UNIVERSITE A VOCATION PROFESSIONNALISANTE

Athénée Saint Joseph Antsirabe

MEMOIRE DE FIN D'ETUDE EN VUE DE L'OBTENTION DU DIPLOME DE MASTER

***Domaine* :** Science et Technologie

Mention : Informatique

***Parcours* :** Génie Logiciel

Par :

5766

Titre :

CONCEPTION ET REALISATION D'UNE PLATEFORME UNIVERSITAIRE

« UniSphere »

Soutenu le, devant les membres du jury composé de :

Président :

Examinateurs :

Encadreur pédagogique :

BP 287 Antsirabe 110
Tél : 4448319/20
E-mail : info@asja-univ.net
Site : www.asja.univ

Année Universitaire : 2024 – 2025

TENY FISAORANA

Voalohany indrindra, dia isaorana betsaka Andriamanitra izay nanome ny zavatra rehetra ary ny fanampiany tamin'ny fanatontosana iti tetik'asa ity :

Mendrika omena fankasitrahan ihany koa ireto olona ireto :

- Ny Révérend Père CUOMO MARIO GUISEPPE, mpanorina ny anjerimanontolo ASJA, ny amin'ny fandavantenany sy ny finianvany nanorina ity sekoly ity mba ahafahan'ny tanora malagasy manohy ny fianarany
- Ny Révérend Père
- Andriamatoa
- Andriamatoa
- Andriamatoa
- Andriamatoa

Fisaorana lehibe ihany koa no omeko an'ireo Ray-aman-dreniko sy ny fianakaviako tamin'ireo fanohanana tamin'ny lafiny vola sy fankaherezan-tsaina nandritry ny dimy (5) taona nianarana tato. Farany, ankasitrahina ihany koa ireo namana rehetra na ny lavitra na ny akaiky izay nanampy tamin'ny fanantontosana ity asa ity

REMERCIEMENTS

Tout d'abord, remercions le Dieu tout puissant qui nous a tout donné et aidé lors l'accomplissement de notre projet de mémoire.

Nous tenons également à exprimer nos remerciements aux personnalités dont les suivent :

- Le Révérend Père CUOMO MARIO GUISEPPE, fondateur de l'université Athénée Saint Joseph Antsirabe, pour sa volonté et son courage de créer cette université afin d'aider les jeunes malagasy à continuer leurs études ;
- Le Révérend Père
- A Monsieur
- A Monsieur
- A Monsieur
- A Monsieur

Un grand merci aussi à nos parents et notre famille, pour leur soutien moral, spirituel et financier tout au long de ces années d'études. Enfin nous exprimons aussi notre gratitude à nos amis et camarades de promotion qui de près ou de loin, ont contribué à l'aboutissement de ce travail.

TABLE DES MATIERES

LISTE DES ABREVIATIONS

ASVS	Application Security Verification Standard
BPMN	Business Process Model and Notation
CERN	Conseil Européen pour la Recherche Nucléaire
CSS	Cascading Style Sheets
DOM	Document Object Model
ERD	Entity Relationship Diagram
HTML	HyperText Markup Language
IMT	Internationnal Business Machines
IOS	iPhone Operating System
ISO	International Organization for Standarization
JSON	JavaScript Object Notation
JWT	JSON Web Token
LMS	Learning Management System
MDA	Model Driven Architecture
MEAN	MongoDB, Express.js, Angular, Node.js
MERN	MongoDB, Express.js, React, Node.js
NPM	Node Package Manager
OMG	Object Management Group
OMT	Object Modeling Technique
OO	Object Oriented
OOD	Object Oriented Design
OOSE	Object Oriented Software Engineering
OWASP	Open Web Application Security Project

PDF	Portable Document Format
RUP	Rational Unified Process
SAAS	Software as a Service
SGBDR	Système de Gestion de Base de Données Relationnelle
SPA	Single Page Application
SQL	Structured Query Language
SUS	System Usability Scale
UI	User Interface
UML	Unified Modeling Language
UNESCO	United Nations Educational, Scientific and Cultural Organization
USB	Universal Serial Bus
VDOM	Virtual Document Object Model
W3S	World Wide Web Consortium
XAMPP	Cross-platform (X), Apache, MySQL/MariaDB, PHP, Perl
XML	eXtensible Markup Language

LISTES DES TABLEAUX ET DES FIGURES

1. Liste des tableaux

2. Liste des figures

INTRODUCTION GENERALE

Actuellement dans le monde, les avancées technologiques ont un impact considérable dans plusieurs domaines tels que le commerce qui est affecté par l'apparition et la propagation des ventes en lignes des produits, la communication qui à transitionne de la lettre à l'utilisation des téléphones portables et bien d'autres domaines. Cette avancée touche aussi l'éducation notamment les enseignements dans les Universités. Grâce à la propagation de l'internet, diverses plateformes de gestion d'apprentissage voient le jour tels que Moodle, Blackboard Learn, Google Classroom, et bien d'autre. Ces plateformes offrent des fonctionnalités diverses qui permettent de gérer les cours en ligne pour les Universités. Grâce à ces plateformes les établissements Universitaires pourront alors adapter et changer leur méthode d'enseignements traditionnelles en enseignements modernisés et adaptés aux avancées de l'informatique

Cependant, ces plateformes ne sont pas entièrement adaptées à la gestion spécifique des cours dans les Universités. Plusieurs contraintes ou limitations ont été constatées sur les plateformes existantes. Les plateformes comme Moodle possèdent de nombreuses fonctionnalités mais sa complexité bloque son adoption pour les enseignants non-initiés, ce qui cause des freinages lors de leur adoption dans les Universités. Il y a aussi les plateformes comme Google Classroom reste très dépendant de l'écosystème de Google, limitant sa personnalisation et son intégration avec d'autres systèmes universitaires. Dans l'ensemble, ces plateformes répondent partiellement aux besoins de l'enseignement supérieur. D'où notre question : comment concevoir une plateforme intuitive et adaptée aux universités ? Comme solution, notre projet de « **conception et réalisation d'une plateforme universitaire UniSphere** » vise à développer une plateforme universitaire conçue spécifiquement pour répondre aux limitations identifiées dans les outils existants. Cette plateforme vise à réunir dans un cadre unique et intuitif, diverses fonctionnalités de gestion éducative, de communication, tout en ayant une prise en main facile et adaptable aux réalités Universités.

Afin de mener à bien notre projet, va départager cet ouvrage en quatre chapitres. Dans le premier chapitre, on va effectuer une présentation générale du projet en commençant par sa provenance jusqu'à ces fonctionnalités. Ensuite dans le deuxième chapitre, l'analyse et la conception du projet, nous y expliquerons en détail les fonctionnalités mentionnées. Puis la réalisation du projet dans le troisième chapitre, nous exposerons les langages de programmation et outils utilisés lors de l'implémentation des fonctionnalités du projet. Enfin dans le dernier chapitre on y effectuera une évaluation de notre projet ainsi que les perspectives d'amélioration.

CHAPITRE 1 GENERALITE DU PROJET

1.1 Introduction

L'Université est une institution privée ou publique d'enseignement supérieur et de recherche qui offre diverses formations académiques dans des domaines variés. Chaque établissement possède des méthodes pédagogiques spécifiques pour transmettre le savoir et les formations et évaluer ses étudiants ; elle assure aussi la communication et la collaboration entre les différents acteurs du milieu universitaire. Notre projet vise à repenser et à moderniser certaines de ces méthodes. Dans ce contexte, ce chapitre présente le contexte général du projet, il décrit la structure et le fonctionnement d'une Université, les limites des outils existants mais aussi les objectifs, avantages et les contraintes qui sont liés à l'implémentation d'une plateforme Universitaire.

1.2 Contexte de mise en œuvre du projet

1.2.1 Définition de l'Université

L'Université est une institution d'enseignement supérieur et de recherche chargée de former des cadres qualifiés, de produire des connaissances, et de contribuer au développement social et économique. Elle regroupe plusieurs départements ou facultés spécialisés dans des disciplines variées telles que les lettres, les sciences, le droit et bien d'autres.

1.2.2 Missions et objectifs d'une Université

En tant qu'établissement éducatif, l'Université se voit accorder diverses missions fondamentales autour des axes suivants :

- La formation : la formation peut se définir d'une manière générale, comme : « l'action d'un formateur s'exerçant sur une ou plusieurs personnes en vue de les adapter techniquement, physiquement et psychologiquement à leur futures fonctions ». Il s'agit à la fois d'un apprentissage de connaissances et d'un apprentissage de méthodes de travail et de savoir-faire mais aussi d'une expérimentation de nouvelles attitudes et de nouveaux comportements. Elle permet l'adaptation à l'emploi, le développement du potentiel des individus, le développement intellectuel et rationnel, la croissance des capacités d'adaptation et de régulation de l'individu dans ses rapports avec son environnement professionnel. [1.01]
- La recherche : la recherche est l'examen minutieux d'une question ou d'un problème de recherche particulier à l'aide de méthodes scientifiques. Selon le sociologue américain Earl Robert Babbie, « la recherche est une enquête systématique visant à décrire, expliquer, prédire

et contrôler le phénomène observe. Elle fait appel à des méthodes inductives et déductives ». [1.02]

- Communication pédagogique : elle désigne un processus par lequel un pédagogue, qu'il soit enseignant ou formateur, transmet des informations, des idées et des compétences à un apprenant afin de favoriser son apprentissage. Ce processus est complexe et implique non seulement la transmission des connaissances mais aussi la création d'un environnement propice à l'apprentissage, l'encouragement de la participation active et la fourniture de rétroactions constructives. [1.03]
- Gestion académique : il s'agit de l'ensemble des processus, des politiques et des pratiques qui régissent l'organisation, l'administration et la supervision des activités académiques. [1.04]

Voici quelques exemples :

- Gestion, modification et création de programmes d'études
- Admission et inscription des personnes étudiantes.
- Cheminement des personnes étudiantes
- Affaires départementales (exemples : justification des postes, gestion des budgets départementaux, etc.).
- Affaires professorales (exemples : dégagements discrétionnaires, approbation de la tâche, etc…)
- Gestion, animation et financement de la recherche.

Ces missions exigent aux Université de posséder une organisation structurée, des procédures efficaces et des méthodes de communication modernes. De nombreux éléments qui peuvent être optimisés en les intégrant dans des solutions numériques.

1.2.3 Organisation fonctionnelle d'une Université

Pour une Université, il est nécessaire d'avoir une organisation structurée et optimisée afin de bien gérer les acteurs qui y sont présents. Nous pouvons distinguer trois (3) acteurs importants dans une Université :

- Etudiants : c'est une personne inscrite dans une formation de l'enseignement supérieur [1.05].

- Enseignants : c'est le personnel professionnel directement impliqué dans l'enseignement des étudiants, dont les enseignants de classe ; les enseignants de l'enseignement spécialisé, et d'autres enseignants qui travaillent avec des étudiants en classe entière, en petits groupes, ou individuellement à l'intérieur ou à l'extérieur d'une classe habituelle [1.06]
- Administration : les administrateurs de l'enseignement organisent et gèrent l'administration, les systèmes de soutien et les activités d'un établissement d'enseignement. Ils accomplissent une série de tâches administratif, de secrétariat, de soutien financier et par d'autres moyens afin de permettre l'exploitation efficace et rentable de l'Université. Ils peuvent contribuer au recrutement d'étudiants, aux relations d'anciens étudiants, au financement, au travail sur les commissions, y compris les conseils universitaires et à l'assurance de la qualité [1.07].

Additionnellement, ces acteurs utilisent des méthodes de communication et de diffusion d'informations variées et souvent éparpillées dans diverses plateformes tels que Facebook, groupes WhatsApp, email, ou autre plateforme externe. Ces acteurs entretiennent une relation étroite et importante pour le bon fonctionnement de l'Université dont chacun joue un rôle spécifique. Les moyens traditionnels de circulation de l'information.

- Communication entre l'administration et les enseignants et les étudiants : les communications officielles tels que les emplois du temps, calendrier universitaire, convocations aux examens, résultats, règlements, sont souvent transmis par :
    - Affichage physique sur les tableaux d'annonces
    - Rassemblement générale périodique
    - Distribuer par des étudiants dans les groupes de discussion sur les réseaux sociaux

Ces moyens sont très éparpillés ce qui favorisent le retard de la distribution de l'information.

- Communication entre les enseignants et les étudiants : les enseignants envoient les supports de cours, devoirs, consignes ou toute autre instruction :
    - Lors des cours présentiels en salle de classe ;
    - Transférer des cours et exercices sous fourme de document électronique en utilisant une clé USB aux étudiants,
    - Par Email personnel
    - Groupe informel sur Facebook ou WhatsApp.

Ces moyens variés rendent les retours (remarques, questions, explications) difficile et non centralisés.

- Communication entre les étudiants et l'administration et les enseignants : les étudiants posent leurs questions, remettent leurs travaux via :
    - En présentiel, lors des cours présentiels ou par rendez-vous ou dépôt de papier
    - Par courriers électroniques personnels ou institutionnels
    - Via les messageries sur les réseaux sociaux.

On peut constater que la circulation des informations importants dans l'université sont effectués par multiple moyens qui ne sont pas centralisés et le plus souvent informel ce qui peut avoir des conséquences indésirables comme les pertes ou retards d'information, confusion sur les informations officielles et ceux non vérifier et difficulté d'accès aux ressources pédagogiques.

1.2.4 Limites constatées

Plusieurs limitations structurelles bloquent les Universités dans leur effort d'organiser et d'optimiser leur communication, leur gestion pédagogique et l'accès à l'information. Ces contraintes se manifestent le plus dans les universités peu numérisés ou mal équipés sur le plan des outils ou plateformes collaboratifs.

Les points négatifs que nous avons constatés :

- Variations de canaux de communication : les échanges effectuer entre les étudiants, les enseignants et l'administration sont pour la plupart du temps faite par le biais de diverses plateformes de communication comme des courriers électroniques institutionnels ou personnels, groupe de messagerie WhatsApp, Messenger, par document imprimés ou affichés. Cette grande diversité provoque un manque de cohérence dans la diffusion et une absence de centralisation des échanges.
- Diffusion des informations : perte et lenteur de la diffusion des informations, la communication des informations importantes est transmise tardivement ou n'arrive pas à l'ensemble des destinataires.
- Accès aux ressource pédagogiques : les support de cours, exercices distribuer par les enseignants sont parfois partagé de manière informelle comme par exemple via des clés USB lors des cours présentiels, par courriers électroniques, par groupe de messagerie sur les réseaux sociaux, ce qui entraine de nombreux problèmes comme accès inégal des ressources entre les

étudiants, entraine une perte ou confusion sur les documents distribuer, empêche une cohésion éducatifs claire en cas d'absence ou de retard.

Ces contraintes accentuent l'intérêt d'une solution numérique centralisée. Une analyse des système existants nous aidera à évaluer les réponses déjà proposés et d'identifier leur insuffisance.

1.3 Analyse de l'existant

Les avancées technologique et la propagation de l'internet, ont provoqué des changements et évolution dans divers domaines dans le monde. Et le domaine de l'éducation n'est pas mise à l'écarts de ces transformations. Ces changements touchent notamment les établissements d'enseignements supérieure ou Université. Cela se manifeste par l'apparition des plateformes d'apprentissage universitaire aussi connu sous le terme de Learning Management System ou LMS. Ces plateformes permettent aux universités de diffuser leurs apprentissages à distance tant que l'internet est disponible. Elles facilitent aussi aux étudiants l'accès aux ressources éducatifs diffuser par l'université, comme les supports de cours, les notes d'évaluation et bien d'autres.

Nous appelons E-learning les modes d'apprentissage qui se repose sur les plateformes LMS. Le marché mondial de l'e-learning repose sur trois principes de base. Premièrement l'accessibilité : les contenus sont accessibles en ligne, ce qui élimine les barrières géographiques et temporelle. Deuxièmement, la flexibilité : les apprenants peuvent suivre les cours à leur propre rythme, ce qui est particulièrement utile pour les employés qui doivent jongler entre travail et formation. Enfin, l'interactivité : les plateformes d'e-learning intègrent des outils interactifs comme les forums de discussion, les vidéos interactives et les quiz, rendant l'apprentissage plus engageant et efficace [1.08]. Ce marché ne cesse de grandir et d'évoluer au fil des années. Cette croissance et évolution peut être attribuer aux avancés technologiques faites ces dernières années mais aussi par nécessité car le domaine de l'éducation sera le premier touché sur toute changements dans les différents domaines dans le monde.

1.3.1 Plateformes Universitaires existant

Actuellement, plusieurs de ces plateformes de type LMS sont déjà présente sur l'internet et coexiste entre eux ; on peut en cite quelque uns :

- Moodle : c'est une plateforme d'apprentissage destinée à fournir aux enseignants, administrateurs et apprenants un système unique robuste, sûr et intègre pour créer des environnements d'apprentissages personnalisés [1.09]

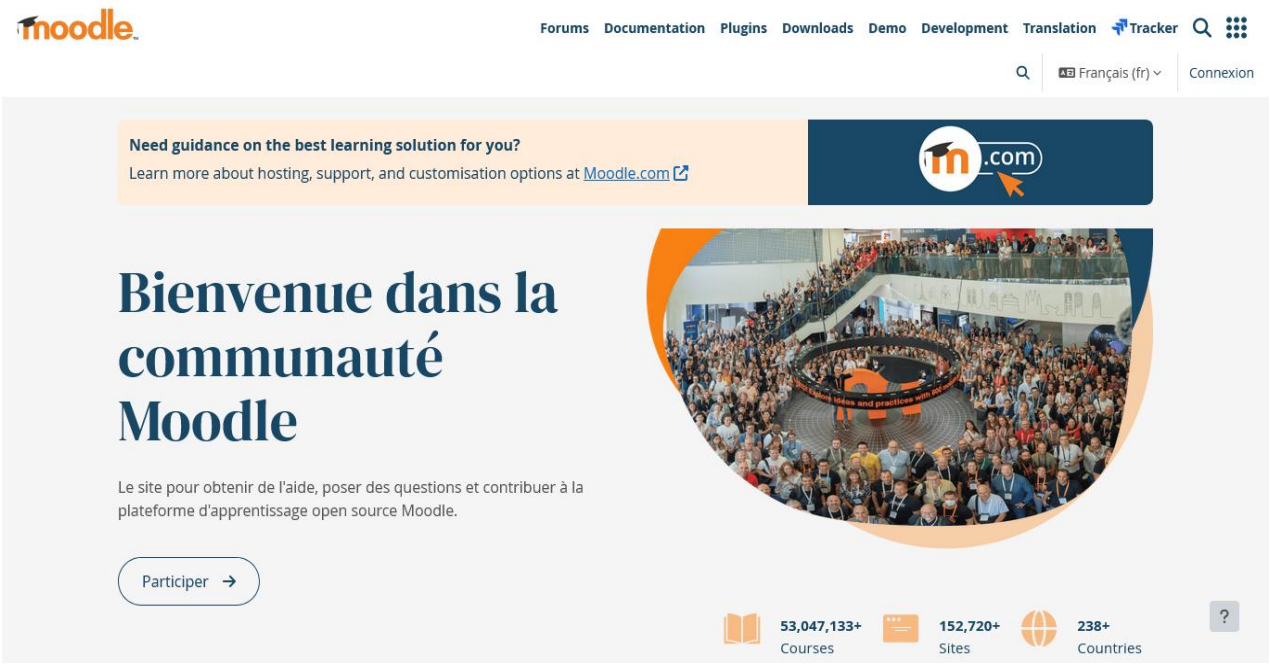

Figure 1.01 : *Plateforme web Moodle*

- Blackboard Learn : c'est un système de gestion de l'apprentissage avancé qui apporte tous les avantages de la technologie dans l'environnement de formation. Blackboard Learn est entièrement configurable en fonction des besoins ou des préférences des établissements. [1.10]

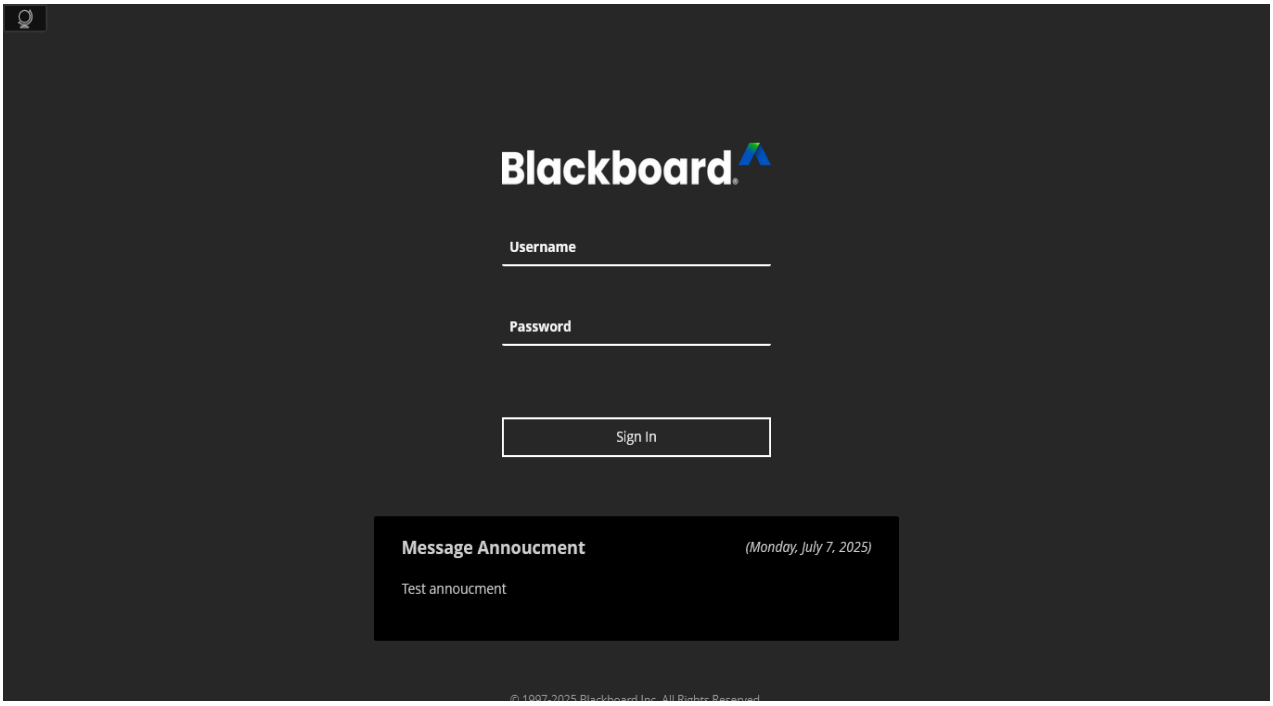

Figure 1.02 : *Plateforme web Blackboard Learn*

- Google Classroom : c'est une plateforme qui aide les enseignants à créer des expériences d'apprentissage stimulantes qu'ils peuvent personnaliser, gérer et évaluer. Intégré à Google Workspace for Education, il permet aux éducateurs de renforcer leur impact et de mieux préparer les étudiants à l'avenir [1.11]

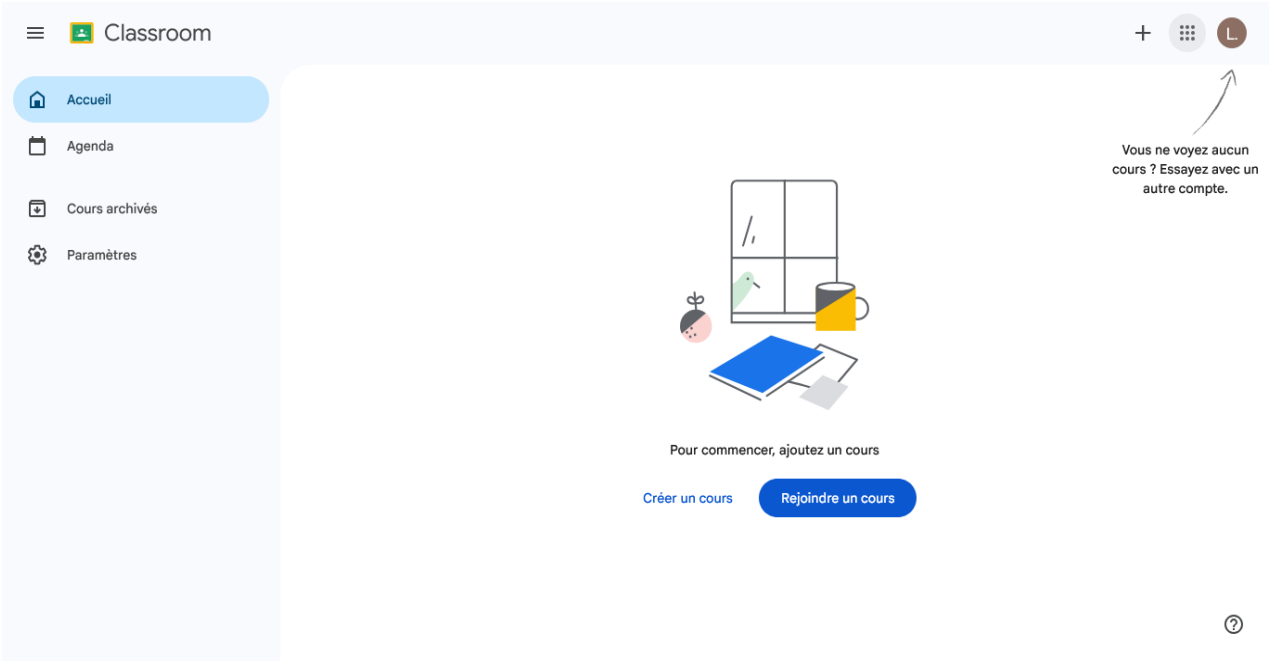

Figure 1.03 : *Plateforme web Google Classroom*

Mise à part les plateformes présente sur l'internet, il y a aussi des plateformes qui sont développés localement. Ces plateformes sont souvent créées pour une Université spécifique et sont parfois limitées aux fonctions de base telle que la gestion des notes, planification des cours et autres, ne sont pas maintenue régulièrement.

1.3.2 Analyse comparative des plateformes

Voici un tableau de comparaison entre les plateformes universitaires cité précédemment :

Tableau 1.01 : *Tableaux comparatifs de divers plateforme universitaire*

Critère	Moodle	Google Classroom
Accessibilité	Facile	Très facile
Coût	Gratuit	Gratuit avec G suite
Ergonomie	Moyen	Simple et intuitive
Fonctionnalité pédagogique	Très riche	Basiques
Personnalisation	Elevée	Faible
Maintenance	Nécessite une équipe IT	Gérée par Google

1.4 Description du projet

Notre projet s'intitule Développement et réalisation d'une plateforme Universitaire qui portera le nom « UniSphere ». Ce projet consistera à concevoir une plateforme web qui nous permettra de résoudre les points négatifs que nous avons constatés sur les Universités. Premièrement, centraliser la méthode de communication entre les étudiants, enseignants et l'administration. Ensuite, facilité l'accès aux ressources éducatifs aux étudiants et enfin optimiser la diffusion des informations partager par l'administration.

1.5 Objectifs du projet

Notre projet « UniSphere » a pour objectif de développer et réaliser une plateforme web qui nous permettra d'améliorer la gestion et la communication dans une université. Cette plateforme doit être capable de répondre aux besoins des acteurs principaux de l'université en modernisant la gestion de l'université et en optimisant la circulation des ressources et de l'information entre eux.

Pour plus de précision, les objectifs du projet sont les suivantes :

- Numériser et structurer le processus de gestion d'université, tels que :
    - L'organisation et la diffusion des ressource pédagogique
    - Gestion des classes et cours

    - Gestion des acteurs : étudiants et enseignants
- Optimiser la communication interne en mettant en place un système de messagerie sécurisé, permettant aux acteurs d'échanger plus facilement.

1.6 Contrainte du projet

Nous devons prendre en compte plusieurs contraintes lors du développement de la plateforme UniSphere, notamment du point de vue technique, organisationnels et humaines. La planification, la conception et même le déploiement de notre plateforme pourrait être influence par ces différentes contraintes.

1.6.1 Contraintes techniques

- Limitation des infrastructures réseau : certaine université ne dispose pas d'une connexion internet stable pour accéder à la plateforme
- Compatibilité sur divers supports : la plateforme doit être disponible sur différents appareils comme les téléphones et les ordinateurs
- Sécurité des données : la protection des données personnelles des utilisateurs est une contrainte importante et nous somme de respecter les normes de sécurité : authentification, confidentialité, sauvegarde

1.6.2 Contraintes organisationnelles

- Multitude de processus internes : chaque université utilise une méthode de gestion différentes ce qui complique l'implémentation d'un système d'unifié
- Absence ou manque de personnel qualifier : le personnel administratifs et les enseignants peuvent rencontrer des difficultés lors de l'utilisation de nouvel outil numérique sans formations adéquate.

1.6.3 Contraintes humaines

- Opposition au changement : l'introduction de la plateforme peut être vue comme une rupture avec les habitudes de travail, ce qui ralenti considérablement son utilisation
- Disponibilité des utilisateurs pour tester la plateforme : difficulté d'impliquer les utilisateurs finaux dans la validation du système.

1.7 Fonctionnalités du projet

Notre plateforme UniSphere offrira un ensemble de fonctionnalité qui répondra aux besoins d'une Université. Ces fonctionnalités ont été conçue dans le but d'améliorer la gestion des ressources pédagogiques et la communication dans l'Université.

1.7.1 Fonctionnalités générales

Les fonctionnalités générales sont accessibles pour tous les utilisateurs de la plateforme :

- Authentification : permet aux utilisateurs inscrit sur la plateforme d'accéder à un tableau de bords.
- Manipulation de profil : permet aux utilisateurs authentifier de manipuler leur profil, cependant ils ne peuvent pas supprimer leur compte.
- Envoie et téléchargement des fichiers : envoyer, télécharger des fichiers sur la plateforme

1.7.2 Fonctionnalités partagées par les « Etudiants » et les « Enseignants »

Les fonctionnalités citées ci-après ne seront disponible que pour les utilisateurs ayant le rôle d'étudiant ou d'enseignant :

- Accéder aux annonces : lecture des annonces postées par l'administration
- Accéder au système de messagerie : permet aux utilisateurs d'envoyer et de recevoir des messages

1.7.3 Fonctionnalités des « Etudiants »

Les fonctionnalités destinées aux étudiants :

- Accéder aux ressources pédagogiques : donne aux étudiants l'accès aux contenus selon leur classe
- Accéder à l'emploi du temps : affichage de l'emploi du temps selon la classe

1.7.4 Fonctionnalités des « Enseignants »

Les fonctionnalités destinées aux enseignants :

- Gestion des cours : ajouter, modifier, supprimer les contenus pour les cours attribués. Attribution des notes.
- Accès aux listes des matières et de classes : accédé à une liste des classes qui lui sont attribué.

1.7.5 Fonctionnalités des « Administrateurs »

Les fonctionnalités destinées aux administrateurs :

- Gestion des utilisateurs : permet aux administrateurs d'activé ou désactiver, modifier mais de manière restreinte les utilisateurs qui sont les étudiants et enseignants.
- Gestion des annonces : permet aux administrateurs de créer, modifier, supprimer des annonces.
- Gestion des classes : permet de créer, modifier, supprimer les classes.
- Gestion des cours : permet d'ajouter, modifier, supprimer des cours.

1.8 Avantages du projet

La création de la plateforme UniSphere présente de nombreux avantages pour une université. Avec l'implémentation de ces outil numériques dans le processus de l'Université, notre projet apporte des améliorations concrète et mesurable :

- Améliorations de la gestion de l'Université : avec la centralisation des données, toutes les informations sur les cours, emplois du temps sont réunir dans en une seule plateforme, ce qui diminue grandement la difficulté d'accès, la mise à jour et la supervision. Cela permet aussi de réduire les tâches répétitives à l'administration comme devoir attribuer des notes aux étudiants, faisant gagner du temps significatif aux personnels administratifs, aux enseignants et aux étudiants ;
- Centralisation de la communication : l'implémentation de la messagerie interne limite la dépendance aux outils externes comme le WhatsApp, Messenger, elle réduit aussi la perte et distorsion d'information et les confusions tout en offrant un moyen de communication sécuriser et fiable. Elle permet aussi de facilité le partages des informations comme les annonces administratives.
- Facilitation des accès aux ressources : avec la plateforme, les étudiants peuvent accéder plus facilement et à distance aux ressources éducatifs depuis n'importe quel appareil connecté à l'internet.

1.9 Conclusion

En définitive, dans le premier chapitre de cet ouvrage, on a pu présenter notre projet, une plateforme universitaire appelé « UniSphere » de manière générale. Nous avons aussi pu décrire son

objectif ainsi que les fonctionnalités qui seront présente mais aussi les contraintes auquel il sera soumis. Dans le prochain chapitre, nous allons effectuer l'analyse et la conception de notre plateforme.

CHAPITRE 2 ANALYSE ET CONCEPTION DU PROJET

2.1 Introduction

Pour chaque développement d'un projet, on doit toujours passer par la phase de conception. Cette phase nous permettra de délimiter le projet mais aussi de déterminer et analyser chaque fonctionnalité qui le composera. La phase de conception nous aidera à donner une fondation stable et solide à notre projet qui s'intitule « UniSphere » et nous permettra de minimiser les problèmes que nous pourrons rencontrés lors de sa réalisation. De ce fait dans ce chapitre, nous allons expliquer la méthode de conception que l'on va utiliser afin de concevoir le projet

2.2 Notation Unified Modeling Language ou UML

Le développement de notre plateforme UniSphere nécessite une conception minutieuse afin de minimiser les problèmes. Pour cela nous utiliserons l'Unified Modeling Language ou Langage de Modélisation Unifié durant la phase de conception.

2.2.1 Définition de la notation UML

L'UML (Unified Modeling Language) est un langage universel de modélisation de logiciel construite à l'aide d'objets et de notation, outil de communication visuelle [2.01]. La notation UML est fréquemment employer lors de la phase de conception d'un projet avec ces modélisations visuelle qui offrent différentes perspectives d'un système d'un projet.

2.2.2 Origine de l'UML

La notation UML n'est pas apparue sans préambule. Il est né dans les années 1990 de la fusion de trois méthodes de développement orienté objet : la méthode Booch de Grady Booch, OMT (Object Modeling Technique) de James Rumbaugh et OOSE (Object-Oriente Software Engineering) de Ivar Jacobson [2.02]. Plus précisément, les Three Amigos du génie logiciel, comme on les appelait alors, avait élaboré d'autres méthodologies. Ils se sont associés pour apporter plus de clarté aux programmeurs en créant de nouvelles normes. La collaboration entre Grady, Booch et Rumbaugh a renforcé les trois méthodes et a amélioré le produit final. Les efforts de ces penseurs ont abouti à la publication des documents UML 0.9 et 0.91 en 1996. Il est rapidement devenu évident que des sociétés comme Microsoft, Oracle et International Business Machines (IBM) voyant l'UML comme un élément critique pour leur développement futur. Elles ont donc mis en place des ressources, accompagnées en cela par de nombreuses autres sociétés et personnes, permettant de développer un langage de

modélisation complet. Les Three Amigos ont publié The Unified Modeling Language User Guide en 1999, qui fut suivi d'une mise à jour comportant des informations sur l'UML 2.0 en 2005 [2.03].

Les Trois Amigos et les méthodes qu'ils ont créées :

- Grady Booch depuis 1981, travaillait à la mise au point de formalismes graphiques pour représenter le développement de programme Orienté Objet (OO). Sa méthode Object Oriented Design (OOD) fut conçue à la demande du ministère de la Défense des États-Unis. Il avait mis au point des diagrammes de classe et d'objet, des diagrammes d'état-transition, et d'autres diagrammes de processus, pour mieux visualiser les programmes pendant leur exécution. Ces diagrammes devaient accompagner et améliorer l'organisation et la structure de programmes écrits en ADA et C++ [2.04].
- James Rumbaugh, alors à la Général Electric, fut l'auteur d'un formalisme graphique, précédant et en cela anticipant UML, appelé Object Modeling Technique (OMT). C'est d'OMT qu'UML s'est indéniablement le plus inspiré. Cette inspiration fut puisée dans les langages à objet mais également dans la modélisation conceptuelle appliquée à l'analyse et au stockage des données. La plupart des diagrammes importants faisant partie d'UML se retrouvaient déjà dans OMT, comme le diagramme de classe et autres diagrammes dynamiques et fonctionnels. C'est de fait Rumbaugh qui, chez IBM aujourd'hui, est le plus impliqué dans la maintenance et l'évolution de son bébé. Il accorde aussi énormément d'importance, au-delà de l'utilisation de ces diagrammes, au suivi d'une méthodologie décomposée en plusieurs phases, de l'analyse à l'implémentation, phases qui, au lieu d'être complètement séquentielles, se recouvrent en partie. Un développement logiciel devrait plutôt se dérouler comme une succession de petites itérations, de courte durée, toutes intégrant de l'analyse et du développement, mais le poids de l'analyse et du développement s'inversant graduellement vers la fin. On retrouve ces directives méthodologiques dans RUP (Rational Unified Process) et dans la programmation agile [2.04].
- Ivar Jacobson, auteur de la méthode Object Oriented Software Engineering (OOSE) est surtout connu pour avoir recentré l'analyse et le développement informatique sur les besoins humains et, en conséquence, pour l'apport dans UML de la notion de cas d'utilisation et du diagramme correspondant. Il s'est toujours intéressé à la nécessité première d'une bonne représentation du cahier des charges de l'application, ainsi que d'une bonne stratégie de développement, également basée sur une succession de phases courtes, dans lesquelles l'analyse et le développement varient en importance durant la progression du projet. C'est essentiellement à

lui que l'on doit RUP, la méthodologie de développement logiciel, qui accompagne souvent l'apprentissage d'UML [2.04].

La version de l'UML 1.0 a été adopté comme standard par l'Object Management Group (OMG) en janvier 1997. Des versions successives ont ensuite été validées, la dernière en date étant l'UML 2.5.1 [2.05].

L'Object Management Group (OMG) est un consortium de normes technologiques international, ouvert à tous et sans but lucratif créé en 1989. Les normes de l'OMG sont appliquées par les entreprises, les utilisateurs, le monde universitaire et les agences gouvernementales. Les groupes de travail de l'OMG développent des normes d'intégration aux entreprises pour un grand nombre de technologies et de secteurs industriels. Les normes de modélisation de l'OMG, dont l'UML et le Model Driven Architecture (MDA), permettent la conception, l'exécution et la maintenance de logiciels et d'autres processus d'une façon visuellement efficace [2.03].

L'OMG (Object Management Group) supervise la définition et la maintenance des spécifications UML. Cette surveillance donne aux ingénieurs et aux programmeurs la possibilité d'utiliser un langage à des fins multiples pendant toutes les phases du cycle de vie du logiciel, quelle que soit la taille du système concerné. L'OMG (Object Management Group) supervise la définition et la maintenance des spécifications UML. Cette surveillance donne aux ingénieurs et aux programmeurs la possibilité d'utiliser un langage à des fins multiples pendant toutes les phases du cycle de vie du logiciel, quelle que soit la taille du système concerné [2.03].

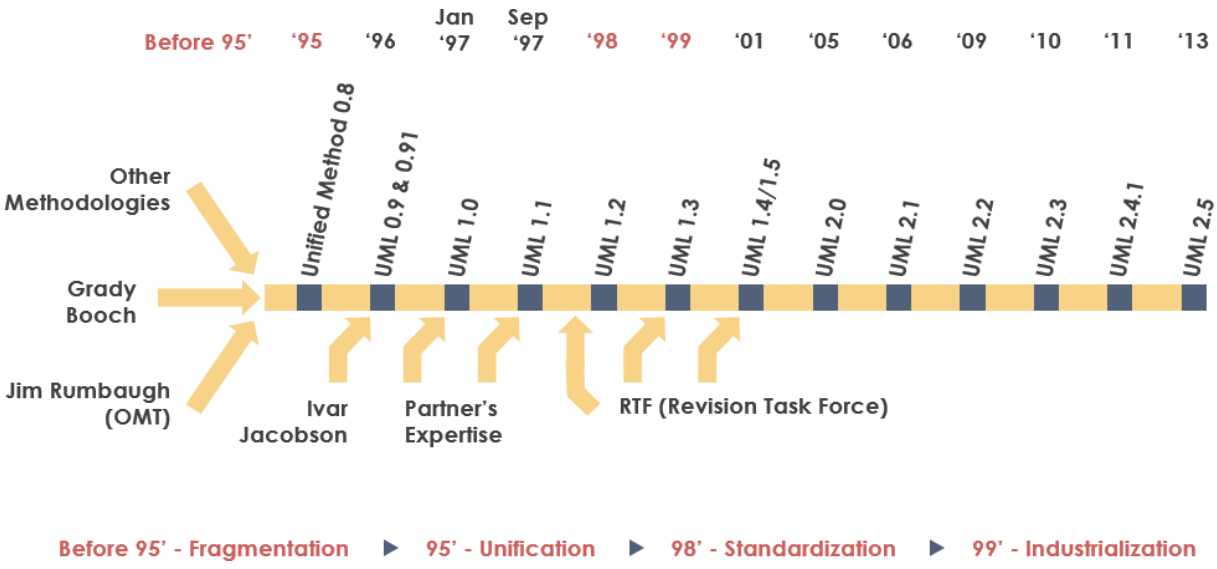

Figure 2.01 : *Ordre Chronologique de l'évolution de l'UML*

2.2.3 Objectifs de l'UML

L'Unified Modeling Language a pour objectif principale de présenter un langage de modélisation permettant de représenter de manière visuelle et simplifier la structure et le comportement de système complexe. Facilité la communication entre les différents acteurs du projet comme les développeurs, concepteurs, chef de projet et le client. L'UML permet de réduire l'imprécision et d'amélioré la précision des besoins du client.

L'Object Management Group définit les objectifs de l'UML comme suite :

- Fournit aux concepteurs de système, ingénieurs logiciels et développeurs de logiciels des outils pour l'analyse, la conception et la mise en œuvre de systèmes logiciels, ainsi que pour la modélisation de processus métier et d'autres processus similaires. [2.03]
- Faire progresser l'industrie en permettant l'interopérabilité des outils de modélisation visuelle orientés objet. Toutefois, pour permettre un échange significatif d'informations de modèles entre outils, il est nécessaire de trouver un accord sur la sémantique et la notation. [2.03]

L'UML réponds aux exigences suivantes :

- Fixer une définition formelle d'un métamodèle basé sur une norme Meta-Object Facility (MOF) commune qui spécifie la syntaxe abstraite de l'UML. La syntaxe abstraite définit l'ensemble des concepts de modélisation UML, leurs attributs et leurs relations, ainsi que les règles permettant d'associer ces concepts afin de créer des modèles UML partiels ou complets. [2.03]
- Fournir une explication détaillée de la sémantique de chaque concept de modélisation UML. La sémantique définit, d'une façon indépendante de la technologie, comment les concepts UML doivent être mis en œuvre par les ordinateurs. [2.03]
- Spécifier des éléments de notation lisibles par l'homme pour représenter chaque concept de modélisation UML, ainsi que les règles pour les combiner au sein d'une grande variété de diagrammes correspondant à différents aspects des systèmes modélisés. [2.03]
- Définir des moyens grâce auxquels les outils UML peuvent être mis en conformité avec cette spécification. Ceci est pris en charge (dans une spécification distincte) par une spécification XML des formats d'échange de modèles correspondants (XMI) qui doivent être réalisés par des outils conformes. [2.03]

2.2.4 Avantages de l'UML

L'UML offre de nombreux avantages lors de la phase de conception d'un quelconque projet informatique, qu'ils s'agissent d'améliorer le flux de travail d'un ingénieur logiciel ou de rationaliser la communication au sein d'une équipe. Passons en revue tous les avantages :

- Simplifier les idées et les systèmes complexes : les diagrammes UML simplifient visuellement les idées abstraites et les systèmes logiciels complexes, facilitant ainsi la collaboration entre les ingénieurs logiciels [2.07].
- Visualisation de code complexes : transformer des lignes de code compliquées en diagrammes visuels rend le développement de logiciels plus simple. Les diagrammes UML donnent une image claire de la manière dont les parties du code sont liées et fonctionnent ensemble, ce qui permet de gagner du temps et de réduire la confusion [2.07].
- Permet aux développeurs d'être sur la même longueur d'onde : UML est un langage visuel standard qui aide les membres de l'équipe à mieux communiquer entre eux, quels que soient leur langue et leur stade de développement. [2.07]
- Permet d'avoir une vue d'ensemble : au cours du processus de développement du logiciel, le fait de pouvoir se référer à un diagramme UML aide les développeurs à rester concentrés sur la conception globale et les objectifs du projet. [2.07].
- Ideal pour les explications non techniques : en plus d'aider les ingénieurs en informatiques, les diagrammes aident les propriétaires de produits, les gestionnaires et les parties prenantes à comprendre les processus et les fonctionnalités des logiciels. Ils comblent le fossé entre les membres techniques et non techniques de l'équipe, favorisant ainsi un meilleur travail d'équipe. [2.07].
- Améliore la collaboration entre les équipes : tous les programmeurs ne comprennent pas et ne se spécialisent pas dans le même type de code et de langage de programmation. En utilisant une notation standard, les diagrammes UML permettent à des programmeurs ayant des compétences différentes de travailler ensemble de manière efficace [2.07].

2.2.5 Les diagrammes de l'UML

Afin de pouvoir concevoir un projet à partir de l'UML, plusieurs diagrammes sont utilisés pour représenter visuellement la structure du système du projet, les communications entre les acteurs et le système et présenter de façon concises le projet au client.

L'UML possède 14 types de diagrammes qui sont repartis dans 2 catégories : le diagramme structurel ou statique et le diagramme comportementaux ou dynamique.

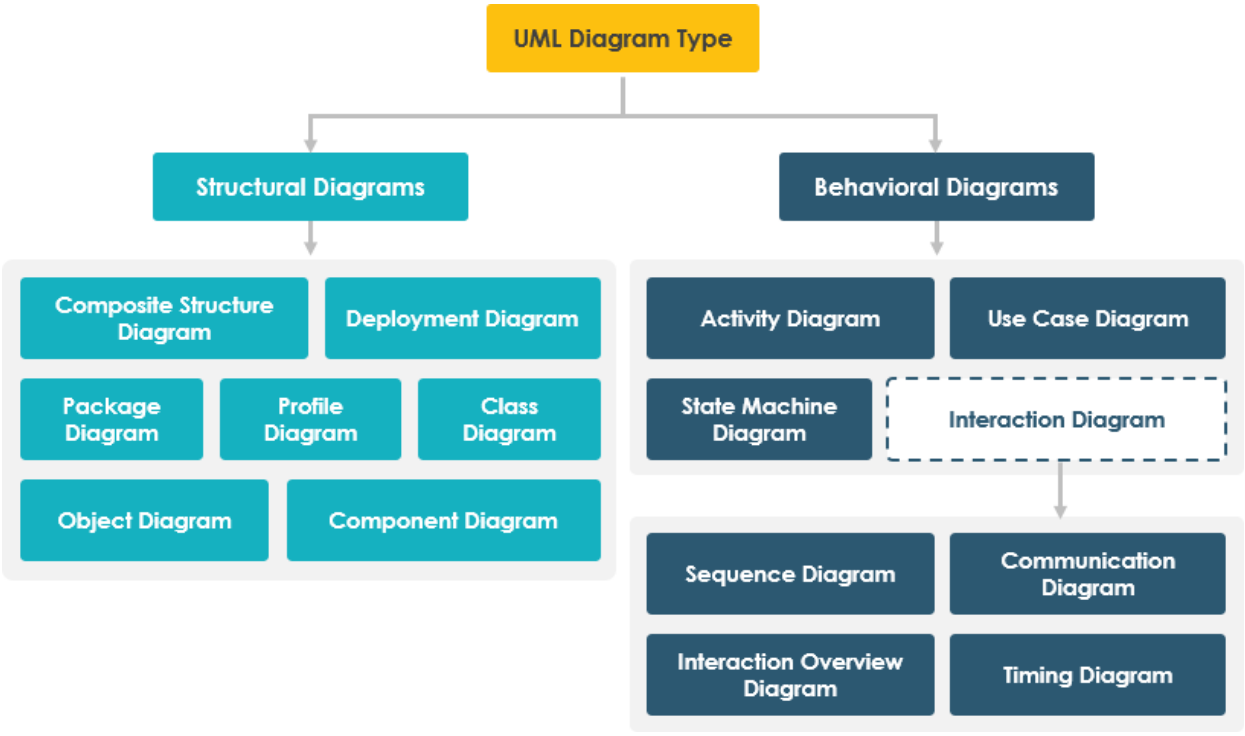

Figure 2.02 : *Hiérarchie Schématique de l'UML*

2.2.5.2 Les diagrammes structurels ou statiques

Les diagrammes structurels sont des représentations visuelles des différentes éléments qui constitue un système et leurs relations statique dans un projet. En d'autres termes, les diagrammes UML structurels, comme leur nom l'indique, illustrent la structure d'un système, notamment les classes, les objets, les paquetages, les composants, etc…, et les relations entre ces éléments [2.08].

- Diagramme de classe : c'est l'une des diagrammes le plus utilisé dans l'UML car ils exposent la structure statique d'un système, notamment les classes, leurs attributs et leurs comportements, ainsi que les liens entre chacune d'elles. Une classe est représentée par un rectangle contenant trois (3) compartiments emplis verticalement. Le compartiment supérieur contient le nom de la classe et est indispensable, tandis que les deux compartiments inférieurs fournissent des détails sur les attributs et les opérations ou comportements de la classe [2.08].

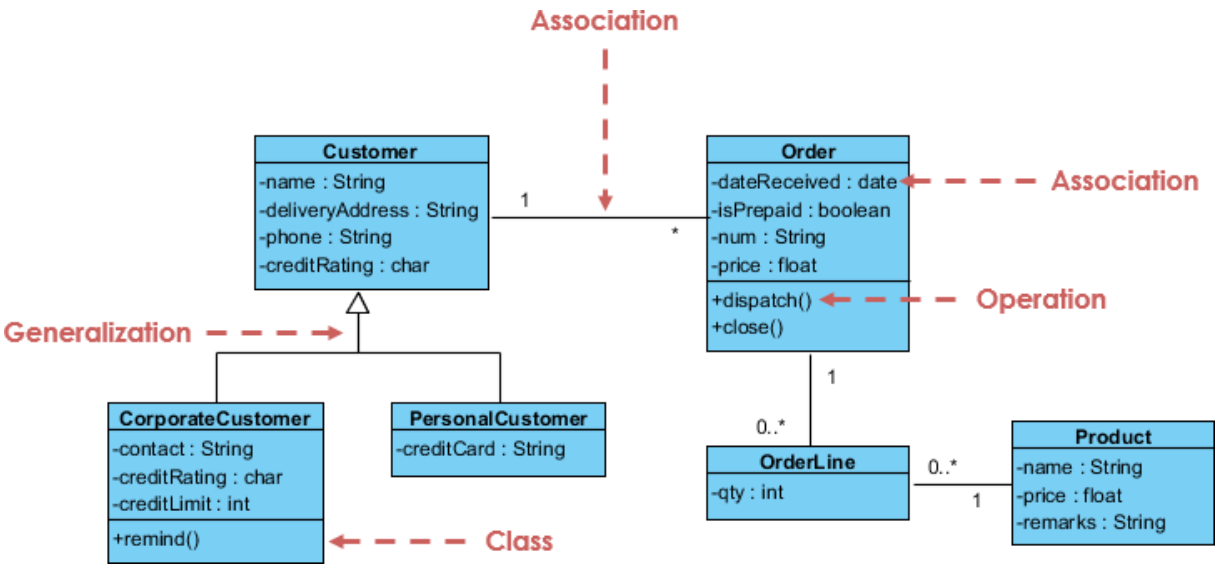

Figure 2.03 : *Exemple de diagramme de classe*

- Diagramme d'objet : les diagrammes d'objets présentent des exemples de structure de données à un moment spécifiques [2.08]. Ces diagrammes sont utilisés comme scenarios de test à un diagramme de classes d'une structure d'un système.

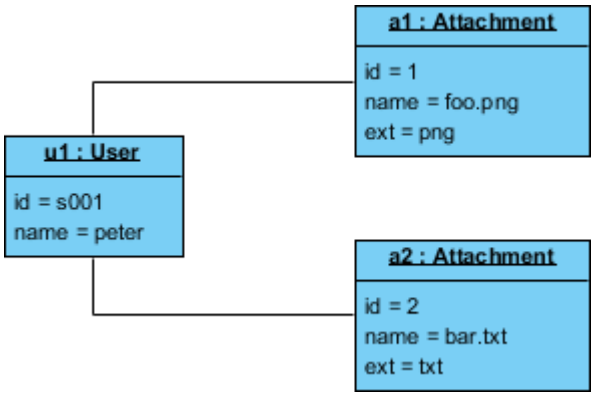

Figure 2.04 : *Exemple de diagramme d'objet*

- Diagramme de composant : c'est une version plus spécifique du diagramme de classes, et les mêmes règles de notation s'appliquent aux deux. Ce type de visuel découpe un système complexe en composants de taille réduite et représente les interactions entre ces derniers [2.08].

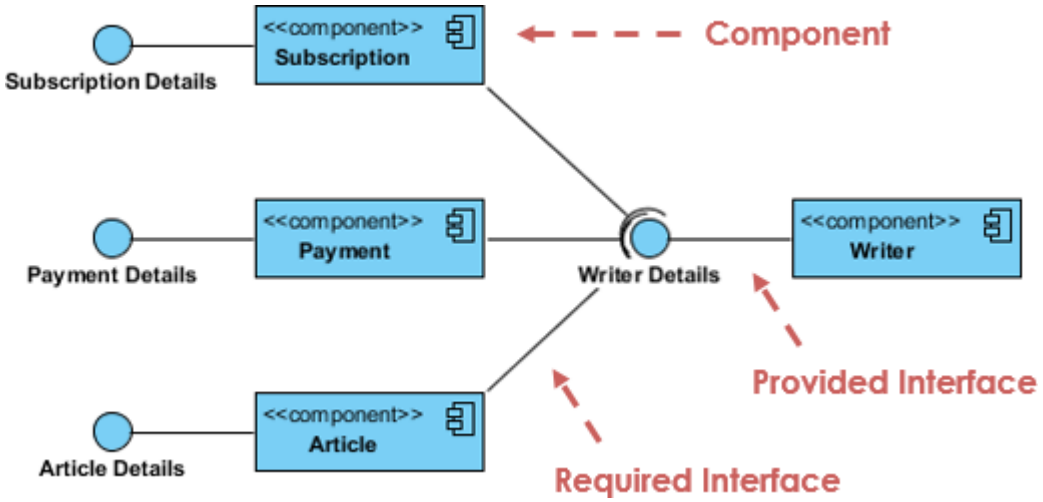

Figure 2.05 : *Exemple de diagramme de composant*

- Diagramme de déploiement : c'est la manière dont les logiciels sont déployés sur les composants matériels d'un système. Ces visuels sont particulièrement utiles pour les ingénieurs système et ils illustrent généralement les performances, l'évolutivité, la maintenabilité et la portabilité. Lorsque les composants matériels sont représentés les uns par rapport aux autres, il est plus facile de suivre l'ensemble des infrastructures informatiques et de s'assurer que tous les éléments sont pris en compte lors d'un déploiement [2.08].

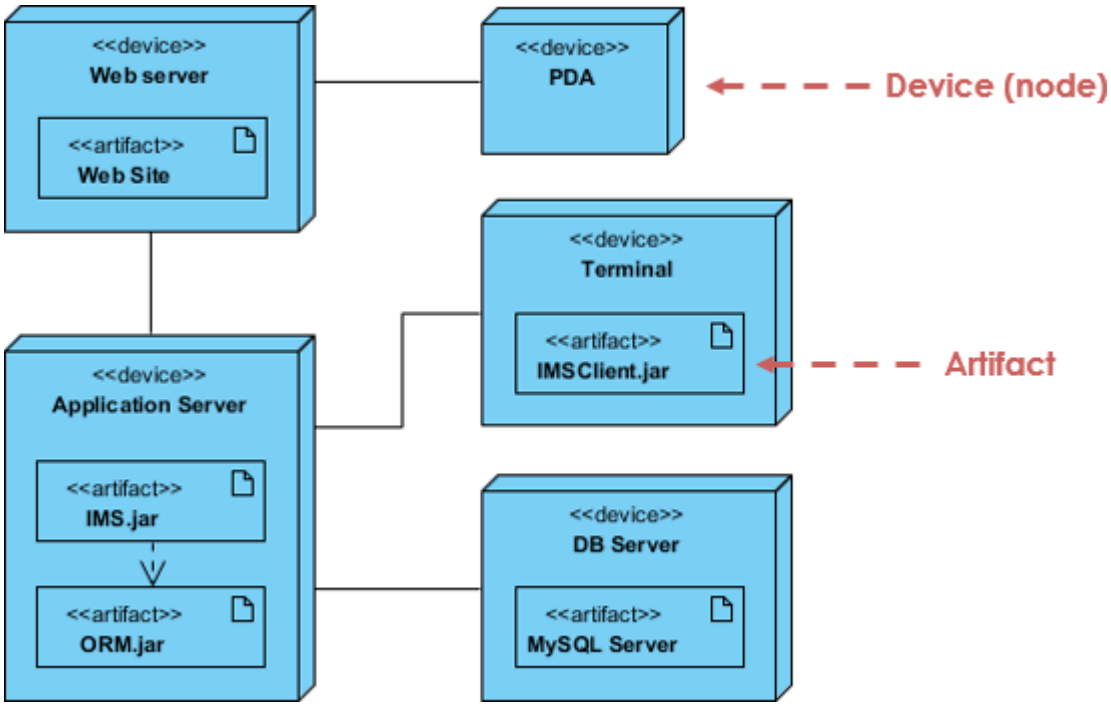

Figure 2.06 : *Exemple de diagramme de déploiement*

- Diagramme de paquetage : ces diagrammes sont utilisés pour illustrer les dépendances entre les différents paquetages d'un système. Ces derniers, représentés sous la forme d'un dossier de fichiers, organisent en groupes les éléments de modèle, tels que les cas d'utilisation ou les classes [2.08].

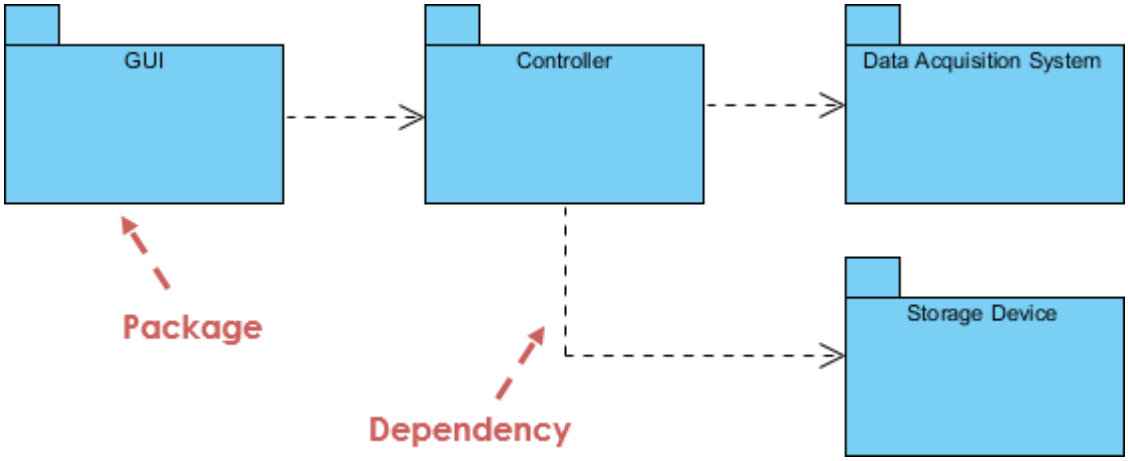

Figure 2.07 : *Exemple de diagramme de paquetage*

- Diagramme de structure composite : ce sont des plans de la structure interne d'un classifieur. Ils peuvent également être utilisés pour illustrer le comportement d'une collaboration ou les interactions du classifieur avec son environnement par le biais des ports. Ils permettent de représenter facilement les composants internes de tout type d'équipement pour mieux en comprendre le fonctionnement [2.08].

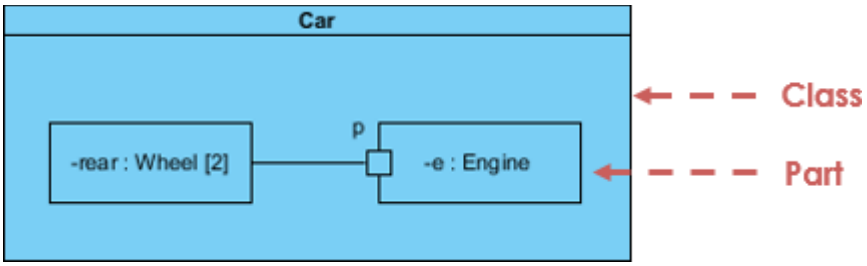

Figure 2.08 : *Exemple de diagramme de structure composite*

- Diagramme de profil : récemment ajouté à UML 2.0, les diagrammes de profil sont uniques et rarement utilisés dans les spécifications. Un diagramme de profil est mieux compris comme un mécanisme d'extensibilité pour personnaliser les modèles UML pour des domaines et des plates-formes spécifiques.

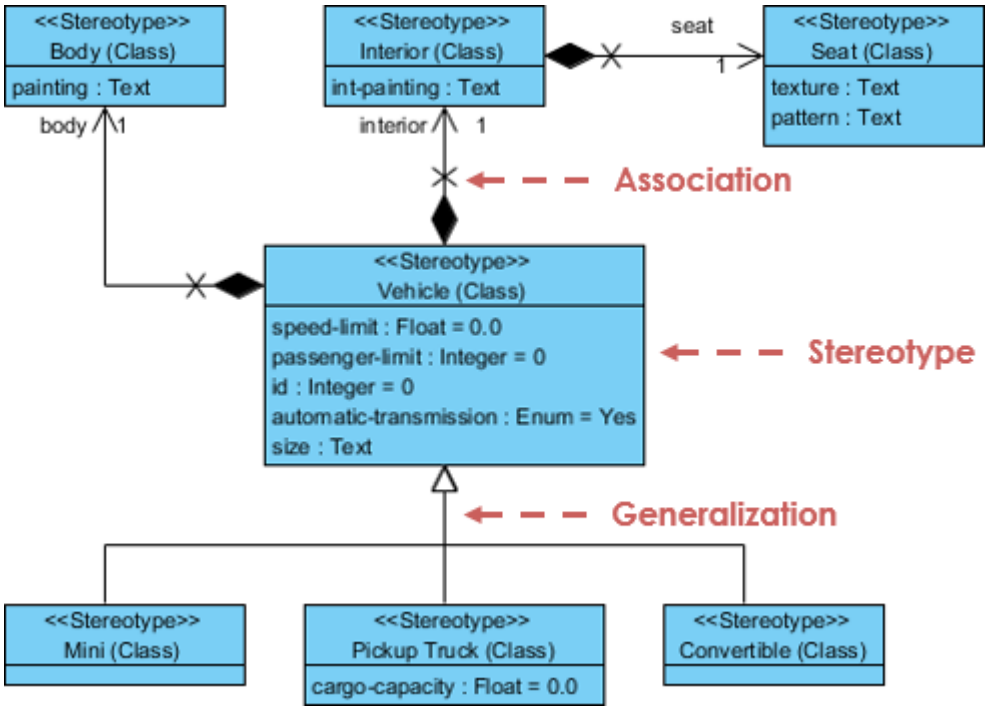

Figure 2.09 : *Exemple de diagramme de profil*

2.2.5.3 Les diagrammes comportementaux ou dynamiques

Les diagrammes comportementaux sont des représentations visuelles des comportement et communication entre les différents éléments et le système d'un projet. En d'autres termes, Ces diagrammes UML représentent la manière dont le système se comporte et interagit avec lui-même et avec les utilisateurs, les autres systèmes et les autres entités [2.08].

- Diagramme de cas d'utilisation : les diagrammes de cas d'utilisation modélisent la manière dont les utilisateurs, représentés sous formes de figurine appelées « acteurs », interagissent avec le système. Ce type de diagramme UML est une vue d'ensemble des relations entre les acteurs et les systèmes, ce qui en fait un excellent outil pour présenter un système à un public non technique [2.08].

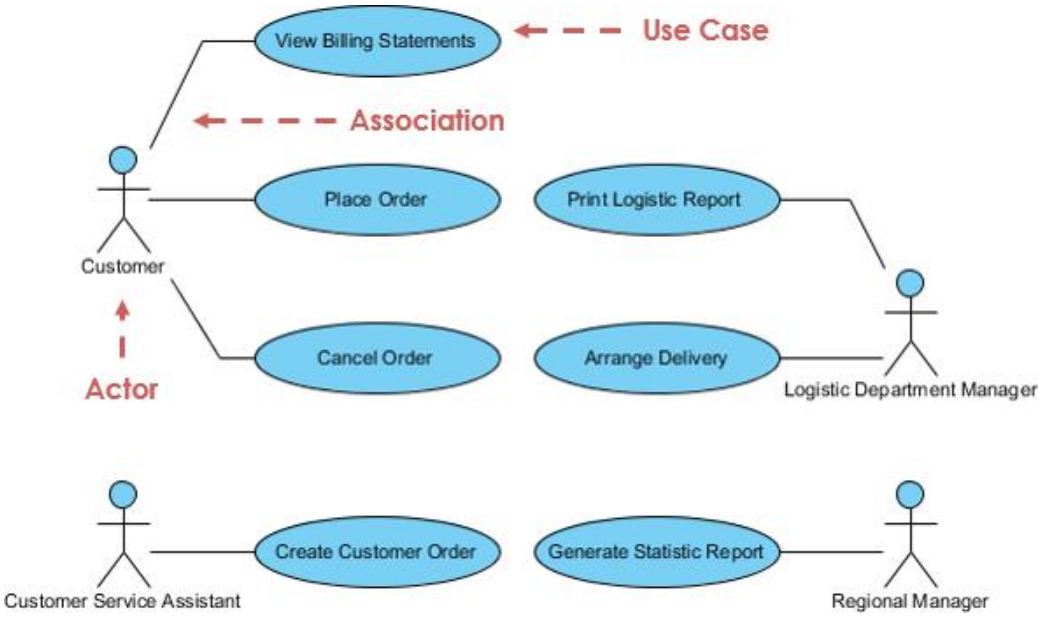

Figure 2.10 : *Exemple de diagramme de cas d'utilisation*

- Diagramme d'activité : ces diagrammes représentent les étapes réalisées dans un cas d'utilisation. Les activités peuvent être séquentielles, ramifiées ou simultanées. Ce type de diagramme UML est utilisé pour montrer le comportement dynamique d'un système, mais il peut également être utile dans la modélisation des processus métier [2.08].

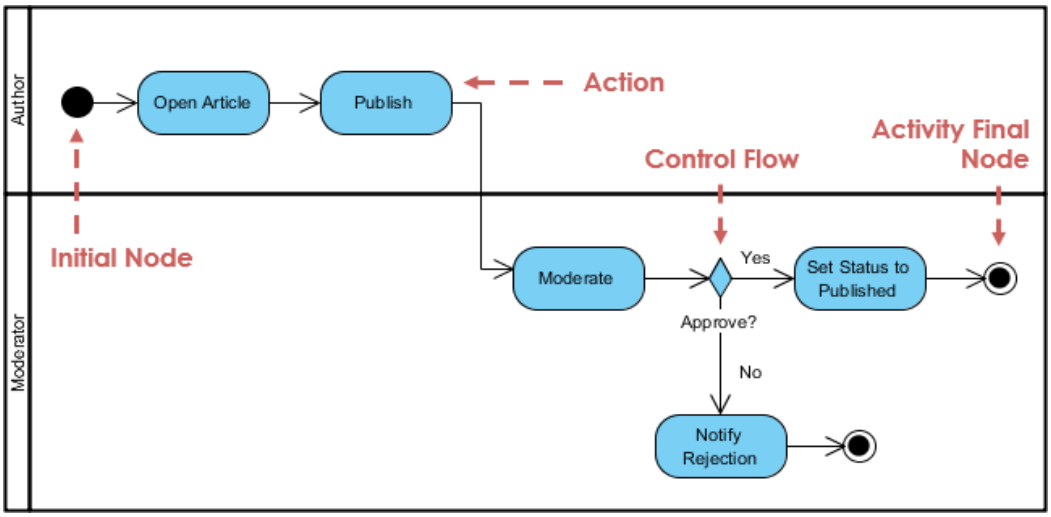

Figure 2.11 : *Exemple de diagramme d'activité*

- Diagramme d'état-transition : ils décrivent les états et les transitions. Les états correspond aux différentes combinaisons d'informations qu'un objet peut contenir, et ce type de diagramme UML permet de visualiser tous les états possibles et la manière dont l'objet passe d'un état à l'autre [2.08].

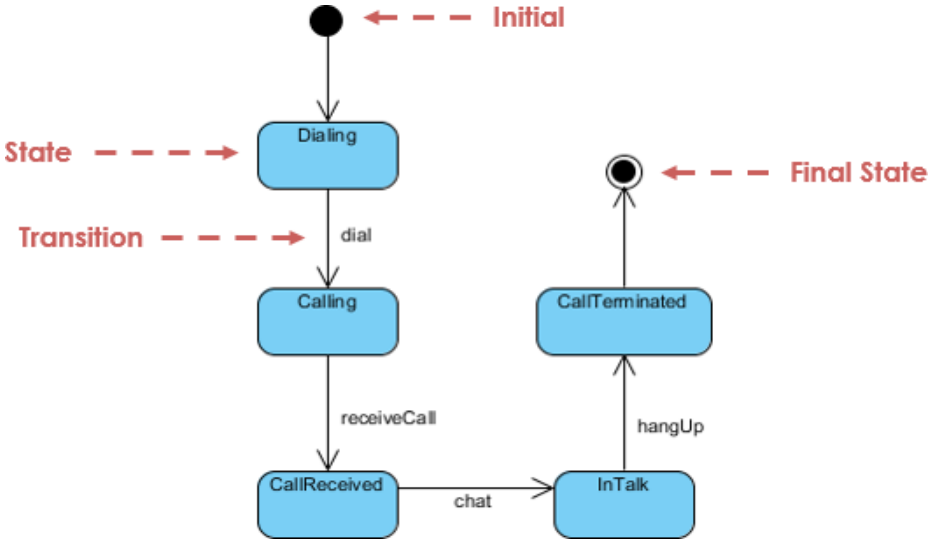

Figure 2.12 : *Exemple de diagramme d'état-transition*

- Diagramme de séquence : un diagramme parfois appelé diagramme d'événements ou scénario d'événements, montre l'ordre dans lequel les objets interagissent. Ils nous permettent ainsi de représenter visuellement des scénarios d'exécution simples [2.08].

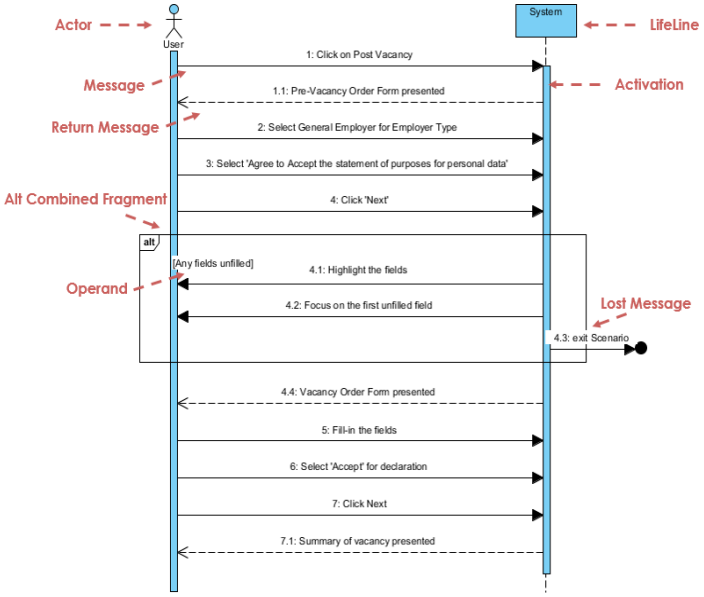

Figure 2.13 : *Exemple de diagramme de séquence*

- Diagramme d'interaction : ce diagramme donne un aperçu du flux de contrôle entre des nœuds en interaction. Ceux-ci incluent les nœuds initiaux, les nœuds finaux de flux, les nœuds finaux d'activité, les nœuds de décision, les nœuds de fusion, les nœuds de bifurcation et les nœuds de jonction [2.08].

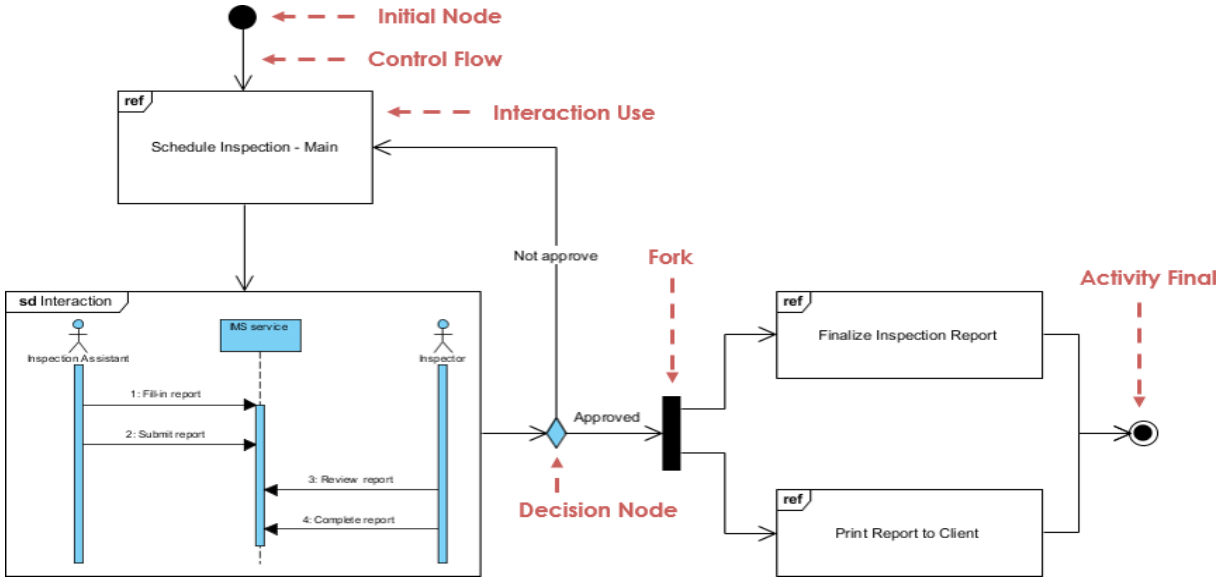

Figure 2.14 : *Exemple de diagramme d'interaction*

- Diagramme de communication : ce diagramme, autrefois appelé diagramme de collaboration, illustrent les liens entre les objets. Ils modélisent la manière dont ces derniers s'associent et se connectent par le biais de messages dans le cadre de la conception architecturale d'un système. Ils peuvent également représenter des scénarios alternatifs dans des cas d'utilisation ou des opérations qui nécessitent la collaboration de différents objets et interactions [2.08].

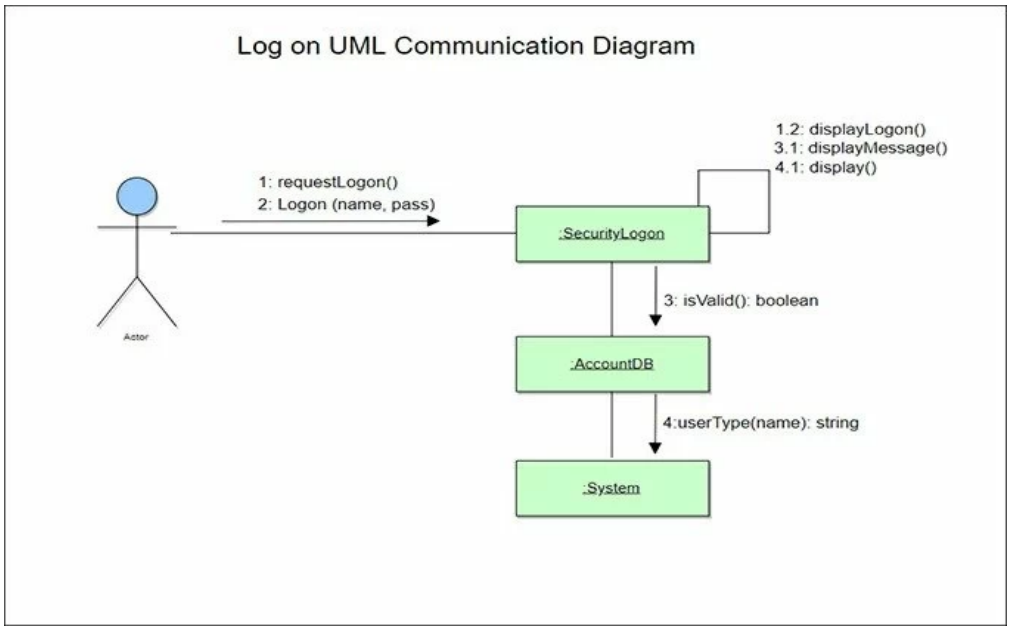

Figure 2.15 : *Exemple de diagramme de communication*

- Diagramme de temps : souvent décrit comme un diagramme de séquence inversé, un diagramme de temps montre comment les objets interagissent entre eux dans un laps de temps donné [2.08].

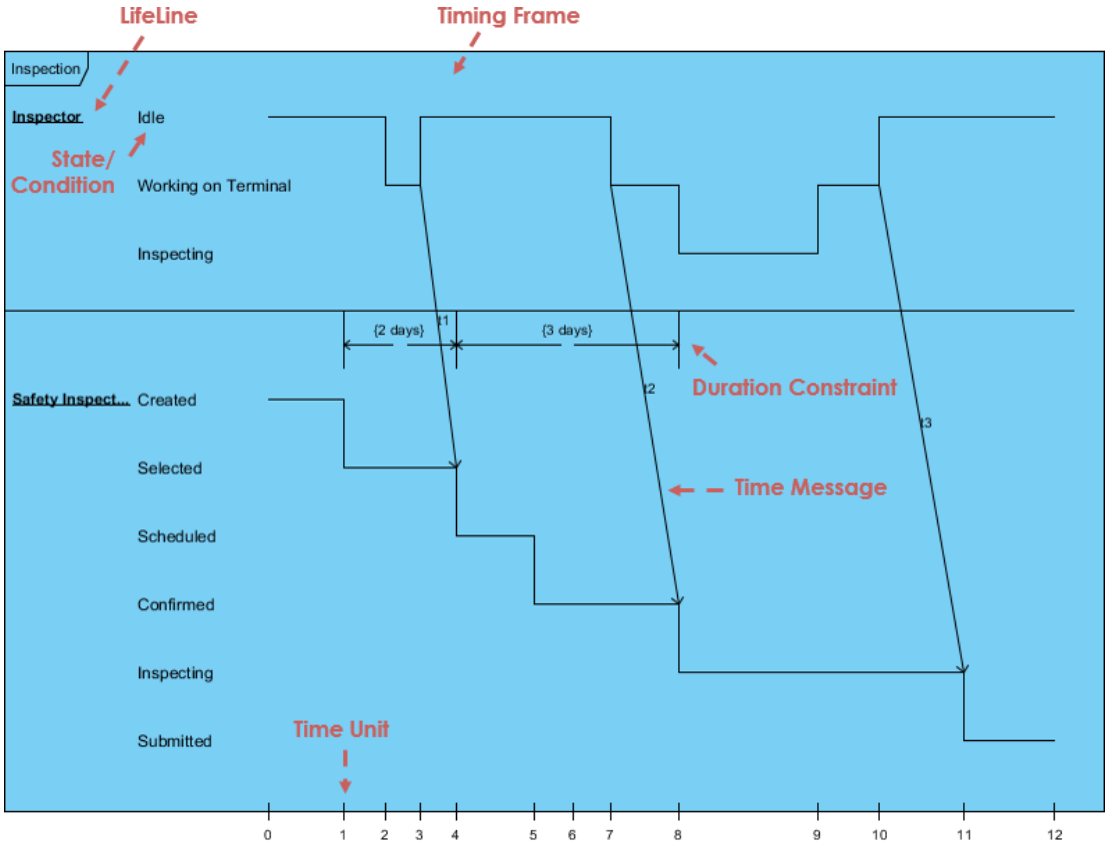

Figure 2.16 : *Exemple de diagramme de temps*

2.3 Conception du projet

Dans le contexte de notre projet, nous emploierons l'Unified Modeling Language lors de sa conception pour sa facilité d'utilisation, de compréhension, son adaptabilité et ces modélisation visuels.

Pour cette conception, nous emploierons les diagrammes suivants :

- Diagramme de cas d'utilisation
- Diagramme de séquence
- Diagramme d'activité
- Diagramme de classe

2.3.1 Diagrammes de cas d'utilisation

Ce diagramme est utilisé pour décrire les fonctionnalités d'un système d'un point de vue utilisateur sous la forme d'actions et de réaction.

Un diagramme de cas d'utilisation se compose des éléments suivants :

- Acteur : c'est le rôle joue par une entité extérieure au système, ayant une interaction directe avec le système, peut-être de type humain ou de type non humaine comme matériel ou logiciel.

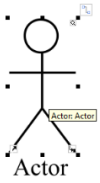

Figure 2.17 : *Acteur*

- Système : le système permet de délimiter le sujet de l'étude, et est constitué de cas d'utilisation

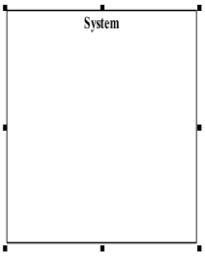

Figure 2.18 : *Système*

- Cas d'utilisation : c'est un moyen de représenter les différentes possibilités d'utiliser un système. Il exprime toujours une suite d'interaction entre un acteur et un système

Figure 2.19 : *Cas d'utilisation*

Notre diagramme de cas d'utilisation est représenté dans la figure :

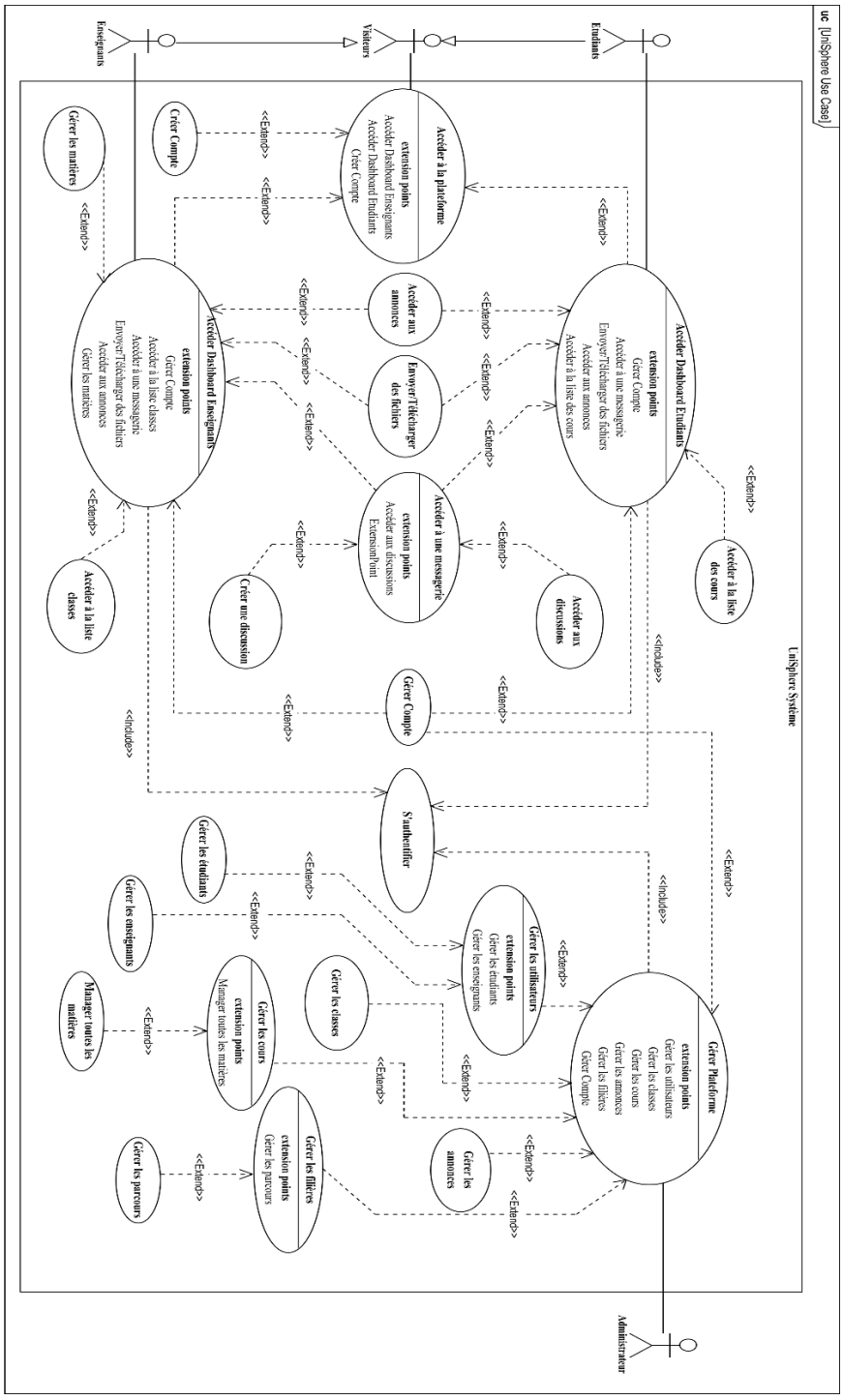

Figure 2.20 : *Diagramme de cas d'utilisation de UniSphere*

2.3.2 Description textuelle

Cette étape de la conception nous permettra de décrire avec plus de détails les cas d'utilisation présentent dans notre diagramme de cas d'utilisation. La description textuelle nous permettra de clarifier les démarches entamer afin d'exécuter une fonctionnalité.

Une description textuelle comprend les éléments suivants :

- Les préconditions : ce sont les conditions nécessaires pour déterminer si le cas d'utilisation peut fonctionner
- Les acteurs : nous permettent de déterminer quel acteur peut utiliser cette fonctionnalité
- Les scénarios : offre la description du fonctionnement du cas d'utilisation dans le cas d'un scénario le plus probable et le plus improbable.

2.3.2.1 Description textuelle de : « S'authentifier »

a- Information

- Nom : S'authentifier
- Acteur : Administrateurs, Etudiants, Enseignants
- Précondition : accès à l'internet, plateforme ouverte, possède déjà un compte

b- Scénario nominal

Tableau 2.01 : *S'authentifier (1)*

N°	Action
1	Accéder à la plateforme
2	La plateforme affiche la page d'accueil
3	L'acteur se dirige vers la page d'authentification
4	Affichage de la page d'authentification
5	L'acteur saisit les informations demandées
6	La plateforme vérifie et valide les informations entrées
7	La plateforme redirige l'acteur vers le tableau de bords qui lui correspond

c- Scénario alternatif

Tableau 2.02 : *S'authentifier (2)*

N°	Action
6	La plateforme invalide les informations saisit
6.1	La plateforme affiche un message d'erreur et attend que l'acteur saisisse à nouveau les informations demandées. Reprise au point 5

2.3.2.2 Description textuelle de : « Créer un Compte »

a- Information

- Nom : créer un compte
- Acteur : visiteur
- Précondition : accès à l'internet, plateforme active

b- Scénario nominal

Tableau 2.03 : *Créer un compte (1)*

N°	Action
1	Accéder à la plateforme
2	La plateforme affiche la page d'accueil
3	L'acteur se dirige vers la page d'inscription
4	La plateforme affiche la page d'inscription
5	L'acteur saisit les informations demandées
6	La plateforme vérifie et valide les informations entrées
7	La plateforme redirige l'acteur vers la page d'authentification

c- Scénario alternatif

Tableau 2.04 : *Créer un compte (2)*

N°	Action
6	La plateforme invalide les informations entrées
6.1	La plateforme affiche la page d'inscription avec un message d'erreur et attend que l'acteur saisisse à nouveau les informations demandées. Reprise au point 5

2.3.2.3 Description textuelle de : « Accéder à une messagerie »

a- Information

- Nom : Accéder à une messagerie
- Acteur : Etudiants, Enseignants
- Précondition : accès à l'internet, plateforme active, être connecter sur la plateforme

b- Scénario nominal

Tableau 2.05 : *Accéder à une messagerie (1)*

N°	Action
1	Inclusion du cas d'utilisation « S'authentifier »
2	La plateforme affiche le tableau de bords qui correspond à l'acteur
3	L'acteur se dirige vers la page de messagerie
4	La plateforme affiche la page de messagerie
5	L'acteur accède aux fonctionnalités de la messagerie : « Créer une discussion », « Accéder à une discussion »

2.3.2.4 Description textuelle de : « Envoyer/Télécharger des fichiers »

a- Information

- Nom : Envoyer/Télécharger des fichiers
- Acteur : Etudiants, Enseignants, Administrateurs
- Précondition : accès à l'internet, plateforme active, être connecter sur la plateforme

b- Scénario nominal

Tableau 2.06 : *Envoyer/Télécharger des fichiers (1)*

N°	Action
1	Inclusion du cas d'utilisation « S'authentifier »
2	Inclusion du cas d'utilisation « Accéder à une messagerie »
3	Inclusion du cas d'utilisation « Accéder à une discussion »
4	La plateforme affiche une discussion privée ou groupée
5	L'acteur appuie sur le bouton envoyer fichier (image ou document)
6	La plateforme affiche un formulaire
7	L'acteur choisi un fichier et envoie sur la plateforme
8	La plateforme vérifie et valide le fichier
9	La plateforme affiche le fichier dans la discussion privée ou groupée

c- Scénario alternatif

Tableau 2.07 : *Envoyer/Télécharger des fichiers (2)*

N°	Action
8	La plateforme invalide le fichier
8.1	La plateforme affiche un message d'erreur et attend que l'acteur fasse une action. Reprise au point 6

2.3.2.5 Description textuelle de : « Créer des discussion »

a- Information

- Nom : Créer des discussions
- Acteur : Etudiants, Enseignants
- Précondition : accès à l'internet, plateforme active, être connecter sur la plateforme

b- Scénario nominal

Tableau 2.08 : *Créer des discussions (1)*

N°	Action
1	Inclusion du cas d'utilisation « S'authentifier »
2	Inclusion du cas d'utilisation « Accéder à une messagerie »
3	La plateforme affiche la page de messagerie
4	L'acteur appuie sur le bouton de création de discussion
5	La plateforme affiche une liste des utilisateurs (étudiants et enseignants)
6	L'acteur sélectionne un ou plusieurs utilisateurs pour participer à la discussion
7	La plateforme crée une discussion dans avec les membres sélectionnés par l'acteur

c- Scénario alternatif

Tableau 2.09 : *Créer des discussion (2)*

N°	Action
6	L'acteur ne sélectionne aucun utilisateur pour la discussion
6.1	La plateforme ne crée pas de discussion. Reprise au point 3

2.3.2.6 Description textuelle de : « Gérer les utilisateurs »

a- Information

- Nom : Gérer les utilisateurs
- Acteur : Administrateurs
- Précondition : accès à l'internet, plateforme active, être connecter sur la plateforme

b- Scénario nominal

Tableau 2.10 : *Gérer les utilisateurs (1)*

N°	Action
1	Inclusion du cas d'utilisation « S'authentifier »
2	La plateforme affiche le tableau de bords de l'administrateur
3	L'acteur se dirige vers la page de gestion d'utilisateur
4	La plateforme affiche la page de gestion d'utilisateur : - Gestion d'étudiants - Gestion d'enseignants
5	L'acteur appuie sur le bouton de gestion d'étudiants
6	La plateforme affiche la page de gestion d'étudiants

c- Scénario alternatif

Tableau 2.11 : *Gérer les utilisateurs (2)*

N°	Action
5	L'acteur appuie sur le bouton de gestion d'enseignants
5.1	La plateforme affiche la page de gestion d'enseignants

2.3.2.7 Description textuelle de : « Gérer les cours »

a- Information

- Nom : Gérer les classes
- Acteur : Administrateur
- Précondition : accès à l'internet, plateforme active, être connecter sur la plateforme

b- Scénario nominal

Tableau 2.12 : *Gérer les classes (1)*

N°	Action
1	Inclusion du cas d'utilisation « S'authentifier »
2	La plateforme affiche le tableau de bords de l'administrateur
3	L'acteur se dirige vers la page de gestion de classes
4	La plateforme affiche une liste des classes
5	L'acteur sélectionne une classe
6	La plateforme affiche les informations de la classe

2.3.3 Diagramme de séquence

Ce diagramme représente les interactions entre objets selon un point de vue temporel, et on y met l'accent sur la chronologie des envois de messages, montre les réactions du système aux actions des utilisateurs, la création et la manipulation des objets. Il document un ou plusieurs scenarios d'un cas d'utilisation.

Un diagramme de séquence se compose :

- Les acteurs : permet de de préciser les utilisateurs qui effectue les actions durant le scénario
- Les objets : qui sont représenté par un rectangle contenant le nom de l'objet
- Ligne de vie : c'est une ligne verticale pointillé dirigée vers le bas à partir de chaque objet, elle symbolise une durée qui dépend du scénario et du comportement modélisée. Elle représente aussi l'ensemble des opérations exécutées par un objet.
- Barre d'activation : c'est un rectangle qui remplace la ligne de vie
- Message : c'est généralement un appel, un signal ou une réponse qui est représenter par des flèches horizontales relient la ligne de vie de l'objet émetteur à la ligne de vie de l'objet récepteur.

2.3.3.1 Diagramme de séquence de : « S'authentifier »

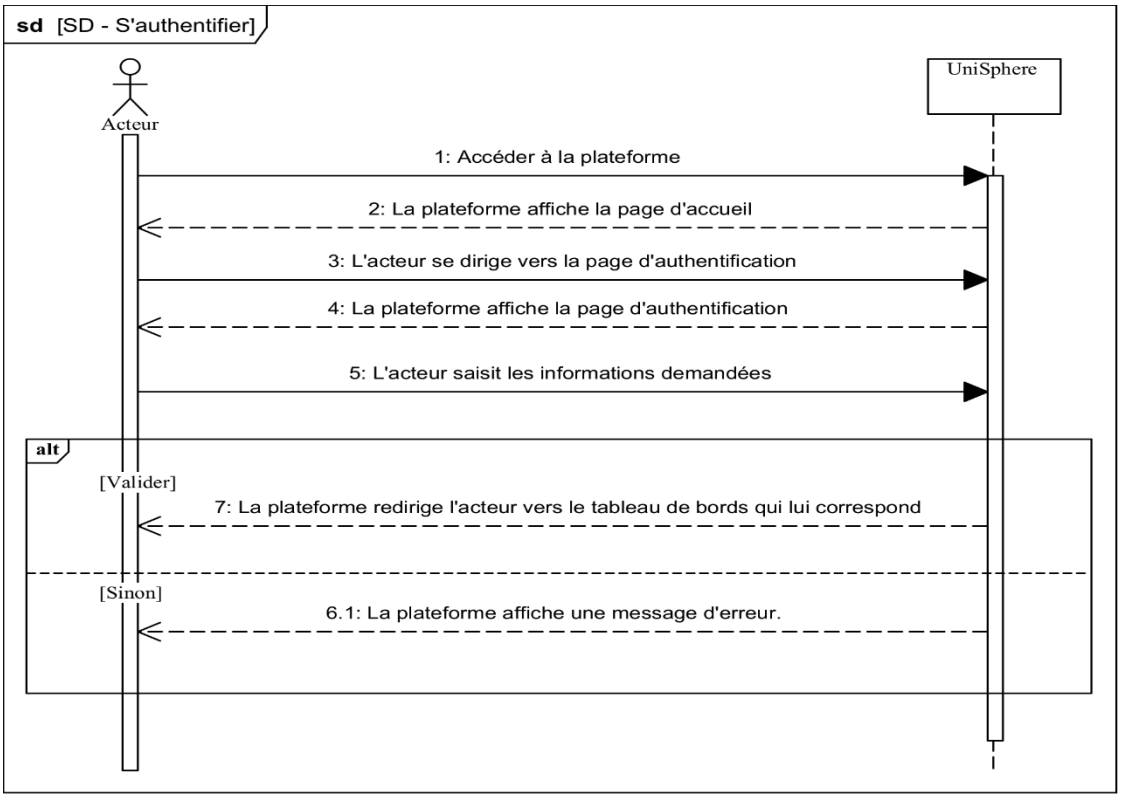

Figure 2.21 : *Diagramme de Séquence : « S'authentifier »*

2.3.3.2 Diagramme de séquence de : « Créer un compte »

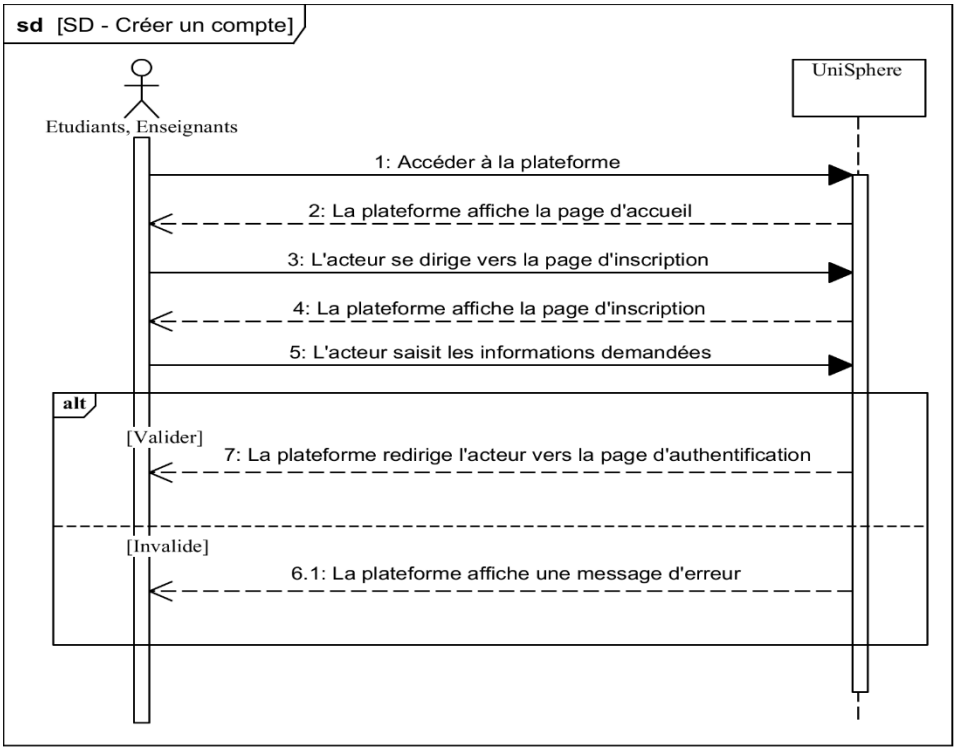

Figure 2.22 : *Diagramme de Séquence : Créer un compte*

2.3.3.3 Diagramme de séquence de : « Accéder à une messagerie »

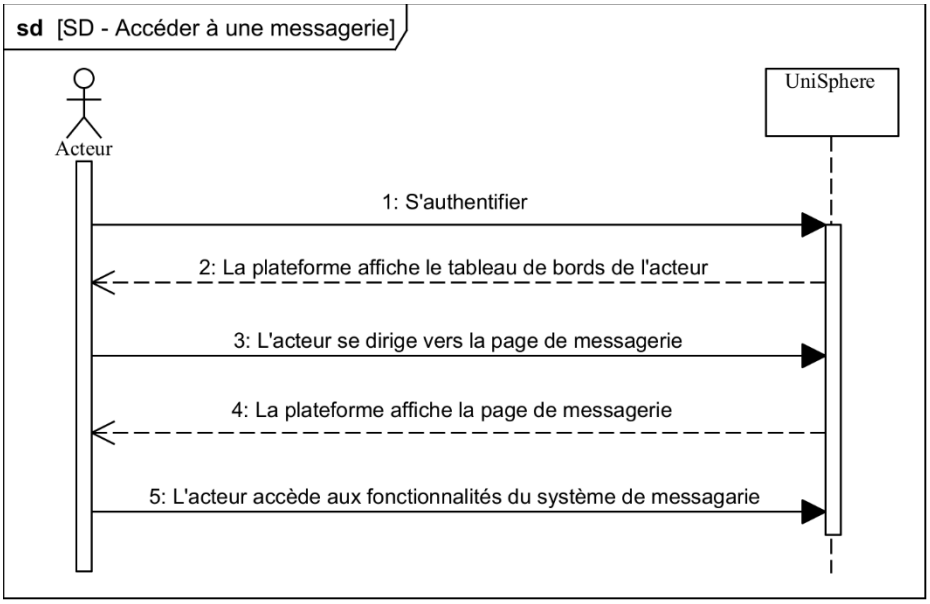

Figure 2.23 : *Diagramme de Séquence : Accéder à une messagerie*

2.3.3.4 Diagramme de séquence de : « Envoyer/Télécharger des fichier »

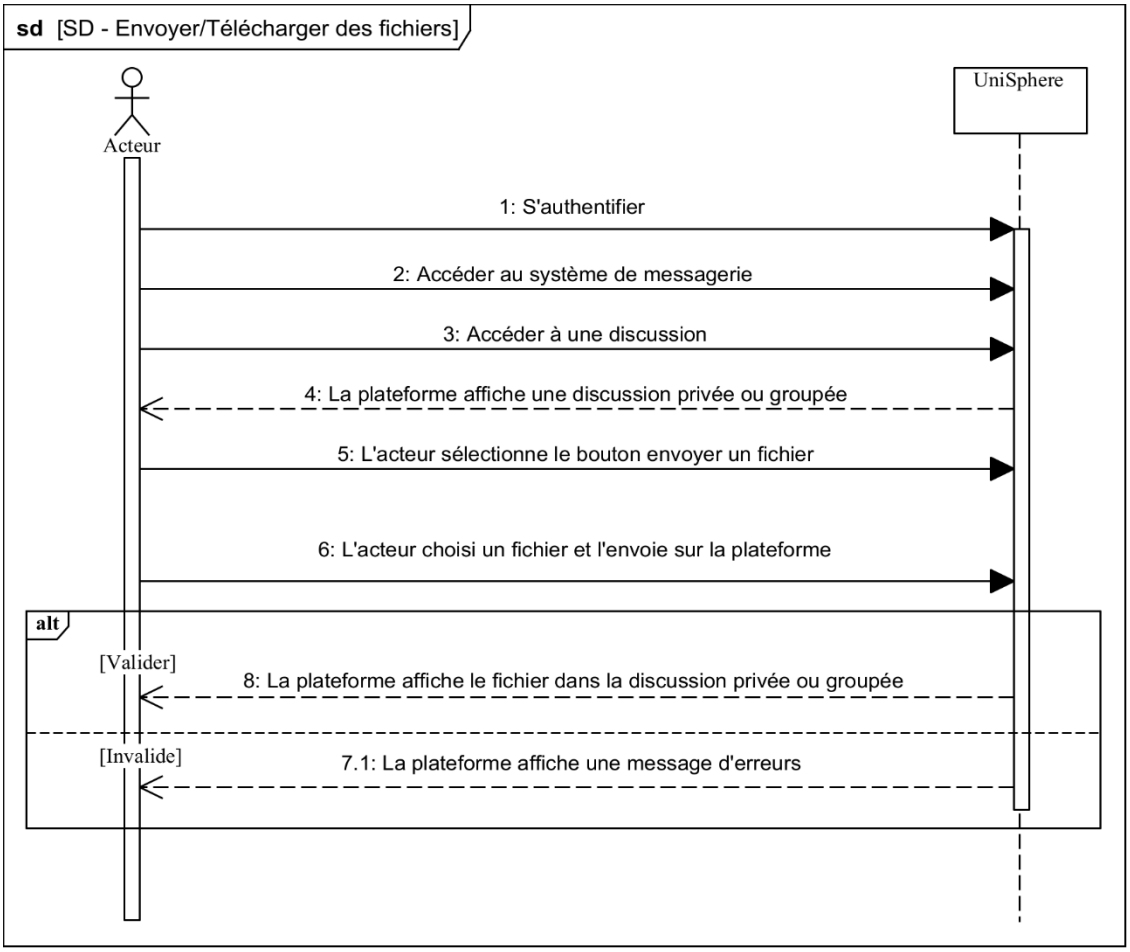

Figure 2.24 : *Diagramme de Séquence : Envoyer/Télécharger des fichiers*

2.3.3.5 Diagramme de séquence de : « Créer une discussion »

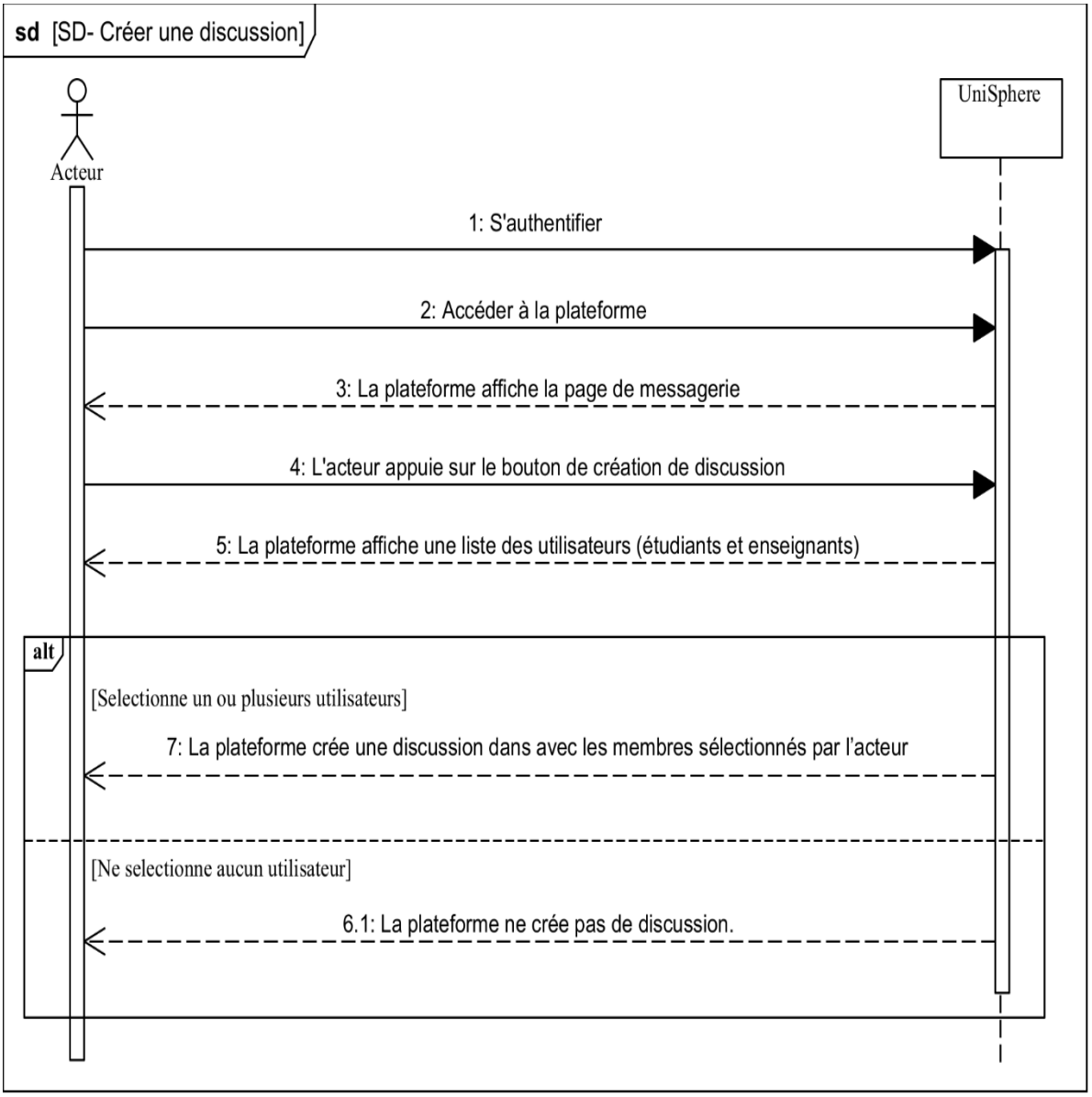

Figure 2.25 : *Diagramme de Séquence : Créer une discussion*

2.3.3.6 Diagramme de séquence de : « Gérer les utilisateurs »

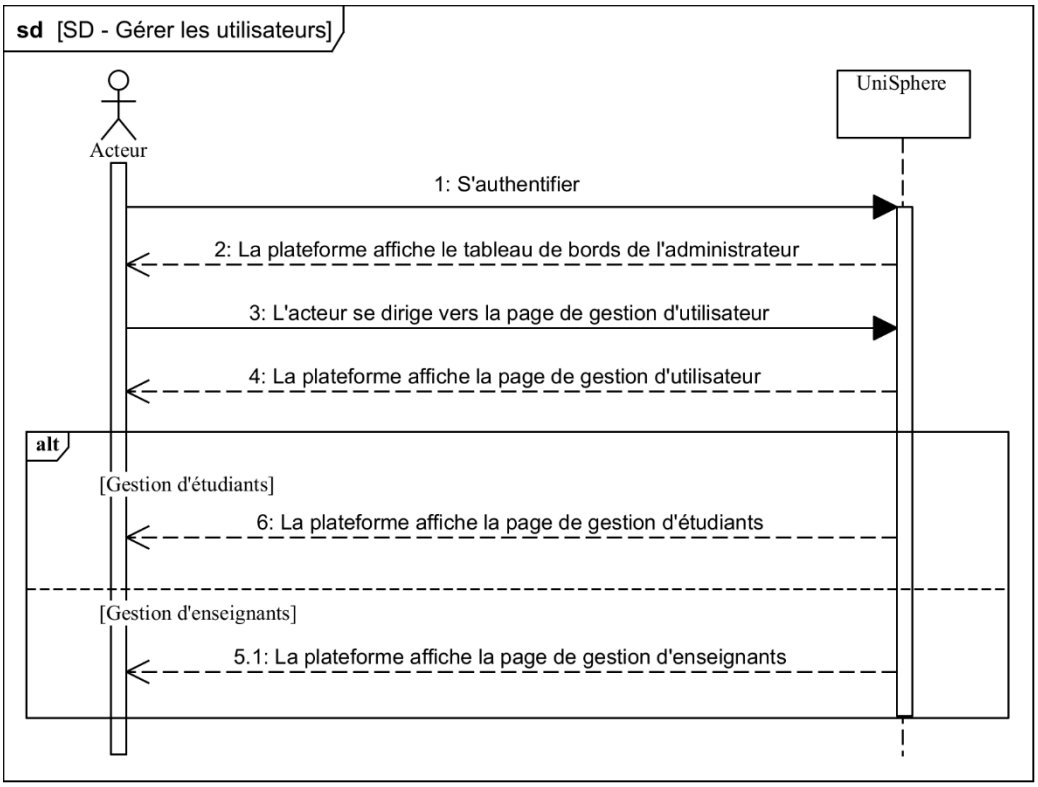

Figure 2.26 : *Diagramme de Séquence : Gérer les utilisateurs*

2.3.3.7 Diagramme de séquence de : « Gérer les classes »

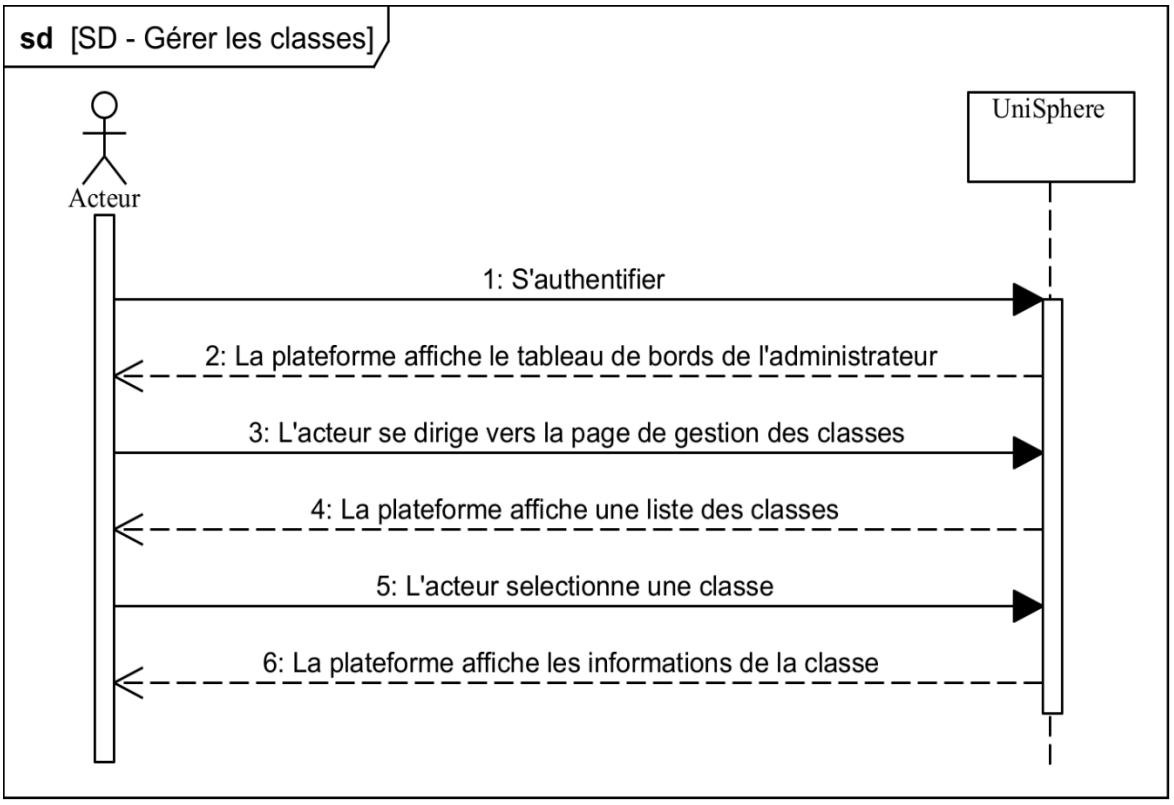

Figure 2.27 : *Diagramme de Séquence : Gérer les classes*

2.3.4 Diagramme d'activité

C'est une variante du diagramme d'état transition, permettant de représenter le comportement interne d'un cas d'utilisation ou processus et prend en charge les comportements parallèles.

Un diagramme d'activité comprend les éléments suivants :

- Action : c'est une étape dans l'activité où les acteurs ou logiciel exécutent une tâche donnée ;
- Etat initial : c'est un élément qui marque le début de l'activité ;
- Etat final : c'est un élément qui marque la fin de toute activité ;
- Activité : représentent le comportement d'un objet ;
- Nœuds de test décision : permet de faire un choix entre plusieurs flots sortant en fonction des conditions de garde de chaque flot. Il ne possède qu'une seule entrée et peut utiliser deux flots de sortie, le premier correspondant à la condition vérifié et l'autre traitant le cas contraire.
- Nœuds de bifurcation ou fourche : permet la création de plusieurs flots concurrents en sortie de la barre de synchronisation à partir d'un flot unique entrant ;

Pour notre plateforme, les diagrammes d'activités sont les suivants :

2.3.4.1 Diagramme d'activité de : « S'authentifier »

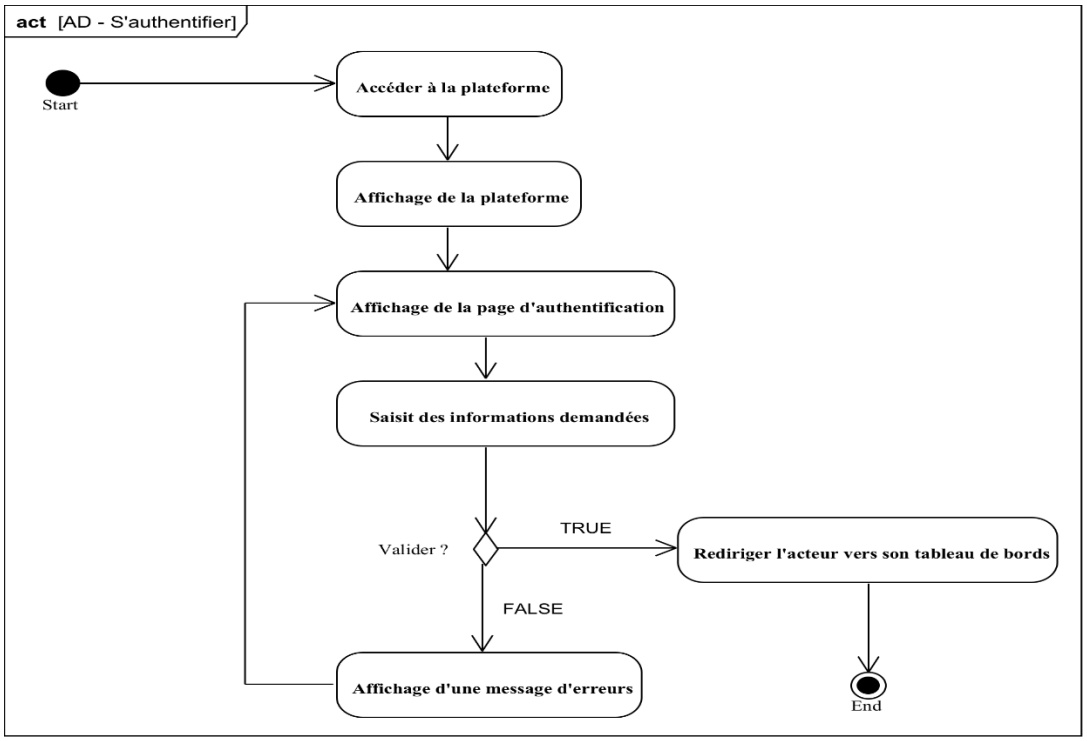

Figure 2.28 : *Diagramme d'activité :S'authentifier*

2.3.4.2 Diagramme d'activité de : « Créer un compte »

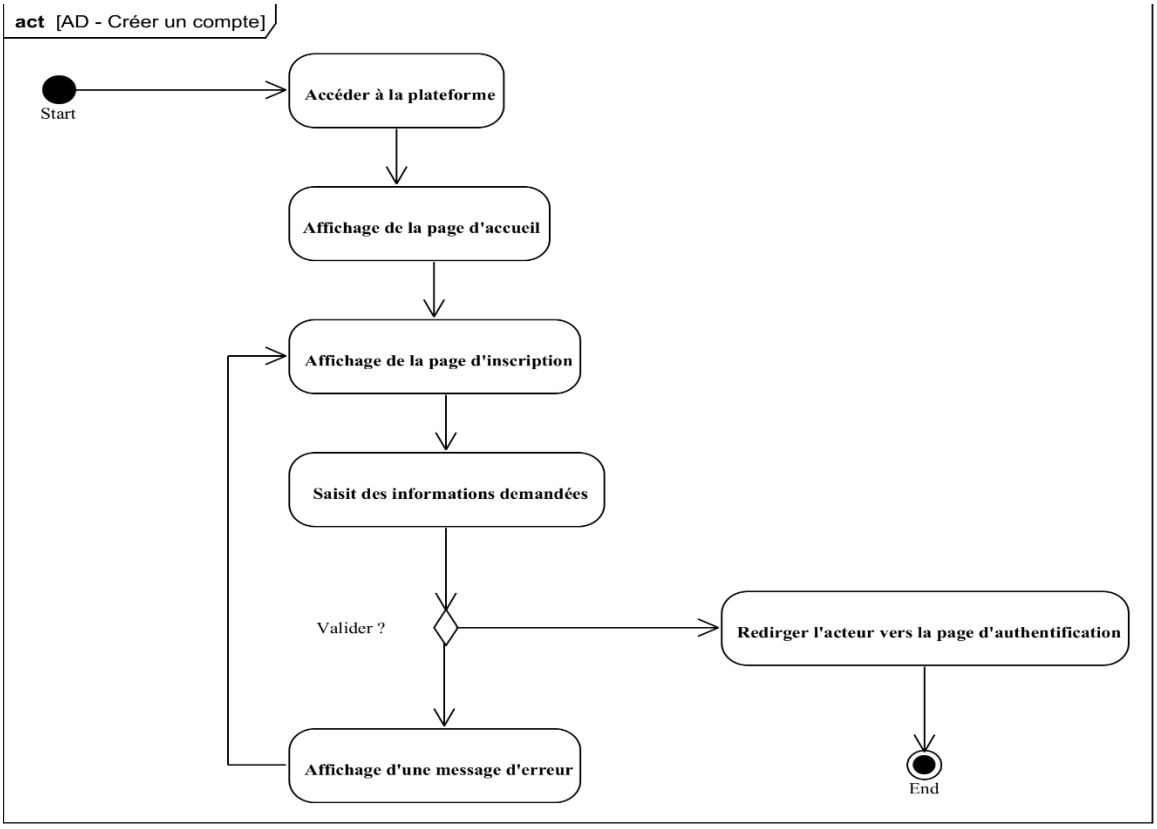

Figure 2.29 : *Diagramme d'activité : Créer un compte*

2.3.4.3 Diagramme d'activité de : « Accéder à un messagerie »

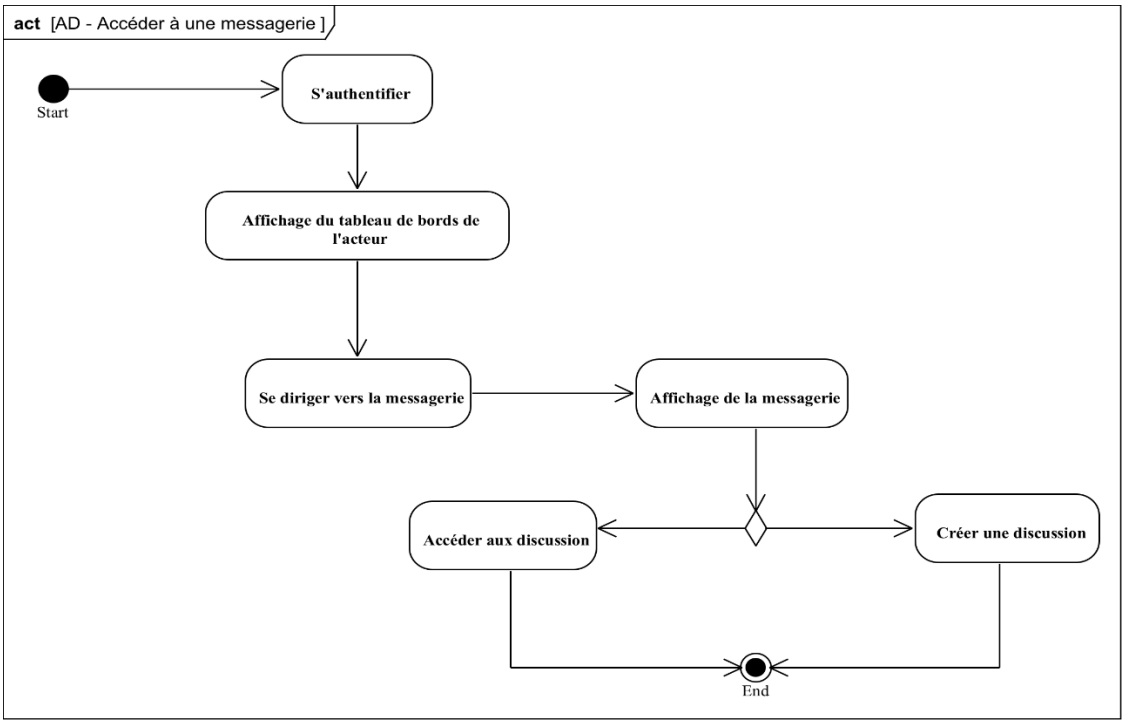

Figure 2.30 : *Diagramme d'activité : Accéder à une messagerie*

2.3.4.4 Diagramme d'activité de : « Envoyer/Télécharger des fichier »

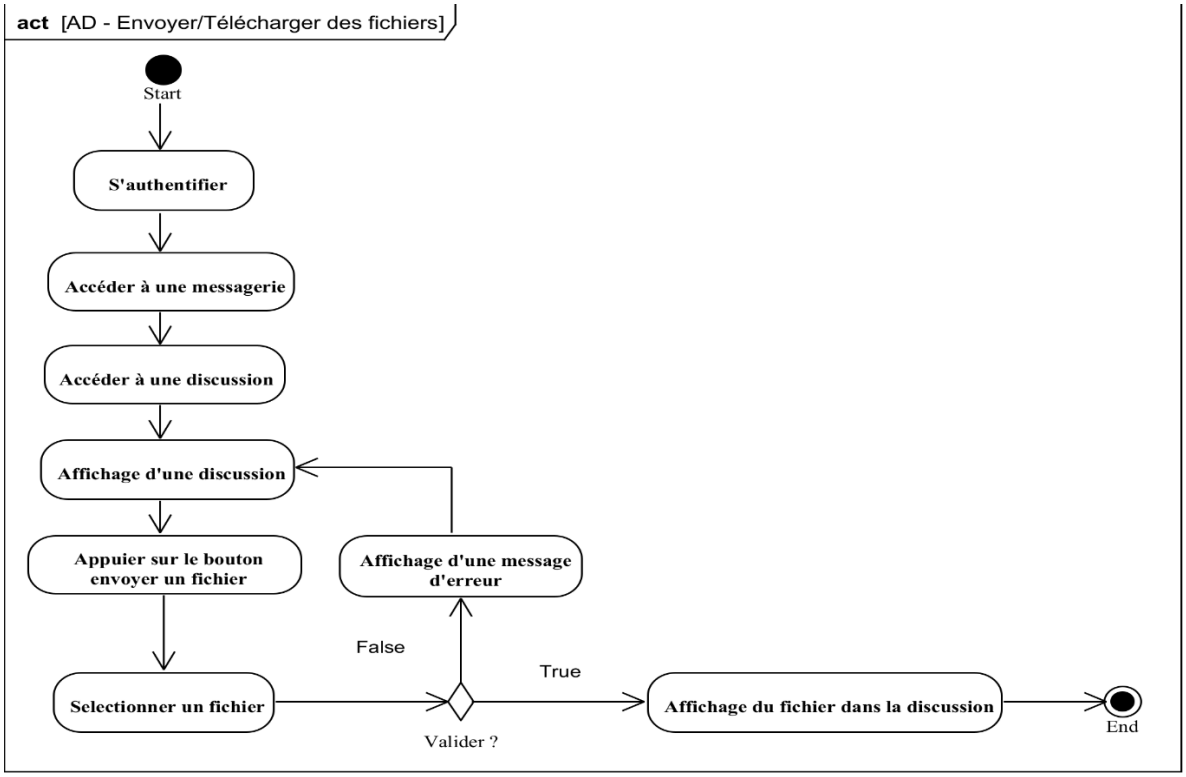

Figure 2.31 : *Diagramme d'activité : Envoyer/Télécharger des fichiers*

2.3.4.5 Diagramme d'activité de : « Créer une discussion »

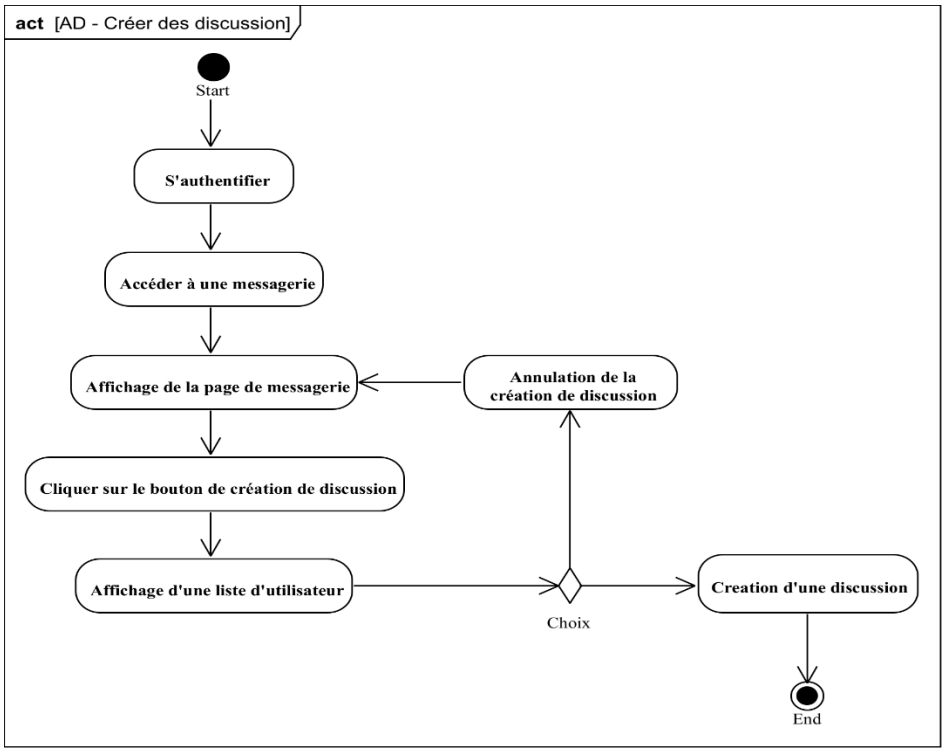

Figure 2.32 : *Diagramme d'activité : Créer des discussions*

2.3.4.6 Diagramme d'activité de : « Gérer les utilisateur »

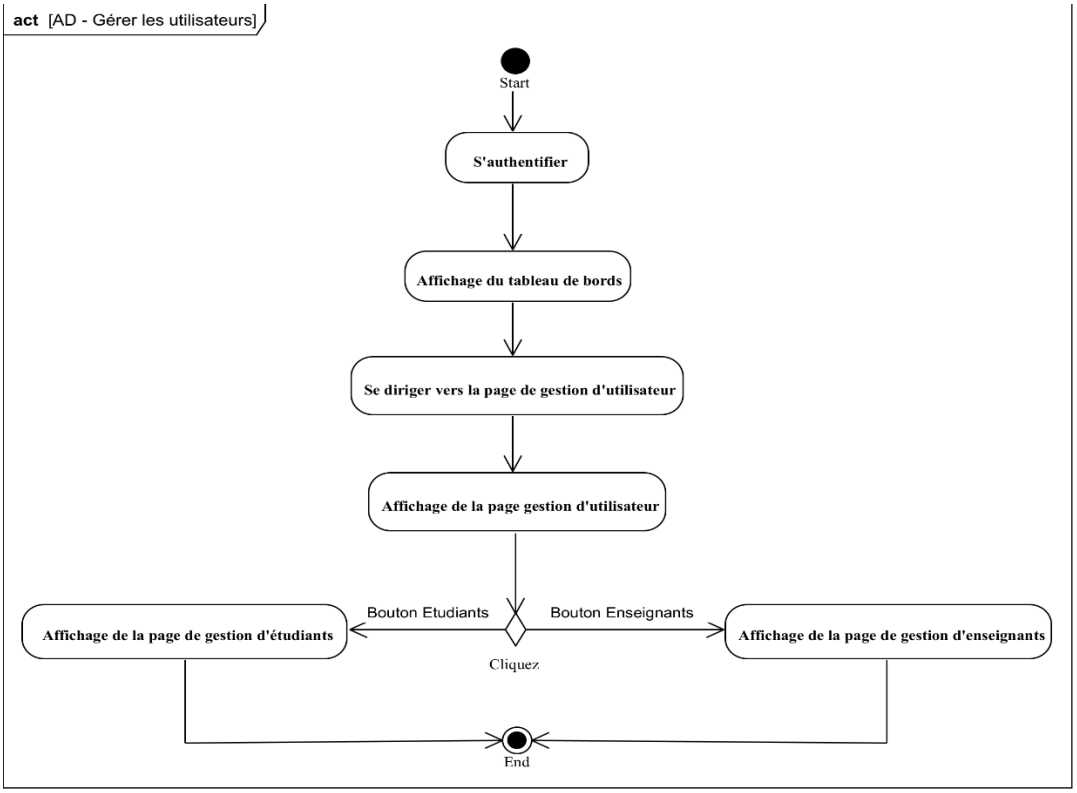

Figure 2.33 : *Diagramme d'activité : Gérer les utilisateurs*

2.3.4.7 Diagramme d'activité de : « Gérer les classes »

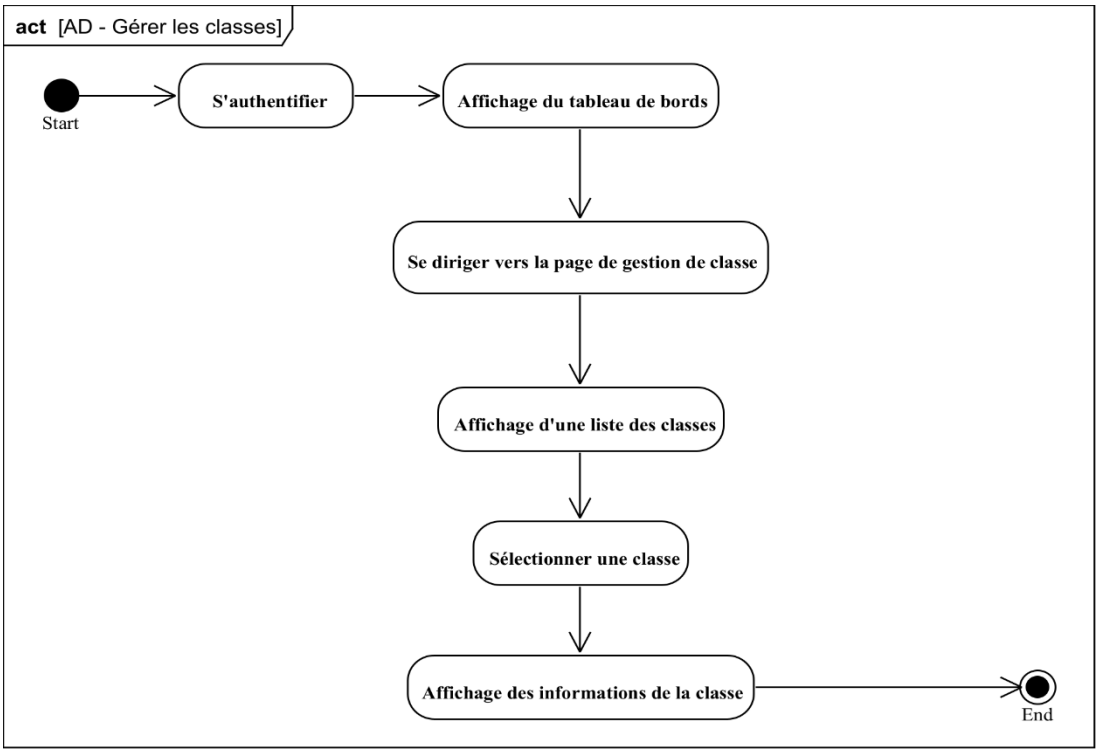

Figure 2.34 : *Diagramme d'activité : Gérer les classes*

2.3.5 Diagramme de classe

Le diagramme de classe est considéré comme le pivot de l'ensemble de la modélisation d'un système. Il permet de donner la représentation statique du système à développer, il est fondé sur le concept d'objet, le concept de classe comprenant les attributs, les opérations et les différents types d'association entre classes.

Ce diagramme est composé des éléments suivants :

- Classe : qui permet de décrire un groupe d'objets et comportant les attributs et les opérations ;
- Attribut : propriété élémentaire d'une classe ;
- Opération : c'est une fonction applicable aux objets d'une classe. Elles décrivent les comportements d'un objet ;
- Association : représente les liens existants entre les instances des classes ;
- Multiplicité : indique un domaine de valeur pour préciser le nombre d'instance d'une classe vis-à-vis d'une autre classe pour une association donnée ;

Le diagramme de classe de notre projet est représenté dans la figure suivante :

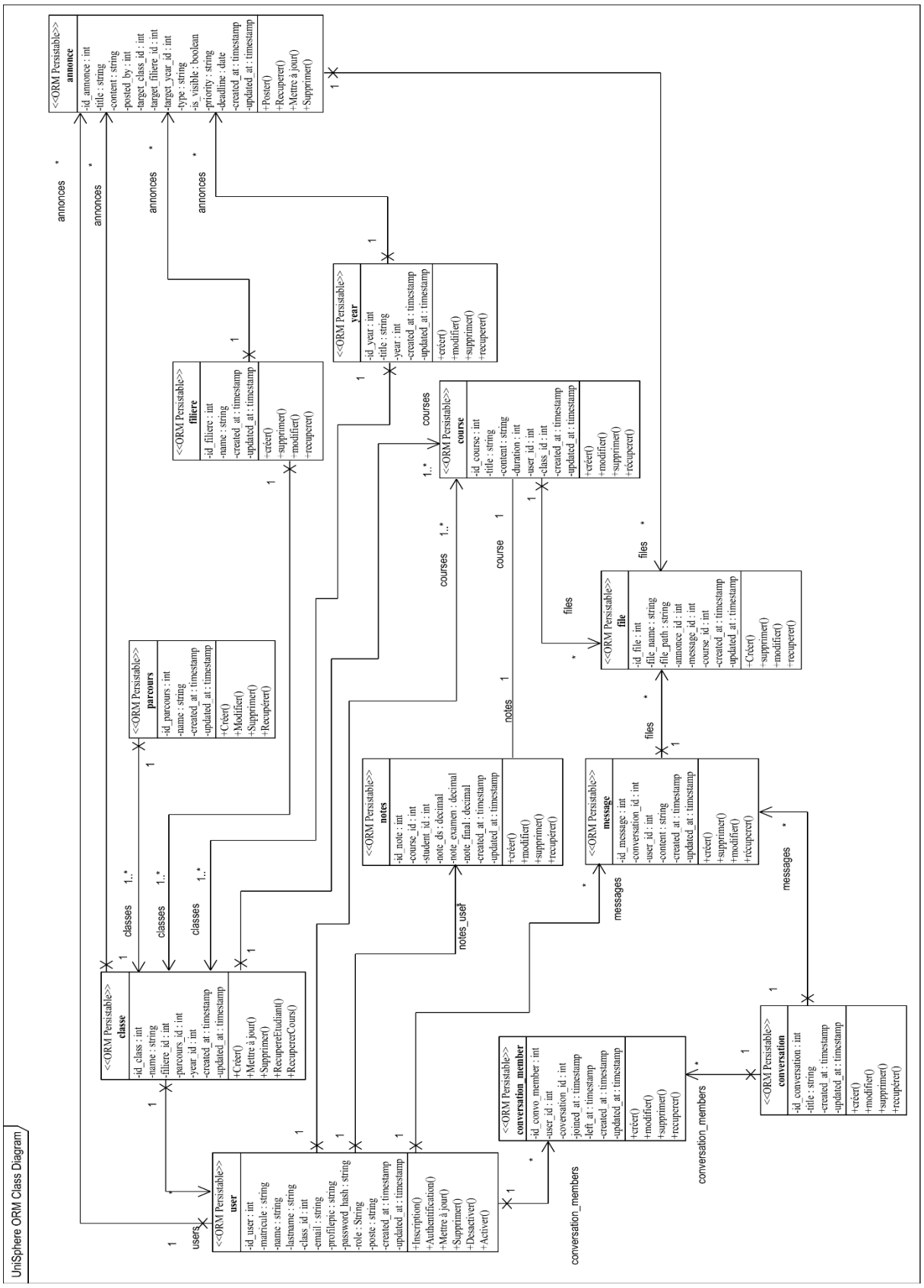

Figure 2.35 : *Diagramme de classe de l'UniSphere*

2.4 Conclusion

Comme nous avons vu dans ce chapitre, la phase de conception nous a permis de détailler les fonctionnalités importantes de notre plateforme. Ainsi dans le prochain chapitre nous allons voir la phase de réalisation de notre projet.

CHAPITRE 3 REALISATION DU PROJET

3.1 Introduction

La phase de réalisation est la partie essentielle durant le développement de notre projet informatique. De ce fait, nous allons présenter les outils et les langages de programmation utilisés pour le développement de ce projet. Ainsi qu'une présentation de la plateforme via ces interfaces graphiques.

3.2 Langages et framework de développement

3.2.1 Les langages de programmations utilisées

3.2.1.1 Langage de balise HyperText Markup Language : HTML

L'HTML signifie « HyperText Markup Language » qu'on peut traduire par « Langage de balises pour l'hypertexte ». Il est utilisé afin de créer et de représenter le contenu d'une page web et sa structure. D'autres technologies sont utilisées avec HTML pour décrire la présentation d'une page et/ou ses fonctionnalités interactives. [3.01]

L' « hypertexte » désigne les liens qui relient les pages web entre elles, que ce soit au sein d'un même site web ou entre différents sites web. Les liens sont un aspect fondamental du web. Ce eux qui forment cette « toile », mot traduit par web en anglais. En téléchargeant du contenu sur l'Internet et en les reliant à des pages créées par d'autres personnes. [3.01]

Le HTML a été inventer par Tim Berners-Lee en 1990 alors qu'il travaillait au CERN de Genève. Son objectif principal était de relise des pages entre elles via des liens hypertexte et de rendre les documents lisibles sur tous les ordinateurs indépendamment de leur système d'exploitation. Depuis le HTML a évolué à travers plusieurs versions : HTML en 1991, HTML+, HTML 2, HTML3.2, HTML 4.01 en 1999 et XHTMLen 2000.

En 2004, le Web Hypertexte Application Technologie working Group (WebWG) a été créé pour relancer le développement du HTML, conduisant à la spécification de HTML 5 en 2007. HTML 5 a été officialiser en aout 2012 et continue de s'améliorer. Cette version met l'accent sur la compatibilité avec les contenus existants et l'adaptation aux besoins des utilisateurs grâce à une approche pragmatique des évolutions technologiques [3.02]

3.2.1.2 Cascading Style Sheet : CSS

Le Cascading Style Sheets ou le CSS, est un langage de feuille de style utilise pour décrire la présentation d'un document écrit en HTML ou XML. Il permet de séparer le contenu HTML de

la présentation, en contrôlant l'apparence des éléments, la mise en page, les polices, les couleurs, etc., sur différents supports comme les écrans, le papier ou la voix

CSS est l'un des langages principaux du Web ouvert et a été standardisé par le World Wide Web Consortium (W3C). Auparavant, le développement des différentes parties de la spécification CSS était réaliser de façon synchrone, permettant d'avoir une version pour l'ensemble de la recommandation. Nous avons donc entendu parler de CSS1, CSS2.1, voire CSS3. Toutefois, il n'y aura pas de version CSS4 ou d'autres version globale numérotée de CSS. [3.03]

3.2.1.3 Langage de programmation : JavaScript

Le JavaScript souvent abréger en « JS » est un langage de script léger, orienté objet, principalement connu comme le langage de script des pages web. Mais il est aussi utilisé dans de nombreux environnements extérieurs aux navigateurs web tels que Node.JS, Apache CouchDB, voire Adobe Acrobat. Le code JavaScript est interprété ou compilé à la volée. C'est un langage à objets utilisant le concept de prototype, disposant d'un typage faible et dynamique qui permet de programmer suivant plusieurs paradigmes de programmation : fonctionnelle, impérative et orienté objet. [3.04]

Le JavaScript est un langage de programmation informatique open source créée en 1995 par Brendan Eich, un programmeur travaillant pour Netscape Communication Corporation. Crées-en seulement 10 jours, le JavaScript est appelé à l'origine « Mocha », mais a ensuite été renommé « LiveScripté » avant de se fixer finalement sur JavaScript, dans une démarche stratégique visant à surfer sur la popularité de Java à l'époque. [3.05]

Avec HTML et CSS, le JavaScript représente en fait l'un des trois principaux langages de programmation utilisé dans le développement web. [3.05]

a- JavaScript moderne à partir de 2015

Le JavaScript continue d'évoluer grâce à des mise à jour régulières de la spécifications ECAMScript. Les fonctionnalités introduites dans les version suivantes (ES2016, ES2017, etc…) comprennent async/await, de nouvelles méthodes et une syntaxe améliorée. Ces dernières années, avec le développement de cadres et de bibliothèques puissants pour les développements frontaux et dorsaux, l'écosystème JavaScript s'est considérablement développé. [3.05]

b- L'utilisation de JavaScript de nos jours

De nos jours, JavaScript est un langage de programmation d'une impressionnante polyvalence qui peut être déployé dans divers aspects du développement web. Sa flexibilité et son large éventail de cadres le rendent idéal pour les types de développement web suivants [3.05] :

- Développement de l'interface utilisateur (UI) : dans le cadre du développement web coté client, JavaScript est essentiel pour créer des interfaces utilisateur dynamiques et interactives. Il est utilisé pour manipuler le Document Object Model ou DOM et répondre aux actions de l'utilisateur, afin d'offrir une expérience transparente et attrayante.
- Développement web coté serveur : avec l'introductions de Node JS, JavaScript peut être utiliser cote serveur, ce qui permet aux développeurs de créer des applications serveur évolutives et performantes. Cela permet d'utiliser un langage unifié pour le développement coté client et cote serveur.
- Développement complet : JavaScript est un composant clé des piles MEAN qui signifie MongoDB, Express.JS, Angular.JS, Node.JS et MERN, MongoDB, Express.JS, React.JS, Node.JS, qui sont des choix populaires pour le développement à pile complète. Ces piles permettent aux développeurs d'utiliser JavaScript dans l'ensemble de l'application, de la base de données au frontend.
- Développement d'application mobile : dans des cadres comme React Native et NativeScript, JavaScript est utiliser pour créer des applications mobiles multiplateformes. Cela permet aux développeurs d'écrire du code une seule fois et de le déployer sur les plateformes IOS et Android.
- Applications à page unique (SPA) : JavaScript est un langage de base pour la construction de SPA, ou une seule page HTML est dynamiquement mise à jour lorsque l'utilisateur interagit avec l'application. Des framework comme Angular.JS, React.JS et Vue.JS simplifient le développement des SPA en fournissant des architectures structurées.
- API Web et intégration : couramment utilisé pour interagir avec des API externes, JavaScript permet l'intégration de services et de données tiers dans les applications web. Cela est essentiel pour créer des applications riches en fonctionnalité et axées sur les données.
- Développement de jeux sur le web : pour créer des jeux sur le web, JavaScript peut être déployé en combinaison avec HTML 5 et CSS3. Des bibliothèques comme Phaser et Three.js facilitent la création de jeux interactifs et visuellement attrayantes directement dans le navigateur.

3.2.2 Les frameworks utilisées

3.2.2.1 Framework Vue JS

Le framework Vue.JS est un framework évolutif pour construire des interfaces utilisateur. A la différence des autre framework monolithiques, Vue a été conçu et pense pour pouvoir être adopté de manière incrémentale. Le cœur de la bibliothèque se concentre uniquement sur la partie vue, et il est vraiment simple de l'intégrer avec d'autres bibliothèque ou projets existants. D'un autre côté, Vue est tout à fait capable de faire tourner des applications web mono-pages quand il est couplé avec des outils modernes et des bibliothèques complémentaires. [3.06]

a. Historique

Vue JS a été créé à l'origine par une seule personne, Evann You, ancien ingénieur de Google ayant notamment travaillé avec Angular.JS puis sur le framework Meteor. Après avoir expérimenté une mécanique de réactivité qu'il jugeait plus intéressante que celle d'Angular, il décide de publier ses premiers résultats en juillet 2013. Cinq ans plus tard, Vue.JS est le 3ème projet sur GitHub en nombre de stars et Evan enchaine les conférences à travers le monde. [3.07]

b. Equipe de développement

Vue JS dispose aujourd'hui d'une équipe internationale d'une trentaine de personnes, constituée de contributeurs bénévoles qui se sont formé avec les années. La décentralisation complétée de l'équipe est à la fois une contrainte et une force qui lui a permis de diffuser le framework beaucoup plus rapidement à plusieurs endroits à la fois. [3.07].

c. Comparaison avec les autres frameworks

Tâchons de comparer Vue avec les deux autres framework JS les plus populaires, React et Angular. [3.07].

Premièrement, les points communs entre ces 3 frameworks :

- Très populaire, utilisés par de grosse entreprise
- Mature, stables, support à long terme financé
- Codebase orienté composants
- Paradigme principalement déclaratif et non impératif
- Adapté aux stacks modernes avec ES6+/ TypeScript
- Large écosystème de composants et d'outillage

Deuxièmement, le positionnement : React, Vue et Angular se positionnent dans des catégories différentes et cela s'observe dès les premières lignes d'introduction. React se définit comme une bibliothèque indépendante du reste de la stack technique, Angular se définit comme le framework unique répondant à tous les besoins, desktop comme mobile. Tandis que Vue se positionne entre deux comme un framework progressif, polyvalent et que l'on peut adopter par étapes. Un juste milieu entre une bibliothèque et un framework tout équipé.

Enfin une comparaison de langage, style et stack technique

Tableau 3.01 : *Comparaison entre divers framework*

	React	Vue.js	Angular
Langage	JSX, (TypeScript)	TypeScript, JSX	TypeScript
Gestion d'état	State centraliser non mutable (flux)	State centraliser en option (Vuex), sinon interne et mutable	State interne et mutable, pas de state centraliser officiel
Gestion de la réactivité	Manuelle (setState + VDOM diffing)	Automatique (observers/Proxies)	Automatique (Zones/dirty-checking)
Stack	Ne se suffit pas à lui-même, à intégrer dans une stack web avec d'autres outils en externe. Ecosystème riche	Quelques outils officiels maintenus par la team Vue, d'autres à chercher en externe. Promeut des solutions tierces si elles sont appropriées	Framework full stack et auto suffisant, avec options de sécurité incluses nativement. Complet mais plus fermé.

State : objet de données représentant l'état de l'application ;

Interne : propre à chaque composant

3.2.2.2 Express.JS

Express JS, parfois aussi appelé « Express », est un framework backend Node.JS minimaliste, rapide et de type Sinatra qui offre des fonctionnalités et des outils robustes pour développer des applications backend évolutives. Il vous offre le système de routage et des fonctionnalités simplifiées pour étendre le framework en développant des composants et des parties plus puissants en fonctions des cas d'utilisations de notre application. [3.08]

C’est de ce fait le framework standard pour le développement de serveur en Node.JS. L’auteur original, TJ Holowaychuck, le décrit comme étant un serveur inspiré de Sinatra dans le sens qu’il est relativement minimaliste tout en permettant d’étendre ses fonctionnalités via des plugins.

Express fournit un ensemble d’outils pour les applications web, les requêtes et les réponses http, le routage et les intergiciels permettant de créer et de déployer des applications à grande échelle, prêtes pour l’entreprise. Il fournit également un outil d’interface de ligne de commande appelé Node Package Manager ou NPM, où l’on peut s’approvisionner en paquets développés. [3.08]

Notamment utilisé dans un large éventail de choses dans l’écosystème JavaScript/Node.JS, nous pouvons également développer des applications, des points de terminaison API, des systèmes de routage et des frameworks avec lui.

Voici quelques-uns des types d’applications créer avec Express JS :

- Applications à page unique
- Outils de collaboration en temps réel
- Applications de streaming
- Application Fintech

3.2.2.3 Socket.IO

Le Socket.IO est une bibliothèque JavaScript open source qui fournit une solution pour la communication en temps réel entre un client et un serveur dans les applications web modernes. Créée par Guillermo Rauch en 2010, elle est développée en continu par une communauté de contributeurs.

Couramment utilisée dans les applications web qui nécessitent une communication en temps réel, telles que les applications de messagerie instantanée, les jeux en ligne et les tableaux de bord en temps réel. Socket.IO fonctionne avec des technologies web telles que Node.JS et les WebSockets pour offrir une communication bidirectionnelle efficace entre un client et un serveur.

Contrairement au WebSockets, Socket.IO n’est pas une spécification officielle, elle a donc sa propre API. Toutefois, elle prend en charge les WebSockets et d’autres protocoles de communication en temps réel tels que Server-Sent Event ou SSE et Long Polling, selon la disponibilité du navigateur et du serveur. [3.09]

Le Socket.IO offre plusieurs avantages par rapport au WebSockets :

- Socket.IO diffuse un message à tous les clients connectés.

- Les proxys et les répartiteurs de charge rendent la mise en œuvre et l'échelle des WebSockets difficiles. Socket.IO prend en charge ces technologies en standard.
- Comme mentionné précédemment, Socket.IO peut passer à d'autre technologie que WebSockets lorsque le client ne le prend pas en charge
- Lors des interruptions de connexion, WebSockets ne se reconnectera pas automatiquement tandis que Socket.IO le fait grâce à un mécanisme de ping/pong, qui vérifie périodiquement l'état de la connexion.

3.2.2.4 Tailwind CSS

Tailwind CSS est un framework utility-First CSS avec des classes prédéfinis que nous pouvons utiliser pour construire et concevoir des pages web directement dans votre balisage. Il nous permet d'écrire du CSS dans notre HTML sous la forme de classes prédéfinies. [3.10]

Créer en 2017 par Adam Wathan, un homme passionné par le développement web et est à l'origine de nombreux projet tels que Laravel Valet, Jigsaw et bien sûr Tailwind CSS. C'est en travaillant sur un projet nommé KiteTail que lui vient l'idée de créer le framework Tailwind CSS. Avec ce framework, il est possible de créer un design d'interface au sein même du fichier HTML. Cette façon de programmer n'interfère pas avec les pratiques recommandées par le W3C comme celle de séparer le HTML des feuilles de styles CSS. Pour utiliser Tailwind, il faut prendre les classes CSS prédéfinies par le framework en les appelant dans un fichier HTML comme ceci : [3.11]

3.3 Logiciels de développement

3.3.1 Editeur de code Visual Studio Code

Visual Studio Code est un éditeur de code source léger, mais puissant, qui s'exécute sur notre bureau et est disponible pour Windows, MacOs et Linux. Il est livré avec la prise en charge intégrée pour JavaScript, TypeScript et Node.JS et dispose d'un riche écosystème d'extensions pour d'autres langages et environnements d'exécution tels que C++, C#, Java, Python, PHP, Go, .Net [3.12].

Il offre de nombreuses fonctionnalités à l'utilisateur comme la coloration syntaxique, l'auto-complétions, la mise en évidence des erreurs, la navigation de code, le débogage, la gestion de versions, l'intégration avec Git. Grâce à l'aide d'une grande variété d'extensions développées par la communauté, l'éditeur permet aux développeurs de personnaliser ce dernier selon leurs besoins.

3.3.2 Visual Paradigm

Visual Paradigm est un outil complet qui intègre de nombreuses fonctionnalités et modules essentiels pour la modélisation et le développement de logiciels. Voici quelques-unes des principales caractéristiques [3.13] :

- Modélisation UML : Visual paradigm prend en charge tous les diagrammes UML standard, y compris les diagrammes de cas d'utilisation, de classes, de séquence, d'activité et d'état.
- Modélisation de processus métier : Grace à la notation BPMN, nous pouvons modéliser, documenter et optimiser des processus métier
- Conception de base de données : Créer des modèles de donnée conceptuels logiques et physiques à l'aide de diagrammes entité-relation ou ERD.
- Architecture d'entreprise : utiliser ArchiMate pour modéliser et documenter l'architecture d'entreprise, y compris les aspects métier, applicatifs et technologiques
- Gestion des exigences : capturer, organiser et gérer les exigences de nos projets de manière centralisée.
- Génération de code : générer automatiquement du code source à partir de nos modèles UML, économisant ainsi du temps et des efforts.
- Développement agile : panifier et suivit de nos sprints, créez des backlog de produits et de sprints, et gérer les taches de manière efficace.

Visual paradigm se distingue par sa capacité à s'intégrer à d'autres outils et plateformes populaires, tels que Git, Jira, Confluence et Microsoft Office. Cette interopérabilité facilite la collaboration au sein des équipes et améliore l'efficacité globale du processus de développement [3.13].

3.3.3 Plateforme GitHub

GitHub est un plateforme web et un service de cloud qui aide les développeurs à stocker et à gérer leur code, ainsi qu'à suivre et contrôler les modifications qui lui sont apportée. Créer en 2008 par Tom Preston-Werner, Chris Wanstrath et PJ Hyett. Preston-Werner a expliqué que l'idée de créer GitHub était venue de l'utilisation de Git. A l'époque, il n'y avait pas de plateforme en ligne permettant de stocker et de partager du code avec d'autre développeurs. [3.15]

GitHub se base sur deux principes : [3.14]

- Contrôle de version : le contrôle de version aide les développeurs à suivre et à gérer les modifications apportées au code d'un projet logiciel. Au fur et à mesure qu'un projet prend de l'ampleur, le contrôle de version devient essentiel.
- Git : c'est un système de contrôle de version open-source spécifique crée par Linus Torvalds en 2005. Concrètement, git est un système de contrôle de version distribué, ce qui signifie que l'ensemble de la base du code et de l'historique est disponible sur l'ordinateur de chaque développeur. Ce qui permet des branchements et une fusion facile.

3.3.4 Plateforme Node.JS

Node JS est une plateforme de développement JavaScript. Ce n'est pas un serveur, ce n'est pas un framework, c'est juste le langage JavaScript avec des bibliothèques permettant de réaliser des actions comme écrire sur la sortie standard, ouvrir/fermer des connections réseau ou encore créer un fichier. [3.16]

On peut souvent le confondre avec un serveur car c'est son origine : Node.JS a été créé par Ryan Dahl dans le but de pouvoir créer des applications temps réel ou le serveur est capable de pousser de l'information au client. C'est dans ce but qu'il utilise la bibliothèque libuv pour réaliser son modèle d'entrer sortie non bloquante. [3.16]

Node.JS présente de nombreux intérêts :

- Logiciel libre sous la licence MIT
- Performance du moteur V8
- Modèle non bloquant
- Communauté très active
- Système de paquet intégrer : NPM

Grâce à une vaste liste de fonctionnalités, Node.JS a connu une croissance rapide au cours de ces dernière année : [3.17]

- Facile : Easy-Node.JS est assez facile à prendre en main. C'est un choix incontournable pour les débutants en développement web.
- Evolutif : il offre une grande évolutivité aux applications. Comme Node.JS est un single-thread, il est capable de gérer un grand nombre de connexions simultanées avec un débit élevé.

- Vitesse : l'exécution non bloquante des threads rend Node.JS encore plus rapide et plus efficace
- Paquets : un vaste ensemble de paquets Node.JS open source est disponible et peut simplifier notre travail
- Backend solide : Node.JS est écrit en C et C++, ce qui le rend rapide et ajoute des fonctionnalités comme le support réseau.
- Multi-plateforme : la prise en charge multi-plateforme nous permet de créer des sites web SaaS, des applications de bureau et même des applications mobiles, le tout en utilisant Node.JS
- Maintenable : Node.JS est un choix facile pour les développeurs, car le frontend et le backend peuvent être gères avec JavaScript comme un seul langage.

3.3.5 Logiciel X(cross) Apache MariaDB Perl PHP : Xampp

Le Logiciel Xampp est un logiciel de développement web permettant de mettre en place un serveur web local, un serveur FTP et un serveur de messagerie électronique qui fonctionne sous tous types de systèmes d'exploitation comme Windows, Linux et Mac. Il contient du MySQL que nous allons utiliser comme base de données pour notre plateforme.

3.4 Base de données : MySQL

Pour bien réaliser notre plateforme, nous avons besoin d'une base de données que nous utiliserons en tandem avec les autre frameworks et logiciels précédemment mentionnés. Cette base nous servira aussi à stocker les informations qui seront communiquer aux utilisateurs de notre plateforme. La base de données que nous utiliserons sera le MySQL.

MySQL est un système de gestion de base de données relationnelles SQL ou Structured Query Language open source développé et supporté par Oracle. [3.18]

MySQL est un SGBDR open source qui utilise SQL pour créer et gérer des bases de données. En tant que base de données relationnelle, MySQL stocke les données dans des tables de lignes et de colonnes organisées en schémas. Un schéma définit l'organisation et le stockage des données et décrit la relation entre les différentes tables. Avec ce format, les développeurs peuvent facilement stocker, extraire et analyser de nombreux types de données, y compris du texte simple, des chiffres, des dates, des heures et plus récemment, des vecteurs et JSON [3.19]

Nous avons choisi d'utiliser MySQL pour diverses raisons :

- La rapidité : le serveur MySQL est très rapide
- La facilité d'utilisation : MySQL est beaucoup plus simple à utiliser que la plupart des serveurs de bases de données commerciaux ;
- API diverse : on peut effectuer diverse opération sur une base MySQL en utilisant des interfaces écrites en C, Perl, C++, Java, Python, PHP ;
- Les connexion et sécurité : MySQL dispose d'un système de sécurité permettant de gérer les personnes et les machines pouvant accéder aux différentes bases.

3.5 Interfaces graphiques

Dans cette partie du chapitre, nous allons présenter quelques illustrations de notre plateforme :

3.5.1 Page d'accueil d'UniSphere

Figure 3.01 : *Page d'accueil de l'UniSphere*

3.5.2 Pages d'authentification

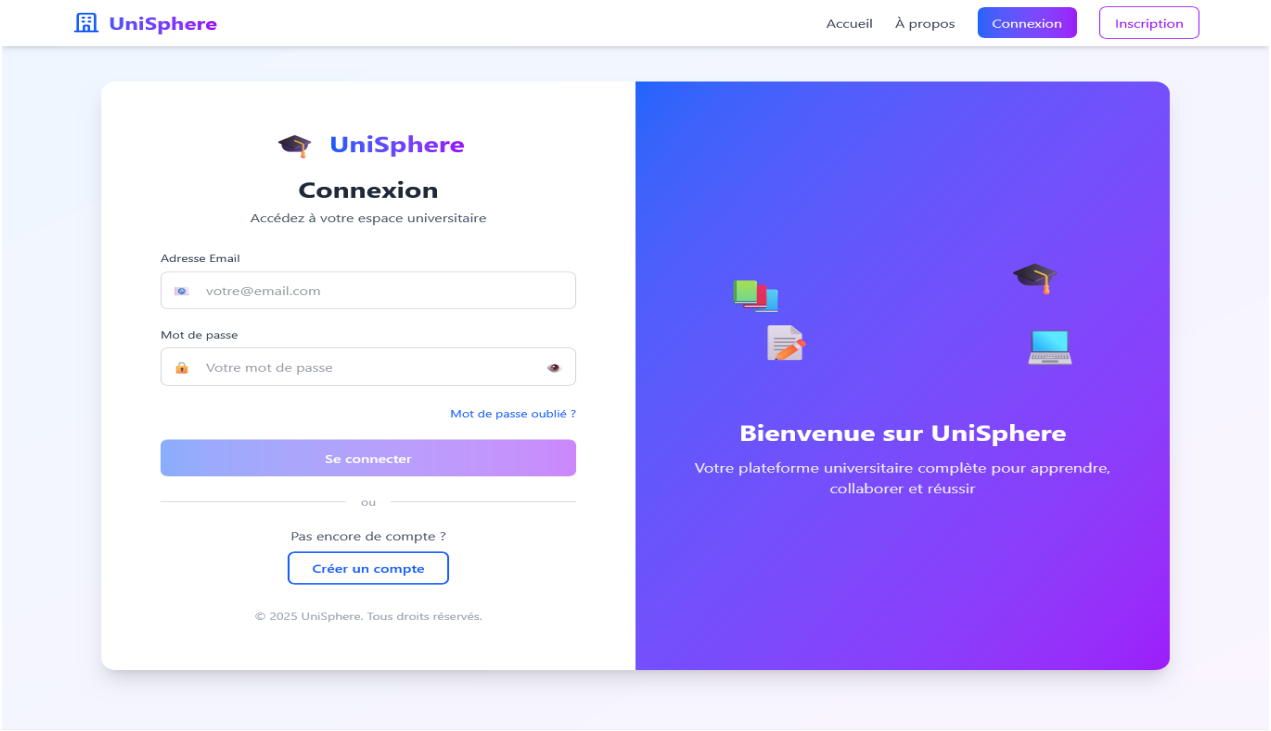

Figure 3.02 : *Page d'authentification*

3.5.3 Tableau de bord étudiant

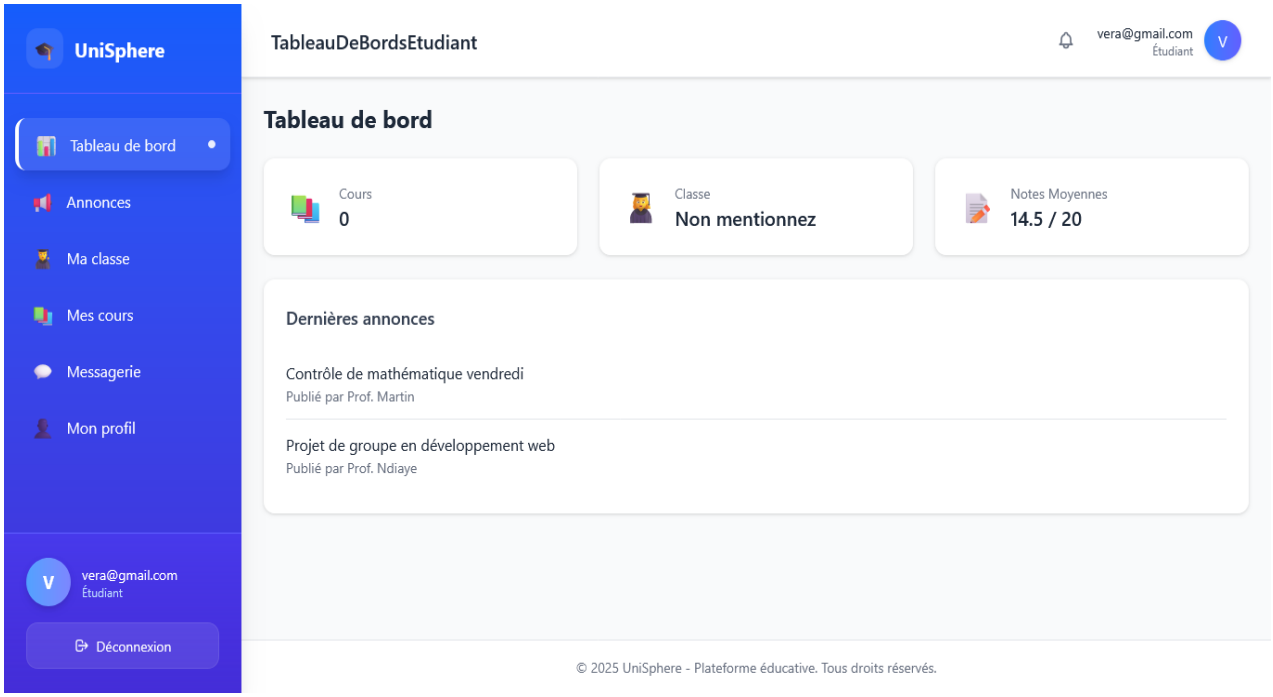

Figure 3.03 : *Tableau de bord : Etudiant*

3.5.4 Liste des cours de l'étudiant

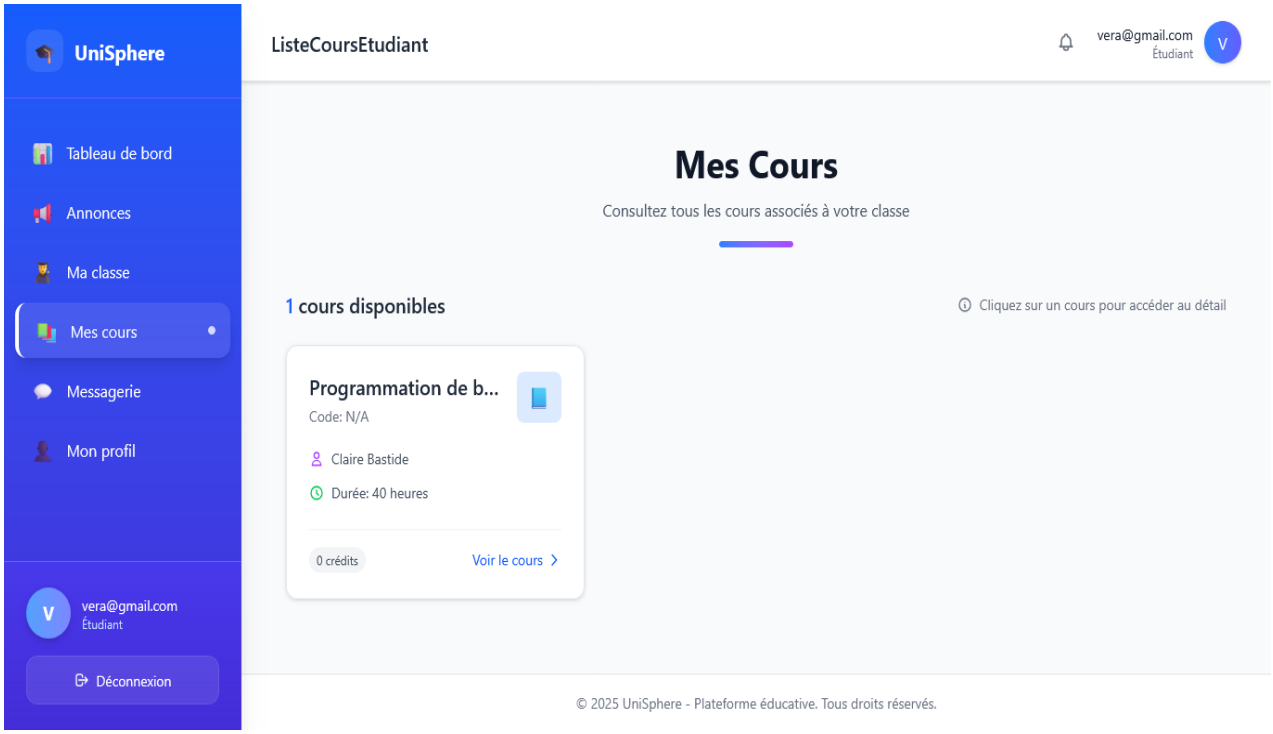

Figure 3.04 : *Liste des cours : Etudiant*

3.5.5 Affichage d'un cours

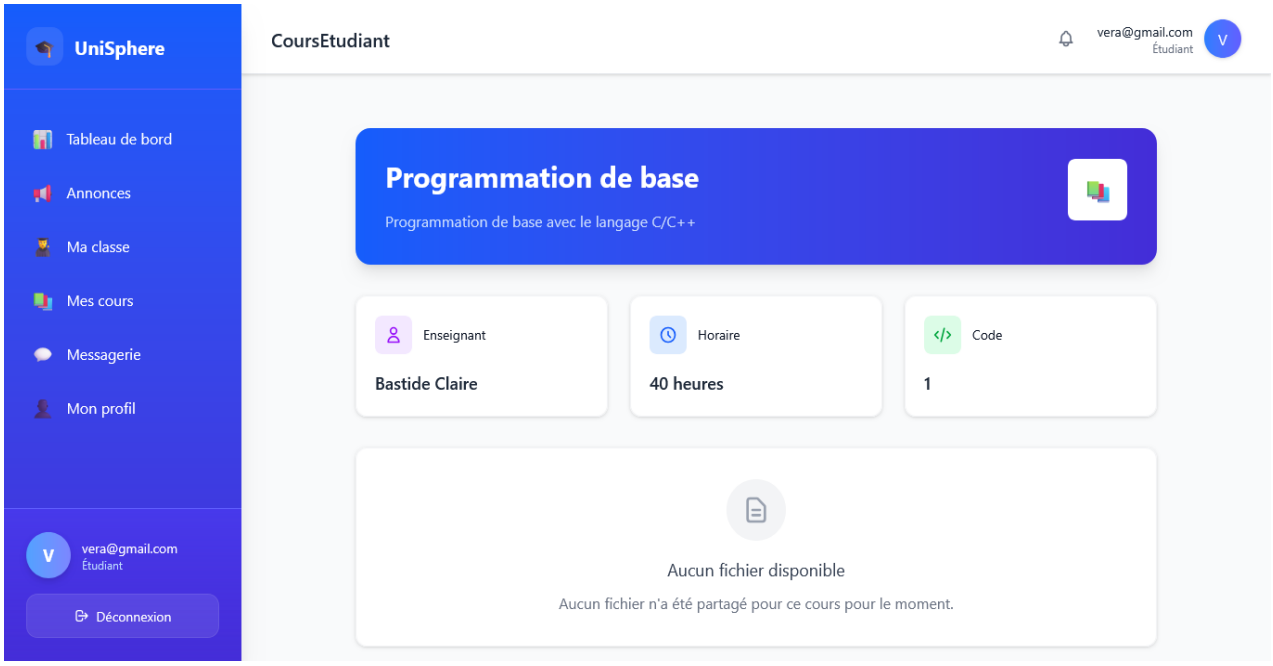

Figure 3.05 : *Affichage d'un cours : Etudiant*

3.5.6 Tableau de bord enseignants

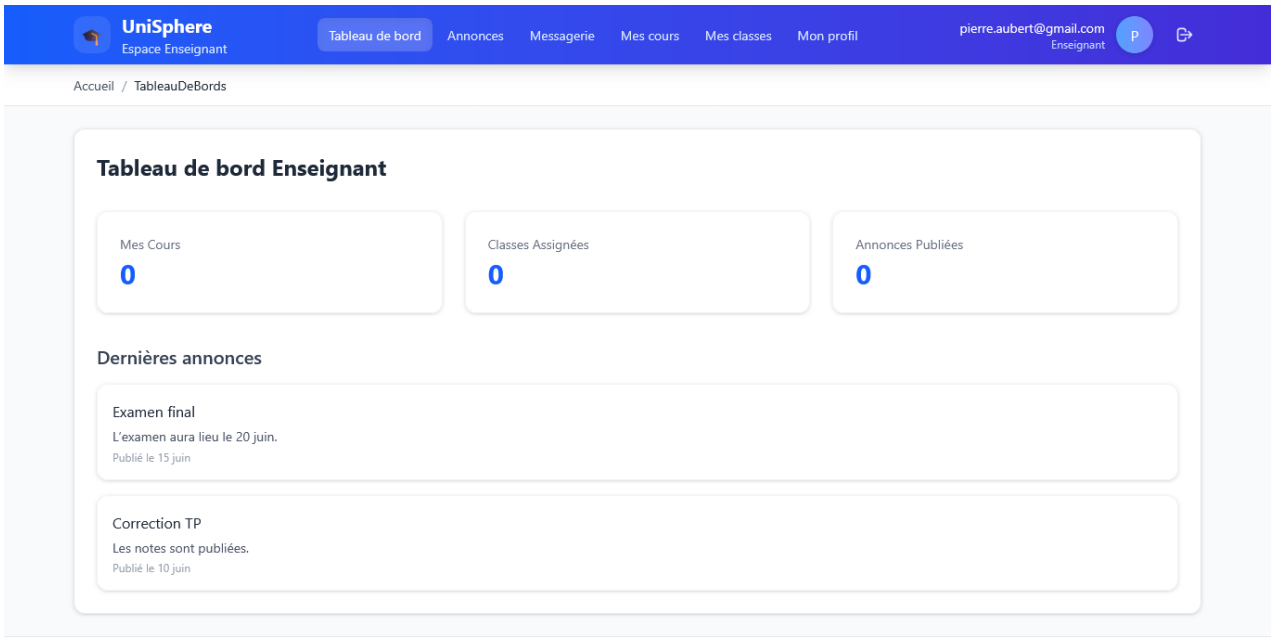

Figure 3.06 : *Tableau de bord : Enseignant*

3.5.7 Liste des classes de l'enseignant

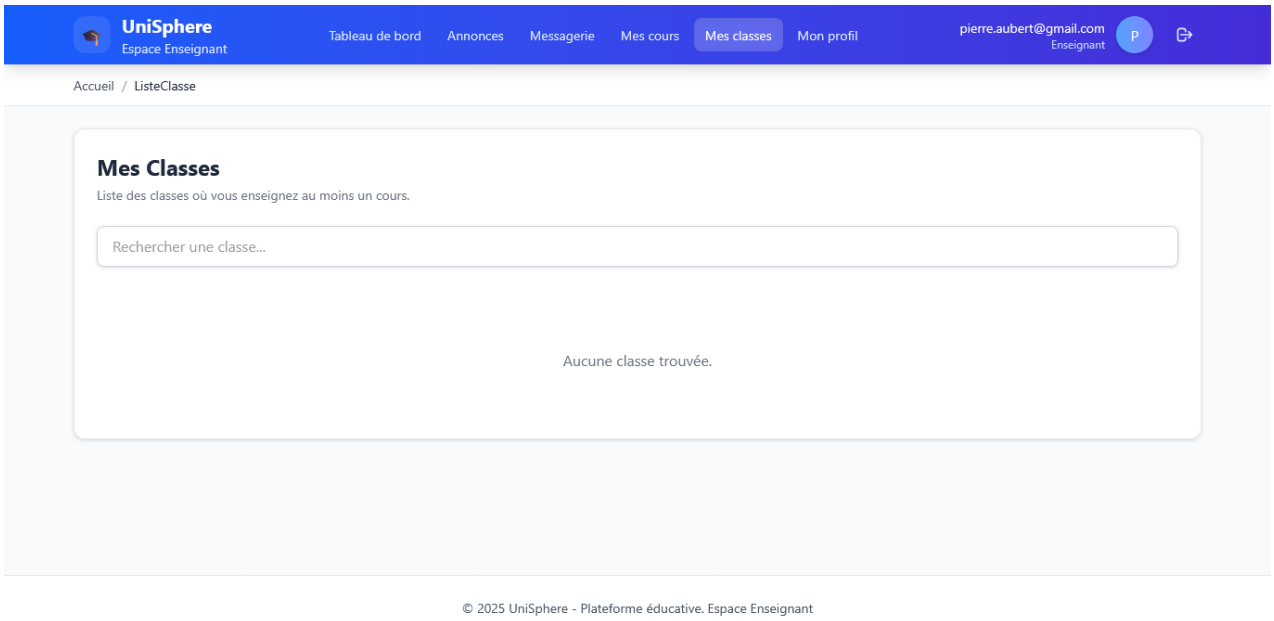

Figure 3.07 : *Liste des classes : Enseignant*

3.5.8 Liste des cours de l'enseignant

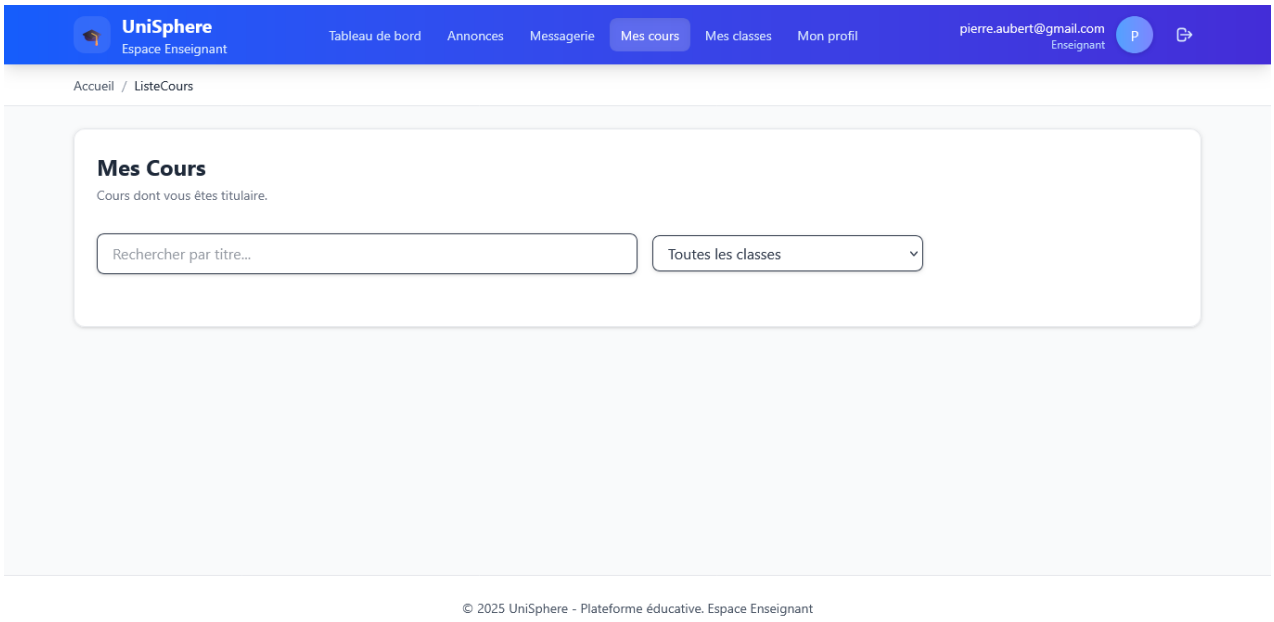

Figure 3.08 : *Liste des cours : Enseignant*

3.5.9 Tableau de bord de l'administrateur

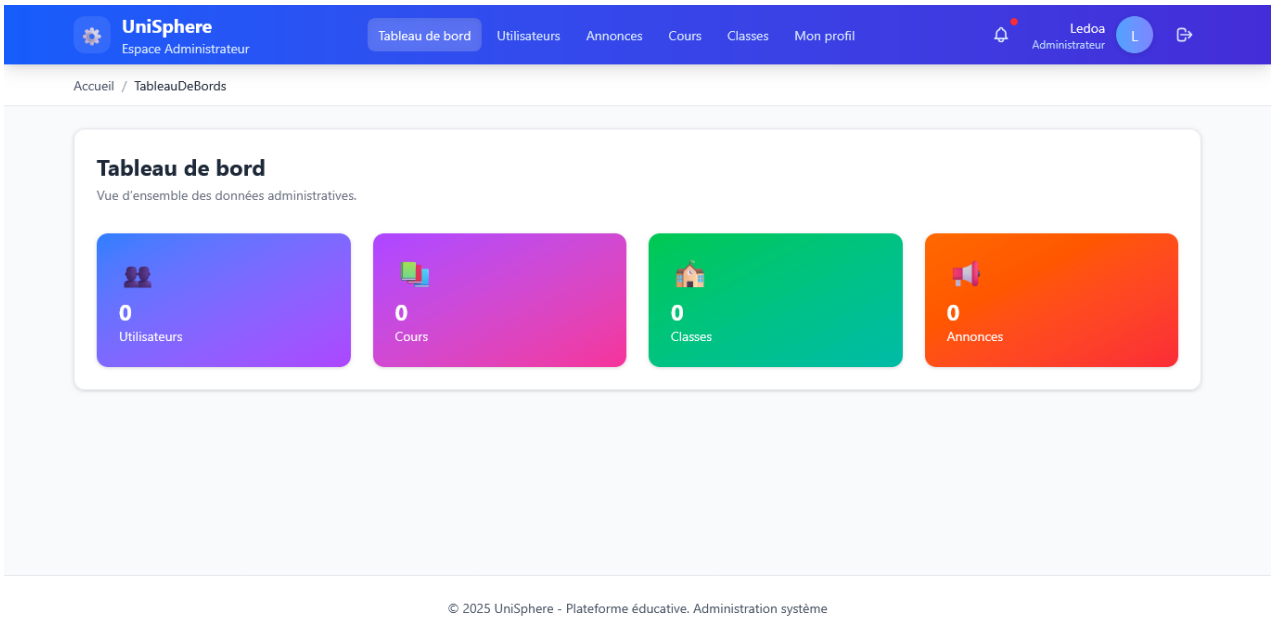

Figure 3.09 : *Tableau de bord : Administrateur*

3.5.10 Gestionnaire des classes

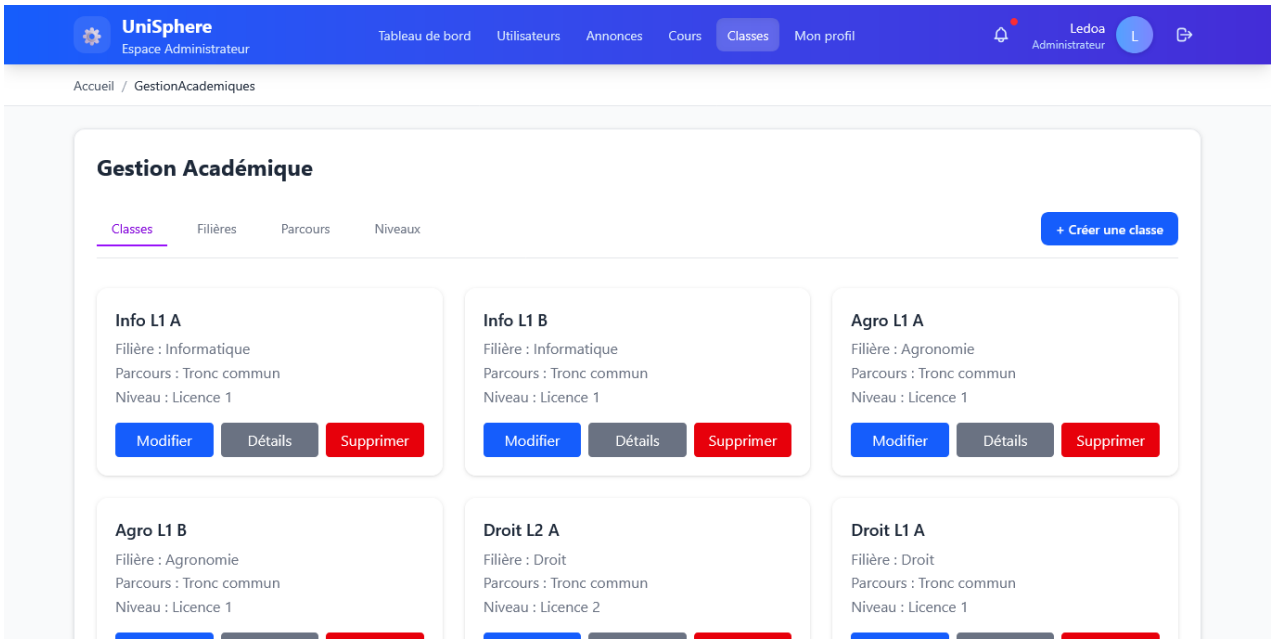

Figure 3.10 : *Gestionnaire des classes*

3.5.11 Gestionnaire des cours

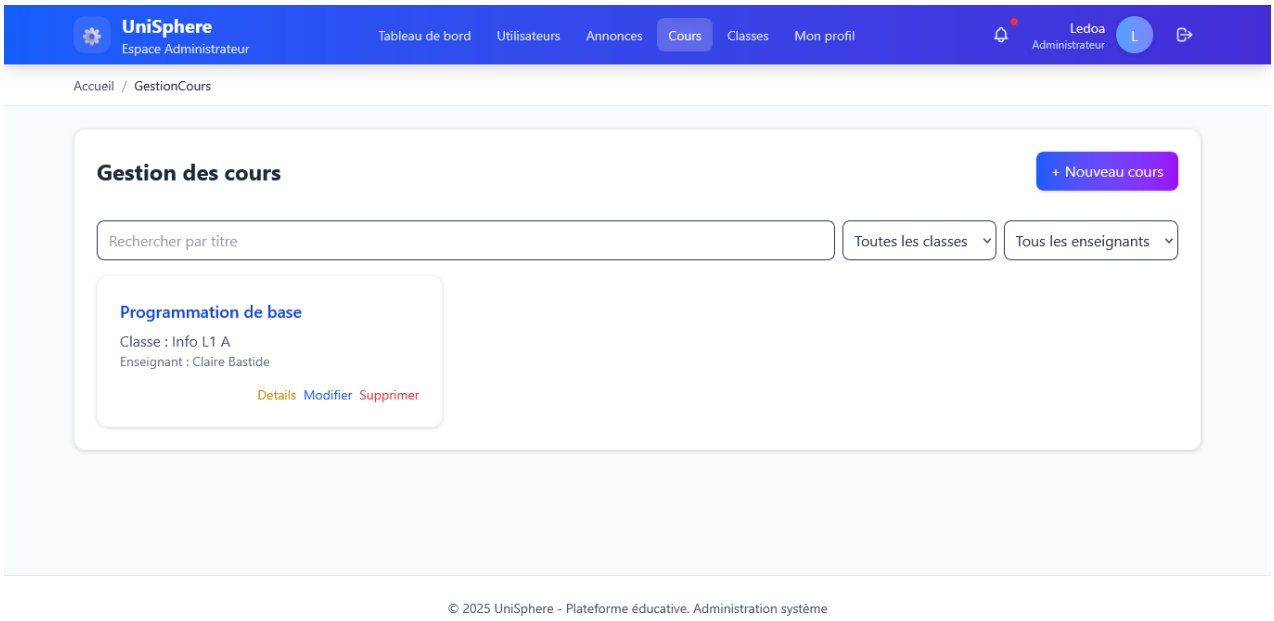

Figure 3.11 : *Gestionnaire des cours*

3.5.12 Gestionnaire des annonces

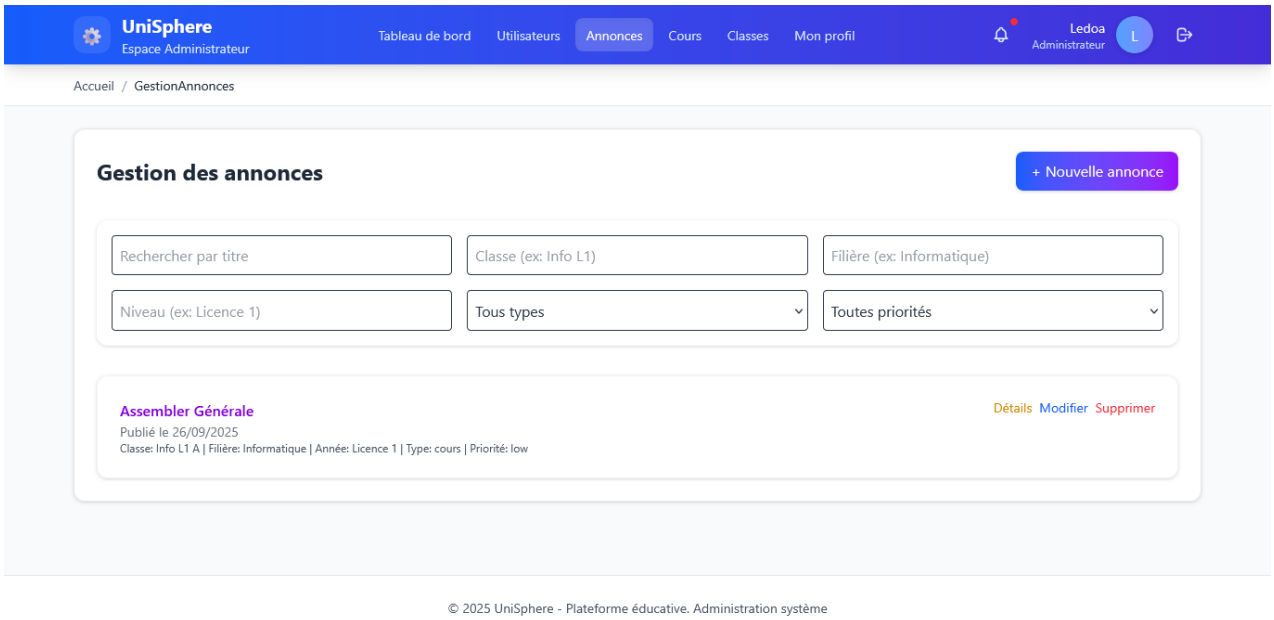

Figure 3.12 : *Gestionnaire des annonces*

3.5.13 Messagerie

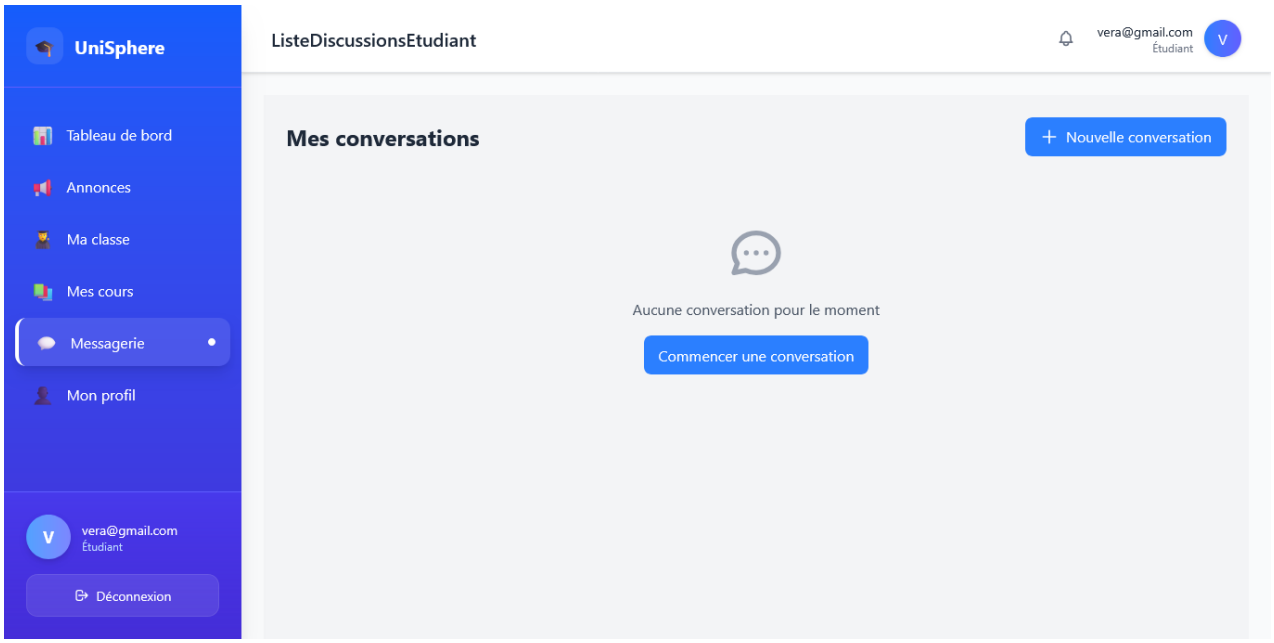

Figure 3.13 : *Messagerie*

3.6 Conclusion

Pour conclure, nous avons pu expliquer les langages de programmation, outils et logiciels utiliser lors du développement de notre plateforme. Mais aussi présenter les résultats par les

illustrations des interfaces de notre plateforme UniSphere. Ainsi dans le prochain chapitre nous allons effectuer à l'évaluation de notre projet et ces perspectives d'avenir.

CHAPITRE 4 EVALUATION ET PERSPECTIVE D'AMELIORATION DU PROJET

4.1 Introduction

La phase d'évaluation constitue une étape cruciale dans le cycle de développement de tout projet informatique [4.01]. Ce chapitre a pour objectif d'évaluer de manière exhaustive la plateforme UniSphere sous plusieurs angles : fonctionnel, technique, ergonomique et pédagogique. Nous allons voir dans ce chapitre les analyses détaillées des résultats obtenus, des limites constatés lors de la réalisation, ainsi que des possibles amélioration future du projet.

4.2 Méthodologie d'évaluation

Afin de mener une évaluation complète et objective, nous avons adopté une approche mixte, combinant plusieurs méthodes de collecte de données quantitatives et qualitatives [4.02]. Cette approche nous permet de trianguler les résultats et d'obtenir une vision complète des forces et faiblesses de la plateforme.

4.2.1 Cadre méthodologie générale

Notre méthodologie s'appuie sur le cadre d'évaluation des technologies éducatives proposé par l'UNESCO [4.03], qui recommande une approche systémique intégrant les dimensions techniques, pédagogiques et organisationnelles. Nous avons adapté ce cadre aux spécificités de notre plateforme en y intégrant des éléments propres aux systèmes de gestion de l'apprentissage modernes.

4.2.2 Protocol d'évaluation détaillé

Tableau 4.01 : *Méthodologie d'évaluation de la plateforme*

Type d'évaluation	Méthode utilisée	Participants	Durée	Objectifs spécifiques	Indicateur mesures
Test fonctionnels	Scénarios de test manuel [4.04]	Développeur 1 personne	3 semaines	Vérifier le bon fonctionnement des fonctionnalités	Taux de réussite des test, nombre de bugs critiques
Tests de performance	Outils Artillery.io, JMeter [4.05]	Environnement de test dédié	1 semaine	Mesurer les temps de réponse et la capacité de charge	Temps de réponse, débit, utilisation ressources
Test de sécurité	Analyse manuelle outils de scan [4.06]	Equipe sécurité 2 personnes	1 semaine	Identifier les vulnérabilité potentielles	Nombre de vulnérabilités critique/moyennes/faibles
Test utilisateurs	Questionnaires, observations directes, entretiens [4.07]	15 utilisateurs 5 par profil	2 semaines	Evaluer l'expérience utilisateur et l'utilité perçue	Score SUS, taux de satisfaction feedback qualitatifs
Comparaison avec les plateforme existants	Analyse comparative structurée	Benchmark avec Moodle et Google Classroom	1 semaine	Positionner UniSphere dans écosystème	Scores comparatifs sur divers critères

4.2.3 Profile des utilisateurs pour les tests

La sélection des utilisateurs tests a été effectuée selon une méthode d'échantillonnage raisonné visant à représenter la diversité des profils d'utilisateurs cibles de la plateforme

Tableau 4.02 : *Caractéristiques détaillées des utilisateurs participants à l'évaluation*

Profil	Nombre	Niveau technique	Expérience avec les LMS	Fréquence d'utilisation prévue
Etudiants	5	Intermédiaire	Aucune	Quotidienne à hebdomadaire
Enseignants	5	Débutant à intermédiaire	Aucune	Quotidienne à hebdomadaire
Administrateurs	5	Avancé	Expérimenté	Quotidienne

4.2.4 Instrument de collecte de données

Nous avons développé et adapté plusieurs instruments de collecte de données spécifiquement pour cette évaluation

- Grille d'évaluation technique : basée sur les norme ISO 25010 [4.08]
- Questionnaire de satisfaction utilisateur : adapté du System Usability Scale (SUS) [4.02]
- Guide d'entretien semi-directif : pour les retours qualitatifs
- Scenario de test utilisateurs : couvrant les principales fonctionnalités
- Checklist de sécurité : basée sur l'OWASP ASVS [4.06]

4.3 Résultats de l'évaluation fonctionnelle

L'évaluation fonctionnelle visait à vérifier que toutes les fonctionnalités planifiées avaient été correctement implémentées et qu'elles répondaient aux besoins exprimés

4.3.1 Etat d'avancement des fonctionnalités

Tableau 4.03 : *Etat détaillé de la réalisation des fonctionnalité principales*

Fonctionnalité	Etat	Taux de réalisation	Complexité	Notes d'implémentation
Authentification	Implémentée	90 %	Faible	JWT avec expiration configurable
Gestion des utilisateurs	Implémentée	90 %	Moyenne	Interface admin complète, gestion des rôles
Messagerie instantanée	Implémentée	75 %	Elevée	Socket.IO avec transfert de fichiers
Gestion des annonces	Implémentée	90 %	Moyenne	Ciblage multi-niveaux, pièces jointes
Gestion des cours	Implémentée	90 %	Moyenne	Uploadé multiples, validation formats
Système de note	Implémentée	85 %	Moyenne	Calcul automatique, historisation
Interface responsive	Implémentée	75 %	Elevée	Compatible mobile, tablette, desktop

4.3.2 Analyse des écarts fonctionnels

Malgré un taux de réalisation global élevée, il y a quelque écart par rapport aux spécification initiales qui ont été identifier :

- Fonctionnalité de gestion des emplois du temps : reportée à une version ultérieure pour des raisons de complexité technique
- Export avancé des données : limitation aux format basique csv, PDF

Tableau 4.04 : *Analyse des écarts fonctionnels et leur impact*

Fonctionnalité manquante	Impact utilisateur	Solution alternative	Planning de correction
Gestion des emplois du temps	Elevée	-	-
Export donnée avancé	Moyen	Exports basique disponibles	Dans 3 mois

4.4 Résultats de l'évaluation technique

L'évaluation technique s'est concentrée sur les aspects performance, sécurité, maintenabilité et évolutivité de la plateforme.

4.4.1 Test de performance

Les tests de performance ont été conduits dans un environnement reflétant les conditions réelles d'utilisation avec une charge graduellement croissante

Tableau 4.05 : *Résultats détaillés des tests de performance*

Métrique	Valeur	Objectif	Seuil d'acceptation	Statut	Observation
Temps de réponse moyen	187ms	< 500ms	< 1000ms	Atteint	Excellente réactivité
Utilisateurs simultanés	512	300	200	Dépassé	Bonne scalabilité
Taux d'erreurs	0.18%	< 1%	<5%	Atteint	Très faible taux d'erreur
Temps de chargement de page	1.8 s	< 3s	< 5s	Atteint	Optimisation correcte
Disponibilité	99.82%	> 99.5%	> 99%	Atteint	Haute disponibilité
Utilisation CPU	68% pic	< 80%	<90%	Atteint	Marge de manœuvre

4.4.2 Evaluation de sécurité

L'évaluation de sécurité a suivi la méthodologie OWASP [4.06] et incluait des tests manuels

Tableau 4.06 : *Résultats de l'audit de sécurité*

Catégorie de risque	Nombre de vulnérabilité	Niveau de risque	Statut	Mesure correctives
Critique	0	Aucun	Conforme	
Élevée	2	Faible	A modifier	Validation renforcée
Moyen	5	Acceptable	Conforme	Surveillance continue
Faible	12	Négligeable	Accepté	Aucune actions

Les principaux points forts identifiés en matière de sécurité :

- Implémentation robuste de l'authentification JWT
- Hachage des mots de passe avec bcrypt
- Protection contre les injections SQL
- Validation stricte des types de fichiers uploadés

4.4.3 Qualité du code et maintenabilité

L'analyse statique du code a été réalisée à l'aide d'outils spécialisés pour évaluer la maintenabilité à long terme

Tableau 4.07 : *Métrique de qualité du code*

Métrique	Valeur	Seuil acceptable	Interprétation
Complexité cyclomatique moyenne	8.2	<15	Bonne maintenabilité
Taux de duplication du code	3.1%	<5%	Très faible duplication
Couverture des tests	78%	>70%	Couverture satisfaisante
Dettes techniques	12 jours	<30 jours	Niveau acceptable
Violations des règles de codage	45	>100	Qualité de code bonne

4.5 Evaluation de l'expérience utilisateur

L’évaluation de l'expérience utilisateur a constitué un volet essentiel de notre démarche, visant à mesurer l'utilisabilité, l'utilité perçue et la satisfaction globale.

4.5.1 Résultats quantitatifs

Tableau 4.08 : *Résultats détaillés du questionnaire de satisfaction sur une échelle de 1 à 5*

Critère d’évaluation	Étudiants (moyenne)	Enseignants (moyenne)	Administrateurs (moyenne)	Moyenne générale	Écart-type
Facilité d’utilisation	4.2	3.8	4.5	4.2	0.35
Performance perçue	4.0	4.2	4.3	4.2	0.15
Utilité perçue	4.5	4.3	4.7	4.5	0.20
Design de l'interface	4.1	3.9	4.2	4.1	0.15
Fonctionnalités communication	4.7	4.5	4.3	4.5	0.20
Gestion des documents	4.3	4.4	4.6	4.4	0.15
Navigation générale	4.0	3.7	4.4	4.0	0.35

Le score SUS (System Usability Scale) moyen calculé est de 82.5, ce qui place UniSphere dans la catégorie "Excellent" selon l'échelle de notation SUS [4.02]

4.5.2 Analyse qualitative des retours utilisateurs

Les entretiens et observations ont permis de recueillir des retours riches et détaillés sur l'expérience utilisateur

Points forts fréquemment mentionnées :

- "La centralisation de toutes les ressources pédagogiques au même endroit est un gain de temps considérable" (Enseignant, 15 ans d'expérience)
- "L'interface moderne et épurée facilite la prise en main, même pour les moins technophiles" (Étudiant, première année)

- "La messagerie intégrée permet des échanges rapides sans avoir à basculer entre différentes applications" (Administrateur)

Points d'amélioration identifier :

- "Sur mobile, certains boutons sont trop petits et difficiles à cliquer précisément" (Retour fréquent)
- "L'absence de notifications push hors ligne limite l'immédiateté des communications" (Besoin exprimé)
- "La courbe d'apprentissage pour les fonctionnalités avancées pourrait être plus progressive" (Enseignant)

4.5.3 Analyse des tâches utilisateurs

L'observation des utilisateurs lors de la réalisation de tâches spécifiques a permis d'identifier les points de friction dans l'interface

Tableau 4.09 : *Temps de réalisation des tâches principales (en secondes)*

Tâche	Expert	Novice	Écart	Difficulté perçue (1-5)
Se connecter	8.2	12.5	4.3	1.2
Consulter un cours	5.3	15.8	10.5	1.8
Envoyer un message	12.1	28.4	16.3	2.4
Uploader un fichier	15.7	42.3	26.6	3.1
Créer une annonce	25.4	68.9	43.5	3.8

4.6 Comparaison avec les plateformes existantes

Une analyse comparative approfondie a été menée pour positionner UniSphere par rapport aux solutions dominantes du marché.

4.6.1 Méthodologie de comparaison

La comparaison s'est appuyée sur une grille d'évaluation comprenant 20 critères répartis en 5 catégories principales, chaque critère étant noté sur 5 points.

Tableau 4.10 : *Grille d'évaluation comparative détaillée*

Catégorie	Critère	Pondération	UniSphere	Moodle	Google Classroom
Fonctionnalités	Gestion des cours	5%	4.5	5	4
Fonctionnalités	Communication	5%	5	3	3.5
Fonctionnalités	Évaluation	5%	4	5	4.5
Technique	Performance	5%	4.5	3.5	4.5
Technique	Sécurité	5%	4	4.5	4.5
Utilisabilité	Interface	5%	4.5	3	4.5
Utilisabilité	Accessibilité	5%	4	3.5	4
Économique	Coût total	5%	5	4	3.5
Économique	Maintenance	5%	4	3	5
Stratégique	Flexibilité	5%	5	4.5	3

Score total sur 100 | 100% | 86.5 | 78.0 | 81.5

4.6.2 Analyse des avantages compétitifs

Points forts d'UniSphere :

- Communication intégrée : solution tout-en-un contrairement aux alternatives
- Flexibilité et personnalisation : architecture modulaire permettant des adaptations poussées
- Coût total de possession : solution open source sans frais de licence

Points faibles relatifs :

- Écosystème de plugins : moins étendu que Moodle
- Intégration écosystémique : moins poussée que Google Classroom
- Maturité : solution nouvelle face à des acteurs établis

4.7 Analyse des limites et contraintes

Une analyse approfondie des limitations identifiées durant les tests permet d'anticiper les challenges pour le déploiement à grande échelle

4.7.1 Limitations techniques

Tableau 4.11 : *Analyse détaillée des limitations techniques*

Limitation	Impact	Gravité	Cause racine	Solution temporaire	Solution long terme
Dépendance Internet	Accès impossible hors ligne	Élevée	Architecture web classique	Aucune	Application mobile offline (V2)
Compatibilité IE	Non supporté	Moyenne	Technologies modernes	Message d'orientation	Pas de support planifié
Stockage fichiers	Limite espace disque	Moyenne	Architecture monolithique	Politique de rétention	Stockage cloud scalable
Performance mobile	Interface moins fluide	Faible	Optimisations insuffisantes		Optimisations CSS avancées

4.7.2 Limitations fonctionnelles

L'analyse des limitations fonctionnelles révèle des écarts entre les attentes utilisateurs et les fonctionnalités actuelles

Principales limitations identifiées :

- Absence de calendrier intégré nécessitant l'utilisation d'outils externes
- Pas de système de visioconférence natif
- Gestion des versions de documents limitée
- Reporting et analytics basiques
- Personnalisation de l'interface utilisateur restreinte

4.7.3 Challenges organisationnels

Le déploiement d'UniSphere soulève également des défis organisationnels qui nécessitent une attention particulière.

Tableau 4.12 : *Analyse des challenges organisationnels*

Challenge	Impact potentiel	Stratégie d'atténuation	Responsable
Résistance au changement	Adoption lente	Programme de formation et accompagnement	Direction pédagogique
Besoin de formation	Sous-utilisation des fonctionnalités	Guides utilisateurs, tutoriels vidéo	Équipe support
Charge administrative	Surcharge équipe IT	Automatisation des processus	DSI
Maintenance continue	Dégradation performance	Plan de maintenance préventive	Equipe technique

4.8 Plan d'amélioration et feuille de route

Sur la base des résultats de l'évaluation, un plan d'amélioration structuré a été élaboré pour guider les évolutions futures de la plateforme.

4.8.1 Court terme : 0-6 mois

Tableau 4.13 : *Plan d'amélioration à court terme*

Amélioration	Priorité	Complexité	Impact	Ressources nécessaires	Échéance
Notifications push [4.19]	Élevée	Moyenne	Élevé	1 développeur, 3 semaines	3 mois
Optimisation mobile	Élevée	Faible	Moyen	1 développeur, 2 semaines	2 mois
Mode sombre	Moyenne	Faible	Faible	1 développeur, 1 semaines	4 mois
Recherche avancée	Moyenne	Moyenne	Moyen	1 développeur, 4 semaines	6 mois

4.8.2 Moyen terme :6-18 mois

Les améliorations à moyen terme visent à combler les lacunes fonctionnelles identifiées et à enrichir l'écosystème de la plateforme.

Améliorations planifiées :

- Application mobile native (React Native)
- Module de visioconférence intégré (WebRTC)
- Système de quiz interactifs avancé
- Tableau de bord Analytics pour enseignants
- Intégration calendrier avec synchronisation

4.8.3 Long terme : +18 mois

La vision long terme s'oriente vers des fonctionnalités innovantes positionnant UniSphere comme une plateforme d'apprentissage nouvelle génération.

Axes stratégiques :

- Intelligence artificielle pour la recommandation personnalisée [4.09]
- Adaptive Learning avec parcours individualisés
- Blockchain pour la certification des compétences

4.9 Impact et retombée potentielles

L'analyse d'impact permet de quantifier les bénéfices attendus du déploiement d'UniSphere dans un établissement d'enseignement supérieur.

4.9.1 Impact pédagogique

Pour les étudiants :

- Accès unifié à l'ensemble des ressources pédagogiques [4.03]
- Autonomie accrue dans le suivi de leur progression académique
- Interaction facilitée avec les enseignants et pairs
- Réduction de la fracture numérique grâce à l'accessibilité multiplateforme

Pour les enseignants :

- Gain de temps significatif dans la gestion des cours [4.10]
- Outils d'évaluation plus efficaces et diversifiés
- Meilleur suivi de l'engagement et de la progression des étudiants

- Flexibilité accrue dans la création et distribution des contenus

4.9.2 Impact organisationnel

Pour l'administration :

- Rationalisation des coûts (réduction des licences logicielles)
- Uniformisation des processus académiques
- Meilleure traçabilité des activités pédagogiques
- Indépendance stratégique vis-à-vis des éditeurs propriétaires

Tableau 4.14 : *Analyse des coûts bénéfices simplifiée*

Poste	Coût/année (solution actuelle)	Coût/année (UniSphere)	Économie	Commentaires
Licences logicielles	75 000 000 AR (25 AR/an)	0 AR	75 000 000 AR	Élimination des coûts de licence Moodle/Blackboard
Maintenance infrastructure	40 000 000 AR (13.3 AR/an)	25 000 000 AR (8.3 AR /an)	15 000 000 AR	Réduction des besoins de maintenance
Formation utilisateurs	60 000 000 AR (20 AR /an	40 000 000 AR (13.3 AR /an	20 000 000 AR	Internalisation partielle du support
Support technique	25 000 000 AR (ponctuel)	35 000 000 AR (investissement initial)	-10 000 000 AR	Investissement formation accru initial
Total	200 000 000 AR	100 000 000 AR	100 000 000 AR	Économie de 50%

4.9.3 Impact sur l'écosystème éducatif

Au-delà de l'établissement, UniSphere contribue à :

- Démocratiser l'accès aux outils numériques éducatifs avancés
- Favoriser l'innovation pédagogique par des fonctionnalités adaptées
- Créer un écosystème open source autour des technologies éducatives
- Réduire la dépendance aux solutions propriétaires internationales

4.10 Perspectives de valorisation et déploiement

Les résultats positifs de l'évaluation ouvrent la voie à plusieurs scénarios de valorisation et de déploiement à plus large échelle

4.10.1 Modèles de déploiement envisageables

Tableau 4.15 : *Analyse des modèles de déploiement possibles*

Modèle	**Avantages**	**Inconvénients**	**Public cible**	**Potentiel économique**
Open source communautaire	Adoption large, contributions externes	Financement incertain	Universités, développeurs	Faible (prestige)
SaaS (Abonnement)	Récurrent revenue, scalabilité	Concurrence forte	Établissements moyens	Élevé
Modèle hybride	Best of both worlds	Complexité de gestion	Marché segmenté	Moyen à élevé
Partenariats institutionnels	Stabilité, impact	Dépendance aux partenaires	Réseaux universitaires	Variable

4.10.2 Stratégie de contribution communautaire

Pour favoriser l'adoption et l'amélioration continue, une stratégie de contribution communautaire structurée est proposée :

- Documentation complète traduite en plusieurs langues
- API ouverte pour les développeurs tiers
- Programme de contributions avec reconnaissance des contributeurs
- Forums et communauté d'entraide et d'échange

4.10.3 Perspectives de recherche et développement

La plateforme UniSphere ouvre également des perspectives pour la recherche dans plusieurs domaines :

- Analyse des données d'apprentissage (Learning Analytics)

- Personnalisation des parcours pédagogiques
- Évaluation de l'impact des technologies sur l'apprentissage
- Interopérabilité entre systèmes éducatifs

4.11 Conclusion

En conclusion, l'évaluation exhaustive menée dans ce chapitre démontre qu'UniSphere atteint ses objectifs principaux avec un niveau de qualité satisfaisant. Dans ce chapitre nous avons pu démontrer que la plateforme répond aux besoins identifiés initialement et présente même des avantages compétitifs significatifs par rapport aux solutions existantes.

CONCLUSION GENERALE

En somme, les avancés technologiques dans le monde ont profondément transformé le secteur éducatif notamment dans les universités. Ces avancés ont faite naitre diverses plateformes très utiliser par les universités dans la gestion des ressources pédagogiques et la communication. Cependant celle-ci ne répondent pas toujours aux besoins de certaine universités. D'où notre projet « UniSphere » qui se présente comme solution simple et concrète pour ces universités. En adaptant les processus de gestion académique et en facilitant les communications entres les acteurs de l'université, notre projet renforce l'efficacité pédagogiques, tout en améliorant la gestion administrative de l'université et la communication entre ces acteurs.

A travers cette ouvrage diviser en quatre chapitres, nous avons détailles les démarches nécessaires de la conception à la réalisation de cette plateforme. Dans le premier chapitre, nous avons présenté le contexte général du projet, repérer les limites des systèmes déjà existants, déterminer les principaux objectifs et fonctionnalités de l'UniSphere. Cette étude a confirmé le besoin d'une solution intégré et intuitive.

Le deuxième chapitre a été dédié à l'analyse et à la conception de notre à l'aide du langage de modélisation UML. Nous avons pu présenter les fonctionnalités à implémentées de notre plateforme avec les diagrammes de cas d'utilisation, de séquence, d'activité et de classe.

Le troisièmes chapitre a été consacré à la phase de réalisation. Dans ce chapitre nous présentons les langages de programmations, les frameworks et les outils essentiels pour la création de notre plateforme UniSphere. Afin que notre plateforme soit performante et sécurisé, nous avons employé les technologies moderne comme Vue.JS, Socket.IO et MySQL. Les interfaces graphiques présenter montrent nos intentions à offrir une expérience utilisateur accessible, et fluide. Ce qui nous offre une compréhension cohérente des interactions entre les acteurs et le système

Enfin, le quatrième chapitre, nous permis d'évaluer la plateforme sur divers points. Les résultats obtenus sont excellents, notre plateforme à atteint les objectifs principaux, répond aux exigences des utilisateurs, facile à maintenir et peut rivaliser aux solutions existantes.

Bien que l'UniSphere soit déjà fonctionnelle et remplisse ses objectifs principaux, il présente encore beaucoup d'amélioration possibles tels que la création d'une version mobile native de la plateforme pour les étudiants et les enseignants ou une version bureau pour les administrateurs montrent de nombreuses possibilités pour notre plateforme.

ANNEXES

Annexe 1 : Présentation de l'Athénée Saint Joseph Antsirabe

A1.1 Historique

L'Athénée Saint Joseph Antsirabe a été créée en 2000 par la congrégation des Pères du Sacré Cœur de Jésus (Dehoniens). Elle se donne pour objectifs de fournir aux étudiants la réactivité, les capacités et l'esprit d'entreprise essentiels pour faire face au monde professionnel. Cette université est située à 11 Km au Nord de la ville d'Antsirabe, sur la Route Nationale 7. Ses 1449 étudiants viennent actuellement des différentes régions de Madagascar et des îles sœurs, les Comores.

Les infrastructure permettent aux différentes mentions d'approfondir leurs spécialités et de se documenter : un laboratoire de physico-chimique, un laboratoire de microbiologie, un laboratoire d'analyse sensorielle, un laboratoire de recherche, un hall technologie, un laboratoire destiné aux Géosciences et Environnements, des laboratoires informatiques, 02 laboratoires de langue, un centre de documentation et d'information avec connexion WIFI, un amphithéâtre pouvant accueillir 1000 personnes et une chapelle.

Les étudiants peuvent également poursuivre leur cursus à l'étranger. L'université s'ouvre à l'internationalisation à travers divers partenariats : l'université de Paris, l'Agro campus à Rennes, l'université de la Réunion.

Figure A1.01 : *Vue aérienne de l'ASJA*

A1.2 Formation

L'université comprend deux domaines d'études séparés, les sciences et technologies et les Sciences de la Société. Ces domaines regroupent plusieurs Mentions qui sont : Droit, Economie et commerce, Sciences Agronomiques, Sciences de la Terre, Informatique, Langues et Cultures. (Cf. Formation Licence ASJA)

A1.3 Système LMD à l'ASJA

La nouvelle architecture retenue pour l'Enseignement Supérieur est articulée selon trois paliers de formation correspondant chacun à un diplôme :

- Le niveau LICENCE, correspond à un cycle de formation de trois années après le baccalauréat. La licence professionnelle permet aux étudiants qui le souhaitent d'acquérir rapidement une qualification professionnelle répondant à des besoins et à des métiers clairement identifiés. L'année de formation articule enseignement théoriques et pratiques, apprentissage de méthodes et d'outils, stage en milieu professionnel de 6 à 8 semaines ;
- Le niveau MASTER, correspond à deux ans supplémentaires après le niveau Licence. Le diplôme de Master vise à permettre aux Universités d'organiser les études entre le grade de Licence et le grade de Master dans le cadre d'un cursus débouchant sur un nouveau diplôme national ;
- Le niveau DOCTORAT, correspondant à trois années supplémentaires après le niveau Master.

Ce nouveau système vise à rendre plus lisible les offres de formation de chaque établissement, adoptant des niveaux et des appellations universelles pour les diplômes

Il permet d'accroître ainsi la fiabilité et la transférabilité des diplômes délivrés par l'université ASJA et faciliter ainsi la mobilité de nos étudiants.

A1.4 L'organigramme hiérarchique

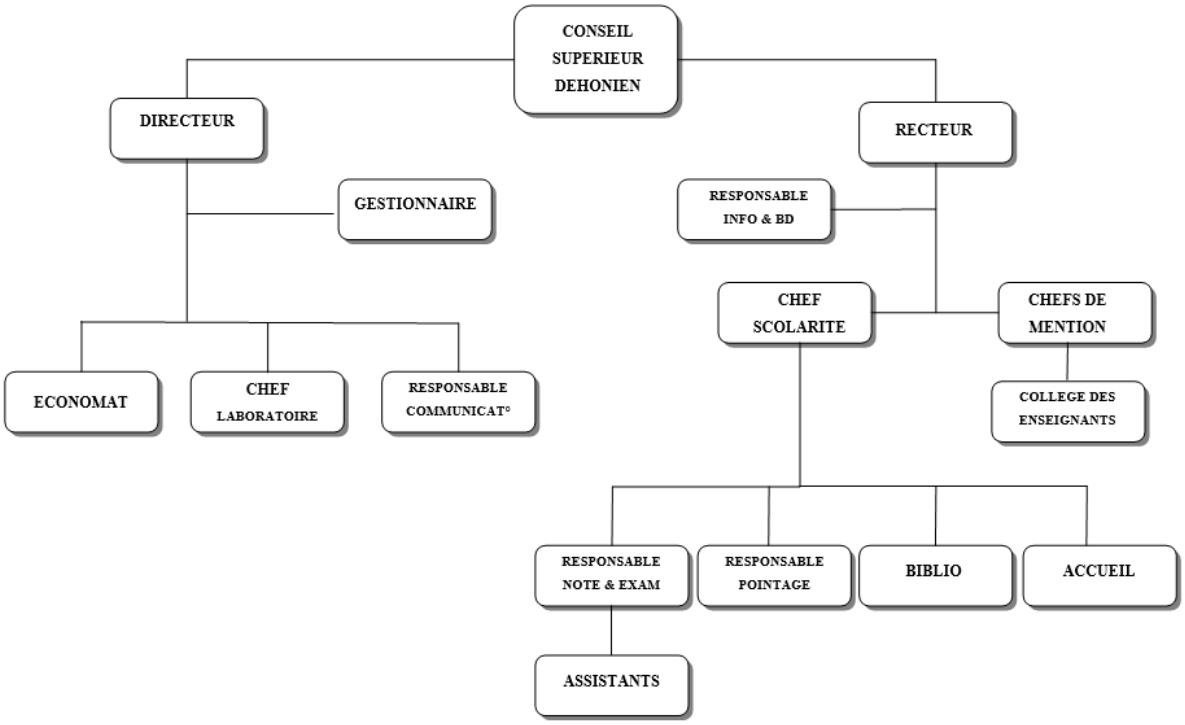

Figure A1.02 : *Organigramme hiérarchique de l'ASJA*

Le Conseil Supérieur est l'organe hiérarchique supérieur de l'ASJA. Il est constitué par le président du Conseil supérieur, le Directeur Général et 3 conseillers, minimum.

Le président du Conseil Supérieur de l'ASJA est l'organe individuel du gouvernement de l'ASJA.

Le Directeur Général qui représente le président du Conseil Supérieur à l'ASJA représente également la Congrégation du Sacré Cœur de Jésus in loco.

Le Recteur de l'ASJA dirige et organise la vie ordinaire et pratique de l'université. Il est le représentant légal de l'ASJA auprès du Ministère de l'Enseignement Supérieur. Il est assisté par les six (6) chefs de Mentions.

Le Conseil Scientifique et le Conseil d'Administration sont des organes d'assistance et d'exécution technique, d'après leur domaine de travail : l'organisation scientifique- pédagogique et la gestion financière-administrative. Le Conseil scientifique fonctionne aussi comme Conseil de Discipline.

Le Conseil scientifique est composé du Recteur, des chefs de Mention, de certains enseignants expérimentés et des professionnels, du Directeur Général de l'ASJA, d'un représentant du Conseil Supérieur, du chef de scolarité et des bénévoles de FIDESCO.

Le Conseil d'Administration est constitué d'un représentant du Président du Conseil Supérieur, du Directeur Général, du Recteur, de l'Econome Régional des Prêtres du Sacré Cœur de Jésus, d'un Gestionnaire (Coopérant FIDESCO), d'une commission restreinte économique nommée par le Supérieur Régional.

Annexe 2 : EXTRAIT DE CODE

A2.1 Extrait de code du backend

```
1  import axios from "axios";
2
3  // URL de base du backend
4  const API_BASE_URL = "http://localhost:8000/api"; // adapter si nécessaire
5
6  // Création d'une instance Axios
7  const api = axios.create({
8    baseURL: API_BASE_URL,
9    timeout: 10000, // 10s
10   headers: {
11     "Content-Type": "application/json",
12   },
13 });
```

Figure A2.01 : *Extrait code Connexion au backend*

```
1  const initSockets = (server) => {
2    io = new Server(server, {
3      cors: {
4        origin: "http://localhost:5173",
5        methods: ["GET", "POST"],
6        credentials: true,
7      },
8    });
9
10   io.on("connection", (socket) => {
11     console.log("🔌 Client connected:", socket.id);
12
13     // Importer les événements conversationnels
14     conversationSockets(socket, io);
15     messageSockets(socket, io);
16
17     socket.on("disconnect", () => {
18       console.log("🔌 Client disconnected:", socket.id);
19     });
20   });
21 };
```

Figure A2.02 : *Extrait code Initialisation du socket*

A2.2 Extrait de code frontend

```vue
import { ref, computed, onMounted } from "vue";
import { useRoute, useRouter } from "vue-router";
import { useAuthStore } from "../stores/auth.store";

const isSidebarOpen = ref(false);
const route = useRoute();
const router = useRouter();
const authStore = useAuthStore();

// Récupérer les informations de l'utilisateur
const user = computed(() => authStore.user);
const userName = computed(
  () => user.value?.email || user.value?.username || "Utilisateur"
);

const userInitial = computed(() => userName.value.charAt(0).toUpperCase());

const loadUser = async () => {
  if (authStore.token && !authStore.user) {
    try {
      await authStore.fetchUserData();
    } catch (error) {
      console.error("Failed to fetch user data");
    }
  }
};

onMounted(() => {
  loadUser();
});
```

Figure A2.03 : *Extrait code student.Layout*

```vue
import { ref, computed, watch } from "vue";
import { useRouter } from "vue-router";
import { useAuthStore } from "../../stores/auth.store";

const router = useRouter();
const authStore = useAuthStore();

const loginData = ref({ email: "", password: "" });

const showPassword = ref(false);
const hasError = ref(false);

// Effacer l'erreur quand l'utilisateur commence à taper
const clearError = () => {
  if (authStore.error) {
    authStore.error = null;
  }
  hasError.value = false;
};

// Validation du formulaire
const isFormValid = computed(() => {
  return loginData.value.email && loginData.value.password;
});

```

Figure A2.04 : *Extrait code Login*

REFERENCES

[1.01] https://www.demos.fr/blog/quest-ce-que-la-formation/, Qu'est-ce que la formation consulter le 31 Juillet 2025

[1.02] https://www.questionpro.com/blog/fr/quest-ce-que-la-recherche/, Qu'est-ce que la recherche ? consulter le 31 Juillet 2025

[1.03] https://www.ekole.fr/blog/la-communication-pedagogique-un-pilier-de-leducation-moderne, Communication Pédagogique, consulté le 31 Juillet

[1.04] https://oraprdnt.uqtr.uquebec.ca/portail/gscw031?owa_no_site=305, FAQ Chantier institutionnel sur les structures de gestion académique, consulter le 31 Juillet 2025

[1.05] https://www.insee.fr/fr/metadonnees/definition/c1525, Définition Etudiants | Insee, consulter le 31 Juillet 2025

[1.06] https://inee.org/fr/glossaire-ESU/enseignant-enseignante, Enseignant/Enseignante, consulté le 31 juillet 2025

[1.07] https://francecarriere.fr/metier/secretaire-dadministration-scolaire-et-universitaire-sasu, Secrétaire d'administration scolaire et universitaire, consulter le 31 Juillet 2025

[1.08] https://docs.moodle.org/4x/fr/%C3%80_propos_de_Moodle, A propos de Moodle, consulté le 31 juillet 2025

[1.09] https://help.blackboard.com/frfr/Learn/Administrator/Hosting, Découvrez Blackboard Learn, consulter le 31 Juillet 2025

[1.10] https://edu.google.com/workspace-for-education/products/classroom/, Classroom Management Tools & Ressources - Google for Education, consulté le 31 Juillet 2025

[2.01] RAKOTONIRINA Andriamitantsoa Soloarinala, « UML », cours 3ème année, parcours développement d'application Informatique, ASJA, A.U. :2022-2023, consulté le 06 Aout 2025

[2.02] https://www.techniques-ingenieur.fr/base-documentaire/technologies-de-l-information-th9/genie-logiciel-42306210/langage-uml-developpement-de-logiciel-et-modelisation-visuelle-h3238/origine-et-objectifs-d-uml-h3238niv10002.html, Langage UML développement de logiciel et modélisation visuel, consulter le 06 Aout 2025

[2.03] https://www.lucidchart.com/pages/fr/langage-uml, Qu'est-ce que le langage UML, consulté le 06 Aout 2025

[2.04] Hugues Bersini, « L'orienté objet », Eyrolles 3ème édition, page 160 consulté le 11 Aout 2025

[2.05] https://www.futura-sciences.com/tech/definitions/informatique-uml-3979/, Définition UML, consulté le 06 Aout 2025

[2.06] https://www.cybermedian.com/fr/unified-modeling-language-uml-introduction/, Présentation du langage de modélisation unifié (UML), consulté le 08 Aout 2025

[2.07] https://miro.com/fr/diagramme/qu-est-ce-qu-un-diagramme-uml/, Qu'est-ce qu'un diagramme UML et à quoi sert - il, consulté le 11 Aout 2025

[2.08] https://www.lucidchart.com/blog/fr/types-de-diagrammes-UML, Les types de diagrammes UML, consulté le 09 Aout 2025

[2.09] https://guides.visual-paradigm.com/understanding-profile-diagrams-in-uml-a-comprehensive-guide, Understanding Profile Diagrams in UML, consulter le 13 Août 2025

[2.10] https://gitmind.com/fr/diagramme-communication-uml.html, Diagramme de Communication UML, consulté le 13 Aout 2025

[3.01] https://developer.mozilla.org/fr/docs/Web/HTML, HTML (HyperText Markup Language), consulter le 20 Septembre 2025

[3.02] https://www.elephorm.com/formation/code-data/xhtmlcss/apprendre-html-5-les-fondamentaux/levolution-du-html-de-sa-creation-aujourdhu, L'évolution de l'HTML : De sa création à aujourd'hui, consulter le 20 Septembre 2025

[3.03] https://developer.mozilla.org/fr/docs/Web/CSS, CSS : Feuille de styles en cascade, consulter le 20 Septembre 2025

[3.04] https://developer.mozilla.org/fr/docs/Web/JavaScript, JavaScript | MDN, consulter le 20 Septembre 2025

[3.05] https://www.bocasay.com/fr/histoire-evolution-langage-javascript/, Histoire et évolution du langage JavaScript, consulter le 20 Septembre 2025

[3.06] https://v2.fr.vuejs.org/v2/guide/, Introduction Vue, consulter le 20 Septembre 2025

[3.07] https://worldline.github.io/vuejs-training/fr/presentation/, Présentation de vue.js, consulter le 20 Septembre 2025

[3.08] https://kinsta.com/fr/base-de-connaissances/qu-est-express-js/, Qu'est-ce que Express JS, consulter le 20 Septembre 2025

[3.09] https://lixtec.fr/socket-io-ou-websockets/, Socket ou WebSockets, consulter le 20 septembre 2025

[3.10] https://kinsta.com/fr/blog/tailwind-css/, Comment utiliser Tailwind CSS pour développer rapidement de superbes sites web, consulter le 20 Septembre 2025

[3.11] https://www.numendo.com/blog/framework/tailwind-css-framework-totalement-personnalisable/, Tailwind CSS le framework totalement personnalisable, consulter le 20 Septembre 2025

[3.12] https://visualstudio.microsoft.com/fr/, Visual Studio : IDE et éditeur de code pour le développement logiciel, consulter le 20 Septembre 2025

[3.13] https://www3.technologyevaluation.com/fr/solutions/53729/visual-paradigm, Visual Paradigm, consulter le 20 Septembre 2025

[3.14] https://kinsta.com/fr/base-de-connaissances/base-de-connaissances-github/, Qu'est-ce que GitHub, consulter le 20 Septembre 2025

[3.15] https://www.nocodefactory.fr/definitions-lowcode/github, GitHub : définition de la plateforme de code, consulter le 20 Septembre 2025

[3.16] https://makina-corpus.com/front-end/introduction-nodejs, Introduction à Node JS, consulter le 20 Septembre 2025

[3.17] https://kinsta.com/fr/base-de-connaissances/qu-est-ce-que-node-js/, Qu'est-ce que le Node.JS et pourquoi l'utiliser, consulter le 20 septembre 2025

[3.18] https://kinsta.com/fr/base-de-connaissances/qu-est-ce-que-mysql/, Qu'est-ce que le MySQL, consulter le 20 Septembre 2025

[3.19] https://www.oracle.com/ca-fr/mysql/what-is-mysql/, MySQL présentation et utilisation, consulter le 20 Septembre 2025

[4.01] Sommerville, I. (2016). *Software Engineering* (10th ed.). Pearson., consulter le 24 Septembre 2025

[4.02] Brooke, J. (1996). SUS: A quick and dirty usability scale. *Usability Evaluation in Industry*, 189-194. Consulter le 24 Septembre 2025

[4.03] https://en.unesco.org/themes/ict-education, UNESCO (2023). *ICT in Education,* consulter le 24 Septembre 2025

[4.04] Patton, R. (2005). *Software Testing* (2nd ed.). Sams Publishing. pp. 45-78, consulter le 24 Septembre 2025

[4.05] Croll, A., & Power, S. (2009). *Complete Web Monitoring*. O'Reilly, consulter le 24 Septembre 2025

[4.06] https://owasp.org/www-project-application-security-verification-standard/, OWASP (2023). *Application Security Vérification Standard,* consulter le 24 Septembre 2025

[4.07] Nielsen, J. (1994). *Usability Engineering*. Morgan Kaufmann, consulter le 24 Septembre 2025

[4.08] ISO 9241-210:2019. Ergonomics of human-system interaction — Part 210: Human-centered design for interactive systems, consulter le 24 Septembre 2025

[4.09] Siemens, G. (2005). *Connectivism: A Learning Theory for the Digital Age*. International Journal of Instructional Technology and Distance Learning, consulter le 24 Septembre 2025

[4.10] Bates, A. W. (2015). *Teaching in a Digital Age*. BCcampus, consulter le 24 Septembre 2025

FICHE DE RENSEIGNEMENTS

Nom :

Prénoms :

Adresse de l'auteur :

Téléphone :

E-mail :

Titre du mémoire : Conception et réalisation d'une plateforme universitaire

Nombre de pages : 80 pages

Nombre de tableaux : 30 tableaux

Nombre de figures : 51 figures

Directeur de mémoire :

Grade :

Téléphone :

Mail :

FAMINTINANA

Ity tetik'asa ity dia fanamboarana ny sehatra an-tserasera ho an'ny oniversite anstoina hoe "UniSphere". Izany dia sehatra fitantanana izay ahafahan'ny oniversite mitantana tsara kokoa ny loharanon-pampianarana sy ireo mpikambana, mora ny fidirana amin'ny loharanon-pampianarana ary mampitokana ny fifandraisana amin'ny alalan'ny rafi-pandefasana hafatra azo antoka. Ny "UniSphere" dia natao tamin'ny fampiasana ny "UML" izay nahafahana hazava ny fifampidinihan'ny rafi-pandrindana sy ny mpampiasa. Rehefa nanao izany tetik'asa izany dia nampiasa ny fitendrehana "JavaScript" miaraka amin'ny "framework Vue JS" ho an'ny ampahany "frontend", "Node JS" sy "Express" ho an'ny ampahany "backend", ary "MySQL" ho an'ny tahirin-kevitra. Ny tolotra fifandraisana dia mampiasa ny "Socket.IO" mba hahazoana valiny avy hatrany.

Teny misongadina: **Oniversite, sehatra, UniSphere, JavaScript, Vue JS, Node JS**

RESUME

Notre projet est le développement d'une plateforme universitaire intitulé « UniSphere ». C'est une plateforme qui permet aux universités de mieux gérer leurs ressources pédagogiques et ces membres, faciliter l'accès au ressource éducatifs et centraliser la communication avec sa messagerie sécuriser. L'UniSphere a été conçu en utilisant le langage de modélisation UML qui nous permis d'éclaircir les interactions entre le système et les acteurs. Lors de la réalisation du projet nous avons employés le langage de programmation JavaScript avec le framework Vue JS pour la partie frontend, Node JS et Express pour le backend et MySQL pour la base de données. Sa fonctionnalité de messagerie utilise le Socket.IO pour une réponse instantanée.

Mots clés : Université, Plateforme, UniSphere, JavaScript, Vue JS, Node JS

ABSTRACT

Our project is the development of a university platform called "UniSphere." It is a university management platform which allows universities to better manage their educational resources and members, facilitates access to educational resources and centralizes communication with its secure messaging system. UniSphere was designed using the UML modeling language, which allows us to clarify the interactions between the system and its users. This project was carried out using the JavaScript language with Vue JS framework for the frontend, Node JS and Express for the backend, and MySQL for the database. Its messaging feature uses Socket.IO for instant response.

Keywords: **University, Platform, UniSphere, JavaScript, Vue JS, Node JS**