensuite, nous discuterons des différentes étapes ayant conduit à la réalisation du projet.

Cela sera fait en analysant la problématique, en examinant une étude de l'existant et en proposant des solutions adéquates.

Enfin, nous détaillerons la méthode employée pour concevoir et implémenter ce projet.

1.2 Présentation de l'organisme d'accueil



FIGURE 1.1 – Logos de Piterion et Neuron-Universe

Neuron-Universe est une startup spécialisée dans la communication de confiance, qui développe des solutions logicielles intégrées afin de minimiser les tâches administratives, rendre les processus d'entreprise plus efficaces et offrir aux utilisateurs davantage de temps pour se concentrer sur ce qui est essentiel. Située à La Marsa, Neuron-Universe appartient au groupe Piterion, un fournisseur international de services en PLM, IT, et ingénierie.

Le groupe se trouve au point de convergence de trois domaines de services, développant des solutions dynamiques pour une grande variété d'industries. Fondé en 2002 à Stuttgart, en Allemagne, le groupe Piterion a son siège social en Allemagne, ainsi qu'une succursale en Tunisie, une société étrangère privée implantée au Pôle Technologique El Ghazela.

1.2.1 Les services

- **PLM "Product Life Cycle Management"** : La gestion professionnelle du cycle de vie des produits (PLM) est plus vaste que jamais et offre encore plus de possibilités en matière de numérisation industrielle.
- **Développement des logiciels industriels** : Avec le domaine de service de développement des logiciels industriels, Piterion Group réduit la complexité inutile et garantit une convivialité maximale.

- **Interaction avec le système humain** : Qu’il s’agisse de lunettes de données ou de données machine projetées en temps réel sur l’écran de la tablette : les applications issues de domaines tels que la réalité augmentée et virtuelle ou l’intelligence artificielle ne jouent pas seulement un rôle de plus en plus important dans le cycle de vie des produits. Piterion Group garantit une adéquation parfaite entre l’homme et la machine, ainsi qu’une intégration optimale dans les systèmes existants.
- **Opérations** : Qu’il s’agisse d’un problème avec un jumeau numérique, d’un dysfonctionnement des lunettes de données ou d’une panne du système : avec une assistance du niveau 2 au niveau 4 ainsi que différents modèles d’assistance mondiaux, sur site, hors site, offshore et nearshore. Piterion Group [**piterion**] garantis vous d’une maintenance et d’un entretien précis et fiables de toutes les applications et composants pertinents pour les produits des entreprises. Piterion Group [**piterion**] résout les erreurs et les dysfonctionnements rapidement et durablement pour un fonctionnement fluide qui exploite toujours tout le potentiel des systèmes industriels, et rend ainsi l’investissement encore plus rentable.

1.2.2 Les secteurs d’étude de cas

Piterion travaille sur plusieurs secteurs d’étude de cas, notamment :

- **Médical** : Recherche des solutions innovantes dans le domaine de la santé.
- **Aérospatiale et Défense** : Développement de technologies avancées pour l’aéronautique et la sécurité.
- **Transport** : Études sur l’amélioration des systèmes de transport et leur efficacité.
- **Agriculture** : Innovations pour améliorer les pratiques agricoles et la gestion des ressources.
- **Automobile** : Solutions liées à l’industrie automobile et à la mobilité durable.
- **Energie** : Études sur la gestion de l’énergie et les énergies renouvelables.
- **Electronique** : Développement de produits électroniques avancés pour divers secteurs.

Ces secteurs sont au cœur des projets de l’entreprise, qui cherche à apporter des solutions innovantes et durables dans chaque domaine.

1.3 Présentation du projet

Notre projet se concentre sur le développement de Pharma, une plateforme pharmaceutique personnalisée et intégrée qui vise à améliorer la gestion des pharmacies et à faciliter la collaboration entre pharmaciens, patients et professionnels de santé. Mais comment fonctionne exactement cette plateforme et quels sont ses principaux atouts ?

- **La plateforme PHARMA** est une solution complète de gestion pharmaceutique, conçue pour permettre aux pharmacies de gérer efficacement leurs stocks, les médicaments, ainsi que les interactions avec les patients et le personnel médical. Cette plateforme se décline en plusieurs modules interconnectés pour répondre aux besoins spécifiques des pharmaciens, des médecins, des secrétaires médicaux et des patients.
- **Un module de Deep Learning** est intégré pour assurer la vérification des interactions médicamenteuses. En s'appuyant sur un modèle d'apprentissage automatique, basé sur un dataset fourni par le ministère de la Santé Tunisienne, cette fonctionnalité permet aux professionnels de santé de vérifier rapidement la compatibilité des traitements et d'optimiser la sécurité des patients.

L'objectif principal de Pharma est de transformer les pharmacies en centres de santé modernes, en offrant une plateforme sécurisée et accessible qui répond aux besoins des utilisateurs, tout en optimisant la gestion et la sécurité des médicaments. Grâce à des technologies avancées, Pharma s'engage à offrir une solution personnalisée et performante pour améliorer la santé et le bien-être de ses utilisateurs.

1.4 Étude de l'existant

Dans cette section, nous nous concentrons sur l'analyse des plateformes concurrentes à notre solution, telles que **AmWell**. Les fonctionnalités et les services proposés par cette application seront étudiés et comparés à ceux de **PHARMA**, afin d'identifier les avantages et les points à améliorer.

1.4.1 AmWell

Comme illustré par la figure 1.2, AmWell est une autre plateforme proposant des services pour la télésanté et de création d'un écosystème de soins numériques.

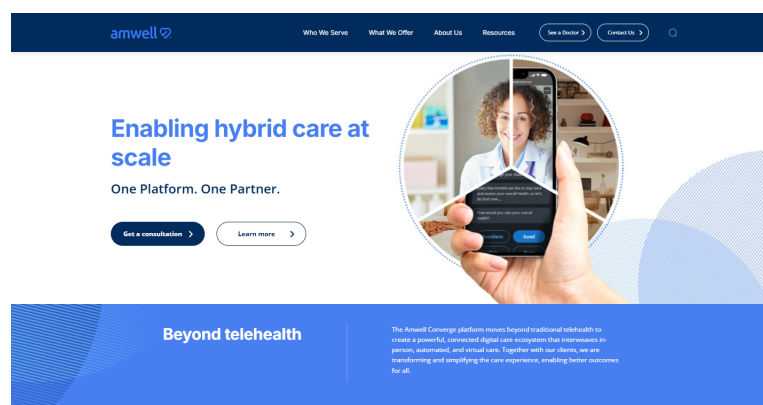


FIGURE 1.2 – Interface AmWell
[2]

- **Avantages :** AmWell[2] est un site web qui aide les utilisateurs à avoir un programme de soins automatisé ainsi qu'il propose également un paiement en ligne pour faire payer les médecins.

De plus, AmWell [2] a un logo simple et classique en utilisant des couleurs constantes.

- **Inconvénients :** pour les inconvénients de Amwell on n'a pas d'accès aux médicaments, pas de réservations des médicaments, pas d'option pour l'interaction médicamenteuse, pas de forums et il n'y a pas des fonctionnalités pour les réclamations pour les patients.

1.4.2 DPM

Comme illustré par la figure 2.2 DPM [3] est une autre plateforme pharmaceutique proposant des services pour la gestion des pharmacies.



FIGURE 1.3 – Interface DPM
[3]

- **Avantages :** DPM est un site web aide les pharmaciens à gérer les aspect administratifs liés à la pharmacie, aux médicaments, réclamations et activités apparentées. Il a aussi un logo simple qui est conforme avec le thème pharmaceutique.
- **Inconvénients :** Pour les inconvénients de DPM le site est pas du tout organisé, il n'y a pas des fonctionnalités pour vérifier l'interaction des médicaments, réserver des médicaments, gérer les stocks des médicaments et consulter les laboratoires médicaux.

1.5 Problématique

La gestion des processus en pharmacie, en collaboration avec les médecins, notamment le suivi des stocks, la gestion des dossiers médicaux, les réservations et les interactions médicamenteuses, constitue un défi majeur pour les professionnels de santé. Face au volume croissant de données et à la nécessité de garantir la sécurité et l'efficacité des traitements, il devient essentiel de disposer d'une solution centralisée, fiable et intuitive. Comment concevoir une plateforme pharmaceutique complète et sécurisée permettant aux pharmaciens, médecins et patients

de gérer efficacement les dossiers médicaux, de vérifier les interactions médicamenteuses et de collaborer autour des soins, tout en assurant une utilisation simple et une gestion optimisée des ressources pharmaceutiques ?

1.6 Solution Proposée

Une approche superficielle ne suffit pas pour relever les défis complexes posés par la gestion pharmaceutique. Une réflexion approfondie est donc indispensable pour concevoir une solution adaptée. Dans cette optique, nous proposons le développement de modules distincts pour une gestion complète d'une plateforme pharmaceutique personnalisée. Cette solution intègre des technologies telles que JWT (JSON Web Tokens) et le Single Sign-On (SSO) pour garantir une connexion fluide et sécurisée entre les différents modules.

PHARMA est une plateforme pharmaceutique personnalisée qui se distingue par son intégration transparente et son objectif de simplifier les tâches des pharmaciens et des médecins, tout en améliorant l'expérience des patients. Elle intègre également des fonctionnalités avancées de Deep Learning (DL) pour analyser et prévenir les interactions médicamenteuses.

Cette plateforme innovante est conçue pour répondre aux divers besoins des professionnels de santé et des patients, en offrant une approche holistique et intelligente de la gestion des médicaments. Grâce à une interface conviviale, les utilisateurs peuvent créer des profils, saisir et gérer leurs informations de manière simple et sécurisée. Ils ont également accès à des rapports et ordonnances médicaux détaillés à tout moment, et peuvent effectuer facilement l'ensemble des processus médicaux, depuis l'inscription jusqu'à la réservation des médicaments, en passant par la vérification des interactions médicamenteuses et de leur compatibilité.

Notre composant utilise le Deep Learning et l'apprentissage automatique pour extraire des informations de la base de données médicamenteuse du Ministère de la Santé tunisien, alertant ainsi les utilisateurs et les prestataires de soins de santé en cas d'interactions médicamenteuses ou de contre-indications potentielles.

De plus, la plateforme "PHARMA" vise à faciliter les tâches des différents professionnels, tels que les médecins et les pharmaciens, en leur permettant de réaliser leurs diverses activités en toute sécurité. Cela inclut la gestion des stocks de médicaments, la gestion des dossiers médicaux électroniques, la publication dans le magazine médical, et la rédaction des rapports médicaux, entre autres.

L'intégration avec les dossiers médicaux électroniques (DME) garantit un accès fluide aux antécédents médicaux, aux diagnostics et aux traitements en cours des patients. Par ailleurs, des mesures de sécurité renforcées et la conformité aux réglementations protègent les données et la vie privée des utilisateurs.

En mettant l'accent sur la précision, la sécurité et l'autonomisation des utilisateurs, cette plateforme représente un outil de transformation qui révolutionne les soins pharmaceutiques et améliore le bien-être des patients.

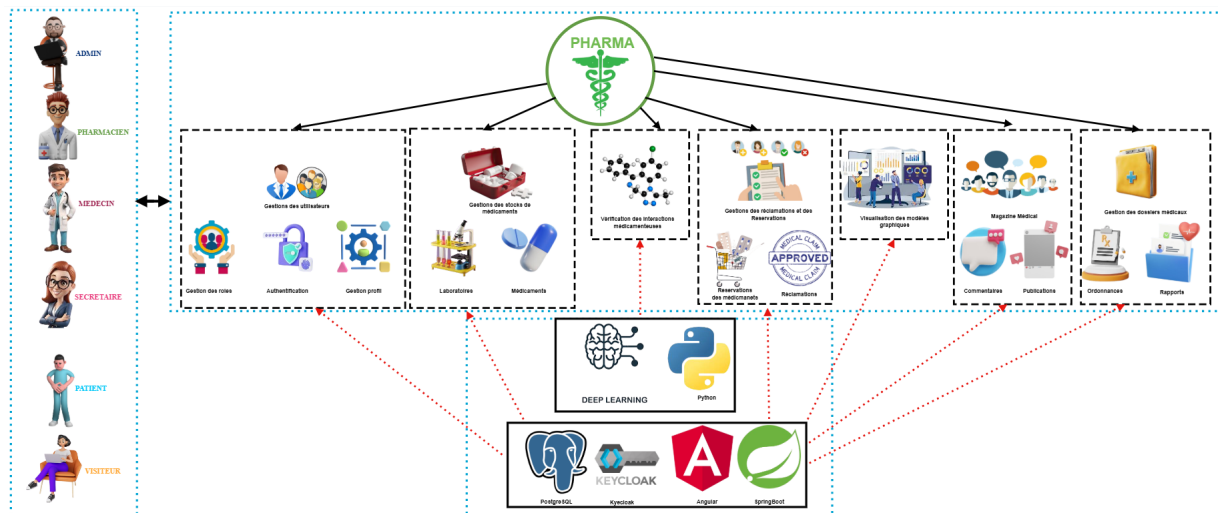


FIGURE 1.4 – Schéma conceptuel global

La figure 1.4, présente le schéma conceptuel pour notre solution proposée qui illustre l'architecture fonctionnelle et technologique de la plateforme pharmaceutique personnalisée "PHARMA". Il met en avant les modules principaux, les rôles des utilisateurs et les technologies utilisées.

1.7 Méthodologie adoptée

Dans notre projet, nous avons choisi d'adopter l'approche Agile, reconnue pour sa flexibilité et sa capacité d'adaptation aux évolutions. Parmi les différentes méthodologies agiles, nous avons opté pour Scrum, un cadre de travail très répandu dans le développement logiciel et d'autres domaines. Scrum repose sur des sprints, des cycles de travail courts, généralement de 2 à 4 semaines, au terme desquels une version utilisable du produit est livrée. Cette méthode nous permet de maintenir un rythme de livraison constant tout en favorisant une adaptation aux besoins évolutifs du projet.

1.7.1 Méthodologies Agiles

Les méthodologies agiles représentent des approches de gestion de projet et de développement logiciel qui mettent l'accent sur la flexibilité, la coopération, et la satisfaction des clients à travers des itérations fréquentes et progressives. Voici quelques-unes des méthodes agiles les plus répandues :

- **Kanban** : Cette méthode vise à optimiser les processus de travail en offrant une vue claire des tâches et en maintenant un flux de travail continu. Elle est utile pour les projets nécessitant une flexibilité accrue et une gestion constante des priorités.
- **Lean** : Basée sur les principes de la production industrielle, la méthode Lean se focalise sur la maximisation de la valeur tout en éliminant les gaspillages. Elle convient bien aux projets nécessitant une efficacité opérationnelle et un processus d'amélioration continue.

- **Extreme Programming** : Cette approche se concentre sur la production de logiciels de qualité supérieure en utilisant des pratiques telles que le développement dirigé par les tests (TDD) et la programmation en binôme. Elle est adaptée aux projets nécessitant une rigueur technique élevée et une collaboration étroite avec le client.
- **Scrum** : Comme l'une des méthodologies agiles les plus populaires, Scrum se focalise sur la livraison d'incréments fonctionnels du produit à des périodes fixes appelées sprints. Elle favorise la collaboration, la souplesse et la responsabilité au sein de l'équipe.

1.7.2 Pourquoi Opter pour la Méthode SCRUM ?

Nous avons sélectionné la méthode SCRUM pour notre projet en raison de ses nombreux avantages :

- **Flexibilité** : SCRUM permet une adaptation rapide aux évolutions, qu'elles concernent les exigences des clients ou les conditions du marché.
- **Livraisons régulières** : Les sprints, qui durent de 2 à 4 semaines, permettent de livrer fréquemment des incréments de produit fonctionnels, garantissant une valeur continue pour le client aussi.
- **Amélioration permanente** : Grâce aux réunions de rétrospective, l'équipe peut régulièrement évaluer et optimiser ses processus et méthodes.
- **Transparence et échanges** : Les réunions quotidiennes de type stand-up et les revues de sprint favorisent une communication transparente et régulière au sein de l'équipe ainsi qu'avec les parties prenantes.
- **Interaction rapprochée avec le client** : Le Product Owner (PO) veille à ce que les retours et les besoins des clients soient intégrés de manière continue dans le développement du produit.

1.7.3 Les rôles dans SCRUM

- **Product Owner (PO)** : Il représente les intervenants ainsi que les clients et se charge de la gestion du backlog produit. Le PO établit les priorités et s'assure que l'équipe se concentre sur les fonctionnalités apportant la meilleure valeur.
- **Scrum Master** : En tant que facilitateur du cadre Scrum, il a pour rôle de veiller à ce que l'équipe respecte les principes de Scrum, surmonte les obstacles rencontrés et améliore son efficacité globale.
- **Équipe de développement** : Formée de développeurs et de spécialistes, cette équipe s'auto-organise et est responsable de livrer un incrément opérationnel à la fin de chaque sprint.

1.7.4 Les Artéfacts SCRUM

- **Product Backlog** : Une liste priorisée de toutes les fonctionnalités, améliorations et corrections à apporter au produit.
- **Sprint Backlog** : Un sous-ensemble du product backlog choisi pour un sprint spécifique, sur lequel l'équipe se concentrera.
- **Incrément** : Le résultat fonctionnel à la fin de chaque sprint, c'est une version du produit utilisable et potentiellement livrable.

1.7.5 Les événements SCRUM

- **Sprint Planning** : Une réunion au début de chaque sprint où l'équipe planifie ce qu'elle va accomplir pendant le sprint à partir du product backlog.
- **Daily Scrum** : Une réunion quotidienne de 15 minutes où l'équipe synchronise ses activités, discute des progrès et identifie les obstacles.
- **Sprint Review** : Une réunion à la fin du sprint pour examiner le travail accompli, obtenir des retours des parties prenantes, et ajuster les priorités.
- **Sprint Retrospective** : Une réunion pour réfléchir sur ce qui s'est bien passé pendant le sprint, ce qui peut être amélioré, et définir des actions d'amélioration pour les prochains sprints.

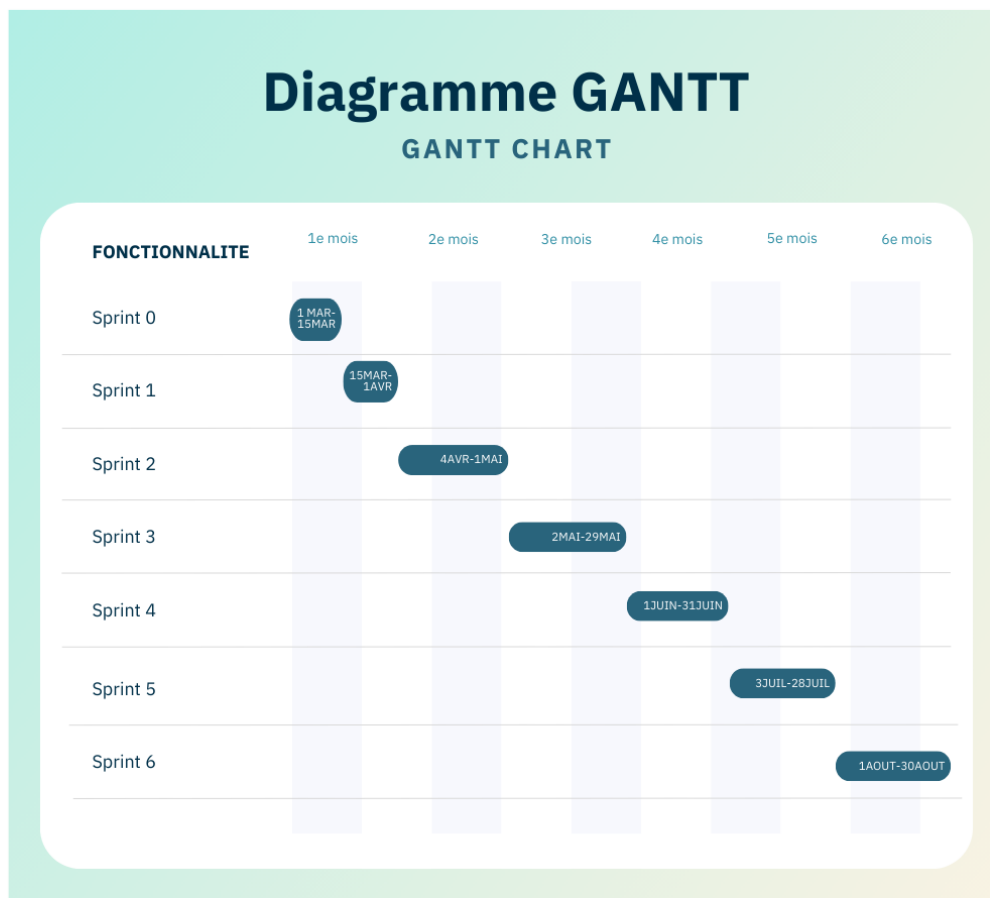


FIGURE 1.5 – Diagramme Gantt

Comme le montre la figure 1.5, le Project Manager, Mr Hafedh DZIRI, a élaboré les user stories, les organisant pour constituer le product backlog. Ainsi, Les tâches ont été priorisées aussi planifiées sur plusieurs sprints.

Comme démontré par la figure 1.6, Agile Scrum repose sur trois piliers essentiels : le Product Owner, responsable de la gestion du backlog et de la priorisation des fonctionnalités ; le Scrum Master, facilitateur qui s'assure que l'équipe respecte les pratiques Scrum et surmonte les obstacles et l'équipe de développement, auto-organisée, chargée de livrer un incrément fonctionnel à chaque sprint. Ces rôles collaborent étroitement pour assurer l'adaptabilité, la livraison continue et la qualité du produit.

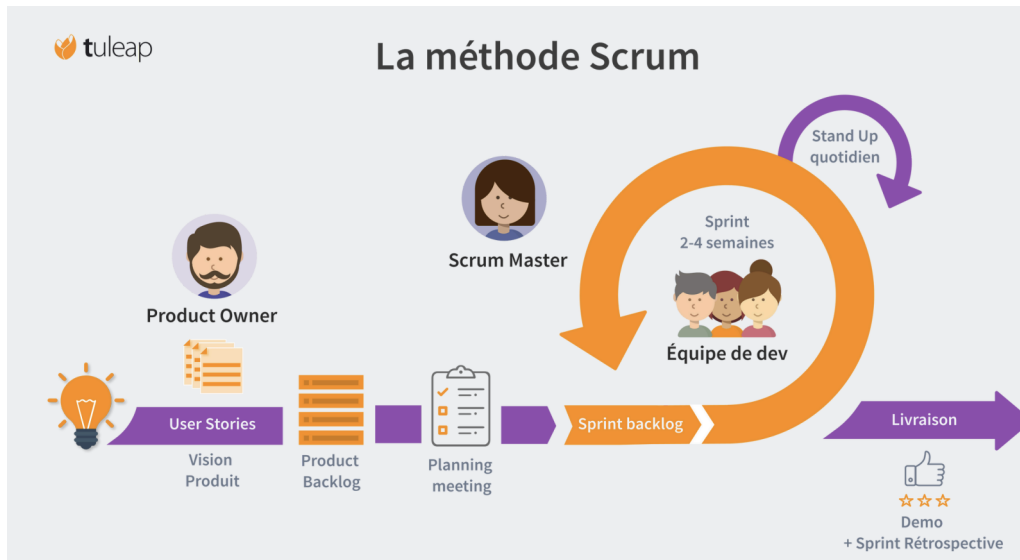


FIGURE 1.6 – Méthode Scrum
[4]

1.8 Conclusion

Dans le premier chapitre, nous avons défini le cadre général du projet, présenté l'entreprise d'accueil, ainsi que les enjeux principaux et la solution proposée. Nous avons examiné la situation actuelle en relation avec la solution proposée. De plus, nous avons présenté l'approche choisie pour assurer le succès du projet.

Chapitre 2

Sprint 0 : Préparation et Planification

2.1	Introduction	15
2.2	Identification des acteurs	15
2.3	Architecture	15
2.3.1	Architecture physique	15
2.3.2	Architecture logique	16
2.4	Identification des besoins	17
2.4.1	Besoins fonctionnels	17
2.4.2	Besoins non fonctionnels	18
2.5	Méthode SCRUM	19
2.5.1	Product Backlog	19
2.5.2	Planning des Sprints	27
2.6	Le diagramme de cas d'utilisation Globale	28
2.7	Diagramme de classe d'analyse	29
2.8	Environnement de travail	30
2.8.1	Environnement matériel	30
2.8.2	Environnement logiciel	30
2.9	Le choix des technologies	32
2.9.1	Frameworks et technologies utilisées	32
2.9.2	Choix de la base de données	33
2.10	Conclusion	34

2.1 Introduction

Dans ce chapitre, nous allons explorer le sprint initial, ou phase de lancement du projet. Nous aborderons l'architecture globale de notre application, les technologies sélectionnées pour accomplir l'ensemble des tâches, ainsi que les langages de programmation utilisés tout au long du processus de développement.

2.2 Identification des acteurs

Dans notre projet, de nombreux acteurs interagissent avec notre système :

- **Administrateur** :Celui qui a une vue d'ensemble de toute application. Celui qui possède une vision globale de toute l'application. La réalisation de ce rôle offre des avantages considérables par rapport aux autres utilisateurs, offrant ainsi des possibilités supplémentaires comme la gestion des profils d'utilisateur.
- **Médecins** :Les médecins examinent méticuleusement les recommandations générées par l'IA, en s'assurant qu'elles correspondent au profil de santé de chaque patient. Ils supervisent la gestion des ordonnances, des rapports et des dossiers médicaux pour suivre les progrès des patients et répondent rapidement à tout événement indésirable ou préoccupation.
- **Patients** : Ils jouent un rôle actif dans leurs soins de santé en consultant leurs ordonnances, rapports et dossiers médicaux, en réservant les médicaments, en vérifiant les interactions entre les médicaments prescrits par leur médecin et en faisant des réclamations auprès de la pharmacie en cas de problème.
- **Sécretares** : Ses principales missions consistent à gérer les dossiers, les ordonnances et les rapports des patients.
- **Pharmaciens** : Au sein du pharmacie, les pharmaciens fournissent leur expertise en matière de gestion des médicaments en gérant les médicaments et leurs stocks, ainsi qu'en gérant les réservations et les réclamations des patients.
- **Visiteur** : cet acteur peut simplement visiter notre Plateforme et consulter l'accueil.

2.3 Architecture

2.3.1 Architecture physique

Comme illustré dans la figure 2.1, cette structure est composée de trois niveaux : le backend, le front-end et le portail API. La partie front-end d'Angular (port 4200) est en communication avec l'API Gateway (port 8098) afin de communiquer avec les microservices du backend. L'authentification et l'autorisation sont assurées par Keycloak (port 8180), tandis que les services sont découverts par Eureka server (port 8761), dont le microservice Authentification (port

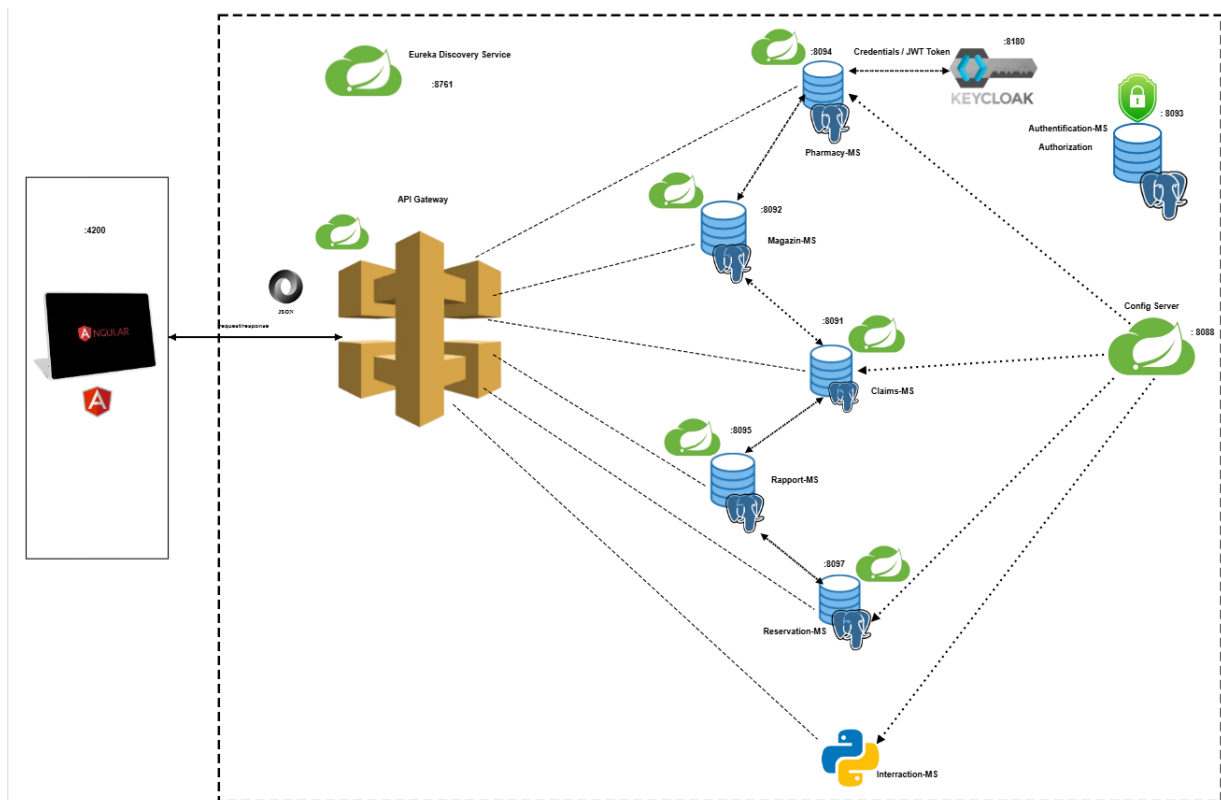


FIGURE 2.1 – Architecture Physique

8093), le microservice de gestion des utilisateurs (port 8081), le microservice Pharmacie (port 8094), le microservice Magazine (port 8092), le microservice Rapport (port 8095), le microservice Réclamation (port 8091) et le microservice réservation (port 8097).

2.3.2 Architecture logique

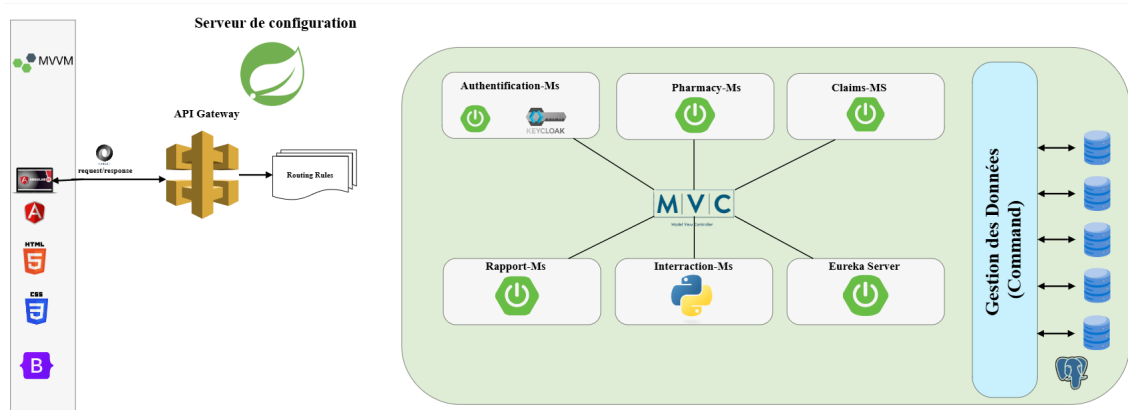


FIGURE 2.2 – Architecture Logique

Comme illustré par la figure 2.2, cette structure logicielle intègre notre application Angular, qui est structurée en différentes couches de modèles, de composants et de services afin de rendre l'interface utilisateur et la logique de communication plus structurées. Les requêtes sont

transmises par l'API Gateway aux microservices back-end. Ce derniers font appel à Spring pour concevoir des services REST, des services métier, et Spring Data pour accéder aux informations. L'authentification est assurée par Keycloak, tandis que la découverte des services est facilitée par Eureka Discovery Server, ce qui permet d'améliorer la communication et la gestion des microservices.

L'architecture repose sur les patrons de conception MVC et MVVM : le modèle MVC est appliqué côté back-end pour structurer les microservices Spring Boot, en séparant la logique métier, l'accès aux données et les contrôleurs. Côté front-end, le modèle MVVM est adopté avec Angular, permettant une séparation claire entre la vue et la logique métier grâce au ViewModel, facilitant ainsi la gestion de l'interface utilisateur et de la communication avec les services.

2.4 Identification des besoins

2.4.1 Besoins fonctionnels

Les besoins fonctionnels doivent répondre aux exigences du système en termes de fonctionnalité.

- Authentification.
- Gestions du profil.
- Gestion du magazine.
- Gestion du stock des médicaments.
- Gestion du laboratoire.
- Gestion des réclamations.
- Gestion des médicaments.
- Envoyer et recevoir des ordonnances.
- Étiquetage des médicaments.
- Gestion des ordonnances.
- Téléchargement des ordonnances.
- Gestion des dossiers médicaux.
- Gestion des discussions.
- Gestion de réservation des médicaments.
- Interaction médicamenteuse.

2.4.2 Besoins non fonctionnels

Afin de mener à bien notre mission, l'application doit examiner certaines caractéristiques et tenir compte de certaines contraintes et exigences.

Scalabilité

- En choisissant une architecture basée sur les microservices, il devient possible d'abolir la saturation de l'application en cas d'accroissement du trafic ou de l'utilisation, car chaque microservice fonctionne de manière indépendante.

Sécurité

- **1. Contexte de sécurité :** Les demandes d'objets critiques ciblant les points d'extrémité d'une API REST sécurisée par Spring Security peuvent avoir à fournir une preuve d'authentification. Afin de vérifier si un utilisateur est authentifié, Spring Security s'appuie sur le contenu de son "Contexte de Sécurité", qui doit être alimenté par un objet "Authentication". La procédure habituelle lorsqu'une requête cible un point d'accès sécurisé et qu'il n'y a pas d'utilisateur authentifié est de rediriger cet utilisateur vers un formulaire de connexion ou vers la page d'accueil.

2. Gestion de la session Spring Security : D'autres situations peuvent conduire à l'effacement du "Contexte de Sécurité". Ces situations dépendent de la politique de gestion des sessions "Spring Security" choisie par l'utilisateur. Quatre configurations distinctes sont disponibles, chacune ayant un impact différent sur le cycle de vie du "Contexte de Sécurité".

- `SessionCreationPolicy.ALWAYS`
- `SessionCreationPolicy.IF-REQUIRED`
- `SessionCreationPolicy.NEVER`
- `SessionCreationPolicy.STATELESS`

3. Utilisation d'un JWT : Un JWT (Json Web Token) [5] est un objet Json contenant des clés et des valeurs, qui est généralement crypté dans une chaîne de caractères. Un JWT est généralement utilisé pour stocker un nom d'utilisateur, une liste d'autorisations et une date d'expiration. Si un tel objet est attaché à une requête, nous pouvons l'utiliser pour générer une authentification si la date d'expiration est valide.

4. Refresh Token : Les Access Tokens ont généralement un délai d'expiration. Si un utilisateur malveillant met la main sur un Access Token valide, il peut effectuer des demandes au nom d'un utilisateur et accéder à des données protégées. Par conséquent, les Access Tokens expirent au bout d'un certain temps et deviennent invalides. Les Refresh Tokens, en revanche, ont une durée de vie plus longue, de sorte que notre application peut les utiliser pour récupérer un nouveau Access Token, afin d'augmenter la sécurité de notre application.

5. les mots de passe cryptés : Nous avons utilisé `BCryptPasswordEncoder` comme bean dans notre configuration `BCrypt`, cependant, génère en interne un sel aléatoire.

Ergonomie et facilité d'utilisation

- Pour l'interface, nous allons simplifier l'utilisation de l'application où chaque gestion doit être regroupée sur un même module. Aussi, nous adopterons des couleurs qui correspondent à une cohérence visuelle pour rendre l'application plus lisible.

Modularité

- Nous mettrons en œuvre l'architecture microservices qui nous permettra de créer notre application comme une suite de petits services indépendants qui peuvent être développés, déployés et maintenus séparément pour assurer la flexibilité, l'évolutivité et la maintenabilité.

2.5 Méthode SCRUM

2.5.1 Product Backlog

Le tableau 2.1 expose le product backlog de notre projet, décrivant les différentes histoires d'utilisateur, leurs niveaux de priorité et leurs estimations :

ID	Module	ID User Story	User Story	Priorité	Estimation
1	Authentification et Gestion des Utilisateurs	1.1	Comme étant utilisateur, je veux créer un compte pour personnaliser mon expérience. Le processus d'inscription doit être intuitif et me permettre de commencer à utiliser le service rapidement.	1	8
		1.2	Sous l'identité d'administrateur système, je veux pouvoir ajouter de nouveaux administrateurs pour partager la charge de travail et renforcer la sécurité de la plateforme.	2	8
		1.3	Comme utilisateur, je veux pouvoir modifier mes informations personnelles (nom, prénom, email, etc.) à tout moment.	1	5

		1.4	Comme étant administrateur et afin de gérer les comptes utilisateurs, je souhaite pouvoir modifier les profils des utilisateurs.	2	5
		1.5	À titre d'administrateur, je veux pouvoir désactiver les comptes utilisateurs pour restreindre leur accès à la plateforme.	3	3
2	Gestion du Pharmacie	2.1	En tant que pharmacien, je souhaite consulter la liste de tous les laboratoires avec leurs détails (nom, téléphone, adresse web) afin de mieux gérer mes fournisseurs.	3	4
		2.2	Comme pharmacien, je souhaite avoir la possibilité de rechercher un laboratoire en utilisant le nom pour obtenir des informations détaillées sur un laboratoire particulier.	2	4
		2.3	En tant que pharmacien, je veux ajouter des stocks de médicaments afin de maintenir un inventaire à jour.	1	8
		2.4	Comme pharmacien, je souhaite consulter la liste complète des stocks disponibles afin de savoir quels médicaments sont en stock.	2	6
		2.5	En tant que pharmacien, je veux pouvoir consulter un stock spécifique par identifiant ou par nom de laboratoire pour voir ses informations détaillées (nom, quantité, date d'arrivée, etc.).	1	8

		2.6	En tant que pharmacien, je veux pouvoir restocker un médicament dont la date d'expiration approche afin d'éviter une rupture de stock.	1	8
		2.7	Comme pharmacien, je veux pouvoir modifier certaines informations d'un stock spécifique (par exemple, nom, quantité, date d'arrivage) pour corriger ou mettre à jour les données.	2	6
		2.8	En tant que pharmacien, je souhaite ajouter de nouveaux médicaments disponibles de manière simple et efficace afin de tenir à jour l'inventaire.	1	8
		2.9	Comme pharmacien, je veux rechercher un médicament spécifique et consulter ses informations détaillées pour faciliter la gestion des stocks.	1	6
		2.10	En tant que pharmacien, je souhaite modifier ou supprimer un médicament pour ajuster les informations disponibles dans l'inventaire.	1	8
		2.11	Comme pharmacien, je veux vérifier la disponibilité d'un médicament afin de mieux répondre aux besoins des patients.	2	6
		2.12	En tant que pharmacien, je souhaite consulter la liste des médicaments disponibles en Tunisie pour être informé de l'offre locale.	3	6

	2.13	En tant que patient, je veux consulter les médicaments disponibles pour faire des réservations.	1	8
	2.14	Comme patient, je souhaite ajouter une réservation de médicament afin de m'assurer que je peux obtenir le traitement dont j'ai besoin.	1	10
	2.15	En tant que patient, je veux voir la liste de mes réservations et leurs détails (statut, médicament, etc.) pour suivre mes commandes.	3	6
	2.16	Comme patient, je veux pouvoir modifier ou annuler une réservation si mes besoins changent.	2	5
	2.17	En tant que pharmacien, je veux approuver ou supprimer une réservation reçue afin de gérer les commandes de mes clients.	1	8
	2.18	Comme pharmacien, je veux consulter la liste complète des réservations reçues pour suivre les demandes des patients.	2	6
	2.19	En tant que pharmacien, je souhaite consulter les réservations actives d'un patient spécifique pour mieux comprendre leurs besoins.	1	10
	2.20	En tant que patient, je veux pouvoir consulter la dose recommandée d'un médicament pour savoir comment l'utiliser correctement.	1	8

		2.21	Comme pharmacien, je souhaite récupérer les détails d'un médicament par son nom pour faciliter la gestion des stocks et des commandes.	2	6
3	Gestion des Réclamations	3.1	En tant que patient, je veux pouvoir placer une réclamation à la pharmacie pour signaler un problème afin que celui-ci soit pris en charge.	2	8
		3.2	Comme patient, je veux pouvoir modifier une réclamation que j'ai déjà soumise (par exemple, type de réclamation, titre, cause) pour corriger ou préciser ma demande.	3	5
		3.3	En tant que patient, je souhaite annuler une réclamation si le problème est résolu ou si je n'ai plus besoin de suivre la plainte.	1	8
		3.4	Comme patient, je veux pouvoir consulter la liste de mes réclamations et filtrer par statut, date ou type pour suivre mes plaintes.	3	5
		3.5	En tant que patient, je souhaite ajouter une pièce jointe à ma réclamation (par exemple, photo, document) pour fournir des preuves ou des informations supplémentaires.	2	5
		3.6	En tant que pharmacien, je veux consulter la liste complète des réclamations reçues avec leurs détails pour traiter les plaintes des patients.	1	10

		3.7	Comme pharmacien, je souhaite modifier le statut d'une réclamation (par exemple, résolu, en attente, en cours, retour) afin de suivre l'avancement du traitement.	1	8
		3.8	En tant que pharmacien, je veux filtrer les réclamations par statut, date ou type pour les gérer efficacement.	3	5
		3.9	Comme pharmacien, je veux pouvoir retirer et supprimer les réclamations annulées afin de maintenir une base de données propre.	2	5
		3.10	En tant que pharmacien, je souhaite récupérer et télécharger les pièces jointes associées à une réclamation pour examiner des preuves ou des documents supplémentaires.	3	5
		3.11	En tant qu'administrateur, je souhaite visualiser les réclamations via des statistiques pour évaluer la qualité du service et identifier les points à améliorer.	1	10
4	Gestion des dossiers médicaux	4.1	En tant que médecin, je veux rédiger une ordonnance ou un rapport médical électronique pour mes patients afin de documenter les traitements et diagnostics.	1	12
		4.2	Comme médecin ou secrétaire, je veux consulter toutes les ordonnances ou les détails d'une ordonnance spécifique pour suivre l'historique médical des patients.	2	6

		4.3	En tant que médecin ou secrétaire, je veux modifier ou supprimer des ordonnances ou rapports inutiles pour garder les dossiers à jour.	2	8
		4.4	Comme médecin ou secrétaire, je souhaite ajouter des pièces jointes (résultats d'examens, analyses..etc) à un rapport médical pour documenter des informations supplémentaires.	1	12
		4.5	En tant que secrétaire, je veux générer un dossier médical pour chaque patient afin de centraliser leurs informations médicales.	1	10
		4.6	Comme secrétaire, je veux envoyer les dossiers médicaux aux patients pour leur fournir un accès à leurs informations de santé.	2	8
		4.7	En tant que patient, je veux rechercher et consulter la liste de toutes mes ordonnances, mes rapport et mes dossiers ou leurs informations détaillés pour mieux comprendre mon historique médical et suivre mes traitements.	2	8
		4.8	Comme patient, je souhaite recevoir mes dossiers médicaux pour avoir une copie de mes informations médicales.	1	10
		4.9	Comme patient, je veux supprimer des ordonnances, rapports, ou dossiers médicaux inutiles afin de mieux organiser mes informations.	2	6

		4.10	En tant que patient, je souhaite télécharger les pièces jointes d'un rapport médical pour examiner des résultats ou des documents associés.	2	8
5	Magazine	5.1	En tant que médecin, je souhaite ajouter un sujet de discussion et partager des posts concernant ce sujet ainsi que leur modification et suppression afin de promouvoir les échanges et discussions dans le magazine.	1	12
		5.2	En tant que médecin ou pharmacien, je souhaite commenter les différents posts publiés ainsi que supprimer et mettre à jours mes commentaires pour participer aux discussions et partager mes avis.	3	8
		5.3	En tant qu'utilisateur du plateforme, je souhaite rechercher et consulter des posts par nom d'utilisateur, sujet, ou autre critère pour trouver les informations qui m'intéressent.	3	6
6	Interaction médicamenteuse	6.1	En tant que patient, pharmacien ou médecin, je souhaite vérifier les interactions possibles entre différents médicaments d'une manière simple et efficace pour m'assurer qu'il n'y a pas de risques de combinaisons dangereuses.	1	12

TABLE 2.1 – Product Backlog

2.5.2 Planning des Sprints

Comme exposé dans le tableau 2.2, on va présenter le planning des sprints ainsi que leurs priorité.

Sprint	Fonctionnalité	Priorité
Sprint 1	Architecture Microservices	Élevée
	Gestion des utilisateurs	Élevée
	Gestion de session	Élevée
Sprint 2	Gestion des médicaments	Élevée
	Gestion du stock des médicaments.	Élevée
	Gestion des laboratoires.	Élevée
	Gestion du réservations de médicaments.	Moyenne
	Gestion des réclamations.	Moyenne
Sprint 3	Gestion des modèles graphiques	Élevée
Sprint 4	Gestion des dossiers médicaux	Moyenne
	Gestion des ordonnances	Élevée
	Gestion des rapports médicaux	Élevée
Sprint 5	Gestion des blogs	Moyenne
	Gestion des commentaires	Moyenne
Sprint 6	Interaction médicamenteuse	Élevée

TABLE 2.2 – Sprint Planning

Le tableau présenté détaille le plan de développement logiciel divisé en plusieurs sprints. Chaque sprint correspond à une itération de développement pendant laquelle l'équipe s'engage à livrer un ensemble de fonctionnalités spécifiques. La priorité de chaque fonctionnalité est également indiquée pour aider à hiérarchiser les tâches.

Ce tableau de planification des sprints offre une vue d'ensemble du projet. Il est un outil précieux pour organiser les travaux et suivre l'avancement du projet. Cependant, il est important de l'affiner et de l'adapter en fonction de l'évolution du projet et des retours de l'équipe.

2.6 Le diagramme de cas d'utilisation Globale

Comme illustré par la figure 2.3, on va détailler nos acteurs et leurs tâches globales.

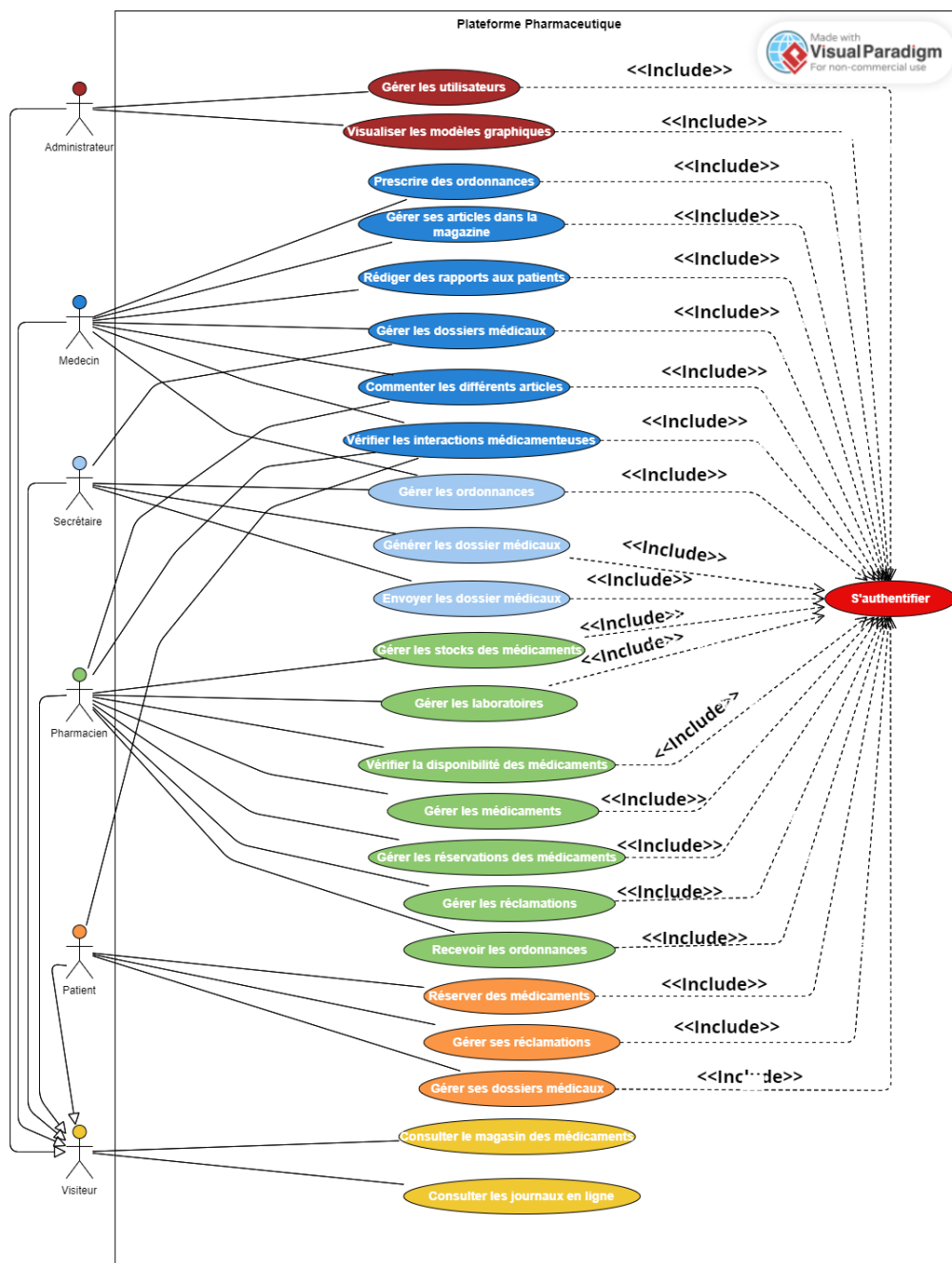


FIGURE 2.3 – Diagramme complet de cas d'utilisation

Ce diagramme de cas d'utilisation global illustre les différentes fonctionnalités offertes par la plateforme pharmaceutique en fonction des rôles des utilisateurs. Chaque acteur (administrateur, médecin, secrétaire, pharmacien, patient, visiteur) est associé à des cas d'utilisation spécifiques correspondant à ses responsabilités et interactions dans le système. Les relations « include » montrent les dépendances fonctionnelles entre les cas d'utilisation, mettant en évidence les fonctionnalités communes ou partagées.

2.8 Environnement de travail

2.8.1 Environnement matériel

Comme présenté dans le tableau 2.3, pendant toute la période de ce stage, nous avons utilisé un ordinateur portable de la marque HP EliteBook avec les spécifications suivantes :

Fabricant	Processeur	Ram	Disque Dur
HP	Intel i5 10 ^{ème} Génération	8 Go	1 To SSD

TABLE 2.3 – Fiche technique de l'ordinateur portable utilisé

2.8.2 Environnement logiciel

Visual Studio 1.95.1 : Visual Studio [6] est un environnement de développement intégré développé par Microsoft et accessible gratuitement. Son adaptabilité, sa simplicité d'utilisation, ainsi que sa capacité à prendre en charge plusieurs langages de programmation ont été les principales raisons pour lesquelles nous l'avons choisi pour créer notre FrontEnd sur Angular.



FIGURE 2.5 – Logo Visual Studio Code
[6]

IntelliJ 2023.3.2 Le logiciel IntelliJ IDEA [7] constitue un environnement de développement intégré conçu par JetBrains et offrant une version gratuite. La sélection de cet IDE pour la création de notre application s'explique par son usage courant pour la programmation en Java et notre BackEnd reposant sur Spring Boot, un framework Java.



FIGURE 2.6 – Logo IntelliJ
[7]

Pycharm 2024.2.4 :PyCharm [8] est un environnement de développement intégré conçu par JetBrains et accessible en version gratuite. Nous avons choisi cet IDE pour le développement de notre application en raison de son utilisation courante pour la programmation en Python, notre BackEnd étant basé sur Flask, un framework Python.



FIGURE 2.7 – Logo Pycharm
[8]

Git : Git [9] est Un système de contrôle de version distribué permet aux développeurs de suivre et de gérer les modifications du code source de manière collaborative. Dans ce projet, l'intégration continue est essentielle, et Git joue un rôle central. Il nous permet de maintenir une branche commune contenant le projet de base (la branche « main ») tout en créant des branches dérivées pour que chaque membre de l'équipe puisse travailler de manière autonome sur ses tâches spécifiques. Les modifications fonctionnelles sont ensuite fusionnées dans la branche principale, garantissant un livrable fonctionnel à chaque itération.



FIGURE 2.8 – Logo Git
[9]

Postman 4.4.4 : Postman [10] est un outil de développement API qui simplifie le processus de création, de partage, de test et de documentation des API. Nous utilisons régulièrement Postman pour tester ainsi appeler les diverses API du BackEnd de notre projet. Sa plateforme conviviale facilite l'envoi de requêtes HTTP, les test et la réception des réponses associées. **Swagger 2.9.2** :



FIGURE 2.9 – Logo Postman
[10]

Swagger [11] est un outil de développement API qui facilite la conception, la construction, la documentation et la consommation d'API RESTful. Nous utilisons Swagger pour générer automatiquement la documentation des API de notre projet, permettant ainsi aux développeurs de tester et d'interagir avec les points de terminaison directement depuis une interface utilisateur conviviale.



FIGURE 2.10 – Logo Swagger
[11]

2.9 Le choix des technologies

2.9.1 Frameworks et technologies utilisées

Angular 16 : Angular [12] Est un framework open source développé par Google, particulièrement adapté pour la création d'applications web dynamiques et réactives. Le choix d'Angular pour notre projet repose sur plusieurs facteurs techniques :

- **Architecture MVVM (Model-View-ViewModel)** : Angular suit cette architecture qui sépare clairement la logique métier et l'interface utilisateur, ce qui facilite la maintenance du code et améliore la testabilité.
- **Two-way data binding** : Angular permet une liaison bidirectionnelle des données, ce qui simplifie la gestion des états de l'application et améliore l'expérience utilisateur en synchronisant automatiquement les données entre le modèle et la vue.
- **Composants réutilisables** : Grâce à son système de composants, Angular permet de créer des éléments modulaires et réutilisables, ce qui optimise la structure du code et réduit les redondances.
- **Écosystème et support** : Avec sa vaste communauté et son intégration avec des outils modernes comme TypeScript, Angular assure une mise à jour régulière et un support à long terme, ce qui est essentiel pour la pérennité du projet.



FIGURE 2.11 – Logo Angular
[12]

Springboot 2.7.8 : [13] Est un framework Java conçu pour simplifier le développement d'applications web et microservices. Les raisons techniques de notre choix incluent :

- **Simplification de la configuration** : Spring Boot minimise la configuration nécessaire grâce à ses conventions par défaut, ce qui permet de gagner du temps et de réduire les erreurs de configuration.
- **Support des microservices** : Le projet étant basé sur une architecture de microservices, Spring Boot facilite le développement de services découplés, permettant une meilleure évolutivité et une gestion efficace des services.

- **Sécurité et gestion des transactions** : Avec des outils comme Spring Security et Spring Data, Spring Boot offre des solutions intégrées pour sécuriser l'application et gérer les accès aux données.

- **Flexibilité et performance** : Spring Boot permet d'optimiser les performances des applications grâce à son architecture légère et à sa capacité à déployer des applications indépendantes sans serveur externe.



FIGURE 2.12 – Logo Springboot
[13]

Keycloak 16 : Keycloak [14] est une plateforme open source de gestion des identités et des accès, développée par Red Hat, qui est fondamentale pour notre projet. Conçue pour simplifier les processus d'authentification et d'autorisation, elle permet une gestion centralisée des utilisateurs et des permissions pour diverses applications et services. En intégrant Keycloak, nous bénéficions d'une solution clé pour sécuriser nos applications, assurant une gestion optimisée des accès et répondant à nos exigences de sécurité et d'authentification.

Les raisons de son intégration dans notre projet sont les suivantes :

Gestion centralisée des utilisateurs : Keycloak permet de centraliser la gestion des utilisateurs et des permissions pour l'ensemble des applications, ce qui simplifie l'administration et assure une sécurité renforcée.

Support des protocoles standard : Keycloak prend en charge des protocoles d'authentification courants tels que OAuth2 et OpenID Connect, ce qui facilite l'intégration avec des applications externes tout en garantissant des mécanismes de sécurité robustes.

Single Sign-On (SSO) : Grâce à Keycloak, les utilisateurs peuvent se connecter une seule fois pour accéder à toutes les applications sécurisées, offrant ainsi une expérience utilisateur fluide et réduisant les risques de failles de sécurité liées à la gestion des sessions.



FIGURE 2.13 – Logo Keycloak
[14]

2.9.2 Choix de la base de données

PostgreSQL 17 : PostgreSQL [15] est un système open source robuste pour la gestion des bases de données relationnelles (SGBDR). Nous avons choisi PostgreSQL pour les raisons

techniques suivantes :

Fiabilité et conformité aux normes SQL : PostgreSQL est reconnu pour sa conformité avec les standards SQL, ce qui garantit une compatibilité optimale avec divers outils et une facilité de migration si nécessaire.

Haute performance et scalabilité : Grâce à son moteur de stockage et à ses optimisations avancées, PostgreSQL est capable de gérer de grandes quantités de données tout en maintenant des performances élevées.

Sécurité avancée : PostgreSQL intègre des mécanismes de sécurité avancés, notamment le chiffrement des données et la gestion fine des accès, ce qui est essentiel pour la protection des informations sensibles dans le cadre de notre projet.



FIGURE 2.14 – Logo PostgreSQL
[15]

2.10 Conclusion

Après l'introduction du contexte, le but de ce chapitre est de fournir une vue précise du projet ainsi que des diverses tâches qui le composent, que nous allons détailler progressivement tout au long de ce rapport.

Chapitre 3

Sprint 1 : Gestion des utilisateurs

3.1	Introduction	36
3.2	Sprint Backlog 1	36
3.3	Partie Analyse	39
3.3.1	Diagramme de cas d'utilisation raffiné "Authentification"	40
3.3.2	Diagramme de cas d'utilisation "gestion des utilisateurs"	40
3.3.3	Diagramme de classe de conception relatif à l'authentification et gestion des utilisateurs :	41
3.4	Partie Conception	41
3.4.1	Diagramme de séquence "authentification"	42
3.4.2	Diagramme de séquence gestion profil	43
3.4.3	Description textuelle de cas d'utilisation "S'authentifier"	43
3.5	Partie Réalisation	44
3.5.1	l'interface d'inscription	44
3.5.2	Détails d'un utilisateur	44
3.5.3	Liste des utilisateurs	45
3.5.4	Assignation des rôles	45
3.5.5	Modifier un utilisateur	46
3.5.6	Ajouter un utilisateur	46
3.6	Conclusion	47

3.1 Introduction

Ce chapitre se focalise sur la documentation des tâches effectuées durant ce premier sprint. En débutant par l'analyse du backlog de ce sprint, nous procéderons ensuite, à la conception et pour conclure, nous évaluerons la mise en œuvre en présentant certaines illustrations d'interfaces.

3.2 Sprint Backlog 1

Comme présenté sur le tableau 3.1, on va détailler le backlog pour notre sprint.

ID	User story	ID Tâche	Tâche	Estimation	Responsable
1.1	Comme étant utilisateur, je veux créer un compte pour personnaliser mon expérience. Le processus d'inscription doit être intuitif et me permettre de commencer à utiliser le service rapidement.	1.1.1	Créer les endpoints pour l'inscription des utilisateurs, y compris la validation et la gestion des données utilisateur	3	Nader ZOUAIRI
		1.1.2	Création du contrôleur et ajout des fonctionnalités nécessaires	2	Nader ZOUAIRI
		1.1.3	Créer un formulaire d'inscription convivial avec des champs pour le nom, l'email, le mot de passe, etc.	1	Nader ZOUAIRI
		1.1.4	Fournir une documentation interne pour l'équipe de développement sur le processus et les détails de l'implémentation de l'inscription.	1	Nader ZOUAIRI

		1.1.5	Écrire des tests unitaires pour la logique d'inscription backend et les composants de formulaire frontend.	1	Nader ZOUAIRI
1.2	Sous l'identité d'administrateur système, je veux pouvoir ajouter de nouveaux administrateurs pour partager la charge de travail et renforcer la sécurité de la plateforme.	1.2.1	Créer un endpoint pour ajouter de nouveaux administrateurs, incluant la validation des données et les contrôles d'autorisation.	3	Nader ZOUAIRI
		1.2.2	Création du contrôleur et ajout des fonctionnalités nécessaires.	2	Nader ZOUAIRI
		1.2.3	Créer un formulaire utilisateur pour permettre à un administrateur système d'ajouter de nouveaux administrateurs.	1	Nader ZOUAIRI
		1.2.4	Documenter les endpoints API pour l'ajout d'administrateurs, y compris les formats de requête et de réponse.	1	Nader ZOUAIRI
		1.2.5	Écrire des tests unitaires pour la logique backend relative à l'ajout d'administrateurs.	1	Nader ZOUAIRI
1.3	Comme utilisateur, je veux pouvoir modifier mes informations personnelles (nom, prénom, email, etc.) à tout moment.	1.3.1	Créer un endpoint pour permettre aux utilisateurs de mettre à jour leurs informations personnelles. Ce service devrait inclure des vérifications pour assurer la validité des nouvelles informations.	1	Nader ZOUAIRI

		1.3.2	Création du contrôleur et ajout des fonctionnalités nécessaires.	1	Nader ZOUAIRI
		1.3.3	Créer un formulaire sur l'interface utilisateur permettant aux utilisateurs de mettre à jour leurs informations personnelles.	1	Nader ZOUAIRI
		1.3.4	Documenter les endpoints API pour la mise à jour des informations personnelles, y compris les formats de requête et de réponse.	1	Nader ZOUAIRI
		1.3.5	Écrire des tests unitaires pour la logique backend relative à la mise à jour des informations personnelles.	1	Nader ZOUAIRI
1.4	Comme étant administrateur et afin de gérer les comptes utilisateurs, je souhaite pouvoir modifier les profils des utilisateurs.	1.4.1	Créer un endpoint permettant aux administrateurs de modifier les informations des profils des utilisateurs, incluant les contrôles d'autorisation nécessaires.	1	Nader ZOUAIRI
		1.4.2	Création du contrôleur et ajout des fonctionnalités nécessaires.	1	Nader ZOUAIRI
		1.4.3	Créer une interface utilisateur permettant aux administrateurs de modifier les informations des profils des utilisateurs, incluant des champs pour toutes les informations modifiables.	1	Nader ZOUAIRI

		1.4.4	Documenter les endpoints API pour la modification des profils des utilisateurs, y compris les formats de requête et de réponse.	1	Nader ZOUAIRI
		1.4.5	Écrire des tests unitaires pour la logique backend relative à la modification des profils des utilisateurs.	1	Nader ZOUAIRI
1.5	À titre d'administrateur, je veux pouvoir désactiver les comptes utilisateurs pour restreindre leur accès à la plateforme	1.5.1	Créer un endpoint permettant aux administrateurs de désactiver les comptes des utilisateurs. Cela inclut la validation des requêtes et les vérifications d'autorisation.	1	Nader ZOUAIRI
		1.5.2	Création du contrôleur et ajout des fonctionnalités nécessaires.	1	Nader ZOUAIRI
		1.5.3	Créer une interface utilisateur permettant aux administrateurs de visualiser les comptes actifs et de les désactiver. Inclure des éléments tels que des boutons ou des listes pour gérer l'état des comptes.	1	Nader ZOUAIRI

TABLE 3.1 – Sprint 1 Backlog

3.3 Partie Analyse

Dans cette section on va présenter le raffinement de nos diagrammes de cas d'utilisation durant cette sprint.

3.3.1 Diagramme de cas d'utilisation raffiné "Authentication"

Comme détaillé par la figure 3.1, on va présenter le raffinement du cas d'utilisation "Authentication"

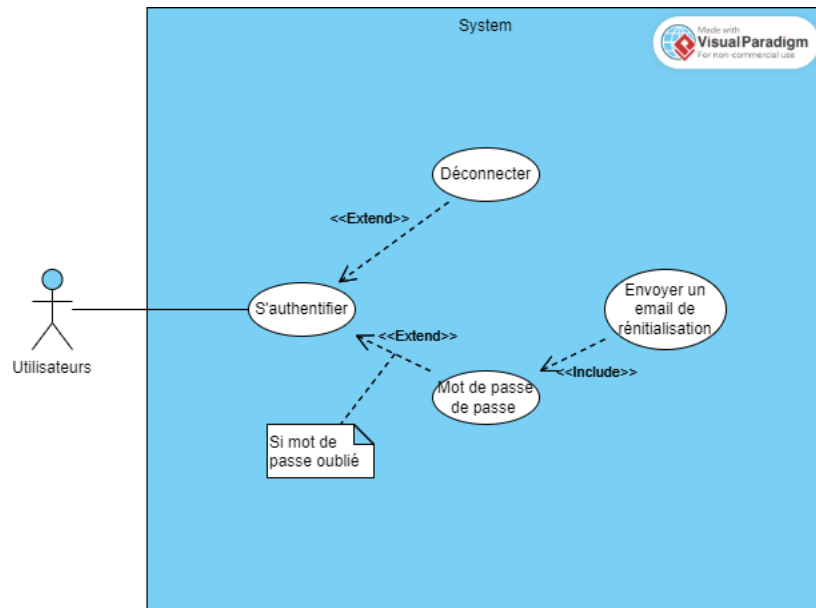


FIGURE 3.1 – Diagramme de cas d'utilisation authentication

Ce diagramme représente les cas d'utilisation liés à l'authentification des utilisateurs, incluant la connexion, la déconnexion, et la gestion du mot de passe oublié. Les relations indiquent les extensions ou dépendances entre ces fonctionnalités pour assurer une gestion sécurisée des accès.

3.3.2 Diagramme de cas d'utilisation "gestion des utilisateurs"

Comme mis en lumière par l'illustration 3.2, on a raffiné le cas d'utilisation "Gestion des utilisateurs"

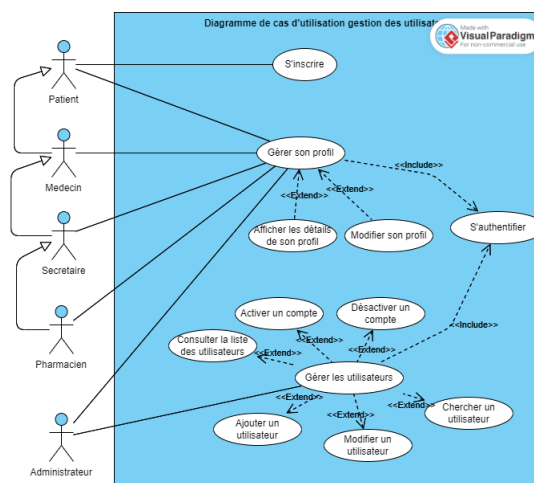


FIGURE 3.2 – Diagramme de cas d'utilisation gestion des utilisateurs

Ce diagramme illustre les cas d'utilisation liés à la gestion des utilisateurs. Il couvre des actions telles que l'inscription, la gestion des profils par les différents rôles, et l'administration des comptes, notamment l'ajout, la modification ou la désactivation d'utilisateurs. Les relations mettent en évidence les dépendances et extensions entre ces fonctionnalités pour garantir une gestion centralisée et efficace des utilisateurs.

3.3.3 Diagramme de classe de conception relatif à l'authentification et gestion des utilisateurs :

Comme interprété dans la figure 3.3, le diagramme contient les classes manipulées au cours de développement de ce sprint au niveau d'authentification et gestion des utilisateurs.

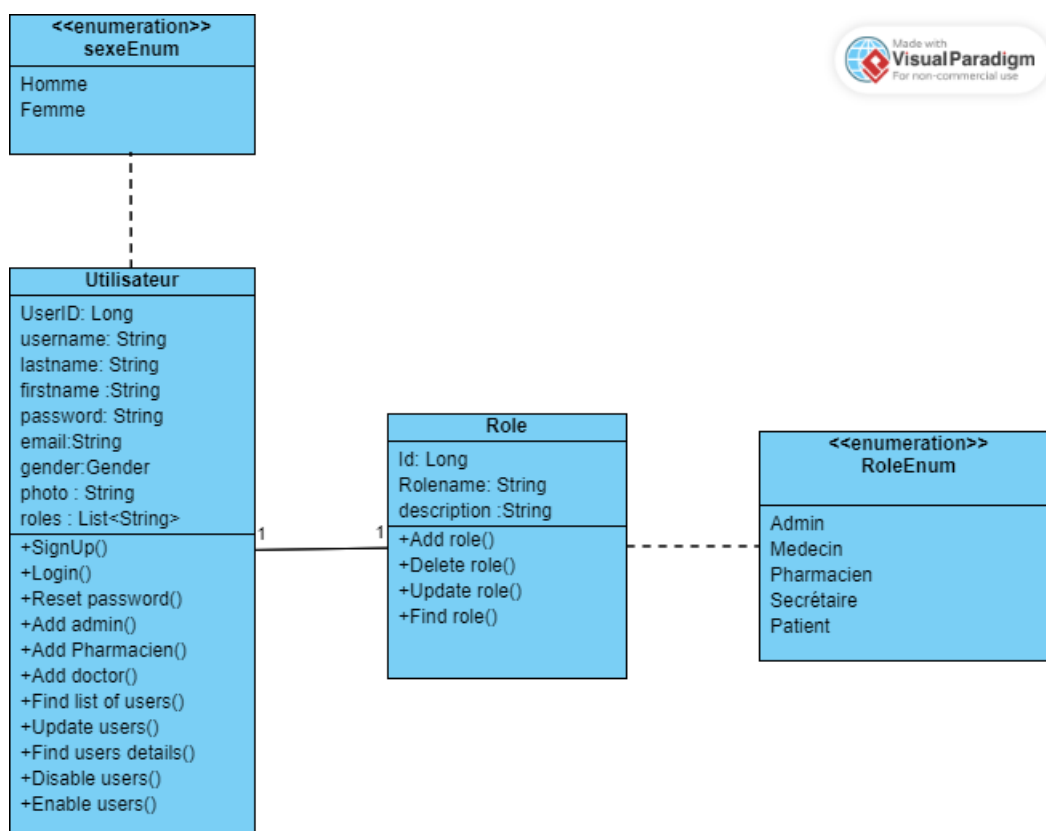


FIGURE 3.3 – Diagramme de classe niveau authentification et gestion utilisateurs

Ce diagramme de classe représente les différentes catégories d'utilisateurs (par exemple, administrateur, médecin) et leurs attributs (nom, prénom, rôle). Les relations entre ces catégories sont également illustrées, montrant par exemple qu'un utilisateur peut avoir plusieurs rôles.

3.4 Partie Conception

Dans cette partie nous allons présenter quelques diagramme de séquence de notre travail ainsi qu'un exemple de description textuelle pour une tâche spécifique.

3.4.1 Diagramme de séquence "authentification"

Comme établi dans la figure 3.4, ce diagramme explique l'interaction entre les différents objets afin de s'authentifier.

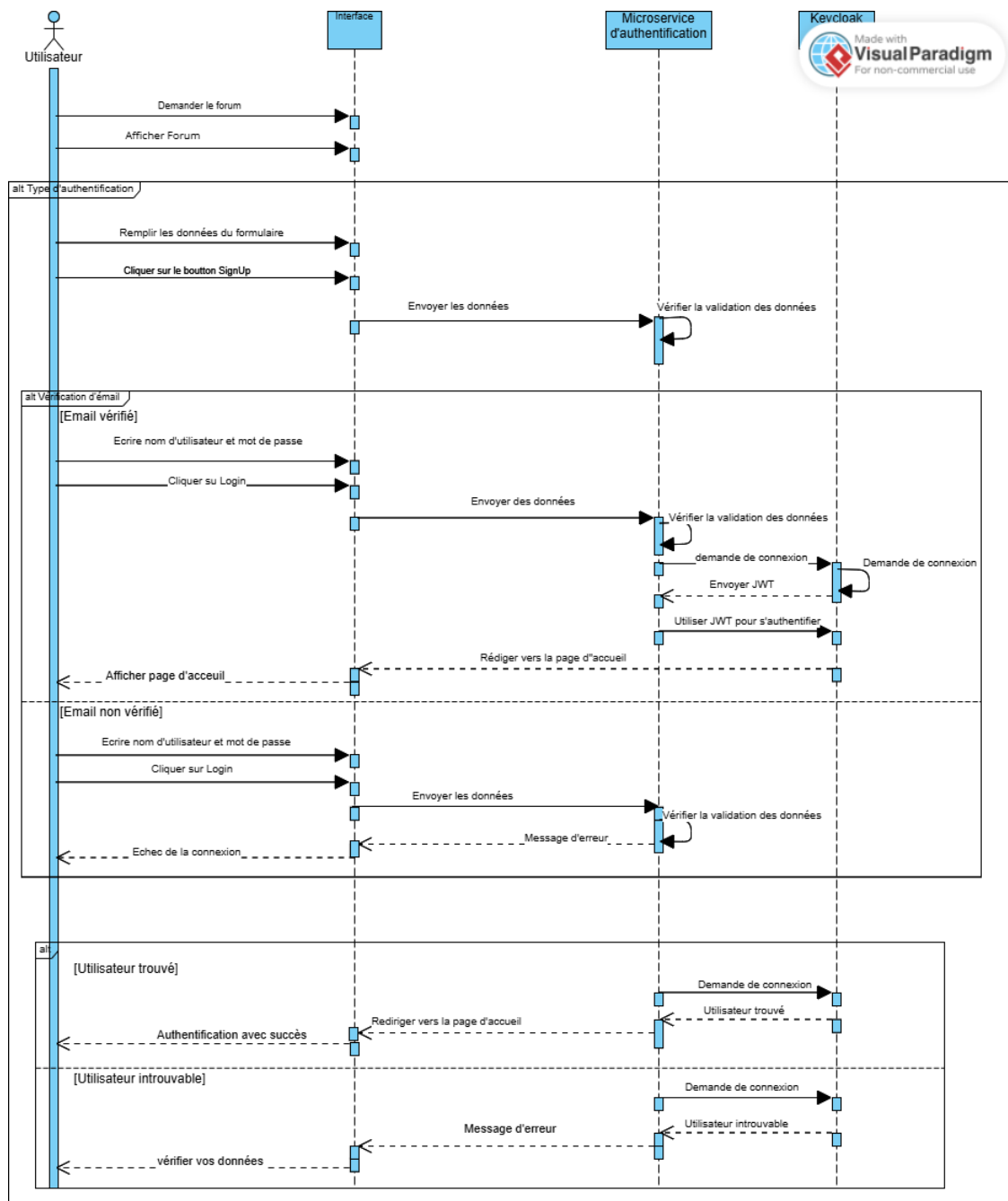


FIGURE 3.4 – Diagramme de séquence authentification

Le diagramme de séquence présenté décrit le processus d'authentification d'un utilisateur sur la plateforme. Il met en évidence les interactions entre l'utilisateur, l'interface utilisateur et le microservice d'authentification, en passant par la vérification de l'email et la génération d'un jeton JWT (JSON Web Token) pour les utilisateurs authentifiés.

3.4.2 Diagramme de séquence gestion profil

La figure 3.5, présente le diagramme de séquence pour la gestion des profils.

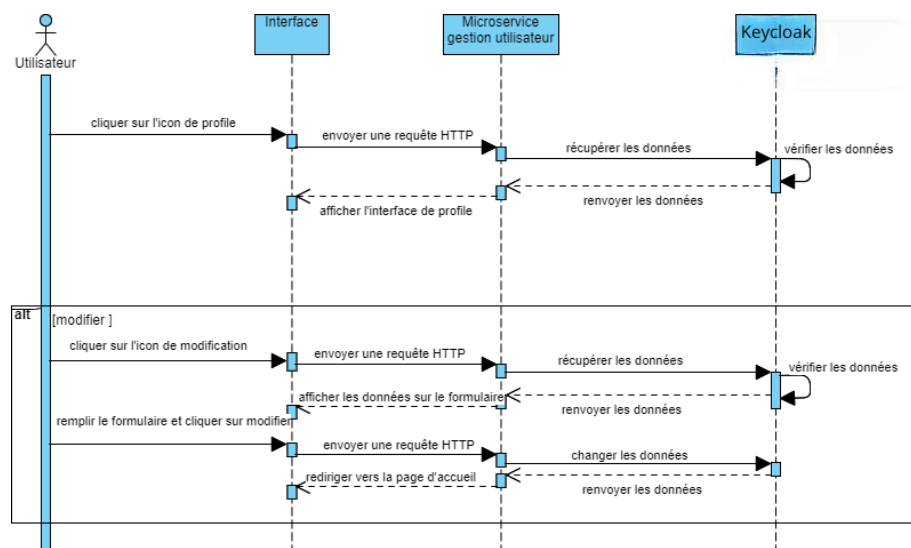


FIGURE 3.5 – Diagramme de Séquence Gestion Profil

3.4.3 Description textuelle de cas d'utilisation "S'authentifier"

Comme montré dans le tableau 3.2, on va détailler le demarche du cas d'utilisation "s'authentifier".

Élément	Description
Cas d'utilisation	S'authentifier
Acteur	Patient
Pré-condition	Le patient souhaite accéder à des fonctionnalités protégées du système.
Post-condition	Le patient est connecté au système et autorisé à accéder aux fonctionnalités correspondantes à son profil.
Scénario nominal	1- Le patient saisit son identifiant (nom d'utilisateur, email) et son mot de passe dans le formulaire d'authentification. 2- Le système vérifie la validité des identifiants en les comparant avec ceux stockés en base de données. 3- Si les identifiants sont corrects, le système crée une session utilisateur et le redirige vers la page d'accueil ou la page correspondant à son profil.
Scénario alternatif	3a- Le système affiche un message d'erreur indiquant que les identifiants sont incorrects. 4a- L'utilisateur a la possibilité de réessayer.

TABLE 3.2 – Description textuelle du cas d'utilisation : S'authentifier

3.5 Partie Réalisation

Dans cette partie on va illustrer quelques interfaces d'utilisateurs de notre plateforme qui concerne cette sprint.

3.5.1 l'interface d'inscription

Comme observé dans l'image 3.6, notre interface d'inscription d'utilisateur.

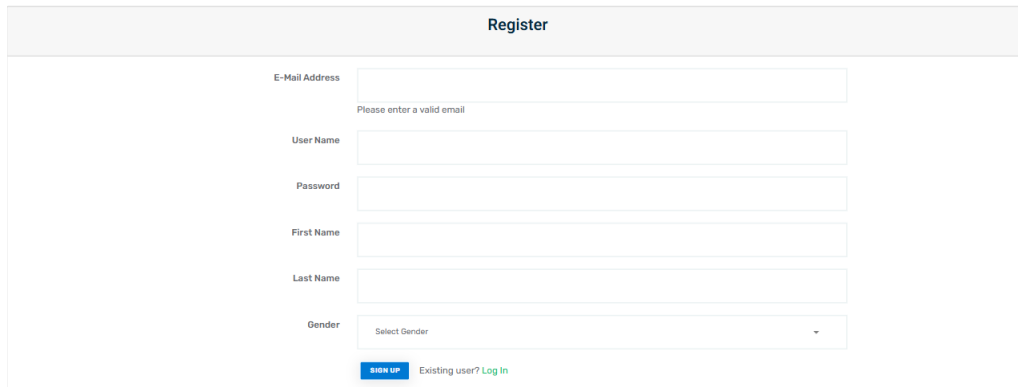
The image shows a web form titled "Register". It contains several input fields: "E-Mail Address" with a placeholder "Please enter a valid email", "User Name", "Password", "First Name", "Last Name", and a "Gender" dropdown menu with the text "Select Gender". At the bottom, there is a blue "SIGN UP" button and a link that says "Existing user? Log In".

FIGURE 3.6 – Réalisation de l'interface d'inscription

L'interface d'inscription permet aux utilisateurs de remplir une formulaire avec ses coordonnées on respectant le contrôle de saisie des champs et de s'inscrire facilement.

3.5.2 Détails d'un utilisateur

Comme identifié par la figure 3.7, voilà notre interface qui présente les détails d'un profil utilisateur.

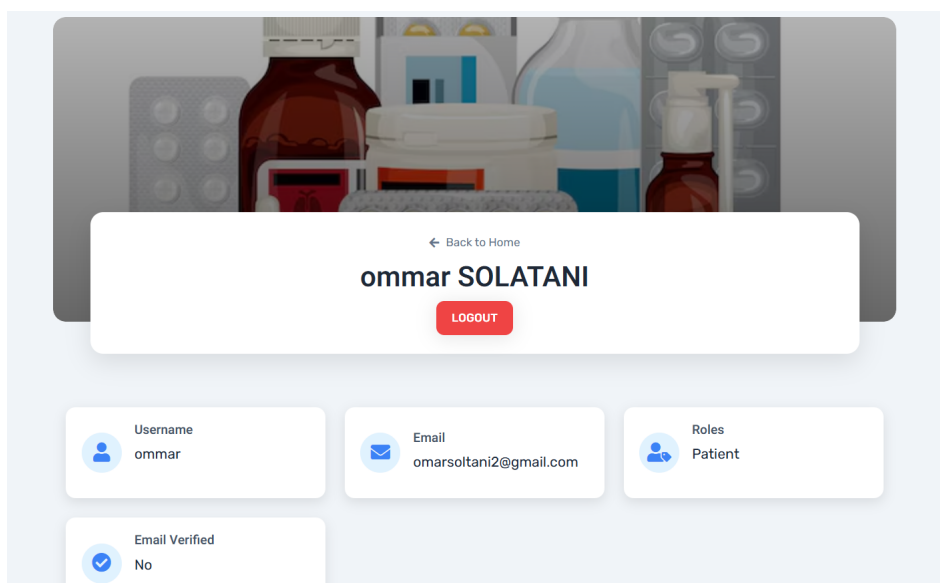
The image shows a user profile page. At the top, there is a "Back to Home" link. Below it, the user's name "ommar SOLATANI" is displayed in bold, with a red "LOGOUT" button underneath. The profile details are organized into four cards: "Username" with the value "ommar", "Email" with the value "omarsoltani2@gmail.com", "Roles" with the value "Patient", and "Email Verified" with a checkmark icon and the value "No". The background of the page features a blurred image of medical supplies like bottles and pills.

FIGURE 3.7 – détails d'un utilisateur

Les détails du profil utilisateur sont affichés dans l'interface des détails, y compris le nom, l'adresse email et le rôle attribué.

3.5.3 Liste des utilisateurs

Comme illustré dans la figure 3.8, voici l'interface d'administration de la liste d'utilisateurs.

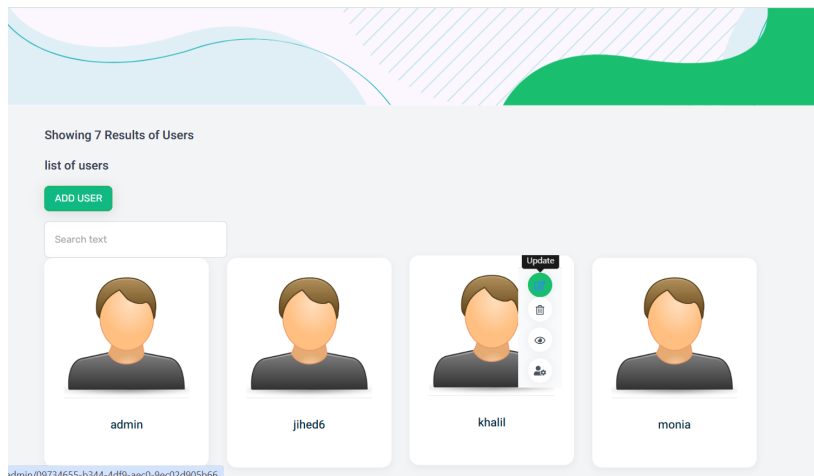


FIGURE 3.8 – Liste des utilisateurs

La liste complète des utilisateurs, avec leurs informations détaillées, est accessible à l'administrateur, avec une fonctionnalité de recherche pour faciliter la navigation à travers deS multiples entrées.

3.5.4 Assignment des rôles

La figure 3.9 présente l'interface de l'assignation des rôles par l'administrateur.

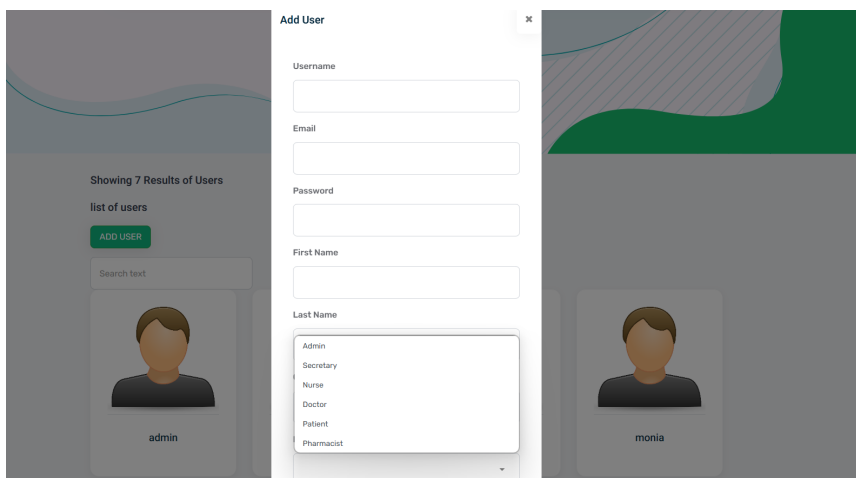
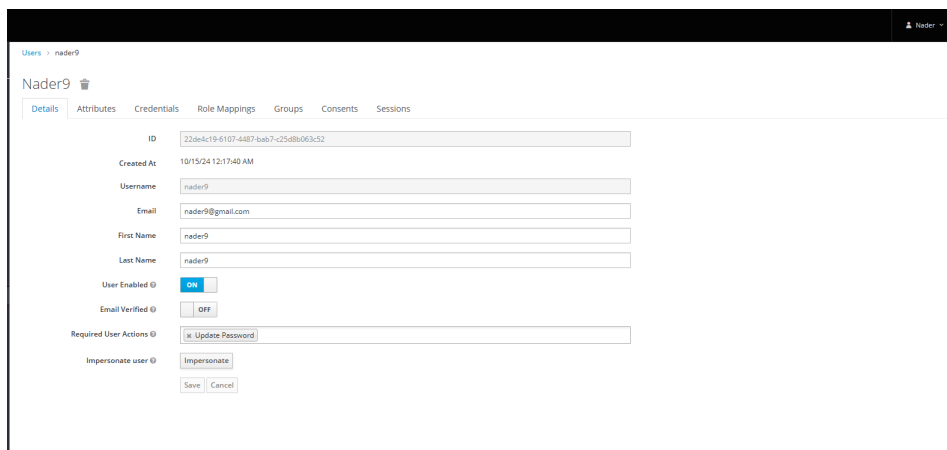


FIGURE 3.9 – Interface d'assignation des rôles

L'administrateur a la possibilité de gérer les rôles des différents utilisateurs à partir de cette interface.

3.5.5 Modifier un utilisateur

La figure 3.10, présente l'interface qui permet la modification des données d'un utilisateur.



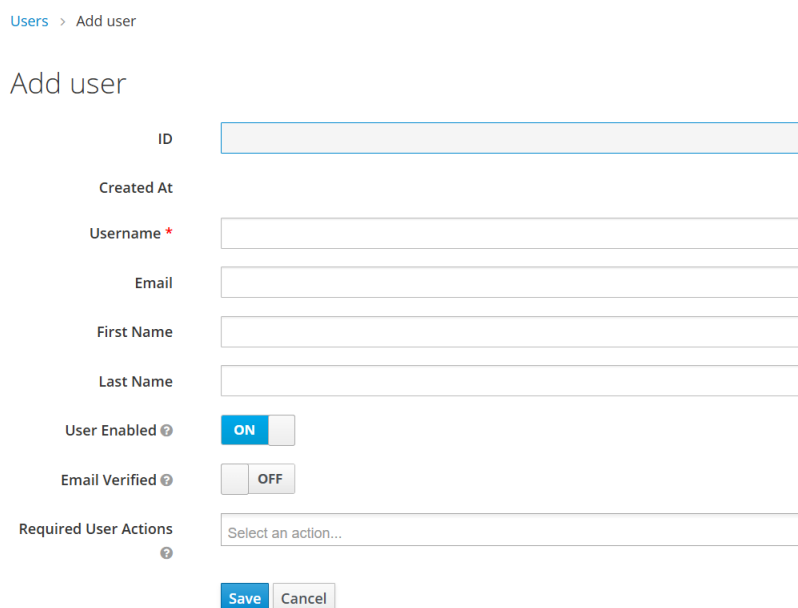
The screenshot shows the 'Users' management interface for 'nader9'. The 'Details' tab is active, displaying various user attributes in a form. The fields include ID (22de4c19-6107-4487-bab7-c25d8b063c52), Created At (10/15/24 12:17:40 AM), Username (nader9), Email (nader9@gmail.com), First Name (nader9), and Last Name (nader9). There are also toggle switches for 'User Enabled' (ON) and 'Email Verified' (OFF). A dropdown menu for 'Required User Actions' is set to 'Update Password', and an 'Impersonate user' button is visible. At the bottom, there are 'Save' and 'Cancel' buttons.

FIGURE 3.10 – Interface gestion des utilisateurs

L'administrateur a la possibilité de consulter, modifier et désactiver un compte d'utilisateur ainsi que ses données.

3.5.6 Ajouter un utilisateur

comme illustré par la figure 3.11, on a l'interface d'ajout des utilisateurs.



The screenshot shows the 'Add user' form in the Nader9 interface. It includes input fields for ID, Created At, Username (marked with a red asterisk), Email, First Name, and Last Name. There are also toggle switches for 'User Enabled' (ON) and 'Email Verified' (OFF). A dropdown menu for 'Required User Actions' is set to 'Select an action...'. At the bottom, there are 'Save' and 'Cancel' buttons.

FIGURE 3.11 – Interface d'ajout d'utilisateur

En utilisant l'interface d'ajout d'un nouvel utilisateur, l'administrateur peut créer un nouveau compte pour un autre administrateur, un pharmacien, etc., en choisissant le rôle approprié. Il collecte les données nécessaires et attribue le rôle souhaité, ce qui simplifie la gestion des utilisateurs et des autorisations sur la plateforme.

3.6 Conclusion

Après le développement et la validation de la gestion des utilisateurs par notre chef de projet, nous pouvons désormais passer au sprint suivant tout en disposant d'un livrable opérationnel sur lequel nous pourrions continuer à travailler.

Chapitre 4

Sprint 2 : Gestion du Pharamcie

4.1	Introduction	49
4.2	Sprint Backlog 2	49
4.3	Partie Analyse	56
4.3.1	Diagramme de cas d'utilisation globale gestion du pharamcie	57
4.3.2	Diagramme de classe de conception niveau "Gestion Pharmacie"	58
4.3.3	Diagramme de cas d'utilisation raffiné "Gérer les stocks des médicaments"	59
4.3.4	Diagramme de cas d'utilisation raffiné "Gérer les réclamations"	59
4.3.5	Diagramme de cas d'utilisation raffiné "Gérer les réservations"	60
4.3.6	Diagramme de cas d'utilisation raffiné "Gérer ses réclamations"	60
4.3.7	Diagramme de cas d'utilisation raffiné "Réserver médicaments"	61
4.3.8	Diagramme de cas d'utilisation raffiné "Gérer médicaments"	61
4.4	Partie Conception	62
4.4.1	Diagramme de séquence "Traiter status"	62
4.4.2	Diagramme de séquence "Réserver médicaments"	62
4.4.3	Description textuelle de cas d'utilisation "Ajouter médicaments"	63
4.5	Partie Réalisation	63
4.5.1	Affichage de liste des médicaments	63
4.5.2	Traitement des réclamations	64
4.5.3	Suppression des réclamations	65
4.5.4	Traitement d'une réclamation	65
4.5.5	Affichage du liste des laboratoires en Tunisie	66
4.6	Conclusion	66

4.1 Introduction

Dans ce chapitre, nous nous concentrerons sur la présentation des tâches accomplies durant le sprint 2. Nous commencerons par le backlog du sprint, puis nous décrirons la conception, et enfin, nous illustrerons des interfaces pour la mise en œuvre.

4.2 Sprint Backlog 2

Comme indiqué dans le tableau 4.1, nous allons examiner de manière détaillée le backlog pour notre deuxième sprint.

ID	User story	ID Tâche	Tâche	Estimation	Responsable
1.1	En tant que pharmacien, je souhaite consulter la liste de tous les laboratoires avec leurs détails (nom, téléphone, adresse web) afin de mieux gérer mes fournisseurs.	1.1.1	Créer une API pour récupérer et gérer les informations des Laboratoires disponibles depuis la base de données.	2	Nader ZOUAIRI
		1.1.2	Concevoir et implémenter une interface utilisateur permettant de visualiser la liste des laboratoires disponibles, incluant des détails tels que les adresses et les emails ..etc.	1	Nader ZOUAIRI
		1.1.3	Tester l'API et l'interface utilisateur pour vérifier que les laboratoires sont correctement affichés et que les informations sont exactes.	1	Nader ZOUAIRI

1.2	Comme pharmacien, je veux pouvoir rechercher un laboratoire par nom ou par identifiant afin d'accéder aux informations détaillées d'un laboratoire spécifique.	1.2.1	Implémenter une API pour récupérer les détails d'un laboratoire spécifique avec un recherche avancé et facile	2	Nader ZOUAIRI
		1.2.2	Créer une interface pour les pharamaciens permettant de rechercher un laboratoire spécifié et afficher ses informations détaillées	1	Nader ZOUAIRI
		1.2.3	Documenter l'API de recherche des laboratoires et les instructions pour utiliser l'interface du pharmacien.	1	Nader ZOUAIRI
1.3	En tant que pharmacien, je veux ajouter des stocks de médicaments afin de maintenir un inventaire à jour.	1.3.1	Mettre en place une API pour ajouter des stocks de médicaments	4	Nader ZOUAIRI
		1.3.2	Concevoir une interface pour les pharamciens qui leur permet d'ajouter facilement les nouveaux stocks de médicaments.	3	Nader ZOUAIRI
		1.3.3	Vérifier que toutes les informations des stocks sont correctement ajoutées, et tester les cas de données manquantes ou incorrectes.	1	Nader ZOUAIRI

1.4	Comme pharmacien, je souhaite consulter la liste complète des stocks disponibles afin de savoir quels médicaments sont en stock.	1.4.1	Créer une API pour récupérer toutes la liste des stocks disponibles pour avoir une vue d'ensemble sur les médicaments qui sont en stock.	2	Nader ZOUAIRI
		1.4.2	Concevoir une interface permettant d'afficher toutes la liste des stocks dans l'espace du personnel du pharamcien.	3	Nader ZOUAIRI
		1.4.3	Vérifier que la liste des stocks s'affiche correctement dans l'espace personnel du pharmacien et que les informations sont à jour.	1	Nader ZOUAIRI
1.5	En tant que pharmacien, je veux pouvoir consulter un stock spécifique par identifiant ou par nom de laboratoire pour voir ses informations détaillées (nom, quantité, date d'arrivage, etc.).	1.5.1	Développer une API pour récupérer les informations sur tous les laboratoires, y compris leurs détails (nom, adresse, numéro téléphone ..etc).	4	Nader ZOUAIRI
		1.5.2	Créer une interface pour les pharmaciens affichant une vue d'ensemble des laboratoires, incluant un tableau ou une liste avec les détails nécessaires.	3	Nader ZOUAIRI

		1.5.3	Vérifier que toutes les données des laboratoires sont correctement affichées et que les informations sont précises et à jour.	1	Nader ZOUAIRI
1.6	En tant que pharmacien, je veux pouvoir restocker un médicament dont la date d'expiration approche afin d'éviter une rupture de stock.	1.6.1	Implémenter une API pour restocker un stock dont la date d'expiration approche en mettant à jour sa date dans la base de données et en synchronisant les modifications.	4	Nader ZOUAIRI
		1.6.2	Créer une interface pour les pharamciens permettant de restocker les différents stocks.	4	Nader ZOUAIRI
1.7	Comme pharmacien, je veux pouvoir modifier certaines informations d'un stock spécifique (par exemple, nom, quantité, date d'arrivée) pour corriger ou mettre à jour les données.	1.7.1	Développer une API pour récupérer les informations d'un stocks et permettre de les modifier afin de faciliter la mise à jours.	2	Nader ZOUAIRI
		1.7.2	Concevoir une interface pour les pharmacien permettant de consulter les details des stocks et les modifier d'une manière ergonomique et pratique et	2	Nader ZOUAIRI
		1.7.3	Vérifier que les informations des stocks modifiés s'affichent correctement.	2	Nader ZOUAIRI

1.8	En tant que pharmacien, je souhaite ajouter de nouveaux médicaments disponibles de manière simple et efficace afin de tenir à jour l'inventaire.	1.8.1	Créer une API pour ajouter un nouveaux médicaments y compris la récupération de ses informations telque le dosage, la Conditionnement Primaire et la classe.	4	Nader ZOUAIRI
		1.8.2	Création du contrôleur et ajout des fonctionnalités nécessaires.	2	Nader ZOUAIRI
		1.8.3	Créer un formulaire utilisateur pour permettre au pharmacien d'ajouter de nouveaux médicaments.	2	Nader ZOUAIRI
1.9	Comme patient, je souhaite ajouter une réservation de médicament afin de m'assurer que je peux obtenir le traitement dont j'ai besoin.	1.9.1	Développer une API qui permet de créer, récupérer, et supprimer des réservations de médicaments.	4	Nader ZOUAIRI
		1.9.2	Création du contrôleur et ajout des fonctionnalités nécessaires.	4	Nader ZOUAIRI
		1.9.3	Concevoir et implémenter une interface utilisateur pour la réservation des médicaments.	2	Nader ZOUAIRI
1.10	En tant que patient, je veux voir la liste de mes réservations et leurs détails (statut, médicament, etc.) pour suivre mes commandes.	1.10.1	Développer une API qui permet de récupérer des réservations de médicaments.	2	Nader ZOUAIRI

		1.10.2	Concevoir et implémenter une interface utilisateur pour la réservation de médicaments qui afficher les détails des réservations, y compris le statut (en attente, confirmé, annulé).	3	Nader ZOUAIRI
		1.10.3	Vérifier que toutes les informations des réservations sont correctement récupérées	1	Nader ZOUAIRI
1.11	Comme patient, je veux pouvoir modifier ou annuler une réservation si mes besoins changent.	1.11.1	Développer une API qui permet de récupérer, mettre à jour, et supprimer des réservations de médicaments.	2	Nader ZOUAIRI
		1.11.2	Concevoir et implémenter une interface utilisateur qui permet aux patients de consulter leurs réservations passées et actuelles, avec une option pour annuler une réservation si nécessaire.	2	Nader ZOUAIRI
		1.11.3	Vérifier que toutes les informations des réservations sont correctement récupérées ou mettre à jours.	1	Nader ZOUAIRI
1.12	En tant que pharmacien, je veux approuver ou supprimer une réservation reçue afin de gérer les commandes de mes clients	1.12.1	Développer une API qui permet au pharmacien de mettre à jour le statut des réservations (approuvée, annulée).	4	Nader ZOUAIRI

		1.12.2	Créer une vue dédiée où le pharmacien peut consulter la liste des réservations reçues et Ajouter des boutons ou options pour approuver ou supprimer chaque réservation, avec confirmation d'action pour éviter les erreurs.	3	Nader ZOUAIRI
		1.12.3	Vérifier que le pharmacien peut approuver ou supprimer les réservations sans erreur.	1	Nader ZOUAIRI
1.13	Comme pharmacien, je veux consulter la liste complète des réservations reçues pour suivre les demandes des patients.	1.13.1	Développer une API qui permet de récupérer toutes les réservations reçues, y compris la filtration des réservations par statut (en attente, approuvée, annulée) et par date afin de faciliter le suivi.	2	Nader ZOUAIRI
		1.13.2	Créer une vue dédiée pour le pharmacien, affichant la liste complète des réservations avec des informations telles que le nom du patient, le médicament, la quantité demandée, la date de réservation et le statut.	2	Nader ZOUAIRI

		1.13.3	Ajouter des fonctionnalités de recherche et de filtrage qui Permet au pharmacien de rechercher des réservations spécifiques en fonction du nom du patient ou du nom du médicament.	2	Nader ZOUAIRI
		1.13.4	Tester les fonctionnalités de recherche et de filtrage pour s'assurer qu'elles fonctionnent de manière fluide et efficace.	1	Nader ZOUAIRI

TABLE 4.1 – Sprint 2 Backlog

4.3 Partie Analyse

Durant cette partie, nous allons présenter notre schéma de cas d'utilisation globale qui vise cette sprit, ensuite le diagramme de classe d'analyse et enfin quelques diagrammes de raffinements pour certaines cas d'utilisations.

4.3.1 Diagramme de cas d'utilisation globale gestion du pharmacie

Dans la figure 4.1, on va présenter la partie concacrée à la gestion du pharmacie.

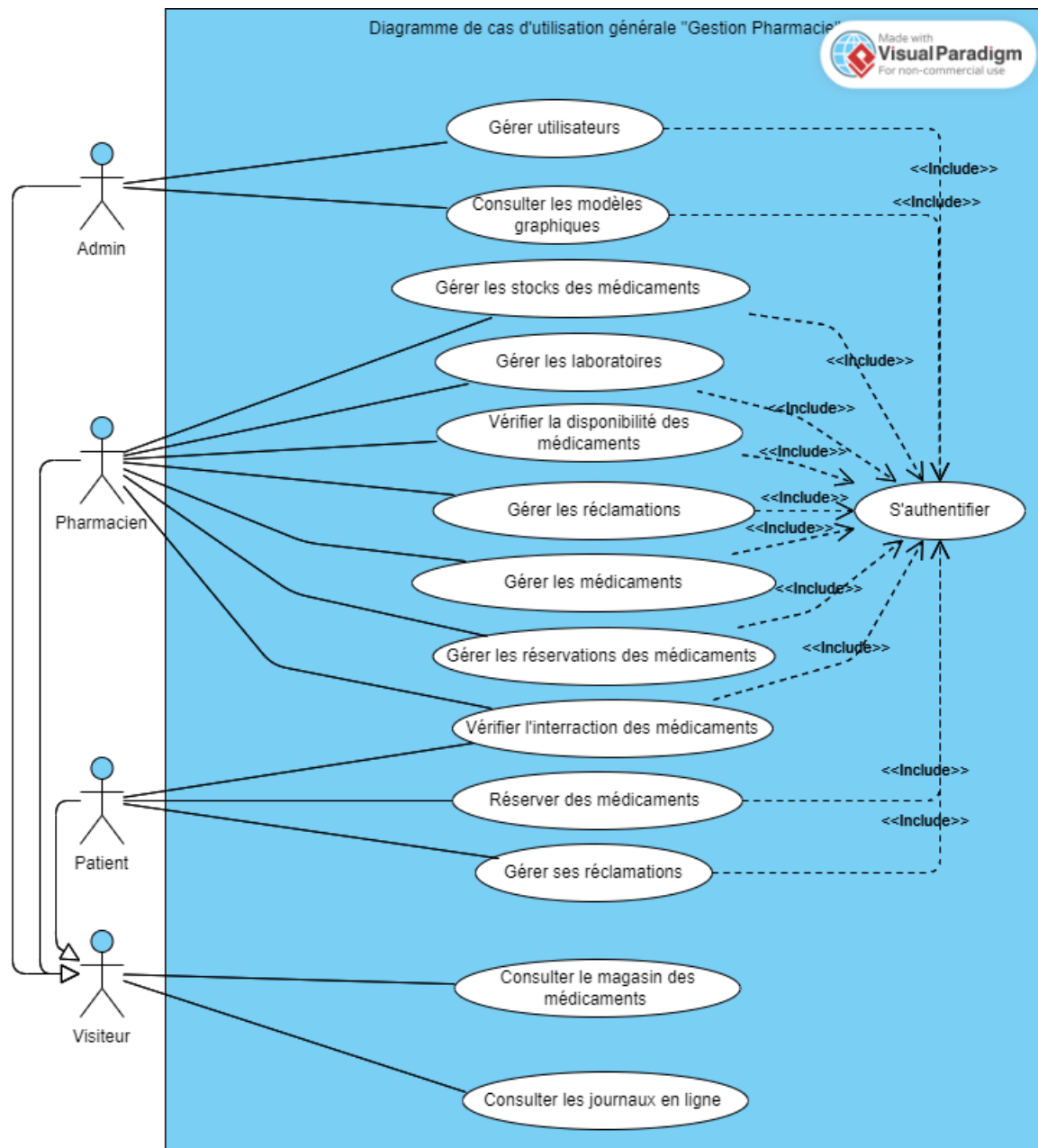


FIGURE 4.1 – Diagramme de cas d'utilisation générale "gestion du pharmacie"

Ce diagramme de cas d'utilisation modélise les interactions entre différents acteurs (administrateur, pharmacien, patient, visiteur) et un système de gestion de pharmacie. Il met en évidence les différentes fonctionnalités offertes par le système, organisées autour du cas d'utilisation central "S'authentifier" qui est inclus dans presque tous les autres cas d'utilisation, indiquant que l'authentification est nécessaire pour accéder à la plupart des fonctionnalités du système.

4.3.2 Diagramme de classe de conception niveau "Gestion Pharmacie"

Comme le montre la figure 4.2, le diagramme contient les classes manipulées au cours du développement de ce sprint au niveau du pharmacie.

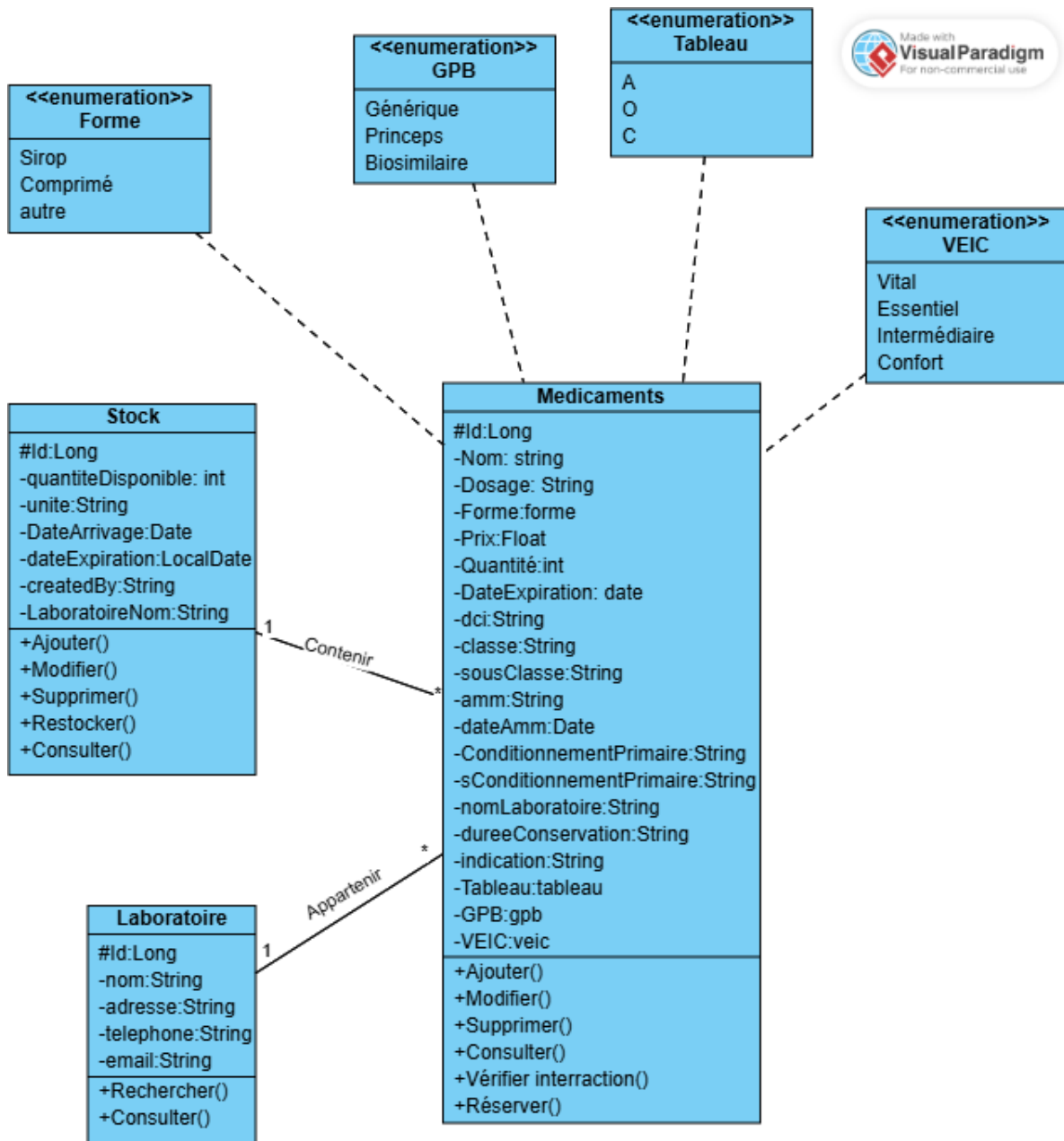


FIGURE 4.2 – Diagramme de classe de conception "Gestion Pharmacie"

Ce diagramme de classes UML représente un modèle conceptuel du système de gestion de pharmacie. Il décrit les différentes entités (médicaments, stock, laboratoire) et leurs relations, ainsi que les attributs et opérations associés à chacune.

4.3.3 Diagramme de cas d'utilisation raffiné "Gérer les stocks des médicaments"

Dans la figure 4.3, on va raffiner le cas d'utilisation "gérer les stocks des médicaments".

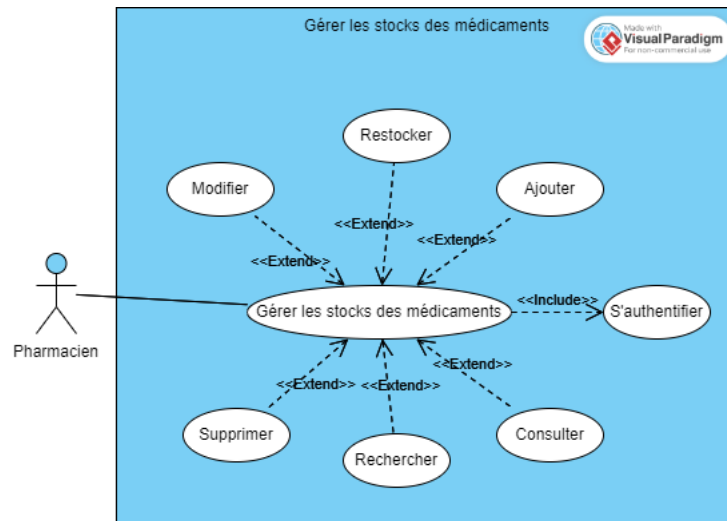


FIGURE 4.3 – Diagramme de cas d'utilisation "Gestion des stocks des médicaments"

Ce diagramme de cas d'utilisation se concentre spécifiquement sur les interactions d'un pharmacien avec le système de gestion des stocks de médicaments. Il met en évidence les différentes actions que le pharmacien peut effectuer sur les stocks, ainsi que les relations entre ces actions.

4.3.4 Diagramme de cas d'utilisation raffiné "Gérer les réclamations"

Comme précisé dans la figure 4.4, on va raffiner le cas d'utilisation du "gestion des réclamations".

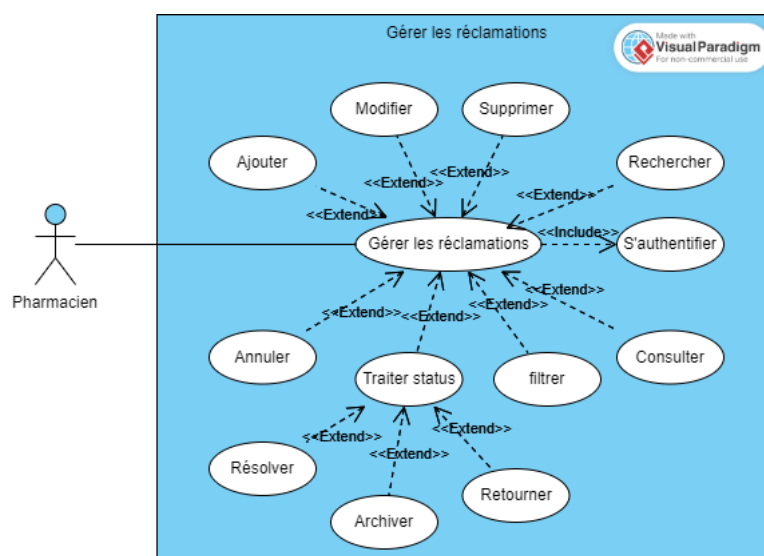


FIGURE 4.4 – Diagramme de cas d'utilisation "Gestion des réclamations"

4.3.5 Diagramme de cas d'utilisation raffiné "Gérer les réservations"

Comme présenté dans la figure 4.5, voila le raffinement du cas d'utilisation gestion des réservations.

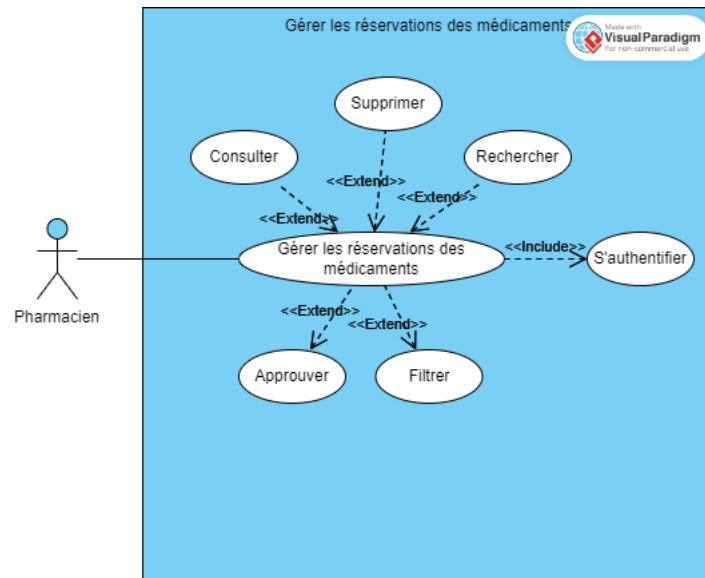


FIGURE 4.5 – Diagramme de cas d'utilisation "Gestion des réservations"

Ce diagramme de cas d'utilisation se concentre spécifiquement sur les interactions d'un pharmacien avec le système de gestion des réservations de médicaments. Il met en évidence les différentes actions que le pharmacien peut effectuer sur les réservations, ainsi que les relations entre ces actions.

4.3.6 Diagramme de cas d'utilisation raffiné "Gérer ses réclamations"

La figure 4.6, illustre le raffinement du cas d'utilisation de "gestion de ses récalamtions" pour le patient.

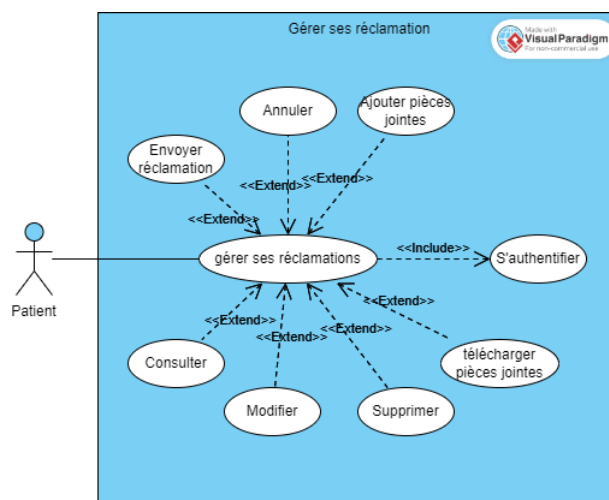


FIGURE 4.6 – Diagramme de cas d'utilisation "Gestion de ses réclamations"

4.3.7 Diagramme de cas d'utilisation raffiné "Réserver médicaments"

Comme exposé dans la figure 4.7, voici le raffinement du cas d'utilisation du réservation des médicaments.

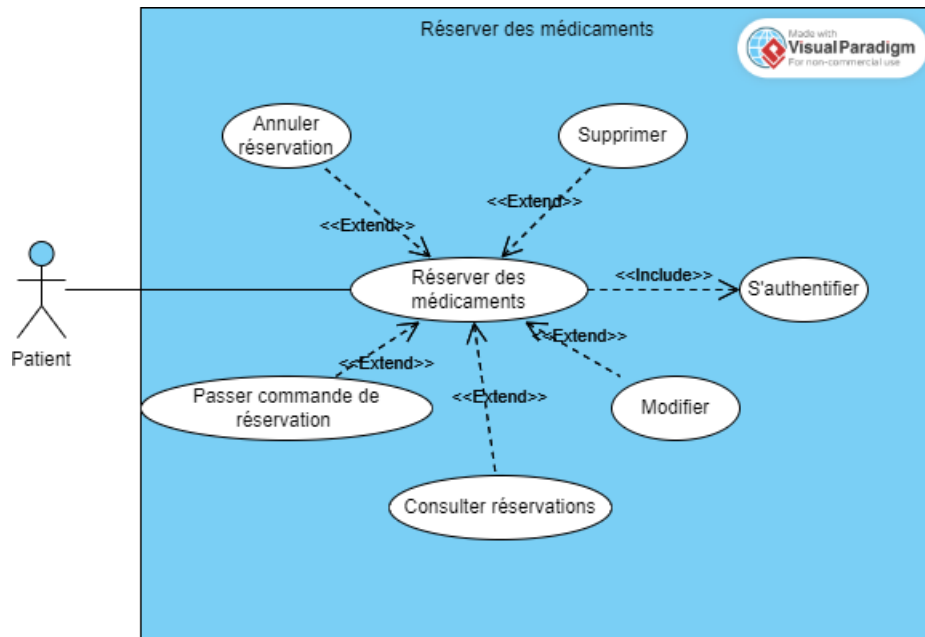


FIGURE 4.7 – Diagramme de cas d'utilisation "Réservation des médicaments"

4.3.8 Diagramme de cas d'utilisation raffiné "Gérer médicaments"

Comme mis en lumière par la figure 4.8, voici le raffinement du cas d'utilisation "réserver des médicaments"

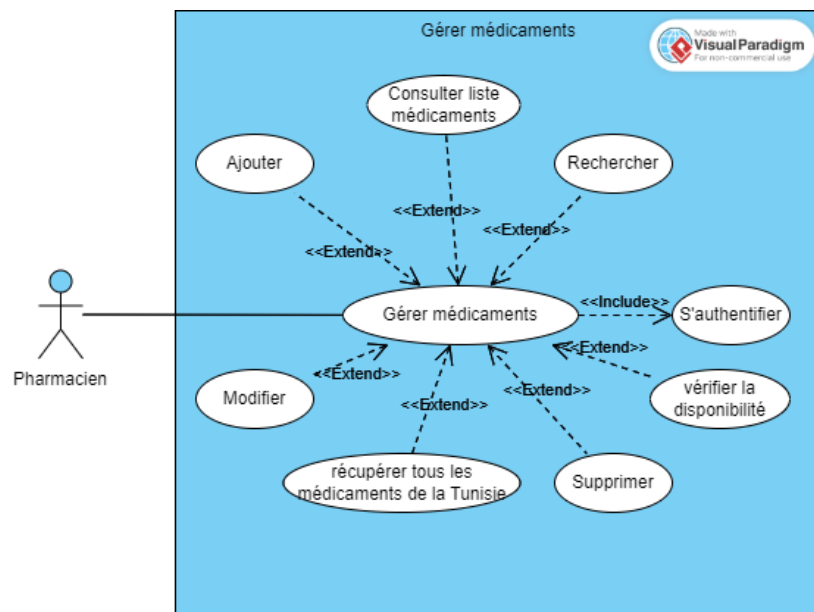


FIGURE 4.8 – Diagramme de cas d'utilisation "Gérer médicaments"

4.4 Partie Conception

4.4.1 Diagramme de séquence "Traiter status"

La figure 4.9 présente le diagramme de séquence du traitement des status.

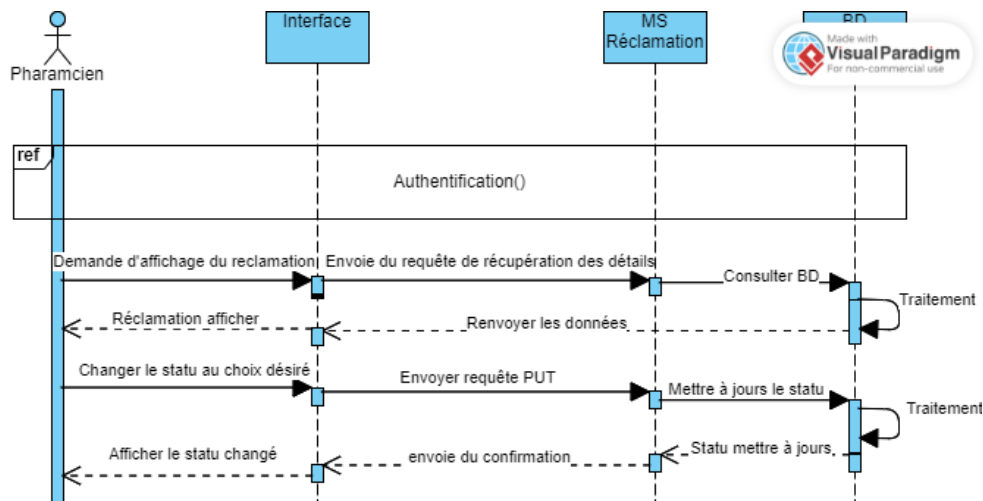


FIGURE 4.9 – Diagramme de séquence "Traiter staus"

Ce diagramme de séquence illustre les interactions entre un pharmacien et un système de gestion des réclamations (MS Réclamation) lors du processus de consultation et de mise à jour du statut d'une réclamation. Il décrit les échanges de messages entre les différents acteurs et les traitements effectués par le système.

4.4.2 Diagramme de séquence "Réserver médicaments"

La figure 4.10, démontre le diagramme de séquence pour la réservation des médicaments.

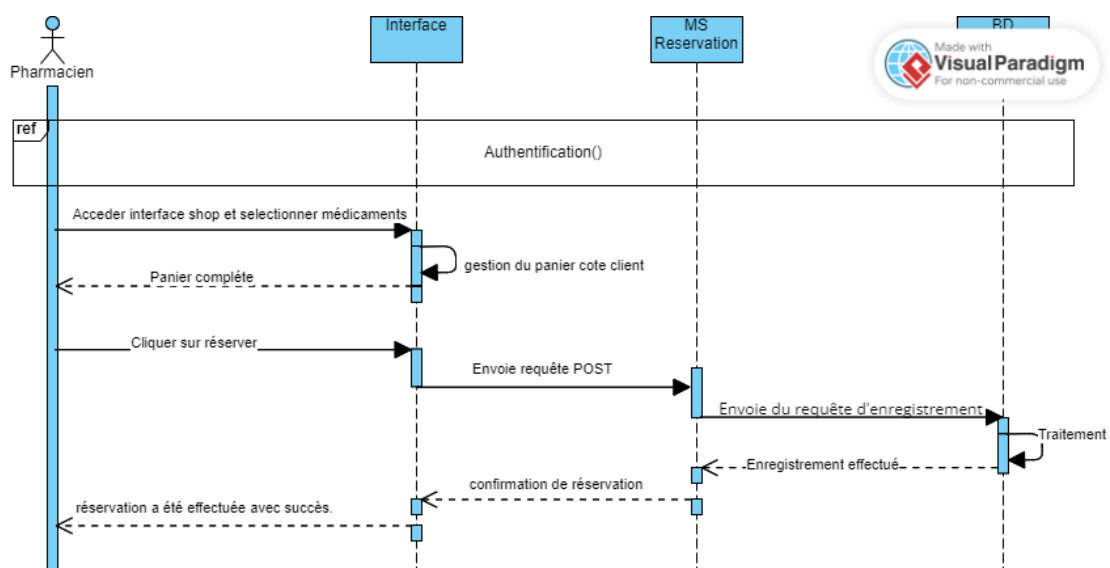


FIGURE 4.10 – Diagramme de séquence "Réserver médicaments"

4.4.3 Description textuelle de cas d'utilisation "Ajouter médicaments"

Le tableau 4.2, présente une description textuelle détaillée pour le cas d'utilisation d'ajout des médicaments

Élément	Description
Cas d'utilisation	Ajouter médicaments
Acteur	Pharmacien
Pré-condition	Le Pharmacien doit être connecté
Post-condition	Médicament ajouté avec succès
Scénario nominal	1- Le pharmacien saisie les informations nécessaires du médicament à ajouter. 2- Le système transmet le nom du médicament, la quantité, la date d'expiration, au Service réservation en backend. 3- Le système accède au fichier excel et parcourt le fichier pour vérifier si le médicament existe en comparant les noms. 4- Si une correspondance est trouvée, les détails du médicament sont extraits. 5- Un nouvel objet médicament est créé avec les informations récupérées et les informations de stock sont également définies. 6- Le système enregistre les objets respectivement dans les répertoires de données médicaments et stock dans la base de données.
Scénarios alternatifs	4a- Si une correspondance n'est pas trouvée le système affiche un message d'erreur.

TABLE 4.2 – Description textuelle du cas d'utilisation : Ajouter médicaments

4.5 Partie Réalisation

4.5.1 Affichage de liste des médicaments

La figure 4.11, illustre l'interface de gestion et consultation des médicaments.

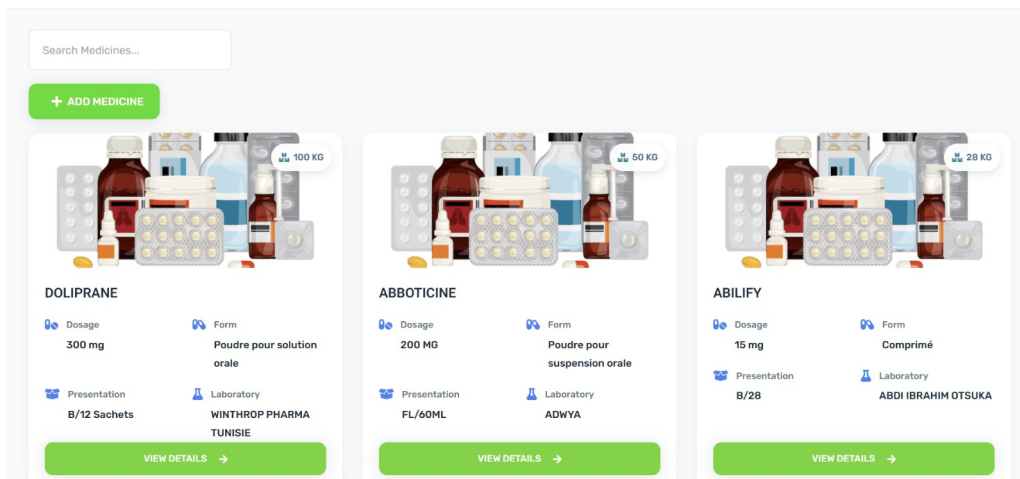


FIGURE 4.11 – Gestion et consultation des médicaments

Cette interface permet au pharmacien de gérer et de consulter les informations détaillées des médicaments. Elle comprend plusieurs fonctionnalités telles que la recherche avancée pour faciliter la recherche des médicaments spécifiques, l'ajout de médicament qui est une fonctionnalité essentielle pour mettre à jour la liste des médicaments disponibles aussi que la fiche de médicament où chacun est affiché sous forme de carte qui présente une image et des détails essentiels, notamment le dosage et le laboratoire ainsi qu'une bouton pour consulter les informations détaillées telque le stock disponible.

4.5.2 Traitement des réclamations

La figure 4.12, montre l'interface de traitement des réclamations.

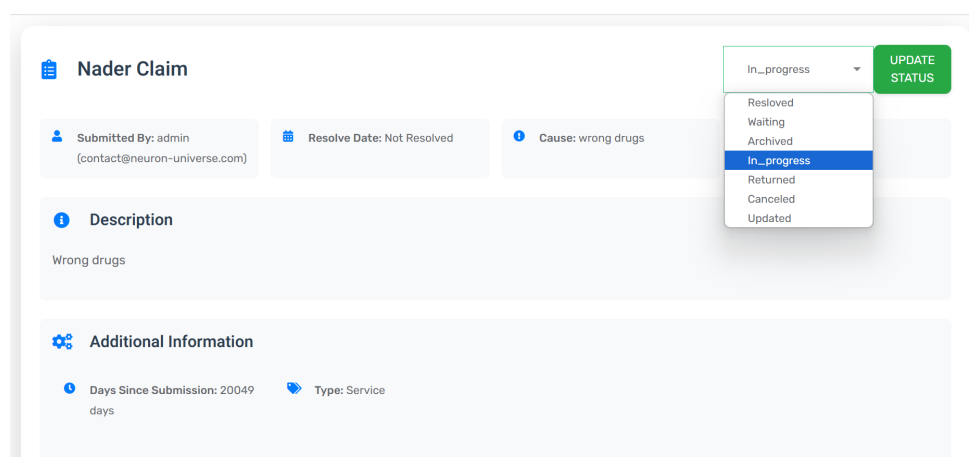


FIGURE 4.12 – Traitement du réclamation

L'objectif de cette interface est de faciliter aux pharmaciens la consultation de toutes les réclamations reçues. Elle est conçue pour être conviviale et faciliter l'accès aux informations des réclamations, puis les traiter facilement grâce à une recherche avancée pour récupérer des réclamations spécifiques selon l'état du patient, le nom, le statut, etc.

4.5.3 Suppression des réclamations

Selon la figure 4.13, voici l'interface dédié au suppression des réclamations.

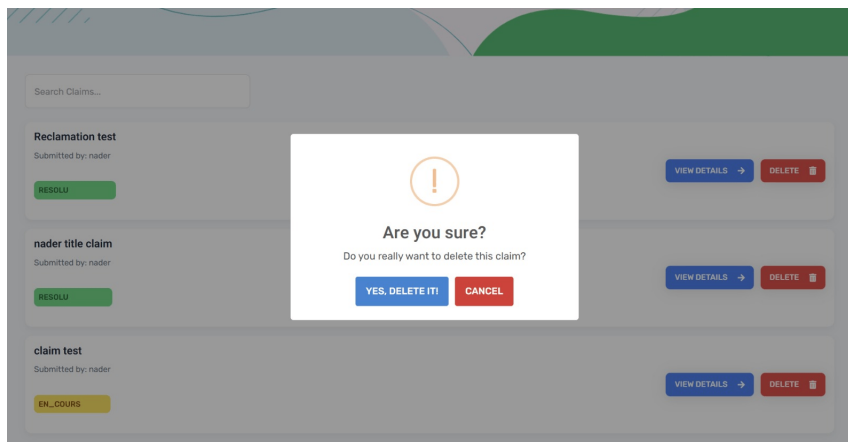


FIGURE 4.13 – Suppression des récalamtions

Cette interface présente une liste de réclamations soumises par divers utilisateurs pour le pharamcien. Chaque réclamation est accompagnée d'un titre, du nom de l'utilisateur qui l'a soumise et d'un bouton de statut supplémentaire.

Une fenêtre de confirmation s'ouvre au centre de l'écran lorsque le pharmacien souhaite supprimer une réclamation inutile, demandant la vérification avant de la supprimer. Ainsi, il peut confirmer la suppression ou annuler l'action.

4.5.4 Traitement d'une réclamation

La figure 4.14, décrit l'interface du traitement des réclamations.

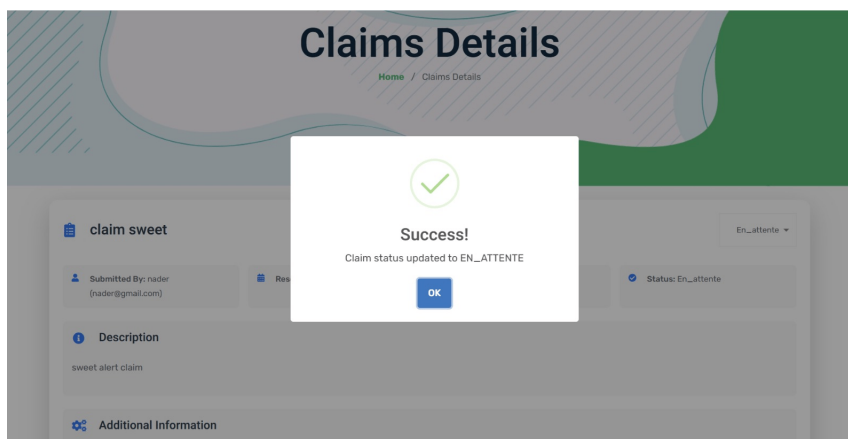


FIGURE 4.14 – Succés de traitement d'une réclamation

Cette interface est conçue pour le pharmacien, il lui montre les détails d'une réclamation soumise par un patient. L'interface comprend des informations telles que le nom du personne qui a soumis la réclamation, son adresse email, et une description de la réclamation. Un bouton permet de changer le statut de la réclamation.

Après une mise à jour du status, une fenêtre de notification de succès apparaît au centre de l'écran avec un message de succès.

4.5.5 Affichage du liste des laboratoires en Tunisie

L'illustration 4.15, présente l'interface de l'affichage du liste des laboratoires en Tunisie.

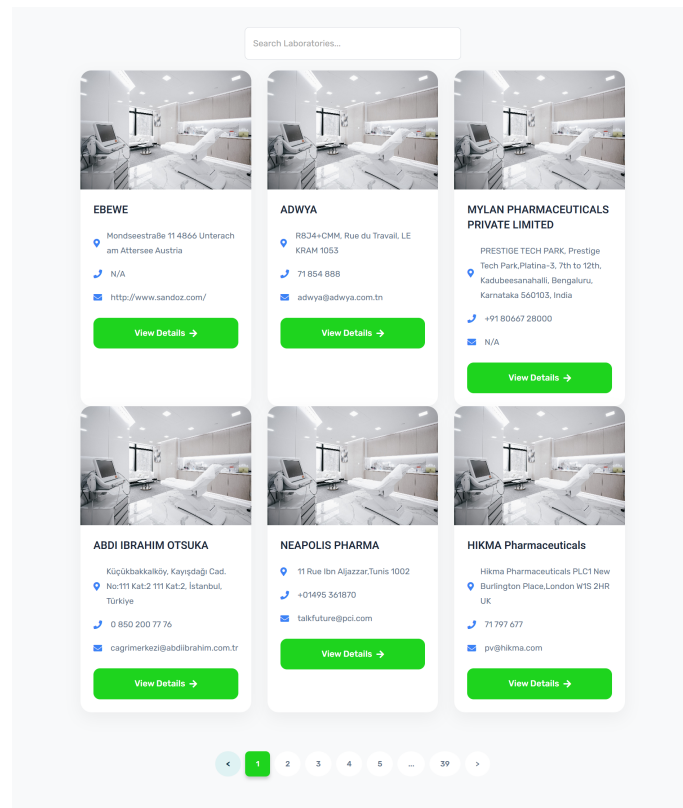


FIGURE 4.15 – Les laboratoires en Tunisie

Cette interface permet au pharmacien de consulter la liste complète des laboratoires en Tunisie, y compris leurs informations détaillées pour faciliter la gestion et le contact avec les fabricants de médicaments. Les fonctionnalités principales incluent la recherche de Laboratoire, la fiche Laboratoire, détails du Laboratoire avec un bouton « VIEW DETAILS » et une pagination est présente pour naviguer à travers une grande liste de laboratoires, permettant un accès rapide et structuré.

4.6 Conclusion

Après avoir achevé le développement du module de gestion du pharmacie, qui facilitera la gestion des médicaments, laboratoires, stocks, réclamations et réservations nous progressons vers le prochain sprint, qui contribue à l'objectif complet du projet.

Chapitre 5

Sprint 3 : Gestion des modèles graphiques

5.1	Introduction	68
5.2	Sprint Backlog 3	68
5.3	Partie Analyse	69
5.3.1	Diagramme de cas d'utilisation raffiné "Visualisation des modèles graphiques"	69
5.4	Partie Conception	69
5.4.1	Diagramme de séquence "Visualisation des modèles graphiques"	69
5.4.2	Description textuelle de cas d'utilisation "Visualiser les modèles graphiques"	70
5.5	Partie Réalisation	71
5.5.1	Affichage des modèles graphiques	71
5.6	Conclusion	72

5.1 Introduction

Dans ce chapitre, nous nous concentrerons sur la présentation des travaux effectués au cours du sprint 3. Nous débuterons avec le Sprint Backlog, suivi de la conception, et enfin, nous montrerons quelques interfaces pour illustrer la mise en œuvre.

5.2 Sprint Backlog 3

Comme présenté dans le tableau 5.1, on va détailler le backlog pour notre troisième sprint.

ID	User story	ID Tâche	Tâche	Estimation	Responsable
1.1	En tant qu'administrateur, je souhaite visualiser les réclamations via des statistiques pour évaluer la qualité du service et identifier les points à améliorer.	1.1.1	Intégration d'un service backend pour le traitement des données de réclamations et le calcul de statistiques.	3	Nader ZOUAIRI
		1.1.2	Développement du contrôleur pour récupérer les données de réclamations et les envoyer au frontend.	2	Nader ZOUAIRI
		1.1.3	Création de graphiques et visualisations pour afficher les statistiques des réclamations.	1	Nader ZOUAIRI
		1.1.4	Test et intégration pour valider les fonctionnalités statistiques.	1	Nader ZOUAIRI
		1.1.5	Ajustements de l'interface utilisateur pour inclure une page de visualisation des statistiques.	1	Nader ZOUAIRI

TABLE 5.1 – Sprint 3 Backlog

5.3 Partie Analyse

5.3.1 Diagramme de cas d'utilisation raffiné "Visualisation des modèles graphiques"

la figure 5.1, présente le raffinement du cas d'utilisation de visualisation des modèles graphiques.

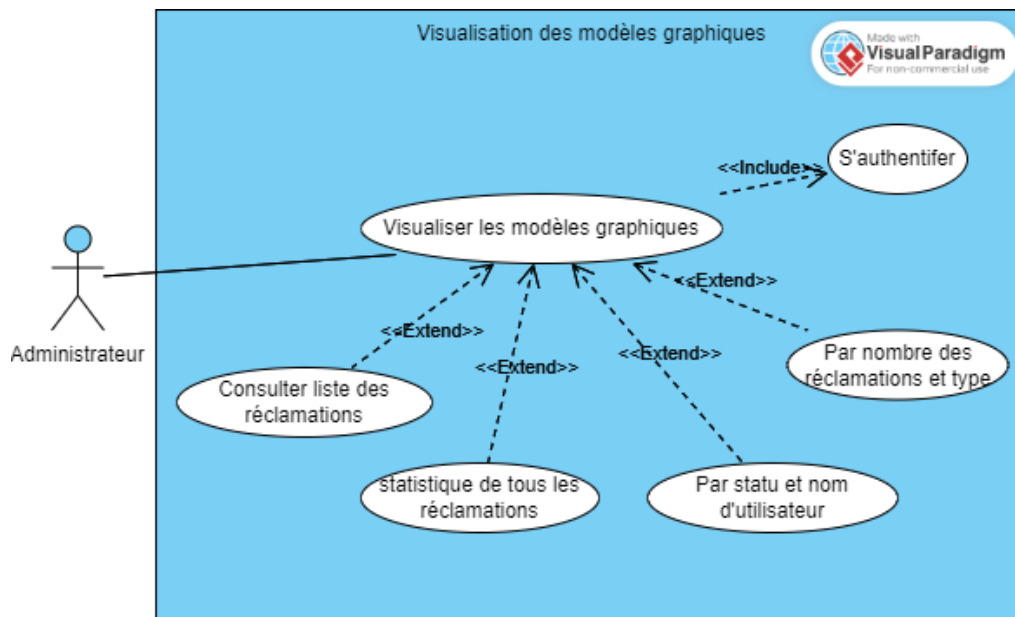


FIGURE 5.1 – Diagramme de cas d'utilisation "Visualisation des modèles graphiques"

Ce diagramme de cas d'utilisation décrit les différentes actions qu'un administrateur peut effectuer pour visualiser des modèles graphiques liés à la gestion des réclamations. Il met en évidence les différentes fonctionnalités offertes à l'administrateur pour analyser les données de manière visuelle.

5.4 Partie Conception

5.4.1 Diagramme de séquence "Visualisation des modèles graphiques"

La figure 5.2, présente le diagramme de séquence pour la visualisation des modèles graphiques.

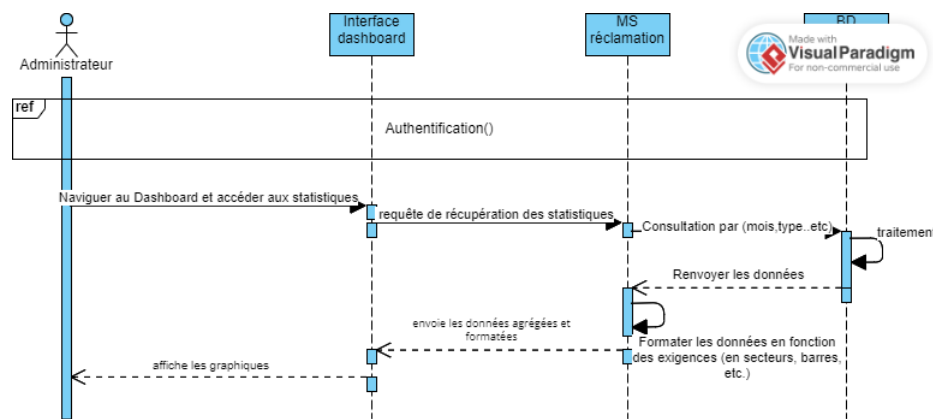


FIGURE 5.2 – Diagramme de séquence "Visualisation des modèles graphiques"

Ce diagramme de séquence illustre le processus d'interaction entre l'administrateur et le système de gestion des réclamations lors de la visualisation de modèles graphiques. Il décrit de manière détaillée les échanges de messages et les traitements effectués par le système pour répondre à la demande de l'administrateur.

5.4.2 Description textuelle de cas d'utilisation "Visualiser les modèles graphiques"

Le tableau 5.2, trace le démarche du cas d'utilisation "Visualiser les modèles graphiques".

Élément	Description
Cas d'utilisation	Visualiser les modèles graphiques
Acteur	Administrateur
Pré-condition	L'administrateur doit être connecté
Post-condition	Affichage des modèles graphiques
Scénario nominal	1- L'administrateur accède à l'interface dashboard et sélectionne l'option de visualisation des statistiques de réclamations. 2- L'interface dashboard envoie une requête au microservice "Réclamation" pour obtenir les statistiques. 3- Le microservice "Réclamation" récupère les données de la base de données en comptant les réclamations par statut, par type et par mois. 4- Les données sont formatées par le microservice "Réclamation" sous forme de modèles graphiques. 5- L'interface dashboard reçoit les données formatées et génère les graphiques correspondants (camembert pour les statuts, barres pour les tendances mensuelles...etc). 6- Le système affiche les graphiques de réclamations pour l'administrateur sur le dashboard.

TABLE 5.2 – Description textuelle du cas d'utilisation : Ajouter médicaments

5.5 Partie Réalisation

5.5.1 Affichage des modèles graphiques

Dans la figure 5.3, on présente l'interface du dashboard admin.

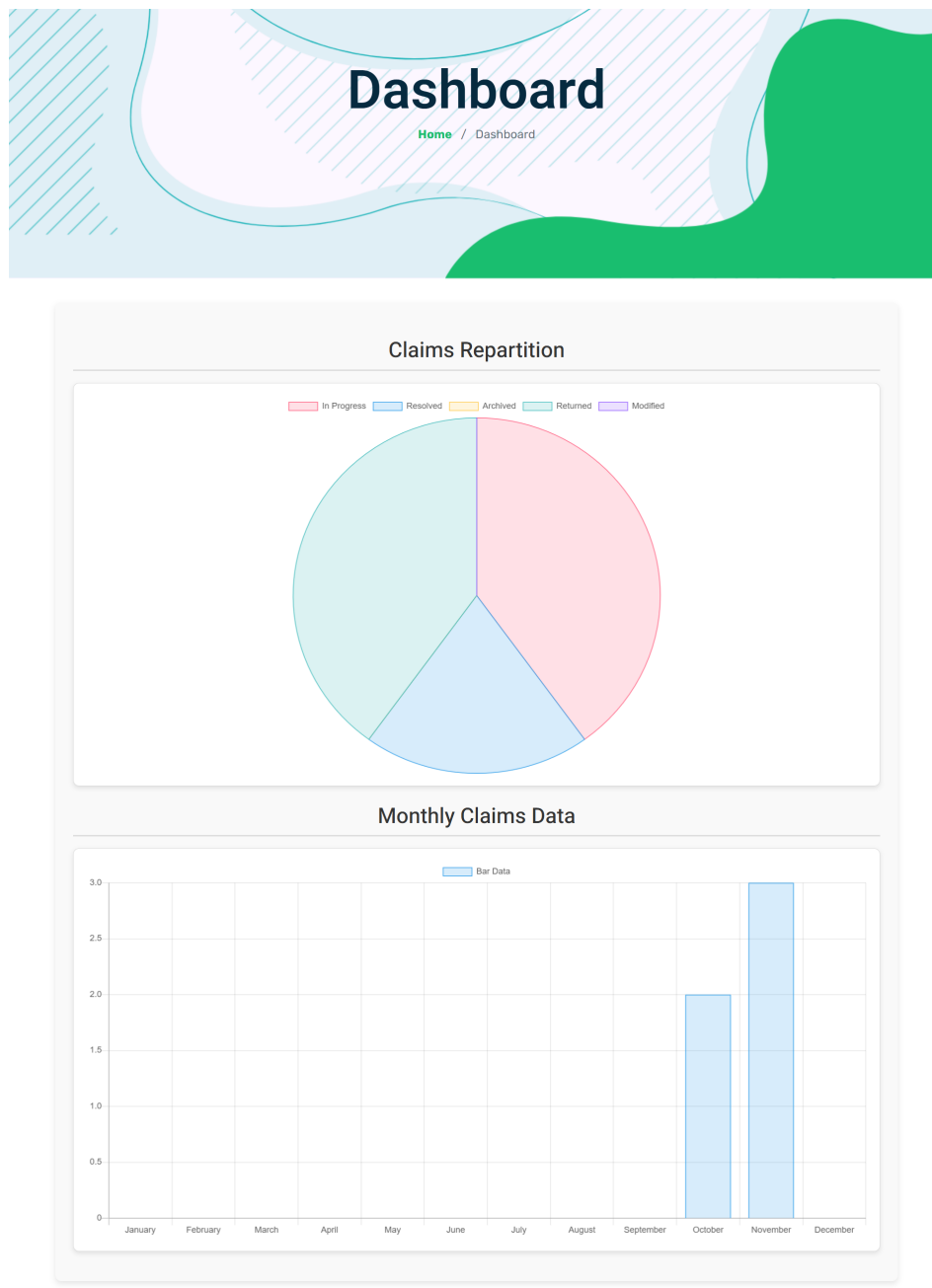


FIGURE 5.3 – modèles graphiques pour l'administrateur

Cette interface est un tableau de bord destiné uniquement à l'admin pour visualiser les réclamations des patients et obtenir une vue d'ensemble. Elle comporte deux graphiques principaux : Répartition des réclamations : Un graphique circulaire montrant la répartition des réclamations par statut, avec différentes catégories telles que "In Progress" (en rose), "Resolved" (en vert), "Archived" (en jaune), "Returned" (en violet), et "Modified" (en bleu). Ce graphique permet à

l'admin de voir rapidement la proportion des réclamations dans chaque statut.

Données mensuelles des réclamations : Un graphique en barres qui représente le nombre de réclamations par mois, facilitant le suivi des réclamations au fil du temps et permettant d'identifier les périodes de forte activité en matière de réclamations.

5.6 Conclusion

Après avoir développé le module de visualisation des modèles graphiques, qui facilitera la visualisation des réclamations et les affichera clairement pour l'administrateur, nous nous préparons pour le prochain sprint, axé sur l'optimisation supplémentaire des objectifs du projet.

Chapitre 6

Sprint 4 : Gestion des dossiers médicaux

6.1	Introduction	74
6.2	Sprint Backlog 4	74
6.3	Partie Analyse	79
6.3.1	Diagramme de cas d'utilisation global : "Gestion dossiers/rapports/Ordonnances"	79
6.3.2	Diagramme de cas d'utilisation raffiné "Gestion des dossiers"	80
6.3.3	Diagramme de cas d'utilisation raffiné "Rédaction du rapport"	80
6.4	Partie Conception	81
6.4.1	Diagramme de classe de conception au niveau "Gestion des dossiers médicaux"	81
6.4.2	Diagramme de séquence "Consulter le dossier médical"	82
6.4.3	Description textuelle de cas d'utilisation "Ajouter pièces jointes"	82
6.5	Partie Réalisation	83
6.5.1	Interface liste des dossiers	83
6.5.2	Gestion du dossier d'un patient	84
6.5.3	Affichage de son ordonnance	85
6.5.4	Téléchargement et impression de l'ordonnance	85
6.6	Conclusion	86

6.1 Introduction

Dans ce chapitre, nous nous concentrerons sur la description du travail accompli durant le sprint 4. Nous débuterons par le Sprint Backlog, suivi par la partie conception et on conclut par une présentation des réalisations.

6.2 Sprint Backlog 4

Comme présenté dans le tableau 6.1, on va détailler le backlog pour notre quatrième sprint.

ID	User story	ID Tâche	Tâche	Estimation	Responsable
1.1	En tant que médecin, je veux rédiger une ordonnance ou un rapport médical électronique pour mes patients afin de documenter les traitements et diagnostics.	1.1.1	Créer des endpoints dans OrdonnanceController et RapportController pour rédiger et sauvegarder une ordonnance et un rapport électroniques.	3	Nader ZOUAIRI
		1.1.2	Implémenter la validation des données dans le service pour vérifier l'existence du dossier patient et les médicaments via le microservice de médicaments.	1	Nader ZOUAIRI
		1.1.3	Ajouter une logique de vérification d'accès pour garantir que seuls les médecins peuvent rédiger ces documents.	2	Nader ZOUAIRI
		1.1.4	Ajouter la gestion des erreurs et des réponses HTTP appropriées dans OrdonnanceController pour les cas d'ordonnances invalides ou de dossiers manquants.	3	Nader ZOUAIRI

		1.1.5	Intégrer les fonctionnalités dans l'interface utilisateur pour permettre aux médecins de rédiger et consulter.	3	Nader ZOUAIRI
1.2	Comme médecin ou secrétaire, je veux consulter toutes les ordonnances ou les détails d'une ordonnance spécifique pour suivre l'historique médical des patients.	1.2.1	Créer un endpoint dans OrdonnanceController pour récupérer la liste complète des ordonnances d'un médecin.	1	Nader ZOUAIRI
		1.2.2	Créer un endpoint dans OrdonnanceController pour récupérer les détails d'une ordonnance spécifique par son identifiant (ID).	1	Nader ZOUAIRI
		1.2.3	Implémenter un service pour rechercher les ordonnances par critères spécifiques (nom du patient, code CNAM, etc.).	1	Nader ZOUAIRI
		1.2.4	Utiliser l'authentification Keycloak dans les endpoints pour obtenir le nom d'utilisateur du médecin ou secrétaire, limitant l'accès aux données.	2	Nader ZOUAIRI
		1.2.5	Intégrer les fonctionnalités dans l'interface utilisateur pour permettre aux médecins ou secrétaire de consulter toutes la liste ou les détails d'une ordonnance.	1	Nader ZOUAIRI

1.3	En tant que médecin ou secrétaire, je veux modifier ou supprimer des ordonnances ou rapports inutiles pour garder les dossiers à jour.	1.3.1	Créer un endpoint dans OrdonnanceController pour mettre à jour une ordonnance spécifique.	1	Nader ZOUAIRI
		1.3.2	Implémenter la logique de mise à jour dans le service, incluant la validation des médicaments et l'association au dossier patient.	1	Nader ZOUAIRI
		1.3.3	Créer un endpoint dans OrdonnanceController pour supprimer une ordonnance spécifique par son identifiant (ID).	1	Nader ZOUAIRI
		1.3.4	Implémenter la logique de suppression dans le service pour vérifier l'existence de l'ordonnance avant de la supprimer.	1	Nader ZOUAIRI
		1.3.5	Utiliser l'authentification Keycloak pour vérifier que l'utilisateur a le rôle "Medecin" ou "secrétaire" pour les opérations de mise à jour.	1	Nader ZOUAIRI
		1.3.6	Intégrer les fonctionnalités dans l'interface utilisateur pour permettre aux médecins de modifier ou supprimer une ordonnance ou un rapport.	3	Nader ZOUAIRI

1.4	Comme médecin ou secrétaire, je souhaite ajouter des pièces jointes (résultats d'examen, analyses..etc) à un rapport médical pour documenter des informations supplémentaires.	1.4.1	Créer un endpoint dans RapportController pour l'ajout des pièces jointes	3	Nader ZOUAIRI
		1.4.2	Implémenter la logique d'ajout dans le service.	4	Nader ZOUAIRI
		1.4.3	Utiliser l'authentification Keycloak pour vérifier que l'utilisateur a le rôle "Medecin" ou "secrétaire" pour l'opération d'ajout des pièces jointes.	3	Nader ZOUAIRI
		1.4.4	Intégrer les fonctionnalités dans l'interface utilisateur pour permettre aux médecins et secrétaire d'ajouter les pièces jointes à un rapport.	2	Nader ZOUAIRI
1.5	En tant que secrétaire, je veux générer un dossier médical pour chaque patient afin de centraliser leurs informations médicales.	1.5.1	Créer un endpoint dans DossierController pour la génération d'un dossier médical.	2	Nader ZOUAIRI
		1.5.2	Implémenter la logique de génération de dossier dans le DossierService.	3	Nader ZOUAIRI
		1.5.3	Utiliser l'authentification Keycloak pour vérifier que l'utilisateur a le rôle "Secrétaire" pour créer des dossiers.	3	Nader ZOUAIRI

		1.5.4	Intégrer les fonctionnalités dans l'interface utilisateur pour permettre au secrétaire de générer des dossiers médicaux pour les patients.	2	Nader ZOUAIRI
1.6	Comme secrétaire, je veux envoyer les dossiers médicaux aux patients pour leur fournir un accès à leurs informations de santé.	1.6.1	Ajouter un bouton dans l'interface utilisateur.	1	Nader ZOUAIRI
		1.6.2	Créer une fonction JavaScript pour appeler l'API d'envoi.	3	Nader ZOUAIRI
		1.6.3	Ajouter un endpoint dans DossierController pour envoyer le dossier.	2	Nader ZOUAIRI
		1.6.4	Implémenter la logique d'envoi d'email dans DossierService.	1	Nader ZOUAIRI
		1.6.5	Créer un EmailService pour gérer l'envoi d'emails, incluant éventuellement le dossier en pièce jointe.	1	Nader ZOUAIRI
1.7	En tant que patient, je souhaite télécharger les pièces jointes d'un rapport médical pour examiner des résultats ou des documents associés.	1.7.1	Endpoint downloadAttachment dans RapportController pour permettre le téléchargement de pièces jointes.	2	Nader ZOUAIRI
		1.7.2	Méthode loadAttachment dans RapportService pour charger le fichier depuis le répertoire des uploads.	2	Nader ZOUAIRI

		1.7.3	Vérification d'autorisation pour garantir que seul le patient propriétaire du rapport peut accéder au fichier.	2	Nader ZOUAIRI
		1.7.4	Mise à jour de l'interface utilisateur pour permettre le téléchargement des fichiers.	2	Nader ZOUAIRI

TABLE 6.1 – Sprint 4 Backlog

6.3 Partie Analyse

6.3.1 Diagramme de cas d'utilisation global : "Gestion dossiers/rapports/Ordonnances"

Dans la figure 6.1, on va présenter la partie concacrée pour la gestion des dossiers médicaux.

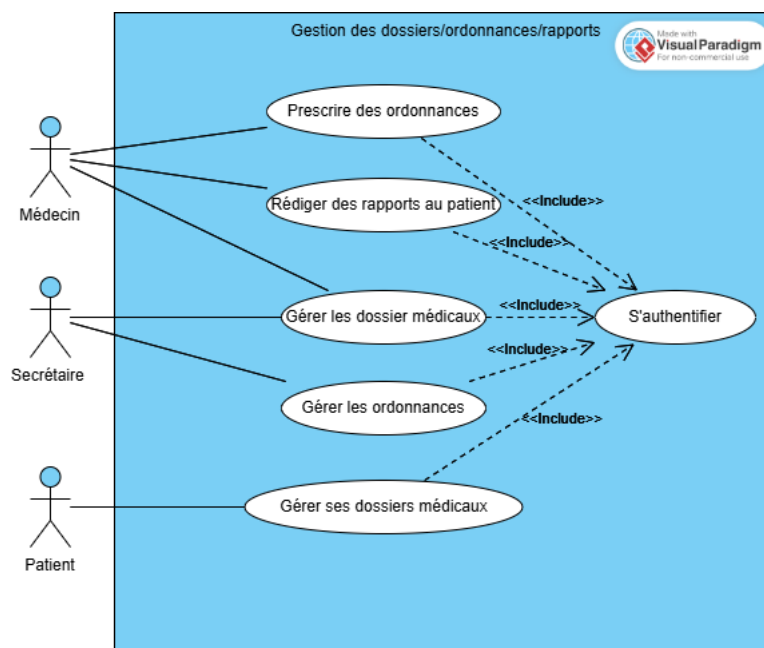


FIGURE 6.1 – Diagramme de cas d'utilisation "Gestion des dossiers/Rapports/Ordonnances"

Ce diagramme de cas d'utilisation représente les différentes fonctionnalités du système de gestion médicale, centré sur les interactions entre trois acteurs principaux : le médecin, le secrétaire et le patient. Il met en évidence les actions que chacun de ces acteurs peut effectuer et les relations entre ces actions.

6.3.2 Diagramme de cas d'utilisation raffiné "Gestion des dossiers"

La figure 6.2, précise le raffinement du cas d'utilisation "Gestion des dossiers médicaux"

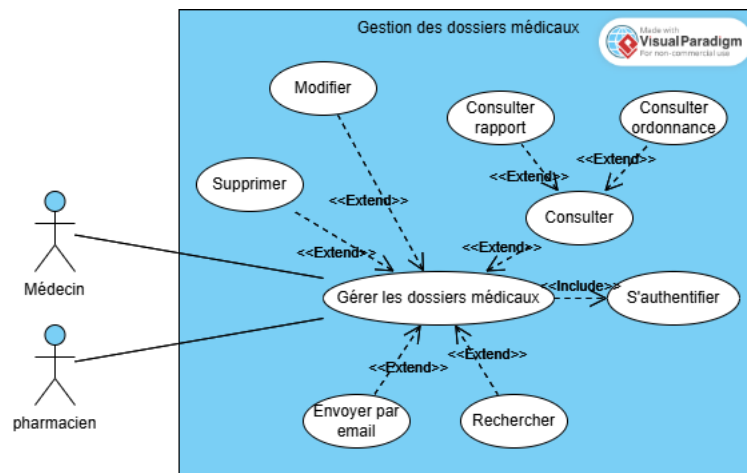


FIGURE 6.2 – Diagramme de cas d'utilisation "Gestion des dossiers médicaux"

Ce diagramme de cas d'utilisation représente les différentes fonctionnalités du système de gestion des dossiers médicaux, centré sur les interactions entre deux acteurs principaux : le médecin et le pharmacien. Il met en évidence les actions que chacun de ces acteurs peut effectuer sur les dossiers médicaux et les relations entre ces actions.

6.3.3 Diagramme de cas d'utilisation raffiné "Rédaction du rapport"

La figure 6.3, précise le raffinement du cas d'utilisation "rédaction du rapport"

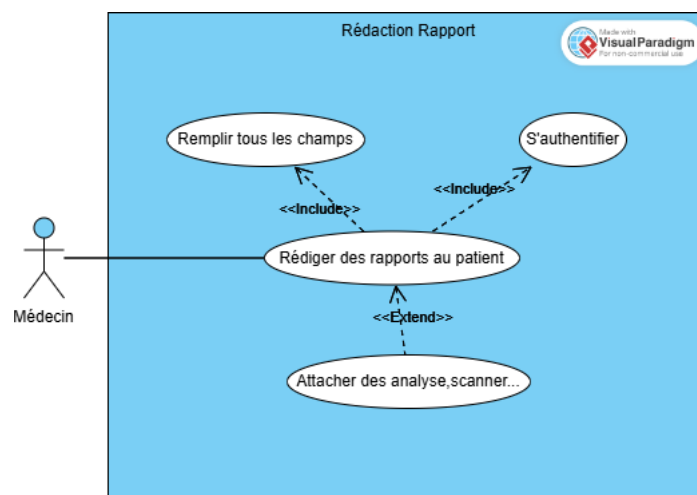


FIGURE 6.3 – Diagramme de cas d'utilisation "Rédaction du rapport"

Ce diagramme de cas d'utilisation se concentre spécifiquement sur la fonctionnalité de rédaction du rapport au patient dans le système de gestion médicale. Il met en évidence les actions qu'un médecin doit effectuer pour réaliser cette tâche, ainsi que les relations entre ces actions.

6.4 Partie Conception

6.4.1 Diagramme de classe de conception au niveau "Gestion des dossiers médicaux"

La figure 6.4, examine le diagramme de classe de conception au niveau du phase de gestion des dossiers médicaux.

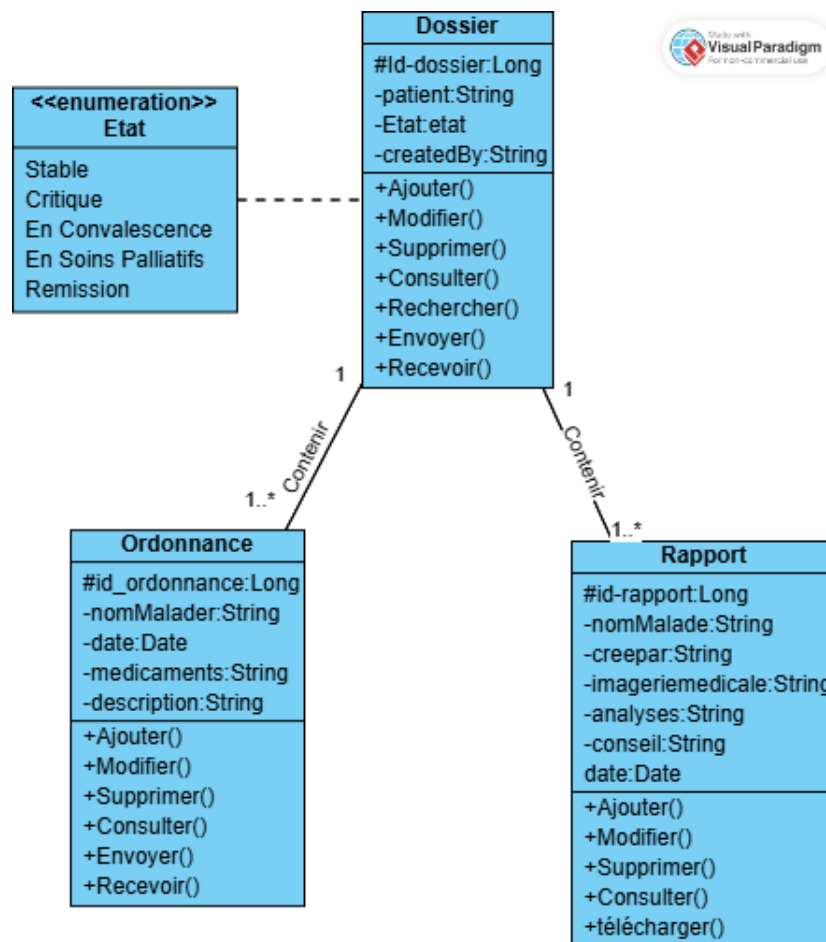


FIGURE 6.4 – Diagramme de classe de conception niveau "Gestion des dossiers médicaux"

Ce diagramme de classes UML modélise le système de gestion des dossiers médicaux. Il définit trois classes principales : Dossier, Ordonnance et Rapport. Ces classes sont liées entre elles par des relations d'association et représentent les éléments clés d'un dossier médical.

6.4.2 Diagramme de séquence "Consulter le dossier médical"

la figure 6.5, illustre le diagramme de séquence de consultation des dossiers médicaux.

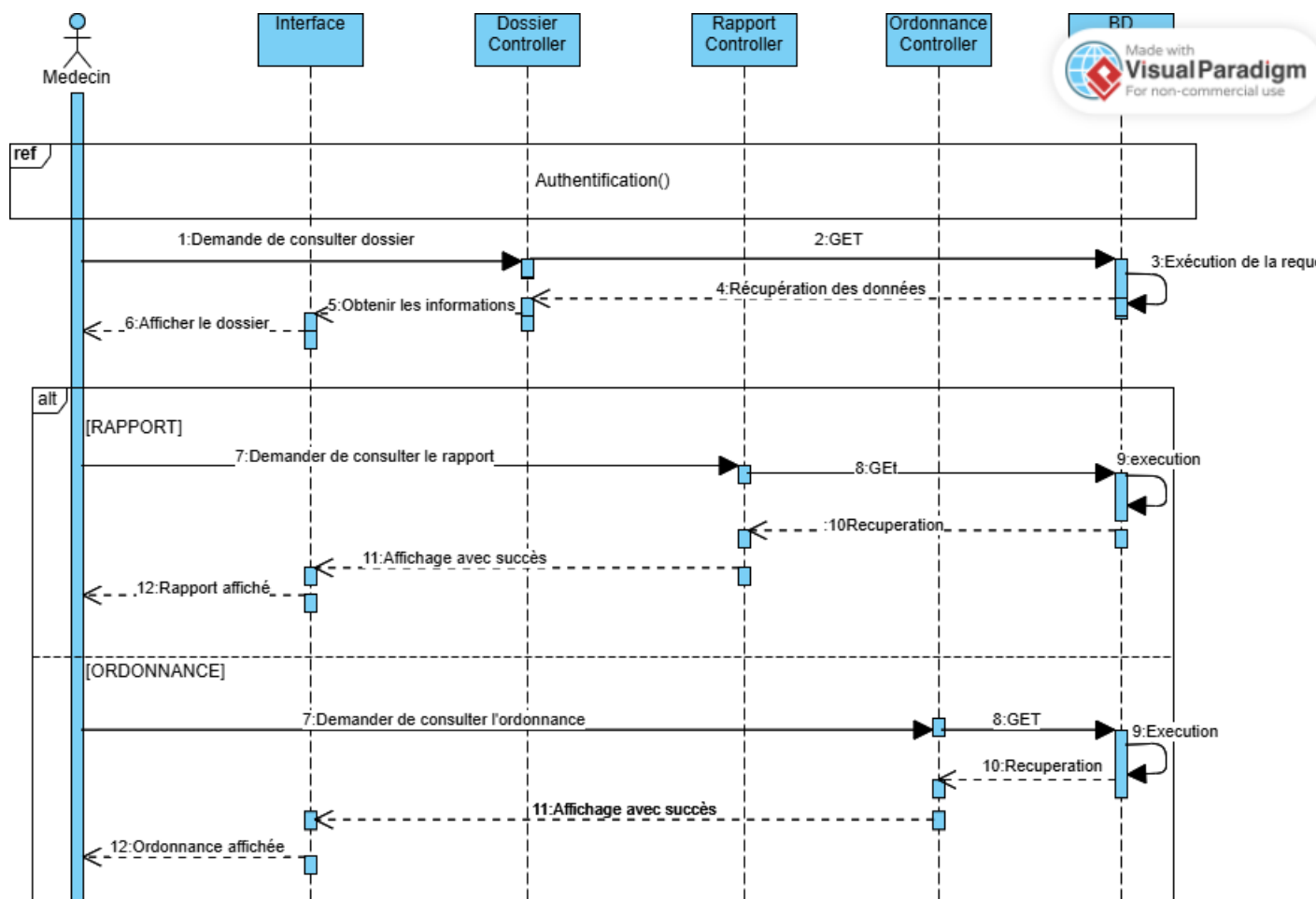


FIGURE 6.5 – Diagramme de séquence "Consulter le dossier médical"

Ce diagramme de séquence UML décrit le déroulement d'une interaction entre un médecin et le système de gestion de dossiers médicaux. Il modélise le processus de consultation d'un dossier médical, y compris les différentes étapes et les acteurs impliqués et il permet de visualiser les interactions entre les différents composants du système et de s'assurer que les exigences fonctionnelles sont bien prises en compte.

6.4.3 Description textuelle de cas d'utilisation "Ajouter pièces jointes"

Le tableau 6.2, expose la description textuelle du cas d'utilisation d'ajout des pièces jointes.

Élément	Description
Cas d'utilisation	Ajouter des pièces jointes au dossier médical (scanners, analyses, etc.)
Acteur	Médecin
Pré-condition	Le Médecin doit être connecté et avoir un dossier médical existant à compléter.
Post-condition	Pièce jointe ajoutée.
Scénario nominal	1- Le médecin sélectionne le dossier médical auquel il souhaite ajouter des pièces jointes. 2- Le médecin charge les fichiers des pièces jointes (scanners, analyses, etc.) via l'interface utilisateur. 3- Le système transmet les fichiers chargés au Attachment-Service en backend. 4- Le AttachmentService enregistre chaque fichier dans un répertoire spécifique et génère des métadonnées (nom, type, date d'ajout, etc.). 5- Un lien est créé entre les fichiers enregistrés et le dossier médical dans la base de données. 6- La pièce jointe est ajoutée avec succès.
Scénarios alternatifs	4a- Si le fichier est trop volumineux ou d'un format non supporté, le système affiche un message d'erreur indiquant que l'ajout a échoué.

TABLE 6.2 – Description textuelle du cas d'utilisation : Ajouter pièce jointe

6.5 Partie Réalisation

6.5.1 Interface liste des dossiers

La figure 6.6, présente l'interface de l'affichage du liste des dossiers.

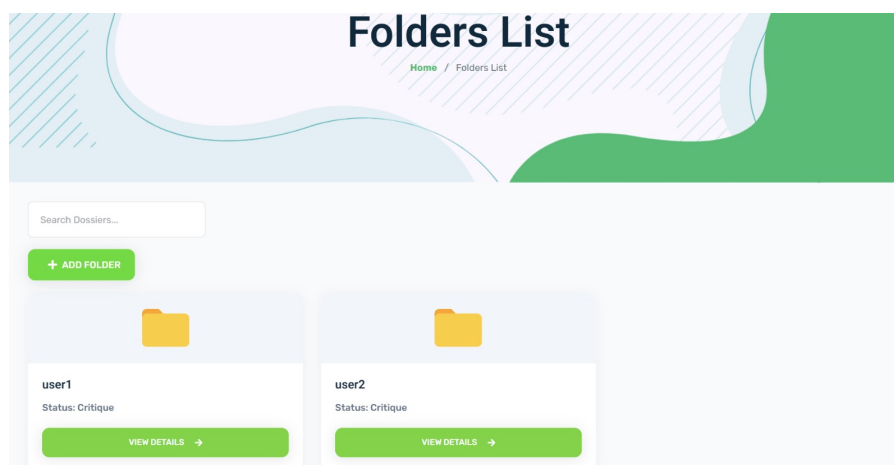


FIGURE 6.6 – Liste des dossiers médicaux

Cette interface a été conçue pour optimiser la gestion des dossiers médicaux des patients. Elle offre au medecin et secrétaire un accès rapide et intuitif à toutes les informations médicales

des patients avec la possibilité de consulter les informations détaillées d'un dossier, la recherche pour retrouver rapidement un dossier spécifique et l'ajout d'un nouveau dossier pour enregistrer facilement les informations d'un nouveau patient et créer son dossier médical, ainsi que l'apparence du status de chaque dossier qui permet de prioriser le travail et de suivre les dossiers qui nécessitent une attention particulière.

6.5.2 Gestion du dossier d'un patient

La figure 6.7, identifie l'interface de gestion des dossier des patients.

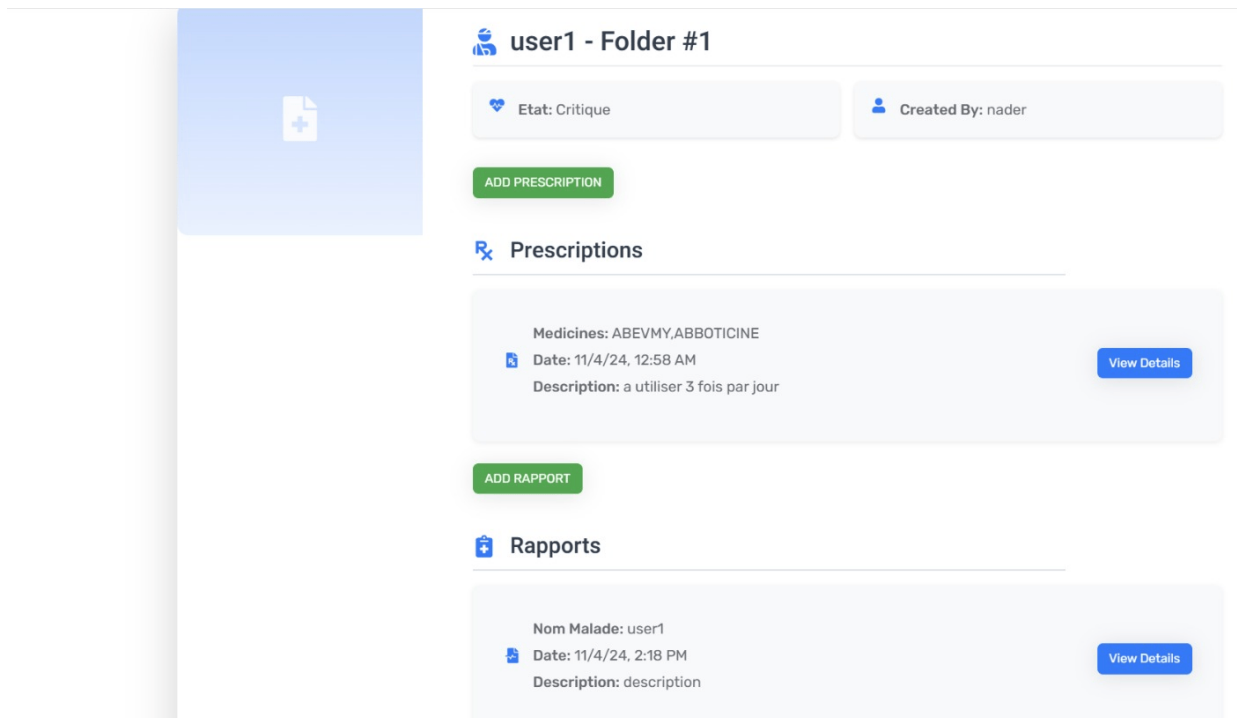


FIGURE 6.7 – Dossier médical d'un patient

Cette interface est conçue pour offrir une vue d'ensemble complète et actualiser le dossier médical d'un patient en centralisant les informations médicales, les ordonnances et les rapports d'un patient.

Elle permet au médecin de consulter rapidement les informations essentielles (antécédents, traitements, etc.), ajouter de nouvelle ordonnance ou rapport, suivre l'évolution de l'état de santé du patient et faciliter la collaboration entre les différents acteurs médicaux.

6.5.3 Affichage de son ordonnance

La figure 6.8, met en évidence l’affichage de l’ordonnance.

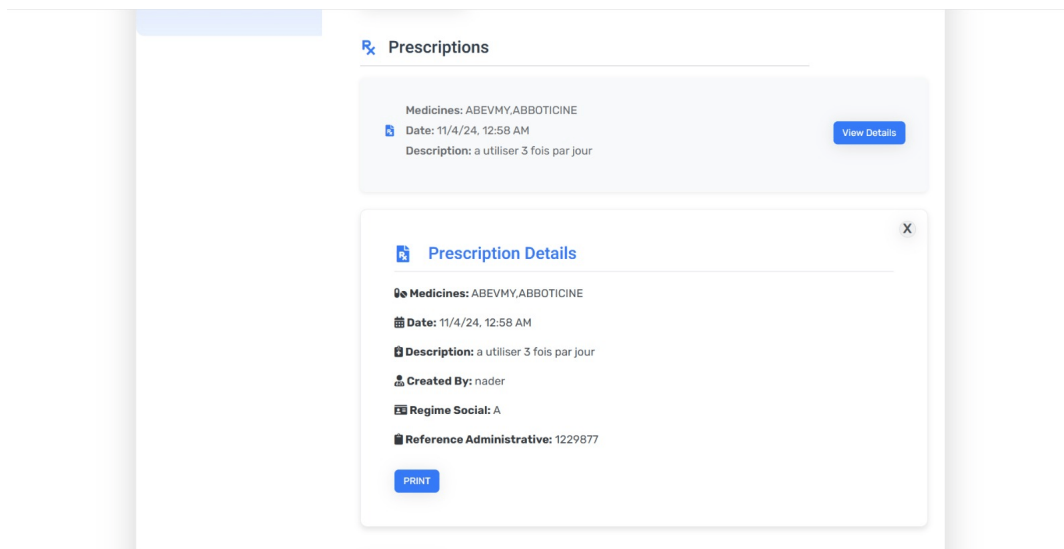


FIGURE 6.8 – Affichage de l’ordonnance

Grâce à cette interface, le patient peut accéder à ses ordonnances et leurs détails en toute sécurité pour suivre ses traitements et consulter les détails des médicaments prescrits par son médecin et aussi le pouvoir de l’imprimer ou la télécharger. Un outil pratique pour mieux gérer sa santé au quotidien.

6.5.4 Téléchargement et impression de l’ordonnance

La figure 6.9, met en évidence l’interface de téléchargement de l’ordonnance.

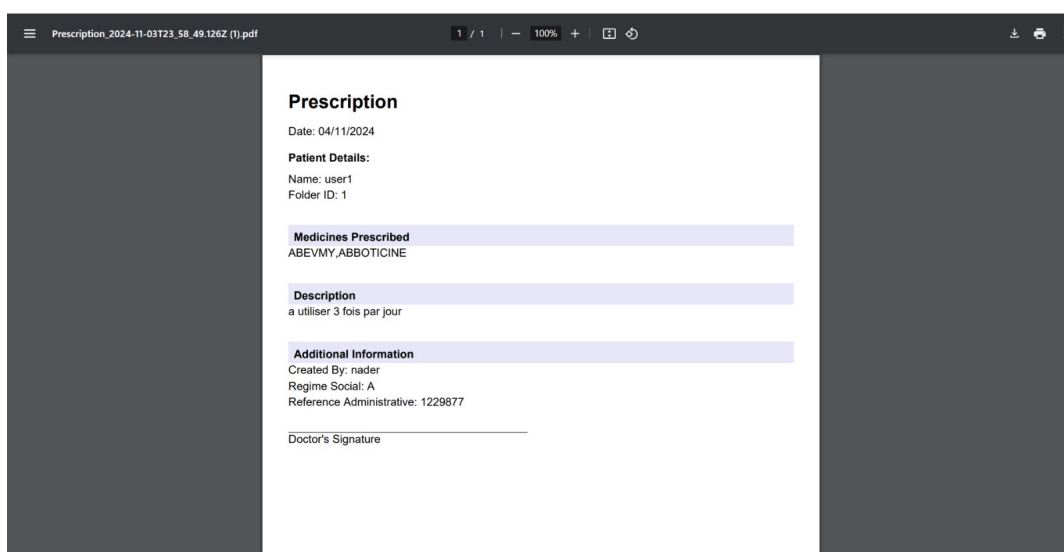


FIGURE 6.9 – téléchargement et impression de l’ordonnance

Soucieux de son confort,nous offrons au patient la possibilité de télécharger ses ordonnances

au format PDF qui est une réplique exacte de celle générée par son médecin. Elle contient toutes les informations nécessaires, y compris son identité, les médicaments prescrits, les dosages et les instructions. Cette fonctionnalité assure une traçabilité complète de vos prescriptions tout en respectant les normes de sécurité en vigueur.

6.6 Conclusion

Après avoir achevé le développement du module de gestion des dossiers médicaux, qui facilitera la responsabilité du médecin et du secrétaire ainsi que les fonctionnalités liés au dossier médical au niveau du patient, nous progressons vers le prochain sprint, qui est la gestion du magazine médicale.

Chapitre 7

Sprint 5 : Gestion du Magazine médicale

7.1	Introduction	88
7.2	Sprint Backlog 5	88
7.3	Partie Analyse	91
7.3.1	Diagramme de cas d'utilisation général niveau "Magazine"	91
7.3.2	Diagramme de cas d'utilisation raffiné niveau "Gestion des articles"	92
7.3.3	Diagramme de cas d'utilisation raffiné niveau "Commenter les articles"	92
7.4	Partie Conception	93
7.4.1	Diagramme de classe de conception niveau "Gestion magazine"	93
7.5	Partie Réalisation	93
7.5.1	Partage des publications	93
7.5.2	Publications et Commentaires	94
7.5.3	Gestion des publications	95
7.6	Conclusion	95

7.1 Introduction

Dans ce chapitre, nous mettrons en lumière les travaux effectués au cours du sprint 5. Nous débuterons par le Sprint Backlog ensuite la partie d'analyse et conception et nous montrerons enfin quelques interfaces pour illustrer la mise en œuvre.

7.2 Sprint Backlog 5

Comme présenté par le tableau 7.1, on va détailler le backlog pour notre cinquième sprint.

ID	User story	ID Tâche	Tâche	Estimation	Responsable
1.1	En tant que médecin, je souhaite ajouter un sujet de discussion et partager des posts dans le magazine.	1.1.1	Créer un endpoint dans PostController pour l'ajout d'un nouveau post sous un sujet de discussion.	1	Nader ZOUAIRI
		1.1.2	Implémenter la logique d'ajout d'un post dans le PostService, en lien avec le sujet de discussion existant.	2	Nader ZOUAIRI
		1.1.3	Utiliser Keycloak pour vérifier que l'utilisateur a le rôle "Médecin" pour pouvoir ajouter des posts.	3	Nader ZOUAIRI
		1.1.4	Intégrer les fonctionnalités dans l'interface utilisateur pour permettre aux médecins d'ajouter des posts.	1	Nader ZOUAIRI
		1.1.5	Ajouter la possibilité de récupérer tous les posts par sujet et par utilisateur dans PostController.	2	Nader ZOUAIRI

		1.1.6	Mettre en place une fonctionnalité de mise à jour et de suppression des posts dans Post-Controller.	2	Nader ZOUAIRI
		1.1.7	Créer une interface pour la visualisation et la gestion des posts par les médecins.	1	Nader ZOUAIRI
1.2	En tant que médecin ou pharmacien, je souhaite commenter les différents posts publiés dans le magazine.	1.2.1	Créer un endpoint dans CommentController pour l'ajout d'un commentaire à un post.	1	Nader ZOUAIRI
		1.2.2	Implémenter la logique d'ajout de commentaire dans le CommentService, en liant chaque commentaire à un post spécifique.	1	Nader ZOUAIRI
		1.2.3	Utiliser Keycloak pour vérifier que l'utilisateur a le rôle "Médecin" ou "Pharmacien" pour ajouter un commentaire.	2	Nader ZOUAIRI
		1.2.4	Intégrer les fonctionnalités dans l'interface utilisateur pour permettre aux médecins et pharmaciens de commenter les posts.	1	Nader ZOUAIRI
		1.2.5	Ajouter une fonctionnalité de récupération de tous les commentaires associés à un post dans CommentController.	1	Nader ZOUAIRI

		1.2.6	Mettre en place la possibilité de mise à jour et de suppression des commentaires par leurs auteurs dans Comment-Controller.	1	Nader ZOUAIRI
		1.2.7	Créer une interface pour la visualisation des commentaires sous chaque post et la gestion des commentaires par les utilisateurs.	1	Nader ZOUAIRI
1.3	En tant qu'utilisateur, je souhaite rechercher des posts par nom d'utilisateur, sujet, ou autre critère.	1.3.1	Ajouter des filtres de recherche dans PostController pour permettre la recherche de posts par utilisateur, sujet, et mot-clé.	1	Nader ZOUAIRI
		1.3.2	Implémenter la logique de recherche dans Post-Service pour rechercher des posts par nom d'utilisateur, sujet, et mot-clé.	1	Nader ZOUAIRI
		1.3.3	Ajouter un endpoint de recherche dans Topic-Controller pour récupérer les topics en fonction de critères spécifiques.	1	Nader ZOUAIRI
		1.3.4	Utiliser Keycloak pour sécuriser l'accès aux fonctionnalités de recherche et s'assurer que seuls les utilisateurs authentifiés y ont accès.	1	Nader ZOUAIRI

		1.3.5	Mettre en place l'interface utilisateur pour afficher les résultats de recherche, incluant les posts, commentaires et topics correspondants.	1	Nader ZOUAIRI
		1.3.6	Tester la fonctionnalité de recherche pour vérifier que les résultats sont précis et filtrés correctement par utilisateur, sujet, ou mot-clé.	1	Nader ZOUAIRI

TABLE 7.1 – Sprint 5 Backlog

7.3 Partie Analyse

7.3.1 Diagramme de cas d'utilisation général niveau "Magazine"

La figure 7.1, représente le diagramme de classe général au niveau de la gestion du magazine.

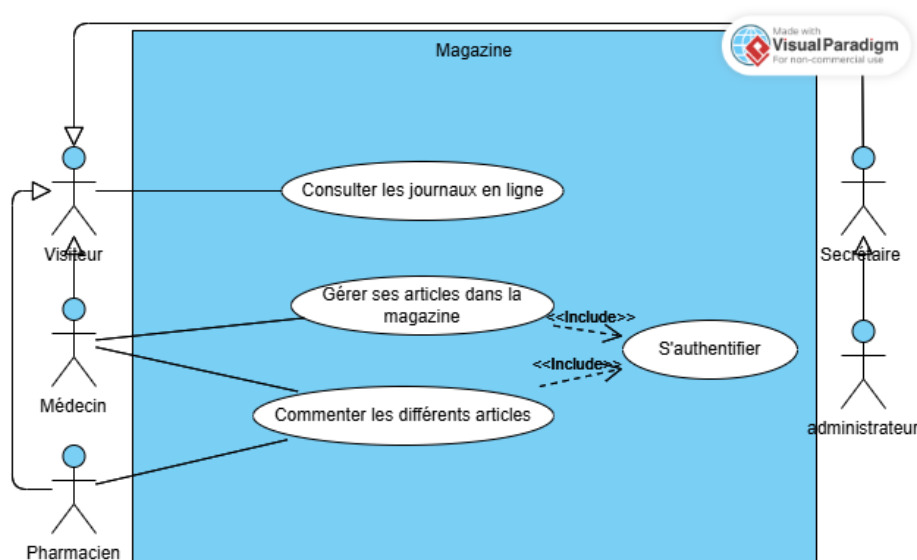


FIGURE 7.1 – Diagramme de cas d'utilisation niveau "Gestion du magazine"

Ce diagramme de cas d'utilisation représente les différentes fonctionnalités du système de gestion de magazine en ligne. Il met en évidence les interactions entre les différents acteurs (visiteur, médecin, pharmacien, secrétaire et administrateur) et les actions qu'ils peuvent effectuer sur le système.

7.3.2 Diagramme de cas d'utilisation raffiné niveau "Gestion des articles"

La figure 7.2, révèle le raffinement du cas d'utilisation du "Gestion des articles". Ce dia-

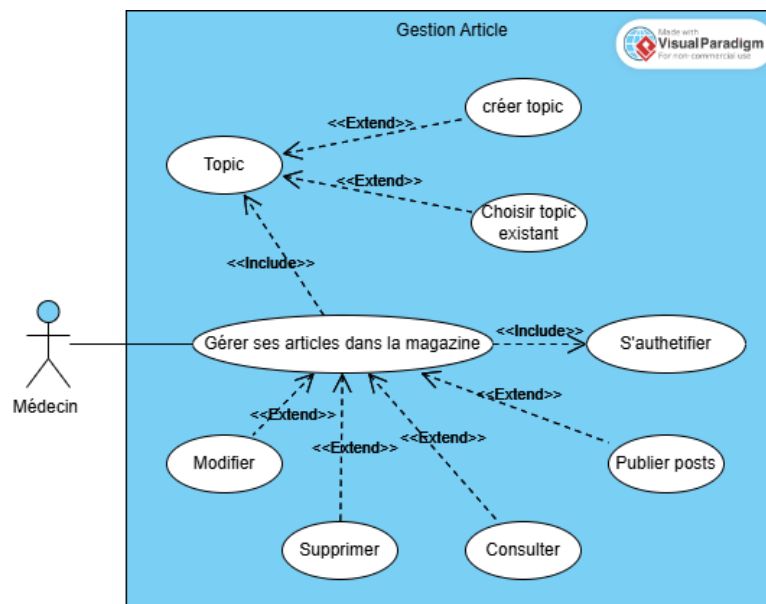


FIGURE 7.2 – Diagramme de cas d'utilisation raffiné niveau "Gestion des articles"

gramme de cas d'utilisation se concentre sur les interactions du médecin avec le système de gestion d'articles, au sein du magazine en ligne spécialisé dans le domaine médical. Il met en évidence les différentes actions que le médecin peut effectuer sur les articles, notamment la création, la modification, la suppression et la consultation.

7.3.3 Diagramme de cas d'utilisation raffiné niveau "Commenter les articles"

La figure 7.3, révèle le raffinement du cas d'utilisation "Commenter les articles".

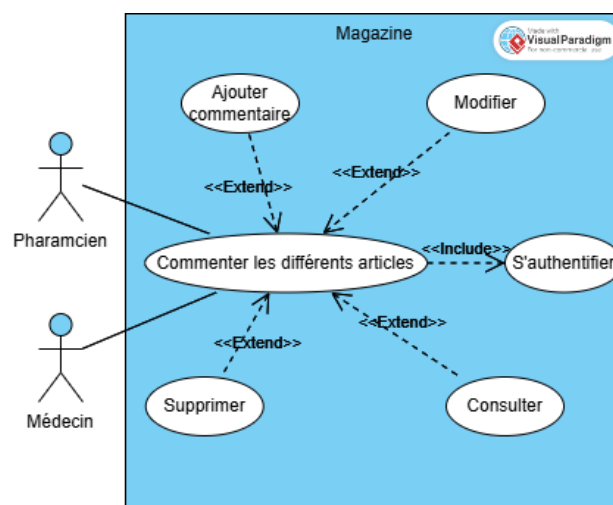


FIGURE 7.3 – Diagramme de cas d'utilisation raffiné niveau "Commenter les articles"

7.4 Partie Conception

7.4.1 Diagramme de classe de conception niveau "Gestion magazine"

La figure 7.4 présente le diagramme de conception au niveau "Gestion magazine".

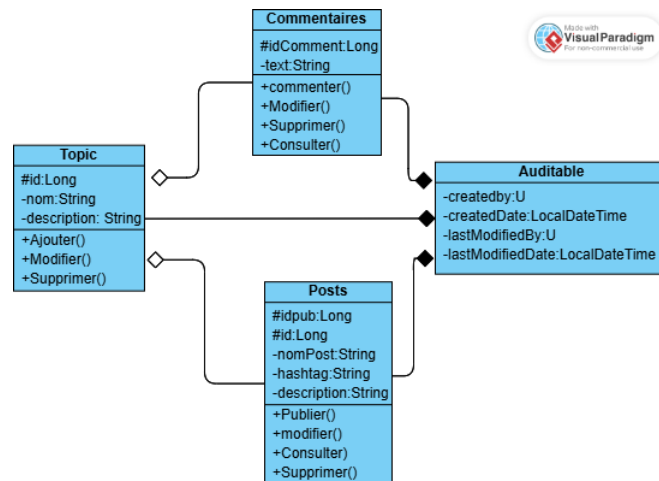


FIGURE 7.4 – Diagramme de classe niveau "Gestion du magazine"

Ce diagramme de classes modélise le système de gestion de forum, lié au magazine en ligne. Il définit trois classes principales : Topic, Posts et Commentaires. Ces classes sont liées entre elles par des relations d'association et d'héritage, et représentent les éléments clés d'un forum de discussion.

7.5 Partie Réalisation

7.5.1 Partage des publications

La figure 7.5 fournit l'interface de partage des publications.

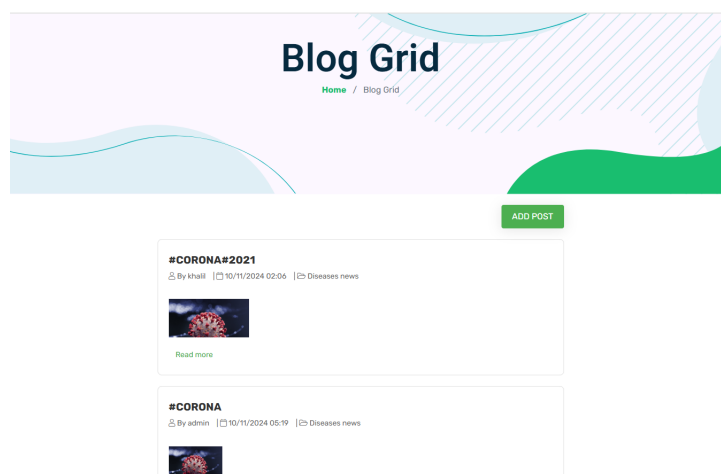


FIGURE 7.5 – Partage des publications

Cette interface offre aux médecins un espace dédié pour créer et animer des discussions autour de sujets médicaux précis. Partagez vos connaissances, posez des questions à vos pairs et trouvez des réponses à vos interrogations. Cet outil de collaboration est conçu pour favoriser le partage de bonnes pratiques, le développement professionnel continu et la mise en réseau des médecins.

7.5.2 Publications et Commentaires

La figure 7.6 fournit l'interface dédiée aux commentaires.

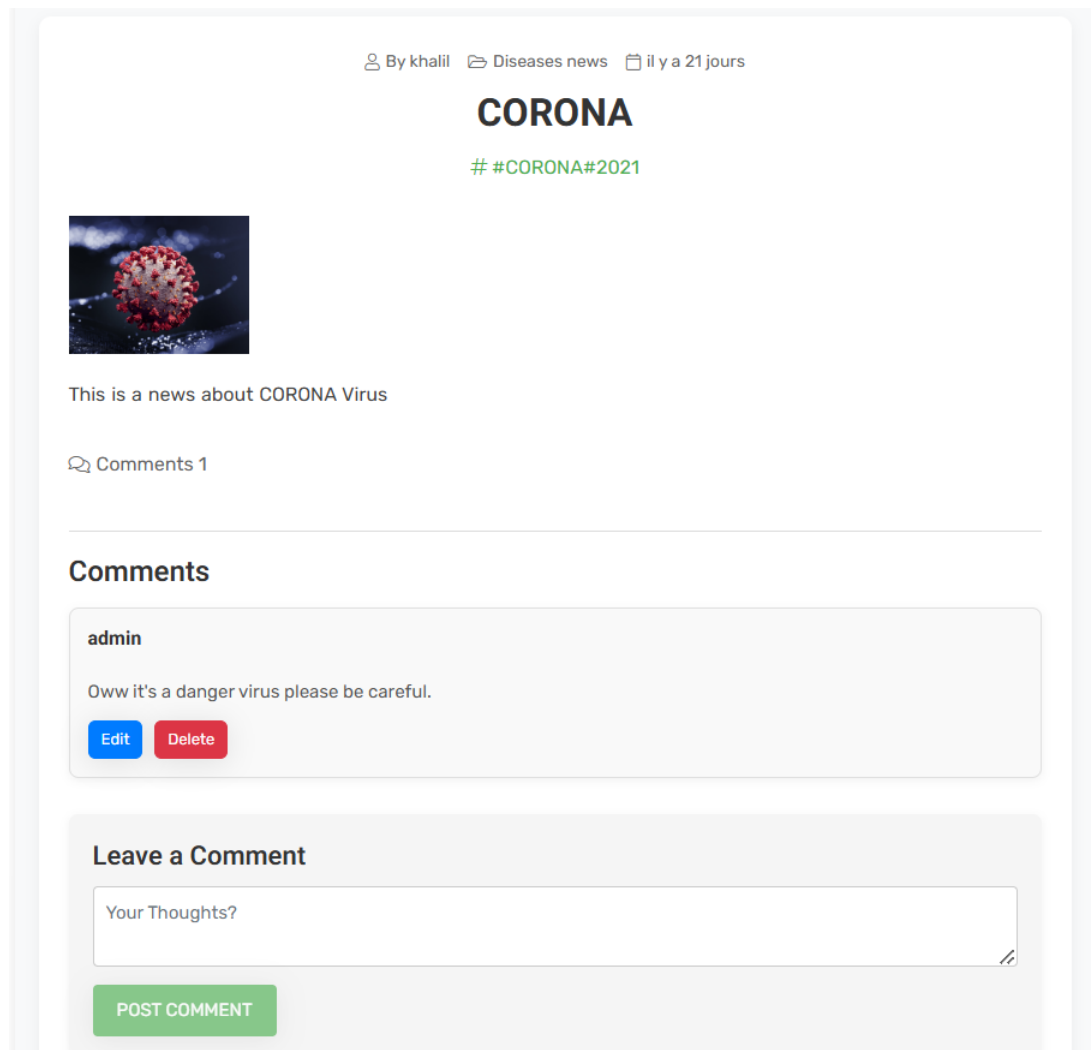


FIGURE 7.6 – Publications et Commentaires

Cette interface est conçue pour favoriser les échanges entre professionnels de santé. Il leur permet de rejoindre la conversation en laissant un commentaire sous chaque publication, partager leur expertise, poser des questions à l'auteur ou à d'autres membres de la communauté, et enrichir le débat.

7.5.3 Gestion des publications

La figure 7.7 présente l'interface de gestion des publications.

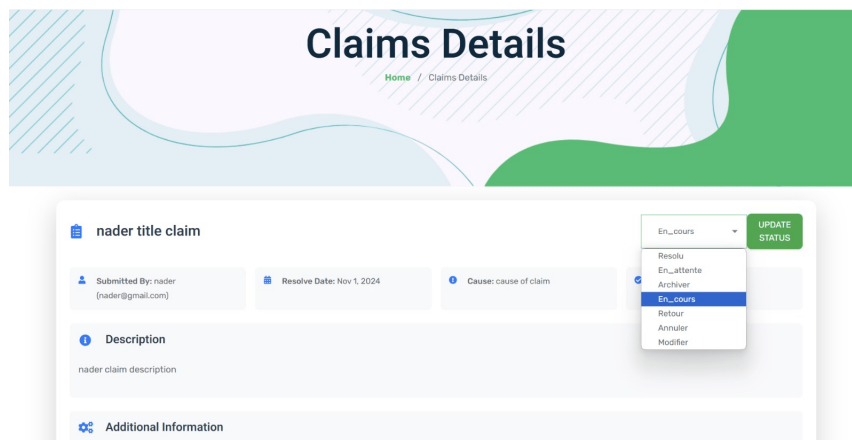


FIGURE 7.7 – Gestion des publications

Cette interface est un outil de modération puissant conçu pour le propriétaire des publications pour maintenir un environnement de discussion sain et sécurisé. Supprimez les commentaires inappropriés et éditez vos publications en quelques clics.

7.6 Conclusion

Cette phase vise à développer un espace de partage de connaissances médicales personnalisé et sécurisé. En offrant aux médecins et pharmaciens la possibilité de publier, de commenter et de consulter des articles sur des sujets spécifiques, nous encourageons la production et la diffusion d'informations médicales à jour. Grâce à des fonctionnalités de recherche et de filtrage, nous permettons aux utilisateurs de trouver rapidement les informations dont ils ont besoin, tout en assurant la confidentialité des données.

Chapitre 8

Sprint 6 : Interaction médicamenteuse

8.1	Introduction	97
8.2	Choix du système d'apprentissage	97
8.2.1	Justification du choix	98
8.3	Mise en place et implémentation du code	98
8.4	Implémentation du code	98
8.4.1	Importation des bibliothèques python nécessaires	98
8.4.2	Chargement et échantillonnage du dataset	99
8.4.3	Définition de la classe DrugDataset	99
8.4.4	Initialisation du tokenizer et du modèle DistilBERT	100
8.4.5	Division des données	100
8.4.6	Définition des métriques d'évaluation	100
8.4.7	Paramètres de formation du modèle	101
8.4.8	Initialisation du Trainer et boucle de formation	101
8.5	Analyse comparative des résultats	102
8.6	Résultat final obtenu	102
8.7	Conclusion	103

8.1 Introduction

Dans ce sprint, l'objectif principal est de développer et d'optimiser un modèle de deep learning "DL" pour prédire les interactions médicamenteuses. Ce modèle, basé sur DistilBERT, une architecture de transformer allégée, a été choisi pour sa capacité à gérer efficacement les données textuelles tout en maintenant des performances élevées.

Nous expliquerons d'abord pourquoi nous avons choisi de migrer du "ML" à "DL", puis nous présenterons et expliquerons les parties les plus importantes du code et conclurons par les derniers résultats obtenus.

8.2 Choix du système d'apprentissage

La figure 8.1 démontre une comparaison visuelle entre le ML et le DL

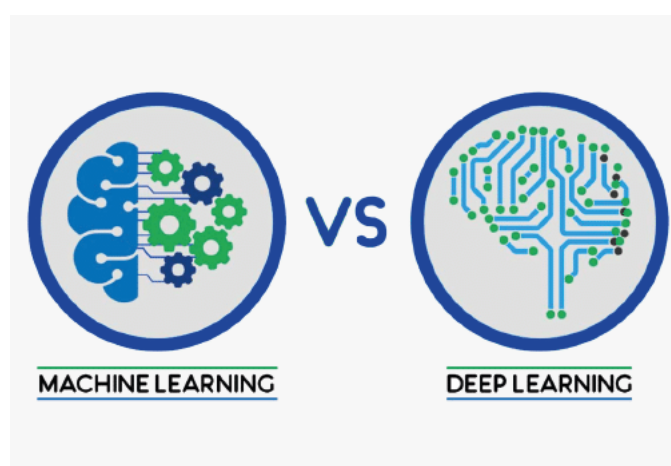


FIGURE 8.1 – Comparaison entre "ML" et "DL"

Pour ce projet de prédiction d'interactions médicamenteuses, une comparaison entre les approches de machine learning et de deep learning s'impose pour justifier notre choix.

Initialement, l'option du machine learning a été envisagée, mais avec plus de 36 000 médicaments, le nombre de combinaisons possibles dépasse le million de paires à traiter, ce qui représente un immense volume de données et de relations complexes. Dans ce contexte, le machine learning, bien que capable de traiter des datasets plus modestes et d'offrir des résultats satisfaisants sur des relations simples, atteint ses limites : les modèles testés ont produit des scores de précision entre 0,10 et 0,20, bien en dessous des attentes pour une application sensible comme la détection d'interactions médicamenteuses.

En revanche, le deep learning, avec des architectures de type transformer comme DistilBERT, s'avère beaucoup plus adapté pour extraire les subtilités des relations entre médicaments dans des datasets volumineux.

Cette approche offre des résultats bien plus prometteurs, avec une précision variant entre 0,46 et 0,71, ce qui montre une meilleure compréhension des interactions possibles grâce à l'apprentissage profond. Ainsi, l'approche deep learning permet non seulement d'améliorer les perfor-

mances du modèle, mais aussi de garantir une évolutivité pour des besoins futurs, rendant cette technologie plus appropriée pour des analyses aussi complexes et volumineuses.

8.2.1 Justification du choix

Notre migration du Machine learning au Deep learning est due à plusieurs raisons :

- **Complexité de la compréhension du langage** Les noms des médicaments et les interactions impliquent des schémas linguistiques nuancés que l'apprentissage profond, en particulier les modèles de transformateurs comme DistilBERT, est conçu pour comprendre mieux que les algorithmes d'apprentissage automatique traditionnels. Les transformateurs gèrent efficacement les relations linguistiques, améliorant ainsi la qualité des prédictions d'interactions.
- **Apprentissage contextuel** Les modèles d'apprentissage profond tels que DistilBERT permettent de saisir le contexte dans les données textuelles, ce qui est crucial pour interpréter avec précision la signification des noms de médicaments et de leurs interactions. Les modèles traditionnels peuvent éprouver des difficultés avec ce niveau de contexte, en particulier lorsqu'ils traitent des modèles et des relations textuels complexes.
- **Précision accrue** Les transformateurs sont généralement plus puissants pour les tâches de traitement du langage naturel et offrent souvent une plus grande précision et de meilleures performances dans l'identification de modèles subtils dans les données. En utilisant DistilBERT, le projet bénéficie de techniques de pointe qui offrent un meilleur pouvoir prédictif.
- **Évolutivité et adaptabilité future** Les méthodes d'apprentissage en profondeur sont plus adaptables à mesure que l'ensemble de données augmente ou si vous devez étendre le projet pour traiter des interactions médicamenteuses plus complexes à l'avenir. Une approche d'apprentissage profond est donc plus évolutive, ce qui permet d'obtenir des solutions plus robustes au fil du temps.

8.3 Mise en place et implémentation du code

Durant cette phase, on va présenter le demarche de notre travaille en Deep Learning on expliquant les parties importantes.

8.4 Implémentation du code

8.4.1 Importation des bibliothèques python nécessaires

```

1  import pandas as pd
2  import torch
3  from torch.utils.data import Dataset, DataLoader
4  from transformers import AutoTokenizer, DistilBertForSequenceClassification, Trainer, TrainingArguments
5  from sklearn.model_selection import train_test_split
6  from sklearn.metrics import accuracy_score, precision_recall_fscore_support
7  import itertools
8  import random

```

FIGURE 8.2 – Importation des bibliothèques

La figure 8.2 montre l'importation des bibliothèques essentielles pour le traitement des données, la création du modèle, et les métriques d'évaluation telles que "Torch", et "pandas". Elle inclut aussi des bibliothèques pour gérer les combinaisons aléatoires de médicaments telle que "itertools".

8.4.2 Chargement et échantillonnage du dataset

```

10 # Load and sample the dataset
11 df = pd.read_excel(io: 'liste_amm.xls', engine='xlrd')
12 drugs = df['Nom'].unique()
13
14 # Set the number of pairs to sample for a smaller dataset
15 num_pairs = 2000 # Increase sample size for more data diversity
16 random.seed(42)
17 sampled_pairs = random.sample(list(itertools.combinations(drugs, r=2)), num_pairs)
18
19 # Ensure balanced labels for training
20 data = []
21 for i, (drug1, drug2) in enumerate(sampled_pairs):
22     label = i % 2 # Alternate labels to balance
23     data.append({"drug1": drug1, "drug2": drug2, "label": label})
24
25 df_interactions = pd.DataFrame(data)
26 df_interactions.to_csv(path_or_buf: "drug_interactions_sampled.csv", index=False)

```

FIGURE 8.3 – Chargement et échantillonnage

La figure 8.3 présente la partie du code qui charge le fichier Excel contenant les noms de médicaments, sélectionne 2000 paires de médicaments aléatoires et crée un jeu de données équilibré en étiquetant alternativement les paires. Le dataset est ensuite enregistré au format CSV pour une utilisation future.

8.4.3 Définition de la classe DrugDataset

```

50 # Initialize DistilBERT tokenizer and model
51 model_name = "distilbert-base-uncased"
52 tokenizer = AutoTokenizer.from_pretrained(model_name)
53 model = DistilBertForSequenceClassification.from_pretrained(model_name, num_labels=2)
54

```

FIGURE 8.4 – Classe DrugDataset

Après le chargement du dataset échantillonné, prêt pour une division en jeux d'entraînement, validation, et test, on a défini la classe "DrugDataset" pour la gestion des paires de médicaments qui hérite de Dataset et gère les paires de médicaments pour le modèle. Elle utilise un tokenizer pour encoder chaque paire sous forme de texte, et retourne les données sous un format compatible avec PyTorch.

8.4.4 Initialisation du tokenizer et du modèle DistilBERT

```
50 # Initialize DistilBERT tokenizer and model
51 model_name = "distilbert-base-uncased"
52 tokenizer = AutoTokenizer.from_pretrained(model_name)
53 model = DistilBertForSequenceClassification.from_pretrained(model_name, num_labels=2)
54
```

FIGURE 8.5 – Initialisation du tokenizer et du modèle DistilBERT

Ici comme le montre la figure 8.5, le tokenizer et le modèle DistilBERT sont initialisés. DistilBERT est préentraîné pour des tâches de classification de texte et ajusté pour prédire l'interaction (2 classes) entre les paires de médicaments.

8.4.5 Division des données

```
55 # Stratified split data into train, validation, and test sets
56 train_df, temp_df = train_test_split(*arrays: df, train_size=0.8, stratify=df['label'], random_state=42)
57 val_df, test_df = train_test_split(*arrays: temp_df, test_size=0.5, stratify=temp_df['label'], random_state=42)
58
59 train_dataset = DrugDataset(train_df, tokenizer)
60 val_dataset = DrugDataset(val_df, tokenizer)
61 test_dataset = DrugDataset(test_df, tokenizer)
```

FIGURE 8.6 – Initialisation du tokenizer et du modèle DistilBERT

La figure 8.6 montre la partie où le dataset est divisé en ensembles d'entraînement 80%, validation 10% et test 10% avec une stratification sur les labels pour un meilleur équilibre. Cela garantit une évaluation fiable et une formation efficace.

8.4.6 Définition des métriques d'évaluation

```
64 def compute_metrics(pred):
65     labels = pred.label_ids
66     preds = pred.predictions.argmax(-1)
67     precision, recall, f1, _ = precision_recall_fscore_support(labels, preds, average='binary')
68     acc = accuracy_score(labels, preds)
69     return {
70         'accuracy': acc,
71         'f1': f1,
72         'precision': precision,
73         'recall': recall
74     }
```

FIGURE 8.7 – Métriques d'évaluation

La figure 8.7 identifie la fonction qui calcule les métriques d'évaluation essentielles pour le modèle : précision, rappel, F1-score et exactitude. Ces mesures aident à évaluer la performance de notre modèle.

8.4.7 Paramètres de formation du modèle

```

76 # Set training arguments with early stopping
77 training_args = TrainingArguments(
78     output_dir='./results',
79     eval_strategy="epoch",
80     save_strategy="epoch",
81     learning_rate=1e-5,
82     per_device_train_batch_size=8,
83     per_device_eval_batch_size=8,
84     num_train_epochs=10,
85     weight_decay=0.01,
86     fp16=True,
87     load_best_model_at_end=True,
88     metric_for_best_model="f1",
89     save_total_limit=2,
90 )

```

FIGURE 8.8 – Paramètres de formation du modèle

La figure 8.8 présente la partie où les arguments de formation spécifient la fréquence d'évaluation, le taux d'apprentissage, le nombre d'époques, et la gestion du poids. Le modèle est sauvegardé lorsque les performances s'améliorent, en suivant le F1-score comme métrique clé.

8.4.8 Initialisation du Trainer et boucle de formation

```

92 # Initialize Trainer
93 trainer = Trainer(
94     model=model,
95     args=training_args,
96     train_dataset=train_dataset,
97     eval_dataset=val_dataset,
98     compute_metrics=compute_metrics
99 )
100
101 # Main training loop
102 if __name__ == '__main__':
103     trainer.train()
104     model.save_pretrained("./fine_tuned_model")
105     tokenizer.save_pretrained("./fine_tuned_model")

```

FIGURE 8.9 – Initialisation du Trainer

La figure 8.9 illustre le Trainer qui est initialisé avec le modèle, les arguments de formation, et les datasets d'entraînement et de validation. Une fois le modèle entraîné, il est sauvegardé pour être réutilisé.

8.5 Analyse comparative des résultats

Durant cette phase on va illustrer quelques images qui présentent la balance entre les résultats obtenus et ensuite le résultat atteint jusqu'à présent.

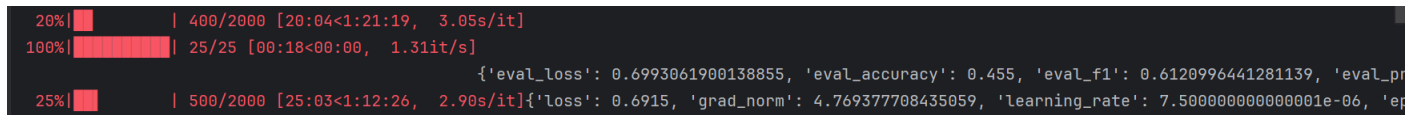


FIGURE 8.10 – 1er exemple du résultat

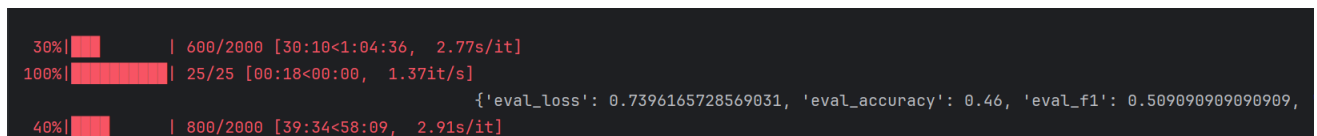


FIGURE 8.11 – 2ème exemple du résultat

Les deux figures 8.10 et 8.11 affichent les performances de notre modèle d'interaction médicamenteuse au fur et à mesure de l'entraînement.

Pour le premier modèle, il montre une précision de 45,5% et un score F1 de 61,2%, ce qui indique que bien que la précision soit modérée, le modèle est déjà capable de capter des informations utiles et offre une capacité de prédiction équilibrée et montre que le modèle commence à apprendre, mais avec des marges d'amélioration importantes, particulièrement en termes de précision.

Pour le deuxième modèle sa précision est de 46% et le score F1 de 50,9%. Bien que la précision n'ait pas énormément augmenté, le score F1 a légèrement baissé, ce qui pourrait indiquer que le modèle commence à atteindre un plateau dans sa capacité d'apprentissage.

8.6 Résultat final obtenu

- **eval_loss : 0.411**
- **eval_accuracy : 71.5%**
- **epoch : 10**

À l'époque 10, la précision atteint 71%, indiquant que notre modèle s'est encore amélioré de manière significative. Après des modifications, il a convergé vers une solution plus optimisée où il a appris les caractéristiques essentielles pour prédire les interactions avec un bon niveau de précision.

C'est pour cela, l'augmentation continue de la précision au fil des époques montre que le modèle de Deep Learning profite d'un entraînement prolongé pour mieux apprendre les interactions médicamenteuses dans un dataset large et complexe.

8.7 Conclusion

L'atteinte d'une précision de 71 % à l'époque 10 justifie le choix d'utiliser une approche Deep Learning, car elle permet au modèle de capturer les complexités et les relations non linéaires entre les médicaments, ce qui serait difficile pour un modèle de Machine Learning classique. Cette précision continue d'augmenter grâce à la capacité du modèle à traiter une grande quantité de données et à s'améliorer avec plus d'itérations.

Conclusion générale et perspectives

En conclusion, notre projet de développement d'une plateforme pharmaceutique personnalisée dédiée aux soins de santé, à la gestion efficace des pharmacies, et à l'optimisation des processus de travail des médecins et secrétaires constitue une avancée significative dans le domaine de la santé numérique. En intégrant les dernières technologies de l'information, cette plateforme offre une solution innovante pour améliorer l'accès aux soins et favoriser une meilleure coordination entre les différents acteurs de la santé.

Les principaux apports de ce projet incluent la simplification de la gestion des médicaments, le renforcement de la communication entre professionnels de santé et patients, ainsi que la réduction des inégalités en matière de santé. En contribuant à améliorer la qualité de vie des populations, nous pensons que cette solution numérique a un impact positif durable sur la société.

Dans une perspective d'avenir, plusieurs extensions sont envisagées pour renforcer davantage l'efficacité et l'accessibilité de la plateforme. Parmi celles-ci : L'ajout d'un **chatbot intelligent** pour guider les utilisateurs dans l'exploration des services disponibles et répondre à leurs questions en temps réel, **l'introduction d'une fonctionnalité d'achat et de paiement en ligne des médicaments**, incluant des options de livraison à domicile pour améliorer le confort des patients et **l'intégration de consultations en ligne**, permettant aux patients d'accéder à des soins médicaux de manière pratique et rapide, tout en optimisant l'expérience des médecins.

Dans l'ensemble, ce projet s'inscrit dans une démarche visant à ouvrir de nouvelles perspectives en télémédecine et à personnaliser davantage les soins. Nous sommes convaincus que ces améliorations contribueront à façonner l'avenir des systèmes de santé en garantissant sécurité, qualité et accessibilité. Notre engagement pour un avenir plus sûr et inclusif se reflète dans chaque aspect de ce projet.

Bibliographie

- [1] <https://www.piterion.com/>. **Piterion:** Consulté le 17 mars 2024.
- [2] <https://business.amwell.com/>. **Site AmWell:** Consulté le 17 mars 2024.
- [3] <http://www.dpm.tn/>. **Site DPM:** Consulté le 17 mars 2024.
- [4] <https://www.bocasay.com/fr/methode-scrum-benefices-developpements-web/>. **Scrum :** Consulté le 18 mars 2024.
- [5] <https://jwt.io/introduction>. **JWT:** Consulté le 20 mars 2024.
- [6] <https://code.visualstudio.com/docs>. **Vs Code:** Consulté le 28 mars 2024.
- [7] <https://www.jetbrains.com/idea/>. **Intellij Idea:** Consulté le 2 mai 2024.
- [8] <https://www.jetbrains.com/help/pycharm/getting-started.html>. **Pycharm:** Consulté le 1 septembre 2024.
- [9] <https://git-scm.com/>. **Git:** Consulté le 20 mars 2024.
- [10] <https://www.postman.com/>. **Postman:** Consulté le 25 mai 2024.
- [11] <https://swagger.io/solutions/api-documentation/>. **Swagger:** Consulté le 20 mai 2024.
- [12] <https://angular.dev/>. **Angular:** Consulté le 18 mars 2024.
- [13] <https://spring.io/projects/spring-boot>. **Spring Boot:** Consulté le 21 mars 2024.
- [14] <https://www.keycloak.org/documentation>. **Keycloak:** Consulté le 24 mars 2024.
- [15] <https://www.postgresql.org/>. **PostgrSql:** Consulté le 20 mars 2024.