Développement d'une application d'aide à la délibération des étudiants de l'Université Nouveaux Horizons

Par :
Wilfried Musanzi Mavula

Directeur :
Dr. Saint Jean Djungu

Co-directeur :
Me. Jonathan Kabemba

Travail de fin de cycle présenté en vue de l'obtention du diplôme de licence en Informatique de Gestion

Année académique 2022 - 2023

RÉSUMÉ

ABSTRACT

RÉMERCIEMENTS

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

0 INTRODUCTION GÉNÉRALE

La délibération est un processus de prise de décision qui implique une discussion et une réflexion approfondies sur un sujet ou une question donnée. C'est une méthode qui permet de discuter, d'examiner et d'analyser les différentes options et perspectives avant de prendre une décision.

Dans le contexte d'une assemblée délibérative, telle qu'un conseil municipal, une assemblée générale ou un comité, la délibération se déroule généralement selon un processus formel qui permet à chaque membre de présenter son point de vue et d'argumenter en faveur de ses positions.

L'objectif de la délibération est d'aboutir à une décision éclairée et consensuelle, qui prend en compte les différents points de vue et les intérêts en présence. Ce processus permet également de renforcer la transparence et la légitimité de la décision prise, en impliquant tous les participants dans la prise de décision.

La délibération est souvent considérée comme une méthode de prise de décision plus démocratique et participative que d'autres approches, telles que la décision unilatérale ou le vote à la majorité simple. Elle est utilisée dans de nombreux domaines, tels que la politique, l'éducation, les affaires, et le droit, et bien d'autres pour n'en citer que ceux-ci.

0.1 CONTEXTE ET MOTIVATION

Ce projet est axé sur la déliberation à l'université Nouveaux Horizons, où cette dernière est, chaque année confrontée à la prise des décisions importantes concernant les étudiants tout au long de leur parcours universitaire. La délibération a un impact significatif sur la vie des étudiants, et il est donc important qu'elle soit faite de manière éclairée et réfléchie car elle détermine le sort des étudiants de manière individuelle, qui passe et qui ne passe pas et avec quelle mention.

La motivation derrière ce projet est de fournir à l'université Nouveaux Horizons un outil pratique, efficace, sur mesure, flexible pour les aider à prendre des décisions éclairées et réduire le circuit de circlution des données.

En fin de compte, ce projet a pour objectif de développer une application web qui permettra à l'université Nouveaux Horizons de gérer la délibération des étudiants ; De la recolte des notes à la publication des résultats.

0.2 MÉTHODES ET TECHNIQUES

Pour la réalisation de ce travail nous avons utilisé plusieurs méthodes et techniques qui nous ont permis de mener à bien notre projet en réunissant les informations nécessaires et en les analysant pour en tirer des conclusions.

0.2.1 *Méthodes*

1. La méthode analytico-déductive : Nous avons analyser la problématique évoquée dans une section ci-haut en partant des faits concrets pour aboutir à une conclusion générale.
2. La méthode descriptive : Nous avons décrit notre problématique de manière précise et objective.
3. La méthode comparative : Nous avons eu à comparer la manière dont la problématique est gérée ailleurs pour en dégager les similitudes et les différences.

0.2.2 *Techniques*

Nous avons utilisée est la technique documentaire et plus précisément la technique de la recherche bibliographique. Nous avons consulté des ouvrages, des articles, des documents et des sites web pour avoir des informations sur les applications existantes et les technologies utilisées pour leur développement. Nous avons aussi consulté des documents sur les méthodes de conception et de développement d'applications web.

Nous avons aussi utilisé la technique de l'entretien pour avoir des informations sur les besoins du corps acédemique de l'Université Nouveaux Horizons et sur les fonctionnalités qu'ils souhaiteraient avoir dans une application d'aide à la délibération également sur la manière dont les universités soeurs gérent cette problématique.

0.3 ETAT DE LA QUESTION

Etant dans une université, qui est loin d'être la seule au pays et dans le monde, nous avons pensé que d'autres universités ont forcément eu à faire face à la même problématique que nous. Nous nous sommes donc intéressés à la manière dont les autres universités gèrent la délibération des étudiants et les outils qu'elles utilisent. Nous avons trouvé des concepts qui nous ont aidé à mieux comprendre la problématique et à mieux cerner les besoins. Nous nous manquerons pas de les mentionner dans la suite de ce travail.

0.4 OBJECTIFS

En vue d'apporter une solution efficace, flexible et solide à la problématique posée, nous avons défini l'objectif de fournir une application web qui :

— permettra de racourcir le circuit de circulation des informations,
— permettra de faciliter la publication des résultats,
— permettra de faciliter le suivi des données des étudiants,
— peut s'intégrer facilement dans le système d'information de l'université, et aux autres applications existantes telles que moodle, Goole Classroom, etc.

0.5 PROBLÉMATIQUE

Actuellement, à l'Université Nouveaux Horizons la déliberation des étudiants se passe commme suit :

— A la fin des examens le professeur chargé du cours envoie soit les notes annuelles soit les notes de tous les travaux ansi que l'examen au décanat de l'Université en copie au doyen de la faculté.
— Le décanat après réception transmet les notes réçues soit au format pdf soit Excel(xlsl) soit manuscrit au président du jury.
— Le jury à son calcul les moyennes et après déliberation communique les résultats aux étudiants.

Vous conviendrez avec moi que le circuit de circlulation des informations et assez long, de plus, il est difficile de gérer les données des étudiants de manière efficace et la publication des résultats n'est pas aussi évidente qu'elle le devrait. Les dites données peuvent être :

— les cours en compléments,

— le(s) relevé(s) de chaque année parce qu'un étudiant peut en avoir plusieurs en une année,
— etc.

1

GÉNÉRALITÉS

1.1 LA DÉLIBÉRATION

La délibération est un processus de réflexion et de prise de décision collectif qui implique l'examen et l'évaluation de différentes options ou perspectives, souvent en vue d'arriver à un consensus ou à une décision commune. La délibération peut avoir lieu dans de nombreux contextes, tels que l'ensignement, les gouvernements, les organisations, les groupes de travail ou même les conversations informelles entre amis et collègues.

Dans une délibération, les participants discutent ouvertement et honnêtement des différents points de vue, arguments et preuves en faveur ou en défaveur de chaque option. L'objectif est souvent d'arriver à une décision qui soit juste, équitable et acceptable pour toutes les parties impliquées.

La délibération peut prendre différentes formes, allant des discussions informelles aux débats structurés et aux processus de consultation plus formels. Les participants peuvent également utiliser des outils tels que des sondages, des votes ou des simulations pour faciliter la prise de décision.

1.2 UNE APPLICATION

En informatique, une application (ou "app" en abrégé) est un logiciel conçu pour effectuer une tâche ou une série de tâches spécifiques sur un ordinateur, un smartphone, une tablette ou un autre appareil électronique. Les applications sont généralement créées pour répondre à un besoin spécifique de l'utilisateur ou pour fournir une fonctionnalité particulière, telle que la gestion de tâches, la navigation GPS, la communication en ligne, le traitement de texte, le montage de photos, etc.

Les applications peuvent être accessibles via un navigateur tel que Goodle Chrome, Firefox, Microsoft Edge, etc ou téléchargées et installées à partir de diverses sources, telles que les boutiques d'applications en ligne, les sites web des développeurs ou les disques d'installation. Certaines applications sont gratuites, tandis que d'autres sont payantes, et les prix peuvent varier considérablement en fonction de la complexité et de la popularité de l'application.

Les applications peuvent être développées pour différents systèmes d'exploitation, tels que iOS, Android, Windows, MacOS, Linux, etc. Les développeurs peuvent utiliser différents langages de programmation et environnements de développement pour créer des applications, tels que JavaScript, Typescript, Swift, Kotlin, Python, C++, etc.

1.3 UN ÉTUDIANT

Un étudiant est une personne qui s'inscrit à un programme d'enseignement dans une école, un collège, une université ou un autre établissement d'enseignement pour poursuivre des études dans un domaine d'intérêt particulier. Les étudiants peuvent poursuivre des

études à différents niveaux d'enseignement, allant de l'éducation primaire à l'enseignement supérieur et à la formation professionnelle.

Les étudiants sont généralement motivés par un désir d'apprendre et d'acquérir des compétences et des connaissances dans leur domaine d'études choisi. Ils peuvent suivre des cours dans une variété de disciplines, telles que les sciences, les mathématiques, les arts, les sciences sociales, les sciences humaines, les affaires, etc.

En tant qu'étudiants, ils sont souvent tenus de participer à des cours, des séminaires, des projets de groupe et des travaux pratiques pour compléter leur programme d'études. Les étudiants peuvent également être impliqués dans des activités parascolaires, telles que des clubs, des équipes sportives, des organisations étudiantes, etc.

Les étudiants peuvent poursuivre différents objectifs en poursuivant des études, tels que l'acquisition de compétences professionnelles, l'obtention de diplômes, la poursuite d'une carrière, la poursuite de la recherche, l'exploration des intérêts personnels, etc.

1.4 WEB

Le web, ou World Wide Web, est un système d'information mondial qui permet d'accéder à des ressources et des services sur Internet à travers des hyperliens et des URL (Uniform Resource Locators). Il a été inventé par Tim Berners-Lee en 1989 et est devenu public en 1993.

Le web est basé sur un ensemble de technologies, notamment le langage de balisage HTML pour la création de pages web, le protocole HTTP (Hypertext Transfer Protocol) pour la communication entre les clients et les serveurs web, et les navigateurs web pour l'affichage des pages web sur les ordinateurs et les appareils mobiles.

Le web permet aux utilisateurs d'accéder à une grande variété de contenus et de services en ligne, tels que des pages web, des documents, des images, des vidéos, des jeux, des applications, des réseaux sociaux, des services de messagerie, des services bancaires en ligne et bien plus encore.

Le web est devenu un élément essentiel de la vie quotidienne de nombreux utilisateurs d'Internet, et est utilisé dans de nombreux domaines, notamment les affaires, l'éducation, les médias, les services gouvernementaux, les soins de santé, le divertissement et la communication.

1.5 UNE API

Une API, ou "Application Programming Interface", est une interface de programmation qui permet à des applications ou des services informatiques de communiquer entre eux.

Plus précisément, une API expose un ensemble de fonctionnalités et de données spécifiques, généralement sous la forme de points d'entrée accessibles à distance par le biais d'Internet. Ces points d'entrée sont souvent basés sur des protocoles standardisés tels que HTTP (utilisé par les API REST) ou SOAP.

Les développeurs peuvent utiliser ces points d'entrée pour échanger des données et des instructions avec l'API et accéder aux fonctionnalités qu'elle expose, sans avoir à connaître les détails internes de la mise en œuvre de l'API.

Les API sont largement utilisées dans les applications web et mobiles, les services cloud, les réseaux sociaux, les systèmes d'intégration d'entreprise, et dans de nombreux autres contextes. Elles permettent aux développeurs de créer des applications plus rapidement en

réutilisant des fonctionnalités existantes et en évitant de devoir écrire du code complexe pour chaque nouvelle fonctionnalité.

1.6 L'UNIVERSITÉ NOUVEAUX HORIZONS

Le désir et la volonté d'offrir à la jeunesse une éducation de qualité sont les grandes motivations à l'origine de la création de l'Université Nouveaux Horizons. A cette immense volonté viennent en surcouche deux faits du vécu social qui sont : L'incitation et suggestion des tiers : Pendant plusieurs années des nombreuses personnes ont incités les promoteurs de l'Université Nouveaux Horizons à créer une institution académique, dans le prolongement de l'enseignement de fin d'études à l' GE D'OR, sans aucun doute on fait mention des parents des élèves de ladite école, des amis, et autres. Le regard de l'élève tourné vers l'étranger pour la suite des études après le cycle de l'enseignement secondaire : Le fait est que des nombreux élèves de la République Démocratique du Congo en fin de leurs études secondaires n'aspirent qu'à s'expatrier, pour aller s'inscrire, sous d'autres cieux car pensant que l'ailleurs et mieux que le chez soi ; dans des Universités supposées de meilleure qualité. Par ceci on aperçoit aisément la critique souvent formulée à l'endroit des titres académiques décernés par l'Université en République Démocratique du Congo. Par ailleurs, un autre fait ayant incité les initiateurs à créer l'Université Nouveaux Horizons est en quelque sorte ce qu'on pourrait appeler "l'héritage colonial" légué aux Universités Congolaises dans leur ensemble celui qui consiste à former des demandeurs d'emploi et non des créateurs d'emplois.

L'Université Nouveaux Horizons a pour but de dispenser des enseignements dans différentes filières, avec accent sur l'esprit d'entreprise (entrepreneuriat) ; de former des hommes et des femmes capables de participer efficacement au développement de la communauté, davantage comme des créateurs que des demandeurs d'emplois. L'UNH assure un enseignement de qualité et de niveau international grâce à des compétences requises, judicieusement recrutées, pour cette fin.

1.7 LES MÉTHODES DE DÉVELOPPEMENT LOGICIEL

Une méthode de développement logiciel est une approche structurée pour planifier, concevoir, construire, tester et maintenir un logiciel de manière efficace et efficiente. Les méthodes de développement logiciel fournissent un cadre pour l'ensemble du processus de développement, de la planification initiale à la livraison du produit final.

Les méthodes de développement logiciel peuvent inclure des techniques, des outils, des pratiques et des processus pour faciliter la collaboration entre les membres de l'équipe de développement, pour gérer les risques de projet, pour assurer la qualité du logiciel, pour planifier et suivre les progrès du projet, etc.

Les méthodes de développement logiciel peuvent varier en fonction des besoins et des objectifs du projet. Certaines méthodes sont axées sur la rapidité et la flexibilité, tandis que d'autres mettent l'accent sur la qualité, la rigueur et la planification minutieuse. Certaines méthodes sont plus adaptées aux projets de grande envergure, tandis que d'autres sont plus adaptées aux projets plus petits et plus agiles.

Il existe plusieurs méthodes de développement logiciel, chacune avec ses propres avantages et inconvénients. Voici un aperçu de quelques-unes des méthodes les plus courantes seront citées ci-dessous.

1.7.1 *La méthode en cascade*

La méthode en cascade est une méthode de développement logiciel qui suit une approche linéaire et séquentielle pour le développement de logiciels. Elle est également connue sous le nom de modèle de cycle de vie en cascade ou de modèle de développement en cascade. La méthode en cascade est composée de plusieurs étapes linéaires, chacune étant exécutée dans l'ordre séquentiel suivant :

1. Analyse des besoins : Dans cette étape, les exigences et les spécifications du logiciel sont identifiées et définies en détail. Cela implique une compréhension approfondie des besoins des utilisateurs, des fonctionnalités nécessaires du logiciel et des contraintes techniques.
2. Conception : Dans cette étape, une architecture de haut niveau du logiciel est conçue en utilisant les exigences et les spécifications établies dans l'étape précédente. Cette étape implique également la conception des interfaces utilisateur, des bases de données et des algorithmes nécessaires pour implémenter les fonctionnalités du logiciel.
3. Implémentation : Dans cette étape, le code est écrit et les fonctionnalités requises sont implémentées selon les spécifications établies dans les étapes précédentes.
4. Test : Dans cette étape, le logiciel est testé pour s'assurer qu'il fonctionne correctement et qu'il répond aux exigences et aux spécifications établies dans les étapes précédentes. Les tests peuvent inclure des tests unitaires, des tests de système, des tests de performance, des tests d'acceptation, etc.
5. Maintenance : Dans cette étape, le logiciel est maintenu et soutenu après sa livraison. Cela peut inclure des mises à jour, des corrections de bogues, des améliorations de performance, etc.

La méthode en cascade est simple et facile à comprendre, mais elle peut être rigide et ne permet pas de réagir facilement aux changements. Cela est dû au fait que chaque étape doit être complétée avant de passer à la suivante, et tout changement apporté à une étape ultérieure peut avoir des répercussions sur les étapes précédentes. Par conséquent, la méthode en cascade est souvent utilisée pour les projets bien définis avec des exigences claires et stables.

1.7.2 *Les méthodes agiles*

Les méthodes agiles sont des méthodes de développement logiciel qui mettent l'accent sur l'adaptabilité, la collaboration, la flexibilité et la réactivité aux changements. Les méthodes agiles cherchent à livrer des logiciels fonctionnels rapidement et régulièrement tout en s'adaptant aux changements des exigences du client et de l'environnement de développement. Voici quelques détails sur les méthodes agiles les plus courantes :

1. Scrum est une méthode agile populaire qui se concentre sur la collaboration étroite entre les membres de l'équipe de développement et le client. Les projets sont organisés en sprints, qui sont des périodes courtes de développement allant de deux à quatre semaines. Pendant chaque sprint, l'équipe de développement travaille pour livrer une fonctionnalité fonctionnelle. À la fin de chaque sprint, l'équipe se réunit pour une réunion de rétrospective pour discuter de ce qui s'est bien passé et de ce qui peut être amélioré.

2. XP (Extreme Programming) : XP est une méthode agile qui se concentre sur l'écoute du client et la rapidité de livraison. XP se concentre sur l'automatisation des tests, la programmation en binôme, la planification continue et l'intégration continue. Les pratiques d'XP aident les équipes à travailler ensemble efficacement et à livrer des logiciels fonctionnels rapidement.
3. Kanban : Kanban est une méthode agile qui se concentre sur la gestion visuelle de projet. Les projets sont organisés en tâches individuelles qui sont représentées sur un tableau Kanban. Les tâches passent par plusieurs étapes, de l'analyse à la livraison. Les équipes de développement travaillent à équilibrer le nombre de tâches en cours de façon à éviter les goulots d'étranglement.
4. Lean : Le développement Lean se concentre sur l'élimination des déchets dans le processus de développement, permettant d'optimiser le flux de travail. La méthode Lean met l'accent sur la collaboration étroite entre les membres de l'équipe de développement et le client, ainsi que sur la livraison rapide et continue de logiciels fonctionnels.

Les méthodes agiles sont généralement utilisées pour les projets de développement logiciel complexes et/ou dynamiques où les exigences du client peuvent évoluer au fil du temps. Les méthodes agiles sont souvent préférées pour leur capacité à s'adapter aux changements, à leur flexibilité et à leur efficacité.

1.7.3 *La méthode en V*

La méthode en V est une méthode de développement logiciel qui est similaire à la méthode en cascade, mais qui met davantage l'accent sur les tests. La méthode en V est également connue sous le nom de modèle de cycle de vie en V ou de modèle de développement en V.

La méthode en V suit un processus linéaire, mais elle met en correspondance chaque étape de développement avec une étape de test correspondante, formant ainsi une forme de V. Les étapes supérieures de la V représentent les étapes de conception et de spécification, tandis que les étapes inférieures représentent les étapes de test et de validation.

1. Analyse des besoins : Dans cette étape, les exigences et les spécifications du logiciel sont identifiées et définies en détail. Cela implique une compréhension approfondie des besoins des utilisateurs, des fonctionnalités nécessaires du logiciel et des contraintes techniques.
2. Spécification : Dans cette étape, les spécifications du logiciel sont détaillées et documentées en utilisant des techniques telles que les diagrammes UML, les descriptions textuelles, etc.
3. Conception : Dans cette étape, une architecture de haut niveau du logiciel est conçue en utilisant les spécifications et les exigences établies dans les étapes précédentes. Cette étape implique également la conception des interfaces utilisateur, des bases de données et des algorithmes nécessaires pour implémenter les fonctionnalités du logiciel.
4. Programmation : Dans cette étape, le code est écrit et les fonctionnalités requises sont implémentées selon les spécifications établies dans les étapes précédentes.
5. Tests unitaires : Dans cette étape, chaque unité de code est testée de manière isolée pour s'assurer qu'elle fonctionne correctement.

6. Tests d'intégration : Dans cette étape, les unités de code sont intégrées pour former des modules fonctionnels qui sont testés pour s'assurer qu'ils fonctionnent ensemble correctement.
7. Tests de système : Dans cette étape, le logiciel est testé dans son ensemble pour s'assurer qu'il fonctionne correctement et qu'il répond aux exigences et aux spécifications établies dans les étapes précédentes.
8. Tests d'acceptation : Dans cette étape, le logiciel est testé par les utilisateurs finaux pour s'assurer qu'il répond à leurs besoins et à leurs attentes.
9. Maintenance : Dans cette étape, le logiciel est maintenu et soutenu après sa livraison. Cela peut inclure des mises à jour, des corrections de bogues, des améliorations de performance, etc.

La méthode en V est rigoureuse et garantit un niveau élevé de qualité, mais elle peut être coûteuse et nécessite une planification minutieuse. La méthode en V est souvent utilisée pour les projets de développement logiciel critiques, tels que les logiciels de mission, les systèmes de contrôle de la sécurité, etc., où la qualité est primordiale.

1.7.4 *La méthode RAD*

La méthode RAD (Rapid Application Development) est une méthode de développement logiciel qui se concentre sur la rapidité de livraison des logiciels en utilisant des cycles de développement courts et itératifs combinés à des techniques de prototypage. Le RAD est souvent utilisé pour les projets de développement logiciel qui ont des exigences changeantes et des délais de livraison serrés.

Le RAD suit un processus de développement rapide qui se compose de quatre phases principales :

1. Planification : Dans cette phase, les exigences du client sont identifiées et une planification détaillée du projet est réalisée. Cette phase implique également la définition des objectifs du projet, la définition du budget et du calendrier, la définition des rôles et responsabilités de l'équipe de développement, etc.
2. Analyse : Dans cette phase, les exigences sont analysées en détail et les spécifications du logiciel sont définies. Cette phase implique également la conception des prototypes pour valider les exigences du client et pour recueillir leurs commentaires.
3. Conception : Dans cette phase, la conception détaillée du système est réalisée en utilisant les spécifications et les commentaires du client. Les prototypes sont également améliorés et affinés en fonction des commentaires du client.
4. Construction : Dans cette phase, le logiciel est développé de manière itérative et incrémentale en utilisant des cycles de développement courts. Les tests sont effectués tout au long de cette phase pour s'assurer que le logiciel est de haute qualité et fonctionne correctement.

Le RAD met l'accent sur la collaboration étroite entre les membres de l'équipe de développement et le client, ainsi que sur la livraison rapide et continue de logiciels fonctionnels. Les avantages du RAD incluent une meilleure réponse aux besoins changeants du client, une livraison plus rapide des logiciels et une meilleure qualité de l'application grâce à la collaboration étroite entre les membres de l'équipe de développement et le client.

Cependant, le RAD peut être inadapté pour les projets qui ont des exigences stables et bien définies ou pour les projets qui nécessitent une planification détaillée et rigoureuse. Le RAD est souvent utilisé pour les projets de développement logiciel qui ont des exigences changeantes et des délais de livraison serrés, tels que les projets de développement de logiciels Web.

1.7.5 *La méthode DevOps*

La méthode DevOps (Development and Operations) est une approche de développement logiciel qui vise à améliorer la collaboration et la communication entre les équipes de développement et d'exploitation, en vue d'accélérer le processus de développement et de déploiement de logiciels.

La méthode DevOps est basée sur trois principes clés :

1. Collaboration : La méthode DevOps encourage la collaboration étroite entre les équipes de développement et d'exploitation, afin de favoriser la communication, la compréhension mutuelle et la prise de décisions en commun.
2. Automatisation : La méthode DevOps utilise l'automatisation pour accélérer le processus de développement et de déploiement de logiciels. L'automatisation peut inclure des tests automatisés, des déploiements automatisés et des intégrations continues.
3. Amélioration continue : La méthode DevOps encourage une culture d'amélioration continue, où les équipes cherchent constamment à améliorer les processus de développement et de déploiement de logiciels en utilisant des feedbacks des utilisateurs finaux et des données de performance.

La méthode DevOps implique également l'utilisation d'outils et de technologies spécifiques, tels que les outils de gestion de source, les plates-formes de conteneurs, les outils de test automatisés et les outils de déploiement automatisé.

Les avantages de la méthode DevOps incluent une plus grande agilité et flexibilité, une amélioration de la qualité du logiciel, une réduction des coûts et une amélioration de la satisfaction des utilisateurs finaux grâce à une livraison plus rapide de logiciels fonctionnels.

Cependant, la mise en œuvre de la méthode DevOps peut être complexe et nécessiter une transformation culturelle significative au sein de l'organisation. La méthode DevOps nécessite également des compétences techniques et une compréhension approfondie des outils et des technologies utilisés.

1.7.6 *Le processus unifié*

Le Processus Unifié (PU), également connu sous le nom de Rational Unified Process (RUP), est une méthode de développement logiciel qui suit une approche itérative et incrémentale. Le PU est un cadre de développement logiciel qui fournit des directives détaillées pour toutes les étapes du développement logiciel, de la planification initiale à la maintenance et la mise à jour continue du logiciel. Le processus Unifié est une méthode de développement logiciel agile de la lignée dite unifiée qui nécessite une adaptation à chaque projet qui y a recours, elle est génerique, itérative et incrémentale.

Le PU est basé sur quatre principes fondamentaux :

— La modélisation visuelle : Le PU utilise des diagrammes UML (Unified Modeling Language) pour représenter le système logiciel.

— La gestion de projet : Le PU utilise une approche de gestion de projet itérative et agile qui permet de s'adapter aux changements des exigences et aux problèmes qui surgissent pendant le développement logiciel.
— La vérification et la validation : Le PU insiste sur la vérification et la validation du logiciel à chaque étape du processus pour s'assurer que le logiciel est conforme aux exigences du client et qu'il est de haute qualité.
— L'architecture centrée sur les cas d'utilisation : Le PU se concentre sur la conception de l'architecture du logiciel en se basant sur les cas d'utilisation qui décrivent les interactions entre les utilisateurs et le système.

Le PU est structuré en quatre phases principales :

1. La phase d'inception : Cette phase consiste à définir le périmètre du projet, à identifier les objectifs et les exigences, et à élaborer une vision claire du système logiciel.
2. La phase d'élaboration : Cette phase consiste à élaborer une architecture détaillée du système logiciel et à développer un plan de développement détaillé.
3. La phase de construction : Cette phase consiste à développer et à tester le logiciel en utilisant une approche itérative et incrémentale.
4. La phase de transition : Cette phase consiste à déployer le logiciel et à assurer la formation des utilisateurs, la maintenance et la mise à jour continue.

1.8 GIT

Git est un système de contrôle de version distribué utilisé pour le suivi des modifications apportées à un ensemble de fichiers, généralement du code source. Il est utilisé par les développeurs de logiciels pour collaborer sur des projets, suivre l'historique des modifications, effectuer des sauvegardes et des récupérations de code, et gérer les conflits de fusion.

Le fonctionnement de Git est basé sur un système de branches, où chaque développeur peut créer une branche de développement séparée pour travailler sur une fonctionnalité ou une correction de bogue spécifique. Les développeurs peuvent ensuite fusionner leurs branches avec la branche principale, généralement appelée "master", pour incorporer les modifications dans le code principal.

Git est largement utilisé dans l'industrie du développement de logiciels, ainsi que dans d'autres domaines où le suivi des modifications est important, tels que la rédaction de documents techniques, la gestion de projets et la collaboration sur des fichiers multimédias. Il est open source et gratuit, et est disponible pour Windows, macOS et Linux.

1.9 GITHUB

GitHub est une plateforme web basée sur Git, qui permet aux développeurs de collaborer sur des projets de développement de logiciels. Elle fournit un ensemble d'outils pour le suivi des versions, la gestion des projets, la collaboration, l'hébergement de code, l'intégration continue, etc.

Les utilisateurs peuvent créer des dépôts (des répertoires Git) sur GitHub pour stocker leur code source, et les partager publiquement ou en privé avec d'autres développeurs. GitHub offre également des fonctionnalités de suivi des problèmes, des demandes de fusion (pull requests), des commentaires, et des outils d'intégration continue pour automatiser les tests, les builds et les déploiements.

GitHub est devenu une plateforme très populaire pour la collaboration et la gestion de projets de développement de logiciels open source, mais il est également utilisé par de nombreuses entreprises pour la gestion de projets internes. Il offre une grande communauté de développeurs et de contributeurs, ainsi que de nombreuses intégrations avec d'autres outils de développement et de gestion de projets.

1.10 LINUX

Linux est un système d'exploitation libre et open source, basé sur le noyau Linux. Il a été développé par Linus Torvalds en 1991 et est maintenant utilisé dans une grande variété d'applications, allant des serveurs web et des supercalculateurs aux ordinateurs personnels et aux appareils mobiles.

Linux est connu pour être stable, sécurisé et flexible, avec une grande variété de distributions (telles que Ubuntu, Fedora, Debian, etc.) qui sont adaptées à différents cas d'utilisation et niveaux d'expertise. Linux est également connu pour son écosystème open source, qui offre une grande variété de logiciels gratuits et open source pour les tâches courantes, tels que la navigation sur le web, l'édition de texte, la création de documents, la gestion de fichiers, la programmation, etc.

Les distributions Linux sont souvent utilisées pour les serveurs web, les supercalculateurs, les systèmes embarqués, ainsi que pour les ordinateurs personnels et les appareils mobiles. Linux est également souvent utilisé comme plateforme de développement pour les développeurs qui travaillent avec des technologies open source, telles que Python, Ruby, Java, Go, Rust, etc.

En raison de sa flexibilité, de sa sécurité et de sa fiabilité, Linux est devenu l'un des systèmes d'exploitation les plus populaires dans l'industrie du développement de logiciels et de l'informatique en général.

1.11 LA MODÉLISATION

La modélisation est le processus de représentation d'un système, d'un processus, d'un objet ou d'un concept sous forme de modèle.

Un modèle est une représentation abstraite et simplifiée d'un système réel, qui peut être utilisée pour comprendre, analyser, prédire et concevoir le système.

1.12 UML (UNIFIED MODELING LANGUAGE)

UML (Unified Modeling Language) est un langage de modélisation visuel utilisé pour représenter des systèmes logiciels. UML fournit un ensemble de notations graphiques standardisées pour décrire les différents aspects d'un système, tels que les exigences, la conception, la mise en œuvre et le déploiement.

Voici les principales notations graphiques utilisées dans UML :

1.12.1 *Diagramme de classes*

Utilisé pour représenter la structure statique du système, en définissant les classes, les attributs, les méthodes et les relations entre les classes.

1.12.2 *Diagramme de cas d'utilisation*

Utilisé pour représenter les interactions entre les acteurs et le système, et pour définir les exigences fonctionnelles du système.

1.12.3 *Diagramme d'objets*

Utilisé pour représenter les instances des classes et les relations entre les objets.

1.12.4 *Diagramme de séquence*

Utilisé pour représenter les interactions entre les objets dans une séquence temporelle, en montrant les messages échangés entre les objets.

1.12.5 *Diagramme de collaboration*

Diagramme de collaboration : Utilisé pour représenter les interactions entre les objets en termes de messages échangés et de relations entre les objets.

1.12.6 *Diagramme d'états-transitions*

Utilisé pour représenter le comportement dynamique d'un objet ou d'un système, en montrant les états possibles et les transitions entre ces états.

1.12.7 *Diagramme d'activités*

Utilisé pour représenter le flux de contrôle dans un processus ou une méthode, en montrant les activités, les décisions et les boucles.

1.12.8 *Diagramme de déploiement*

Utilisé pour représenter la configuration physique du système, en montrant les composants matériels et logiciels et leurs relations.

1.13 HTML (HYPERTEXT MARKUP LANGUAGE)

L'HTML (Hypertext Markup Language) est un langage de balisage utilisé pour créer des pages web et des applications web. Il permet de structurer le contenu d'une page web en utilisant des balises qui décrivent le type de contenu et la manière dont il doit être affiché.

Les éléments HTML peuvent être utilisés pour définir des titres, des paragraphes, des images, des liens, des formulaires, des tableaux et bien plus encore. Les balises HTML sont écrites sous forme de code et sont utilisées pour décrire la structure et le contenu d'une page web.

En utilisant l'HTML, les développeurs web peuvent créer des pages web interactives et dynamiques qui peuvent être affichées dans n'importe quel navigateur web. L'HTML est souvent combiné avec d'autres langages de programmation, tels que le CSS (Cascading

Style Sheets) pour la mise en forme et la présentation de la page, et le JavaScript pour la programmation côté client et la manipulation dynamique de la page.

L'HTML est un langage de base pour la création de sites web et est largement utilisé dans l'industrie du développement web. Il est également régulièrement mis à jour pour répondre aux besoins changeants des développeurs et des utilisateurs web.

1.14 CSS (CASCADING STYLE SHEETS)

Le CSS (Cascading Style Sheets) est un langage de feuilles de style utilisé pour définir la présentation visuelle et la mise en forme des pages web écrites en HTML (Hypertext Markup Language).

Le CSS est utilisé pour séparer la présentation visuelle d'une page web de son contenu, ce qui permet aux développeurs web de créer des designs et des mises en page plus sophistiqués et plus flexibles. Les feuilles de style CSS peuvent être utilisées pour définir les couleurs, les polices de caractères, les marges, les bordures, les effets de transition, les dispositions, les tailles et bien plus encore.

En utilisant le CSS, les développeurs web peuvent créer des pages web plus accessibles, plus cohérentes et plus conviviales pour les utilisateurs finaux. Les feuilles de style peuvent également être utilisées pour créer des designs responsifs, qui s'adaptent automatiquement à la taille et à la résolution de l'écran.

Le CSS est un langage de base pour la création de sites web modernes et est largement utilisé dans l'industrie du développement web. Il est également régulièrement mis à jour pour répondre aux besoins changeants des développeurs et des utilisateurs web.

1.15 UN LANGUAGE DE PROGRAMMATION

Un langage de programmation est un langage formel utilisé pour écrire des instructions qui seront exécutées par un ordinateur. Les langages de programmation sont utilisés pour créer des logiciels, des applications, des sites web, des systèmes d'exploitation, des jeux vidéo et de nombreux autres types de programmes informatiques.

Un langage de programmation se compose d'un ensemble de règles syntaxiques et sémantiques qui définissent la structure et le sens des instructions. Les instructions sont écrites sous forme de code source, qui est ensuite traduit en code machine par un compilateur ou un interprète.

Les langages de programmation peuvent être classés en plusieurs catégories, telles que :

1. Les langages de programmation impératifs : qui décrivent les étapes à suivre pour accomplir une tâche. Les langages de programmation impératifs peuvent être classés en deux catégories : les langages de programmation procéduraux et les langages de programmation orientés objet.
2. Les langages de programmation fonctionnels : qui se concentrent sur les fonctions mathématiques et l'évaluation d'expressions, etc.
3. Les langages de programmation orientés objet : qui sont basés sur des objets et leurs interactions, etc. Les langages de programmation orientés objet peuvent être classés en deux catégories : les langages de programmation orientés objet basés sur les classes et les langages de programmation orientés objet basés sur les prototypes.
4. Les langages de script : qui sont utilisés pour automatiser des tâches et des processus, etc. Les langages de script sont généralement interprétés plutôt que compilés.

5. Les langages de programmation déclaratifs : qui décrivent le résultat souhaité plutôt que les étapes à suivre pour l'obtenir,

Il existe de nombreux langages de programmation différents, chacun ayant ses propres avantages et inconvénients. Voici quelques-uns des langages de programmation les plus courants :

— C++,
— JavaScript,
— Java,
— CSharp,
— Ruby,
— Python,
— Etc.

1.16 UN FRAMEWORK

Un framework est un ensemble de composants logiciels préconçus qui fournissent une structure pour faciliter le développement d'applications. Ces composants incluent souvent des bibliothèques, des modules, des classes, des fonctions et des interfaces qui peuvent être utilisés pour résoudre des problèmes courants dans le développement d'applications, tels que l'authentification des utilisateurs, la gestion de base de données, le traitement de fichiers, la sécurité, etc.

Les frameworks sont souvent conçus pour permettre aux développeurs de se concentrer sur la logique métier de leur application, plutôt que sur les détails techniques de bas niveau. Ils fournissent une structure de base pour l'application, qui peut être étendue et personnalisée en fonction des besoins spécifiques de l'application.

Il existe de nombreux frameworks différents pour différents langages de programmation et pour différents types d'applications. Par exemple, dans le développement web, des frameworks populaires incluent, NestJs pour Javascript, Ruby on Rails pour Ruby, Django pour Python, Laravel pour PHP, et ASP.NET pour CSharp. Pour le développement d'applications mobiles, des frameworks comme React Native pour JavaScript et Flutter pour Dart sont de plus en plus populaires.

Les avantages d'utiliser un framework sont nombreux. Ils permettent aux développeurs de gagner du temps en fournissant une base solide pour l'application, ils peuvent accélérer le développement et la maintenance, et ils peuvent rendre le code plus facile à lire et à maintenir en raison de la structure préconçue et cohérente.

En résumé, un framework en développement logiciel est un ensemble de composants logiciels préconçus qui fournissent une structure pour faciliter le développement d'applications. Il permet aux développeurs de se concentrer sur la logique métier de leur application plutôt que sur les détails techniques de bas niveau, et peut accélérer le développement et la maintenance de l'application.

1.17 NODEJS

Node.js est une plateforme de développement open source qui permet d'exécuter du code JavaScript côté serveur. Elle a été développée en 2009 par Ryan Dahl et est basée sur le moteur JavaScript V8 de Google, qui est également utilisé dans le navigateur Chrome.

Node.js permet aux développeurs de créer des applications web dynamiques et évolutives en utilisant JavaScript des deux côtés de la plateforme, côté client et côté serveur. Il est souvent utilisé pour créer des serveurs web, des applications en temps réel, des API, des outils en ligne de commande, des scripts pour l'automatisation de tâches, etc.

Node.js est particulièrement adapté aux applications nécessitant une grande évolutivité, car il utilise un modèle événementiel non-bloquant (non-blocking event-driven model) qui permet de gérer de grandes quantités de connexions simultanées sans ralentir le serveur. Il est également facile à apprendre pour les développeurs qui connaissent déjà JavaScript, car il utilise la syntaxe et les structures de contrôle de base de JavaScript.

Node.js dispose d'une grande communauté de développeurs et de contributeurs qui ont créé de nombreuses bibliothèques et modules pour faciliter le développement d'applications. Il est également compatible avec de nombreuses autres technologies de développement web, telles que MongoDB, Express.js, React, Angular, etc.

En résumé, Node.js est une plateforme de développement JavaScript côté serveur qui permet de créer des applications web évolutives et dynamiques, en utilisant un modèle événementiel non-bloquant et en étant compatible avec de nombreuses autres technologies web.

1.18 UNE BASE DES DONNÉES

Une base de données est une collection organisée de données qui sont stockées de manière à permettre un accès facile et efficace, ainsi que des opérations de gestion de données. Elle est utilisée pour stocker et organiser des informations pour une utilisation ultérieure.

Dans le contexte du développement logiciel, une base de données est souvent utilisée pour stocker les données de l'application, telles que les informations utilisateur, les commandes, les messages, les transactions, etc. Elle permet également aux développeurs d'accéder facilement à ces données, de les organiser et de les manipuler à l'aide de requêtes SQL (Structured Query Language), qui est le langage de programmation standard pour interagir avec les bases de données relationnelles.

1.18.1 *Bases de données relationnelles*

Les bases de données relationnelles sont les plus couramment utilisées dans le développement logiciel. Elles stockent les données dans des tables avec des relations entre elles, et permettent aux développeurs de les manipuler à l'aide de requêtes SQL. MySQL, PostgreSQL et Microsoft SQL Server sont des exemples de bases de données relationnelles populaires.

1.18.2 *Bases de données NoSQL*

Les bases de données NoSQL (Not Only SQL) sont une alternative aux bases de données relationnelles. Elles sont souvent utilisées pour des applications à grande échelle et nécessitant une grande évolutivité. Les bases de données NoSQL stockent les données sous forme de documents, de graphes ou de paires clé-valeur, et utilisent souvent des langages de requête spécifiques plutôt que SQL. MongoDB, Cassandra et Couchbase sont des exemples de bases de données NoSQL populaires.

1.18.3 *Bases de données orientées objet*

Les bases de données orientées objet sont conçues pour stocker des objets plutôt que des tables. Elles sont souvent utilisées dans les applications de programmation orientée objet, où les données sont stockées sous forme d'objets avec des attributs et des méthodes. db4o et ObjectDB sont des exemples de bases de données orientées objet populaires.

1.18.4 *Bases de données en mémoire*

Les bases de données en mémoire stockent les données en mémoire vive plutôt que sur un disque dur, ce qui permet des temps d'accès plus rapides et une meilleure performance. Elles sont souvent utilisées dans les applications nécessitant une réponse rapide, comme les jeux en ligne et les applications financières. Redis et Memcached sont des exemples de bases de données en mémoire populaires.

1.19 une spa (single page application)

Une SPA (Single Page Application) est une application web qui fonctionne sur une seule page, où tout le contenu est chargé dynamiquement sans avoir besoin de recharger la page complète lorsqu'un utilisateur interagit avec l'application.

Contrairement aux applications web traditionnelles qui ont plusieurs pages, une SPA est construite comme un ensemble de composants qui sont chargés dynamiquement, en réponse aux interactions de l'utilisateur. Les données sont généralement récupérées de manière asynchrone à l'aide d'API (Application Programming Interface) et de technologies côté client telles que JavaScript et AJAX.

Les avantages d'une SPA comprennent une expérience utilisateur plus rapide et plus fluide, car les interactions sont plus rapides et plus fluides que dans une application web traditionnelle. En outre, une SPA peut permettre aux développeurs de réduire la complexité du code et d'améliorer la maintenabilité en utilisant des frameworks et des bibliothèques qui facilitent le développement d'applications web dynamiques.

Cependant, une SPA peut également présenter des inconvénients, notamment une dépendance accrue aux technologies côté client, ce qui peut rendre l'application plus vulnérable aux attaques de sécurité et poser des problèmes de compatibilité avec certains navigateurs.

1.20 ui / ux

Le UI (User Interface) fait référence à l'interface utilisateur, c'est-à-dire la partie visible et interactive d'une application ou d'un système informatique qui permet à l'utilisateur d'interagir avec la technologie.

L'UI englobe l'ensemble des éléments visuels et interactifs tels que les boutons, les menus, les champs de saisie, les images, les icônes, les animations, etc. Elle est conçue pour offrir une expérience utilisateur (UX) optimale et intuitive, en facilitant la navigation et la compréhension du système.

Le développement de l'UI est une partie importante du processus de développement logiciel, qui nécessite une collaboration étroite entre les développeurs et les designers pour s'assurer que l'interface utilisateur répond aux besoins et aux attentes des utilisateurs finaux.

Le UX (User Experience) est l'expérience utilisateur globale d'un système informatique ou d'une application, qui comprend toutes les interactions que l'utilisateur a avec le système, avant, pendant et après l'utilisation.

Le UX englobe l'ensemble des aspects de l'interface utilisateur, y compris la navigation, la disposition des éléments, la cohérence visuelle, la facilité d'utilisation, la rapidité de réponse, la convivialité, etc. Le but du UX est de fournir une expérience utilisateur optimale et agréable, en répondant aux besoins et aux attentes des utilisateurs finaux.

1.21 LE DÉPLOIEMENT

Le déploiement en développement logiciel fait référence au processus de mise en production d'une application ou d'un logiciel, c'est-à-dire de le rendre disponible et fonctionnel pour les utilisateurs finaux.

Le déploiement implique plusieurs étapes, notamment la préparation de l'environnement de production, la configuration des serveurs, la mise en place des bases de données et des systèmes de stockage, la compilation du code source, la mise en place des fichiers de configuration, la vérification des dépendances, et la résolution de tout problème de compatibilité.

Une fois que l'application est prête à être déployée, elle est souvent transférée sur un serveur de production ou un environnement de cloud computing, où elle peut être mise à la disposition des utilisateurs finaux. Les développeurs peuvent mettre en place des stratégies de déploiement automatique pour faciliter le processus de déploiement et réduire le risque d'erreurs humaines.

Le déploiement est une étape clé du cycle de vie du développement logiciel, car il permet de rendre l'application disponible pour les utilisateurs finaux, ce qui peut générer des retours d'utilisateurs et des données d'utilisation qui peuvent être utilisés pour améliorer l'application dans les versions futures.

1.22 UN SERVEUR

Dans le contexte des applications web, un serveur est un ordinateur ou un logiciel qui fournit des services à d'autres ordinateurs ou logiciels. Le serveur est souvent utilisé pour héberger des applications web, des sites web ou des API, et pour répondre aux demandes des clients, qui peuvent être des navigateurs web ou des applications clientes.

Dans le contexte de l'architecture client-serveur, le serveur est la partie centrale qui gère les données et les traitements, tandis que les clients communiquent avec le serveur pour obtenir des informations et effectuer des opérations. Le serveur peut être un serveur de base de données, un serveur d'application, un serveur de fichiers, etc.

Dans le contexte des applications distribuées, un serveur peut être un nœud qui fournit des services à d'autres nœuds dans un réseau peer-to-peer ou dans un réseau de type client-serveur.

1.22.1 *Le serveur mutualisé*

Un serveur mutualisé est un type de serveur qui est partagé entre plusieurs utilisateurs, qui utilisent tous les mêmes ressources du serveur, telles que la mémoire, la puissance de traitement, l'espace de stockage, etc.

Dans un environnement de serveur mutualisé, chaque utilisateur dispose d'un compte d'hébergement séparé, mais tous les comptes partagent les mêmes ressources du serveur. Cela signifie que les performances et la stabilité du serveur peuvent être affectées par les besoins en ressources des autres utilisateurs, en particulier si certains utilisateurs consomment une grande quantité de ressources.

Les serveurs mutualisés sont souvent utilisés pour héberger des sites web de petite ou moyenne taille, des blogs, des forums ou des applications qui n'ont pas besoin de beaucoup de ressources système. Ils sont également moins coûteux que les autres options d'hébergement, comme les serveurs dédiés ou les VPS.

Cependant, en raison du partage des ressources, les serveurs mutualisés peuvent présenter des inconvénients, tels que des temps de chargement plus lents, des problèmes de sécurité et des limitations sur les logiciels et les fonctionnalités disponibles.

1.22.2 *Le serveur dédié*

Un VPS (Virtual Private Server) est un type de serveur qui permet à plusieurs utilisateurs de partager les ressources d'un même serveur physique, tout en ayant chacun leur propre environnement de serveur virtuel isolé. Un VPS (Virtual Private Server) est un type de serveur qui permet à plusieurs utilisateurs de partager les ressources d'un même serveur physique, tout en ayant chacun leur propre environnement de serveur virtuel isolé.

En d'autres termes, un VPS est un serveur virtuel qui fonctionne sur un serveur physique et qui est divisé en plusieurs instances isolées les unes des autres. Chaque instance, appelée "machine virtuelle", dispose de son propre système d'exploitation, de ses propres ressources système et de ses propres applications, ce qui permet à chaque utilisateur de personnaliser son environnement de serveur selon ses besoins.

Les VPS sont souvent utilisés pour héberger des sites web, des applications et des services en ligne, car ils offrent une flexibilité et une évolutivité importantes. Ils permettent également aux utilisateurs de bénéficier des avantages d'un serveur dédié, tels que des performances élevées, une sécurité renforcée et un accès root complet, sans avoir à investir dans leur propre matériel ou à supporter les coûts associés à la gestion d'un serveur physique. En d'autres termes, un VPS est un serveur virtuel qui fonctionne sur un serveur physique et qui est divisé en plusieurs instances isolées les unes des autres. Chaque instance, appelée "machine virtuelle", dispose de son propre système d'exploitation, de ses propres ressources système et de ses propres applications, ce qui permet à chaque utilisateur de personnaliser son environnement de serveur selon ses besoins.

Les VPS sont souvent utilisés pour héberger des sites web, des applications et des services en ligne, car ils offrent une flexibilité et une évolutivité importantes. Ils permettent également aux utilisateurs de bénéficier des avantages d'un serveur dédié, tels que des performances élevées, une sécurité renforcée et un accès root complet, sans avoir à investir dans leur propre matériel ou à supporter les coûts associés à la gestion d'un serveur physique.

2

ÉTAT DE L'ART

Avant de commencer à développer notre application, nous avons d'abord comparer les différentes applications existantes pour en dégager les fonctionnalités clés, nous avons ensuite identifié les fonctionnalités susceptibles de répondre aux besoins de l'Université Nouveaux Horizons et nous avons ensuite justifié nos choix de technologies et fonctionnalités.

2.1 PRÉSENTATION DES PRINCIPALES APPLICATIONS EXISTANTES

Nous sommes conscients qu'il en existe certainement lusieurs autres, mais nous avons choisi de nous limiter à celles mentioner ci dessous car elles sont les plus utilisées, les plus populaires et couvrent la plupart des besoins des leurs utilisateurs.

2.1.1 *Moodle*

1. **Présentation de l'application**
   Moodle est un système de gestion de l'apprentissage (LMS) open source, c'est-à-dire un logiciel permettant la création, la gestion, la distribution et la surveillance de cours en ligne. Il permet aux enseignants et aux formateurs de créer des cours en ligne interactifs, d'organiser des activités d'apprentissage, de communiquer avec les étudiants et de suivre leur progression.
2. **Fonctionalités clés**
   — Gestion des cours : Moodle permet de créer des cours en ligne et d'organiser des activités d'apprentissage en utilisant des outils tels que des forums de discussion, des leçons, des wikis, des glossaires, des quiz, des devoirs, des sondages, etc.
   — Gestion des utilisateurs : Moodle permet de gérer les utilisateurs, de les inscrire aux cours, de les suivre et de leur attribuer des rôles spécifiques (enseignant, étudiant, administrateur, etc.).
   — Communication : Moodle offre plusieurs outils de communication, y compris les forums de discussion, les messages privés, les chats en temps réel et les webinaires.
   — Suivi des progrès : Moodle permet aux enseignants de suivre les progrès des étudiants en visualisant leurs notes, leurs activités et leurs contributions sur les forums de discussion et autres outils.
   — Personnalisation : Moodle offre de nombreuses options de personnalisation pour les enseignants et les administrateurs, notamment la personnalisation de l'apparence du site, la gestion des catégories de cours, la création de rapports personnalisés, etc.
   — Accessibilité : Moodle est conforme aux normes d'accessibilité pour les personnes handicapées, ce qui permet à tous les utilisateurs de profiter pleinement des activités d'apprentissage en ligne.
3. **Inconvénients et limites**
   — Expertise technique requise : Moodle est un logiciel complexe dont l'installation et la configuration requièrent une certaine expertise technique. Si vous ne disposez

pas des compétences nécessaires, vous devrez peut-être faire appel à un consultant ou à un développeur pour vous aider.
— Sécurité : Moodle est une cible populaire pour les pirates informatiques, il est donc important de prendre des mesures pour sécuriser votre site Moodle. Cela inclut l'utilisation de mots de passe forts, l'installation de mises à jour de sécurité, et la surveillance du site pour toute activité suspecte.
— Evolutivité : Moodle peut être adapté à un grand nombre d'utilisateurs, mais il peut être difficile de gérer un grand site Moodle. Si vous envisagez d'utiliser Moodle pour une grande organisation, vous devrez peut-être engager un consultant ou un développeur pour vous aider à gérer votre site.
— Moodle est conçu pour répondre aux besoins généraux de l'enseignement en ligne. Il n'est pas conçu pour répondre aux besoins spécifiques d'une institution.

2.1.2 *PowerSchool*

1. **Présentation de l'application**
   PowerSchool est un système d'information pour les établissements scolaires qui permet de gérer les informations relatives aux étudiants, aux enseignants et aux programmes académiques. Il est utilisé par plus de 45 millions d'utilisateurs dans le monde entier, notamment dans les écoles primaires, secondaires et les établissements d'enseignement supérieur.
2. **Fonctionalités clés**
   — Gestion des inscriptions et des admissions : PowerSchool permet de gérer les inscriptions et les admissions des étudiants, y compris la collecte des informations de base, la gestion des documents requis, la communication avec les parents et les étudiants, et la planification des cours.
   — Gestion des notes et des absences : PowerSchool permet aux enseignants d'enregistrer les notes et les absences des étudiants, et de partager les informations avec les parents et les étudiants.
   — Gestion des emplois du temps et des calendriers académiques : PowerSchool permet de gérer les emplois du temps des enseignants et des étudiants, ainsi que les calendriers académiques et les événements scolaires.
   — Gestion des frais de scolarité et des paiements : PowerSchool permet de gérer les frais de scolarité et les paiements, y compris la collecte et le suivi des paiements, la génération de factures, et la communication avec les parents et les étudiants.
   — Communication avec les parents et les étudiants : PowerSchool permet de communiquer avec les parents et les étudiants via des messages personnalisés, des bulletins électroniques, des alertes automatiques, et d'autres outils de communication.
   — Gestion des bibliothèques : PowerSchool permet de gérer les bibliothèques de l'établissement, y compris la gestion des prêts de livres, la gestion des retours, et la gestion des réservations de livres.
   — Rapports et analyses : PowerSchool permet de générer des rapports et des analyses sur les performances académiques des étudiants, les tendances de fréquentation, les statistiques de paiement, et bien plus encore.

3. **Inconvénients et limites**
   — Coût : Le coût initial de mise en place de PowerSchool peut être élevé, en particulier pour les établissements scolaires plus petits. Les coûts peuvent également augmenter avec l'ajout de fonctionnalités supplémentaires.
   — Complexité : PowerSchool est un système complexe qui peut nécessiter une formation pour les administrateurs, les enseignants et les utilisateurs finaux. Les établissements scolaires doivent également consacrer du temps et des ressources à la configuration et à la gestion du système.
   — Personnalisation : Bien que PowerSchool offre de nombreuses fonctionnalités, il peut être difficile de personnaliser le système pour répondre aux besoins spécifiques de chaque établissement. Les établissements peuvent être limités dans leur capacité à personnaliser le système en fonction de leurs besoins spécifiques.
   — Dépendance : Les établissements qui utilisent PowerSchool dépendent du système pour gérer les données des étudiants et de l'établissement. En cas de panne du système ou de perte de données, cela peut avoir des conséquences importantes pour l'établissement.

2.1.3 *Esis Salama*

1. **Présentation**
   Esis salama est une institution d'enseignement supérieure de référence ic en République Démocratique du Congo, ce qui lui vaut une mention dans ce présent travail c'est la manière dont elle gère la délibération.
2. **Outils utilisés**
   — Excel : Ce dernier est un tableur édité par Microsoft pour les systèmes d'exploitation Windows et Mac OS X. Il est principalement utilisé pour le calcul, l'analyse de données et les graphiques. Excel est l'un des programmes les plus populaires pour la gestion des données et des calculs.
   — Plateforme développée en interne : Cette dernière facilite la publication des résultats des étudiants et permet aux étudiants de consulter leurs résultats et de faire des recours en ligne.
3. **Inconvénients et limites**
   — Cette façon de faire est très chronophage,
   — Nécessite beaucoup de ressources humaines

2.2 CONCLUSION

En conclusion nous retiendront que l'idéal serait de produire une application qui comblerait les lacunes des applications existantes et les étendrait mais l'idéal n'est pas toujours réalisable. Nous proposons donc une solution qui resoud les gros inconvenients et limitations des applications existantes et qui est facilement extensible et modifiable.

Les apports sont :

— Le fait que cette application soit développée en interne permettra de l'adapter aux besoins de l'Université Nouveaux Horizons et de rester indépendant des éditeurs de logiciels.
— L'application sera facilement extensible et modifiable.
— L'application peut facilement s'intégrer au système existant.

— L'application permettra de réduire les coûts en temps et ressources humaines car elle fera la grande partie du travail certe sous la supervision humaine.

3

CONCEPTION

Dans ce chapitre nous allons faire le tour des différentes décisions techniques prises, et les raisons qui nous ont poussé à les prendre. Nous allons commencer par présenter les technologies utilisées, puis nous allons parler du choix architectural de l'application, et enfin nous allons parler de la modélisation et des différents modèles utilisés.

3.1 CHOIX TECHNIQUES ET MOTIVATIONS

3.1.1 *MySQL*

Le choix de la base de données est une étape importante dans le développement d'une application. Il existe plusieurs types de bases de données, chacune avec ses propres avantages et inconvénients. Dans ce projet, nous avons choisi d'utiliser une base de données relationnelle, car elle est la plus adaptée à notre cas d'utilisation. Elle nous permet de stocker les données de manière structurée, et de les manipuler facilement à l'aide de requêtes SQL. Nous avons choisi d'utiliser MysQL, car c'est une base de données relationnelle populaire, open source et gratuite, avec une grande communauté de développeurs et une documentation complète.

3.1.2 *Architecture de l'application*

Nous avons choisi une architecture client-serveur pour notre application. Le serveur est responsable de la logique métier et de la gestion des données, et le client est responsable de l'interface utilisateur. Le client communique avec le serveur via une API REST (Representational State Transfer), qui est un ensemble de conventions et de bonnes pratiques pour la conception d'API web. L'API REST permet au client d'effectuer des opérations CRUD (Create, Read, Update, Delete) sur les données stockées sur le serveur. Nous avons choisi d'utiliser une API REST, car elle est simple à mettre en œuvre et facile à utiliser. Elle permet également de séparer la logique métier de l'interface utilisateur, ce qui facilite la maintenance et l'évolutivité de l'application. Nous avons choisi d'utiliser JSON (JavaScript Object Notation) comme format de données pour l'API REST, car il est léger, facile à lire et à écrire, et facilement extensible.

3.1.3 *UML*

Nous avons utilisé UML (Unified Modeling Language) pour modéliser notre application. UML est un langage de modélisation graphique standard utilisé pour représenter la structure et le comportement des systèmes logiciels. Il est utilisé pour visualiser, spécifier, construire et documenter les artefacts d'un système logiciel. UML est composé de plusieurs types de diagrammes, chacun représentant un aspect différent du système. Nous avons utilisé les diagrammes de cas d'utilisation, de classes, de séquence et d'activité pour modéliser notre application.

3.1.4 *Typescript*

Nous avons choisi d'utiliser Typescript pour le développement de notre application. Typescript est un langage de programmation open source développé par Microsoft. Il est conçu pour le développement d'applications JavaScript à grande échelle. Il ajoute des fonctionnalités au JavaScript, comme le typage statique, les classes, les interfaces, les modules, etc. Il est compilé en JavaScript, et peut donc être utilisé sur n'importe quel navigateur web ou serveur web. Nous avons choisi d'utiliser Typescript, car il est facile à apprendre et à utiliser, et il nous permet de détecter les erreurs de programmation avant l'exécution du code. Le plus gros avantages de Typescript est qu'il est facile à apprendre et à utiliser, et il nous permet de détecter les erreurs de programmation avant l'exécution du code, de plus il est compilé en JavaScript, et peut donc être utilisé sur n'importe quel navigateur web ou serveur web ce qui nous permet d'avoir un code client et serveur en Typescript on ne change donc pas de langage entre le client et le serveur.

3.1.5 *NestJS*

Nous avons choisi d'utiliser NestJS pour le développement l'API(partie serveur). NestJS est un framework open source pour le développement d'applications Node.js. Il est basé sur Express, et utilise Typescript. Il est conçu pour créer des applications évolutives et efficaces. Il utilise une architecture modulaire, et est composé de plusieurs modules, chacun avec sa propre fonctionnalité.

3.1.6 *NextJs*

Nous avons choisi d'utiliser NextJS pour le développement de notre interface utilisateur (partie client). web. NextJS est un framework open source pour le développement d'applications React. Il est basé sur React, et utilise Typescript. Il est conçu pour créer des applications web évolutives et efficaces. Il utilise une architecture modulaire, et est composé de plusieurs modules, chacun avec sa propre fonctionnalité.

3.1.7 *Git*

Nous avons choisi d'utiliser Git pour la gestion de version de notre code. Git est un système de contrôle de version distribué open source. Il est conçu pour gérer les petits et grands projets avec rapidité et efficacité. Il est utilisé pour suivre les modifications apportées au code source, et pour faciliter la collaboration entre les développeurs. Nous avons choisi d'utiliser Git, car il est facile à apprendre et à utiliser, et il nous permet de suivre les modifications apportées au code source, et de revenir à une version antérieure du code source si nécessaire.

3.1.8 *GitHub*

Nous avons choisi d'utiliser GitHub pour l'hébergement de notre code. GitHub est un service d'hébergement de code source basé sur Git. Il est conçu pour faciliter la collaboration entre les développeurs. Il est utilisé pour héberger des projets open source et privés. Nous

avons choisi d'utiliser GitHub, car il est facile à apprendre et à utiliser, et il nous permet de faciliter la collaboration entre les développeurs.

3.1.9 *Tailwindcss*

Nous avons choisi d'utiliser Tailwindcss pour le développement de notre interface utilisateur (partie client). web. Tailwindcss est un framework open source pour le développement d'interfaces utilisateur. Il est conçu pour créer des interfaces utilisateur personnalisées et réactives. Il est utilisé pour créer des interfaces utilisateur réactives et personnalisées. Nous avons choisi d'utiliser Tailwindcss, car il est facile à apprendre et à utiliser, et il nous permet de créer des interfaces utilisateur réactives et personnalisées.

3.1.10 *Fedora*

Fedora est une distribution Linux libre et open source, développée par la communauté et soutenue par Red Hat, une entreprise spécialisée dans les logiciels open source. Fedora est connue pour être une distribution Linux innovante et à la pointe de la technologie, avec une forte orientation vers les développeurs et les utilisateurs avancés.

Fedora utilise le gestionnaire de paquets RPM (RPM Package Manager) pour installer et gérer les logiciels, et propose une large sélection d'applications open source pour les tâches courantes, telles que la navigation sur le web, la création de documents, la gestion de fichiers, la programmation, etc. Il est également connu pour son support des technologies émergentes telles que Flatpak, qui permet d'emballer des applications pour qu'elles s'exécutent sur n'importe quelle distribution Linux.

Fedora est souvent utilisé comme une plateforme de développement pour les développeurs qui travaillent avec des technologies open source, telles que Python, Ruby, Java, Go, Rust, etc. Il est également populaire auprès des utilisateurs avancés qui cherchent à personnaliser leur environnement de bureau et à expérimenter avec de nouvelles technologies. Fedora est une distribution Linux communautaire et soutenue par la communauté, ce qui signifie que les utilisateurs peuvent participer au développement et à l'amélioration de la distribution en contribuant à la documentation, aux tests, aux correctifs, etc.

Conclusion

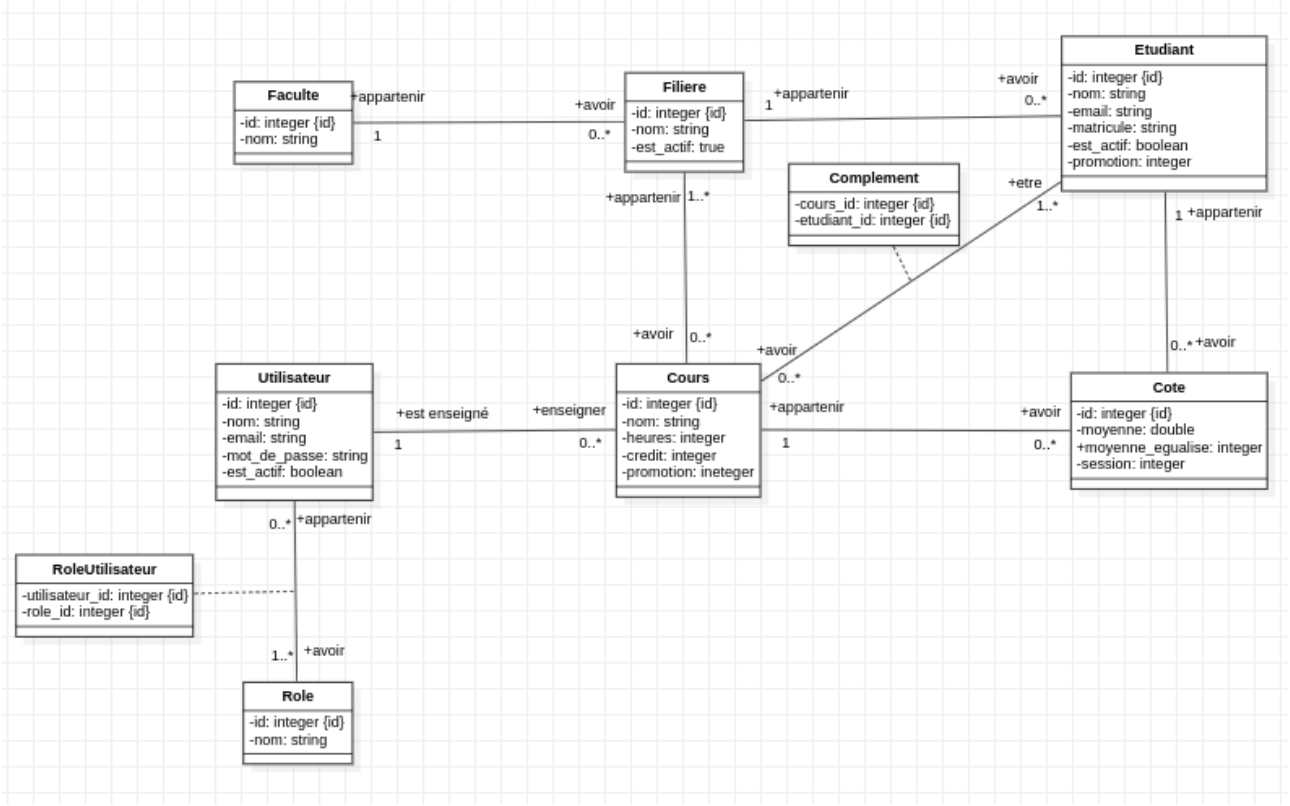
Nous retiendrons par ailleurs que toutes les décisions prises ne sont pas forcément les meilleures, mais elles ont été prises en fonction des contraintes et des objectifs du projet. Nous voulons non seulement résoudre le problème de la délibération, mais aussi fournir une solution qui soit facile à maintenir et à faire évoluer à l'avenir et facilement adaptable à d'autres problèmes similaires surtout extensible.

Le choix d'une API offre une grande flexibilité et une grande extensibilité car la solution peut être utilisée par d'autres applications et peut être facilement intégrée à d'autres systèmes et peut être utilisée pour concevoir un autre type d'application (mobile, web, desktop, etc.).

3.2 MODÈLES

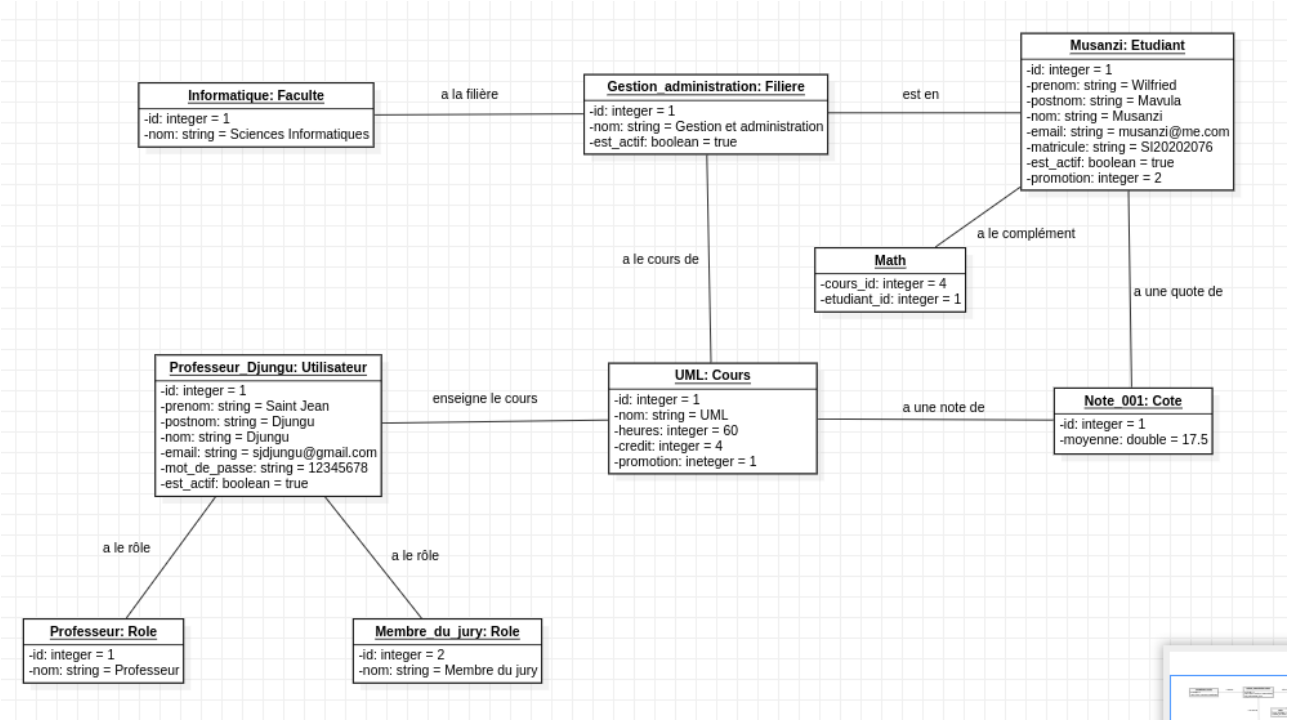
3.2.1 *Diagramme de classes*

FIGURE 1 – Diagramme de classes



3.2.2 *Diagramme d'objets*

FIGURE 2 – Diagramme d'objets



3.2.3 *Diagramme de cas d'utilisation*

FIGURE 3 – Cas d'utilisation : Gestion des facultés

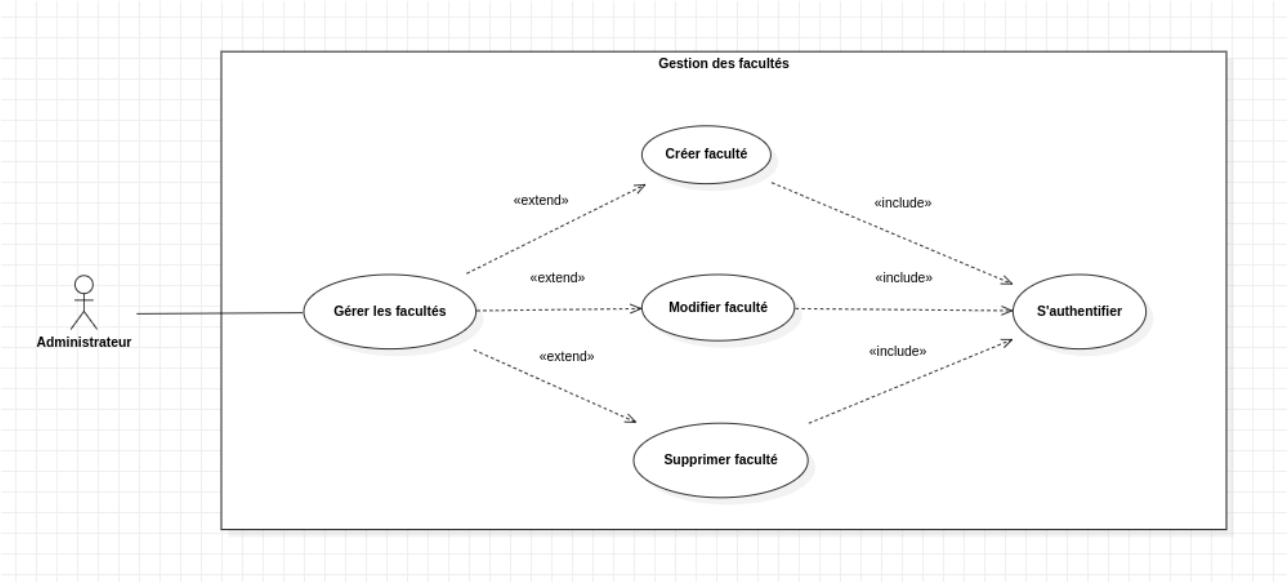


FIGURE 4 – Cas d'utilisation : Gestion des filières

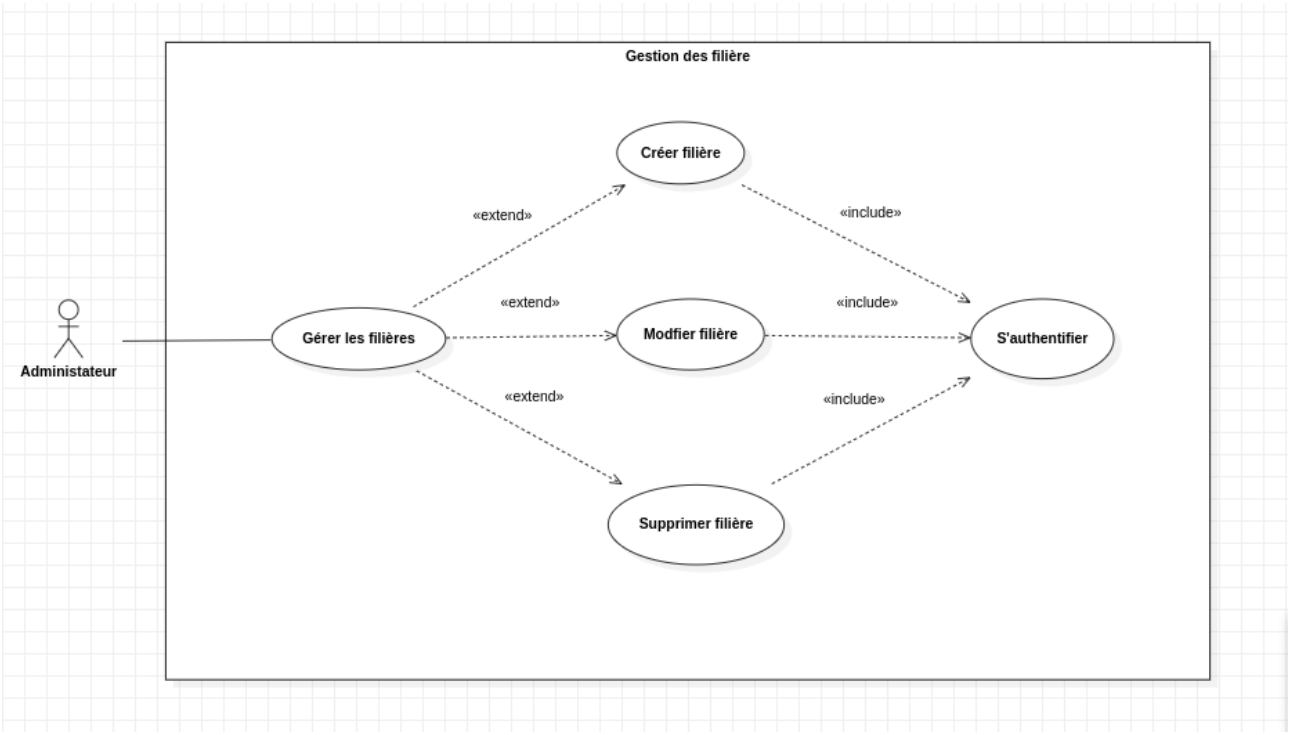


FIGURE 5 – Cas d'utilisation : Gestion des cours

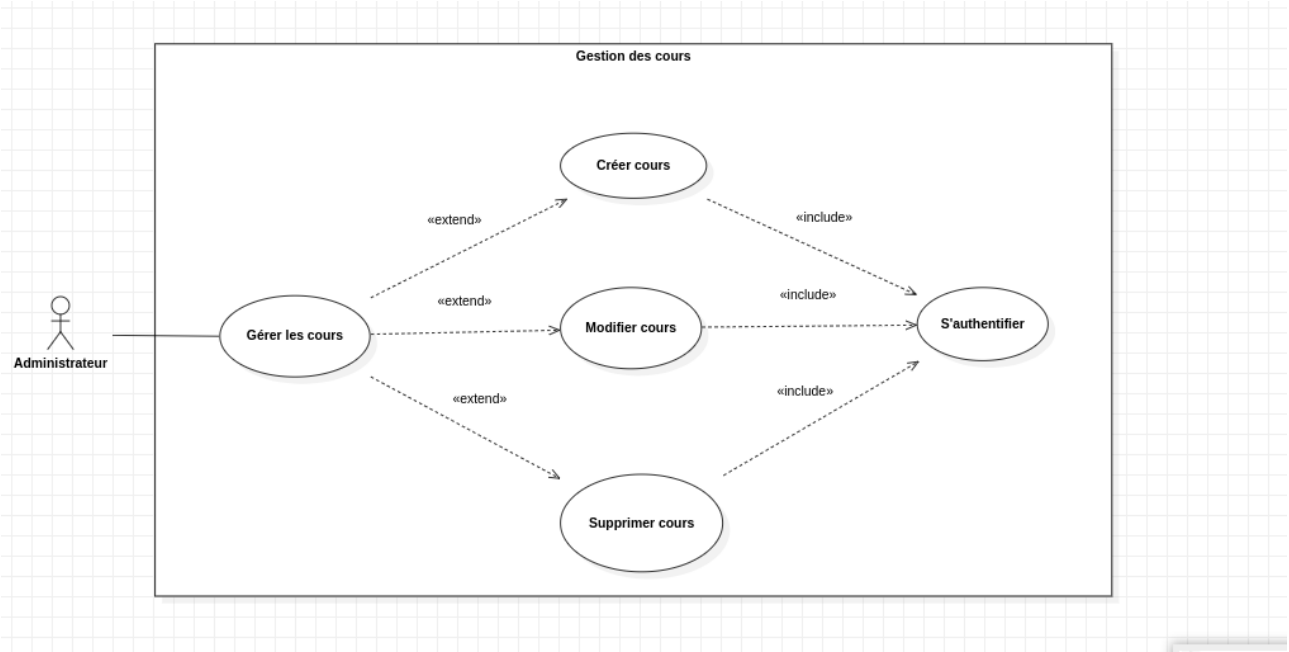


FIGURE 6 – Cas d'utilisation : Gestion des cotes

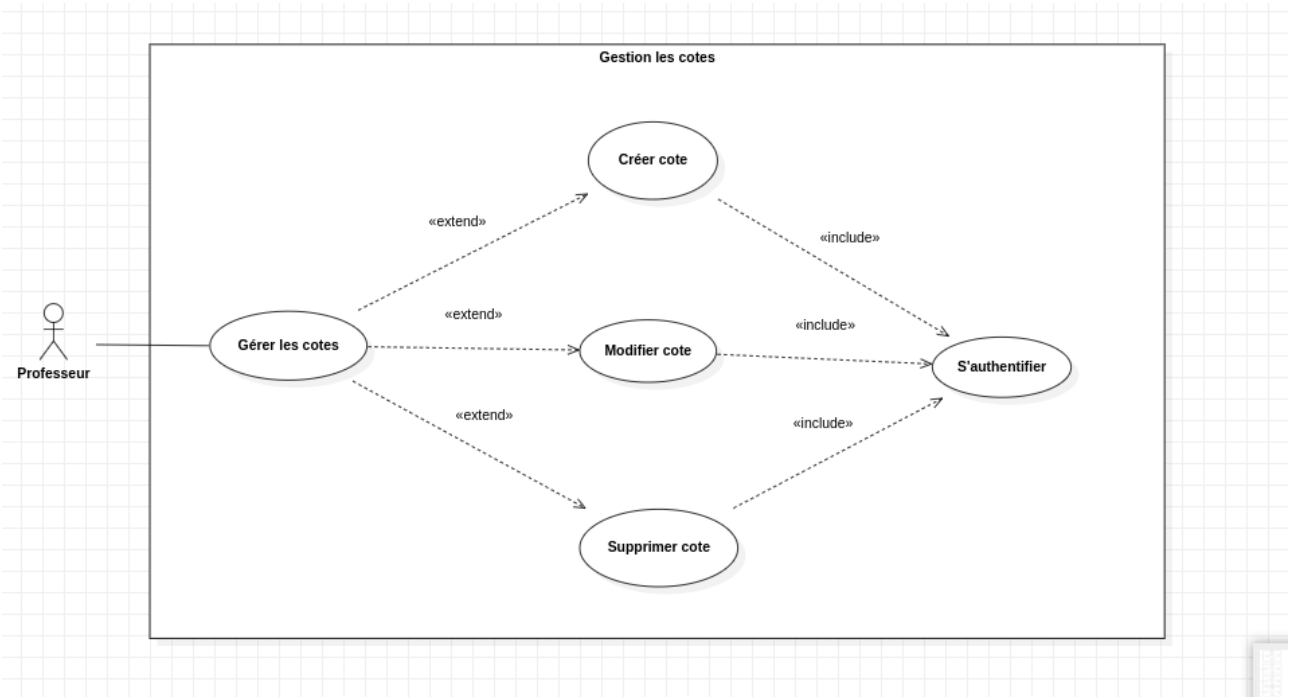


FIGURE 7 – Cas d'utilisation : Gestion des étudiants

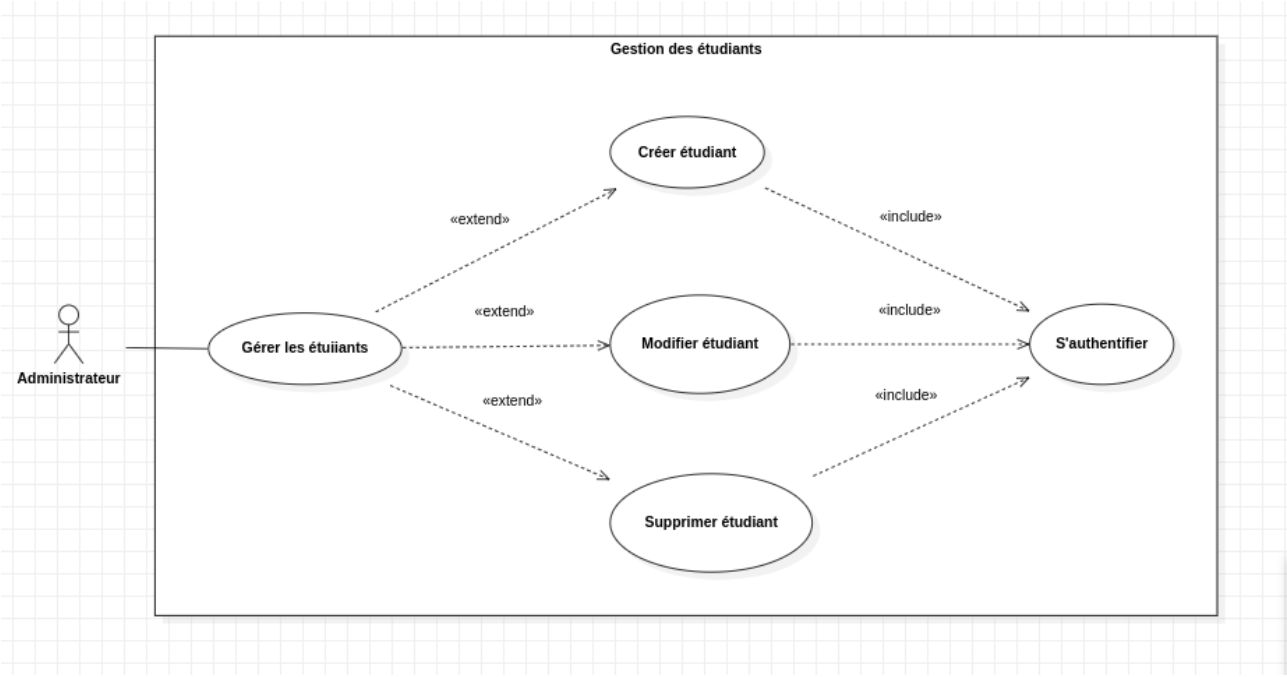


FIGURE 8 – Cas d'utilisation : Gestion des utilisateurs

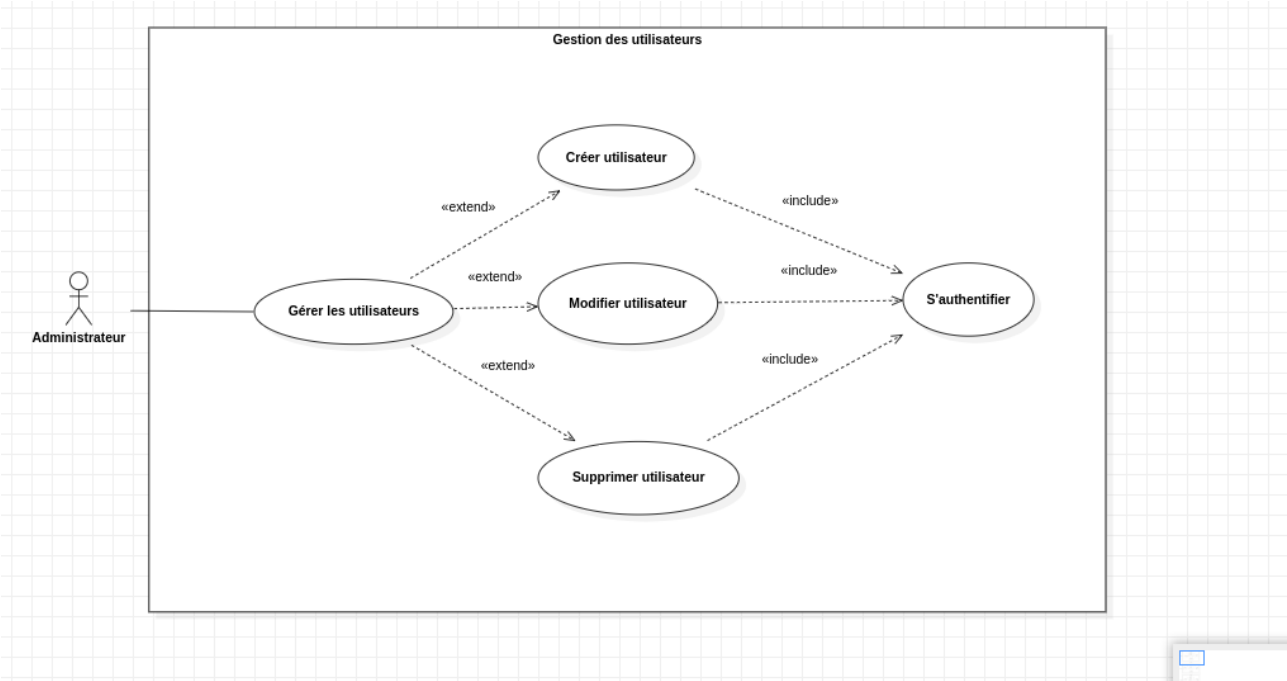


FIGURE 9 – Cas d'utilisation : Gestion des rôles

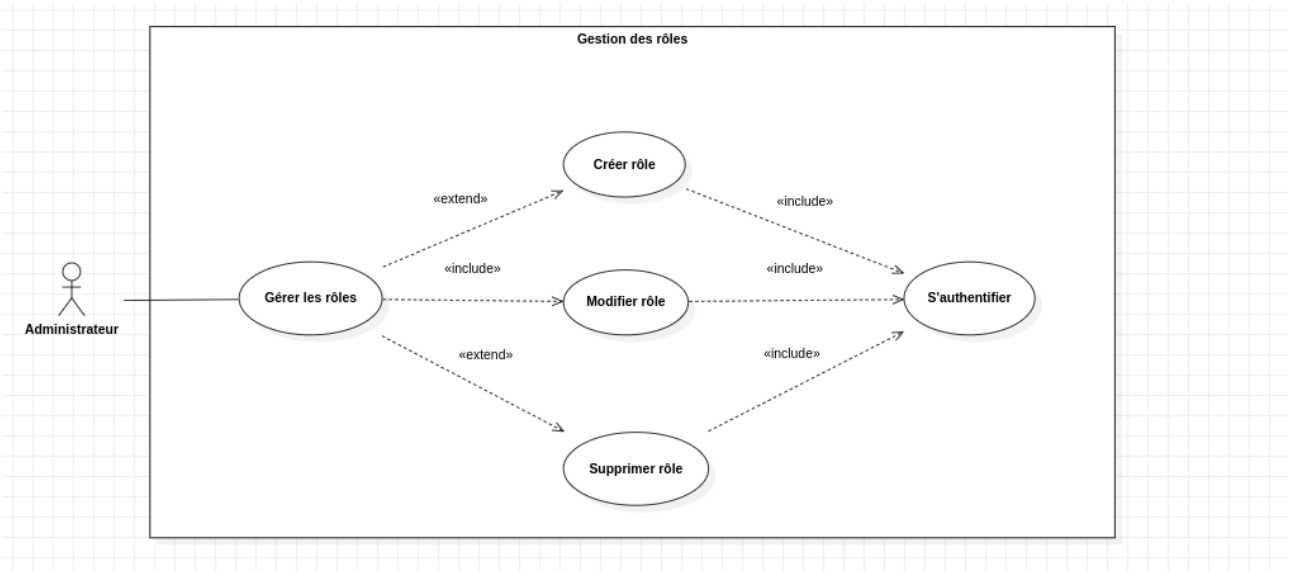


FIGURE 10 – Cas d'utilisation : Création des PDFs(Relevés)

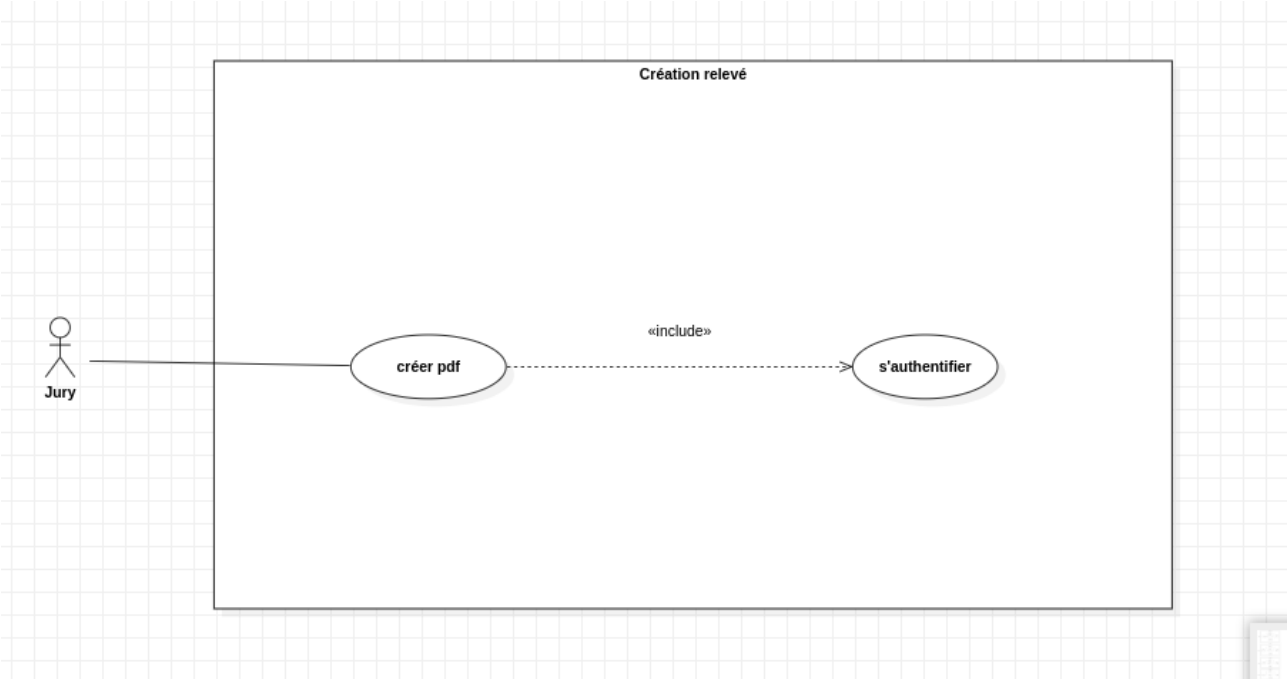
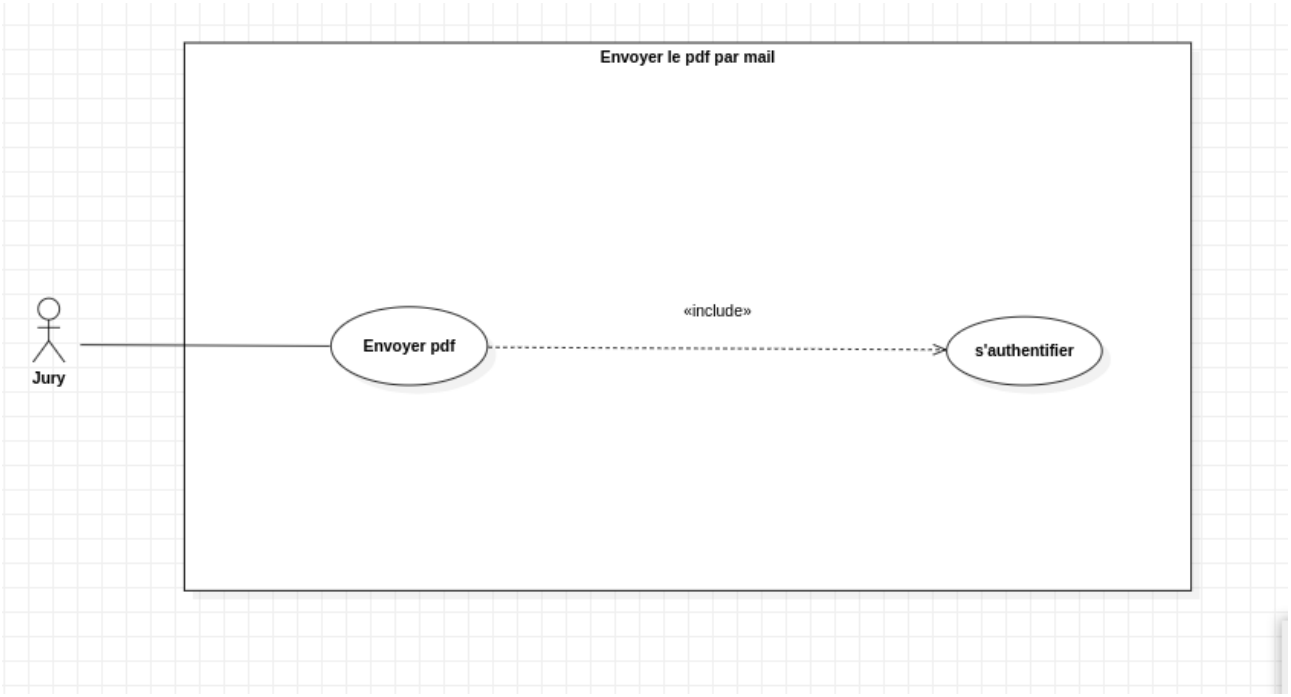
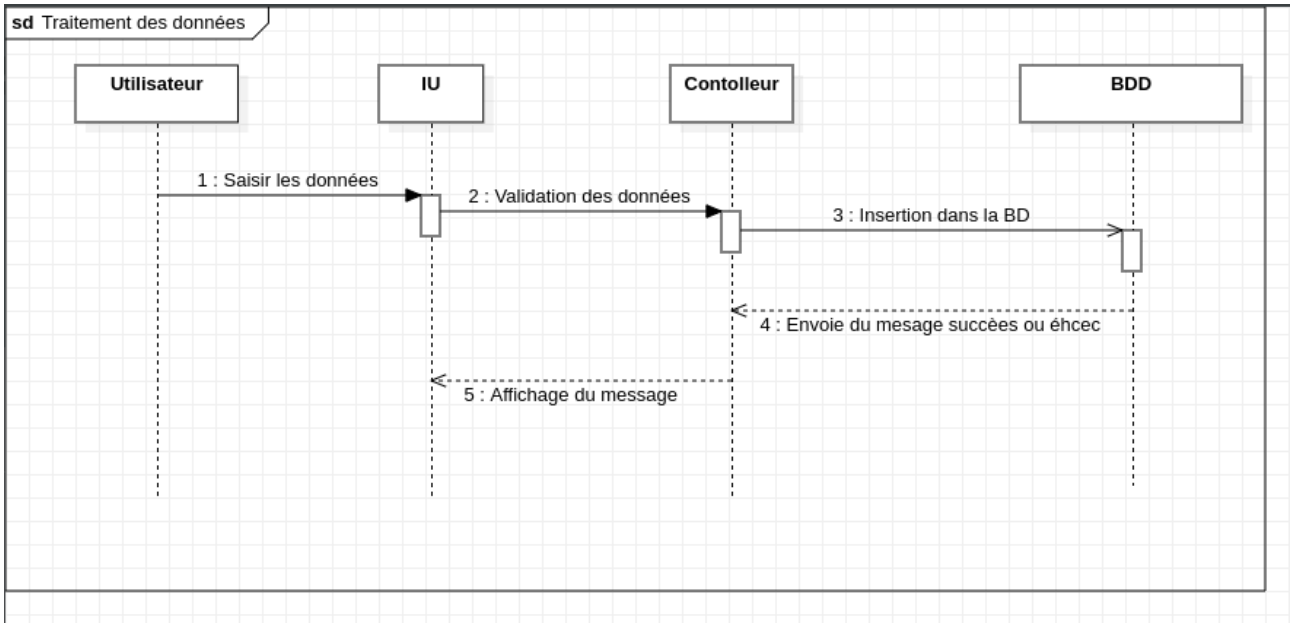


FIGURE 11 – Cas d'utilisation : Envoyer le pdf par mail



3.2.4 *Diagramme de séquence*

FIGURE 12 – Diagramme de séquence



4 PRÉSENTATION DES RÉSULTATS

4.1 PRÉSENTATION DE L'API REST

L'API comporte 44 routes au total que nous avons catégorisées en 9 groupes selon la ressource qu'elles manipulent. Nous allons présenter ces routes dans l'ordre de leur création.

4.1.1 *Les routes de l'authentification*

Préfixe des routes :

```
1    localhost:3000/auth
```

Méthodes	Routes	Paramètres	Corps de la requête	Réponses
POST	/login	-	{ email, mot de passe }	{ jeton }
GET	/profile	-	-	utilisateur : { id, nom, email, mot de passe, est_actif }
PATCH	/profile	-	{ nom, email }	{ status, message }
PATCH	/update-password	-	{ Ancien mot de passe, nouveau mot de passe }	{ status, message }

4.1.2 *Les routes des facultés*

Préfixe des routes :

`localhost:3000/faculties`

Méthodes	Actions	Routes	Paramètres	Corps de la requête	Réponses
POST	Créer	/	-	{ nom }	{ status, message }
GET	Lire	/	-	-	facultés[] : { id, nom }
GET	Lire	/ :id	{ id }	-	facultés : { id, nom }
PATCH	Modifier	/ :id	{ id }	{ nom }	{ status, message }
DELETE	Supprimer	/ :id	{ id }	-	{ status, message }

4.1.3 *Les routes des filières*

Préfixe des routes :

`localhost:3000/fields`

Méthodes	Actions	Routes	Paramètres	Corps de la requête	Réponses
POST	Créer	/	-	{ nom, faculté }	{ status, message }
GET	Lire	/	-	-	filières[] : { id, nom, faculté }
GET	Lire	/ :id	{ id }	-	filière : { id, nom, faculté }
PATCH	Modifier	/ :id	{ id }	{ nom, faculté }	{ status, message }
DELETE	Supprimer	/ :id	{ id }	-	{ status, message }

4.1.4 *Les routes des rôles*

Préfixe des routes :

`localhost:3000/roles`

Méthodes	Actions	Routes	Paramètres	Corps de la requête	Réponses
POST	Créer	/	-	{ nom }	{ status, message }
GET	Lire	/	-	-	rôles[] : { id, nom }
GET	Lire	/ :id	{ id }	-	rôles : { id, nom }
PATCH	Modifier	/ :id	{ id }	{ nom }	{ status, message }
DELETE	Supprimer	/ :id	{ id }	-	{ status, message }

4.1.5 *Les routes des utilisateurs*

Préfixe des routes :

`localhost:3000/users`

Méthodes	Actions	Routes	Paramètres	Corps de la requête	Réponses
POST	Créer	/	-	{ nom, email, roles[] }	{ status, message }
GET	Lire	/	-	-	rôles[] : { id, nom, email, roles[] }
GET	Lire	/ :id	{ id }	-	utilisateur : { id, nom, email, roles[] }
PATCH	Modifier	/ :id	{ id }	{ nom, email, roles[] }	{ status, message }
DELETE	Supprimer	/ :id	{ id }	-	{ status, message }

4.1.6 *Les routes des étudiants*

Préfixe des routes :

`localhost:3000/users`

Méthodes	Actions	Routes	Paramètres	Corps de la requête	Réponses
POST	Créer	/	-	{ nom, email, matricule, filière, promotion, compléments }	{ status, message }
GET	Lire	/	-	-	étudiants[] : { id, nom, email, matricule, filière, promotion, compléments }
GET	Lire	/ :id	{ id }	-	utilisateur : { id, nom, email, roles[] }
PATCH	Modifier	/ :id	{ id }	{ nom, email, roles[] }	{ status, message }
DELETE	Supprimer	/ :id	{ id }	-	{ status, message }
GET	Lire	fields/ :fieldId/promotions/promotionId	{ fieldId, promotionId }	-	étudiants[] : { id, nom, email, matricule, filière, promotion }
GET	Lire	courses/ :coursID	{ coursId }	-	étudiants[] : { id, nom, email, matricule, promotion, compléments }

4.1.7 *Les routes des cours*

Préfixe des routes :

```
localhost:3000/courses
```

Méthodes	Actions	Routes	Paramètres	Corps de la requête	Réponses
POST	Créer	/	-	{ nom, heures, crédits, promotion, enseignant, filière }	{ status, message }
GET	Lire	/	-	-	cours[] : { id, nom, heures, crédits, promotion, enseignant, filière }
GET	Lire	/ :id	{ id }	-	cours : { id, nom, heures, crédits, promotion, enseignant, filière }
PATCH	Modifier	/ :id	{ id }	{ nom, heures, crédits, promotion, enseignant, filière }	{ status, message }
DELETE	Supprimer	/ :id	{ id }	-	{ status, message }

4.1.8 *Les routes des cotes*

Préfixe des routes :

```
localhost:3000/grades
```

Méthodes	Actions	Routes	Paramètres	Corps de la requête	Réponses
POST	Créer	/	-	{ moyenne, moyenne_eg, session, promotion_etudiant, cours, étudiant }	{ status, message }
GET	Lire	/	-	-	cotes[] : { id, moyenne, moyenne_eg, session, promotion_etudiant, cours, étudiant }
GET	Lire	/ :id	{ id }	-	cote : { id, moyenne, moyenne_eg, session, promotion_etudiant, cours, étudiant }
PATCH	Modifier	/ :id	{ id }	{ id, moyenne, moyenne_eg, session, promotion_etudiant, cours, étudiant }	{ status, message }
DELETE	Supprimer	/ :id	{ id }	-	{ status, message }

4.1.9 *Les routes des délibérations*

Préfixe des routes :

```
localhost:3000/deliberations
```

Méthodes	Actions	Routes	Paramètres	Corps de la requête	Réponses
GET	Lire	/ :id	{ id }	-	{ nom, matricule, email, filière, cours[] }
GET	Lire	generate-reports/ :id	{ id }	-	{ status, message }
GET	Lire	send-reports/ :id	{ id }	-	{ status, message }