

Université Abdelmalek Essaâdi
Faculté des Sciences et Techniques de Tanger
Cycle d'Ingénieur Géo-Information
Module : Développement Mobile

*MedWay : Application Mobile
d'Accès Intelligent aux Services de
Santé*



RÉALISÉS PAR :

LHANNAOUI Leila

NEBKHOUT Hanae

ALLALI Mohamed Amine

EL OUALI Taha

ENCADRÉE PAR :

Pr. KHALFAOUI Hafida

ANNÉE UNIVERSITAIRE :

2024 / 2025

Dédicace

Je dédié ce travail :

A mon cher père,

Pour tous les sacrifices que vous avez faits, pour tout votre amour et préoccupation, sans l'inspiration, l'enthousiasme et le soutien que vous m'avez donnés, je ne serais jamais devenu la personne que je suis aujourd'hui. Vous serez toujours mon étoile la plus brillante.

A ma chère mère,

Si Dieu a mis le paradis sous les pieds des mères, ce n'est pas pour rien. Tu représentes pour moi le symbole de la bonté par excellence, l'exemple du dévouement qui n'a pas cessé de m'encourager et de prier pour moi. Ta prière et ta bénédiction m'ont été d'un grand secours pour mener à bien mes études.

A mes sœurs et mon frère,

Pour ses soutiens moral et leurs conseils précieux tout au long de mes études.

A mes enseignants, mes collègues et amis qui m'ont soutenu et encouragé durant ces années d'études.

LEILA L'HANNAOUI

Remerciements

La réalisation de ce projet a été rendue possible grâce au soutien, à l'encadrement et à la bienveillance de nombreuses personnes que nous souhaitons chaleureusement remerciés.

Nous exprimons notre profonde gratitude à Madame Hafida KHALFAOUI, enseignante à la Faculté des Sciences et Techniques de Tanger, pour son encadrement rigoureux, ses conseils avisés et sa disponibilité constante tout au long de notre travail. Son implication et sa confiance ont été essentielles dans l'avancement et la réussite de ce projet.

Nous tenons également à remercier l'ensemble de nos enseignants pour leur engagement, la qualité de leur enseignement et les compétences qu'ils nous ont transmises tout au long de notre parcours universitaire.

Enfin, nous adressons nos remerciements les plus sincères à toutes les personnes qui, de près ou de loin, ont contribué à la réussite de ce projet. Même si leurs noms ne figurent pas ici, nous leur témoignons toute notre reconnaissance pour leur aide précieuse, leur soutien ou leurs encouragements.

Résumé

Ce rapport présente le processus complet de conception, de développement et d'évolution de l'application MedWay, une solution mobile intelligente dédiée à l'accès aux services de santé. Il s'articule autour de six chapitres complémentaires, chacun couvrant une étape clé du projet, de l'analyse des besoins à la projection vers des perspectives d'avenir.

Le chapitre I est consacré à l'étude préliminaire et à l'analyse des besoins. Il introduit le domaine d'application, identifie les problèmes rencontrés par les usagers du système de santé, et définit les objectifs fonctionnels et non fonctionnels. Une étude comparative d'applications existantes est également présentée afin de positionner MedWay dans son contexte technologique et fonctionnel.

Le chapitre II détaille l'extraction des données via l'API Overpass. Il décrit les outils utilisés et les étapes de la requête permettant de récupérer des données de géolocalisation des établissements de santé, nécessaires à la phase de développement. Cette partie met en évidence l'exploitation d'une source de données ouverte et dynamique pour enrichir les fonctionnalités de recherche.

Le chapitre III aborde la conception de l'application mobile. Il présente les choix technologiques effectués (langages, Framework, architecture logicielle) et propose une modélisation complète à

travers des diagrammes de cas d'utilisation, de classes et de séquences. Cette phase de conception a permis de structurer l'application selon une logique modulaire et évolutive.

Le chapitre IV est dédié à la réalisation et au développement. Il décrit l'architecture du projet, les principales fonctionnalités implémentées, ainsi que les aspects liés à la sécurité, à la gestion des erreurs et à l'optimisation des performances. L'intégration d'un chatbot intelligent, basé sur un modèle de langage exécuté localement, est également décrite comme une avancée majeure dans l'interaction utilisateur.

Le chapitre V présente les maquettes de l'application, conçues pour garantir une expérience utilisateur fluide, intuitive et conforme aux principes d'ergonomie mobile. Ces maquettes ont guidé le développement des interfaces graphiques et ont servi de base à la validation fonctionnelle auprès des utilisateurs cibles.

Enfin, le chapitre VI expose les perspectives d'évolution de MedWay. Il propose des pistes d'amélioration fonctionnelle, technique et stratégique, comme l'intégration de la téléconsultation, l'interopérabilité avec les systèmes de santé publique, ou encore la certification HDS. Ces évolutions visent à garantir la pérennité de l'application et à renforcer son impact dans l'écosystème de l'e-santé.

Table des matières

INTRODUCTION GÉNÉRALE 1

CHAPITRE I

ÉTUDE PRÉLIMINAIRE ET ANALYSE DES BESOINS

I. Introduction	4
II. Présentation générale du domaine	4
III. Analyse du besoin	5
1. Identification du problème	5
2. Public cible de l'application	5
3. Objectifs fonctionnels et non fonctionnels	5
4. Contraintes techniques et organisationnelles	6
IV. Étude comparative (état de l'art)	6
1. Applications similaires existantes	6
2. Avantages et limites des solutions existantes	7
3. Positionnement de notre solution.....	7
V. Conclusion.....	8

CHAPITRE II

EXTRACTION DES DONNÉES VIA L'API OVERPASS

I. Introduction	10
II. Outils et bibliothèques utilisés	10
III. Description de la requête Overpass	10
IV. Étapes du traitement et de l'extraction des données.....	11

V. Exportation Excel et mise en forme.....	12
VI. Conclusion	14

CHAPITRE III

CONCEPTION DE L'APPLICATION MOBILE

I. Introduction	16
II. Choix des outils et technologies	16
1. Langages et environnement de développement	16
2. Librairies et frameworks utilisés.....	18
3. Architecture logicielle choisie	21
III. Conception fonctionnelle et technique.....	22
1. Diagramme de cas d'utilisation.....	22
1.1 Présentation du diagramme	22
1.2 Description des acteurs	24
1.3 Cas d'utilisation principaux	25
2. Diagramme de classe	26
2.1 Présentation du diagramme	26
2.2 Description des acteurs	27
2.3 Relations entre les classes	29
3. Diagramme de séquence.....	29
3.1 Diagramme de séquence - Patient.....	30
3.2 Diagramme de séquence – Professionnel de Santé.....	31
3.3 Diagramme de séquence – Administrateur	32
IV. Conclusion	33

CHAPITRE IV

RÉALISATION ET DÉVELOPPEMENT

I. Introduction	35
II. Architecture du projet.....	35
III. Implémentation des fonctionnalités	37
IV. Intégration et sécurité.....	40
1. Sécurisation des données	40
2. Gestion des erreurs et exceptions.....	41
3. Optimisation des performances.....	41
V. Conclusion.....	42

CHAPITRE V

MAQUETTES DE L'APPLICATION

I. Introduction	44
II. Inscription de l'utilisateur	44
1. Inscription du patient	44
2. Inscription du professionnel de santé.....	45
III. Authentification de l'utilisateur	46
IV. Fonctionnalités de l'administrateur.....	47
V. Fonctionnalités du professionnel de santé	50
1. Gestion des établissements de santé.....	50
1.1 Création du profil d'un établissement de santé.....	51
1.2 Mise à jour des disponibilités	52
1.3 Consultation des files d'attente	54
2. Chatbot intelligent IA	56

3. Gestion du profil	58
VI. Fonctionnalités du patient.....	61
1. Recherche d'établissements	61
2. Création, modification et suppression du dossier médical	63
3. Prise de rendez-vous et consultation de la liste des rendez-vous	66
4. Chatbot intelligent IA.....	68
5. Gestion du profil	70
V. Conclusion.....	72

CHAPITRE VI

PERSPECTIVES D'ÉVOLUTION

I. Introduction	74
II. Nouvelles perspectives fonctionnelles.....	74
III. Améliorations techniques envisagées.....	75
IV. Perspectives stratégiques.....	76
V. Conclusion.....	77
CONCLUSION.....	79

Liste des figures

Figure 1 : Exportation des Données	13
Figure 2 : Java.....	17
Figure 3 : XML	17
Figure 4 : Android Studio	18
Figure 5 : Firebase Authentication	19
Figure 6 : Cloud Firestore.....	19
Figure 7 : OpenStreetMap	20
Figure 8 : Glide	20
Figure 9 : IA / Chatbot	21
Figure 10 : Diagramme de Cas d'Utilisation.....	23
Figure 11 : Diagramme de Classe.....	27
Figure 12 : Diagramme de Séquence - Patient.....	30
Figure 13 : Diagramme de Séquence – Professionnel de Santé.....	31
Figure 14 : Diagramme de Séquence – Administrateur.....	32
Figure 15 : Architecture du Projet.....	36
Figure 16 : Inscription du Patient	45
Figure 17 : Inscription du Professionnel de Santé.....	46
Figure 18 : Authentification de l'Utilisateur	47
Figure 19 : Tableau de Bord du Administrateur	49
Figure 20 : Fonctionnalités de l'Administrateur	50
Figure 21 : Création du Profil d'un établissement de Santé	51
Figure 22 : Liste des établissements de Santé	52
Figure 23 : Mise à Jour des Disponibilités	53
Figure 24 : Liste des Disponibilités	54
Figure 25 : Consultation des Files d'Attente.....	55

Figure 26 : File d'Attente	56
Figure 27 : Chatbot Intelligent IA.....	57
Figure 28 : Gestion du Profil	60
Figure 29 : Avis	61
Figure 30 : Recherche d'établissements	62
Figure 31 : Option de Filtrage	63
Figure 32 : Création, Modification et Suppression du Dossier Médical	
Figure 33 : Fiche Dossier Médical.....	66
Figure 34 : Prise de Rendez-vous	67
Figure 35 : Liste des Rendez-vous.....	68
Figure 36 : Chatbot IA	69
Figure 37 : Gestion du Profil	71
Figure 38 : Avis	72

Liste des tableaux

Table 1 : Type d'Établissement de Santé.....13

INTRODUCTION GÉNÉRALE

Avec l'évolution rapide des technologies mobiles et de l'intelligence artificielle, le secteur de la santé se digitalise de plus en plus afin d'améliorer l'accessibilité, l'efficacité et la qualité des services offerts aux patients et aux professionnels. Dans cette dynamique, les applications mobiles de santé jouent un rôle clé dans la modernisation du système de soins, en permettant une gestion fluide des informations médicales, une meilleure coordination entre les acteurs de santé, et une interaction simplifiée avec les patients.

L'application **MedWay** intègre un ensemble complet de fonctionnalités conçues pour améliorer à la fois l'expérience des patients et l'efficacité des professionnels de santé. Elle permet notamment la gestion intégrale du dossier médical du patient, incluant les informations personnelles, les antécédents, les traitements en cours et les résultats médicaux. En parallèle, **MedWay** assure une authentification sécurisée des professionnels de santé via Firebase Authentication, garantissant un accès confidentiel et individualisé aux données médicales. Grâce à un système de recherche d'établissements de santé en temps réel, les utilisateurs peuvent trouver rapidement les structures médicales adaptées, en fonction de plusieurs critères grâce à un filtrage avancé (type, localisation, spécialité, disponibilité, etc.).

MedWay propose également la prise de rendez-vous en ligne et la gestion des disponibilités, permettant aux patients de planifier facilement leurs consultations et aux établissements de mieux organiser leurs créneaux. Un espace professionnel personnalisé est dédié aux professionnels de santé, leur offrant la possibilité de gérer leurs établissements, les créneaux de disponibilité et les files d'attente associées. Pour renforcer l'autonomie des utilisateurs, l'application intègre un chatbot intelligent, capable de les guider dans leurs démarches, de répondre aux questions courantes et de faciliter la navigation au sein de l'application. Enfin, **MedWay** met à disposition un système d'avis et de notation, permettant aux patients de partager leurs retours sur les établissements et les services médicaux, contribuant ainsi à plus de transparence et à l'amélioration continue de la qualité des soins.

Grâce à une architecture basée sur le modèle MVVM, des outils modernes comme Java, XML, Firebase (Authentication et Firestore), ainsi qu'une interface ergonomique, **MedWay** se positionne comme une solution mobile complète et évolutive dédiée à la santé connectée.

Ce rapport est structuré en cinq chapitres principaux, reflétant les grandes étapes du projet :

- ❖ **CHAPITRE I : ÉTUDE PRÉLIMINAIRE ET ANALYSE DES BESOINS**, qui présente le contexte, les objectifs du projet ainsi que l'identification des besoins fonctionnels et techniques.
- ❖ **CHAPITRE II : EXTRACTION DES DONNÉES VIA L'API OVERPASS**, qui explique le processus de collecte des établissements de santé à Tanger en utilisant l'API Overpass, en détaillant la requête utilisée, les outils Python exploités et l'exportation des résultats.
- ❖ **CHAPITRE III : CONCEPTION DE L'APPLICATION MOBILE**, qui décrit l'architecture technique, les maquettes de l'interface, les schémas de base de données et les choix de conception.
- ❖ **CHAPITRE IV : RÉALISATION ET DÉVELOPPEMENT**, qui détaille la mise en œuvre de l'application, les fonctionnalités développées, les technologies utilisées et les tests effectués.
- ❖ **CHAPITRE V : MAQUETTES DE L'APPLICATION**, qui présente les maquettes de l'interface utilisateur, illustrant la navigation et l'ergonomie prévues pour l'application.
- ❖ **CHAPITRE VI : PERSPECTIVES D'ÉVOLUTION**, qui expose les évolutions possibles de l'application, telles que l'optimisation des performances, l'ajout de nouvelles fonctionnalités (messagerie, intelligence artificielle, recommandations), l'amélioration de l'ergonomie et le renforcement de la sécurité des données.

CHAPITRE I

ÉTUDE PRÉLIMINAIRE ET

ANALYSE DES BESOINS

I. Introduction

Avant de concevoir une application mobile, il est indispensable d'étudier le contexte dans lequel elle s'inscrit et d'analyser les besoins qu'elle vise à satisfaire. Cette étape permet de mieux comprendre les attentes des utilisateurs, d'identifier les fonctionnalités clés, et de poser les bases solides du projet.

Dans le cadre de **MedWay**, une application dédiée à la gestion des services de santé, cette analyse préliminaire vise à définir les enjeux du secteur, à cerner les besoins des patients et des professionnels de santé, et à évaluer les solutions existantes. Ce chapitre présente donc le domaine d'application, les besoins fonctionnels et techniques identifiés, ainsi qu'une étude comparative des applications similaires pour mieux positionner notre solution.

II. Présentation générale du domaine

Le domaine de la santé connaît depuis plusieurs années une transformation profonde grâce à l'intégration des technologies numériques. Face à une demande croissante en soins, à une gestion de plus en plus complexe des données médicales, et à la nécessité d'optimiser l'organisation des établissements de santé, les solutions mobiles s'imposent comme des outils indispensables. Elles permettent non seulement de faciliter la communication entre les professionnels de santé et les patients, mais aussi d'améliorer la gestion des dossiers médicaux, la planification des consultations, et le suivi des traitements.

Dans ce contexte, les applications mobiles de santé offrent une réponse efficace aux limites des systèmes traditionnels, souvent basés sur des processus manuels, fragmentés et chronophages. Ces outils numériques permettent de centraliser les informations, d'automatiser certaines tâches administratives, et de rendre les services de santé plus accessibles, notamment via des fonctionnalités comme les notifications, la traduction, ou les services d'assistance automatisée.

C'est dans cette dynamique que s'inscrit le projet **MedWay**, qui ambitionne de fournir une solution mobile complète et intuitive, destinée à la fois aux patients et aux professionnels. En s'appuyant sur les technologies mobiles et les services cloud,

MedWay vise à moderniser la gestion des établissements de santé, tout en améliorant la qualité et la rapidité des soins.

III. Analyse du besoin

L'analyse du besoin consiste à identifier le problème à résoudre, définir le public cible de l'application, établir les objectifs fonctionnels et non fonctionnels, et prendre en compte les contraintes techniques et organisationnelles du projet.

1. Identification du problème

Le projet **MedWay** répond à un besoin réel observé dans le secteur de la santé : malgré les nombreuses avancées technologiques, la gestion des établissements médicaux reste souvent complexe et peu optimisée. Les professionnels doivent gérer manuellement de nombreuses tâches administratives, ce qui engendre une perte de temps et un risque accru d'erreurs. Les patients, quant à eux, rencontrent fréquemment des difficultés d'accès à l'information, des délais d'attente importants, et une communication limitée avec les professionnels de santé. Ces problèmes soulignent l'urgence de proposer une solution numérique centralisée, simple d'usage et efficace.

2. Public cible de l'application

L'application cible deux catégories principales d'utilisateurs. D'une part, les **professionnels de santé** (médecins, infirmiers, personnel administratif), qui auront accès à des outils de gestion de leur profil, de leurs disponibilités, de leurs établissements et des files d'attente. D'autre part, les **patients**, qui pourront créer, modifier et consulter leur dossier médical, rechercher des établissements de santé, prendre des rendez-vous, accéder à un assistant virtuel, déposer des avis sur l'application.

3. Objectifs fonctionnels et non fonctionnels

Les besoins fonctionnels de **MedWay** sont multiples et ciblent une automatisation efficace des processus médicaux et administratifs. L'application permet la création, modification, consultation et suppression des dossiers médicaux, avec la gestion détaillée des antécédents, traitements et résultats. Elle propose un système d'inscription et d'authentification sécurisé pour les professionnels, ainsi qu'un espace personnel pour gérer leurs établissements, créneaux de disponibilité et files d'attente.

De plus, un chatbot intelligent aide les utilisateurs à naviguer dans l'application ou à poser des questions médicales générales. Un système de notation et d'avis permet aux patients d'évaluer les services reçus.

Sur le plan des besoins non fonctionnels, **MedWay** vise à offrir une interface ergonomique, une navigation fluide, une sécurité renforcée des données (grâce à Firebase), ainsi qu'une structure modulaire facilitant la maintenance. À plus long terme, un support multiplateforme et une intégration avec d'autres systèmes médicaux ou API pourront être envisagés pour étendre les possibilités de l'application.

4. Contraintes techniques et organisationnelles

Plusieurs contraintes techniques et organisationnelles doivent être prises en compte pour garantir la réussite du projet. Le développement suit une architecture MVVM afin d'assurer la clarté, la modularité et la maintenabilité du code. Le stockage des données est réalisé via Firebase et Cloud Firestore, avec des règles de sécurité strictement adaptées pour protéger les informations sensibles et garantir l'accès uniquement aux utilisateurs autorisés.

Enfin, le projet respecte rigoureusement les standards de sécurité, de performance et d'accessibilité du développement Android, garantissant ainsi une application fiable, rapide et inclusive pour tous les utilisateurs.

IV. Étude comparative (état de l'art)

L'étude comparative analyse les applications similaires existantes, en mettant en évidence leurs avantages et leurs limites. Elle permet de positionner notre solution en fonction des besoins identifiés et d'offrir une alternative plus efficace et adaptée.

1. Applications similaires existantes

Plusieurs applications mobiles dans le domaine de la santé connectée sont déjà présentes sur le marché. Parmi les plus populaires, on trouve des applications comme **MyChart**, **Doctolib**, et **Health Mate**. Ces applications permettent aux utilisateurs de gérer leurs rendez-vous médicaux, accéder à leur dossier médical, et suivre leurs traitements. Cependant, peu d'entre elles offrent une solution complète qui

intègre à la fois les professionnels de santé et les patients dans une même interface mobile.

2. Avantages et limites des solutions existantes

❖ Avantages

Les applications existantes offrent généralement une interface conviviale et une gestion facile des rendez-vous et des informations médicales. Elles permettent également une meilleure communication entre le patient et le professionnel de santé.

❖ Limites

Ces solutions ont souvent une portée limitée, ne couvrant qu'une partie du parcours de soins. Par exemple, certaines n'intègrent pas la recherche en temps réel des établissements avec des filtres de recherche avancés, ou bien la gestion des dossiers médicaux peut être fragmentée. De plus, la sécurité et la confidentialité des données sont des préoccupations majeures, souvent non entièrement traitées par ces applications.

3. Positionnement de notre solution

MedWay se distingue des autres solutions en offrant une plateforme mobile complète, sécurisée et intuitive, qui couvre l'ensemble du parcours de soin pour les patients comme pour les professionnels de santé.

Elle propose les fonctionnalités suivantes :

- ❖ **Recherche d'établissements de santé en temps réel**, avec un système de **filtrage avancé** (type, localisation, spécialité, disponibilité, etc.).
- ❖ **Prise de rendez-vous** en ligne et **gestion des disponibilités**.
- ❖ **Création, consultation, modification et suppression du dossier médical**, incluant les antécédents, traitements et résultats médicaux.
- ❖ **Authentification sécurisée** des professionnels de santé via Firebase Authentication.
- ❖ **Espace professionnel personnalisé**, permettant la gestion des établissements, des créneaux de disponibilité et des files d'attente.

- ❖ **Chatbot intelligent**, pour guider les utilisateurs dans leurs démarches ou répondre à des questions médicales générales.
- ❖ **Système d'avis et de notation**, permettant aux patients de partager leurs retours sur les établissements et services médicaux.
- ❖ **Génération de rapports statistiques par l'administrateur**, permettant l'analyse de l'activité de la plateforme (utilisation, rendez-vous, évaluations, etc.).

Grâce à cette offre riche et intégrée, **MedWay** se positionne comme une solution de santé connectée innovante, fiable et accessible, répondant aux enjeux actuels de digitalisation des parcours de soins. Son architecture basée sur MVVM garantit un développement modulaire, clair et évolutif, respectant les standards Android en matière de sécurité, performance, accessibilité et ergonomie.

V. Conclusion

L'étude préliminaire a permis d'identifier les lacunes des solutions existantes en matière de santé connectée, tout en précisant les attentes des utilisateurs cibles, qu'il s'agisse des patients ou des professionnels de santé. Grâce à cette analyse approfondie, nous avons défini les besoins fonctionnels et techniques de l'application **MedWay**, en intégrant des fonctionnalités innovantes telles que la gestion du dossier médical, la prise de rendez-vous, la géolocalisation en temps réel des établissements, un chatbot intelligent, et une interface multilingue, etc.

Ces éléments constituent la base sur laquelle repose la conception de l'application, en assurant une réponse pertinente, sécurisée et évolutive aux enjeux actuels du domaine médical numérique.

CHAPITRE II

EXTRACTION DES DONNÉES

VIA L'API OVERPASS

I. Introduction

Afin de collecter des données géospatiales précises et actualisées sur les établissements de santé à Tanger, nous avons exploité l'API Overpass, un outil performant permettant d'interroger la base de données OpenStreetMap (OSM). Ces données sont indispensables pour l'analyse spatiale, la visualisation cartographique et le développement d'applications dédiées à la santé publique.

II. Outils et bibliothèques utilisés

Pour l'extraction et le traitement des données issues d'OpenStreetMap via l'API Overpass, plusieurs bibliothèques Python ont été mobilisées :

- ☞ **overpy** : bibliothèque permettant d'interagir facilement avec l'API Overpass en envoyant des requêtes OSM et en récupérant les données sous forme d'objets Python.
- ☞ **pandas** : utilisée pour la manipulation, la structuration et le nettoyage des données extraites.
- ☞ **openpyxl** : permet l'exportation des données vers un fichier Excel (.xlsx) avec une mise en forme personnalisée.
- ☞ **tqdm** : facilite le suivi du traitement grâce à l'affichage de barres de progression.
- ☞ **time** : utilisée pour mesurer la durée d'exécution des différentes étapes du script.

III. Description de la requête Overpass

Le script a pour objectif d'extraire les établissements de santé situés dans la ville de Tanger, en ciblant une zone géographique délimitée par une boîte englobante (Latitude : 35.70 à 35.85, Longitude : -5.90 à -5.70). La requête Overpass est construite pour interroger les entités pertinentes dans OpenStreetMap, en se basant sur des balises sémantiques associées aux services de santé.

Les entités sélectionnées incluent les **nœuds** et les **chemins** portant les balises suivantes :

- ☞ "amenity" : avec des valeurs telles que hospital, clinic, doctors, pharmacy, dentist, veterinary, nursing_home, social_facility.
- ☞ "healthcare" : quelle que soit la valeur associée (filtrage plus large).

Voici la requête Overpass utilisée :

```
[out:json][timeout:300];
(
node["amenity"~"hospital|clinic|doctors|pharmacy|dentist|veterinary|
nursing_home|social_facility"](35.70,-5.90,35.85,-5.70);

way["amenity"~"hospital|clinic|doctors|pharmacy|dentist|veterinary|n
ursing_home|social_facility"](35.70,-5.90,35.85,-5.70);

node["healthcare"]的文化(35.70,-5.90,35.85,-5.70);

way["healthcare"]文化(35.70,-5.90,35.85,-5.70);

);
out center;
```

Cette requête permet d'obtenir une liste complète et à jour des infrastructures de santé disponibles dans la zone ciblée, en tenant compte aussi bien des points individuels que des bâtiments (polygones).

IV. Étapes du traitement et de l'extraction des données

Le processus d'extraction et de traitement des données à partir d'OpenStreetMap via l'API Overpass s'est déroulé en plusieurs étapes clés :

☞ Connexion à l'API Overpass

Utilisation de la bibliothèque overpy pour établir une connexion avec le serveur Overpass et permettre l'exécution de requêtes personnalisées.

➲ Exécution de la Requête Overpass

Envoi de la requête définie précédemment afin de récupérer les établissements de santé présents dans la zone géographique ciblée (ville de Tanger).

➲ Extraction des Informations Pertinentes

Pour chaque nœud ou chemin retourné, les balises utiles sont extraites : nom de l'établissement, type d'amenity ou de healthcare, coordonnées géographiques (latitude, longitude), adresse, numéro de téléphone, etc.

➲ Organisation des Données

Création d'un tableau structuré en utilisant pandas.DataFrame, facilitant la manipulation et la visualisation des données collectées.

➲ Exportation vers Excel

Sauvegarde des données dans un fichier Excel (.xlsx) avec une mise en forme personnalisée grâce à la bibliothèque openpyxl, incluant les en-têtes, le formatage des colonnes, et une présentation claire pour exploitation future.

V. Exportation Excel et mise en forme

À l'issue du traitement, les données extraites sont exportées dans un fichier Excel intitulé **All_Health_Facilities_Tanger.xlsx**, généré à l'aide de la bibliothèque **openpyxl**. Cette étape vise à faciliter l'analyse et l'exploitation des résultats sous une forme lisible et soignée. Le fichier Excel présente les caractéristiques suivantes :

➲ En-têtes Stylisés : les titres des colonnes sont affichés en gras pour une meilleure lisibilité.

➲ Ajustement Automatique : la largeur des colonnes est adaptée automatiquement au contenu, garantissant un affichage optimal des informations.

❖ Format Structuré : les données sont organisées de manière claire et hiérarchisée, prêtes à être utilisées pour des analyses ultérieures ou des visualisations.

Name	Type	Latitude	Longitude	Address	Phone	Emergency	Wheelchair	Opening Hours	Beds	Specialty	Source
Clinique Tingis	Clinic	35.770451	-5.8073402			no	unknown				OpenStreetMap
Pharmacie Anegay	Pharmacy	35.7851172	-5.8124517			no	unknown				OpenStreetMap
Pharmacie Bâb Bhar	Pharmacy	35.7878722	-5.8117732			no	unknown				OpenStreetMap
Pharmacie Al Maghrib	Pharmacy	35.7826008	-5.8129687			no	unknown				OpenStreetMap
Pharmacie La Cité Universitaire	Pharmacy	35.7338104	-5.8346532			no	unknown				OpenStreetMap
Pharmacie Moderne	Pharmacy	35.7840650	-5.8126868			no	unknown				OpenStreetMap
Pharmacie Cervantes	Pharmacy	35.7826706	-5.8104385			no	unknown				OpenStreetMap
Unnamed Facility	dentist	35.7783815	-5.8086548			no	unknown				OpenStreetMap
Pharmacie Masjid Al Azhar	Pharmacy	35.7606341	-5.8354840			no	unknown				OpenStreetMap
Centre de Santé Bni Ouraghel	Doctor Office	35.7459556	-5.8249800			no	unknown				OpenStreetMap
Pharmacie Al Aouama Al Gharbia	Pharmacy	35.7261824	-5.8038898			no	unknown				OpenStreetMap
Pharmacie Al Hanae	Pharmacy	35.7308043	-5.8029761			no	unknown				OpenStreetMap
Pharmacie Al Machkouri	Pharmacy	35.7328935	-5.8152164			no	unknown				OpenStreetMap
Dr. El Alaoui	Doctor Office	35.7805578	-5.8126490	+212 539933030		no	unknown	Mo-Fr 09:00-16:00; Sa 09:00-14:00			OpenStreetMap
Pharmacie Marbella	Pharmacy	35.7254593	-5.7564234			no	unknown				OpenStreetMap
Pharmacie De La Plage	Pharmacy	35.7788382	-5.8025471			no	unknown				OpenStreetMap
Pharmacie Aswak Assalam	Pharmacy	35.7379507	-5.8738926			no	unknown				OpenStreetMap
Arrazi Radiologie	Clinic	35.7762857	-5.8149609			no	unknown				OpenStreetMap
Pharmacie du Soleil	Pharmacy	35.7748656	-5.8101820			no	unknown				OpenStreetMap
Centre De Biologic Medecine	Clinic	35.7734271	-5.8098306			no	unknown				OpenStreetMap

Figure 1 : Exportation des Données

#	Nom du Type d'Établissement de Santé
1	Clinique
2	Laboratoire
3	Pharmacie
4	Hôpital
5	Cabinet Dentaire
6	Cabinet Médical (Médecin général)

Table 1 : Type d'Établissement de Santé

VI. Conclusion

L'utilisation de l'API Overpass, couplée à des bibliothèques Python puissantes, a permis d'extraire efficacement des données géospatiales fiables et à jour sur les établissements de santé à Tanger. Grâce à cette approche, nous avons pu collecter et structurer des informations précieuses (noms, types, coordonnées, etc.) directement issues d'OpenStreetMap. Ces données constituent une base solide pour alimenter notre application mobile, offrir des services localisés aux utilisateurs, et renforcer la précision des fonctionnalités de recherche et de visualisation des établissements de santé dans la ville.

CHAPITRE III

CONCEPTION DE

L'APPLICATION MOBILE

I. Introduction

La phase de conception constitue une étape fondamentale du cycle de développement de l'application **MedWay**. Elle permet de structurer les composants techniques et fonctionnels de manière claire et cohérente, en vue de garantir la qualité, la maintenabilité et l'évolutivité du système.

Ce chapitre présente la conception globale de l'application **MedWay**, en s'appuyant sur les choix technologiques, l'organisation fonctionnelle et la modélisation technique. Il décrit les outils utilisés, les interactions entre les utilisateurs et le système, ainsi que les structures de données et les échanges entre composants à travers des diagrammes UML de cas d'utilisation, de classes et de séquence.

II. Choix des outils et technologies

Le choix des outils et technologies repose sur la sélection des langages, plateformes et environnements de développement les plus adaptés, ainsi que des librairies et frameworks permettant d'optimiser le développement. L'architecture logicielle retenue assure la structuration efficace de l'application en fonction de ses exigences.

1. Langages et environnement de développement

Le choix des outils de développement pour l'application **MedWay** repose sur la nécessité de garantir une performance optimale, une compatibilité étendue et une flexibilité dans les évolutions futures. Pour assurer une expérience fluide et une interface moderne, nous avons sélectionné des technologies adaptées à l'écosystème Android tout en gardant en tête les exigences de sécurité, de scalabilité et de maintenabilité.

Ce choix stratégique a également permis de garantir une bonne interaction entre les différentes couches de l'application, de la gestion du logique métier à l'affichage des interfaces utilisateur.

⌘ Java

Le langage principal utilisé pour implémenter la logique métier de l'application. Il gère la communication avec les services back-end, comme **Firebase**, et les interactions de l'utilisateur au sein de l'application. Java reste le pilier du projet en raison de sa stabilité et de sa robustesse dans les environnements Android.



Figure 2 : Java

⌘ XML

Utilisé pour la conception des interfaces utilisateur. **XML** permet de structurer les vues, les widgets et de définir l'apparence des écrans, tout en respectant les standards d'Android. Ce langage permet aussi de séparer la logique de l'interface, facilitant ainsi la maintenance et la modification de l'application.



Figure 3 : XML

Android

La plateforme de développement principale est **Android**, afin de s'assurer de la compatibilité avec une large gamme de smartphones utilisés dans les environnements professionnels de santé. Android Studio, l'environnement de développement intégré (IDE) officiel, sera utilisé pour le codage, le test et le déploiement de l'application.

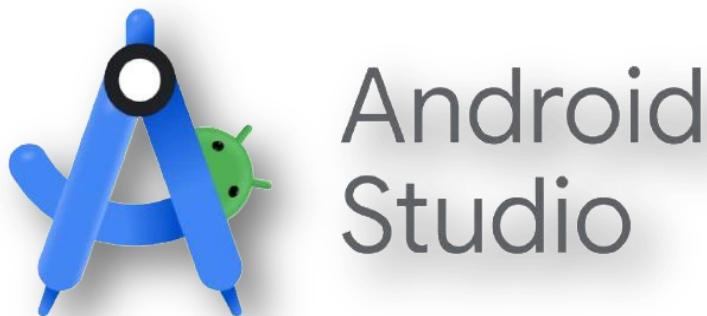


Figure 4 : Android Studio

2. Librairies et frameworks utilisés

Pour garantir une expérience fluide et performante tout en respectant les meilleures pratiques, **MedWay** utilise un ensemble de librairies et de frameworks modernes qui facilitent le développement de fonctionnalités avancées tout en assurant la stabilité et la sécurité de l'application.

Firebase Authentication

Cette librairie est utilisée pour gérer l'inscription et l'authentification des utilisateurs de manière sécurisée. Elle permet de gérer différents types de connexions, tels que par email/mot de passe ou via des services tiers (Google, Facebook, etc.), tout en assurant une sécurité optimale des données sensibles.



Figure 5 : Firebase Authentication

Cloud Firestore

Ce service est utilisé pour le stockage des données en temps réel. Il permet de structurer les informations sous forme de collections et de documents, facilitant ainsi la gestion des données d'utilisateurs, d'établissements, de rendez-vous, et de dossiers médicaux. Firestore garantit une synchronisation rapide et une accessibilité élevée aux données.



Figure 6 : Cloud Firestore

OpenStreetMap

Ces bibliothèques permettent d'intégrer des cartes interactives open source au sein de l'application. Elles sont utilisées pour afficher la localisation des établissements, offrant ainsi aux utilisateurs une navigation géographique fluide et précise.



Figure 7 : OpenStreetMap

Picasso/Glide

Ces deux bibliothèques sont utilisées pour le chargement, le traitement et l'affichage efficace des images. Elles permettent de gérer les images à distance (notamment les avatars des utilisateurs et les photos des établissements) tout en optimisant la mémoire et la performance de l'application.



Figure 8 : Glide

Material Components

Ce framework est intégré pour respecter les bonnes pratiques en matière d'UI/UX recommandées par Google. Il permet de créer des interfaces cohérentes et modernes, adaptées aux directives de design d'Android, tout en garantissant une expérience utilisateur fluide et agréable.

IA / Chatbot

Un chatbot intégré à l'application permet d'offrir une assistance conversationnelle aux utilisateurs. Il facilite la recherche d'informations, la gestion des démarches et la navigation dans l'application en fournissant des réponses automatiques et pertinentes.



Figure 9 : IA / Chatbot

3. Architecture logicielle choisie

L'application **MedWay** adopte une architecture **MVVM** (Model – View – ViewModel), qui garantit une séparation nette des responsabilités entre les différentes couches de l'application, à savoir les données, l'interface utilisateur et la logique métier. Ce modèle permet :

-  **Une séparation claire des responsabilités**, facilitant la gestion et l'évolution de l'application tout en réduisant les dépendances entre les différentes couches.
-  **Une meilleure testabilité et maintenabilité du code**, grâce à une organisation modulaire qui simplifie l'ajout de nouvelles fonctionnalités et le débogage.
-  **Une intégration optimale avec LiveData et ViewModel**, deux composants clés d'Android permettant de gérer de manière réactive

l'interface utilisateur et le cycle de vie des activités et fragments, assurant ainsi une réactivité constante et une gestion efficace des données.

III. Conception fonctionnelle et technique

La phase de conception de l'application **MedWay** repose sur la création de modèles et de diagrammes permettant de clarifier la structure et les interactions au sein de l'application. Ces diagrammes aident à visualiser les différents cas d'utilisation, la structure des données et les processus métier qui animent l'application. Cette section présente les diagrammes clés qui définissent l'architecture fonctionnelle et technique de **MedWay**.

1. Diagramme de cas d'utilisation

1.1 Présentation du diagramme

Ce diagramme illustre les principales fonctionnalités de l'application et la manière dont les utilisateurs interagiront avec celles-ci. Chaque cas d'utilisation représente une action ou un scénario que l'utilisateur peut réaliser dans l'application, et montre les relations entre l'utilisateur et les différentes fonctionnalités (comme l'authentification, la gestion des établissements, la prise de rendez-vous, etc.). Le diagramme sert à définir clairement les attentes fonctionnelles et à orienter la conception des interfaces et des processus métier.

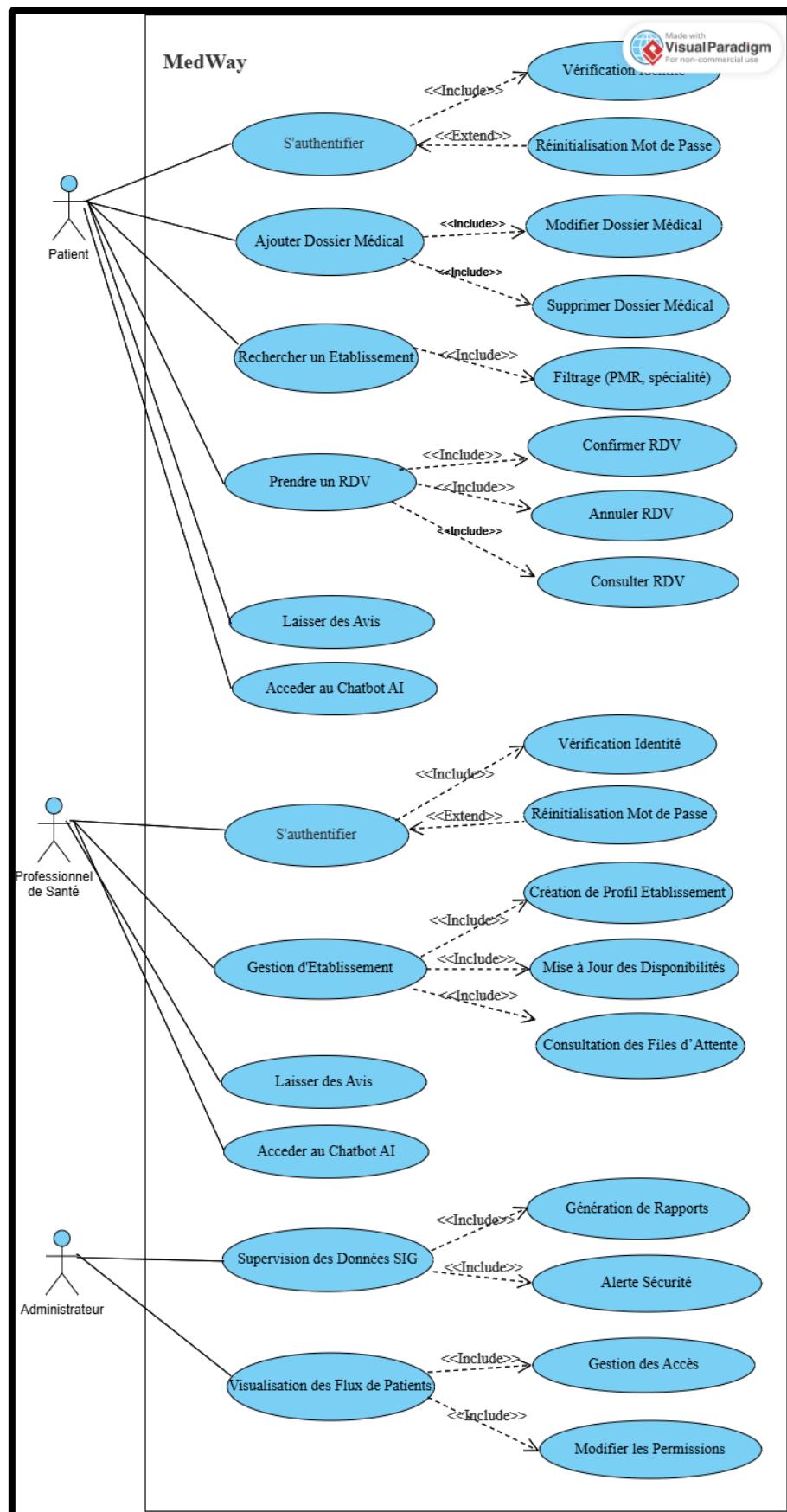


Figure 10 : Diagramme de Cas d'Utilisation

1.2 Description des acteurs

Le système MedWay fait intervenir trois types d'acteurs principaux, chacun ayant un rôle spécifique dans l'utilisation de la plateforme. Ces acteurs interagissent avec différentes fonctionnalités selon leurs besoins et responsabilités au sein du système.

Patient

Utilisateur principal du système. Il peut :

- ❖ S'authentifier pour accéder à son espace personnel.
- ❖ Gérer son dossier médical (ajout, modification, suppression).
- ❖ Rechercher un établissement de santé (avec filtres comme PMR ou spécialité).
- ❖ Prendre, consulter, confirmer ou annuler un rendez-vous.
- ❖ Laisser des avis sur les établissements ou professionnels.
- ❖ Utiliser le chatbot AI pour poser des questions ou obtenir de l'aide.

Professionnel de Santé

Utilise l'application pour gérer son activité médicale. Il peut :

- ❖ S'authentifier pour accéder à l'interface professionnelle.
- ❖ Gérer un établissement (création de profil, mise à jour des disponibilités, consultation des files d'attente).
- ❖ Laisser des avis.
- ❖ Accéder au chatbot AI.

Administrateur

Supervise l'ensemble du système. Il peut :

- ❖ Superviser les données SIG (génération de rapports, alertes de sécurité).
- ❖ Visualiser les flux de patients.
- ❖ Gérer les accès et les permissions des utilisateurs.

1.3 Cas d'utilisation principaux

Les cas d'utilisation principaux du système représentent les fonctionnalités essentielles accessibles par les utilisateurs. Ces cas permettent de répondre aux besoins fonctionnels de manière claire et structurée selon le profil de l'acteur.

☞ S'authentifier

Permet à tout utilisateur d'accéder à son espace personnel en fonction de son rôle (patient, professionnel ou administrateur). Ce processus inclut la vérification de l'identité ainsi qu'une option de réinitialisation du mot de passe en cas d'oubli.

☞ Ajouter un Dossier Médical

Offre au patient la possibilité de créer et de gérer son dossier médical personnel, incluant la mise à jour ou la suppression des informations de santé.

☞ Prendre un Rendez-Vous

Permet au patient de réserver un créneau avec un établissement médical. Il peut ensuite consulter, confirmer ou annuler ce rendez-vous selon ses disponibilités.

☞ Rechercher un Établissement

Facilite la recherche d'établissements de santé à partir de critères spécifiques tels que la spécialité médicale, l'accessibilité PMR ou la localisation.

☞ Gestion d'Etablissement

Le professionnel de santé peut créer ou modifier le profil de son établissement, gérer les créneaux de disponibilité, et suivre les files d'attente des patients.

☞ Supervision des Données SIG

L'administrateur supervise les informations géographiques du système (SIG), génère des rapports d'activité et surveille les alertes de sécurité afin d'assurer la fiabilité et la conformité des données.

Visualisation des Flux de Patients

Offre à l'administrateur une vue d'ensemble des mouvements et de la répartition des patients, permettant une meilleure planification et une gestion optimisée des ressources.

Laisser un Avis

Permet aux patients et aux professionnels de santé de partager leur retour d'expérience sur les établissements ou les services reçus, contribuant ainsi à l'amélioration continue de la qualité.

Utiliser le Chatbot IA

Fournit une assistance automatisée grâce à une intelligence artificielle capable de répondre aux questions des utilisateurs, les orienter dans l'application ou les aider dans leurs démarches.

2. Diagramme de classe

2.1 Présentation du diagramme

Ce diagramme décrit la structure statique de l'application en définissant les classes principales, leurs attributs et leurs relations. Il aide à représenter les entités et leurs interactions au sein du système, comme les relations entre les utilisateurs, les établissements, les rendez-vous et les dossiers médicaux. Ce diagramme permet de modéliser les objets de l'application, leurs propriétés et les méthodes qui y sont associées.

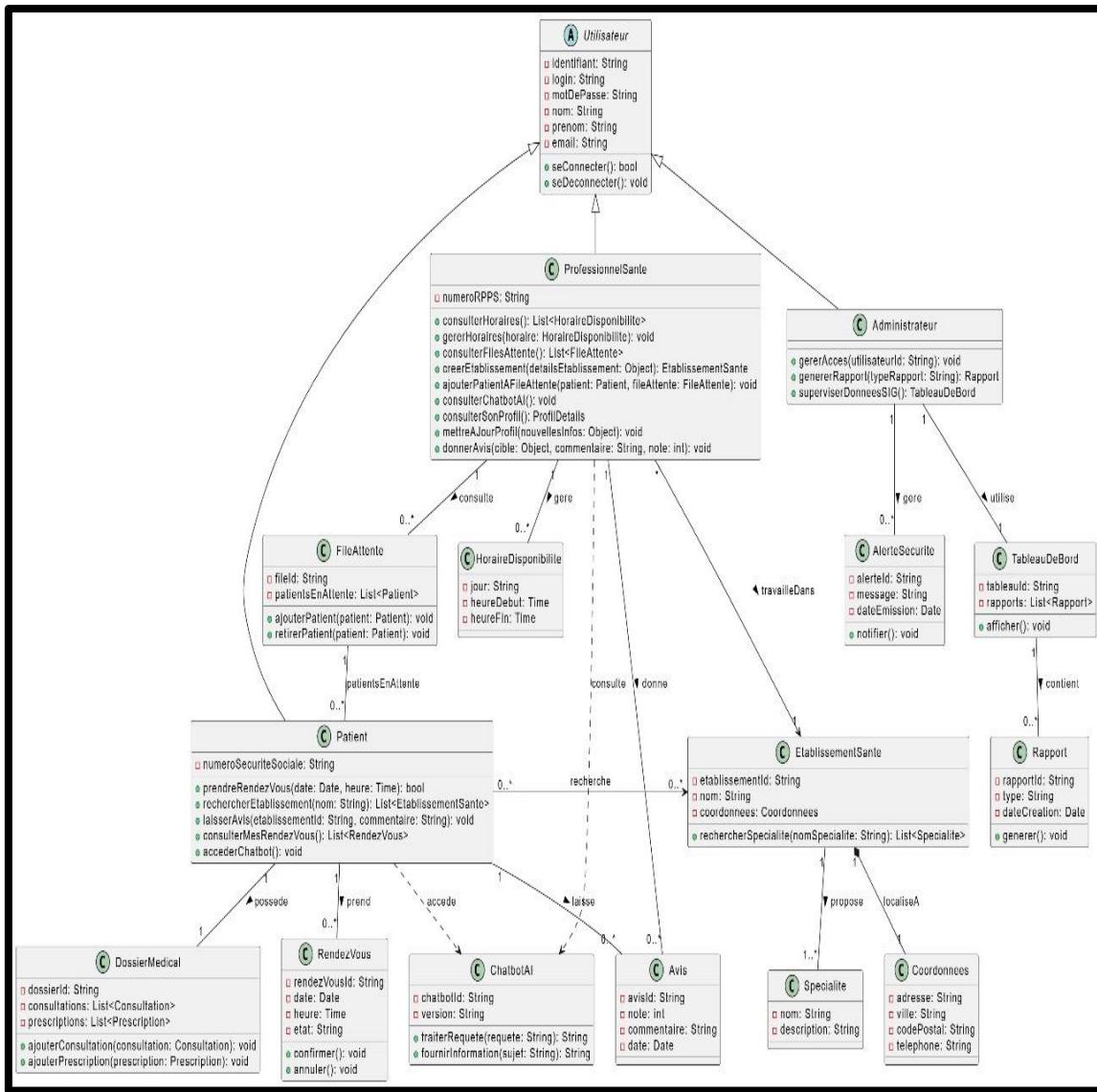


Figure 11 : Diagramme de Classe

2.2 Description des acteurs

Le diagramme de classes présente les entités principales du système **MedWay**, leurs attributs et les méthodes associées. Il reflète la structure orientée objet utilisée pour implémenter les fonctionnalités décrites dans les cas d'utilisation.

❖ **Utilisateur** est la classe de base, commune à tous les types d'acteurs (Patient, ProfessionnelDeSante, Administrateur). Elle contient les informations générales comme l'identifiant, le login, le mot de passe, le nom, prénom et email, ainsi que des méthodes pour se connecter et se déconnecter.

- ❖ **Patient**, héritant d'Utilisateur, possède des méthodes spécifiques telles que la prise de rendez-vous, la consultation des rendez-vous, la recherche d'établissements, la gestion de son dossier médical et l'accès au chatbot. Il peut également laisser des avis.
- ❖ **ProfessionnelDeSante**, également hérité d'Utilisateur, peut gérer les horaires de disponibilité, créer un établissement, consulter les files d'attente, interagir avec le chatbot et donner des avis. Il peut également consulter et modifier les informations de son profil.
- ❖ **Administrateur** hérite de Utilisateur et dispose de privilèges étendus : il peut gérer les accès, générer des rapports, et superviser les données SIG via un tableau de bord.
- ❖ **DossierMedical** est associé à un patient et regroupe les consultations et les prescriptions. Il peut être mis à jour par l'ajout de nouvelles consultations ou prescriptions.
- ❖ **RendezVous** contient les informations liées à un rendez-vous (date, heure, statut) et est associé à un patient.
- ❖ **EtablissementSante** regroupe les informations d'un établissement (nom, coordonnées, spécialités). Il peut être recherché par spécialité.
- ❖ **FileAttente** est utilisée pour gérer les patients en attente. Elle permet d'ajouter ou de retirer un patient.
- ❖ **HoraireDisponibilite** décrit les plages horaires des professionnels de santé.
- ❖ **Avis** représente les retours des utilisateurs sur les établissements. Il contient une note, un commentaire, une date, et l'identifiant de l'auteur.
- ❖ **ChatbotAI** est la classe chargée de répondre aux requêtes des utilisateurs via une IA, en fournissant des informations sur différents sujets.
- ❖ **AlerteSecurite** permet de signaler des problèmes ou risques dans le système. Elle contient un message, une date, et peut déclencher des notifications.
- ❖ **TableauDeBord** permet à l'administrateur d'accéder à un ensemble de rapports. Il sert de point de centralisation pour la supervision du système.

- ❖ **Rapport** contient les informations issues des analyses générées (type, date, contenu).

2.3 Relations entre les classes

Les relations dans ce diagramme décrivent les associations logiques entre les entités du système :

- ❖ Les classes **Patient**, **ProfessionnelDeSante** et **Administrateur** héritent de la classe **Utilisateur**, partageant ainsi les méthodes de connexion.
- ❖ Un **Patient** est lié à un **DossierMedical**, et peut avoir plusieurs **RendezVous**. Il est aussi associé à des **Avis** qu'il peut laisser sur un **EtablissementSante**.
- ❖ Un **ProfessionnelDeSante** peut être lié à plusieurs **HoraireDisponibilite**, gérer un ou plusieurs **FileAttente**, et est associé à un **EtablissementSante**.
- ❖ La classe **FileAttente** contient une liste de **Patients** en attente et propose des méthodes pour ajouter ou retirer un patient.
- ❖ **EtablissementSante** peut proposer plusieurs **Specialite**, et est localisé via une instance de la classe **Coordonnees**.
- ❖ Le **ChatbotAI** est accessible aussi bien par les patients que les professionnels de santé, via des requêtes de type string et des réponses personnalisées.
- ❖ **AlerteSecurite** est utilisée dans la supervision du système, et est liée au **TableauDeBord**, qui regroupe aussi plusieurs **Rapports**.
- ❖ Un **Rapport** est généré par l'**Administrateur**, accessible via le **TableauDeBord** pour l'analyse des données SIG.

3. Diagramme de séquence

Le diagramme de séquence présente les interactions entre les objets du système au fil du temps. Il illustre les étapes nécessaires pour accomplir des tâches spécifiques, comme l'inscription d'un utilisateur, la recherche d'un établissement ou la gestion d'un rendez-vous. Ce diagramme permet de visualiser la chronologie des événements et l'ordre dans lequel les messages sont envoyés entre les objets.

3.1 Diagramme de séquence - Patient

Ce diagramme illustre les interactions entre un patient et le système de gestion médicale à travers l'interface de l'application. Le processus débute par l'inscription et l'authentification du patient, où le système vérifie les informations via la base de données avant de retourner les résultats à l'interface. Ensuite, le patient peut rechercher un établissement, ce qui entraîne l'envoi d'une requête par le système à la BDD pour filtrer les établissements, et renvoyer les résultats pour affichage. Lors de la sélection d'un rendez-vous (RDV), le système vérifie les disponibilités et enregistre le RDV dans la BDD avant de renvoyer une confirmation. Le patient peut aussi consulter et éditer ses dossiers médicaux. L'application permet également d'envoyer un message à un chatbot, qui analyse le message et renvoie une réponse. Enfin, le patient peut laisser un avis. Ce diagramme met en évidence une interaction fluide et structurée entre le patient, l'interface, le système et la base de données.

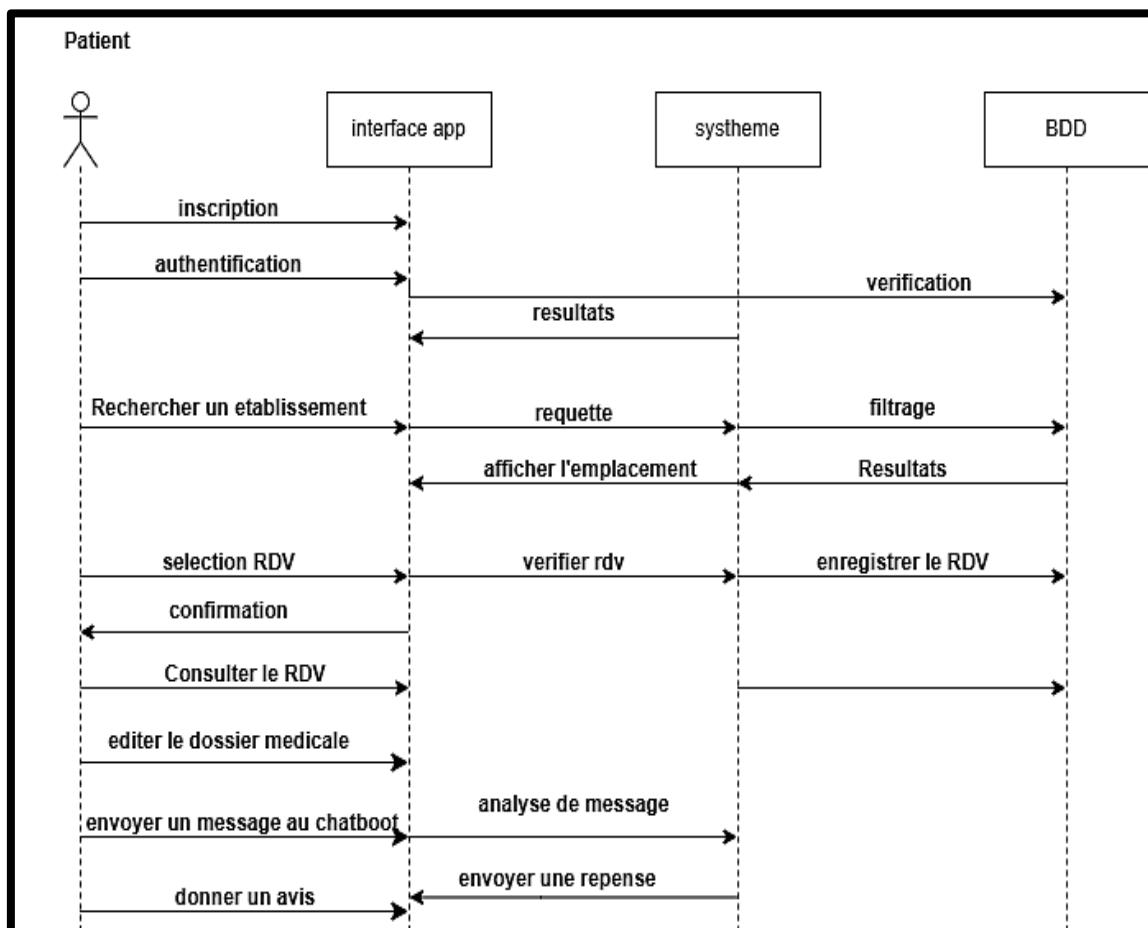


Figure 12 : Diagramme de Séquence - Patient

3.2 Diagramme de séquence – Professionnel de Santé

Ce diagramme présente les interactions entre un médecin et le module de gestion de l'application. Le médecin commence par une inscription et une authentification, vérifiée par le système. Ensuite, il peut créer un établissement, ce qui implique que le système enregistre les nouvelles données dans la base de données. Le médecin peut consulter les files d'attente des patients, organiser les horaires disponibles, et ces modifications sont enregistrées et confirmées par la BDD. Il peut aussi accéder au chatbot basé sur l'intelligence artificielle pour poser des questions ou obtenir de l'aide. Enfin, le médecin a la possibilité de laisser un avis. Le diagramme montre que le médecin dispose de fonctionnalités de gestion des données médicales et organisationnelles, tout en interagissant avec un module intelligent d'aide à la décision.

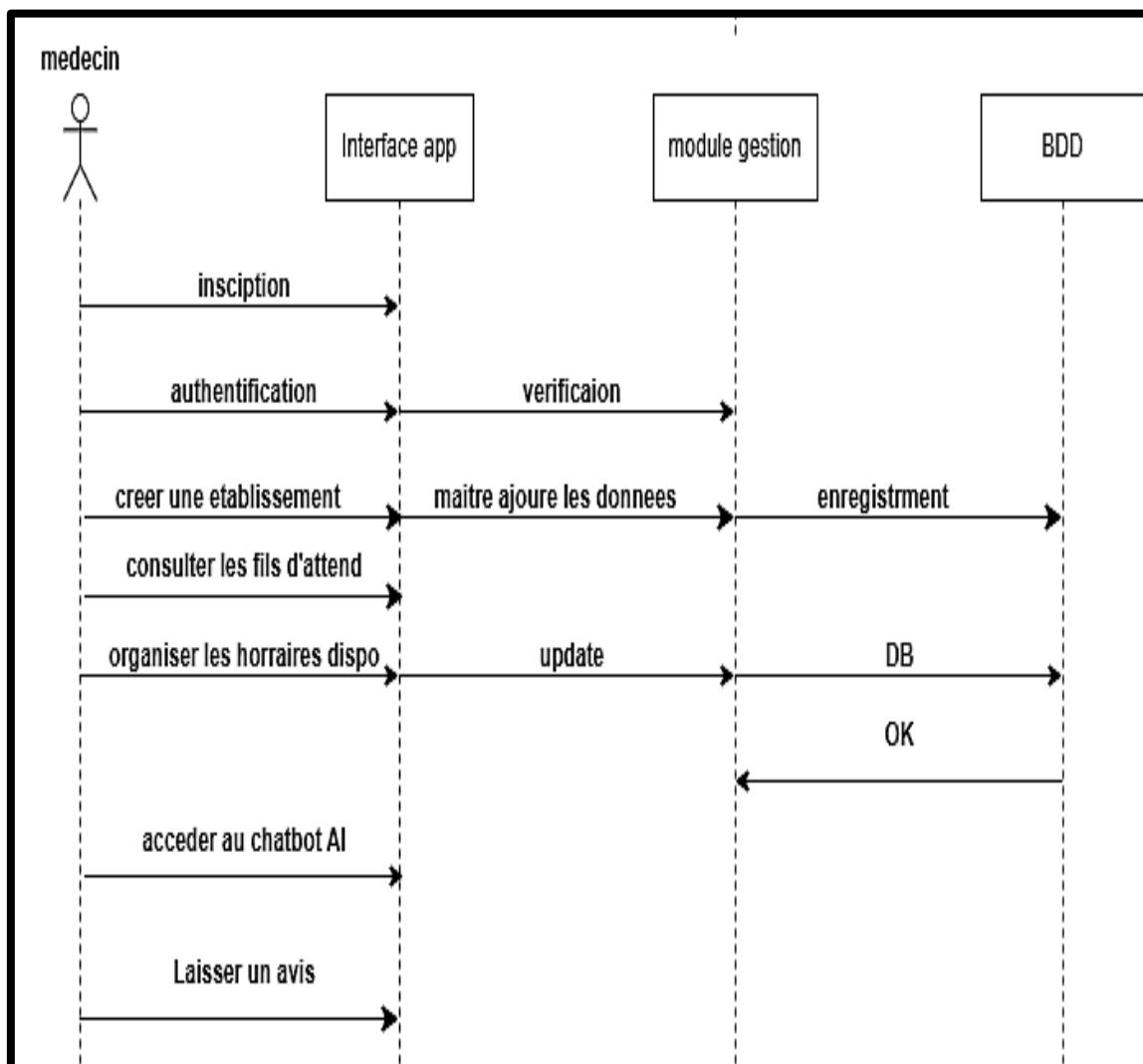


Figure 13 : Diagramme de Séquence – Professionnel de Santé

3.3 Diagramme de séquence – Administrateur

Le dernier diagramme concerne les actions de l'administrateur via une interface dédiée. Après une connexion et une vérification par le système, l'administrateur peut consulter les données, ce qui implique l'envoi d'une requête au module SIG (système d'information géographique), qui interroge la base de données. Les données récupérées permettent d'afficher des résultats sous forme de carte graphique. L'administrateur peut aussi ajouter un nouvel établissement, ce qui est pris en charge par le module SIG qui enregistre l'ajout dans la base de données. Enfin, il peut gérer les accès, probablement en modifiant les autorisations ou les profils des utilisateurs. Ce diagramme montre que l'administrateur a un rôle de supervision, de gestion des établissements et de contrôle des accès au système.

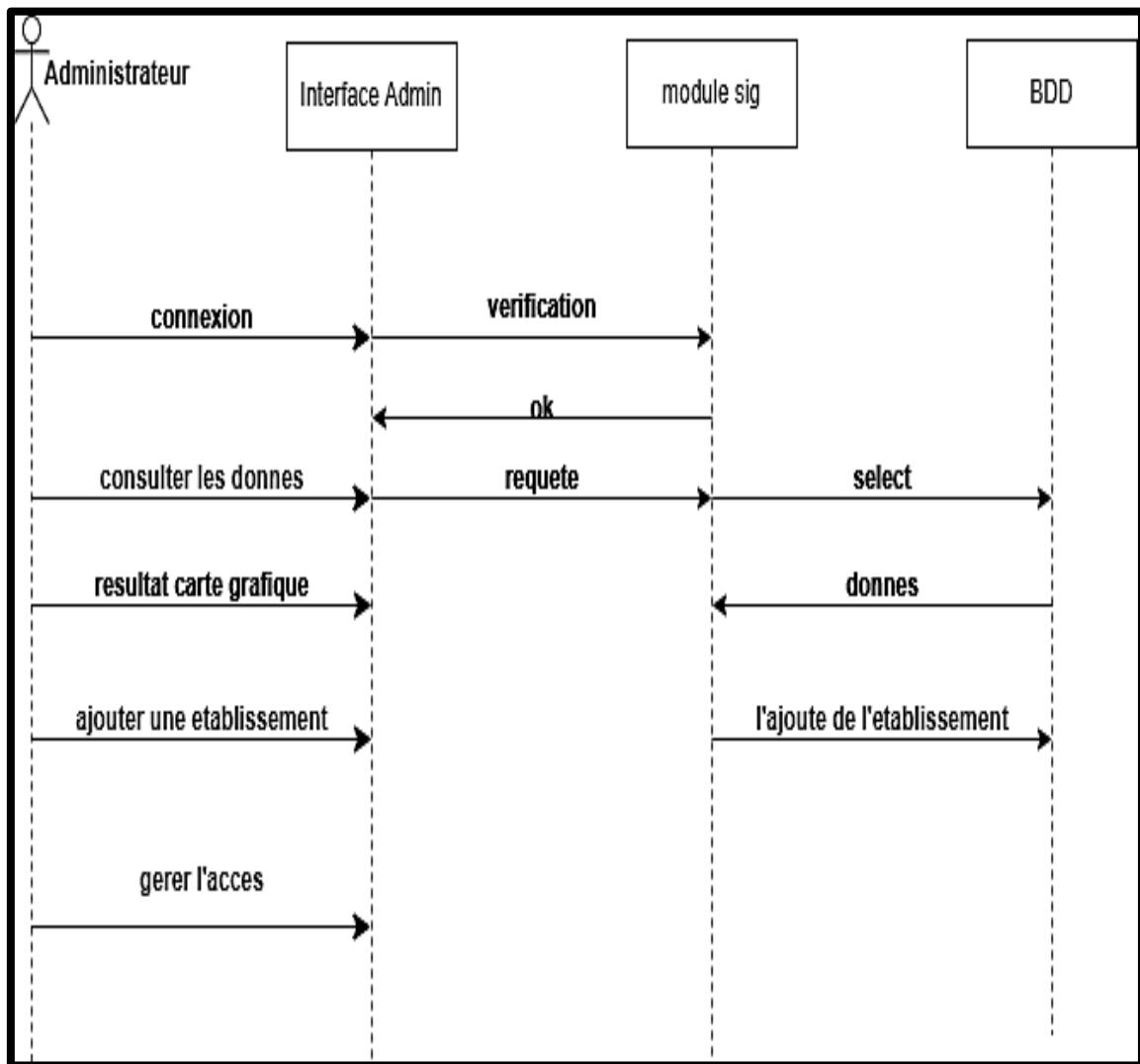


Figure 14 : Diagramme de Séquence – Administrateur

IV. Conclusion

Le chapitre sur la conception de l'application **MedWay** a permis de définir les bases solides de son développement, en mettant l'accent sur le choix des outils, des technologies et de l'architecture adaptée aux besoins de l'application. Grâce à l'utilisation de frameworks modernes et à une approche centrée sur l'utilisateur, **MedWay** est conçu pour offrir une expérience fluide et intuitive. Les diagrammes UML, tels que les cas d'utilisation, les diagrammes de classe et de séquence, ont permis de structurer et de clarifier les processus métier, facilitant ainsi la compréhension et la gestion des interactions complexes au sein de l'application.

CHAPITRE IV

RÉALISATION ET DÉVELOPPEMENT

I. Introduction

Le chapitre III se concentre sur la phase cruciale de la réalisation et du développement de l'application **MedWay**. Après avoir défini les bases théoriques et conceptuelles dans le chapitre précédent, il est maintenant temps de passer à l'implémentation concrète des fonctionnalités et à la mise en place de l'architecture technique du projet. Ce chapitre détaille l'architecture du projet, la manière dont les différentes parties de l'application sont organisées et interagissent, ainsi que les processus de développement des fonctionnalités essentielles telles que la gestion des utilisateurs, l'intégration des bases de données locales et distantes, et la mise en œuvre d'interfaces dynamiques.

II. Architecture du projet

L'architecture de l'application **MedWay** a été conçue pour offrir une structure modulaire, flexible et maintenable, facilitant ainsi l'évolution du projet au fil du temps. Elle repose sur une organisation claire des dossiers et des modules, avec une séparation nette entre la gestion des données, la logique métier et l'interface utilisateur. Cette architecture garantit une intégration fluide des différentes fonctionnalités de l'application, telles que la gestion des utilisateurs, l'interaction avec Firebase et l'affichage dynamique des données.

L'objectif est de maintenir une cohérence dans le code tout en assurant la performance, la sécurité et la scalabilité de l'application. La structure est également pensée pour faciliter l'ajout de nouvelles fonctionnalités sans perturber les éléments existants.

Voici l'architecture globale de l'application **MedWay**, pensée pour assurer une intégration fluide, une haute performance et une évolutivité optimale tout au long du développement et de son évolution :

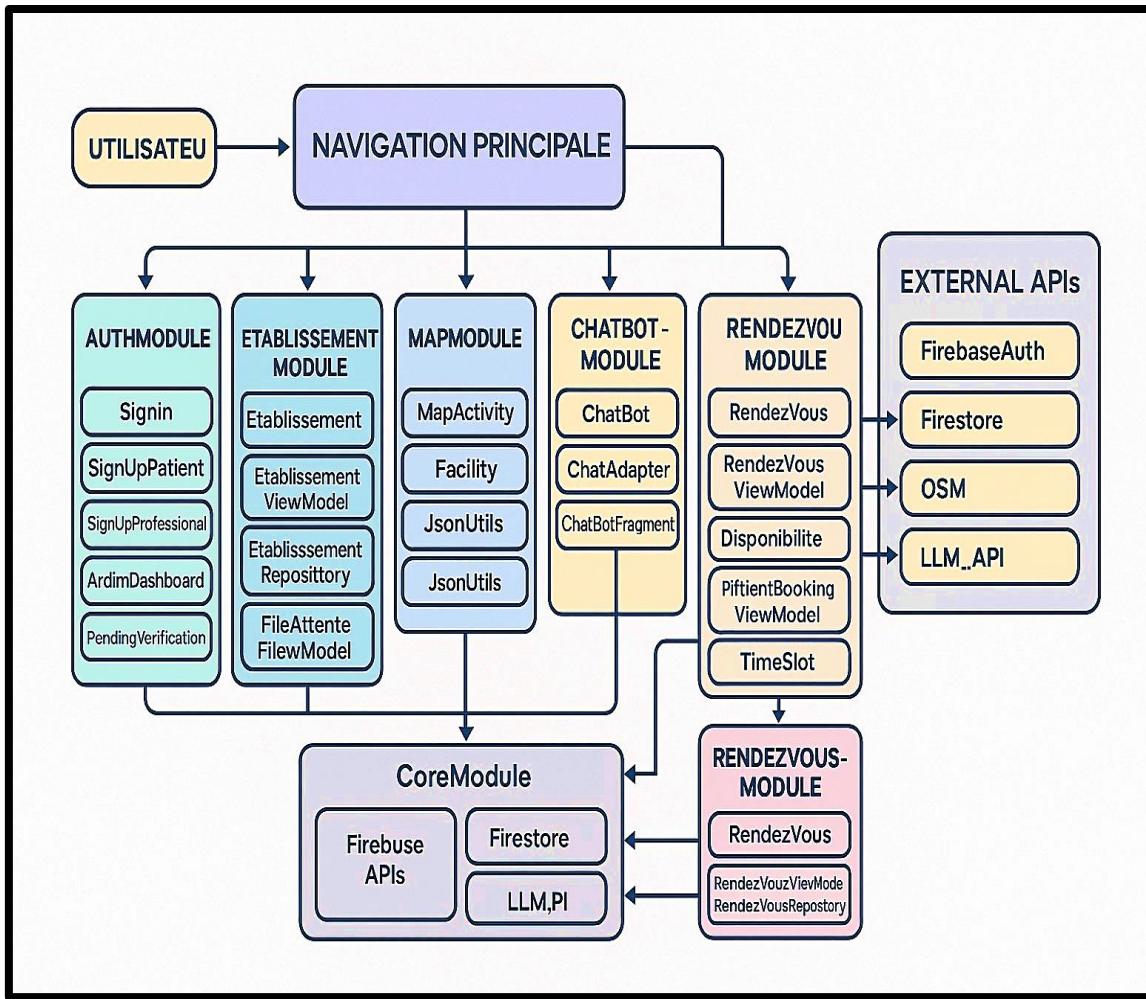


Figure 15 : Architecture du Projet

L'architecture de l'application **MedWay** est modulaire, avec une **navigation principale** reliant l'utilisateur aux différents modules fonctionnels. Le **AuthModule** gère l'inscription et la connexion des patients et professionnels de santé, ainsi que l'accès aux interfaces personnalisées selon le rôle. Chaque module contient ses propres composants (UI, ViewModel, Repository), assurant une séparation claire des responsabilités.

Le **EtablissementModule** permet la gestion des établissements, de leurs disponibilités et des files d'attente, tandis que le **MapModule** exploite des utilitaires pour afficher les établissements sur une carte. Le **ChatbotModule**, quant à lui, fournit une assistance intelligente via une interface conversationnelle intégrée. Le **RendezVousModule** centralise la gestion des créneaux, de la prise de rendez-vous et du suivi des réservations.

Tous les modules communiquent avec le **CoreModule**, qui encapsule les appels aux **API Firebase (Auth, Firestore)**, aux **services de localisation (OSM)** et à une **API d'intelligence artificielle (LLM/PI)** pour les interactions avancées. Cette architecture modulaire garantit une application évolutive, maintenable et facilement extensible avec de nouveaux services.

III. Implémentation des fonctionnalités

L'implémentation des fonctionnalités constitue le cœur du développement de l'application **MedWay**. Dans cette section, nous détaillons les principales fonctionnalités mises en place pour répondre aux besoins spécifiques des utilisateurs tout en garantissant une expérience fluide et efficace. Chaque fonctionnalité a été soigneusement développée en suivant les principes de l'architecture MVVM, en utilisant les outils et technologies appropriés, et en veillant à l'intégration correcte avec la base de données locale et distante.

& Gestion des Utilisateurs (Inscription, Connexion)

L'une des premières étapes dans la mise en place de **MedWay** est la gestion de l'authentification des utilisateurs. Nous avons opté pour **Firebase Authentication**, un service robuste et sécurisé pour la gestion des inscriptions et des connexions. L'authentification permet aux utilisateurs de créer un compte, de se connecter à l'application et de maintenir un accès sécurisé à leurs données personnelles et médicales. Les fonctionnalités incluent la gestion des mots de passe, la validation des informations d'inscription et l'authentification par email.

& Base de Données Distante avec Fonctionnalités Locales (Firestore)

L'application repose sur une architecture connectée à une base distante via Firebase Firestore. Toutes les données sont stockées et synchronisées en temps réel dans le cloud, garantissant un accès à jour et centralisé aux informations. Bien qu'aucune base locale comme Room ne soit utilisée, des mécanismes de persistance automatique fournis par Firestore assurent un fonctionnement partiel hors ligne, avec re-

synchronisation automatique dès le retour de la connectivité. Cette approche garantit une gestion efficace des données tout en offrant une expérience utilisateur fluide.

& Affichage Dynamique (RecyclerView, LiveData)

Pour une interface utilisateur réactive et fluide, **RecyclerView** est utilisé pour afficher dynamiquement les listes d'éléments (par exemple, les établissements médicaux, les rendez-vous, etc.). **LiveData**, intégré avec **ViewModel**, permet de lier les données à l'interface de manière réactive, garantissant que les informations affichées sont toujours actualisées sans avoir besoin d'interventions manuelles. Cela garantit une expérience utilisateur fluide et en temps réel.

& Ajout, Modification, Suppression de Données

L'une des fonctionnalités principales de l'application est la gestion des données utilisateur, y compris l'ajout, la modification et la suppression d'informations telles que les établissements médicaux, les rendez-vous ou les dossiers médicaux. Grâce à l'architecture MVVM et l'utilisation de **Firebase Firestore**, ces opérations sont réalisées de manière rapide et sécurisée. Chaque modification de données déclenche automatiquement une mise à jour de l'interface utilisateur, assurant une cohérence totale entre la base de données et l'affichage.

& Fonctionnalités Supplémentaires

N Recherche d'Établissements de Santé en Temps Réel avec Filtrage Avancé

Les utilisateurs peuvent rechercher des établissements de santé en temps réel, avec un système de filtrage avancé par type d'établissement, localisation, spécialité, disponibilité, et bien plus encore. Cette fonctionnalité garantit aux utilisateurs de trouver rapidement les établissements correspondant à leurs besoins spécifiques.

N Prise de Rendez-Vous en Ligne et Gestion des Disponibilités

L'application permet aux utilisateurs de prendre des rendez-vous en ligne en consultant les disponibilités des professionnels de santé. Un système de rappels automatiques est intégré pour notifier les utilisateurs avant leurs rendez-vous, garantissant ainsi une gestion optimale des créneaux.

❖ Gestion Complète du Dossier Médical

Les utilisateurs peuvent créer, consulter, modifier et supprimer leur dossier médical, incluant les antécédents médicaux, les traitements suivis, ainsi que les résultats des examens et consultations. Cela permet aux patients de garder un historique médical complet et facilement accessible.

❖ Authentification Sécurisée via Firebase Authentication

L'authentification des utilisateurs est gérée de manière sécurisée grâce à **Firebase Authentication**, garantissant l'accès sécurisé aux informations personnelles et médicales des utilisateurs, en toute confiance.

❖ Espace Professionnel Personnalisé

L'application inclut un espace professionnel personnalisé permettant aux professionnels de santé de gérer les établissements, les créneaux de disponibilité et les files d'attente. Cette fonctionnalité facilite la gestion de leur emploi du temps et des demandes des patients.

❖ Chatbot Intelligent

Un chatbot intelligent est intégré pour guider les utilisateurs dans leurs démarches, répondre à des questions médicales générales et offrir une assistance interactive lors de l'utilisation de l'application.

❖ Système d'Avis et de Notation

Un système d'avis et de notation est mis en place pour permettre aux patients de partager leurs retours sur les établissements et les services médicaux, contribuant à améliorer la qualité des soins et l'expérience utilisateur.

❖ Génération de Rapports Statistiques

Un module de génération de rapports est mis à disposition de l'administrateur, lui permettant d'analyser en temps réel l'activité de la plateforme (fréquentation, prises de

rendez-vous, évaluations, interactions), afin d'optimiser la gestion globale du système et d'orienter les prises de décision.

IV. Intégration et sécurité

Dans cette section, nous abordons les aspects relatifs à l'intégration et à la sécurité de l'application **MedWay**, en mettant l'accent sur la sécurisation des données, la gestion des erreurs et exceptions, ainsi que l'optimisation des performances pour garantir une expérience utilisateur fiable et fluide.

1. Sécurisation des données

La sécurité des données des utilisateurs est une priorité pour **MedWay**. Pour assurer la confidentialité et la protection des informations personnelles et médicales, plusieurs techniques de sécurisation sont mises en place :

☞ Hash des Mots de Passe

Tous les mots de passe des utilisateurs sont cryptés avant d'être stockés dans Firebase Authentication pour éviter toute fuite ou compromission de données. Nous utilisons un algorithme de hachage sécurisé (tel que SHA-256) pour garantir que les mots de passe ne sont jamais stockés en clair.

☞ Règles de Sécurité Firebase

Les Firebase Security Rules sont utilisées pour définir des autorisations d'accès strictes à la base de données Firebase Firestore. Seuls les utilisateurs authentifiés peuvent accéder à leurs propres données, et des règles spécifiques sont mises en place pour garantir que chaque utilisateur ou professionnel de santé n'accède qu'à ses informations pertinentes. Ces règles sont essentielles pour assurer une gestion sécurisée des données sensibles telles que les dossiers médicaux.

☞ Protection des Communications

Toutes les données échangées entre l'application et Firebase sont sécurisées via HTTPS, garantissant ainsi une transmission chiffrée des informations entre les utilisateurs et les serveurs.

2. Gestion des erreurs et exceptions

Une gestion efficace des erreurs et des exceptions est cruciale pour assurer la stabilité de l'application et une expérience utilisateur sans accroc. **MedWay** intègre une série de mécanismes de gestion des erreurs pour capturer et gérer les exceptions afin de prévenir toute perte de données ou interruption de service. Ces mécanismes incluent :

☞ Traitement des Exceptions

Toute exception (par exemple, une tentative d'accès non autorisé ou une erreur de connexion à la base de données) est capturée et traitée afin d'afficher des messages d'erreur compréhensibles pour l'utilisateur tout en évitant les crashes de l'application.

☞ Logging et Monitoring

Des outils de logging sont utilisés pour suivre les erreurs critiques et les comportements anormaux de l'application. Par exemple, Firebase Crashlytics permet de suivre les crashes en temps réel et de fournir des rapports détaillés pour corriger rapidement les problèmes.

☞ Affichage des Messages d'Erreur

Lorsque des erreurs surviennent (par exemple, lors de la connexion ou de la mise à jour des données), l'application affiche des messages d'erreur clairs et compréhensibles pour l'utilisateur, lui permettant de prendre les mesures appropriées, comme réessayer l'action ou contacter l'assistance.

3. Optimisation des performances

Pour garantir une expérience utilisateur fluide et réactive, l'application **MedWay** a été conçue avec un accent particulier sur l'optimisation des performances. Plusieurs stratégies ont été mises en place pour améliorer la vitesse, la réactivité et l'efficacité de l'application, y compris :

☞ Chargement et Gestion Optimisée des Données

L'application utilise des outils tels que LiveData et RecyclerView pour assurer un affichage dynamique et réactif des données. Cela permet d'éviter le

rechargement complet des données, limitant ainsi l'usage de la bande passante et améliorant la vitesse de réponse.

Gestion de la Mémoire

L'application optimise l'utilisation de la mémoire en chargeant uniquement les données nécessaires au moment où elles sont requises. De plus, Glide et Picasso sont utilisés pour le chargement des images de manière asynchrone, évitant ainsi des ralentissements dus à un chargement excessif de ressources visuelles.

Optimisation de la Base de Données

En utilisant Firebase Firestore, les données sont automatiquement synchronisées avec les serveurs distants. Cependant, l'application optimise les requêtes pour éviter les appels excessifs à la base de données, réduisant ainsi le temps de réponse et la consommation de données.

V. Conclusion

Ce chapitre a présenté les différentes étapes techniques de développement de l'application **MedWay**, depuis l'architecture du projet jusqu'à l'implémentation des principales fonctionnalités. L'accent a été mis sur la sécurité, la performance, l'ergonomie et la fiabilité du système. Grâce à une intégration cohérente des outils comme Firebase, Firestore, et OpenStreetMap, l'application offre une expérience riche, fluide et sécurisée, adaptée aux besoins des utilisateurs du secteur de la santé.

CHAPITRE V

MAQUETTES DE

L'APPLICATION

I. Introduction

Les maquettes jouent un rôle fondamental dans le processus de conception d'une application. Elles permettent de représenter visuellement la structure et l'organisation des interfaces avant le développement. Ce chapitre présente les différentes maquettes élaborées afin de définir clairement l'apparence, la disposition des éléments et les parcours utilisateurs.

II. Inscription de l'utilisateur

1. Inscription du patient

L'inscription du patient permet à chaque utilisateur d'accéder à l'application mobile **MedWay** de façon sécurisée et personnalisée. À travers une interface intuitive, le patient renseigne son nom, son adresse email et son mot de passe. Ces informations sont essentielles pour authentifier l'utilisateur et garantir la confidentialité des données médicales. Ce processus constitue une étape fondamentale pour permettre un accès structuré aux services de santé offerts par l'application, tels que la prise de rendez-vous, le suivi médical et la communication avec les professionnels.



Figure 16 : Inscription du Patient

2. Incription du professionnel de santé

L'inscription du professionnel de santé permet à chaque praticien d'accéder à l'application mobile **MedWay** de manière sécurisée et personnalisée. À travers une interface intuitive, le professionnel renseigne son nom, son adresse email, son mot de passe, son numéro de licence médicale et sa spécialité. Ces informations assurent l'authentification de l'utilisateur et la fiabilité des données médicales échangées. Ce processus constitue une étape clé pour garantir un accès contrôlé et structuré aux services offerts par l'application.



Figure 17 : Inscription du Professionnel de Santé

III. Authentification de l'utilisateur

L'interface de connexion de l'application mobile **MedWay** permet aux utilisateurs, qu'ils soient patients ou professionnels de santé, d'accéder à leur espace personnel de manière sécurisée. Elle offre deux champs simples pour saisir l'adresse email et le mot de passe, avec une option pour afficher ou masquer ce dernier. Un lien "Mot de passe oublié ?" facilite la récupération d'accès en cas d'oubli. L'interface propose également un accès rapide à l'inscription, en fonction du profil de l'utilisateur.

Ce point d'entrée assure une authentification fiable et un accès personnalisé aux fonctionnalités de **MedWay**.



Figure 18 : Authentification de l'Utilisateur

IV. Fonctionnalités de l'administrateur

L'interface de l'administrateur dans l'application **MedWay** est conçue pour offrir un contrôle global et une supervision efficace du système. Sur le tableau de bord mobile, l'administrateur a accès à la gestion des demandes d'inscription des

professionnels de santé. Chaque demande affiche des informations essentielles telles que le nom, l'adresse e-mail, le numéro de licence et la spécialité du médecin. Pour chaque inscription, deux actions sont possibles : **accepter** ou **rejeter** la demande. Un bouton de rafraîchissement permet également de mettre à jour la liste des demandes en temps réel.

L'administrateur peut effectuer la **supervision des données SIG**, ce qui inclut la **génération de rapports** et la réception d'**alertes de sécurité**, afin de garantir un bon fonctionnement et une surveillance proactive du système. Il a également la possibilité de **visualiser les flux de patients**, c'est-à-dire suivre en temps réel les mouvements des patients à travers les établissements de santé. Cette fonctionnalité s'accompagne de la **gestion des accès**, permettant de définir les droits d'utilisation, et de la **modification des permissions**, qui offre la possibilité d'ajuster les priviléges des utilisateurs selon leur rôle ou besoin.

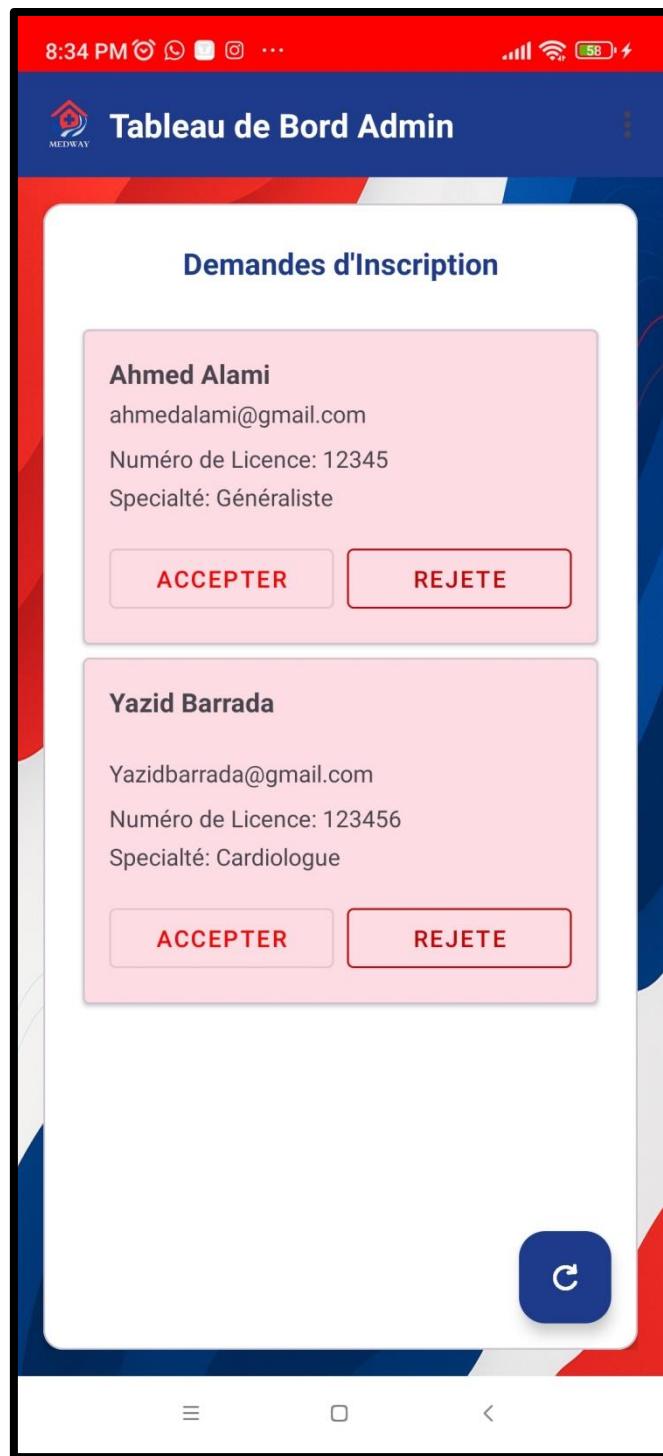


Figure 19 : Tableau de Bord du Administrateur



Figure 20 : Fonctionnalités de l'Administrateur

V. Fonctionnalités du professionnel de santé

1. Gestion des établissements de santé

1.1 Crédation du profil d'un établissement de santé

L'interface de création d'établissement dans **MedWay** permet aux professionnels de santé d'enregistrer de nouvelles structures médicales via un formulaire complet comprenant le nom, l'adresse, les coordonnées, le responsable et le type d'établissement. Elle intègre également une liste actualisée des établissements ajoutés, offrant une vision d'ensemble du réseau médical. Ce module central garantit une gestion uniforme des structures de santé tout en permettant leur identification claire au sein de la plateforme.

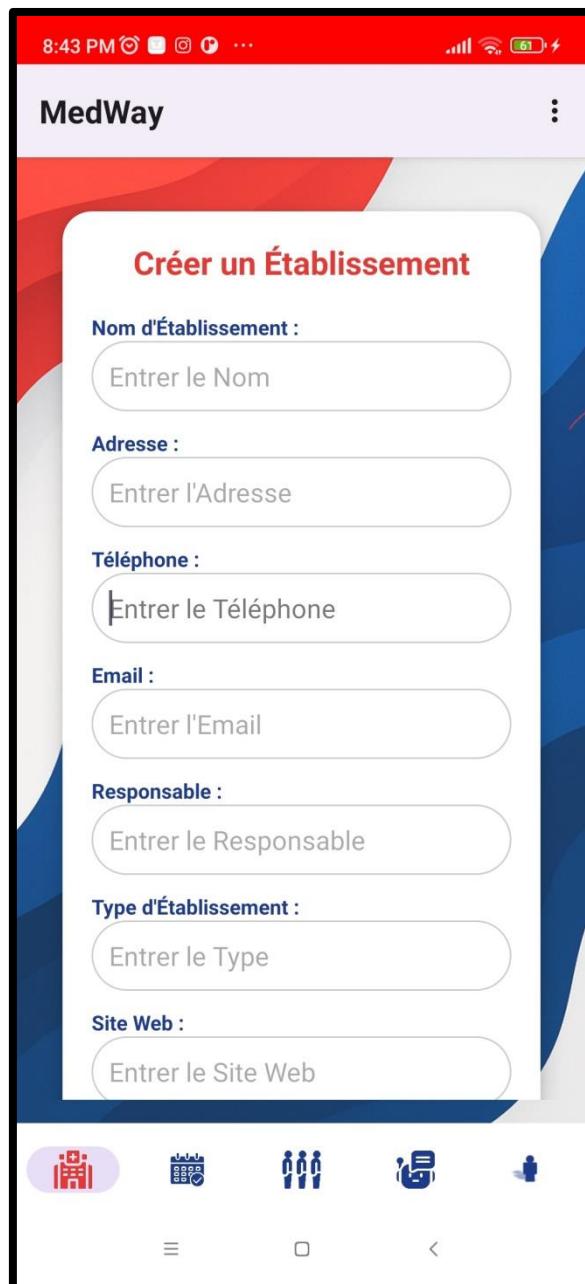


Figure 21 : Crédation du Profil d'un établissement de Santé

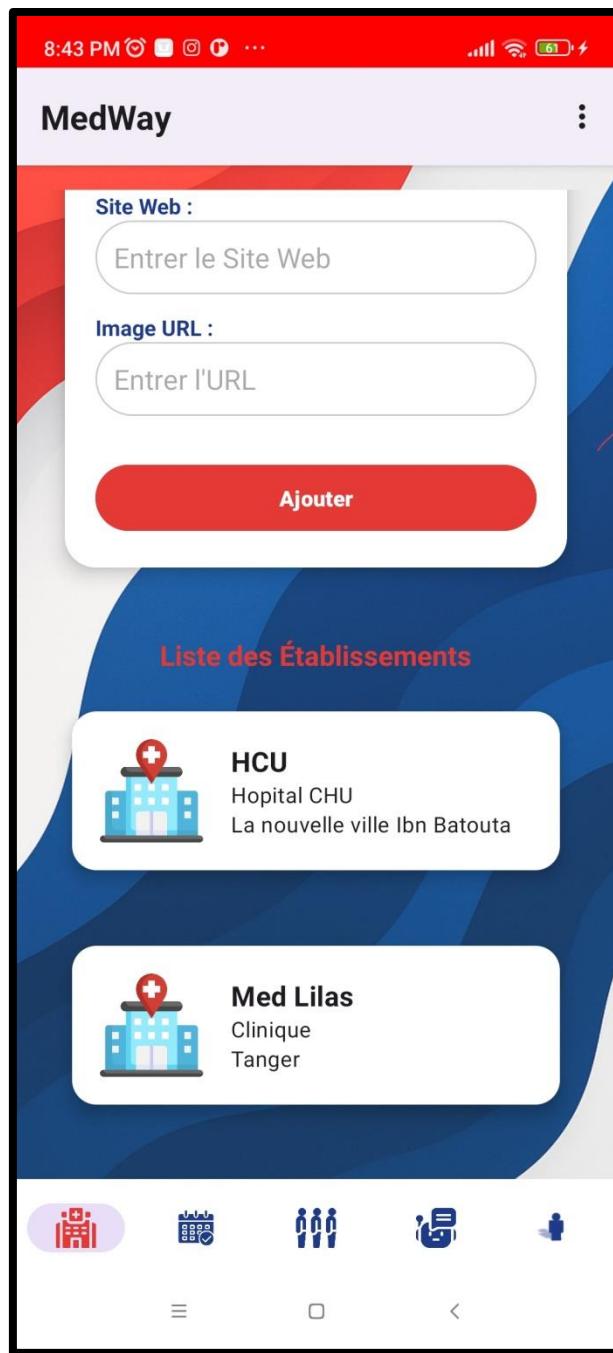


Figure 22 : Liste des établissements de Santé

1.2 Mise à jour des disponibilités

L'interface de gestion des disponibilités des établissements de santé dans **MedWay** permet aux administrateurs de configurer les créneaux d'ouverture de leur structure de manière sécurisée et centralisée. Elle offre des champs intuitifs pour sélectionner l'établissement, définir les jours et horaires, avec une option simple pour activer chaque journée. La validation via le bouton "Enregistrer Disponibilité" assure une mise à jour fiable des plannings qui impacte l'ensemble des professionnels rattachés

à l'établissement. Ce module clé garantit une gestion cohérente des disponibilités institutionnelles et une harmonisation des agendas au sein de la structure médicale. Une liste complète des disponibilités enregistrées est accessible, permettant une visualisation rapide de l'ensemble des plages horaires configurées. Cette vue d'ensemble facilite le contrôle et la vérification des horaires d'ouverture de l'établissement.

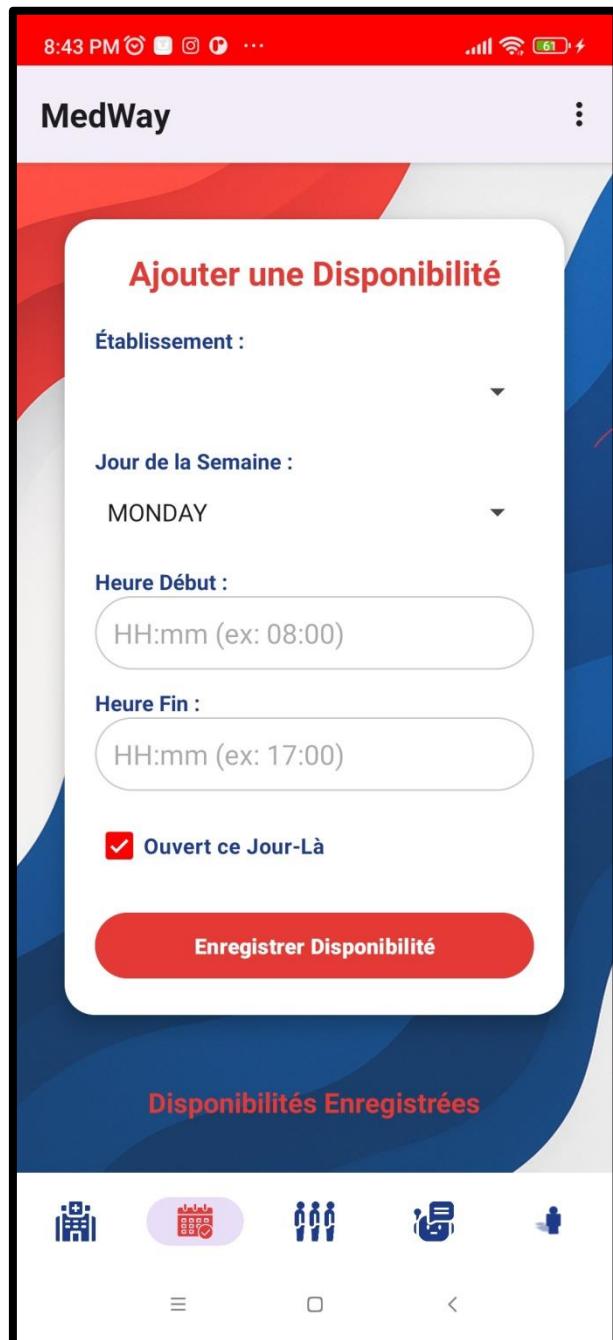


Figure 23 : Mise à jour des Disponibilités

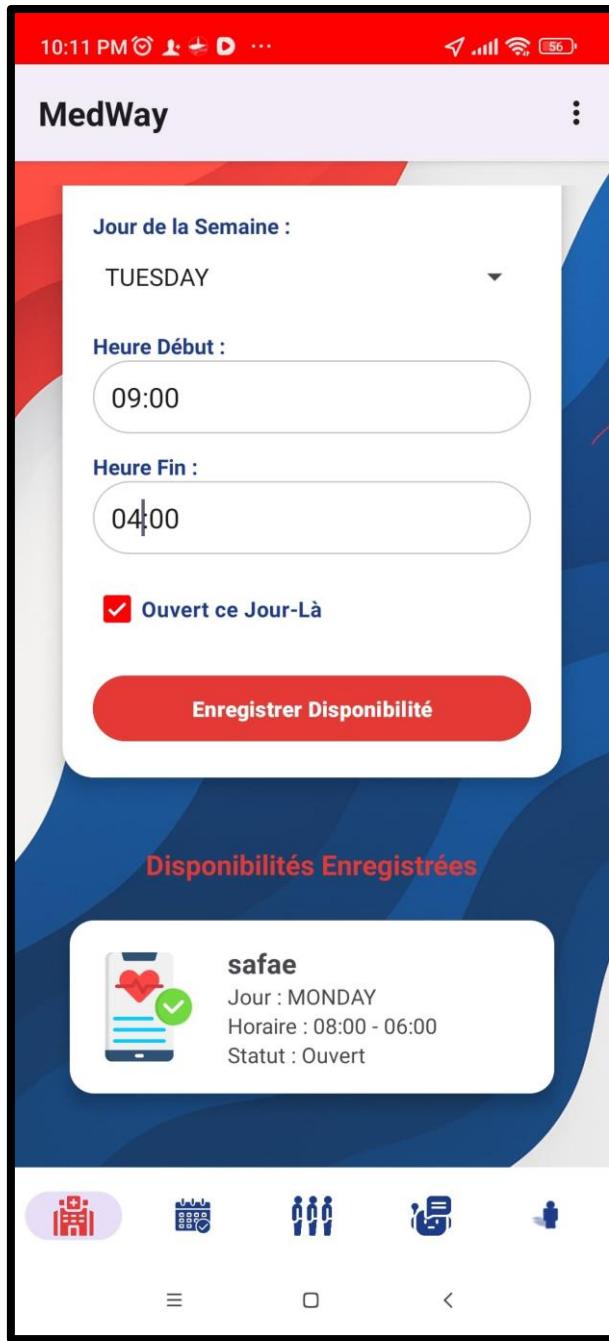


Figure 24 : Liste des Disponibilités

1.3 Consultation des files d'attente

L'interface d'ajout de clients à la file d'attente dans **MedWay** permet aux professionnels de santé d'enregistrer et de gérer les patients en attente de consultation de manière structurée et efficace. Elle propose des champs clairs pour saisir les informations du client (nom, prénom), l'heure d'arrivée, le motif de la visite et le statut actuel. La validation via le bouton "Ajouter à la File" assure une intégration immédiate dans le système de gestion des flux patients. Ce module essentiel comprend également

une liste complète et actualisée en temps réel de tous les patients en attente, offrant une vision claire et organisée de la file d'attente.

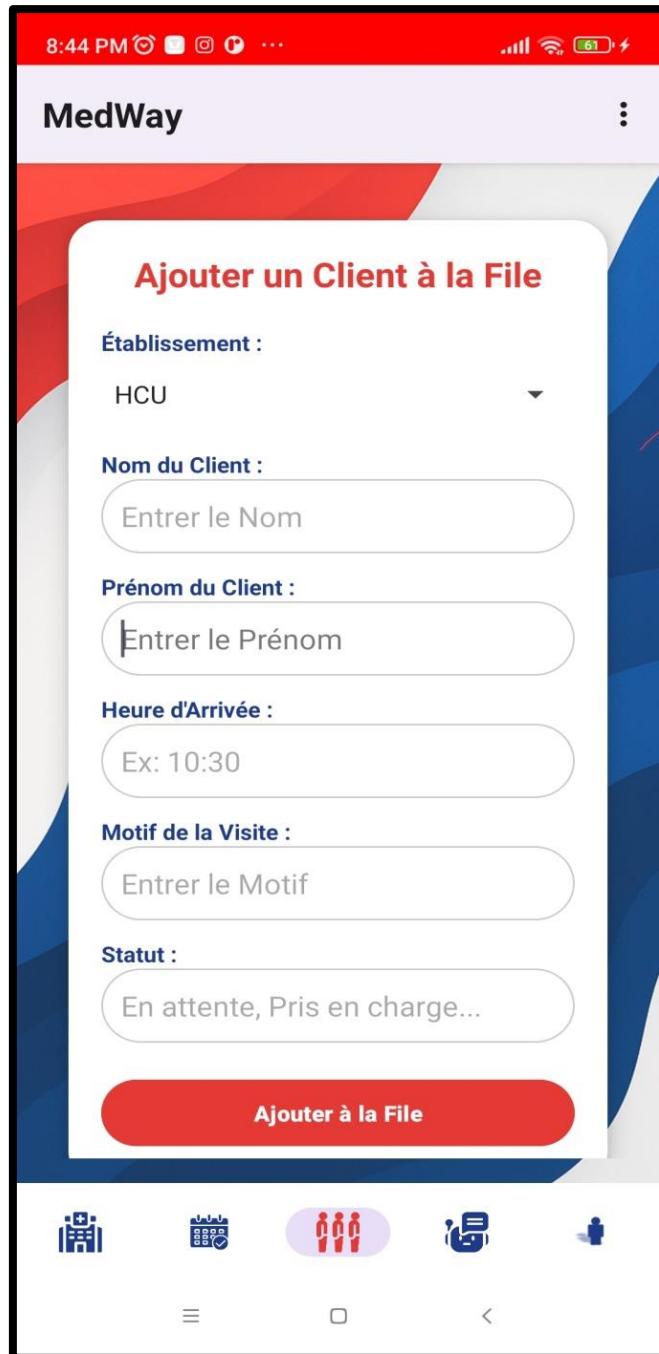


Figure 25 : Consultation des Files d'Attente

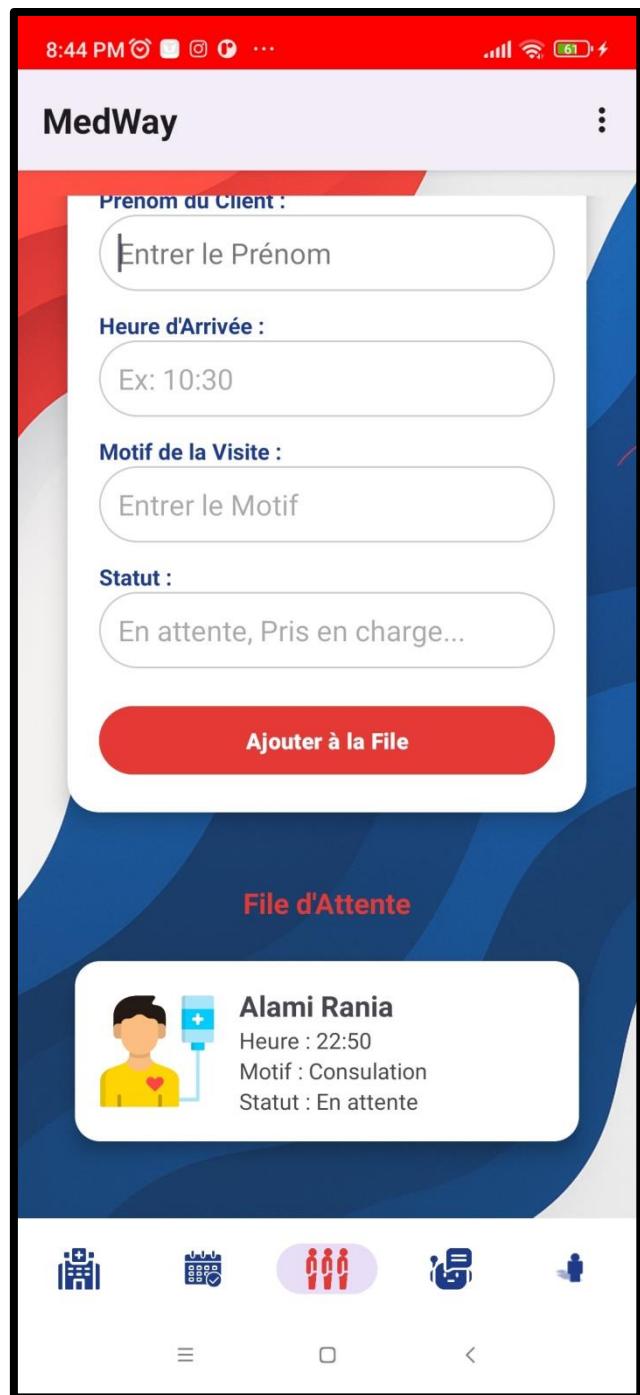


Figure 26 : File d'Attente

2. Chatbot intelligent IA

Le chatbot intelligent IA de **MedWay** pour les professionnels de santé est un outil d’assistance rapide et fonctionnel, conçu pour répondre de manière claire aux questions liées à l’utilisation de l’application. Il permet, par exemple, de mettre à jour ses disponibilités en guidant l’utilisateur vers la section appropriée de son profil, ou de créer un profil d’établissement en expliquant les étapes à suivre dans la gestion

administrative. En cas d'oubli de mot de passe, il indique simplement la procédure de réinitialisation via l'adresse e-mail professionnelle. Il peut aussi aider à consulter les patients en attente en redirigeant vers la section des files d'attente. Grâce à ce système interactif, les professionnels obtiennent rapidement des réponses concrètes à leurs besoins quotidiens, ce qui facilite leur organisation et améliore leur efficacité dans l'usage de l'application.



Figure 27 : Chatbot Intelligent IA

3. Gestion du profil

La gestion du profil dans **MedWay** constitue un élément fondamental de l'expérience des professionnels de santé, leur offrant une maîtrise optimale de leur compte grâce à des fonctionnalités à la fois intuitives et sécurisées. La mise à jour du profil permet aux praticiens de modifier aisément leurs informations professionnelles, y compris leurs coordonnées, leur numéro de licence et leur spécialité médicale, via une interface transparente qui assure la confidentialité des données sensibles. La publication d'avis donne aux professionnels l'opportunité de partager leurs retours d'expérience, contribuant ainsi à l'amélioration continue de l'application tout en enrichissant les échanges au sein de la communauté médicale. Enfin, la déconnexion sécurisée protège l'accès au compte en un simple clic, garantissant la protection des informations professionnelles, particulièrement essentielle dans le cadre d'une utilisation sur des postes partagés ou des environnements médicaux.



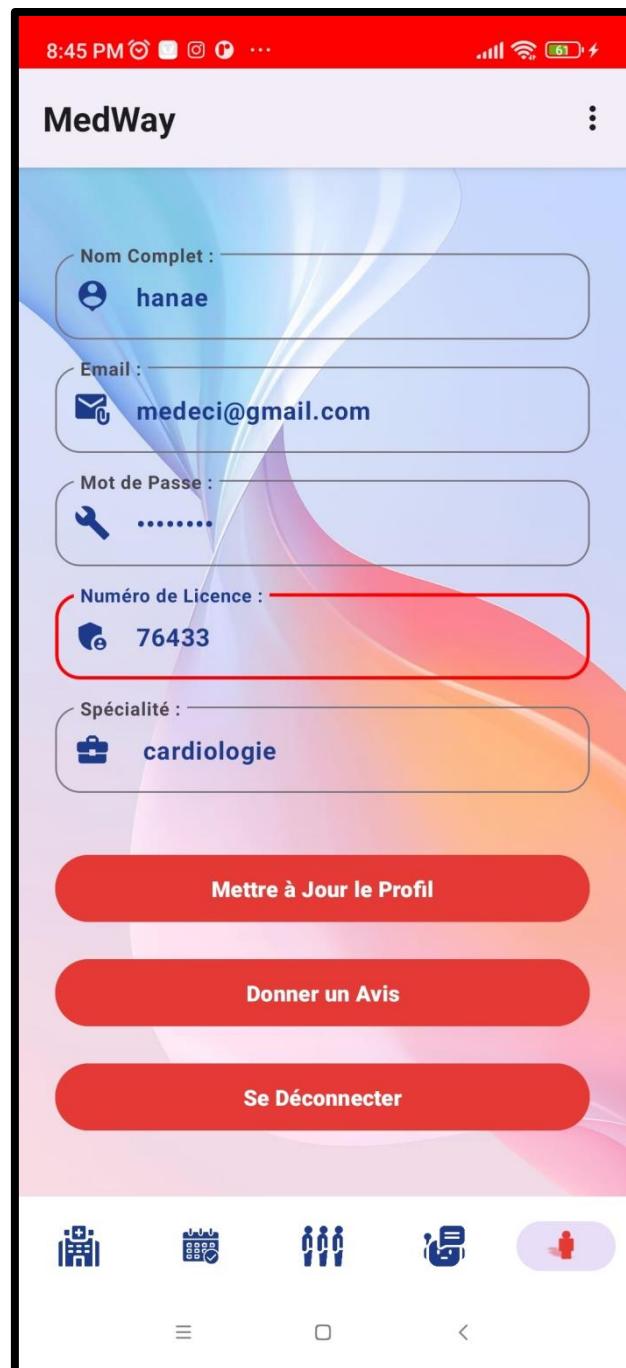


Figure 28 : Gestion du Profil

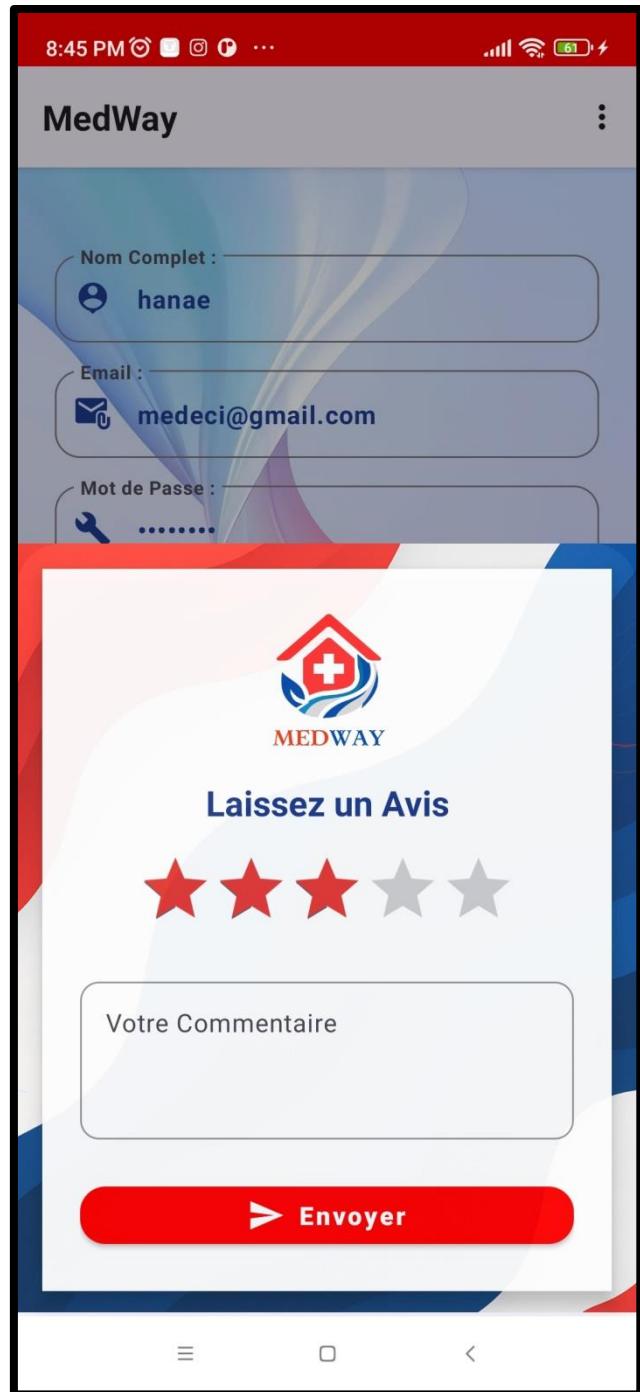


Figure 29 : Avis

VI. Fonctionnalités du patient

1. Recherche d'établissements

L'interface de recherche d'établissements dans **MedWay** permet aux utilisateurs de trouver facilement des structures de santé en saisissant des mots-clés, avec des options de filtrage avancées incluant la distance maximale (en kilomètres) et le

type d'établissement (tous types ou spécifique). Les boutons "Annuler" et "Appliquer" offrent une gestion intuitive des critères de recherche, permettant aux patients d'affiner rapidement leurs résultats pour localiser l'établissement le plus adapté à leurs besoins.

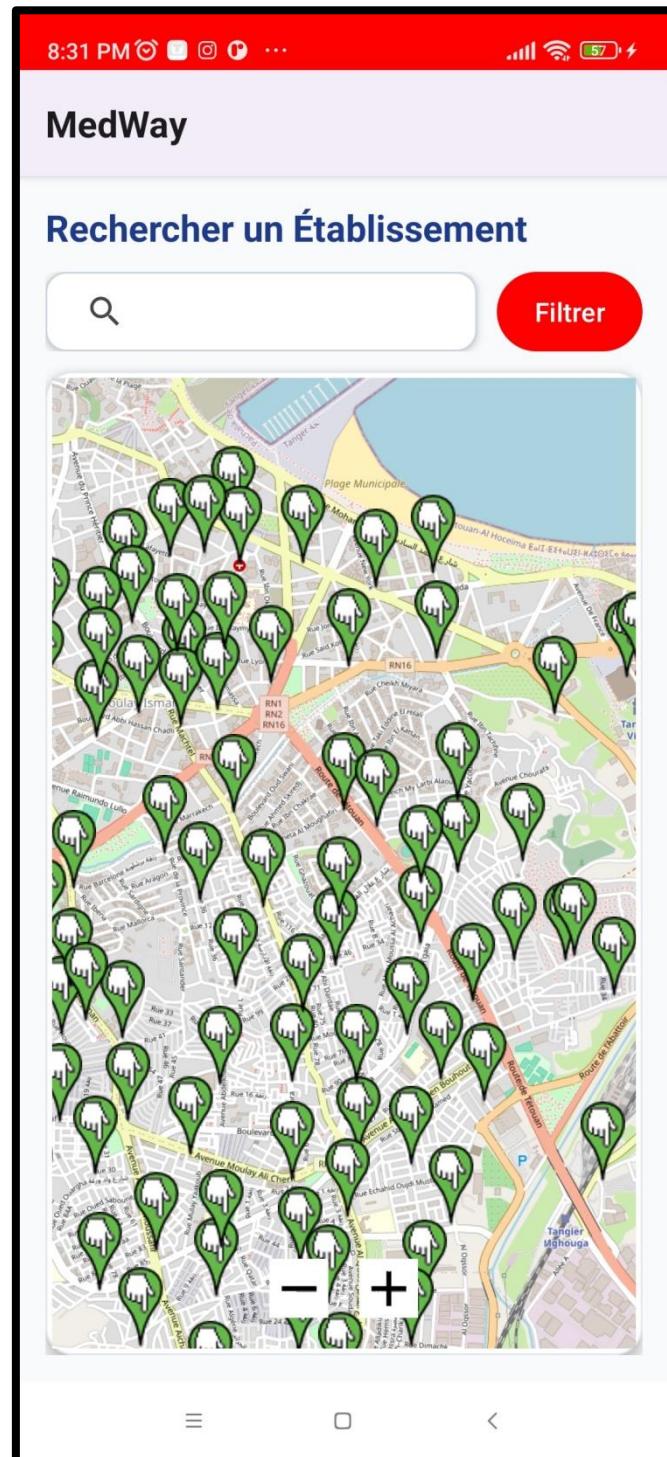


Figure 30 : Recherche d'établissements

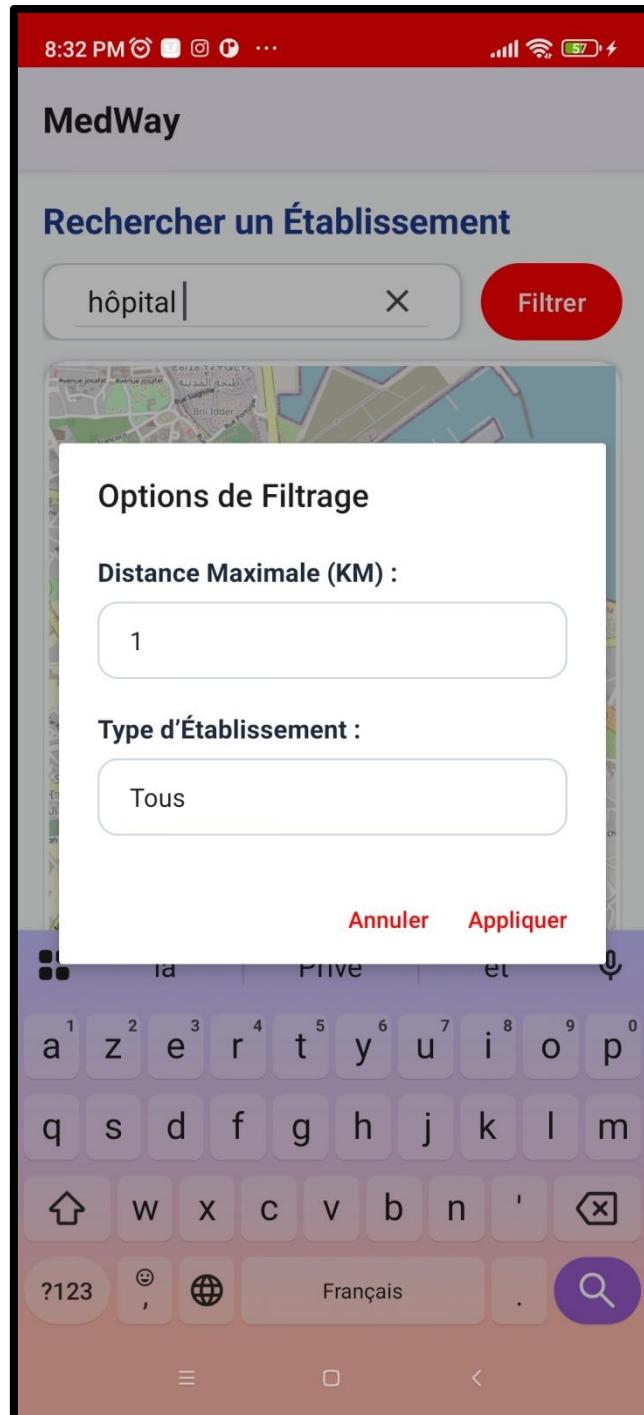
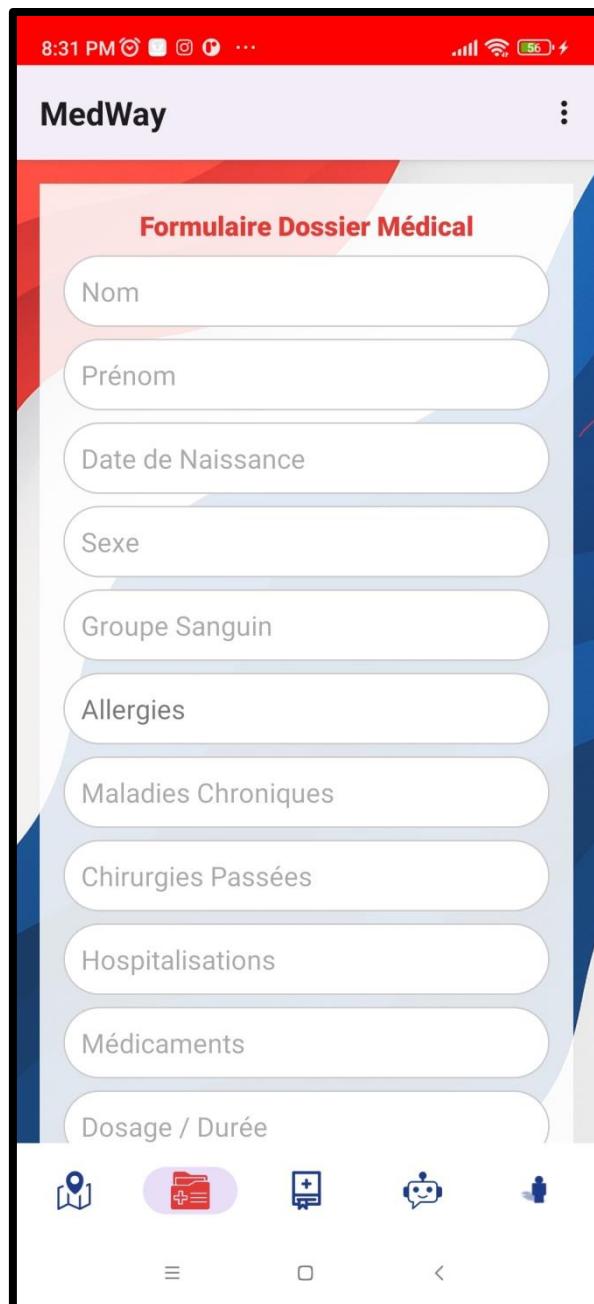


Figure 31 : Option de Filtrage

2. Crédation, modification et suppression du dossier médical

L'interface de gestion des dossiers médicaux dans **MedWay** offre aux patients un processus complet et sécurisé pour créer et administrer leur profil santé. Le parcours commence par un formulaire intuitif permettant de saisir les informations personnelles (nom, prénom, date de naissance), les caractéristiques médicales essentielles (groupe sanguin, allergies) et l'historique santé complet (maladies

chroniques, chirurgies passées, hospitalisations, traitements en cours avec dosage). Après validation via le bouton "Sauvegarder", le système génère une fiche médicale claire et détaillée regroupant l'ensemble des informations. Cette fiche définitive, présentée de manière organisée, intègre deux fonctionnalités clés : un bouton "Modifier" pour mettre à jour les données et un bouton "Supprimer" permettant d'effacer définitivement le dossier.



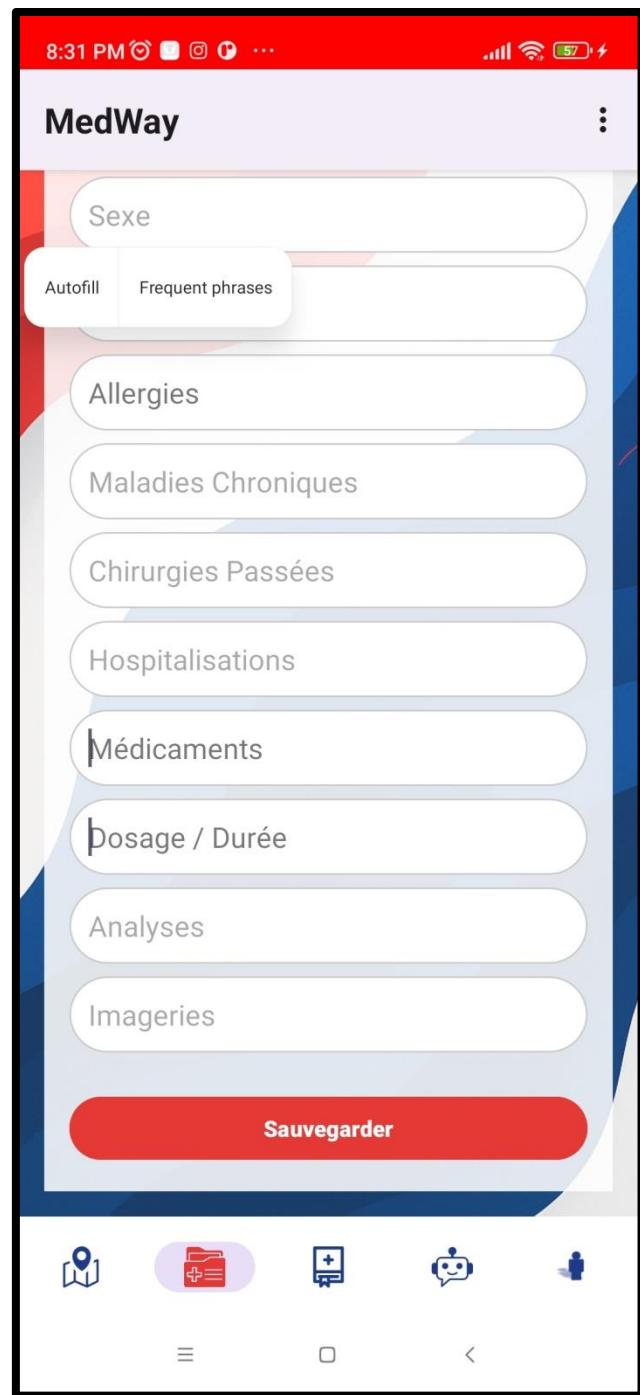


Figure 32 : Crédit, Modification et Suppression du Dossier Médical

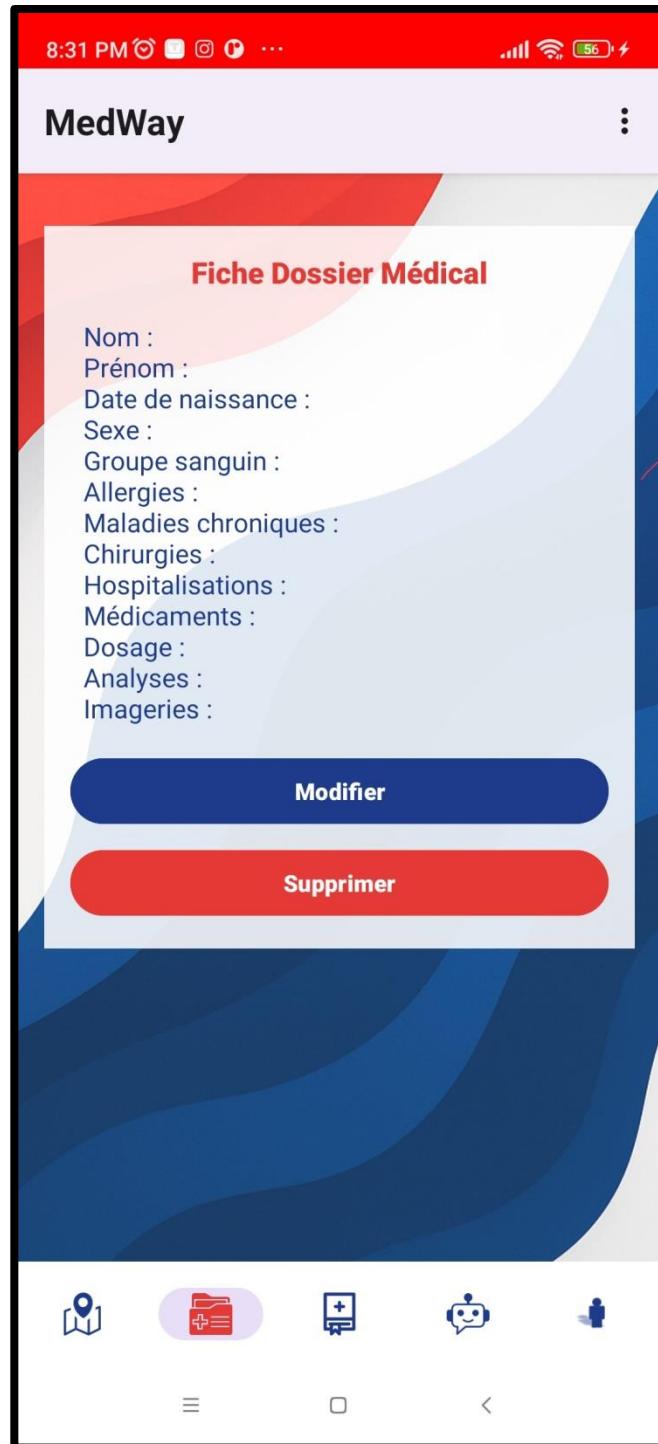


Figure 33 : Fiche Dossier Médical

3. Prise de rendez-vous et consultation de la liste des rendez-vous

L'interface de prise de rendez-vous dans **MedWay** propose une sélection intuitive des créneaux disponibles, affichant clairement les plages horaires accessibles pour chaque établissement. Les patients peuvent filtrer les disponibilités par structure

médicale, ce qui leur permet de visualiser facilement les horaires proposés par chaque établissement partenaire. Une fois le rendez-vous confirmé, il s'ajoute automatiquement à la liste personnelle des consultations programmées, offrant ainsi une vision centralisée des prochains rendez-vous. Cette interface simplifiée permet une prise de rendez-vous rapide et efficace, tout en garantissant une gestion optimale des disponibilités par établissement.

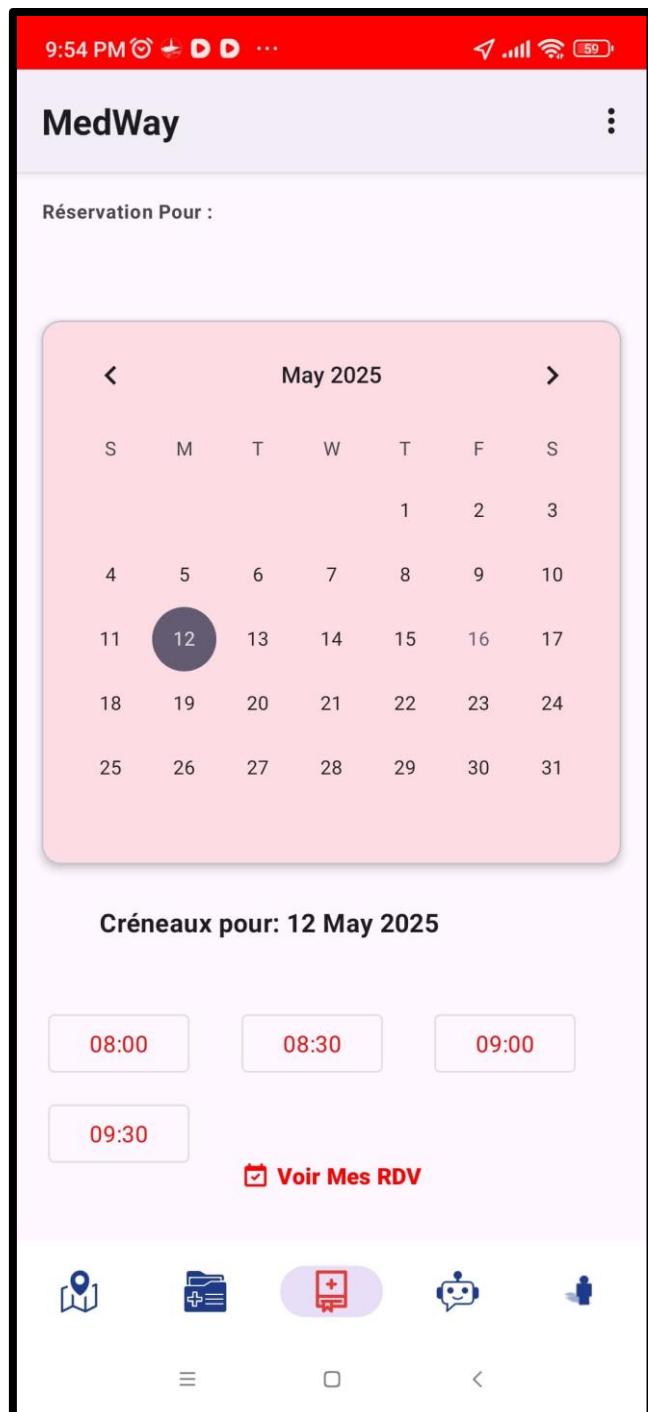


Figure 34 : Prise de Rendez-vous

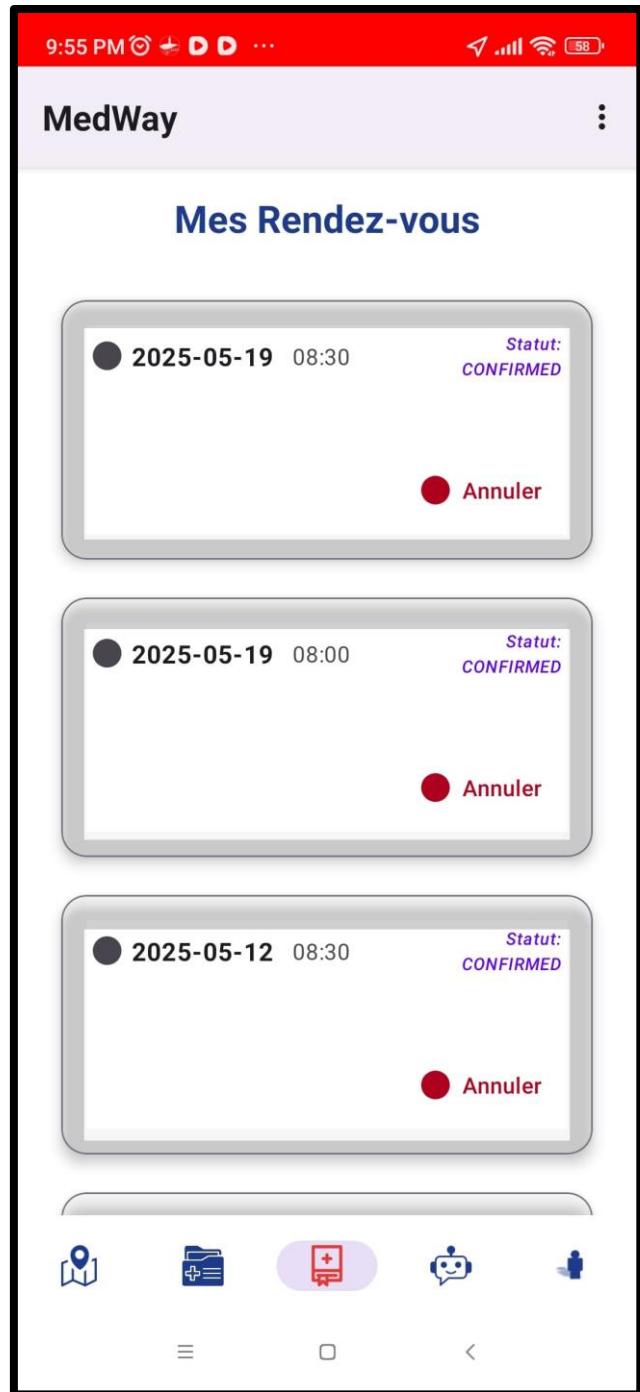


Figure 35 : Liste des Rendez-vous

4. Chatbot intelligent IA

Le chatbot intelligent IA de l'application **MedWay**, destiné aux patients, est un assistant virtuel accessible et convivial, conçu pour accompagner l'utilisateur tout au long de son parcours médical. Dès l'ouverture de l'interface, il engage automatiquement la conversation avec un message d'accueil personnalisé, prêt à fournir de l'aide. Ce chatbot est capable de répondre à une grande variété de demandes, telles que la prise,

l'annulation ou la confirmation de rendez-vous médicaux, la recherche d'un établissement de santé selon différents critères (comme la spécialité ou l'accessibilité PMR), ou encore la gestion du dossier médical (ajout, modification ou suppression). Il peut aussi guider le patient dans les démarches de connexion, la vérification d'identité, ou la réinitialisation de mot de passe.

Grâce à son intelligence artificielle, il offre une assistance rapide, 24h/24, permettant aux patients de trouver facilement des réponses à leurs questions sans avoir besoin d'un contact humain direct. Ce chatbot améliore ainsi l'expérience utilisateur en rendant les services médicaux plus accessibles, fluides et autonomes.

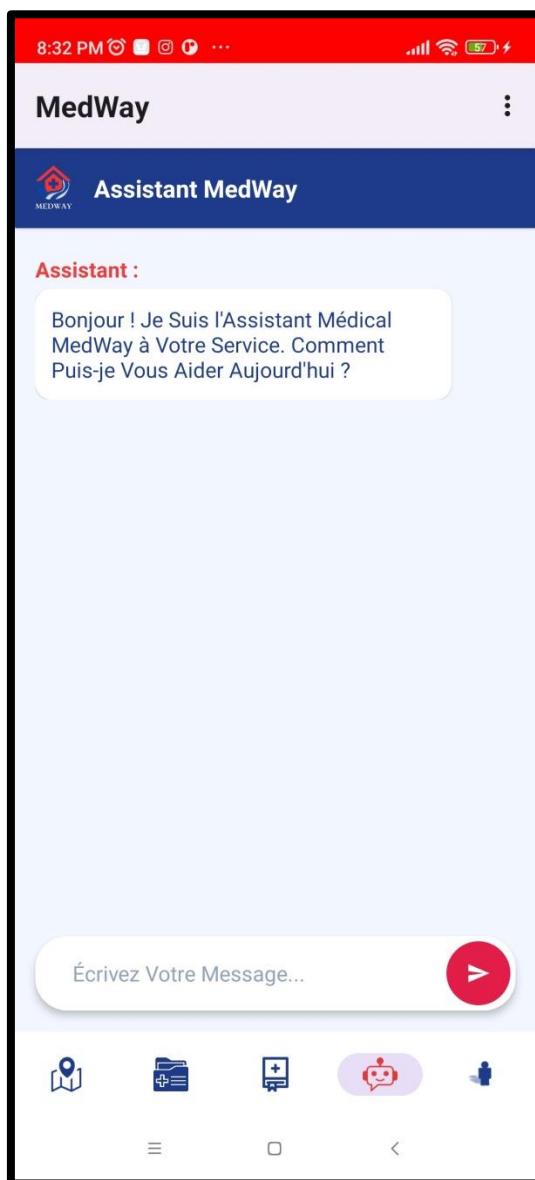


Figure 36 : Chatbot IA

5. Gestion du profil

La gestion du profil dans **MedWay** constitue un élément central de l'expérience utilisateur, offrant aux patients une maîtrise complète de leur compte grâce à des fonctionnalités intuitives et sécurisées. La mise à jour du profil permet aux utilisateurs de modifier aisément leurs informations personnelles, y compris leur identité, leurs coordonnées et leur mot de passe, via une interface claire qui garantit la confidentialité des données. La publication d'avis donne voix aux patients pour partager leurs retours d'expérience, contribuant ainsi à l'amélioration continue des services tout en informant la communauté utilisateur. Enfin, la déconnexion sécurisée protège l'accès au compte en un simple clic, assurant la protection des données sensibles en cas d'utilisation sur des appareils partagés.

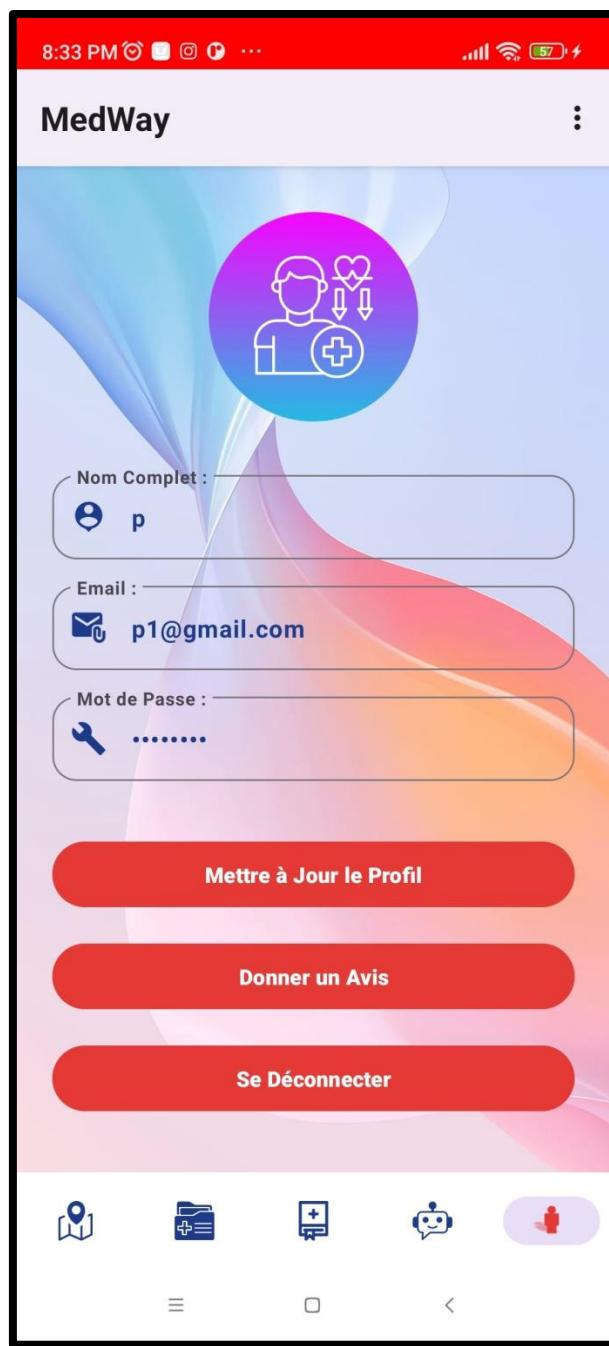


Figure 37 : Gestion du Profil

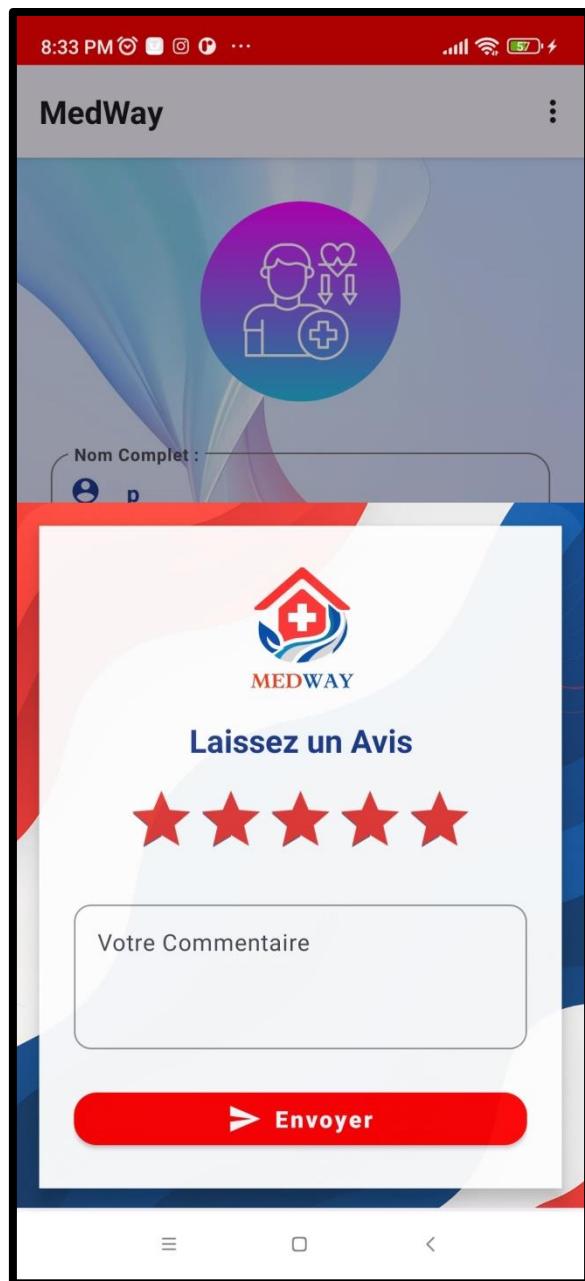


Figure 38 : Avis

V. Conclusion

Les maquettes présentées dans ce chapitre ont été réalisées après le développement de l'application, dans le but de documenter visuellement l'interface utilisateur et d'illustrer le fonctionnement global du système. Elles permettent de mieux comprendre les parcours utilisateurs, les fonctionnalités intégrées ainsi que l'ergonomie proposée.

CHAPITRE VI

PERSPECTIVES D'ÉVOLUTION

I. Introduction

L’application **MedWay** a été conçue dans le but d’optimiser le parcours de soins et de faciliter l’interaction entre patients et professionnels de santé. Elle s’appuie sur une architecture robuste et une logique orientée utilisateur, garantissant une expérience fluide, intuitive et centrée sur les besoins réels du secteur médical.

Toutefois, dans un environnement numérique en constante évolution, il est essentiel d’envisager des pistes d’amélioration et d’innovation continue. Ces perspectives visent à renforcer la qualité du service, améliorer l’accessibilité, répondre aux futurs besoins des utilisateurs, et assurer l’adaptabilité de la solution à long terme, tant sur le plan technique que fonctionnel.

II. Nouvelles perspectives fonctionnelles

L’extension des fonctionnalités constitue un levier essentiel pour améliorer la valeur ajoutée de l’application. De nouveaux services centrés sur la connectivité, l’accessibilité et la personnalisation peuvent significativement renforcer l’expérience utilisateur.

❖ Téléconsultation Intégrée

L’intégration d’un module de visioconsultation sécurisé permettrait aux patients de consulter des professionnels de santé à distance, réduisant ainsi les délais d’accès aux soins. Des créneaux spécifiques pourraient être planifiés directement depuis l’application, avec envoi de notifications et enregistrement de l’historique des consultations. À l’issue de chaque téléconsultation, la transmission numérique des ordonnances offrirait une continuité de soin immédiate.

❖ Portail Web Synchronisé

Le développement d’une version Web de l’application accessible aussi bien aux patients qu’aux professionnels permettrait d’améliorer l’accessibilité du service sur différents supports. Ce portail en ligne offrirait une gestion synchronisée des rendez-vous, des dossiers médicaux et des disponibilités. Une interface ergonomique, adaptée

aux écrans d'ordinateur, faciliterait la navigation et encouragerait l'usage dans les environnements professionnels.

❖ **Intégration avec Objets Connectés (IoT santé)**

L'interopérabilité avec des objets connectés tels que des montres ou capteurs médicaux permettrait de recueillir automatiquement des données physiologiques comme la tension artérielle, la glycémie ou la fréquence cardiaque. Ces données seraient intégrées directement dans le dossier médical du patient, facilitant ainsi un suivi en temps réel et une meilleure réactivité médicale.

❖ **Système de Messagerie Sécurisée**

La création d'un système de messagerie instantanée sécurisé favoriserait la communication directe entre patients et professionnels de santé. Ce canal de communication chiffré permettrait des échanges confidentiels, avec des notifications en temps réel et un suivi structuré des conversations, tout en respectant les normes de sécurité et de confidentialité des données de santé.

III. Améliorations techniques envisagées

Pour garantir la fiabilité, la fluidité et la sécurité de **MedWay** à long terme, des optimisations techniques ciblées doivent être envisagées. Elles permettraient de renforcer les performances du système, d'assurer une meilleure disponibilité dans tous les contextes d'utilisation, et de maintenir un haut niveau de sécurité face aux exigences réglementaires croissantes.

❖ **Support Hors Ligne et Synchronisation Différée**

La mise en place d'un support hors ligne permettrait aux utilisateurs, notamment dans des zones à faible connectivité, d'accéder à leurs dossiers médicaux et à l'historique des consultations sans nécessiter de connexion Internet. Une fois la connexion rétablie, le système procéderait automatiquement à la synchronisation différée des données, garantissant ainsi la cohérence et la mise à jour du contenu entre les différents appareils.

Intelligence Artificielle Avancée

L'intégration de modèles d'intelligence artificielle permettrait de proposer des recommandations personnalisées de professionnels ou d'établissements en fonction du profil et des besoins du patient. Des algorithmes prédictifs pourraient anticiper les créneaux disponibles ou les pics de demande. De plus, un chatbot médical intelligent, entraîné sur des corpus médicaux fiables, renforcerait l'assistance à l'utilisateur en offrant des réponses instantanées, sécurisées et adaptées au contexte de santé.

Optimisation des Performances

Pour offrir une navigation plus fluide et réduire les temps d'attente, l'optimisation des appels Firebase est essentielle. Cela passe par une gestion efficace des requêtes et par la mise en place de solutions de mise en cache locales, telles que DataStore ou SharedPreferences. Ces mécanismes permettent de stocker temporairement certaines données sur l'appareil de l'utilisateur, réduisant ainsi les appels réseau tout en améliorant la réactivité et l'expérience utilisateur.

Sécurité Renforcée

La protection des données de santé étant primordiale, le renforcement de la sécurité passe notamment par l'ajout d'une double authentification (2FA) lors de la connexion. En complément, un système de surveillance des connexions suspectes pourrait être mis en œuvre pour détecter les activités anormales. Un journal des accès sensibles permettrait également d'assurer une traçabilité complète des opérations effectuées sur les informations confidentielles.

IV. Perspectives stratégiques

Au-delà des aspects fonctionnels et techniques, l'évolution de **MedWay** repose également sur une stratégie de déploiement à long terme. Cette stratégie doit prendre en compte l'intégration aux écosystèmes de santé existants, la viabilité économique du projet, ainsi que le respect des normes et certifications en vigueur dans le domaine des données de santé.

¤ Interopérabilité avec les Systèmes de Santé Publique

L'avenir de **MedWay** passe par une intégration fluide avec les infrastructures de santé publique telles que le Dossier Médical Partagé (DMP) ou d'autres plateformes nationales. Cette interopérabilité impliquerait la connexion à des bases de données officielles via des API sécurisées, permettant d'échanger des données médicales de manière standardisée, tout en garantissant la sécurité et la traçabilité des flux d'information. Un tel raccordement offrirait une continuité de soin optimale et renforcerait la légitimité de l'application dans le parcours médical des patients.

¤ Monétisation et Modèle Economique

Pour assurer la pérennité du projet, un modèle économique hybride pourrait être mis en place. Une offre freemium permettrait à tous les utilisateurs d'accéder gratuitement aux fonctionnalités de base, tandis que des services premium — tels que la prise de rendez-vous prioritaire, l'espace de stockage étendu ou l'accès à des spécialistes — seraient proposés sous forme d'abonnements. Par ailleurs, un partenariat avec des établissements de santé, via un abonnement professionnel, pourrait constituer une source de revenus supplémentaire tout en favorisant leur fidélisation.

¤ Certification et Normalisation

Dans le domaine de la e-santé, la conformité réglementaire est un prérequis incontournable. **MedWay** devra s'engager dans une démarche de certification HDS (Hébergement de Données de Santé), garantissant la protection, la confidentialité et la disponibilité des données sensibles. De plus, l'application devra respecter scrupuleusement le Règlement Général sur la Protection des Données (RGPD) ainsi que les normes ISO relatives à la sécurité de l'information, assurant ainsi la crédibilité et la confiance des utilisateurs comme des partenaires institutionnels.

V. Conclusion

Les évolutions envisagées pour **MedWay** ont pour objectif de consolider son rôle central dans l'écosystème de la e-santé, tant sur le plan social que technologique. En enrichissant continuellement ses services, l'application pourra non seulement répondre aux besoins croissants des utilisateurs, mais aussi améliorer la qualité et l'accessibilité

des soins. Elle contribuera ainsi à construire une santé numérique plus inclusive, connectée et résiliente face aux enjeux de demain.

CONCLUSION

CONCLUSION

Le développement de l'application **MedWay** s'inscrit dans une démarche innovante visant à améliorer l'accès aux services de santé grâce aux technologies mobiles. À travers ce projet, nous avons pu concevoir, réaliser et évaluer une solution complète, intelligente et adaptée aux besoins réels des patients comme des professionnels de santé.

De l'analyse des besoins à l'implémentation technique, en passant par la conception fonctionnelle et l'optimisation des performances, chaque étape a été abordée avec rigueur et méthode. L'intégration d'un chatbot intelligent, la gestion avancée des rendez-vous, la personnalisation des espaces professionnels et le respect des bonnes pratiques de sécurité illustrent la richesse fonctionnelle de l'application.

Ce projet a également permis de mettre en œuvre des compétences transversales en développement mobile, en architecture logicielle, en interaction homme-machine, ainsi qu'en manipulation de données issues d'APIs ouvertes. Les choix technologiques ont été orientés vers la fiabilité, la scalabilité et la facilité d'évolution du système.

Bien que de nombreuses fonctionnalités soient déjà opérationnelles, l'application **MedWay** reste ouverte à de futures évolutions. Les perspectives proposées (téléconsultation, interopérabilité avec les systèmes de santé publique, certification HDS...) témoignent de notre volonté de faire de **MedWay** une plateforme durable et capable d'accompagner les transformations numériques du secteur médical.

En définitive, ce projet nous a permis de concrétiser une idée innovante tout en répondant à un enjeu sociétal majeur : simplifier et sécuriser l'accès aux soins à l'ère du numérique.

Webographies

MDN Web Docs, (n.d.). API - Glossaire,
<https://developer.mozilla.org/fr/docs/Glossary/API>

OWASP Foundation, (2023). Authentication Cheat Sheet,
https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

Auth0, (n.d.). Auth0 Documentation, <https://auth0.com/docs/>

Google Firebase, (2024). Cloud Firestore Documentation,
<https://firebase.google.com/docs/firestore>

OpenStreetMap, (n.d.). Accueil OpenStreetMap,
<https://www.openstreetmap.org/>

Overpass Turbo, (n.d.). Interface Overpass Turbo, <https://overpass-turbo.eu/>

OpenStreetMap Wiki, (n.d.). Overpass API Documentation,
https://wiki.openstreetmap.org/wiki/Overpass_API

Android Developers, (2024). Guide d'architecture Android (MVVM),
<https://developer.android.com/jetpack/guide>

Android Developers, (2024). Recyclerview,
<https://developer.android.com/guide/topics/ui/layout/recyclerview>

Glossaire

API (Application Programming Interface) : Interface permettant à des systèmes logiciels de communiquer et d'échanger des données ou des services. Les API sont essentielles pour l'intégration de différentes applications ou services externes. Une API permet d'automatiser des tâches et de récupérer ou envoyer des informations via des requêtes programmatiques.

Authentification : Processus de vérification de l'identité d'un utilisateur avant de lui accorder l'accès à des services sécurisés. Elle repose sur des éléments d'identification tels que les identifiants (nom d'utilisateur, email) et un mot de passe ou d'autres mécanismes comme l'authentification multifactorielle (MFA). Cette étape garantit que l'utilisateur est bien celui qu'il prétend être.

Base de Données : Système structuré permettant de stocker, organiser, et gérer des informations. Les bases de données peuvent être relationnelles ou non relationnelles (NoSQL) et sont utilisées pour des opérations de lecture, d'écriture, et de mise à jour des données. Elles assurent la persistance et l'intégrité des données dans les applications.

Dossier Médical : Ensemble de documents et informations liées à la santé d'un patient, telles que ses antécédents médicaux, traitements, allergies, diagnostics, et résultats de tests. Il peut être créé, mis à jour, consulté ou supprimé selon les droits d'accès et les exigences légales de confidentialité et de sécurité.

Établissement de Santé : Structure organisationnelle qui fournit des services médicaux aux patients. Cela inclut des hôpitaux, des cliniques, des centres de soins spécialisés, des cabinets médicaux, etc. Les établissements de santé sont souvent classifiés en fonction de leur spécialité et de leurs capacités de prise en charge.

File d'Attente : Système de gestion des patients ou utilisateurs en attente d'un service ou d'une consultation. Il sert à organiser l'ordre de passage et à optimiser les flux dans

des contextes tels que les hôpitaux, cliniques, ou tout autre type de service où les utilisateurs doivent patienter.

Filtrage Avancé : Mécanisme de recherche sophistiqué permettant aux utilisateurs de spécifier plusieurs critères pour affiner leurs résultats. Le filtrage avancé peut inclure des critères multiples comme la localisation géographique, la spécialité, la disponibilité, ou encore la proximité avec un lieu spécifique.

Firestore : Base de données NoSQL en temps réel développée par Google. Firestore permet de stocker des données sous forme de documents et de collections, avec une synchronisation instantanée entre les clients et le serveur. C'est une base de données évolutive, flexible et performante, idéale pour les applications mobiles.

Fragment : Composant modulaire de l'interface utilisateur dans les applications Android. Un fragment représente une portion d'interface (partie de l'écran) qui peut être ajoutée ou retirée dynamiquement d'une activité. Cela permet de créer des interfaces plus flexibles et modulaires.

Interface Utilisateur (UI) : L'interface utilisateur est l'ensemble des éléments visuels et interactifs d'une application qui permettent à l'utilisateur de communiquer avec le système. Cela inclut les boutons, les champs de texte, les menus et toute autre composante graphique permettant de naviguer et d'interagir avec l'application.

MVVM (Model-View-ViewModel) : Modèle d'architecture logicielle qui sépare clairement la logique de l'application en trois parties :

- ❖ **Model** : La gestion des données et des interactions avec la base de données.
- ❖ **View** : L'interface graphique avec laquelle l'utilisateur interagit.
- ❖ **ViewModel** : La couche qui lie le Model à la View, en fournissant des données sous forme prête à être affichée. Cette architecture permet une séparation des responsabilités et une meilleure maintenabilité du code.

Notification : Mécanisme permettant d'informer un utilisateur d'un événement ou d'une mise à jour. Les notifications peuvent être push ou locales et sont souvent utilisées pour rappeler des événements importants comme des rendez-vous ou des alertes urgentes. Elles sont essentielles pour maintenir l'engagement des utilisateurs.

OpenStreetMap : Projet collaboratif visant à créer une carte libre et ouverte du monde. Les données géographiques, comme les routes, les bâtiments, et autres points d'intérêt, sont collectées et mises à jour par la communauté. OpenStreetMap est souvent utilisé dans des applications nécessitant des informations géospatiales.

Overpass API : API permettant d'interroger des bases de données OpenStreetMap pour extraire des informations géographiques spécifiques. L'Overpass API permet de filtrer les données par type d'objet, géolocalisation, ou autre attribut. Elle est particulièrement utile pour obtenir des données précises sur les lieux d'intérêts, tels que les établissements de santé.

Profil Utilisateur : Ensemble des informations personnelles associées à un utilisateur, telles que son nom, adresse, email, rôle, et préférences. Le profil utilisateur est souvent utilisé pour personnaliser l'expérience et adapter les services proposés selon les caractéristiques et les besoins de l'utilisateur.

RecyclerView : Composant d'interface Android permettant de gérer et d'afficher des listes dynamiques et longues de données. Le RecyclerView est optimisé pour les performances en recyclant les vues et en minimisant la consommation de mémoire. Il est souvent utilisé pour afficher des éléments tels que des listes d'utilisateurs, d'établissements, ou de rendez-vous.

Rendez-Vous : Moment programmé pour qu'un utilisateur consulte un service ou un professionnel de santé. Les rendez-vous peuvent être pris, modifiés, ou annulés en fonction de la disponibilité et des préférences des utilisateurs. Les systèmes de gestion des rendez-vous incluent souvent des rappels et des notifications.

RGPD (Règlement Général sur la Protection des Données) : Règlement de l'Union Européenne visant à protéger les données personnelles des individus. Le RGPD impose des exigences strictes concernant la collecte, le traitement, la conservation et la sécurité des données personnelles. Il est essentiel pour garantir la confidentialité et la conformité légale des applications manipulant des données sensibles.

Sécurité des Données : Ensemble de mesures et de pratiques permettant de protéger les informations contre la divulgation non autorisée, la perte, ou la modification. La sécurité des données repose sur des stratégies de chiffrement, de contrôle d'accès, de

sauvegarde et de détection des intrusions pour assurer l'intégrité et la confidentialité des données.

Spécialité Médicale : Domaine spécifique de la médecine dans lequel un professionnel de santé a acquis une expertise. Les spécialités incluent des domaines tels que la cardiologie, la dermatologie, l'ophtalmologie, etc. Les utilisateurs peuvent filtrer et rechercher des établissements en fonction de la spécialité des professionnels qu'ils recherchent.

Synchronisation en Temps Réel : Processus permettant de garantir que toutes les données sont actualisées instantanément sur tous les appareils connectés à un système. La synchronisation en temps réel est cruciale pour des applications où la précision et la mise à jour rapide des informations sont essentielles, comme dans le cas de la gestion des rendez-vous ou des fichiers médicaux.

Annexes

Annexe 1 : AndroidManifest.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools">
4          <uses-permission android:name="android.permission.INTERNET" />
5          <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
6          <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
7
8          <application
9              android:allowBackup="true"
10             android:dataExtractionRules="@xml/data_extraction_rules"
11             android:fullBackupContent="@xml/backup_rules"
12             android:icon="@mipmap/ic_launcher"
13             android:label="MedWay"
14             android:roundIcon="@mipmap/ic_launcher_round"
15             android:supportsRtl="true"
16             android:theme="@style/Theme.medcare"
17             tools:targetApi="31"
18             android:networkSecurityConfig="@xml/network_security_config">
19
20             <activity
21                 android:name=".MainActivity"
22                 android:exported="true">
23                 <intent-filter>
24                     <action android:name="android.intent.action.MAIN" />
25                     <category android:name="android.intent.category.LAUNCHER" />
26                 </intent-filter>
27             </activity>
28
29             <activity android:name=".auth.SignInActivity" android:exported="false" />
30             <activity android:name=".auth.SignUpProfessionalActivity" android:exported="false" />
31             <activity android:name=".admin.PendingVerificationActivity" android:exported="false" />
32             <activity android:name=".admin.AdminDashboardActivity" android:exported="false" android:theme="@style/Theme.medcare.NoActionBar" />
33             <activity android:name=".auth.SignUpPatientActivity" android:exported="false" />
34             <activity android:name=".map.MapActivity" android:exported="false" />
35
36             ! </application>
37         </manifest>
```

Annexe 2 : MainActivity.java

```
1 package com.example.medcare;
2
3 > import ...
40
41 </> public class MainActivity extends AppCompatActivity {
42
43     14 usages
44     private static final String TAG = "MainActivity";
45     8 usages
46     private FirebaseAuth mAuth;
47     2 usages
48     private FirebaseFirestore db;
49
50
51     13 usages
52     private BottomNavigationView bottomNavigationView;
53     3 usages
54     private FrameLayout fragmentContainer;
55     9 usages
56     private String currentUserRole = null;
57
58
59
60
61
62
63     @Override
64     protected void onCreate(Bundle savedInstanceState) {
65         super.onCreate(savedInstanceState);
66         setContentView(R.layout.activity_main);
67
68         mAuth = FirebaseAuth.getInstance();
69         db = FirebaseFirestore.getInstance();
70
71         bottomNavigationView = findViewById(R.id.bottom_navigation);
72         fragmentContainer = findViewById(R.id.fragment_container);
73
74         if (bottomNavigationView == null || fragmentContainer == null) {
75             Log.e(TAG, msg: "FATAL: Core UI elements not found!");
76             handleFatalError();
77             return;
78         }
79     }
80 }
```

```

70
71     @Override
72     protected void onStart() {
73         super.onStart();
74         FirebaseUser currentUser = mAuth.getCurrentUser();
75         if (currentUser == null) {
76             redirectToSignIn();
77         } else {
78             verifyUserAccessAndSetupUI(currentUser.getUid());
79         }
80     }
81
82     4 usages
83     private void redirectToSignIn() {
84         Intent intent = new Intent(packageContext: MainActivity.this, SignInActivity.class);
85         intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
86         startActivity(intent);
87         finish();
88     }
89
90     1 usage
91     private void handleFatalError() {
92         Toast.makeText(context: this, text: "Erreur critique de l'interface.", Toast.LENGTH_LONG).show();
93         if (mAuth != null && mAuth.getCurrentUser() != null) mAuth.signOut();
94         redirectToSignIn();
95     }
96
97     1 usage
98     @SuppressLint("SetTextI18n")
99     private void verifyUserAccessAndSetupUI(String userId) {
100        DocumentReference userRef = db.collection(collectionPath: "users").document(userId);
101        userRef.get().addOnCompleteListener(task -> {
102            if (task.isSuccessful() && task.getResult() != null && task.getResult().exists()) {
103                DocumentSnapshot document = task.getResult();
104                String userRole = document.getString(field: "role");
105                String userStatus = document.getString(field: "status");
106                this.currentUserRole = userRole;
107
108                boolean showMainUI = false;
109                Intent redirectIntent = null;
110
111                Log.d(TAG, msg: "Role check: userRole = " + userRole);
112
113                if ("admin".equals(userRole)) {
114                    redirectIntent = new Intent(packageContext: MainActivity.this, AdminDashboardActivity.class);
115                } else if ("professional".equals(userRole) && "approved".equals(userStatus)) {
116                    showMainUI = true;
117                } else if ("pending_professional".equals(userRole)) {
118                    if ("pending".equals(userStatus)) {
119                        redirectIntent = new Intent(packageContext: MainActivity.this, PendingVerificationActivity.class);
120                    } else {
121                        handleInvalidAccess("rejected".equals(userStatus) ? "Your professional application was rejected." : "Account status error.");
122                        return;
123                    }
124                } else if ("patient".equals(userRole)) {
125                    showMainUI = true;
126                }
127
128                if (showMainUI) {
129                    Log.d(TAG, msg: "Setting up Main UI for role: " + this.currentUserRole);
130                    setupBottomNavigationForRole();
131                } else if (redirectIntent != null) {
132                    Log.d(TAG, msg: "Redirecting to: " + redirectIntent.getComponent().getShortClassName());
133                    redirectIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
134                    startActivity(redirectIntent);
135                    finish();
136                } else {
137                    handleInvalidAccess(toastMessage: "Account access error or unhandled state.");
138                }
139            } else {
140                Log.e(TAG, msg: "Firestore profile check failed.", task.getException());
141                handleInvalidAccess(toastMessage: "Error loading profile. Please log in again.");
142            }
143        }
144    }

```

```

105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139

```

```
140     });
141 }
142
143     3 usages
144     private void handleInvalidAccess(String toastMessage){
145         Toast.makeText( context: this, toastMessage, Toast.LENGTH_LONG).show();
146         if (mAuth != null) mAuth.signOut();
147         redirectSignIn();
148     }
149
150     1 usage
151     @SuppressLint(" ResourceType") |
152     private void setupBottomNavigationForRole() {
153         if (bottomNavigationView == null || currentUserRole == null) return;
154
155         bottomNavigationView.getMenu().clear();
156         int menuResId;
157         int initialFragmentId;
158
159         Log.d(TAG, msg: "Inflating menu for role: " + currentUserRole);
160
161         try {
162             if ("professional".equals(currentUserRole)) {
163                 menuResId = R.menu.bottom_menu;
164                 initialFragmentId = R.id.nav_etablissement;
165                 bottomNavigationView.inflateMenu(menuResId);
166                 setupNavigationListenerProfessional();
167             } else if ("patient".equals(currentUserRole)) {
168                 menuResId = R.menu.bottom_nav_menu_patient;
169                 initialFragmentId = R.id.nav_book_appointment;
170                 bottomNavigationView.inflateMenu(menuResId);
171                 setupNavigationListenerPatient();
172             } else {
173                 Log.e(TAG, msg: "Unsupported role for bottom nav: " + currentUserRole);
174                 bottomNavigationView.setVisibility(View.GONE);
175             }
176         }
177     }
178 }
```

```

169         setupNavigationListenerPatient();
170     } else {
171         Log.e(TAG, msg: "Unsupported role for bottom nav: " + currentUserRole);
172         bottomNavigationView.setVisibility(View.GONE);
173         return;
174     }
175 } catch (Exception e) {
176     Log.e(TAG, msg: "Error inflating menu resource!", e);
177     Toast.makeText(context: this, text: "Erreur de chargement du menu.", Toast.LENGTH_LONG).show();
178     bottomNavigationView.setVisibility(View.GONE);
179     replaceFragment(new Fragment());
180     return;
181 }
182
183 bottomNavigationView.setVisibility(View.VISIBLE);
184
185 if (getSupportFragmentManager().findFragmentById(R.id.fragment_container) == null) {
186     Log.d(TAG, msg: "Setting initial fragment ID: " + initialFragmentId);
187     if (bottomNavigationView.getMenu().findItem(initialFragmentId) != null) {
188         bottomNavigationView.setSelectedItemId(initialFragmentId);
189     } else {
190         Log.e(TAG, msg: "Default menu item ID " + initialFragmentId + " not found in inflated menu!");
191         loadFallbackFragment();
192     }
193 }
194 }
195
196 /**
197 * usage
198 */
199 private void setupNavigationListenerProfessional() {
200     bottomNavigationView.setOnItemSelectedListener(item -> {
201         int id = item.getItemId();
202         Fragment selectedFragment = null;
203         if (id == R.id.nav_etablissement) selectedFragment = new EtablissementFragment();
204         else if (id == R.id.nav_disponibilite) selectedFragment = new DisponibiliteFragment();
205         else if (id == R.id.nav_file) selectedFragment = new FileAttenteFragment();
206         else if (id == R.id.nav_chatbot) selectedFragment = new ChatBotFragment();

```

```
204     else if (id == R.id.nav_profile) selectedFragment = new ProfilFragment();
205
206
207     // Add more items...
208     if (selectedFragment != null) replaceFragment(selectedFragment);
209     return selectedFragment != null;
210   });
211 }
212
213
214 1 usage
215 private void setupNavigationListenerPatient() {
216   bottomNavigationView.setOnItemSelectedListener(item -> {
217     int id = item.getItemId();
218     Fragment selectedFragment = null;
219
220     if (id == R.id.nav_book_appointment) {
221       selectedFragment = new PatientEstablishmentListFragment();
222     } else if (id == R.id.nav_my_appointments) {
223       selectedFragment = new PatientAppointmentsFragment();
224     } else if (id == R.id.nav_chatbot) {
225       selectedFragment = new ChatbotFragment2();
226     } else if (id == R.id.nav_medical_dossier) {
227       selectedFragment = new MedicalRecordFragment();
228     } else if (id == R.id.nav_map) {
229
230       Intent intent = new Intent( packageContext: MainActivity.this, MapActivity.class);
231       startActivity(intent);
232       return true;
233     }
234
235     if (selectedFragment != null) {
236       replaceFragment(selectedFragment);
237       return true;
238     }
239   }
240 }
```

```

239         return false;
240     });
241 }
242
243 1 usage
244
245     private void loadFallbackFragment() {
246         Log.w(TAG, msg: "Loading fallback fragment due to missing initial item ID.");
247         Fragment fallbackFragment;
248         if ("patient".equals(currentUserRole)) {
249             fallbackFragment = new PatientEstablishmentListFragment();
250         } else if ("professional".equals(currentUserRole)) {
251             fallbackFragment = new EtablissementFragment();
252         } else {
253             fallbackFragment = new Fragment();
254         }
255         replaceFragment(fallbackFragment);
256     }
257
258 4 usages
259
260     private void replaceFragment(Fragment fragment) {
261         if (fragmentContainer == null || fragment == null) return;
262         FragmentManager fragmentManager = getSupportFragmentManager();
263         Fragment currentFragment = fragmentManager.findFragmentById(R.id.fragment_container);
264         if (currentFragment != null & currentFragment.getClass().equals(fragment.getClass())) {
265             Log.d(TAG, msg: "Fragment already displayed: " + fragment.getClass().getSimpleName());
266             return;
267         }
268         FragmentTransaction transaction = fragmentManager.beginTransaction();
269         transaction.replace(R.id.fragment_container, fragment, fragment.getClass().getSimpleName());
270         transaction.commitAllowingStateLoss();
271         Log.d(TAG, msg: "Replacing fragment with: " + fragment.getClass().getSimpleName());
272     }

```

```

273
274
275
276
277     @Override
278     public boolean onCreateOptionsMenu(Menu menu) {
279         getMenuInflater().inflate(R.menu.main_options_menu, menu);
280         return true;
281     }
282
283
284     @Override
285     public boolean onOptionsItemSelected(@NonNull MenuItem item) {
286         final int itemId = item.getItemId();
287         if (itemId == R.id.action_logout) {
288             Log.d(TAG, msg: "Logout selected.");
289             mAuth.signOut();
290             redirectToSignIn();
291             return true;
292         }
293         return super.onOptionsItemSelected(item);
294     }
295

```

Annexe 3 : activity_sign_up.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:padding="24dp"
8      tools:context=".auth.SignInActivity">
9
10
11     <ImageView
12         android:id="@+id/backArrowSignIn"
13         android:layout_width="wrap_content"
14         android:layout_height="wrap_content"
15         android:src="@drawable/ic_arrow_back"
16         app:layout_constraintStart_toStartOf="parent"
17         app:layout_constraintTop_toTopOf="parent"
18         android:contentDescription="Back" />
19
20     <TextView
21         android:id="@+id/textViewSignInTitle"
22         android:layout_width="wrap_content"
23         android:layout_height="wrap_content"
24         android:text="Sign In"
25         android:textSize="24sp"
26         android:textStyle="bold"
27         app:layout_constraintTop_toBottomOf="@+id/backArrowSignIn"
28         app:layout_constraintStart_toStartOf="parent"
29         app:layout_constraintEnd_toEndOf="parent"
30         android:layout_marginTop="16dp" />
31
32     <com.google.android.material.textfield.TextInputLayout
33         android:id="@+id/textFieldEmailSignIn"
34         style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
35         android:layout_width="0dp"
36         android:layout_height="wrap_content"
37         android:layout_marginTop="32dp"
38         android:hint="Enter your email"
39         app:startIconDrawable="@drawable/ic_email"
40         app:layout_constraintTop_toBottomOf="@+id/textViewSignInTitle"
41         app:layout_constraintStart_toStartOf="parent"
42         app:layout_constraintEnd_toEndOf="parent">
43
44         <com.google.android.material.textfield.TextInputEditText
45             android:id="@+id/editTextEmailSignIn"
46             android:layout_width="match_parent"
47             android:layout_height="wrap_content"
48             android:inputType="textEmailAddress" />
49     </com.google.android.material.textfield.TextInputLayout>
50
51     <com.google.android.material.textfield.TextInputLayout
52         android:id="@+id/textFieldPasswordSignIn"
53         style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
54         android:layout_width="0dp"
55         android:layout_height="wrap_content"
56         android:layout_marginTop="16dp"
57         android:hint="Enter your password"
58         app:startIconDrawable="@drawable/ic_lock"
59         app:endIconMode="password_toggle"
60         app:layout_constraintTop_toBottomOf="@+id/textFieldEmailSignIn"
61         app:layout_constraintStart_toStartOf="parent"
62         app:layout_constraintEnd_toEndOf="parent">
```

```
63    <com.google.android.material.textfield.TextInputEditText  
64        android:id="@+id/editTextPasswordSignIn"  
65        android:layout_width="match_parent"  
66        android:layout_height="wrap_content"  
67        android:inputType="textPassword" />  
68    </com.google.android.material.textfield.TextInputLayout>  
69  
70    <TextView  
71        android:id="@+id/textViewForgotPassword"  
72        android:layout_width="wrap_content"  
73        android:layout_height="wrap_content"  
74        android:text="Forgot password?"  
75        android:layout_marginTop="8dp"  
76        android:textColor="@color/design_default_color_primary"  
77        app:layout_constraintTop_toBottomOf="@+id/editTextPasswordSignIn"  
78        app:layout_constraintEnd_toEndOf="parent"/>  
79  
80    <ProgressBar  
81        android:id="@+id/progressBarSignIn"  
82        style="?android:attr/progressBarStyle"  
83        android:layout_width="wrap_content"  
84        android:layout_height="wrap_content"  
85        android:visibility="gone"  
86        app:layout_constraintBottom_toBottomOf="@+id/buttonSignIn"  
87        app:layout_constraintEnd_toEndOf="@+id/buttonSignIn"  
88        app:layout_constraintStart_toStartOf="@+id/buttonSignIn"  
89        app:layout_constraintTop_toTopOf="@+id/buttonSignIn" />  
90
```

```
91     <com.google.android.material.button.MaterialButton  
92         android:id="@+id/buttonSignIn"  
93         android:layout_width="0dp"  
94         android:layout_height="wrap_content"  
95         android:layout_marginTop="24dp"  
96         android:text="Sign In"  
97         android:paddingTop="12dp"  
98         android:paddingBottom="12dp"  
99         app:cornerRadius="24dp"  
100         app:layout_constraintTop_toBottomOf="@+id/textViewForgotPassword"  
101         app:layout_constraintStart_toStartOf="parent"  
102         app:layout_constraintEnd_toEndOf="parent" />  
103  
104     <LinearLayout  
105         android:id="@+id/layoutSignUpOptions"  
106         android:layout_width="wrap_content"  
107         android:layout_height="wrap_content"  
108         android:orientation="vertical"  
109         android:gravity="center_horizontal"  
110         android:layout_marginTop="24dp"  
111         app:layout_constraintTop_toBottomOf="@+id/buttonSignIn"  
112         app:layout_constraintStart_toStartOf="parent"  
113         app:layout_constraintEnd_toEndOf="parent">  
114  
115     <TextView  
116         android:layout_width="wrap_content"  
117         android:layout_height="wrap_content"  
118         android:text="Need an account?" /> <!-- Optional prompt -->  
119  
120     <TextView  
121         android:id="@+id/textViewSignUpPatientLink"  
122         android:layout_width="wrap_content"  
123         android:layout_height="wrap_content"  
124         android:layout_marginTop="8dp"  
125         android:text="Sign up as Patient"  
126         android:textColor="@color/design_default_color_primary"  
127         android:minHeight="48dp"  
128         android:gravity="center" /> <!-- Min touch target size -->  
129  
130  
131     <TextView  
132         android:id="@+id/textViewSignUpProfessionalLink"  
133         android:layout_width="wrap_content"  
134         android:layout_height="wrap_content"  
135         android:layout_marginTop="8dp"  
136         android:text="Sign up as Professional"  
137         android:textColor="@color/design_default_color_primary"  
138         android:minHeight="48dp"  
139         android:gravity="center" /> <!-- Min touch target size -->  
140  
141     </LinearLayout>  
142 </androidx.constraintlayout.widget.ConstraintLayout>
```

Annexe 4 : Génération de Rapports Statistiques

