Cahier des Charges

Projet d'Intégration

TeachHub

Effectué par :

Omri Islem
Ilef Neji
Jlassi Ariem

Année universitaire : 2024-2025

ISET Rades

1 Description générale

L'application est conçue pour automatiser et simplifier la gestion des cours, devoirs, et évaluations dans un cadre éducatif. Elle s'appuie sur des règles précises pour garantir une cohérence dans la gestion des notes, la soumission des travaux, et l'organisation des cours et des groupes d'étudiants.

Administration des cours

Pour créer un cours, l'enseignant doit fournir des informations essentielles, notamment le nom du cours, son coefficient et le nombre de crédits associés. Chaque cours a un coefficient attribué, qui sera utilisé pour calculer la moyenne générale des étudiants.

Exceptions

- Aucun cours ne peut être créé sans nom, coefficient ou crédits. Une validation est mise en place pour éviter ces omissions.
- Si un cours a déjà des travaux ou devoirs associés avec des pondérations, les coefficients ne peuvent plus être modifiés.

Un cours peut être modifié après sa création, mais toute modification des coefficients ou des crédits doit être justifiée pour éviter de fausser les moyennes calculées.

Supervision des devoirs

Chaque devoir ou travail est associé à un cours précis et possède un barème d'évaluation. L'enseignant peut définir la pondération de chaque devoir par rapport à la note finale du cours.

Exceptions

- La pondération totale des devoirs d'un cours ne peut pas excéder 100%. Si l'enseignant tente de dépasser cette limite, une alerte est générée.
- Si un barème personnalisé est choisi, il doit respecter certaines limites (par exemple, un barème ne peut pas être inférieur à 1 ou supérieur à 100).
- Si un devoir n'a pas été rendu par un étudiant, il est noté par défaut comme "non rendu" (note égale à 0) à moins qu'un motif de retard ou une exception ne soit approuvée par l'enseignant.
- Si le total des coefficients ne fait pas 100%, une alerte est envoyée à l'enseignant pour ajuster les pondérations. Aucun calcul de moyenne ne sera effectué tant que la pondération n'est pas correcte.

L'enseignant peut choisir un barème standard (sur 20, sur 100) ou définir un barème personnalisé pour chaque devoir. Chaque devoir a une date limite, après laquelle les soumissions ne sont plus acceptées.

Encadrement des étudiants et des sous-groupes

Les enseignants peuvent inviter des étudiants à rejoindre un cours soit en leur envoyant un code d'inscription, soit directement par e-mail. Les étudiants doivent s'inscrire et confirmer leur présence avant d'accéder aux travaux et devoirs.

L'enseignant peut diviser les étudiants en sous-groupes pour faciliter la gestion des travaux en petits groupes. Chaque sous-groupe peut avoir des devoirs spécifiques qui diffèrent des autres groupes.

Exceptions

- Un étudiant ne peut pas appartenir à plus d'un sous-groupe pour un même devoir. S'il est déjà assigné à un sous-groupe, une alerte empêche sa réaffectation multiple.

Calcul des notes et des moyennes

Formule pour la note finale d'un cours

La note finale d'un cours est calculée en fonction des devoirs soumis et évalués, des pondérations définies, et du coefficient du cours.

$$\text{Note finale} = \sum \left(\frac{\text{Note du devoir}}{\text{Barème}} \times \text{Pondération} \right) \tag{1}$$

Exceptions

- Si un étudiant ne soumet pas de devoir, une note de 0 est attribuée automatiquement, sauf en cas de justification valable (maladie, problème technique).
- Si un barème est modifié après la publication d'un devoir, les devoirs déjà soumis et notés seront automatiquement réévalués en fonction du nouveau barème.
- Si le total des coefficients ne fait pas 100%, une alerte est envoyée à l'enseignant pour ajuster les pondérations.

Calcul de la moyenne d'un cours

La moyenne d'un cours est calculée en fonction des devoirs soumis, des pondérations définies, et du coefficient du cours.

$$\text{Moyenne du cours} = \frac{\sum (\text{Note du devoir} \times \text{Coefficient})}{\sum \text{Coefficients}} \tag{2}$$

Chaque cours possède un nombre de crédits, et la validation d'un cours est basée sur l'obtention d'une note supérieure à un certain seuil (par exemple, 10/20).

2 Besoins Fonctionnels

2.1 Priorité Élevée

Gestion des cours

- Permettre la création de cours en fournissant des informations telles que le nom, le coefficient et les crédits.
- Offrir la possibilité de réajuster les informations d'un cours existant, avec des justifications pour les changements des coefficients et des crédits.

Gestion des devoirs et travaux

- Autoriser l'ajout de travaux et devoirs, avec la définition de la note, la pondération, le barème de notation et la date limite.

Soumission et évaluation des devoirs

- Permettre aux utilisateurs de soumettre des devoirs en ligne avant la date limite.
- Offrir des outils pour télécharger et évaluer les devoirs soumis.

2.2 Priorité Normale

Interaction et communication

- Faciliter l'invitation d'étudiants à rejoindre un cours via un code ou par e-mail.
- Organiser les étudiants en sous-groupes pour la gestion des travaux.

Suivi des performances

- Fournir la possibilité de consulter les notes et le détail des évaluations après notation.
- Assurer le suivi des performances des étudiants, incluant l'accès aux évaluations et aux moyennes.

2.3 Priorité Faible

Gestion des exceptions

- Permettre la gestion des demandes de prolongation de délais pour les devoirs non rendus, en fonction de justifications valables.
- Offrir la possibilité de modifier directement la note des étudiants en cas de circonstances exceptionnelles (maladie, problème technique, etc.).

Notifications et alertes

- Envoyer des notifications concernant les devoirs, les dates limites, et les résultats aux utilisateurs concernés.
- Générer des alertes pour signaler des incohérences, comme des pondérations qui ne totalisent pas 100 % ou des modifications des coefficients non justifiés.

2.4 Objectifs globaux

- Assurer une plateforme centralisée pour la gestion des cours, des devoirs et des évaluations.
- Promouvoir une méthode d'évaluation claire et transparente.
- Améliorer l'efficacité de la gestion des devoirs et des travaux, tout en facilitant l'interaction entre les utilisateurs.

3 Les acteurs

3.1 Enseignant

Rôle :
L'enseignant est responsable de la création, de la gestion et de l'évaluation des cours et des devoirs. Il configure les barèmes de notation, gère les étudiants, invite des participants et évalue leurs performances. Son rôle consiste également à fournir des feedbacks aux étudiants et à suivre leurs progrès académiques. Dans l'école, il joue le rôle de gestionnaire pédagogique, assurant l'organisation des activités d'enseignement.

3.2 Étudiant

Rôle :
L'étudiant est l'utilisateur final de l'application. Il rejoint les cours, consulte les devoirs assignés, soumet ses travaux en ligne et consulte ses résultats et évaluations. Il est actif dans l'apprentissage et l'amélioration de ses performances. Au sein de l'entreprise éducative, il représente l'apprenant, qui profite des services pédagogiques mis en place.

4 Diagramme cas d'utilisation

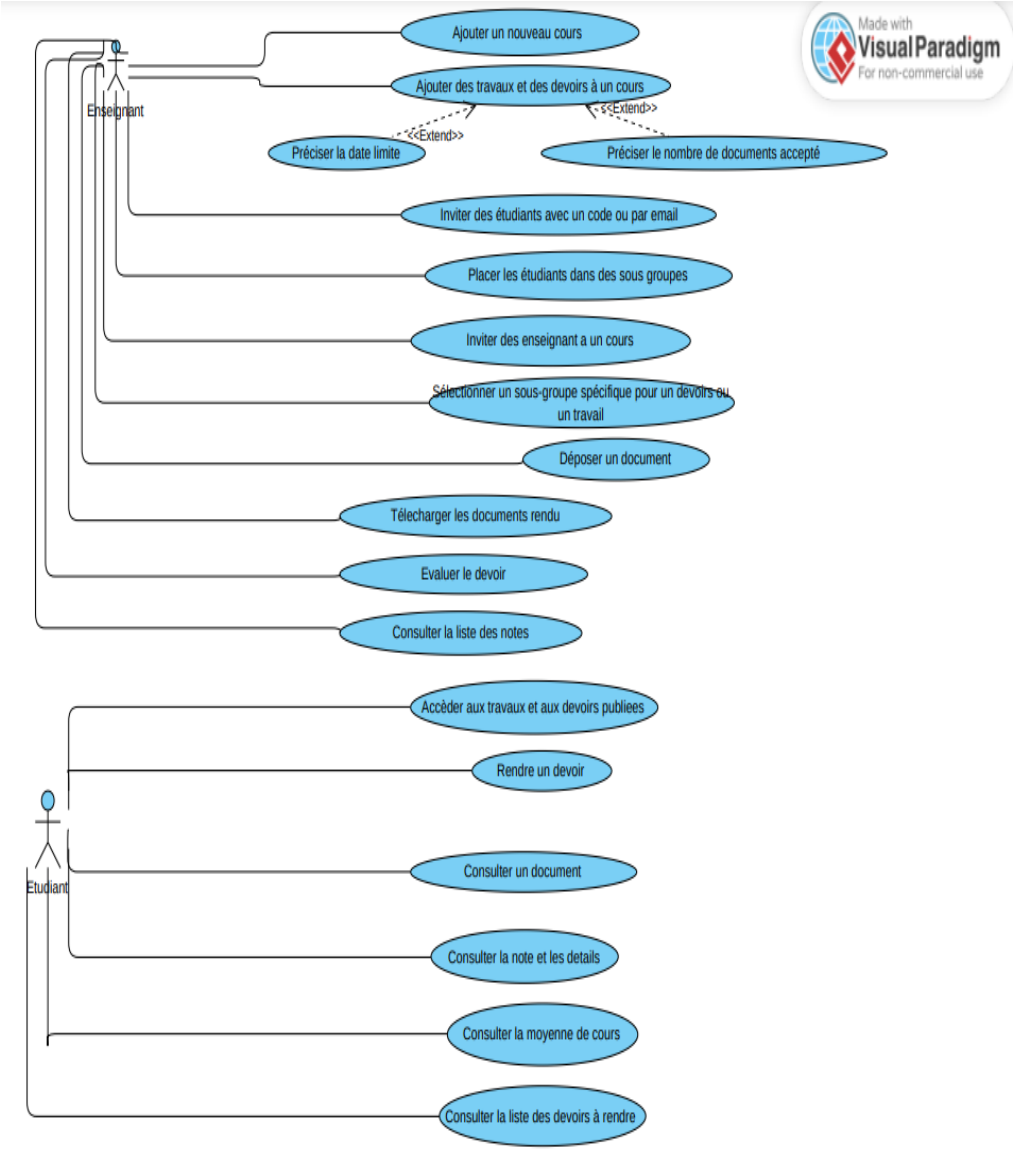

Figure 1: Diagramme cas d'utilisation

5 Product backlog

Sprint	User Story	Priorité	Scrum Master	Estimation du Temps
1	- En tant qu'enseignant, je dois être capable d'ajouter un nouveau cours avec son nom, son coefficient, ses crédits, et comment la moyenne sera calculée.	Haute	Jlassi Mariem	1 semaine
	- En tant qu'enseignant, je dois pouvoir ajouter des travaux et des devoirs en précisant le barème d'évaluation, dans quelle note du cours ils seront comptabilisés et avec quelle pondération.	Haute		
	- En tant qu'étudiant, je dois pouvoir accéder aux travaux et aux devoirs publiés.	Normale		
2	- En tant qu'enseignant, je dois pouvoir inviter des étudiants à mon cours avec un code cours ou par mail et les placer dans des sous-groupes.	Haute	Omri Islem	1 semaine
	- En tant qu'enseignant, je dois pouvoir inviter d'autres enseignants à mon cours par mail.	Normale		
	- En tant qu'enseignant, je dois pouvoir sélectionner un sous-groupe d'étudiants ou les étudiants concernés par un travail ou un devoir.	Normale		
	- En tant qu'enseignant, je dois pouvoir déposer un document.	Normale		
	- En tant qu'étudiant, je dois pouvoir rendre un devoir.	Haute		
	- En tant qu'étudiant, je dois pouvoir consulter un document.	Normale		
	- En tant qu'enseignant, je dois pouvoir télécharger un devoir rendu.	Normale		
3	- En tant qu'enseignant, je dois pouvoir préciser pour un devoir ou un questionnaire la date limite et le nombre de documents acceptés.	Normale	Neji Ilef	1 semaine
	- En tant qu'enseignant, je dois pouvoir évaluer un devoir en utilisant le barème.	Haute		
	- En tant qu'étudiant, je dois pouvoir consulter ma note et le détail de l'évaluation du devoir.	Haute		
4	- En tant qu'enseignant, je dois pouvoir consulter les évaluations réalisées par devoir, par groupe d'étudiants ou par étudiant sous forme de tableau.	Normale	Mariem Jlassi	1 semaine
	- En tant qu'étudiant, je dois pouvoir consulter l'ensemble des évaluations et la moyenne du cours.	Normale		

6 Les technologies utilisées

6.1 Angular

Angular est un framework JavaScript open source, apparu pour la première fois en 2009. Il a été créé par Misko Hevery et Adam Abrons et est totalement avalé et supporté par Google. Angular possède une communauté de particuliers et de sociétés. Il combine des modèles déclaratifs, une injection de dépendance, des outils de bout en bout et des meilleures pratiques intégrées pour résoudre les problèmes de développement. Ce framework permet aux développeurs de créer des applications qui résident sur le web, les mobiles ou le bureau.

Figure 2: Logo Angular

6.2 Spring Boot

Spring Boot est un framework Java conçu pour simplifier le développement d'applications en se basant sur Spring. Il permet de créer des applications autonomes prêtes pour la production avec une configuration minimale grâce à l'auto-configuration et aux dépendances prédéfinies. Il intègre un serveur d'applications embarqué, ce qui facilite le déploiement, et offre des outils comme Spring Boot Actuator pour la surveillance. Utilisé principalement pour créer des microservices, il réduit la complexité de la configuration Spring traditionnelle, accélérant ainsi le développement.

Figure 3: Logo SpringBoot

6.3 Postgres pgAdmin 4

Postgres pgAdmin 4 est une évolution majeure de l'outil d'administration avec interface graphique du serveur de base de données PostgreSQL. Il est destiné à succéder à

pgAdmin 3. Cette nouvelle version est dotée de deux modes de fonctionnement :

- Le mode station de travail, qui est un mode local à une machine de bureau.
- Le mode Web, qui est un mode serveur (web) destiné pour les machines serveurs.

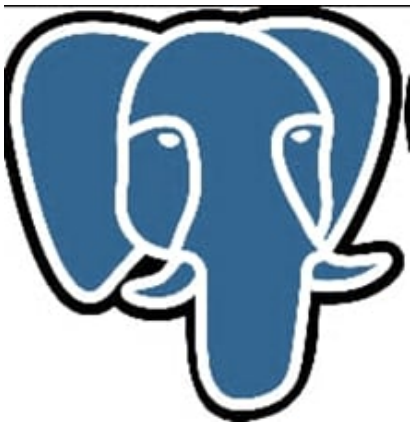

Figure 4: Logo PostgreSQl

6.4 Flutter

Flutter est un framework open-source développé par Google, utilisé pour créer des applications mobiles, web et desktop à partir d'une seule base de code. Il permet de développer des applications multiplateformes (iOS, Android, Windows, macOS, etc.) en utilisant un seul langage de programmation, Dart. Flutter se distingue par ses performances élevées, son rendu graphique rapide grâce à son propre moteur graphique, et la possibilité de concevoir des interfaces utilisateur attrayantes et personnalisées avec des widgets intégrés. Il est apprécié pour sa flexibilité, sa rapidité de développement et sa large communauté.

Figure 5: Logo Flutter

7 Diagramme de déploiement

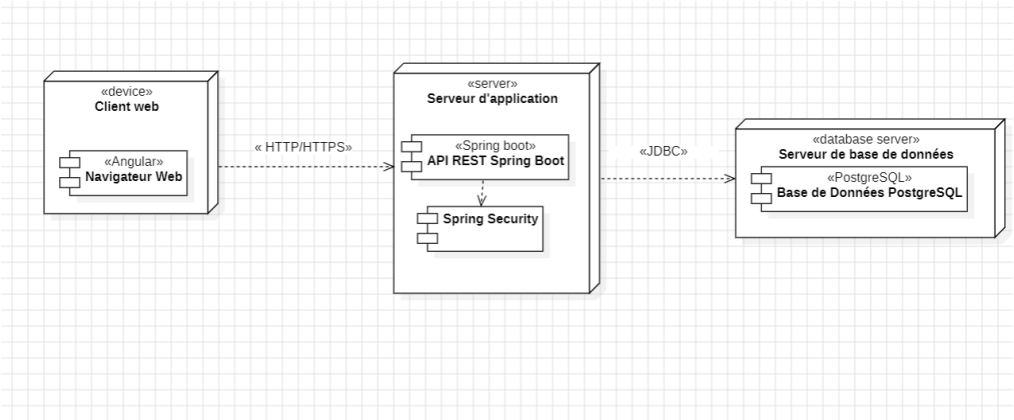

Figure 6: Diagramme de déploiement

8 Besoin non fonctionnelles

Capacité fonctionnelle

- Conformité stricte aux spécifications fonctionnelles et respect des règles de gestion métier.
- Validation des fonctionnalités par une modélisation claire (diagrammes UML, cas d'utilisation) et implémentation modulaire des services métier.

Tests :

- Tests fonctionnels pour garantir que chaque fonctionnalité correspond exactement aux besoins exprimés.

Facilité d'utilisation

- Conception d'une interface utilisateur intuitive et ergonomique, avec des flux simplifiés pour minimiser l'effort de l'utilisateur.
- Application des principes d'ergonomie (cohérence, retour visuel, simplicité) pour rendre l'application facilement compréhensible et exploitable.

Tests :

- Tests ergonomiques pour évaluer l'expérience utilisateur avec des spécialistes de l'ergonomie.

Maintenabilité

- Utilisation d'une architecture modulaire (par exemple, MVC) qui permet de localiser facilement les changements sans affecter l'ensemble du système.
- Adoption des bonnes pratiques de développement (comme les principes SOLID) pour garantir la lisibilité et la modularité du code.

Tests :

- Tests de régression pour s'assurer que les modifications n'introduisent pas de nouvelles erreurs dans le logiciel.