



ECOLE MAROCAINE DES
SCIENCES DE L'INGENIEUR
Membre de
HONORIS UNITED UNIVERSITIES

IIR

INGÉNIERIE INFORMATIQUE & RÉSEAUX

ANNÉE UNIVERSITAIRE

2025

OPTION

MÉTHODES INFORMATIQUES APPLIQUÉES
À LA GESTION DES ENTREPRISES (MIAGE)

PROJET DE FIN D'ANNÉE

THÈME

Application (web+mobile) pour la gestion des
devoirs et des notes

RÉALISÉ PAR

ZAOUIA Mouad - ZAKI ELIDRISSI Abdallah

HNKICH Reda – BOUDRIGA Ismail

ENCADRÉ PAR

NOM DE L'ENCADRANTE 1 : BOUSQAOUI HALIMA

NOM DE L'ENCADRANTE 2 : NADIRI ABDELJALIL

NOM DE L'ENCADRANTE 3 : ESSABAR DRISS

NOM DE L'ENCADRANTE 4 : CHAREF AYOUB

Remerciement

Je souhaite exprimer ma profonde gratitude à Madame BOUSQAoui Halima, professeure de Java EE à l'EMSI Marrakech, pour son accompagnement exemplaire tout au long de la réalisation de ce projet. Son expertise technique, sa pédagogie remarquable et son engagement constant ont été des atouts majeurs dans l'orientation et l'avancement de mon travail. Madame BOUSQAoui n'a cessé de faire preuve d'une grande disponibilité et d'un professionnalisme sans faille, en apportant des conseils judicieux, des remarques pertinentes et un regard critique toujours constructif. Grâce à ses interventions, j'ai pu renforcer ma maîtrise des concepts avancés du développement Java EE, améliorer la qualité de mon code, et acquérir une rigueur méthodologique essentielle dans le cadre d'un projet informatique structuré. Au-delà de l'aspect technique, j'ai également beaucoup appris en matière de gestion de projet, d'autonomie et de persévérance, éléments que Madame BOUSQAoui a su encourager avec bienveillance. Je lui suis infiniment reconnaissant pour sa confiance, son soutien constant et l'inspiration qu'elle représente en tant qu'enseignante et encadrante.

Abstract

This document presents the summary of my end-of-studies project, which focuses on the design and development of a web and mobile application for managing homework assignments and student grades. The primary objective of this project is to offer a simple, intuitive, and efficient digital platform that facilitates communication between teachers and students regarding course activities and evaluations.

The application is structured around two main user roles: teachers and students. Teachers can create and manage courses, assign homework, collect student submissions, and grade assignments. Students, on the other hand, can join courses using a code, submit homework, and consult their grades and feedback. The platform aims to streamline academic workflow by centralizing data and automating repetitive tasks.

Developed using **React** and **Tailwind CSS** for the frontend and **Android Native** for the mobile application, the system features a responsive and modern user interface. Although this version does not include backend functionality, it establishes a solid foundation for future integration with technologies such as Spring Boot and PostgreSQL.

This project highlights the importance of user-centered design and modular architecture in building educational tools that are adaptable, maintainable, and scalable for real-world use.

Keywords: Homework Management, Grade Tracking, Educational Platform, React, Tailwind CSS, User Roles, UI/UX Design.

Résumé

Le présent document résume le travail réalisé dans le cadre de mon Projet de Fin d'Études Appliqué (PFA), qui a porté sur la conception et le développement d'une application web et mobile de gestion des devoirs et des notes. Ce projet vise à faciliter la communication et le suivi pédagogique entre enseignants et élèves, en proposant une solution numérique intuitive, centralisée et accessible.

L'application, développée en **React** avec **Tailwind CSS** pour le frontend, et en **Android Native** pour la version mobile, permet aux enseignants de créer des cours, d'attribuer des devoirs, de collecter les soumissions et d'attribuer des notes. De leur côté, les élèves peuvent consulter les cours auxquels ils sont inscrits, soumettre leurs devoirs et suivre leurs notes. L'interface est conçue de manière responsive et ergonomique, avec des animations douces assurées par **Framer Motion** pour améliorer l'expérience utilisateur.

Même si le projet a été développé dans un environnement statique (sans backend fonctionnel pour l'instant), une attention particulière a été portée à l'architecture du système, à l'organisation des composants, ainsi qu'à la gestion des rôles (professeur / élève) et à l'expérience utilisateur.

Ce projet constitue une base solide pour une future version pleinement fonctionnelle, intégrant une authentification sécurisée, une base de données relationnelle et une gestion dynamique des interactions en temps réel.

Mots-clés : Gestion des devoirs, Suivi pédagogique, Application web et mobile, React, Android Native, Interface responsive, Enseignant, Élève.

Liste des abréviations

PFA	Projet de Fin d'Année
UI	Interface Utilisateur
UX	Expérience Utilisateur
HTTP	Protocole de transfert hypertexte
JSON	JavaScript Object Notation
API	Interface de programmation d'application
UML	Unified Modeling Language
API REST	Interface de Programmation Applicative RESTful
JWT	JSON Web Token
SCRUM	Méthode agile

Liste des figures

Figure 1 Cas d'utilisation : Gestion des devoirs et des notes.....	15
Figure 2 Diagramme de séquence (Partie Professeur).....	16
Figure 3 Diagramme de séquence (Partie Élève)	17
Figure 4 Diagramme de classe.....	19
Figure 5 Page d'authentification.....	31
Figure 6 Professeur Dashboard.....	32
Figure 7 Créer un cour.....	33
Figure 8 L'affichage des cours.....	33
Figure 9 Gestion des cours	34
Figure 10 Gestion des parametres	34
Figure 11 Gestion de profile (Professeur)	35
Figure 12 Élève Dashboard	36
Figure 13 Les cours d'un Élève	36
Figure 14 l'affichage des devoirs d'un cour	37
Figure 15 L'affichage des notes.....	37
Figure 16 Gestion de profile (Élève)	38
Figure 17 Login (Partie mobile)	39
Figure 18 Éleve Dashboard	40
Figure 19 Les cours d'un Éleve	41

Table des matières

Remerciement	2
Abstract.....	3
Résumé	4
Liste des abréviations.....	5
Introduction générale	1
Chapitre 1 : Contexte général du projet.....	2
1.1 Introduction.....	3
1.2 Description du Projet	3
1.2.1 Sujet du projet	3
1.2.2 Problématique.....	4
1.2.3 Objectifs.....	5
1.3 Étude fonctionnelle	6

1.3.1	Les besoins fonctionnels	6
1.3.2	Les besoins non fonctionnels	7
1.4	Démarche et Planification	8
1.4.1	La méthode SCRUM.....	8
1.4.2	Pourquoi SCRUM ?.....	8
Chapitre 2	: Analyse et conception	12
2.1	Introduction.....	13
2.2	Analyse et conception.....	13
2.2.1	Le formalisme UML.....	13
2.2.2	Le choix de UML	13
2.2.3	Identification des acteurs	14
2.2.4	Diagramme de cas d'utilisation.....	15
2.2.5	Diagramme des sequences.....	16
2.2.6	Diagramme De Classe	19
2.3	Conclusion	20
Chapitre 3	: Technologies et outils techniques.....	21
3.1	Introduction.....	22
3.2	Choix technologiques.....	22
3.2.1	Frontend Web – React.js	22
3.2.2	Frontend Mobile – Android native.....	23
3.2.3	Backend – Spring Boot	24
3.2.4	Base de données – MySQL	25

3.2.5	API REST – Communication entre frontend et backend	26
3.3	Choix Environnement de développement	27
3.3.1	VS Code	27
3.3.2	IntelliJ	28
3.3.1	Android Studio	28
3.4	Conclusion	29
Chapitre 4	: Réalisation	30
4.1	Introduction.....	31
4.1	Page d’authentification.....	31
4.4	Espace Professeur.....	32
4.4.1	Page Principal.....	32
4.4.2	Page de Création d’un cour	33
4.4.3	Page de l’affichage d’un cour.....	33
4.4.4	Page de gestion des devoirs.....	34
4.4.5	Page de gestion des paramètres.	34
4.4.6	Page de gestion de profile.	35
4.5	Espace Élève	36
4.5.1	Page de Page Principal	36
4.5.2	Page de des Cours	36
4.5.3	Page de l’affichage des devoirs d’un cour	37
4.5.4	Page de l’affichage des notes	37
4.5.5	Page de gestion de profile.	38

4.6	Partie Mobile	38
4.6.1	Page Login :	39
4.6.2	Page d'accueil :	39
4.6.3	Page des cours :	41
4.7	Conclusion.....	41
	Conclusion et perspective	42
	Webographie.....	43

Introduction générale

Dans un contexte éducatif en constante évolution, la digitalisation des processus pédagogiques est devenue une nécessité pour répondre aux exigences de rapidité, d'efficacité et de transparence. Les établissements d'enseignement, qu'ils soient secondaires ou supérieurs, cherchent à moderniser la gestion des cours, des devoirs et du suivi des étudiants à travers des solutions numériques innovantes.

C'est dans cette optique que s'inscrit le présent projet, qui porte sur la **conception et le développement d'un système de gestion des devoirs et des notes**. Ce projet vise à faciliter la communication entre les enseignants et les élèves, en offrant une plateforme centralisée permettant de créer, distribuer, soumettre et corriger les devoirs en ligne. L'objectif est de simplifier le suivi pédagogique, d'optimiser le temps de gestion administrative des enseignants et de renforcer l'implication des élèves dans leur apprentissage.

La version actuelle de l'application repose sur une interface **frontend développée avec React et Tailwind CSS**, assurant une expérience utilisateur moderne, responsive et intuitive. Une application mobile native Android a également été conçue pour répondre aux besoins des utilisateurs en mobilité. Bien que cette version ne comporte pas encore de logique backend, elle constitue une base solide pour une future intégration avec des technologies comme **Spring Boot** pour le backend et **MySQL** pour la gestion des données.

À travers ce projet, nous visons non seulement à répondre à un besoin réel du monde éducatif, mais aussi à mettre en œuvre les connaissances techniques acquises durant notre formation, tout en adoptant une démarche rigoureuse de conception logicielle.

Chapitre 1 : Contexte général du projet

1.1 Introduction

Ce chapitre introduit le projet en le situant dans son cadre éducatif et technologique. Nous présenterons d'abord les défis liés à la gestion traditionnelle des devoirs et des notes, puis les besoins des enseignants et des élèves. Ensuite, nous décrirons le client, l'équipe projet, ainsi que les rôles et responsabilités. Enfin, nous aborderons l'étude des solutions existantes, la problématique à résoudre, la solution proposée, et l'organisation du projet.

1.2 Description du Projet

1.2.1 Sujet du projet

Le projet consiste en la création d'un système de gestion des devoirs et des notes, visant à automatiser et centraliser les interactions entre enseignants et élèves. Ce système permettra aux enseignants de gérer efficacement les remises de devoirs, les corrections et l'attribution des notes, tout en offrant aux élèves une visibilité claire sur leur progression et leurs échéances. Les fonctionnalités principales incluent la gestion des informations des cours et des utilisateurs, la soumission et la correction des devoirs, ainsi que l'affichage en temps réel des résultats et des notifications. En intégrant une base de données PostgreSQL pour le stockage des informations, le projet propose une interface web et mobile conviviale, facilitant le suivi scolaire et renforçant l'organisation des tâches pédagogiques. Ce système vise à améliorer l'efficacité du processus éducatif, tout en favorisant une gestion transparente et équitable des devoirs et des notes.

1.2.2 Problématique

La gestion traditionnelle des devoirs et des notes dans de nombreuses institutions éducatives repose encore sur des méthodes manuelles ou des outils non centralisés, tels que des carnets papier, des fichiers dispersés ou des communications informelles. Cela engendre plusieurs difficultés :

- **Manque de centralisation** : Les informations concernant les devoirs, les notes et les délais sont souvent éparpillées, ce qui complique leur suivi et gestion pour les enseignants et les élèves. Les enseignants doivent jongler entre divers canaux de communication pour collecter, corriger et attribuer les notes, ce qui est chronophage et source d'erreurs.
- **Manque de visibilité pour les élèves** : Les élèves ne disposent pas d'une vue claire et actualisée de leur progression, de leurs notes ou des échéances. Cela peut créer de la confusion, des oublis et un manque de motivation, car ils ne savent pas toujours où ils en sont dans leurs études.
- **Efficacité limitée** : La planification et la correction manuelles des devoirs entraînent une surcharge de travail pour les enseignants, avec un risque de retard dans les corrections et l'attribution des notes. De plus, cela augmente la probabilité d'erreurs humaines dans l'enregistrement des résultats.
- **Absence d'automatisation** : L'absence d'un système automatisé pour gérer la répartition des tâches et le suivi des travaux crée une lourdeur administrative. Chaque enseignant ou secrétaire doit prendre en charge un grand nombre de tâches répétitives, augmentant le temps consacré à des tâches administratives plutôt qu'à l'enseignement.

1.2.3 Objectifs

L'idée Les objectifs du projet de gestion des devoirs et des notes se déclinent comme suit :

- **Automatisation de la Gestion des Devoirs et des Notes** : Développer un système permettant l'automatisation de la soumission des devoirs, la correction et l'attribution des notes, en fonction de critères définis comme les délais, les types de devoirs et les performances des élèves. Cela permettra de réduire le temps consacré à la gestion manuelle et d'assurer une distribution équitable des responsabilités pédagogiques.
- **Gestion Efficace des Utilisateurs (Enseignants et Élèves)** : Créer une interface conviviale pour permettre aux enseignants de gérer facilement les informations des élèves, les devoirs soumis, ainsi que les résultats. Les élèves pourront également suivre leur progression et consulter leurs notes et échéances en temps réel.
- **Suivi et Reporting des Résultats Scolaires** : Mettre en place un système de suivi qui permet de consulter et d'analyser l'historique des devoirs, les performances des élèves, ainsi que des rapports détaillés sur la répartition des tâches pédagogiques. Cela permettra une meilleure gestion des évaluations et une planification proactive des devoirs.
- **Facilitation de la Communication** : Intégrer des fonctionnalités permettant aux enseignants de notifier les élèves de la soumission des devoirs, de l'attribution des notes, et d'informer rapidement des changements ou des retards dans le processus. Cela garantira une communication fluide et transparente entre les enseignants et les élèves.
- **Adaptabilité et Flexibilité** : Concevoir le système pour qu'il soit adaptable aux besoins spécifiques des enseignants et des élèves, et qu'il puisse gérer des imprévus comme des ajustements de date ou des modifications de devoirs.
- **Sécurisation des Données** : Assurer la protection des données personnelles des utilisateurs, y compris les résultats scolaires et les informations sensibles, en intégrant des mesures de sécurité adaptées et en respectant les normes de confidentialité.
- **Amélioration de l'Efficacité Pédagogique** : En automatisant la gestion des devoirs et des notes et en réduisant les erreurs humaines, le système vise à améliorer l'efficacité pédagogique, permettant aux enseignants de se concentrer davantage sur l'enseignement et moins sur les tâches administratives.

1.3 Étude fonctionnelle

L'étude fonctionnelle est effectivement une étape cruciale dans la conception d'un système d'information. Elle consiste à analyser les besoins des utilisateurs, à identifier les fonctionnalités attendues du système et à définir les spécifications fonctionnelles. Cette approche permet de se concentrer sur les objectifs à atteindre plutôt que sur les moyens pour y parvenir.

1.3.1 Les besoins fonctionnels

Les besoins fonctionnels décrivent les fonctionnalités spécifiques que le système doit offrir pour répondre aux attentes des utilisateurs. Voici quelques exemples :

1. Gestion des Utilisateurs

- Création, modification et suppression des comptes (professeurs et élèves).
- Authentification des utilisateurs (connexion par e-mail et mot de passe).
- Gestion des rôles (professeur/élève).

2. Gestion des Cours

- Création, modification et suppression des cours par les professeurs.
- Inscription des élèves aux cours via un code d'inscription.
- Affichage des cours pour les professeurs et les élèves.

3. Gestion des Devoirs

- Création et gestion des devoirs par les professeurs.
- Soumission des devoirs par les élèves (texte ou fichier).
- Correction des devoirs par les professeurs avec attribution des notes.

4. Suivi des Notes et Résultats

- Consultation des notes et feedback par les élèves.
- Suivi des devoirs et des performances par les professeurs.

5. Notifications

- Notifications des échéances, soumissions et corrections de devoirs.
- Alertes sur les changements ou retards des devoirs.

6. Interface Utilisateur

- Interface conviviale et intuitive pour les professeurs et les élèves.
- Navigation simple et claire entre les différentes sections.

7. Sécurité et Confidentialité

- Protection des données personnelles et des résultats scolaires.
- Sécurisation des informations sensibles avec cryptage.

1.3.2 Les besoins non fonctionnels

Les besoins non fonctionnels concernent les caractéristiques et les qualités du système qui ne sont pas directement liées à des fonctionnalités spécifiques, mais qui sont essentielles pour sa performance globale. Voici quelques exemples :

- **Performance**

- **Temps de réponse rapide** : L'application doit charger rapidement et répondre instantanément aux actions des utilisateurs pour garantir une expérience fluide.
- **Gestion du trafic élevé** : Le système doit être capable de gérer de nombreux utilisateurs simultanément (par exemple, plusieurs enseignants et élèves connectés en même temps) sans affecter la performance.
- **Optimisation des requêtes** : Les requêtes vers la base de données (pour les cours, devoirs, notes) doivent être optimisées pour éviter des délais de chargement importants.

- **Sécurité**

- **Protection des données sensibles** : Les informations personnelles des utilisateurs (comme les emails, les notes, les devoirs soumis) doivent être protégées contre tout accès non autorisé.
- **Authentification sécurisée** : Utiliser des mécanismes de sécurité solides comme des mots de passe cryptés et des options d'authentification tierce (par exemple, Google Identity) pour assurer que seuls les utilisateurs autorisés accèdent à leur compte.
- **Respect des normes de confidentialité** : L'application doit respecter les lois et normes de confidentialité des données personnelles (par exemple, le RGPD) pour protéger la vie privée des utilisateurs.

- **Accessibilité**

- **Conception responsive** : L'application doit s'adapter à tous les appareils (bureau, tablette, smartphone).
- **Compatibilité multi-navigateurs** : Fonctionnement sur les navigateurs populaires (Chrome, Firefox, Safari, Edge).
- **Interface claire et lisible** : Interface ergonomique et lisible pour tous les utilisateur

1.4 Démarche et Planification

1.4.1 La méthode SCRUM

La méthode Scrum est une méthode agile de gestion de projets informatiques privilégiant la communication, et facilitant les réorientations opportunes. C'est désormais la méthode privilégiée pour les démarches dites "agiles". Fort de son succès dans l'univers informatique, elle est maintenant déployée en entreprise comme nouvelle organisation du fonctionnement en "mode projet".

1.4.2 Pourquoi SCRUM ?

Scrum est de très loin la méthodologie la plus utilisée parmi les méthodes Agile existantes. Elle est donc la plus éprouvée, documentée et supportée. Livres, blogs, formations, vidéos, associations, conférences traitant de Scrum ne manquent pas et bon nombre de ces ressources sont accessibles gratuitement. On pourrait pratiquement parler d'un standard Agile. Un autre atout important : **Scrum** est simple à comprendre. Sa maîtrise est en revanche difficile.

Les experts de Scrum, même ses fondateurs, le décrivent comme un « cadre de travail permettant de répondre à des problèmes complexes et changeants tout en livrant de manière productive et créative des produits de la plus grande valeur possible » Scrum propose un modèle de contrôle de processus basé sur l'empirisme. Il s'appuie sur trois piliers.

- **La transparence** : Scrum met l'accent sur le fait d'avoir un langage commun entre l'équipe et le management, qui doit permettre à tout observateur d'obtenir rapidement une bonne compréhension du projet.
- **L'inspection** : Scrum propose de faire le point sur les différents artefacts produits à intervalle régulier, afin de détecter toute variation indésirable.
- **L'adaptation** : Si une dérive est constatée pendant l'inspection, le processus doit alors être adapté. Scrum fournit des rituels, durant lesquels cette adaptation est possible. Il s'agit de sprint planning, de daily scrum, et de sprint review qu'on va détailler dans la section suivante.

1.5 Identification des sprints

L'identification des sprints consiste à prévoir les missions du projet tout au long des phases consistant le cycle de développement.

Grâce aux réunions tenues avec les profs, les différentes phases du projet ainsi que leur déroulement ont été clarifiées selon l'approche Agile, le tableau suivant décrit les durées, les objectifs ainsi que les activités effectuées durant les sprints :

Durée globale de ce projet est : *du 21 Mars 2025 au 10 May 2025.*

Tableau 1 :Identification des Sprints

SPRINT	OBJECTIFS	TÂCHES
Sprint 0	Collecte et analyse des besoins	<ul style="list-style-type: none"> ✓ Définir les objectifs du projet ✓ Identifier les acteurs (Professeur, Élève) ✓ Recueillir les besoins fonctionnels et non-fonctionnels ✓ Rédiger le cahier des charges ✓ Préparer l'environnement de développement (React.js, Android Studio, Spring Boot, PostgreSQL) ✓ Réaliser un prototype UI (Figma ou autre)
Sprint 1	Authentification & Création de cours	<ul style="list-style-type: none"> ✓ Spécifier les besoins ✓ Concevoir les interfaces de login/inscription ✓ Implémenter l'authentification JWT ✓ Créer la fonctionnalité "Créer un cours avec code auto-généré" côté professeur
Sprint 2	Accès aux cours et soumission de devoirs	<ul style="list-style-type: none"> ✓ Implémenter la fonctionnalité "Rejoindre un cours avec code" pour l'élève ✓ Implémenter l'upload de devoirs (fichier/texte) côté élève ✓ Gérer l'enregistrement des soumissions côté backend
Sprint 3	Correction et notation des devoirs	<ul style="list-style-type: none"> ✓ Permettre au professeur de voir les devoirs soumis ✓ Implémenter l'attribution de notes ✓ Stocker les notes dans la base de données

Sprint 4	Consultation des notes et finalisation	✓ Implémenter la page "Mes notes" côté élève ✓ Connecter à l'API de récupération des notes ✓ Tests finaux (fonctionnels, UI, backend) ✓ Préparation à la soutenance et documentation finale
----------	--	--

1.6 Identification des sprints

L'identification La planification est une phase d'avant-projet essentielle. Elle consiste à établir une chronologie claire des différentes étapes du projet et à estimer les délais associés à chacune d'elles. Cela permet de structurer le travail, d'en assurer le bon déroulement et de faciliter le suivi tout au long de sa réalisation.

Pour modéliser le déroulement de mon projet, j'ai utilisé un **diagramme de GANTT**, un outil efficace de planification et d'ordonnancement des tâches dans le temps. Grâce à sa représentation visuelle simple et claire, ce diagramme m'a permis de suivre précisément l'état d'avancement du projet, de visualiser la séquence des tâches, leur durée et l'organisation générale du calendrier de développement.

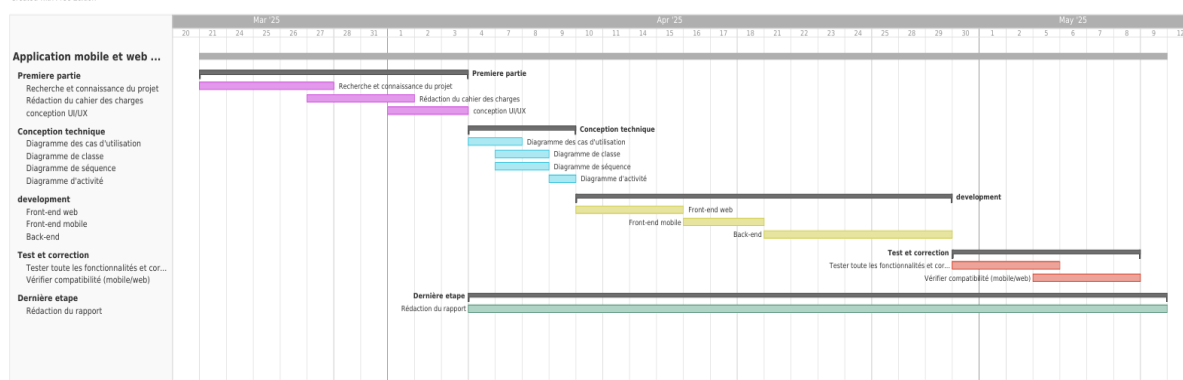
Afin d'organiser le travail de manière structurée, le projet a été découpé en **plusieurs phases**, elles-mêmes divisées en **tâches et sous-tâches**. La première étape a été consacrée à la **recherche et à la compréhension du projet**, suivie par la **rédaction du cahier des charges** et la **conception UI/UX** de l'application.

La deuxième phase portait sur la **modélisation technique**, avec la réalisation des diagrammes UML (cas d'utilisation, classes, séquence, activité), avant de passer à la phase de **développement** de l'application :

- **Frontend web** avec React.js
- **Frontend mobile** avec Android natif
- **Backend** avec Spring Boot et PostgreSQL

Enfin, une phase de **test et de correction** a été menée afin de s'assurer du bon fonctionnement des fonctionnalités sur les deux plateformes, suivie par la **rédaction du rapport final**.

Le diagramme de Gantt ci-dessous illustre cette planification sur l'ensemble de la période du projet.



1.7 Conclusion

Dans ce chapitre, nous avons présenté le contexte général du sujet de mon projet de fin d'étude, sa problématique, son objectif, et sa gestion. Dans le deuxième chapitre, nous allons procéder à l'analyse et la conception du projet.

Chapitre 2 : Analyse et conception

2.1 Introduction

Dans ce chapitre, nous nous consacrerons sur l'étude et l'analyse fonctionnelle du projet, étape clé dans le processus de conception et de réalisation. Cette phase a pour objectif de définir précisément les besoins du client, ainsi que les exigences fonctionnelles et techniques du système à mettre en œuvre. Nous y analyserons les différents acteurs du système, établirons le diagramme de cas d'utilisation, ainsi que le diagramme de séquence et le diagramme de classe. Ce chapitre mettra également en lumière le périmètre d'intervention, déterminant ainsi l'étendue de notre domaine d'action.

2.2 Analyse et conception

2.2.1 Le formalisme UML

UML (Unified Modeling Language) est un langage de modélisation visuel utilisé par les développeurs pour représenter graphiquement les objets, états et processus dans un logiciel ou un système. Il aide à créer un modèle structuré d'un projet, facilitant la communication avec des spécialistes externes. Bien qu'il soit principalement utilisé pour le développement de logiciels orientés objet, la version 2.0 de UML est aussi adaptée pour représenter des processus de gestion. UML n'étant pas une méthode, l'utilisation des diagrammes dépend des préférences des utilisateurs. Le diagramme de classes est central dans UML, et certaines méthodes, comme le processus unifié, utilisent l'ensemble des diagrammes pour développer des modèles par itérations successives. D'autres approches préfèrent ne modéliser que certaines parties d'un système, notamment les sections critiques difficiles à comprendre à partir du code seul.

2.2.2 Le choix de UML

Aujourd'hui, les diagrammes UML restent des outils standards et essentiels pour les développeurs, gestionnaires de projet, chefs d'entreprise, entrepreneurs technologiques, et professionnels de divers secteurs. Les raisons de l'adoption de ce langage de modélisation sont les suivantes :

- UML facilite la communication,
- Il simplifie la complexité et améliore la qualité du travail,

- UML n'est pas lié à un langage de programmation spécifique et peut donc être utilisé avec n'importe quel langage.

2.2.3 Identification des acteurs

Les acteurs impliqués dans le système de gestion des devoirs et des notes sont les suivants :

➤ **Professeur :**

- **Rôle principal :** Créer et gérer les cours, ainsi que superviser le processus d'évaluation des élèves.
- **Actions spécifiques :**
 - **Créer un cours :** Le professeur peut créer un cours en renseignant les informations nécessaires (titre, matière, niveau, description, etc.). Lors de la création, un **code unique** pour le cours est généré automatiquement pour permettre aux élèves de s'inscrire.
 - **Gérer les devoirs :** Le professeur peut ajouter des devoirs au cours, définir des dates limites, et attribuer des notes aux élèves une fois les devoirs soumis.
 - **Corriger les devoirs :** Le professeur peut consulter les devoirs soumis par les élèves et leur attribuer des notes.
 - **Gérer les élèves :** Le professeur peut voir la liste des élèves inscrits au cours et suivre leurs performances.

➤ **Élève :**

- **Rôle principal :** Participer aux cours, soumettre des devoirs et consulter ses résultats.
- **Actions spécifiques :**
 - **Rejoindre un cours :** L'élève utilise un **code de cours** unique généré par le professeur pour accéder au cours et en devenir membre.
 - **Soumettre des devoirs :** L'élève soumet ses devoirs via l'interface du cours en ligne. Il peut soit télécharger un fichier, soit entrer un texte en fonction du type de devoir.
 - **Consulter les devoirs :** L'élève peut voir les devoirs assignés, connaître les dates limites et avoir un aperçu des consignes.
 - **Consulter ses notes :** Une fois les devoirs corrigés, l'élève peut consulter ses notes et, si disponible, les retours du professeur sur sa performance.

2.2.4 Diagramme de cas d'utilisation

Dans le cadre du projet de gestion des devoirs et des notes, le diagramme de cas d'utilisation a pour but de représenter visuellement les interactions entre les différents acteurs du système et les fonctionnalités offertes par celui-ci. Ce diagramme permet de comprendre les actions que chaque utilisateur (acteur) peut effectuer dans le système.

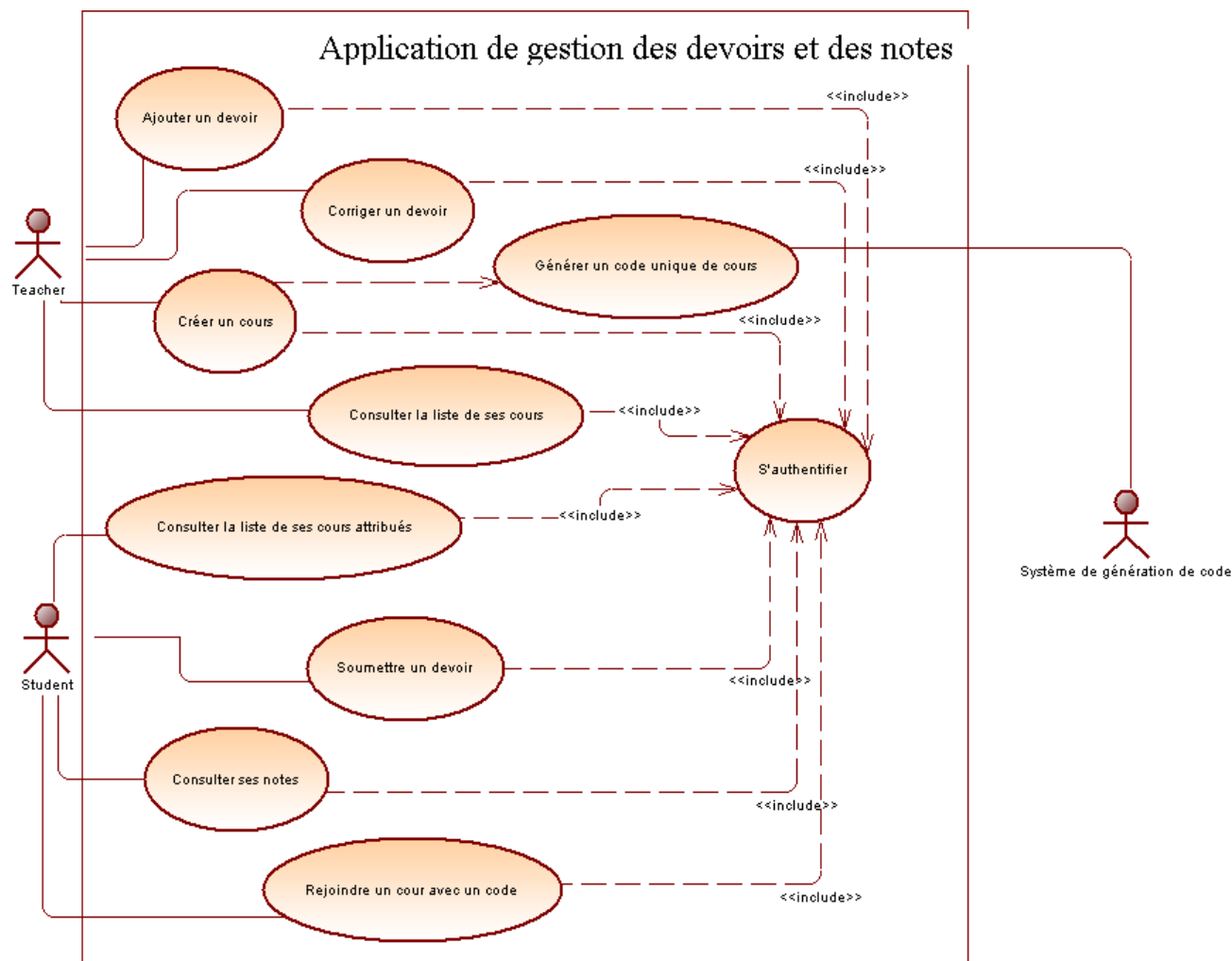


Figure 1 Cas d'utilisation : Gestion des devoirs et des notes

➤ Cas d'utilisation :

• Professeur :

- Créer un cours (avec génération automatique du code de cours).
- Gérer les devoirs (ajouter, modifier, supprimer).
- Corriger les devoirs soumis par les élèves.
- Consulter les performances des élèves (voir les soumissions et les notes).

• Élève :

- Rejoindre un cours en utilisant le code unique.

- Soumettre un devoir (en ligne ou fichier).
- Consulter les devoirs à venir (dates limites et consignes).
- Consulter ses notes et les commentaires des professeurs.

2.2.5 Diagramme des sequences

Le diagramme présenté ci-dessous illustre le déroulement du cas d'utilisation "S'authentifier". Selon ce scénario, Le développeur doit entrer le nom d'utilisateur et le mot de passe fourni par l'administrateur, le développeur doit changer obligatoirement le premier mot de passe, si ce dernier est changé alors il est redirigé vers la page d'accueil sinon un message d'erreur s'affiche.

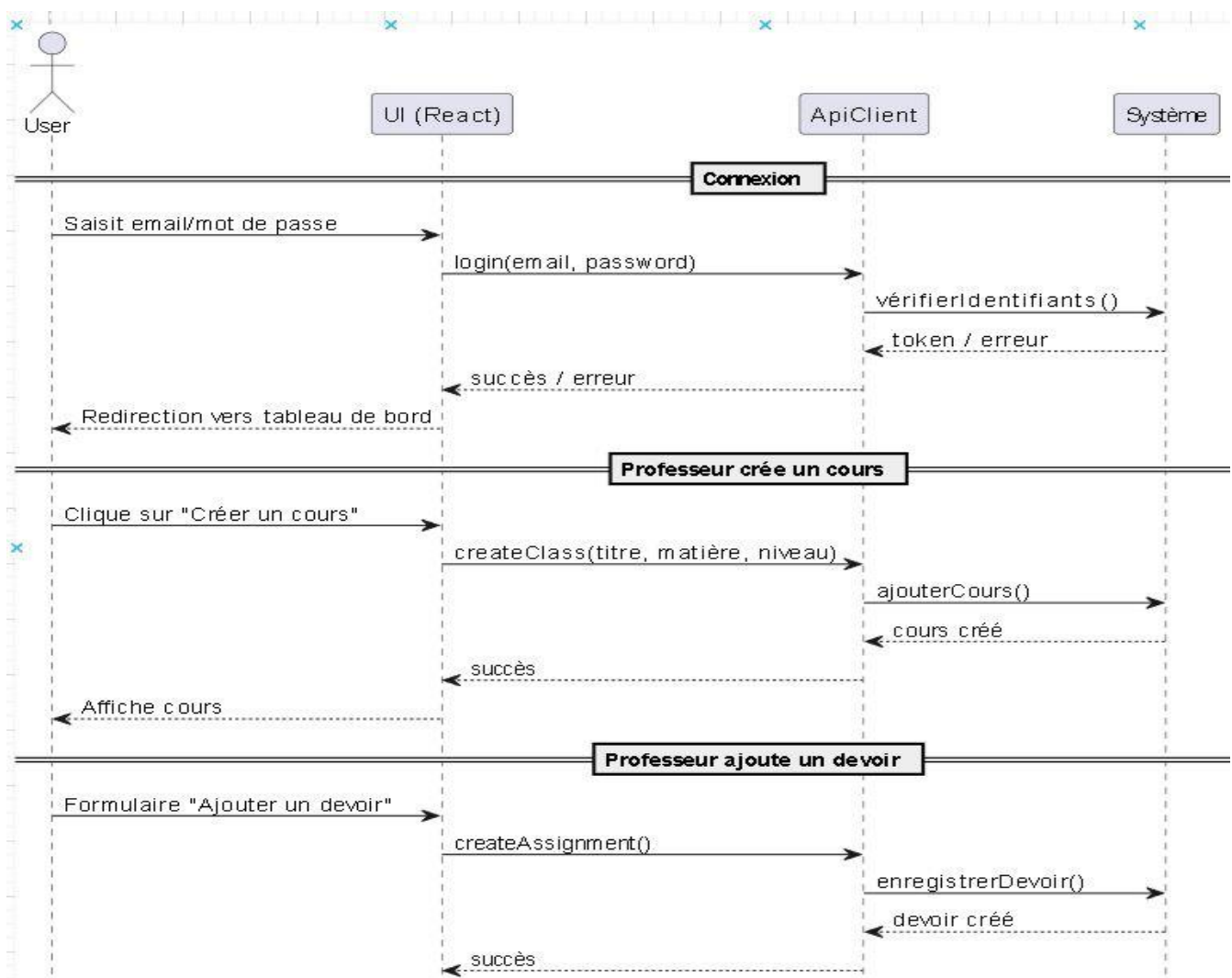


Figure 2 Diagramme de séquence (Partie Professeur)

Ce diagramme illustre les principales interactions entre l'utilisateur (professeur), l'interface utilisateur (UI en React), et le système backend pour trois scénarios clés de l'application :

- **Connexion** : l'utilisateur saisit ses identifiants. L'interface transmet les données au

système via une API qui vérifie les informations et retourne un token ou une erreur.

- **Création d'un cours** : après authentification, le professeur peut créer un cours en renseignant son titre, sa matière et son niveau. Le système génère automatiquement un code et enregistre le cours.
- **Ajout d'un devoir** : le professeur ajoute un devoir lié à un cours. L'interface transmet les informations au backend qui enregistre le devoir et confirme sa création.

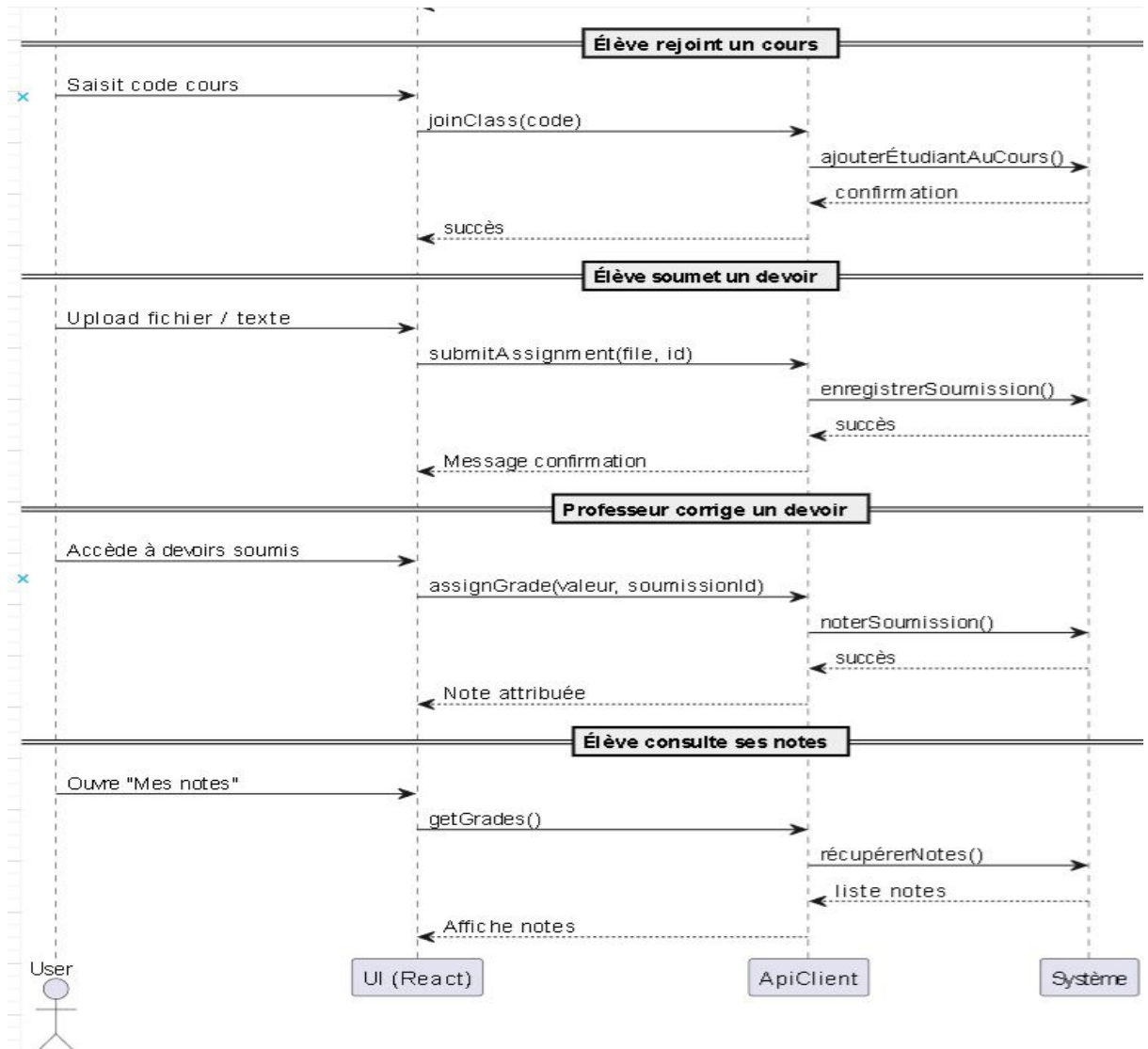


Figure 3 Diagramme de séquence (Partie Élève)

Ce diagramme détaille quatre scénarios clés liés à la gestion des devoirs et des notes dans l'application :

- **Élève rejoint un cours** : l'élève entre le code du cours pour rejoindre

automatiquement le groupe.

- **Élève soumet un devoir** : il envoie un fichier ou un texte via l'interface, qui est enregistré par le système.
- **Professeur corrige un devoir** : il attribue une note à une soumission en accédant à la liste des devoirs reçus.
- **Élève consulte ses notes** : il peut accéder à ses résultats via l'interface qui récupère les notes depuis le système.

2.2.6 Diagramme De Classe

Ce diagramme présente une vue statique du projet, et permet d'identifier les classes intervenantes dans le projet leurs attributs et les relations entre elles. Notre système se compose des classes présentées dans les diagrammes ci-dessous :

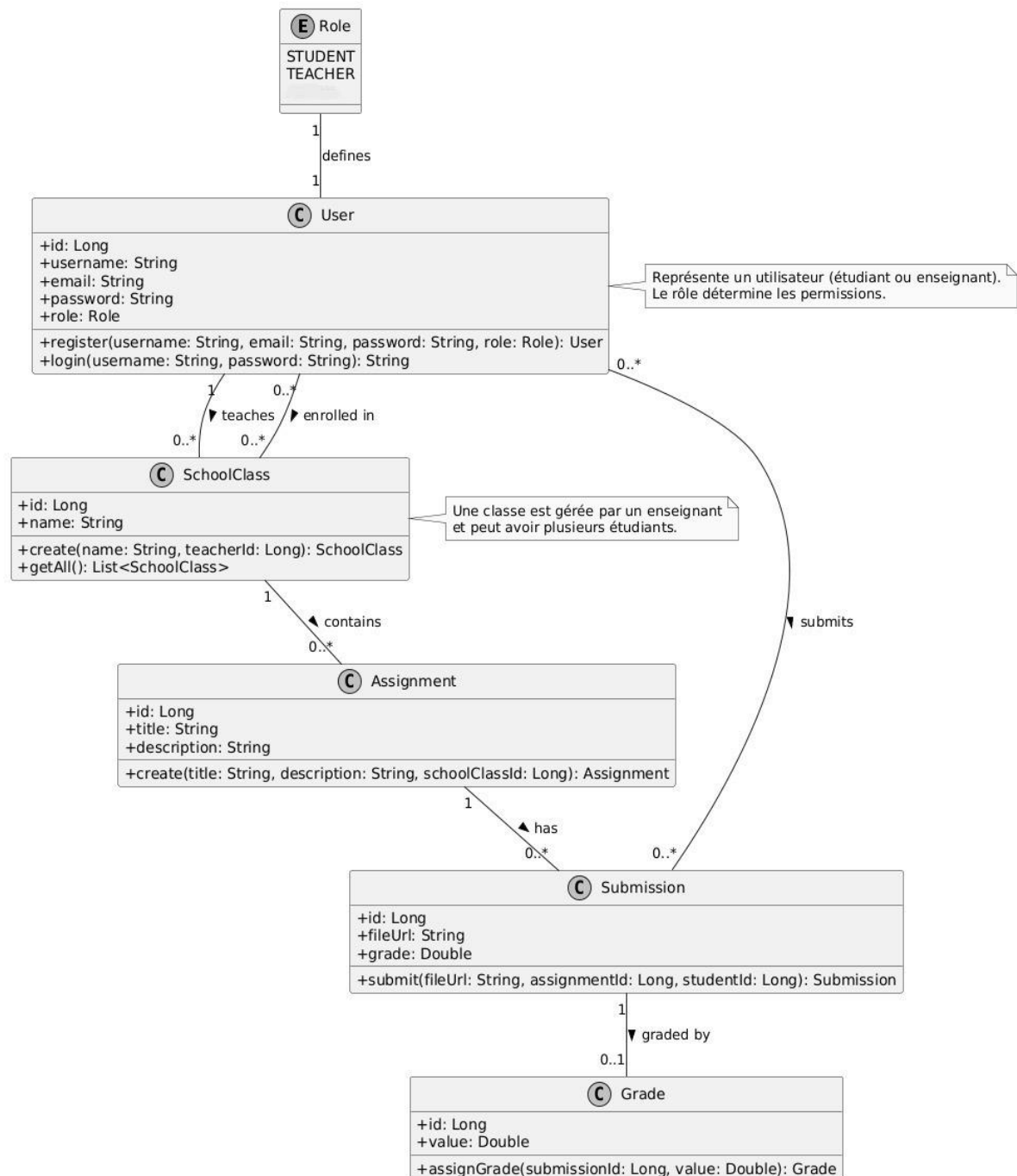


Figure 4 Diagramme de classe

Le diagramme de classes présenté modélise l'architecture principale d'un système de gestion pédagogique, permettant aux utilisateurs d'interagir avec des classes, des devoirs et des évaluations. Ce système est destiné à un environnement scolaire ou universitaire.

Classes principales

User : Représente les utilisateurs du système, qui peuvent être des étudiants, ou des enseignants, selon leur rôle. Chaque utilisateur possède un identifiant, un nom d'utilisateur, une adresse email, un mot de passe et un rôle. Des méthodes telles que register et login permettent la gestion de l'authentification.

SchoolClass : Représente une classe encadrée par un enseignant. Une classe peut contenir plusieurs étudiants et regrouper plusieurs devoirs. Elle est associée à un nom et à un identifiant, et peut être créée via la méthode create.

Assignment : Modélise un devoir ou un exercice à rendre, avec un titre, une description, et une association à une classe. Il peut être créé via une méthode dédiée.

Submission : Représente un rendu de devoir effectué par un étudiant. Chaque soumission est associée à un devoir, à un étudiant, et peut être notée. Elle contient un lien vers le fichier soumis et une note potentielle.

Grade : Permet d'attribuer une note à une soumission. La note est liée à une soumission spécifique via la méthode assignGrade.

Role (énumération) : Définit les différents types d'utilisateurs possibles dans le système : STUDENT, TEACHER

2.3 Conclusion

Dans ce chapitre, nous avons mis en lumière des étapes cruciales du cycle de vie d'une application, servant de liaison entre la phase de spécification et celle de réalisation. Cela s'est manifesté par la présentation de l'étude fonctionnelle, qui repose sur la collecte des besoins des utilisateurs et l'identification des acteurs ainsi que de leurs rôles. Et nous avons utilisé le formalisme UML pour représenter les différents aspects de notre système, tels que les acteurs, les cas d'utilisations et les séquences. Ce chapitre nous a ainsi permis de passer de la conception à la réalisation de notre projet. Dans le prochain chapitre, nous nous concentrerons sur la citation des différents outils et technologies utilisés pour la mise en œuvre de ce projet.

Chapitre 3 : Technologies et outils techniques

3.1 Introduction

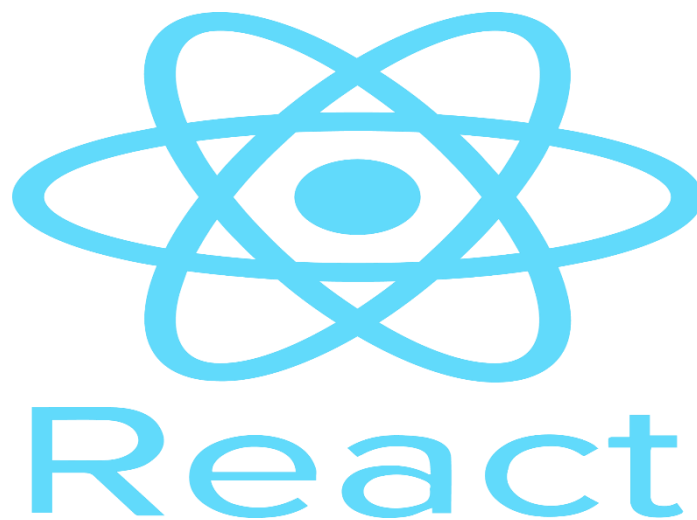
Ce chapitre vise à fournir une brève description de tous les outils et technologies sélectionnés pour la mise en œuvre de ce projet.

3.2 Choix technologiques

3.2.1 Frontend Web – React.js

React [1] *React* (aussi appelé **React.js** ou **ReactJS**) est une bibliothèque open source JavaScript pour créer des interfaces utilisateurs. Elle est maintenue par Meta (anciennement Facebook) ainsi que par une communauté de développeurs individuels et d'entreprises depuis 2013.

Le but principal de cette bibliothèque est de faciliter la création d'application web monopage, via la création de composants dépendant d'un état et générant une page (ou portion) HTML à chaque changement d'état.



3.2.2 Frontend Mobile – Android native

Android Native [2] Le **développement natif** fait référence au processus de **création d'applications spécifiques à une plateforme**, comme Android ou iOS, en utilisant les langages de programmation recommandés par ces plateformes. Par exemple, Java et Kotlin sont couramment utilisés pour le développement d'applications **Android**, tandis que Swift est le langage de programmation de choix pour les applications **iOS**. Le développement natif est souvent préféré pour sa **performance et son intégration optimales** avec les fonctionnalités du système d'exploitation.

Le développement natif offre **une expérience utilisateur supérieure**, car il permet de tirer pleinement parti des fonctionnalités de la plateforme. Cependant, le développement natif peut être **plus coûteux et plus long** que le développement hybride, car il nécessite **la création d'une application distincte pour chaque plateforme**. De plus, le développement natif nécessite une expertise dans plusieurs langages de programmation, ce qui peut augmenter les coûts de développement.

Le développement natif offre également une **plus grande flexibilité en termes de fonctionnalités de l'application**. Avec le développement natif, les développeurs peuvent accéder à toutes les fonctionnalités du système d'exploitation, ce qui n'est pas toujours possible avec le développement hybride.



3.2.3 Backend – Spring Boot

Spring Boot [3] Java Spring Framework (Spring Framework) est une infrastructure open source d'entreprise couramment utilisée qui permet de créer des applications autonomes de production qui fonctionnent sur la machine virtuelle Java (JVM).

Java Spring Boot (Spring Boot) est un outil qui accélère et simplifie le développement d'applications Web et de microservices avec Spring Framework grâce à trois fonctionnalités principales :

- Configuration automatique
- Approche directive de la configuration
- Possibilité de créer des applications autonomes

Ces fonctionnalités fonctionnent ensemble pour fournir un outil qui permet de configurer une application Spring avec une configuration et une installation minimales.



3.2.4 Base de données – MySQL

MySQL [4] est un système de gestion de bases de données relationnelles SQL open source développé et supporté par Oracle.

C'est la réponse courte, en une phrase, à la question « qu'est-ce que MySQL », mais décomposons cela en termes un peu plus humains.

Une base de données n'est qu'une collection structurée de données qui est organisée pour en faciliter l'utilisation et la récupération. Pour un site WordPress, ces « données » sont des choses comme le texte de vos articles de blog, des informations pour tous les utilisateurs enregistrés sur votre site, des données chargées automatiquement, des configurations de paramètres importants, etc.

MySQL n'est qu'un système populaire qui peut stocker et gérer ces données pour vous, et c'est une solution de base de données particulièrement populaire pour les sites WordPress. SonarQube.



3.2.5 API REST – Communication entre frontend et backend

API REST [5] Une API REST est une interface de programmation d'application (API) qui permet d'établir une communication entre plusieurs logiciels. Grâce à elle, des logiciels d'applications utilisant différents systèmes d'exploitation peuvent interagir et partager des informations par l'intermédiaire du protocole HTTP.

Dans sa forme complète, l'API REST est une interface de programmation d'applications de transfert d'état représentationnel, appelée plus communément service web API REST. Ainsi, lorsqu'un utilisateur effectue une requête via une API RESTful, celle-ci transfère une représentation de l'état des ressources requises au système client.

Cela signifie que les développeurs n'ont pas besoin d'installer des bibliothèques ou des logiciels supplémentaires pour tirer parti d'une conception d'API REST.



3.3 Choix Environnement de développement

3.3.1 VS Code

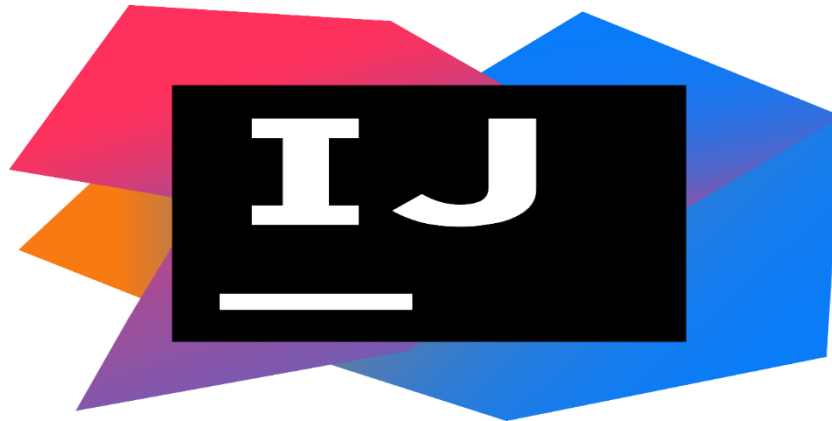
Visual Studio Code (VSCode) [5] est un **éditeur de code source** et un environnement de développement intégré (IDE) de Microsoft. Il est open-source et cross-platform, c'est-à-dire qu'il fonctionne sur Windows, Linux et Mac. Il a été conçu pour les développeurs web, mais il prend en charge de nombreux autres langages de programmation tels que C++, C#, Python, Java, etc. Il offre de nombreuses fonctionnalités comme la coloration syntaxique, l'auto-complétion, la mise en évidence des erreurs, la navigation de code, le débogage, la gestion de versions, l'intégration avec Git, et beaucoup d'autres. Il est également extensible à l'aide d'une grande variété d'extensions développées par la communauté, permettant aux développeurs de personnaliser l'éditeur selon leurs besoins.



Visual Studio Code

3.3.2 IntelliJ

l'environnement de développement intégré (IDE) officiel pour la création d'applications pour le système d'exploitation Android. Conçu et maintenu par Google, il est mis à disposition des développeurs afin de leur faciliter le processus de développement d'applications mobiles. Basé sur le logiciel IntelliJ IDEA de JetBrains, Android Studio offre une multitude d'outils et de fonctionnalités adaptées au développement Android.



3.3.1 Android Studio

Android Studio [7] est l'environnement de développement intégré (IDE) officiel pour la création d'applications pour le système d'exploitation Android. Conçu et maintenu par Google, il est mis à disposition des développeurs afin de leur faciliter le processus de développement d'applications mobiles. Basé sur le logiciel IntelliJ IDEA de JetBrains, Android Studio offre une multitude d'outils et de fonctionnalités adaptées au développement Android.



3.4 Conclusion

Dans ce chapitre, nous avons pris le temps de vous présenter les outils et technologies que nous avons soigneusement sélectionnés pour la réalisation de notre projet. Chaque élément a été choisi pour sa pertinence en vue d'atteindre notre objectif final.

Ces ressources varient, allant des langages de programmation aux outils informatiques spécialisés, chacun jouant un rôle précis dans la conception, l'amélioration et la mise en œuvre de notre solution.

Le chapitre suivant illustrera comment nous avons utilisé ces outils pour concrétiser notre projet.

Chapitre 4 : Réalisation

4.1 Introduction

Ce chapitre est consacré à la présentation de la phase de réalisation dans lequel je présenterai un ensemble de captures d'écrans des pages les plus pertinentes réalisées de l'application Monitoring

4.1 Page d'authentification

La figure 5 illustre l'**interface de connexion** de la version web de l'application, développée avec **React.js** et **Tailwind CSS**. L'interface est centrée sur la page avec un **design moderne et épuré**, sur fond **dégradé violet-indigo**.

L'utilisateur peut :

- Se connecter à l'aide de son **email et mot de passe**,
- Ou utiliser la **connexion via Google** grâce aux **Google Identity Services**.

Après authentification réussie, l'application effectue une **redirection automatique vers le tableau de bord correspondant au rôle de l'utilisateur** (prof ou élève)

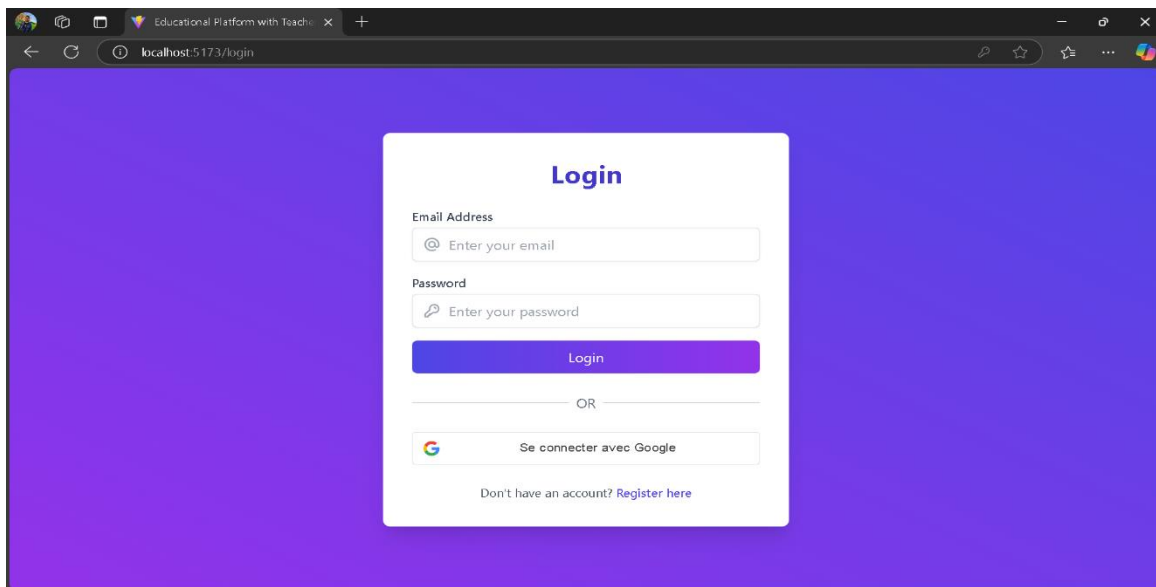


Figure 5 Page d'authentification

4.4 Espace Professeur

4.4.1 Page Principal

La figure 6 représente le tableau de bord enseignant de l'application, affichant un accueil personnalisé "Welcome Back, Mouad" et fournissant une synthèse complète des activités pédagogiques. L'interface présente d'abord les statistiques claires : 3 cours (dont 1 matière enseignée), 5 activités à suivre, 4 étudiants inscrits et 5 copies à corriger. Elle liste ensuite les cours détaillés (Introduction à Python, Sécurité Réseau, Structures de Données) avec leurs codes, disciplines, effectifs d'étudiants et descriptions. Des éléments interactifs comme la recherche de cours, le bouton "Create Course" et la moyenne globale (85%) complètent ce tableau de bord conçu pour une gestion pédagogique optimale, alliant clarté des informations (grâce à une organisation visuelle en colonnes et listes à puces) et fonctionnalités essentielles pour le suivi académique.

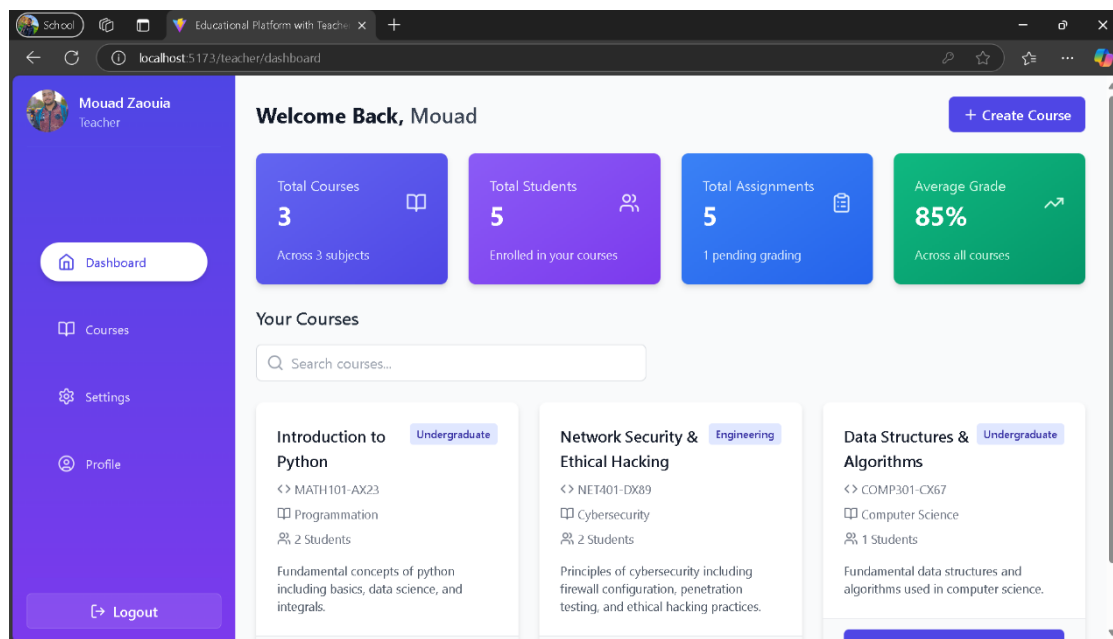
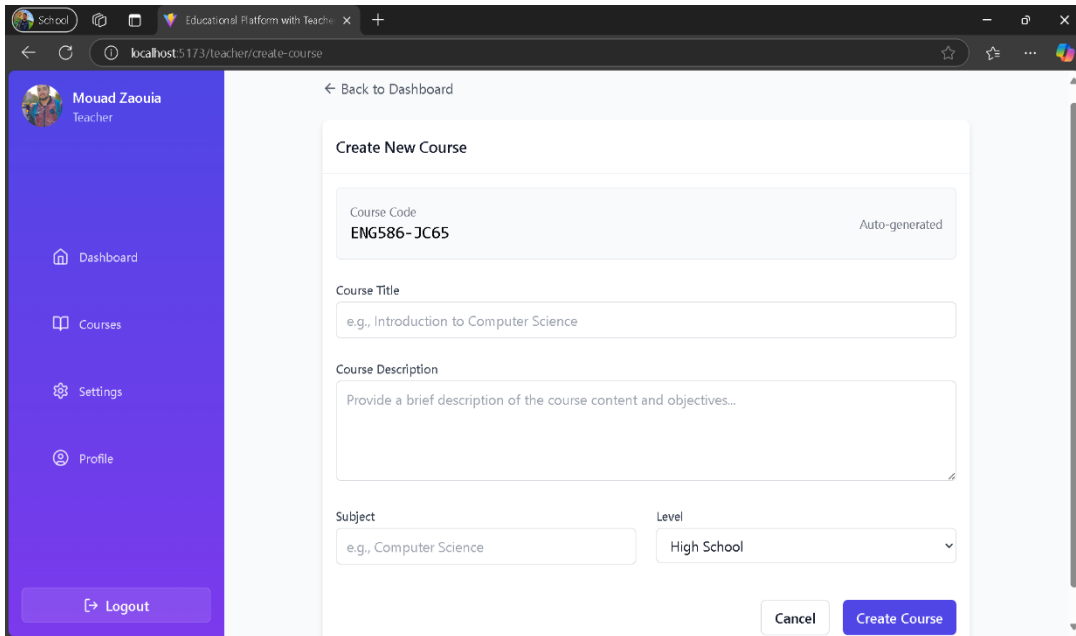


Figure 6 Professeur Dashboard

4.4.2 Page de Création d'un cour

La figure illustre l'interface dédiée à la création de nouveaux cours par les enseignants.



The screenshot shows a web browser window with the URL `localhost:5173/teacher/create-course`. The interface features a purple sidebar on the left with the user profile 'Mouad Zaouia, Teacher' and navigation links for 'Dashboard', 'Courses', 'Settings', and 'Profile'. A 'Logout' button is at the bottom of the sidebar. The main content area is titled 'Create New Course' and includes a 'Back to Dashboard' link. The form contains the following fields:

- Course Code:** A text box containing 'ENG586-JC65' with an 'Auto-generated' label.
- Course Title:** A text box with the placeholder 'e.g., Introduction to Computer Science'.
- Course Description:** A large text area with the placeholder 'Provide a brief description of the course content and objectives...'.
- Subject:** A text box with the placeholder 'e.g., Computer Science'.
- Level:** A dropdown menu currently showing 'High School'.

At the bottom right of the form are 'Cancel' and 'Create Course' buttons.

Figure 7 Créer un cour

4.4.3 Page de l'affichage d'un cour

Cette figure illustre l'interface personnalisée permettant à un enseignant de gérer l'ensemble de ses cours.

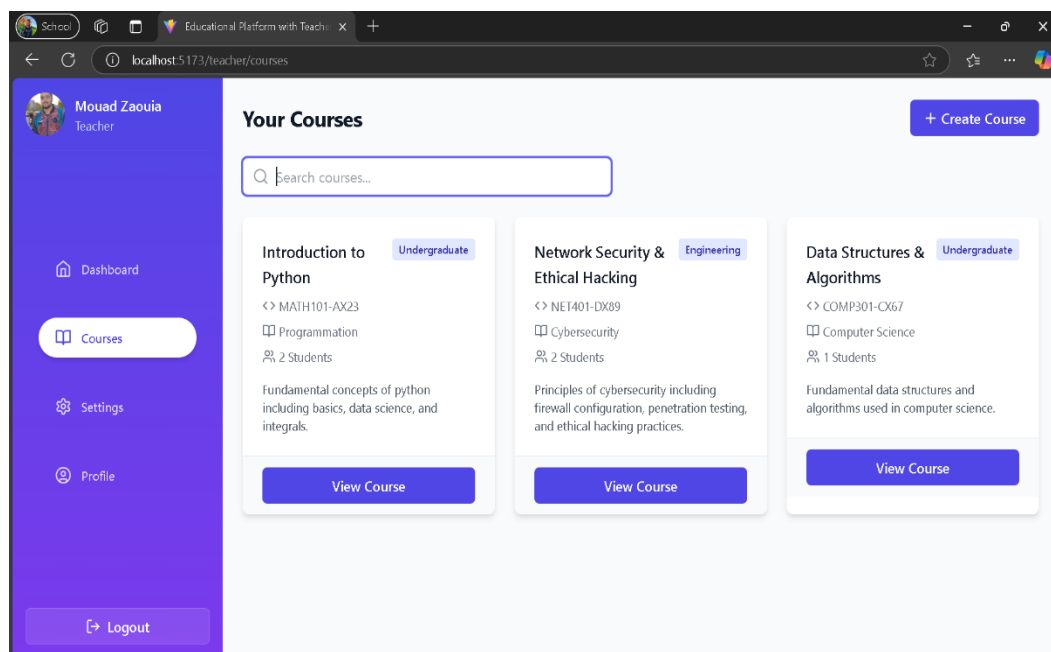


Figure 8 L'affichage des cours

4.4.4 Page de gestion des devoirs

Cette figure présente l'interface dédiée à la gestion des devoirs pour le cours *"Introduction to Python"*

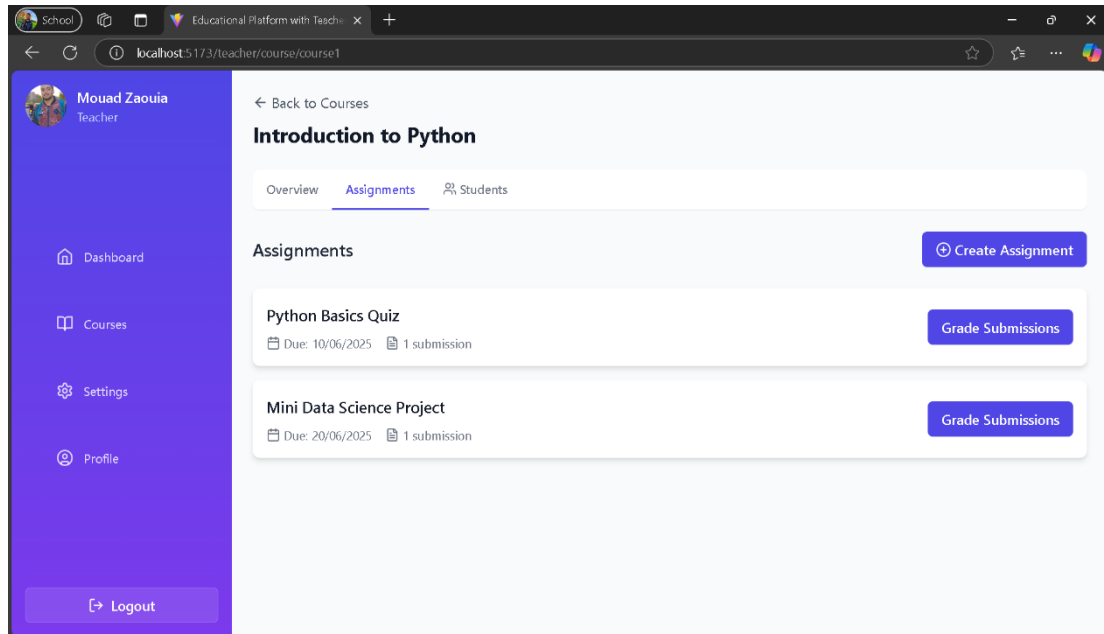


Figure 9 Gestion des cours

4.4.5 Page de gestion des paramètres.

Cette figure présente l'interface dédiée à la gestion des paramètres de l'application.

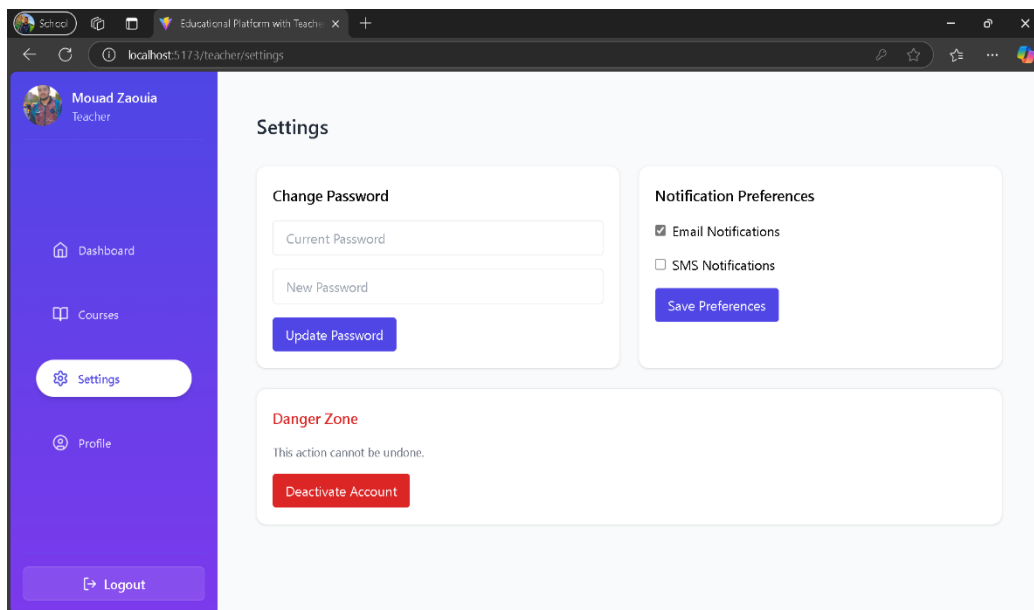


Figure 10 Gestion des parametres

4.4.6 Page de gestion de profile.

Cette figure présente l'interface dédiée à la gestion de profile.

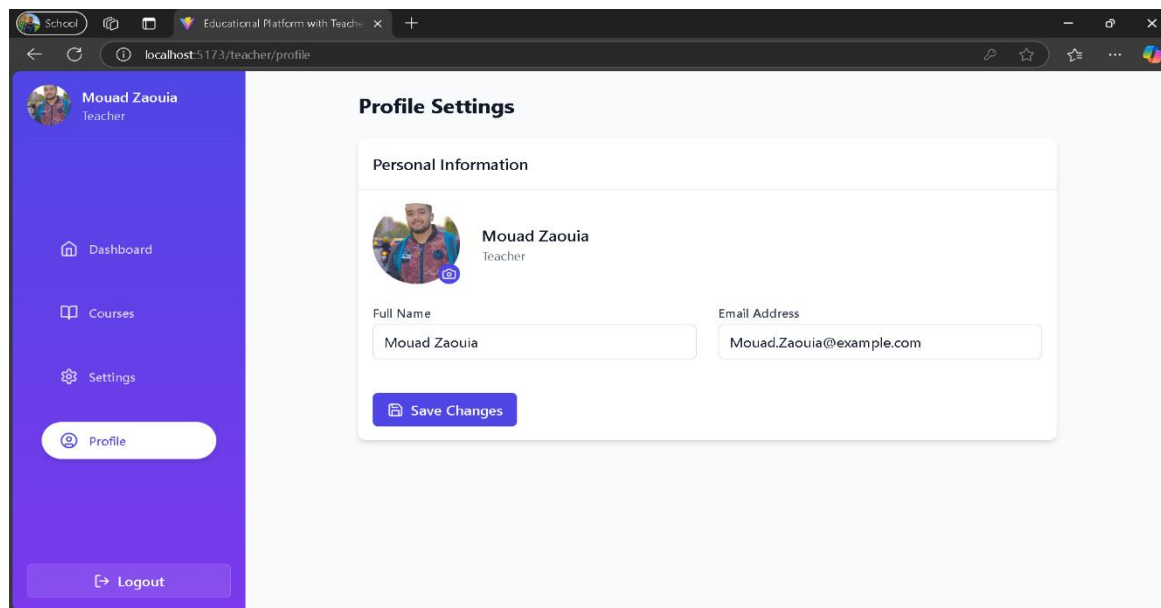


Figure 11 Gestion de profile (Professeur)

4.5 Espace Élève

4.5.1 Page de Page Principal

Cette interface présente le tableau de bord personnel d'un étudiant, affichant ses cours inscrits (*Introduction à Python* et *Cybersécurité*), ses performances académiques (moyenne de 53%, meilleure note de 90%) et les fonctionnalités de gestion (recherche de cours, option "Join a Course").

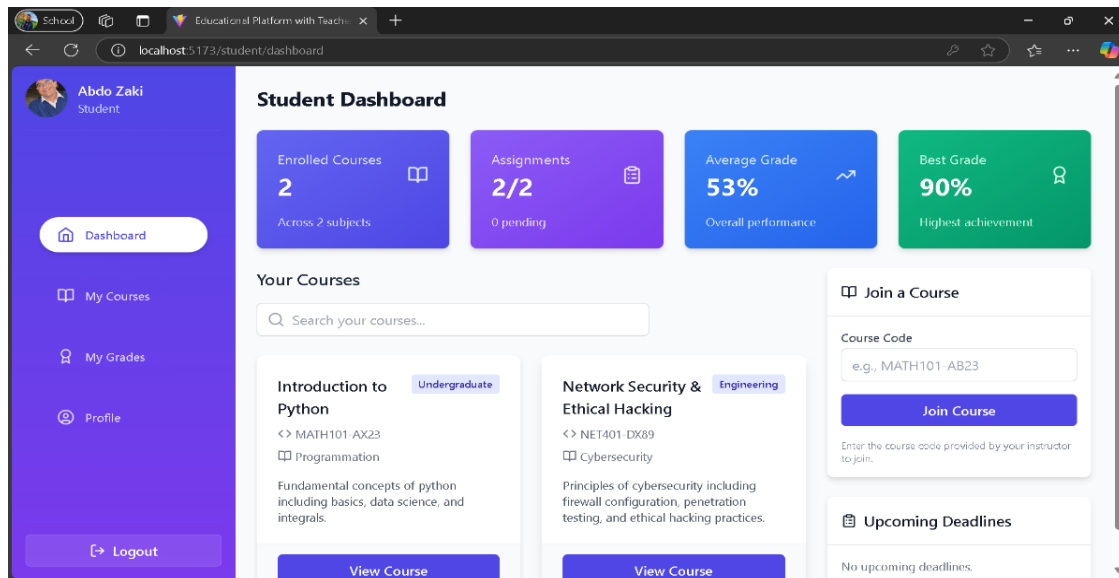


Figure 12 Élève Dashboard

4.5.2 Page de des Cours

On illustre les cours qui sont assignés à cet élève.

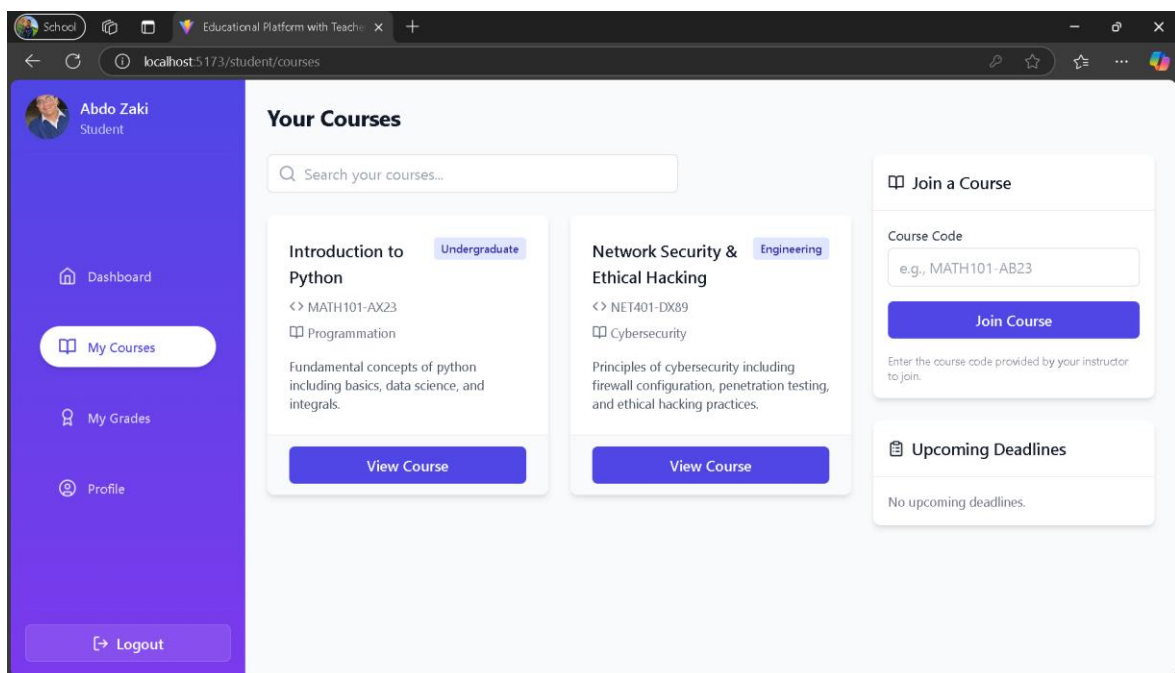


Figure 13 Les cours d'un Élève

4.5.3 Page de l'affichage des devoirs d'un cour

On présente ici la page qui permet de visualiser les devoirs d'un cours ainsi que leur état.

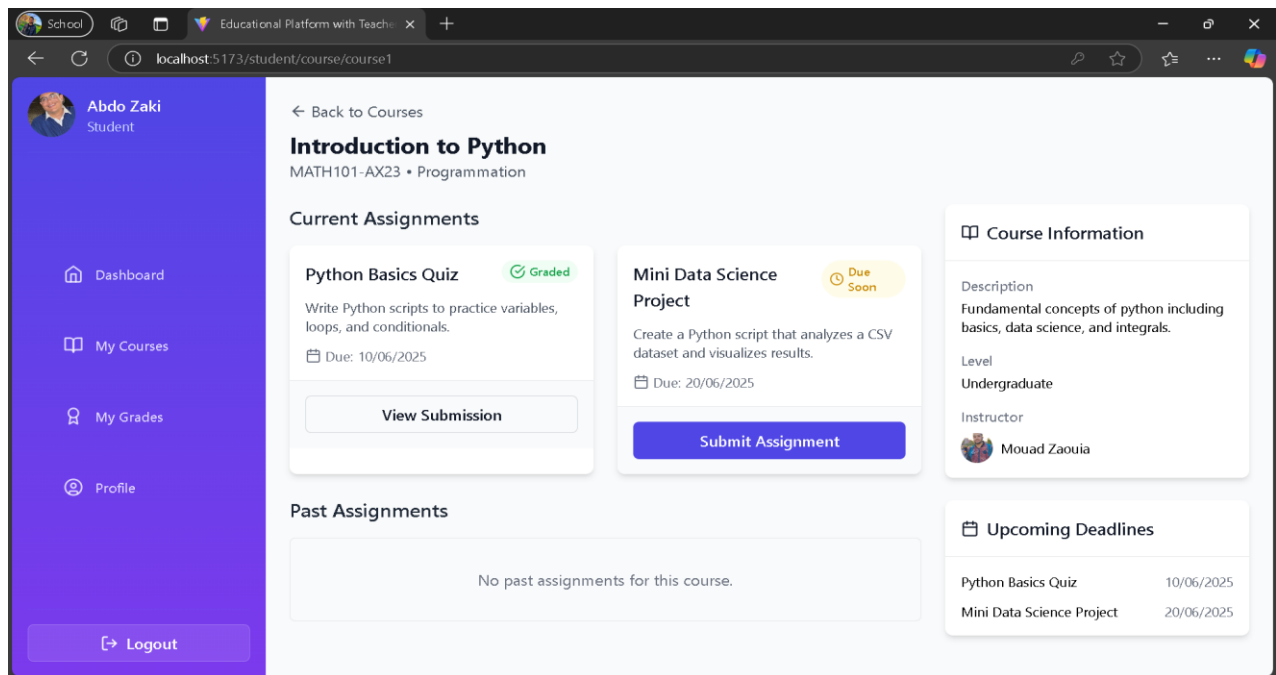


Figure 14 l'affichage des devoirs d'un cour

4.5.4 Page de l'affichage des notes

Cette figure affiche les notes obtenues par l'élève dans les différents devoirs, ainsi que la moyenne générale.

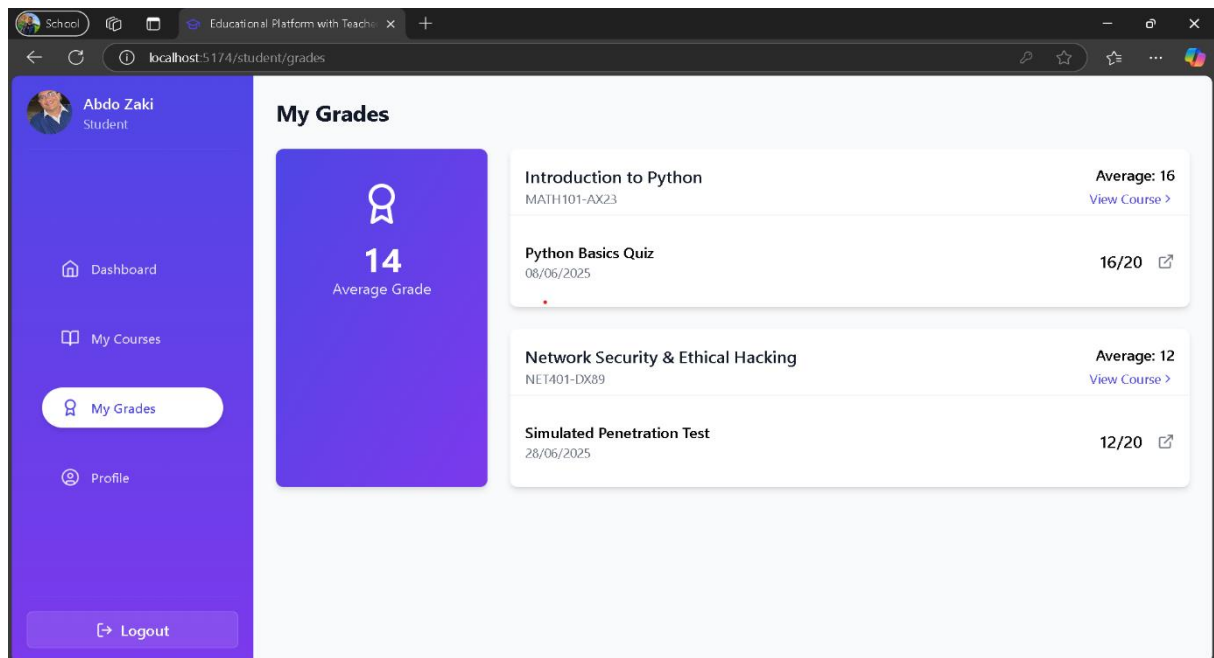


Figure 15 L'affichage des notes

4.5.5 Page de gestion de profile.

Cette figure présente l'interface dédiée à la gestion de profile.

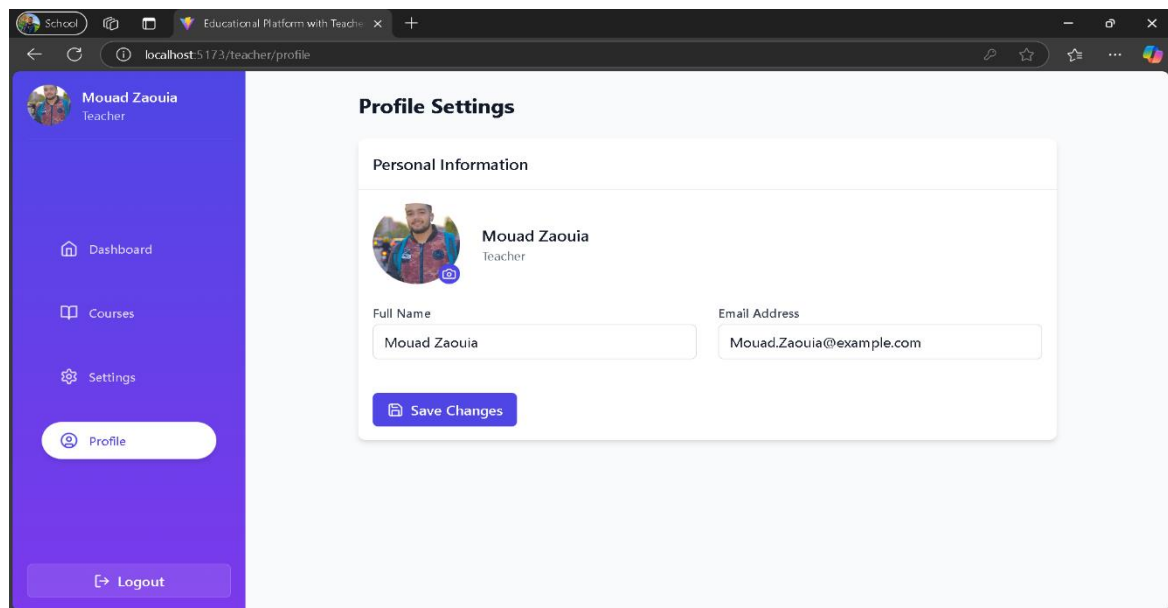


Figure 16 Gestion de profile (Élève)

4.6 Partie Mobile

La version mobile de l'application est exclusivement destinée aux **élèves**, et a été développée en **Android natif** à l'aide de **Java** et **Kotlin**. Elle permet aux étudiants de suivre leurs cours, consulter leurs devoirs et soumettre leurs travaux de manière simple et intuitive, directement depuis leur smartphone.

4.6.1 Page Login :

Ce figure illustre la page d'authentification pour accéder à l'application :

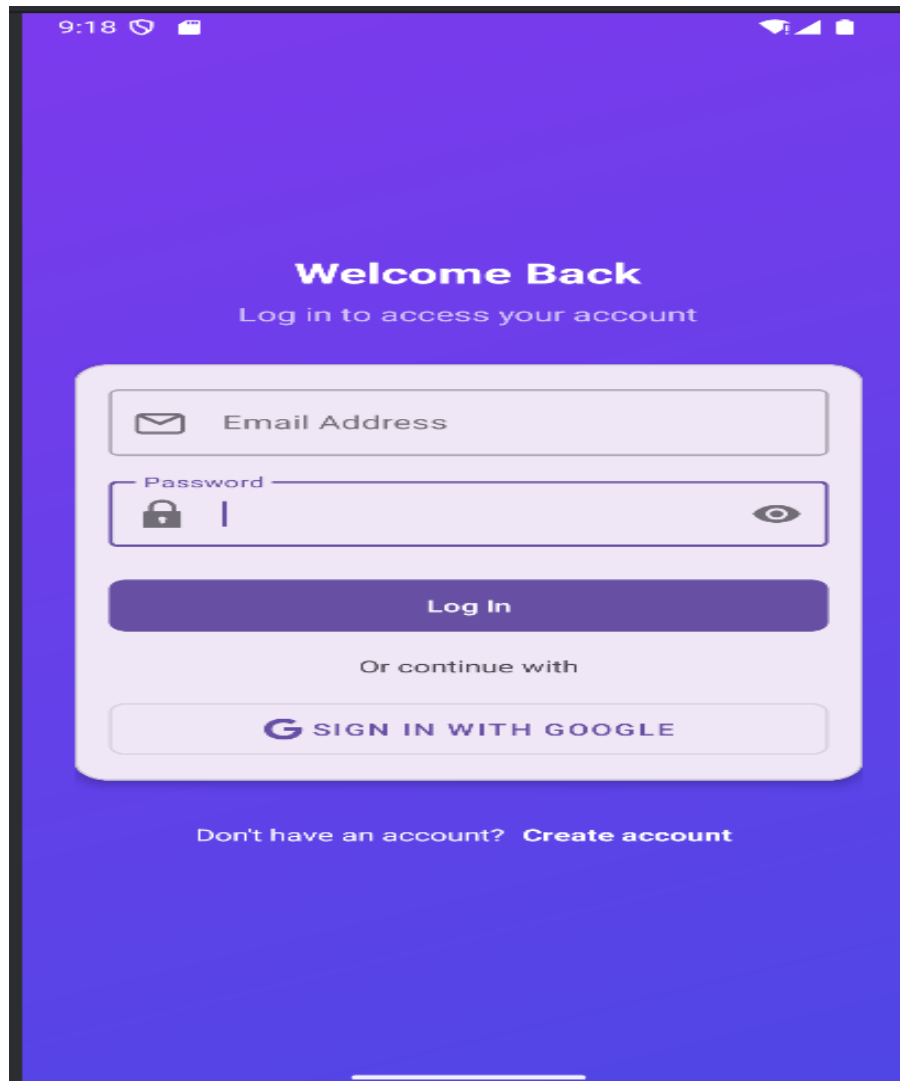


Figure 17 Login (Partie mobile)

4.6.2 Page d'accueil :

Cette interface présente le tableau de bord personnel d'un étudiant, affichant ses cours inscrits (*Introduction à Python et Cybersécurité*),

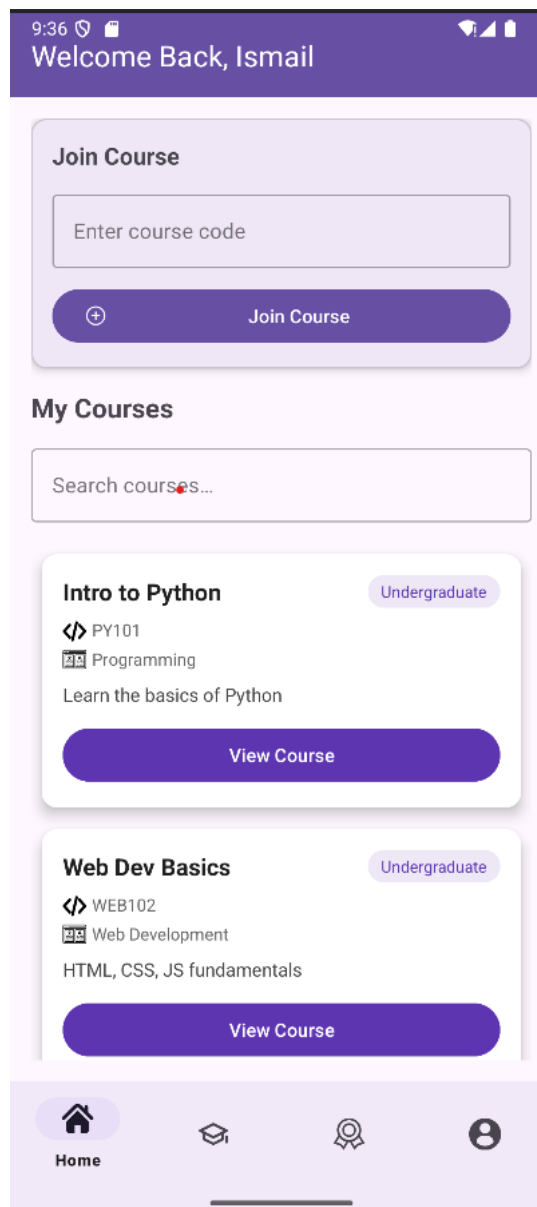


Figure 18 Éleve Dashboard

4.6.3 Page des cours :

Cette figure affiche les notes obtenues par l'élève dans les différents devoirs, ainsi que la moyenne générale.

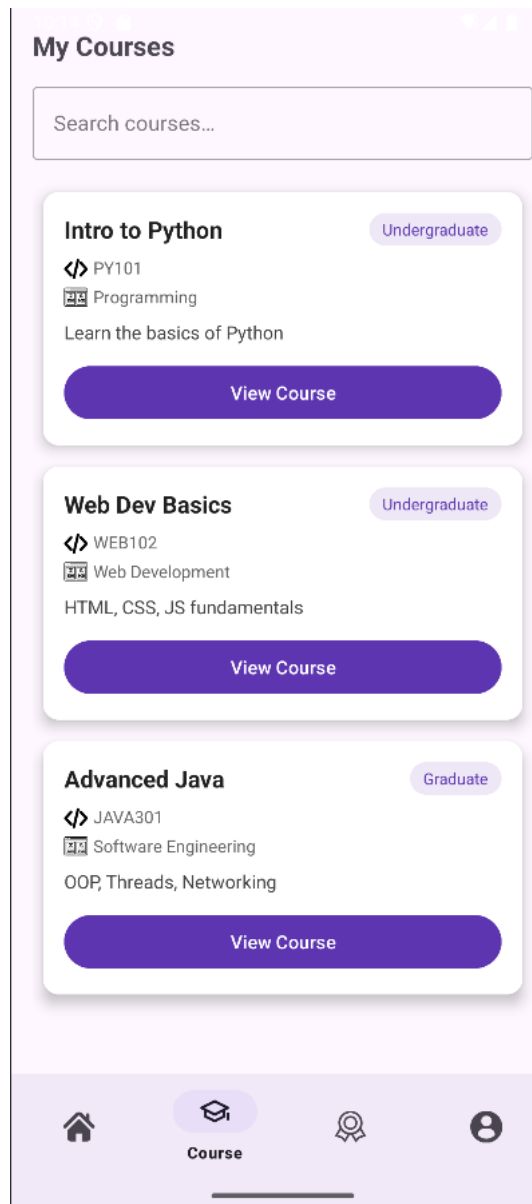


Figure 19 Les cours d'un Éleve

4.7 Conclusion

Ce dernier chapitre a fait l'objet de la phase de réalisation du projet dont lequel j'ai présenté les prises d'écran les plus pertinentes des fonctionnalités réalisées.

Conclusion et perspective

Pour conclure, ce projet de **gestion des devoirs et des notes** a permis de concevoir une application web et mobile qui facilite la relation pédagogique entre les **professeurs et les élèves**, notamment en ce qui concerne la **création de cours**, la **soumission de devoirs**, la **correction** et la **consultation des notes**.

L'application a été développée avec **React.js** pour le web, **Android natif** pour la version mobile, et un backend en **Spring Boot** avec une base de données **MySQL**. Elle offre une interface simple et intuitive, adaptée à chaque rôle.

Parmi les fonctionnalités réalisées, on trouve la génération automatique de codes de cours, l'upload et la correction des devoirs, ainsi que l'accès aux notes pour chaque élève.

Certaines fonctionnalités comme les **notifications automatiques** ou la **génération de statistiques pédagogiques** n'ont pas pu être finalisées, mais elles sont envisagées dans les perspectives d'amélioration.

Ce projet m'a permis de consolider mes compétences en développement fullstack, en gestion de projet et en conception logicielle tout en répondant à un besoin concret dans le domaine éducatif.

Webographie

- [1] «React.js,» 2013. [En ligne]. Available: <https://fr.wikipedia.org/wiki/React>
Date d'accès : 06 Mai 2025.

- [2] «Android Native,» 2008. [En ligne]. Available:
<https://esokia.com/fr/blog/d%C3%A9veloppement-natif-vs-hybride>
Date d'accès : 06 Mai 2025.

- [3] «Spring Boot,» 2014. [En ligne]. Available: <https://www.ibm.com/fr-fr/topics/java-spring-boot> Date d'accès : 06 Mai 2025.

- [4] «MySQL,» 1995. [En ligne]. Available: <https://kinsta.com/fr/base-de-connaissances/qu-est-ce-que-mysql/> Date d'accès : 06 Mai 2025.

- [5] «Vs Code,» 2015. [En ligne]. Available: <https://bility.fr/definition-visual-studio-code/> Date d'accès : 06 Mai 2025.

- [6] «IntelliJ,» 2001. [En ligne]. Available: https://fr.wikipedia.org/wiki/IntelliJ_IDEA
Date d'accès : 06 Mai 2025.

- [7] «Android Studio,» 2014. [En ligne]. Available: <https://www.v-labs.fr/glossaire/android-studio/> Date d'accès : 06 Mai 2025.