République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Ain Temouchent Belhadj Bouchaib

Faculté des sciences et technologies
Département des Mathématiques et de l'Informatique

Mémoire

En vue de l'obtention du Diplôme de Master en Informatique

Option :

Réseaux et Ingénierie des Données (RID)

Présenté par :
Mr.ZIANE CHERIF Ayoub

Mr.HADDOU Ali Badr

Système de gestion intelligente et de géolocalisation en temps réel pour les services de protection civile.

Encadrant :
Dr BENDIABDALLAH Mohammed Hakim
Maitre de conférence "B" à U.A.T.B.B.
Soutenu le : 13/07/2021

Devant le jury composé de :

Président :	*M^r BOUCHAKOUR ERRAHMANI* Hichem (M.C.B)	U.A.T.B.B.
Examinateurs :	*M^r BENOMAR* Mohammed Lamine (M.C.B)	U.A.T.B.B.

Année universitaire 2020/2021

Dédicace

Nous dédions ce modeste mémoire à :

Nos chers parents :

Qu'aucune dédicace ne puisse exprimer ce que nous leur devons, pour tous leurs sacrifices, leur gentillesse, leur amour, leur tendresse, leur soutien et leurs prières tout au long de nos études. Que ce travail soit le témoignage de notre profond amour et de notre gratitude "Que Dieu vous garde".

Nos chères soeurs et nos chers frères :

Pour leurs constants encouragements et leur soutien moral, nous leur dédions ce modeste ouvrage en témoignage de notre grand amour et de notre infinie gratitude.

Toutes nos Familles :

À toutes les familles ZIANE CHERIF & MEKRI et HADDOU & BEKRITI, pour leur soutien tout au long de notre parcours universitaire, Que ce travail soit l'accomplissement de vos souhaits tant allégués, et la fuite de votre soutien indéfectible.

Tous nos amis :

Pour leur aide et leur soutien moral durant l'élaboration du travail de fin d'études.

Remerciement

Avant tout, nous remercions le bon DIEU de nous avoir aidés à accomplir ce modeste travail.

Nous profitons de cette occasion pour exprimer nos plus sincères remerciements et notre reconnaissance à :

Notre encadreur : Mr BENDIABDALLAH Mohammed Hakim pour sa disponibilité, ses conseils, son orientation et ses encouragements tout au long de notre recherche.

Membres du jury Mr BOUCHAKOUR ERRAHMANI Hichem et Mr BENOMAR Mohammed Lamine qui ont accepté d'évaluer et d'examiner ce travail.

Nos familles et amis qui, par leurs prières et leurs encouragements, ont pu surmonter tous les obstacles.

Enfin, nous ne saurions terminer ces remerciements sans associer toutes les personnes qui ont participé à l'exécution de ce modeste travail.

Résumé

La protection civile a plusieurs missions, dans notre travail nous nous sommes intéressé à la mission de la protection des personnes et les secours d'urgence, qui représentent la grande majorité des interventions de la protection civile.

Après l'émergence des technologies mobiles beaucoup d'applications ont été développées dans divers domaines. Cependant, nous avons proposé une plateforme en ligne et en temps réel qui améliore la communication entre les services de protection civile et les hôpitaux afin d'éviter les complications de santé des malades voire même les pertes humaines.

Cette plateforme n'est pas bénéfique seulement pour les patients mais aussi pour le personnel médical et les ambulanciers. Il s'agit de faire une gestion globale du service réception et orientation de l'hôpital ainsi que de toutes ces ressources afin de faciliter la tâche du personnel médical ainsi que les ambulanciers.

Mots clés : Android, Java, Firebase, Real-time Database, Firebase Authentication, Géolocalisation, Fused Location Provider, Intelligence artificielle, Protection civile.

Abstract

Civil protection has several missions, in our work we are interested in the mission of protection of people and emergency relief, which represent the vast majority of interventions of civil protection.

After the emergence of mobile technologies, many applications have been developed in various fields. However, we have proposed an online and real-time platform that improves communication between civil protection services and hospitals in order to avoid health complications of patients and even human losses.

This platform is not only beneficial for patients, but also for medical staff and paramedics. It is about making a global management of the hospital as well as all its resources in order to facilitate the task of the medical staff as well as the ambulance drivers.

Keywords : Android, Java, Firebase, Real-time Database, Firebase Authentication, Geolocation, Fused Location Provider, Artificial Intelligence, Civil protection.

ملخص

الحماية المدنية لها عدة مهام ، في عملنا نحن مهتمون بمهمة حماية الناس والإغاثة في حالات الطوارئ ، والتي تمثل الغالبية العظمى من تدخلات الحماية المدنية.

بعد ظهور تقنيات الهاتف المحمول تم تطوير العديد من التطبيقات في مختلف المجالات. لذلك ، اقترحنا منصة عبر الإنترنت وفي الوقت الاني تعمل على تحسين الاتصال بين خدمات الحماية المدنية والمستشفيات من أجل تجنب المضاعفات الصحية للمرضى وحتى الخسائر البشرية.

هذه المنصة ليست مفيدة فقط للمرضى ولكن أيضا للموظفين الطبيين والمسعفين. يتعلق الأمر بإجراء إدارة عامة للمستشفى بالإضافة إلى جميع موارده من أجل تسهيل مهمة الطاقم الطبي وكذلك المسعفين.

كلمات مفتاحية: أندرويد، جافا، فايربيس، قاعدة بيانات في الوقت الفعلي، فايربيس مصادقة، تحديد الموقع الجغرافي، مزود الموقع المدمج، الذكاء الاصطناعي، حماية مدنية.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Avec l'ère technologique actuelle où tout fonctionne principalement sur les smartphones et les applications numériques, le besoin de services rapides et efficaces est important dans tous les aspects, en particulier en ce qui concerne les services médicaux.

Les défis que l'ambulancier rencontre quotidiennement sont de trouver le moyen de vérifier la disponibilité des services requis de l'hôpital, ainsi que le chemin le plus optimal en matière de temps et de fluidité vers cet hôpital, surtout dans les cas d'urgence . Dans ces cas, le temps et la disponibilité des informations en temps réel peuvent sauver plusieurs vies humaines.

L'objectif principal de ce travail est de réduire le temps de transport du patient vers l'hôpital optimal en fonction de plusieurs facteurs tels que la distance entre l'hôpital et l'ambulance et sa saturation.

Un autre objectif est de gérer le service de réception de l'hôpital et de toutes ses ressources afin de faciliter la tâche du personnel médical et d'apporter les informations nécessaires en temps réel aux ambulanciers afin qu'ils puissent transporter les patients le plus rapidement possible vers l'hôpital optimal.

Ce mémoire est organisé en trois chapitres : Le premier est consacré aux différentes architectures et technologies pour le développement des applications mobiles. Le deuxième chapitre présente la problématique et l'objectif du travail, l'étude de l'existant, et la solution proposée . Le dernier chapitre sera consacré à la conception UMl (digrammes ..) ,la réalisation ainsi que les différents outils de développements, bases de données utilisées et quelques interfaces de l'application.

Chapitre 1

Généralités sur les applications mobiles

1.1 Introduction

Le choix des bonnes technologies web et mobiles est essentiel pour tout projet de développement. Les anciennes technologies peuvent limiter la durée de vie du logiciel. Des technologies mal adaptées peuvent ne pas convenir, limitant les performances, les fonctionnalités et même l'évolutivité futures. Le nombre de technologies modernes pour le développement web et mobile ne cesse de croître ce qui rend le choix difficile.

Dans ce chapitre, nous donnons un bref aperçu des différents types de technologies et de leurs objectifs dans la création de logiciels. Nous dressons ensuite la liste des technologies les plus courantes pour chacune d'entre elles.

1.2 Les techonologies web

1.2.1 Introduction

Une application web (ou web app) est un logiciel d'application qui s'exécute sur un serveur web, contrairement aux logiciels informatiques qui sont exécutés localement sur le système d'exploitation (OS) de l'appareil. L'utilisateur accède aux applications web par le biais d'un navigateur web avec une connexion réseau active. Ces applications sont programmées à l'aide d'une structure modélisée client-serveur - l'utilisateur ("client") reçoit des services par l'intermédiaire d'un serveur hors site hébergé par un tiers. Parmi les exemples d'applications web couramment utilisées, citons : le courrier électronique, la vente au détail en ligne, la banque en ligne et les enchères en ligne. [7]
Il existe plusieurs façons de cibler les appareils mobiles lors de la création d'applications web :

- Le Responsive Web Design peut être utilisé pour rendre une application web - qu'il s'agisse d'un site web classique ou d'une application à page unique - consultable sur de petits écrans qui fonctionnent bien avec des écrans tactiles.
- Les applications web progressives (PWA) sont des applications web qui se chargent comme des pages web ou des sites web ordinaires, mais qui peuvent offrir à l'utilisateur des fonctionnalités telles que le travail hors ligne et l'accès au matériel de l'appareil, traditionnellement réservées aux applications mobiles natives.

1.2.2 Conception web réactive

- **Définition**

La conception web réactive (ou Responsive Web Design) est une approche du développement web décrite pour la première fois par Ethan Marcotte en 2010, soit cinq ans avant la conception de Progressive Web App. [8]

- **Architecture**

  Fondamentalement, la philosophie du responsive web design est que la conception et le développement doivent être réalisés dans le but de répondre à l'appareil de l'utilisateur - ce qui signifie répondre au comportement, à la taille, à la plate-forme et à l'orientation de l'appareil utilisé. Cet objectif est atteint grâce à l'utilisation de grilles fluides, d'images flexibles et de requêtes média CSS :

  - Grilles fluides

  Les sites web responsives conçus avec des grilles fluides peuvent mieux gérer les différentes tailles d'écran sur le marché car, au lieu de définir des dimensions basées

sur les pixels, la grille fluide adopte un nouveau calcul basé sur le pourcentage.

— Images flexibles

Les images sur le web ne sont pas naturellement fluides, mais avec certaines configurations (propriété de largeur définie à 100% et propriété de hauteur définie à auto), toute image peut être rendue réactive sur tous les appareils.

— Requêtes multimédias CSS

Si une page web réactive avec des images flexibles et des grilles fluides est techniquement réactive, elle n'est pas aussi belle qu'elle pourrait l'être. C'est là que les requêtes média CSS entrent en jeu, car elles sont utilisées pour créer une expérience encore meilleure, adaptée aux différents appareils. Ces expériences personnalisées sont souvent introduites par l'ajout de points d'arrêt qui prennent effet à des tailles d'écran spécifiques.

— **Exemples**

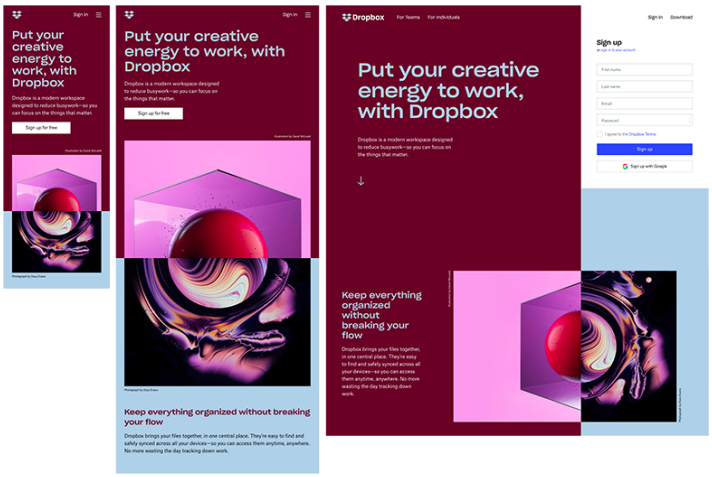

Figure 1.1: Site web Dropbox [1]

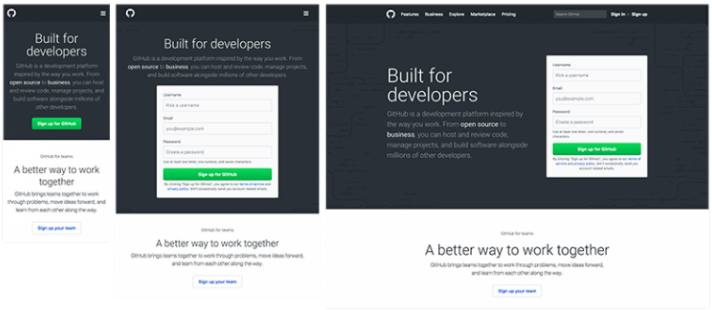

Figure 1.2: Site web Github [1]

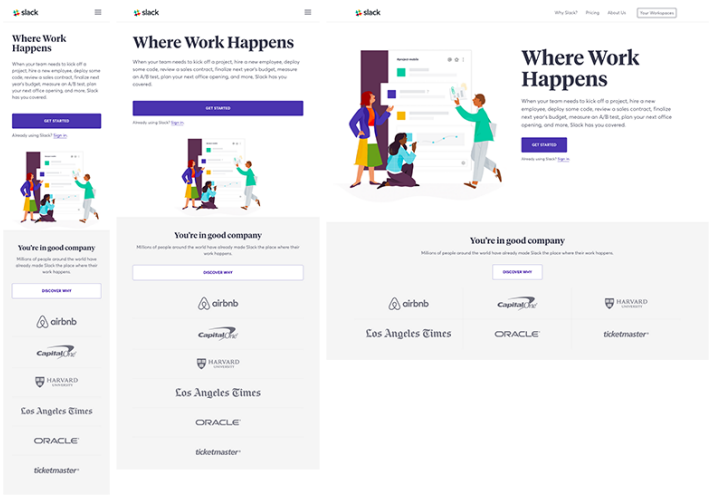

Figure 1.3: Site web Slack [1]

1.2.3 Applications web progressives

— **Définition**

L'application web progressive (ou Progressive Web App) est la prochaine évolution naturelle du web en raison de ses nombreux avantages par rapport au site web réactif typique. La partie "progressive", peut être expliquée comme suit : "au fur et à mesure que l'utilisateur établit une relation avec l'application, celle-ci devient de plus en plus puissante".

Une PWA est un site web semblable à une application, doté de (presque) toutes les fonctionnalités que l'on peut attendre d'une application mobile native, notamment les notifications push, les capacités hors ligne, etc. De ce fait, l'expérience globale est un cran au-dessus de celle de son équivalent en site web réactif, puisque la PWA peut conserver tous les avantages supposés d'une plateforme web. [8]

— **Architecture**

Il est en fait assez facile de résumer les principaux composants d'une PWA. En gros, pour rendre possibles toutes les fonctionnalités progressives susmentionnées, voici les exigences :

— Manifeste de l'application web

Le manifeste de l'application web est un fichier JSON qui fournit les métadonnées nécessaires au processus d'installation de votre PWA et détermine la façon dont votre PWA se présente sur l'écran d'accueil de l'utilisateur.

— Travailleurs de service

Universellement considéré comme le composant fondamental qui rend possible toutes les fonctionnalités progressives des PWA, le travailleur de service s'exécute indépendamment du navigateur et sur un thread différent du JavaScript principal.

— Contextes sécurisés

En tant que nouvelle norme du web, il est nécessaire qu'une PWA soit séparée par un protocole sécurisé - HTTPS. Cela garantit une communication sécurisée entre l'utilisateur et le serveur et, en retour, une expérience sans risque.

— **Exemples des applications PWA**

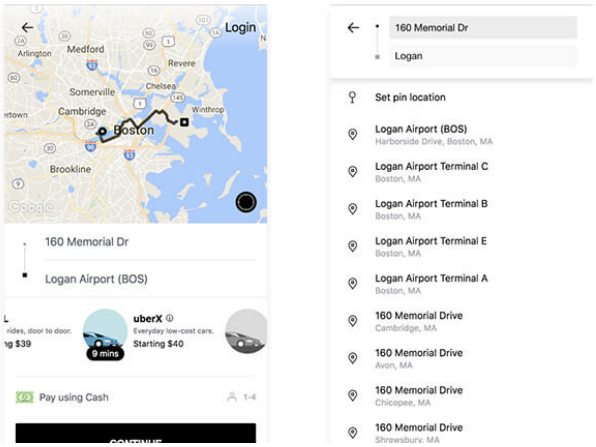

Figure 1.4: Site web Uber [2]

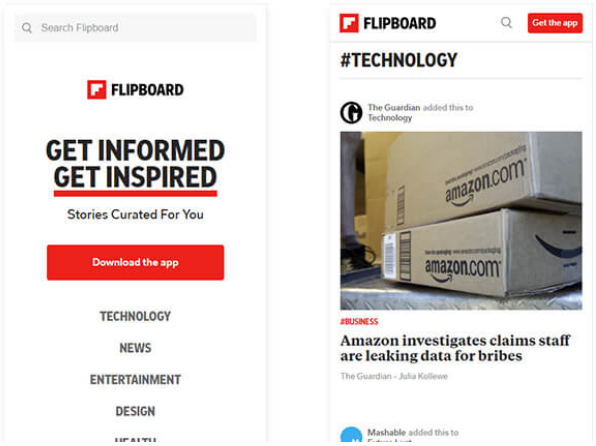

Figure 1.5: Site web Flipboard [2]

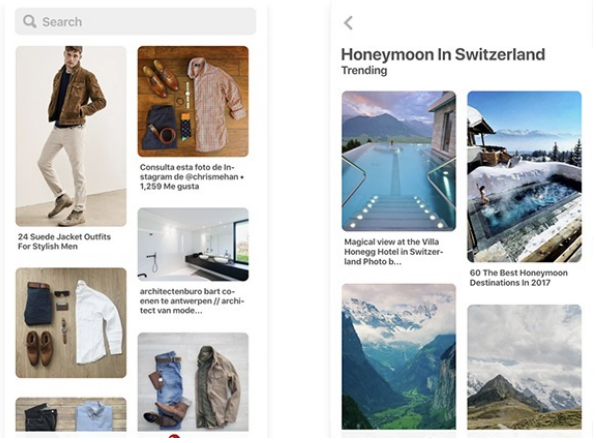

Figure 1.6: Site web Pinterest [2]

1.3 Les techonologies d'application mobile

1.3.1 Introduction

Une application mobile est un programme informatique ou un logiciel conçu pour fonctionner sur un appareil mobile tel qu'un téléphone, une tablette ou une montre. À l'origine, les applications étaient destinées à faciliter la productivité, notamment par le biais de bases de données de courriels, de calendriers et de contacts, mais la demande du public pour les applications a entraîné une expansion rapide dans d'autres domaines, tels que les jeux mobiles, l'automatisation des usines, les services GPS et de géolocalisation, le suivi des commandes et l'achat de billets, de sorte qu'il existe aujourd'hui des millions d'applications disponibles. Les applications sont généralement téléchargées à partir de plateformes de distribution d'applications exploitées par le propriétaire du système d'exploitation mobile, comme l'App Store (iOS) ou le Google Play Store.

Les applications mobiles peuvent être classées par de nombreuses méthodes. Un schéma commun consiste à distinguer les applications natives, multiplateformes et hybrides. [9]

1.3.2 Applications natives

Une application native est généralement écrite dans un langage de programmation pour un système d'exploitation particulier. Par rapport à d'autres types de produits, les applications natives offrent des performances constantes et sont parfois plus fiables. Selon la plateforme pour laquelle une application est développée, des langages de programmation spécifiques sont utilisés. Pour iOS, il s'agit principalement d'Objective-C et de Swift, tandis que les développeurs Android écrivent en Java ou en Kotlin.

Le passage au langage natif est le rêve de beaucoup de propriétaires d'applications, mais tous ne peuvent pas se le permettre. La raison principale est que pour faire fonctionner une application sur plusieurs plateformes, il faut développer et maintenir une application pour chaque plateforme séparément. Et pour beaucoup d'entrepreneurs, le développement d'une application native pour plusieurs plateformes coûte une fortune.

En termes simples, les applications natives impliquent la création de deux applications différentes avec des jeux de codes différents pour chaque plateforme Android et iOS. [3]

- **Avantages**
  - Vitesse élevée

    Comme les applications mobiles natives n'ont pas un code trop complexe, elles ont tendance à fonctionner plus rapidement que les autres applications. De nombreux éléments de l'application s'affichent rapidement car ils sont préchargés à l'avance.
  - Fonctionne bien hors ligne

    Les applications natives fonctionnent sans problème même en l'absence de connexion à Internet. Il est évident qu'une telle application est beaucoup plus pratique pour les utilisateurs, car ils peuvent accéder à toutes les fonctionnalités en déplacement ou en avion lorsqu'il n'y a pas de connexion.
- **Inconvénients**
  - Pas de code réutilisable

    Si un développeur souhaite créer des applications natives à la fois pour Android et iOS, il devra développer deux applications natives distinctes. Évidemment, cela prendrait beaucoup plus de temps et d'efforts que de développer une application

mobile multiplateforme avec une base de code réutilisable ou une application hybride avec un code backend partagé.

— Implique plus de talent

Les applications natives étant essentiellement spécifiques à un langage, les entreprises ont généralement du mal à trouver un développeur compétent pour développer une application native en parallèle. Si l'on compare les applications natives aux applications multiplateformes, si une entreprise souhaite toucher un public plus large, elle devra engager deux équipes de développement pour les applications natives. Alors qu'elle aurait pu se contenter d'une seule dans le cas d'une application multiplateforme.

— **Exemples**

La plupart des applications que les utilisateurs de smartphones installent quotidiennement sont natives. Voici quelques-uns des exemples les plus marquants de ce dont le développement d'applications natives est capable.

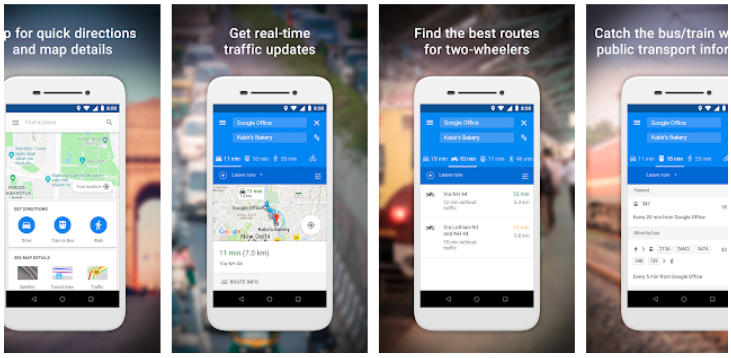

Figure 1.7: Application mobile Google Maps [3]

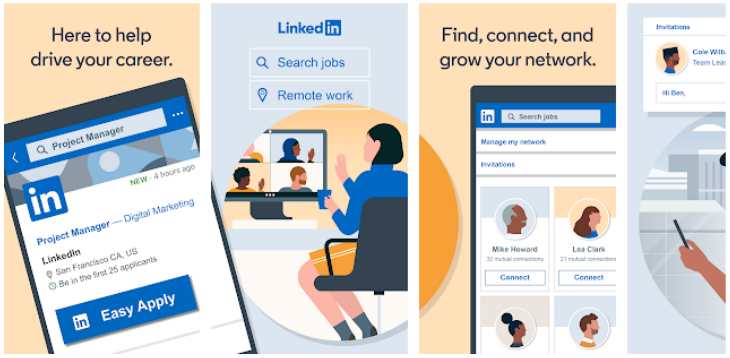

Figure 1.8: Application mobile Linkedin [3]

1.3.3 Applications hybrides

Les applications hybrides sont le juste milieu entre les applications natives et les applications web. Elles se composent de deux parties : le code du backend et un visualisateur natif qui peut être téléchargé pour afficher le backend dans une vue web. Contrairement aux applications web, les applications mobiles hybrides ne nécessitent pas de navigateur pour l'accès et peuvent tirer parti de n'importe quel plugin et des API d'un appareil. Elles sont moins chères à développer que les applications natives, mais leurs performances sont généralement plus faibles. [3]

— **Avantages**
  — Développement plus rapide

  Étant donné que l'application utilise le même code dorsal pour toutes les plateformes, la création d'une application hybride ne prend pas beaucoup de temps. En fait, tout ce qu'un développeur doit faire, c'est créer un shell natif pour afficher le code qui a déjà été développé en tant qu'application web. N'oubliez pas, cependant, que les applications hybrides comportant de nombreuses fonctionnalités peuvent prendre encore plus de temps, il est donc préférable de rester simple.
  — Maintenance simple

  Les applications hybrides étant basées sur la technologie web, elles sont plus faciles à entretenir que les applications natives ou hybrides, plus complexes en termes de programmation.
— **Inconvénients**
  — Impossible d'accéder hors ligne

  Étant donné que les applications hybrides sont essentiellement basées sur le web, elles ne fonctionnent pas sans connexion Internet. De plus, comme tous les éléments de l'application doivent être chargés, la vitesse de performance est généralement plus lente. Inutile de dire que cela impose une tonne de limitations de connexion à l'utilisateur de l'application. Les problèmes de connexion constituent une différence importante entre le développement d'applications natives et hybrides.
  — Incohérences du système d'exploitation

  Étant donné que les applications hybrides partagent une base de code, certaines fonctionnalités peuvent être prises en charge par Android et ne pas s'afficher sur un appareil iOS et vice versa. Il faut plus de sprints de test pour identifier les incohérences et une tonne de modifications pour corriger ces problèmes.
— **Exemples**

  Si, à première vue, les applications hybrides peuvent sembler n'être qu'une solution moins cher, en réalité, une bonne partie des meilleures applications de médias sociaux sont en fait hybrides. Jetons un coup d'œil aux exemples d'applications hybrides les plus populaires :

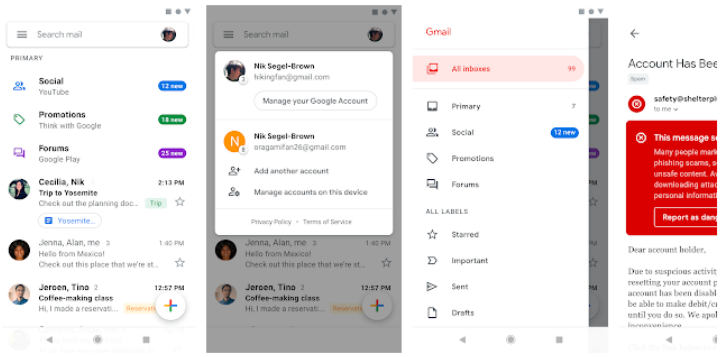

Figure 1.9: Application mobile Gmail [3]

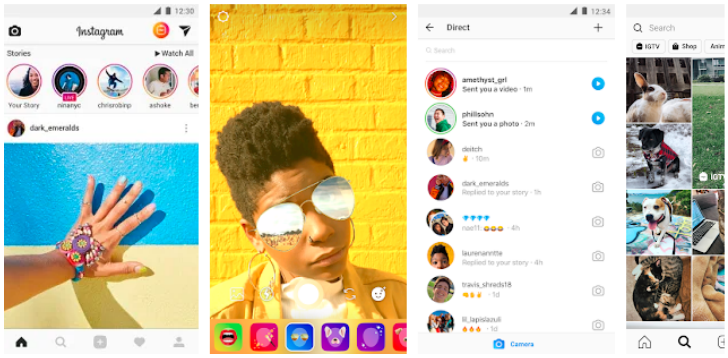

Figure 1.10: Application mobile Instagram [3]

1.3.4 Applications multiplateformes

Il existe différents systèmes d'exploitation fonctionnant sur divers smartphones, Android et iOS étant les plus utilisés. Chacune de ces plateformes utilise un environnement de programmation distinct avec son propre langage et son API.

Ainsi, la nécessité pour les développeurs mobiles d'atteindre la plus grande base d'utilisateurs possible, quelle que soit leur plate-forme de prédilection, a fait que les applications mobiles multiplateformes ont gagné en valeur. Des outils comme Xamarin, Flutter ou React-native ont rendu le développement d'applications multiplateformes largement populaire et accessible. [3]

- **Avantages**
  - Rentabilité

    La question des applications natives par rapport aux applications multiplateformes peut être débattue, mais lorsqu'il s'agit de rentabilité, le développement multiplateforme l'emporte haut la main. Outre le fait que la plupart des outils de développement multiplateforme sont disponibles sans réserve, cette approche évite de devoir faire appel à des développeurs distincts pour créer des applications pour différentes plateformes.
  - Réutilisable

    Avec les applications multiplateformes, les développeurs ne doivent plus écrire un code unique pour chaque système d'exploitation. Ils peuvent au contraire utiliser une base de code commune pour transférer le code sur différentes plateformes.
- **Inconvénients**
  - Processus de développement complexe

    Il faut un développeur compétent pour créer une application qui serait bien adaptée à quelques plateformes. En fait, il faut tenir compte de toutes les petites différences entre les systèmes d'exploitation et le matériel sur lequel ils fonctionnent, surtout lorsqu'il s'agit de mettre en œuvre une interface et des fonctionnalités complexes.
  - Des intégrations difficiles

    Les développeurs peuvent rencontrer des difficultés lors de l'intégration d'applications multiplateformes aux paramètres locaux et lors de l'engagement d'un fournisseur de services en nuage tiers. Le code d'une application multiplateforme HTML5 est compliqué en raison de la programmation de type callback utilisée pour communiquer avec les plugins natifs.
- **Exemples**

  La plupart des applications que les utilisateurs de smartphones installent quoti-

diennement sont natives. Voici quelques-uns des exemples les plus marquants de ce dont le développement d'applications natives est capable.

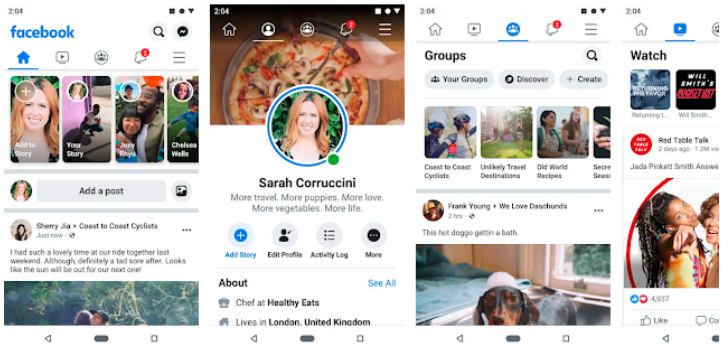

Figure 1.11: Application mobile Facebook [3]

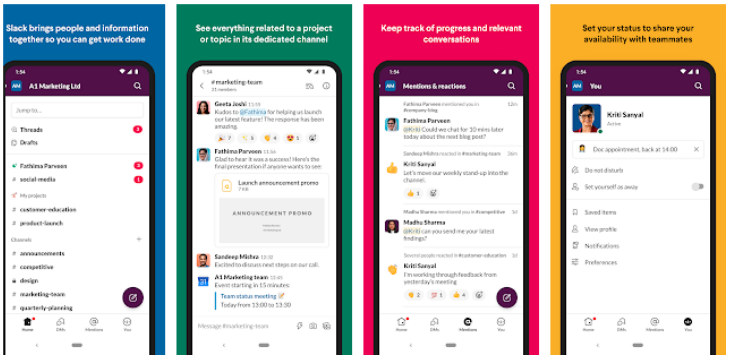

Figure 1.12: Application mobile Skype [3]

1.4 Système d'exploitation iOS

1.4.1 Définition

iOS (anciennement iPhone OS) est un système d'exploitation mobile créé et développé par Apple Inc. exclusivement pour son matériel. C'est le système d'exploitation qui alimente de nombreux appareils mobiles de la société, notamment l'iPhone et l'iPod Touch ; le terme comprenait également les versions fonctionnant sur les iPad jusqu'à ce que le nom iPadOS soit introduit avec la version 13 en 2019. Il s'agit du deuxième système d'exploitation mobile le plus installé au monde, après Android. Il constitue la base de trois autres systèmes d'exploitation fabriqués par Apple : iPadOS, tvOS et watchOS. Il s'agit d'un logiciel propriétaire, bien que certaines parties de celui-ci soient en open source sous la licence Apple Public Source License et d'autres licences. [10]

1.4.2 Historique

L'iPhone a été lancé en juin 2007 et le 5 septembre 2007, Apple a lancé l'iPod Touch qui possédait la plupart des fonctionnalités non téléphoniques de l'iPhone. En juin 2010, Apple a rebaptisé iPhone Os en iOS. La première génération d'iPad est sortie en avril 2010 et l'iPad Mini en novembre 2012. [10]

1.4.3 Fonctionnalités

iOS possède de nombreuses fonctionnalités, dont les suivantes, sans s'y limiter [11] :
— Multitâche
— Médias sociaux
— iCloud
— Achats intra-application
— Centre de jeu
— Centre de notification
— Accéléromètre
— Gyroscope
— API puissantes
— GPS
— Processeur haut de gamme
— Accessibilité
— Bluetooth
— Orientations
— Intégration de la caméra
— Services de localisation
— Cartes
— Courriel
— Pages web
— Contacts
— Messages

1.4.4 Architecture

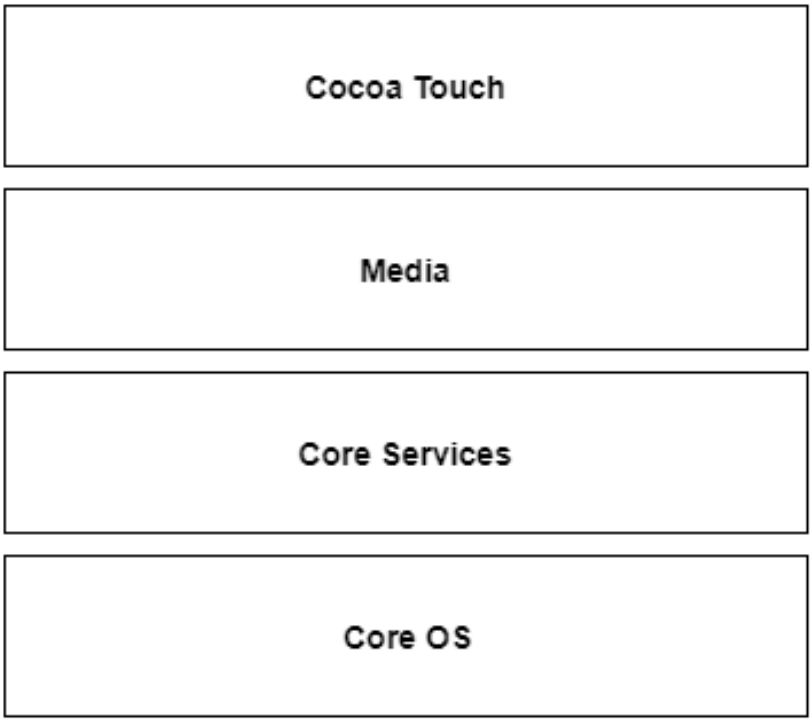

Figure 1.13: L'architecture iOS [4]

— **Core OS**
Toutes les technologies iOS sont basées sur les fonctionnalités de bas niveau fournies par la couche Core OS. Ces technologies comprennent le Core Bluetooth Framework, le External Accessory Framework, l'Accelerate Framework, le Security Services Framework, le Local Authorisation Framework, etc. [4]

— **Core Services**
Il existe de nombreux cadres disponibles dans la couche des services de traitement. Les détails de certains d'entre eux sont donnés ci-dessous.
— Cloudkit Framework
Les données peuvent être déplacées entre l'application et iCloud en utilisant le Cloudkit Framework.
— Core Foundation Framework
Il fournit les fonctionnalités de gestion et de service des données pour les applications iOS.
— Core Data Framework
Le modèle de données de l'application modèle vue contrôleur est géré à l'aide du Core Data Framework.
— Address Book Framework
Le framework Carnet d'adresses permet d'accéder à la base de données des contacts de l'utilisateur.
— Core Motion Framework
Toutes les données basées sur le mouvement de l'appareil sont accessibles à l'aide du Core Motion Framework.

— Healthkit Framework
Les informations relatives à la santé de l'utilisateur peuvent être traitées par ce nouveau framework.
— Core Location Framework
Ce framework fournit les informations de localisation et de cap aux différentes applications.

— **Media**
La couche média permet toute la technologie graphique, audio et vidéo du système. Les différents frameworks sont :
— UIKit Graphics
Il permet de concevoir des images et d'animer le contenu des vues.
— Core Graphics Framework
Il prend en charge le rendu vectoriel 2D et le rendu d'images et constitue le moteur de dessin natif des applications iOS.
— Core Animation
La technologie Core Animation optimise l'expérience d'animation des applications.
— Media Player Framework
Ce cadre prend en charge la lecture de listes de lecture et permet à l'utilisateur d'utiliser sa bibliothèque iTunes.
— AV Kit
Ce kit fournit diverses interfaces faciles à utiliser pour la présentation vidéo.

— **Cocoa Touch**
La couche tactile de cacao fournit les cadres suivants :
— EventKit Framework
Ceci montre les interfaces standard du système utilisant des contrôleurs de vue pour visualiser et modifier les événements liés au calendrier.
— GameKit Framework
Cela permet aux utilisateurs de partager leurs données de jeu en ligne à l'aide du centre de jeu.
— MapKit Framework
Cela permet d'obtenir une carte déroulante qui peut être intégrée à l'interface utilisateur de l'application.

1.5 Système d'exploitation Android

1.5.1 Définition

Android est un système d'exploitation mobile basé sur une version modifiée du noyau Linux et d'autres logiciels open source, conçu principalement pour les appareils mobiles à écran tactile tels que les smartphones et les tablettes.

Android est développé par un consortium de développeurs connu sous le nom d'Open Handset Alliance et parrainé commercialement par Google. Il a été dévoilé en novembre 2007, et le premier appareil Android commercial, le HTC Dream, a été lancé en septembre 2008. [12]

1.5.2 Historique

Android a débuté en 2003 en tant que projet de la société de technologie américaine Android Inc., visant à développer un système d'exploitation pour appareils photo numériques.
En 2004, le projet a changé pour devenir un système d'exploitation pour smartphones. Android Inc., a été racheté par la société américaine de moteurs de recherche Google Inc., en 2005. Chez Google, l'équipe Android a décidé de baser son projet sur Linux, un système d'exploitation open source pour ordinateurs personnels.

1.5.3 Fonctionnalités

— **Messagerie**
Les SMS et MMS sont des formes de messagerie disponibles, y compris la messagerie texte filetée et Android Cloud To Device Messaging (C2DM) et maintenant la version améliorée de C2DM, Android Google Cloud Messaging (GCM) fait également partie des services Android Push Messaging.
Les téléphones Android ont également la possibilité d'envoyer et de recevoir des RCS via l'application de messagerie (si prise en charge par l'opérateur). [13]

— **Correction automatique et dictionnaire**
Le système d'exploitation Android a une fonctionnalité intéressante appelée Correction automatique. Lorsqu'un mot est mal orthographié, Android recommande les mots significatifs et corrects correspondant aux mots disponibles dans le dictionnaire.
Les utilisateurs peuvent ajouter, modifier et supprimer des mots du dictionnaire selon leur souhait.

— **Navigateur web**

Le navigateur web disponible dans Android est basé sur le moteur de mise en page open-source Blink (anciennement WebKit), associé au moteur JavaScript V8 de Chromium. Ensuite, le navigateur Android utilisant WebKit a obtenu un score de 100/100 au test Acid3 sur Android 4.0 ICS ; le navigateur basé sur Blink a actuellement une meilleure prise en charge des normes.
L'ancien navigateur web est appelé de manière variable "navigateur Android", "navigateur AOSP", "navigateur de stock", "navigateur natif" et "navigateur par défaut" (à partir du moment où il était toujours le navigateur par défaut). À partir d'Android 4.4 KitKat, Google a commencé à octroyer des licences à Google Chrome (un logiciel propriétaire) séparément d'Android, mais généralement fourni avec (ce que la plupart des fournisseurs d'appareils ont fait). Depuis Android 5.0 Lollipop, le navigateur WebView que les applications peuvent utiliser pour afficher du contenu web sans quitter l'application a été séparé du reste du micrologiciel Android afin de faciliter les mises à jour de sécurité séparées par Google.

— **Fonctionnalités basées sur la voix**
La recherche vocale Google est disponible depuis la version initiale. Les actions vocales pour les appels, les SMS, la navigation, etc. sont prises en charge à partir d'Android 2.2. À partir d'Android 4.1, Google a étendu les actions vocales avec la possibilité de répondre et de lire les réponses à partir du Knowledge Graph de Google lorsqu'il est interrogé avec des commandes spécifiques.
La capacité de contrôler le matériel n'a pas encore été mise en œuvre.

— **Multi-touches**

Android a un support natif pour le multi-touch qui a été initialement mis à disposition dans des combinés tels que le HTC Hero. La fonctionnalité a été désactivée à l'origine au niveau du noyau (peut-être pour éviter de violer les brevets d'Apple sur la technologie d'écran tactile à l'époque).
Google a depuis publié une mise à jour pour le Nexus One et le Motorola Droid qui permet le multi-touch de manière native.

— **Multitâche**

Le multitâche des applications, avec une gestion unique de l'allocation de mémoire, est disponible.

— **Capture d'écran**

Android prend en charge la capture d'une capture d'écran en appuyant simultanément sur les boutons d'alimentation et de l'écran d'accueil.
Avant Android 4.0, les seules méthodes de capture d'une capture d'écran étaient via des personnalisations (applications) du fabricant et de tiers, ou autrement en utilisant une connexion PC (outil de développement DDMS).
Ces méthodes alternatives sont toujours disponibles avec la dernière version d'Android.

— **Enregistrement TV**

Android TV prend en charge la capture et la relecture.

— **Appel video**

Android ne prend pas en charge les appels vidéo natifs, mais certains combinés ont une version personnalisée du système d'exploitation qui le prend en charge, soit via le réseau UMTS (comme le Samsung Galaxy S), soit sur IP.
Les appels vidéo via Google Talk sont disponibles dans Android 2.3.4 (Gingerbread) et versions ultérieures. Gingerbread permet au Nexus S de passer des appels Internet avec un compte SIP.
Cela permet une numérotation VoIP améliorée vers d'autres comptes SIP et même des numéros de téléphone. Skype 2.1 propose des appels vidéo sous Android 2.3, y compris la prise en charge de la caméra frontale.
Les utilisateurs de l'application Google+ pour Android peuvent effectuer un chat vidéo avec d'autres utilisateurs de Google+ via Hangouts.

— **Prise en charge de plusieurs langues**

Android prend en charge plusieurs langues.

— **Accessibilité**

La synthèse vocale intégrée est fournie par TalkBack pour les personnes malvoyantes ou sans vision. Des améliorations pour les personnes malentendantes sont disponibles, de même que d'autres aides.

1.5.4 Architecture Android

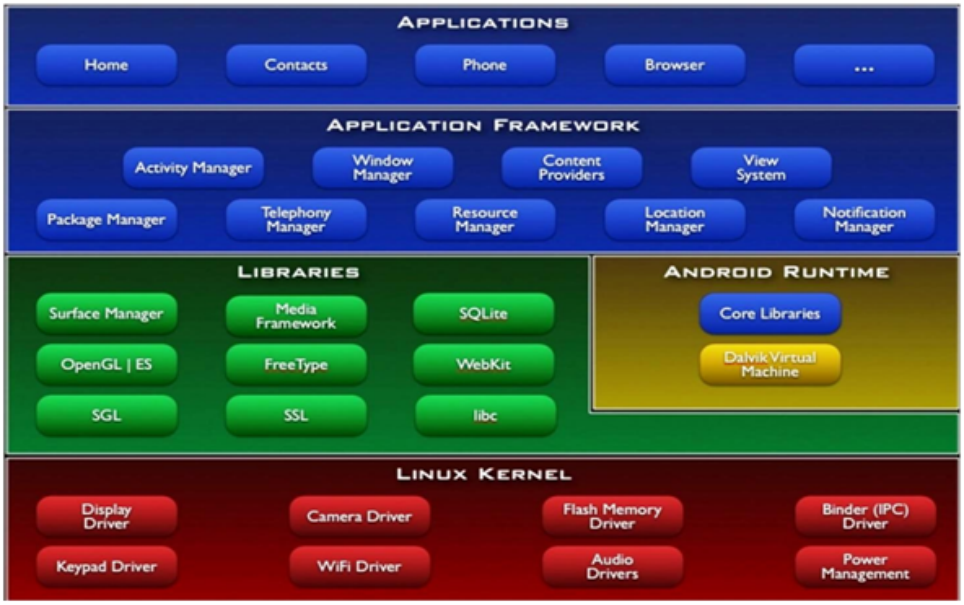

Figure 1.14: L'architecture Android [5]

— **Noyau Linux**

Android utilise le puissant noyau Linux et prend en charge un large éventail de pilotes matériels. Le noyau est le cœur du système d'exploitation qui gère les demandes d'entrée et de sortie du logiciel.
Cela fournit des fonctionnalités système de base comme la gestion des processus, la gestion de la mémoire, la gestion des périphériques comme la caméra, le clavier, l'affichage, etc. le noyau gère tout.

Linux est vraiment bon en réseau et il n'est pas nécessaire de l'interfacer avec le matériel périphérique.
Le noyau lui-même n'interagit pas directement avec l'utilisateur mais interagit plutôt avec le shell et d'autres programmes ainsi qu'avec les périphériques matériels du système.

— **Bibliothèques**

Au sommet d'un chenil Linux, il y a un ensemble de bibliothèques comprenant des navigateurs web open source tels que WebKit, la bibliothèque libc. Ces bibliothèques sont utilisées pour lire et enregistrer de l'audio et de la vidéo.
Le SQLite est une base de données utile pour le stockage et le partage des données d'application. Les bibliothèques SSL sont responsables de la sécurité Internet, etc.

— **Runtime Android**

Le Runtime Android fournit un composant clé appelé Dalvik Virtual Machine qui est une sorte de machine virtuelle java. Il est spécialement conçu et optimisé pour Android. La VM Dalvik est la machine virtuelle de processus dans le système d'exploitation Android. C'est un logiciel qui exécute des applications sur les appareils Android.

La VM Dalvik utilise les fonctionnalités de base de Linux telles que la gestion de la mémoire et le multithreading en langage Java.
La VM Dalvik permet à chaque application Android d'exécuter son propre processus. La VM Dalvik exécute les fichiers au format .dex.

— **Cadre d'application**

La couche cadre d'application fournit de nombreux services de niveau supérieur aux applications telles que le gestionnaire de fenêtres, le système d'affichage, le gestionnaire de packages, le gestionnaire de ressources, etc. Les développeurs d'applications sont autorisés à utiliser ces services dans leurs applications.

— **Applications**

Vous trouverez toutes les applications Android dans la couche supérieure et vous écrirez votre application et l'installerez sur cette couche. Des exemples de telles applications sont les contacts, les livres, les navigateurs, les services, etc. Chaque application joue un rôle différent dans l'ensemble des applications. [5]

1.5.5 Langages de programation Android

— **Java**

Tout d'abord, Java était le langage officiel pour le développement d'applications Android (mais il a maintenant été remplacé par Kotlin) et, par conséquent, c'est aussi le langage le plus utilisé. De nombreuses applications du Play Store sont construites en Java, et c'est aussi le langage le plus soutenu par Google. En plus de tout cela, Java a une grande communauté en ligne pour le soutien en cas de problèmes.

Cependant, Java est un langage compliqué à utiliser pour un débutant car il contient des sujets complexes comme les constructeurs, les exceptions de pointeur nul, la concurrence, les exceptions vérifiées, etc. De plus, le kit de développement logiciel (SDK) d'Android augmente la complexité à un nouveau niveau !

Dans l'ensemble, Java est un langage idéal pour profiter pleinement des joies du développement d'applications Android. Cependant, il peut être un peu complexe pour les débutants qui préfèrent commencer par quelque chose de plus facile et y revenir ensuite. [14]

— **Kotlin**

Désormais, Kotlin est le langage officiel pour le développement d'applications Android déclaré par Google depuis 2019. Kotlin est un langage de programmation multiplateforme qui peut être utilisé comme une alternative à Java pour le développement d'applications Android. Il a également été présenté comme un langage Java secondaire "officiel" en 2017. Kotlin peut interopérer avec Java et il fonctionne sur la machine virtuelle Java.

La seule différence notable est que Kotlin supprime les fonctionnalités superflues de Java, telles que les exceptions de pointeur nul. Il supprime également la nécessité de terminer chaque ligne par un point-virgule. En bref, Kotlin est beaucoup plus simple à essayer pour les débutants que Java et il peut également être utilisé comme "point d'entrée" pour le développement d'applications Android.

— **C++**

Le C++ peut être utilisé pour le développement d'applications Android à l'aide du kit de développement natif Android (NDK). Cependant, une application ne peut pas être créée entièrement en C++ et le NDK est utilisé pour implémenter des parties de l'application en code natif C++. Cela permet d'utiliser les bibliothèques de code C++ pour l'application si nécessaire.

Bien que le C++ soit utile pour le développement d'applications Android dans certains cas, il est beaucoup plus difficile à mettre en place et beaucoup moins flexible. Il peut également entraîner un plus grand nombre de bogues en raison de

sa complexité accrue. Il est donc préférable d'utiliser Java plutôt que C++ car il n'offre pas un gain suffisant pour compenser les efforts requis.

— **C#**
C# est assez similaire à Java et est donc idéal pour le développement d'applications Android. Comme Java, C# met également en œuvre la collecte des déchets, ce qui réduit les risques de fuites de mémoire. De plus, C# a une syntaxe plus propre et plus simple que Java, ce qui rend le codage plus facile.
Auparavant, le plus gros inconvénient de C# était qu'il ne pouvait fonctionner que sur les systèmes Windows car il utilisait le .NET Framework. Toutefois, ce problème a été résolu par Xamarin. Android (anciennement Mono pour Android), qui est une mise en œuvre multiplateforme de l'infrastructure de langage commune. Désormais, les outils Xamarin.Android peuvent être utilisés pour écrire des applications Android natives et partager le code sur plusieurs plateformes.

— **Python**
Python peut être utilisé pour le développement d'applications Android, même si Android ne prend pas en charge le développement Python natif. Cela peut se faire à l'aide de divers outils qui convertissent les applications Python en paquets Android pouvant fonctionner sur les appareils Android.
Kivy, par exemple, est une bibliothèque Python à code source ouvert utilisée pour le développement d'applications mobiles. Elle prend en charge Android et encourage également le développement rapide d'applications. Cependant, l'inconvénient est qu'il n'y aura pas d'avantages natifs pour Kivy car elle n'est pas supportée en natif.

— **Corona**
Corona est un kit de développement logiciel qui peut être utilisé pour développer des applications Android en utilisant Lua. Il possède deux modes opérationnels, à savoir Corona Simulator et Corona Native. Le Corona Simulator est utilisé pour construire directement des applications tandis que le Corona Native est utilisé pour intégrer le code Lua avec un projet Android Studio pour construire une application utilisant des fonctionnalités natives.
Si Lua est un peu limité par rapport à Java, il est aussi beaucoup plus simple et sa courbe d'apprentissage est plus facile. En outre, il existe des fonctions de monétisation intégrées ainsi que divers actifs et plugins qui enrichissent l'expérience de développement d'applications. Corona est surtout utilisé pour créer des applications graphiques et des jeux, mais il n'est en aucun cas limité à cela.

— **HTML, CSS, JavaScript**
Les applications Android peuvent être créées à l'aide de HTML, CSS et JavaScript en utilisant le cadre Adobe PhoneGap qui est alimenté par Apache Cordova. Le cadre PhoneGap permet essentiellement d'utiliser les compétences en matière de développement web pour créer des applications hybrides qui s'affichent par le biais d'une "WebView" mais qui sont conditionnées comme une application.
Si le cadre Adobe PhoneGap est suffisant pour les tâches de base dans le domaine du développement d'applications Android, il ne nécessite guère de programmation, à l'exception de JavaScript. Et comme il faut beaucoup de travail pour créer une application de qualité, il est préférable d'utiliser les autres langages de cette liste pour être considéré comme un véritable développeur Android.

Nom	Numéro (s) de version	Date de sortie initiale stable	Niveau API
Pas de nom officiel	1.0	23 septembre 2008	1
Pas de nom officiel	1.1	9 février 2009	2
Cupcake	1.5	27 avril 2009	3
Donut	1.6	15 septembre 2009	4
Eclair	2.0 - 2.1	26 octobre 2009	5 - 7
Froyo	2.2 - 2.2.3	20 mai 2010	8
Gingerbread	2.3 - 2.3.7	6 décembre 2010	9 - 10
Honeycomb	3.0 à 3.2.6	22 février 2011	11 - 13
Ice Cream Sandwich	4.0 à 4.0.4	18 octobre 2011	14 - 15
Jelly Bean	4.1 - 4.3.1	9 juillet 2012	16 - 18
KitKat	4.4 - 4.4.4	31 octobre 2013	19 - 20
Lollipop	5.0 - 5.1.1	12 novembre 2014	21 - 22
Marshmallow	6.0 - 6.0.1	5 octobre 2015	23
Nougat	7.0 - 7.1.2	22 août 2016	24 - 25
Oreo	8.0	21 août 2017	26
Oreo	8.1	5 décembre 2017	27
Pie	9	6 août 2018	28
Android 10	10	3 septembre 2019	29
Android 11	11	8 septembre 2020	30
Android 12	12	TBA	31

Table 1.1: Historique des versions d'Android [6]

1.5.6 Historique des versions d'Android

L'historique des versions du système d'exploitation mobile Android a commencé avec la sortie publique de la version bêta d'Android le 5 novembre 2007. La première version commerciale, Android 1.0, est sortie le 23 septembre 2008.
Android est continuellement développé par Google et l'Open Handset Alliance (OHA), et il a vu plusieurs mises à jour de son système d'exploitation de base depuis la version initiale.

1.6 Comparaison entre les systèmes d'exploitation Android et iOS

Critère	Android	iOS
Développeur	Google	Apple
Famille	Linux	OS X, Unix
Programmé en	C, C++, Java	Objective-C, C, C++, Swift
Widgets	Oui	Non, sauf dans le Notification Center
Langages Principales	Java, Kotlin	Objective-C, Swift
Code source	Source ouverte	Fermé, avec des composants de source ouverte

Table 1.2: Comparaison entre les systèmes d'exploitation Android et iOS

1.7 Conclusion

Dans ce chapitre, nous avons présenté les différentes technologies web et mobiles et même les systèmes d'exploitation concernés par le développement mobile, plus particulièrement le système d'exploitation Android en étalant son architecture, ses fonctionnalités ainsi que les langages de programmation utilisés pour développer dans le système.

Chapitre 2

Étude de l'existant et approche proposée

2.1 Introduction

La protection civile, communément appelé « pompiers » décrit le service qui englobe tous les acteurs de la sécurité civile et de la lutte contre l'incendie. La Protection civile a pour mission la protection des personnes, des biens et de l'environnement, elle comprend aussi bien les secours d'urgence aux personnes victimes d'accidents, de catastrophes et leur évacuation vers les hôpitaux. Les secours à domicile (accident domestique, hémorragie, arrêt cardiaque...) ou sur la voie publique (blessés sur la voie publique ou dans les lieux publics, accidents de la circulation routière. . .) qui représentent la majorité des interventions. L'augmentation de nombres d'interventions rend les méthodes traditionnelles insuffisantes et le personnel de la protection civile se trouve dépassé, d'où l'intérêt de recourir au technologies modernes qui fournissent une aide substantielle au personnel de la protection civile, et réduit les pertes humaines en offrant les moyens efficaces d'interventions.

Dans ce chapitre nous allons faire une étude des différents travaux qui ont été faites sur ce domaine et nous allons conclure le chapitre par notre approche proposée.

2.2 Problématique

Dans notre travail nous allons nous intéresser sur la mission de la protection des personnes et les secours d'urgence , qui représente la grande majorité des interventions de la protection civile. Lorsque les ambulanciers arrivent chez le patient, ils l'emmènent généralement à l'hôpital le plus proche, qui peut être ou non le plus rapide en matière de temps, en raison des embouteillages et de l'infrastructure routière pour n'en nommer que quelques-uns. Une fois le patient arrivé à l'hôpital, il est soumis à de multiples tests médicaux dont certains auraient pu être effectués au préalable sur le chemin de l'hôpital. Après toutes les difficultés pour arriver à l'hôpital à temps, le médecin responsable peut rejeter le patient en raison de problèmes de saturation, du manque de personnel ou même du manque de ressources médicales,.

Ces problèmes peuvent malheureusement conduire à des complications voir le décès du patient. Et la cause principale est le manque de moyens de communication entre les différents établissements de santé, ces derniers fonctionnent indépendamment les uns des autres.

Après l'émergence des technologies mobiles beaucoup d'applications ont été développées dans divers domaines. Cependant, nous avons pensé à utiliser ces technologies pour créer une plateforme en ligne et en temps réel qui améliore la communication entre les services de protection civile et les hôpitaux afin d'éviter les complication et les pertes humaines .

Cette plateforme n'est pas bénéfique seulement pour les patients mais aussi pour le personnel médical et les ambulanciers. Il s'agit de gérer le service de réception de l'hôpital ainsi que de toutes ces ressources afin de faciliter la tâche du personnel médical ainsi que les ambulanciers.

2.3 Étude de l'existant

2.3.1 En Algerie

Jusqu'à présent, en Algérie, il n'y a malheureusement aucune solution moderne qui a été proposé pour les services de la protection civile, ces derniers utilisent les méthodes traditionnelles (manuelles).

2.3.2 A l'étranger

À l'étranger, les services de protection civile utilisent souvent le système manuel, avec certains pays comme les États-Unis et le Royaume-Uni où le patient n'est pas toujours dirigé vers l'urgence ou l'hôpital le plus proche en raison de l'utilisation du mécanisme de détournement des ambulances.

Le déroutement des ambulances est une stratégie controversée visant à désengorger temporairement les services d'urgence. Lorsqu'un hôpital invoque le statut de déroutement, les ambulances qui arrivent sont dirigées vers d'autres établissements. En réponse à l'encombrement des urgences, des patients moins gravement blessés sont transportés vers d'autres établissements proches.

À court terme, le déroutement des ambulances donne un répit au service d'urgence qui a invoqué le statut de déroutement, lui permettant de retrouver un fonctionnement optimal pendant qu'il traite le surplus de patients. Toutefois, si la situation se prolonge, elle peut créer un effet domino, en incitant les établissements voisins, désormais engorgés par les patients déroutés, à se mettre eux-mêmes en état de déroutement. Cela peut également entraîner des retards dans les soins médicaux pour les patients se trouvant ailleurs dans le système de soins de santé. Si une ambulance ne peut pas amener les patients à l'établissement le plus proche, ils doivent être transportés sur de plus longues distances pour recevoir le traitement nécessaire. Cette augmentation du temps de transport peut réduire la disponibilité des ambulances pour de nouveaux appels pour d'autres patients en attente de services médicaux d'urgence.[15]

Il existe plusieurs plateformes qui ciblent le service médical, mais aucun d'entre eux ne se concentre sur l'idée que nous présentons, Les plus proches que nous avons trouvées sont les plateformes qui indiquent les temps d'attente aux urgences, l'une d'entre elles est l'application mobile Doctr.

— **L'application Doctr**

Doctr donne accès aux temps d'attente et aux taux d'occupation des salles d'urgence au Canada en temps réel, pour mission de faciliter l'accès aux services de santé au Canada. Il affiche les hôpitaux et les cliniques sans rendez-vous les plus proches du patient, ce qui facilite la prise de rendez-vous avec un médecin ou une infirmière.

En fonction de l'endroit où se trouve le patient, Doctr lui fournira une liste de cliniques sans rendez-vous et de centres de soins d'urgence où il peut se rendre pour des urgences mineures comme alternative à la salle d'urgence. Doctr est disponible partout dans la province de Québec et est actuellement en test bêta en Colombie-Britannique, en Alberta, au Manitoba et à l'Île-du-Prince-Édouard. [16]

2.4 Solutions suggérées

— **Construire plus d'hôpitaux**
L'une des solutions est de construire plus d'hôpitaux et d'urgences dans des endroits stratifiés en fonction de la population de chaque zone, les zones densément peuplées ayant plus d'hôpitaux que les zones moins peuplées.
Ces hôpitaux devraient disposer d'équipements de pointe, d'une grande capacité d'accueil et d'une équipe médicale importante travaillant 24 heures sur 24.
Ils doivent également tenir compte de la santé et du bien-être dans leur conception, ces considérations peuvent aller de caractéristiques telles que la lumière du jour, les vues, la bonne qualité de l'air intérieur, le confort acoustique et thermique à d'autres plus cruciales comme l'accessibilité, le contrôle des infections et l'orientation des visiteurs.

— **Plateforme numérique**
Utiliser les technologies modernes pour créer une plateforme qui facilite la connexion et la communication entre les ambulances, les patients et les hôpitaux dans le but de réduire le temps de réponse aux urgences.
Cette plateforme est une application mobile qui utilise des technologies de pointe pour géolocaliser l'ambulance et lui fournir des caractéristiques en temps réel des hôpitaux proches tels que la distance, le nombre de lits, les médecins actifs, les temps d'attente aux urgences et bien plus encore.
Cette gestion en temps réel aide l'ambulancier à choisir l'hôpital optimal (en matière de temps du trajet et de disponibilité et d'autres facteurs) pour sauver la vie du patient.

2.5 Solution retenu

Si la construction de nouveaux établissements de santé peut réduire le temps de transport des patients, elle reste très limitée en l'absence d'une réelle mise en œuvre de la technologie qui apporterait une grande amélioration.

L'utilisation d'outils numériques tels que les smartphones dans la vie de tous les jours sont incontournables et apportent une aide précieuse dans différents domaines notamment le domaine médical.

Cette application Android est basée sur la géo-localisation et la collecte des informations en temps réel des hôpitaux qui se trouvent à proximité. Elle serait en mesure de localiser le complexe patient-ambulance et de le mettre en relation avec l'unité d'urgence disponible, la plus proche et la mieux équipée , et de faire une évaluation de la sévérité afin de prendre toutes les mesures nécessaires pour offrir au patient les meilleurs soins possibles.

— **Les éléments de l'application**
  — Le patient
  Un patient est tout bénéficiaire de services de soins de santé qui sont dispensés par des professionnels de la santé. Le patient est le plus souvent malade ou blessé et a besoin d'être soigné par un médecin, une infirmière ou tout autre prestataire de soins de santé.Si les blessures du patient mettent sa vie en danger, il sera traité par un personnel d'urgence médicale.
  — L'ambulance
  Une ambulance est un véhicule équipé médicalement qui transporte des patients

vers des centres de traitement, tels que des hôpitaux. En général, le patient reçoit des soins médicaux avant d'arriver à l'hôpital.
Les ambulances sont utilisées pour répondre aux urgences médicales par les services médicaux d'urgence. À cette fin, elles sont généralement équipées de gyrophares et de sirènes. Elles peuvent transporter rapidement des ambulanciers et d'autres premiers intervenants sur les lieux, transporter des équipements pour administrer les soins d'urgence et transporter les patients à l'hôpital ou vers d'autres soins définitifs. La plupart des ambulances sont conçues à partir de fourgons ou de camionnettes. D'autres prennent la forme de motos, de voitures, de bus, d'avions et de bateaux. [17]

- L'hôpital
  Un hôpital est un établissement de soins de santé qui traite les patients avec un personnel médical et infirmier spécialisé et des équipements médicaux. Le type d'hôpital le plus connu est l'hôpital général, qui dispose généralement d'un service d'urgence pour traiter les problèmes de santé urgents, qu'il s'agisse de victimes d'incendies ou d'accidents ou d'une maladie soudaine. [18]
  Un service d'urgence, également connu sous le nom d'unité d'urgence ou simplement d'urgences, est une installation de traitement médical spécialisée dans la médecine d'urgence, c'est-à-dire les soins aigus des patients qui se présentent sans rendez-vous préalable, soit par leurs propres moyens, soit par ceux d'une ambulance. Le service des urgences se trouve généralement dans un hôpital ou un autre centre de soins primaires.
  En raison de la nature imprévue de la présence des patients, le service doit fournir un traitement initial pour un large éventail de maladies et de blessures, dont certaines peuvent mettre la vie en danger et nécessiter une attention immédiate. [19]

- **Sélection de l'hôpital**
  - **Les paramètres de sélection**
    Il existe plusieurs paramètres pour choisir l'hôpital optimal :
    - La distance
      La distance mesure la longueur du trajet entre l'hôpital et la position actuelle de l'ambulance transportant le patient. Ce paramètre est le plus important, car il permet de calculer le temps de trajet, ce qui aide l'ambulancier à décider de la procédure à suivre.
    - Le nombre de lits
      Un lit d'hôpital est un lit spécialement conçu pour les patients hospitalisés ou les personnes nécessitant une certaine forme de soins de santé. Ces lits présentent des caractéristiques particulières, tant pour le confort et le bien-être du patient que pour la commodité du personnel soignant. Ce paramètre est également important car la capacité du nombre de patients dans l'établissement est mesurée en "lits" disponibles.
    - Le nombre de médecins actifs
      Le nombre de médecins actifs est le nombre de médecins qui sont en service et prêts à traiter des patients à un moment donné dans un hôpital donné. Ce paramètre est essentiel car il indique la capacité et la disponibilité de l'hôpital à recevoir des patients.
    - Le nombre de patients en attente
      Le nombre de patients en attente est le nombre de patients qui se trouvent

dans l'hôpital à un moment donné et qui sont traités par un médecin en lui attribuant un lit. Ce paramètre est essentiel car il indique en fonction du niveau de gravité du cas du patient s'il doit attendre ou non.

— **La corrélation entre les paramètres**

Bien que les quatre paramètres soient essentiels, ils diffèrent par leur niveau d'importance. Les paramètres nombre de médecins actifs et nombre de patients en attente sont calculés en corrélation car le rapport entre eux est plus pertinent.

— Algorithme

```
Debut
        #Declaration et initiation des facteurs constants
        constant entier distanceFacteur <- 1, patientsDocteursFacteur <- 10, litsPresFacteur <- 12;

        #Declaration de valeurs maximales
        reel distanceMax, patientsDocteursMax, litsPresMax;

        #Distance maximale
        distanceMax <- hopitalList.getMaxDistance();


        #Rapport maximal entre le nombre de patients en attente et le nombre de docteurs pr ts avec une penalite si le nombre de docteurs est egal a 0
        patientsDocteursMax <- hopitalList.getMaxNbrPatientsDocteurs();


        #Nombre maximal de lits pr ts
        litsPresMax <- hopitalList.getMaxNbrLitsPres();


        #remplir les points pour chaque hopital
        pour i de 0 a hopitalList.taille() faire

            #Calcul de resultat de distance en divisant sur la valeur maximale pour la normalisation et sur le facteur d'importance
            reel distanceResulta <- (hopitalList[i].getDistance() / distanceMax) / distanceFacteur;

            #Calcul de resultat du rapport entre le nombre de patients en attente et le nombre de docteurs pr ts en divisant sur la valeur maximale pour la normalisation et sur le facteur d'importance
            reel patientsDocteursResulta <- (hopitalList[i].getNbrPatientsDocteurs() / patientsDocteursMax) / patientsDocteursFacteur;

            #Calcul de resultat du nombre de lits pr ts avec une penalite si le nombre de docteurs est egal a 0 en divisant sur la valeur maximale pour la normalisation et sur le facteur d'importance
            reel litsPresResulta <- (1 - hopitalList[i].getNbrLitsPres() / litsPresMax) / litsPresFacteur;
```

```
31
32              #Diviser les resultats obtenus sur 3
33              reel point <- (distanceResulta+
    patientsDocteursResulta+litsPresResulta) / 3;
34
35              #Attribuer le point obtenu sur l'attribut de point
    a chaque hopital
36              hopitalList[i].point <- point;
37
38
39          finpour
40
41          #tri par fusion avec une complexite O(n*log(n)) qui
    favorise les hopitaux avec des points plus petits
42          hopitalList.tri();
43
44  Fin
```

Listing 2.1: Algorithme de sélection des hôpitaux

— Exemple

Nom	Docteurs actifs	Patients en attente	Lits	Distance	Classement
Dr Benzerdjeb	5	1	56	12.64 Km	#1
Medjber Tami	6	1	219	48.13 Km	#2
CHU Usto	5	2	200	56.53 Km	#3
CHU Mustapha Pacha	7	1	549	402.27 Km	#4
EPH Bachir Mentouri	7	1	190	403.66 Km	#5
CHU Lamine Debaghine	5	2	240	403.25 Km	#6
EPH Azeffoun	6	1	317	522.56 Km	#7
EPH Ghardaia	7	1	101	546.87 Km	#8
El Bir	5	7	185	701.3 Km	#9
CHU Annaba	4	11	88	813.56 Km	#10
Aoulef El Arab	5	16	120	956.34 Km	#11

Table 2.1: Exemple de sélection d'hôpitaux

2.6 Conclusion

Dans ce chapitre, nous avons défini la problématique et l'objectif du travail en présentant l'étude de l'existant que ce soit localement ou à l'étranger, en détaillant l'approche que nous proposons comme solution au problème posé.

Chapitre 3

Conception et réalisation

3.1 Introduction

Dans ce chapitre nous présenterons la conception UML ainsi que les outils, langages et environnements de développement retenus pour la réalisation, le fonctionnement de l'application puis nous passerons en revue l'ensemble de ses interfaces.

3.2 Conception UML

Le langage de modélisation unifié (UML) est un langage de modélisation polyvalent et évolutif dans le domaine de l'ingénierie logicielle, destiné à fournir un moyen standard de visualiser la conception d'un système. [20]

La création d'UML a été motivée à l'origine par le désir de normaliser les systèmes de notation et les approches disparates de la conception de logiciels.

Les diagrammes UML représentent deux vues différentes du modèle d'un système :

— Vue statique (ou structurelle)
met l'accent sur la structure statique du système en utilisant des objets, des attributs, des opérations et des relations. Elle inclut les diagrammes de classes.

— Vue dynamique (ou comportementale)
met l'accent sur le comportement dynamique du système en montrant les collaborations entre les objets et les changements des états internes des objets. Cette vue inclut les diagrammes de séquence.

— **Diagramme de cas d'utilisation**
Un diagramme de cas d'utilisation est une représentation graphique des interactions possibles d'un utilisateur avec un système. Un diagramme de cas d'utilisation montre différents cas d'utilisation et différents types d'utilisateurs du système et sera souvent accompagné par d'autres types de diagrammes. Les cas d'utilisation sont représentés par des cercles ou des ellipses. Les acteurs sont souvent représentés par des figures en stick. [21]
Le diagramme est présenté dans la figure 3.1.

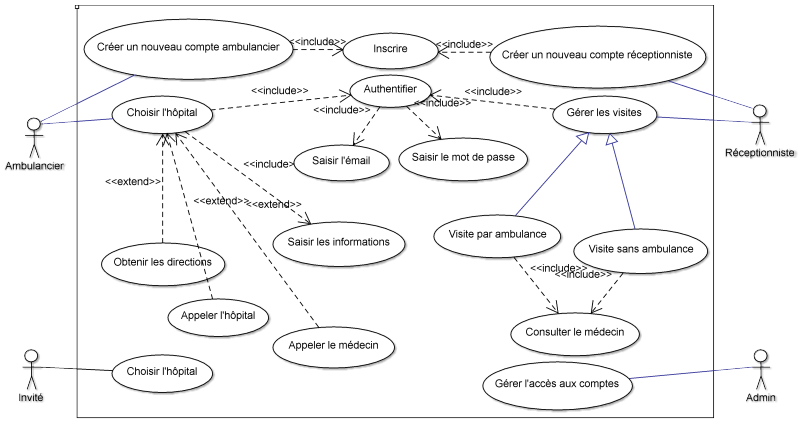

Figure 3.1: Diagramme de cas d'utilisation générale de l'application

— **Diagramme de classe**
Le diagramme de classes est la principale brique de la modélisation orientée objet. Il est utilisé pour la modélisation conceptuelle générale de la structure de l'application, et pour la modélisation détaillée, en traduisant les modèles en code de

programmation. Les diagrammes de classes peuvent également être utilisés pour la modélisation des données.
Les classes dans un diagramme de classes représentent à la fois les éléments principaux, les interactions dans l'application, et les classes à programmer. [22]
Les diagrammes sont présentés dans les figures 3.2 et 3.3.

— **La sous-application ambulancier**

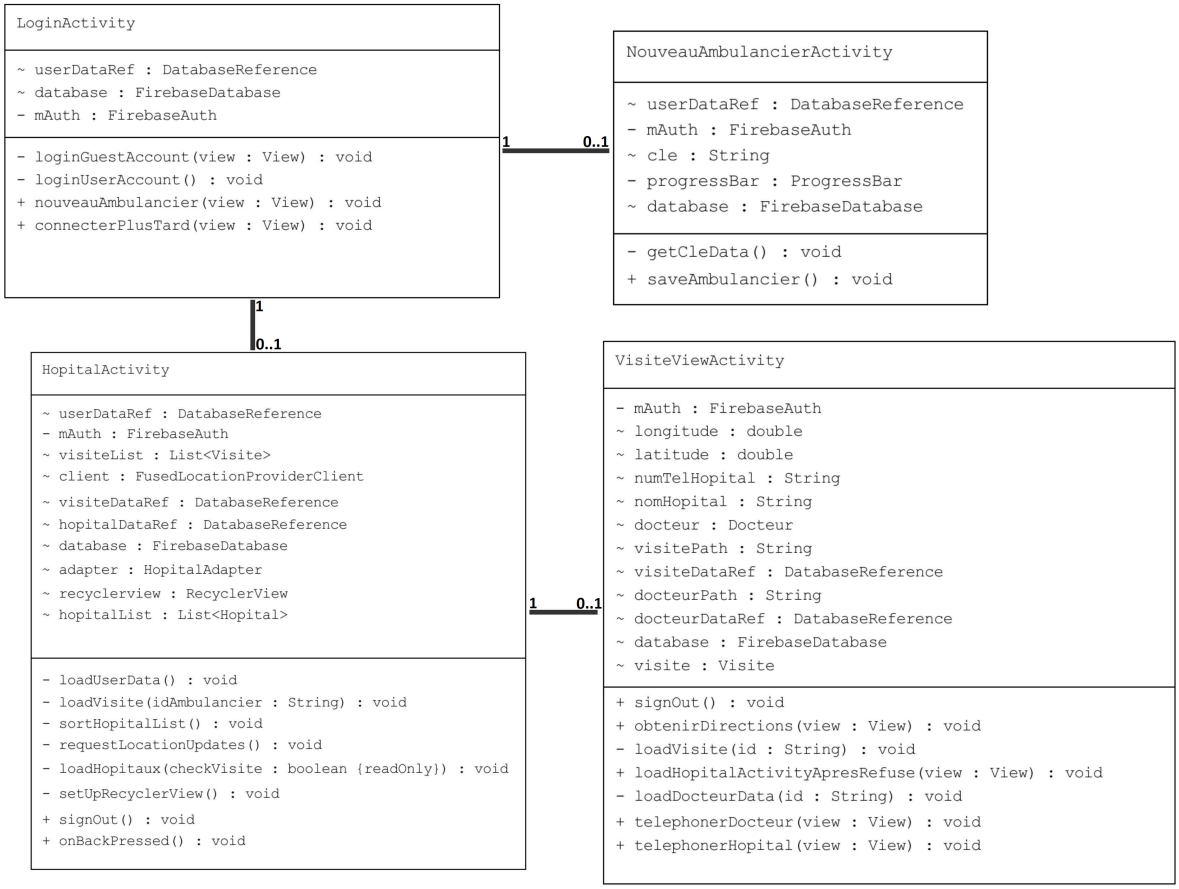

Figure 3.2: Diagramme de classe "ambulancier"

— **La sous-application de gestion de l'hôpital**

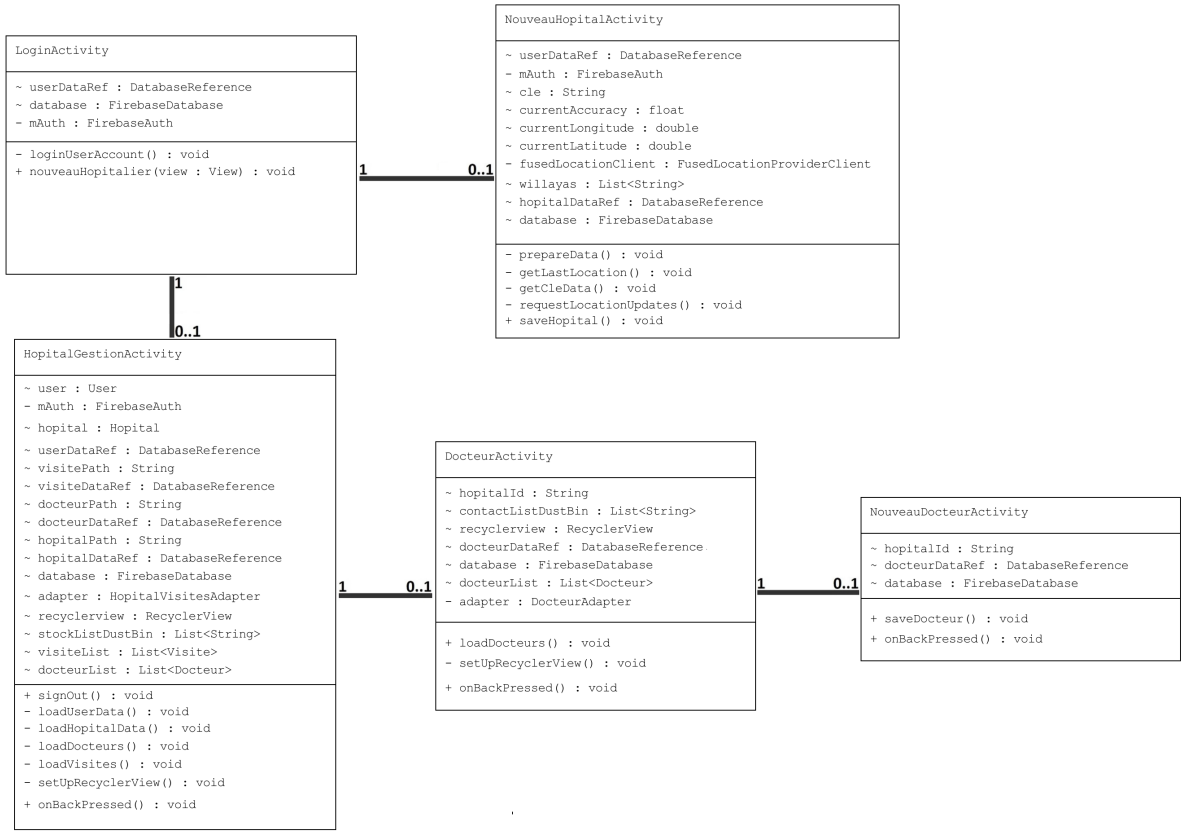

Figure 3.3: Diagramme de classe "gestion de l'hôpital"

— **Diagramme de sequence**
Un diagramme de séquence montre les interactions entre les objets dans une séquence temporelle. Il décrit les objets impliqués dans le scénario et la séquence des messages échangés entre les objets nécessaires à la réalisation de la fonctionnalité du scénario. Les diagrammes de séquence sont généralement associés à des réalisations de cas d'utilisation dans la vue logique du système en cours de développement. Les diagrammes de séquence sont parfois appelés diagrammes d'événements ou scénarios d'événements. [23]
Les diagrammes sont présentés dans les figures 3.4, 3.5, **??** et 3.6.
— **La sous-application ambulancier**
— Diagramme de séquence du cas "s'authentifier"

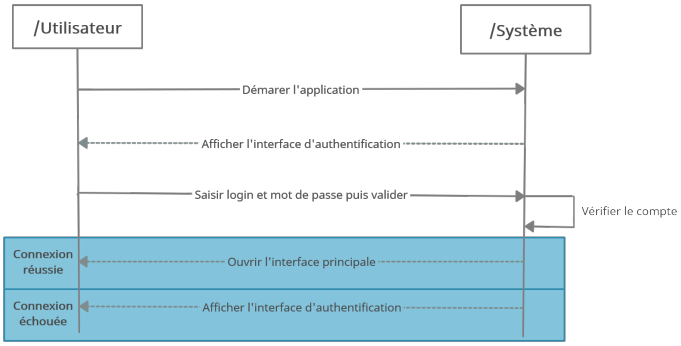

Figure 3.4: Diagramme de séquence du cas "s'authentifier"

— Diagramme de séquence du cas "visite par ambulance"

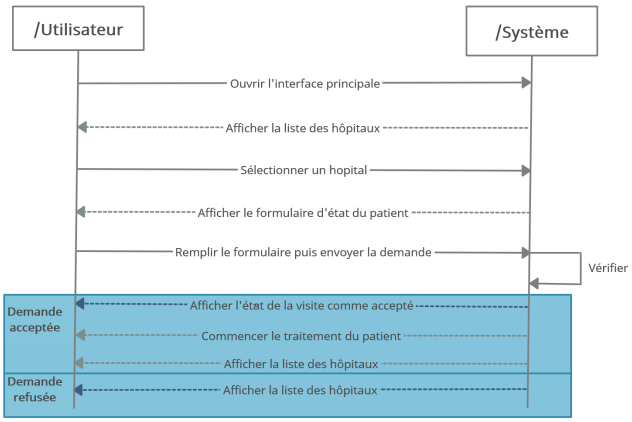

Figure 3.5: Diagramme de séquence du cas "visite par ambulance"

— **La sous-application de gestion de l'hôpital**
— Diagramme de séquence du cas "visite sans ambulance"

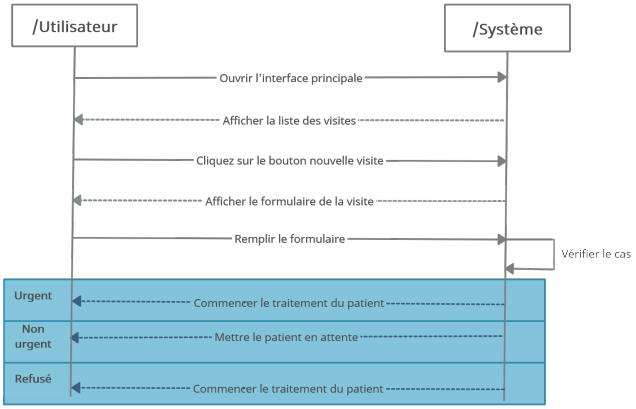

Figure 3.6: Diagramme de séquence du cas "visite sans ambulance"

3.3 Réalisation

3.3.1 Introduction

La réalisation représente l'étape qui suit immédiatement la phase de conception. C'est l'aboutissement final de notre projet. La réalisation de notre application se répartie en deux (02) sous-applications principales :

- **La sous-application de gestion de l'hôpital**
  Elle permettra aux personnel de l'hôpital de gérer les visites des patients que ce soit qu'ils viens via l'ambulance ou autrement.
- **La sous-application ambulancier**
  Elle sera installée sur le périphérique mobile des ambulancies.

3.3.2 Choix des outils et langages de développement

- **Java**
  Java est l'un des outils idéaux de programmation polyvalente, avancée et orientée objet. Il intègre les concepts les plus intéressants des technologies informatiques récentes dans une plate-forme de développement riche et homogène. Nous avons choisi le langage Java pour toutes ses caractéristiques qui ont largement contribué à son énorme succès, à savoir :
  - **Programmation orientée objet**
    Java adopte la programmation orientée objet (POO) - un concept de codage dans lequel le développeur définit non seulement le type de données et leur structure, mais aussi l'ensemble des fonctions qui leur sont appliquées. De cette façon, la structure des données devient un objet qui peut désormais être manipulé pour créer des relations entre différents objets.
  - **Langage de haut niveau avec une syntaxe simple et une courbe d'apprentissage moyenne**
    Java est un langage de haut niveau, ce qui signifie qu'il ressemble beaucoup au langage humain. Contrairement aux langages de bas niveau qui ressemblent au code machine, les langages de haut niveau doivent être convertis à l'aide de compilateurs ou d'interpréteurs. Cela simplifie le développement, en rendant un langage plus facile à écrire, à lire et à maintenir.
  - **Protection contre les failles de sécurité**
    Certaines des caractéristiques de Java peuvent protéger contre les failles de sécurité courantes. Tout d'abord, par rapport au C, Java n'a pas de pointeurs. Un pointeur est un objet qui stocke l'adresse mémoire d'une autre valeur, ce qui peut entraîner un accès non autorisé à la mémoire. Deuxièmement, il possède un gestionnaire de sécurité, une politique de sécurité créée pour chaque application où le développeur peut spécifier des règles d'accès. Cela lui permet d'exécuter les applications Java dans un " sandbox ", éliminant ainsi les risques de nuisance.
  - **Indépendance vis-à-vis de la plate-forme**
    Cela signifie qu'un programme Java peut être créé sous Windows, compilé en bytecode, puis l'application peut être exécutée sur toute autre plate-forme prenant en charge une machine virtuelle Java (JVM). Dans ce cas, une JVM sert de niveau d'abstraction entre le code et le matériel.
  - **Gestion automatique de la mémoire**

Les développeurs Java ne doivent pas se soucier d'écrire manuellement du code pour les tâches de gestion de la mémoire grâce à la gestion automatique de la mémoire (AMM) et au garbage collection, une application qui gère automatiquement l'allocation et la désallocation de la mémoire.

— **Multithreading**
En programmation, un thread est la plus petite unité de traitement. Pour maximiser l'utilisation du temps CPU, Java permet d'exécuter ces threads simultanément - dans un processus appelé multithreading.
Les threads partagent la même zone de mémoire, de sorte que le passage d'un thread à l'autre prend peu de temps. Ils sont également indépendants, de sorte que si un thread fait face à une exception, cela n'affecte pas les autres threads.

— **Stabilité et communauté massive**
Java est un langage de programmation très populaire grâce à la communauté, au soutien d'Oracle et à la multitude d'applications et de langages qui tournent sur la JVM. En outre, de nouvelles versions de Java sont régulièrement publiées avec des fonctionnalités nouvelles et intéressantes.
Java dispose d'un écosystème extrêmement vaste de bibliothèques et de frameworks bien testés pour tous les cas d'utilisation. Java est très probablement l'un des premiers langages que les développeurs débutants rencontrent au cours de leurs études, car il existe plus de 1000 cours liés à Java sur Udemy et plus de 300 sur Coursera. [24]

— **Firebase**
Firebase est une plateforme de développement logiciel lancée en 2011 par Firebase inc, et rachetée par Google en 2014. Commencée comme une base de données en temps réel, elle compte désormais 14 services (dont 3 actuellement en version bêta), et des API dédiées. L'ensemble de la plateforme est une solution Backend-as-a-Service à la fois pour les applications mobiles et web qui comprend des services pour la création, le test et la gestion des applications.
Les solutions BaaS permettent au développeur d'éliminer la nécessité de gérer des bases de données dorsales et d'obtenir le matériel correspondant. Au lieu de cela, il peut les intégrer à son application via des API dédiées à chaque service distinct. Dans le cas de Firebase, il en existe 7 qui couvrent tout le spectre des technologies de back-end pour une application. La liste des plateformes avec lesquelles Firebase s'intègre comprend Android, iOS, Web et Unity.
Mais, il y a beaucoup plus à apprendre sur la plate-forme, car elle comprend divers services pour travailler avec un back-end géré. Dans la section suivante, nous donnerons un aperçu général des deux services de la plateforme Firebase que nous avons utilisés dans notre application mobile. [25]

— **Firebase Realtime Database**
Firebase Realtime Database a été le premier produit à apparaître sous le drapeau de Firebase, il s'agit donc du service le plus établi et le plus stable de toute la plateforme. Realtime Database est essentiellement un stockage en cloud NoSQL qui peut être connecté à l'application pour fournir un accès en temps réel aux données sur différentes plateformes. L'un des avantages est que la base de données peut fonctionner hors ligne, en mettant en cache les données dans la mémoire du dispositif, et après s'être reconnecté à l'internet, en les synchronisant.
Les données sont stockées en JSON et peuvent être interrogées par les utilisa-

teurs. En termes de sécurité, Realtime Database fournit un accès aux données basé sur les autorisations. Cela peut être fait avec l'aide de Firebase Authentication, et en donnant des autorisations par identité d'utilisateur ou par règles de sécurité.

- **Firebase Authentication**
  Firebase Authentication est une fonctionnalité d'authentification Google adaptée aux applications utilisant Firebase. Elle permet d'utiliser une interface utilisateur prédéfinie ou personnalisée pour l'authentification des utilisateurs, et de connecter les utilisateurs via des informations d'identification personnalisées, des e-mails ou des médias sociaux.

- **NoSQL**
  Comme ces bases de données ne sont pas limitées à une structure de table, elles sont appelées NoSQL. Ce type de système de gestion de base de données est considéré comme orienté vers les documents et souvent appelé bases de données non relationnelles. Les données non structurées telles que les articles, les photos, les vidéos et autres sont rassemblées dans un seul document. Les données sont simples à interroger mais ne sont pas toujours classées en lignes et colonnes comme dans une base de données relationnelle. Les bases de données non relationnelles ou NoSQL sont généralement mises à l'échelle horizontalement en ajoutant des serveurs. [26]

- **Fused Location Provider API**
  Les applications peuvent tirer parti des signaux fournis par plusieurs capteurs de l'appareil pour déterminer sa position. Cependant, il n'est pas simple de choisir la bonne combinaison de signaux pour une tâche spécifique dans différentes conditions. Trouver une solution qui soit également économe en batterie est encore plus compliqué.
  Le fournisseur de localisation fusionné est une API de localisation dans les services Google Play qui combine intelligemment différents signaux pour fournir les informations de localisation dont l'application a besoin.
  Le fournisseur de localisation fusionné gère les technologies de localisation sous-jacentes, telles que le GPS et le Wi-Fi, et fournit une API simple qui permet de spécifier la qualité de service requise. Par exemple, le développeur peut demander les données les plus précises disponibles, ou la meilleure précision possible sans consommation d'énergie supplémentaire. [27]
  - **Prise en charge des scénarios de localisation courants**
    - Dernière localisation connue
      Grâce à l'API de fournisseur de localisation fusionnée, l'application peut demander le dernier emplacement connu de l'appareil de l'utilisateur. L'obtention du dernier emplacement connu est généralement un bon point de départ pour les applications qui ont besoin d'informations de localisation.
    - Paramètres de localisation
      Lors de la demande d'informations de localisation, de nombreuses sources de localisation différentes, telles que le GPS et le Wi-Fi, sont utilisées. Il peut être difficile de décider quelles sources utiliser, mais l'API de fournisseur de localisation fusionnée élimine les approximations en modifiant automatiquement les paramètres système appropriés. Tout ce que l'application doit faire, c'est spécifier le niveau de service souhaité.
    - Mises à jour des localisations

Outre le dernier emplacement connu, l'API du fournisseur d'emplacement fusionné peut fournir des mises à jour de l'emplacement à un rappel dans l'application à des intervalles spécifiques. Le développeur peut spécifier l'intervalle souhaité comme un paramètre de la qualité de service. En utilisant les mises à jour de la localisation, l'application peut fournir des informations supplémentaires telles que la direction et la vitesse.

3.3.3 Choix de l'environnement de développement

— **Android studio**
Android Studio est l'environnement de développement intégré (IDE) officiel du système d'exploitation Android de Google, construit sur le logiciel IntelliJ IDEA de JetBrains et conçu spécifiquement pour le développement Android. Il est disponible en téléchargement sur les systèmes d'exploitation basés sur Windows, macOS et Linux ou sous forme de service par abonnement en 2020.Il remplace Eclipse Android Development Tools (E-ADT) comme principal IDE pour le développement d'applications Android natives.
Android Studio a été annoncé le 16 mai 2013 lors de la conférence Google I/O. Il était en phase de preview en accès anticipé à partir de la version 0.1 en mai 2013, puis est entré en phase bêta à partir de la version 0.8 qui a été publiée en juin 2014.La première build stable a été publiée en décembre 2014, à partir de la version 1.0.
Le 7 mai 2019, Kotlin a remplacé Java comme langage préféré de Google pour le développement d'applications Android Java est toujours pris en charge, tout comme C++. [28]

— **Android SDK**
Le SDK Android (Software Development Kit) est un ensemble d'outils de développement utilisés pour développer des applications pour la plate-forme Android. Ce SDK fournit une sélection d'outils nécessaires à la création d'applications Android et veille à ce que le processus se déroule aussi bien que possible. Que le développeur crée une application en utilisant Java, Kotlin ou C#, il a besoin du SDK pour la faire fonctionner sur n'importe quel appareil Android. Il peut également utiliser un émulateur afin de tester les applications qu'il a créées.
Aujourd'hui, le SDK Android est également fourni avec Android Studio, l'environnement de développement intégré qui permet de travailler et de gérer au mieux de nombreux outils. [29]
Le SDK Android comprend les éléments suivants :
— Bibliothèques requises.
— Un débogueur.
— Un émulateur.
— Une documentation pertinente pour les interfaces de programme d'application (API) Android.
— Des exemples de code source.
— Des tutoriels pour le système d'exploitation Android.

3.3.4 Captures d'interfaces de l'application

— **L'application ambulancier**

— Interface d'authentification
Cette interface permet à l'utilisateur, soit de se authentifier ou créer un nouveau compte en cliquant sur le bouton "Nouveau ambulancier" s'il s'agit d'un ambulancier, soit d'aller vers l'interface d'accueil invité en cliquant le button "Se connecter plus tard" s'il s'agit d'un utilisateur invité.

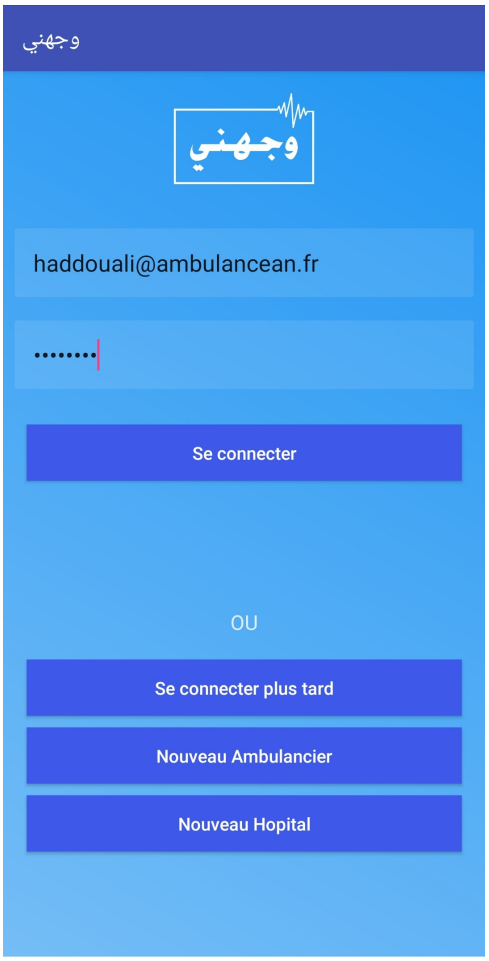

Figure 3.7: Interface d'authentification

— Interface nouveau ambulancier
Cette interface permet de créer un nouveau compte ambulancier.

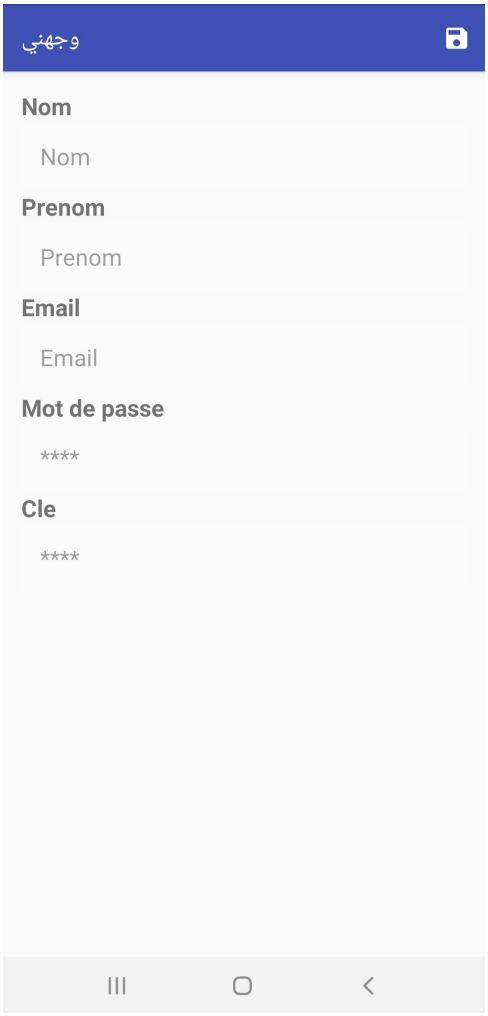

Figure 3.8: Interface nouveau ambulancier

— Interface d'accueil de l'invité
Cette interface permet d'afficher la liste des unités d'urgence proches sans la possibilité de réserver une visite à l'hôpital.

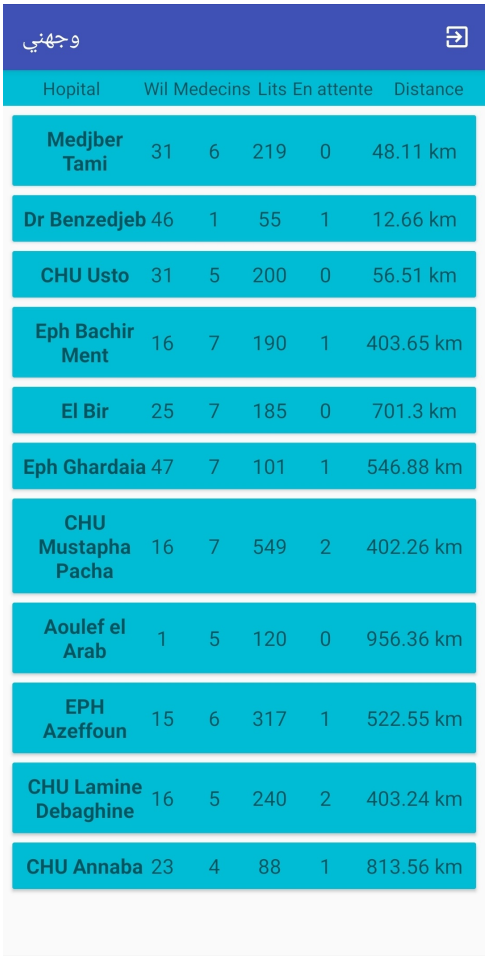

Figure 3.9: Interface d'accueil de l'invité

— Interface d'accueil de l'ambulancier
Cette interface est la principale, elle permet d'afficher la liste des unités d'urgence proches de l'ambulance, triées selon plusieurs caractéristiques.

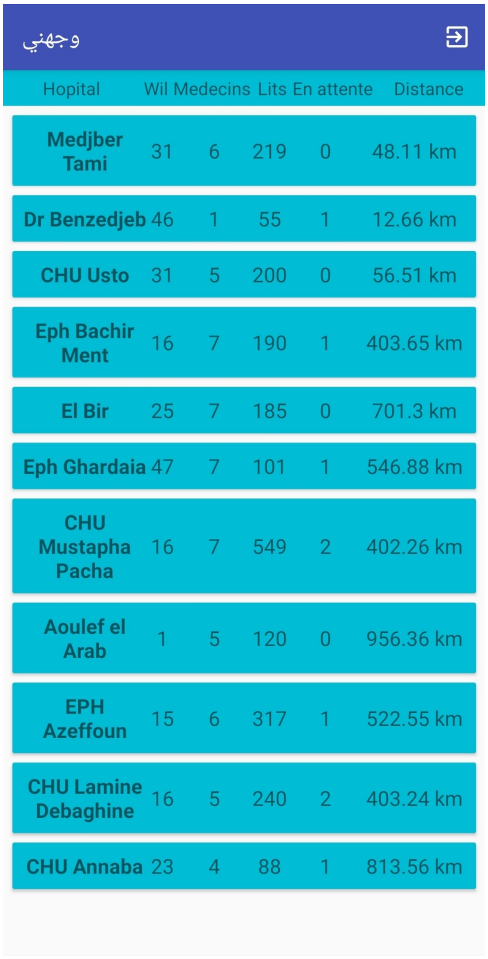

Figure 3.10: Interface d'accueil de l'ambulancier

— Interface de remplissage des données de la visite
Après avoir choisi l'unité d'urgence une fenêtre s'ouvre, cette fenêtre permet à l'ambulancier de remplir un formulaire sur le cas d'urgence du patient, et l'envoyer au personnel médical responsable de l'hôpital sélectionné.

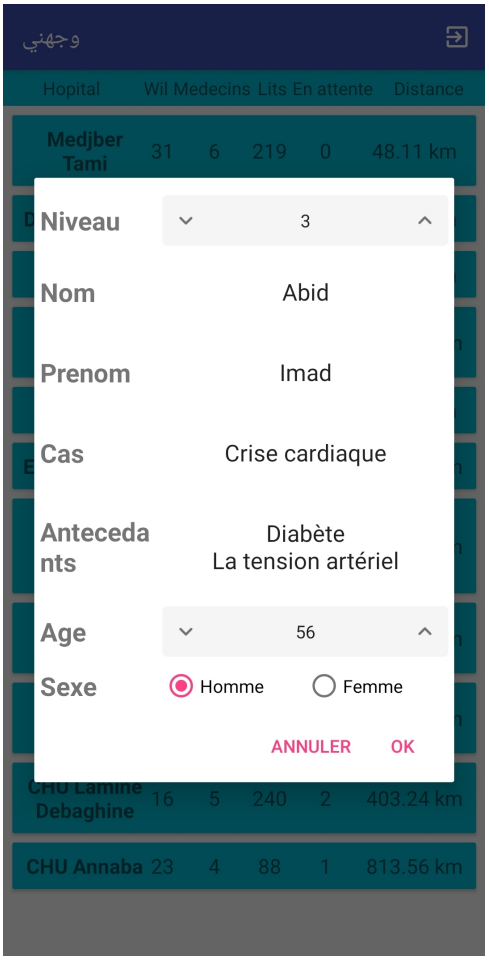

Figure 3.11: Interface de remplissage des données de la visite

— Interface l'état de la demande
Dans cette interface, l'ambulancier suivra l'état de sa demande à l'hôpital, qu'elle soit acceptée ou refusée. Avec la possibilité d'appeler l'hôpital et de se diriger vers lui.

Figure 3.12: Interface l'état de la demande

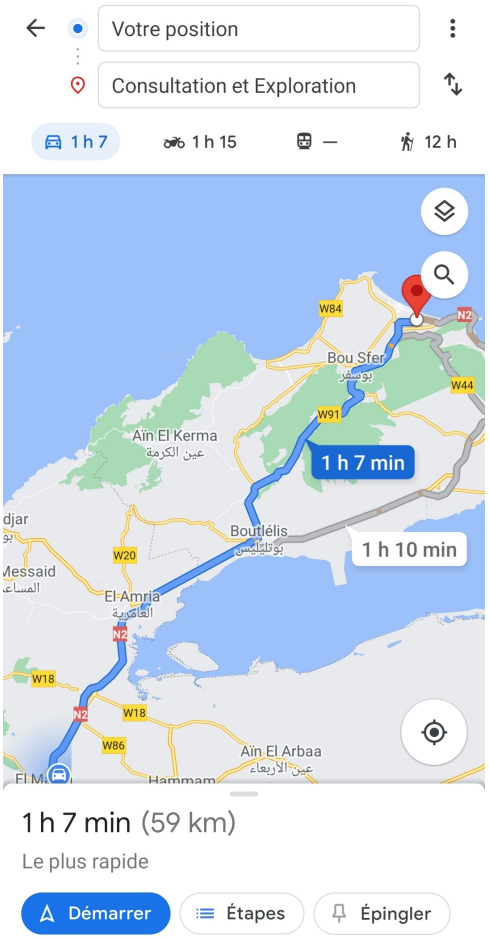

Figure 3.13: Interface du trajet vers l'hôpital

— Interface de la sous-application de gestion de l'hôpital
Après que l'ambulancier a envoyé les données du patient à l'hôpital, le personnel médical décide d'accepter ou de refuser le cas.

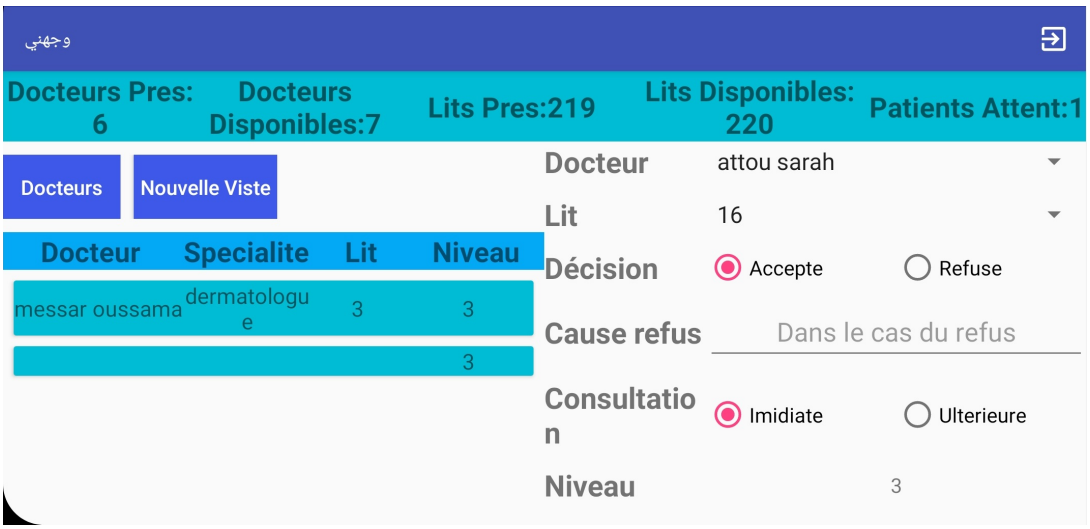

Figure 3.14: Interface de la sous-application de gestion de l'hôpital

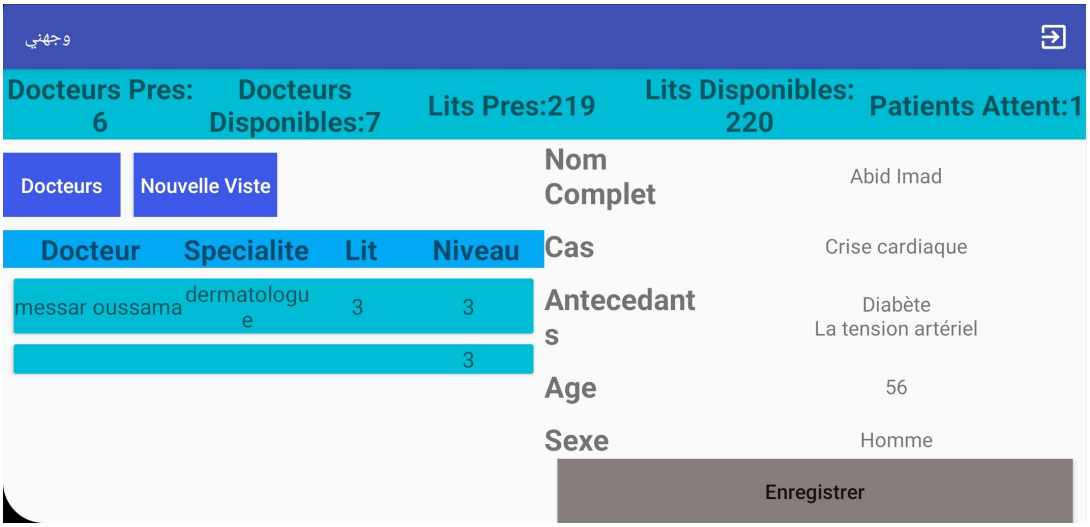

Figure 3.15: Interface de la sous-application de gestion de l'hôpital

— Interface de l'état de demande (acceptée)
Une fois que le personnel médical a accepté le cas, l'ambulancier peut également appeler directement le médecin responsable.

Figure 3.16: Interface de l'état de demande (accepte)

— Interface de la sous-application de gestion de l'hôpital

Après l'arrivée de l'ambulance, le personnel médical commencera le traitement du patient, à ce moment le transport du patient par l'ambulancier prendra fin et il pourra transporter de nouveaux patients.

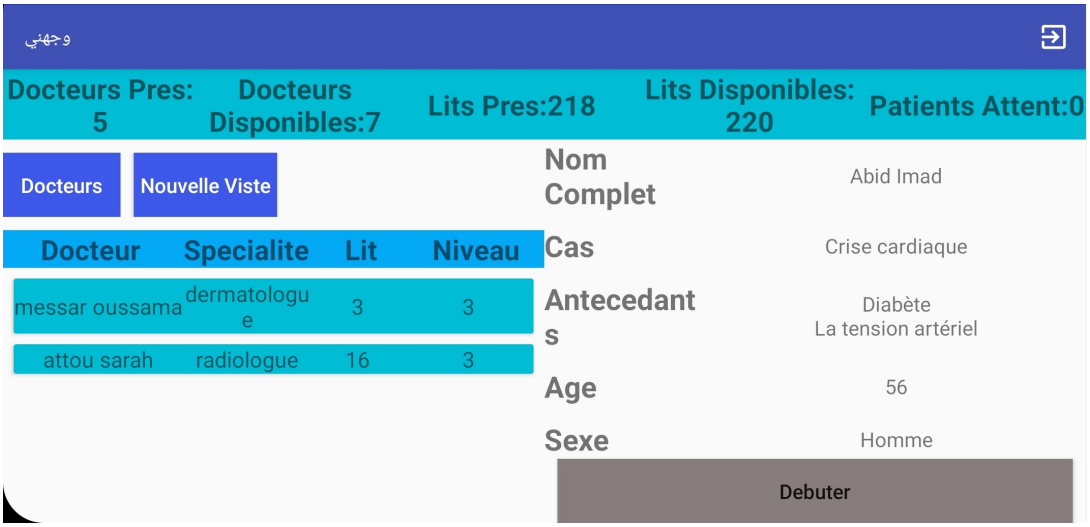

Figure 3.17: Interface de la sous-application de gestion de l'hôpital

— **La sous-application de gestion de l'hôpital**
  — Interface d'authentification
  Cette interface permet au personnel de l'hôpital responsable sur la gestion des visites, soit de se authentifier, soit de créer un nouveau compte en cliquant sur le bouton "Nouvel hôpital".

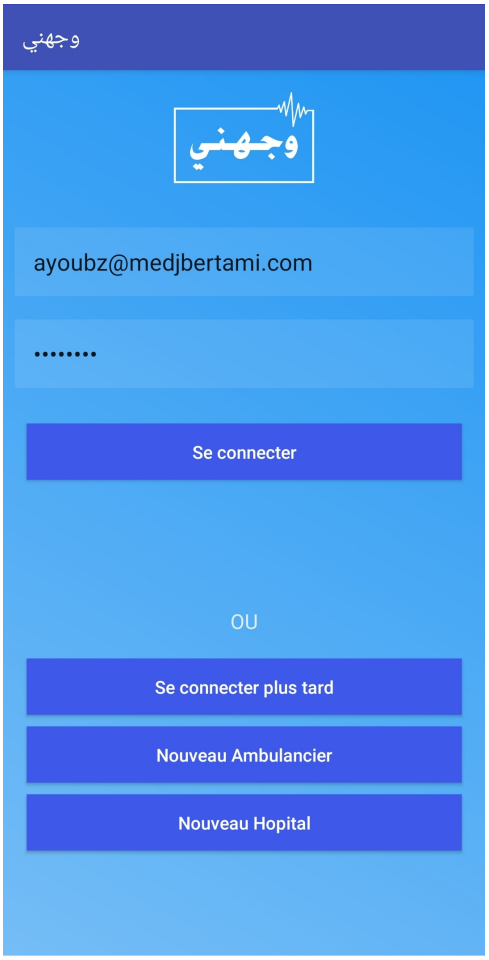

Figure 3.18: Interface d'authentification

— Interface Nouvel hôpital
Cette interface permet de créer un nouvel hôpital.

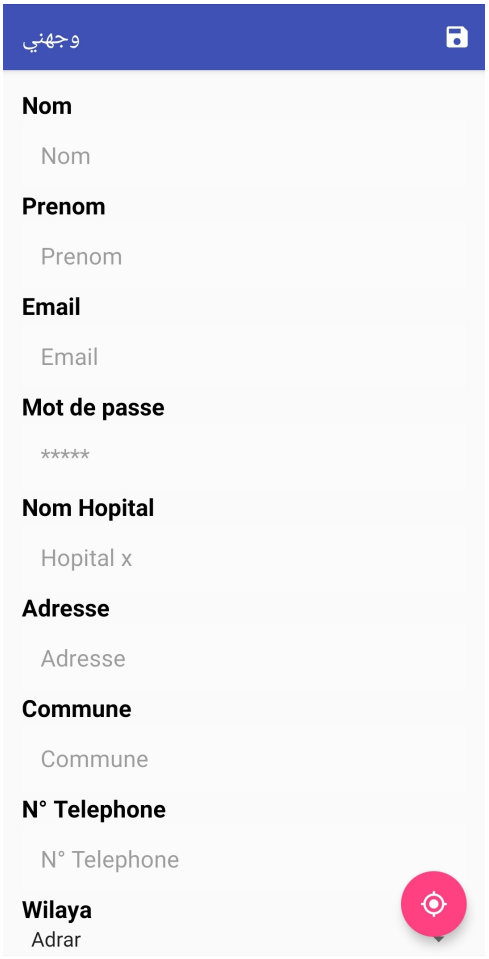

Figure 3.19: Interface Nouvel hôpital

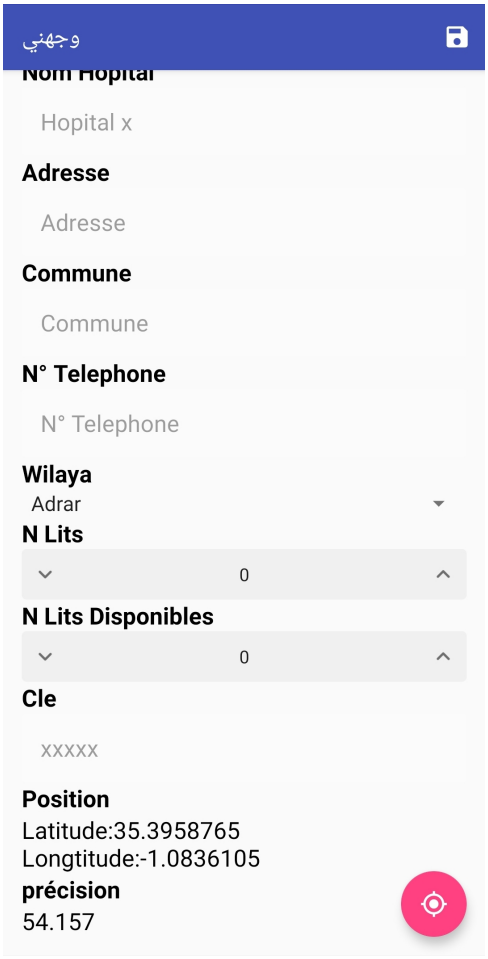

Figure 3.20: Interface Nouvel hôpital

— Interface liste des docteurs
Cette interface permet d'afficher la liste des docteurs qui travaillent dans l'hôpital.

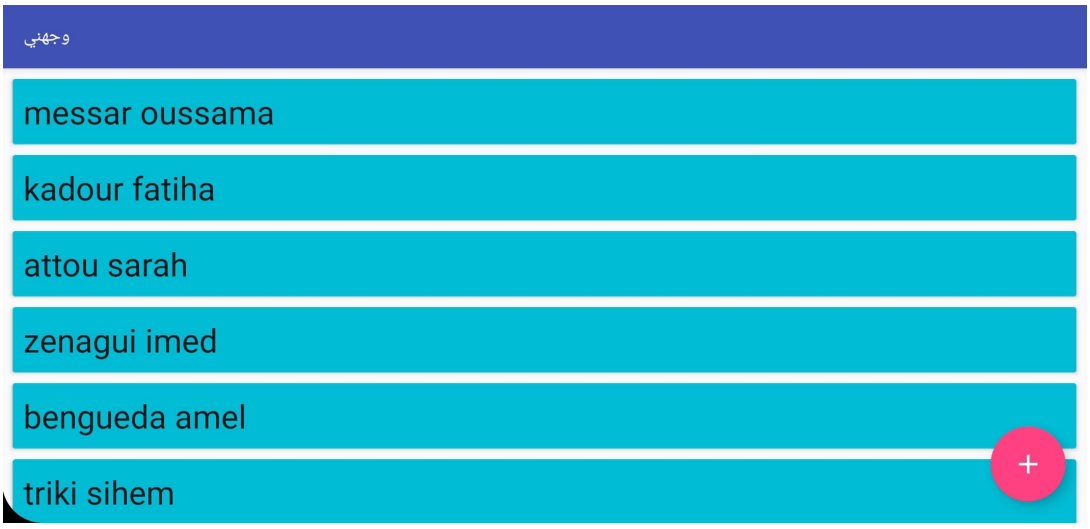

Figure 3.21: Interface liste des docteurs

— Interface nouveau docteur

Cette interface permet de créer un nouveau docteur.

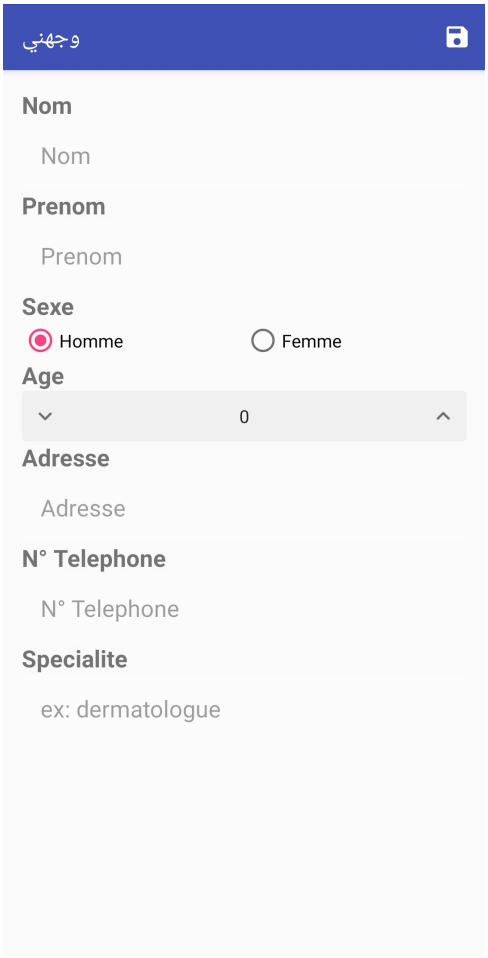

Figure 3.22: Interface nouveau docteur

— Interface d'accueil de la sous-application de gestion de l'hôpital
Cette interface est la principale, elle permet de gérer les visites des patients venant en ambulance ou sans ambulance.
Les patients qui arrivent aux urgences sans ambulance peuvent être enregistrés dans le système pour organiser le flux des patients et fournir des données en direct.

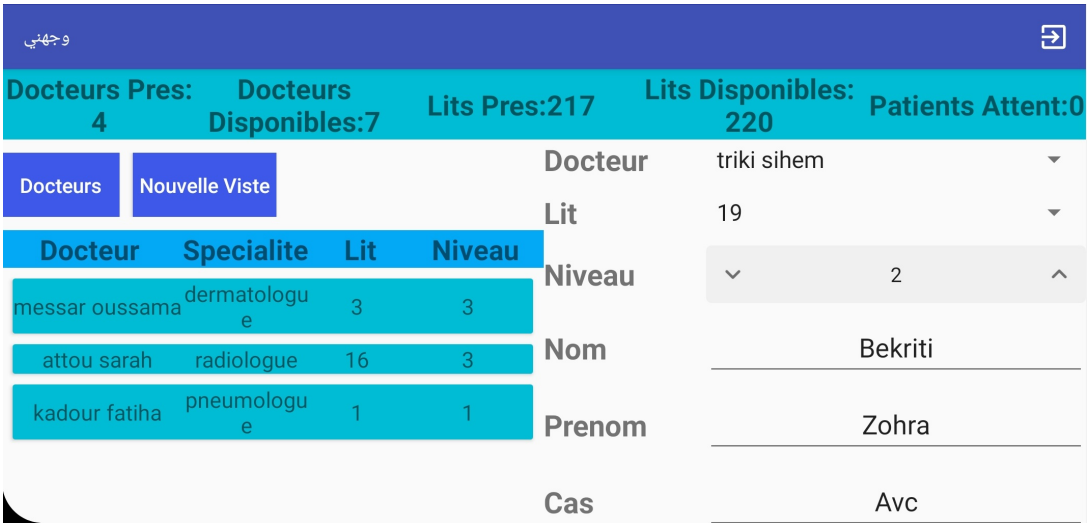

Figure 3.23: Interface d'accueil de la sous-application de gestion de l'hôpital

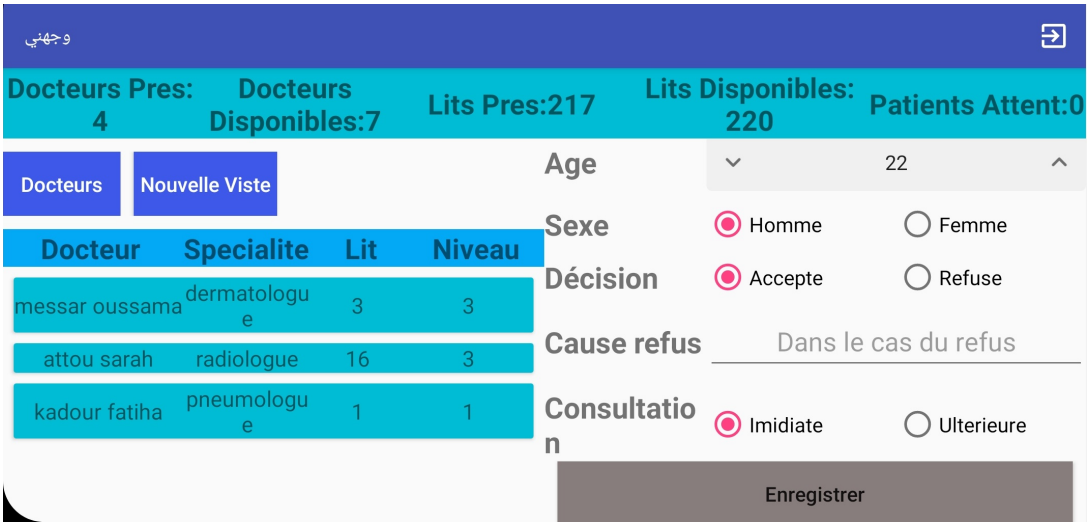

Figure 3.24: Interface d'accueil de la sous-application de gestion de l'hôpital

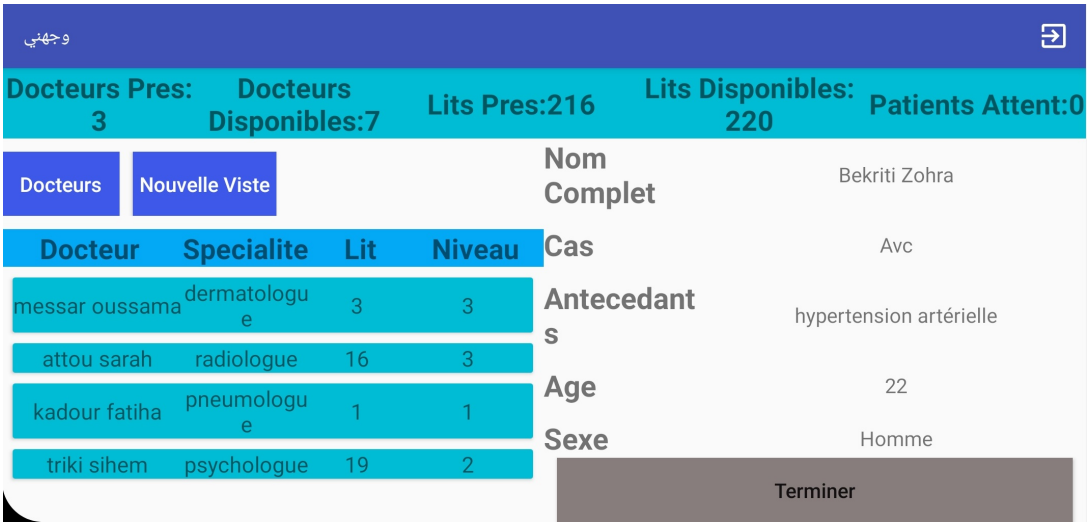

Figure 3.25: Interface d'accueil de la sous-application de gestion de l'hôpital

3.4 Conclusion

Dans ce chapitre, nous avons présenté la conception UML de notre application Android, y compris les diagrammes de cas d'utilisation, de classe et de séquence. Nous avons également donné les définitions des outils de développement et des environnements utilisés pour construire l'application.

Finalement, des aperçus des activités sont présentés afin de décrire l'interface utilisateur de l'application réalisée.

Conclusion générale

Ce projet nous a permis de concevoir, développer et intégrer une application mobile sous la plateforme Android qui a fortement occupé le marché des terminaux mobiles.

La réalisation de ce travail est organisée en trois phases, nous avons commencé par une étude préalable de notre projet au cours de laquelle nous avons traité d'une façon générale les différentes architectures, systèmes et technologies pour le développement des applications mobiles et d'une façon détaillé le système Android en étudiant son architecture, ses versions et ses outils de développements.

La deuxième phase concerne la problématique et l'objectif du travail en présentant l'étude de l'existant que ce soit en Algérie ou à l'étranger, et de manière détaillée la solution proposée.

La troisième phase concerne la conception et la réalisation de notre application, nous avons utilisé le langage de modélisation UML pour la partie modélisation et l'environnement de développement Android Studio pour la réalisation.

Enfin, nous avons réussi à créer un outil simple et efficace qui améliore la communication entre les services de protection civile et les hôpitaux afin d'éviter les complications ainsi que les pertes humaines.

L'outil réalisé est très satisfaisant. Néanmoins, plusieurs perspectives sont encore possibles, par exemple faire une gestion générale de l'hôpital avec une automatisation notamment en ce qui concerne les ressources hospitalières à l'aide de l'intelligence artificielle, tenir compte du délai de la visite dans le choix de l'hôpital, gérer les médecins en garde ou encore localiser le patient en temps réel afin de réduire le temps de réponse des services de protection civile.

Bibliographie

Bibliographie

[1] 11 Powerful Examples of Responsive Web Design, May 2021. [Online ; accessed 27. May 2021].

[2] 12 Best Examples of Progressive Web Apps (PWAs) in 2021, Apr 2021. [Online ; accessed 27. May 2021].

[3] Siddharth Sharma. Native vs Hybrid vs Cross-Platform — What To Choose ? *Medium*, Aug 2020.

[4] Apple iOS Architecture, May 2021. [Online ; accessed 25. May 2021].

[5] Android Operating System : Introduction, Features & Its Applications, Mar 2021. [Online ; accessed 27. May 2021].

[6] Contributors to Wikimedia projects. Android version history - Wikipedia, May 2021. [Online ; accessed 27. May 2021].

[7] Wikipedia. Web application — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Web%20applicationoldid=1023808310, 2021. [Online ; accessed 24-May-2021].

[8] Responsive Web Design and Progressive Web App (PWA) : The Differences - SimiCart, Apr 2021. [Online ; accessed 24. May 2021].

[9] Wikipedia. Mobile app — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Mobile%20appoldid=1017884699, 2021. [Online ; accessed 24-May-2021].

[10] Wikipedia. IOS — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=IOSoldid=1024609361, 2021. [Online ; accessed 24-May-2021].

[11] iOS Introduction | IOS Tutorial | Intellipaat.com, May 2021. [Online ; accessed 24. May 2021].

[12] Wikipedia. Android (operating system) — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Android%20(operating%20system)oldid=1023978311 2021. [Online ; accessed 27-May-2021].

[13] Contributors to Wikimedia projects. List of features in Android - Wikipedia, Apr 2021. [Online ; accessed 27. May 2021].

[14] Top Programming Languages for Android App Development - GeeksforGeeks, Mar 2021. [Online ; accessed 25. May 2021].

[15] Ambulance Diversion | Health Affairs Brief, May 2021. [Online ; accessed 26. May 2021].

[16] Doctr - Accès aux soins de santé sans stress, Oct 2020. [Online ; accessed 26. May 2021].

[17] Wikipedia. Ambulance — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Ambulanceoldid=1027696783, 2021. [Online ; accessed 12-June-2021].

[18] Wikipedia. Hospital — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Hospitaloldid=1025544759, 2021. [Online ; accessed 12-June-2021].

[19] Wikipedia. Emergency department — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Emergency%20departmentoldid=1027069440, 2021. [Online ; accessed 12-June-2021].

[20] Wikipedia. Unified Modeling Language — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Unified%20Modeling%20Languageoldid=1020625438 2021. [Online ; accessed 05-June-2021].

[21] Wikipedia. Use case diagram — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Use%20case%20diagramoldid=1019341507, 2021. [Online ; accessed 13-June-2021].

[22] Wikipedia. Class diagram — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Class%20diagramoldid=1024380898, 2021. [Online ; accessed 05-June-2021].

[23] Wikipedia. Sequence diagram — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Sequence%20diagramoldid=1019345802, 2021. [Online ; accessed 05-June-2021].

[24] The Good and the Bad of Java Programming, May 2021. [Online ; accessed 30. May 2021].

[25] The Good and the Bad of Firebase Backend Services, May 2021. [Online ; accessed 31. May 2021].

[26] Comparing Database Management Systems : MySQL, PostgreSQL, MSSQL Server, MongoDB, Elasticsearch and others, May 2021. [Online ; accessed 31. May 2021].

[27] Fused Location Provider API | Google Developers, Aug 2017. [Online ; accessed 31. May 2021].

[28] Wikipedia. Android Studio — Wikipedia, the free encyclopedia. http ://en.wikipedia.org/w/index.php ?title=Android%20Studiooldid=1022911296, 2021. [Online ; accessed 31-May-2021].

[29] Android SDK Tutorial For Beginners | A Complete Guide | Edureka, Nov 2019. [Online ; accessed 31. May 2021].