RÉPUBLIQUE DU BÉNIN
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ D'ABOMEY-CALAVI

INSTITUT DE FORMATION ET DE RECHERCHE EN INFORMATIQUE

BP 526 Cotonou Tel : +229 21 14 19 88
http://www.ifri-uac.net Courriel : contact@ifri.uac.bj

MÉMOIRE

pour l'obtention du

Diplôme de Licence en Informatique

Option : Génie Logiciel

Présenté par :
Charbel **SONON**

Application pour la gestion et le suivi des absences des étudiants en milieu universitaire

Sous la supervision :
Ing. Jean-Pierre **KOUKPAKI**

Bon pour le dépôt

Membres du jury :

OUSMANE Matine	Docteur	Cour Suprême Bénin	Président
Sylvain ALLE	Ingénieur	AKS-Pictures	Examinateur
KOUKPAKI Jean-Pierre	Ingénieur	JTEK Solutions	Rapporteur

Année Académique : **2023-2024**

Sommaire

Dédicace ii

Remerciements iii

Résumé iv

Abstract v

Liste des figures vi

Liste des tableaux viii

Liste des acronymes ix

Introduction 1

1 Etat de l'art 3

2 Analyse des choix techniques et Conception 10

3 Résultats et Discussion 28

Conclusion 44

Bibliographie 45

Webographie 46

Table des matières 47

Dédicace

Ce mémoire est dédié à ma famille, source infinie de soutien et d'amour

Remerciements

Je tiens à exprimer ma profonde gratitude au Seigneur pour sa bienveillance et son amour, qui m'ont permis d'avoir la santé et l'intelligence nécessaires à la réalisation de ce mémoire.

Mes remerciements s'adressent également à mon directeur de mémoire, Monsieur Jean-Pierre KOUKPAKI, pour son engagement et sa supervision tout au long de cette rédaction.

Je n'oublie pas mes proches, dont les prières et le soutien moral m'ont aidé à surmonter les obstacles et à progresser.
Mes remerciements vont également à la direction du professeur Eugène EZIN, Directeur de l'Institut de Formation et de Recherche en Informatique (IFRI) à l'Université d'Abomey-Calavi, à toute l'administration et les enseignants de l'IFRI ainsi qu'à mes camarades de promotion pour ces trois années de formation enrichissantes.

Enfin, je tiens à exprimer ma reconnaissance à tous ceux qui, de près ou de loin, ont contribué à l'élaboration de ce mémoire, ainsi qu'aux membres du jury pour l'évaluation de mon travail.

Résumé

La gestion des absences en milieu universitaire constitue un enjeu important dans l'optique de garantir le suivi et la régularité des étudiants, principaux acteurs du secteur. Dans ce contexte nous avons dans notre projet conçu une application web et mobile dédiée à la gestion des absences. Cette application a pour but de simplifier et d'automatiser le processus de gestion de la présence et de générer des rapports statistiques afin d'améliorer l'efficacité des services administratifs et d'assurer un suivi plus précis des étudiants. Le développement de cette application a été fait en utilisant les technologies modernes telles que Flutter, Dart, NodeJs, Firebase et suivant une méthodologie de développement AGILE, la méthodologie SCRUM.

Mots clés :gestion des absences,application web et mobile,suivi des étudiants

Abstract

Managing absences in the university environment is an important issue to ensure the monitoring and regularity of students, who are the main actors in the sector. In this context, we design and develop in our project a web and mobile application dedicated to managing absences. This application will simplify and automate the recording of attendance and generate statistical reports to improve the efficiency of administrative services and ensure more precise monitoring of students. The development of this application was done using modern technologies such as Flutter, Dart, NodeJs, Firebase and following an AGILE methodology, the SCRUM methodology.

Key words: absence management, web and mobile application, student monitoring

Liste des figures

1.1 EducPro Solution 4
1.2 EasyLog QR App 5
1.3 Reconnaissance faciale 5
1.4 ZohoPeople App 6

2.1 Logo Flutter 10
2.2 Logo Dart 11
2.3 Firebase 11
2.4 MySQL 11
2.5 Logo Node.Js 12
2.6 Logo WampServer 13
2.7 Logo VS Code 13
2.8 Parts de marché des systèmes d'exploitation au Bénin 14
2.9 QR Code 14
2.10 Cycle de vie de la méthode SCRUM 15
2.11 Le langage UML 16
2.12 Diagramme de cas d'utilisation 18
2.13 Diagramme de Séquence Authentification 22
2.14 Diagramme de Séquence Ajouter Etudiant 23
2.15 Diagramme de Séquence Signer la Présence 24
2.16 Diagramme de Séquence Marquer la Présence manuellement 25
2.17 Diagramme de Séquence Signer la Présence 26
2.18 Diagramme de Classe 27

3.1 Schéma global de l'application 28
3.2 Logo de l'application 29
3.3 Premier slider de la page d'intro 30
3.4 Second slider de la page d'intro 30
3.5 Page de pré connexion 30
3.6 Page de connexion 30
3.7 Page d'inscription administrateur 31
3.8 Page des paramètres 32
3.9 Dashboard Administrateur 33
3.10 Gestion de filières 34
3.11 Création de filière 34
3.12 Gestion d'étudiants 34
3.13 Ajout d'étudiant par formulaire 35

3.14 Ajout d'étudiants par fichier 35
3.15 Gestion des professeurs 35
3.16 Ajout d'un nouveau professeur 35
3.17 Gestion des cours 36
3.18 Ajout d'un nouveau cours 36
3.19 Gestion des requêtes 36
3.20 Dashboard Professeur 37
3.21 Gestion des Séances d'un cours 38
3.22 Création de séance 38
3.23 Prise de présence Code QR 39
3.24 Prise de présence simple 39
3.25 Liste des étudiants du cours 40
3.26 Présence générale d'un étudiant 40
3.27 Résultat de la présence d'une séance 40
3.28 Dashboard Etudiant 41
3.29 Faire Une requête 42
3.30 fig :ScanPresence 42
3.31 Résultat du scan du code QR 42
3.32 Consulter Présence 43

Liste des tableaux

1.1 Comparaison des outils de suivi des présences 7

2.1 Comparaison entre REST et GraphQL 12
2.2 Rôles des différents acteurs 17
2.3 Description du processus d'authentification 19
2.4 Description du processus de lancement de la prise de présence 20
2.5 Description du processus de signature de la présence par les étudiants 21

Liste des acronymes

API :

Application Programming Interface 12, 47

HTTP :

HyperText Transfert Protocole 12

IDE :

Integrated Development Environment 13

PDF :

Portable Document Format 2, 8

QR :

Quick Response 4, 6, 8, 14, 21, 48

REST :

Representational State Transfer 12

SGBD :

Système de Gestion de Base de Données 11, 47

SQL :

Structured Query Langage 11

UML :

Unified Modeling Language 10, 16

VPN :

Virtual Private Network 6, 8

WAMP :

Windows Apache MySql Php 13

Introduction Générale

La gestion des absences en milieu universitaire est un défi important pour assurer le suivi et la régularité des étudiants, acteurs essentiels du secteur de l'éducation supérieure. Les établissements d'enseignement supérieur sont confrontés à des défis liés à la gestion et au suivi des absences, en raison de l'augmentation du nombre d'étudiants, de la complexité croissante des emplois du temps et des longues procédures administratives. Il est donc crucial de mettre en place une solution efficace pour optimiser la gestion des absences et améliorer la communication entre les différents acteurs du secteur.

Problématique

La gestion des absences des étudiants représente un défi majeur pour les établissements universitaires. En effet, la régularité des étudiants est essentielle pour garantir le bon déroulement des cours et la réussite académique.De plus tout établissement sérieux se doit d'avoir un œil sur l'assiduité de chacun de ses étudiants.
Cependant, les méthodes actuelles de gestion des absences au sein de nos établissements s'avèrent fastidieuses et difficiles rendant le travail lent et mal structuré.
Parmi les problèmes recensés, nous pouvons citer :

- **Problème de gestion** :l'enregistrement des présences se fait soit avec un fichier Excel pour chaque séance soit avec des feuilles de papier, ce qui constitue une perte de temps et une mauvaise organisation de travail.
- **Difficulté d'accès à l'information** : chaque liste de présence représente un fichier Excel indépendant ou une feuille de papier, ce qui amène à parcourir plusieurs fichiers pour consulter la présence et à s'encombrer de beaucoup de paperasses.
- **Mauvaise gestion de l'information** : l'absence d'une base de données commune pour stocker les informations de présence peut entraîner la perte de ces dernières.
- **Sécurité** : Tout individu peut accéder aux informations, les modifier ou les effacer.

Il est donc nécessaire de développer des outils afin d'informatiser le processus pour faciliter cette gestion et assurer un suivi efficace des absences.

Contexte et Justification

Dans le contexte actuel de l'enseignement supérieur, la gestion efficace des absences des étudiants est devenue un enjeu majeur. La diversité des raisons d'absence, qu'elles soient liées à des contraintes personnelles, des problèmes de santé ou d'autres facteurs, rend nécessaire la mise en place d'un système de suivi et de gestion des absences. En effet, les méthodes traditionnelles de suivi des absences, comme l'utilisation des feuilles de papier, se révèlent inefficaces et sujettes à des erreurs. Une application dédiée à cette tâche permettrait non seulement de simplifier la collecte des données d'absence, mais aussi de faciliter la communication entre les étudiants, les enseignants et l'administration tout en réduisant les pertes de temps avec les procédures administratives. Ce projet de mémoire vise donc à concevoir et développer une application qui offrira la possibilité de centraliser les données et de les rendre facilement accessibles à l'ensemble des acteurs du système .

Objectifs

L'objectif général de ce projet est de concevoir et développer une application web et mobile(Android) facilitant la gestion des absences.
Cette application devra :

- Faciliter le processus de marquage de la présence
- Faciliter la consultation de la présence des étudiants
- Faciliter la génération des rapports de présence au format PDF
- Gérer les requêtes des étudiants
- Diminuer les erreurs
- Garantir la sécurité à travers une procédure d' authentification

Organisation du mémoire

Ce mémoire est structuré en trois parties principales. En plus de ce chapitre d'introduction, notre travail sera divisé en trois chapitres :

Le premier chapitre aborde l'état de l'art qui présente certains des modèles existant et donne un aperçu du modèle que nous choisissons d'étudier.

Le deuxième chapitre détaille l'analyse, la modélisation et les choix techniques opérés en vue du développement de l'application.

Enfin, le troisième chapitre concerne la présentation du résultat de notre travail , son évaluation et propose des pistes d'amélioration pour les futures versions.

Chapitre 1

Etat de l'art

Introduction

L'étude préalable est une étape fondamentale dans tout processus de développement parce qu'elle permet d'évaluer la situation actuelle et d'en tirer les diverses conséquences. Il existe plusieurs applications de gestion des présences des étudiants. Dans ce chapitre nous en présenterons quelques-unes et collecterons les points importants pour créer notre application.

1.1 Les solutions existantes

Il existe plusieurs applications ou études abordant la problématique de la gestion des présences des étudiants. Dans ce chapitre, nous en présentons quelques-unes et recueillons les points importants pour la création de notre application.

1. **EducPro Solution**

   Sur le plan national (au Bénin), nous avons la plateforme EducPro Solution. [12] EducPro Solution est une platforme béninoise de suivi de vie scolaire. Elle offre des fonctionnalités de gestion de présence aux professeurs qui peuvent notamment marquer (cocher) les élèves présents et générer des rapports pour chaque jour.

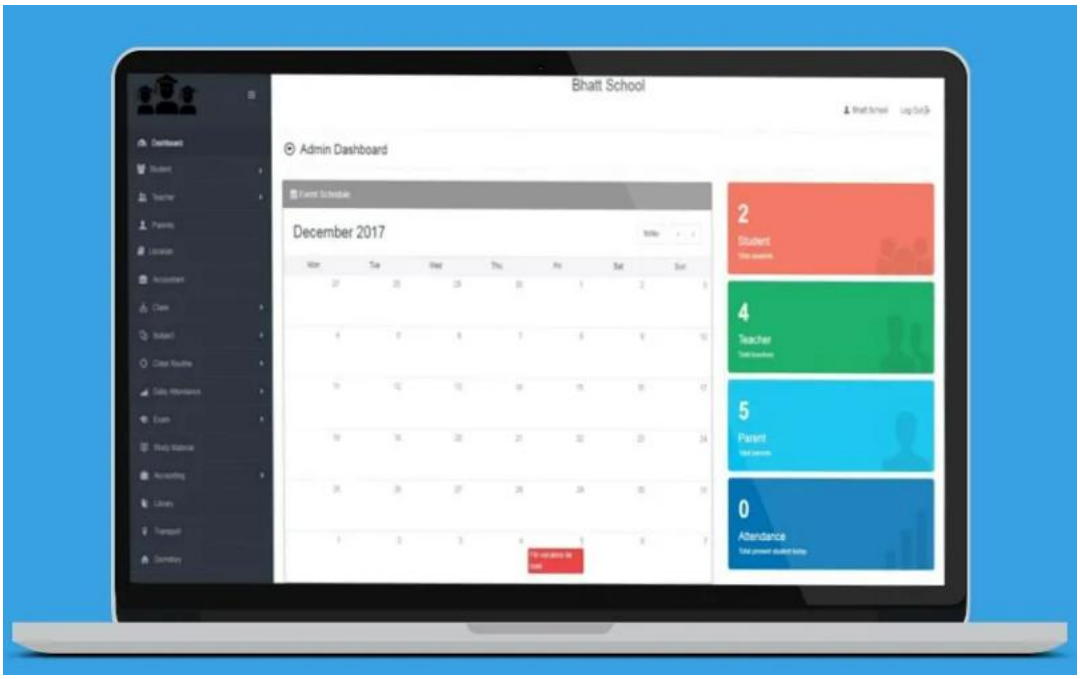


FIGURE 1.1 : EducPro Solution

2. **L'étude de Yong Wei Lun [2]**
   Yong Wei Lun (2019) [2] a mené une étude sur l'importance d'utiliser des systèmes basés sur des codes QR pour suivre la fréquentation des étudiants en milieu universitaire. Les résultats de l'étude suggèrent que la mise en œuvre de codes QR peut constituer un moyen sûr et efficace de capturer des informations sur les présences. Les étudiants peuvent facilement scanner le code QR affiché en classe à l'aide d'une application mobile, éliminant ainsi le besoin de prise de présence manuelle ou de listes papier. Pour empêcher toute manipulation ou tricherie, le système propose de mettre à jour les codes QR toutes les 10 secondes, ce qui garantit que les étudiants absents n'ont pas la possibilité d'obtenir un code récent facilement. Cette mesure renforce l'intégrité du système de suivi de l'assiduité. De plus, les étudiants peuvent accéder à leurs relevés de présence par souci de transparence et identifier les domaines à améliorer dans leur engagement académique. Des notifications sont également envoyées lorsque le pourcentage d'assiduité d'un étudiant tombe en dessous d'un certain seuil prédéfini, permettant une intervention précoce pour résoudre les problèmes d'assiduité.

3. **EasyLog QR Booking App [5]**
   EasyLog a développé une application qui permet aux employés d'enregistrer facilement leurs présences à un endroit spécifique en scannant un code QR[5]. Installée sur le smartphone d'un employé, l'application permet de scanner un code QR pour identifier une zone de travail comme un chantier de nettoyage ou de construction. Alternativement, l'application peut être installée sur une tablette et configurée pour enregistrer un processus lorsqu'un travailleur scanne son badge code QR individuel, par exemple sur une machine dans une usine.

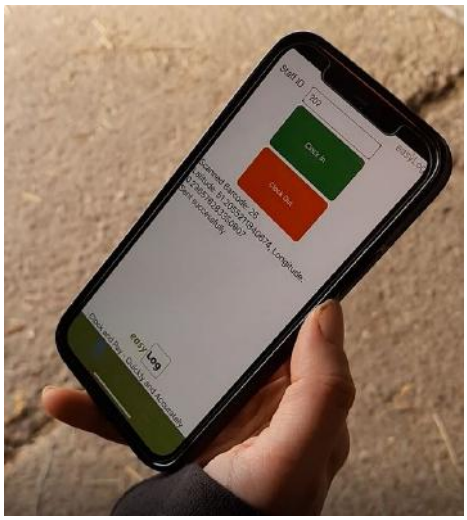


FIGURE 1.2 : EasyLog QR App

Caractéristiques :

- Identification automatique de l'employé
- Génération et exportation de rapport
- Enregistrement des entrées et sorties des employés

4. **Le système de Varadharajan et co. [1]**
   Varadharajan et co. (2016)[1] ont présenté un système de gestion automatique des présences utilisant la technologie de reconnaissance faciale. Cette approche innovante implique une caméra haute définition dans la salle de classe pour capturer les visages détectés, qui sont ensuite vérifiés par rapport à la base de connaissances du système. Les élèves dont le visage est reconnu sont marqués comme présents, tandis que les informations d'absence sont envoyées aux parents par SMS. Ce système suit non seulement la fréquentation des élèves, mais détecte également les personnes non autorisées dans la classe, renforçant ainsi la sécurité de l'école.



FIGURE 1.3 : Reconnaissance faciale

5. **Zoho People[10]**
   Zoho People [10] est une solution Ressources Humaines complète qui comprend aussi des fonctionnalités de gestions de présence basées sur la reconnaissance faciale en fournissant une vue détaillée des heures et absences des employés.

   Caractéristiques :

   - Marquage de la présence avec une fonctionnalité de géolocalisation
   - Prise de présence basée sur la reconnaissance faciale
   - Existence de diverses vues

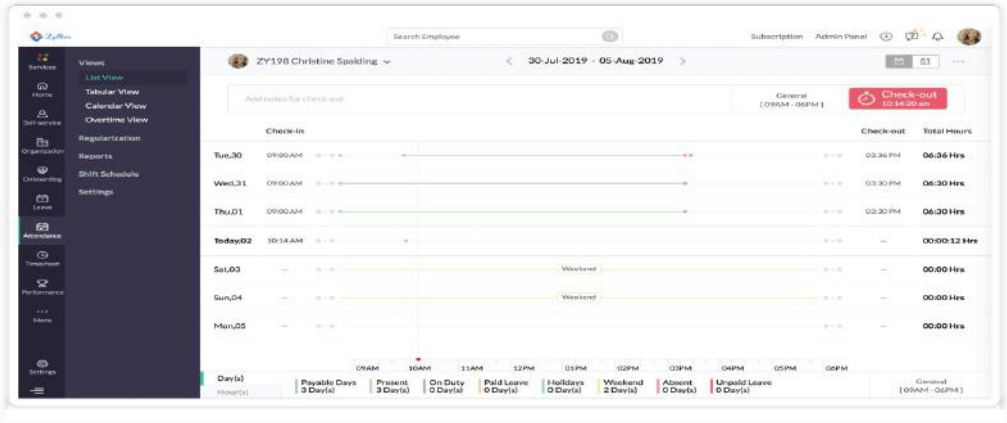


FIGURE 1.4 : ZohoPeople App

1.2 Comparaison des technologies de suivi de présence

Le tableau ci-dessous 1.1 compare quelques technologies de suivi des présences. Les solutions à faible coût utilisent la géolocalisation et le code QR. Cependant nous considérons que la solution avec le code QR est la meilleure car la solution utilisant la géolocalisation est facilement contournable avec les outils tels que les VPN.

Technologies	Description	Forces	Limites
• **Etude de Young Wei Lun** • **EasyLog QR Booking App**	Le système enregistre la présence des étudiants après qu'ils aient chacun scanné le Code QR affiché en classe à l'aide de leurs smartphone	• Prise de la présence • Simulation facile • Faible coût	• Uniquement en ligne • Interface non conviviale • Absence d'une fonctionnalité de communication • Ne fonctionne pas hors ligne
• **Le système de Varadharajan** • **Zoho People**	Le système enregistre la présence des étudiants une fois qu'ils sont reconnus par le lecteur de reconnaissance faciale et selon leurs positions	• Prise de la présence • Différentes vues pour les employés avec plusieurs informations • Détection des personnes non autorisées dans la classe • Fiabilité et sécurité	• Simulation difficile • Coût élevé • Absence d'une fonctionnalité de communication
EducPro Solution	Ce système en ligne permet aux enseignant de marquer numériquement la présence des élèves	• Prise de la présence • Communication en ligne • Gestion de la vie scolaire	• Non adapté à l'enseignement supérieur • Coût élevé • Uniquement en ligne

TABLE 1.1 : Comparaison des outils de suivi des présences

Dans le contexte de l'université béninoise, il ne serait pas pratiquer d'adopter des systèmes de reconnaissance faciale ou de pointage en ligne car le coût est relativement élevé. La mise en place de telles solutions requière un nombre important d'équipements car chaque classe doit être équipée avec les appareils nécessaires selon la solution utilisée. Les solutions basées sur QR Code et Géolocalisation se révèlent être les solutions les plus adéquates.

1.3 Présentation de notre Application Solution *"EasyAttend"*

Après avoir étudié les applications existantes, nous arrivons à la conclusion que la plupart d'entre elles ne sont pas adéquates au système universitaire béninois. La majorité d'entre elles présentent des insuffisances dans le sens où si tous les étudiants n'utilisent pas l'application mobile pour pointer, la plateforme ne peut pas être efficace car le professeur n'a pas la possibilité de marquer les présences à son niveau. De plus certaines d'entre elles peuvent être contournées avec des outils tels que les VPN . Il est aussi à noter le coût élevé d'accès à certaines de ces plateformes. Nous avons donc décidé de créer une application mobile sous Android et Web destinée à la fois aux étudiants, aux enseignants et aux administrateurs et répondant à la plupart des besoins. Notre application offrira la possibilité aux enseignants d'enregistrer très facilement et rapidement la présence des étudiants.

1.3.1 Objectifs de notre application

L'objectif principal de notre application est de concevoir et de développer une application web et mobile(Android) facilitant la gestion des absences.
Cette application devra permettre de :

- Prendre facilement la présence
- Faciliter la consultation des listes de présence
- Générer des rapports statistiques au format PDF
- Gérer les requêtes des étudiants
- Diminuer les erreurs
- Garantir la sécurité à travers une procédure d' Authentification
- Structurer correctement le travail

Pour cela, notre application de gestion des absences des étudiants appelée «EasyAttend» devrait contenir trois parties :

Partie Étudiant : Elle permet à l'étudiant de consulter la liste de ses cours et sa présence pour chaque cours. Elle lui permet aussi d' imprimer ses rapports de présence pour chaque cours au PDF, de signer sa présence en scannant un code QR généré par l'enseignant et de faire des requêtes auprès de l'administration.

Partie Professeur : Elle permet à l'enseignant de consulter la liste de présence de chaque séance, la liste de présence générale d'un étudiant en particulier. Elle lui permet également d'imprimer les listes. L'enseignant pourra également prendre la présence manuellement dans l'application ou simplement ouvrir une séance code QR donnant l'autorisation aux étudiants pour signer leur présence par scan. Les étudiants ne pourront scanner le code QR que si la séance de présence est ouverte. L'enseignant peut démarrer et arrêter une séance à sa guise.

Partie Administrateur : Les administrateurs disposent de tous les avantages qu'ont les enseignants. En plus de cela, un administrateur peut créer des comptes pour les étudiants et les enseignants, ajouter des cours et des filières et gérer les requêtes des étudiants.

Conclusion

Dans ce chapitre, nous avons présenté quelque différentes applications de gestion de présence existantes et le sommaire de notre solution qui sera une application sous Android et Web pour la gestion des absences des étudiants ainsi que ses objectifs.

Chapitre 2

Analyse des choix techniques et Conception

Introduction

Dans ce chapitre, nous allons aborder l'analyse et la conception de notre système. Notre objectif est de prévoir, d'étudier et de collecter les informations nécessaires pour gérer efficacement notre système. Pour ce faire, nous utiliserons le langage UML, qui nous permettra de réduire les ambiguïtés grâce à un vocabulaire précis et indépendant des langages orientés objet, et de modéliser nos applications selon une vision orientée objet.

2.1 Analyse des choix techniques

La réalisation de notre application a nécessité l'utilisation d'un ensemble de technologies et d'outils. Dans cette section, nous présenterons notre environnement de développement à travers les technologies et les outils utilisés pour la réalisation de l'application.

2.1.0.1 Frontend

Le front-end est la partie avec laquelle un utilisateur interagit directement. Pour le front-end, nous avons utilisé plusieurs technologies.

Flutter

Flutter est un framework d'interface utilisateur gratuit et open-source de Google qui permet de créer de superbes interfaces utilisateur. Il change la donne car il permet de créer des applications pour plusieurs plateformes, à la fois iOS, Android et même le Web, en utilisant une seule base de code et un seul langage de programmation.



FIGURE 2.1 : Logo Flutter

Dart

Dart est un langage de programmation optimisé pour les applications sur plusieurs plateformes. Il est développé par Google et est utilisé pour créer des applications mobiles, de bureau, de serveur et web.



FIGURE 2.2 : Logo Dart

2.1.1 BACKEND

Le Back-end, contrairement au Front-end, représente tout ce que l'utilisateur ne voit pas. Il s'agit de la programmation côté serveur destiné mettre en place l'ensemble des traitements et contrôles de l'application. Dans notre solution nous avons opté pour différentes approches pour gérer notre backend.

2.1.1.1 Authentification

Concernant l'authentification des utilisateur de notre application, nous avons choisi d'opter pour le service d'authentification de Firebase .

Firebase est une plateforme de développement d'applications web et mobiles soutenue par Google, pour aider les développeurs à offrir des expériences d'applications plus riches. Firebase gère sa propre infrastructure avec un bel ensemble d'outils pour simplifier le travail du développeur en lui fournissant des kits de développement et un tableau de bord en ligne. Firebase offre des fonctionnalités d' **authentification** et **base de données en temps réel**.[3]



FIGURE 2.3 : Firebase

2.1.1.2 Système de Gestion de Base de données (SGBD)

Notre choix de SGBD s'est porté sur MySql. MySQL est un système de gestion de bases de données relationnelles (SGBDR). Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public que par des professionnels[14] . Le principe d'une base de données relationnelle est d'enregistrer les informations dans des tables, qui représentent des regroupements de données par sujets (table des etudiants, table des professeurs, table des filières, par exemple). Les tables sont reliées entre elles par des relations. MySql supporte le langage de requête SQL.



FIGURE 2.4 : MySQL

2.1.1.3 API

Une API (Application Programming Interface), en français Interface de Programmation Applicative, est un ensemble de définitions et de protocoles qui facilite la création et l'intégration des applications. En termes simples, une API permet à différentes applications de communiquer et d'échanger entre elles indépendamment des détails de leurs mise en œuvre.
Un service web est un type d'API, accessible via internet, souvent via le protocole HTTP dans différents formats. Les API Web les plus populaires sont REST et GraphQL. Les API REST sont basées sur le modèle architectural REST (Representational State Transfer). Elles utilisent des URIs pour identifier des ressources, et des verbes HTTP pour effectuer des opérations sur ces ressources (GET, POST, PUT, PATCH, DELETE…). Les API GraphQL sont basées sur un langage de requêtes dédié. Elles permettent aux clients de demander exactement les données dont ils ont besoin.[9]

Caractéristique	REST	GraphQL
Avantages	- Bien établies et largement utilisées - Relativement faciles à comprendre et à mettre en œuvre - Compatibles avec de nombreux langages de programmation et frameworks	- Plus efficaces pour les requêtes complexes - Permettent de récupérer des données associées ou imbriquées en un seul appel - Plus flexibles que les API REST
Inconvénients	- Moins efficaces pour les requêtes complexes - Peuvent nécessiter plusieurs requêtes pour récupérer des données associées ou imbriquées	- Moins bien établies que les API REST - Peuvent être plus difficiles à comprendre et à mettre en œuvre

TABLE 2.1 : Comparaison entre REST et GraphQL

Nous estimons que REST est plus pratique et répond équitablement au besoin actuel. Sa simplicité, sa large utilisation, sa courbe d'apprentissage réduite et sa proximité avec d'autres technologies Web dans la philosophie de conception en font la plus appropriée pour notre projet.

2.1.1.4 Serveur avec Node.Js

Node.js est un environnement d'exécution Javascript asynchrone non bloquant. Un tel environnement procure au Javascript la possibilité de s'exécuter sur le serveur (comme c'est le cas pour PHP, Python ou Perl...) afin de créer des applications backend ou natives (comme les applications mobiles). [4]. Associé au framework ExpressJs, NodeJs nous a permis de mettre en place notre API. Express.js est un framework pour construire des applications web basées sur Node.js. C'est de fait le framework standard pour le développement de serveur en Node.js.



FIGURE 2.5 : Logo Node.Js

2.1.1.5 WampServer

WampServer est une plateforme de développement Web de type WAMP, permettant de faire fonctionner localement des scripts PHP. WampServer n'est pas en soi un logiciel, mais un environnement comprenant deux serveurs (Apache et MySQL), un interpréteur de script (PHP), ainsi que phpMyAdmin pour l'administration Web des bases MySQL.



FIGURE 2.6 : Logo WampServer

2.1.2 Environnement de développement :Visual Studio Code

VS Code est un environnement de développement intégré (IDE) qui permet d'éditer du code.Les fonctionnalités incluent la prise en charge du débogage, la mise en évidence de la syntaxe, la complétion intelligente du code (IntelliSense), les snippets, la refactorisation du code et Git intégré.



FIGURE 2.7 : Logo VS Code

2.2 Choix de la plateforme client

Pour la plateforme client, nous avons choisi une approche multiplateforme en développant à la fois une **application mobile** et une **application Web**. Ce choix est fait en raison de la nécessité d'offrir une expérience utilisateur cohérente et fluide, quel que soit le dispositif utilisé. Les applications mobiles sont devenues essentielles dans notre vie quotidienne, offrant une accessibilité immédiate, tandis que les applications web sont accessibles depuis n'importe quel navigateur web.

2.3 Choix de la plateforme mobile

Comme le montre la figure suivante 2.8 [13], le système d'exploitation Android domine largement le marché béninois. Android est un système d'exploitation performant et moderne mais qui ne demeure pas pour autant moins simple et moins souple. De plus Android est open source et offre un environnement adéquat aux développeurs.

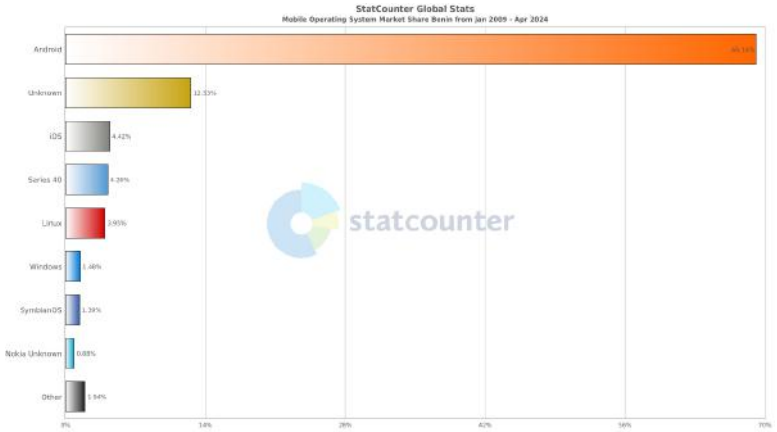


FIGURE 2.8 : Parts de marché des systèmes d'exploitation au Bénin

Le principal concurant de Android est ios de Apple. Les fournisseurs d'appareils Android offrent des produits accessibles et de première classe aux personnes du monde entier. Cela permet aux consommateurs de trouver ce qui correspond à leur budget à partir d'une variété d'appareils.
Cela nous montre que Android est clairement plus avantageux dans notre contexte et c'est pour cela que nous l'avons choisi comme support pour développer notre application !

2.4 Le Code de Réponse Rapide (QR)

Un QR code contient un motif dans un graphique carré, dans lequel des informations sont intégrées. Ce carré comporte des points et des lignes en noir et blanc, lus par l'application.[6] Les QR codes sont utilisés partout où les données doivent être lues rapidement. Vous pouvez les trouver sur des billets, des distributeurs automatiques, des timbres, des prospectus, des affiches et bien d'autres endroits. Chaque appareil photo, qu'il s'agisse d'un smartphone ou d'une tablette, peut lire et traiter les QR codes avec une application correspondante. Certains téléphones peuvent nécessiter l'utilisation d'une application de lecture QR, qui peut être téléchargée gratuitement à partir des magasins Android ou iPhone.



FIGURE 2.9 : QR Code

2.5 La méthodologie du projet

Pour la réalisation de notre application nous nous sommes basés sur une méthode de type AGILE, *SCRUM*. Une méthode AGILE permet de mieux communiquer avec le client et d'avoir une meilleure visibilité du produit livrable tout en assurant la qualité en continu et la détection des problèmes au fur et à mesure.

2.5.1 La méthodologie SCRUM

Le principe de la méthodologie SCRUM est de développer des application complexes en organisant le projet autour de "sprint" de développement (les itérations) d'une durée allant généralement de deux à quatre semaines. Un exemple typique d'utilisation de Scrum est un projet de développement informatique où le client n'a pas encore défini toutes les fonctionnalités dont il a besoin et où une certaine souplesse d'organisation est donc nécessaire. Avec des livraisons toutes les 4 semaines en moyenne, le client reçoit un logiciel fonctionnel à chaque itération. Plus nous avançons dans le projet, plus le logiciel est complet et possède de plus en plus de fonctionnalité.[15] La figure 2.10 illustre le cycle de vie de la méthode SCRUM :

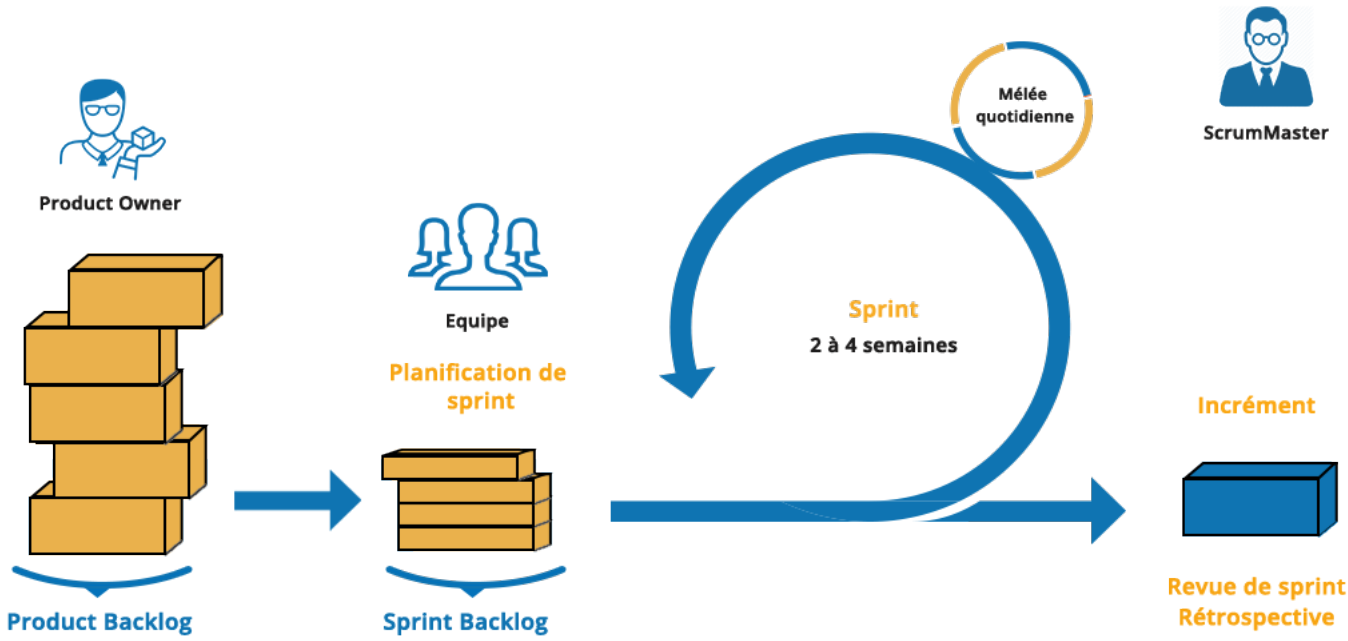


FIGURE 2.10 : Cycle de vie de la méthode SCRUM

Comme nous pouvons le remarquer dans cette figure, pour mettre en place la méthode SCRUM, il faut dans un premier temps identifier les différentes fonctionnalités de l'application. Ces dernières forment le backlog du produit.
Ensuite, il faut faire une planification du sprint pour définir le plan détaillé d'une itération.
Les sprints durent environ deux à quatre semaines durant lesquelles il y a toujours des réunions quotidiennes entre les différents intervenants du projet afin de faire le point sur l'état d'avancement des différentes tâches en cours, les difficultés rencontrées ainsi que les tâches restantes à effectuer. Une fois le produit partiel est prêt, on effectue une vérification de la conformité de ce qui a été fait

durant le sprint et on peut l'améliorer en procédant à l'étape de rétrospective.

2.6 Modélisation

Afin de modéliser les fonctionnalités de notre application, nous utiliserons le langage UML.

2.6.1 Le langage UML



FIGURE 2.11 : Le langage UML

Le langage UML (Unified Modeling Language, ou Langage de Modélisation Unifié) a été pensé pour être un langage de modélisation visuelle commun, et riche sémantiquement et syntaxiquement. Il est destiné à l'architecture, la conception et la mise en œuvre de systèmes logiciels complexes par leur structure aussi bien que leur comportement. Dans l'ensemble, les diagrammes UML décrivent la limite, la structure et le comportement du système et des objets qui s'y trouvent.L'UML n'est pas un langage de programmation, mais il existe des outils qui peuvent être utilisés pour générer du code en plusieurs langages à partir de diagrammes UML. L'UML a une relation directe avec l'analyse et la conception orientées objet. [8]
Le langage UML est également indépendant du langage de programmation utilisé pour le développement d'une application. Ainsi, les diagrammes de cas d'utilisation, de classes et de séquence serviront de base pour la modélisation de notre application.

2.6.2 Identification des acteurs

Un acteur est une entité extérieure au système modélisé, et qui interagit directement avec celui ci. Pour notre application ,nous avons recensé trois (03) acteurs :

- Les étudiants
- Les professeurs
- Les administrateurs

2.6.2.1 Rôle des acteurs

Le tableau 2.2 suivant présente les rôles de chacun des acteurs

Acteur	Rôles
Étudiant	• S'authentifier • Consulter la liste de ses cours • Consulter sa présence d'un cours après avoir sélectionné le cours • Imprimer un rapport de présence • Scanner une présence • Faire une requête • Gérer son profil
Professeur	• S'authentifier • Consulter les cours qui lui sont attribués • Gérer les séances d'un cours(Créer, Supprimer) • Prendre la présence d'une séance (manuellement ou par code QR) • Consulter la présence d'une séance • Consulter la présence d'un seul étudiant • Imprimer les rapports de présence • Gérer son profil
Administrateur	• S'authentifier • Gérer les étudiants (Ajout, modification, suppression) • Gérer les professeurs (Ajout, modification, suppression) • Gérer les filières (Ajout, modification, suppression) • Gérer les cours (Ajout, modification, suppression) • Gérer la présence (comme les professeurs) • Gérer les requêtes des étudiants (Approuver, Rejeter, Mettre en attente) • Consulter et Imprimer les rapports de présence • Gérer son profil

TABLE 2.2 : Rôles des différents acteurs

2.6.3 Le Diagramme de cas d'utilisation

2.6.3.1 Définition

Le diagramme de cas d'utilisation constitue la première étape de l'analyse en UML. Un diagramme de cas d'utilisation sert à résumer les informations des acteurs d'un système et leurs interactions avec ce dernier. Il permet de recueillir, d'analyser, d'organiser les besoins et de recenser les grandes fonctionnalités du dit système.

Le diagramme 2.12 suivant présente globalement les différentes tâches que les utilisateurs peuvent effectuer dans notre application.

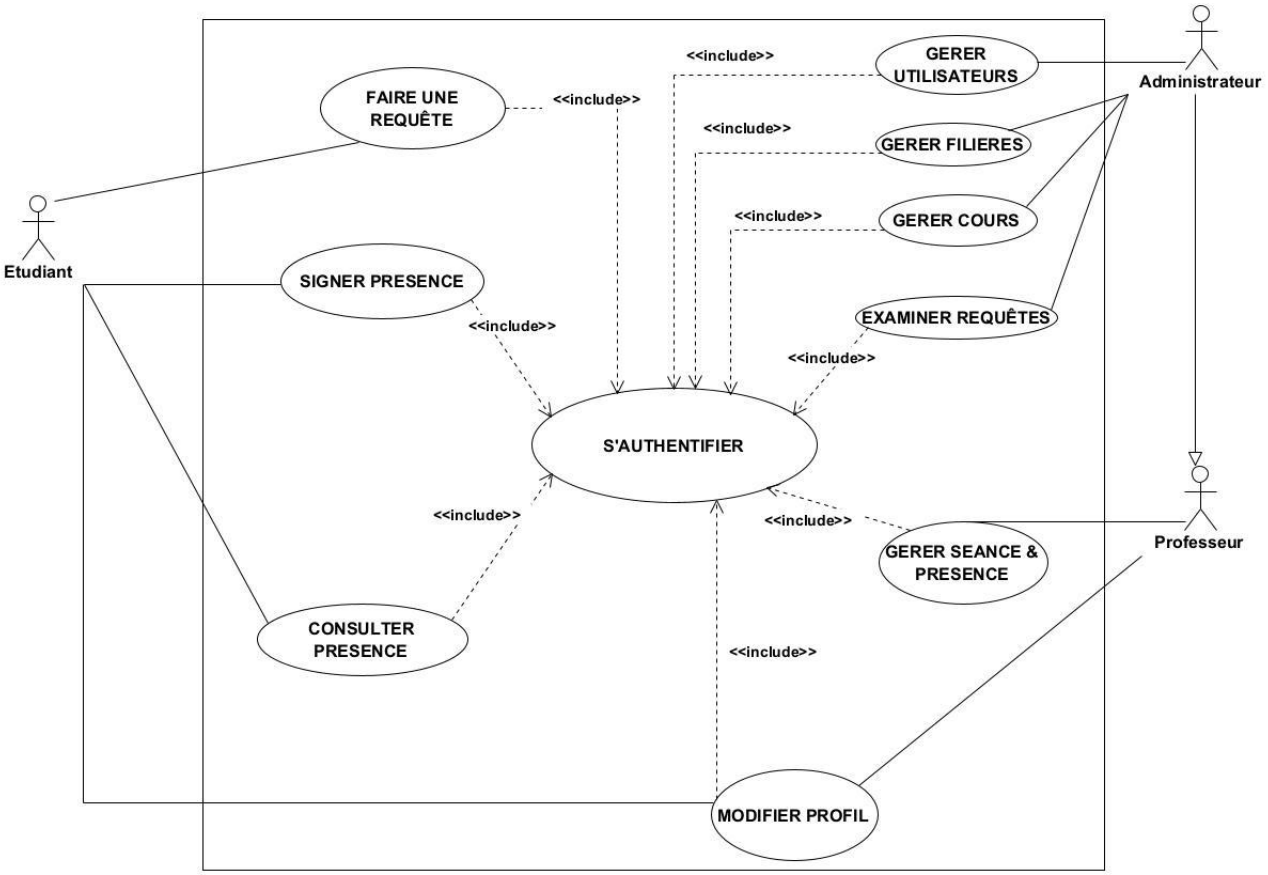


FIGURE 2.12 : Diagramme de cas d'utilisation

2.6.4 Les Scénarios

Les tableaux suivants, nous décriront 3 cas d'utilisations autour desquels tournent notre application de gestion de présence. Il s'agit des cas d'utilisation suivant :

1. Authentification des acteurs
2. Lancement de la présence QR Code par l'enseignant
3. Signature de la présence par les étudiants

2.6.4.1 Scénario "Authentification"

Nom :	Authentification
Acteur :	Administrateur, Professeurs, Etudiants
Objectif :	Garantir l'identité des utilisateurs et leur permettre d'accéder aux fonctionnalités qui leur correspondent
Scénario :	• L'utilisateur sélectionne son rôle à l'ouverture de l'application (Etudiant, Professeur, Administrateur) • L'utilisateur accède à la page d'authentification • L'application lui affiche le formulaire de connexion • L'utilisateur entre son e-mail et son mot de passe • L'application vérifie les identifiants de l'utilisateur • L'application redirige l'utilisateur vers l'interface correspondante
Scénario alternatif :	• Identifiants incorrects • Accès non autorisé • Affichage d'un message d'erreur et arrêt du cas d'utilisation

TABLE 2.3 : Description du processus d'authentification

2.6.4.2 Scénario "Lancement de la présence"

Nom :	Autorisation de la prise de présence
Acteur :	Professeur
Objectif :	Permet au professeur de prendre la présence des étudiants par QR Code
Scénario :	• Le professeur sélectionne le cours puis la séance cible après son authentification • Le professeur lance la séance de présence • L'application génère un code QR spécifique à la séance • Les étudiants passe signer la présence en scannant le code QR • L'application enregistre chaque étudiant qui scanne • Le professeur ferme la séance de présence
Scénario alternatif :	• Une erreur se produit : Affichage d'un message d'erreur et arrêt du cas d'utilisation

TABLE 2.4 : Description du processus de lancement de la prise de présence

2.6.4.3 Scénario "Signer la présence"

Nom :	Signer la présence
Acteur :	Etudiant
Objectif :	Permet aux étudiants de signer la présence d'une séance
Scénario :	• L'étudiant accède à la page de signature de présence après son authentification • L'étudiant lance le scan puis scanne le code QR affiché sur l'appareil du professeur • Le système enregistre la présence de l'étudiant • Les étudiants passe signer la présence en scannant le code QR affiché sur l'appareil du professeur • L'application enregistre la présence du professeur
Scénario alternatif :	• Une erreur se produit : Affichage d'un message d'erreur et arrêt du cas d'utilisation • La séance de présence est fermée • Le code QR ne correspond à aucune séance

TABLE 2.5 : Description du processus de signature de la présence par les étudiants

2.6.5 Les diagrammes de Séquence

2.6.5.1 Définition

Le diagramme de séquence fait parties des diagrammes comportementaux (dynamique) et plus précisément des diagrammes d'interactions. Il permet de représenter des échanges entre les différents objets et acteurs du système en fonction du temps.
A moins que le système à modéliser soit extrêmement simple, nous ne pouvons pas modéliser la dynamique globale du système dans un seul diagramme. Nous ferons donc appel à un ensemble de diagrammes de séquences chacun correspondant à une sous fonction du système, généralement d'ailleurs pour illustrer un cas d'utilisation.[11]

2.6.5.2 Diagramme de Séquence «Authentification»

Nous représenterons ici le diagramme de séquence du cas d'utilisation qui précède tous les autres : l'authentification.

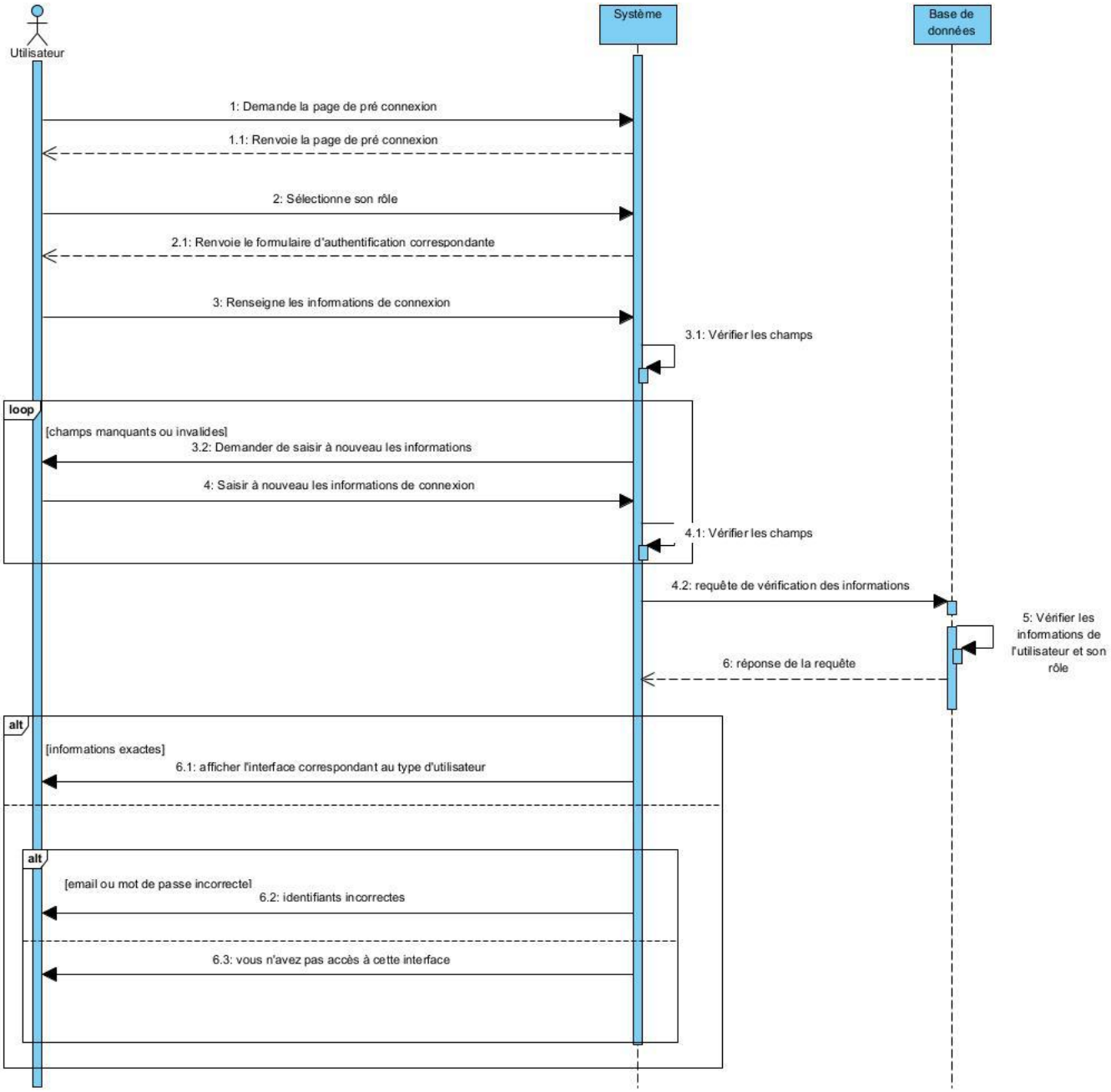


FIGURE 2.13 : Diagramme de Séquence Authentification

2.6.5.3 Diagramme de séquence «Ajouter Etudiant»

Nous représenterons ici le diagramme de séquence du cas d'utilisation qui permet aux administrateurs d'ajouter des étudiants.

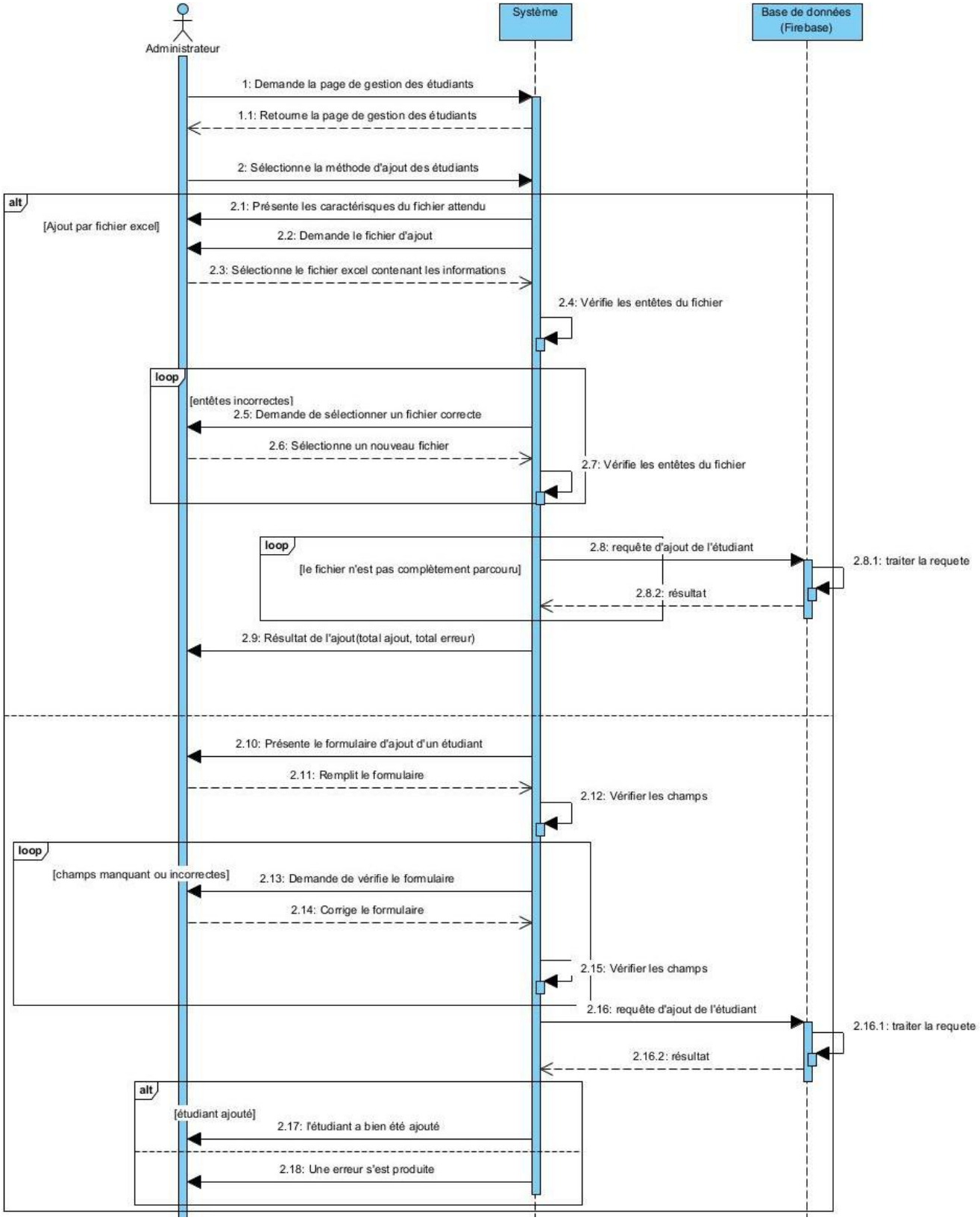


FIGURE 2.14 : Diagramme de Séquence Ajouter Etudiant

2.6.5.4 Diagramme de séquence «Signer La Presence»

Nous représenterons ici le diagramme de séquence du cas d'utilisation qui permet aux étudiants de signer la présence par code QR.

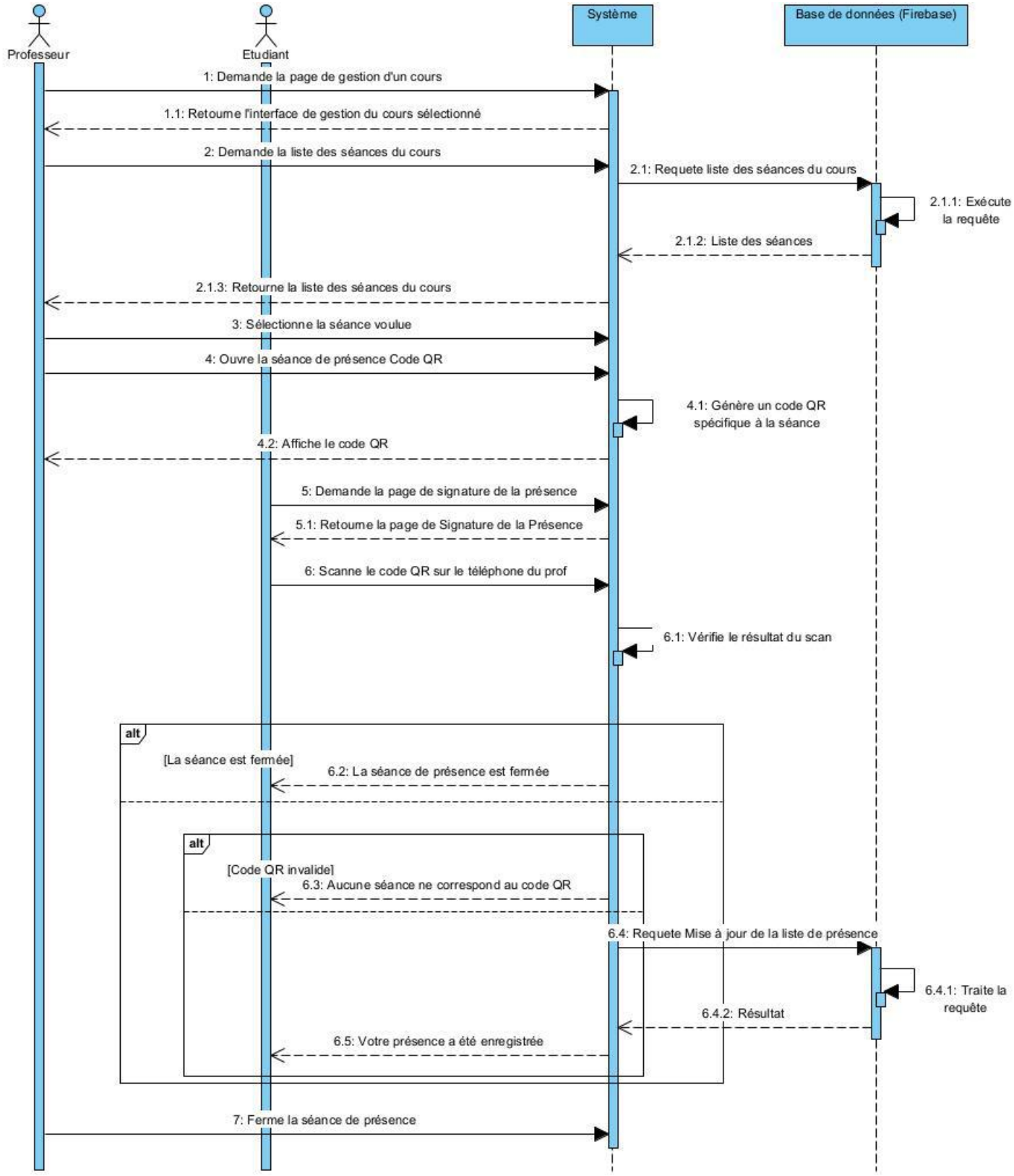


FIGURE 2.15 : Diagramme de Séquence Signer la Présence

2.6.5.5 Diagramme de séquence «Marquer la présence manuellement»

Nous représenterons ici le diagramme de séquence du cas d'utilisation qui permet aux enseignants et aux administrateurs de marquer la présence manuellement en cochant dans une liste, les étudiants présents. Il s'agit d'une alternative à la prise de présence par QR Code.

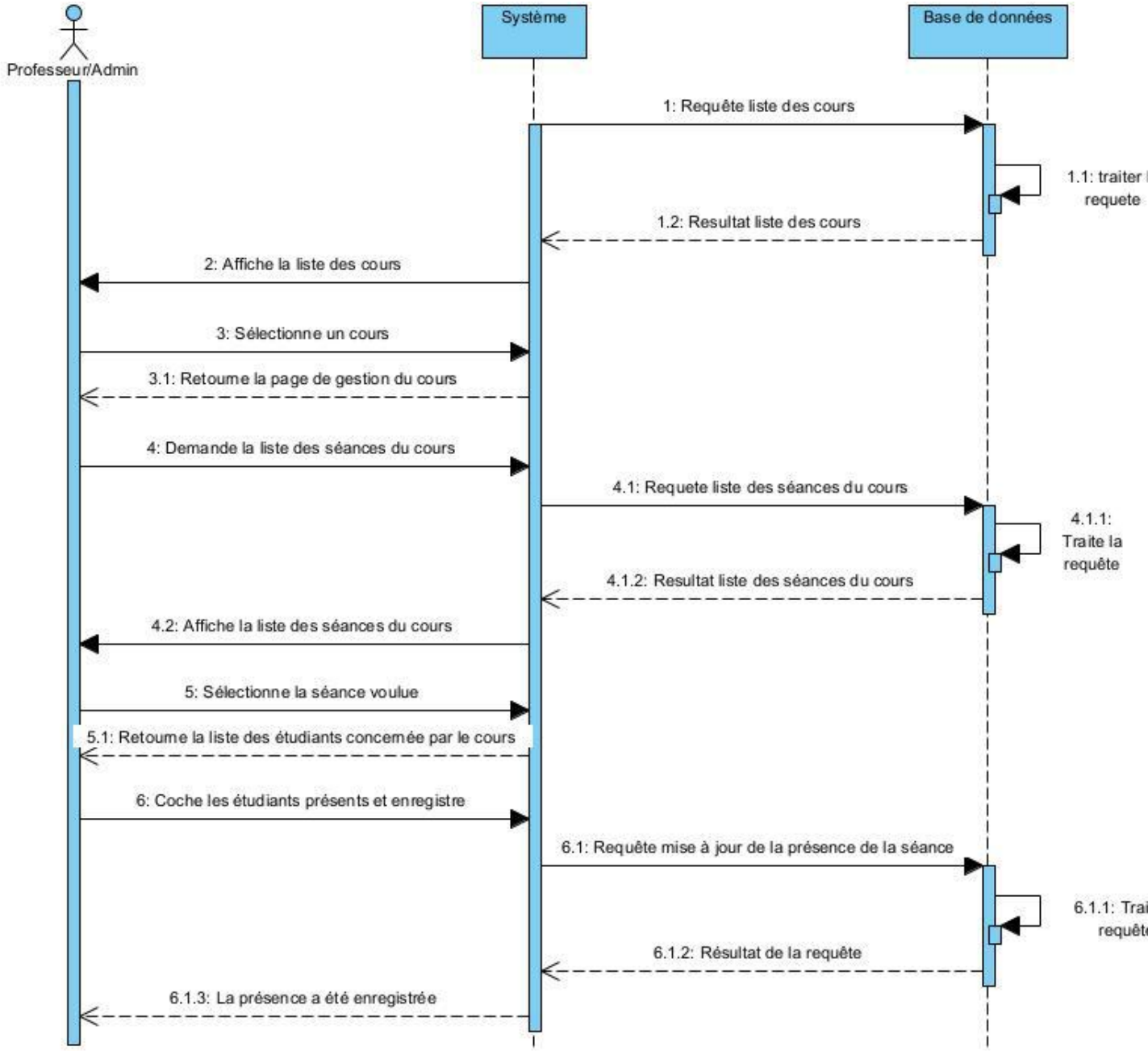


FIGURE 2.16 : Diagramme de Séquence Marquer la Présence manuellement

2.6.5.6 Diagramme de séquence «Consulter et Imprimer la présence d'une séance»

Nous représenterons ici le diagramme de séquence du cas d'utilisation qui permet aux professeurs et aux administrateurs de consulter et d'imprimer la présence d'une séance.

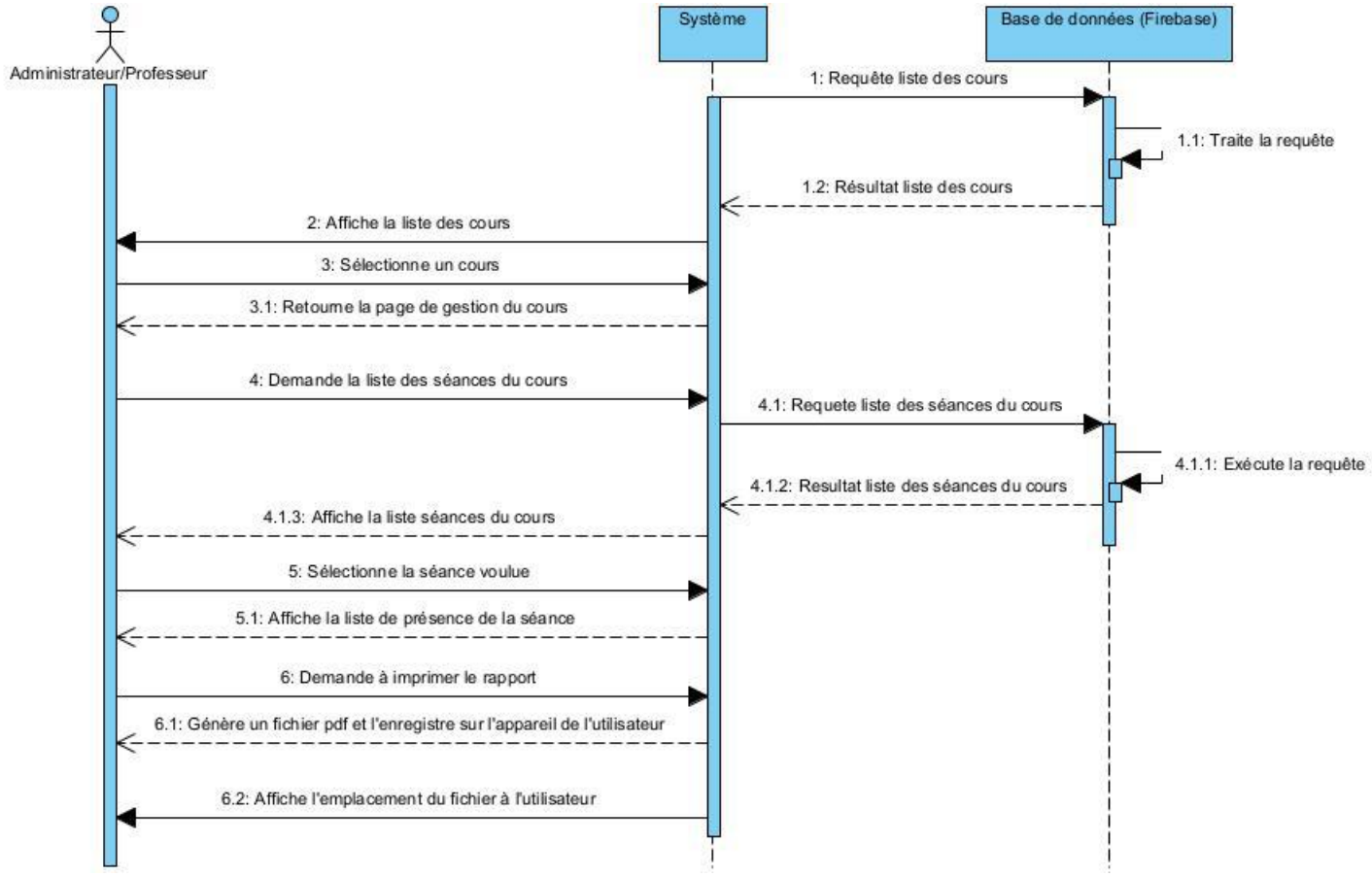


FIGURE 2.17 : Diagramme de Séquence Signer la Présence

2.6.6 Le diagramme de classes

Le diagramme de classes est l'un des types de diagrammes UML le plus utile, car il décrit clairement la structure d'un système particulier en modélisant ses classes, ses attributs, ses opérations et les relations entre ses objets[7]

Notre diagramme de classe se présente comme suit :

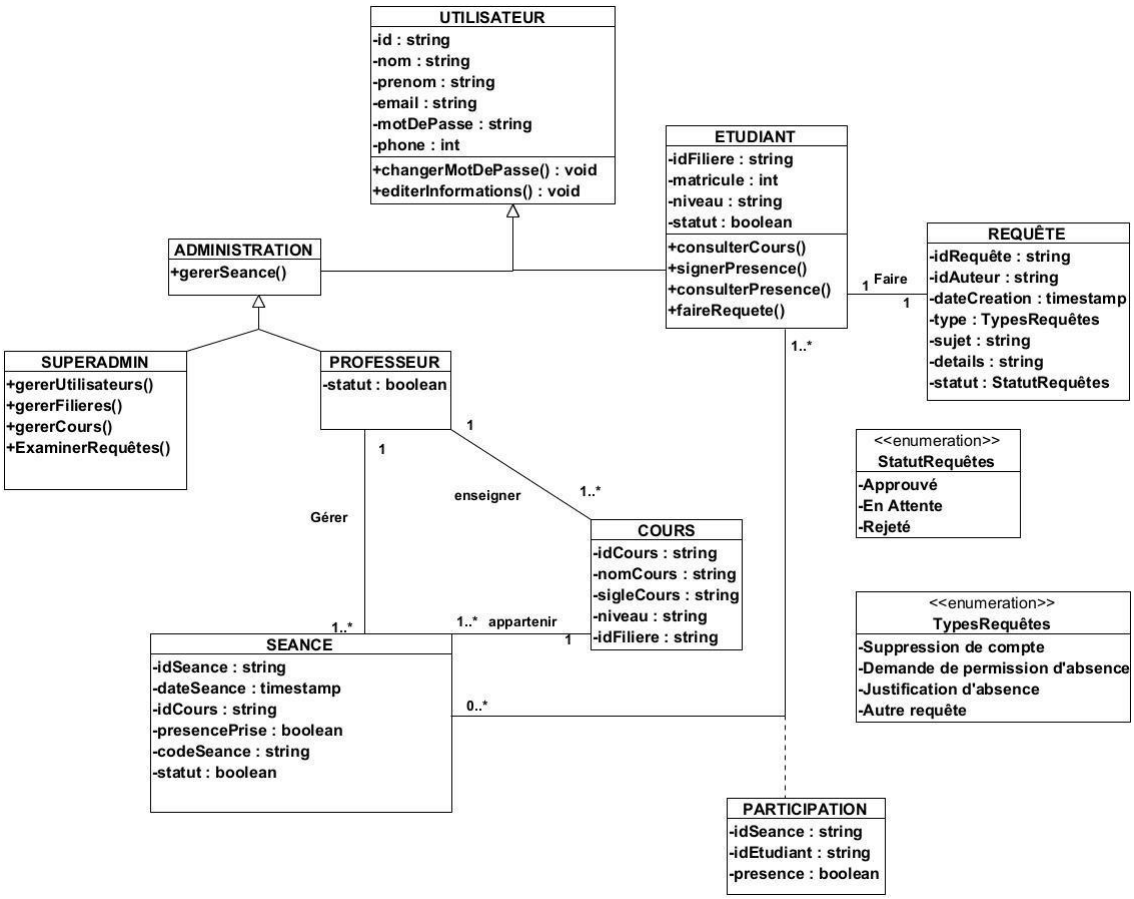


FIGURE 2.18 : Diagramme de Classe

Conclusion

Dans ce chapitre, nous avons présenté les différents diagrammes qui ont permis d'avoir l'architecture globale de notre projet ainsi que les différents outils utilisés. Dans le chapitre suivant nous présenterons les résultats de notre travail ainsi que les perspectives d'avenir.

Chapitre 3

Résultats et Discussion

Introduction

Ce chapitre présente le fruit de notre travail ainsi que ses limites et les perspectives d'avenir. L'interface dans son état actuel est en cours de perfectionnement mais l'application reste fonctionnelle.

3.1 Présentation de l'application

3.1.1 Schéma global de l'application

La figure suivante présente graphiquement le schéma global de l'application à travers les différents acteurs et leurs rôles principales dans l'application.

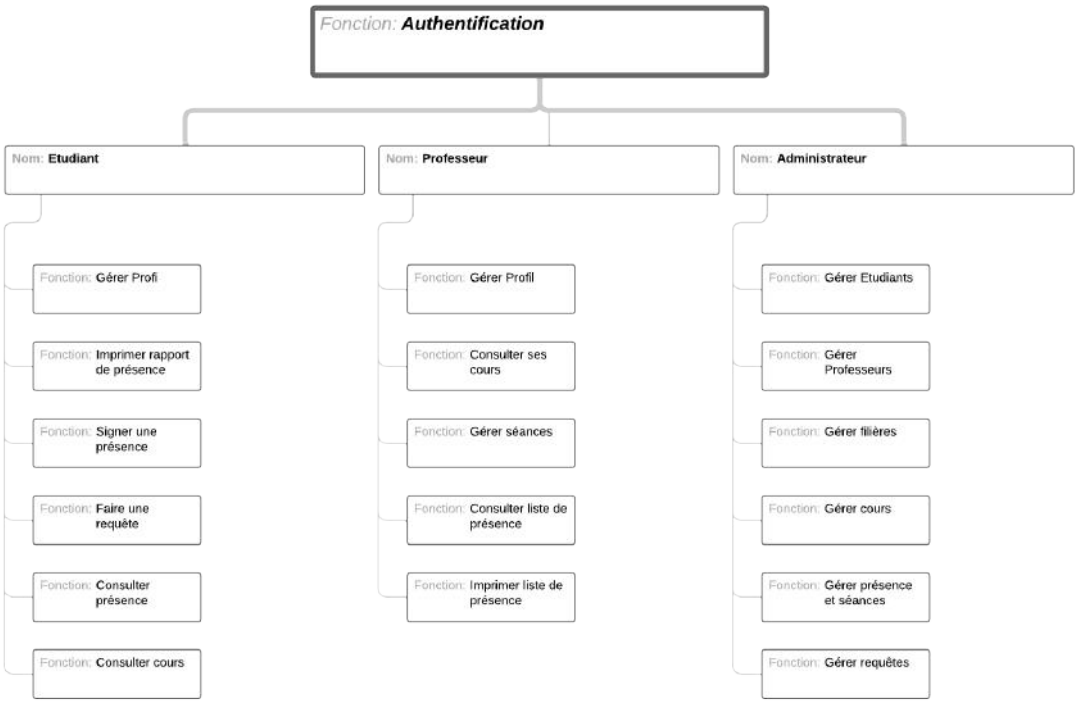


FIGURE 3.1 : Schéma global de l'application

3.1.2 Logo de l'application

Le logo de notre application est défini par une combinaison des mots "Easy"(Facile en Anglais) et "Attend"(Participer en Anglais) font référence symbolique à un marquage facile de sa participation, de sa présence. Ce visuel traduit clairement l'objectif de la plateforme qui est de faciliter le processus de signature de la présence.



FIGURE 3.2 : Logo de l'application

3.1.3 Pages d'Intro

L'utilisateur qui ouvre l'application pour la première fois sur son appareil verra cet écran d'introduction lui souhaitant la bienvenue dans l'application.

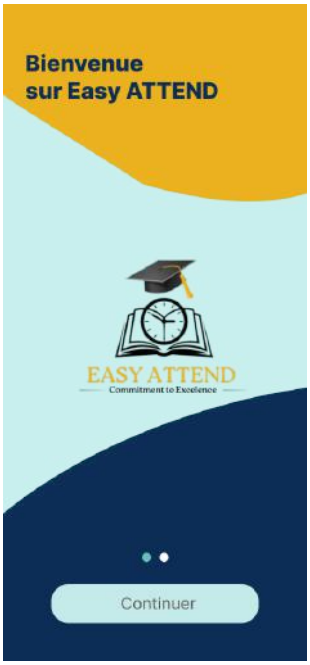


FIGURE 3.3 : Premier slider de la page d'intro



FIGURE 3.4 : Second slider de la page d'intro

3.1.4 Pages d'authentification

3.1.4.1 Page de connexion

L'écran de connexion permet aux utilisateurs d'accéder à l'application. L'utilisateur sélectionne son établissement et son rôle, saisit ses identifiants afin d'accéder à son compte.



FIGURE 3.5 : Page de pré connexion



FIGURE 3.6 : Page de connexion

3.1.4.2 Page d'inscription

Pour les nouveaux administrateurs, une page d'inscription est disponible afin de créer un compte.



FIGURE 3.7 : Page d'inscription administrateur

3.1.5 Gestion de profil

La page profil permet aux utilisateurs de modifier leurs informations personnelles à savoir : photo, numéro de téléphone ou mot de passe par exemple. Il leur permet également de consulter les politiques de confidentialité ainsi que les informations sur le développeur.

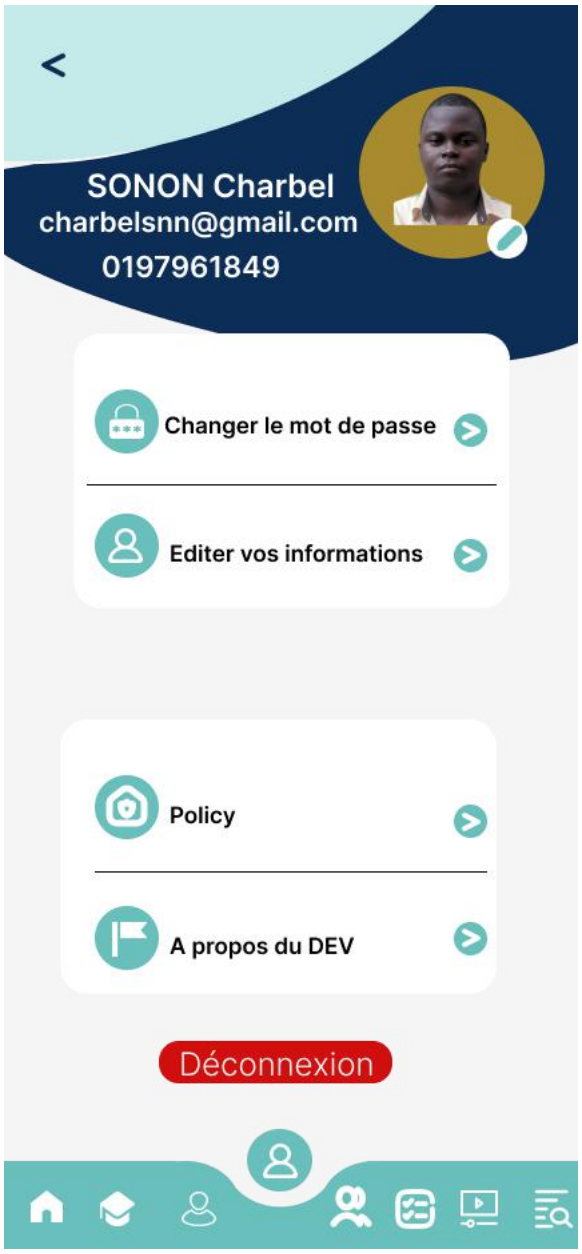


FIGURE 3.8 : Page des paramètres

3.1.6 Interface Administrateur

Nous présentons ici l'interface des administrateurs.

3.1.6.1 Dashboard

Le tableau de bord de l'administrateur regroupe des statistiques sur le nombre d'étudiants inscrits,le nombre de professeurs, le nombre de cours et le nombre de requêtes en attente de traitement.



FIGURE 3.9 : Dashboard Administrateur

3.1.6.2 Gestion Filieres

La page de gestion des filières permet à l'administrateur de consulter, créer, supprimer et modifier les filières.



FIGURE 3.10 : Gestion de filières



FIGURE 3.11 : Création de filière

3.1.6.3 Gestion Etudiants

La page de gestion des étudiants permet à l'administrateur de consulter la liste des étudiants avec des filtres, d'ajouter de nouveaux étudiants individuellement ou via un fichier Excel, ainsi que de modifier, supprimer ou restaurer un étudiant.



FIGURE 3.12 : Gestion d'étudiants

Pour ajouter un étudiant, l'administrateur peut le faire à travers un formulaire ou ajouter plusieurs étudiants depuis un fichier.



FIGURE 3.13 : Ajout d'étudiant par formulaire



FIGURE 3.14 : Ajout d'étudiants par fichier

3.1.6.4 Gestion Professeurs

La page de gestion des professeurs permet à l'administrateur de consulter, ajouter, modifier, supprimer ou restaurer des professeurs.



FIGURE 3.15 : Gestion des professeurs



FIGURE 3.16 : Ajout d'un nouveau professeur

3.1.6.5 Gestion Cours

Cette section permet à l'administrateur de consulter les cours selon les filtres disponibles, d'ajouter des cours en choisissant la filière, le niveau et le professeur, ainsi que de supprimer des cours.



FIGURE 3.17 : Gestion des cours



FIGURE 3.18 : Ajout d'un nouveau cours

3.1.6.6 Gestion Requêtes

Les administrateurs peuvent consulter et examiner les différentes demandes des étudiants, telles que les permissions d'absence, les justifications d'absence ou tout autre type de requête. Ils ont la possibilité de les approuver, de les rejeter ou de les mettre en attente.

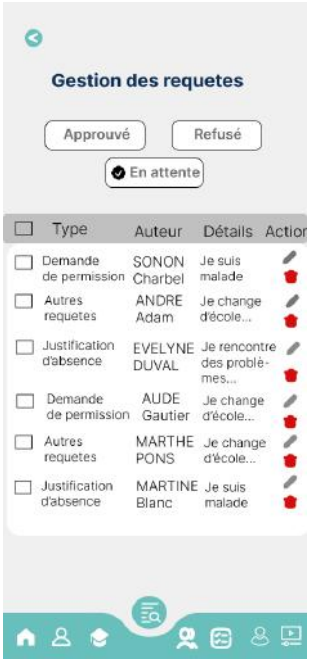


FIGURE 3.19 : Gestion des requêtes

3.1.7 Interface Professeur

Nous présentons ici l'interface des professeurs lorsqu'ils sont connectés

3.1.7.1 Dashboard

Le professeur connecté peut accéder à la liste des cours qui lui sont assignés via cette interface de tableau de bord. Il a aussi un aperçu statistique du nombre de filières qu'il tient ainsi que du nombre de cours à sa charge.



FIGURE 3.20 : Dashboard Professeur

3.1.7.2 Cours et Séances

Ici nous avons l'interface affichant la liste des séances d'un cours. Pour chaque séance, le professeur peut consulter la présence, prendre une présence simple ou une présence QR Code. Le professeur peut également supprimer la séance.Le professeur peut également créer de nouvelles séances.



FIGURE 3.21 : Gestion des Séances d'un cours



FIGURE 3.22 : Création de séance

Dans la section nous présenterons l'interface permettant de prendre la présence de deux manières possibles : en utilisant un code QR ou de manière plus interactive en cochant le nom des étudiants.

3.1.7.3 Prise de présence

Le professeur peut prendre la présence de deux manières : en utilisant un code QR ou en cochant manuellement les noms des étudiants sur une liste. Ces deux manières offrent une flexibilité appréciable dans la gestion des présences. Avec le code QR, les étudiants peuvent simplement scanner leur code personnel à l'entrée de la classe, ce qui permet un enregistrement rapide et efficace. Cette méthode réduit également le risque d'erreurs liées à une lecture manuelle. D'un autre côté, la liste manuelle peut être utile dans des situations où la technologie pourrait faire défaut ou en cas de difficulté d'accès à un réseau.



FIGURE 3.23 : Prise de présence Code QR

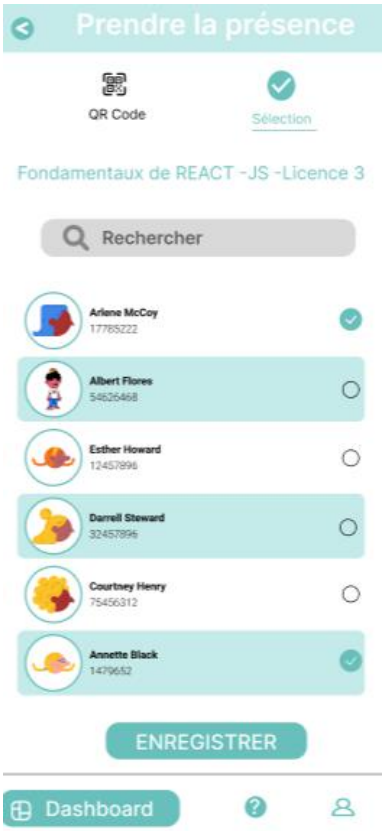


FIGURE 3.24 : Prise de présence simple

En outre, chaque méthode a ses propres avantages. Le code QR facilite la collecte des données, permettant au professeur de suivre les tendances d'assiduité et d'identifier rapidement les absents. Cela peut également aider à prévenir la fraude, car chaque étudiant doit être présent pour scanner son code. Cependant, la méthode manuelle, bien que plus lente, garde un aspect humain dans le processus, favorisant l'interaction entre le professeur et ses étudiants. Dans tous les cas, l'important est de choisir la méthode qui s'adapte le mieux au contexte et aux besoins de la classe.

3.1.7.4 Consultation de présence

Les images suivantes illustrent les interfaces permettant au professeur de consulter la liste des étudiants d'un cours, de vérifier la présence à une séance et d'observer la présence générale d'un étudiant dans l'ensemble du cours.

Consultation de la liste des étudiants du cours(de la filière)

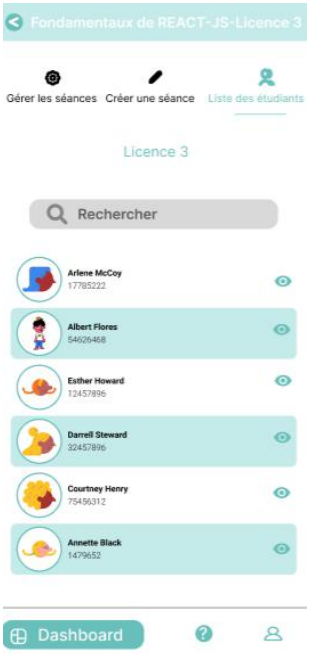


FIGURE 3.25 : Liste des étudiants du cours

Consultation du rapport de présence général d'un étudiant à un cours et de celui d'une séance

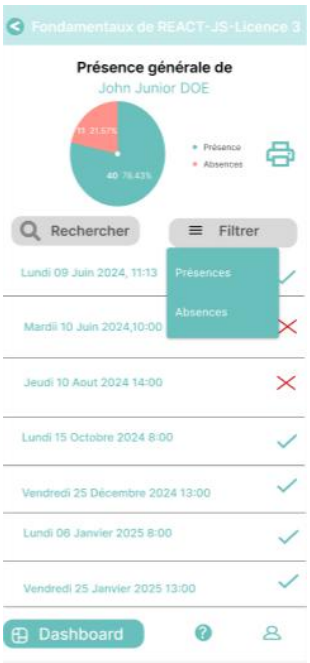


FIGURE 3.26 : Présence générale d'un étudiant

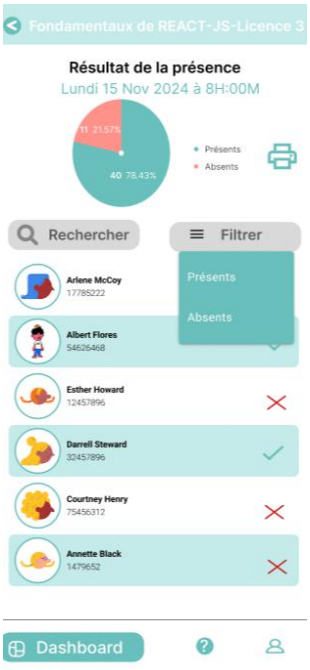


FIGURE 3.27 : Résultat de la présence d'une séance

3.1.8 Interface Etudiant

Nous présentons ici l'interface Etudiant

3.1.8.1 Dashboard

Ceci est le tableau de bord d'un étudiant lui permettant de visualiser les cours auxquels il doit assister selon sa formation, sa filière. Il a aussi un aperçu statistique du nombre de cours ainsi que sa requête auprès de l'administration si éventuellement il en avait.



FIGURE 3.28 : Dashboard Etudiant

3.1.8.2 Faire une requête

Cette page permet à un étudiant de faire une demande auprès de l'administration ou de consulter l'état d'avancement d'une demande qu'il a déjà soumise.



FIGURE 3.29 : Faire Une requête

3.1.8.3 Signer Présence

Cette interface permet à un étudiant de scanner un code QR pour signer sa présence, avec des instructions pour l'aider à effectuer cette tâche. Si l'étudiant est trop éloigné de l'établissement, il recevra un message d'erreur.



FIGURE 3.30 : fig :ScanPresence

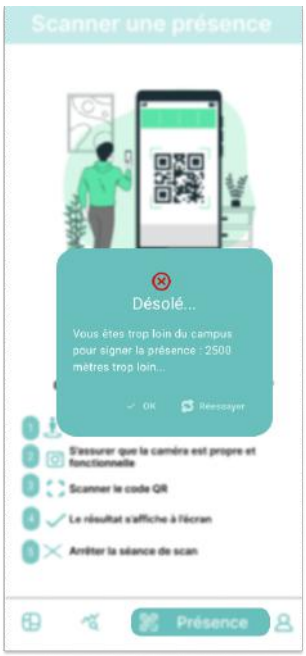


FIGURE 3.31 : Résultat du scan du code QR

3.1.8.4 Consulter présence

Cette page permet à un étudiant de consulter son rapport de présence à un cours donné.

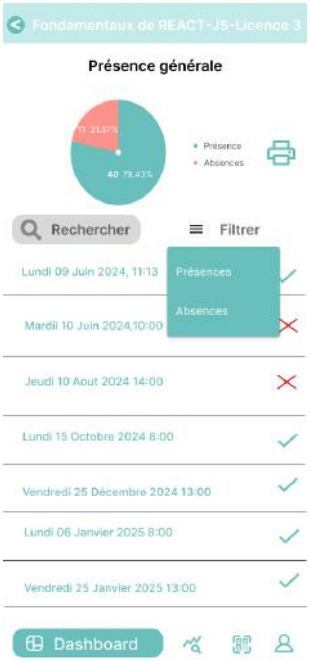


FIGURE 3.32 : Consulter Présence

3.2 Discussion

Notre solution EasyAttend, est conçue pour apporter une aide à l'administration universitaire dans la gestion de la présence et de l'assiduité des étudiants. Notre solution offre la possibilité aux différents acteurs(Administration, Etudiants, Professeurs) d'abandonner les papiers ou registres physiques ainsi que les fichiers Excels afin de centraliser les données au sein d'une base de données commune et fiable. Les interfaces sont simples et l'application est facile d'utilisation. Vu les résultats obtenus nous pouvons dire que notre solution permet d'enregistrer et de consulter les informations de présence concernant les étudiants. Elle permet aussi aux étudiants de faire des requêtes auprès de l'administration. Néanmoins, elle présente des insuffisances qu'il y a lieu de corriger lors de futures mises à jour.Il s'agit de :

1. Notre application n'est actuellement optimisée que pour une seule université
2. L'absence d'emploi du temps incorporé
3. La sécurité de l'application peut encore être améliorée

Conclusion

Dans ce chapitre on a exposé le fruit de notre travail. Les résultats obtenus au terme de la phase de développement de l'application ont été présentés sous forme de captures d'écran et un rapport des insuffisances a également été présenté.

Conclusion Générale

Dans ce travail, nous avons proposé une solution capable de faciliter le suivi de la présence des étudiants au sein des universités. Le processus existant peut être considéré comme toujours traditionnel car il se base encore sur l'utilisation des feuilles de papier ou des fichiers Excel, ce qui cause des problèmes d'inefficacité, d'erreur et de perte de temps. Notre thème de mémoire a alors consisté à proposer une solution permettant d'améliorer ce processus. Nous avons conçu une plateforme qui permet d'automatiser le processus, de réduire les erreurs, centraliser les données et de contribuer à améliorer la rétention et la persévérance des étudiants. Afin d'atteindre les objectifs du projet, nous avons utilisé une technique populaire et très utilisée dans des applications modernes. Il s'agit de la technologie des codes QR. La prise de présence avec code QR est l'option la moins chère et la plus adaptée parmi les techniques connues. Nous estimons que nous avons réalisé une bonne étude du problème. Nous avons abouti à une bonne première version de la solution proposée. Comme suite à ce travail, nous avons comme perspectives :

1. Elargir la solution à plusieurs établissements
2. Ajouter une couche de sécurité basée sur la reconnaissance faciale
3. Intégrer un système de messagerie
4. Ajouter de nouvelles fonctionnalités de prise de présence optimisées pour les cours en ligne

Bibliographie

[1] S. B. K. S. H. E Varadharajan, R Dharani. “automatic attendance management system using face detection”. 2016.

[2] Y. W. Lun. "smart attendance system using qr code". 2019.

Webographie

[3] Nicolas Belhamri. "présentation firebase". https://www.boryl.fr/glossaire/firebase/, Consulté le 22 Avril 2024.

[4] Chiny. "qu'est ce que node js ?". https://www.chiny.me/qu-est-ce-que-node-js-21-1.php, Consulté le 16 Mai 2024.

[5] EasyLog. "easylog qr booking app". https://clocking.easylog.co.uk/job-recording-with-qr-code-apps/, Consulté le 19 Avril 2024.

[6] Ionos. "qu'est ce qu'un qr code". https://www.ionos.fr/digitalguide/web-marketing/vendre-sur-internet/quest-ce-quun-qr-code/, Consulté le 22 Avril 2024.

[7] Lucidchart. "le diagramme de classe". https://www.lucidchart.com/pages/fr/diagramme-de-classes-uml, Consulté le 01 Mai 2024.

[8] Lucidchart. "le langage uml". https://www.lucidchart.com/pages/fr/langage-uml, Consulté le 23 Avril 2024.

[9] SWOOD Partners. "api rest vs graphql". https://swoodpartners.medium.com/api-rest-vs-graphql-diff%C3%A9rences-et-similarit%C3%A9s-7a45f22b7196, Consulté le 16 Mai 2024.

[10] Zoho People. "zoho people attendance management system using face detection". https://www.zoho.com/people/attendance-tracker.html, Consulté le 19 Avril 2024.

[11] remy manu. "le diagramme de séquence". http://remy-manu.no-ip.biz/UML/Cours/coursUML5.pdf, Consulté le 29 Avril 2024.

[12] EducPro Solution. "educpro solution". https://www.educprosolution.com/, Consulté le 09 Novembre 2024.

[13] StatCounter. "comparaison des systèmes d'exploitation mobile au bénin". https://gs.statcounter.com/os-market-share/mobile/benin/#monthly-200901-202404-bar, Consulté le 22 Avril 2024.

[14] Wikipedia. "mysql". https://fr.wikipedia.org/wiki/MySQL, Consulté le 16 Mai 2024.

[15] L'équipe éditoriale. "la méthodologie scrum". https://www.nutcache.com/fr/blog/methodologie-scrum/, Consulté le 10 Juin 2024.

Table des matières

Dédicace ii

Remerciements iii

Résumé iv

. iv

Abstract v

. v

Liste des figures vi

Liste des tableaux viii

Liste des acronymes ix

Introduction 1

1 Etat de l'art 3

Introduction 3

1.1 Les solutions existantes 3

1.2 Comparaison des technologies de suivi de présence 6

Conclusion 7

1.3 Présentation de notre Application Solution *"EasyAttend"* 8

1.3.1 Objectifs de notre application 8

2 Analyse des choix techniques et Conception 10

Introduction 10

2.1 Analyse des choix techniques 10

2.1.0.1 Frontend 10

2.1.1 BACKEND 11

2.1.1.1 Authentification 11

2.1.1.2 Système de Gestion de Base de données (SGBD) 11

2.1.1.3 API 12

2.1.1.4 Serveur avec Node.Js 12

2.1.1.5 WampServer 13

2.1.2 Environnement de développement :Visual Studio Code 13

2.2 Choix de la plateforme client 13

2.3 Choix de la plateforme mobile 14

2.4 Le Code de Réponse Rapide (QR) . . . 14
2.5 La méthodologie du projet . . . 15
2.5.1 La méthodologie SCRUM . . . 15
2.6 Modélisation . . . 16
2.6.1 Le langage UML . . . 16
2.6.2 Identification des acteurs . . . 16
2.6.2.1 Rôle des acteurs . . . 16
2.6.3 Le Diagramme de cas d'utilisation . . . 18
2.6.3.1 Définition . . . 18
2.6.4 Les Scénarios . . . 18
2.6.4.1 Scénario "Authentification" . . . 19
2.6.4.2 Scénario "Lancement de la présence" . . . 20
2.6.4.3 Scénario "Signer la présence" . . . 21
Conclusion . . . 21
2.6.5 Les diagrammes de Séquence . . . 21
2.6.5.1 Définition . . . 21
2.6.5.2 Diagramme de Séquence «Authentification» . . . 22
2.6.5.3 Diagramme de séquence «Ajouter Etudiant» . . . 23
2.6.5.4 Diagramme de séquence «Signer La Presence» . . . 24
2.6.5.5 Diagramme de séquence «Marquer la présence manuellement» . . . 25
2.6.5.6 Diagramme de séquence «Consulter et Imprimer la présence d'une séance» . . . 26
2.6.6 Le diagramme de classes . . . 27

3 Résultats et Discussion 28
Introduction . . . 28
3.1 Présentation de l'application . . . 28
3.1.1 Schéma global de l'application . . . 28
3.1.2 Logo de l'application . . . 29
3.1.3 Pages d'Intro . . . 30
3.1.4 Pages d'authentification . . . 30
3.1.4.1 Page de connexion . . . 30
3.1.4.2 Page d'inscription . . . 31
3.1.5 Gestion de profil . . . 32
3.1.6 Interface Administrateur . . . 33
3.1.6.1 Dashboard . . . 33
3.1.6.2 Gestion Filieres . . . 34
3.1.6.3 Gestion Etudiants . . . 34
3.1.6.4 Gestion Professeurs . . . 35
3.1.6.5 Gestion Cours . . . 36
3.1.6.6 Gestion Requêtes . . . 36
3.1.7 Interface Professeur . . . 37
3.1.7.1 Dashboard . . . 37
3.1.7.2 Cours et Séances . . . 38
3.1.7.3 Prise de présence . . . 39

3.1.7.4 Consultation de présence 40
3.1.8 Interface Etudiant 41
3.1.8.1 Dashboard 41
3.1.8.2 Faire une requête 42
3.1.8.3 Signer Présence 42
3.1.8.4 Consulter présence 43
3.2 Discussion 43
Conclusion 43

Conclusion **44**

Bibliographie **45**

Webographie **46**

Table des matières **47**