A picture containing logo

Description automatically generated

DEPARTEMENT TECHNOLOGIES de l’Informatique

**M**inistère de l’**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

**D**irection **G**énérale des **É**tudes **T**echnologiques

**I**nstitut **S**upérieur des **É**tudes **T**echnologiques de **S**ousse

**المعهد العالي للدراسات التكنولوجية بسوسة**

Logo Société d’accueil

Électromécanique

N° 2022/

RAPPORT DE STAGE DE FIN D’ETUDES

Pour l’obtention du Diplôme de Licence

en Technologies de l’Informatique

Parcours : Multimédia et Développement Web

Intitulé :

Conception et Réalisation de la

Plateforme Pharmacie

Organisme d’accueil

Médical Innovation Technology

**Encadré par :**

Jalel Boughizane

Amel HARHIRA

**Elaboré par:**

Mouhamed Karim Zrouga

Soulaima Ben Chedly

Nom de l’Enseignant Encadrant :

Signature :

Nom de l’Encadrant Industriel :

Tampon & Sign. de l’Encadrant Ind. :

Mots-Clés :

Résumé :

Organisme d’accueil :

****

**Année Universitaire**

**2021– 2022**

Dédicaces

Du profond de mon cœur, je dédie ce travail à tous ceux qui me sont chers,

À mes chers parents, Que ce travail soit l’expression de ma reconnaissance pour vos sacrifices consentis, votre soutien moral et matériel que vous n’avez cesse de prodiguer. Vous avez tout fait pour mon bonheur et ma réussite. Qu ‘Allah vous préserve en bonne santé et vous accorde une longue vie.

À tous mes chers amis, pour le soutien que vous m’aviez offert, je vous dis MERCI.

Mouhamed karim zrouga

Soulaima ben chedly

Remerciement

C’est avec un grand plaisir que nous réservons cette page, en signe de gratitude et de profonde reconnaissance, à tous ceux qui nous ont aidé, de près ou de loin, à la réalisation de ce travail.

Nous tenons à exprimer nos sincères gratitudes et respects à notre encadrant Jalel Boughizane pour le sérieux de son encadrement, son soutien, et sa collaboration et pour les conseils et les orientations qu’il n’a cessé de nous prodiguer tout au long de l’élaboration de ce projet. Aux membres du jury : nous sommes très sensibles à l’honneur que vous nous faites en acceptant de juger notre soutenance.

Nous ne saurons enfin clore cette liste sans remercier notre institut supérieur des études technologiques de Sousse qui nous a favorisé cette occasion pour enrichir nos connaissances et nous a préparé à s’intégrer dans la vie professionnelle, et tous nos enseignants pour leur formation et leurs précieux conseils tout au long de nos années d’études à l’ISET.

Nous ne saurons enfin clore cette liste sans remercier notre Institut Supérieur des Études Technologiques de Sousse, qui nous a favorisé cette occasion pour enrichir nos connaissances et nous a préparé à s’intégrer dans la vie professionnelle, et tous nos enseignants pour leur formation et leurs précieux conseils tout au long de nos années d’études à l’ISET.

Mouhamed karim zrouga

Soulaima ben chedly

Tableau des matières

[Introduction Générale 1](#_Toc106007558)

[Chapitre 1 : Présentation du cadre de projet 3](#_Toc106007559)

[1. Présentation de l’organisme 3](#_Toc106007560)

[2. Présentation de projet 4](#_Toc106007561)

[2.1. Problématique 4](#_Toc106007562)

[2.2. Description de l’existant 4](#_Toc106007563)

[2.3. Critique de l’existant 4](#_Toc106007564)

[2.4. Solution proposée 5](#_Toc106007565)

[3. Conclusion 6](#_Toc106007566)

[Chapitre 2 : Etude préalable 7](#_Toc106007567)

[1. Identification des acteurs et des fonctionnalités attendues 7](#_Toc106007568)

[1.1. Les besoins fonctionnels 8](#_Toc106007569)

[1.2. Les besoins non fonctionnels 9](#_Toc106007570)

[1.3. Diagramme de cas d’utilisation global 9](#_Toc106007571)

[1.4. Diagramme de Classe 11](#_Toc106007572)

[1.5. Dictionnaire de données 12](#_Toc106007573)

[1.6. Modèle relationnel 14](#_Toc106007574)

[1.7. Schéma relationnel 16](#_Toc106007575)

[2. Méthodologie 16](#_Toc106007576)

[2.1. Méthodologies de travail: 16](#_Toc106007577)

[2.2. presentation du Scrum: 16](#_Toc106007578)

[2.3. Répartitions des rôles dans Scrum 17](#_Toc106007579)

[3. Pilotage du projet avec Scrum 18](#_Toc106007580)

[3.1. User Stories 18](#_Toc106007581)

[3.2. Product Backlog 18](#_Toc106007582)

[3.3. Enchainement des Sprints 20](#_Toc106007583)

[4. Conclusion 21](#_Toc106007584)

[Chapitre 3 : Etude Théorique 22](#_Toc106007585)

[1. Les Patterns 22](#_Toc106007586)

[2. Application Web et mobile 23](#_Toc106007587)

[2.1. Application Web 23](#_Toc106007588)

[2.2. Les principes de la partie Frontend (Angular) 24](#_Toc106007589)

[2.3. Application Mobile 27](#_Toc106007590)

[2.4. Application mobile Android 27](#_Toc106007591)

[2.5. Les Notions d’Android 32](#_Toc106007592)

[2.6. Types des applications mobile 34](#_Toc106007593)

[3. Les principes de la partie backend (Spring Boot) 36](#_Toc106007594)

[4. Communication client et service 38](#_Toc106007595)

[4.1. Web Services 38](#_Toc106007596)

[4.2. Communication angular et spring boot 39](#_Toc106007597)

[4.3. Communication Android et Spring Boot 40](#_Toc106007598)

[5. Google Cloud Platform services 41](#_Toc106007599)

[5.1. Google Authentication service 41](#_Toc106007600)

[5.2. API Google Maps 41](#_Toc106007601)

[6. Google AdMob 44](#_Toc106007602)

[7. Déploiement 46](#_Toc106007603)

[8. Conclusion 48](#_Toc106007604)

[Chapitre 4 : Mise en œuvre de l’application 49](#_Toc106007605)

[1. Sprint 0 : Environnement de travail et Configuration du système 49](#_Toc106007606)

[1.1. Environment logiciel 50](#_Toc106007607)

[1.2. Environnent Matériel 54](#_Toc106007608)

[1.3. Préparation de la machine virtuelle 54](#_Toc106007609)

[1.4. Conclusion 56](#_Toc106007610)

[2. Sprint 1 implémentation des APIs 57](#_Toc106007611)

[2.1. Analyse du sprint implémentation des APIs 57](#_Toc106007612)

[2.2. Revenu du sprint implémentation des APIs 61](#_Toc106007613)

[2.3. Conclusion 62](#_Toc106007614)

[3. Sprint 2 : Authentication et gestion de profile 63](#_Toc106007615)

[3.1. Analyse du sprint Authentication et gestion de profile 63](#_Toc106007616)

[3.2. Notion Application 69](#_Toc106007617)

[3.3. Revenu du sprint Gestion d’accès 70](#_Toc106007618)

[3.4. Conclusion 73](#_Toc106007619)

[4. Sprint 3 : Recherche pharmacie 74](#_Toc106007620)

[4.1. Analyse de sprint Recherche pharmacie 74](#_Toc106007621)

[4.2. Notion appliquer 76](#_Toc106007622)

[4.3. Revenu de sprint Recherche pharmacie 80](#_Toc106007623)

[4.4. Conclusion 81](#_Toc106007624)

[5. Sprint 4 : Messagerie et Google AdMob 82](#_Toc106007625)

[5.1. Analyse de sprint Messagerie et Google AdMob 82](#_Toc106007626)

[5.2. Notion appliquer 83](#_Toc106007627)

[5.3. Revenu de sprint Messagerie et Google AdMob 85](#_Toc106007628)

[5.4. Conclusion 86](#_Toc106007629)

[6. Sprint 5 : Administration 87](#_Toc106007630)

[6.1 Analyse de sprint Administration 87](#_Toc106007631)

[6.2 Notions appliquées 89](#_Toc106007632)

[6.3 Revenu de sprint Administration 91](#_Toc106007633)

[6.4 Conclusion 96](#_Toc106007634)

[Conclusion Générale 97](#_Toc106007635)

[Webographie 98](#_Toc106007636)

Liste des figures

[Figure 1 : Logo de l’entreprise Medical Innovation Technology 3](#_Toc106007372)

[Figure 2 : les acteurs de système 7](#_Toc106007373)

[Figure 3 : Diagramme de cas d’utilisation globale 10](#_Toc106007374)

[Figure 4 : Diagramme de classe 11](#_Toc106007375)

[Figure 5 : Schéma relationnel 16](#_Toc106007376)

[Figure 6 : Processus Scrum 17](#_Toc106007377)

[Figure 7:Pattern MVC 22](#_Toc106007378)

[Figure 8 : Pattern Singleton 23](#_Toc106007379)

[Figure 9 : Architecture application web 24](#_Toc106007380)

[Figure 10 : Structure d’une vue Angular 25](#_Toc106007381)

[Figure 11 : Data Binding et directives structurelles 26](#_Toc106007382)

[Figure 12:Logo Android 27](#_Toc106007383)

[Figure 13: architecture Android 28](#_Toc106007384)

[Figure 14 : Architecture d’une application Android 29](#_Toc106007385)

[Figure 15 :Composition d’une activité 30](#_Toc106007386)

[Figure 16 : Android Studio Développent Environnent 32](#_Toc106007387)

[Figure 17:Structure Retrofit 32](#_Toc106007388)

[Figure 18:Architecture Recycler View 33](#_Toc106007389)

[Figure 19 : Type d’application mobile 34](#_Toc106007390)

[Figure 20 :Architecture Springboot 36](#_Toc106007391)

[Figure 21:JPA Base Construction 38](#_Toc106007392)

[Figure 22 : Architecture Angular et springboot 39](#_Toc106007393)

[Figure 23 :Communication android et springBoot 40](#_Toc106007394)

[Figure 24 : Consommation google plateforme service 41](#_Toc106007395)

[Figure 25 : Bouton google authentification 41](#_Toc106007396)

[Figure 26 :API iternaire 42](#_Toc106007397)

[Figure 27 :API places android 42](#_Toc106007398)

[Figure 28 : Caractéristique d’un marker 43](#_Toc106007399)

[Figure 29 : Comprendre le flux des autorisations 44](#_Toc106007400)

[Figure 30 :Annonce bannieérs 45](#_Toc106007401)

[Figure 31 : Annonce Interstitiel 45](#_Toc106007402)

[Figure 32 :Annonces native 46](#_Toc106007403)

[Figure 33 : SpringBoot build 46](#_Toc106007404)

[Figure 34 : Déploiement 47](#_Toc106007405)

[Figure 35 : Android Build 48](#_Toc106007406)

[Figure 36 :APK non signé 48](#_Toc106007407)

[Figure 37:Architecture physique 49](#_Toc106007408)

[Figure 38 :Logo eclipse 50](#_Toc106007409)

[Figure 39:Logo android studio 50](#_Toc106007410)

[Figure 40: Logo Visual studio code 50](#_Toc106007411)

[Figure 41:Logo Postman 51](#_Toc106007412)

[Figure 42: Logo PuTTY 51](#_Toc106007413)

[Figure 43:Logo Workbench 51](#_Toc106007414)

[Figure 44:logo Virtual Private Server 52](#_Toc106007415)

[Figure 45:Logo Cloudinary 52](#_Toc106007416)

[Figure 46 : Logo logiciel Visual Paradigm 52](#_Toc106007417)

[Figure 47 :Logo GitHub 53](#_Toc106007418)

[Figure 48 :Logo Photoshop 53](#_Toc106007419)

[Figure 49 : Logo slack 53](#_Toc106007420)

[Figure 50 : Logo Figma 54](#_Toc106007421)

[Figure 51: Default Tomcat home page 55](#_Toc106007422)

[Figure 52 :Tomcat Webapp Manager 56](#_Toc106007423)

[Figure 53 : Prototype du diagramme par package 57](#_Toc106007424)

[Figure 54 : Diagramme de class par package 59](#_Toc106007425)

[Figure 55 : Diagramme de déploiement 60](#_Toc106007426)

[Figure 56 :Arborescence partie backend 61](#_Toc106007427)

[Figure 57:Find pharma data base 61](#_Toc106007428)

[Figure 58 : Tester les API avec la logiciel Postman 62](#_Toc106007429)

[Figure 59 : Endpoint après déploiement 62](#_Toc106007430)

[Figure 60 : Diagramme de cas d’utilisation sprint 1 63](#_Toc106007431)

[Figure 61 : Diagramme de séquence d’authentification 66](#_Toc106007432)

[Figure 62 : Diagramme de séquence d’inscription 67](#_Toc106007433)

[Figure 63 : Diagramme de séquence oublier mot de passe 68](#_Toc106007434)

[Figure 64 : Gérer un projet sur google cloud plateforme 69](#_Toc106007435)

[Figure 65 : Configuration de google sign 70](#_Toc106007436)

[Figure 66:interface inscription 71](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007437)

[Figure 67:interface d’accueil 71](#_Toc106007438)

[Figure 68: authentication avec google 72](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007439)

[Figure 69: interface inscription 72](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007440)

[Figure 71 : interface changer mot de passe 72](#_Toc106007441)

[Figure 70: interface validation 72](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007442)

[Figure 73 : interface profile utilisateur 73](#_Toc106007443)

[Figure 72: interface édit profile utilisateur 73](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007444)

[Figure 74 : Diagramme de cas d’utilisation sprint 3 74](#_Toc106007445)

[Figure 75 : Diagramme de séquence recherche médicaments 76](#_Toc106007446)

[Figure 76 :Maps SDK 77](#_Toc106007447)

[Figure 77 : API Google MAPS 78](#_Toc106007448)

[Figure 78:API key 78](#_Toc106007449)

[Figure 79: google maps permission 78](#_Toc106007450)

[Figure 80: Dependance 79](#_Toc106007451)

[Figure 81:interface d'accueil 80](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007452)

[Figure 82: recherche pharmacie 80](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007453)

[Figure 83: liste de pharmacie 81](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007454)

[Figure 84 :la direction de pharmacie 81](#_Toc106007455)

[Figure 85 : Diagramme de cas d’utilisation sprint4 82](#_Toc106007456)

[Figure 86 : Discuter médecin 83](#_Toc106007457)

[Figure 87 :compte google admob 84](#_Toc106007458)

[Figure 88 Google AdMob dependance 84](#_Toc106007459)

[Figure 89: Implementation key adMob 84](#_Toc106007460)

[Figure 91 : interface de discussion pharmacie 85](#_Toc106007461)

[Figure 90:l''interface de discussion médecin 85](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007462)

[Figure 92 : Diagramme de class sprint 5 87](#_Toc106007463)

[Figure 93 : Diagramme de classe gérer produit sprint4 87](#_Toc106007464)

[Figure 94 : Diagramme de séquence ajouter produit 89](#_Toc106007465)

[Figure 95:List utilisateurs 91](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007466)

[Figure 96:Dashboard Administrateur 91](#_Toc106007467)

[Figure 97: Liste de pharmacie 92](file:///C:\Users\karim\Desktop\imp\final1.docx#_Toc106007468)

[Figure 98: Interface statistique 92](#_Toc106007469)

[Figure 99:Interface d'authentification administrateur 93](#_Toc106007470)

[Figure 100:Interface de statistiques 93](#_Toc106007471)

[Figure 101:Interface de profile administrateur 94](#_Toc106007472)

[Figure 102:Interface de la gestion des utilisateurs 94](#_Toc106007473)

[Figure 103:Interface de la gestion des produits 95](#_Toc106007474)

[Figure 104:Interface de la gestion des pharmacies 95](#_Toc106007475)

Liste des tableaux

[Tableau 1 : Critique de l’existant 5](#_Toc106006614)

[Tableau 2 : les besoins fonctionnels 9](#_Toc106006615)

[Tableau 3 : Dictionnaire des données 13](#_Toc106006616)

[Tableau 4 : Product backlog 19](#_Toc106006617)

[Tableau 5: Planification temporelle 21](#_Toc106006618)

[Tableau 6 : Comparaison entre différents types d’applications mobile 36](#_Toc106006619)

[Tableau 7 :Environnement matériel 54](#_Toc106006620)

[Tableau 8 : Description Textuelle « Inscription » 64](#_Toc106006621)

[Tableau 9 :Description Textuelle « Authentication » 65](#_Toc106006622)

[Tableau 10 : Description Textuelle « réinitialiser mot passe » 66](#_Toc106006623)

[Tableau 11 : Description Textuelle « Recherche pharmacie » 75](#_Toc106006624)

[Tableau 12 : Description Textuelle « Gérer profile pharmacie » 76](#_Toc106006625)

[Tableau 13 : Description Textuelle « discussion médecin, pharmacie » 82](#_Toc106006626)

[Tableau 14 : Description Textuelle « modifier profile » 88](#_Toc106006627)

[Tableau 15 : Description Textuelle « ajouter Produit » 88](#_Toc106006628)

Liste Abréviation

* MIT : Médical Innovation Technology
* SPA : Single Page Application
* DOM : Document Object Model
* BDD : Base De Données.
* REST :Representational state transfer
* JPA : Java Persistance API
* HTTP :HyperText Transfer Protocol.
* IDE :Integrated Development Environment..
* MVC :Modèle Vue Contrôleur.
* MySql :My Structured Query Language.
* SGBD :System de Gestion de Base de Données.
* SQL :Structured Query Language.
* UML: Unified Modeling language
* APK : Android Package Kit
* API: Application Programming Interface
* GPS : Global Positioning System

# Introduction Générale

Les technologies de l'information et de la communication ont été la révolution la plus importante et innovante qui a marqué ces dernières décennies. En effet, loin d'être un phénomène éphémère ces technologies nous ont apporté du confort dans notre vie quotidienne par leurs capacités à traiter l’information dans des délais raisonnables. Cette révolution a permis l’émergence de la notion de la portabilité et de la mobilité qui permet un accès distant, instantané et un flux sans interruption d’informations. En effet, cela est symbolisé par l’apparition des différents appareils de haute technologie tels que les Smartphones et les tablettes qui sont dotés de plusieurs applications pratiques.

Ces dernières années, la téléphonie mobile a été sans doute le secteur le plus dynamique, le plus rentable et le plus innovant de toute l’Industrie des Télécommunications. Le marché des Smartphones, essentiellement, connaît un véritable essor dans lequel les acteurs habituels essaient de s'engouffrer. Google, ayant réalisé le potentiel de ce marché, a décidé de s'y introduire en sortant un nouveau système d'exploitation Android.

Dans ce cadre d’intégration, notre projet de fin d’études consiste à développer une application mobile qui offre à l’utilisateur une recherche de la pharmacie la plus proche.

Ce rapport détaillera les différentes phases par lesquelles nous sommes passés par, afin d’aboutir à une plateforme (front End et Back end) fiable et satisfaisante. Au début, nous commencerons par un premier chapitre « Cadre général du projet » qui présente une description du contexte général du projet.

Nous enchaînerons par la suite avec le deuxième chapitre « Etude préalable » qui comportera les besoins fonctionnels et non fonctionnels de l’application, et qui tracera les différents cas d’utilisations de notre plateforme de manière et nous détaillerons notre méthodologie de développement et des différentes releases de l’application réalisée en présentant les différents sprints.

Le troisième chapitre intitulé « Étude Théorique » sera consacré pour une analyse des technologies et des principes utilisés.

Dans le quatrième chapitre intitulé « Mise en œuvre de l’application » qui est dédié à la présentation des différents sprints de l’application réalisée en présentant leurs digrammes de conception ainsi que le développement associé aux interfaces nécessaires et les notions appliquées.

Nous terminerons ce rapport par une conclusion générale récapitulant tout notre travail.

Chapitre 1 : Présentation du cadre de projet

Ce chapitre sera destiné, en premier lieu, à la présentation de l’organisme d’accueil, ensuite nous présenterons notre cadre de projet et finalement, nous ferons une étude de l’existant.

1. Présentation de l’organisme

Nous avons effectué le stage au sein de la société Medical Innovation Technology (MIT) qui à proposé ce projet. Ce chapitre sera destiné, en premier lieu, à la présentation de l’organisme d’accueil, ensuite nous présenterons notre cadre de projet. Puis, nous ferons une étude de l’existant, nous le critiquerons et nous finirons par énoncer la méthodologie de la conception utilisée.

* A propos de Medical Innovation Technology

Medical Innovation Technology est une société spécialisée dans le domaine l'industrie des soins de santé, a été fondée en septembre 2019 à Tunis. Aujourd'hui, MIT utilise les technologies numériques pour améliorer la communication entre le patient et le spécialiste, fournir des traitements cliniques efficaces, optimiser le diagnostic des maladies dangereuses, réduire les menaces pour la santé publique.

Text

Description automatically generated with medium confidence

Figure 1 : Logo de l’entreprise Medical Innovation Technology

1. Présentation de projet

Dans cette partie, nous présenterons notre projet. Au début nous exposerons notre problématique ainsi que la description du sujet et les missions attendues.

### **Problématique**

De nos jours, la recherche de la disponibilité de certains médicaments est devenu de plus en plus fatigante et le fait de trouver un médicament dans la première ou deuxième pharmacie est devenu très difficile. Dans ce cadre, notre application vient pour résoudre ce problème et permettre à l’utilisateur de rechercher la disponibilité des médicaments dans la pharmacie la plus proche.

### **Description de l’existant**

La conception et la réalisation de tout projet doivent être précédées par une étude de l’existant qui détermine les points faibles et les points forts des systèmes actuels, tenant compte des objectifs à atteindre.

La recherche que nous avons menée sur les solutions disponibles sur le marché, nous a permis d’exposer par la suite quelques applications qui s’y intéressent de près :

* Localisation de pharmacie.
* PharMap.

### **Critique de l’existant**

|  |  |  |
| --- | --- | --- |
| Application | Localisation de pharmacie | PharMap |
| Objective | * Trouvez des entrées près de votre emplacement et n’importe où dans le monde. L’application affiche des éléments dans une liste ainsi qu’une carte et vous permet de naviguer facilement vers un objet.. | * PharMap vient de résoudre ce problème en vous indiquant suivant votre position géographique, la pharmacie de garde la plus proche de vous, ainsi que l’itinéraire qui y mène. Elle vous donne également la possibilité de contacter ces pharmacies pour vérifier la disponibilité d’un médicament. |
| Functionalities | * Localiser les pharmacies. * Horaires des pharmacies. | * Localiser les pharmacies. * Vérifier la disponibilité d’un médicament. * Horaires des pharmacies. * Contacter pharmacie (appel, WhatsApp). |
| Point fort | * L’application combine plusieurs bases de données qui la rend très fiable et robuste. |  |
| Point faible | * Manque le numéro de téléphone. * Service avec publicité. | * Pas de géolocalisation (Map). |

Tableau 1 : Critique de l’existant

### **Solution proposée**

Il s’agit d’une application mobile à travers laquelle l’utilisateur peut atteindre la pharmacie et la parapharmacie la plus proche qui lui fournit le médicament souhaité et faire la comparaison des prix entre les produits pharmaceutiques, ainsi qu’autoriser le médecin à connaître l’existence ou la pénurie de médicaments dans les pharmacies et diriger directement le patient vers eux, et permet au pharmacien pour évaluer les types de médicaments les plus requis.

* Objective :
* Trouver une pharmacie de garde ouverte à proximité.
* Que ce soit la journée, la nuit ou le week-end, les utilisateurs de l’application peuvent trouver simplement la pharmacie ou bien la pharmacie de garde la plus proche et qui est ouvert et suivre l’itinéraire pour s’y rendre.
* Vérifier la disponibilité des produits pharmaceutiques.
* Chercher le prix le moins chers des produits pharmaceutiques.

1. Conclusion

Dans ce chapitre, nous avons décrit le cadre du projet en y introduisant l’organisme d’accueil ainsi que la problématique et la solution proposée. Ensuite, nous avons détaillé quelques solutions existantes. Le chapitre suivant sera consacré à l’analyse et la spécification des besoins tout en présentant les différentes fonctionnalités du projet.

Chapitre 2 : Etude préalable

L’objectif de ce chapitre consiste à présenter les spécifications fonctionnelles et non-fonctionnelles du projet. Dans le cadre de notre projet, nous sommes appelés à définir l’ensemble des fonctionnalités de l’application, de les concevoir et de les réaliser et nous finissons par énoncer la méthodologie de conception utilisée.

1. Identification des acteurs et des fonctionnalités attendues

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositifs matériels ou autre système) qui interagit directement avec le système en étude.

Les acteurs qui interagissent avec notre système sont représentés dans la figure 2.

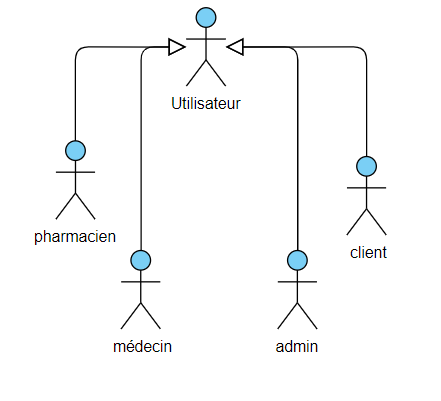


Figure 2 : les acteurs de système

Dans le cas de notre projet nous avons 4 utilisateurs de l’application :

* Administrateur : C’est un utilisateur qui permet d’assurer la gestion des comptes admin, client, pharmacie et médecin. C’est l’administrateur de notre système qui assure la gestion des comptes.
* Médecin : C’est un utilisateur du système. Il accède aux applications   pour chercher la pharmacie la plus proche ou pour voir la disponibilité de produits pharmaceutiques.
* Client : C’est un utilisateur du système. Il accède à l’application   pour chercher la pharmacie la plus proche.
* Pharmacien : C’est un utilisateur du système. Il accède à l’application pour ajouter la position et les coordonnées de sa pharmacie Ainsi que répondre aux questions des médecins et gérer les produits pharmaceutiques

### **Les besoins fonctionnels**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Acteur | Pharmacien | Client | Médecin | Administrateur | |
| Action | * S’authentifier * Gérer profile | | | | |
| * Inscription | | | |  |
|  | * Recherche des produit pharmaceutique | | | * Gestion des comptes client, médecin et pharmacien et gestion des produits |
|  | | * Recherche disponibilités médicaments | |  |
| * Ajouter une pharmacie sur le map * Ajouter un ou plusieurs produits |  | | | |

Tableau 2 : les besoins fonctionnels

### **Les besoins non fonctionnels**

Un besoin non fonctionnel est un besoin spécifiant les propriétés que le système doit avoir mais qui ne sont pas reliés directement au comportement du système. Les principaux besoins non fonctionnels de notre application se résument comme suit :

* Rapidité : il faut développer une application avec un temps de réponse minimal.
* Flexibilité : la possibilité d’ajouter d’autres fonctionnalités et d’apporter des mises à jour à l’application.
* Fiabilité : l’application doit toujours être en mesure de fonctionner correctement.
* Sécurité : l’application doit garantir la confidentialité et l’intégrité des informations ainsi que le traitement de l’identification des utilisateurs et la gestion des droits.
* Simplicité : le système doit être simple à utiliser. La navigation entre les pages doit être souple et facile.
* Modularité : l’application doit avoir un code simple et facile à maintenir et à comprendre.

### **Diagramme de cas d’utilisation global**

Les diagrammes des cas d’utilisation sont des diagrammes UML utilisés pour donner une vision globale du comportement fonctionnel d’un système logiciel. Ils sont utiles pour des présentations auprès de la direction ou des acteurs d’un projet, mais pour le développement, le cas d’utilisation est plus approprié. Un cas d’utilisation représente une unité discrète d’interaction entre un utilisateur (humain ou machine) et un système. Il correspond à une unité significative de travail. Dans un diagramme des cas d’utilisation, les utilisateurs sont appelés acteurs « Actors » qui interagissent avec les cas d’utilisation « use cases ».

Diagram

Description automatically generated

Figure 3 : Diagramme de cas d’utilisation globale

### **Diagramme de Classe**

Diagram

Description automatically generated

Figure 4 : Diagramme de classe

### **Dictionnaire de données**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tableau | Attribut | Obligatoire | Pk/Fk | Type | Description |
| User | Iduser | oui | Pk | Numérique | Identifiant de User |
| Nom | oui | - | Chaine | Nom de l’utilisateur |
| prenom | oui | - | Chaine | Prenom de l’utilisateur |
| email | oui | - | Chaine | Email de l’utilisateur |
| password | oui | - | Chaine | Password de l’utilisateur |
| tel | oui | - | Chaine | Telephone de l’utilisateur |
| Image | non | - | Chaine | Image de l’utilisateur |
| Role | oui | - | Chaine | Role de l’utilisateur (Admin, Medecin, Pharmacien, Client) |
| sexe | oui | - | Chaine | Sexe de l’utilisateur (Male, Female) |
| Create\_At | oui | - | Date | Date de l’insertion de l’utilisateur dans la base de données |
| Update\_At | oui | - | Date | Date de la dernière modification de l’utilisateur dans la base de données |
| state | oui | - | Boolean | L’état d’utilisateur (active ou non active) |
| Produit | Idproduit | oui | Pk | Numérique | Identifiant de la produit |
| Nomproduit | oui | - | Chaine | Nom de la produit |
| Constructeur | oui | - | Chaine | Constructeur de la produit |
| Reference | oui | - | Chaine | Reference de la produit |
| Image | non | - | Chaine | Image de l’utilisateur |
| Create\_At | oui | - | Date | Date de l’insertion du produit dans la base de données |
| Update\_At | oui | - | Date | Date de la dernière modification du produit dans la base de données |
| State | oui | - | Boolean | La disponibilité de produit |
| Prix | oui | - | Numérique | Le prix de produit |
| type | oui | - | chaine | Le type de produit |
| pharmacie | Idpharmacie | oui | pk | Numerique | Identifiant de pharmacie |
| nompharmacie | oui | - | chaime | Nom de la pharmacie |
| localisation | oui | - | chaine | Les Coordonnées géographique de la pharmacie |
| image | oui | - | Chaine | L’image de pharmacie |
| Doc\_sign | oui | - | Chaine | Document de signature |
| State | oui | - | Chaine | La disponibilité de pharmacie |
| adresse | oui | - | Chaine | L’adresse de pharmacie |
| horaires | oui | - | Numérique | Le temps de travail de pharmacie |
| Created\_at | non | - | Date | Date de l’insertion de la pharmacie dans la base de données |
| Updated\_at | oui | - | Date | Date de la dernière modification de la pharmacie dans la base de données |
| adresse | oui | - | Chaine | L’adresse de pharmacie |
| médecin | idmedecin | oui | PK | Numérique | L’identifiant de médecin |
| cin | oui | - | Chaine | Le numéro de la carte d’identité |
| telcabinet | oui | - | Chaine | Numéro du téléphone du cabinet |
| Pharmacien | Idpharmacien | oui | PK | Numérique | Identifiant de pharmacien |
| cin | oui | - | Chaine | Le numéro de la carte d’identité |
| conversation | idconversation | oui | PK | Numérique | L’identifiant de la conversation |
| content | oui | - | Chaine | Le contenu de conversation |
| Date | oui | - | Date | Date de conversation |
| order | oui | - | Numérique | L’ordre de l’envoyeur de message |
| Admin | idadmin | oui | PK | Numérique | Identifiant d’admin |
| Médicaments | idmedecimaents | oui | PK | Numérique | Identifiant de médicaments |
| prix | oui | - | Numérique | Prix de médicaments |
| cnam | oui | - | Boolean | Si les médicaments est inclut au cnam |
| paramédical | idpara | oui | PK | Numérique | Identifiant de paramédical |
| complémentaire | idcom | oui | PK | Numérique | Identifiant de complémentaire |
| Client | idclient | oui | PK | Numérique | Identifiant de client |

Tableau 3 : Dictionnaire des données

Notre diagramme de classe contient 15 tables :

* Les 4 Tables client, médecin, pharmacien, admin : chaque table qui contient les informations nécessaires pour l’un des 4 acteurs sont des tables fils qui hérite de la table mère User.
* Médicament, paramédical et complémentaire (relation héritage).
* Table médicament, paramédical et complémentaire, sont des tables fils qui hérite de la table mère produit.
* Table Conversation est une table qui contient des champs qui décrivent la date d’envoie d’un message du médecin au pharmacien et inversement.

### **Modèle relationnel**

Le concepteur d’une base de données relationnelle doit élaborer un schéma relationnel de la base de données. Cette activité consiste à définir toutes les relations de la base de données et leurs attributs.

Règles de passage au modèle relationnel : Les règles utilisées pour le passage du diagramme de classes de notre application web au modèle Relationnel sont les suivantes :

• Toute entité deviennent une relation ayant pour clé primaire son identifiant.

• Chaque propriété se transforme en attribut.

• Toute association non hiérarchique (de type [n, n]) devient une relation. La clé primaire est formée par la concaténation (juxtaposition) de l’ensemble des identifiants des entités reliées. Toutes les propriétées éventuelles deviennent des attributs qui ne peuvent pas faire partie de la clé.

* Légende :

La clé primaire est soulignée

Les clés étrangères sont suivies d’un symbole #

* Héritage :

Produit (idproduit, nom, produit, référence, constructeur, description,image,prix,

nom,state,type,reated\_at,updated\_at).

Médicament (idmed, prix, cnam).

Paramédical (id para).

Complémentaires (idcom).

* Association :

Pharmacie avec 3 tables (médicament ,paramédical,complémentaires).

Optimisation : Produit (idproduit, nom, référence, constructeur, description, image, prix, cnam, type, created\_at ,updated\_at,type,state ).

Une seule table pharma\_prod (#idproduit, idpharmacie, etat, prix).

Pharmacie (idpharmacie #iduser, horaire, pharmajour, pharmaweak, image, Location, nompharmacie, doc\_sgin,cretaed\_at,updtaed\_at,adresse).

* Héritage :

User (iduser, nom, prenom, tel, date\_nes, email, password, sexe, image,reated\_at

,reated\_at,state).

Client (idclient).

Pharmacien (idpharmacien, cin).

Médecin (idmed, cin, tél cabinet).

* Optimisation :

Une seule table User (iduser, nom, prenom, tel, date\_nes, email, password, sexe, image, cin, telcabinet, role,reated\_at,reated\_at ,state #idpharmacie)

### **Schéma relationnel**

Le schéma suivant nous présente les tables de notre base de données illustré.

Diagram

Description automatically generated

Figure 5 : Schéma relationnel

1. Méthodologie

### **Méthodologies de travail:**

Plusieurs méthodes existent, mais le choix dépend de la nature du projet. Si le domaine du projet est bien maîtrisé, le cycle de vie en cascade suffit. Mais, comme nous ne pouvons pas tout prévoir dès le début ou si les besoins sont incomplets comme dans notre cas, alors il faut utiliser les méthodes itératives et incrémentales telles que les méthodes agiles.

Parmi les méthodes agiles, nous pouvons citer « SCRUM » que nous avons choisi pour notre projet, car il assure une bonne définition et répartition des tâches de telle façon que le projet devient extrêmement organisé et la planification des tâches devient instinctive.

### **presentation du Scrum:**

Scrum est un Framework d’organisation de développement de produits complexes. Il est défini par ses créateurs comme un cadre de travail permettant de répondre à des problèmes complexes et changeants tout en livrant de manière productive et créative des produits de la plus grande valeur possible. Scrum est considéré comme une méthode agile.

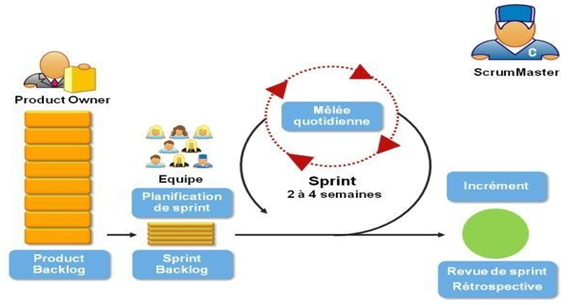


Figure 6 : Processus Scrum

Nous avons choisi le Framework Scrum pour les raisons suivantes :

* La souplesse et la réactivité.
* Augmentation de la productivité.
* Deadlines intégrés c’est-à-dire que chacun possède une responsabilité qu’il doit l’accomplir jusqu’à la fin.
* Flexibilité, en effet grâce à Scrum on a pu accélérer la réalisation des projets et les améliorer.
* Transparence et respect ça veut dire que SCRUM exige la transparence et qu’on est une équipe qui travaille ensemble pour atteindre un objectif, y’a plus de patron, on est tous les mêmes.

### **Répartitions des rôles dans Scrum**

* Le Product Owner : celui qui porte la vision du produit à réaliser en général c’est le client.
* Le SCRUM Master : est l’animateur d’une équipe qui applique la méthode agile qui consiste à aider l’équipe à travailler. Il est le garant de Scrum sur le projet. Principalement agir en facilitateur, va protéger l’équipe en cas des obstacles empêchant l’équipe,
* Le SCRUM team : c’est l’équipe qui réalise le produit, elle se compose de 4 à 10 personnes en général.

## **Pilotage du projet avec Scrum**

### **User Stories**

Les histoires d’utilisateurs sont des brèves descriptions simples d’une fonctionnalité

Présente de point de vue de la personne qui souhaite utiliser la nouvelle fonctionnalité, généralement un utilisateur ou un client du système. Ils suivent généralement un modèle simple.

En tant que <type d'utilisateur>, je veux <un objectif> pour que <une raison>

### **Product Backlog**

Le Product Backlog est une responsabilité du Product Owner. Elle est la liste des besoins du projet : les fonctionnalités et les besoins non-fonctionnels demandés par le client. Les éléments du backlog de produit, appelé aussi les histoires utilisateurs, précisent la fonctionnalité désirée par le client, généralement, écrits sous la forme suivante « En Tant que X, je veux Y ».

Le backlog de produit présenté dans le tableau comprend les champs suivants :

* ID : C’est un nombre unique et auto-incrémenté pour chaque histoire utilisateur.
* Thème : Pour mieux ordonner les user stories.
* User Story : comporte la description d’une fonctionnalité désirée par le client suivant la forme « En tant que ...Je veux... ».
* Priorité : Identifié par le producteur lors des discussions avec le client final.Complexité : La complexité est évaluée entre 3 niveaux faible, moyenne et élevée.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Thème | User story | Priorité | complexité |
| 1 | Authentication et gestion de profile | * En tant qu’utilisateur de l’application, je veux me connecter à l’application. * En   tant   qu’utilisateur   je   veux   récupérer l’accès   à   l’application   avec   l’option   mot de passe oublié * En   tant   qu’user (client/médecin/admin) ,   je   veux   créer un compte. * gestion de profile | 1 | faible |
| 2 | Recherche Pharmacie | * En tant que client, admin, médecin, pharmacien, je veux rechercher la pharmacie la plus proche. * En tant que client, médecin, je veux rechercher le produit pharmaceutique le moins cher. | 2 | élevée |
| 3 | Messagerie | * En tant que médecin je veux discuter (question/réponse) avec le pharmacien. * En tant que médecin je veux rechercher les produits pharmaceutique | 3 | moyenne |
| 4 | Administration | * En tant qu’administrateur je veux gérer les comptes d’utilisateurs et les produits. | 4 | Moyenne |

Tableau 4 : Product backlog

### **Enchainement des Sprints**

Authentication Gestion profile

Administration

se

Messagerie

Recherche pharmacie

**Sprint 1**

**se**

**Sprint 4**

se

**Sprint 3**

**Sprint 2**

Administration

se

Dans notre projet, nous avons estimé de réaliser notre application dans une durée approximative de quatre mois.

Le tableau ci-dessous montre le planning que nous avons adaptée pour planifier notre projet d’une meilleure façon possible et pour une visualisation rapide de l’avancement des différentes tâches.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Mois | Février 2022 | | | | Mars 2022 | | | | Avril 2022 | | | | Mai 2022 | | | | Juin 2022 | | | | |
| Semaine | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Etude préalable |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Analyse des besoins et spécification |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Conception |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Auto-formation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Développement |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Test et validation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Rédaction du rapport |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Tableau 5: Planification temporelle

1. Conclusion

Dans ce chapitre, nous avons modélisé notre application en apportant des réponses à nos questions de conception. En s’appuyant sur l’analyse des besoins de notre application, nous avons pu modéliser sous forme de diagrammes de cas d’utilisation, puis concevoir notre application sous forme de diagrammes de classe et schémas relationnelle. Il nous reste de concevoir et de créer notre application.

Chapitre 3 : Etude Théorique

Au cours de ce chapitre, nous allons parler de notre auto-formation et nos recherches réalisées avant de commencer la réalisation du projet pour le maîtriser et ne pas rencontrer des difficultés en temps de développement. Mise en œuvre de l’application

Ce chapitre comporte les notions de base des Framework Spring Boot, Angular et les notions de base de développement mobile sous Android.

1. Les Patterns

* Pattern MVC (Modèle-vue-contrôleur) :

MVC est un modèle de conception formé de trois composant :

Le modèle : son rôle est d’implanter les fonctionnalités de l’application, indépendamment des aspects interactifs.

Vue : est responsable de l’interface, ce qui recouvre essentiellement dans notre cas les fragments HTML qui sont assemblés pour constituer les pages du site.

Contrôleurs : le rôle des contrôleurs est de récupérer les données utilisateurs, les filtrer et les contrôler, puis déclencher le traitement approprié (via le modèle) et finalement de déléguer la production du document de sortie à la vue.

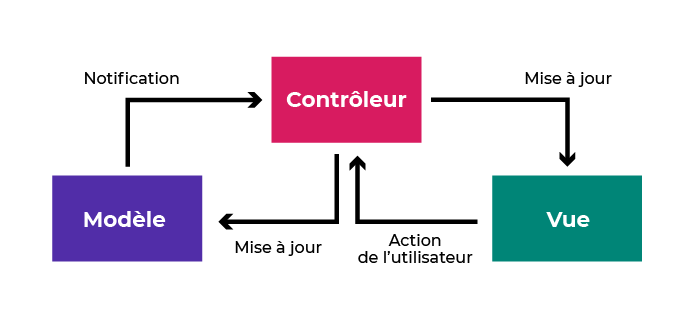


Figure 7:Pattern MVC

* Pattern singleton:

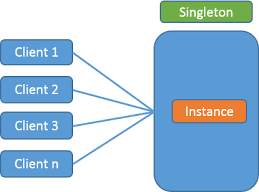


Figure 8 : Pattern Singleton

Le modèle Singleton est l’un des modèles de conception, il est utilisé lorsqu’une seule instance d’une classe est requise pour contrôler l’action tout au long de l’exécution. Les classes Singleton sont utilisées pour la connexion à la base de données.

## **Application Web et mobile**

### **Application Web**

Une application web désigne une plateforme applicatif hébergé sur un serveur et accessible via un navigateur web.

Contrairement à un logiciel traditionnel, l’utilisateur d’une application web n’a pas besoin de l’installer sur son ordinateur. Il lui suffit de se connecter à l’application à l’aide de son navigateur. La tendance actuelle est d’offrir une expérience utilisateur et des fonctionnalités équivalentes aux logiciels directement installés sur les ordinateurs

Le développement d’une application web divisé en deux parties :

* Front end

La partie ’Frontend’ représente la couche supérieure du projet et elle correspond à la couche de présentation qui réagit directement avec les utilisateurs.

* Back end

L’application ’Backend’ représente la couche intermédiaire entre l’application ’Frontend’ et la base de données. C’est le cœur du projet et elle contrôle la fonctionnalité de base de l’application en effectuant un traitement détaillé. Elle correspond donc à la couche métier dans l’architecture à trois niveaux. Elle reçoit les entrées et les demandes des utilisateurs de l’application ’Frontend’, les traite et les connecte à la couche de base de données pour effectuer des opérations de lecture/écriture puis renvoie une réponse à l’application ’Frontend’ pour afficher le résultat.

### **Les principes de la partie Frontend (Angular)**

Dans cette partie nous allons présenter quelques notions qui décrivent le Framework Angular.

Angular est un Framework permettant de créer des applications coté cliente d’une seule page (spa) à l’aide de HTML et de Type Script.

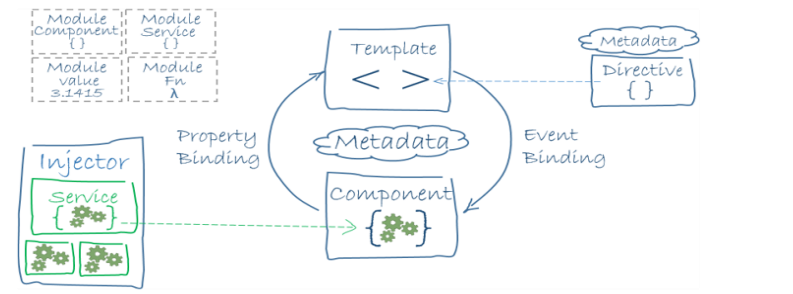


Figure 9 : Architecture application web

Angular permet de mettre en place des applications web modulaires afin de mieux structurer le code et de faciliter la maintenance, la réutilisation et le partage.

* Modules : Un module Angular ou NgModule est un mécanisme permettant de regrouper des composants, des services, directives etc. définir leurs dépendances, et définir leur visibilité. Les NgModules peuvent importer des fonctionnalités exportées depuis d’autres NgModules et exporter quelques fonctionnalités pour que d’autres NgModules puissent les utiliser. Chaque application Angular comprend au moins un module : le module racine ou root, appelé par défaut AppModule

Création d’un component : ng generate module nom\_module

* Composant : L’un des principaux concepts d’Angular est les composants. Les composants représentent les briques élémentaires qui constituent l’application. Un composant contrôle une portion de l’affichage graphique de l’application appelée vue. Une vue peut être par exemple l’ensemble de la page web, une fenêtre de tchat, une barre de navigation, etc. À chaque composant, est associée une classe qui définit sa logique (fichier type script) et un Template html. Une application Angular possède toujours un composant racine appelé conventionnellement App Component. Un composant peut contenir un ou plusieurs composants.

Création d’un component : ng generate component nom\_component

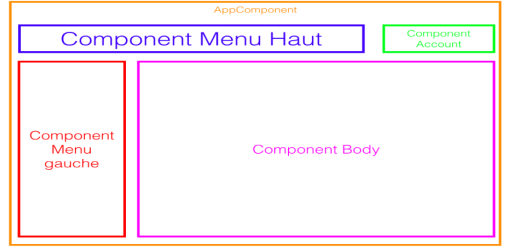
**

Figure 10 : Structure d’une vue Angular

* [Classes Métier](https://iset.uvt.tn/mod/resource/view.php?id=415686): Pour modéliser les entités métier au niveau de l’application, vous allez utiliser des classes TypeScript

Créer un class: ng generate class nom\_class.

* [Classes](https://iset.uvt.tn/mod/resource/view.php?id=415686) services : une dépendance est généralement l’instance d’une classe permettant de factoriser certaines fonctionnalités ou d’accéder à un état partagé permettant ainsi aux composants de communiquer entre eux

Créer un service : ng generate service nomService.

* Data Binding et Directives Structurelles

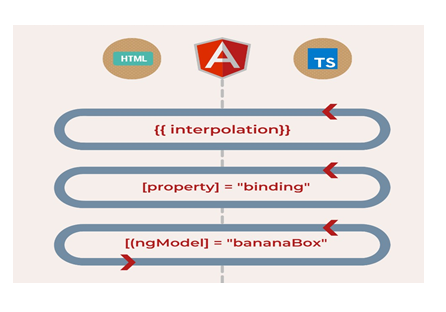


Figure 11 : Data Binding et directives structurelles

* Interpolation : pour afficher la propriété d’une variable du composant dans la vue html correspondante, on utilise l’interpolation dans son template component.html : {{variable}}.
* Les directives sont des instructions intégrées dans le DOM. Quand angular lit le template et rencontre une directive qu’il reconnaît, il suit les instructions correspondantes.

Exemple : au lieu d’afficher le contenu d’un tableau à la main vous allez utiliser une boucle pour parcourir automatiquement le tableau : la directive \*ngFor La directive \*ngIf permet de réaliser un traitement spécifique selon la condition à traiter.

* Property binding est une technique qui aidera à lier et contrôler n’importe quelle propriété d’un élément html.
* http Client

Http Client est un module angulaire qui permet à notre application de communiquer avec les services backend via le protocole http. Vous pouvez effectuer toutes les requêtes http, y compris GET, POST, PUT, et DELETE. Vous pouvez également modifier les en-têtes pour insérer des paramètres d’autorisation ou pour spécifier le type de contenu dont votre application a besoin, par exemple JSON, XML, etc.

### **Application Mobile**

Une application mobile est, en premier lieu, un logiciel. Un programme téléchargeable sur smartphone ou tablette qui comporte un fichier qui est installé puis exécuté par le système d’exploitation de votre mobile. Ce fichier est codé dans un langage de développement spécifique à votre appareil.

### **Application** **mobile Android**

Le système d’exploitation Android est un système d’exploitation open source principalement utilisé dans les appareils mobiles. Il est écrit principalement en Java et basé sur le système d’exploitation Linux. Le système d’exploitation Android est symbolisé par un logo Robot Android de couleur verte.

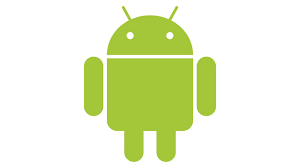


Figure 12:Logo Android

* Architecture d’Android :

L’architecture Android est une pile logicielle de composants permettant de prendre en charge les besoins d’un appareil mobile. La pile logicielle Android contient un noyau Linux, une collection de bibliothèques c / c ++. Le système d’exploitation Android est une pile de composants logiciels divisée en cinq sections et quatre couches principales, comme indiqué ci-dessous dans le diagramme d’architecture.

* Un noyau Linux (multitâches) des bibliothèques graphiques, multimédias
* Une machine virtuelle Java adaptée : la Dalvik Virtual Machine.
* Un Framework applicatif proposant des fonctionnalités de gestion de fenêtres, de téléphonie et de gestion de contenu.
* Couche Application : Les applications constituent la couche supérieure de l’architecture Android. En effet, ces applications sont préinstallées, prenons quelques exemples comme les applications “*Home*”, “*contacts*”, “*camera*” et “*gallery*” montrés dans la figure suivante.

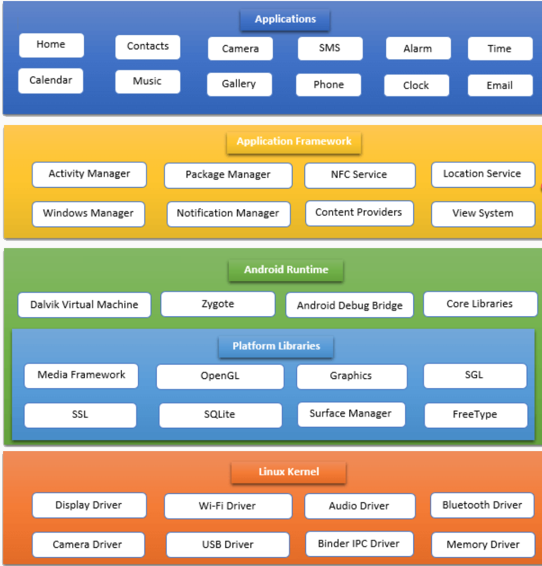


Figure 13: architecture Android

* Architecture d’une application Android :

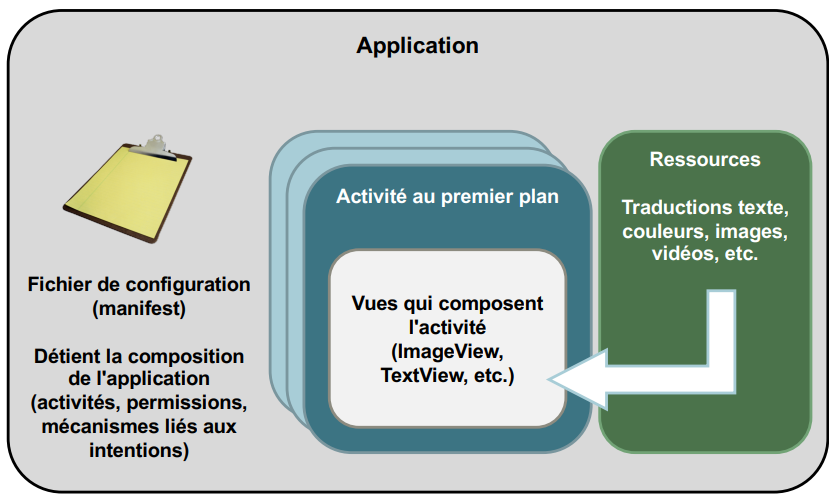


Figure 14 : Architecture d’une application Android

Une application Android est un assemblage de composants liés grâce à un fichier de configuration. Les Activités à Classe Java Les Vues et contrôles à Fichiers XML. Les Ressources à Interfaces (*Layout*), Images, Textes, audio, vidéo, … Le fichier de configuration à AndroidManifest.xml.

* Activités (*Activity*) :

Une activité est la composante principale pour une application Android. Elle dicte l’interface de l’utilisateur et manipule l’interaction de l’utilisateur avec l’écran du smartphone. Une activité est constituée d’un contexte d’une application et d’une seule et unique interface graphique.

Graphical user interface, diagram

Description automatically generated

Figure 15 :Composition d’une activité

* Les services :

Un service est le composant qui s’exécute en arrière-plan pour effectuer des opérations à long terme.

* Fournisseurs de contenu (*Content Provider*) :

Les *Content Providers* sont des gestionnaires de données comme l’exprime leurs noms, ils servent à accéder à des données depuis l’application. Nous pouvons accéder aux fichiers stockés dans le téléphone ainsi que d’autre données stockées dans l’application, grâce aux *content providers.* L’accès aux fichiers stockés dans le téléphone ainsi que d’autres données depuis l’application grâce aux.

* *Views* :

Les *Views* sont les composants de base de l’interface graphique. Ce sont les éléments UI, y compris les éléments de formulaire (Les boutons, *TextView*, *Check Box*…)  Et les éléments multimédias (*ImagesView*, *ImageButton*…).

* Layouts :

Une mise en page définit la structure d’une interface utilisateur dans l’application, comme dans une activité. Tous les éléments de la présentation sont créés à l’aide d’une hiérarchie d’objets *View* et *ViewGroup*. Une vue dessine généralement quelque chose que l’utilisateur peut voir et avec lequel il peut interagir.

* Fragments :

Un fragment définit généralement une partie d’une interface utilisateur. Les fragments doivent être intégrés aux activités ; ils ne peuvent pas fonctionner indépendamment des activités.

* Ressources :

Les ressources sont utilisées pour tout ; de la définition des couleurs, des images, des dispositions, des menus jusqu’à la définition des valeurs de chaîne.

* *Manifest* :

Un fichier de configuration qui contient des informations sur le *package*, y compris les composants de l’application tels que les activités, les services, les récepteurs de diffusion, les fournisseurs de contenu, les permissions, etc.

Chaque application Android nécessite un fichier de configuration : AndroidManifeste.xml . Il décrit :

* Le contexte de l’application
* Les activités
* Les récepteurs d’intents
* Les fournisseurs de contenu
* Les permissions.
* Développement d’une application mobile (Android) :

Afin de développer une application Android il faut savoir 3 choses, le langage de programmation (JAVA ou KOTLIN), XML et Android studio en tant qu’IDE. Java ou kotlin sont strictement nécessaire pour la création d’une application Android.

XML est pour optimiser le développement Android. Ce langage de balisage est utilisé pour gérer l’affichage du contenu à l’écran. Les interfaces à l’écran sont représentées par des classes appelées « Activités ».

Graphical user interface, application

Description automatically generated

Figure 16 : Android Studio Développent Environnent

### **Les Notions d’Android**

* Retrofit :

Retrofit est un client http de type sécurisé pour Android et Java. La mise à niveau facilite la gestion des données JSON ou XML qui sont analysées en objets Java. Retrofit prend en charge tous les appels http et la transmission de données entre les applications clientes Android et un serveur distant.

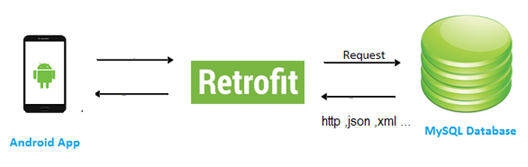


Figure 17:Structure Retrofit

* *RecyclerView* :

Android *RecyclerView* est une version plus avancée de *ListView* avec des performances améliorées et d’autres avantages. La classe *RecyclerView* étend la classe *ViewGroup* et implémente l’interface *ScrollingView*. *RecyclerView* est principalement utilisé pour concevoir l’interface utilisateur avec un contrôle précis sur les listes et les grilles des applications Android.

Diagram

Description automatically generated

Figure 18:Architecture Recycler View

* Structure de *Recyclerview*:

Pour implémenter une *RecyclerView* de base, trois sous-parties doivent être construites pour offrir aux utilisateurs le degré de contrôle dont ils ont besoin pour créer différentes conceptions de leur choix.

La mise en page de l’élément : la mise en page de la carte est une mise en page XML qui sera traitée comme un élément de la liste créée par *RecyclerView*.

* La classe Data : Est une classe Java personnalisée qui agit comme une structure pour contenir les informations de chaque élément de *RecyclerView*.
* L’adaptateur : Permet de faire la liaison (Bind) entre la vue RecyclerView et une liste de données.
* Le *ViewHolder*: Est une classe Java et elle est une partie de l’adaptateur qui stocke la référence aux vues de mise en page de la carte qui doivent être modifiées dynamiquement lors de l’exécution du programme par une liste de données obtenues par des bases de données en ligne ou ajoutées dans d’autres façon.
* *Shared Préférence*

L’une des options de stockage de données les plus intéressantes qu’Android offre à ses utilisateurs est les préférences partagées. Les préférences partagées sont la manière dont on peut stocker et récupérer de petites quantités de données primitives sous forme de paires clé/valeur dans un fichier sur le stockage de l’appareil, telles que *String*, *int*, *float*, *Boolean* qui constituent vos préférences dans un fichier XML à l’intérieur de l’application sur le stockage de l’appareil. Les préférences partagées peuvent être considérées comme un dictionnaire ou une paire clé/valeur.

### **Types des applications mobile**

Il existe trois types de base d’applications mobiles tel que :

* Une application mobile web : peuvent fonctionner sur n’importe quel appareil mobile ou système d’exploitation, car elles sont fournies à l’aide d’un navigateur mobile.
* Une application native
* Une application hybride : il s’agit d’une combinaison d’applications natives et Web.

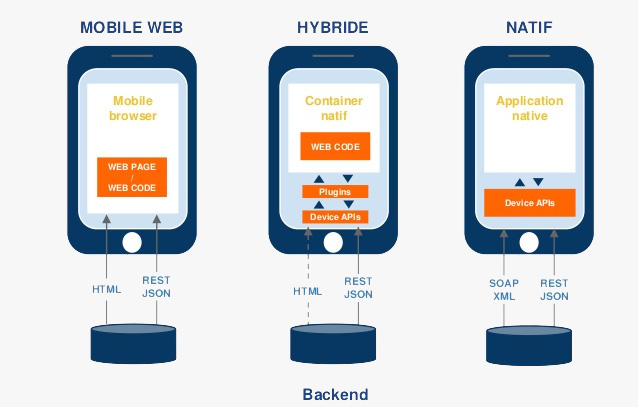


Figure 19 : Type d’application mobile

* Application native :

Une application native est spécifique à une plateforme Android ou iOS. Le langage de programmation est différent : Java pour Android et Swift pour iOS. Il a l’avantage d’utiliser les fonctionnalités intégrées du système d’exploitation. Ces candidatures sont distribuées via sa boutique (*Play Store*, *App Store*, etc.). Développer une application native permet généralement d’utiliser toutes les fonctionnalités liées au système d’exploitation ciblé (GPS, accéléromètre, caméra, etc.) et permet également de proposer des applications généralement plus riches que les applications web HTML5. Une fois téléchargées et installées, certaines applications peuvent être utilisées sans Connexion Internet.

Cependant, une application native ne peut être utilisée que par les mobinautes utilisant le système d’exploitation mobile. Il faut donc multiplier le nombre des applications natives si nous voulons toucher le plus grand nombre d’utilisateurs mobiles.

* Application web mobile :

Les applications mobiles web sont réellement des sites web développés en HTML et conçus pour être utilisés sur les tailles des écrans mobiles. Ils sont accessibles via le navigateur web du smartphone et ne nécessitent pas forcément de télécharger l’application.

Le point fort c’est qu’on n’a qu’à développer un seul format qui marche sur toutes les plateformes donc un gain de temps de développement énorme et une facilité de mise à jour.

Cependant l’application web n’utilise pas la mémoire du smartphone donc si l’application nécessite d’intégrer plusieurs fonctionnalités elle sera longue à charger.

* Application hybride :

Application Hybride Tout comme les applications web, ce type est développé à partir des langages web qui vont par suite s’intégrer à des « *WebView* » dans la partie native. En plus de ça, ce sont des applications multiplateformes qui peuvent utiliser les fonctionnalités natives.

Néanmoins, les applications hybrides ne peuvent pas bien parfois s’adapter au système d’exploitation du smartphone pour des raisons de résolution et d’autres.

* Application native hybride :

Les applications hybrides natives peuvent être développées en utilisant Flutter avec Dart comme langage de programmation ou React Native avec JavaScript. Par exemple, en utilisant Flutter, L’application est écrite en Dart sauf qu’elle se compile en native code binaire. Généralement, ces applications fonctionnent de la même manière que les applications natives.

* Comparaison entre différents types d’applications mobiles :

Il est important de comprendre les différents types d’applications. Notre objectif ici est de donner une vision et une compréhension globale des forces et faiblesses des plateformes native, hybride ou application web. Le tableau suivant sert à nous donner une compréhension globale entre différentes implémentations d’applications.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Paramètres | Web Apps | Native Apps | Hybride Web App | Hybride Native Apps |
| Mise sur le marché rapide | ++ | - | + | ++ |
| Accès à  Matériel de l’appareil | - | ++ | + | + |
| Haute qualité  Convivialité et performances | - | ++ | - | ++ |
| Normes UI/UX | - | ++ | - | ++ |

Tableau 6 : Comparaison entre différents types d’applications mobile

1. Les principes de la partie backend (Spring Boot)

Spring boot est un Framework open source pour construire et définir l’infrastructure d’une application Java, où il facilite le développement et les tests.

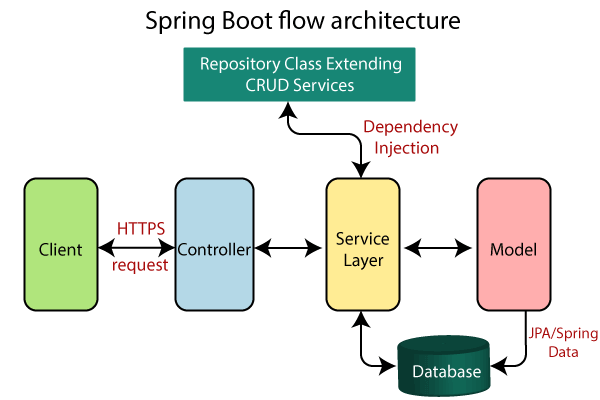


Figure 20 :Architecture Springboot

* Couche Controller : Son rôle est de générer la réponse à la requête http demandée par l’utilisateur. Il se charge d’analyser et de traiter la requête de l’utilisateur.
* Couche Service : implémentation des traitements métiers spécifiques à l’application.
* Couche Model : implémentation des objets métiers qui seront manipulés par les autres couches.
* Couche Repository : interaction avec les sources de données externes (base donnée)
* Couche view : Il s’agit de l’interface utilisateur de l’application qui présente les fonctionnalités et les données de l’application à l’utilisateur.
* Spring Boot Annotations

Est une forme de métadonnées qui fournit des données sur un programme qui ne fait pas partie du programme lui-même. Ils n’ont aucun effet direct sur le fonctionnement du code.

Les annotations Spring Boot n’utilisent pas XML et utilisent à la place le principe de la convention sur la configuration.

L’annotation utiliser soit avant une déclaration attribut, méthode, classe Exemple :

@Service : Il est utilisé au niveau de la classe. Il montre que la classe annotée est une classe de service.

@RequestMapping : est utilisé pour mapper la requête http. Il est utilisé avec la classe ainsi que la méthode.

@id : il est utilisé avec les attributs pour montrer que la variable est une clé primaire de la classe.

@Autowired : une classe avec l’annotation @Autowired résoudra automatiquement l’instance et l’injectera dans la classe qui l’a déclarée. (Instance singleton).

* Spring Boot dépendances

Gérer la gestion des dépendances est une tâche difficile pour les grands projets spring boot la résolution de ce problème s’effectue en fournissant un ensemble de dépendances; Il vous fallait juste d’injecter la dépendance dont vous avez besoin : spring-boot-starter-\*\*dependency\*\* .

Exemple :

* Si vous voulez utiliser spring et mongodb, il suffit d’inclure spring-boot-starter-data-mongodb dans le projet.
* JPA

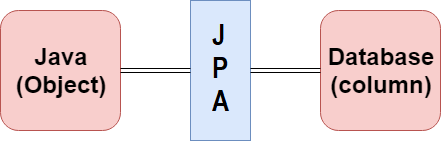


Figure 21:JPA Base Construction

JPA (Java Persistance API) est une interface de programmation Java utilisée pour développer et maintenir une relation entre un objet et une base de données relationnelle en mappant un état d’objet à une colonne de base de données. Il est capable de gérer facilement diverses opérations de base de données telles que l’insertion, la mise à jour, la suppression, etc.

1. Communication client et service

### **Web Services**

Les services Web sont des protocoles qui permettent la communication et l’échange de données entre des systèmes informatiques distribués. Le grand avantage des services web est que la communication peut se faire entre des systèmes informatiques d’architectures différentes.

 Les systèmes qui exploitent les services peuvent être situés dans des environnements de développement totalement différents.

Les services web de type ”REST” également appelés Les services Web RESTful » sont des services Web basés sur les technologies Web. C’est-à-dire qu’ils exposent leurs fonctionnalités à travers des « URI » accessibles par le protocole. http. Les messages de réponse peuvent avoir différentes représentations (XML, JSON,. . . ).

* Architecture REST :

REST qui signifie *REPresentational* *State Transfer* est un style d’architecture logicielle permettant de fournir des normes entre les systèmes informatiques sur le Web, ce qui facilite la communication entre les systèmes. Il offre une alternative plus simple et légère, au lieu d’utiliser XML pour faire une requête, REST repose une simple URL.

Les API REST sont basées sur http (HyperText Transfer Protocol), l’échange est basé sur des requêtes client et serveur, Un client lance une requête http et le serveur renvoie une réponse.

REST peut utiliser quatre opérations principales de requête http qui sont : PUT, GET, POST et DELETE :

PUT : La méthode PUT existe pour mettre à jour une ressource sur le serveur.

GET : La méthode GET existe pour récupérer des données d’une ressource, elle demande une représentation de la ressource spécifiée.

POST :  La méthode POST est utilisée pour envoyer des données dans une requête et souvent pour l’ajouter à la ressource précisée dans la partie URI de la première ligne de la requête.

DELETE :  La méthode DELETE supprime la ressource indiquée sur le serveur.

### **Communication angular et spring boot**

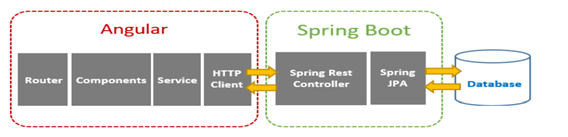


Figure 22 : Architecture Angular et springboot

Template html contient des événements comme click Button qui déclenche l’exécution des fonctions situé à la logique de chaque composant et ces méthodes sont injectées par un service http Client qui est responsable de l’appel des requêtes http GET, Post... à l’application Spring Boot principale. Http Client renvoie un Observable avec l’objet Response.

Lorsque la demande(request) arrive du backend les restcontroller responsable à la réception des request fait intervenir les autres coucher (model, repository, service) via injection (annotation @Autowried) pour réaliser les traitements demander.

### **Communication Android et Spring Boot**

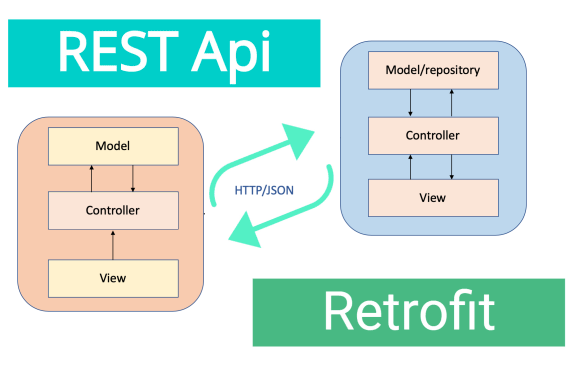


Figure 23 :Communication android et springBoot

L’interface de l’utilisateur contient des événements qui déclenche l’exécution des fonctions situé à la logique de chaque activité et ces méthodes utilisent les services fournis par la bibliothèque retrofit qui permet d’envoyé des demandes à l’application Spring Boot.

Lorsque la demande arrive du backend les restcontroller responsable a la réception des request font intervenir les autres coucher pour réaliser les traitements demander et retourner une réponse sous forme Json.

1. Google Cloud Platform services

Les API Google sont des interfaces de programmation de Google Cloud Platform. Elles constituent un élément clé de Google Cloud Platform, vous permettant d’implémenté des service google facilement à vos applications.

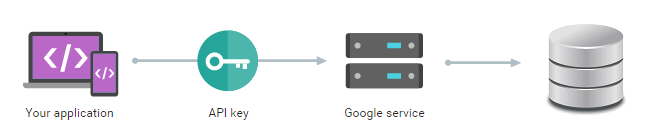


Figure 24 : Consommation google plateforme service

Une clé API est une simple chaîne cryptée utilisée pour appeler certaines API qui n’ont pas besoin d’accéder aux données utilisateurs privés

### **Google Authentication service**

 Google permet aux utilisateurs de se connecter via leur compte Gmail dans leur application Android. La signature Google nous permet d’obtenir très facilement et rapidement tous les détails de l’utilisateur tels que les détails du profil, l’image, les contacts.

L’intégration de la connexion Google dans votre application Android permet aux utilisateurs de se connecter rapidement sans avoir besoin du processus d’inscription complexe.

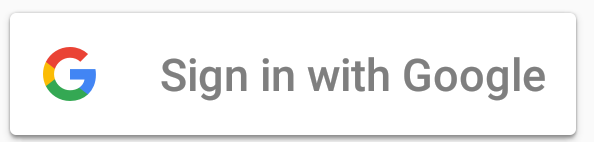


Figure 25 : Bouton google authentification

### **API Google Maps**

La plateforme Google Maps fait partie de la plate-forme Google Cloud. Avec Maps, Routes et Places, il inclut tous les services nécessaires pour développer des cartes interactives et les intégrer dans leurs propres applications

* 1. **API Itinéraire**

Calculer les directions entre les emplacements et recherchez des directions pour plusieurs modes de transport, y compris le transport en commun, la voiture. L’API Directions est un service qui calcule les directions entre les emplacements à l’aide d’une requête http.

Avec l’API Directions, vous pouvez :

* Recherchez des itinéraires pour plusieurs modes de transport, y compris les transports en commun, la voiture, la marche ou le vélo.
* Renvoyez des directions en plusieurs parties à l’aide d’une série de points de cheminement.



Figure 26 :API iternaire

* 1. **API Places pour Android**

Fournissez aux utilisateurs d’applications mobiles des informations supplémentaires sur l’emplacement actuel ou sélectionné via l’API Google Places pour Android et iOS et affichez les emplacements à proximité sur la carte.



Figure 27 :API places android

* Marker

Un marqueur identifie un emplacement sur une carte. Par défaut, un marqueur utilise une image standard. Les marqueurs peuvent afficher des images personnalisées, dans ce cas ils sont généralement appelés « icônes ». Les marqueurs et les icônes sont des objets de type marqueur.

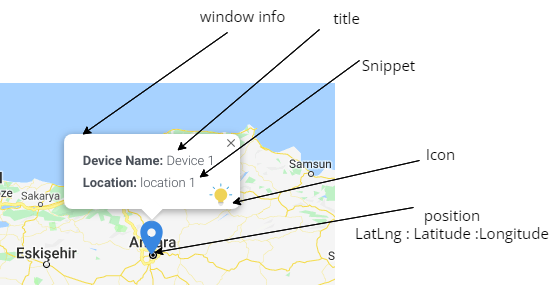


Figure 28 : Caractéristique d’un marker

* GPS

Pour obtenir la position GPS actuelle par programme à partir du GPS dans Android il y a deux méthodes :

Si vous utilisez google map et souhaitez obtenir l’emplacement actuel de l’appareil ; FusedLocation est la meilleure méthode. Si vous n’utilisez pas google map et souhaitez obtenir l’emplacement actuel de l’appareil ; LocationManager est la meilleure méthode.

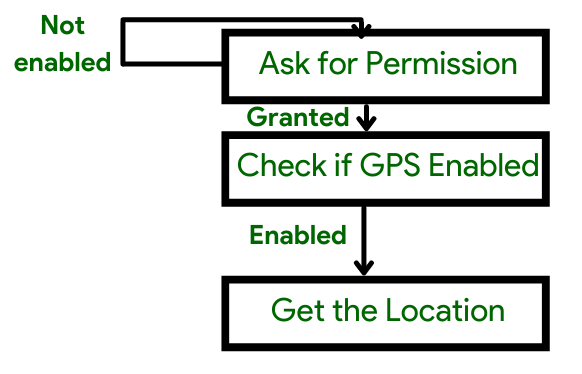


Figure 29 : Comprendre le flux des autorisations

Pour obtenir la position actuelle avec Android :

Il y a deux méthodes :

* Si vous utilisez google map et souhaitez obtenir l’emplacement actuel de l’appareil FusedLocation est la meilleure méthode.
* Si vous n’utilisez pas google map et souhaitez obtenir l’emplacement actuel de l’appareil LocationManager est la meilleure méthode.

1. Google AdMob

Google AdMob permet aux développeurs de gagner de l’argent facilement grâce aux annonces de grande qualité diffusées dans leurs applications mobiles. AdMob maximise la valeur de chaque impression en combinant la demande des annonceurs à l’échelle mondiale, les formats d’annonces innovants et une technologie avancée de monétisation des applications.

* Formats d’annonces
* Bannière :

Les bannières publicitaires sont des annonces textuelles ou illustrées en forme rectangulaire qui occupent une place dans la mise en page d’une application

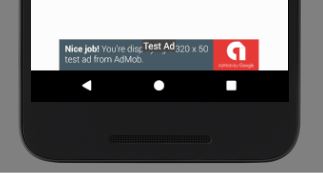


Figure 30 :Annonce bannieérs

* Interstitiel :

Les annonces interstitielles sont des annonces en plein écran qui couvrent l’ensemble de l’interface utilisateur de l’application.

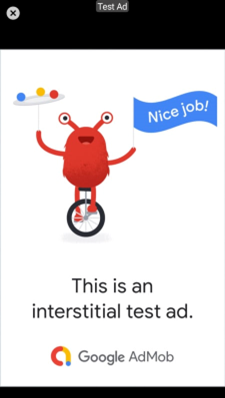


Figure 31 : Annonce Interstitiel

* Annonces natives :

 Ces annonces statiques ou vidéo s’affichent lors des pauses ou des points de transition naturels, ce qui permet de générer des expériences de marque attrayantes sans perturber la fluidité de l’application.



Figure 32 :Annonces native

1. Déploiement

Déploiement de l’application :

* Angular build

Compiler l’application et vérifier les erreurs.

Transpilation : navigateur ne supporte pas typescript inversement pour java script donc il doit être converti ; Ts en Js. Lorsque vous êtes prêt à générer, cependant, vous devez utiliser la ng build commande pour générer l’application (ng build –prod).

* Spring boot build

Build D’abord, il faut définir votre projet avec comme packaging de type war ou jar.

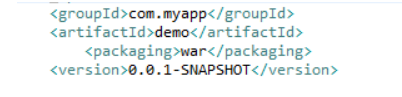


Figure 33 : SpringBoot build

Exécuter en tant que > Installer Maven >Il générera le fichier war dans le dossier target.

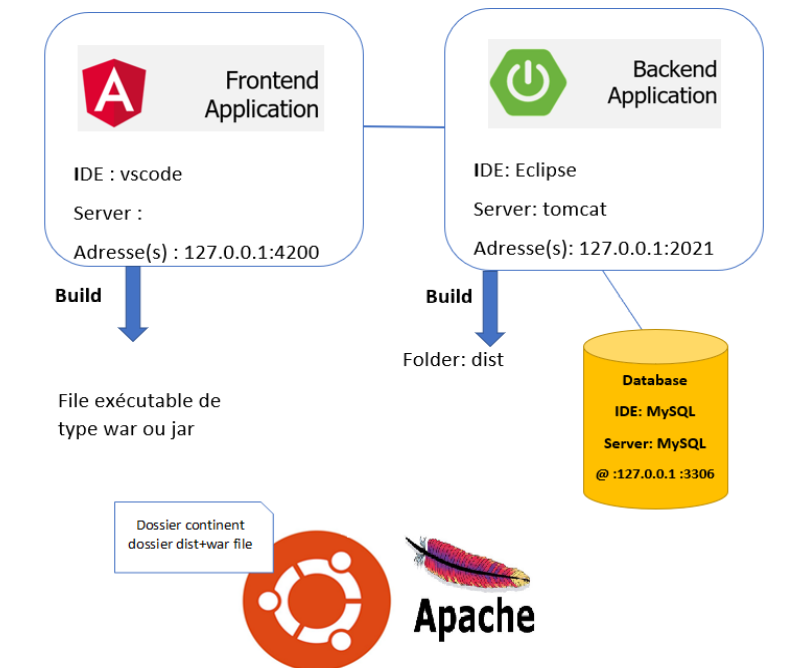


Figure 34 : Déploiement

Pour le déploiement du projet sur Ubuntu (serveur apache) nous recourons aux étapes de déploiement de l’application suivantes : pour la version angular (frontend) déployé sur un serveur virtuel.

-Génération de la version de production de l’application : nous réalisons cette opération via angular cli par la commande “ng build –prod” qui génère ensemble des fichiers de production dans le répertoire “/dist”. C’est ce répertoire qui sera le répertoire virtuel de base lors de notre application sur apache.

-Pour la version spring boot (backend) : Création du fichier war pour le déploiement en production du projet angular-cli sur le serveur Apache Tomcat via outil Installer Maven :

- Intégration Angular avec Spring Boot : Les applications Web Angular et Spring en tant qu’unité déployable (Fichier WAR).

- Nous copions le dossier dist dans le dossier cible Spring ‘’src/main/resources/static’’.

* Android build (Création d’un fichier APK signé)



Figure 35 : Android Build

Apres avoir terminé la phase de conception et de programmation de l’application, l’étape de signature et de remise d’une clé (keystore) à l’application a fire d’être valide pour la publication La première étape est vote travail sur la production du fichier Android- debug.apk et ensuite votre travail est de produire le fichier android-release.apk différence est grande entre eux, le premier n’est pas valide pour la publication et le dernier a une clé et ne signature, et le débogage en a Eta supprime, et donc il est valide pour la publication Google Play rejettera vote application et vous verrez une .erreur indiquant que l’application n’est pas adaptée a la publication.



Figure 36 :APK non signé

1. Conclusion

Ce chapitre comporte les premières étapes nécessaires à la bonne réalisation d’un projet. C’est une étape qui facilite le travail au cours du développement. Toute au long cette partie nous avons utilisés seulement la recherche et la définition des notions.

Chapitre 4 : Mise en œuvre de l’application

Ce chapitre est le dernier volet de ce rapport, il décrit la phase d’implémentation de notre solution. Dans un premier temps, nous présentons les ressources matérielles et logicielles utilisées lors de la réalisation de l’application. Dans un second temps, nous présentons les différentes fonctionnalités de notre application à travers quelques captures d’écran regroupées en sprints.

1. Sprint 0 : Environnement de travail et Configuration du système

Architecture physique

L’architecture physique de notre application est constituée de 3 parties :

• l’appareil téléphonique Android.

• Un serveur Web permettant d’héberger notre API.

• Une base de données qui enregistre les données nécessaires.

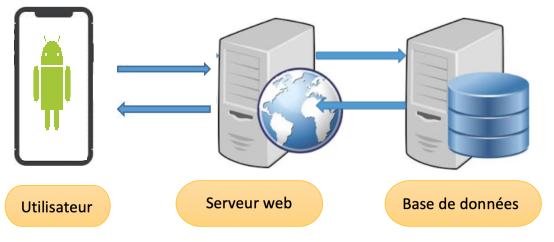


Figure 37:Architecture physique

### **Environment logiciel**

* + Eclipse : Est un environnement de développement intégré libre extensible et universel en s’appuyant principalement sur java.

Figure 38 :Logo eclipse

* + Android Studio : Est un environnement de développement pour développer des applications Android. Il est basé sur IntelliJ IDEA. Il permet principalement d’éditer les fichiers Java et les fichiers de configuration d’une application Android.



Figure 39:Logo android studio

* + - Visual Studio Code : Est un éditeur de code source léger et puissant, open source disponible pour Windows, MacOs et linux.

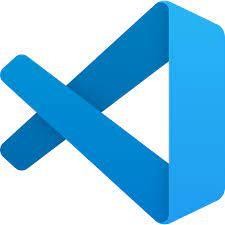


Figure 40: Logo Visual studio code

* + Postman : Une plateforme API complète permet d’exécuter des requêtes, tester, déboguer, créer des serveurs factices, surveiller, exécuter des tests automatisés et documenter une API.

Figure 41:Logo Postman

* + PuTTY : un logiciel permet d’effectuer des connexions à distance avec ssh (Secure Socket Shell).



Figure 42: Logo PuTTY

* + - Workbench : un logiciel qui vous permettra de gérer et d’administrer des bases de données MySQL, et ce, via une interface graphique.



Figure 43:Logo Workbench

* + VPS : une abréviation de *Virtual Private Server* en anglais. C’est un serveur logique ou virtuel plutôt qu’un serveur physique. Un serveur physique peut héberger plusieurs serveurs virtuels.



Figure 44:logo Virtual Private Server

* + Cloudinary : une solution de gestion dynamique des actifs pour vos sites Web et applications mobiles, couvrant tout, des téléchargements d’images et de vidéos à l’optimisation et à la livraison.



Figure 45:Logo Cloudinary

* + Visual Paradigm : un logiciel de modélisation UML *Visual Paradigm for UML.* C’est un bon outil pour réaliser des spécifications.



Figure 46 : Logo logiciel Visual Paradigm

* + GitHub : un site web et un service de cloud qui aide les développeurs à stocker et à gérer leurs codes. Ainsi qu’à suivre et contrôler les modifications qui leurs sont apportées.



Figure 47 :Logo GitHub

* + Photoshop : Est un logiciel de retouche, de traitement et de dessin assisté par ordinateur édite par Adobe. Il est principalement utilisé pour le traitement de photographies numériques, mais sert également à la création d’images.



Figure 48 :Logo Photoshop

* + Slack : Est une plateforme de communication collaborative propriétaire ainsi qu’un logiciel de gestion de projets.

Shape

Description automatically generated with medium confidence

Figure 49 : Logo slack

* + Figma : Est une plateforme de conception pour les équipes qui construisent des produits ensemble. Né sur le Web, Figma aide les équipes à créer, partager, tester et livrer de meilleures conceptions du début à la fin. Qu’il s’agisse de consolider des outils, de simplifier les flux de travail ou de collaborer entre équipes et fuseaux horaires, Figma rend le processus de conception plus rapide, plus efficace et plus amusant tout en gardant tout le monde sur la même longueur d’onde.

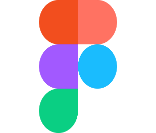


Figure 50 : Logo Figma

### **Environnent Matériel**

Voici les caractéristiques techniques des machines que nous avons utilisées :

|  |  |
| --- | --- |
| Pc asus | • processeur: Core i5,  • ram: 12 GO,  • disque Dur : 1 TB,  • système d’exploitation : Windows |
| Pc Hp | • processeur: Core i7,  • ram: 16 GO,  • disque Dur : 1 TB,  • système d’exploitation : Windows |
| Smartphone Samsung j4+ | • ram : 4GO  • système d’exploitation : Android 9.0 |
| Smartphone Samsung s8 | • ram : 4GO  • système d’exploitation : Android 9.0 |

Tableau 7 :Environnement matériel

### **Préparation de la machine virtuelle**

Dans le monde professionnel tout le jeu de tester doit être appliquer sur une machine virtuelle puis refaire la même procédure dans *virtual private system* (VPS)

* Configuration de la machine virtuelle

Il faut connaître la configuration de VPS et installer une machine de même configuration sur notre machine.

* L’accès au VPS

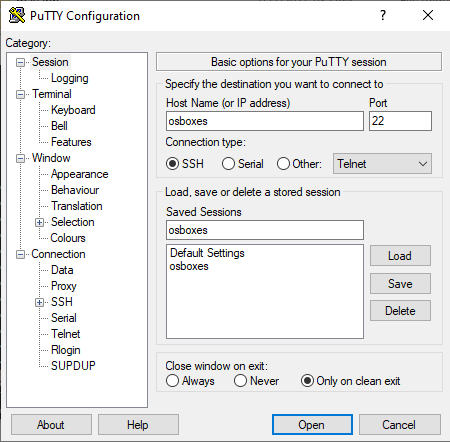


Figure 39 : Connexion au serveur distant

* Installation et configuration Apache Tomcat

Etape 1 : installer java

Vérifier si Java est installé

Etape 2: installer tomcat Text

Description automatically generated

Figure 51: Default Tomcat home page

* Test l’état de tomcat
* Lancer server tomcat: 192.68.1.9:8080

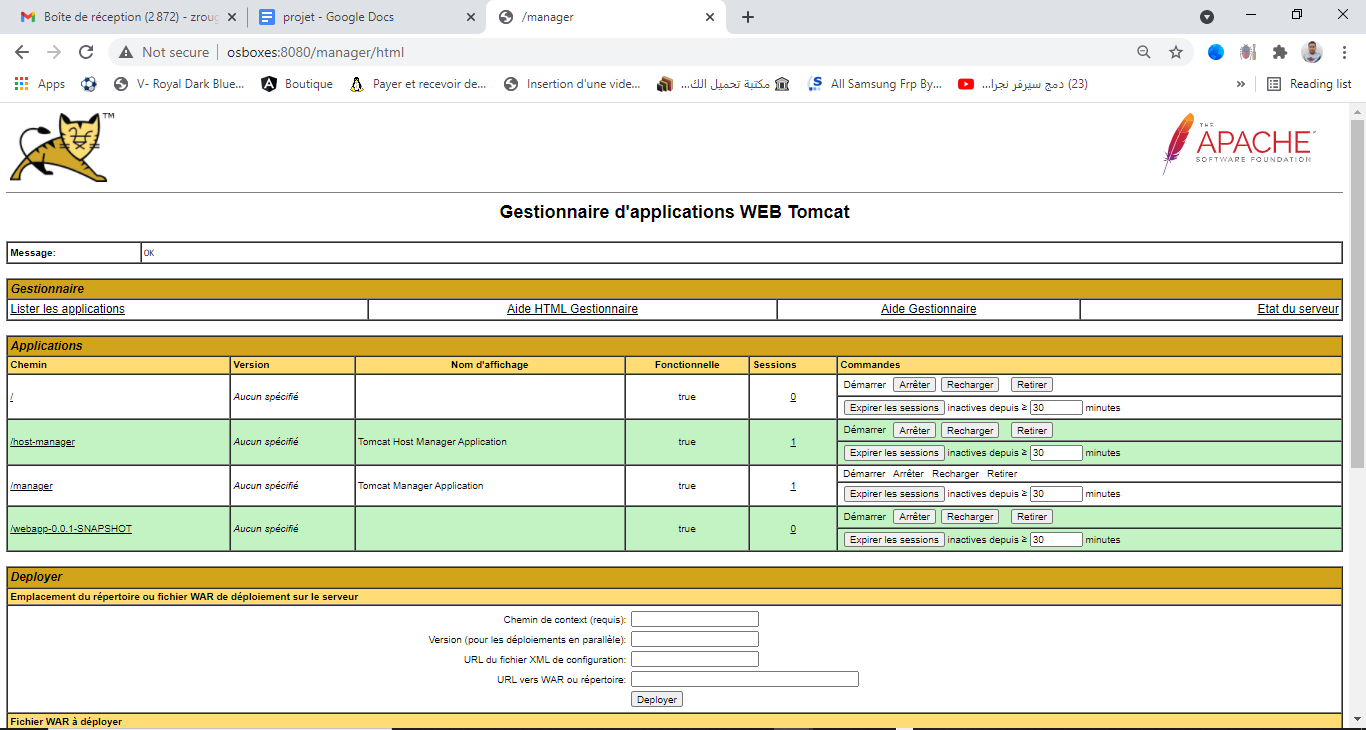


Figure 52 :Tomcat Webapp Manager

* Installation et configuration de Mysql
* Installation de mysql
* Lancer Mysql et Créer un utilisateur
* Configuration de VPS (ovh)

- Répéter la même procédure appliquée à la machine virtuelle.

### **Conclusion**

Dans cette partie nous avons présenté l’environnement logiciel et matériel utilisé et nous avons passé en revue parles aux diffèrent étapes de préparation de l’environnement de déploiement.

1. Sprint 1 implémentation des APIs

### **Analyse du sprint implémentation des APIs**

Pour la partie backend nous avons adopté l’architecture MVC. Cette architecture présente un cadre d’organisation de code en accord avec sa nature. Elle permet une séparation du code en trois couches : la couche modèle, la vue et le contrôleur.

On applique pattern MVC qui décompose backend Springboot en 4 packages :

Model : création des classes model

Repository : création des interfaces repository, chaque entité possède une interface héritée de l’interface Jpa Repository.

Service : chaque repository est injecté dans une ou plusieurs classe service (instance singleton) chaque service correspond à une classe model.

Controller : chaque service est injecté dans un ou plusieurs classe Controller (instance singleton) chaque controller correspond à une classe model.

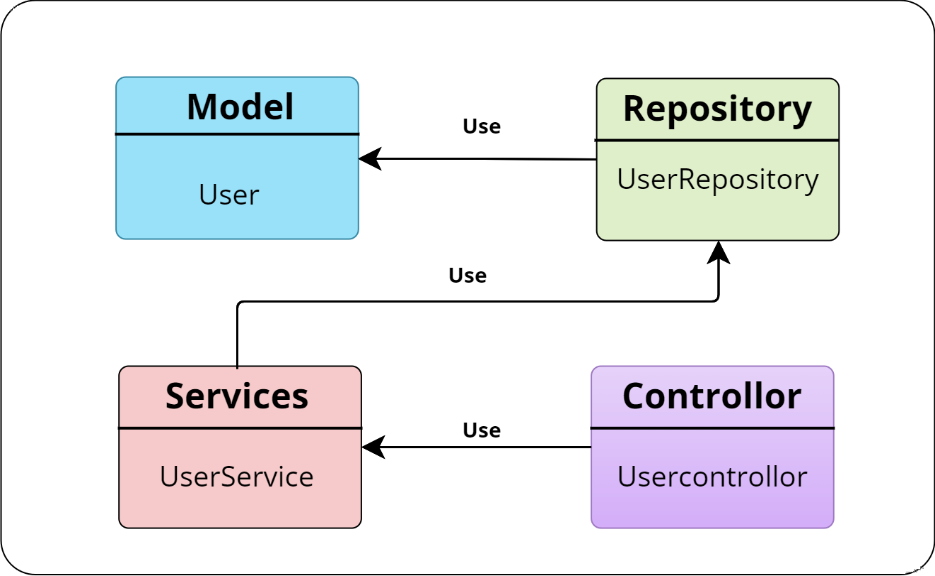
Exemple prototype de notre application (gestion utilisateurs)

Figure 53 : Prototype du diagramme par package

Diagramme de classe par package

Le diagramme de packages/diagramme de paquetages permet de découper l’ensemble des classes (d’un modèle de classes) en sous-ensembles de classes en fonction de leurs dépendances, d’une tâche commune. En clair, on essaye de regrouper toutes les classes qui sont fortement dépendantes entre elles et faiblement dépendantes avec le reste des classes. Notre diagramme de classe par package selon backend spring boot contient 4 packages illustrer dans la figure suivante :

Model : contient les classes entités formé par 5 tables :

• Table User : c’est une table associative entre table Conversation et pharmacie : chaque user

peut avoir un ou plusieurs conversations et une seule pharmacie.

• Table pharmacie : c’est une table qui contient les informations nécessaires pour une pharmacie, chaque pharmacie peut avoir plusieurs produits.

Services : formé par 5 sous classe qui hérite de la classe parent abstract services, chaque service traite une classe entité c’est t-à-dire manipuler une table dans la base de données.

Repository : contient l’interface implémentée a une interface JPARepository qui contient à son tour des méthodes CRUD (Delete, Save, Find, Findall …) pour manipuler la source de données ces interfaces est consommée par les classes services.

Controller : contient la classe responsable a la réception et la réponse de la demande des APIs du frontend (client) en utilisant les classes services.

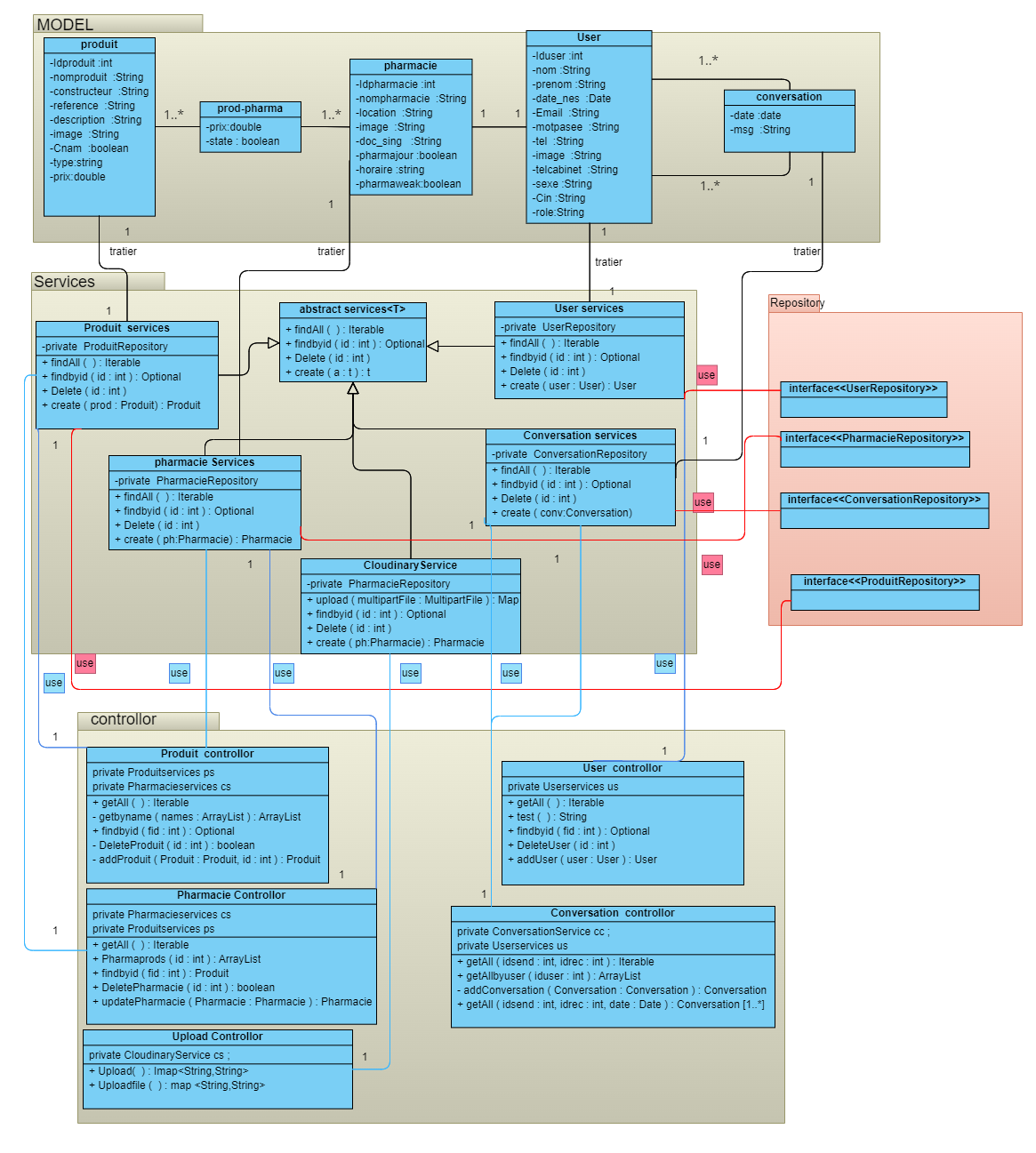
Figure 54 : Diagramme de class par package

Diagramme de déploiement :

Le diagramme de déploiement est un diagramme UML qui montre la configuration physique des différents éléments qui participent à l’exécution du système, ainsi que les instances de composants qu’ils supportent. Il est constitué de « nœuds » connectés par des liens physiques. Les principaux nœuds de notre diagramme de déploiement sont :

* Le Client : c’est le navigateur ou appareil Android, il permet à un utilisateur d’accéder au serveur.
* Le serveur : C’est le serveur principal qui abrite les différents composants logiciels de notre application. Ces composants sont entre autres :
* L’Application Web : l’application proprement dit qui est déployé sur le serveur web.
* Le Serveur Web : assure la gestion des connexions et des requêtes du client.
* La Base de données : c’est le composant qui s’occupe du stockage et de la gestion des données.

On a utilisé dans notre projet les connexions suivantes :

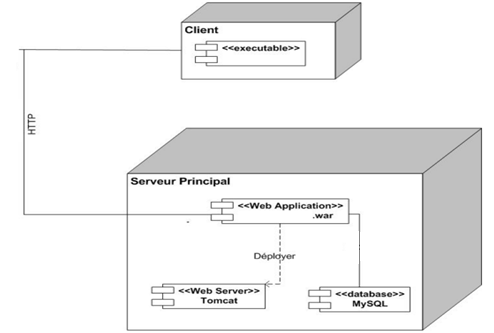


Figure 55 : Diagramme de déploiement

HTTPS : c’est un protocole qui permet de récupérer les données dont le contenu est formaté en JSON par le serveur.

### **Revenu du sprint implémentation des APIs**

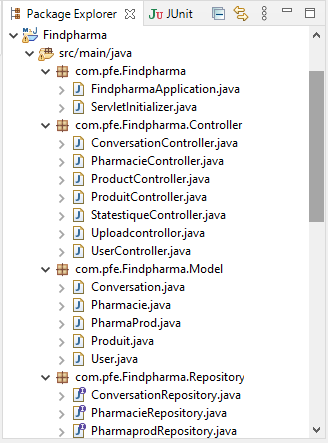


Figure 56 :Arborescence partie backend

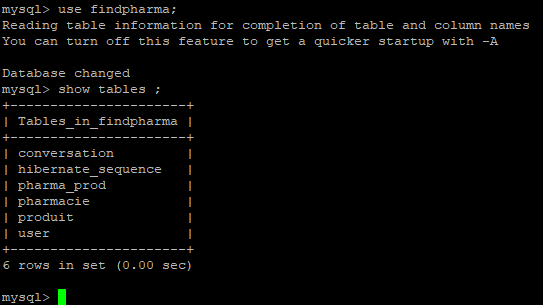


Figure 57:Find pharma data base

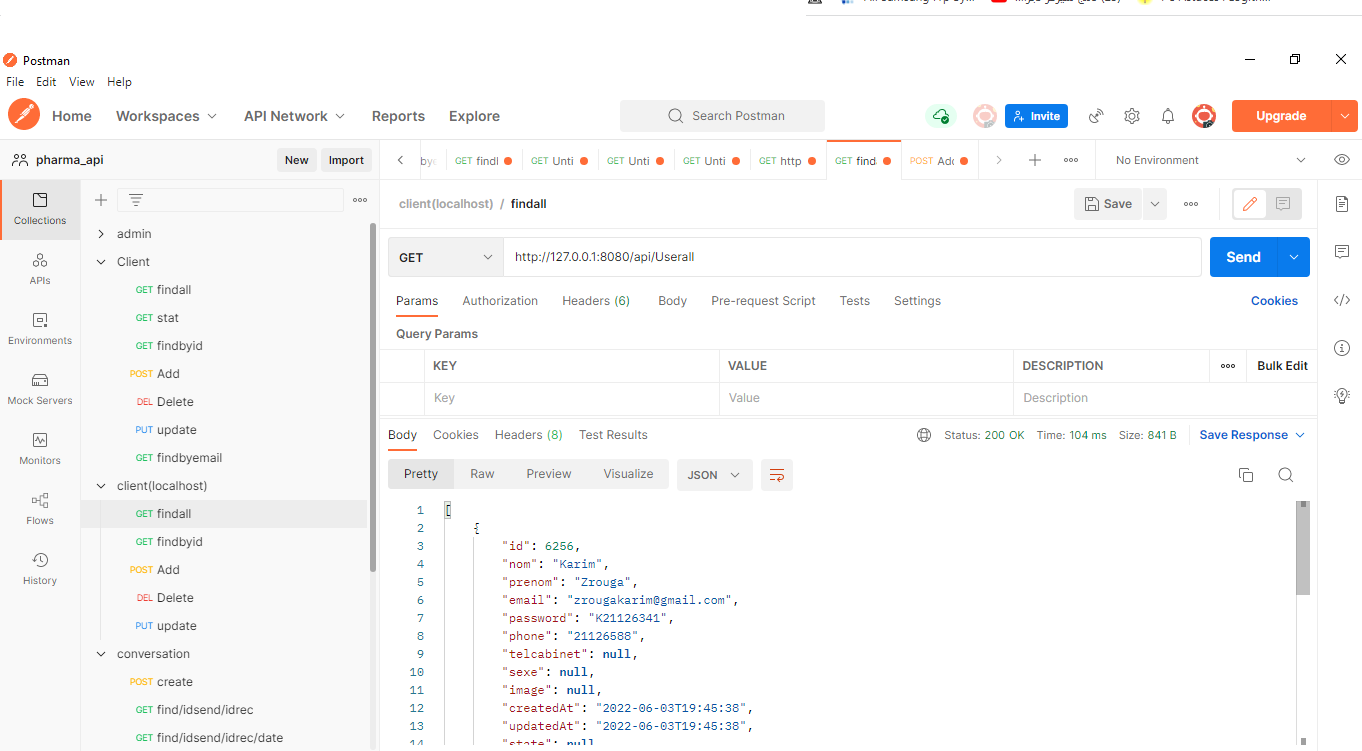


Figure 58 : Tester les API avec la logiciel Postman

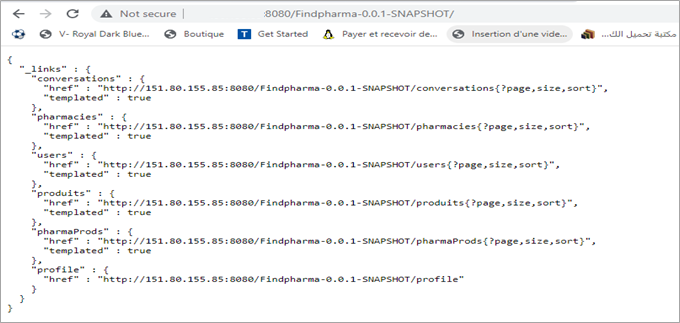


Figure 59 : Endpoint après déploiement

### **Conclusion**

Ce sprint a été consacré à la présentation de l’architecture de backend et le développement de toutes les API. Enfin, nous avons modéliser le diagramme de déploiement de notre système et tester les API’s développée.

1. Sprint 2 : Authentication et gestion de profile

### **Analyse du sprint Authentication et gestion de profile**

Dans cette partie nous présentons la phase d’analyse de sprint 1 par un diagramme des cas d’utilisation et les diagrammes des séquences et la description textuelle et les notions appliquées.

* Diagramme de cas d’utilisation sprint 1

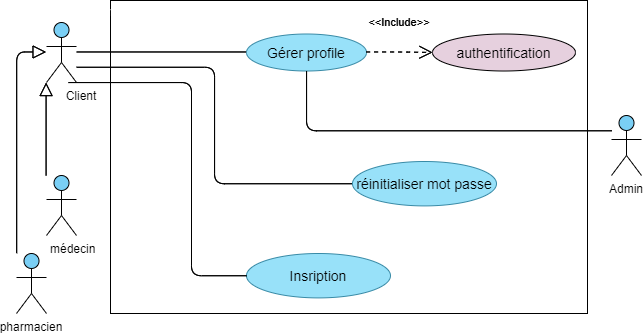


Figure 60 : Diagramme de cas d’utilisation sprint 1

* Description textuelle

|  |  |
| --- | --- |
| Cas d’utilisation | Inscription |
| Acteur | Médecin, Client, Pharmacien |
| résumer | L’utilisateur peut créer un compte dans l’application |
| Précondition | Inscription préalable |
| Postcondition | Une fois l’utilisateur est inscrit, il a l’accès à tous les services de l’application |
| Scénario nominal | 1. L’utilisateur demande de s’inscrire. 2. Le système lui affiche le formulaire spécifique d’inscription. 3. L’utilisateur remplit le formulaire attentivement. 4. Le système valide les champs et envoie un code de validation 5. L’utilisateur saisit le code de validation 6. L’utilisateur sera dirigé vers la page d’accueil |
| Scénario alternative | 1. Si le formulaire contient des champs vides 2. 1.Un message d’erreur est affiché 3. Si l’utilisateur entre une adresse email incorrecte 4. 1.Un message d’erreur est affiché 5. Si l’utilisateur entre un mot de passe ne vérifiant pas le critère demandé : 6. 1.Le système affiche un message d’erreur à l’utilisateur. 7. Si l’utilisateur entre un code de validation incorrect 8. 1.Le système affiche un message d’erreur à l’utilisateur. |

Tableau 8 : Description Textuelle « Inscription »

|  |  |
| --- | --- |
| Cas d’utilisation | Authentication |
| Acteur | Médecin, Client, Pharmacien, Administrateur |
| résumer | L’utilisateur doivent se connecter afin de profiter  de tous les services |
| Précondition | Inscription préalable |
| Postcondition | Une fois l’utilisateur est connecté, il a l’accès à toute l’application. |
| Scénario nominal | 1. L’utilisateur saisit son e-mail et son mot de passe. 2. Le système vérifie l’existence de l’e-mail et de mot de passe dans la base de données. 3. L’utilisateur est dirigé vers l’accueil |
| Scénario alternative | 1. L’utilisateur saisit des coordonnées in- correctes :   Le système vérifie qu’une des coordonnées n’existe pas dans la base de données et affiche un message d’erreur |

Tableau 9 :Description Textuelle « Authentication »

|  |  |
| --- | --- |
| Cas d’utilisation | réinitialiser mot passe |
| Acteur | Médecin, Client, Pharmacien |
| résumer | L’utilisateur peut récupérer son mot de passe s’il l’a oublié. |
| Précondition | Inscription préalable |
| Postcondition | Après avoir saisi correctement l’e-mail, le client a un nouveau formulaire pour changer le mot de passe. |
| Scénario nominal | 1. L’utilisateur navigue dans l’interface de mot de passe oublié. 2. L’interface de changer mot de passe est affichée pour l’utilisateur. 3. L’utilisateur doit saisir son e-mail correctement. 4. Le system envoie un code de confirmation a l’utilisateur. 5. L’utilisateur saisit le code de confirmation 6. L’interface de saisir à nouveau le mot de passe est affichée pour l’utilisateur. 7. L’utilisateur saisit de nouveau le mot de passe 8. Le system revient à l’interface de l’authtification. |
| Scénario alternative | 1. Si le client entre une adresse e-mail incorrecte : 2. 1.Un message d’erreur est affiché 3. Si le client entre un mot de passe ne vérifiant pas le critère demandé : 4. 1.Le système affiche un message d’erreur à l’utilisateur. |

Tableau 10 : Description Textuelle « réinitialiser mot passe »

* Diagramme de séquence

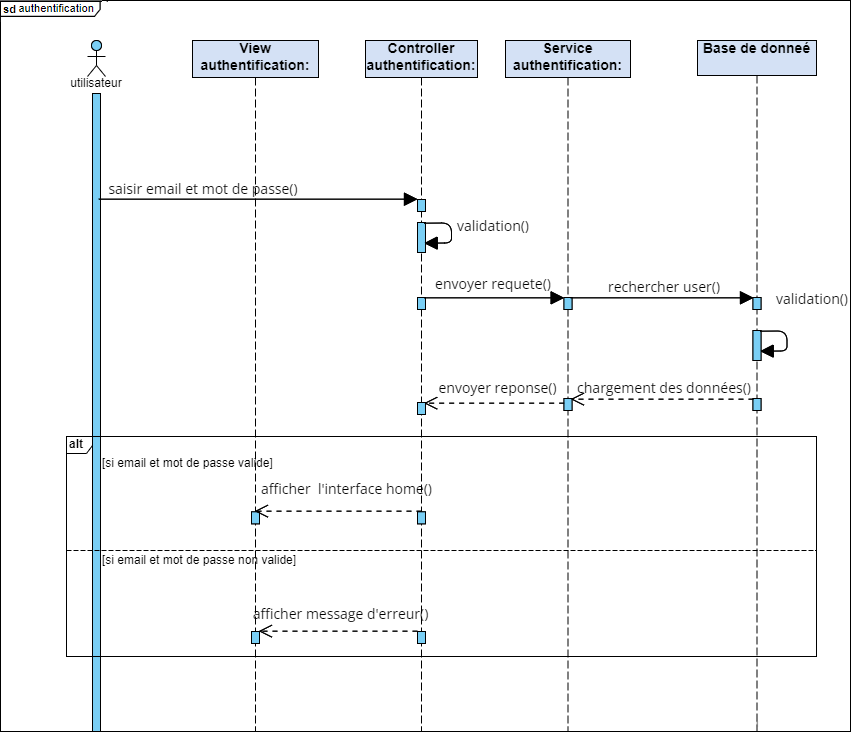


Figure 61 : Diagramme de séquence d’authentification

* L’authentification est une obligation pour avoir accès à tous les services de l’application. L’authentification se fait via l’interface appropriée en saisissant correctement l’adresse e-mail et le mot de passe. Si les coordonnées sont valides, l’utilisateur sera redirigé vers l’interface d’accueil. Dans le cas opposé, un message d’erreur sera affiché.

A picture containing calendar

Description automatically generated

Figure 62 : Diagramme de séquence d’inscription

* Notre application fournit à ses utilisateurs des services comme la localisation de la pharmacie la plus proche. Pour Profitez de tous les services de l’application, l’utilisateur doit passer par le Processus d’inscription et remplir correctement les formulaires requis.

Calendar

Description automatically generated with medium confidence

Figure 63 : Diagramme de séquence oublier mot de passe

* Notre application offre à ses utilisateurs la possibilité de réinitialiser les mots de passe. En cliquant sur Mot de passe oublié, une interface de réinitialisation de mot de passe apparaîtra, l’utilisateur doit saisir son adresse e-mail. Si cette adresse e-mail est fonctionnelle, il recevra un message qui contient un code de validation et le système le redirigera vers l’interface Changement de mot de passe sinon un message d’erreur s’affichera. Si le nouveau mot passe saisit confirme les critères demandés alors le système redirigera l’utilisateur vers l’interface d’authentification pour se connecter avec son nouveau mot de passe. Sinon, un message d’erreur s’affichera.

### **Notion Application**

* Authentification avec google

1. Créer un projet Google

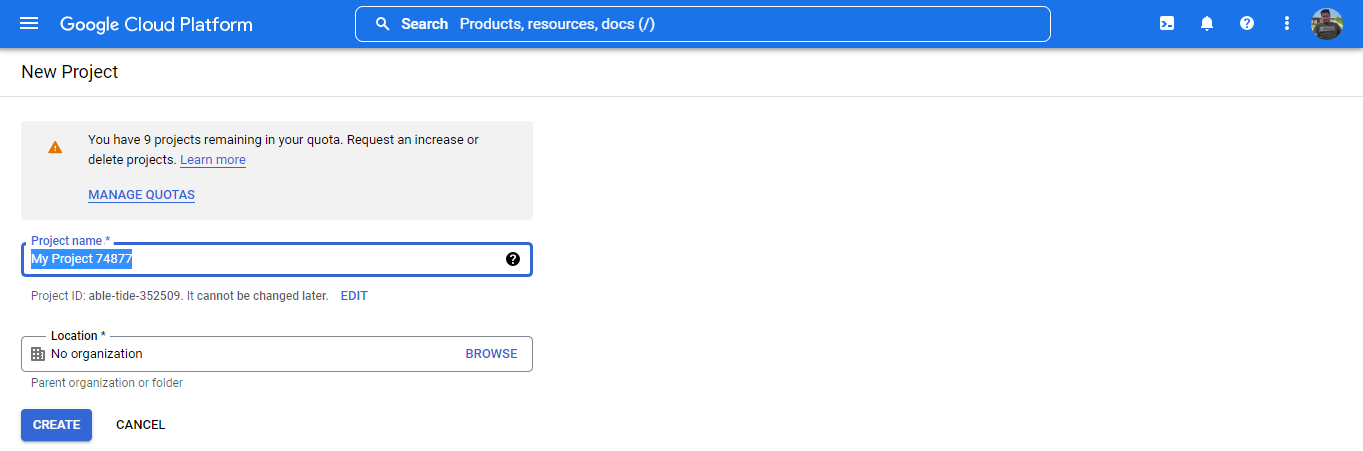
Pour utiliser les services offerts par google cloud plateforme il faut d’abord avoir un compte google

Figure 64 : Gérer un projet sur google cloud plateforme

1. Configuration de Google Sign

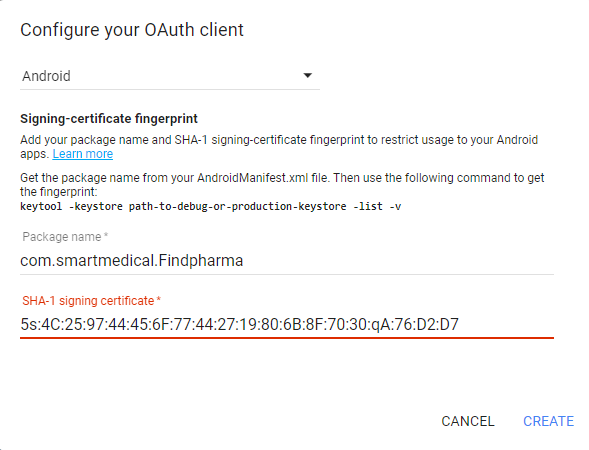


Figure 65 : Configuration de google sign

D’abord pour configurer le projet sur la console développeur de Google, il faut avoir un compte google.

1. Entrez le nom du projet si vous ne créez pas de projet.
2. Entrez ensuite le nom du package de projet selon votre choix.
3. Ajoutez notre clé SHA1.
4. Ajoutez le fichier google-services. Json au dossier d’application de votre projet
5. Ajoutez la dependence



1. Ajoutez un bouton pour la connexion
2. Configuration classe logique
3. Obtenez les informations du profil :

Après avoir connecté un utilisateur à Google on peut accéder aux informations de base du profil de l’utilisateur. Vous pouvez également obtenir leur adresse e-mail, nom…

### **Revenu du sprint Gestion d’accès**

L’authentification est une étape primordiale dans notre application. Cette interface permet à l’utilisateur d’insérer son e-mail et son mot de passe. Si les coordonnées s’avèrent incorrectes, l’interface sera actualisée pour inviter l’opérateur à vérifier ses paramètres, une fois les données sont valides l’utilisateur aura accès à notre application

* Authentification

Lors de l’exécution de l’application, la première interface qui s’affiche c’est celle de l’authentification qui contient deux interfaces ; une pour l’authentification comme montre la figure 65, ainsi qu’un Button pour s’authentifier avec un compte google comme nous le voyons sur les figures 66 et 68.

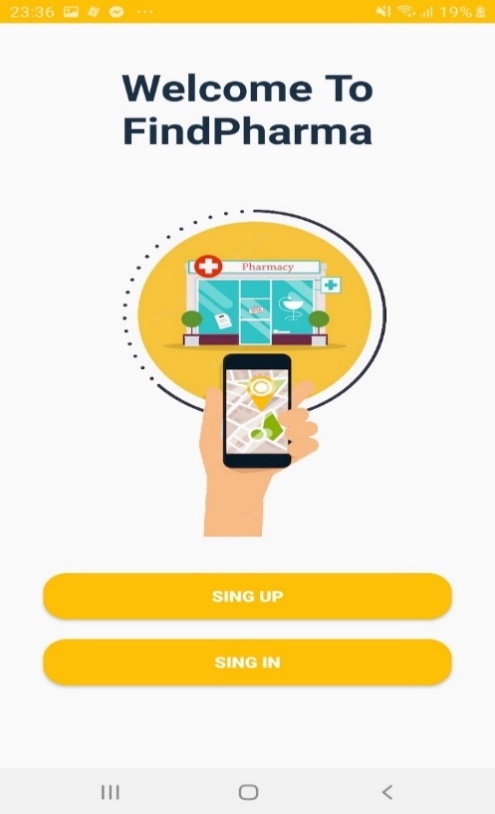
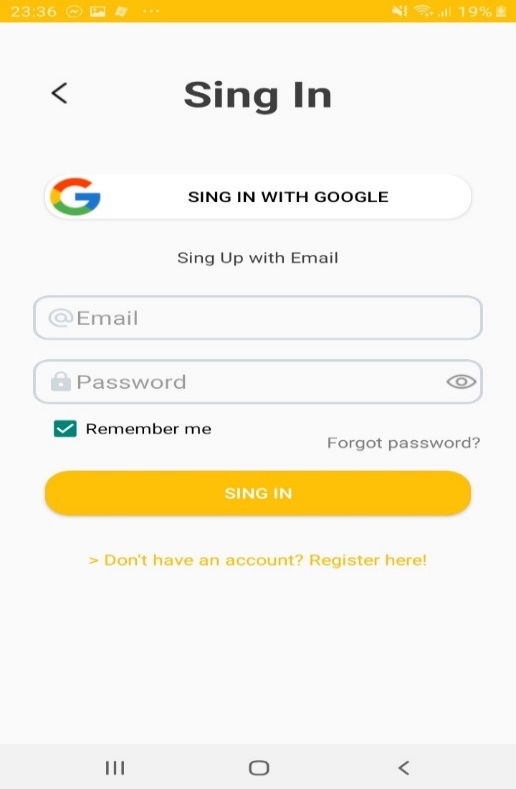


Figure 66:interface inscription

Figure 67:interface d’accueil

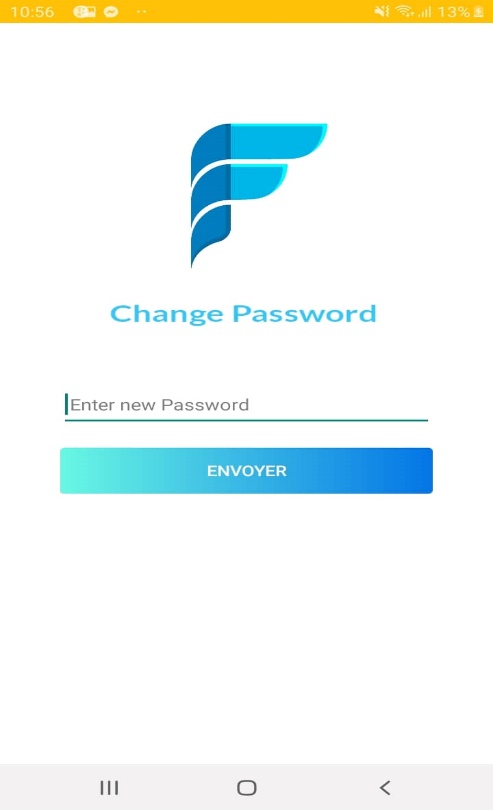
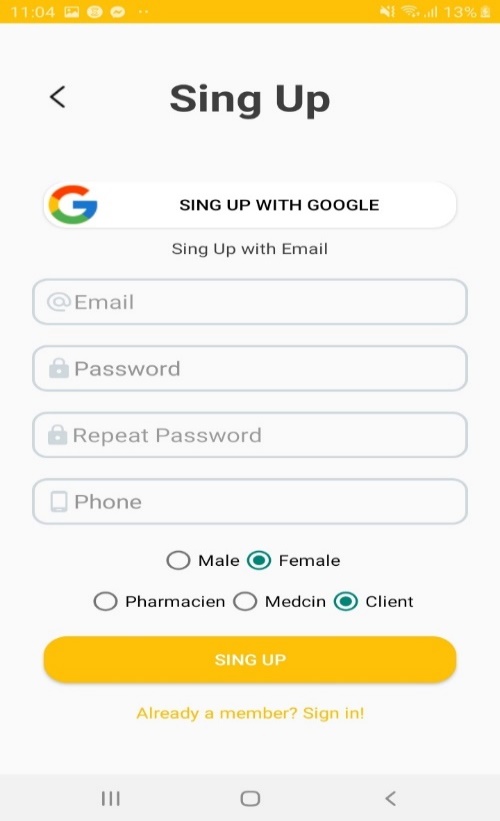
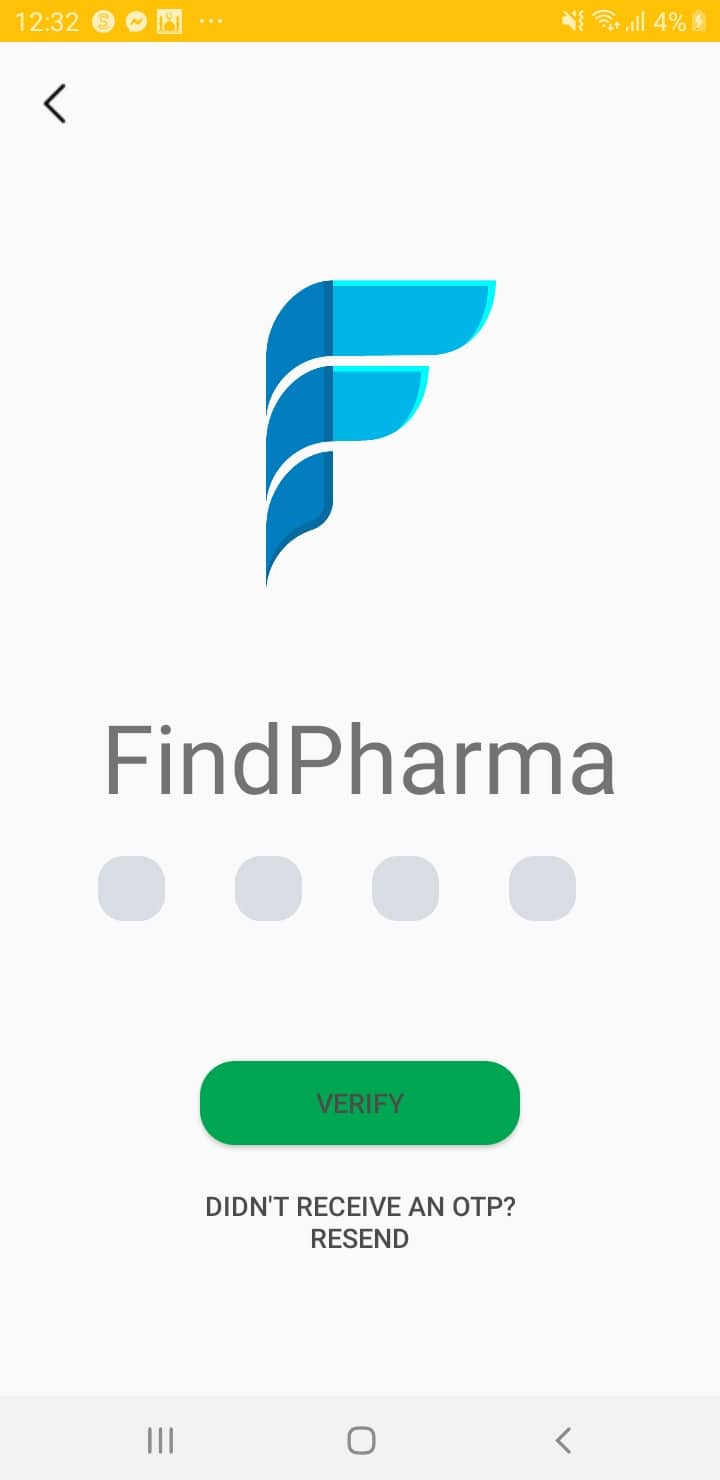
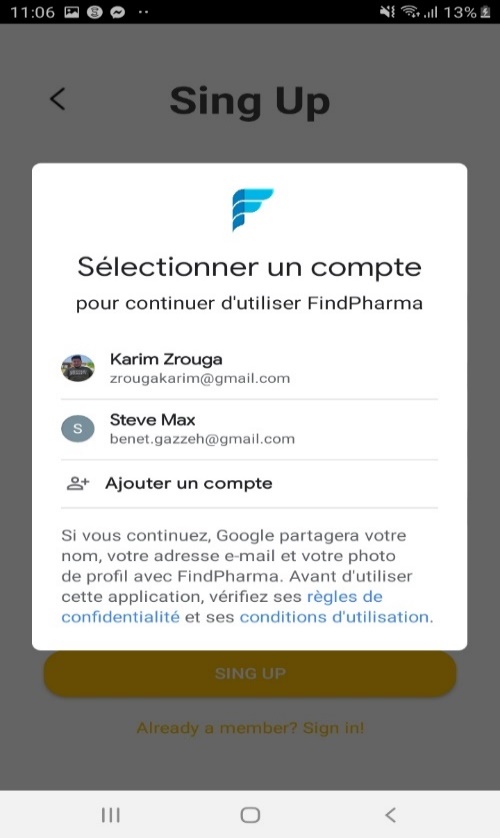


Figure 68: authentication avec google

Figure 69: interface inscription

Figure 71 : interface changer mot de passe

Figure 70: interface validation

* Gestion profile

Apres qu’un utilisateur soit authentifié, il peut accéder à l’interface de ces informations et peut par suite modifier son compte comme montre les figures suivantes.

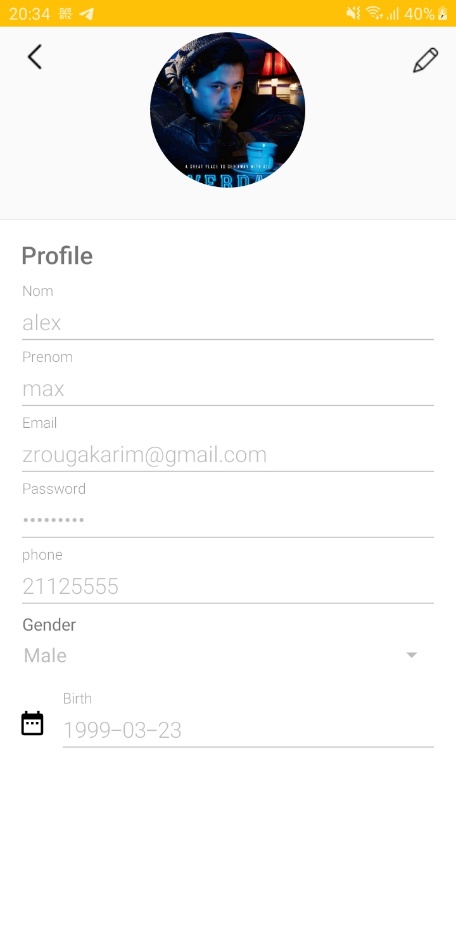
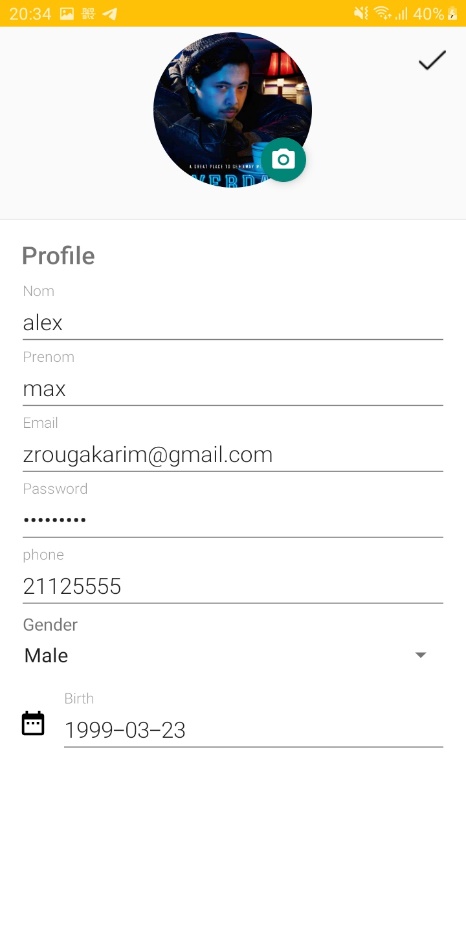


Figure 73 : interface profile utilisateur

Figure 72: interface édit profile utilisateur

### **Conclusion**

Dans ce deuxième sprint nous avons présenté les processus de l’authentification et l’inscription, nous avons cité les étapes de l’authentification avec le service google et pour l’authentification En second lieu, nous avons illustrée la partie réalisation avec quelques interfaces de notre application.

1. Sprint 3 : Recherche pharmacie

### **Analyse de sprint Recherche pharmacie**

Dans cette partie nous présentons la phase d’analyse de sprint 3 par un diagramme des cas d’utilisation, les diagrammes de séquences et la description textuelle et les notions appliqués.

* Diagramme de cas d’utilisation

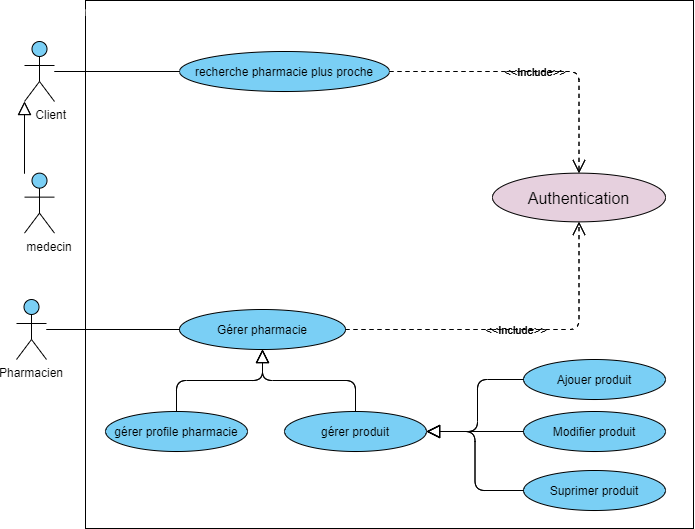
La figure montre le diagramme de séquence ou le pharmacien peut gérer sa pharmacie et le client peut rechercher les médicaments dans la pharmacie la plus proche.

Figure 74 : Diagramme de cas d’utilisation sprint 3

|  |  |
| --- | --- |
| Cas d’utilisation | Rechercher pharmacie |
| Acteur | Médecin, Client |
| résumer | L’utilisateur peut voir la disponibilité des médicaments dans la pharmacie la plus proche |
| Précondition | Authentification |
| Post-condition | Les positions de pharmacie(marker) sera affichée sur le MAP. |
| Scénario nominal | 1-Le système affiche le MAP  3- L’utilisateur saisit le nom d’un médicament  4-Le système recherche le besoin de l’utilisateur  5- Le système affiche la pharmacie la plus proche qui possède les médicaments |
| Scénario alternative | 1 le nom de médicaments est incorrects  1.1Le système affiche un message d’erreur |

Tableau 11 : Description Textuelle « Recherche pharmacie »

|  |  |
| --- | --- |
| Cas d’utilisation | Gérer profile pharmacie |
| Acteur | Pharmacien |
| résumer | L’utilisateur peut modifier les informations de sa pharmacie. |
| Précondition | Authentification |
| Post-condition | Opération demandé réussite |
| Scénario nominal | 1-Le système affiche le formulaire remplit  3- pharmacien demande de modifier l’formation de la pharmacie.  2- pharmacien modifier le formulaire.  3- Le système lui affiche un formulaire qui permet d’effectuer des changements.  Le système vérifie la validité des nouvelles modifications.  4.Le système envoie ces modifications à la base de données |
| Scénario alternative | 1. Si le formulaire contient un champ vide    1. Le système affiche un message d’erreur. |

Tableau 12 : Description Textuelle « Gérer profile pharmacie »

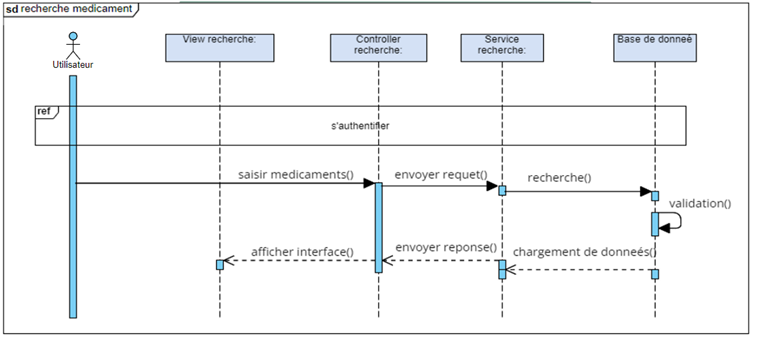
* Diagramme de séquence

Figure 75 : Diagramme de séquence recherche médicaments

➔ Apres l'authentification et la recherche de pharmacie le médecin peut accéder à l’interface de discussion et envoyer des messages au pharmacien.

### **Notion appliquer**

Étape 1. Activer les API

Pour l’utilisation de l’API Cloud, vous devez l’activer pour votre projet.

Pour activer une API, procédez comme suit :

Ouvrez la bibliothèque d’API Google Cloud Console.

* Sélectionnez un projet dans la liste des projets ou créez-en un nouveau.
* Dans la bibliothèque d’API, sélectionnez l’API que vous souhaitez activer.
* Cliquez sur ACTIVER sur la page API.

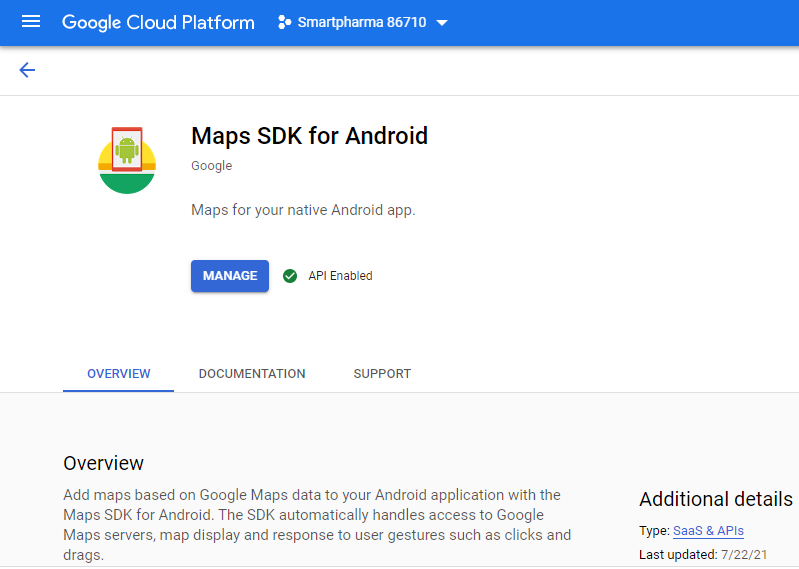


Figure 76 :Maps SDK

Étape 2 : Obtention des clés API et l’utilisation avec Android

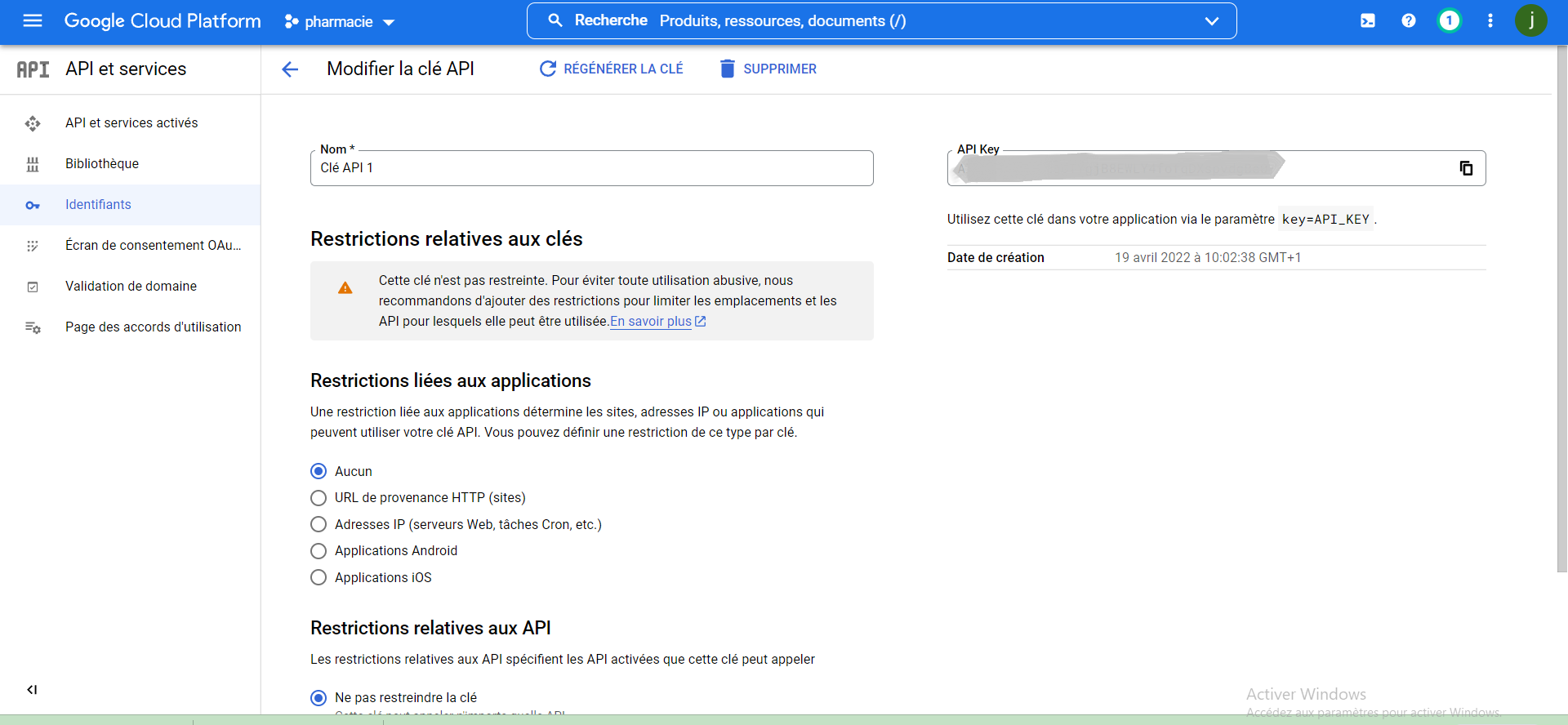
* Obtention de clés API google maps

Figure 77 : API Google MAPS

* Implémentation google maps avec Android

1. Ajout de Api key et permission nécessaire

Pour utiliser Service google map, vous devez déclarer cette clé (key) dans AndroidManifest.xml.



Figure 78:API key



Figure 79: google maps permission

ACCESS FINE LOCATION :

Celui-ci utilise votre GPS, ainsi que le WIFI et les données mobiles, pour déterminer votre position exacte. Vous recevrez l’emplacement exact de votre utilisateur si vous l’utilisez.

ACCESS COARSE LOCATION :

L’inclure dans votre application vous permet d’identifier la position de l’appareil via le WIFI ou les données cellulaires mobiles (ou les deux). En utilisant cette autorisation, l’approximation est proche du niveau de la ville.

ACCESS BACKGROUND LOCATION :

Si vous souhaitez accéder à l’emplacement de l’utilisateur en arrière-plan, vous devez également ajouter l’autorisation ACCESS BACKGROUND LOCATION en plus de l’une des deux autorisations précédentes.

1. Ajouter dépendance nécessaire

Vous devez ajouter la bibliothèque Google Maps à votre projet. Sur Android Studio

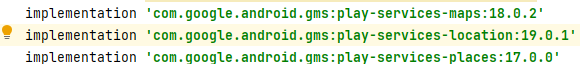


Figure 80: Dependance

1. ﻿Ajouter une carte﻿ à une application

* Le moyen d’ajouter une carte à une application consiste à la spécifier dans le fichier XML de mise en page de l’interface utilisateur pour une activité par l’ajout d’une l’instance Fragment
* La classe activitée doit être implémentés dans l’interface OnapReadyCallback en redéfinitionnant les méthodes nécessaires.
* Implémentation google directions avec Android

A travers cet partie, nous allons montrer comment générer un itinéraire entre deux emplacements dans notre application.

1. Les mêmes étapes précédant (google maps)
2. Get Current location
3. Créer une fonction pour générer l’URL de direction
4. Créer une fonction pour décoder la polyligne
5. Ajouter event listener pour markers

### **Revenu de sprint Recherche pharmacie**

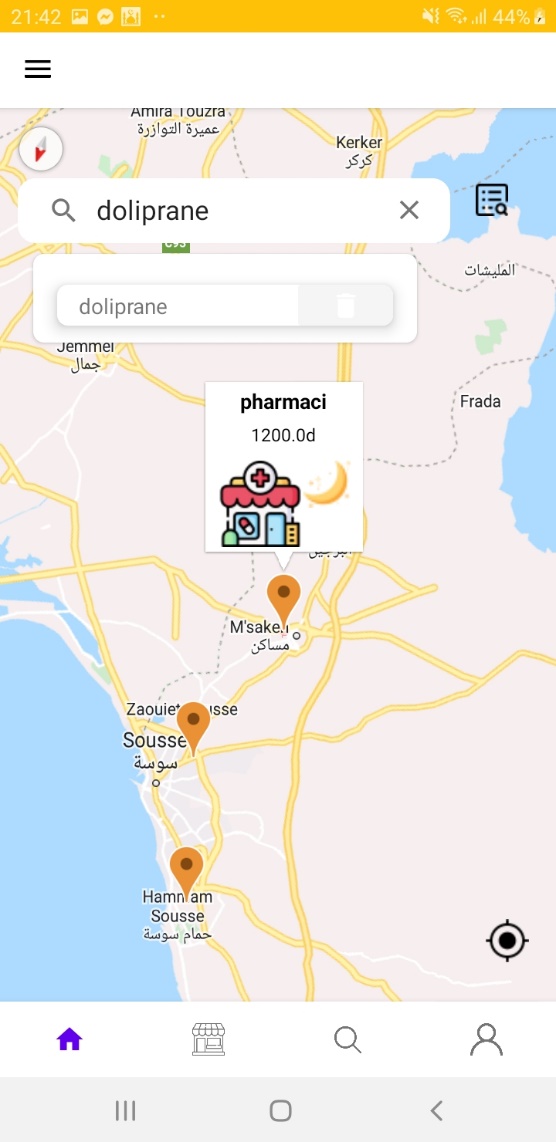


Figure 81:interface d'accueil

Figure 82: recherche pharmacie

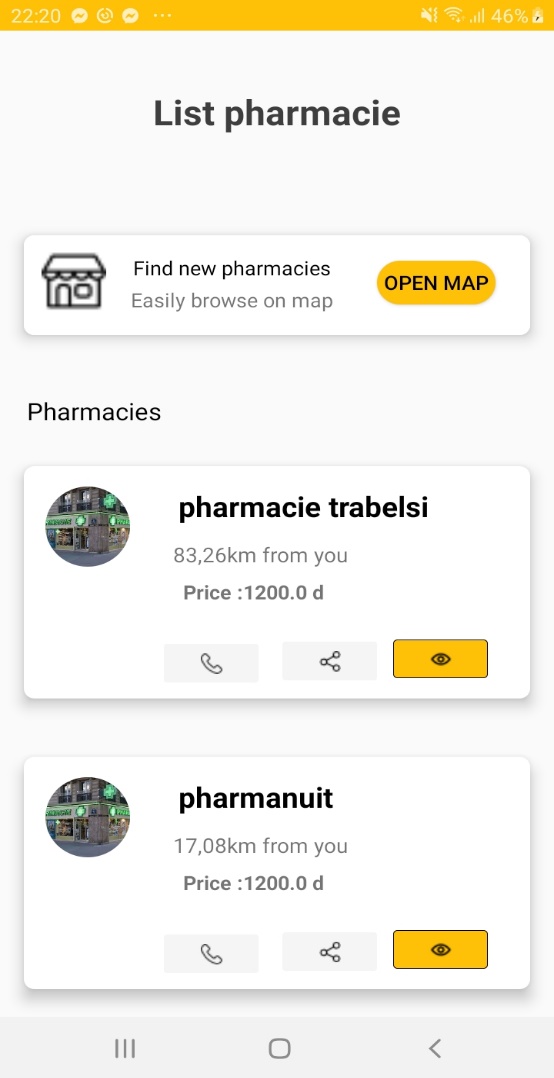


Figure 83: liste de pharmacie



Figure 84 :la direction de pharmacie

### **Conclusion**

Ce sprint présente la recherche de la pharmacie la plus proche avec son diagramme de cas d’utilisation et diagramme de séquence ainsi que sa description. Les notions que nous avons utilisée de L’api google map.

1. Sprint 4 : Messagerie et Google AdMob

### **Analyse de sprint Messagerie et Google AdMob**

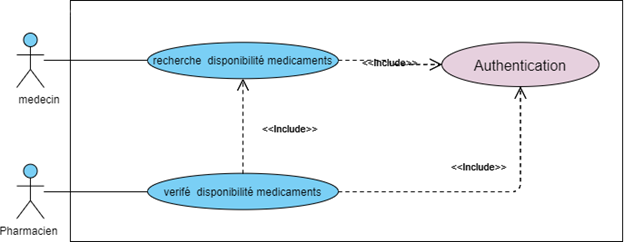


Figure 85 : Diagramme de cas d’utilisation sprint4

* Description textuelle

|  |  |
| --- | --- |
| Cas d’utilisation | Discussion médecin, pharmacie |
| Acteur | Médecin, Pharmacie |
| Précondition | La discussion sera affichée sur l’écran |
| Post-condition | La discussion sera affichée sur l’écran |
| Scénario nominal | 1-Le système affiche la discussion  3- le médecin envoie un message  4-Le système enregistre la discussion et envoie le message au pharmacie |
| Scénario alternative | 3-1 le message est vide  3-1-2-revenir à l’étape 1 du scénario nominal |

Tableau 13 : Description Textuelle « discussion médecin, pharmacie »

* Diagramme de séquence

Graphical user interface

Description automatically generated

Figure 86 : Discuter médecin

* L'application offre au médecin la possibilité d'entrer en discussion avec la pharmacie la plus proche après l'authentification et la recherche des produits pharmaceutiques.

### **Notion appliquer**

Implementation google admob

* Creation compte google AdMob

La première étape consiste à s’inscrire sur la plate-forme Google AdMob à l’aide d’un compte Google

Figure 87 :compte google admob

* Configurons notre application

Nous allons débuter par quelques éléments de configuration générale qui nous serviront aussi bien pour l’affichage des annonces.

* Du côté de Gradle

Nous allons donc ajouter dans le fichier build gradle de notre application, une dépendance au module Ads des Google Play Services grâce à la ligne suivante :



Figure 88 Google AdMob dependance

* Le Manifest

Maintenant que le module Ads des Google Play Services est utilisable dans notre application, nous allons attaquer la mise à jour du manifest de notre application à travers la modification du fichier AndroidManifest.xml et plus précisément à travers l’ajout de méta-data AdMob qui représente ID de l’application (identifié dans leAdMobUI )

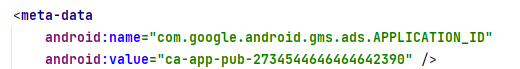


Figure 89: Implementation key adMob

* Du côté Activity

1-Initialiser le SDK Google Mobile Ads

Avant de charger les annonces, demandez à votre application d’initialiser le SDK Google Mobile Ads en appelant MobileAds.initialize(), qui initialise le SDK et rappelle un écouteur de fin de processus après l’initialisation.

1. Sélectionnez un format d’annonce
2. Ajouter AdView à la presentation xml

Pour afficher les annonces, la première étape consiste à placer un container AdView dans la mise en page de *Activity* ou Fragment dans lequel vous souhaitez l’afficher et faire la liaison avec une activité.

### **Revenu de sprint Messagerie et Google AdMob**

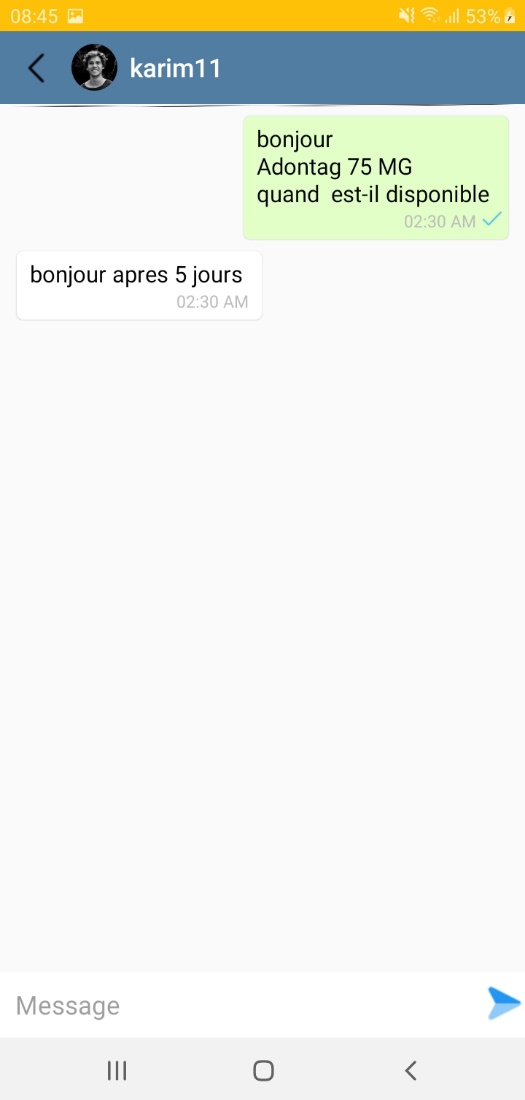


Figure 91 : interface de discussion pharmacie

Figure 90:l''interface de discussion médecin

### **Conclusion**

A travers ce sprint, nous avons inspecté le cas d’utilisation du coté médecin et décrit les interactions avec le pharmacien. En second lieu, nous avons illustré son diagramme de cas d’utilisation et diagramme de séquence ainsi que sa description. Les notions que nous avons utilisées avec google AdMob.

1. Sprint 5 : Administration

### **6.1 Analyse de sprint Administration**

* Diagramme de cas d’utilisation

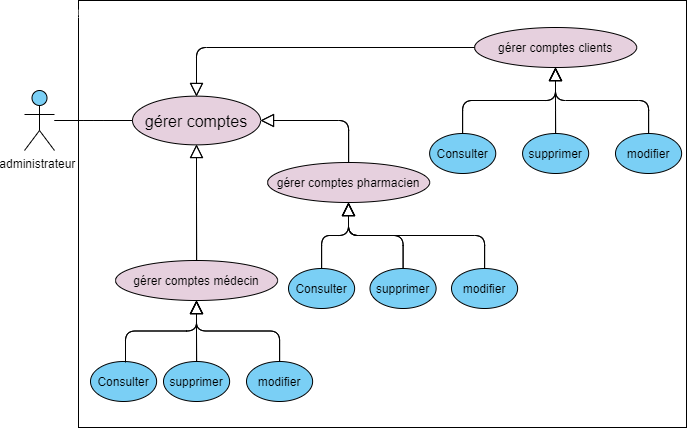


Figure 92 : Diagramme de class sprint 5

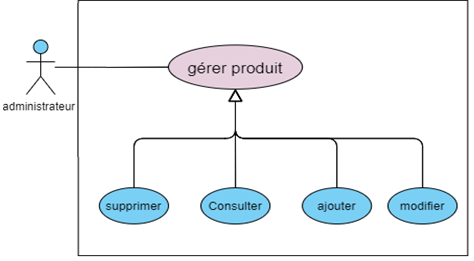


Figure 93 : Diagramme de classe gérer produit sprint4

* Description textuelle

|  |  |
| --- | --- |
| Cas d’utilisation | Modifier profil utilisateur |
| Acteur | Administrateur |
| Précondition | Authentification |
| Post-condition | Affiche L’interface de modification profil |
| Scénario nominal | 1-Le système affiche l’interface de modification avec un formulaire rempli  2-L’admin modifier les informations de l’utilisateur  3-Le système enregistre la modification  4-affiche la liste des utilisateurs |
| Scénario alternative | 2- si formulaire contient un champ obligatoire vide  2.1-system affiche un message d’erreurs |

Tableau 14 : Description Textuelle « modifier profile »

|  |  |
| --- | --- |
| Cas d’utilisation | Ajouter produit |
| Acteur | Administrateur |
| Précondition | Authentification |
| Post-condition | Affiche L’interface d’ajout de produit |
| Scénario nominal | 1-Le système affiche le formulaire d’ajouter un produit  2-l’admin remplit les informations du produit  3-Le système enregistre le produit  4-affiche la liste des produits |
| Scénario alternative | 2- si formulaire contient un champ obligatoire vide  2.1-system affiche un message d’erreurs |

Tableau 15 : Description Textuelle « ajouter Produit »

* Diagramme de séquence

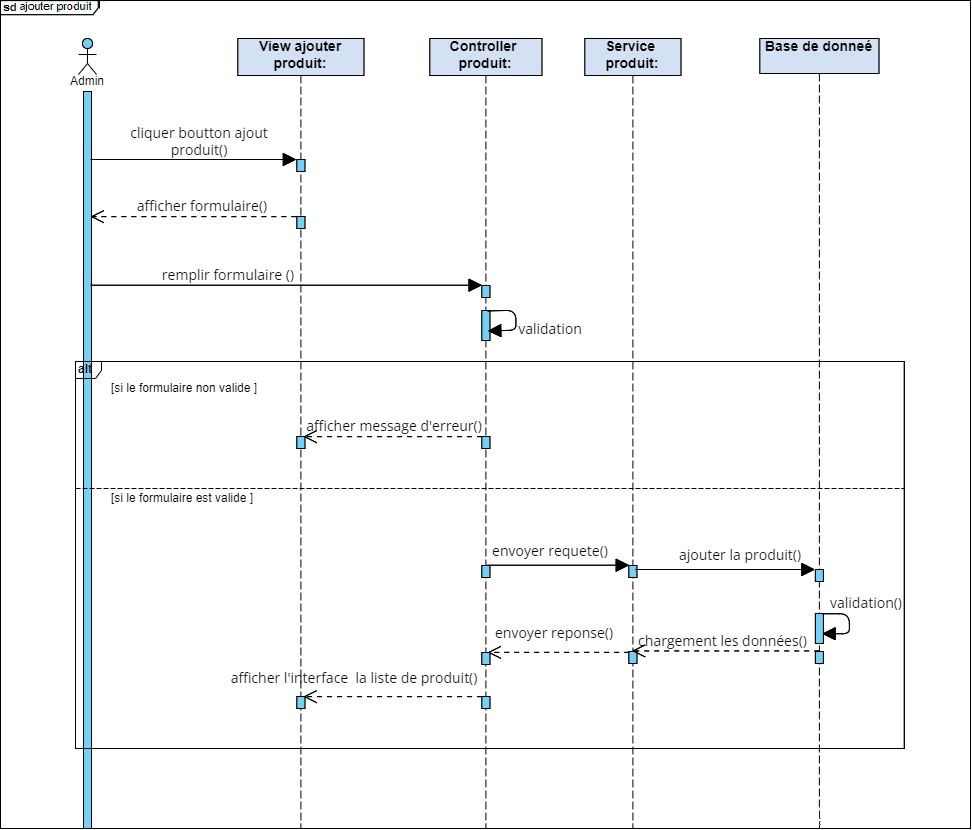


Figure 94 : Diagramme de séquence ajouter produit

➔ L’administrateur peut ajouter des produits pharmaceutiques à l’application et le système valide les informations de produit, si le formulaire est valide, il les ajoute à l’application.

### **6.2 Notions appliquées**

L’alimentation la base de données de notre application est une étape primordiale pour répondre au besoin de notre client pour cela l'administrateur alimente cette base via un document Excel qui est disposé par la pharmacie centrale. Chaque ligne de ce document correspond à un objet produit dans notre base de données.

* Alimentation :

La lecture fichier Excel avec spring boot La bibliothèque Apache POI prend en charge les fichiers .xls et .xlsx, il fournit par l’interface Workbook pour modéliser un fichier Excel, et les interfaces Sheet, Row et Cell qui modélisent les éléments d’un fichier Excel.

Créons une méthode qui ouvre un fichier .xlsx, puis lit le contenu de la première feuille du fichier.

La méthode de lecture du contenu de la cellule varie en fonction du type de données dans la cellule. Le type du contenu de la cellule peut être déterminé à l’aide de la méthode getCellType() de l’ interface Cell.

Étape 1 : pour commencer, nous devrons ajouter les dépendances suivantes à notre fichier pom.xml : Apache POI.

Étape 2 : Création des modèles.

Étape 3 : Création du dépôt du model (Repository), une interface fournit un ensemble des méthodes pour la manipulation du tableau correspondant aux models dans la base de données.

Étape 4 : Créer la classe ModelServiceconvert.

Dans l’étape suivante, implémentons une classe ModelServiceconvert chargée d’ouvrir des fichiers .xlsx et .xls, parcourir ce dernier et envoyer une liste de type model créé.

Étape 5 : Créer la classe ModelServicecrud.

Dans l’étape suivante, implémentons une ModelServicecrud chargée de manipuler notre base de données selon la liste de données générée par la classe ModelServiceconvert.

Étape 6 : Implémentons le Modèle Controller Comme dernière étape, nous allons implémenter une classe nommée Modèle Controller. Il sera responsable du traitement des requêtes arrivant sur nos apis.

### **6.3 Revenu de sprint Administration**

* Dashboard mobile

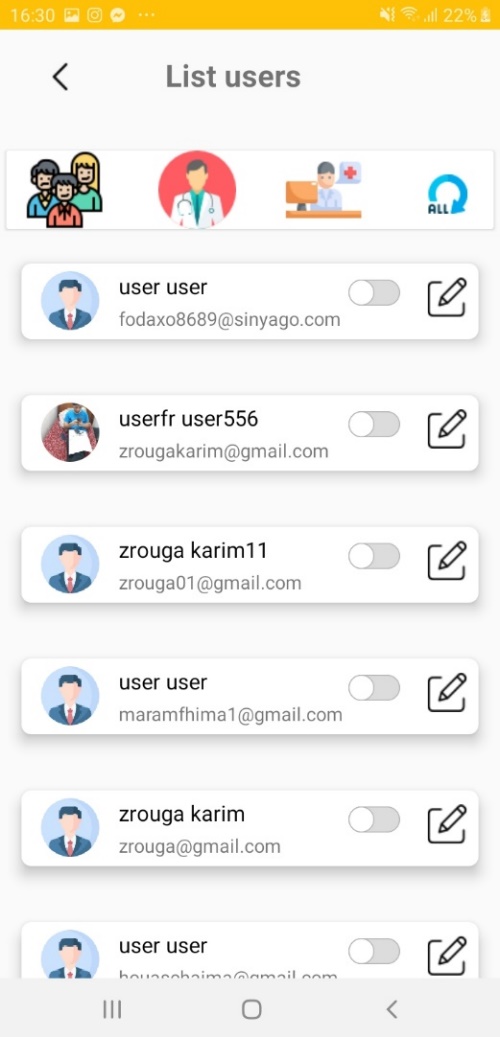
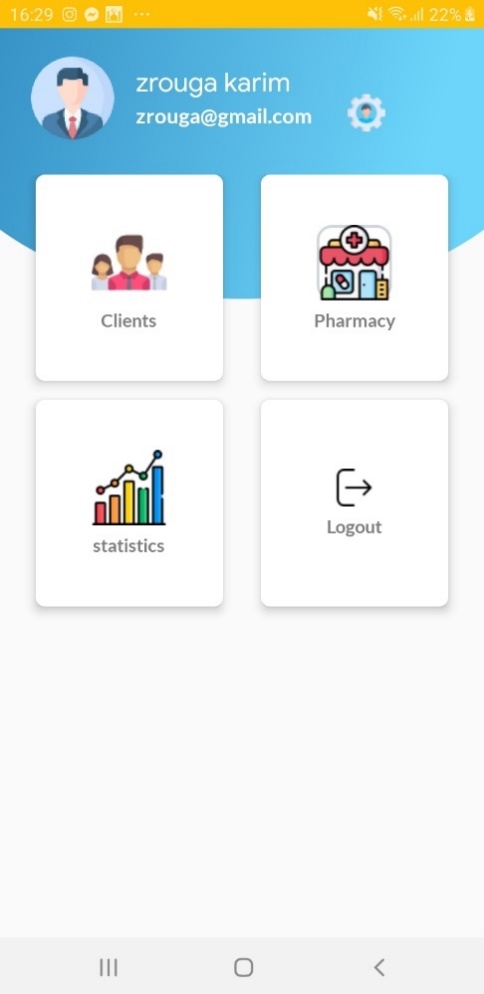


Figure 95:List utilisateurs

Figure 96:Dashboard Administrateur

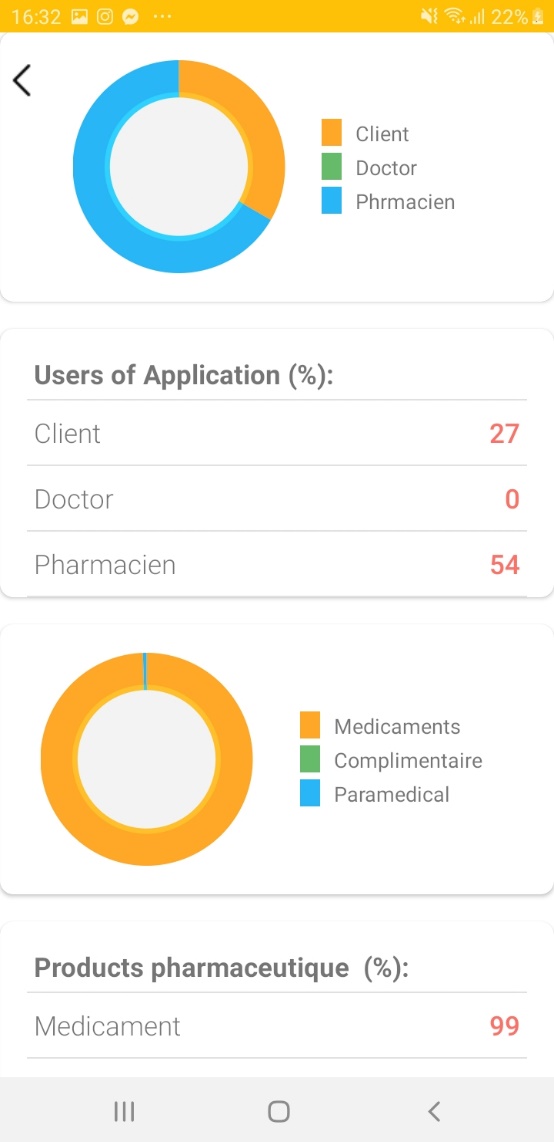
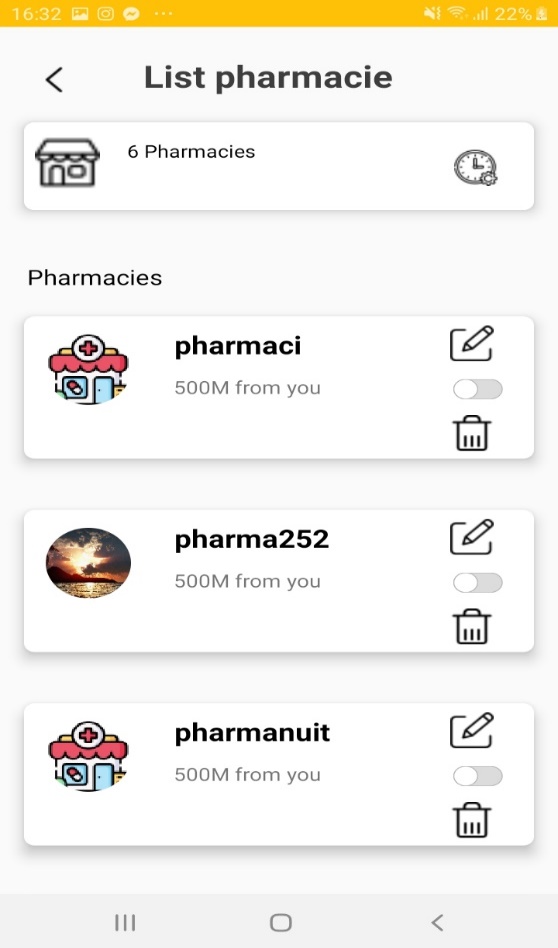
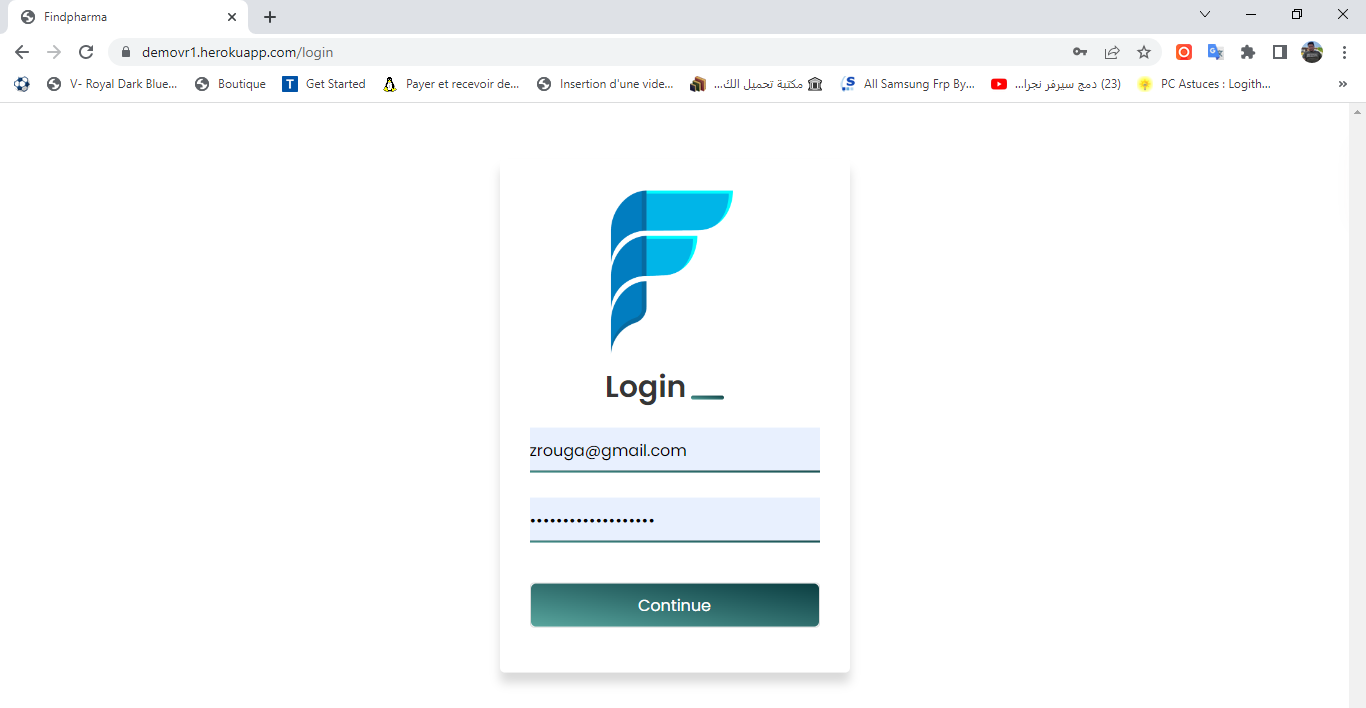


Figure 97: Liste de pharmacie

Figure 98: Interface statistique

* Dashboard Web

L’interface ci-dessous est mise à la disposition de l’administrateur pour qu’il puisse s’authentifier pour gérer les données systèmes. Il doit juste saisir son e-mail et son mot de passe.



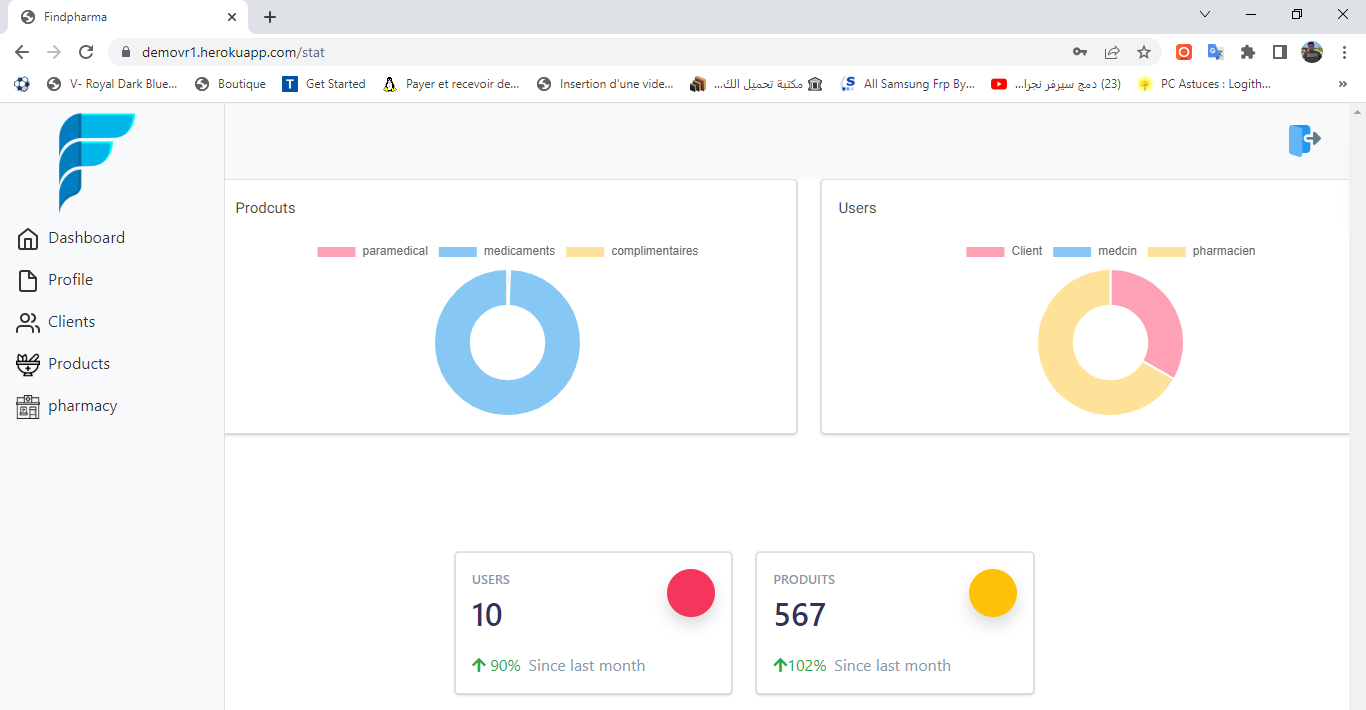
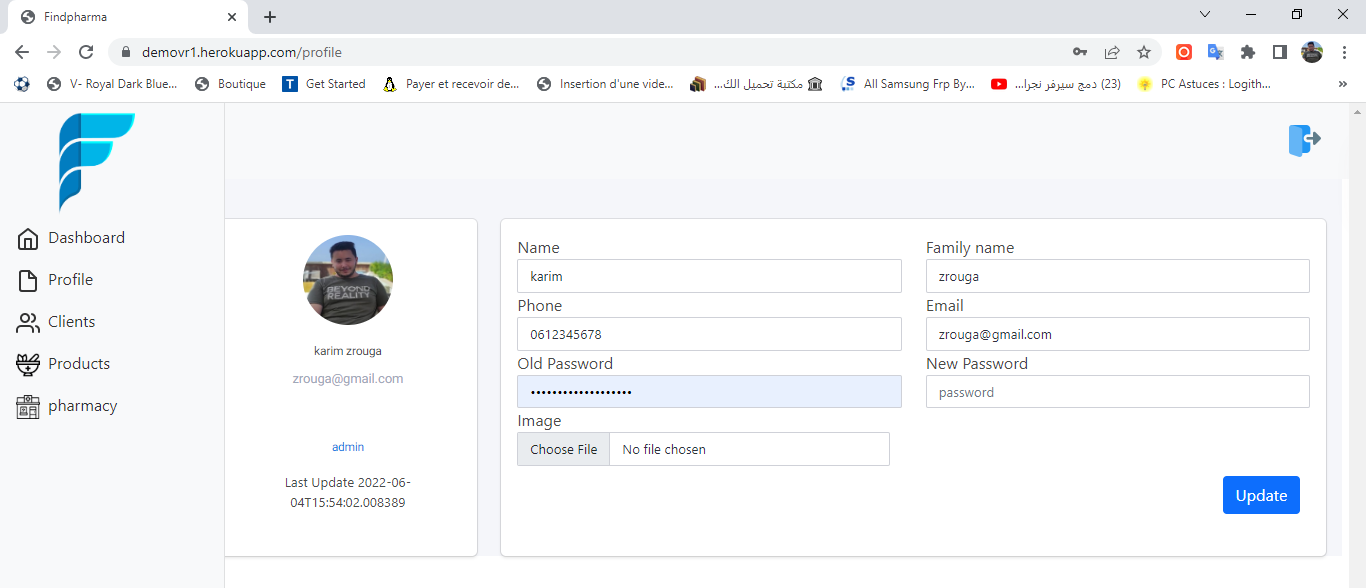
Figure 99:Interface d'authentification administrateur

Figure 100:Interface de statistiques

L’interface ci-dessous est le profil de l’administrateur pour qu’il puisse gérer ses informations.

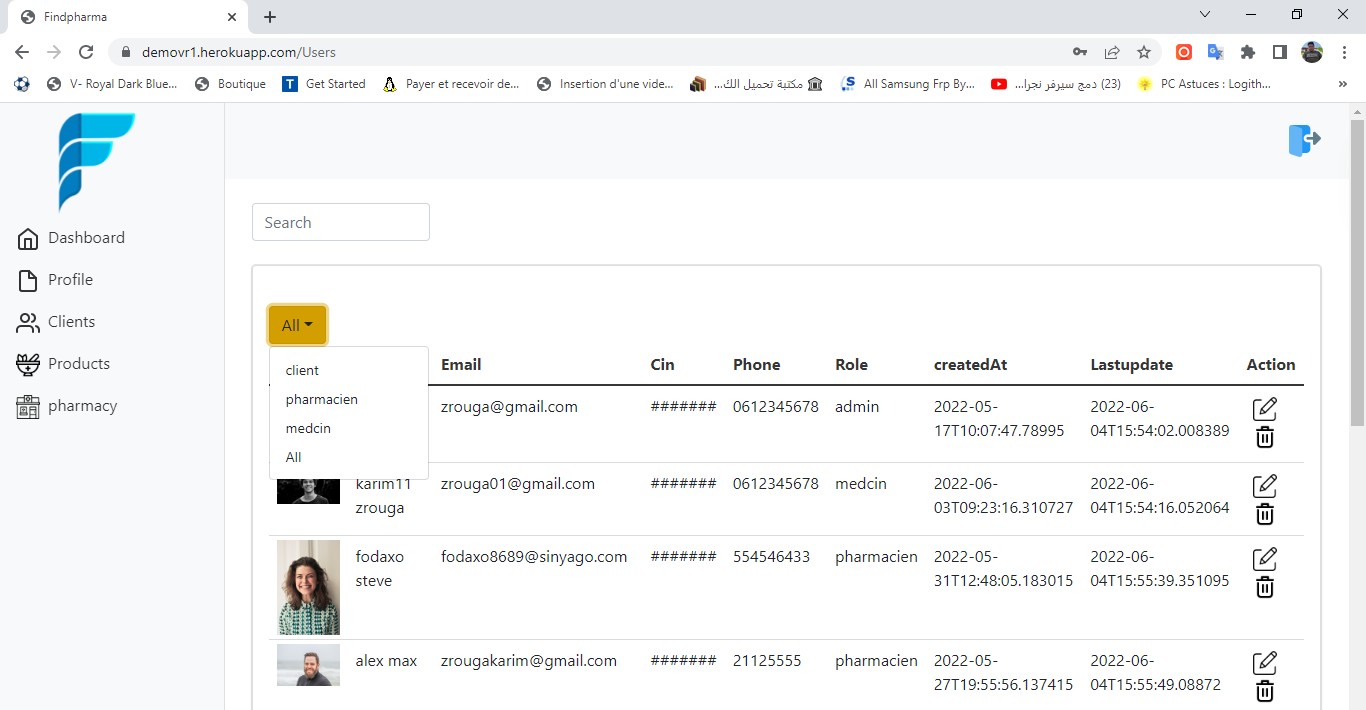
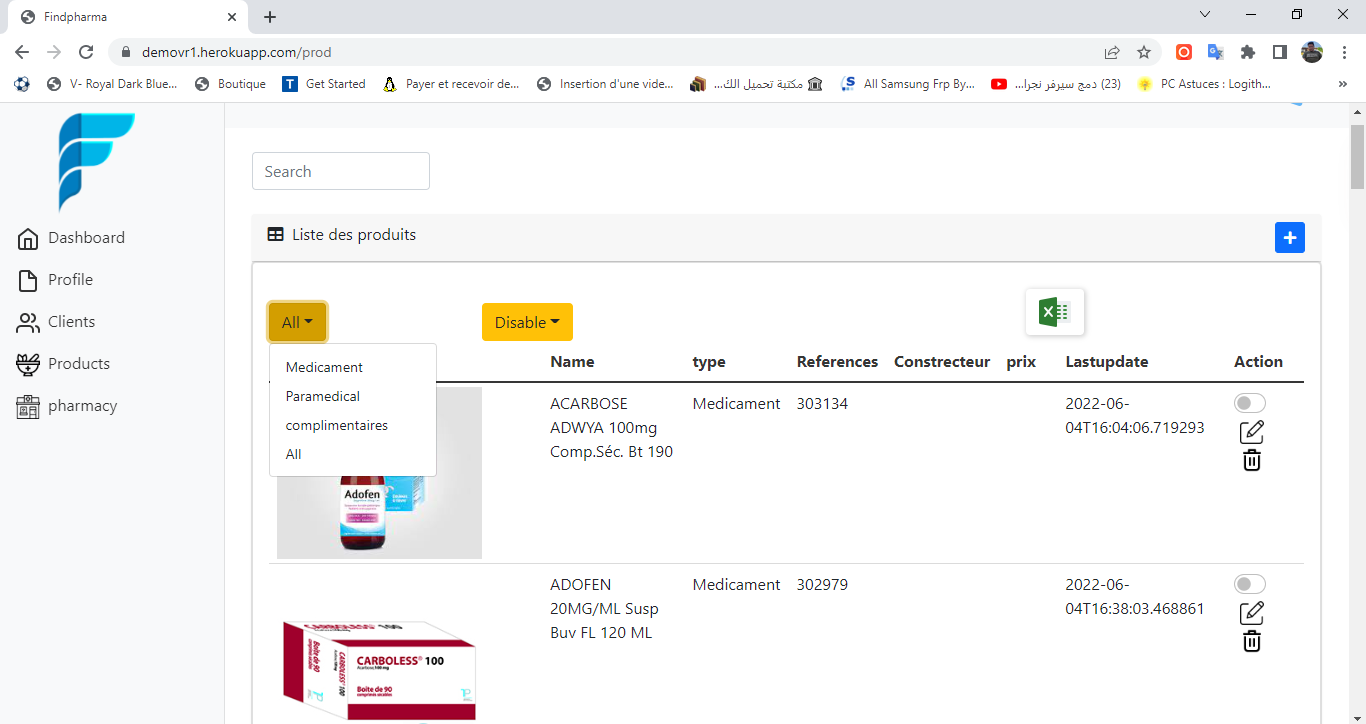
Figure 101:Interface de profile administrateur

Figure 102:Interface de la gestion des utilisateurs



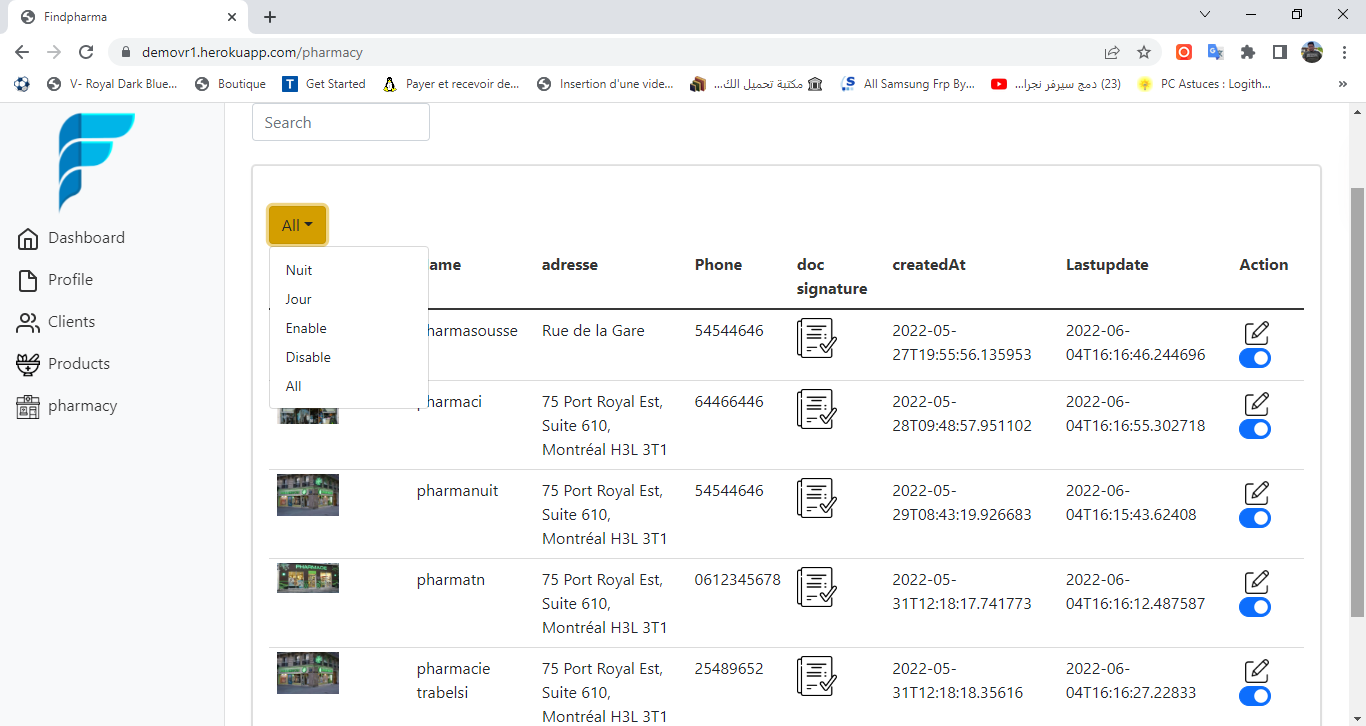
Figure 103:Interface de la gestion des produits

Figure 104:Interface de la gestion des pharmacies

### **6.4 Conclusion**

Finalement, ce sprint présente les fonctionnalités de l’administrateur détaillées par le diagramme de cas d’utilisation et le diagramme de séquence.

Conclusion Générale

Ce rapport nous a permis de présenter notre travail réalisé dans le cadre du projet de fin d’études, effectué au sein de la société « Medical Innovation Technology ». Ce projet a consisté la conception et la réalisation d’une application pharmacie. Il a été implémenté avec deux Framework Spring boot pour le backend et Android, angular pour le Frontend communiquant entre eux en utilisant des apis individuellement créées dans le backend et des apis offertes par google Platform cloud. Donc, il a nécessité de suivre une formation sur ces notions puisque c’est la première fois que nous travaillons avec.

Pour la réalisation de ce projet, une étude préalable de quelques logiciels proposant le même service était très utile pour dégager les fonctionnalités à mettre en œuvre et définir les spécifications de notre application.

La méthode UML, utilisée dans la phase de conception, nous a permis de bien modéliser notre système de manière à ce que sa compréhension devienne facile et que le développement soit plus rapide.

Lors de l’étape de l’implémentation, nous avons également eu recours, à divers outils et technologies tels que tester backend sur machine virtuel avant le déploiement ce qui nous a permis de gagner du temps.

Nous pouvons améliorer plus ce projet dans le futur en ajoutant d’autres fonctionnalités, comme, par exemple, system de notification pour la discussion ou pour les nouvelles actualités des produits pharmaceutique et développer un module d’historique de recherche de médicament.

Finalement, ce projet de fin d’étude a été une bonne expérience pour nous, dans la mesure où il nous a permis de mettre en pratique les connaissances théoriques et les compétences acquises au fil de trois ans d’études à l’Institut Supérieurs des Etudes Technologiques de Sousse, et les confronter avec les exigences demandées par la société. En plus, il nous a été une occasion pour une ouverture sur le monde professionnel et pour l’approfondissement de nos connaissances avec des nouvelles technologies.

Webographie

Les références suivantes nous ont aidé a réalisé des parties dans notre projet :

|  |  |
| --- | --- |
| [1] | «poi.apache,» [En ligne]. Available: https://poi.apache.org/ . [Accès le 12 06 2022]. |
| [2] | «Installation de lombok sous eclipse,» [En ligne]. Available: https://www.baeldung.com/lombok-ide. [Accès le 02 03 2022]. |
| [3] | «Installation de Postman,» [En ligne]. Available: https://www.baeldung.com/postman-testing-collections. [Accès le 03 03 2022]. |
| [4] | «installer spring tool sous eclipse,» [En ligne]. Available: https://devstory.net/10249/installer-spring-tool-suite-pour-eclipse. [Accès le 03 03 2022]. |
| [5] | «Intégrer Google authentication dan notre application Android,» [En ligne]. Available: https://developers.google.com/identity/sign-in/android/start-integrating. [Accès le 22 03 2022]. |
| [6] | «Shared Preference in Android Implementation,» [En ligne]. Available: https://data-flair.training/blogs/shared-preference-in-android/. [Accès le 15 03 2022]. |
| [7] | «different des application mobile,» [En ligne]. Available: https://blog.trigent.com/different-types-of-mobile-applications-native-hybrid-and-web-apps/. [Accès le 02 04 2022]. |
| [8] | «bibliothèque Picasso,» [En ligne]. Available: https://guides.codepath.com/android/Displaying-Images-with-the-Picasso-Library. [Accès le 01 04 2022]. |
| [9] | «MySQL,» [En ligne]. Available: wikipédia MySQL https://fr.wikipedia.org/wiki/MySQL . [Accès le 4 4 2022]. |
| [10] | «gestion projet scrum,» [En ligne]. Available: https://agiliste.fr/gestion-de-projet-simple/. [Accès le 10 03 2022]. |

d

Intitulé du stage :

Conception et Réalisation de la Plateforme Pharmacie

Réalisé par les étudiants :

**Mouhamed Karim Zrouga && Soulaima Ben Chedly**

Organisme d’accueil :

**Médical Innovation Technology**

Mots-Clés:

**Android, Angular, Spring boot, MySQL, java, Restful, paradigm, Scrum**

Résumé :

Notre projet consiste à la conception et la réalisation d'une plateforme Pharmacie, permettant de faciliter la recherche et la disponibilité des produits pharmaceutiques dans la pharmacie la plus proche ainsi qu'autoriser le médecin à connaître l'existence ou la pénurie des médicaments dans les pharmacies et diriger directement le patient vers eux.

Tampon &Sign. de l’Encadrant Ind. :

Signature:

Nom de l’Enseignant Encadrant :

Jalel boughizane

Nom de l’Encadrante Industriel :

Amel HARHIRA