ENSIAS

جامعة محمد الخامس بالرباط
Université Mohammed V de Rabat

Rapport du Projet Intégré Semsetre 5

Conception et réalisation d'une Application de gestion des rendez-vous chez les coiffeurs

---

Ingénierie du Web et Informatique Mobile- Option Web Intelligente

Réalisé par :
BOUTAHIRI Soufiane
BERRACHDI Mohamed

Encadré par :
M. JAMAL EL HACHIMI
Membre de jury :
Mm. Laila CHEIKHI

Année universitaire 2021-2022

Remerciement

Nous tenons tout d'abord, à remercier Dieu le tout puissant et miséricordieux, qui nous a donné la force et la patience d'accomplir ce travail.

Nous tenons, ensuite, à exprimer nos remerciements à tous ceux qui ont contribué par leurs aides précieuses, leurs conseils fructueux et leurs encouragements, et qui nous ont permis de réaliser ce projet dans la meilleure considération.

Nous voulons rendre un hommage particulier à notre encadrant monsieur **Jamal El Hachimi** pour son soutien et ses précieux conseils, aux membres du jury qui ont bien voulu nous honorer de leur présence afin d'évaluer notre travail.

Un grand merci à toutes les personnes qui nous ont soutenus de près ou de loin au cours de la réalisation de ce modeste travail.

Résumé

Aujourd'hui la réservation en ligne est devenue une nécessité. elle résoud divers problèmes liés à la complexité et à la démographie croissante.

Dans notre projet, nous allons résoudre un problème dans ce contexte : "**prendre un rendez-vous avec un coiffeur**".

Le problème d'attente dans les salons de coiffure s'avère un problème hebdomadaire ou mensuel pour certaines personnes.

Dans notre projet, nous utiliserons les technologies mobiles et web pour développer une solution qui offre à ses utilisateurs, une gestion et une prévisibilité sur les services chez un coiffeur.

Mots clés : JAVA, XML, MongoDB, NoSQL, Android, filtrage collaboratif, Spring boot, Azure, Cloud, Git, Rest API.

Abstract

Online booking has become a necessity in our generation, solving many problems due to complexity and increasing demographics.

In our project, we will solve a problem in this context : that is "making an appointment with a hairdresser".

Waiting in hairdressing salons becomes a weekly or monthly problem for some people.

In our project, we will use mobile and web technologies to develop a solution that offers to these users, management, and predictability of services at a hairdresser.

Keywords : JAVA, XML, MongoDB, Android, collaborative filtering, spring boot, Azur, cloud.

Sommaire

Table des figures

Introduction Générale

La prise de rendez-vous fait partie de notre quotidien, permet de résoudre des problèmes liés à la gestion de temps et ensuite avoir une prévision sur notre planning, puis donner une prévision claire à nos clients et ainsi garantir la satisfaction de nos clients.

Parmi les secteurs mal organisés, on note le secteur de coiffure, qui s'avère très important dans le monde artisanale. C'est un secteur indispensable dans notre vie de tous les jours, mais qui manque de mauvaise gestion, et mauvaise structuration. Par exemple, un client peut attendre **de 15 min jusqu'à 2 h** chez un salon de coiffeur.[1]

Dans cette optique de gestion des services et transformation digitale, nous allons essayer de résoudre ce problème. Nous allons développer une solution Web-Mobile basée sur les technologies Android, comme une plateforme mobile, dans le côté frontend. Le choix de la plateforme Android est due à plusieurs caractéristiques :

- Facilité d'utilisation
- Exploitation des ressources de la plateforme tels que (GPS, Camera, Senseurs ...)
- Le nombre énorme d'utilisateurs

Dans le côté backend, nous avons choisi les technologies Spring Boot, Azur cloud, MongoDB, pour des raisons de scalabilité de maintenabilité et facilité de développement.

Notre application permet d'identifier deux types d'utilisateurs : client et coiffeur. Le client peut consulter les services, la disponibilité et les informations du coiffeur, qui a son rôle pour ajouter des rendez-vous et des services, et gérer sa file d'attente.

Tout au long de ce rapport nous allons développer en détail notre projet. Le premier chapitre est consacré à l'étude du problème et détection du périmètre et des besoins.

Le deuxième chapitre sera dédié à l'analyse et la conception où nous allons traiter le diagramme de cas d'utilisation, le diagramme de séquence, le diagramme de classe.

Le troisième chapitre sera dédié à la partie de recommandation de notre application avec le filtrage collaboratif.

Dans le quatrième chapitre, nous allons entamer la réalisation, où nous allons parler des outils de travail et présenter graphiquement les pages de notre application mobile à travers des captures d'écran.

Chapitre 1

Contexte général de projet

Introduction

Dans ce chapitre, nous commencerons par la présentation de la problématique, ainsi qu'une délimitation du périmètre, et spécification des besoins. [2]

1.1 Contexte et définition du problème

Aujourd'hui, les services numériques sont inclus dans toutes les activités socio-économiques, teles que les services d'achat et vente, banque, paiement ..., cette pénétration de technologies nous permet de bien surmonter les problèmes dûs à l'explosion démographique.

De nos jours, en France, il y a plus de **85 700 salon de coiffeur**[3], avec un chiffre d'affaire par salon de **70 000 euro**. Cependant, ce secteur souffre d'absence de digitalisation qui génère beaucoup des problèmes tels que :

1- Moyen d'attente chez les coiffeurs (environ une semaine pour prendre un rendez-vous en France).
2- Manque de recommandation collaborative sur un coiffeur.
3- L'invisibilité sur la disponibilité d'un coiffeur.

1.2 objectifs

Nous voulons offrir un espace pour les coiffeurs et les clients, ce dernier peut trouver les coiffeurs les plus proches, avec une bonne qualité et recommandation des coiffeurs basée sur le filtrage collaboratif.

Les coiffeurs bénéficient d'un système de gestion de rendez-vous et file d'attentes, ainsi d'un espace où ils mettent en place leurs services et leurs réalisations.

1.3 Périmètre

Les acteurs principaux sont les coiffeurs et les clients, les entreprises de cosmétique et de produits de coiffure.

1.4 Etudes des besoins

Il s'agit des fonctionnalités du système. Ce sont les besoins spécifiant un comportement d'entrée / sortie du Système.
L'application mobile doit permettre à l'utilisateur d'affecter les opérations suivantes :

1.4.1 les besoins fonctionnels

Pour bien décrire les besoins fonctionnels de notre application, nous présenterons les services qui doivent être fournis aux différents utilisateurs.
Ces besoins se regroupent dans les diagrammes des cas d'utilisation.

Clinet :

1. Un système d'authentification
2. Lister les coiffeurs les plus proches.
3. Recommander un coiffeur
4. Demander un rendez-vous
5. Consulter le profil du coiffeur
6. Contacter un coiffeur
7. Consulter la disponibilité actuelle du coiffeur
8. Consulter la file d'attente.
9. Suivre un coiffeur.

Coiffeur :

1. Poster des travaux.(Coupe de cheveux ..)
2. Gérer la file d'attente
3. Indique la disponibilité
4. Advertisement

1.4.2 les besoins non fonctionnels

Les besoins non fonctionnels sont des besoins en matière de performance, de type de matériel ou de conception. Ces besoins peuvent concerner les contraintes d'implémentation.

L'extensibilité : L'application web devra être extensible. Autrement dit, nous pouvons avoir une possibilité d'ajouter des nouvelles fonctionnalités ou de modifier les fonctionnalités déjà existantes.

La sécurité : Les données des utilisateurs sont tenues à l'écart de tout risque externe et nous protégeons la vie privée des utilisateurs.

La simplicité et la performance : L'application web a un design simple facile à utiliser, et la performance du site poète dont le délai de réponse quelle que soit l'action de l'utilisateur.

1.5 Choix de la conduite de projet

Les méthodes agiles sont de plus en plus utilisées pour les projets de grande taille, car elles offrent une meilleure adaptabilité, visibilité et gestion des risques.

Elles peuvent tout aussi bien être utilisées pour les projets qui manquent de documentations détaillées. Le client peut alors suivre l'évolution du projet et l'adapter selon ses besoins.

En revanche, les méthodes classiques seront plus utilisées s'il y a une idée très précise du projet avec un cahier des charges et planning très détaillé où on peut anticiper tous les risques possibles.

Thème	Approche traditionnelle	Approche agile
Cycle de vie	En cascade ou en V, sans rétroaction possible, phases séquentielles.	Itératif et incrémental.
Planification	Prédictive, caractérisée par des plans plus ou moins détaillés sur la base d'un périmètre et d'exigences définies et stables au début du projet.	Adaptative avec plusieurs niveaux de planification (macro- et microplanification) avec ajustements si nécessaires au fil de l'eau en fonction des changements survenus.
Documentation	Produite en quantité importante comme support de communication, de validation et de contractualisation.	Réduite au strict nécessaire au profit d'incréments fonctionnels opérationnels pour obtenir le feedback du client.
Équipe	Une équipe avec des ressources spécialisées, dirigées par un chef de projet.	Une équipe responsabilisée où l'initiative et la communication sont privilégiées, soutenue par le chef de projet.
Qualité	Contrôle qualité à la fin du cycle de développement. Le client découvre le produit fini.	Un contrôle qualité précoce et permanent, au niveau du produit et du processus. Le client visualise les résultats tôt et fréquemment.
Changement	Résistance voire opposition au changement. Processus lourds de gestion des changements acceptés.	Accueil favorable au changement inéluctable, intégré dans le processus.
Suivi de l'avancement	Mesure de la conformité aux plans initiaux. Analyse des écarts.	Un seul indicateur d'avancement : le nombre de fonctionnalités implémentées et le travail restant à faire.
Gestion des risques	Processus distinct, rigoureux, de gestion des risques.	Gestion des risques intégrée dans le processus global, avec responsabilisation de chacun dans l'identification et la résolution des risques. Pilotage par les risques.
Mesure du succès	Respect des engagements initiaux en termes de coûts, de budget et de niveau de qualité.	Satisfaction client par la livraison de valeur ajoutée.

FIGURE 1.1 – Comparaison entre méthodes agile et classique

On peut donc conclure que la méthode agile est la plus convenable dans notre cas puisque c'est un projet flexible et très grand.

1.5.1 Choix de la méthode agile

Scrum est une méthode agile pour la gestion de projets conçue pour améliorer grandement la productivité des équipes auparavant paralysées par des méthodologies plus lourdes. Cette méthode est simple à mettre en place et permet d'augmenter une visibilité sur le projet.

Son utilisation est initialement prévue pour la gestion de projets de développement, elle a par exemple été utilisée avec succès pour englober Extreme Programming et d'autres méthodes de développement. Cependant, elle peut théoriquement s'appliquer à n'importe quel contexte où un groupe de personnes ont besoin de travailler ensemble pour atteindre un but commun.

L'approche SCRUM suit les principes de la méthodologie Agile, c'est-à-dire l'implication et la participation active du client tout au long du projet. C'est une méthode de gestion de projet ou plutôt un Framework de management de projet qui a pour objectif d'améliorer la productivité de l'équipe et qui se compose de plusieurs éléments fondamentaux :
des rôles, des événements et des artefacts.

FIGURE 1.2 – Valeurs de SCRUM

Le principe de base de Scrum est de focaliser l'équipe de façon itérative sur un ensemble de fonctionnalités à réaliser constituant les Sprints. Chaque Sprint possède un but à atteindre, défini par le Directeur de produit, à partir duquel sont choisies les fonctionnalités à implémenter.

Un sprint aboutit toujours à la livraison d'un produit partiel fonctionnel. Pendant ce temps, le Scrum Master a la charge de réduire au maximum les perturbations extérieures et de résoudre les problèmes non techniques de l'équipe.

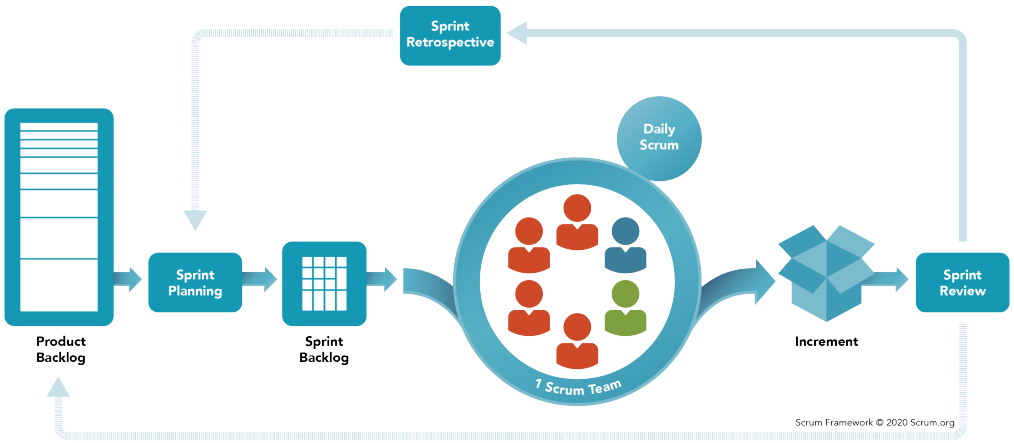

FIGURE 1.3 – Framework SCRUM

Pour l'application de ce framework nous avons utilisé la plateforme JERA, dans laquelle nous avons créé les PBIs, nous avons utilisé "Story Point" comme mesure, et Planning Poker comme méthode d'estimation.

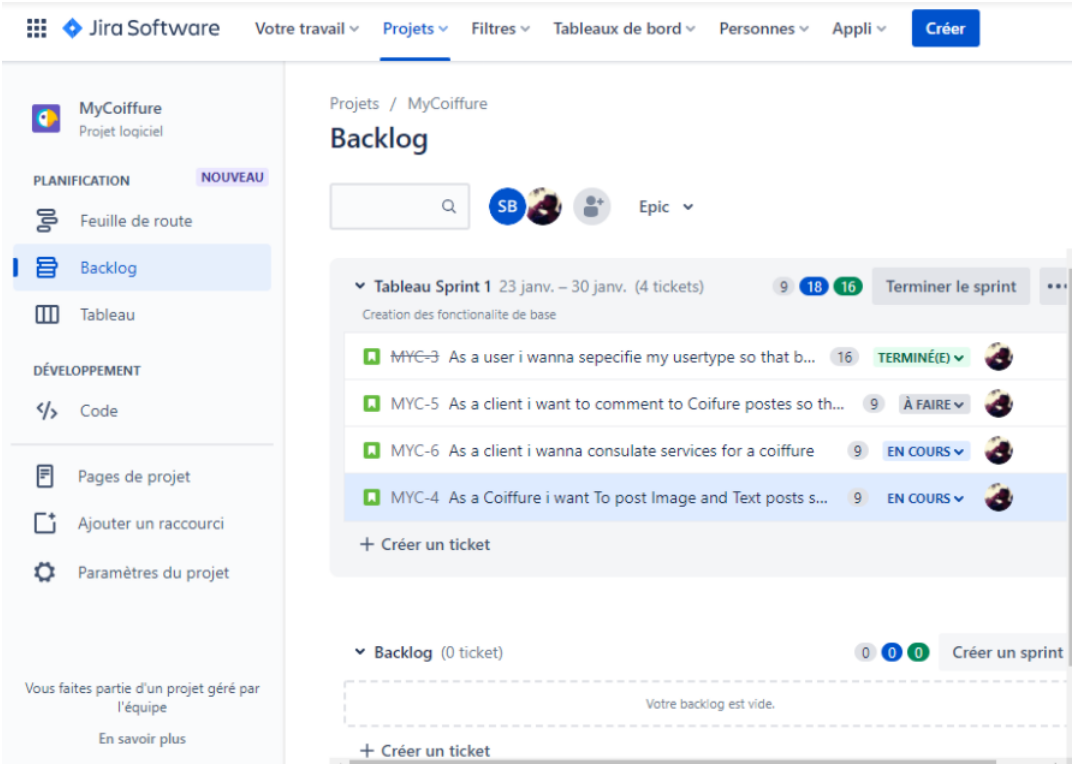

FIGURE 1.4 – Plateforme Jir

1.6 Planification du projet

Avant de se lancer dans la réalisation du projet, il est nécessaire de prendre le temps de découper celui-ci en tâche, afin de planifier l'exécution de ces tâches et le temps à allouer à chacune. La figure ci-dessous résume la planification de notre projet.

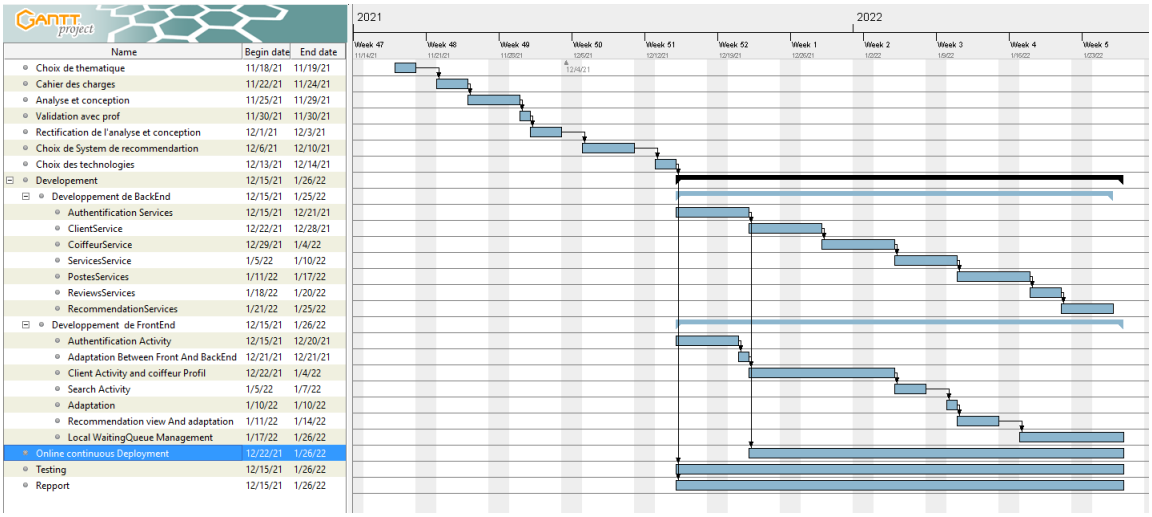

FIGURE 1.5 – Diagramme de GANTT de projet.

Conclusion

Au cours de ce chapitre, nous avons abordé l'étude préliminaire du projet, les besoins fonctionnels et non fonctionnels ainsi que les acteurs de l'application, les applications et le framework que nous avons utilisés dans la gestion de projet.
Le chapitre suivant sera dédié à la phase d'analyse et conception.

Chapitre 2

Analyse et conception du projet

Introduction

La phase d'analyse et de la conception présente une étape essentielle dans le cycle de développement d'un projet. En fait, cette étape vise à clarifier les besoins de notre projet et de définir les structures et les modèles à suivre lors de la phase de l'implémentation de l'application. Cette phase sera basée sur l'UML.[4]

2.1 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation nous permet de montrer les interactions existantes entre le système et les entités externes au système, ces entités externes sont désignées comme des acteurs.

Notre système possède trois acteurs principaux :

Le client -Acteur interne

1. Consulter la liste des coiffeurs.
2. Demander des rendez-vous avec des coiffeurs.
3. Évaluer et suivre des coiffeurs.

Le coiffeur -Acteur interne

1. Permet la gestion de son file d'attente.
2. Lancer une campagne publicitaire.
3. Créer et partager des posts qui présente leurs travaux.

Le système -Acteur externe

1. Il prend en charge la recommandation des coiffeurs aux clients et la gestion des advertisement des coiffeurs

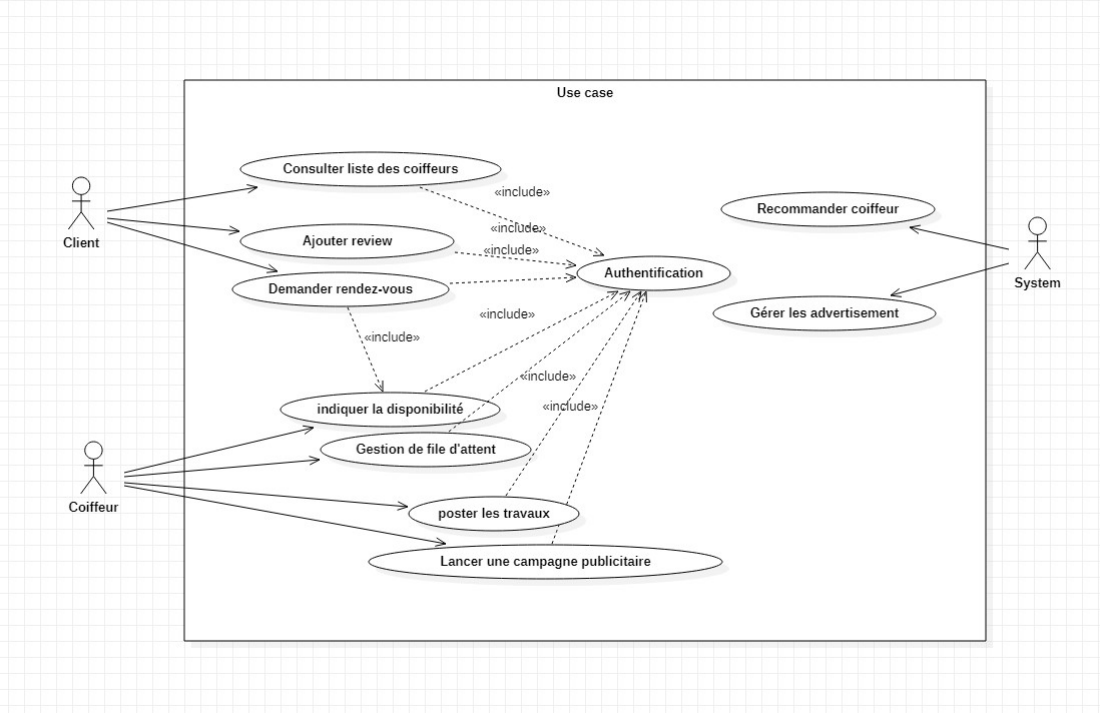

FIGURE 2.1 – Diagramme de cas d'utilisations

2.2 Diagrammes de séquence

Le diagramme de séquence fait partie des diagrammes qui décrivent l'aspect dynamique du système, les interactions entre les objets et les acteurs ou entre les objets eux-mêmes, en mettant l'accent sur la chronologie des messages échangés.

Dans cette partie on va élaborer deux diagrammes de séquence :

- Diagramme de séquence pour **"Ajouter un rendez-vous"**
- Diagramme de séquence pour **"Publier un post"**

2.2.1 Demander un rendez-vous

Description : Ce cas d'utilisation permet au client de prendre un rendez-vous avec le coiffeur.

Acteur : Le client

Pré-condition : Le client doit posséder un compte sur notre application

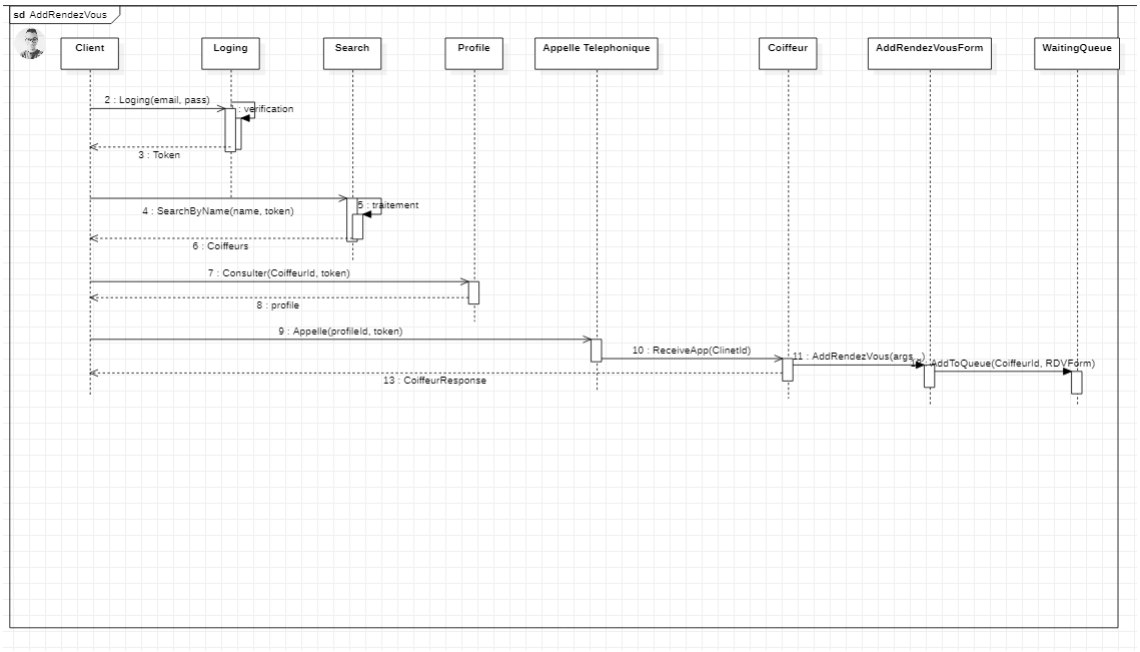

FIGURE 2.2 – Diagramme De séquence : Demander un rendez-vous

2.2.2 Publier un post

Description : Ce cas d'utilisation permet au coiffeur de postuler leur travail dans notre réseaux (My Coiffeur)

Acteur : Le coiffeur

Pré-condition : Le coiffeur doit compléter les informations de son profile

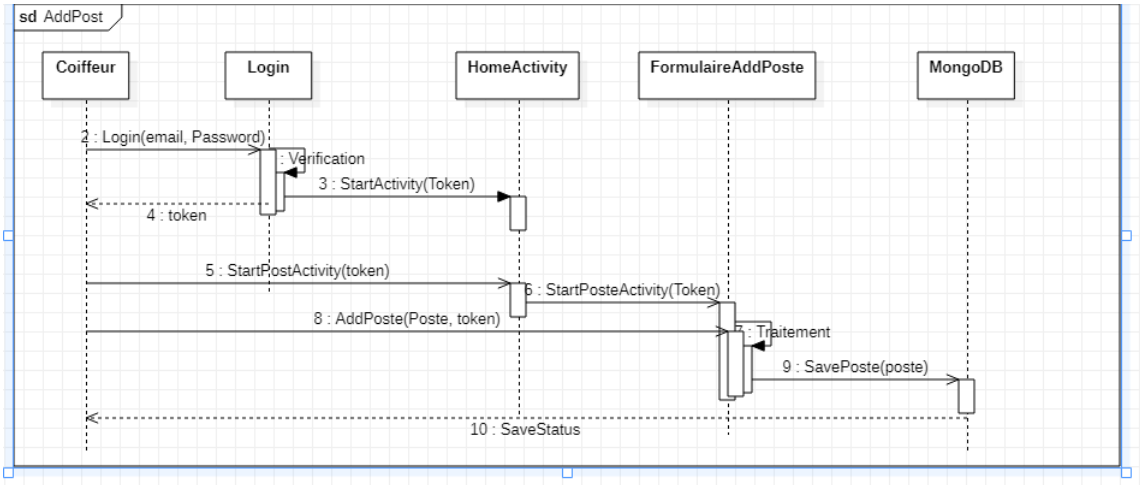

FIGURE 2.3 – Diagramme de séquence : Ajout de poste

2.3 Diagramme de classe

Le diagramme de classe est une modélisation statique du système en termes de classes et les relations entre ces classes. Son intérêt réside dans la modélisation des entités du système d'information.

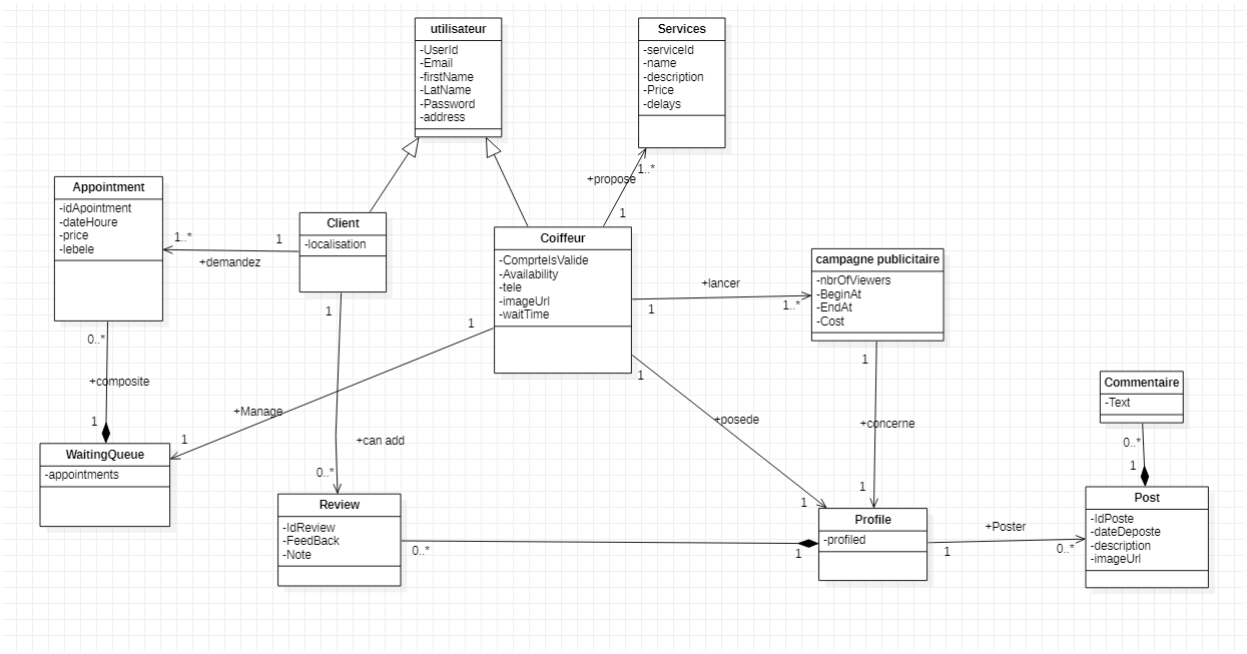

FIGURE 2.4 – Diagramme des classes

conclusion

Dans cette partie nous avons élaboré les différents diagrammes tels que, le cas d'utilisation, de séquence et le digramme de classe. Selon le langage de modélisation unifié UML ,cette partie est essentielle pour n'importe quel projet avant de passer à la phase de développement.

Chapitre 3

le système de recommandation

Introduction

Dans ce chapitre, nous présentons de façon détaillée le domaine des **systèmes de recommandation**, en mettant l'accent sur **les concepts de base et les techniques du filtrage existantes**, et enfin nous allons utiliser **la corrélation** comme une **métrique** de similarité entre les coiffeurs.

3.1 Définition des systèmes de recommandation

Les systèmes de recommandation ont été définis de plusieurs façons. La définition la plus populaire et la plus générale que nous citons ici est celle de Robin Burke [Burke, 2002] que nous avons traduite ainsi :

Système de recommandation : *Système capable de fournir des recommandations personnalisées ou permettant de guider l'utilisateur vers des ressources intéressantes ou utiles au sein d'un espace de données important.*

L'objectif d'un système de recommandation est de fournir des objets (logement, vêtements,...) similaire a ses préférences, il permettra de réduire le temps que l'utilisateur met pour chercher d'autres objets similaires et intéressants pour lui, et aussi pour trouver des objets qui satisfont leur besoins.

Il existe plusieurs techniques de recommandation. La plupart de ces systèmes sont issus du machine learning et statistique.
On cite par exemple :

Filtrage collaboratif Basé sur la Mémoire :

1. Fondé sur les objets (Item Based)
2. Fondé sur les utilisateur (User Based)

Filtrage collaboratif Basé sur les modèles :

1. K-Nearest Neighbour
2. Singular Value Decomposition
3. Non-Negative Matrix Factorization

3.2 histoire des systèmes de recommandation

Les premiers systèmes de recommandation se réduisent aux systèmes de filtrage collaboratif. Ils remontent au début des années 1990s, cette période à laquelle ils sont reconnus comme étant un domaine de recherche indépendant.

Parmi les systèmes pionniers dans ce domaine, nous citons à titre d'exemple les systèmes :

- Tapestry (Goldberg, Nichols, Oki, Terry, 1992).
- GroupLens/NetPerceptions (Resnick, Iacovou.
- Suchak, Bergstrom, Riedl, 1994).
- Ringo/Firefly (Shardanand and Maes, 1995).

Les racines des systèmes de recommandation remontent aux travaux étendus dans les sciences cognitives, la théorie d'approximation, la recherche documentaire, la théorie de la prévoyance et ont également des liens avec la science de la gestion et le marketing, dans la modélisation des choix du consommateur (Adomavicius Tuzhilin, 2005).

Après on a la nouvelle génération des système de recommandation qu'est les système hybride, cette génération est une combinaison entre filtrage de contenu et filtrage collaboratif.

3.3 Choix de l'algorithme

Dans notre application nous avons choisi d'utiliser l'approche basée sur la similarité entre les objets, pour les raisons suivantes :

1. Facilité d'implémentation
2. Consommation des ressources (CPU, RAM).
3. Ainsi que les approches objet-objet sont beaucoup plus rapides que les approches utilisateur-utilisateur.

Ensuite, les profils des utilisateurs changent rapidement et l'ensemble du modèle du système doit être recalculé, alors que les évaluations moyennes des objets ne changent pas aussi rapidement, ce qui conduit à des distributions d'évaluations plus stables dans le modèle, et le modèle n'a donc pas besoin d'être reconstruit aussi souvent.

3.4 Analyse et description des données pour la partie recommandation

3.4.1 introduction

Le KDD (Knowledge data discovery), fait référence au vaste processus de recherche de connaissances dans les données et met l'accent sur l'application de haut niveau de méthodes d'exploration de données particulières. Il intéresse les chercheurs dans les pays de l'apprentissage automatique, de la reconnaissance de formes,des

bases de données, des statistiques, de l'intelligence artificielle, de l'acquisition de connaissances pour des systèmes experts et de la visualisation de données..

L'objectif unificateur du processus KDD est d'extraire des connaissances des données dans le contexte de grandes bases de données.

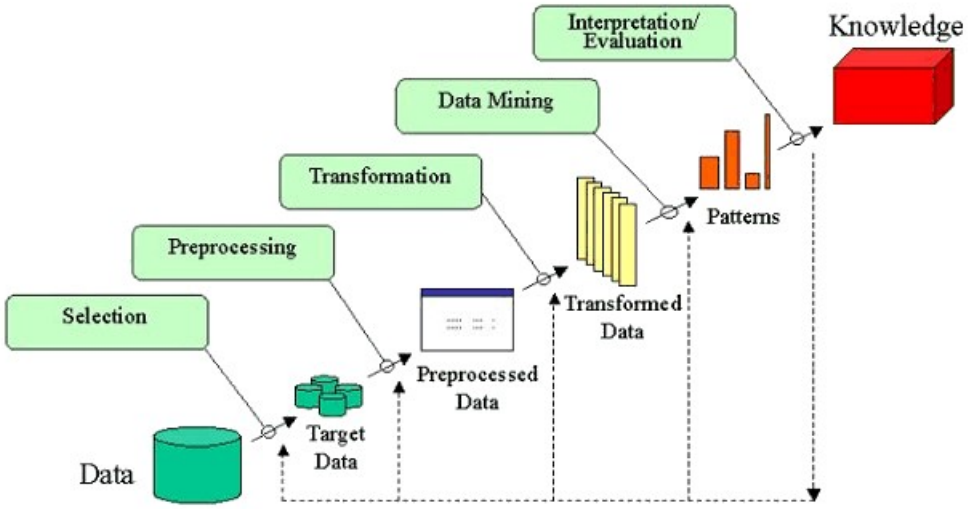

FIGURE 3.1 – les phases de KDD

3.4.2 Data integration and data cleaning

Dans cette phase de KDD, nous avons importé les données de notre base des données MongoDb à partir de la collection "Reviews".

	clientId	profileId	note
0	Q3v7c0KQrZoHl1y5pFu9	WqcyXAucJXgpjBKx6il7	5
1	0sSuylHZoOz5ndDvlisF	WqcyXAucJXgpjBKx6il7	5
2	0sSuylHZoOz5	WqcyXAucJXg1	0
3	0sSuylHZoOz6	WqcyXAucJXg2	4
4	0sSuylHZoOz7	WqcyXAucJXgpjBKx6il7	2
...	...	...	...
100	0sSuylHZoOz19	WqcyXAucJXgpjBKx6il7	4
101	0sSuylHZoOz5ndDvlisF	WqcyXAucJXg1	3
102	0sSuylHZoOz5	WqcyXAucJXg2	0
103	0sSuylHZoOz6	WqcyXAucJXgpjBKx6il7	3
104	0sSuylHZoOz98	WqcyXAucJXg1	4

105 rows × 3 columns

FIGURE 3.2 – Tables des Review sous format dataframe

Notre table de reviews contient 100 reviews de 3 coiffeurs.

Après l'obtention de la table des reviews, nous la transformons à l'aide d'opérations de pivotement, qui nous permettra à transformer cette table, en une table intermédiaire pour appliquer le mesure voulue.

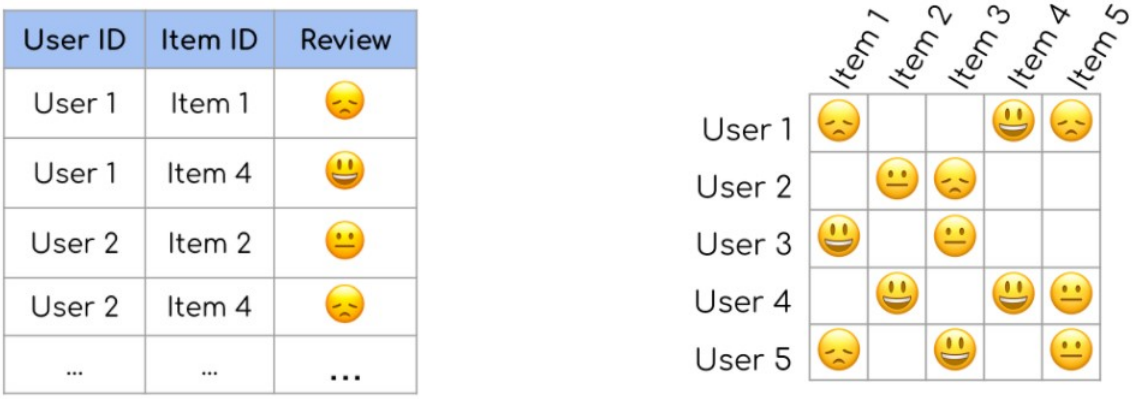

FIGURE 3.3 – Image illustrative d'opération de pivotement.

profileId clientId	WqcyXAucJXg1	WqcyXAucJXg2	WqcyXAucJXgpjBKx6il7
0sSuylHZoOz10	NaN	NaN	0.0
0sSuylHZoOz100	NaN	NaN	2.0
0sSuylHZoOz101	3.0	NaN	NaN
0sSuylHZoOz102	NaN	3.0	NaN
0sSuylHZoOz103	NaN	NaN	2.0
...	...	...	...
0sSuylHZoOz96	NaN	1.0	NaN
0sSuylHZoOz97	NaN	NaN	2.0
0sSuylHZoOz98	1.5	NaN	NaN
0sSuylHZoOz99	NaN	4.0	NaN
Q3v7c0KQrZoHl1y5pFu9	NaN	NaN	5.0

101 rows × 3 columns

FIGURE 3.4 – Table de reviews après pivotement.

Après l'opération de pivotement, nous remplirons les valeurs nulles de la colonne avec les évaluations moyennes du coiffeur correspondant, afin de pouvoir effectuer la corrélation de Pearson. Dans notre cas, nous supposons des évaluations moyennes pour un client qui n'a pas de coiffeur évalué.

3.4.3 Data mining

Dans cette phase nous allons calculer la matrice de similitude, à l'aide de la corrélation :

$$p_{xy} = \frac{Cov(x, y)}{\sigma_x * \sigma_y}$$

profileId	WqcyXAucJXg1	WqcyXAucJXg2	WqcyXAucJXgpjBKx6il7
profileId			
WqcyXAucJXg1	1.000000	-0.000997	0.003088
WqcyXAucJXg2	-0.000997	1.000000	-0.009997
WqcyXAucJXgpjBKx6il7	0.003088	-0.009997	1.000000

FIGURE 3.5 – Table de corrélation entre les coiffeurs.

Après avoir analysé ce tableau, nous pouvons déduire que le coiffure dans la première colonne, peut corréler avec le coiffeur la troisième colonne, en revanche, il ne peut pas corréler avec la coiffure deuxième colonne.
Ainsi, nous pouvons proposer aux clients le coiffeur dans la colonne une, les services du coiffeur dans la troisième colonne.

3.5 Conclusion

Dans ce chapitre nous avons utilisé une méthode de recommandation afin de recommander des coiffeurs à nos utilisateurs. Cette technique s'appelle le filtrage collaboratif basé sur les objets.

Chapitre 4

Mise en œuvre du Projet

Introduction

Cette partie du rapport est basée sur les trois chapitres précédents et se concentre principalement sur les outils et l'architecture que nous avons utilisés pour créer les fonctionnalités de notre projet.
Nous allons montrer ces fonctionnalités à travers des captures d'écran et des commentaires.

4.1 Architecture du projet

La conception de l'architecture est une phase particulièrement importante du développement d'un logiciel de n'importe quelle sorte. Elle conditionne sa stabilité, son efficacité et sa pérennité. Au contraire, certaines applications peuvent présenter des faiblesses dues à une architecture mal pensée, pas ou plus adaptée au contexte.

Notre application mobile MyCoiffeur est divisée en deux couches globalement séparées ; une couche qui tourne sur la machine du client (Front-end) et l'autre qui fonctionne sur le serveur (Back-end), et chacune de ces couches dispose de sa propre architecture.

4.1.1 Architecture du front-end

Dans les premières étapes du développement d'Android, les apprenants écrivent le code de manière à créer finalement une classe MainActivity qui englobe toute la logique de mise en œuvre de l'application mobile . Cette approche de développement d'applications fait que l'activité Android est fortement liée à l'interface utilisateur et au processus de traitement des données de l'application. En outre, elle engendre des difficultés de maintenance et de mise à l'échelle de ces applications mobiles.

Pour éviter ces problèmes de maintenabilité, de lisibilité, et d'évolutivité, les développeurs préfèrent définir des couches de code bien séparées. En appliquant des modèles d'architecture logicielle, on peut organiser le code de l'application pour séparer les préoccupations.

L'architecture MVP (Model - View - Presenter) sur laquelle le front-end de notre application mobile est basé, est l'une des modèles d'architecture les plus populaires et valable pour l'organisation de développement des application mobile Android.

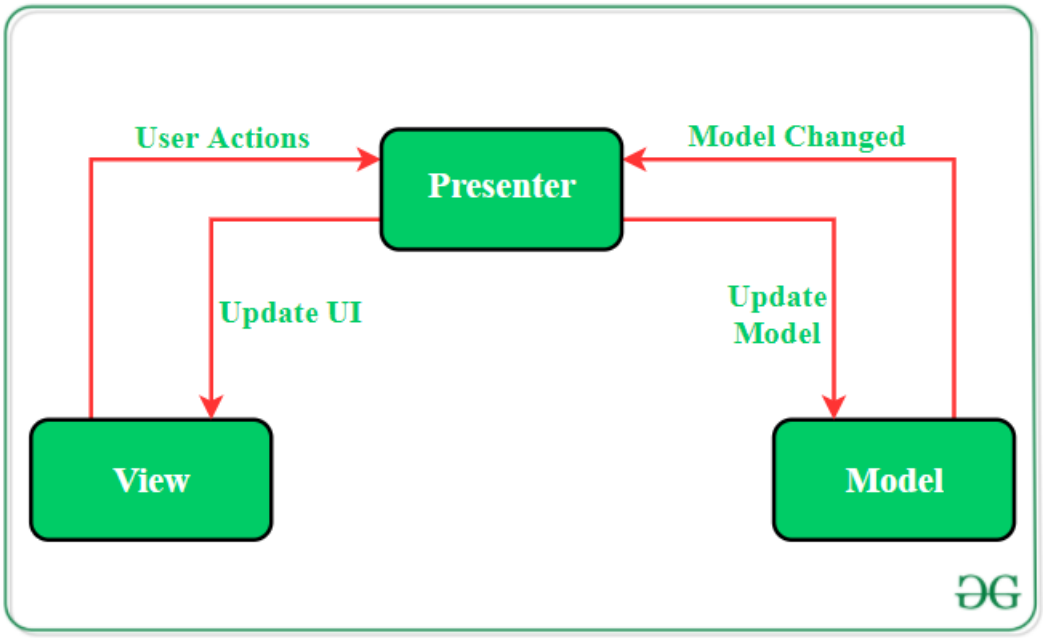

FIGURE 4.1 – Architecture logiciel MVP

4.1.2 Architecture du back-end

REST est un style d'architecture logicielle qui définit l'ensemble de règles à utiliser pour créer des services Web. Les services Web qui suivent le style architectural REST sont appelés services Web RESTful. Il permet aux systèmes demandeurs d'accéder aux ressources Web et de les manipuler en utilisant un ensemble de règles uniformes et prédéfinies. L'interaction dans les systèmes basés sur REST se produit via le protocole de transfert hypertexte (HTTP) d'Internet.

Un système Restful se compose d'un :

client qui demande les ressources.

serveur qui a les ressources.

Il est important de créer une API REST conformément aux normes de l'industrie, ce qui facilite le développement et augmente l'adoption par les clients.

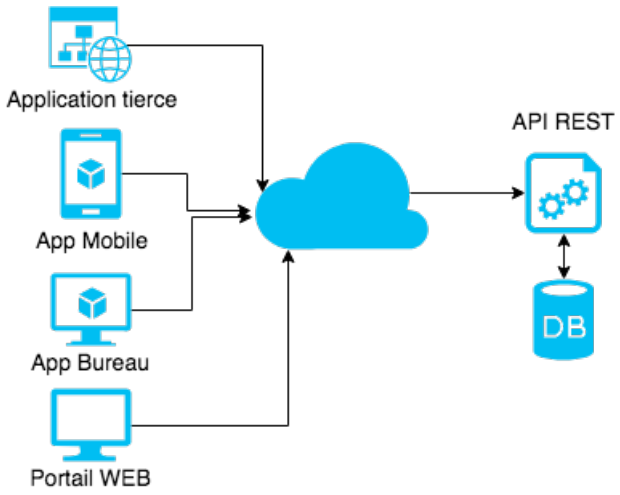

FIGURE 4.2 – REST API Schemal MVC

4.2 Les outils de développement

4.2.1 Front-end

Technologie XML

XML signifie Extensible Markup Language. Tout comme HTML (ou HyperText Markup Language), XML est également un langage de balisage.
Il a été créé comme un moyen standard d'encoder des données dans des applications Internet. Cependant, contrairement à HTML, XML est sensible à la casse, nécessite que chaque balise soit correctement fermée et préserve les espaces.
Les balises XML ne sont pas prédéfinies dans XML. Nous devons définir nos propres Tags. Xml en tant que tel est bien lisible à la fois par l'homme et par la machine. De plus, il est évolutif et simple à développer.
Dans Android, nous utilisons XML pour concevoir nos mises en page car XML est un langage léger, il n'alourdit donc pas notre mise en page.Dans Android, l'utilisation de XML a plusieurs objectifs, chaque objectif nécessite un type spécifique de fichiers xml :

- - Définissez les interfaces graphique de l'application mobile,le dossier **Layout XML** qui contiennent tous les éléments (vues) de notre application.
- - Définissez tous les composants de l'Application dans le fichier **Manifest.xml**
- - etc..

Language Java [3] [1]

Java est le langage de programmation le plus populaire dans le monde, c'est un langage idéal pour développer une application mobile native pour Android. Il présente l'avantage d'être utilisé par des milliers de développeurs à travers le monde, partageant leurs connaissances et retours d'expérience. Java compile le code source utilisé pour développer l'application en code binaire que les machines mobiles sont capables de lire.

Sélectionner Java pour développer une application mobile est un choix stratégique. En effet, c'est un langage qui permet de coder en native pour Android, le système d'exploitation le plus populaire, détenant plus de 80 % du marché. Pour coder une application mobile en Java, utilisez Android Studio, Il s'agit d'un environnement de développement multiplateforme complet et riche.

4.2.2 Back-end

REST API avec Spring Boot framework

Spring Boot est un framework base sur Java et l'ecosysteme Spring , développé par l'équipe de Pivotal, conçu pour simplifier le démarrage et le développement de nouvelles applications Spring. Le framework propose une approche méthodique de la configuration, qui empêche les développeurs de redéfinir la même configuration à plusieurs endroits dans le code. En ce sens, Boot vise à être un acteur majeur dans le domaine croissant du développement rapide d'applications.[5]
Spring Boot peut s'expliquer simplement par l'illustration ci-dessous :

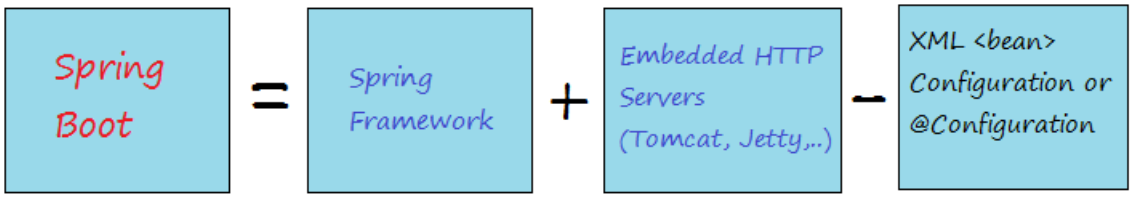

FIGURE 4.3 – illustration du framework Spring Boot

Les avantages de Spring Boot
— Optimisation de la gestion des dépendances.
— L'autoconfiguration.
— La gestion des propriétés.
— Le monitoring et la gestion du programme.
— Le déploiement.
Swagger

Swagger est un projet open source lancé par une Startup en 2010. L'objectif est de mettre en place un Framework qui va permettre aux développeurs de documenter et de designer des API, tout en maintenant une synchronisation avec le code.

Parallèlement au développement du Framework, une spécification Swagger a été mise en place pour définir le standard à respecter pour designer et documenter son API. Le projet a attiré l'attention de nombreux développeurs, et est devenu la technologie la plus populaire pour designer et décrire les API RESTful.
MongoDB et Atlas Cloud

MongoDB Atlas est un service de base de données multi- cloud conçu par les mêmes personnes qui ont créé

MongoDB. Atlas simplifie le déploiement et la gestion de des bases de données NoSQL il offre la polyvalence dont nous avons besoin pour créer des applications globales résilientes et performantes sur les fournisseurs de cloud de notre choix comme AWS, Google Cloud, Azure.

Microsoft Azure

Microsoft Azure regroupe différents services Cloud. Comme chez les concurrents(AWS, IBM Cloud, Google Cloud..), on retrouve notamment un service de stockage, des machines virtuelles, et des réseaux de diffusion de contenu.
Azure propose également des services exploitant les technologies propriétaires de Microsoft comme Active Directory ou SQL Server.
Nous avons utilisé Azurepour déployer notre REST API de notre application mobile.

Python

Python est un langage de programmation open source riche, interprété et facile à apprendre souvent utilisé dans le domaine du Machine Learning, du Big Data et de la Data Science. Pour construire notre système de recommandation qui base sur l'algorithme filtrage collaborative on a utilisé de nombreux bibliothèques de python tel que : La librairie Pandas : Fournit des structures de données puissantes et simples à utiliser, ainsi que les moyens d'opérer rapidement des opérations sur ces structures. NumPy est une bibliothèque open source pour langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux.

Postman

Postman est un logiciel qui se focalise sur les tests des APIs. Il permet d'envoyer tous sort de requête http tel que GET, POST, PUT, UPDATE,... Cette logiciel devenu très populaire pour tester les Micro services et des REST APIs, notamment grâce à

sa simplicité et ses fonctionnalités très spécialisées.

Git

Git est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Thorvald, cet outil ne permet de créer des dépôts locaux et de gérer les versions de nos fichiers.

Github action - CI pipeline

4.3 Les interfaces de notre application mobile

Cette partie du rapport sera dédiée à la présentation du fonctionnement de l'application mobile MyCoiffeur à travers des captures d'écran et des commentaires

4.3.1 Les interfaces utilisateur

-Création d'un compte utilisateur de type Client. L'utilisateur détermine le type de compte à créer, soit Coiffeur soit Client.

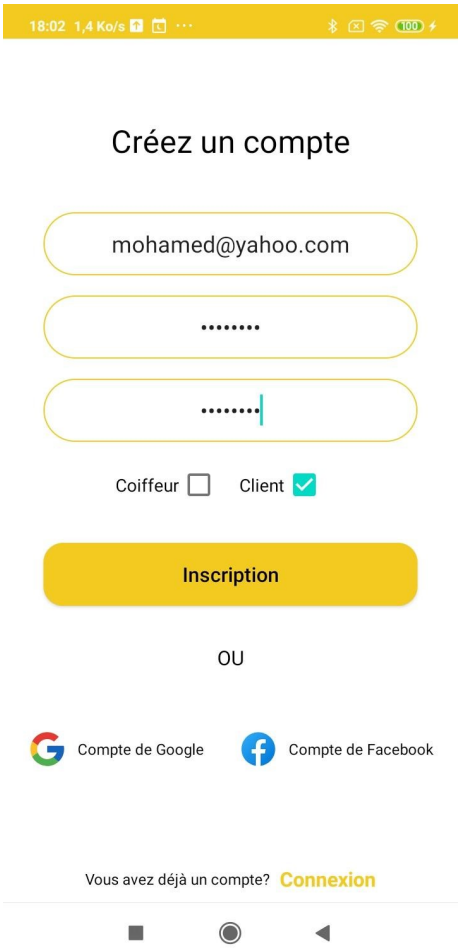

FIGURE 4.4 – sign up du client

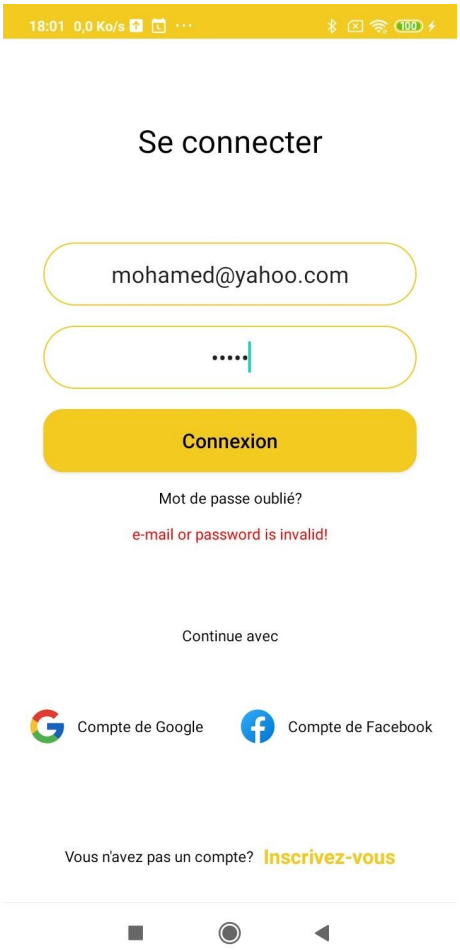

FIGURE 4.5 – logIn du client

FIGURE 4.6 – Interfaces de Connexion

-Lors de la première connexion à l'application MyCoiffeur, il est demandé d'ajouter quelques informations. Dans le cas où la connexion est faite par un coiffeur, l'application demande des informations sur les services offerts par l'utilisateur.

-Une fois que le client a réussi les étapes précédentes, l'interface principale du client s'affiche.

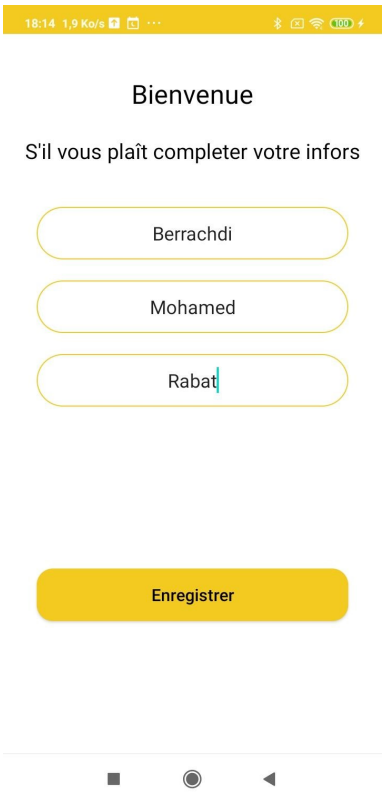

FIGURE 4.7 – Compléter le profil pour le cas du client

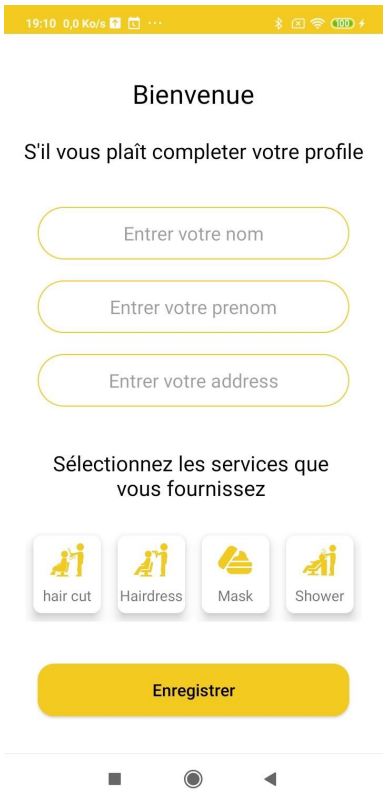

FIGURE 4.8 – Compléter le profil pour le cas du clientdu coiffeur

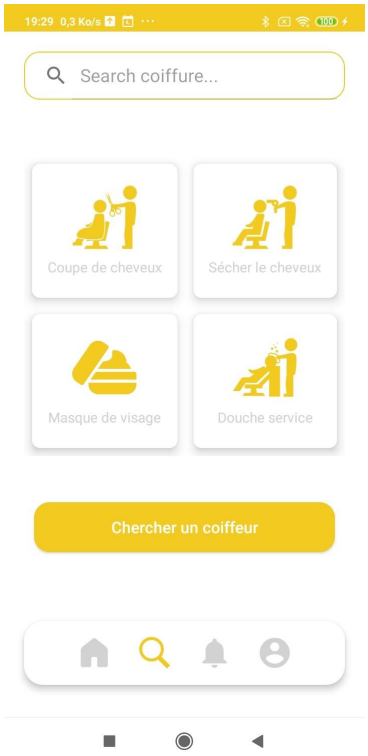

FIGURE 4.9 – L'interface principale du client

-Le client peut rechercher des coiffeurs en utilisant deux approches différentes : sur la base du nom d'utilisateur du coiffeur,dans ce cas là, la liste des coiffeurs sera classée en fonction de l'emplacement du coiffeur par rapport au client, L'évaluation du coiffeur, L'autre approche est basée sur les services

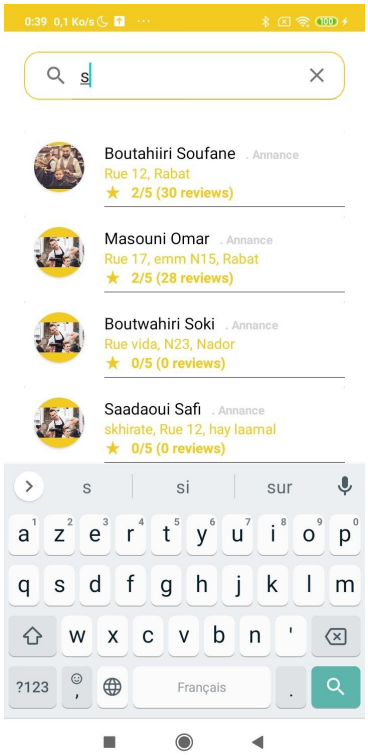

FIGURE 4.10 – Résultats de la recherche

-le client a le droit de consulter le profil de n'importe quel coiffeur. Voici un exemple du profil du coiffeur "Boutahiri Soufane". le client peut contacter le coiffeur sur son numéro de téléphone pour prendre un rendez-vous, pour voir la liste des postes publiés par le coiffeur etc.

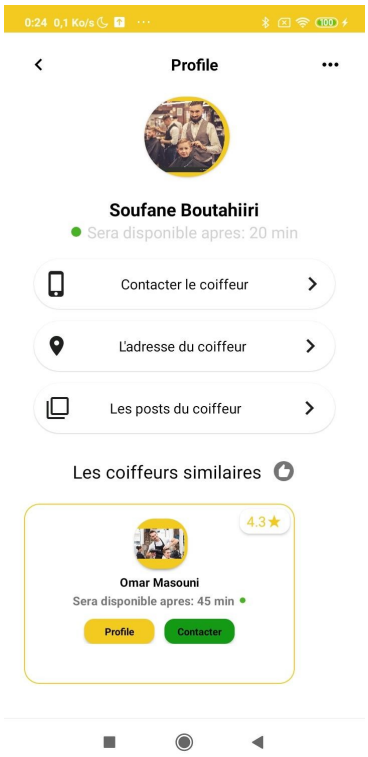

FIGURE 4.11 – profile du coiffeur et la liste des coiffeurs recommande

-en dessous de la liste des recommandations, vous trouverez les avis des clients sur le coiffeur.

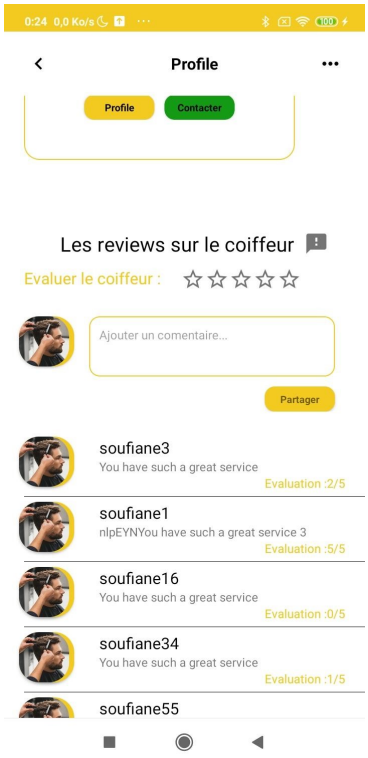

FIGURE 4.12 – les avis des clients

-d'autre part le coiffeur visualise son profil sur cette forme

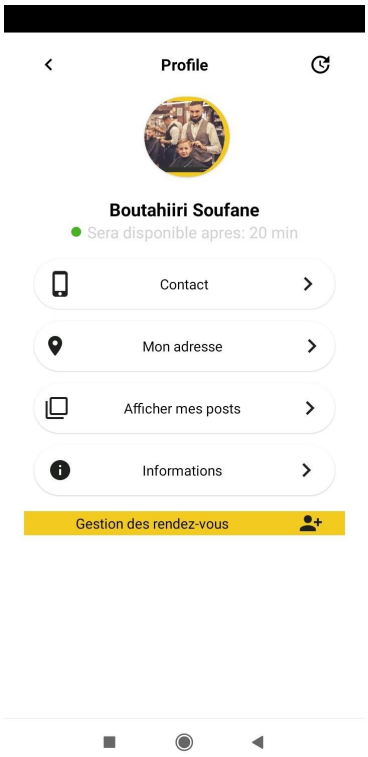

FIGURE 4.13 – Profile du coiffeur

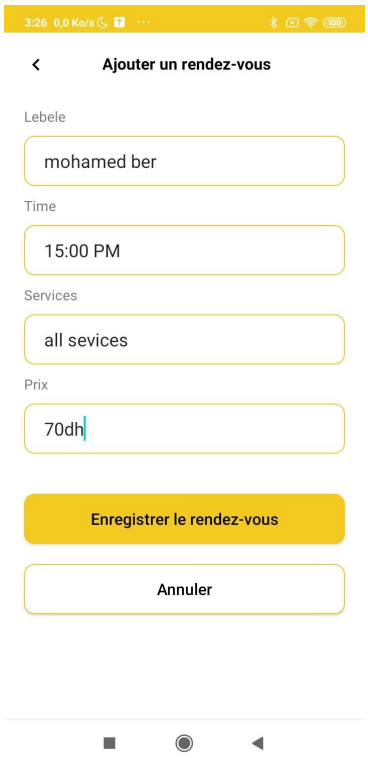

FIGURE 4.14 – Ajouter un rendez vous

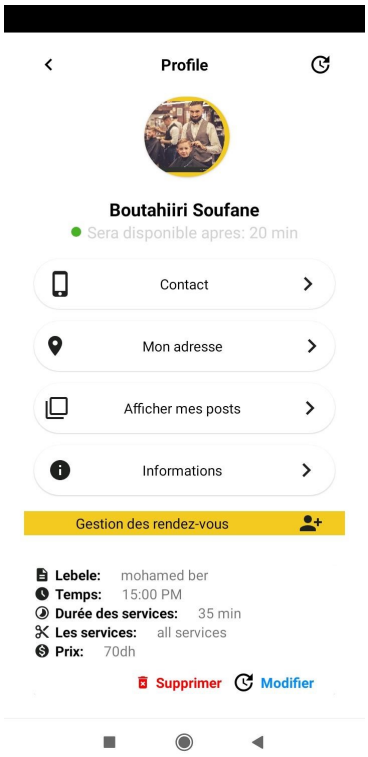

FIGURE 4.15 – Liste des rendez-vous

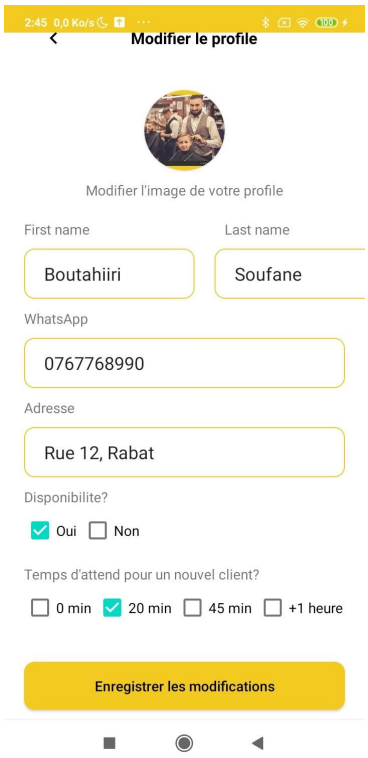

FIGURE 4.16 – Modifier les informations

Conclusion

Dans ce chapitre nous avons détaillé les technologies utilisées pour la réalisation de notre projet ainsi que les fonctionnalités de base de l'application à travers un ensemble de captures d'écran. .

Conclusion général

Tout au long de ce rapport, nous avons présenté les différentes étapes de la réalisation de notre projet du cinquième semestre qui consiste sur la mise en place d'une "application mobile pour la gestion des rendez-vous chez les coiffeurs".
Nous avons illustré dans ce rapport les étapes de notre mission commençant par la présentation de la problématique et l'objectif de notre projet. Nous avons utilisé la méthode agile SCRUM dans la conduite de notre projet.
Puis, pour l'analyse et la conception de ce projet, nous avons utilisé le langage UML, ce qui a permis de mener correctement la tâche d'analyse, ses besoins et la tâche de conception. Les scénarios sont aussi détaillés afin d'expliquer toutes les tâches faites par notre application.
Pour la partie de système des recommandations, nous avons décrit en détails les approches existantes pour développer un système de recommandation, ainsi que les raisons de notre choix de système "Item-Based". Nous avons utilisé python avec la bibliothèque pandas pour développer notre système. Ainsi, la phase de développement et mise en œuvre nous a donné l'occasion de travailler avec plusieurs technologies et faire le lien entre nos connaissances académiques, base de données, data science et implémenter cette application.

Ce projet nous a permis d'approfondir nos connaissances théoriques et pratiques, de stimuler un esprit d'initiative et de créativité, et notamment dans le domaine de développement des applications mobiles basées sur architecture RESTful.

Nous avons pu réaliser le maximum de notre cahier de charges tout en suivant la méthodologie Scrum et en se basant sur plusieurs technologies et frameworks : Spring Bot, MongoDb, Android.

Ce projet fédérateur nous a appris de suivre la méthode efficace pour assurer un travail de groupe. Il a forcé en nous l'autonomie pour résoudre les problèmes au cas où ils se présentent. Il nous a permis également d'être attentifs aux indications de notre encadrant, comment être bien organisés pour accomplir dans les meilleurs délais, et dans de meilleures conditions les tâches qui nous sont confiées.

Toutefois, nous comptons tester notre application avec des salons de coiffure, nous planifions également de le réaliser très prochainement et d'avoir un feedback de leur part afin d'améliorer et ajouter de nouvelles fonctionnalités et pourquoi pas le commercialiser.

Webographie

[1] ESPACE-CONCOURS. *Les chiffres du secteur de la coiffure*. URL : https://www.espace-concours.fr/cap-coiffure/chiffres-cles-secteur.

[2] K. NAFIL. *Rédaction d'un document Cahier des Charges*. URL : https://knafil.blogspot.com/2017/10/redaction-dun-document-cahier-des.html.

[3] Frédéric ESPIAU. *«Pas de place avant fin mai» : chez les coiffeurs, les délais d'attente explosent*. URL : https://www.lefigaro.fr/conso/pas-de-place-avant-fin-mai-chez-les-coiffeurs-les-delais-d-attente-explosent-20200512#:~:text=Depuis%20lundi%5C%2C%20de%20nombreux%20Fran%C3%A7ais,pour%20obtenir%20un%20rendez-vous..

[4] L. CHIEKHI. *Cours UML 2020*.

[5] *Spring boot*. URL : https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/.