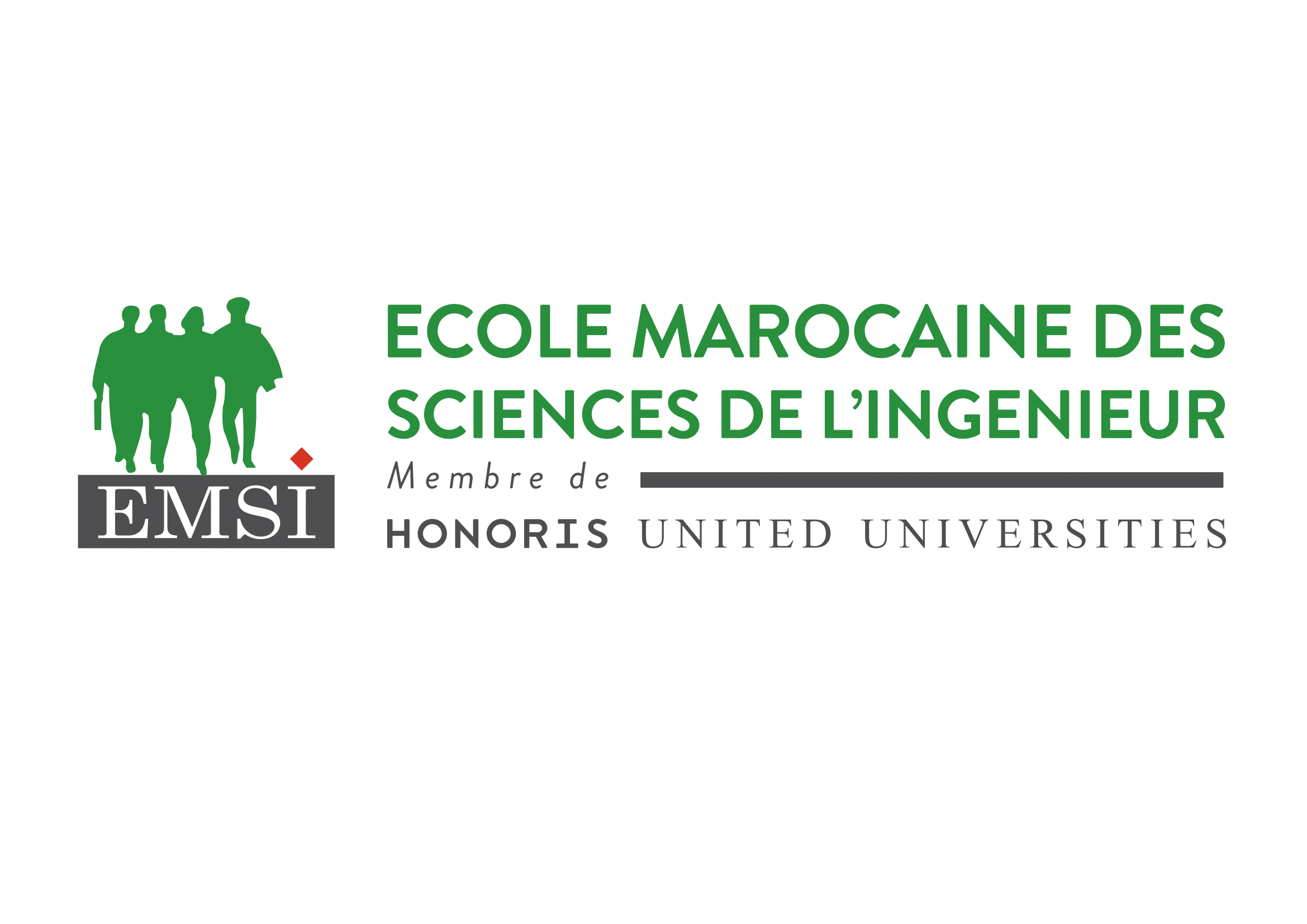
****bar5

**Réalisé par :**

**ABID Hamza**

**ALALOU Yassine**

**SEKKAT Ali**

**Encadré par :**

**Mr BOURKOUKOU Othmane**

**Rapport PFA**

**Année: 2019/2020**

**Conception et réalisation d'une application web de gestion de cabinet médical**

Ingénierie Informatique et Réseaux

**Dédicace**

*Nous dédions ce modeste travail et notre profonde gratitude à :*

*Nos parents pour l’éducation qu’ils nous prodiguent, avec tous les moyens et au prix des sacrifices qu’ils consentent à notre égard, pour le sens de devoir qu’ils nous ont enseigné depuis notre enfance.*

*Nos chers amis pour leur soutien et encouragement.*

*Et à toutes nos familles et à tous ceux que nous aimons.*

*Nous espérons que ce rapport va satisfaire toute personne qui aura l’occasion de le lire.*

**Remerciements**

*Avant d’aborder ce rapport, nous tenons à exprimer notre reconnaissance à tous ceux qui ont contribué de près ou de loin à la rédaction de ce rapport.*

*Nous tenons d’abord à remercier* ***Mr BOURKOUKOU*** *pour avoir bien voulu encadrer ce travail ainsi que pour sa riche contribution et ses précieux conseils.*

*Nous tenons aussi à remercier l’ensemble du corps enseignant de l’EMSI Marrakech.*

**Abstract**

As part of the fifth-year's final project, we implemented a project aiming in the design and development of a Web application for advanced management of a medical office.

The main goal of our project is to develop a Web application that offers the ability to make appointments and manage your practice in a simple way, as well as keeping the traceability of the tasks performed.

**Keywords** : J2EE, TypeScript, Spring Boot, Angular, Hibernate, Consultation, Rendez-vous, Cabinet medical, EMSI.

**Résumé**

Dans le cadre du mini projet de la 5me année, notre mission était de réaliser une application web de gestion de cabinet médical.

Cette gestion se fait à travers divers modules selon les fonctionnalités de chaque partie concernée par l’application.

Pour mener à bien ce travail, nous avons opté pour la méthode en cascade vu les avantages qu’elle offre et également le langage de modélisation UML, pour analyser et concevoir les différentes fonctionnalités de l’application. En ce qui concerne la partie technique l’application a été développée en technologies JAVAEE, à savoir le Framework Sping boot du cote back-end et Angular du cote front-end, et manipuler une base de données relationnelle MySQL.

Le présent rapport constitue une synthèse du travail réalisé durant la période de notre mini-projet. Il représente les différentes étapes de l’élaboration du projet.

**Mots clés**: J2EE, TypeScript, Spring Boot, Angular, Hibernate, Consultation, Rendez-vous, EMSI.

**Glossaire**

**HTML :** Hyper Text Markup Language

**IDE :** [Integrated Development Environment](https://fr.wikipedia.org/wiki/Environnement_de_d%C3%A9veloppement_int%C3%A9gr%C3%A9).

**JEE :** Java Enterprise Edition**.**

**JS**: Java Script.

**PDF** : Portable Document Format.

**RDV** : Rendez-vous.

**TS**: Type Script.

**UML:** Langage de Modélisation Unifié.

**Table des matières**

[**Introduction Générale** 1](#_Toc28874812)

[**Chapitre 1** : **Contexte général** 2](#_Toc28874813)

[**1.1 Introduction** 3](#_Toc28874814)

[**1.2 Présentation du projet** 3](#_Toc28874815)

[1.2.1 La problématique 3](#_Toc28874816)

[1.2.2 L'objectif du projet 3](#_Toc28874817)

[**1.3 Cahier des charges** 3](#_Toc28874818)

[1.3.1 Besoins fonctionnels : 3](#_Toc28874819)

[1.3.2 Besoins non fonctionnels : 4](#_Toc28874820)

[**1.4 Planification du projet** 4](#_Toc28874821)

[**1.6 Conclusion** 7](#_Toc28874822)

[**Chapitre 2:Analyse et Conception** 8](#_Toc28874823)

[**2.1 Introduction** 9](#_Toc28874824)

[**2.2 Diagramme de cas d’utilisation** 9](#_Toc28874825)

[**2.3 Description textuelle des cas d’utilisation** 11](#_Toc28874826)

[**2.4 Diagramme de séquence** 12](#_Toc28874827)

[**2.5 Diagramme de classes** 14](#_Toc28874828)

[**2.6 Conclusion** 15](#_Toc28874829)

[**Chapitre 3: Réalisation** 16](#_Toc28874830)

[**3.1 Introduction** 17](#_Toc28874831)

[**3.2 Le design pattern MVC** 17](#_Toc28874832)

[**3.3 Technologies utilisés** 18](#_Toc28874833)

[**3.4 Outils utilisés** 19](#_Toc28874834)

[**3.5 Captures d’écran du projet** 20](#_Toc28874835)

[**3.6 Conclusion** 24](#_Toc28874836)

[**Conclusion Générale et Perspectives** 25](#_Toc28874837)

[**Références** 26](#_Toc28874838)

**Liste des figures**

[Figure 1.1 tableau des besoins non fonctionnels 4](#_Toc29291587)

[Figure 1.2 Modèle de cycle de vie en cascade 6](#_Toc29291588)

[Figure 1.3 Délai du processus de réalisation 7](#_Toc29291589)

[Figure 1.4 Planning de réalisation 7](#_Toc29291590)

[Figure 2.1 Diagramme de cas d’utilisation admin 10](#_Toc29291591)

[Figure 2.2 Diagramme de cas d'utilisation Secrétaire 10](#_Toc29291592)

[Figure 2.3 Les deux types de diagramme de séquences 12](#_Toc29291593)

[Figure 2.4 Diagramme de séquence du cas « Authentification » 13](#_Toc29291594)

[Figure 2.5 Diagramme de séquence du cas « Consultation » 14](#_Toc29291595)

[Figure2.6 Diagramme de classes du projet 15](#_Toc29291596)

[Figure 3. 1 Interactions entre les couches du patron MVC 18](#_Toc29291597)

[Figure 3.2 Page Login 20](#_Toc29291598)

[Figure 3.3 Page de Register 21](#_Toc29291599)

[Figure 3.4 Ajout d'un rendez-vous 21](#_Toc29291600)

[Figure 3.5 Activation des comptes 22](#_Toc29291601)

[Figure 3.6 Impression de l'ordonnance 22](#_Toc29291602)

[Figure 3.7 gestion des comptes 23](#_Toc29291603)

[Figure 3.8 Consultation 23](#_Toc29291604)

# **Introduction Générale**

Actuellement, le monde connaît une avance technologique considérable dans tous les secteurs et les domaines scientifiques, économiques et cela grâce à l'informatique qui est une science qui étudie les techniques du traitement automatique de l'information. Avant l'invention de l'ordinateur, on enregistrait toutes les informations manuellement sur des supports en papier ce qui engendrait beaucoup de problèmes tel que la perte de temps considérable dans la recherche de ces informations ou la dégradation de ces dernières et plus la sécurisation des données. ...etc. C’est dans ce cadre que cette application se situe pour mettre en œuvre la gestion avancée d’un cabinet médical. Afin d'augmenter la fiabilité, l'efficacité de l'effort humain et faciliter les tâches pénibles au sein du cabinet.

Le présent rapport décrit les phases du déroulement du projet et est organisé en cinq chapitres:

Le premier chapitre présente le cadre et les objectifs du projet et aborde la démarche suivie pour la conduite du projet. Le deuxième chapitre comporte une étude préliminaire du projet en faisant une étude de l’existant, des captures des besoins et le cahier des charges. Le troisième chapitre propose l’analyse et la conception du projet tout en définissant la méthodologie et les phases de conception . Le quatrième chapitre va présenter les outils techniques utilisés dans au sein du projet c'est- à-dire le patron de conception, la justification du choix du modèle MVC, langages et technologies de développement. Le cinquième chapitre va présenter les captures de cette application avec un petit commentaire pour chacune.

*Chapitre 1 :*

# Chapitre 1 : **Contexte général**

## **1.1 Introduction**

Ce chapitre présente en premier lieu une présentation du projet c’est-à-dire présenter la problématique, l’objectif, l’étude fonctionnelle et non-fonctionnelle du projet. En suite le processus de développement, puis la planification des taches du projet, et finalement une conclusion.

## **1.2 Présentation du projet**

### 1.2.1 La problématique

Comme nous pouvons le constater, le protocole actuel pour gérer les cabinets médicaux présente certaines limites :

Obligation de se déplacer toujours même pour une simple prise, modification, suppression ou recherche d’un RDV.

Obligation de chercher dans plusieurs documents afin de trouver les fiches de RDV, consultations, médicaments et plannings voulues.

Obligation de se rende aux anciens documents pour voir les statistiques des RDV.

### 1.2.2 L'objectif du projet

Dans le souci d’apporter une valeur ajoutée et un meilleur service aux employés (médecins et secrétaires) et patients, créer une application accessible via tous les ordinateurs du cabinet et patient sera efficace pour surmonter les problèmes cités avant.

## **1.3 Cahier des charges**

### 1.3.1 Besoins fonctionnels :

Dans cette partie nous avons cité l’ensemble des fonctionnalités que doit offrir l’application aux utilisateurs tel que :

- **Authentification :** chaque utilisateur doit posséder un login et un mot de passe spécifique qui lui permet de vérifier son identité, afin d’autoriser l’accès de cette entité en toute sécurité.

**-La gestion des comptes :** Cette partie permet l’ajout et la suppression des différents utilisateurs (médecin (admin) / patient/ secrétaire).

- **La gestion du cabinet :** Elle s’agit de la gestion des fiches des patients, la gestion des consultations, ainsi que la gestion des rendez-vous.

**-La gestion du planning :** Elle concerne la vérification des rendez-vous ainsi que la possibilité au patient et secrétaire de saisir un rendez-vous via le net directement.

**-La gestion des consultations :** Elle concerne la vérification des rendez-vous ainsi que la possibilité d’ajouter un diagnostic et son prix par le médecin.

**-La gestion des patients :** Elle concerne l’ajout, la modification ainsi que la suppression des patients sans oublier leurs consultations.

**-La gestion des statistiques :** Avoir les statistiques qui concernent les rendez-vous.

### 1.3.2 Besoins non fonctionnels :

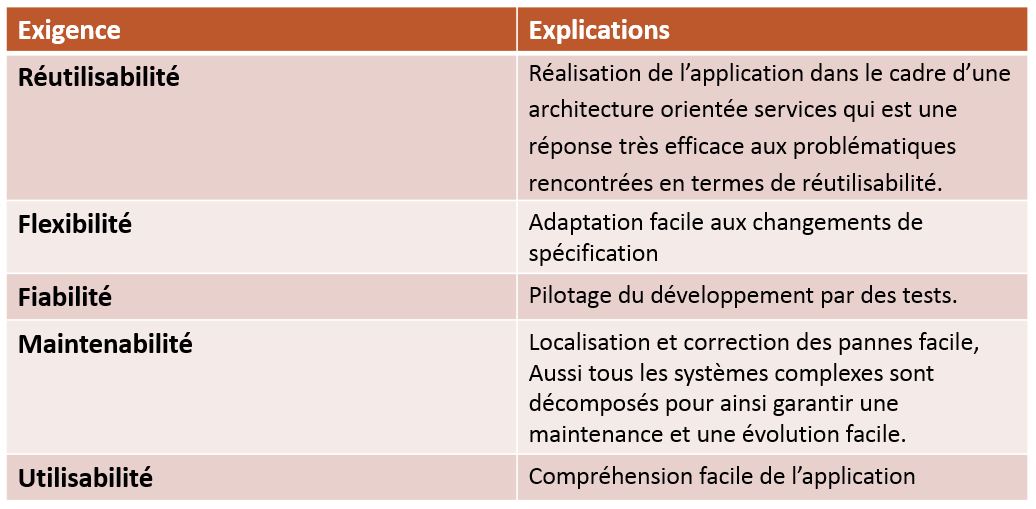
****

Figure 1.1 tableau des besoins non fonctionnels

## **1.4 Planification du projet**

***Cycle de vie du projet :***

Le cycle de vie du projet définit les phases qui relient le début d'un projet à sa fin.

Le nombre de phases ainsi que leur appellation peuvent varier d’une application à une autre. La définition du cycle de vie du projet peut aider le chef de projet à déterminer s'il doit traiter l'étude de faisabilité en tant que première phase du projet ou en tant que projet séparé et indépendant.

Notre projet se compose de 4 phases :

* Phase d’étude (cahier de charges)
* Phase de conception (chapitre 2)
* Phase de réalisation (chapitre 3)
* Phase de test (tout au long de l’implémentation)

Il existe différents types de modèles **UML**entrant dans la réalisation d'un logiciel. Ces modèles prendront en compte toutes les étapes de la conception du système.

Parmi les grandes familles des modèles :

* Modèle en cascade
* Cycle en V
* Cycle en spirale
* Cycle itératif

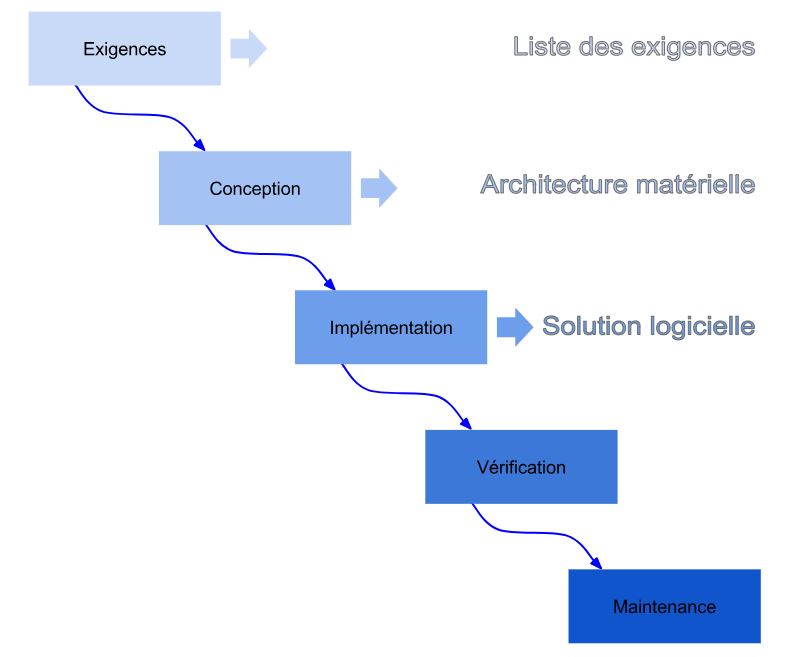
Le modèle de cycle de vie qu’on a utilisé est le modèle en cascade :

Figure 1.2 Modèle de cycle de vie en cascade

***Diagramme de Gantt :***

Le diagramme de Gantt répond à la question « Quelles sont les tâches qui doivent se réaliser à une date donnée ? ».

Ce digramme se présente sous la forme d’un graphique avec la liste des tâches en ordonnée et le temps en abscisse.

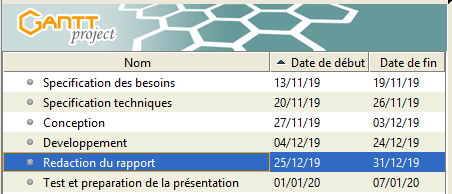


Figure 1.3 Délai du processus de réalisation

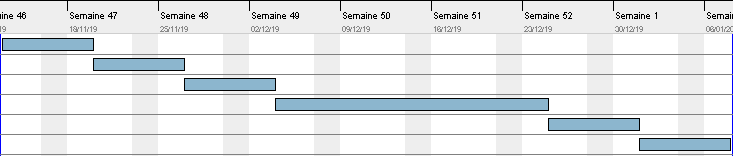


Figure 1.4 Planning de réalisation

## **1.6 Conclusion**

Le succès du projet se mesure, en effet, à la satisfaction du client et à la qualité du résultat, c’est-à-dire à la conformité du produit, à ce qui est attendu, livré dans le respect du délai imparti et du budget alloué.

Avant d’entamer une étude détaillée du projet, il nous a fallu réaliser une étude de l’existant pour déterminer la démarche préconisée par l'entreprise.

Nous devons nous doter alors d’une solution informatique à la fois simple, pratique et robuste.

Chapitre 2 :

# **Chapitre 2:Analyse et Conception**

## **2.1 Introduction**

Comme n’importe quel type de projet, un projet informatique nécessite une phase d’analyse, suivi d’une étape de conception.

**Phase d’analyse :** on cherche d’abord à bien comprendre et à décrire de façon précise les besoins des utilisateurs ou des clients. Que souhaitent-ils faire avec l’application ? Quelles fonctionnalités veulent-ils ? Pour quel usage ? Comment l’action devrait-elle fonctionner ?

**Phase de conception :** on apporte plus de détails à la solution et on cherche à clarifier des aspects techniques. Elle consiste à répondre à la question "comment faire le système».

La conception d'un S.I n'est pas évidente car il faut réfléchir à l'ensemble de l'organisation que l'on doit mettre en place.

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

Le **diagramme de cas d'utilisation** est un diagramme UML utilisé pour donner une vision globale du comportement fonctionnel d'un système logiciel.

Un cas d'utilisation représente une unité discrète d'interaction entre un utilisateur (humain ou machine) et un système. Il est une unité significative de travail. Dans un diagramme de cas d'utilisation, les utilisateurs sont appelés acteurs, ils interagissent avec les cas d'utilisation.

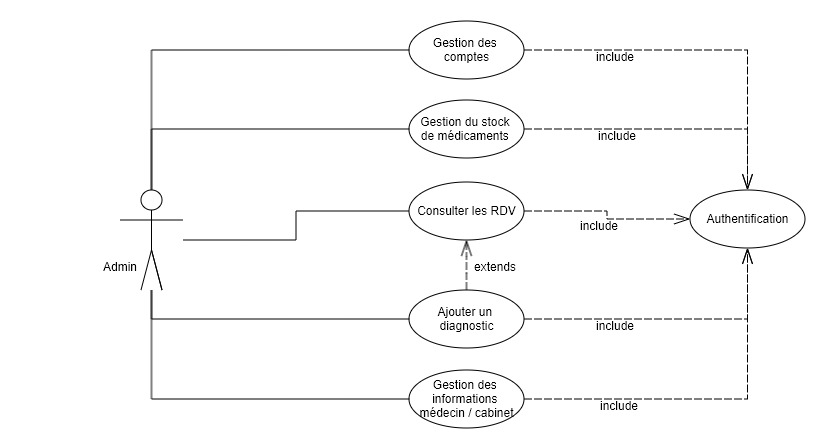


Figure 2.1 Diagramme de cas d’utilisation admin

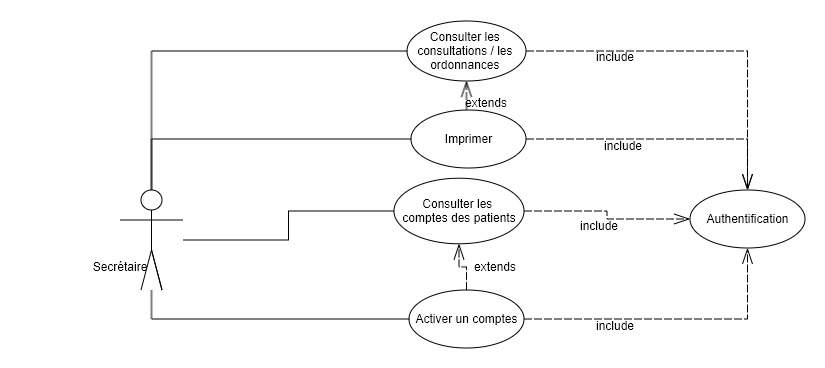


Figure 2.2 Diagramme de cas d'utilisation Secrétaire

## **2.3 Description textuelle des cas d’utilisation**

Le diagramme de cas d'utilisation décrit les grandes fonctions d'un système du point de vue des acteurs, mais n'expose pas de façon détaillée le dialogue entre les acteurs et les cas d'utilisation.

Un simple nom est tout à fait insuffisant pour décrire un cas d'utilisation.

Chaque cas d'utilisation doit être documenté pour qu'il n'y ait aucune ambigüité concernant son déroulement.

On décrit le cas d'utilisation par le scénario principal comme une séquence d'étapes, chaque étape dans le cas d'utilisation est un élément de l'interaction entre l'acteur et le système.

* Gestion des comptes :

|  |  |
| --- | --- |
| Titre | Recherche des candidats |
| Préconditions | L’utilisateur doit au préalable s’être authentifié. |
| Description | CRUD compte. |
| Scénario Nominal | * L’utilisateur sélectionne la page « Comptes». * Le système affiche les comptes. * L’utilisateur choisi d'ajouter, modifier, supprimer un compte. * Le système enregistre le résultat. |
| Scénario alternatif | * L’utilisateurdécide de quitter la consultation des comptes. |
| Scénario d’erreur | * L'utilisateur oublie un champ important. |
| Post Condition | Le système affiche la liste des comptes |

* Gestion des RDV :

|  |  |
| --- | --- |
| Titre | Recherche des candidats |
| Préconditions | L’utilisateur doit au préalable s’être authentifié. |
| Description | CRUD RDV |
| Scénario Nominal | * L’utilisateur sélectionne la page « RDV». * Le système affiche les rendez-vous. * L’utilisateur choisi d'ajouter, modifier, supprimer un RDV. * Le système enregistre le résultat. |
| Scénario alternatif | * L’utilisateurdécide de quitter la consultation des RDV. |
| Scénario d’erreur | * L'utilisateur oublie un champ important. |
| Post Condition | Le système affiche la liste des RDV. |

## **2.4 Diagramme de séquence**

**Les diagrammes de séquences** modélisent l’aspect dynamique du système. Ils permettent de décrire comment les éléments du système interagissent entre eux et avec les acteurs.

Les objets au cœur d'un système interagissent en s'échangent des messages.

Les acteurs interagissent avec le système au moyen d'IHM (Interfaces Homme-Machine).

Il existe deux types de diagrammes de séquence : diagramme de séquence analyse et diagramme de séquence conception

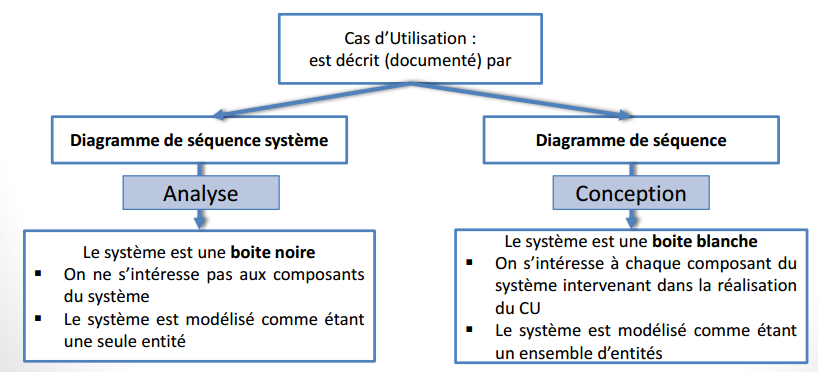


Figure 2.3 Les deux types de diagramme de séquences

Nous modéliserons dans ce projet des diagrammes de séquence analyse (nous avons pris comme exemple les cas d’utilisation « Authentification » et « Consulter les RDV »).

***Diagramme de séquence du cas « Authentification » :***

Quand l’utilisateur souhaite se connecter, le système lui affiche la page de login.

L’utilisateur saisi donc ses informations d’identification, qui seront vérifiés juste après par le système.

Tant que le nom d’utilisateur ou mot de passe saisi est erroné, le système donne la main à l’utilisateur pour entrer ses informations à nouveau.

Une fois les bonnes informations sont saisies, l’interface convenable est affichée.

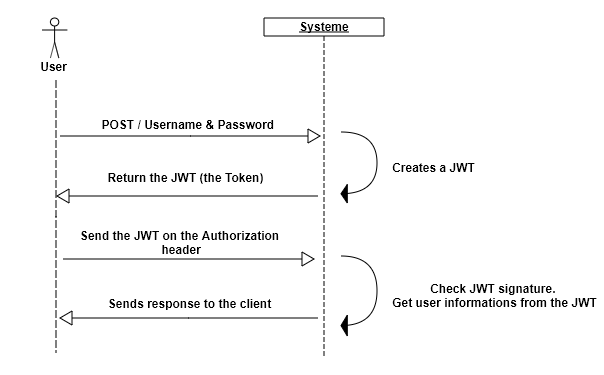
****

Figure 2.4 Diagramme de séquence du cas « Authentification »

***Diagramme de séquence de prise de rendez-vous :***

Ce diagramme de séquence décrit l’interaction de la secrétaire avec l’objet patient pour réaliser l’ajout d’un rendez-vous.

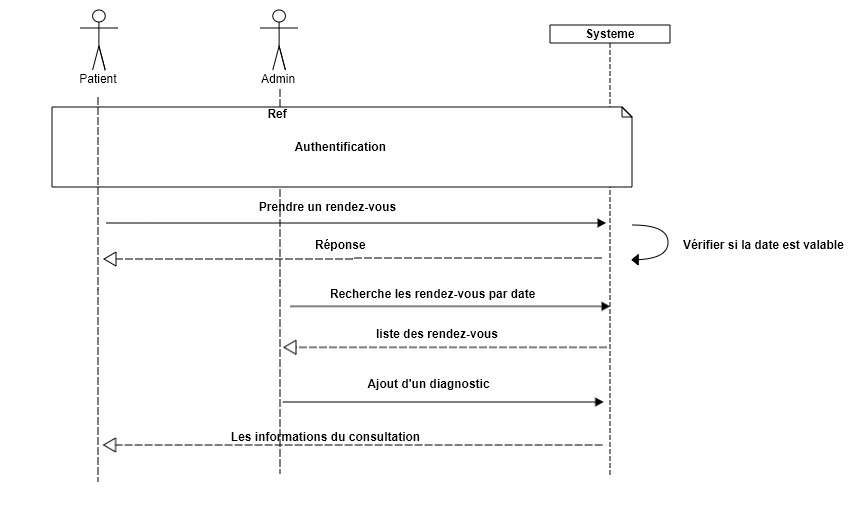


Figure 2.5 Diagramme de séquence du cas « Consulter les rendez-vous »

## **2.5 Diagramme de classes**

Le **diagramme de classes** est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que les différentes relations entre celles-ci. Ce diagramme fait partie de la partie statique d'UML(1) car il fait abstraction des aspects temporels et dynamiques.

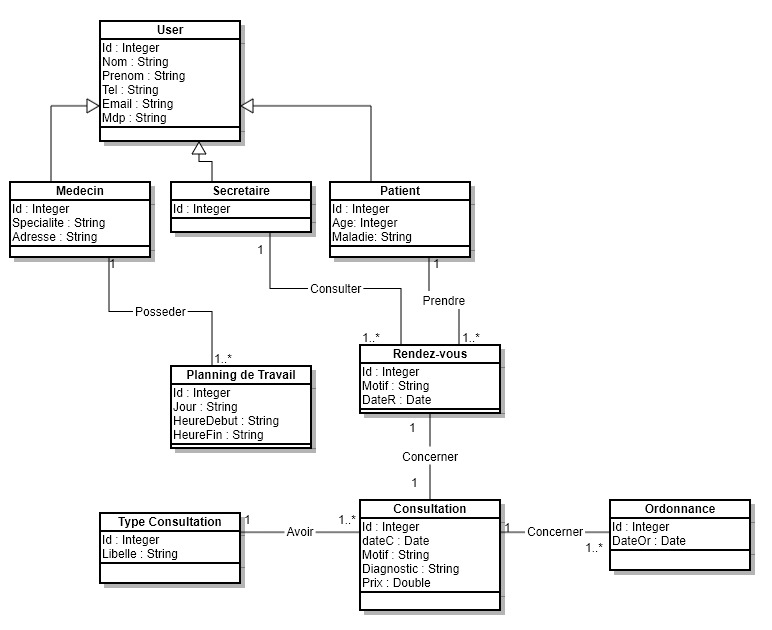
****

Figure2.6 Diagramme de classes du projet

## **2.6 Conclusion**

Nous avons discuté dans ce chapitre les 3 diagrammes **cas d’utilisation**, **séquence**, et **classe**, qui nous ont permis de répondre aux questions « à quoi sert le système », « quelles sont les structures et liens entre les objets composant le système » et « comment les éléments du système interagissent entre eux et avec les acteurs ».

Ceci conclu les phases d’analyse et conception, ce qui nous mène ensuite à la phase de réalisation du projet qui est présentée dans le chapitre suivant.

Chapitre 3 :

# **Chapitre 3: Réalisation**

## **3.1 Introduction**

Après avoir mené les phases de l'étude des besoins, l’analyse des spécifications et la conception du système, nous avons entamé la phase de la réalisation.

Cette phase porte sur la mise en œuvre de l’application. Ainsi nous retrouvons deux parties :

* La première passe en revue les différentes technologies et outils de développement utilisés pour la réalisation.
* La deuxième présente la mise en œuvre effective de l'application en expliquant son architecture technique et met l'accent sur les différents modules réalisés, accompagnée de quelques captures d’écran.

## **3.2 Le design pattern MVC**

Le patron MVC permet de séparer les couches logiques d'une application. Il constitue une façon d'organiser une application en distinguant trois entités ayant chacune un rôle précis dans l'interface:

* Modèle : Représente les données de l'application (accès et mise à jour).
* Vue : Représente interface utilisateur (entrées et sorties). N'effectue aucun traitement, elle se contente simplement d'afficher les données fournis par le modèle.
* Contrôleur : Responsable de la gestion des événements et la synchronisation via l’interprétation des requêtes du client pour lui envoyer la vue correspondante. La synchronisation entre la vue et le modèle se passe avec le pattern Observer. Ce dernier permet de générer des événements lors d'une modification du modèle et d'indiquer à la vue qu'il faut se mettre à jour. Le schéma suivant décrit les interactions entre les différentes couches.

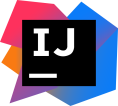
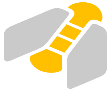


Figure 3. 1 Interactions entre les couches du patron MVC

## **3.3 Technologies utilisés**

*  : Le Framework jee***SPRING BOOT*** : Spring boot est un nouveau projet de l’éco-systèmeSpring.Comme son nom l’indique, Spring boot permet de démarrer un projet Spring en simplifiant tous les aspects configuration qui sont habituellement rébarbatifs. Il a été conçu pour simplifier le travail du développeur utilisant Spring sans pour autant remettre en cause les fondements du framework.Le frameworkSpring pouvait être considéré comme complexe à mettre en oeuvre, lourd (beaucoup de composants), avec des dépendances parfois mal maîtrisées, et, dans l’esprit, plutôt adapté à une utilisation sur des serveurs d’application.[1]
* ***Angular***: est un [framework](https://fr.wikipedia.org/wiki/Framework" \o "Framework) coté client [JavaScript](https://fr.wikipedia.org/wiki/Framework_JavaScript) [libre](https://fr.wikipedia.org/wiki/Logiciel_libre) et [open source](https://fr.wikipedia.org/wiki/Open_source)développé par Google, basé sur TypeScript qui utilise une hiérarchie de composants, ces derniers contient aussi du code HTML. Il permet de développer des [pagesweb](https://fr.wikipedia.org/wiki/Page_web" \o "Page web).[2]
*  ***TypeScript :*** est un [langage de programmation](https://fr.wikipedia.org/wiki/Langage_de_programmation) [libre](https://fr.wikipedia.org/wiki/Logiciel_libre) et [open source](https://fr.wikipedia.org/wiki/Open_source) développé par [Microsoft](https://fr.wikipedia.org/wiki/Microsoft) qui a pour but d'améliorer et de sécuriser la production de code [JavaScript](https://fr.wikipedia.org/wiki/JavaScript). C'est un sur-ensemble de JavaScript (c'est-à-dire que tout code JavaScript correct peut être utilisé avec TypeScript).[3]
* *** Hibernate*** : est un [framework](https://fr.wikipedia.org/wiki/Framework" \o "Framework) [open source](https://fr.wikipedia.org/wiki/Open_Source) gérant la [persistance](https://fr.wikipedia.org/wiki/Persistance_(informatique)) des [objets](https://fr.wikipedia.org/wiki/Programmation_orient%C3%A9e_objet) en [base de données relationnelle](https://fr.wikipedia.org/wiki/Base_de_donn%C3%A9es_relationnelle). Hibernate est adaptable en termes d'architecture, il peut donc être utilisé aussi bien dans un [développement](https://fr.wikipedia.org/wiki/D%C3%A9veloppeur) client lourd, que dans un environnement web léger de type [Apache Tomcat](https://fr.wikipedia.org/wiki/Apache_Tomcat) ou dans un environnement [Java EE](https://fr.wikipedia.org/wiki/Java_EE) complet.[4]
* ***Github*** :  Est un service web d'[hébergement](https://fr.wikipedia.org/wiki/H%C3%A9bergeur_web) et de gestion de développement de logiciels, utilisant le [logiciel de gestion de versions](https://fr.wikipedia.org/wiki/Logiciel_de_gestion_de_versions) [Git](https://fr.wikipedia.org/wiki/Git). Ce site est développé en [Ruby on Rails](https://fr.wikipedia.org/wiki/Ruby_on_Rails) et [Erlang](https://fr.wikipedia.org/wiki/Erlang_(langage)) par Chris Wanstrath, PJ Hyett et Tom Preston-Werner. GitHub propose des comptes professionnels payants, ainsi que des comptes gratuits pour les projets de [logiciels libres](https://fr.wikipedia.org/wiki/Logiciels_libres).[5]

## **3.4 Outils utilisés**

* ***IntelliJ*** IDEA également appelé « IntelliJ », « IDEA » ou « IDJ » est un environnement de développement intégré (en anglais [IntegratedDevelopmentEnvironment](https://fr.wikipedia.org/wiki/Environnement_de_d%C3%A9veloppement_int%C3%A9gr%C3%A9" \o "Environnement de développement intégré) - IDE) de technologie [Java](https://fr.wikipedia.org/wiki/Java_(technique)) destiné au développement de logiciels informatiques. Il est développé par [JetBrains](https://fr.wikipedia.org/wiki/JetBrains" \o "JetBrains).[6]
* ***PowerAMC :***représente un logiciel qui nous permet de modéliser les traitements informatiques et leurs bases de données associées et qui gère la plupart des diagrammes spécifiés dans la norme UML 2.0, elle est basée sur le langage de modélisation UML (UnifiedModelingLanguage) que nous avons adopté pour toute la suite de notre travail. [7]
* ***Xampp :***est un ensemble de [logiciels](https://fr.wikipedia.org/wiki/Logiciel) permettant de mettre en place facilement un [serveur Web](https://fr.wikipedia.org/wiki/Serveur_Web) local. [8]
* ***Gantt Project :*** est logiciel libre de gestion de projet écrit en java, ce qui permet de l’utiliser dans divers système d’exploitation, il permet d’éditer un diagramme de gant. [9]

## **3.5 Captures d’écran du projet**

Ci-dessous quelques captures d’écran prises depuis différents pages du projet :

Après l'authentification l'utilisateur est redirigé vers l’écran d’accueil :

L'ajout un message de succès s'affiche comme suit :

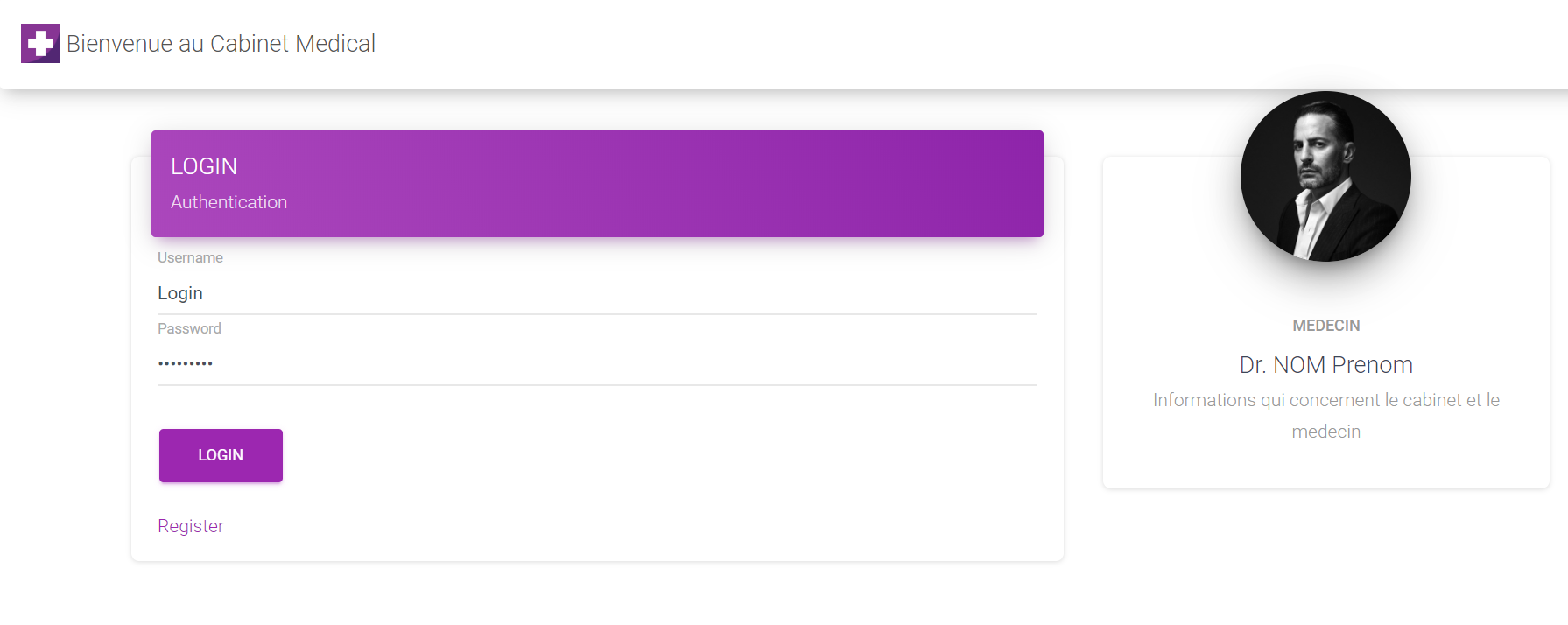


Figure 3.2 Page Login

La figure ci-dessus représente l’interface qui concerne l’authentification ou l’internaute s’authentifie. Si l’internaute n’a pas de compte, il peut alors créer un compte, c’est ce qui représente la figure ci-dessous.

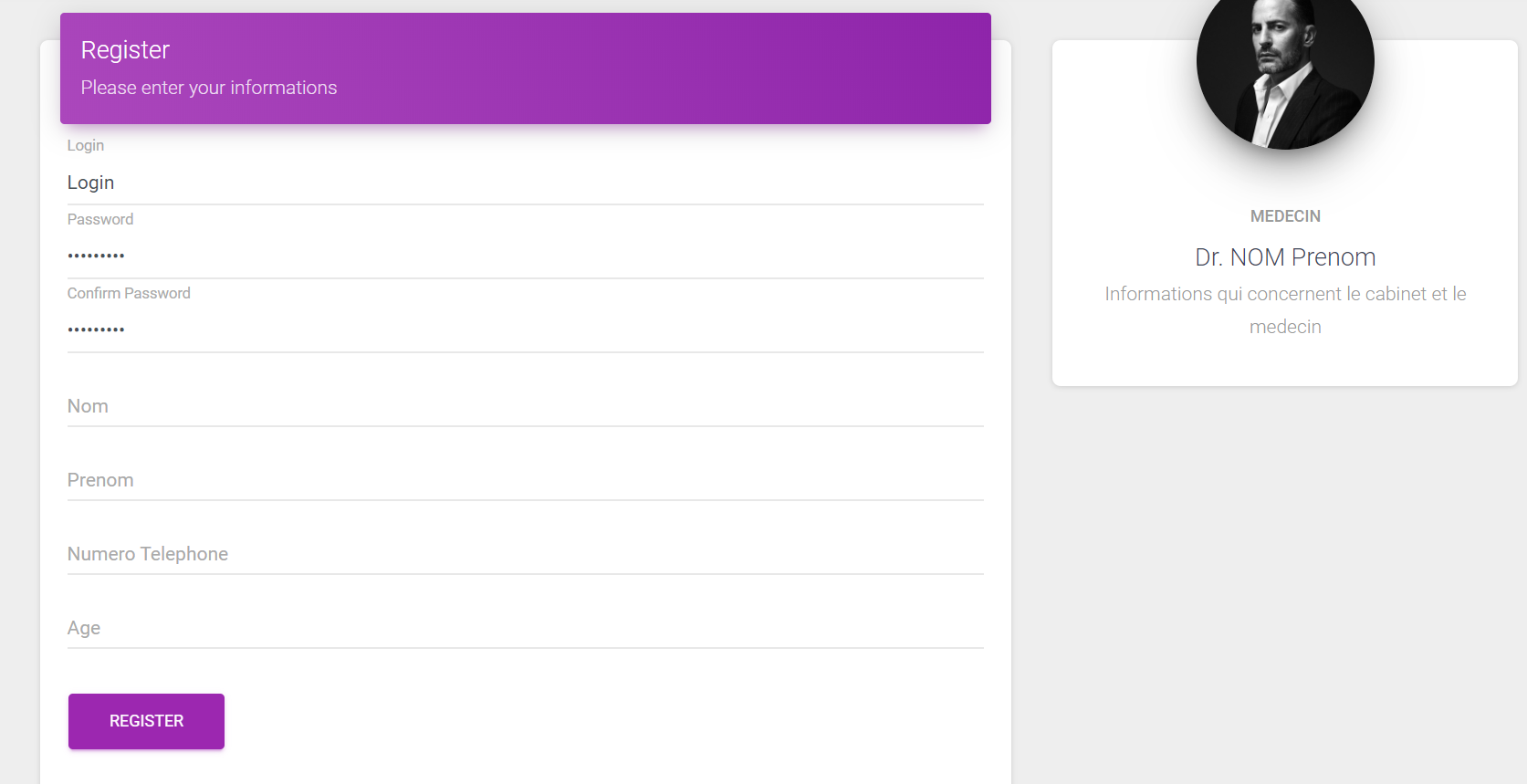


Figure 3.3 Page de Register

La figure suivante illustre l’interface ou le patient peut prendre un rendez-vous.

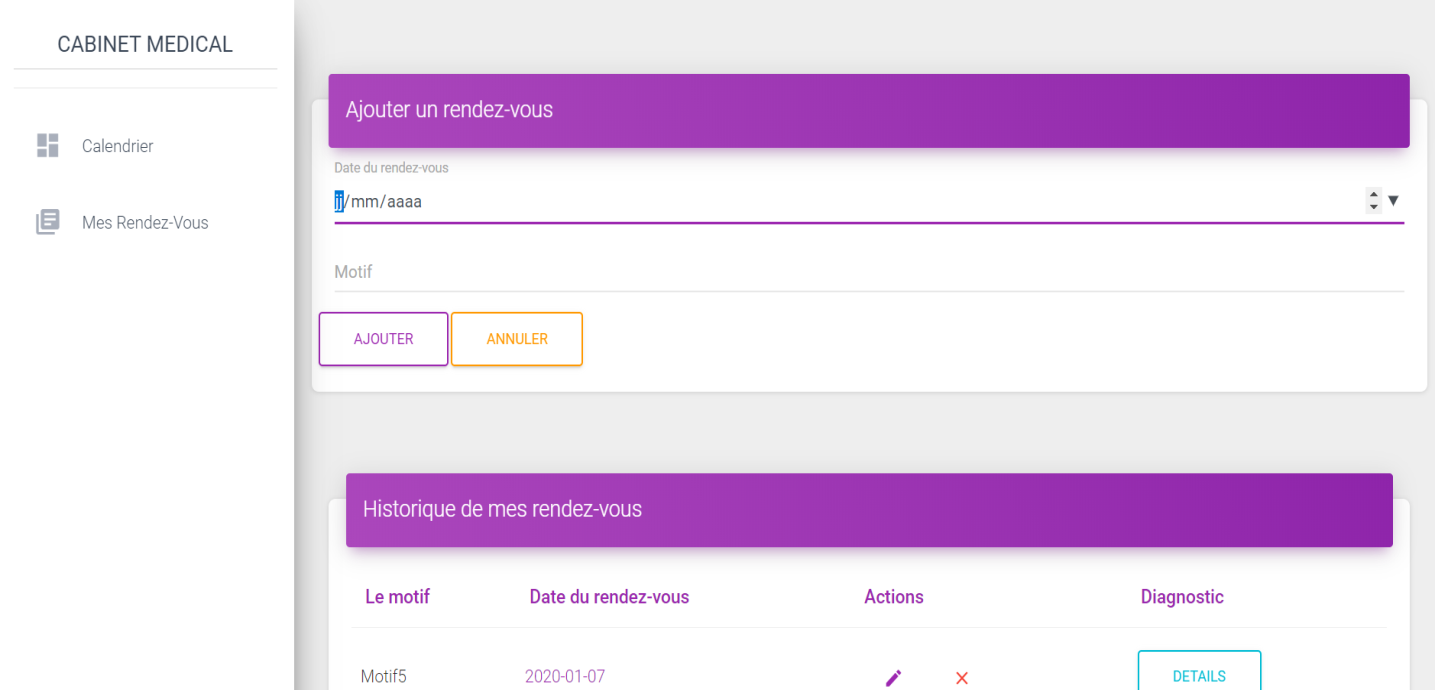


Figure 3.4 Ajout d'un rendez-vous

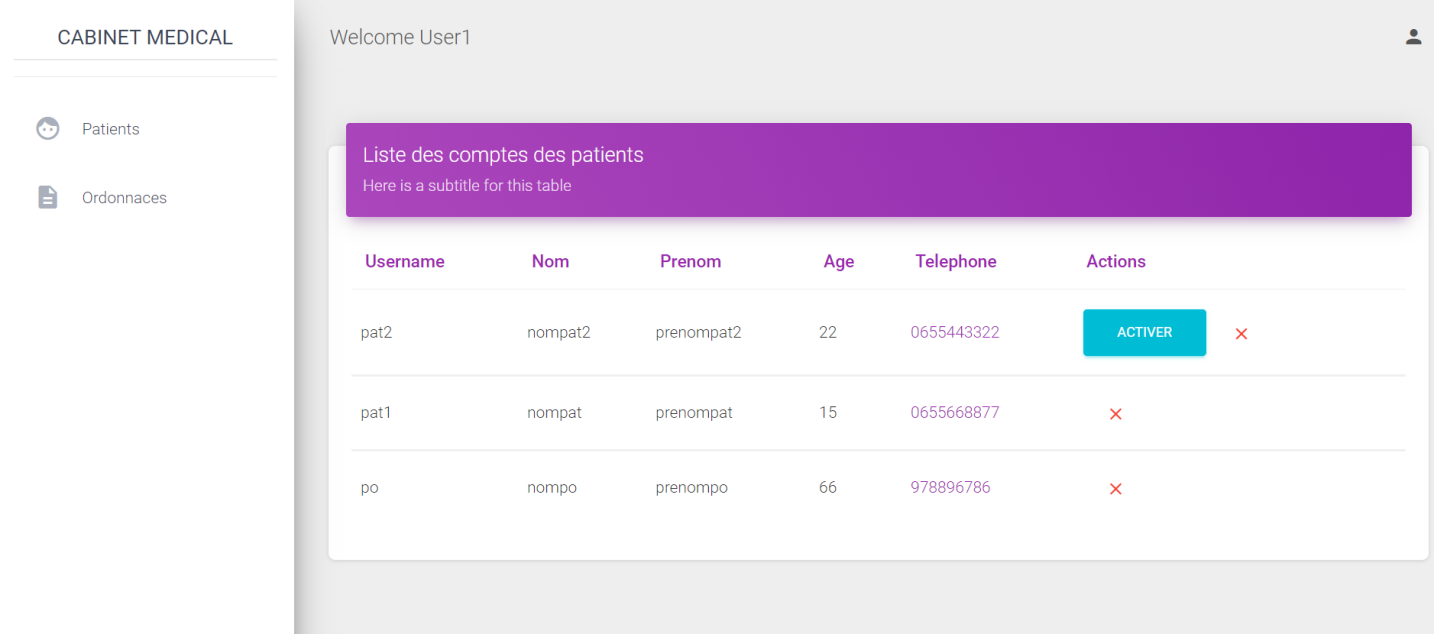


Figure 3.5 Activation des comptes

Cette figure représente l’interface qui permet au secrétaire l’activation des comptes des patients.

La figure suivante montre l’interface ou le secrétaire peut consulter l’historique des consultations, et trouver les consultations par date, il peut également imprimer l’ordonnance.

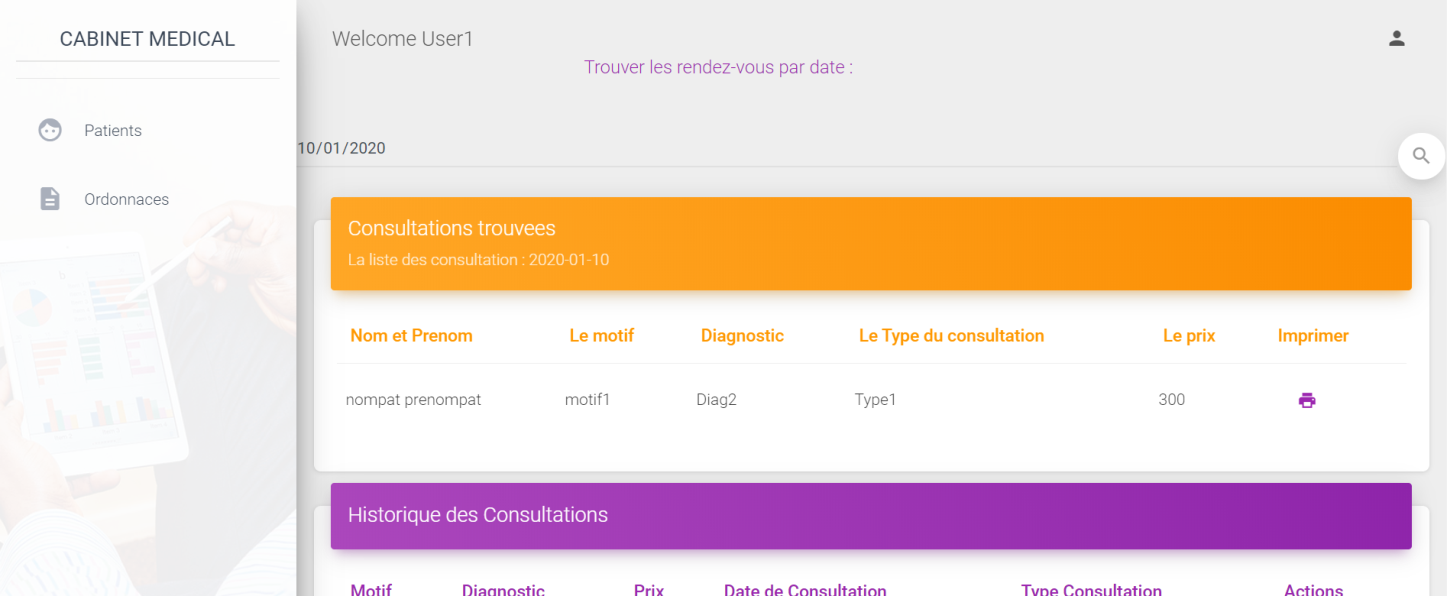


Figure 3.6 Impression de l'ordonnance

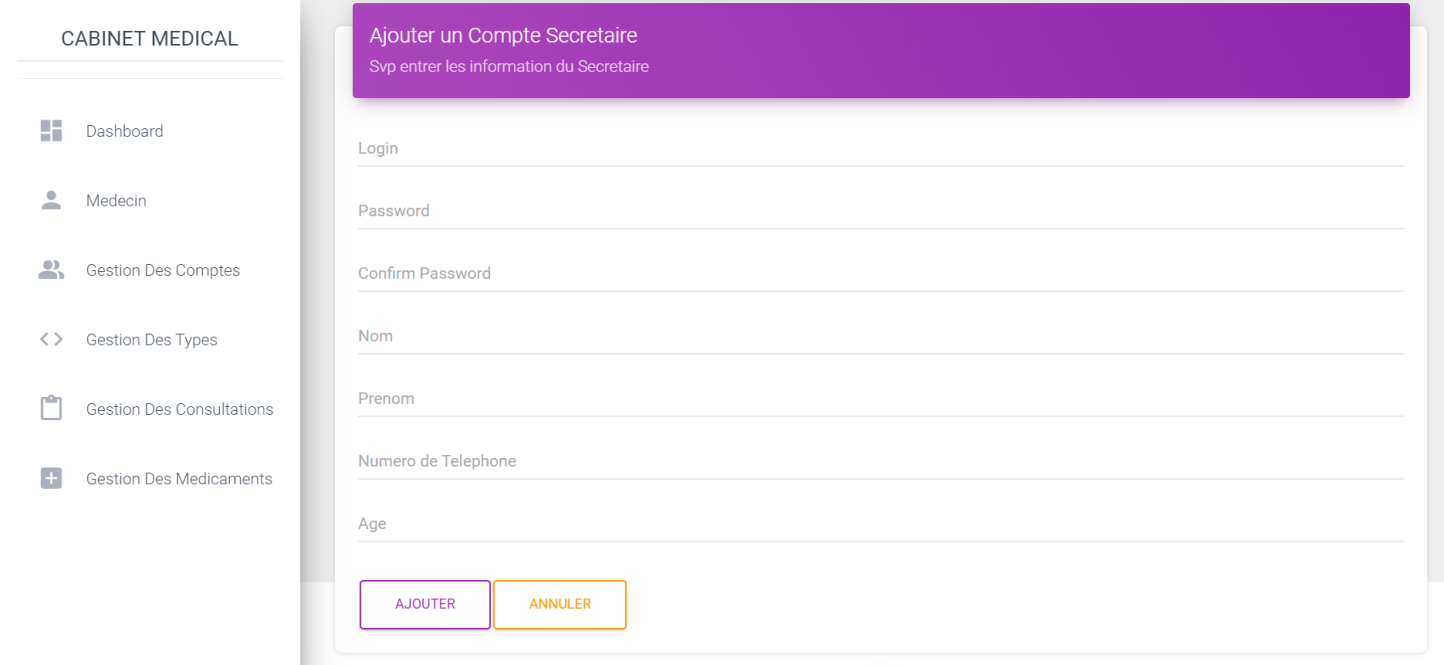


Figure 3.7 gestion des comptes

La figure ci-dessus représente le formulaire à partir duquel l’administrateur peut ajouter un secrétaire, et il peut aussi gérer les informations des comptes du système.

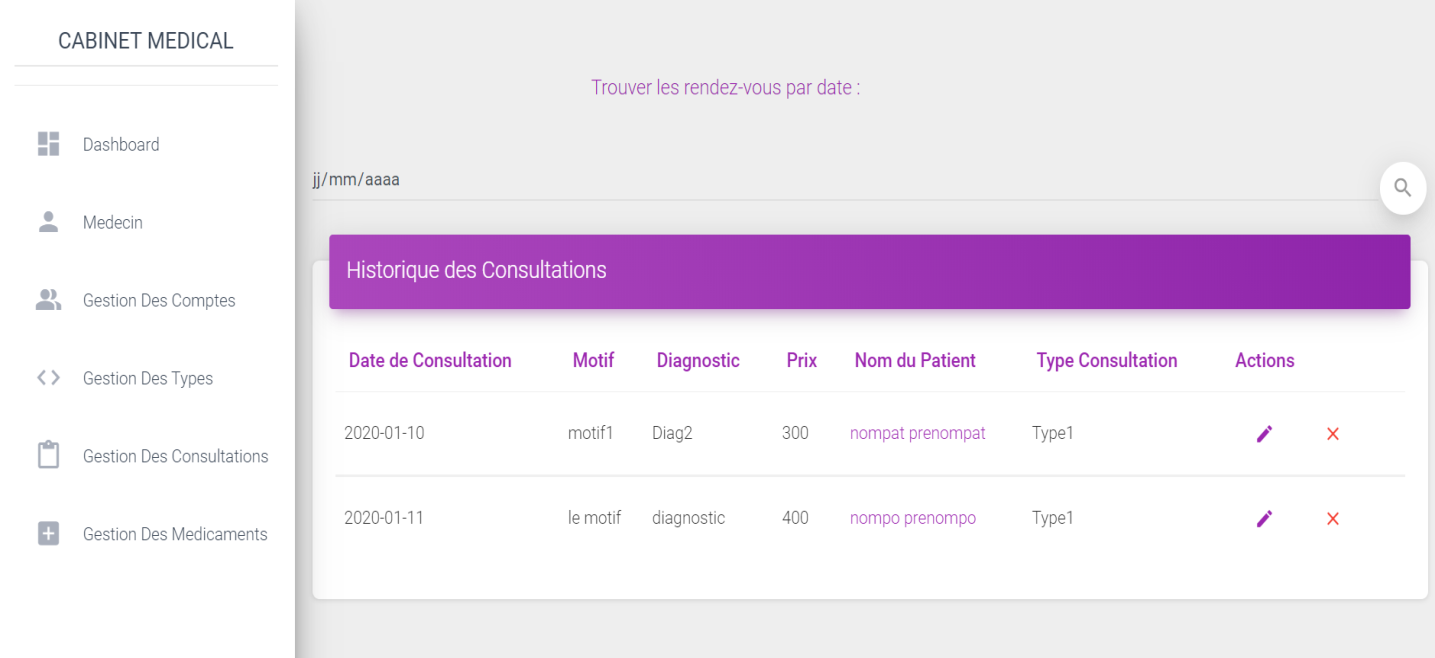


Figure 3.8 Consultation

Cette figure ci-dessus résume le processus de la consultation ou le médecin cherche la liste des rendez-vous par jour, et durant chaque consultation il peut ajouter un diagnostic, le prix, et le type de la consultation.

## **3.6 Conclusion**

Ce chapitre nous a permis de présenter les différentes technologies utilisées pour réaliser ce projet tout en mettant l’accent sur l’utilité de chacun de ces outils.

Nous avons aussi donné une idée sur le fonctionnement de l’application web à travers quelques captures d’écran.

La dernière partie restante du rapport concerne une conclusion générale ainsi que les perspectives du projet.

# **Conclusion Générale et Perspectives**

Notre mini-projet consistait en la conception et le développement d’une application de gestion d’un cabinet médical. Un tel projet vise principalement à améliorer la gestion du cabinet en offrant l'accessibilité et le stockage des données à tout moment.

Pour atteindre ces objectifs, nous avons utilisé des outils puissants et de diverses nouvelles technologies. Nous avons commencé par définir les principales fonctionnalités attendues du système cible. Ensuite, nous avons abordé l’étude fonctionnelle de l’application en utilisant le formalisme UML. Puis, nous avons défini l’architecture applicative du système tout en déterminant les Frameworks techniques nécessaires à sa réalisation, avant d’entamer l’étude conceptuelle dans laquelle nous avons élaboré la conception de la solution. Enfin, nous avons abordé la phase réalisation qui consiste à implémenter les fonctionnalités du système.

Ce mini-projet nous a permis de renforcer plusieurs notions telles que la gestion optimale du temps et l'esprit du travail en groupe. La réalisation du mini-projet nous a permis, aussi, de raffiner nos capacité de conception et de renforcer nos compétences en matière de technologies telles que Spring boot , Angular, TypeScript ...

Toutefois des améliorations de notre travail sont encore possibles. Comme perspective à ce travail : Avoir un calendrier pour mieux faciliter et organiser la réservation des rendez-vous ainsi l’organisation du planning de travail du médecin, Ensuite développé la partie mobile.

# **Références**

**[1] Outils et technologies utilises : Spring boot :** [**https://openclassrooms.com/fr/courses/4668056-construisez-des-microservices/5122425-decouvrez-le-framework-spring-boot**](https://openclassrooms.com/fr/courses/4668056-construisez-des-microservices/5122425-decouvrez-le-framework-spring-boot) **28/12/2019**

**[2] Outils et technologies utilises : Angular** [**https://fr.wikipedia.org/wiki/Angular**](https://fr.wikipedia.org/wiki/Angular) **28/12/2019**

**[3] Outils et technologies utilises : TypeScript :** [**https://fr.wikipedia.org/wiki/TypeScript**](https://fr.wikipedia.org/wiki/TypeScript) **28/12/2019**

**[4] Outils et technologies utilises : Hibernate :** [**https://fr.wikipedia.org/wiki/Hibernate**](https://fr.wikipedia.org/wiki/Hibernate) **28/12/2019**

**[5] Outils et technologies utilises : Github :** [**https://fr.wikipedia.org/wiki/Github**](https://fr.wikipedia.org/wiki/Github) **28/12/2019**

**[6] Outils et technologies utilises : Intellij :** [**https://fr.wikipedia.org/wiki/IntelliJ\_IDEA**](https://fr.wikipedia.org/wiki/IntelliJ_IDEA) **28/12/2019**

**[8] Outils et technologies utilises : Xampp :** [**https://www.apachefriends.org/fr/index.html**](https://www.apachefriends.org/fr/index.html) **28/12/2019**