

**Royaume Du Maroc**

Université Privée De Fès

Faculté des Sciences De L’ingénieur

Génie Informatique 4

**Rapport**

Sujet

Rapport De mini-Projet : Gestion SPA

Enseignante : Naoual boukil

Réalisé par : Younes Darrassi

Année universitaire : 2022/2023

Table of Contents

[I. Chapitre 1 : INTRODUCTION 3](#_Toc124806040)

[1. Remerciements 3](#_Toc124806041)

[2. Présentation du sujet 4](#_Toc124806042)

[3. Diagramme De Gantt 4](#_Toc124806043)

[II. Chapitre 2 : Etude De l’existent 5](#_Toc124806044)

[III. Chapitre 3 : Spécification Des Besoin 6](#_Toc124806045)

[1. Introduction 6](#_Toc124806046)

[2. Besoin Fonctionnels 6](#_Toc124806047)

[3. Besoins non fonctionnels 7](#_Toc124806048)

[IV. Chapitre 4 : Analyse et conception 8](#_Toc124806049)

[1) Gestion du projet 8](#_Toc124806050)

[A. Cycle de vie d’un logiciel : 8](#_Toc124806051)

[B. Cycle de vie en cascade : 10](#_Toc124806052)

[C. Cycle de vie en V : 11](#_Toc124806053)

[2) Methodologie de conception (Spring boot MVC) 12](#_Toc124806054)

[3) Diagrammes 12](#_Toc124806055)

[ INTRODUCTION 12](#_Toc124806056)

[**1.** **Définition de l’UML** 12](#_Toc124806057)

[**2.** **Analyse de cas d’utilisation « S’authentifier »** 13](#_Toc124806058)

[**3.** **Analyse de cas d’utilisation « gérer Rendez-vous »** 15](#_Toc124806059)

[[1] Diagramme de cas d’utilisation « gérer Rendez-vous » 15](#_Toc124806060)

[[2] Description : 15](#_Toc124806061)

[**4.** **Analyse de cas d’utilisation « gérer compte client » :** 17](#_Toc124806062)

[5. Diagramme De Class 19](#_Toc124806063)

[6. Diagramme d’activités : 19](#_Toc124806064)

[7. Diagramme de séquence 24](#_Toc124806065)

[V. Charte graphique et structure de l’application 25](#_Toc124806066)

[A. Charte graphique 25](#_Toc124806067)

[ Wireframe De Login 25](#_Toc124806068)

[ Wireframe D’Ajoute D’un Client 25](#_Toc124806069)

[ Wireframe List Des Clients 26](#_Toc124806070)

[ Wireframe D’ajoute D’un Rendezvous 26](#_Toc124806071)

[ Wireframe List Des Rendezvous 27](#_Toc124806072)

[ Wireframe D’Ajoute Service 27](#_Toc124806073)

[ Wireframe List Service 28](#_Toc124806074)

[B. Arboressance 28](#_Toc124806075)

[VI. Realisation 29](#_Toc124806076)

[i **Choix technologiques** 29](#_Toc124806077)

[ Introduction 29](#_Toc124806078)

[[1] Spring Boot DevTools 29](#_Toc124806079)

[[2] Thymeleaf 29](#_Toc124806080)

[[3] Spring Data JPA 30](#_Toc124806081)

[[4] Spring Web 30](#_Toc124806082)

[[5] Maven 30](#_Toc124806083)

[[6] Mysql driver 30](#_Toc124806084)

[[7] Bootstrap 31](#_Toc124806085)

[[8] Spring Security 31](#_Toc124806086)

[VII. Conclusion et perspectives 32](#_Toc124806087)

Table De Figure

[Figure 1 : Diagramme De Gantt 7](#_Toc124806088)

[Figure 2 : modèle en cascade 12](#_Toc124806089)

[Figure 3 : modèle en V 13](#_Toc124806090)

[Figure 4 : diagramme de cas d’utilisation « s’authentifier » 15](#_Toc124806091)

[Figure 5 : Diagramme de cas d’utilisation « gérer Rendez-vous » 17](#_Toc124806092)

[Figure 6 : Diagramme de cas d’utilisation « gérer compte client » 19](#_Toc124806093)

[Figure 7 : Class Diagramme 21](#_Toc124806094)

[Figure 8 : diagramme d’activité « ajouter produit médical » 22](#_Toc124806095)

[Figure 9 : Diagramme d’activité « créer compte » 23](#_Toc124806096)

[Figure 10 : Diagramme d’activité « inscrire » 24](#_Toc124806097)

[Figure 11 : Diagramme d’activité « réservation » 25](#_Toc124806098)

[Figure 12 : Diagramme de séquence « s’authentifier » 26](#_Toc124806099)

[Figure 13 : Wireframe De Login 27](#_Toc124806100)

[Figure 14 : Wireframe D’ajoute D’un Client 27](#_Toc124806101)

[Figure 15 : Wireframe List Des Clients 28](#_Toc124806102)

[Figure 16 : Wireframe D’ajoute D’un Rendezvous 28](#_Toc124806103)

[Figure 17 : Wireframe List Des Rendezvous 29](#_Toc124806104)

[Figure 18 : Wireframe D’ajoute Service 29](#_Toc124806105)

[Figure 19 : Wireframe List Service 30](#_Toc124806106)

[Figure 20 : Arboressance 30](#_Toc124806107)

[Figure 21 :spring boot logo 31](#_Toc124806108)

Table De Tableau

[Tableau 1 : sommaire d’authentification 16](#_Toc124806109)

[Tableau 2 : Sommaire de Rendez-vous 18](#_Toc124806110)

[Tableau 3 : description de cas d’utilisation <<gérer compte client>> 20](#_Toc124806111)

# Chapitre 1 : INTRODUCTION

## Remerciements

Je tiens à remercier dans un premier temps, toute l'équipe pédagogique de UPF et les intervenants professionnels responsables de la formation de la filière GENIEINFO, pour avoir assuré la partie théorique et Pratique. Je remercie également Dr. Naoual Boukil pour l’aide et les conseils concernant les missions évoquées dans ce rapport.

## Présentation du sujet

Le projet Gestion SPA a pour objectif de gérer les informations relatives aux différents salons de coiffure. Il comporte quatre tables de données principales : **Salon**, **Rendez-vous**, **Service** et **Client**.

La table Salon contient les informations sur les salons, comme l'identifiant unique (id\_salon), le nom et l'adresse.

La table Rendez-vous contient les informations sur les rendez-vous pris auprès des salons, comme la date du rendez-vous, la date de demande, l'identifiant du salon et l'identifiant du client.

La table Client contient les informations sur les clients, comme l'identifiant unique (id\_client), le nom, le prénom, le numéro de téléphone et la date de naissance.

La table Service contient les informations sur les services proposés par les salons, comme l'identifiant unique (id\_service), le nom, le prix et l'identifiant du salon.

Les données de chaque table sont reliées entre elles à travers les identifiants uniques. Cela permet de facilement retrouver les informations relatives à un salon, un rendez-vous ou un client en utilisant ces identifiants.

Ce système permet à l'admin de créer des clients et de gérer les rendez-vous, de suivre les clients, de gérer les services et les prix

L'utilisateur a la possibilité de consulter les rendez-vous pris et télécharger l'affiche de son rendez-vous pour une meilleure gestion de son planning.

## Diagramme De Gantt

Le diagramme de Gantt est un outil graphique utilisé pour planifier et suivre les projets. Il est souvent utilisé pour visualiser les tâches à effectuer, les dépendances entre les tâches et les dates d'échéance. Il permet également de suivre l'avancement des travaux.

Il est généralement représenté sous forme de barres horizontales, où chaque barre représente une tâche à effectuer. La longueur de la barre indique la durée de la tâche, et sa position horizontale indique la date de début et de fin. Les barres peuvent être regroupées en fonction de la phase du projet ou de la personne responsable de la tâche. Des flèches peuvent être utilisées pour indiquer les dépendances entre les tâches.

Le diagramme de Gantt est souvent utilisé pour planifier les projets de construction, les projets informatiques et les projets de recherche et développement. Il est également utilisé pour planifier les projets de marketing et de communication. Il peut être utilisé à différents niveaux de détail, allant des grandes tâches globales jusqu'aux tâches les plus détaillées.

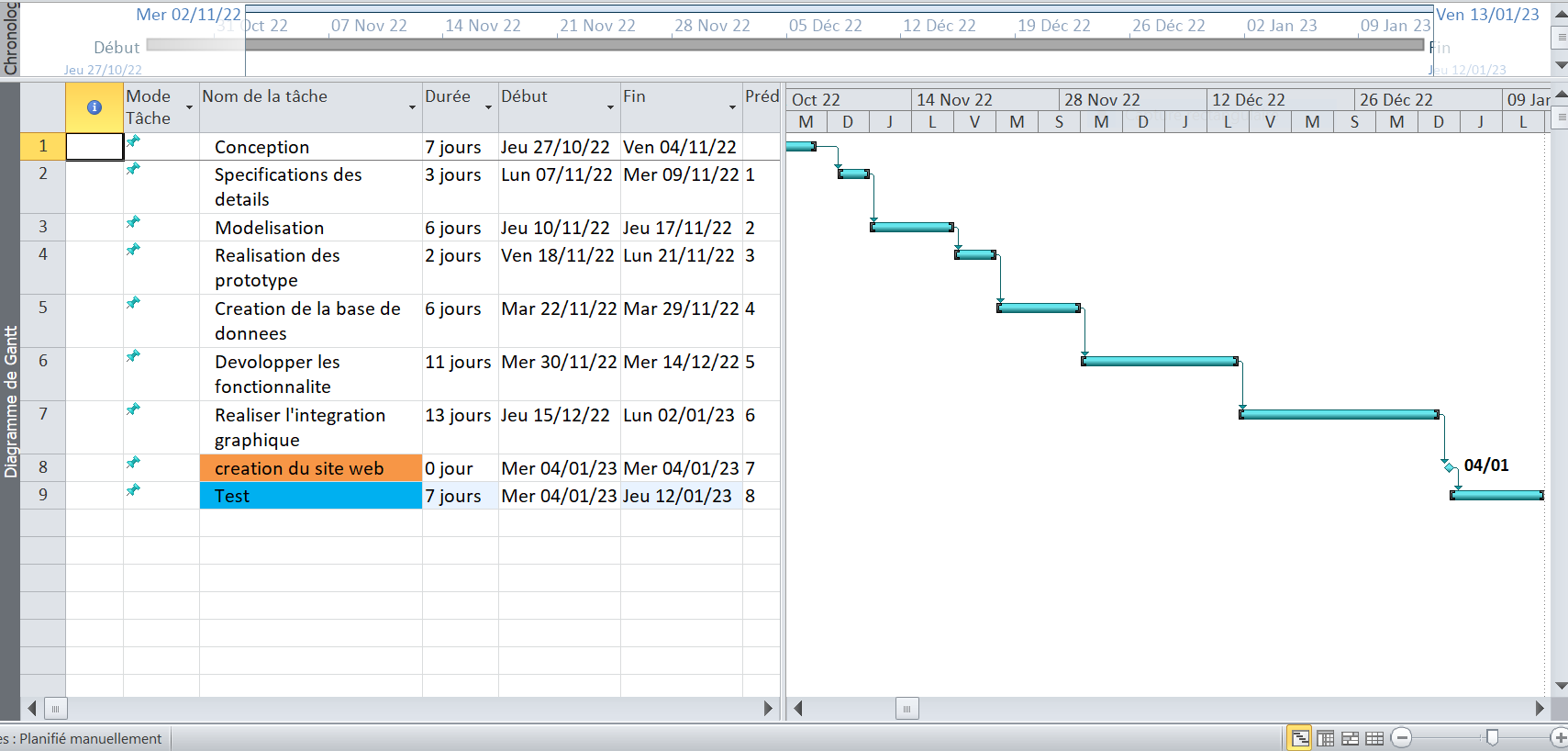
Il permet aux utilisateurs de visualiser rapidement les délais, les dépendances et les risques potentiels, de suivre l'avancement des tâches et d'ajuster les plans en conséquence. Il est un outil pratique pour la gestion de projet, il permet de planifier, d'organiser et de suivre les tâches pour une meilleure coordination et une meilleure efficacité.

Figure  : Diagramme De Gantt

# Chapitre 2 : Etude De l’existent

L'étude de l'existant du système permet de comprendre les fonctionnalités et les processus actuels utilisés pour gérer les informations relatives aux salons de coiffure, aux rendez-vous, aux clients et aux services. Ce système permet à l'administrateur de créer des clients et de gérer les rendez-vous, de suivre les clients, de gérer les services et les prix.

Les données sont stockées dans 4 tables: Salon, Rendez-vous, Client et Service. La table Salon contient les informations de base sur les salons, comme l'identifiant unique (id\_salon), le nom et l'adresse. La table Rendez-vous contient les informations sur les rendez-vous pris auprès des salons, comme la date du rendez-vous, la date de demande, l'identifiant du salon et l'identifiant du client. La table Client contient les informations sur les clients, comme l'identifiant unique (id\_client), le nom, le prénom, le numéro de téléphone et la date de naissance. La table Service contient les informations sur les services proposés par les salons, comme l'identifiant unique (id\_service), le nom, le prix et l'identifiant du salon.

En plus de cela, l'utilisateur a la possibilité de consulter les rendez-vous pris et de télécharger l'affiche de son rendez-vous pour une meilleure gestion de son planning. Le système permet donc à l'administrateur de gérer les informations de manière efficace et à l'utilisateur de suivre ses rendez-vous de manière simple.

# Chapitre 3 : Spécification Des Besoin

## Introduction

La spécification des besoins est un document qui décrit les fonctionnalités et les exigences nécessaires pour répondre aux besoins d'un système ou d'un projet. Il est utilisé pour communiquer les besoins aux développeurs, aux designers et aux autres parties prenantes, et pour s'assurer que tout le monde est sur la même longueur d'onde.

* La spécification des besoins pour un système de gestion de rendez-vous pour salons de coiffure devrait comprendre les éléments suivants :
* Descriptions des utilisateurs : décrire les différents types d'utilisateurs (administrateur, employé, client) et leurs rôles dans le système.
* Fonctionnalités clés : décrire les fonctionnalités clés du système, telles que la gestion des rendez-vous, la gestion des clients, la gestion des services et des prix, et la génération de rapports.
* Contraintes : décrire les contraintes techniques et opérationnelles, telles que les exigences en matière de sécurité, les exigences en matière de performance et les exigences en matière de compatibilité avec les systèmes existants.
* Exigences en matière d'interface utilisateur : décrire les exigences en matière d'interface utilisateur, telles que les exigences en matière de design, les exigences en matière de navigation et les exigences en matière de facilité d'utilisation.
* Acceptation : décrire les critères d'acceptation pour le système, tels que les tests à effectuer et les critères de performance à atteindre.

En résumé, la spécification des besoins est un document clé pour garantir que le système

de gestion de rendez-vous pour salons de coiffure répond aux besoins des utilisateurs et aux exigences techniques. Il est utilisé pour définir les objectifs du système, décrire les fonctionnalités clés, les contraintes et les exigences en matière d'interface utilisateur. Il est également utilisé pour définir les critères d'acceptation et les tests à effectuer pour s'assurer que le système répond aux exigences définies. En utilisant une spécification des besoins détaillée, on peut garantir que le système final répondra aux besoins des utilisateurs et sera fonctionnel, fiable et facile à utiliser.

## Besoin Fonctionnels

Les besoins fonctionnels pour un système de gestion de rendez-vous pour salons de coiffure pourraient inclure:

1. Gestion des clients : la possibilité pour l'administrateur de créer, mettre à jour et supprimer des clients dans la base de données.
2. Gestion des rendez-vous : la possibilité pour l'administrateur de programmer, mettre à jour et annuler des rendez-vous pour les clients, avec la possibilité de voir les rendez-vous à venir et passés.
3. Gestion des services : la possibilité pour l'administrateur de gérer les services proposés par le salon, y compris l'ajout, la modification et la suppression de services, ainsi que la gestion des prix.
4. Suivi des clients : la possibilité pour l'administrateur de suivre les clients, y compris les rendez-vous passés et les services utilisés.
5. Accès utilisateur : la possibilité pour les clients de consulter leurs rendez-vous à venir et de télécharger l'affiche de leur rendez-vous pour une meilleure gestion de leur planning.
6. Génération de rapports: la possibilité pour l'administrateur de générer des rapports sur les rendez-vous, les clients, les services et les revenus pour aider à la prise de décision et à la gestion de la performance du salon.

En résumé, les besoins fonctionnels décrivent les fonctionnalités clés du système qui doivent être développées pour répondre aux besoins des utilisateurs et pour gérer efficacement les informations relatives aux salons de coiffure, aux rendez-vous, aux clients et aux services.

## Besoins non fonctionnels

Les besoins non fonctionnels pour un système de gestion de rendez-vous pour salons de coiffure pourraient inclure:

1. Sécurité : la sécurité des données, y compris les informations sur les clients et les rendez-vous, doit être protégée contre les accès non autorisés.
2. Fiabilité : le système doit être fiable et disponible en tout temps pour les utilisateurs.
3. Performances : le système doit être capable de gérer un grand nombre de rendez-vous et de clients sans ralentir les performances.
4. Conformité : le système doit être conforme aux réglementations et aux normes de l'industrie en matière de protection de la vie privée et de sécurité des données.
5. Scalabilité : le système doit être capable de s'adapter à une croissance future du nombre de clients et de rendez-vous.
6. Interopérabilité : le système doit être compatible avec les systèmes et les technologies existants pour un intégration facile.
7. Facilité d'utilisation : le système doit être facile à utiliser pour les utilisateurs, avec une interface utilisateur claire et intuitive.

En résumé, les besoins non fonctionnels décrivent les exigences qui ne sont pas directement liées aux fonctionnalités du système, mais qui sont importantes pour garantir que le système est sécurisé, fiable, performant, conforme, évolutif et facile à utiliser.

# Chapitre 4 : Analyse et conception

Pour l'analyse et la conception d'un système de gestion de rendez-vous pour salons de coiffure, il pourrait être nécessaire de réaliser les étapes suivantes:

* Analyse des utilisateurs : identifier les différents types d'utilisateurs (administrateur, employé, client) et leurs rôles dans le système, ainsi que leurs besoins et leurs objectifs.
* Analyse des exigences : identifier les fonctionnalités clés du système, les contraintes techniques et les exigences en matière d'interface utilisateur à partir des besoins des utilisateurs.
* Conception des modèles : créer des modèles pour les différentes parties du système, tels que les diagrammes de classes, les diagrammes de séquence, les diagrammes d'état, les diagrammes de cas d'utilisation, et les diagrammes de déploiement.
* Conception de la base de données : concevoir la structure de la base de données pour stocker les informations sur les salons, les rendez-vous, les clients et les services.
* Conception de l'interface utilisateur : concevoir l'interface utilisateur pour qu'elle soit intuitive, conviviale et facile à utiliser pour les utilisateurs.
* Vérification : vérifier que les modèles et les spécifications répondent aux besoins et aux exigences identifiés lors de l'analyse.

En résumé, l'analyse et la conception sont des étapes clés pour garantir que le système de gestion de rendez-vous pour salons de coiffure répond aux besoins des utilisateurs et aux exigences techniques.

## Gestion du projet

### Cycle de vie d’un logiciel :

Le « cycle de vie d'un logiciel » désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition. L'objectif d'un tel découpage est de permettre de définir des jalons intermédiaires permettant la validation du développement logiciel, c'est-à-dire il a conformité du logiciel avec les besoins exprimés, et la vérification du processus de développement, c'est-à-dire l'adéquation des méthodes mises en œuvre. Il comprend généralement au minimale les activités suivantes :

* + **Définition des objectifs**, consistant à définir la finalité du projet et son inscription dans une stratégie globale.
  + **Analyse des besoins et faisabilité**, c'est-à-dire l'expression, le recueil et la formalisation des besoins du demandeur (le client) et de l'ensemble des contraintes.
  + **Conception générale,** Il s'agit de l'élaboration des spécifications de l'architecture générale du logiciel.
  + **Conception détaillée**, consistant à définir précisément chaque sous-ensemble du logiciel.
  + **Codage** (Implémentation ou programmation), soit la traduction dans un langage de programmation des fonctionnalités définies lors de phases de conception.
  + **Tests unitaires**, permettant de vérifier individuellement que chaque sous-ensemble du logiciel est implémenté conformément aux spécifications.
  + **Intégration**, dont l'objectif est de s'assurer de l'interfaçage des différents éléments (modules) du logiciel. Elle fait l'objet de tests d'intégration consignés dans un document.
  + **Qualification** (ou recette), c'est-à-dire la vérification de la conformité du logiciel aux spécifications initiales.
  + **Documentation**, visant à produire les informations nécessaires pour l'utilisation du logiciel et pour des développements ultérieurs.
  + **Mise en production, Maintenance**, comprenant toutes les actions correctives (maintenance corrective) et évolutives (maintenance évolutive) sur le logiciel.

### Cycle de vie en cascade :

En génie logiciel, le modèle en cascade est une version répandue du modèle de gestion du cycle de développement des systèmes et des applications. Souvent considéré comme l'approche classique du développement, ce modèle décrit un cycle linéaire et séquentiel. Son alternative la plus connue aujourd’hui est la méthodologie Agile. Mais Une fois qu'une application se trouve en phase de test, il est très difficile de revenir en arrière afin de modifier un élément mal conçu lors des phases précédentes.

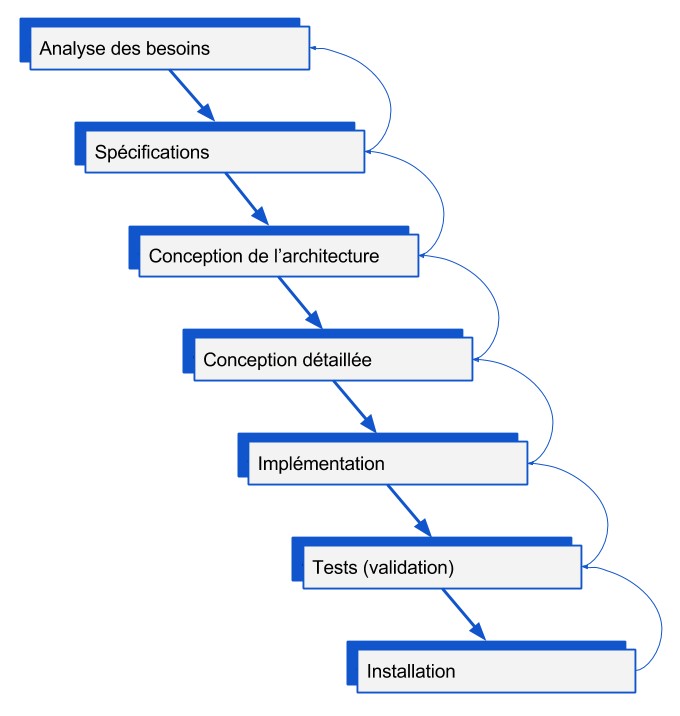


Figure  : modèle en cascade

### Cycle de vie en V :

Le modèle du cycle en V est un modèle conceptuel de gestion de projet imaginé suite au problème de réactivité du modèle en cascade. Il permet, en cas d'anomalie, de limiter un retour aux étapes précédentes. Les phases de la partie montante doivent renvoyer de l'information sur les phases en vis-à-vis lorsque des défauts sont détectés, afin d'améliorer le logiciel.

|  |
| --- |
| Analyse des besoins et faisabilité |

|  |
| --- |
| Recette |

Tests de

Validation

Spécifications

Tests

D’intégration

Conception

Architecturale

Tests unitaire

Conception

Détaillée

Codage

Figure  : modèle en V

* **Pourquoi je choisis le model en V ?**

Dans ce cas je choisis le model en V pour les raisons suivantes :

* Ce model faire le parallèle entre les phases de conception et les phases de test.
* Permettant de discipliner le processus de développement.
* Simple et facile à comprendre
* Force la documentation : une phase ne peut se terminer avant qu’un document Soit validé
* Le test est inhérent à chaque phase
* Les progrès sont tangibles (pour l’équipe de développement)

## Methodologie de conception (Spring boot MVC)

Spring Boot est un cadre de développement qui facilite la création d'applications web basées sur Spring Framework. **Il utilise le modèle MVC** (Modèle-Vue-Contrôleur) pour structurer les applications. Voici une méthodologie de conception possible pour utiliser Spring Boot MVC :

* Étape de planification : définir les objectifs de l'application et les fonctionnalités nécessaires, identifier les utilisateurs et les exigences.
* Étape de conception : créer un modèle conceptuel de l'application en utilisant des diagrammes UML, tels que les diagrammes de classes, les diagrammes de séquence, les diagrammes de cas d'utilisation, etc.
* Étape de développement : créer les modèles, les vues et les contrôleurs en utilisant Spring Boot et Spring MVC. Les modèles représentent les données de l'application, les vues représentent la présentation des données et les contrôleurs gèrent les interactions avec les utilisateurs.
* Étape de test : tester les différentes parties de l'application pour s'assurer qu'elles fonctionnent correctement et qu'elles répondent aux exigences.
* Étape de déploiement : déployer l'application sur un serveur web et la rendre disponible pour les utilisateurs.
* Étape de maintenance : maintenir et améliorer l'application en fonction des besoins des utilisateurs et des exigences évolutives.

En utilisant cette méthodologie, vous pouvez développer des applications web robustes et évolutives en utilisant Spring Boot MVC. Il permet également de respecter les bonnes pratiques de développement et de faciliter la maintenance future.

## Diagrammes

### INTRODUCTION

Dans ce présent chapitre, je propose d’analyser les différents cas d’utilisation que je viens de repérer à travers l’activité de capture des besoins afin de déterminer les différentes classes intervenant dans chacun des cas et de repérer le séquencèrent des flux pour chaque scénario de réalisation. Il s’agit donc là d’une activité importante, qui sert de base pour la conception.

### **Définition de l’UML**

UML n'est pas une méthode, une description normative des étapes de la modélisation : les auteurs d'UML ont en effet estimé qu'il n'était pas opportun de définir une telle méthode en raison de la diversité des cas particuliers. Ils ont préféré être plus modestes, et définir un langage graphique permettant de représenter, de communiquer les divers aspects d'un système d'information (aux graphiques sont bien sûr associés des textes qui expliquent leur contenu). On dit parfois qu'UML est un métalangage car il fournit les éléments permettant de construire le modèle qui sera, lui, le langage de l'entreprise.

UML comporte 12 diagrammes standards représentant autant de "vues" d'un système d'information. Ces diagrammes se répartissent en trois catégories : quatre représentent la structure statique de l'application (diagrammes de classe, d'objet, de composant et de déploiement) ; cinq représentent son comportement dynamique (diagrammes de cas d'utilisation, de séquence, d'activité, de collaboration et d'état) ; trois représentent la façon dont on peut organiser et gérer les modules qui composent le programme (diagrammes de packages, sous-systèmes et modèles)

### **Analyse de cas d’utilisation « S’authentifier »**

#### **Diagramme de cas d’utilisation « s’authentifier »**

Figure  : diagramme de cas d’utilisation « s’authentifier »

#### **Description**

|  |
| --- |
| Sommaire d’authentification |
| **Titre :** S’authentifier |
| **But :** Authentification et autorisation d’accès. |
| **Résumé :** Le client introduit son login et mot de passe pour accéder au système. |
| **Acteur :** Client |
| **DESCRIPTION DES ENCHAINEMENTS conditions** |
| Le client doit avoir un compte  Accès à son espace privé. |
| SCENARIO NOMINAL |
| 1. Le client demande l’accès au système 2. Le système affiche le formulaire d’authentification. 3. Le client saisit son login et son mot de passe. 4. Le système vérifie les champs (champs obligatoires…). 5. Le système vérifie l’existence de l’utilisateur. 6. Si le client est identifié, le système affiche l’interface de privée. |

Tableau  : sommaire d’authentification

### **Analyse de cas d’utilisation « gérer Rendez-vous »**

### Diagramme de cas d’utilisation « gérer Rendez-vous »

Figure  : Diagramme de cas d’utilisation « gérer Rendez-vous »

### Description :

|  |  |
| --- | --- |
| **Sommaire de gérer Rendez-vous** | |
| **Titre :** Gérer rendez-vous | |
| **But :** Réserver un rendez-vous. | |
| **Résumé :** après la consultation de nouveauté de site le client choisis son réservation et télécharger le rendez-vous | |
| **Acteur :** Client et administrateur | |
| **DESCRIPTION DES ENCHAINEMENTS** | |
| Pré conditions | Post conditions |
| -authentification  -créer compte | - Réservation prête pour le client.  -consulter les réservation disponibles |
| SCENARIO NOMINAL | |
| 1. Le client introduit son login et son mot de passe  **2.** Le système vérifie les champs (champs obligatoires...).   1. Le système vérifie l’existence de l’utilisateur. 2. Si le client est identifié, le système affiche l’interface de réservation. 3. Le client choisi le Produit disponible et le réserver. | |
| ENCHAINEMENT ALTERNATIF | |
| **E1 :** Champs obligatoires non valides et/ou vides   1. Le système affiche un message d’erreur.   Le scénario reprend de **1.**  **E2 :** Login ou mot de passe non valide   1. Le système affiche un message d’erreur « Accès refusé »,   Le scénario reprend de **1.** | |

Tableau  : Sommaire de Rendez-vous

### **Analyse de cas d’utilisation « gérer compte client » :**

#### Diagramme de cas d’utilisation « gérer compte client »

Figure  : Diagramme de cas d’utilisation « gérer compte client »

#### Description :

|  |  |
| --- | --- |
| **Sommaire de gérer compte client** | |
| **Titre :** Gérer compte client | |
| **But :** l’administrateur consulte le compte user | |
| **Résumé :** le client créé une compte et l’administrateur le consulte pour l’ajouter ou le modifier ou le supprimer | |
| **Acteurs :** administrateur | |
| **DESCRIPTION DES ENCHAINEMENTS** | |
| Pré conditions | Post conditions |
| -authentification | Enregistrement de l’opération qui a eu Lieu dans la base de données. Affichage d’un message de succès |
| **SCENARIO NOMINAL** | |
| 1 : l’administrateur demande au système de lui retourner la liste des utilisateurs 2 : l’administrateur peut consultée les informations sur un compte.   1. : Le système affiche le compte 2. : l’administrateur peut supprimer le compte en question s’il veut. | |
| **ENCHAINEMENT ALTERNATIF** | |
| **E1 :** Coupure de connexion : fermeture de la session de Super Administrateur.  1 : Problème avec la base de données.  **E2 :** client n’a pas de compte  Système reprend de 1 | |

Tableau  : description de cas d’utilisation <<gérer compte client>>

### Diagramme De Class

Figure  : Class Diagramme

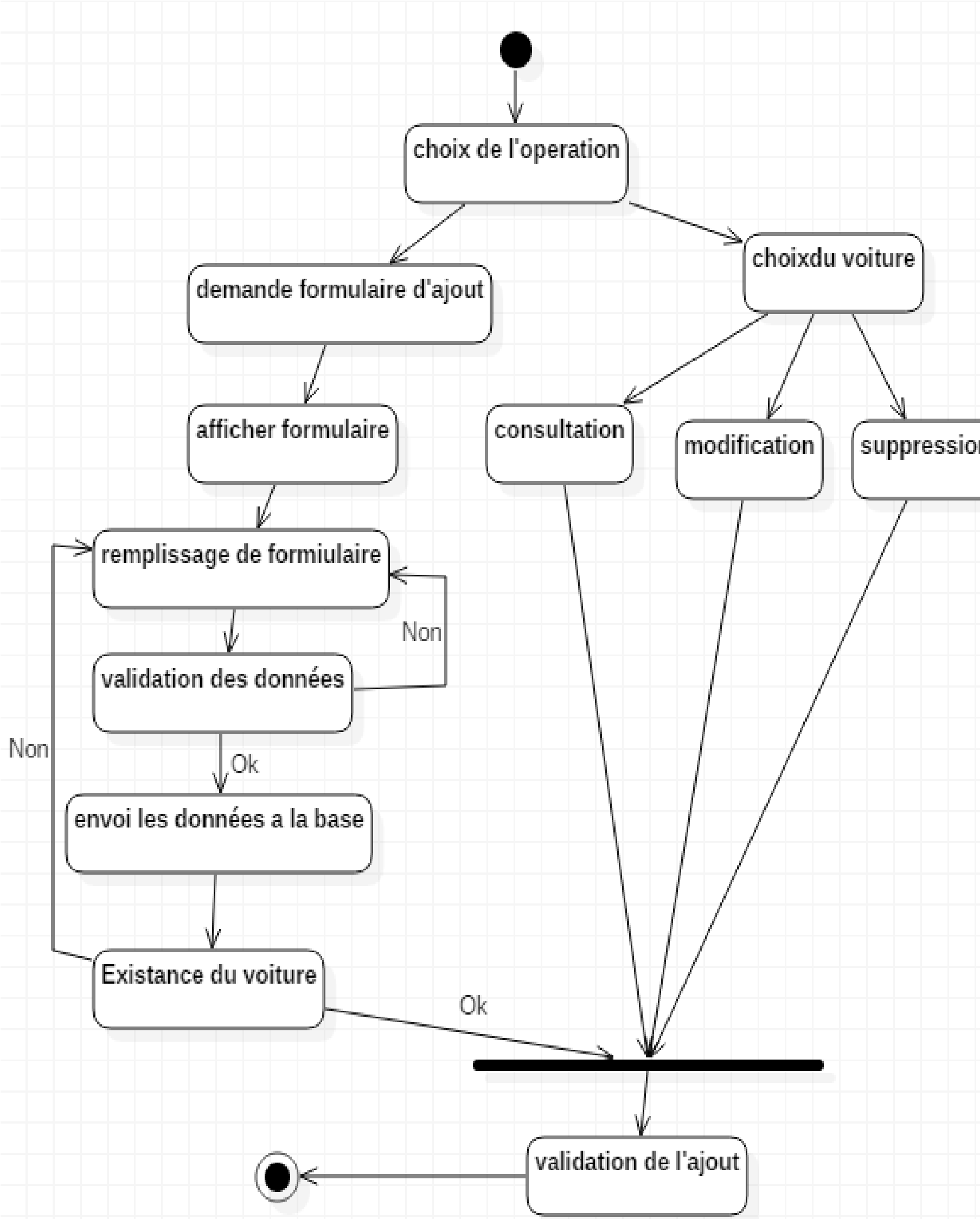
### Diagramme d’activités :

Un diagramme d'activités permet de modéliser un processus interactif, global ou partiel pour un système donné (logiciel, système d'information). Il est recommandable pour exprimer une dimension temporelle sur une partie du modèle, à partir de diagrammes de classes ou de cas d'utilisation, par exemple. Le diagramme d'activités est une représentation proche de l'organigramme ; la description d'un cas d'utilisation par un diagramme d'activités correspond à sa traduction algorithmique. Une activité est l'exécution d'une partie du cas d'utilisation, elle est représentée par un rectangle aux bords arrondis. Dans ce qui suit, nous présentons les diagrammes d’activités pour quelques cas d’utilisation dans notre système.

#### Diagramme d’activité « ajouter un salon ou service ou client ou rendez-vous »

Existence service

Service

Figure  : diagramme d’activité « ajouter produit médical »

#### Diagramme d’activité créer compte

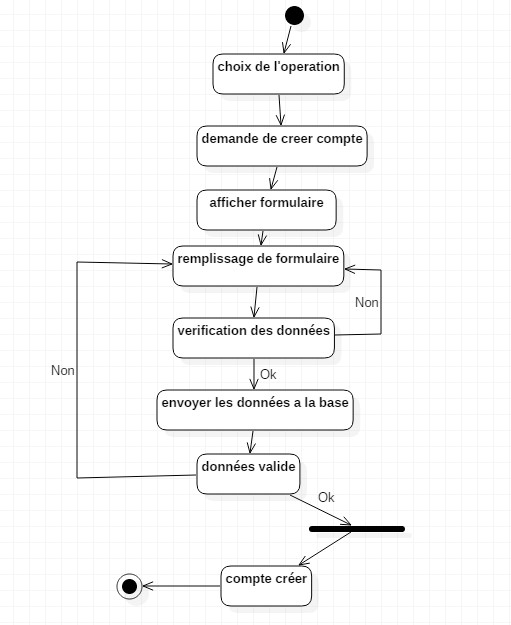


Figure  : Diagramme d’activité « créer compte »

#### Diagramme d’activité « inscription »

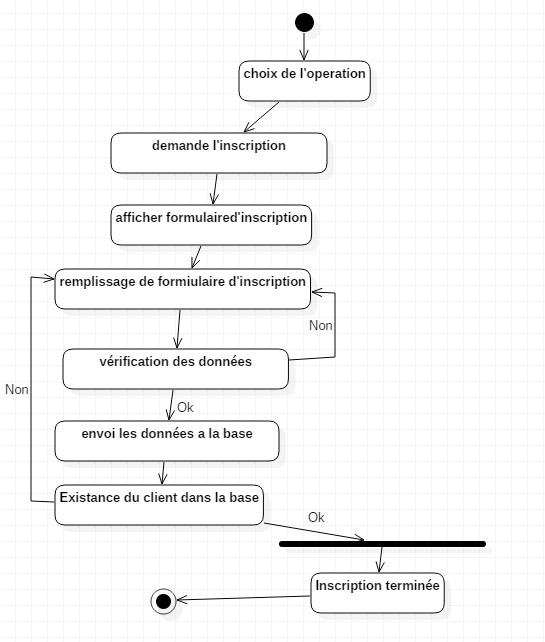
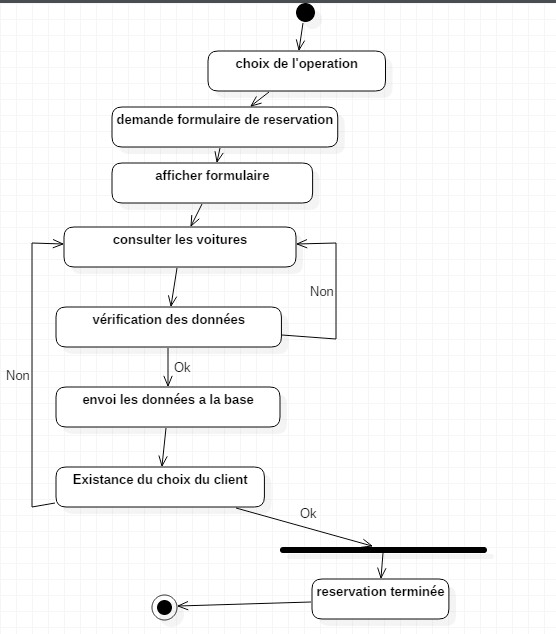


Figure  : Diagramme d’activité « inscrire »

#### Diagramme d’activité « réservation »



Consulter les réservations

Figure  : Diagramme d’activité « réservation »

### Diagramme de séquence

Les diagrammes de séquences sont la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation UML. Dans ce qui suit, je présente le diagramme de séquence pour chaque cas d’utilisation dans mon application

#### Diagramme de séquence « authentification »

Figure  : Diagramme de séquence « s’authentifier »

# Charte graphique et structure de l’application

## Charte graphique

### Wireframe De Login

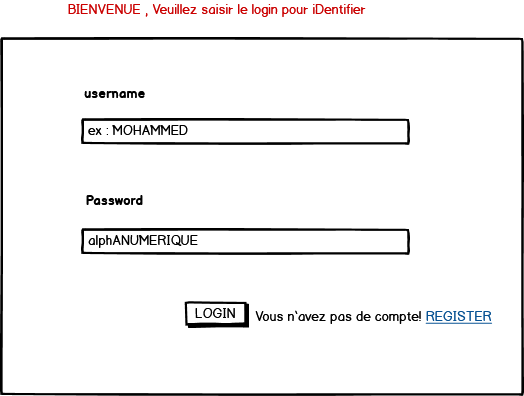


Figure  : Wireframe De Login

### Wireframe D’Ajoute D’un Client

Figure  : Wireframe D’ajoute D’un Client

### Wireframe List Des Clients

Figure  : Wireframe List Des Clients

### Wireframe D’ajoute D’un Rendezvous

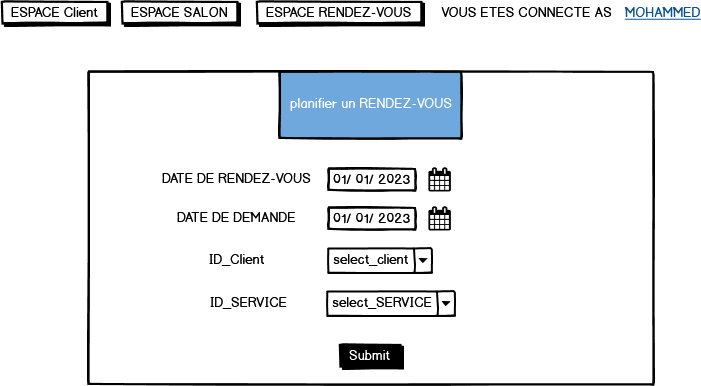


Figure  : Wireframe D’ajoute D’un Rendezvous

### Wireframe List Des Rendezvous

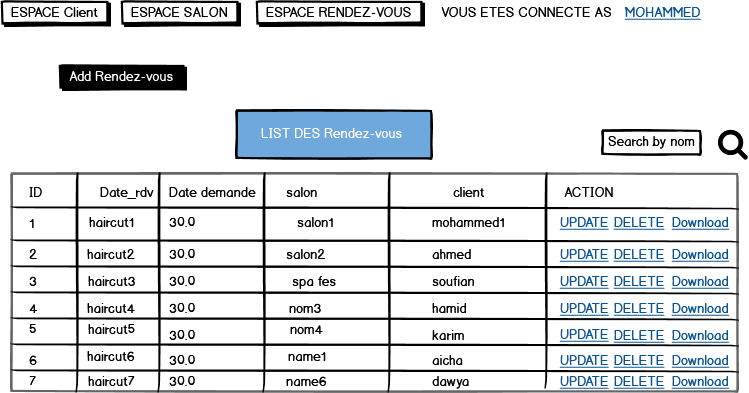


Figure  : Wireframe List Des Rendezvous

### Wireframe D’Ajoute Service

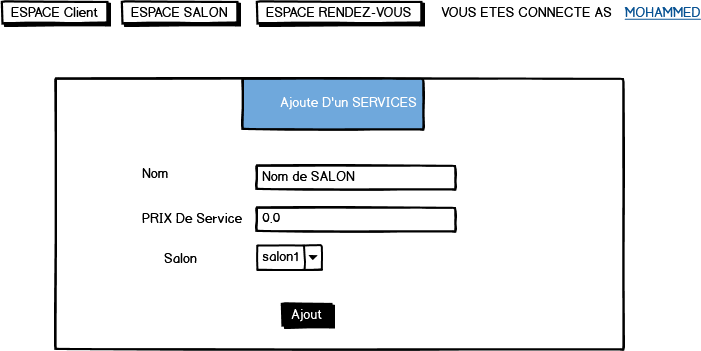


Figure  : Wireframe D’ajoute Service

### Wireframe List Service

Figure  : Wireframe List Service

## Arboressance

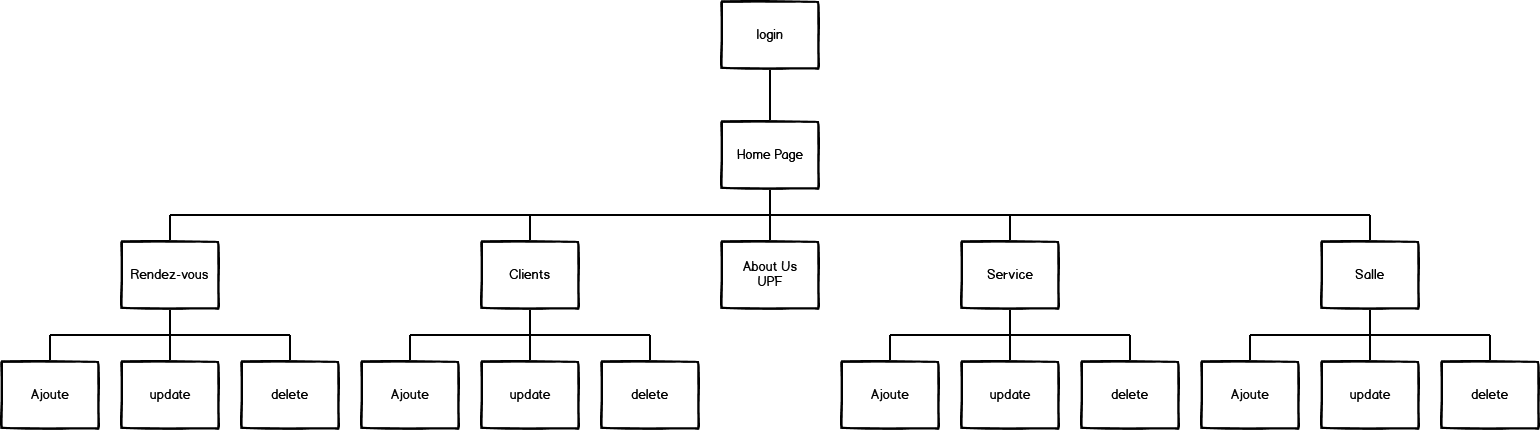


Figure  : Arboressance

# Realisation

## **Choix technologiques**

### Introduction

Spring Boot est un framework qui facilite la création d'applications web basées sur Spring Framework. Il permet de configurer rapidement et facilement un projet avec les fonctionnalités couramment utilisées dans les applications web, telles que la gestion de la sécurité, les contrôleurs REST, les vues et les services d'accès aux données.

Il offre également des fonctionnalités telles que l'auto-configuration, qui permet de configurer automatiquement les composants de l'application en fonction des dépendances présentes dans le projet, et l'intégration de différents outils de développement comme les outils de démarrage rapide, les outils de déploiement et les outils de surveillance.

De plus, Spring Boot offre une grande flexibilité en matière de configuration, permettant aux développeurs de s'adapter aux besoins de leur projet. Il facilite également la mise en place de tests automatisés et offre un support pour les différentes bases de données et technologies de stockage.

En résumé, Spring Boot est un choix populaire pour les développeurs car il fournit une configuration rapide, une grande flexibilité, une facilité de mise en place de tests automatisés et un support pour les différentes bases de données et technologies de stockage.



Figure  :spring boot logo

### Spring Boot DevTools

Spring Boot DevTools est un ensemble d'outils qui facilitent le développement d'applications Spring. Il offre des fonctionnalités telles que le redémarrage automatique de l'application lors de modifications de code, une console de débogage intégrée, et la possibilité de surveiller les dépendances de l'application.

### Thymeleaf

Thymeleaf est un moteur de modèle de langage de balisage utilisé pour générer des vues dans les applications web. Il permet de créer des modèles HTML riches et de les rendre avec des données dynamiques. Il possède une syntaxe facile à apprendre et est compatible avec les principaux frameworks web.

### Spring Data JPA

Spring Data JPA est une bibliothèque qui facilite l'utilisation de JPA (Java Persistence API) pour gérer les données de l'application. Il fournit des outils pour créer des requêtes à la base de données en utilisant des méthodes de déclaration, et permet également de gérer les transactions de manière transparente.

### Spring Web

Spring Web est un module de Spring qui fournit un ensemble de composants pour développer des applications web, comme des contrôleurs pour gérer les requêtes HTTP, des formulaires pour la validation des données, et des outils pour gérer les sessions et la sécurité.

### Maven

Maven est un outil de gestion et de construction de projets qui permet de gérer les dépendances et les plugins utilisés dans un projet. Il facilite la gestion des dépendances en permettant de déclarer les dépendances nécessaires dans un fichier de configuration, et télécharge automatiquement les dépendances manquantes lors de la construction du projet.

Maven est très utile pour les projets Spring Boot car il permet de gérer les dépendances de Spring et les plugins Spring Boot, ainsi que les dépendances d'autres bibliothèques utilisées dans le projet. Il permet également de générer des rapports de couverture de code, des javadocs et des archives de distribution.

Utiliser Maven avec Spring Boot permet également de faciliter les déploiements en utilisant les plugins de déploiement de Maven qui automatise les tâches de build, de test et de déploiement.

En résumé, utiliser Maven avec Spring Boot permet de faciliter la gestion des dépendances, de générer des rapports, de faciliter les déploiements et d'automatiser les tâches de build, test et déploiement. Il est donc un choix logique pour les projets Spring Boot.

### Mysql driver

MySQL est un système de gestion de bases de données relationnelles populaire utilisé pour stocker et gérer les données d'une application. Le pilote MySQL (MySQL driver) est un logiciel qui permet à une application de se connecter à une base de données MySQL et d'exécuter des requêtes SQL.

Il existe plusieurs pilotes MySQL disponibles, mais l'un des plus couramment utilisés est le pilote MySQL Connector/J. Il est maintenu par Oracle, l'entreprise qui développe MySQL, et est compatible avec les versions récentes de MySQL. Il est également compatible avec les différents frameworks de développement, tels que Spring et Hibernate.

Il est open-source, ce qui permet aux développeurs de l'utiliser gratuitement, avec une grande flexibilité pour l'intégration avec les différentes plateformes de développement. Il est également compatible avec les différentes versions de Java, et est donc un choix populaire pour les projets Java qui utilisent MySQL.

En résumé, le pilote MySQL Connector/J est un choix populaire pour les développeurs car il est maintenu par Oracle, est open-source, est compatible avec les différents frameworks de développement, est compatible avec les différentes versions de Java et permet de se connecter et d'exécuter des requêtes sur une base de données MySQL.

### Bootstrap

Utiliser Bootstrap avec Spring Boot permet de combiner les avantages de ces deux technologies pour créer des applications web modernes et professionnelles.

Spring Boot permet de configurer rapidement et facilement un projet avec les fonctionnalités couramment utilisées dans les applications web, telles que la gestion de la sécurité, les contrôleurs REST, les vues et les services d'accès aux données.

Bootstrap, d'autre part, est un framework de design de sites web open-source qui permet de créer des sites web réactifs et adaptatifs en utilisant des composants prédéfinis. Il fournit des styles de base pour les éléments couramment utilisés dans les sites web tels que les boutons, les formulaires, les tables, les grilles, etc. Il permet également de faciliter le développement d'interfaces utilisateur en fournissant des composants prédéfinis pour les éléments couramment utilisés tels que les menus déroulants, les onglets, les modales, etc.

En utilisant Bootstrap avec Spring Boot, les développeurs peuvent créer des applications web qui ont une apparence professionnelle et moderne, tout en bénéficiant de la flexibilité et de la puissance des fonctionnalités offertes par Spring Boot. Cela permet également d'économiser du temps et des efforts en utilisant les composants prédéfinis de Bootstrap pour la création de l'interface utilisateur, plutôt que de les créer à partir de zéro.

En résumé, utiliser Bootstrap avec Spring Boot permet de combiner les avantages de ces deux technologies pour créer des applications web modernes et professionnelles, tout en économisant du temps et des efforts en utilisant les composants prédéfinis de Bootstrap pour la création de l'interface utilisateur.

### Spring Security

**Spring Security** est un framework de sécurité pour les applications Spring qui permet de protéger les applications web et les services REST en fournissant une authentification et une autorisation robustes. Il est intégré à Spring Boot, ce qui facilite la configuration et l'implémentation de la sécurité dans les applications Spring Boot.

Voici quelques raisons pour lesquelles Spring Security est un choix populaire pour les applications Spring Boot:

* Intégration facile: Spring Security est intégré à Spring Boot, ce qui facilite la configuration et l'implémentation de la sécurité dans les applications Spring Boot.
* Flexibilité: Spring Security est très flexible et permet de configurer facilement l'authentification et l'autorisation pour répondre aux besoins de l'application.
* Support pour différents types d'authentification: Spring Security prend en charge plusieurs types d'authentification, tels que l'authentification basée sur des noms d'utilisateur et des mots de passe, les jetons d'accès OAuth2, les jetons JWT, et l'authentification par jeton d'accès.
* Prise en charge des fonctionnalités de sécurité courantes: Spring Security inclut de nombreuses fonctionnalités courantes de sécurité, telles que la prise en charge des rôles, la gestion des sessions, la protection contre les attaques CSRF, et la prise en charge des politiques de mot de passe.
* Grande communauté de développeurs: Spring Security a une grande communauté de développeurs qui ajoute régulièrement de nouvelles fonctionnalités et corrige les bogues. Cela signifie que les problèmes sont généralement résolus rapidement et que de nouvelles fonctionnalités sont souvent ajoutées.

# Conclusion et perspectives

La conclusion découle des résultats obtenus au cours de la réalisation de ce projet. En résumé, ce projet a permis de mettre en place une application de gestion de rendez-vous pour un salon de coiffure. Les fonctionnalités mises en place permettent la gestion des rendez-vous, des clients, des salons et des services proposés par le salon.

*Perspectives*

Les perspectives pour ce projet pourraient être l'ajout de nouvelles fonctionnalités telles que la gestion des employés, la gestion des stocks de produits, la génération de rapports et l'intégration d'un système de paiement en ligne. Il est également possible d'améliorer l'expérience utilisateur en ajoutant des notifications pour les rendez-vous et en offrant une meilleure navigation.

Il est également possible d'étendre cette application à d'autres types de salons de beauté, tels que les salons de manucure, les salons de massage, etc. Enfin, il est possible de déployer cette application sur des plateformes mobiles pour permettre aux utilisateurs de gérer leurs rendez-vous depuis n'importe où.