

**REPUBLIQUE TUNISIENNE**  
**MINISTRE DE L'ENSEIGNEMENT SUPERIEUR**  
**ET DE LA RECHERCHE SCIENTIFIQUE**  
**UNIVERSITE TUNIS EL MANAR**



FACULTE DES SCIENCES DE TUNIS DEPARTEMENT DES SCIENCES DE  
L'INFORMATIQUE

**RAPPORT**  
DE PROJET DE FIN D'ETUDES :

---

# **Application de gestion des absences par Qr code**

---

REALISE Au SEIN DE



Présenté par : Marnissi Skander  
Et  
Saidi Hamdi  
Encadré par : M.Kamel Ben Salem

**ANNEE UNIVERSITAIRE : 2017-2018**

# Dédicaces

## **A nos très chers pères et nos très chères mères**

Dont leurs mérites, leurs sacrifices, leurs qualités humaines qui nous ont permis de vivre ce jour :  
Les mots nous manquent pour exprimer toute la reconnaissance, la fierté et le profond amour que nous leur  
portons pour chaque sacrifice qu'ils ont consenti afin que nous réussissions.  
Qu'ils trouvent ici le témoignage de notre attachement, notre reconnaissance, gratitude et respect.  
Tous nos sentiments de reconnaissance pour eux.

## **A mes grands-parents :**

Qui m'ont toujours encouragé, soutenu, et motivé. C'est grâce à eux que j'en suis arrivé ici.

## **A nos familles :**

Que nous ne pouvons nommer de peur d'en oublier nos attachements et nos affections les plus sincères.

## **A nos ami(e)s :**

A tous ceux qui ont su nous apporter aide et soutien aux moments propices, nous dédions ce travail,  
reconnaissant et remerciant chaleureusement.

---

## Remerciements

C'est avec un grand plaisir que je réserve cette page en signe de gratitude et de profonde reconnaissance à tous ceux qui nous ont aidé pendant la réalisation de ce travail, et d'une façon exceptionnelle :

**Notre encadrant :** M. KAMEL BEN SALEM, professeur à la Faculté Des Sciences de Tunis (FST), pour son aide précieuse, sa qualité d'encadrement et le savoir qu'il nous a transmis.

**Mme BASMA SASSI :** professeur à la Faculté Des Sciences Economiques et de Gestion de Tunis (FSEG), pour son aide et ses conseils.

**M.MOHAMED NAOUI et M. SAMIR ELLOUMI :** professeurs à la Faculté Des Sciences de Tunis (FST), pour leurs aides et leurs intérêts envers notre sujet.

Finalement, nous saisissons cette occasion pour remercier les membres du jury tout en espérant qu'ils trouvent dans ce rapport les qualités, la clarté et la motivation qu'ils attendent.

# Table des matières

Liste des figures.....	i
Liste des Tableaux.....	ii
Introduction Général .....	iii
<b>I Cadre du projet</b>	<b>1</b>
I.1 Cadre du projet .....	2
I.2 Présentation du sujet .....	2
I.3 Conclusion .....	2
<b>II Etude Préalable</b>	<b>3</b>
II.1 Introduction .....	4
II.2 Inspiration .....	4
II.3 Etude de l'existant .....	5
II.3.1 Description et analyse du processus.....	5
II.3.1.1 Inconvénients .....	5
II.4 Solution proposée .....	6
II.5 Technologie QR code : Quick Response Code .....	6
II.6 Méthodologie de travail et modélisation.....	7
II.6.1 Méthodologie de travail .....	7
II.6.1.1 Etude comparative entre Firebase et SQLite.....	8
II.6.1.2 Les patrons de conception.....	10
II.6.2 Modélisation .....	11
<b>III Spécification des besoins</b>	<b>12</b>
III.1 Détermination des besoins.....	13
III.2 Spécification des besoins fonctionnels pour l'application mobile.....	13
III.2.1 BF1 : Détection et lecture du QR code .....	13
III.2.2 BF2 : Transmission des données dans une base de données orienté documents (Firebase) .....	13
III.3 Spécification des besoins fonctionnels pour l'application bureautique.....	13
III.3.1 BF1 : Génération d'un QR code .....	13
III.3.2 BF2 : Synchronisation des données pour afficher la liste des présences et absences .....	14
III.3.3 BF3 : Transmissions automatique du résultat dans la base de donnée de la scolarité .....	14
III.4 Spécification des besoins non fonctionnels de l'application .....	14
III.4.1 BNF1 : Disponibilité .....	14

III.4.2	BNF2 : Extensibilité .....	14
III.4.3	BNF3 : Temps de réponse .....	14
III.4.4	BNF4 : Fiabilité .....	14
III.4.5	BNF5 : Robustesse .....	14
III.4.6	BNF5 : Sécurité .....	15
III.5	Spécification des besoins détaillée.....	15
III.5.1	Identification des acteurs.....	15
III.5.1.1	Analyse et description des acteurs.....	15
III.5.1.2	Identification et description des acteurs.....	16
<b>IV</b>	<b>Conception</b> .....	<b>31</b>
IV.1	Conception globale .....	32
IV.1.1	Architecture générale de l'application .....	32
IV.1.1.1	Architecture générale de l'application bureautique .....	34
IV.1.1.2	Architecture générale de l'application mobile .....	35
IV.1.2	Conception détaillée.....	36
IV.1.2.1	Diagramme de classes .....	36
IV.1.2.2	Diagramme de séquences .....	37
IV.1.2.3	Représentation de la structure de données dans Firebase .....	43
<b>V</b>	<b>Réalisation</b> .....	<b>45</b>
V.1	Environnement de travail .....	46
V.1.1	Environnement matériel .....	46
V.1.2	Environnement logiciel et choix technologique .....	46
V.1.2.1	Logiciels utilisés.....	46
V.1.2.2	Choix technologiques.....	52
V.1.3	Application bureautique .....	53
V.1.4	Application mobile .....	69

# Table des figures

II.1	Faculté en Malaisie.....	4
II.2	Exemple d'un Qr code.....	7
II.3	Exemple d'itération selon la méthode agile SCRUM.....	8
II.4	Structure d'un patron MVC.....	10
III.1	Cas d'utilisation global.....	16
III.2	Cas d'utilisation « Gérer les agents administratifs » .....	17
III.3	Cas d'utilisation « Gérer les professeurs » .....	20
III.4	Cas d'utilisation « Gérer les étudiants ».....	23
III.5	Cas d'utilisation « Gérer les présences » .....	26
IV.1	Architecture de l'application Quick Attendance .....	32
IV.2	Architecture de l'application Quick Attendance .....	33
IV.3	Architecture de l'application bureautique.....	34
IV.4	Architecture de l'application mobile.....	35
IV.5	Diagramme de classes.....	36
IV.6	Diagramme de séquence du cas d'utilisation « S'authentifier » .....	38
IV.7	Diagramme de séquence « Créer un QR code » .....	39
IV.8	Diagramme de séquence « Consulter présences » .....	41
IV.9	Représentation de la structure de données dans Firebase .....	43
V.1	Présentation de l'interface de l'authentification. ....	53
V.2	Présentation de l'interface de la récupération de mot de passe oublié.....	54
V.3	Présentation de l'interface de l'accueil du professeur .....	55
V.4	Présentation de l'interface du générateur de Qr code pour le professeur .....	56
V.5	Présentation de l'interface de l'affichage des présences dans la séance .....	57
V.6	Présentation de l'interface de l'accueil de l'agent administratif.....	58
V.7	Présentation de l'interface de la barre d'options de l'agent administratif .....	59
V.8	Présentation de l'interface de l'ajout d'un étudiant par l'agent administratif .....	60
V.9	Présentation de l'interface de l'ajout d'un professeur par l'agent administratif .....	61
V.10	Présentation de l'interface de l'accueil de l'administrateur .....	62
V.11	Présentation de l'interface de la barre d'options de l'administrateur.....	63
V.12	Présentation de l'interface de l'ajout d'un agent administratif par l'administrateur .....	64

V.13 Présentation de l'interface de la confirmation après chaque ajout.....	65
V.14 Présentation de l'interface de l'ajout de l'étudiant sur Firebase.....	66
V.15 Présentation de l'interface de l'ajout de la présence après le scan sur Firebase.....	67
V.16 Présentation d'un exemple de QR code généré par le professeur.....	68
V.17 Présentation de l'interface d'authentification .....	69
V.18 Présentation de l'interface de la liste déroulante.....	70
V.19 Présentation de l'interface du scan.....	71
V.20 Présentation de l'interface du scan.....	72
V.21 Présentation de l'interface des informations .....	73

# Liste des tableaux

II.1 "Choix des outils" .....	9
III.1 Description textuelle du cas d'utilisation « Ajouter un agent administratif » .....	18
III.2 Description textuelle de cas d'utilisation « Modifier un agent administratif » .....	18
III.3 Description textuelle de cas d'utilisation « Supprimer un agent administratif » .....	19
III.4 Description textuelle du cas d'utilisation « Ajouter un professeur » .....	21
III.5 Description textuelle de cas d'utilisation « Modifier un professeur » .....	21
III.6 Description textuelle de cas d'utilisation « Supprimer un professeur » .....	22
III.7 Description textuelle du cas d'utilisation « Ajouter un étudiant » .....	24
III.8 Description textuelle de cas d'utilisation « Modifier un étudiant » .....	24
III.9 Description textuelle de cas d'utilisation « Supprimer un étudiant » .....	25
III.10Description textuelle de cas d'utilisation « Ajouter un étudiant présent » .....	27
III.11Description textuelle de cas d'utilisation « Modifier l'état de présence d'un étudiant » .....	27
III.12Description textuelle de cas d'utilisation « Créer un QR code » .....	28
III.13Description textuelle de cas d'utilisation « Scanner le QR code » .....	29
IV.1 Scénario de diagramme de séquence « Créer un QR code » .....	40
IV.2 Scénario de diagramme de séquence « Consulter présences » .....	42



## Introduction Générale

Parmi les éléments qui représentent les critères d'une formation de qualité nous trouvons le suivi de l'assiduité des étudiants.

Vu le nombre élevé des étudiants, cette tâche demeure dans certain cas difficile à réaliser, ou peut entraîner une énorme perte de temps.

C'est dans ce cadre que se situe notre travail de « Projet de Fin d'Étude » au sein de la Faculté des sciences de Tunis où nous sommes appelés à proposer une solution de gestion d'absences tout en profitant des nouvelles technologies et en minimisant les anomalies de tâches classiques.

Notre projet consiste à utiliser la technologie du QR code pour effectuer l'appel de présence en cours en un laps de temps ; dès la rentrée en classe, l'enseignant affiche en diapo une image représentant un QR code, et les étudiants doivent le scanner pour inscrire leurs présences.

Ce rapport décrit les détails du projet et il est organisé comme suit :

Dans le premier chapitre, nous commençons par la présentation du cadre générale et la présentation de l'organisme d'accueil. Ensuite, nous allons présenter le sujet.

Dans le deuxième chapitre nous allons faire une étude préalable et donc étudier ce qui existe déjà, puis proposer une solution, et enfin présenter la méthodologie de travail.

Une fois que l'étude préalable est accomplie, nous allons définir les besoins fonctionnels et non fonctionnels que devra satisfaire notre application. Nous aurons alors une vision plus claire, grâce aux diagrammes de cas d'utilisation.

Après avoir défini nos besoins, le quatrième chapitre sera consacré à la conception basée sur les diagrammes UML.

Le dernier chapitre consiste à décrire la réalisation de notre application, nous y éclairerons les étapes d'achèvement du travail et mettrons l'accent sur les environnements et technologies utilisés.

Enfin, nous terminerons notre rapport par une conclusion générale contenant des proposition pour faire évoluer notre application en ajoutant plusieurs fonctionnalités.

# **Chapitre I**

## **Cadre du projet**

## Introduction

L'objectif principal de ce chapitre introductif est de mettre en valeur le contexte général du projet. Nous présentons dedans l'entreprise qui nous a conviés pour la réalisation de notre projet que nous présenterons ensuite son sujet.

### I.1 Cadre du projet

Dans le cadre de leur formation pour l'intention du diplôme de licence en informatique, les étudiants de la Faculté des sciences de Tunis sont appelés à réaliser un projet dans un , ou viser pour le milieu professionnel. C'est dans cette perspective que nous réalisons notre Projet de Fin d'Etudes, afin d'accomplir les acquis développés et enrichis au fil des années de formation académique, et pour nous initier à la vie professionnelle et pratique.

### I.2 Présentation du sujet

Via la réalisation de ce projet, La Faculté Des Science vise à automatiser et faciliter la gestion des absences en intégrant les QR codes.

**Le projet comporte Une application mobile et bureautique :**

Une application **mobile** capable de s'en servir de la caméra de son support électronique pour détecter et analyser les QR codes dans le but de marquer la présence de l'étudiant.

Une **application Bureautique (Desktop)** qui comporte deux volets :

- Le premier sert à générer les QR codes et consulter les absences du côté du professeur.
- Le second sert à la gestion de la base de données de la scolarité du côté de l'administration.

### I.3 Conclusion

Ce premier chapitre a permis de situer notre travail dans son cadre général via la présentation de son objectif ainsi que la société d'accueil qui le propose.

Les exigences et les contraintes de notre projet seront détaillées dans les chapitres suivants.

## **Chapitre II**

### **Etude Préalable**

## II.1 Introduction

Dans ce chapitre, nous présentons l'étude préalable qui a précédé la mise en place de notre application. Cette partie repose, en premier lieu, pour parvenir à présenter une bonne spécification des besoins, nous sommes donc tenus d'organiser et de bien analyser les fonctionnalités et les résultats attendus par notre projet. Nous allons commencer par une étude de la solution existante pour pouvoir en dégager les défauts et mettre en valeur l'apport de la solution.

## II.2 Inspiration

Notre projet a été inspiré par un modèle mis en test en Malaisie qui consiste à confirmer des présences en scannant un QR code spécifique présenté par la faculté au début de chaque cours, nous espérons que notre projet sera un premier pas vers l'avancement technologique pour améliorer le système éducatif de notre pays et pour faire avancer la qualité de l'enseignement Tunisien.



FIGURE II.1 – Faculté en Malaisie

## II.3 Etude de l'existant

Afin d'approfondir notre compréhension du sujet et avoir une meilleure idée sur le travail demandé et ses fonctionnalités, nous avons effectué une critique sur ce qui existe dans la majorité des facultés tunisiennes et ce en observant le processus de la gestion des absences qui se déroule ainsi :

- 1. Distribution de la feuille de présences.
- 2. Récupération de la feuille après écriture des noms, prénoms, cin et groupes de présence des étudiants.
- 3. Appliquer un appel pour vérifier la cohérence des informations inscrites.
- 4. Transmettre cette feuille à l'administration.
- 5. Enregistrement des présences et absences dans la base de données de l'administration.

### II.3.1 Description et analyse du processus

Lors d'une séance de cours ou de travaux dirigés le professeur ou assistant doit préparer une liste vierge qui servira comme liste de présences.

Le professeur distribue tout d'abord la feuille en demandant aux étudiants d'inscrire leurs numéros de carte d'identité nationale ou le numéro de leur carte d'étudiants puis leurs noms leurs prénoms leurs groupes (si c'est une séance de travaux pratiques) et enfin de signer.

Il récupère ensuite cette feuille, et fait un contrôle en faisant l'appel ; **SI** l'étudiant appelé ne répond pas par « présent » quand son nom sera évoqué, le professeur le supprime de la liste **SINON** il passe au nom suivant.

Ce dernier va enfin transmettre cette feuille de présence à l'administration qui va ensuite enregistrer cette liste dans sa base de données.

#### **Remarque :**

L'appelle n'est souvent pas appliqué par les professeurs ou assistants.

#### II.3.1.1 Inconvénients

Ce processus présente une multitude d'inconvénients :

- Risque de perte de la feuille de présence.
- Perte de temps pour chaque transfert du côté du professeur.
- Risque de fraude lors de l'inscription des présences.
- Perte de temps dans l'inscription des données dans la base de données de l'administration.
- La non fiabilité des informations écrites sur la feuille de présences.

## II.4 Solution proposée

En analysant le sujet et en étudiant la solution existante, nous pouvons dégager les objectifs à atteindre durant ce projet.

Afin d'automatiser ce processus et d'en railler les inconvénients notre solution consiste, en premier lieu, à réaliser une application **mobile** sous plateforme Android qui permet de scanner un QR code qui sera afficher dans un des slides du cours du professeur ou bien au début d'une séance de travaux dirigée par l'assistant dans le but d'inscrire automatiquement la présence de l'étudiant avec tous ses données personnelles : numéro carte d'identité , nom , prénom , section , date dans une base de données orienté document.

En second lieu, elle consiste à réaliser une application **bureautique(desktop)** qui permet aux professeurs ou assistants de générer un QR code pour la séance voulu et récupérer automatiquement les présences et absences.

Enfin en troisième lieu, à crée une session pour administrateur dans la même application dans le but de gérer : consulter, modifier, ajouter ,supprimer les données dans la base de données de la scolarité et de créer des statistiques grâce aux données enregistrées.

## II.5 Technologie QR code : Quick Response Code

Le QR code est un type moderne de code à barres bi-dimensionnel. Avantage par rapport à ses ancêtres par la quantité d'information immense qu'il peut stocker et la facilité de reconnaissance des données par les applications capables de l'utiliser il est devenu très populaire dans tous les domaines ; vie courante multimédia sécurité gestion de stocks etc. ...

C'est pour cela que nous avons observé le potentiel d'exploiter ses avantages pour la gestion et le transfert rapide et fluides des informations dans notre solution.

La figure ci-dessous représente un QR code crypté qui contient les informations nécessaires pour confirmer une présence tels que le département, la classe la matière et un code secret utiliser par l'algorithme de décryptage personnaliser pour maintenir la sécurité de nos codes.





FIGURE II.2 – Exemple d'un Qr code

## II.6 Méthodologie de travail et modélisation

### II.6.1 Méthodologie de travail

Le développement de ce projet est effectué par la méthode agile Scrum[4].

Les méthodes agiles sont de méthodologies essentiellement dédiées à la gestion de projets informatiques. Elles reposent sur des cycles de développement intégratifs et adaptatifs en fonction des besoins évolutifs du client. Elles permettent notamment d'impliquer l'ensemble des collaborateurs ainsi que le client dans le développement du projet.

Ces méthodes permettent généralement de mieux répondre aux attentes du client en un temps limité (en partie grâce à l'implication de celui-ci) tout en faisant monter les collaborateurs en compétences. Ces méthodes constituent donc un gain en productivité ainsi qu'un avantage compétitif tant du côté du client que du côté du fournisseur.

Créée en 2002, la méthode Scrum est une méthode agile, elle s'appuie sur le découpage des projets en itérations « sprints » qui peut avoir une durée qui varie en moyenne entre deux semaines et un mois.

Avant chaque sprint, les tâches sont estimées en temps et en complexité à l'aide de certaines pratiques comme le « planning poker », une manière ludique de chiffrer la complexité des tâches ou évolutions à l'aide de cartes à l'instar du célèbre jeu dont le nom est repris. Ces estimations permettent à la fois de planifier les livraisons, mais aussi d'estimer le coût de ces tâches auprès du client. Les fonctionnalités (encore appelées « User stories ») qui sont l'objet d'un sprint constituent ce que l'on appelle un « sprint backlog » du produit éventuellement livrable à la fin du sprint. Il est nécessaire de distinguer le sprint backlog du « Product backlog » qui lui correspond à l'ensemble des fonctionnalités attendues pour le produit sur l'ensemble des sprints.

La méthode Scrum est aussi caractérisée par une « mêlée » quotidienne, encore appelée « Morning » où



Indiquent tour à tour les tâches qu'ils ont effectuées la veille, les difficultés rencontrées et enfin ce sur quoi ils vont poursuivre leur travail le jour suivant.

Cela permet d'évaluer l'avancement du projet, de mobiliser des ressources là où cela est le plus nécessaire, mais aussi de venir en aide aux collaborateurs rencontrant des difficultés lorsque celles-ci ont déjà été rencontrées auparavant par d'autres membres de l'équipe.

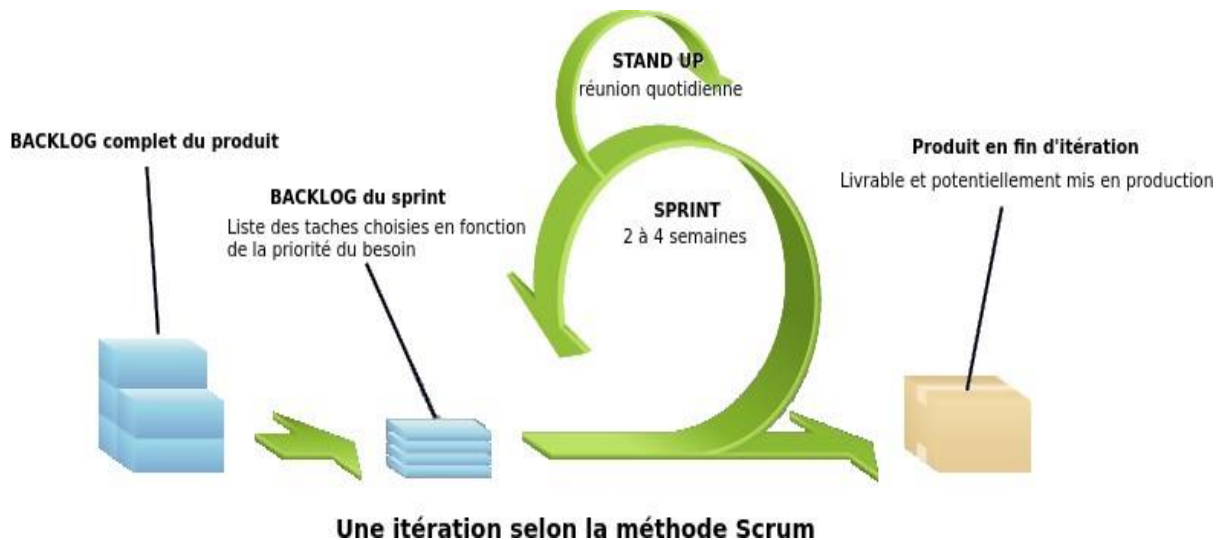


FIGURE II.3 – Exemple d'itération selon la méthode agile scrum

### II.6.1.1 Etude comparative entre Firebase et SQLite

La figure représente une comparaison générale entre Firebase Database et SQLite la plus utilisée dans les applications mobiles ; Malgré le charme d'open source avec SQLite, Firebase offre beaucoup plus de service intéressants et innovants gratuitement citant :

- 1. Authentification.
- 2. Analyse de données et des crashes.
- 3. Machine Learning.
- 4. Prédictions et surveillance de la performance.

Avec la caractéristique « Schema-Free » Firebase semble plus facile à manipuler en offrant de plus une mise à jour en temps réel des données stockées c'est à dire tout changement dans la base de données est pris en charge immédiatement. Par conséquent, et vue la migration vers le Cloud des bases de données nous avons choisi Firebase Database comme base de données pour notre application mobile.

TABLE II.1 – "Choix des outils"

<b>DB</b>	<b>Firestore Database</b>	<b>SQLite</b>
Description	Cloud-hosted, stockage de document en temps réel.	Base de données populaire utilisée par SGBDR.
Modèle primaire	Orienté document.	Relationnel.
Modèle primaire	Google.	Dwayne Richard Hipp.
Développé par	-Open Source	Relationnel.
Licence	Freemium/commercial.	Open source.
Basée Cloud	Oui.	Non.
API et autre Méthodes d'accès	Android, IOS RESTful HTTP API JavaScript API	ADO.NET JDBC ODBC
Langages de programmation supporté	Java JavaScript Objective-C	Java, JavaScript, Objective-C Python, Ruby, Scala, C C#, C++, etc..

### II.6.1.2 Les patrons de conception

Les patrons de conception [1] (en anglais Design Patterns) sont un recueil de bonnes pratiques de conception pour un certain nombre de problèmes récurrents en programmation orientée objet.

#### Le patron de conception Ioc :

L'inversion de contrôle Ioc [2] (en anglais Inversion of Control) ou l'injection de dépendances (en anglais Dependency Injection) est un patron qui a pour objectif de faciliter l'intégration de composants entre eux.

Le concept est basé sur le fait d'inverser la façon dont sont créés les objets.

#### Le patron de conception MVC :

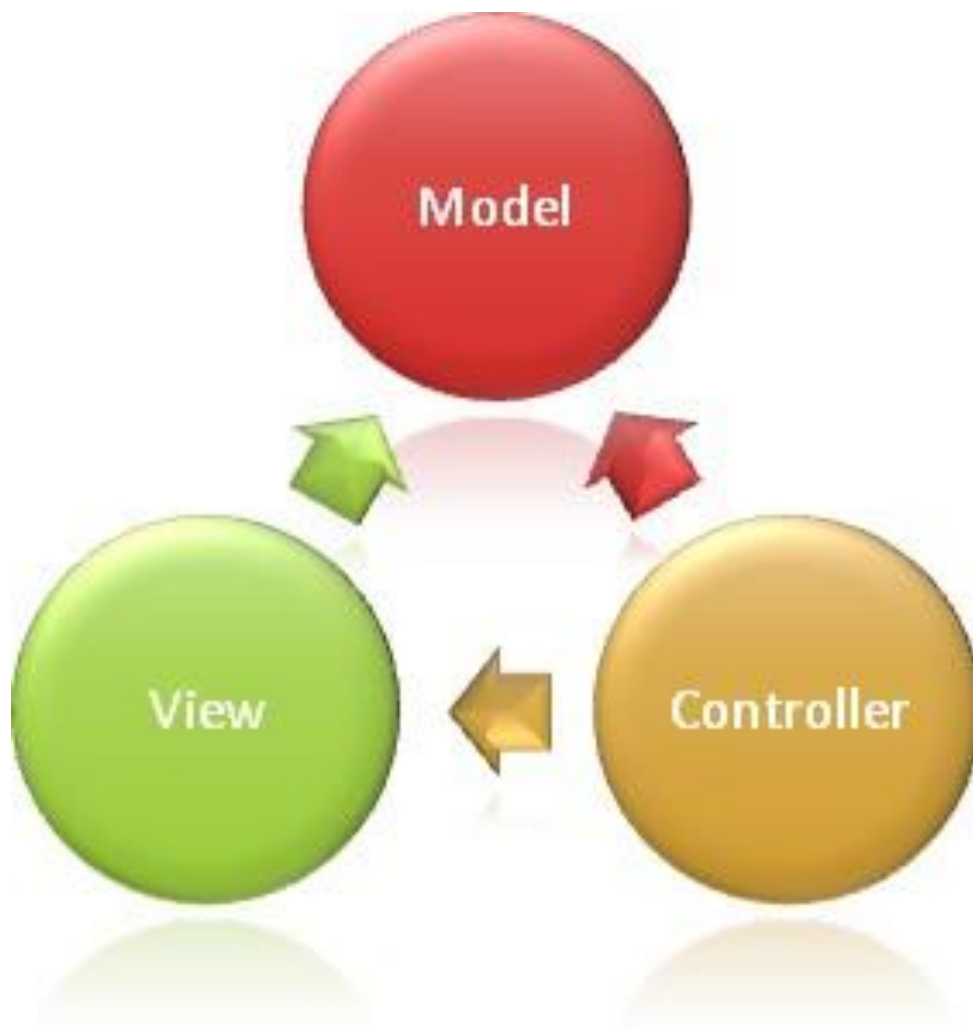


FIGURE II.4 – Structure d'un patron MVC

Modèle-vue-contrôleur [3] ou MVC est un motif d'architecture logicielle destiné aux interfaces graphiques.

Une application conforme au motif MVC comporte trois types de modules : les modèles, les vues et les contrôleurs.

**Modèle** : Élément qui contient les données ainsi que de la logique en rapport avec les données : validation, lecture et enregistrement.

Il peut, dans sa forme la plus simple, contenir uniquement une simple valeur, ou une structure de données plus complexe. Le modèle représente l'univers dans lequel s'inscrit l'application.

Par exemple pour une application de banque, le modèle représente des comptes, des clients, ainsi que les opérations telles que dépôt et retraits, et vérifie que les retraits ne dépassent pas la limite de crédit.

**Vue** : Partie visible d'une interface graphique. La vue se sert du modèle, et peut être un diagramme, un formulaire, des boutons, etc.

Une vue contient des éléments visuels ainsi que la logique nécessaire pour afficher les données provenant du modèle. Dans une application de bureau classique, la vue obtient les données nécessaires à la présentation du modèle en posant des questions.

Elle peut également mettre à jour le modèle en envoyant des messages appropriés. Dans une application web une vue contient des balises HTML.

**Contrôleur** : Module qui traite les actions de l'utilisateur, modifie les données du modèle et de la vue.

## II.6.2 Modélisation

Nous avons choisi de réaliser la modélisation de nos diagrammes en UML car il correspond le plus à notre méthodologie de travail. En effet, presque tous les sprints de développement logiciel nécessitent des diagrammes tels que le diagramme de cas d'utilisation, de séquence et de classe ce qui rend UML le choix le plus compatible.

## Conclusion

Dans ce chapitre, nous avons présenté la solution existante en mettant l'accent sur ses limites et la méthodologie utilisée. Ceci nous permettra de développer les besoins fonctionnels et non fonctionnels, ainsi que les acteurs de notre solution proposée dans le chapitre suivant.

## **Chapitre III**

# **Spécification des besoins**

## **Introduction**

Nous présentons dans cette partie une description du projet avec une étude du contexte fonctionnel du système. Cette description se base sur la détermination des besoins non fonctionnels et des besoins fonctionnels qui nous permettront d'identifier et de décrire les cas d'utilisation du système.

### **III.1 Détermination des besoins**

Le chapitre précédent a permis de définir les différentes fonctionnalités qui seront mises en disposition des acteurs principaux. Ces fonctionnalités dégagées sont ainsi classées en besoins fonctionnels et besoins non fonctionnels.

### **III.2 Spécification des besoins fonctionnels pour l'application mobile**

#### **III.2.1 BF1 : Détection et lecture du QR code**

L'application permet d'identifier et de lire un QR code dans un flux vidéo en temps réel et de lire les informations qu'il contient.

#### **III.2.2 BF2 : Transmission des données dans une base de données orienté documents (Firestore)**

L'application permet d'envoyer les données personnelles de l'étudiant et la confirmation de sa présence automatiquement après avoir effectué le scan du QR code affiché dans la séance.

### **III.3 Spécification des besoins fonctionnels pour l'application bureautique**

#### **III.3.1 BF1 : Génération d'un QR code**

L'application permet de générer un QR code contenant des informations sur la séance : le département de la section, la matière.



### **III.3.2 BF2 : Synchronisation des données pour afficher la liste des présences et absences**

L'application permet de synchroniser les données qui ont été transmis lors du scan dans la base de données orienté document (Firestore) et affiche le résultat sous forme d'un tableau en distinguant les étudiants absents et présents

### **III.3.3 BF3 : Transmissions automatique du résultat dans la base de données de la scolarité**

L'application permet d'envoyer automatiquement le résultat après la synchronisation des données dans la base de données de la scolarité.

## **III.4 Spécification des besoins non fonctionnels de l'application**

### **III.4.1 BNF1 : Disponibilité**

L'application doit être toujours disponible, Pour ça, elle ne doit pas dépendre d'une connexion internet.

### **III.4.2 BNF2 : Extensibilité**

L'application devra être réaliser avec une architecture évolutive, qui rendra simple l'ajout des fonctionnalités.

### **III.4.3 BNF3 : Temps de réponse**

Pour une expérience utilisateur fluide, toutes tâches de l'application doivent s'exécuter dans un temps raisonnable.

### **III.4.4 BNF4 : Fiabilité**

Pour avoir un bon résultat qui ne comporte pas d'incohérence les informations, les résultats de requête et les traitements des données doivent s'appuyer sur des données qui sont correctes.

### **III.4.5 BNF5 : Robustesse**

L'application doit être pertinente de ses réactions à des situations anormales comme les erreurs dues aux utilisateurs dans les événements qu'il déclenche avec les interfaces et les données qu'il soumet à l'application,

clarté des messages et arrêt avant traitement dans le cas où les données n'ont pas de sens ou qu'elles sont incohérents ou bien des erreurs dues à l'environnement.

### III.4.6 BNF<sub>5</sub> : Sécurité

L'application doit être infaillible et sûre pour éviter la modification des données et les tentatives de fraudes.

## III.5 Spécification des besoins détaillée

La conception détaillée d'un projet vise à expliquer en détail l'organisation et la répartition des tâches entre les modules et les objets constituant le projet afin de préparer la phase de réalisation. Elle a pour but d'expliquer les solutions choisies afin de mettre en place les modules et les différents scénarios qui seront exécutés par l'application. En outre, cette phase englobe aussi la préparation des plans de tests unitaire de chaque module qui permettent de voir si le module répond aux spécifications du projet ou non.

### III.5.1 Identification des acteurs

Afin de schématiser de façon simple et expressive les différentes fonctionnalités et les attentes des utilisateurs vis à vis de notre application nous présentons ci-dessous les diagrammes de cas d'utilisation correspondants.

Nous commençons tout d'abord par l'identification des différents acteurs intervenant dans le fonctionnement de l'application, puis nous déterminons les services rendus pour chaque acteur.

#### III.5.1.1 Analyse et description des acteurs

Notre application présente quatre acteurs principaux :

##### **Administrateur :**

L'administrateur est le responsable de la gestion des agents administratifs. Il se situe au sommet de la hiérarchie de l'application. Un administrateur dispose d'un accès privilégié à travers son « Log in » et son « Mot de passe ».

##### **Agent administratif :**

Un agent administratif est le responsable de la gestion des professeurs et des étudiants. Il a le privilège de consulter les résultats finals de présences avec la possibilité de faire des statistiques. Un administrateur possède un accès privilégié à travers son « Log in » et son « Mot de passe ».

##### **Professeur :**

Le professeur est le responsable de la création des QR code spécifique et de la récupération des résultats post-traitement, il a aussi le droit de gérer l'état de présences des étudiants ainsi il peut changer les résultats de présences manuellement si besoin. Un professeur dispose d'un accès personnel à travers son « Log in » et son « Mot de passe ».

##### **Étudiant :**

L'étudiant est l'acteur principale de l'application mobile, il scanne le QR code afficher durant le cours par le professeur dans le but de confirmer sa présence, il a aussi le droit de consulter ses absences. Un étudiant dispose d'un accès personnel à travers son « Log in » et son « Mot de passe ».

### III.5.1.2 Identification et description des acteurs

Suite à l'identification des acteurs supposés dans le système, nous allons maintenant identifier les cas d'utilisation.

#### Cas d'utilisation global

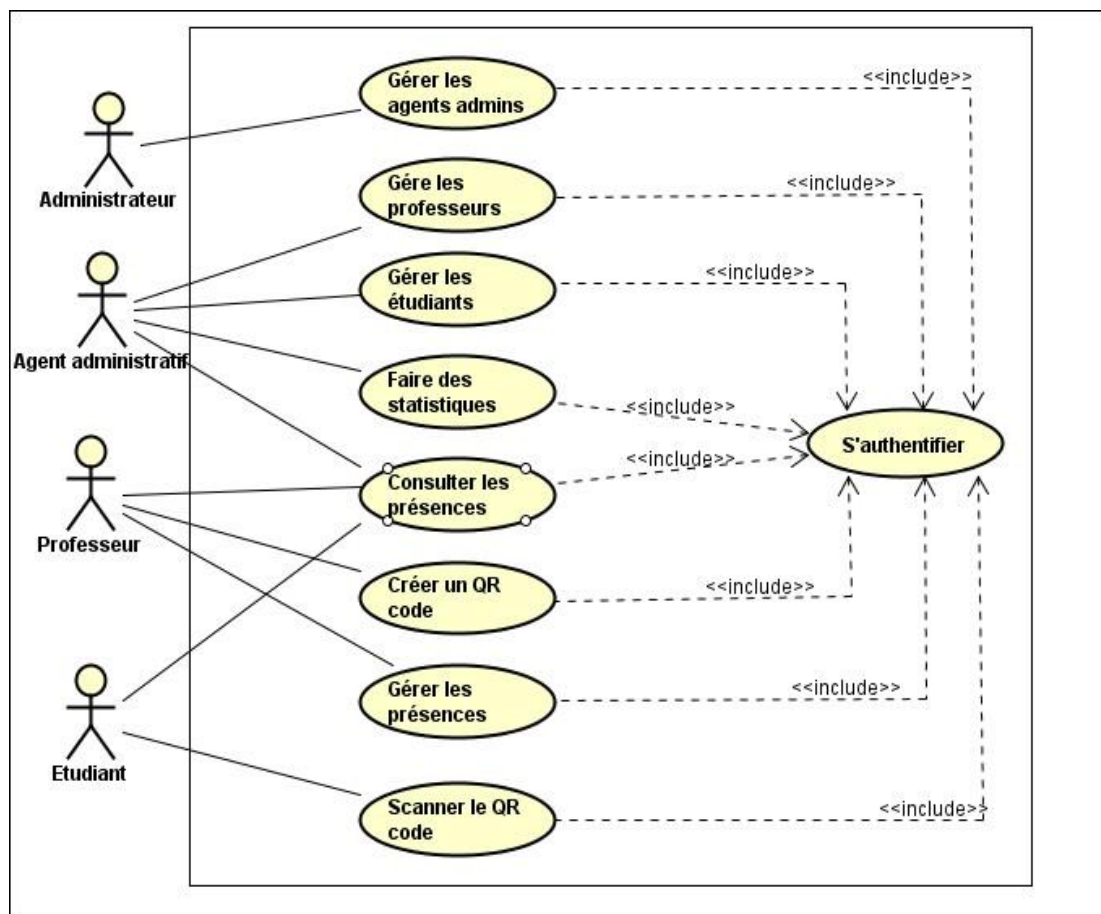


FIGURE III.1 – Cas d'utilisation global

La figure 3.1 représente une visualisation de diagramme de cas d'utilisation global de l'application « Quick Attendance » en prenant en compte les deux parties mobile et desktop sans montrer les détails. Les interactions entre les quatre acteurs vont être détaillées plus bas en développant chaque cas d'utilisation.

#### Cas d'utilisation « Gérer les agents administratifs »

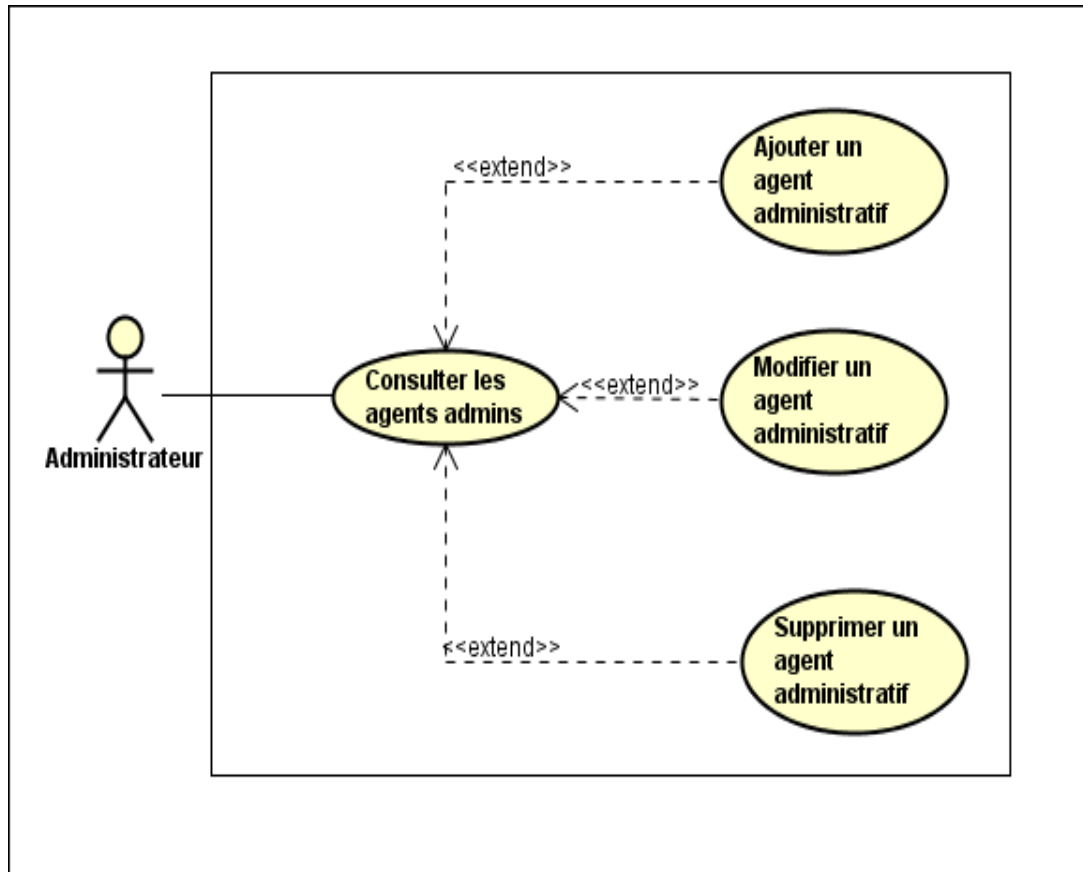


FIGURE III.2 – Cas d'utilisation « Gérer les agents administratifs »

La figure 3.2 représente le diagramme de cas d'utilisation « Gérer les agents administratifs ». Nous allons maintenant détailler ce processus à l'aide d'une description textuelle.

#### Cas d'utilisation « Ajouter un agent administratif »

TABLE III.1 – Description textuelle du cas d'utilisation « Ajouter un agent administratif »

Acteur	Administrateur.
Précondition	Administrateur authentifié.
Post-traitement	Agent administratif ajouté.
Scénario	<ul style="list-style-type: none"> <li>• L'admin ajoute les informations du nouvel agent administratif.</li> <li>• Il confirme l'ajout.</li> <li>• L'agent admin est ajouté.</li> </ul>

Le tableau 3.1 représente la description textuelle du cas d'utilisation « Ajouter un agent administratif »

#### Cas d'utilisation « Modifier un agent administratif »

TABLE III.2 – Description textuelle de cas d'utilisation « Modifier un agent administratif »

Acteur	Administrateur.
Précondition	Administrateur authentifié et liste des agents administratifs affichés.
Post-traitement	Agent administratif modifié.
Scénario	<ul style="list-style-type: none"> <li>• L'admin sélectionne l'agent administratif à modifier.</li> <li>• Il établit les changements nécessaires.</li> <li>• Il confirme les changements.</li> <li>• L'agent administratif est modifié.</li> </ul>

Le tableau 3.2 représente la description textuelle du cas d'utilisation « Modifier un agent administratif »

### Cas d'utilisation « Supprimer un agent administratif »

TABLE III.3 – Description textuelle de cas d'utilisation « Supprimer un agent administratif »

Acteur	Administrateur.
Précondition	Administrateur authentifié et liste des agents Administratifs affichés.
Post-traitement	Agent administratif supprimé.
Scénario	<ul style="list-style-type: none"> <li>• 1. L'admin sélectionne l'agent à supprimer.</li> <li>• 2. Il clique sur le bouton « Supprimer ».</li> <li>• 3. Il confirme la suppression de l'agent sélectionné.</li> <li>• 4. L'agent est supprimé.</li> </ul>

Le tableau 3.3 représente la description textuelle du cas d'utilisation « Supprimer un agent administratif ».

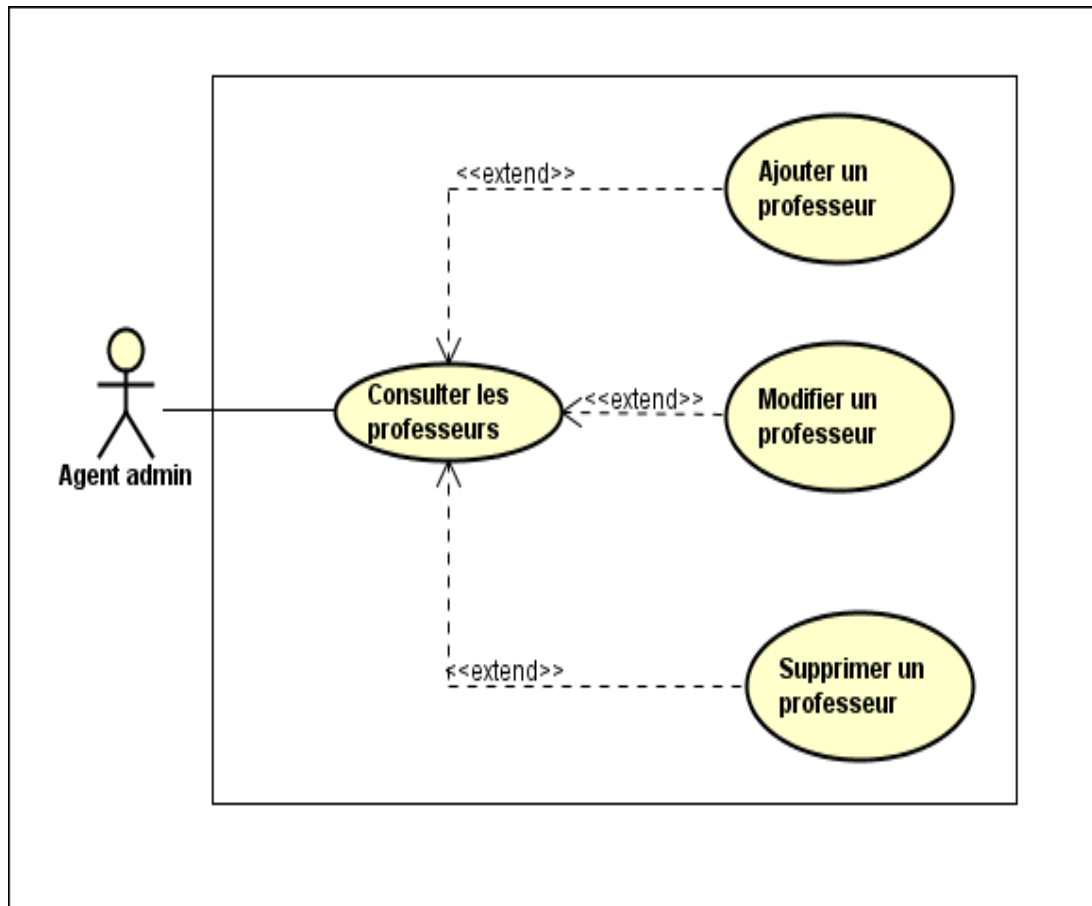
**Cas d'utilisation « Gérer les professeurs »**

FIGURE III.3 – Cas d'utilisation « Gérer les professeurs »

La figure 3.3 représente le diagramme de cas d'utilisation « Gérer les professeurs ». Nous allons maintenant détailler ce processus à l'aide d'une description textuelle.

#### Cas d'utilisation « Ajouter un professeur »

TABLE III.4 – Description textuelle du cas d'utilisation « Ajouter un professeur »

Acteur	Agent admin.
Précondition	Agent admin authentifié.
Post-traitement	Professeur ajouté.
Scénario	<ul style="list-style-type: none"> <li>• L'agent ajoute les informations du nouveau professeur.</li> <li>• Il confirme l'ajout.</li> <li>• Le professeur est ajouté.</li> </ul>

Le tableau 3.4 représente la description textuelle du cas d'utilisation « Ajouter un professeur ».

#### Cas d'utilisation « Modifier un professeur »

TABLE III.5 – Description textuelle de cas d'utilisation « Modifier un professeur ».

Acteur	Agent admin.
Précondition	Agent admin authentifié et liste des professeurs affichée.
Post-traitement	Professeur modifié.
Scénario	<ul style="list-style-type: none"> <li>• L'agent sélectionne le professeur à modifier.</li> <li>• Il établit les changements nécessaires.</li> <li>• Il confirme les changements.</li> <li>• Le professeur est modifié.</li> </ul>



Le tableau 3.5 représente la description textuelle du cas d'utilisation « Modifier un professeur ».

**Cas d'utilisation « Supprimer un professeur »**

TABLE III.6 – Description textuelle de cas d'utilisation « Supprimer un professeur ».

Acteur	Administrateur.
Précondition	Administrateur authentifié et liste des professeurs affichée.
Post-traitement	Professeur supprimé.
Scénario	<ul style="list-style-type: none"><li>• L'admin sélectionne le professeur à supprimer.</li><li>• Il clique sur le bouton « Supprimer ».</li><li>• Il confirme la suppression du professeur sélectionné.</li><li>• 8. Le professeur est supprimé.</li></ul>

Le tableau 3.6 représente la description textuelle du cas d'utilisation « Supprimer un professeur ».

#### Cas d'utilisation « Gérer les étudiants »

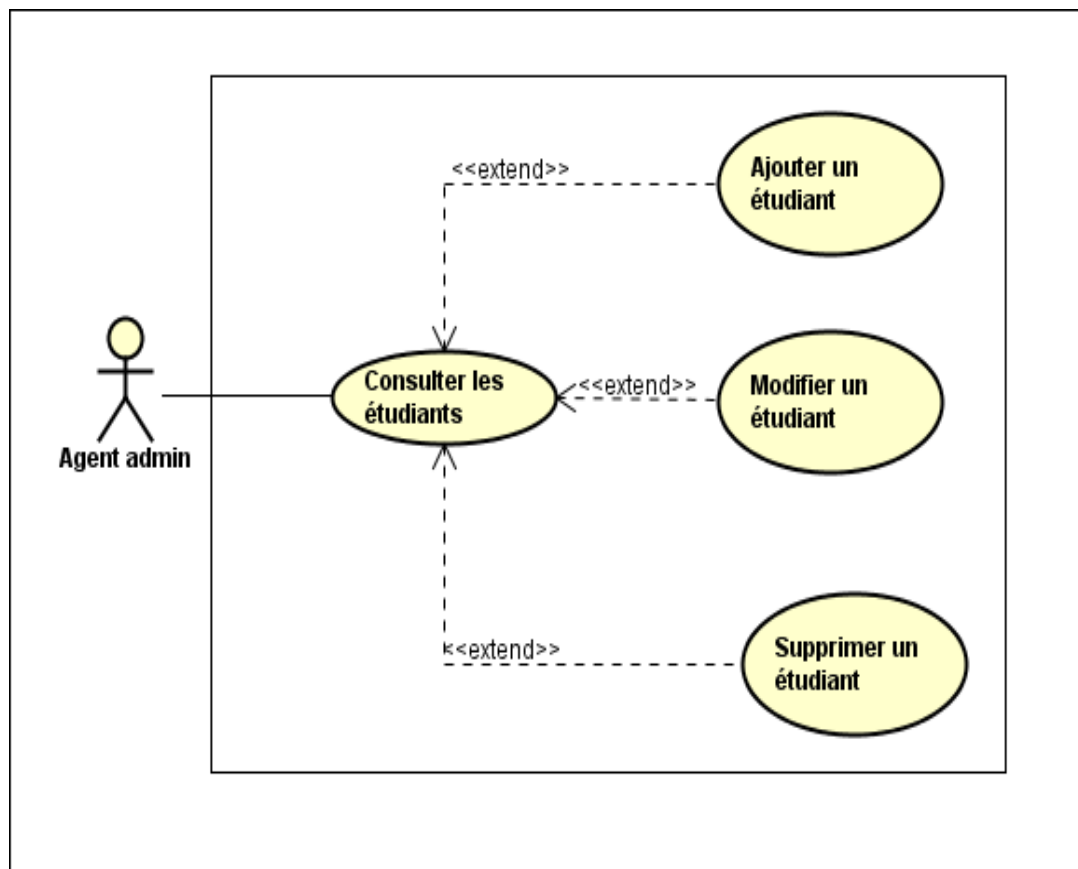


FIGURE III.4 – Cas d'utilisation « Gérer les étudiants »

La figure 3.4 représente le diagramme de cas d'utilisation « Gérer les étudiants ». Nous allons maintenant détailler ce processus à l'aide d'une description textuelle.

#### Cas d'utilisation « Ajouter un étudiant »

TABLE III.7 – Description textuelle du cas d'utilisation « Ajouter un étudiant »

Acteur	Agent admin..
Précondition	Agent authentifié.
Post-traitement	Étudiant ajouté.
Scénario	<ul style="list-style-type: none"> <li>• L'agent ajoute les informations du nouvel étudiant.</li> <li>• Il confirme l'ajout.</li> <li>• L'étudiant est ajouté.</li> </ul>

Le tableau 3.7 représente la description textuelle du cas d'utilisation « Ajouter un étudiant ».

#### Cas d'utilisation « Modifier un étudiant »

TABLE III.8 – Description textuelle de cas d'utilisation « Modifier un étudiant ».

Acteur	Agent admin.
Précondition	Agent authentifié et liste des étudiants affichée.
Post-traitement	Étudiant modifié.
Scénario	<ul style="list-style-type: none"> <li>• L'agent sélectionne l'étudiant à modifier.</li> <li>• Il établit les changements nécessaires.</li> <li>• Il confirme les changements.</li> <li>• 4. L'étudiant est modifié.</li> </ul>

Le tableau 3.8 représente la description textuelle du cas d'utilisation « Modifier un étudiant ».

#### Cas d'utilisation « Supprimer un étudiant »

TABLE III.9 – Description textuelle de cas d'utilisation « Supprimer un étudiant ».

Acteur	Agent admin.
Précondition	Agent authentifié et liste des étudiants affichée.
Post-traitement	Étudiant supprimé.
Scénario	<ul style="list-style-type: none"> <li>• L'agent sélectionne l'étudiant à supprimer.</li> <li>• Il clique sur le bouton « Supprimer ».</li> <li>• Il confirme la suppression de l'étudiant sélectionné.</li> <li>• L'étudiant est supprimé.</li> </ul>

Le tableau 3.9 représente la description textuelle du cas d'utilisation « Supprimer un étudiant ».

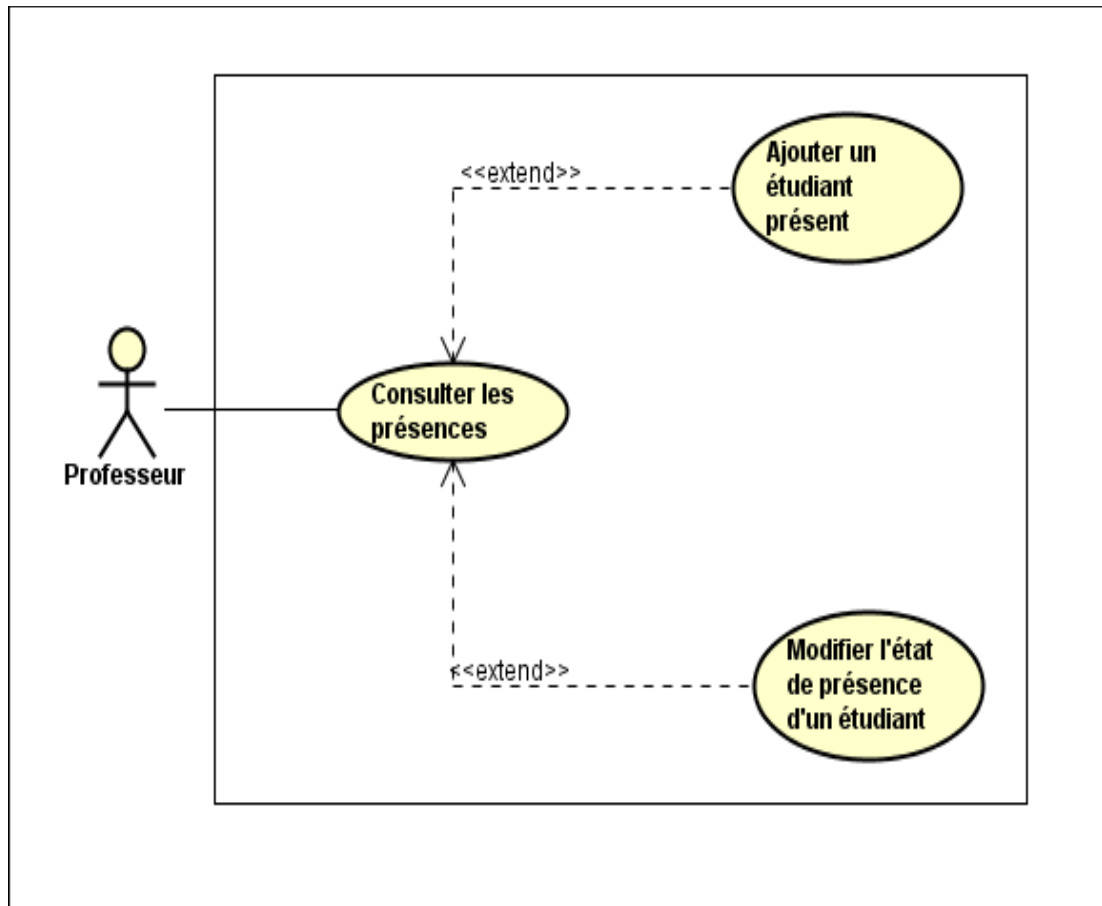
**Cas d'utilisation « Gérer les présences »**

FIGURE III.5 – Cas d'utilisation « Gérer les présences »

La figure 3.5 représente le diagramme de cas d'utilisation « Gérer les présences ». Nous allons maintenant détailler ce processus à l'aide d'une description textuelle.

#### Cas d'utilisation « Ajouter un étudiant présence »

TABLE III.10 – Description textuelle de cas d'utilisation « Ajouter un étudiant présent ».

Acteur	Professeur.
Précondition	Professeur authentifié et liste des présences affichée.
Post-traitement	Étudiant présent ajouté à la liste des présences.
Scénario	<ul style="list-style-type: none"> <li>• Le professeur clique le bouton « Ajouter ».</li> <li>• Il présente les informations nécessaires.</li> <li>• Il confirme les changements.</li> <li>• L'étudiant est ajouté à la liste des présences.</li> </ul>

Le tableau 3.10 représente la description textuelle du cas d'utilisation « Ajouter un étudiant présence ».

#### Cas d'utilisation « Modifier l'état de présence d'un étudiant »

TABLE III.11 – Description textuelle de cas d'utilisation « Modifier l'état de présence d'un étudiant »

Acteur	Professeur.
Précondition	Professeur authentifié et liste des présences affichée.
Post-traitement	État de présences de l'étudiant modifié et synchroniser avec les résultat de traitement.
Scénario	<ul style="list-style-type: none"> <li>• Le professeur sélectionne l'étudiant en question.</li> <li>• Il clique le bouton « Modifier ».</li> <li>• Il modifie les informations de présences et présente ses remarques si besoin.</li> <li>• Il confirme les changements.</li> </ul> <p>L'état de présence de l'étudiant et mis à jour.</p>

Le tableau 3.11 représente la description textuelle du cas d'utilisation « Modifier l'état de présence d'un étudiant ».

#### Cas d'utilisation « Créer un QR code »

TABLE III.12 – Description textuelle de cas d'utilisation « Crée un QR code »

Acteur	Professeur.
Précondition	Professeur authentifié.
Post-traitement	QR code généré et téléchargé sur l'ordinateur du professeur.
Scénario	<ul style="list-style-type: none"> <li>• Le professeur sélectionne la classe en question.</li> <li>• Il sélectionne aussi la matière à enseignée.</li> <li>• Il clique sur « parcourir » pour sélectionner l'emplacement de QR code généré.</li> <li>• Le QR code est créé et enregistré dans l'emplacement sélectionné.</li> </ul>

Le tableau 2.12 représente la description textuelle du cas d'utilisation « Créer un QR code ».

**Cas d'utilisation « Scanner le QR code »**

TABLE III.13 – Description textuelle de cas d'utilisation « Scanner le QR code »

Acteur	Étudiant.
Précondition	Étudiant authentifié.
Post-traitement	QR code scanné décrypté et les informations sont extraites.
Scénario	<ul style="list-style-type: none"><li>• L'étudiant clique sur le bouton « Scan » de son application mobile.</li><li>• L'application indique le succès ou l'échec de l'opération.</li><li>• Le Scan est terminé.</li></ul>

Le tableau 3.13 représente la description textuelle du cas d'utilisation « Scanner le QR code ».



## Conclusion

Au cours de ce chapitre, nous avons étudié les différents besoins de notre application pour concevoir une modélisation avec des diagrammes UML qui vont être représentés dans le chapitre suivant.

## **Chapitre IV**

### **Conception**

## Introduction

Ce chapitre sera consacré à la conception de notre application. D'abord, nous exposerons l'architecture générale de l'application. Puis, nous expliquerons la conception en plus de détails à travers son état statique, à l'aide du diagramme de classes et de son état dynamique à travers les diagrammes de séquences.

### IV.1 Conception globale

C'est l'artéfact le plus important dans le processus Scrum, puisqu'il contient l'ensemble des fonctionnalités du produit souhaité.

#### IV.1.1 Architecture générale de l'application

La figure 4.1 et 4.2 présente l'architecture générale de l'application. Elle se compose de plusieurs couches afin de minimiser le couplage et assurer la flexibilité.

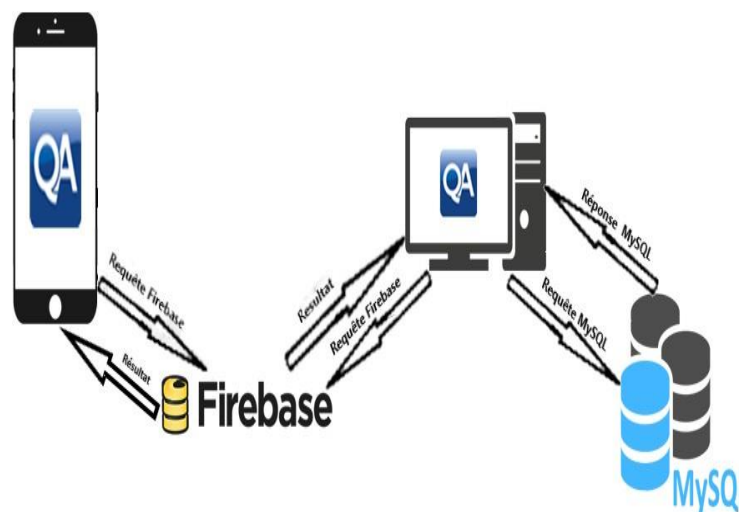


FIGURE IV.1 – Architecture de l'application Quick Attendance

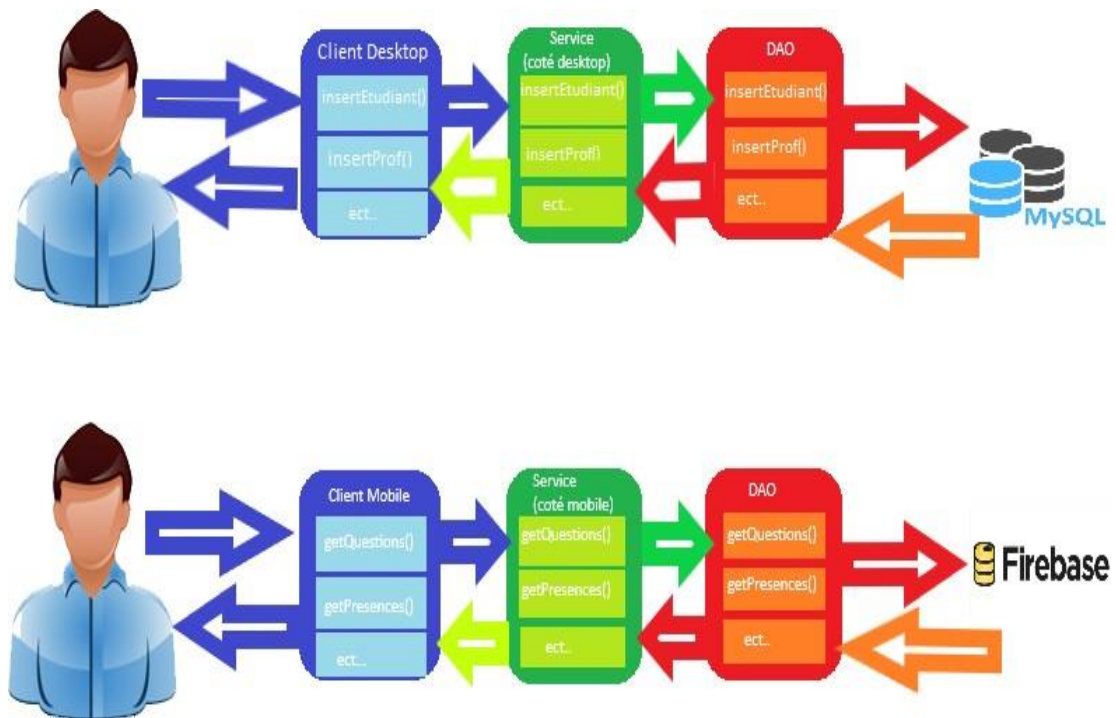


FIGURE IV.2 – Architecture de l'application Quick Attendance

Les différentes couches de l'application sont :

- 1. Client bureautique et / ou mobile : cette couche fournit à chaque utilisateur les interfaces graphiques nécessaires assurant la consultation et l'usage des services disponibles.
- 2. Serveur : il répond aux besoins de ses clients desktop et/ou mobile et leurs fournit les réponses nécessaires.
- 3. La base de données bureautique(local) : elle est responsable du stockage des données provenant de l'application desktop et/ou mobile.
- 4. La base de données mobile : elle représente une solution efficace qui permet l'utilisation de l'application en mode déconnecté « offline », les données s'envoient quand le client se connecte à internet.

#### IV.1.1.1 Architecture générale de l'application bureautique

La figure 4.3 représente l'architecture générale de l'application desktop et les couches qui la composent.



FIGURE IV.3 – Architecture de l'application bureautique

Les différentes couches de l'application bureautique sont :

- 1. Couche présentation : la couche d'interaction avec les utilisateurs, elle est responsable de l'affichage des données.
- 2. Couche service : cette couche est composée de différents services métiers de notre application.
- 3. Couche d'accès aux données (DAO) : cette couche permet de manipuler la base de données à travers la technique « Object Relationnel Mapping », qui permet la conversion des objets en données relationnelles et vice-versa.

#### IV.1.1.2 Architecture générale de l'application mobile

La figure 4.4 représente l'architecture de l'application mobile en mode déconnecté « Offline ».

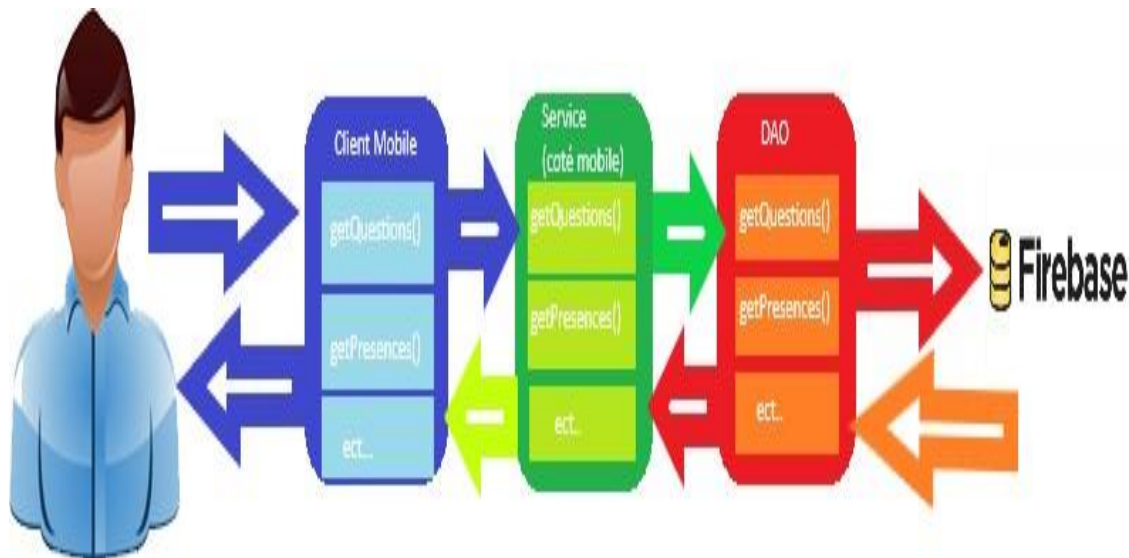


FIGURE IV.4 – Architecture de l'application mobile

En mode connecté « Online », l'application mobile admet la même architecture que l'application bureautique.

### IV.1.2 Conception détaillée

Après avoir présenté la conception de notre application d'une manière générale. Nous allons dans ce qui suit la détailler. Tout d'abord, nous allons présenter le diagramme de classes, les diagrammes de séquences et enfin la structure de donnée sur Firebase.

#### IV.1.2.1 Diagramme de classes

Le diagramme de classe est principalement considéré comme l'un des diagrammes les plus importants dans le développement orienté objet. Il représente l'architecture conceptuelle de l'application, il décrit les classes que l'application utilise ainsi que leur association, héritage, agrégation etc.. . .

La figure ci-dessous, représente le diagramme de classes.

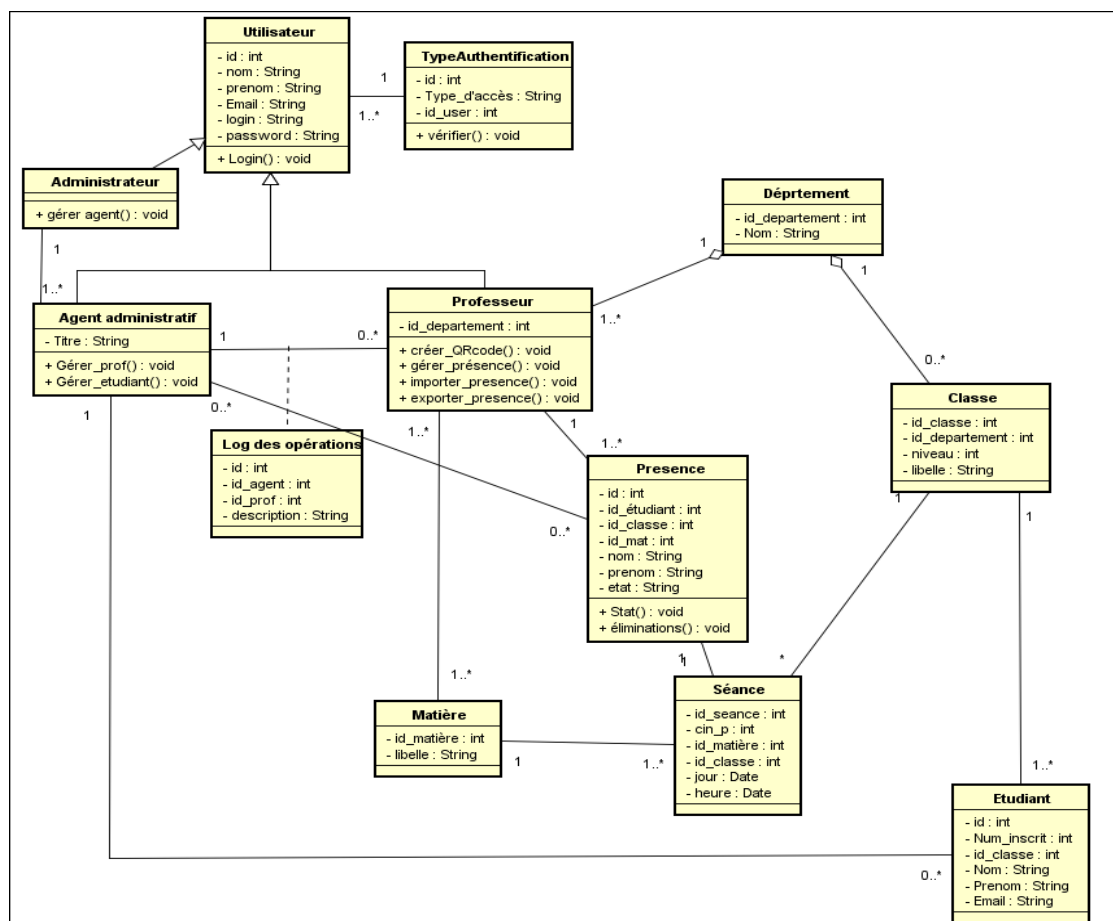


FIGURE IV.5 – Diagramme de classes

Chaque Utilisateur (Administrateur, Agent administratif, professeur) s'authentifie en utilisant son log in et mot de passe et le système va afficher la page d'accueil adéquate selon le type d'authentification (Administrateur, agent administratif, professeur).

Le professeur peut créer un QR code spécifique en fournissant la classe et la matière à enseigner, le système va enregistrer le QR code généré dans l'emplacement précisé par le professeur. Ce dernier attache l'image reçue au début de son support de cours. Le QR code à ce stade contient la moitié des informations nécessaires pour la détermination des présences des étudiants tels que le département, la classe, la matière et un code secret le reste des informations va être ajouté après le traitement mobile.

A la fin de la séance, le professeur clique sur le bouton « Synchroniser » pour recevoir la liste des étudiants avec leur état de présence (Présent ou Absent) et cette liste va automatiquement être exportée vers la base de données de la faculté. Le professeur peut aussi modifier cette liste dans des cas spécifiques (Retard, changement de temps de la séance etc..) ses changements vont être pris en charge dès qu'il les confirme. Un agent administratif peut ajouter, modifier ou supprimer un professeur et il a aussi un accès aux présences des étudiants.

Un Administrateur est le responsable sur le déroulement de l'application, il peut gérer les agents administratifs et possède le type d'accès le plus élevé.

#### **IV.1.2.2 Diagramme de séquences**

Un diagramme de séquence système est une représentation expressive d'une interaction entre les acteurs et le système. Dans cette partie, nous illustrons quelques scénarios associés à des cas d'utilisation à l'aide des diagrammes de séquences.



**Diagramme de séquence « S'authentifier » :**

L'authentification est une opération nécessaire pour garantir la sécurité de notre application d'une part et pour contrôler l'accès aux pages adéquates par chaque utilisateur, selon son rôle, d'autre part. La figure 4.6 présente le diagramme de séquence du cas d'utilisation « S'authentifier ».

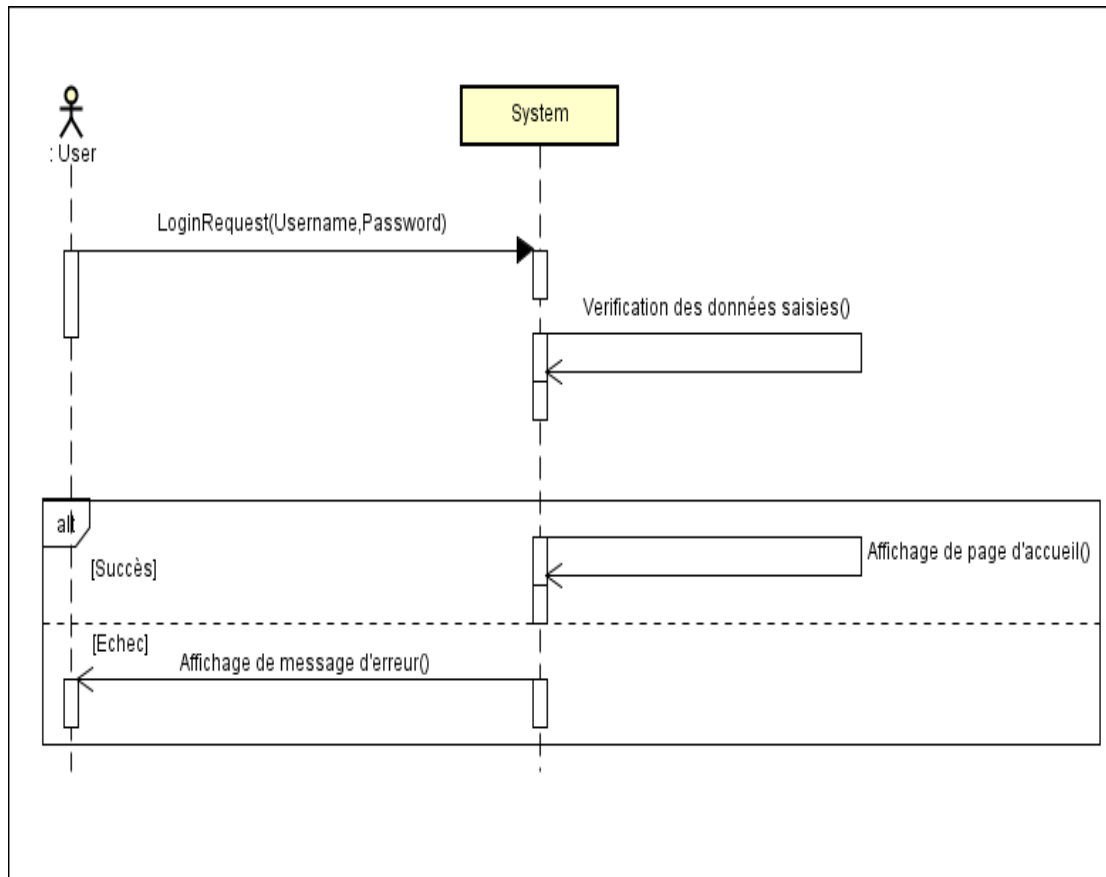


FIGURE IV.6 – Diagramme de séquence du cas d'utilisation « S'authentifier »

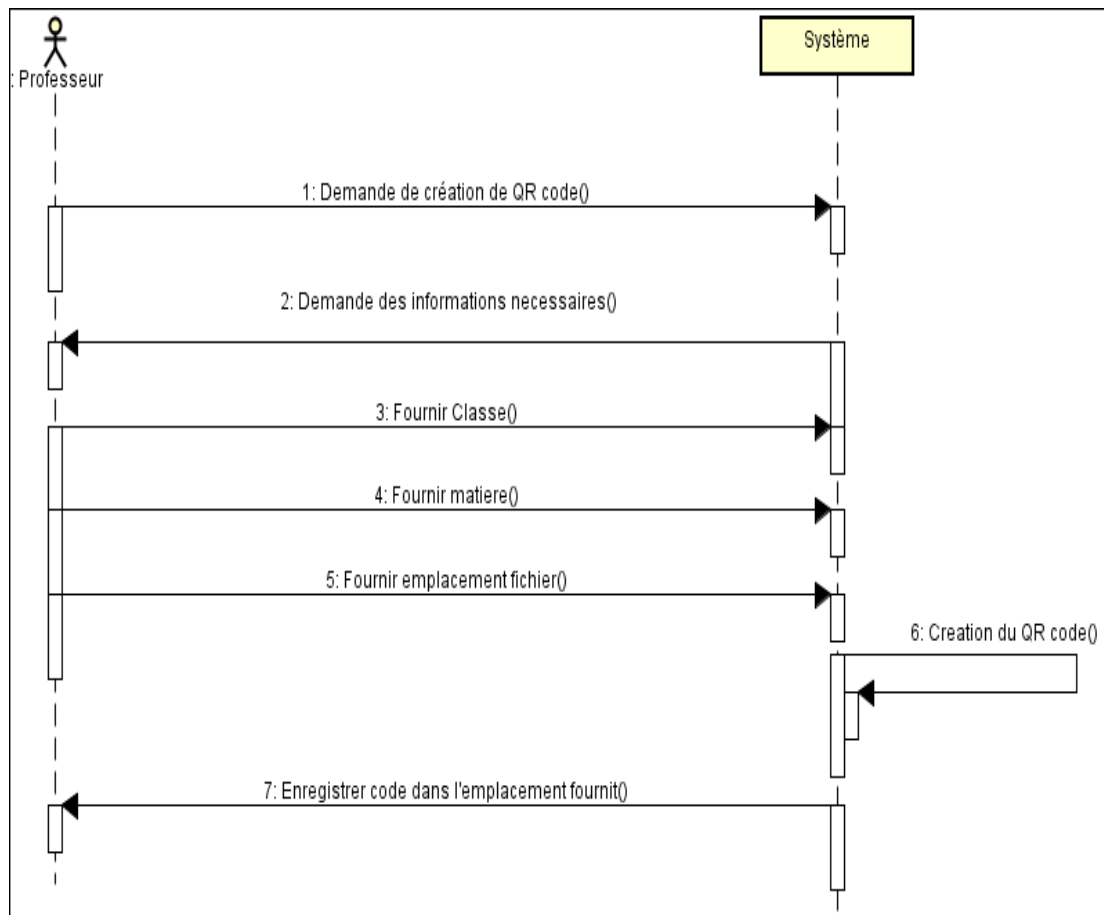
**Diagramme de séquence « Créer un QR code » :**

FIGURE IV.7 – Diagramme de séquence « Créer un QR code »

**Scénario de diagramme de séquence « Créer un QR code » :**

Acteur	Professeur.
Précondition	Professeur authentifié.
Post-traitement	QR code créer et enregistrer dans l’emplacement demandé.
Scénario	<ul style="list-style-type: none"> <li>• Le professeur navigue vers le frame de création de QR code.</li> <li>• Des « combo-box » à modifier vont être afficher pour le professeur.</li> <li>• Le professeur les remplit par les informations adéquates.</li> </ul> <p>Le système crée le QR code et l’enregistre dans l’emplacement demandé.</p>

TABLE IV.1 – Scénario de diagramme de séquence « Créer un QR code »

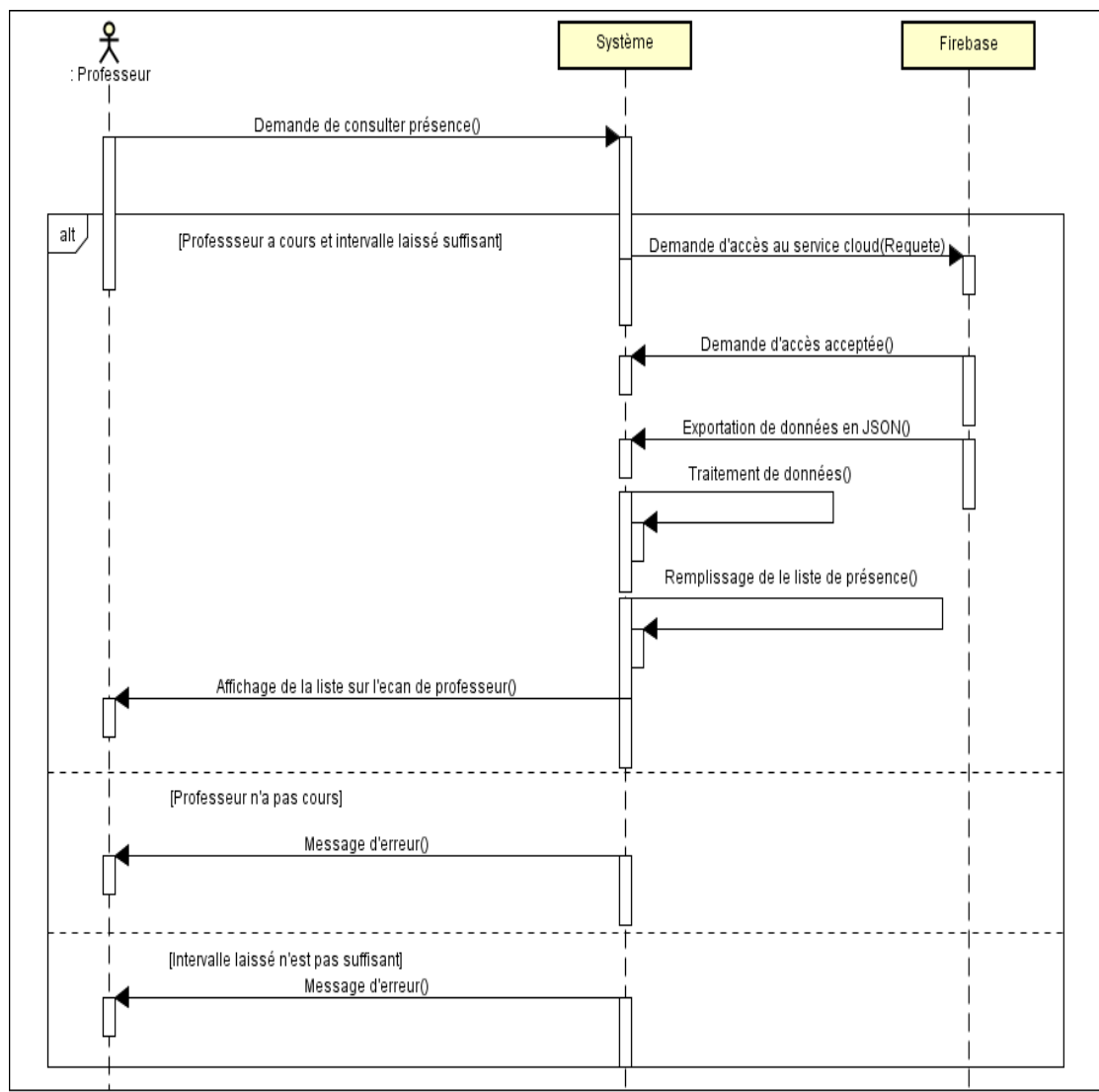
**Diagramme de séquence « Consulter présences » :**

FIGURE IV.8 – Diagramme de séquence « Consulter présences »

**Scénario de diagramme de séquence « Consulter présences » :**

Acteur	Professeur.
Précondition	Professeur authentifié.
Post-traitement	Liste des présences affichée.
Scénario	<ul style="list-style-type: none"> <li>• Le professeur navigue vers le frame de consultation de présence.</li> <li>• Il clique sur le bouton « Synchroniser ».</li> <li>• Le système se connecte à Firebase et demande les informations nécessaires</li> </ul> <p>Le système affiche la liste sur l'écran de professeur.</p>
Exception	<ul style="list-style-type: none"> <li>• Le professeur n'a pas cours l'heure de la demande.</li> <li>• Le professeur n'a pas laissé un intervalle de temps avant la demande.</li> </ul>

TABLE IV.2 – Scénario de diagramme de séquence « Consulter présences »

### IV.1.2.3 Représentation de la structure de données dans Firebase

Vue la simplicité du fonctionnement du côté mobile de notre application et que nous avons besoin d'un traitement avec une performance élevée et un déroulement des requêtes en temps réel, Firebase Database et le choix du NoSQL ici semble adéquat.

Les données sur Firebase sont stockées sous formes d'objet JSON ou encore un arbre JSON, les informations ajoutées à cet arbre deviennent de nouveaux nœuds portant des clés associées automatiquement par une interface de programmation applicative (l'API) ou manuellement choisies par l'administrateur de la base de données. La figure ci-dessous va illustrer la structure de notre base de données :

```
1  {
2    "Users" : {
3      "Nom_Departement_ici" : {
4        "Nom_Classe_ici" : {
5          "Id_etudiant_ici" : {
6            "FirstName" : "//Prenom",
7            "Historique" : {
8              "-LDeP-hcpr0Q3xTW-MrK" : "//Presences1",
9              "-LjdnedJLEODED12564D" : "//Presences2",
10             "-LdeJCH548PODGHyedfH" : "//Presences3",
11           },
12           "LastName" : "//Nom",
13           "Num_ins" : "//Numero d'inscription",
14           "Presences" : "//Dernière Presence Stockée"
15         }
16       }
17     }
18   }
19 }
```

FIGURE IV.9 – Représentation de la structure de données dans Firebase

**Users :** Considéré comme la racine de notre document JSON.

**Département :** Le nom de département en question.

**Classe :** Le nom de la classe en question.

**ID étudiant :** En réalité va être la CIN de l'étudiant.

**Historique :** stocke toutes les présences pour des statistiques

**Présences :** stocke les données du dernier «Scan» de cet étudiant et joue un rôle important dans la détermination de l'état de présence de l'étudiant.

**FirstName, LastName, NumIns :** Informations sur l'étudiant.

## Conclusion

Dans ce chapitre, nous avons étudié la conception de notre application. Nous avons présenté l'architecture générale des applications desktop et mobile. Et enfin nous avons abouti à la représentation de quelques diagrammes afin de détailler la conception proposée. Dans le chapitre qui suit, nous expliciterons la phase de réalisation.

## **Chapitre V**

### **Réalisation**



## Introduction

Après avoir finalisé l'étape de la conception, nous décrivons dans ce chapitre l'implémentation de notre application.

Cette étape est assez importante et consistante tant pour nous que pour la faculté. Car c'est à terme de cette étape que nous pouvons voir le fruit de notre travail en phase d'exploitation dans la société.

Nous passons à la réalisation de notre projet. Au niveau de ce chapitre, nous argumentons notre choix matériel et logiciel en présentant l'architecture du système ainsi que les technologies utilisées pour l'implémentation de l'application.

## V.1 Environnement de travail

### V.1.1 Environnement matériel

Ce projet a été réalisé sur **un ordinateur Lenovo Légion Y520** et un **smartphone HUAWEI Y6 II** ayant les caractéristiques suivantes :

#### Lenovo Légion Y520 :

- **Processeur** : Intel Core i7 7700HQ.
- **Mémoire** : 8 Go DDR 4.
- **Graphisme** : NVIDIA GeForce GTX 1050 Ti.
- **Système d'exploitation** : Windows 10 pro 2017.

#### HUAWEI Y6 II :

- **Processeur** : HiSilicon Kirin 620- 1.2 GHz.
- **Mémoire** : 2 Go.
- **Graphisme** : ARM Mali-450 MP4
- **Système d'exploitation** : Android 6.0 Marshmallow.

### V.1.2 Environnement logiciel et choix technologique

#### V.1.2.1 Logiciels utilisés

Dans cette section vous trouvez les différents logiciels utilisés pour mettre en œuvre l'application.

**NetBeans :**

NetBeans est un environnement de développement (EDI) open source, disponible sur différents systèmes d'exploitation (Windows, Linux, Solaris, Mac OS x). Il permet la prise en charge native de divers langages tels que C, C++, JavaScript, XML, Groovy, le PHP et le HTML. NetBeans nous a permis de développer notre application grâce à son outil de « Build » pour les projets Java et son interface de programmation applicative (API) JavaFX. Fonctions principales :

- Modification de code assistée.
- Import rapide des classes Java.
- Debugging intégré.
- Test et intégration continus.
- Editeur de texte XML, DTD et CSS.
- Accessibilité pour tout l'IDE, Plateforme ou applicatif.

**Android Studio :**

Android studio [5] est l'environnement de développement intégré « IDE » officiel dédié aux développeurs des applications sous la plate-forme Android, cet « IDE » extensible et polyvalent offre bien de fonctionnalités et d'outils pour améliorer le processus de développement des applications Android. Ce dernier offre un système de « Build » basé Gradle, un émulateur riche en options et l'Android SDK [6] (Software Development

Kit) qui est un ensemble d'outils de développement utilisés pour développer des applications Android. Le SDK Android comprend : des bibliothèques requises, un debugger, un émulateur, une documentation utile pour les interfaces de programme d'application Android (API), un échantillon de code sources et des tutoriels pour le système d'exploitation Android.

### **MySQL Workbench :**



MySQL Workbench est un logiciel qui permet la gestion et l'administration de bases de données. Grâce à son interface graphique, ce dernier permet la création, la modification et la suppression des tables, des comptes utilisateurs, et d'effectuer toutes les opérations essentielles à la gestion d'une base de données.

#### **Fonctions principales :**

- Visualisation graphique des performances.
- Sauvegarde et restauration de données.
- Transfert entre SGBD.

### **WampServer :**



WampServer (WAMP) n'est pas un logiciel en lui-même, mais un environnement qui comporte trois serveurs (Apache, MySQL et Maria DB), un interpréteur de script (PHP) et enfin phpMyAdmin pour l'administration web des bases MySQL. WAMP est une plateforme disposant d'une interface d'administration permettant de gérer et d'administrer ses serveurs à travers d'un tray icon. Il permet aussi de faire fonctionner des scripts PHP en local (sans avoir à se connecter à internet ou à un serveur externe).

**Firebase :**

Firebase[7] est un groupe de services d'hébergement pour tout type d'application ( Android , iOS, Javascript, Node.js, Java , ... ) . Il propose d'héberger en NoSQL et en temps réel des bases de données, du contenu, de l'authentification sociale (Google, Facebook, Twitter et GitHub), et des notifications, ou encore des services, tel que par exemple un serveur de communication en temps réel. Lancé en 2011 sous le nom d'Envolve, par Andrew Lee et par James Templin, le service est racheté par Google en octobre 2014. Il appartient aujourd'hui à la maison mère de Google : Alphabet. Toute l'implémentation et la gestion serveur de Firebase est à la charge exclusive de la société Alphabet. Les applications qui utilisent Firebase intègrent une bibliothèque qui permet les diverses interactions possibles.

**SceneBuilder :**

SceneBuilder est un outil de présentation visuelle qui permet aux utilisateurs de concevoir rapidement des interfaces d'application JavaFX, sans avoir à coder. Les utilisateurs peuvent glisser et déposer des composants de l'interface dans une zone de travail, modifier leurs propriétés, appliquer des feuilles de style et le code FXML de la mise en page qu'ils créent est automatiquement généré en arrière-plan. Le résultat est un fichier FXML qui peut ensuite être combiné avec un projet Java en liant l'interface utilisateur à la logique de l'application.

**Git :**

Git est un logiciel de gestion de version décentralisé, qui est très pratique pour le stockage de projet en cours de travail. Il est très précis en termes de mise à jour d'un projet il offre aux utilisateurs une multitude de commandes permettant l'implémentation des dernières modifications sur un projet en cours de réalisation, ce qui représente un gain de temps extrêmement important.

**Adobe Photoshop :**

Édité par Adobe, Photoshop est un logiciel qui a révolutionné la photographie grâce à ses fonctionnalités de dessin, traitement et de retouche d'image assisté par l'ordinateur. Il est principalement utilisé pour le traitement de photographies numériques, mais sert également à la création d'images.

**Adobe After effects :**

Édité par Adobe, After effects est un logiciel, à la base, de montage vidéo qui est devenu ensuite un outil de composition et d'effets visuels. Ce dernier permet la création d'effets spéciaux et d'animations graphiques pour tous supports et à partir de n'importe quel type de sources.

**Maven :**



Maven [8] est un outil open-source de « Build » pour les projets Java populaire, conçu pour supprimer les tâches difficiles du processus de Build. Maven utilise une approche déclarative, où le contenu et la structure du projet sont décrits, plutôt qu'une approche par tâche utilisée par exemple par Ant ou les fichiers « make » traditionnels. Cela aide à mettre en place des standards de développements au niveau d'une société et réduit le temps nécessaire pour écrire et maintenir les scripts de build. Le cœur d'un projet Maven 4 est le modèle objet projet (appelé POM pour Project Object Model). Il contient une description détaillée de votre projet, avec en particulier des informations concernant le versionnage et la gestion des configurations, les dépendances, les ressources de l'application, les tests, les membres de l'équipe, la structure et bien plus encore. Le POM prend la forme d'un fichier XML (pom.xml) qui est placé dans le répertoire de base de votre projet.

**Astah :**



Astah est un outil de conception UML, son nom provient de de l'acronyme JAVA et UML.

### V.1.2.2 Choix technologiques

#### Choix technologique de l'application bureautique :

##### Firestore SDK admin :

Avec le SDK Admin [9], vous pouvez lire et écrire des données dans la base de données en temps réel avec des privilèges d'administrateur complets, ou avec des privilèges limités à granularité plus fine.

##### ZXING :

ZXing « Zebra Crossing » est une bibliothèque de traitement d'image de codes-barres et de QR code implémentée en Java, avec des ports vers d'autres langages. Il prend en charge les produits 1D, 1D industriel et 2D.

##### JavaFX :

JavaFX est une famille de produits et de technologies de Sun Microsystems qui appartient à Oracle. Actuellement JavaFX est constitué de JavaFX Script et de JavaFX Mobile, bien que d'autres produits soient prévus. Les produits JavaFX ont pour but de créer des applications internet riches (RIA).

##### JavaMail :

JavaMail [10] est une API qui permet d'utiliser le courrier électronique (e-mail) dans une application écrite en Java (application cliente, applet, servlet, EJB, ...). Son but est d'être facile à utiliser, de fournir une souplesse qui permette de la faire évoluer et de rester le plus indépendant possible des protocoles utilisés

##### JavaSE :

JavaSE (Java Platform, Standard Edition) Environnement d'exploitation Java installé sur l'ordinateur de l'utilisateur. Il contient le moteur d'exécution Java Virtual Machine (JVM) et les routines logicielles nécessaires à l'exécution de programmes Java sur un poste de travail.

##### CSS3 :

Les feuilles de style en cascade, généralement appelées CSS3 [11] de l'anglais Cascading Style Sheets, est un type de fichier permettant de définir la présentation des documents HTML/XML.

#### Choix technologique de l'application mobile :

##### FireBase authentication :

Avec Firebase authentication [12], les utilisateurs se connectent à l'application à l'aide d'un système qu'ils connaissent déjà et qu'ils utilisent en toute confiance. L'application peut ainsi enregistrer les données de l'utilisateur de manière sécurisée dans le cloud, pour que ce dernier bénéficie d'une même expérience personnalisée sur tous ses appareils.

##### XML :

Nous avons utilisé le XML afin de créer les interfaces graphiques de l'application Android.

### V.1.3 Application bureautique

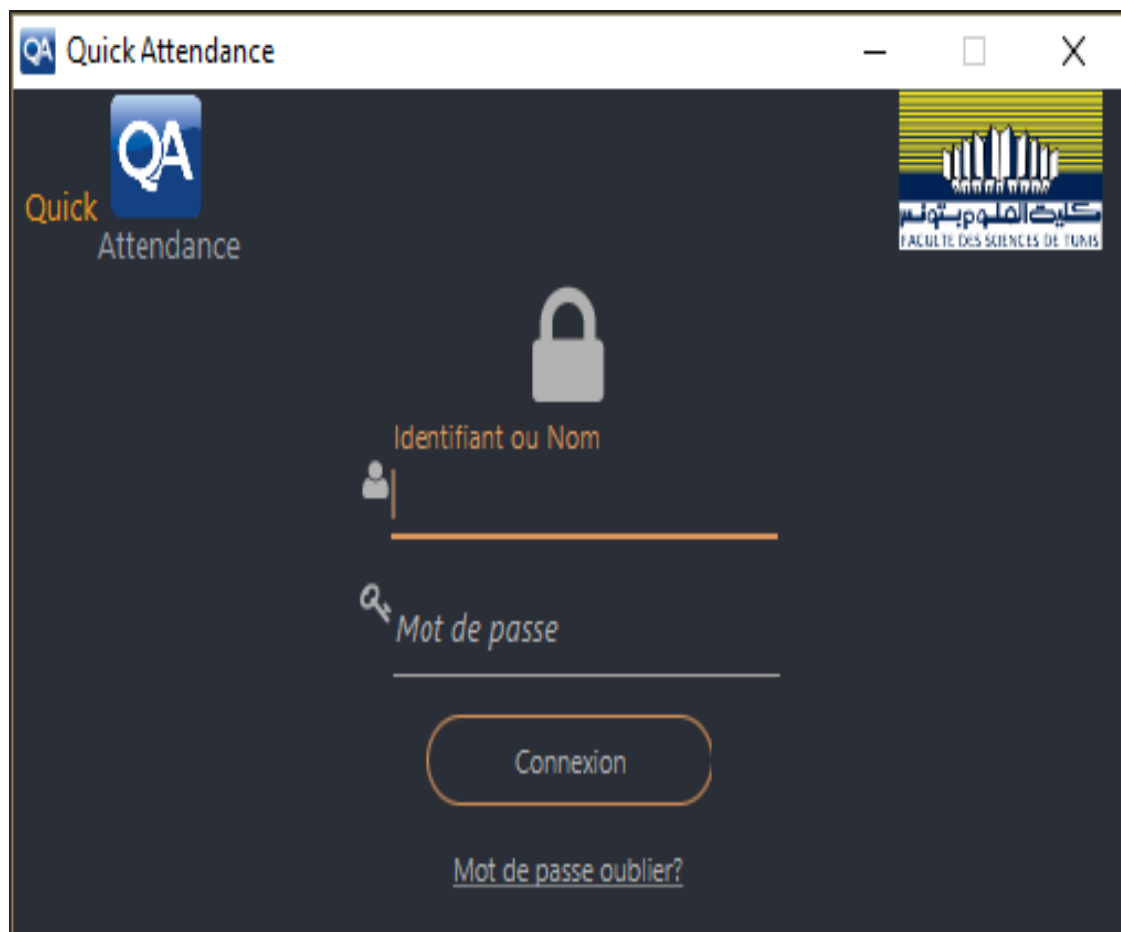


FIGURE V.1 – Présentation de l'interface de l'authentification.

La figure V.1 représente l'interface de l'authentification.



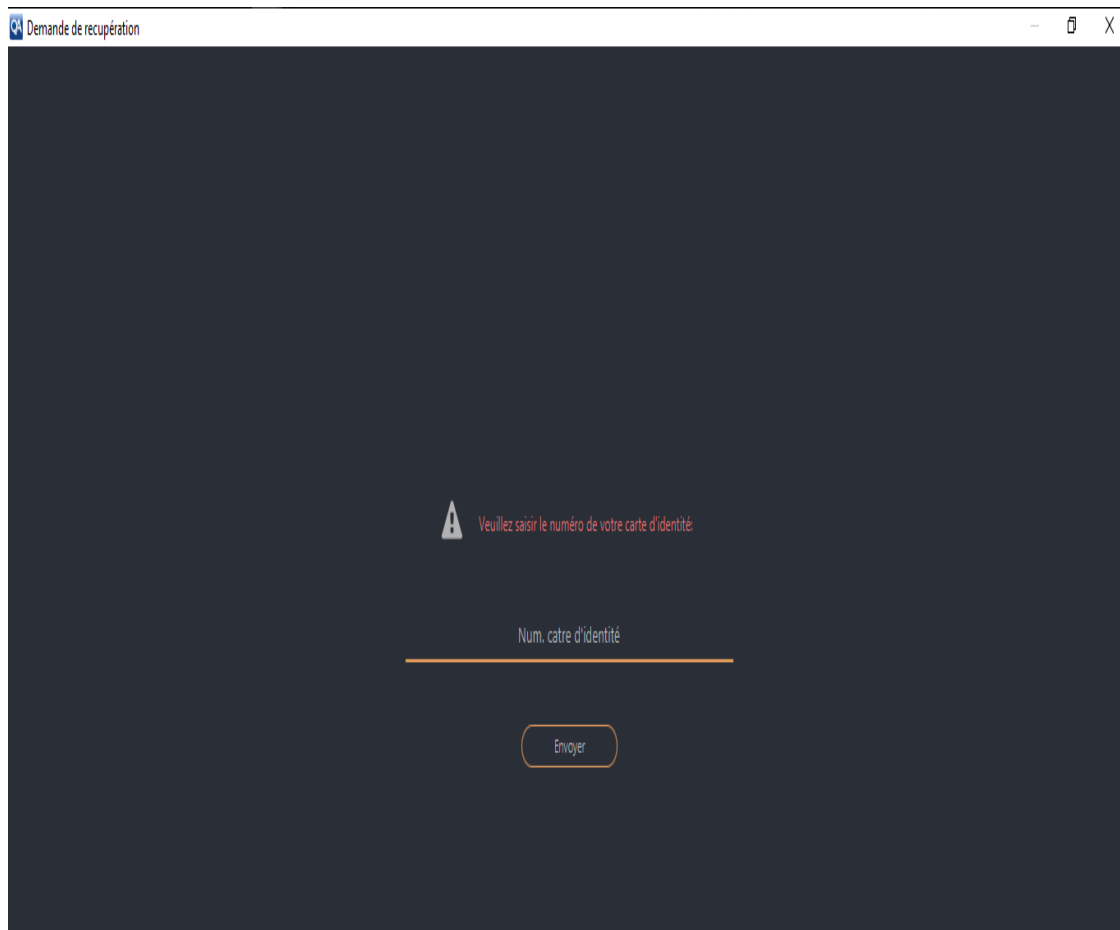


FIGURE V.2 – Présentation de l'interface de la récupération de mot de passe oublié

La figure V.2 représente l'interface de la récupération de mot de passe oublié.

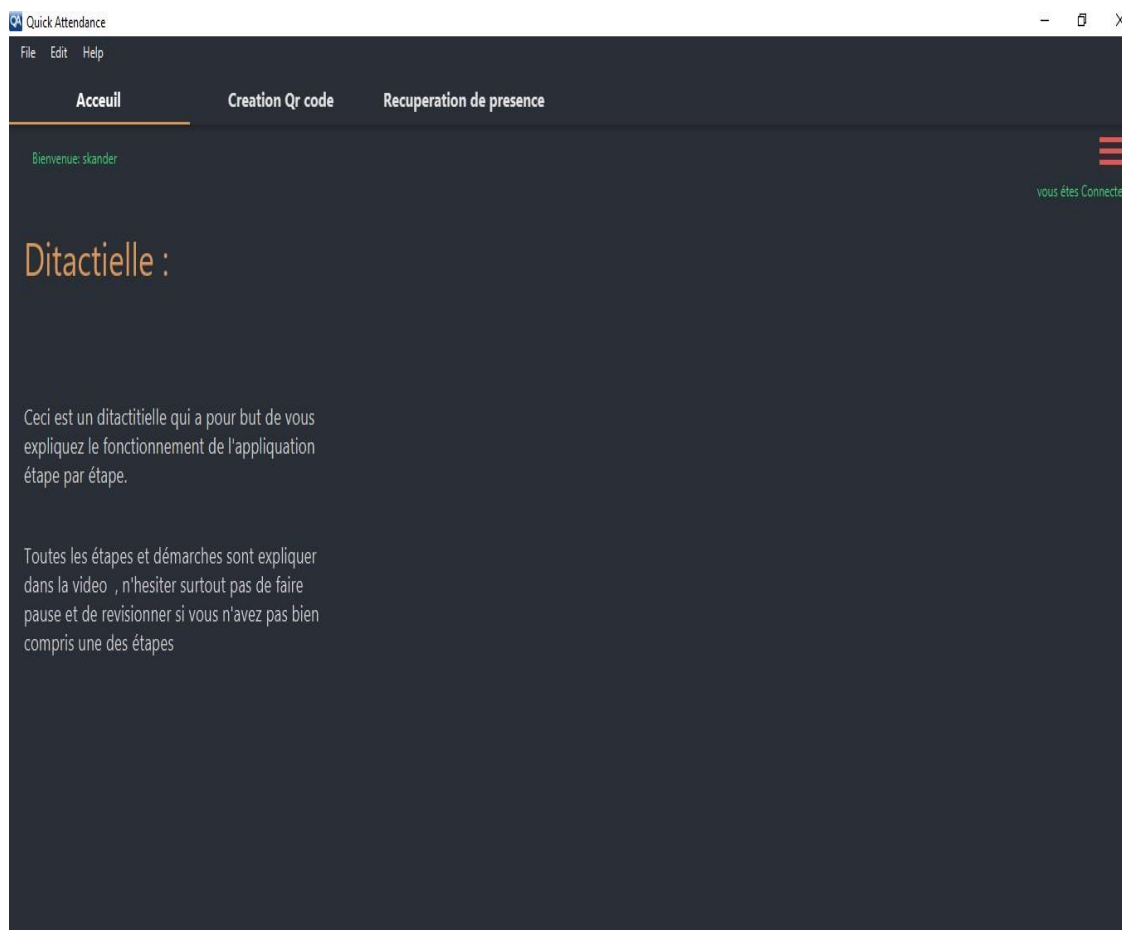


FIGURE V.3 – Présentation de l'interface de l'accueil du professeur

La figure V.3 représente l'interface de l'accueil du professeur.

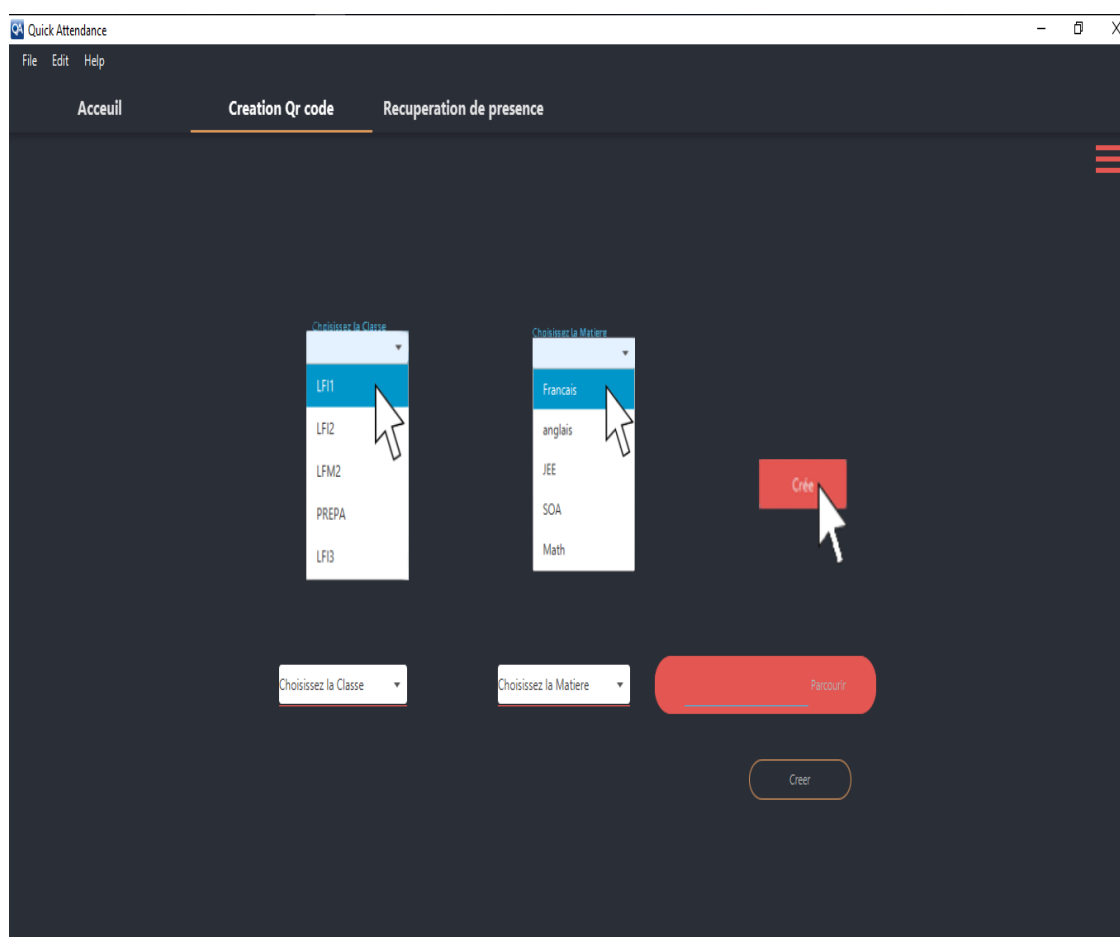


FIGURE V.4 – Présentation de l'interface du générateur de Qr code pour le professeur

La figure V.4 représente l'interface du générateur de Qr code pour le professeur.

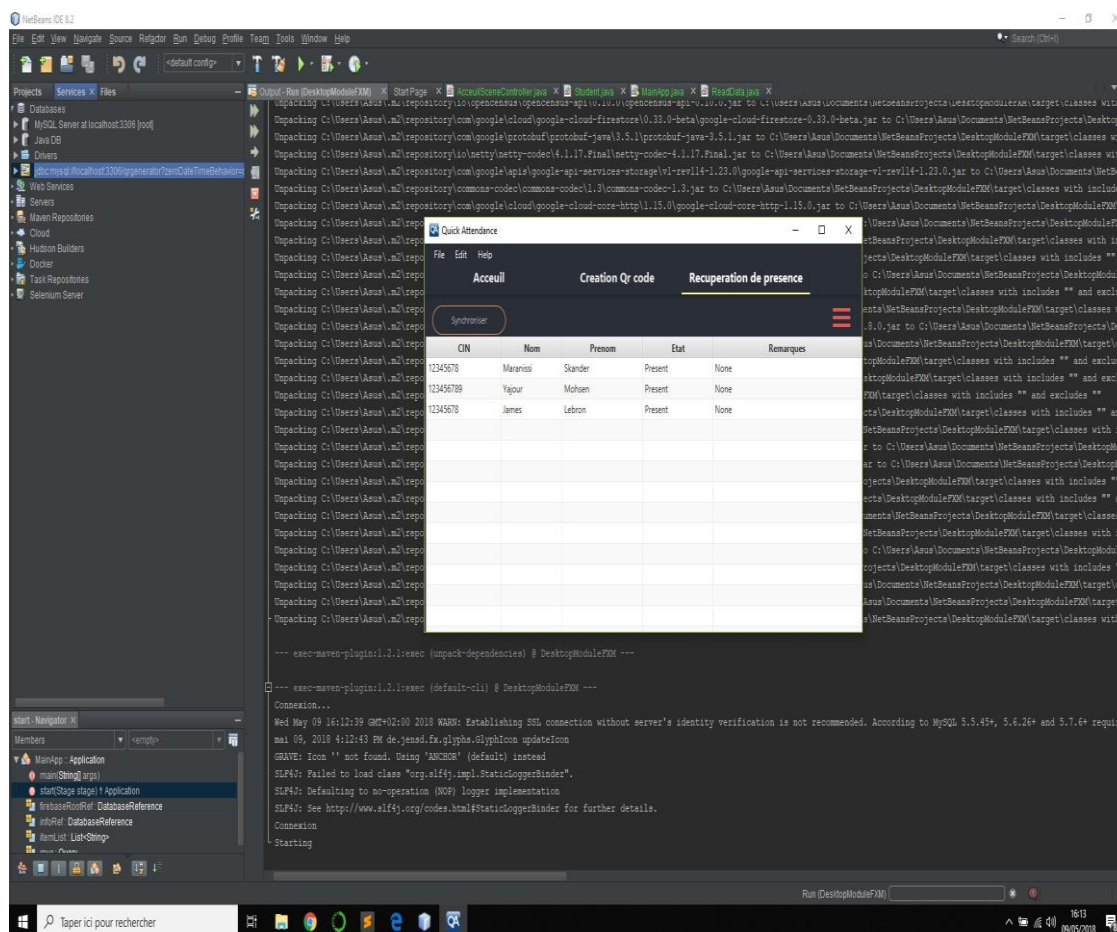


FIGURE V.5 – Présentation de l'interface de l'affichage des présences dans la séance

La figure V.5 représente l'interface de l'affichage des présences dans la séance.

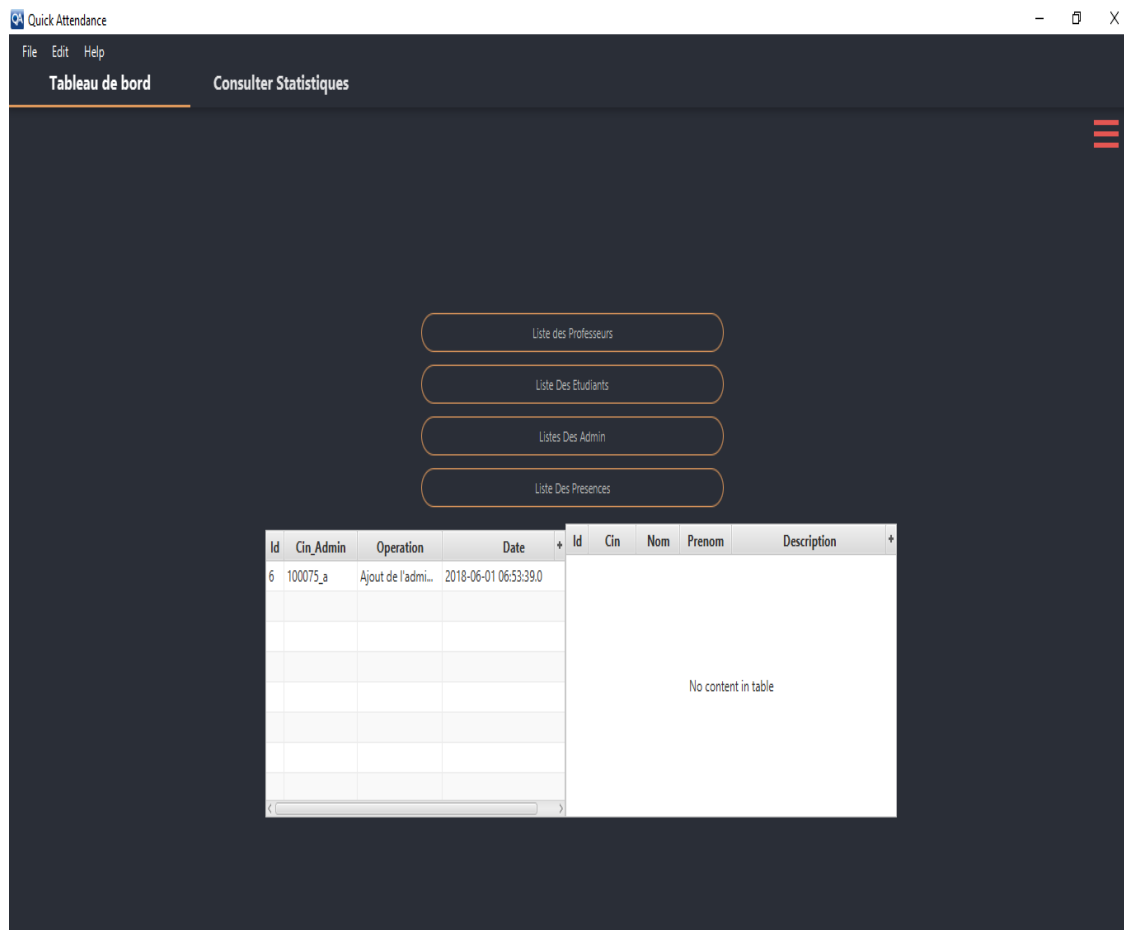


FIGURE V.6 – Présentation de l'interface de l'accueil de l'agent administratif

La figure V.6 représente l'interface de l'accueil de l'agent administratif.

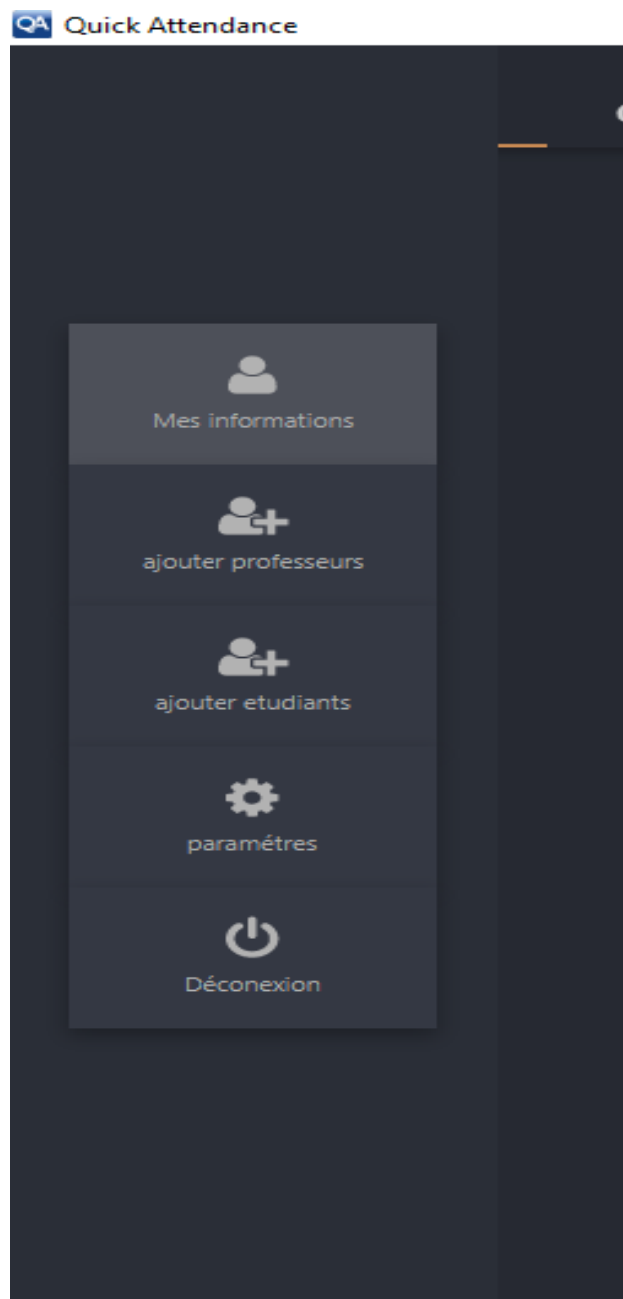
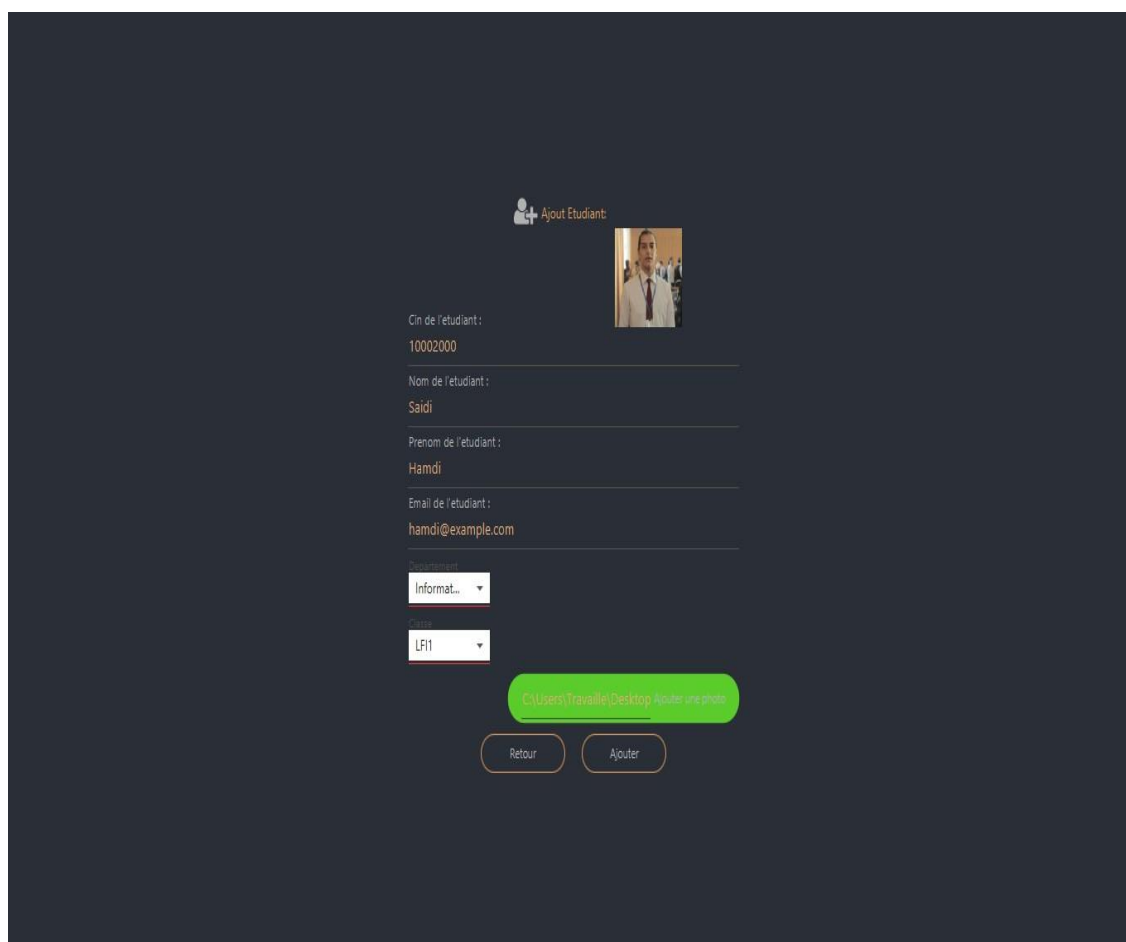


FIGURE V.7 – Présentation de l'interface de la barre d'options de l'agent administratif

La figure V.7 représente l'interface de la barre d'options de l'agent administratif.



Ajout Etudiant

Cin de l'etudiant :  
10002000

Nom de l'etudiant :  
Saidi

Prenom de l'etudiant :  
Hamdi

Email de l'etudiant :  
hamdi@example.com

Diplome  
Informat...

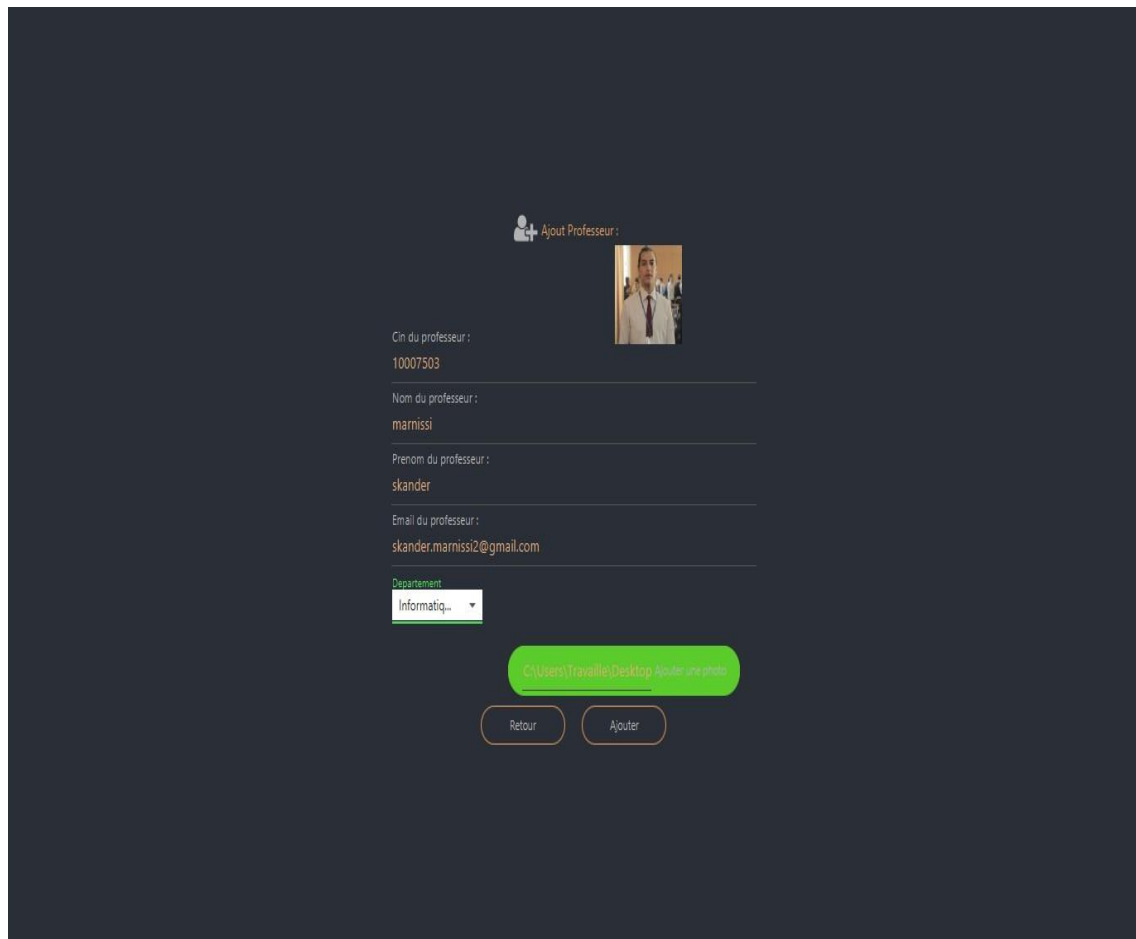
Matiere  
LFI1

C:\Users\Travail\Desktop Ajouter une photo

Retour Ajouter

FIGURE V.8 – Présentation de l'interface de l'ajout d'un étudiant par l'agent administratif

La figure V.8 représente l'interface de l'ajout d'un étudiant par l'agent administratif.



Ajout Professeur :

Cin du professeur :  
10007503

Nom du professeur :  
marnissi

Prénom du professeur :  
skander

Email du professeur :  
skander.marnissi2@gmail.com

Département :  
Informatiq...

C:\Users\Travail\Desktop Ajouter une photo

Retour Ajouter

FIGURE V.9 – Présentation de l'interface de l'ajout d'un professeur par l'agent administratif

La figure V.9 représente l'interface de l'ajout d'un professeur par l'agent administratif.



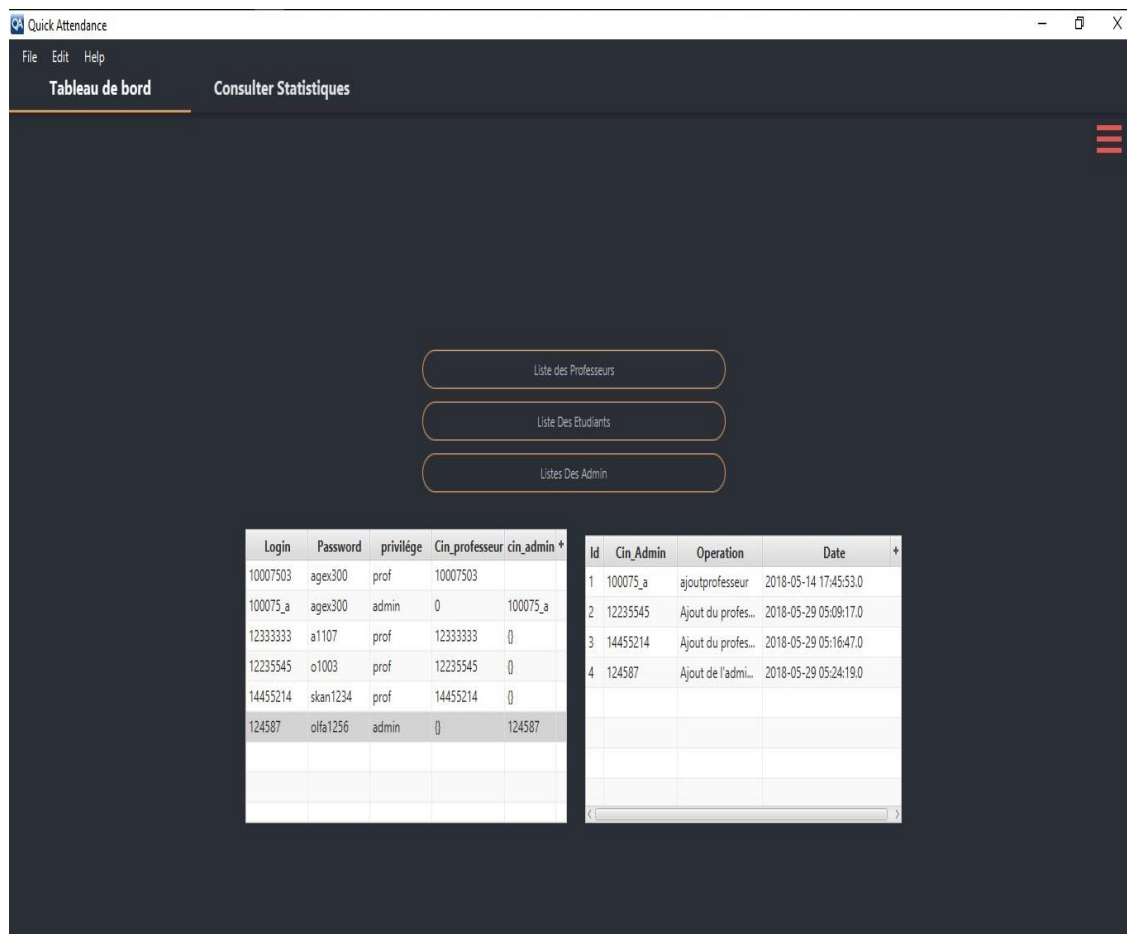


FIGURE V.10 – Présentation de l'interface de l'accueil de l'administrateur

La figure V.10 représente l'interface de l'accueil de l'administrateur.

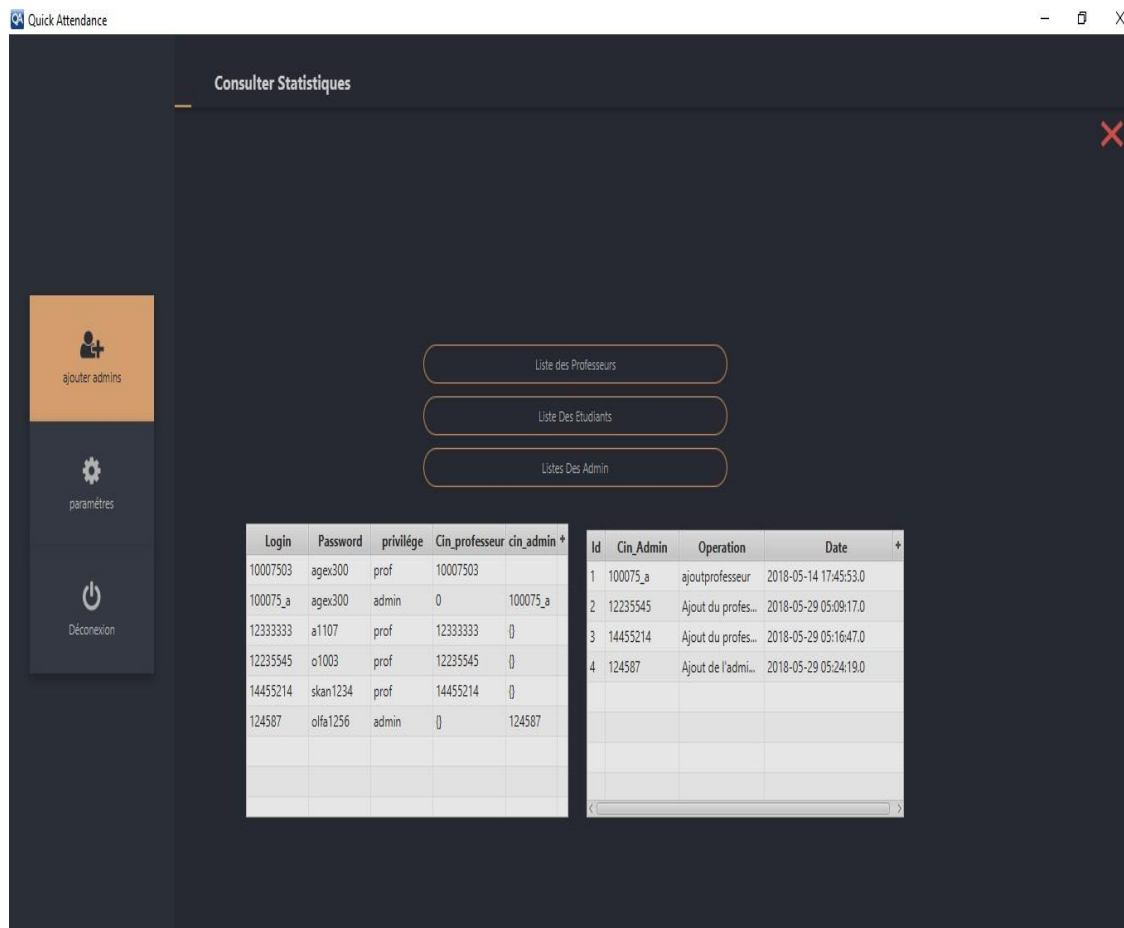
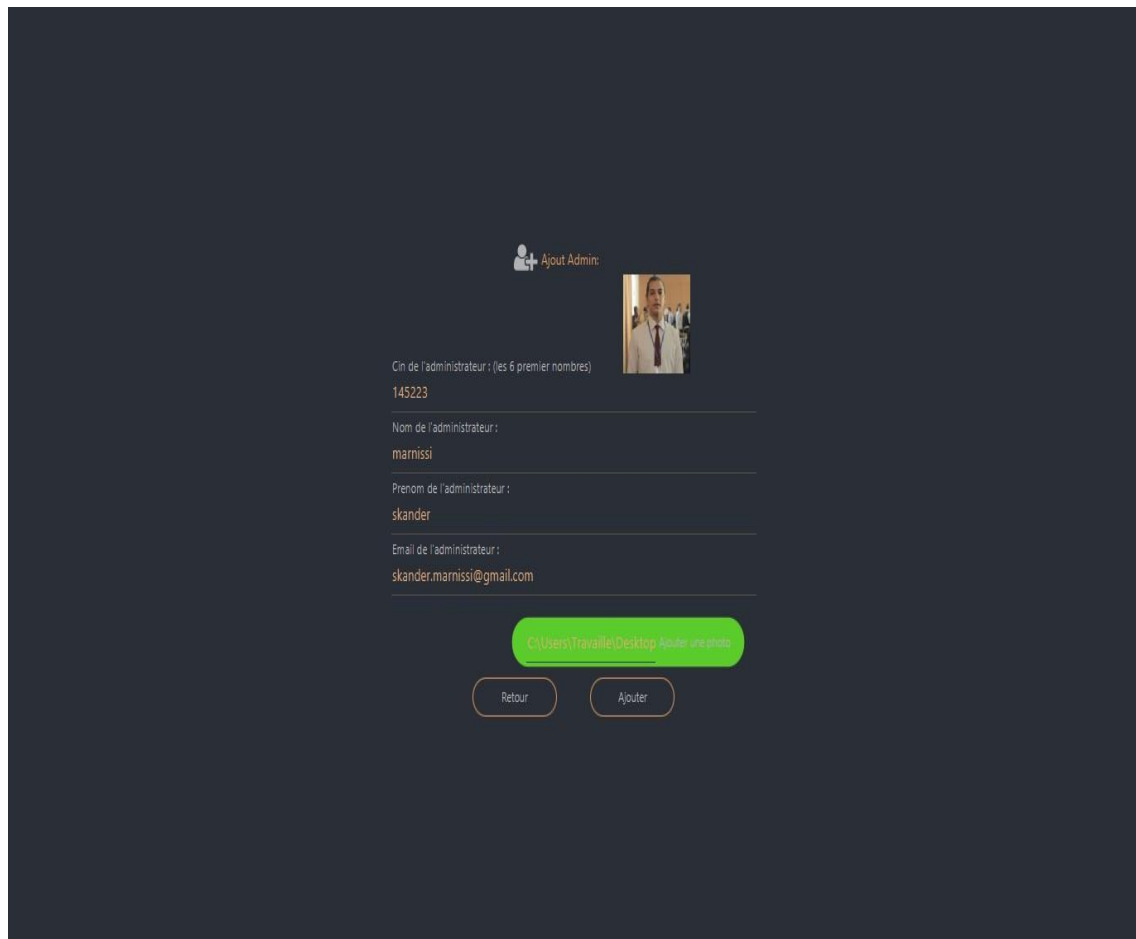


FIGURE V.11 – Présentation de l'interface de la barre d'options de l'administrateur

La figure V.11 représente l'interface de la barre d'options de l'administrateur.



Ajout Admin

Cin de l'administrateur : (les 6 premier nombres)  
145223

Nom de l'administrateur :  
marnissi

Prenom de l'administrateur :  
skander

Email de l'administrateur :  
skander.marnissi@gmail.com

C:\Users\Travail\Desktop Ajouter une photo

Retour Ajouter

FIGURE V.12 – Présentation de l'interface de l'ajout d'un agent administratif par l'administrateur

La figure v.12 représente l'interface de l'ajout d'un agent administratif par l'administrateur.

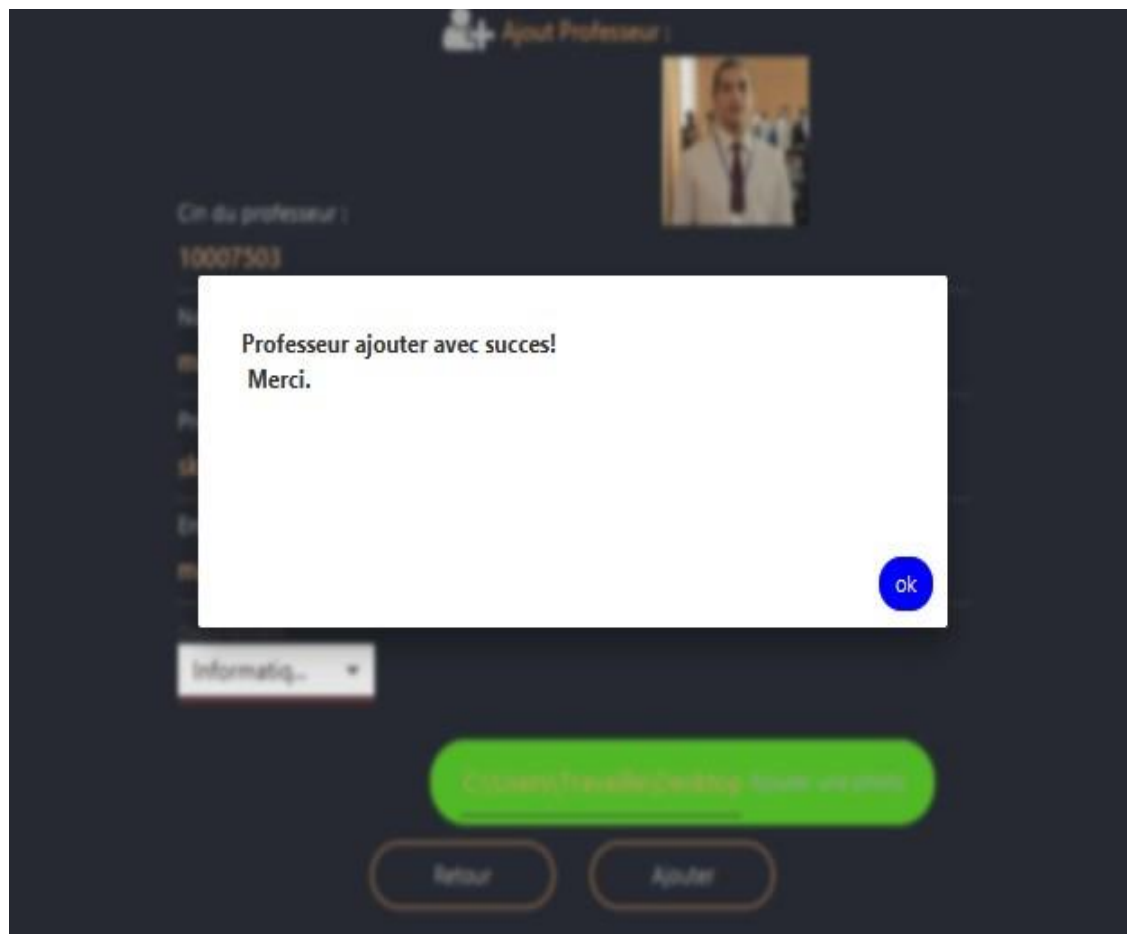


FIGURE V.13 – Présentation de l'interface de la confirmation après chaque ajout

La figure v.13 représente l'interface de la confirmation après chaque ajout.

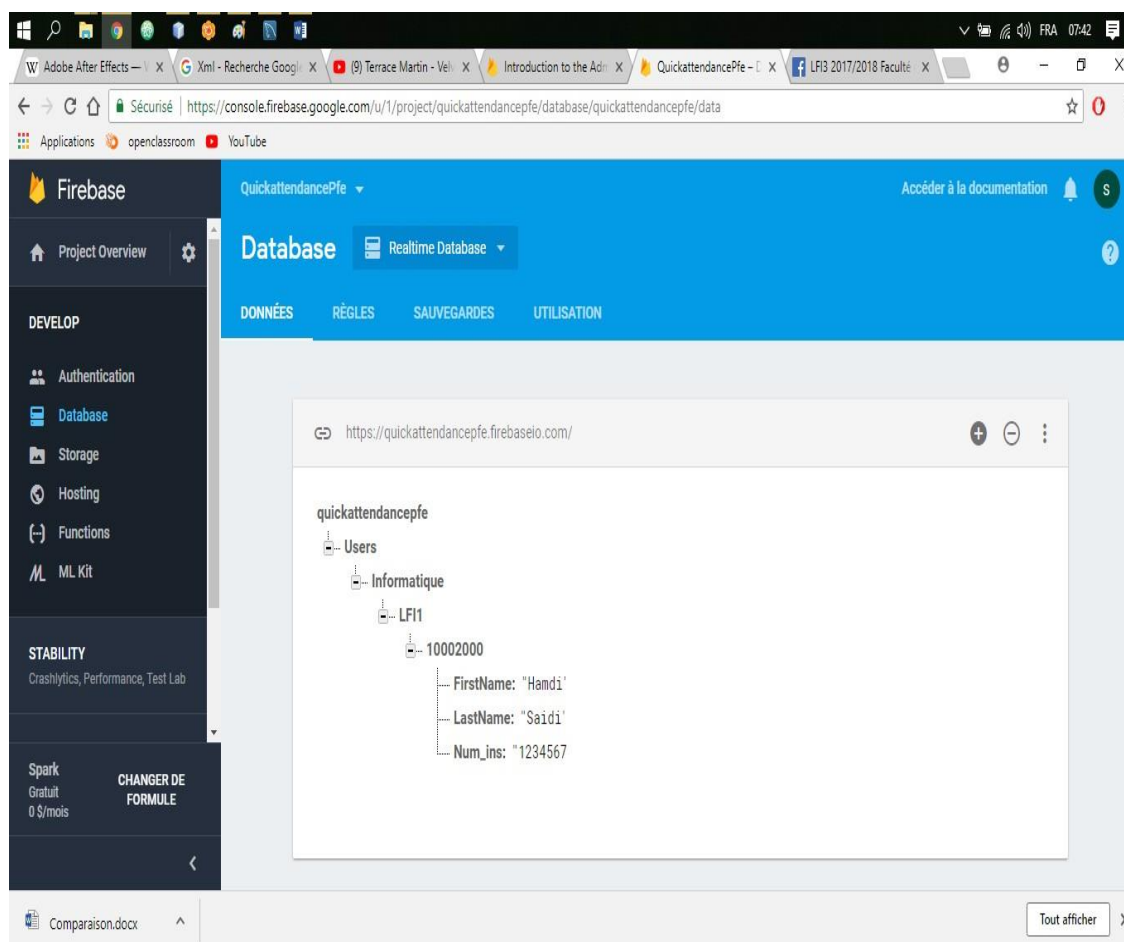


FIGURE V.14 – Présentation de l’interface de l’ajout de l’étudiant sur Firebase

La figure V.14 représente l’ajout de l’étudiant sur Firebase.

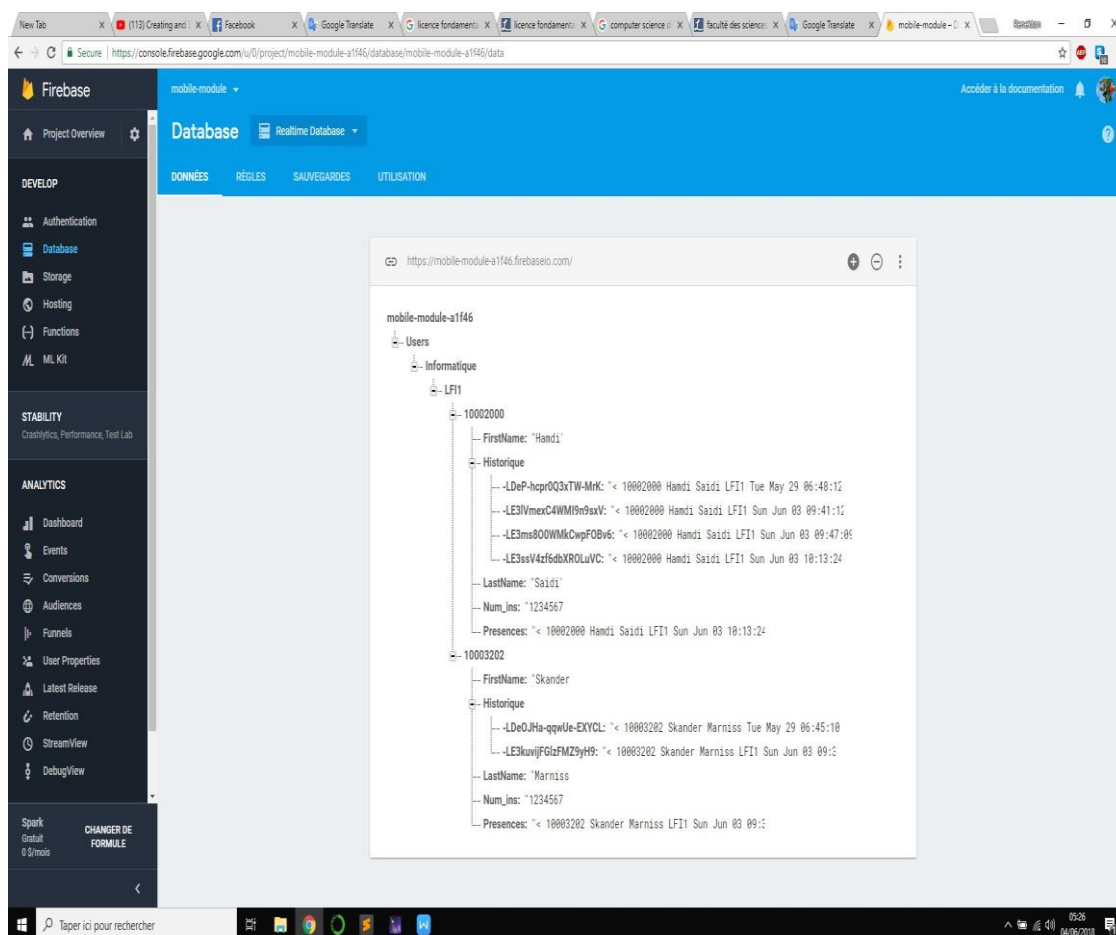


FIGURE V.15 – Présentation de l'interface de l'ajout de la présence après le scan sur Firebase

La figure V.15 représente l'ajout de l'étudiant de la présence après le scan sur Firebase.

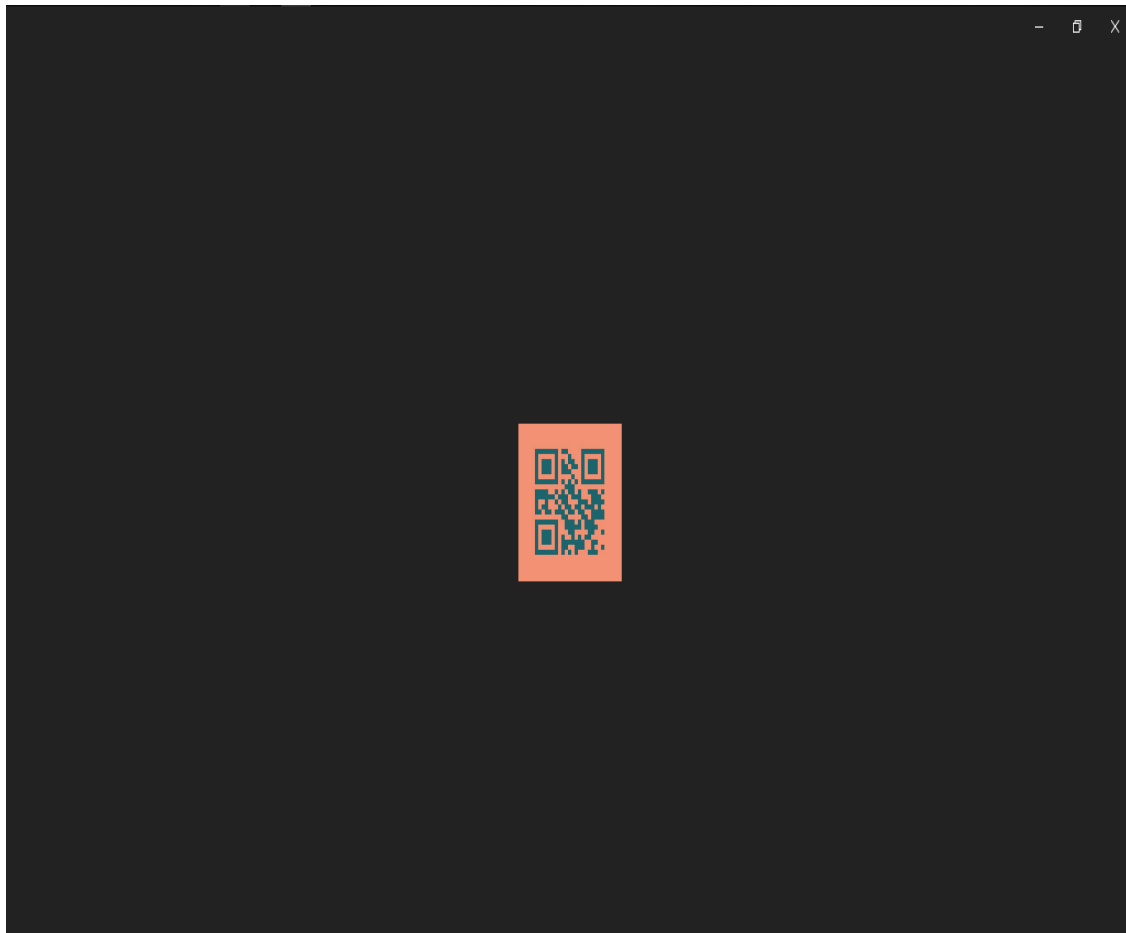


FIGURE V.16 – Présentation d'un exemple de QR code généré par le professeur

La figure V.16 représente un exemple de QR code généré par le professeur.

### V.1.4 Application mobile

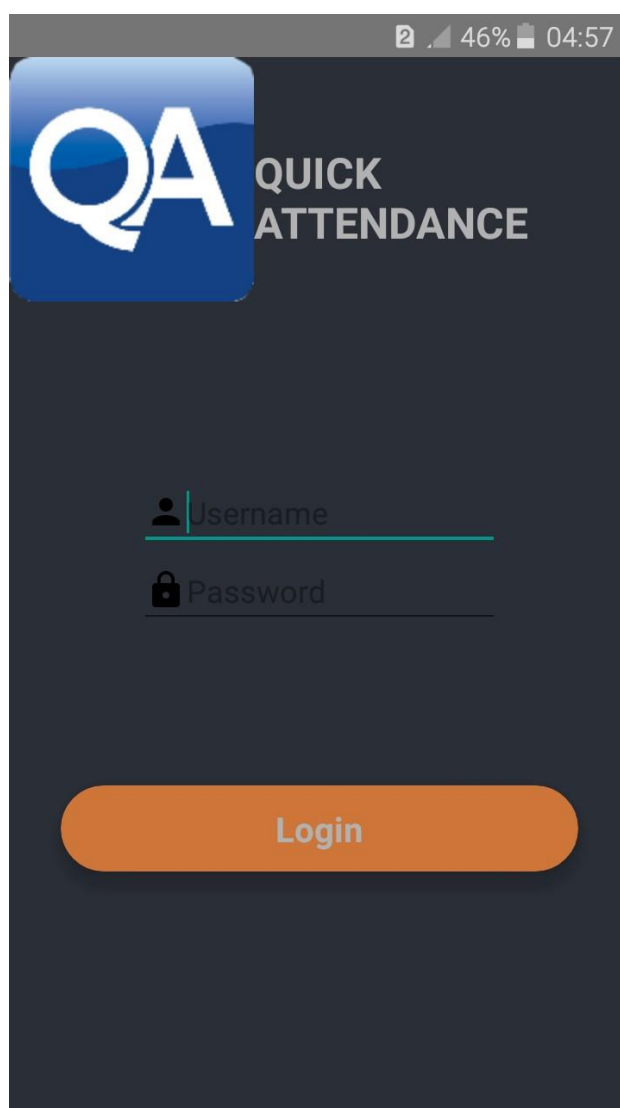


FIGURE V.17 – Présentation de l'interface d'authentification

La figure V.17 représente l'interface d'authentification.



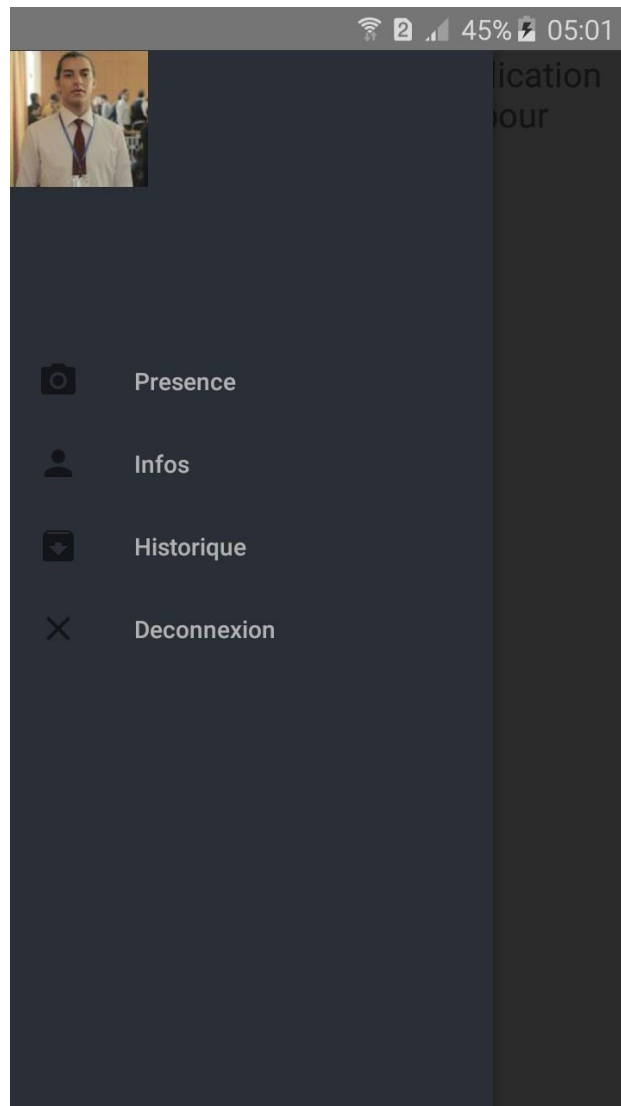


FIGURE V.18 – Présentation de l'interface de la liste déroulante

La figure V.18 représente l'interface de la liste déroulante.

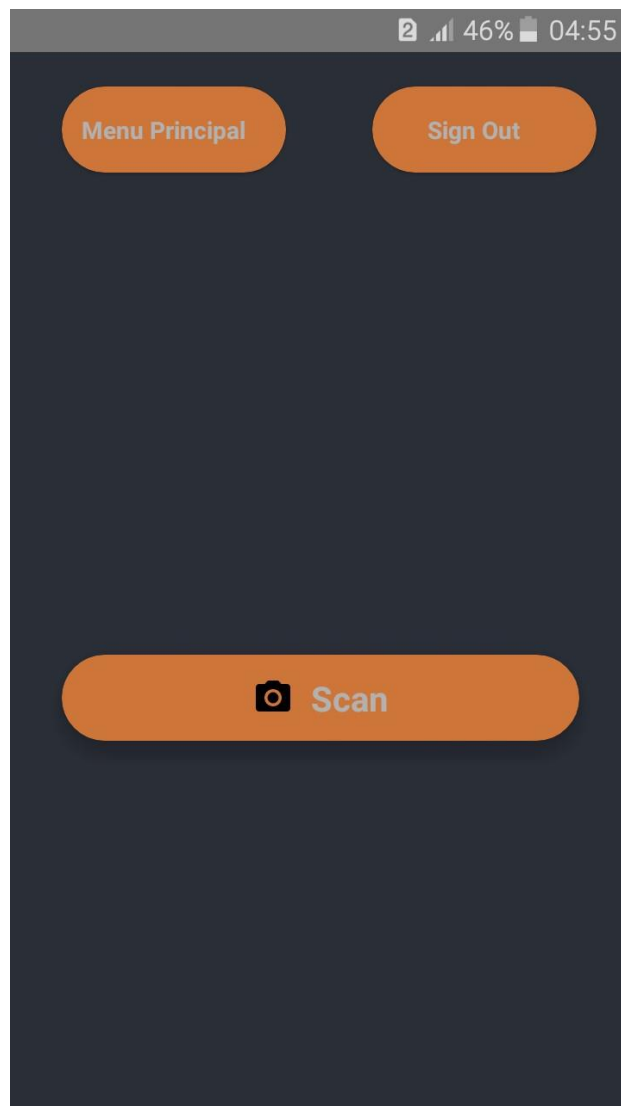


FIGURE V.19 – Présentation de l'interface du scan

La figure V.19 représente l'interface du scan.

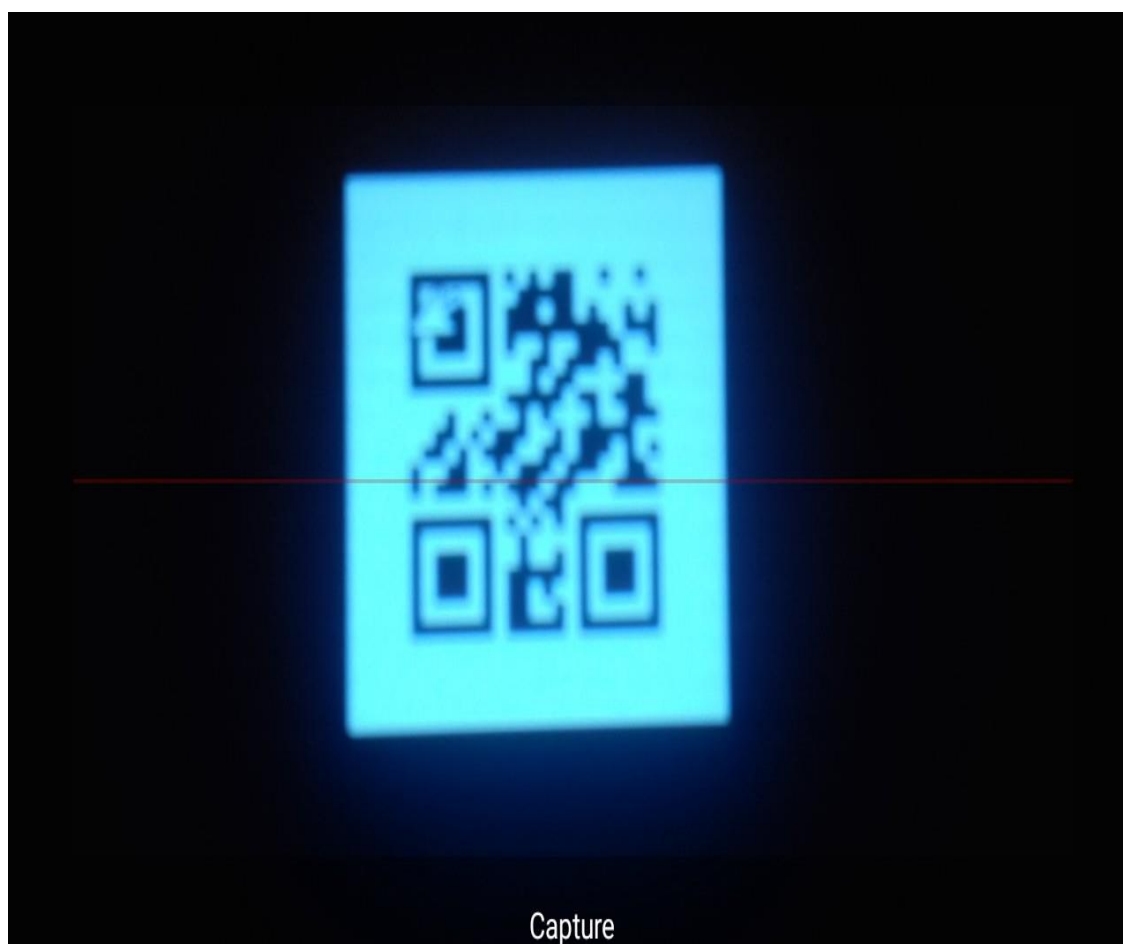


FIGURE V.20 – Présentation de l'interface du scan

La figure V.20 représente l'interface du scan.

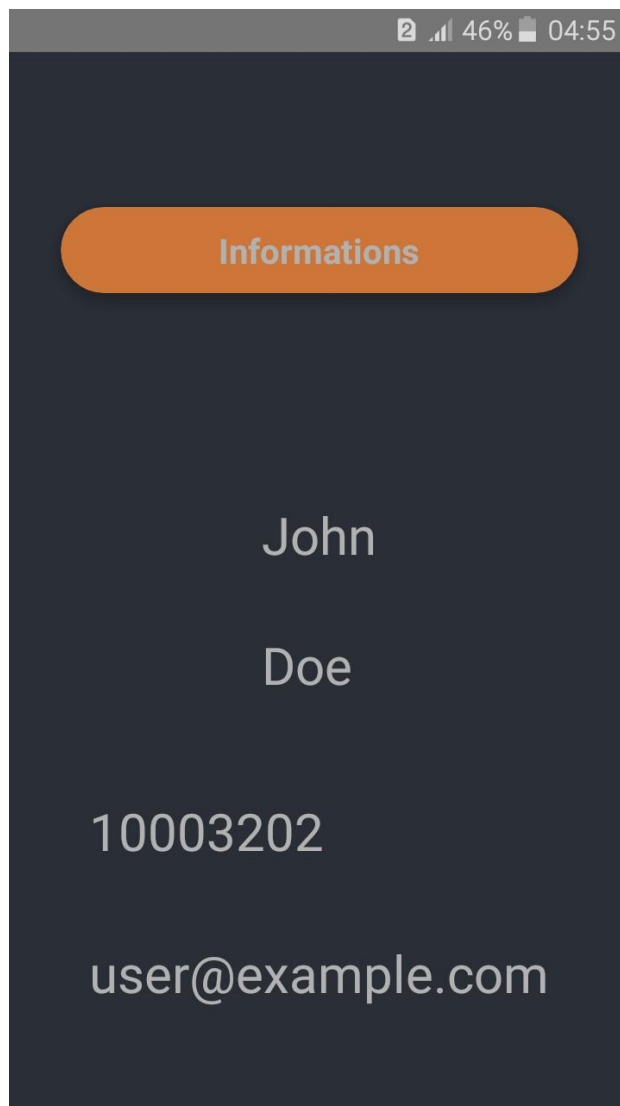


FIGURE V.21 – Présentation de l'interface des informations

La figure V.21 représente l'interface des informations.

## Conclusion

A ce stade, nous atteignons la fin du projet. A travers ce chapitre, nous avons présenté l'environnement matériel et logiciel de notre projet. Nous avons aussi présenté l'architecture de technique de notre application. Et nous avons présenté quelques interfaces graphiques résumant tout le travail que nous avons réalisé.

# Conclusion générale

L'objectif de ce projet de fin d'étude était de développer une application sous plateforme Android et Windows pour la gestion des absences par QR code.

Cette application est faite pour faciliter la tâche de la gestion d'absence au sein de notre faculté.

Le présent rapport détaille les différentes étapes par lesquelles nous sommes passés pour mettre en place cette application. Pour arriver à ce résultat nous avons d'abord commencé par l'analyse de l'existant qui nous a permis d'étudier les inconvénients d'un exemple de processus de gestion des absences et la méthodologie de travail appliquée.

Ensuite nous avons déterminé nos acteurs principaux, ainsi que leurs besoins qui nous ont permis de faire la réalisation du diagramme de cas d'utilisation des différentes tâches réalisables par notre projet juste après nous avons présenté l'architecture de l'application. Puis vient la présentation de la conception détaillée de la solution proposée.

Enfin, dans le dernier chapitre, nous avons donné une idée sur l'environnement de travail ainsi que les technologies utilisées.

Nous avons aussi décrit notre application à travers quelques captures d'écran détaillées.

Après la finalisation, le projet présente plusieurs fonctionnalités dont :

- 1. La gestion du personnel.
- 2. Le Scan d'un Qr code avec un appareil mobile pour marquer la présence.
- 3. La gestion automatique des présences enregistrée par les étudiants.
- 4. Enregistrement des présences et absences dans la base de données de l'administration.
- 5. Faire des statistiques.

Ce projet nous a donné l'occasion de confirmer nos connaissances dans le développement Android et Java, et surtout toucher aux plusieurs aspects du cycle de vie d'un produit logiciel tels que la mise en place de l'environnement de travail, la phase de test. De plus il nous a donné l'opportunité d'avoir une idée approfondie sur la panoplie de plusieurs technologies et outils.

Nous espérons, enfin, que ce travail apporte progrès et satisfaction aux membres du jury et aux personnes intéressées par cette étude.

## Bibliographie

- [1] : « Design Patterns », <https://www.commentcamarche.com/contents/474-design-patterns>, (03/06/2018).
- [2] : « Spring : théorie et pratique », <https://zekey.developpez.com/articles/spring>, (03/06/2018).
- [3] : « Méthodologie de développement MVC », <https://tahe.developpez.com/web/php/mvc>, (03/06/2018).
- [4] : « Présentation des méthodes agiles et Scrum », [https://ineumann.developpez.com/tutoriels/alm/agile scrum](https://ineumann.developpez.com/tutoriels/alm/agile%20scrum), (03/06/2018).
- [5] : « Meet Android Studio », <https://developer.android.com/studio/intro>, (03/06/2018).
- [6] : « Android SDK », <https://www.techopedia.com/definition/4220/android-sdk>, (03/06/2018).
- [7] : « Firebase », <https://www.developpez.com/actu/137290>, (03/06/2018).
- [8] : « Introduction à Maven 2 », <https://dcabasson.developpez.com/articles/java/maven/introduction-maven2>, (03/06/2018).
- [9] : « Add Firebase SDK Admin », <https://firebase.google.com/docs/admin/setup>, (03/06/2018).
- [10] : « JavaMail », <https://www.jmdoudoux.fr/java/dej/chap-javamail.htm>, (03/06/2018).
- [11] : « CSS », <https://www.futura-sciences.com/tech/definitions/internet-css-4050>, (03/06/2018).
- [12] : « Faciliter la connexion », <https://developer.android.com/distribute/best-practices/develop/firebase-authentication?hl=fr>, (03/06/2018).