REPUBLIQUE DEMOCRATIQUE SENEGALAISE

****

Université Assane Seck de Ziguinchor UFR Sciences et Technologie
Département Informatique

MEMOIRE DE FIN D'ETUDE POUR L'OBTENTION DU DIPLOME DE MASTER

Mention : Informatique, Spécialité : Génie Logiciel

Thème : Conception et mise en place d'une plateforme web pour l'exploitation de l'environnement de gestion des stages hospitaliers de l'UFR des sciences de la Santé de l'UASZ

Soutenu le 25/06/2024

Présenté par :

M. Alassane NDIAYE

Sous la supervision de **:**

Pr Ousmane DIALLO

Sous la direction de **:**

Dr El hadji Malick NDOYE

Pr Ousmane DIALLO

M. Diéré DIEDHIOU

Membres du jury

Pr Ibrahima DIOP	Professeur Assimilé	Président, Examinateur	UASZ
M. Malaw NDIAYE	Assistant	Rapporteur	UASZ
Dr El hadj Malick NDOYE	Maitre conférences Assimilé	Encadreur	UASZ
Pr Ousmane DIALLO	Professeur Assimilé	Co-encadreur	UASZ

Année académique: 2022-2023

Résumé

Notre mémoire de master s'inscrit dans le cadre d'un stage au sein de l'Unité de Formation et de Recherche (UFR) des Sciences de la Santé (2S) de l'Université Assane Seck de Ziguinchor (UASZ).

Depuis le début des stages des étudiants en 2013, leur gestion a toujours été réalisée manuellement (format papier) par le service pédagogique. Avec l'augmentation continue du nombre d'étudiants, cette méthode devient de plus en plus laborieuse, entraînant des retards dans la publication des notes, la perte de documents tels que les fiches de présence et les notes de services, ainsi qu'un temps de recherche considérable pour accéder aux informations spécifiques à un étudiant.

Pour remédier à ces problèmes, notre projet propose l'automatisation de la gestion et du suivi des stages via le développement d'une plateforme web dédiée. L'objectif principal de ce mémoire est la conception et mise en place d'une plateforme web permettant une gestion efficace pour l'exploitation de l'environnement de gestion des stages hospitaliers de l'UFR 2S qui se base sur un module : le module de la gestion des étudiants, englobant tous les aspects liés aux étudiants : orientation dans les hôpitaux, spécialités et services, suivi de leur présence, notes obtenues, rotations entre spécialités et services, et génération des résultats.

Pour mener à bien ce projet, nous avons adopté une approche AGILE, spécifiquement la méthode SCRUM. La phase de spécification des besoins a impliqué l'identification des acteurs, la décomposition du module en sous-modules, et la définition des fonctionnalités spécifiques à chaque sous-module. Lors de la phase d'analyse des besoins et de conception, divers diagrammes ont été élaborés pour représenter le système sous différents angles.

Le développement du front-end utilise l'architecture Model-View-Controller (MVC), tandis que le back-end suit un modèle en couches. La notation Unified Modeling Language (UML) et l'outil PowerAMC ont été utilisés pour concevoir les diagrammes, et l'outil Git a facilité la collaboration et la gestion des versions.

Résumé

L'application est développée avec la technologie MERN (MySQL, Express, React, Node), les tests sont effectués avec l'outil Insomnia, et le design est réalisé en utilisant un template Bootstrap téléchargé en ligne et personnalisé selon nos besoins.

Mot clés : plateforme web dédiée ; le module de gestion des étudiants ; approche AGILE ; SCRUM ; Model-View-Controller (MVC) ; modèle en couches ; Unified Modeling Language (UML), MERN, Insomnia ; template Bootstrap.

Abstract

Our master's thesis is part of an internship within the Training and Research Unit (UFR) of Health Sciences (2S) at Assane Seck University of Ziguinchor (UASZ). Since the beginning of student internships in 2013, their management has always been handled manually (paper format) by the educational service. With the continuous increase in the number of students, this method has become increasingly laborious, leading to delays in grade publication, loss of documents such as attendance sheets and service notes, and considerable search time to access specific student information.

To address these issues, our project proposes the automation of internship management and tracking through the development of a dedicated web platform. The main objective of this thesis is to design a web platform for efficient management of the internship environment at UFR 2S. This platform will include a module: the student management module, encompassing all aspects related to students: hospital orientation, specialties and services, attendance tracking, grades obtained, rotations between specialties and services, and result generation.

To successfully carry out this project, we adopted an AGILE approach, specifically the SCRUM method. The requirements specification phase involved identifying stakeholders, breaking down modules into sub-modules, and defining the specific functionalities for each sub-module. During the requirements analysis and design phase, various diagrams were created to represent the system from different perspectives.

The front-end development uses the Model-View-Controller (MVC) architecture, while the back-end follows a layered model. Unified Modeling Language (UML) notation and the PowerAMC tool were used to design the diagrams, and the Git tool facilitated collaboration and version management.

The application is developed using the MERN stack (MySQL, Express, React, Node), tests are performed using the Postman tool, and the design is created using a Bootstrap template downloaded online and customized to our needs.

Abstract

Keywords: dedicated web platform; student management module; AGILE approach; SCRUM; Model-View-Controller (MVC); layered model; Unified Modeling Language (UML); MERN; Insomnia; Bootstrap template.

Dédicaces

Par la grâce d'Allah, le tout puissant, je dédie ce travail :

À mon père et ma mère

C'est grâce à vous que nous sommes présents en ce jour béni. Vous n'avez épargné aucun effort afin de nous voir évoluer dans des conditions exemplaires. Vous êtes et demeurez les plus remarquables des parents. Je souhaite sincèrement que ce travail témoigne de mon amour filial incommensurable à votre égard.

À mes frères et sœurs

Puissiez-vous demeurer solidaires et unis dans l'existence, tout en restant fidèles à l'instruction bienveillante inculquée par nos chers parents.

Que Dieu continue d'éclairer vos chemins.

À tous mes amis et camarades de promo

Vos soutiens, conseils et encouragements me tiennent à cœur.

À particulièrement mon grand frère Mory NDIAYE

Pour tout ce qu'il endure pour nous, que Dieu continue de guider tes pas, frère.

Remerciements

***T**out d'abord, j'exprime ma profonde reconnaissance envers **Allah** le Tout-Puissant pour m'avoir octroyé la force, la patience et le courage nécessaire afin de parvenir jusqu'à ce stade et accomplir cette tâche.*

***J**e souhaite exprimer toute ma gratitude envers mes encadreurs **Docteur El Hadji Malick NDOYE**, **Professeur Ousmane DIALLO** et mon Maître Stage **Monsieur Diéré DIEDHIOU** pour vos soutiens, vos disponibilités, encouragements et vos remarques extrêmement précieuses qui ont contribué à l'amélioration de la qualité de ce travail.*

***Je** tiens également à exprimer mes sincères remerciements aux membres du jury, à savoir **Monsieur Ibrahima DIOP, Monsieur Malaw NDIAYE,** enseignants-chercheurs distingués de l'Université Assane Seck de Ziguinchor. Je suis très reconnaissant de l'intérêt que vous avez manifesté à mon travail, ainsi que du temps précieux que vous avez bien voulu consacrer à son évaluation. Vos suggestions précieuses seront grandement appréciées.*

***Je** tiens à exprimer mes sincères remerciements à tous les travailleurs de la **D**irection du service pédagogique de l'urf santé de l'Université. Je souhaite témoigner ma gratitude pour votre aide et votre disponibilité tout au long de mon stage. Vous avez fait preuve d'une générosité sans faille en me fournissant toutes les informations essentielles qui m'ont permises d'accomplir ce travail avec succès.*

***Je** souhaite également exprimer mes sincères remerciements à mes chers amis pour vos encouragements et votre soutien indéfectible.*

***Mes** vifs remerciements s'adressent également à tous les enseignants du Département Informatique de l'Université Assane Seck de Ziguinchor pour la formation qu'ils ont eu le soin de nous donner.*

***Je** vous exprime ma gratitude la plus sincère, chers camarades de promotion, compagnons de chambre à savoir **Abdou Karim DRAME, Massier NDIAYE, Moussé NDIAYE** et toutes les personnes qui ont apporté leur contribution à l'élaboration méticuleuse de ce travail.*

Table des matières

Liste des figures

Liste des tableaux

Abréviations et sigles

CD : Livraison Continue.
CI : Intégration Continu.
UFR 2S : Unité de Formation et de Recherche des Sciences de la Santé.
SP : Service Pédagogique.
SA : Service Administratif.
SG : Service Généraux.
SF : Service Finance.
PGS : Projet de Gestion des Stages.
HTTP : HyperText Transfer Protocol (ou protocole de transfert hypertexte en français).
IaC : Infrastructure as Code.
IDE : Integrated Development Environment (Environnement de Développement Intégré en français).
PATS : Personnel Administratif, Techniques et de Service.
PER : Personnel d'Enseignement et de Recherche.
SGBD : Système de Gestion de Base de Données.
TDD : Test Driven Development (ou Développement Piloté par des Tests en français).
TIC : Technologies de l'Information et de la Communication.
UASZ : Université Assane Seck de Ziguinchor.
UML : Unified Modeling Langage, ou langage de modélisation unifié en français.
XP : Programmation Extrême.

Introduction générale

Dans le cadre de sa contribution à la formation des futurs diplômés, l'UFR 2S de l'Université Assane Seck de Ziguinchor, en collaboration avec les Centres Hospitaliers, amène régulièrement ses étudiants de médecine de la L2 en D3 en stage dans les services de ces hôpitaux tout au long de l'année.

Ainsi, pour une bonne gestion et suivi de ces stages, l'UFR met en place une collaboration avec le personnel de chaque hôpital. En principe, au démarrage des stages, le chef du service pédagogique envoie les listes des étudiants dans les hôpitaux par courrier. Pour passer dans le service, l'étudiant doit se rendre au niveau du service pédagogique pour obtenir une note de prise de service. Les notes obtenues pour chaque fin de stage sont envoyées sous format papier, de même que les listes de présence.

Cependant malgré son utilité, la gestion manuelle du déroulement de ces stages par le service pédagogique de l'UFR 2S pose plusieurs problèmes notamment la perte d'informations, la complexité de la répartition des stagiaires dans les différents hôpitaux, spécialités et services ainsi que la lourdeur du processus de suivi, etc.

Pour remédier à ces problèmes, le chef du service pédagogique a proposé lors d'un conseil d'UFR académique la mise en place d'une plateforme web. Après validation, le développement d'une application web, pour gérer de manière dématérialisée les stages des étudiants, a été entrepris.

Le projet de Gestion des Stages, composé de plusieurs modules : gestion des utilisateurs, gestion des hôpitaux, gestion des années académiques et gestion des étudiants, vise à automatiser la gestion des stages des étudiants afin de permettre au responsable pédagogique de répartir les étudiants dans les hôpitaux, dans les différentes spécialités, de faire la rotation après chaque semestre et services sans se déplacer. Il permet aux étudiants de connaitre leurs lieux de stage, la date (début du stage) et la durée (le nombre de semaine que va durer le stage). Cela permet également aux personnels des hôpitaux (maîtres de stage, secrétaires, chefs de départements) de travailler dans de meilleures conditions.

Il vise à recueillir et à suivre toutes les informations relatives aux stages des étudiants dans les centres hospitaliers, dématérialisant ainsi toute information relative aux stages des étudiants, notamment les sites de stage, les spécialités, les services, les évaluations, l'assiduité et la

traçabilité des informations pour faciliter le passage d'information entre les utilisateurs. Cette plateforme est essentielle pour la réflexion, la planification et le suivi des stages des étudiants en santé, tout en centralisant les actions de tous les intervenants impliqués dans ces stages à l'UASZ.

Adoptant une approche de développement Agile, en particulier la méthode SCRUM, une collaboration étroite avec les utilisateurs impliqués dans chaque aspect du système a été favorisée. Dans la phase de spécification des besoins, les utilisateurs ont été identifiés et les cas d'utilisation pour chaque module ont été définis. L'analyse des besoins et la conception du système ont été réalisées en élaborant des diagrammes d'activités, des diagrammes de séquences, ainsi que des analyses fonctionnelles et des architectures physique et logique. Des diagrammes de classes ont été utilisés pour la modélisation en employant la notation UML et l'outil PowerAMC, avec une collaboration via l'outil Git pour la gestion des versions.

Pour le développement, deux Framework principaux ont été utilisés : React.js pour le front-end et Express (Node.js) pour le back-end, ainsi que les technologies associées. La base de données MySQL a été utilisée pour stocker les données, tandis que l'API REST a été mise en place pour créer les services, avec Insomnia utilisé pour les tests. Un modèle de conception a été fourni par le responsable pédagogique et adapté selon les besoins.

Notre stage de fin d'études s'est déroulé au sein du Service Pédagogique de l'UFR Santé dans le cadre du Projet de Gestion des Stages (GS). Notre mission spécifique était de concevoir et mise en place d'une plateforme web pour l'exploitation de l'environnement de gestion des stages hospitaliers de l'UFR des sciences de la Santé de l'UASZ.

La structure de notre mémoire est conçue comme suit :

Chapitre 1 : Présentation de l'environnement institutionnel, du projet GS, du contexte et des problématiques du stage, ainsi que des objectifs visés.

Chapitre 2 : Description des différents processus de conception et justification de celui choisi pour notre stage.

Chapitre 3 : Spécification des besoins en identifiant les acteurs impliqués et en déterminant les besoins fonctionnels du système à travers les cas d'utilisation.

Chapitre 4 : Analyse des besoins, présentation des résultats de la conception générale et détaillée, exposition des architectures et des modèles sélectionnés.

Chapitre 5 : Détails sur les outils utilisés pour l'implémentation de l'application, suivi des tests, du déploiement et présentation de quelques interfaces de l'application.

Chapitre I : Contexte et Problématique

Introduction

Ce chapitre s'ouvre, par une présentation de de l'Université Assane Seck de Ziguinchor (UASZ), l'UFR 2S qui est l'entité qui a servi de cadre à notre stage au sein du service pédagogique (SP), considéré comme le client et les futurs utilisateurs de l'application que nous développons. Enfin, nous abordons le contexte spécifique du stage, en identifiant la problématique traitée et les objectifs que nous cherchons à atteindre dans cette expérience professionnelle.

I.1. Présentation de la structure d'accueil

I.1.1. L'Université Assane Seck de Ziguinchor

L'UASZ a été fondée en 2008 par le décret 2008-537 du 22 mai en tant que troisième université du Sénégal. Fonctionnant selon le système Licence Master Doctorat (LMD), elle offre une variété de formations dans différents domaines académiques. L'UASZ a débuté ses activités en février 2007 avec trois Unités de Formation et de Recherche (UFR) initiales,

- UFR des Sciences Economiques et Sociales
- UFR des Sciences et Technologies
- UFR des Lettres, Arts et Sciences Humaines.

A ces dernières s'ajoute l'ufr des Sciences de la Santé (2S) (intégrée en 2011) pour répondre aux besoins croissants de la région.

I.1.2. L'UFR 2S

L'UFR des Sciences de la Santé constitue l'un des piliers majeurs de l'Université Assane Seck de Ziguinchor (UASZ), offrant une formation médicale de qualité supérieure et contribuant activement au développement des services de santé locaux. Intégrée à l'université en 2011, cette UFR fonctionne selon des principes similaires aux autres composantes universitaires, avec notamment un service pédagogique, des départements et des responsables de formation.

Cette UFR se démarque par la rigueur de ses programmes de formation et ses standards élevés, préparant ainsi les étudiants à une réussite professionnelle dans le domaine médical. Les stages pratiqués, effectués dans des établissements de santé agréés, offrent aux étudiants

une expérience précieuse et diversifiée dans différentes spécialités médicales, allant de la médecine interne à la psychiatrie en passant par la gynécologie et l'ophtalmologie, entre autres.

Concernant l'organisation des stages, d'importantes décisions ont été prises au sein de l'UFR. Par exemple, chaque étudiant, du niveau Licence 2 jusqu'au doctorant 3, effectue des stages lors des semestres 1 et 2. Ces stages sont soigneusement planifiés et supervisés pour garantir leur qualité. Pour assurer un suivi rigoureux, l'UFR procède chaque année ou semestre à l'inscription des étudiants dans les établissements de santé partenaires.

Pour garantir la transparence et le bon déroulement des stages, les étudiants sont assignés à des spécialités spécifiques sous la supervision d'un responsable de spécialité et ensuite à des services spécifiques. Pendant toute la durée du stage, le secrétariat assure le suivi de l'assiduité des étudiants. À la fin du stage, le maître de stage attribue les notes en conformité avec les critères préétablis. Cette approche assure une expérience de stage enrichissante et conforme aux normes professionnelles les plus élevées.

I.1.2.1. Organisation de L'UFR

L'UFR 2S est structurée autour de plusieurs services qui sont :

- **A- Le Service Administratif :** ce service est dirigé par le directeur du chef de service administratif, il englobe le service financier, les services généraux, et une partie du service pédagogique. Son rôle est de gérer la gestion des ressources humaines, la gestion des finances, la coordination des opérations quotidiennes, la gestion de l'information et de la communication. Il veille sur la conformité règlementaire, le soutien à la direction et aux employés.
- **B- Le service Pédagogique :** ce service est chargé de gérer tous les aspects liés à l'enseignement et à l'apprentissage au sein de l'organisation. Ses responsabilités incluent la planification des programmes d'études, l'organisation des cours et des examens, la gestion des inscriptions des étudiants, le suivi de leur progression académique, ainsi que la coordination des activités pédagogiques avec les enseignants et les départements académiques. Ils sont tous sous l'autorité du directeur du chef du service administratif et du directeur adjoint.
- **C- Le Service Financier :** ce service est responsable de la gestion des ressources financières de l'organisation. Ses tâches principales comprennent la tenue des comptes, la gestion des budgets, la facturation des frais de scolarité ou autres, la gestion des

paiements et des recouvrements, ainsi que la préparation des rapports financiers pour la direction et les autorités de tutelle. Ils sont tous sous l'autorité du directeur du chef du service administratif.

D- Les Services Généraux : ce service englobe un large éventail de tâches qui soutient le fonctionnement quotidien de l'organisation. Cela peut inclure la gestion des locaux et des installations, l'approvisionnement en fournitures de bureau et en équipements, la maintenance des infrastructures, la gestion des services de restauration et de nettoyage, ainsi que la coordination de la logistique pour les événements et les réunions. Ils sont tous sous l'autorité du directeur du chef du service administratif.

E- **Les Départements :** ils sont au nombre de quatre :

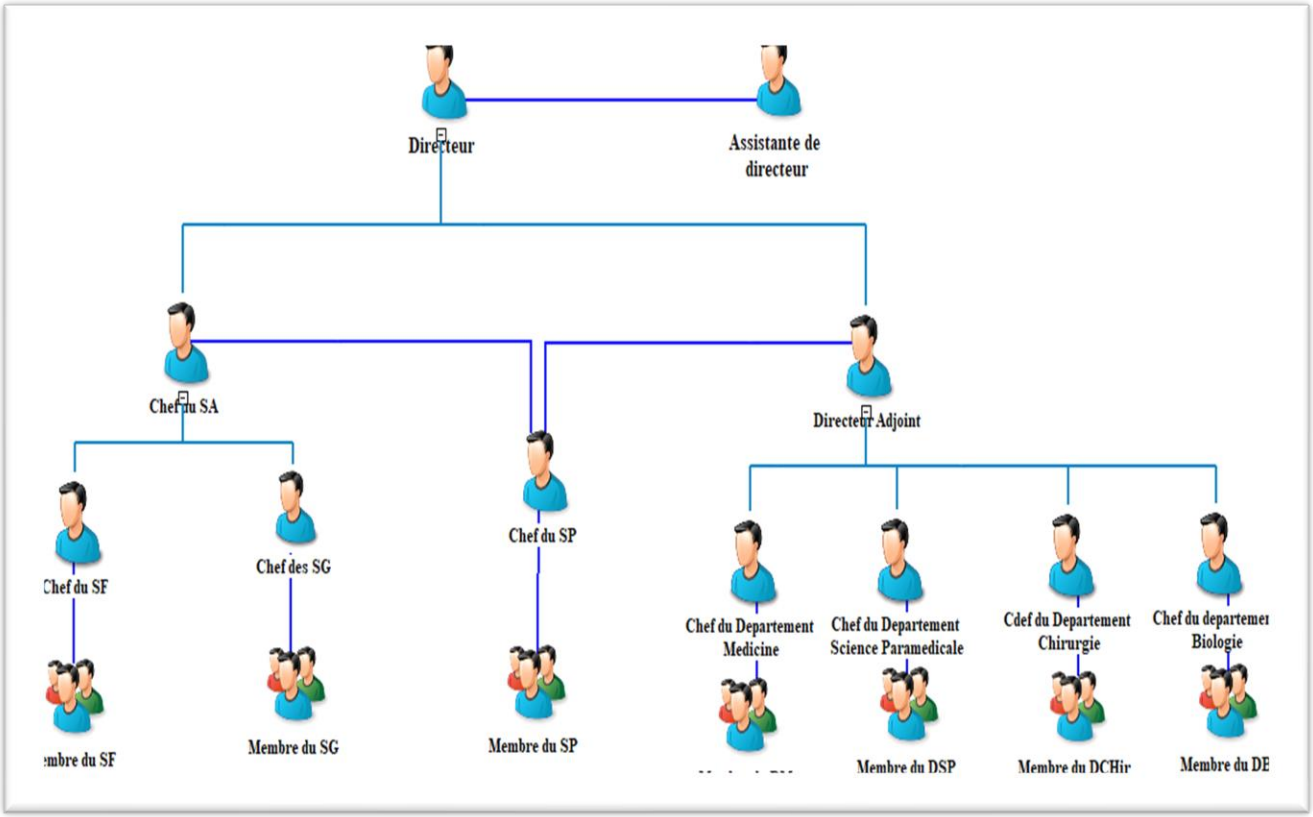
a) Le Département Médecine : géré par le chef du département médicine, ce département remplit un large éventail de fonctions essentielles, allant de la formation des futurs médecins à la recherche médicale et à la prestation de soins de santé, contribuant ainsi à l'avancement des connaissances médicales et à l'amélioration des soins de santé dans la communauté.

b) Le Département Chirurgie : géré par le chef du département chirurgie, ce département remplit un ensemble diversifié de fonctions qui soutiennent l'éducation médicale, la recherche chirurgicale et la prestation de soins de santé de haute qualité. Son rôle essentiel est de former la prochaine génération de chirurgiens, promouvoir l'innovation dans le domaine chirurgical, et améliorer les résultats pour les patients.

c) Le Département Biologie et Exploration Fonctionnelle : gère par le chef du département Biologie, ce département joue un rôle central dans l'avancement des connaissances en biologie, la formation des futurs biologistes et la recherche de solutions aux défis sociétaux et environnementaux. Son expertise et ses efforts contribuent à mieux comprendre le monde vivant et à relever les défis de notre époque

d) Le Département des Sciences Paramédicales : Le département des Sciences Paramédicales joue un rôle crucial dans la formation, la recherche et le développement professionnel des professionnels de la santé paramédicaux. Son objectif principal est de préparer des praticiens compétents, bien informés et éthiques, capables de fournir des soins de haute qualité et de contribuer à l'amélioration de la santé et du bien-être des patients. Il est dirigé par le chef du département des sciences paramédicales.

Chaque service est constitué de division à sa tête un chef de division. Ils sont tous sous l'autorité du directeur de l'UFR.



La figure ci-dessous (***figure 1***) représente l'organigramme de l'UFR

I.1.3. Les Missions

Nous allons donner d'abord la mission de l'UFR 2S en suite celle du service pédagogique dans le cadre de la gestion des stages des étudiants.

I.1.3.1. Mission de L'URF 2S

L'UFR des Sciences de la Santé de l'Université de Ziguinchor est un établissement éducatif et de recherche affilié à l'Université de Ziguinchor. Son objectif principal est de former des professionnels de haut niveau pour répondre aux besoins en ressources humaines dans le domaine de la santé au Sénégal. La création de l'UFR des Sciences de la Santé répond à la nécessité d'améliorer l'accès aux soins de santé dans la région de la Casamance qui est souvent éloignée et isolée. En particulier, elle vise à renforcer les capacités médicales de la

Figure 1:Organisation de l'UFR 2S

région, où les structures de santé sont de plus en plus sollicitées par des patients venant de toute la sous-région.

Cette UFR contribue également à former des spécialistes qui, en plus de dispenser des cours, peuvent intervenir dans les hôpitaux locaux pour traiter diverses pathologies. Depuis son ouverture, l'UFR des Sciences de la Santé de l'Université de Ziguinchor a adopté le système LMD (Licence-Master-Doctorat) et a commencé ses activités avec un effectif de 43 étudiants.

I.1.3.2. Mission du service pédagogique dans la gestion des stages

Le service Pédagogique est chargé de gérer tous les aspects liés à l'enseignement et à l'apprentissage au sein de l'organisation de l'UFR.

Sa mission dans le processus de la gestion des stages des étudiants est d'organiser les stages des étudiants, coordonne leurs suivis dans les hôpitaux, spécialités, services.

En effet, tout étudiant concerné par le stage doit passer dans le service pédagogique pour une prise de note de service avant de démarrer son stage dans un service. Il doit connaitre dans quel l'hôpital il doit le faire, quelle spécialité, et tout ça est c'est chef du service pédagogique qui le coordonne. Donc, sa mission dans la gestion des stages est de coordonner et assure le suivi.

I.2. Description du sujet

Pour décrire, le sujet de notre projet nous allons donner son contexte d'abord, en suite après examinassions de la situation, nous allons identifier les défis rencontrés, en fin proposer des solutions pour le remédier.

I.2.1. Contexte du sujet

Depuis le début des stages des étudiants de la première promotion en 2013, la gestion de ces stages au sein de l'UFR des Sciences de la Santé (UFR 2S) de l'Université Assane Seck de Ziguinchor (UASZ) se fait de manière manuelle. Jusqu'à présent, le service pédagogique n'a pas eu accès à un outil informatique dédié pour gérer efficacement les stages de leurs étudiants.

Ainsi, pour répartir les étudiants dans les différents sites de stages, le responsable pédagogique procède de manière manuelle en utilisant des papiers volants. Ce processus peut parfois être sujet à des pertes de documents, ce qui peut compromettre la planification des stages. De même, l'attribution des notes, le suivi de la présence des étudiants et d'autres tâches

administratives sont également gérés de manière manuelle, ce qui peut être chronophage et sujet à des erreurs.

Dans ce contexte où la gestion des stages au sein de l'UFR des Sciences de la Santé se fait encore de manière manuelle, se pose une problématique majeure : celle des multiples défis rencontrés par le personnel du service pédagogique dans l'accomplissement de leurs tâches administratives.

I.2.2. Problématiques

Après avoir examiné la situation actuelle, nous avons identifié plusieurs défis rencontrés par le personnel du service pédagogique dans la gestion des stages des étudiants. Voici quelques difficultés :

- La rédaction des notes de service devient très complexe lorsque le nombre d'étudiants stagiaires est élevé. Cela peut être fastidieux et prendre beaucoup de temps.
- Le processus de dépôt des listes d'étudiants en stage dans les hôpitaux est ardu pour le personnel, nécessitant des déplacements fréquents et souvent fatigants.
- La sélection des stagiaires et leur suivi deviennent des tâches complexes en raison du manque d'outils efficaces pour gérer ces processus.
- L'accès aux informations des stagiaires est difficile car il faut constamment rechercher leurs dossiers, ce qui peut être chronophage.
- Les étudiants doivent systématiquement demander une note de service à chaque service où ils doivent se rendre en stage, ce qui ajoute à la charge administrative.
- Les documents peuvent être perdus, mal classés ou endommagés en raison de leur manipulation fréquente, ce qui peut compromettre la gestion des stages.
- Les dossiers occupent beaucoup d'espace physique et leur stockage peut devenir problématique.
- Les dossiers ne sont pas sécurisés, ce qui signifie que n'importe qui peut y accéder, posant ainsi un risque de confidentialité.
- L'absence de statistiques rend difficile l'évaluation et le suivi de la performance des étudiants en stage, ainsi que l'analyse des tendances et des besoins en matière de formation, etc.

Face aux défis rencontrés par le personnel du service pédagogique dans la gestion manuelle des stages des étudiants, il devient impératif d'explorer une solution informatisée afin de

pallier ces difficultés et d'améliorer significativement l'efficacité et la fiabilité de ce processus crucial pour l'UFR 2S.

I.2.3. Solutions proposées

Après tous ces défis relevés, nous avons proposé le développement d'une application web dédiée à la gestion des stages des étudiants. Cette application vise à simplifier et à rationaliser les processus de gestion des stages au sein de l'UFR des Sciences de la Santé.
Nous avons identifié cinq acteurs principaux qui interagiront avec l'application de différentes manières :

- L'administrateur de l'application
- Le responsable pédagogique
- Les chefs de département
- Les secrétaires
- Les maîtres de stage

Voici un aperçu des fonctionnalités et des rôles attribués à chaque acteur dans l'application :

- Le responsable pédagogique importe les fiches d'étudiants de chaque niveau et les répartit dans les hôpitaux et les spécialités de manière équitable.
- Il peut également modifier cette répartition selon les besoins.
- Les chefs de département consultent les listes des étudiants de leur spécialité et les assignent à leurs services respectifs avec une date de début de stage.
- Les secrétaires sont chargés de suivre l'assiduité des étudiants pendant leurs stages et envoient régulièrement des listes de présence aux chefs de département.
- Les maîtres de stage notent les étudiants à la fin de chaque stage.
- Les chefs de département gèrent les rotations des étudiants en fonction des notes attribuées, en veillant à ce qu'un étudiant n'ayant pas obtenu de note dans un service précédent ne puisse pas accéder à un autre service.
- Les comptes des utilisateurs sont ajoutés par l'administrateur.
- Le responsable pédagogique organise la rotation semestrielle des étudiants, passant de la médecine à la chirurgie et vice versa, en début de semestre 2.
- À la fin de chaque semestre, le responsable pédagogique peut imprimer les fiches de stage individuelles des étudiants ainsi que celles de toute une promotion du même niveau.

- Pour assurer une accessibilité maximale et éviter le besoin d'installation sur les postes, l'application sera développée en tant qu'application web, permettant un accès permanent via un navigateur web et une connexion Internet.

Après avoir dressé le contexte et identifié les défis inhérents à la gestion manuelle des stages des étudiants au sein de l'UFR des Sciences de la Santé, nous proposons une solution informatisée afin de remédier à ces difficultés et d'améliorer l'efficacité de ce processus. En poursuivant cette démarche, nous détaillerons les objectifs généraux et spécifiques visant à mettre en place un site web dédié à la gestion des stages, avec pour ambition d'optimiser la planification, le suivi et la performance des étudiants au cours de leurs périodes de stage.

I.3. Objectifs attendus

Pour automatiser la gestion des stages des étudiants cet UFR nous avons fixes un objectif général qui en découle des objectifs spécifiques :

I.3.1. Objectif général

L'objectif général est de mettre en place un site web pour la gestion des stages des étudiants de l'UFR des sciences de la santé (2S) de l'Université Assane Seck de Ziguinchor.

Le principal avantage dans l'utilisation d'un tel outil informatique est la possibilité de dématérialiser le travail fait jusqu'ici par le responsable pédagogique (supports papiers) car la plupart des informations sur les étudiants pour les stages sont sur des papiers. En effet, ceci est une façon de pérenniser toutes les informations pour les restituer en cas de besoin.

I.3.2. Objectifs spécifiques

Les principaux objectifs attendus dans l'automatisation de la gestion des stages et du suivi sont :

- de faciliter la gestion des stages en orientant les étudiants dans les hôpitaux sans se déplacer.
- de faciliter la gestion des stages en orientant équitablement les étudiants vers les spécialités médicales et chirurgicales.
- l'application doit également permettre de gérer le transfert d'étudiants entre les spécialités et services.
- de faciliter le pointage des étudiants.
- de faciliter l'attribution des notes aux étudiants.

- de garder la traçabilité des informations relatives aux stages des étudiants, etc.

À la fin du projet, nous souhaitons obtenir une application web fiable, compréhensible et facile à utiliser pour les utilisateurs du système.

Conclusion

Cette première phase nous a offert une compréhension approfondie de la situation actuelle et des défis auxquels sont confrontés les différents acteurs impliqués dans la gestion des stages. Forts de cette vision clarifiée, nous avons pu établir des objectifs spécifiques visant à résoudre les contraintes identifiées.

Cependant, pour concrétiser ces objectifs, il est essentiel de mettre en place une méthodologie de développement appropriée. Ainsi, dans la deuxième phase de notre travail, nous détaillons le processus de développement à suivre pour ce projet. Cela nous permet de structurer notre approche, d'organiser les différentes étapes du développement de l'application et de garantir son efficacité et sa pertinence pour répondre aux besoins des utilisateurs.

Chapitre II : Processus de conception et développement

Introduction

Le processus de développement d'une application est une série d'étapes organisées visant à concevoir, créer, tester et déployer des applications logicielles. Chaque étape peut être complexe et exige des compétences techniques spécifiques. Selon les besoins du projet et de l'équipe de développement, il est crucial d'adopter une méthodologie appropriée telle qu'**Agile, DevOps, Lean, en Cascade ou autres**.

Ces méthodologies visent principalement à réduire les risques et les coûts. Bien qu'elles diffèrent dans leur approche, elles partagent toutes un objectif commun aider les équipes de développement à produire des logiciels de haute qualité dans des délais serrés et à moindre coût[1].

Ce chapitre vise à présenter brièvement quelques-unes de ces méthodologies avant d'expliquer les critères qui ont guidé le choix de la méthode spécifique pour ce projet.

I. Méthode en cascade

I.1. Définition

La méthode en cascade, également connue sous le nom de modèle en cascade ou cycle de vie en cascade, ou "waterfall" en anglais, est une approche de gestion de projet qui suit une séquence linéaire et séquentielle.

Elle divise le projet en différentes étapes distinctes qui doivent être accomplies successivement et de manière interdépendante.

En d'autres termes, chaque étape ne peut démarrer que lorsque la précédente est terminée et approuvée[2]

I.2. Les étapes essentielles de la méthode en cascade

Le modèle en cascade divise le cycle de vie d'un projet en sept phases distinctes[2]

- **Phase une : Planification :**avant de plonger dans le vif du sujet, il est primordial de planifier le projet. Cela implique de définir précisément les besoins et les exigences de toutes les parties impliquées. Cette phase prend du temps, car elle nécessite une analyse minutieuse pour établir un plan détaillé du projet. Le résultat de cette étape est un document clé appelé "document des exigences du projet", qui

résume les différentes étapes du projet, les responsabilités attribuées à chacune, les ressources nécessaires et les délais[3].

- **I.2.2. Phase deux : Analyse :** dans cette phase, vous examinez de près la liste des besoins et des exigences établies précédemment afin de créer un cahier des charges. Ce document détaille les spécifications fonctionnelles du projet[4].
- **Phase trois : Conception :** une fois l'analyse terminée, vous passez à la conception du produit en vous basant sur les spécifications établies précédemment. L'équipe de conception affine les détails fonctionnels du projet. Par exemple, dans le développement logiciel, elle décide des technologies à utiliser, des langages de programmation, etc. Il est même possible de créer un prototype à cette étape pour mieux visualiser le produit final[5].
- **Phase quatre : Mise en œuvre,** ici, c'est le moment pour les développeurs de donner vie au produit en se basant sur le document des exigences établi précédemment ainsi que sur les spécifications détaillées pendant les phases d'analyse et de conception. Dans le domaine du développement logiciel, cela implique généralement du codage ou de la programmation[6].
- **Phase cinq : Tests,** une fois la mise en œuvre achevée, le produit passe par une phase de tests approfondis. L'équipe de contrôle qualité examine le produit pour s'assurer qu'il respecte les exigences initiales. Les testeurs recherchent les éventuelles erreurs et proposent des corrections avant le déploiement. Toutes les anomalies détectées sont consignées dans un rapport pour référence futur[7].
- **Phase six : Déploiement,** c'est l'étape où le produit est finalement lancé ou livré à l'utilisateur final ou aux clients. On parle souvent de "lancement du produit" ou de "livraison des livrables"[8].
- **Phase sept : Maintenance,** après le déploiement, il est possible que des ajustements ou des améliorations soient nécessaires. La phase de maintenance intervient alors pour assurer le bon fonctionnement continu du produit. Cela inclut les réparations, les mises à jour et les évolutions nécessaires tout au long de la durée de vie du produit[9].

En somme, ces phases sont réalisées de manière séquentielle, chaque phase étant achevée avant le début de la suivante.

La figure ci-après (***figure 2***) représente les phases de la méthode en cascade

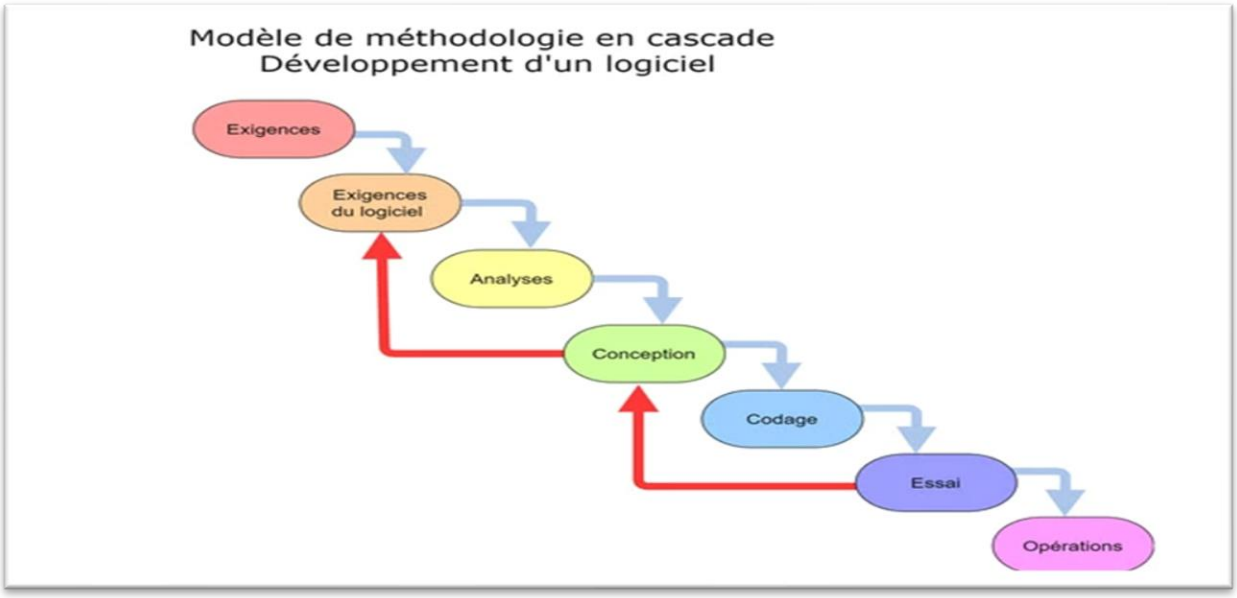


Figure 2:Exemple de modèle en cascade [2]

I.3. Avantages et Inconvénients de la méthode en cascade

La méthode cascade présente à la fois des avantages et des inconvénients.

Parmi ses avantages, on peut citer :

- Une planification structurée qui permet une meilleure anticipation des besoins et des étapes du projet.
- Une documentation détaillée qui facilite la compréhension et la traçabilité des décisions prises.
- Un suivi facile grâce à la séquence linéaire des phases du projet.

En revanche, cette méthode présente également des inconvénients, tels que :

- L'absence de flexibilité pour faire face aux imprévus, Comme le processus avance de manière linéaire et séquentielle, il ne peut pas facilement s'adapter aux retards ou aux obstacles inattendus. Par exemple, si une pièce nécessaire pour la fabrication n'arrive pas à temps, cela peut entraîner un retard dans le calendrier ou même suspendre complètement le projet.
- L'impossibilité de revenir en arrière une fois une phase terminée, ce qui peut rendre les corrections ou les ajustements plus complexes.
- Un contrôle qualité tardif, car les tests n'ont lieu qu'après la mise en œuvre, ce qui peut entraîner des retards ou des coûts supplémentaires en cas de problèmes détectés.

En résumé, la méthode cascade convient particulièrement aux projets où les objectifs sont clairement définis dès le départ, où les exigences restent stables et où la mise en œuvre peut être parfaitement maîtrisée.

II. Méthode ou concept Agile

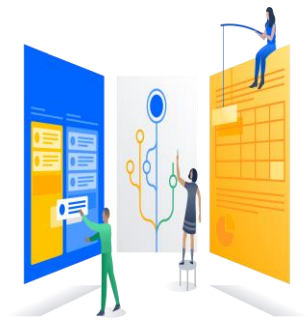
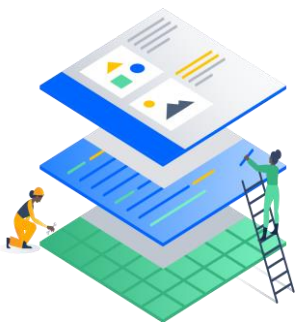
II.1. Présentation de la méthode Agile

L'Agilité représente une approche itérative pour la gestion de projets et le développement de logiciels, mettant en avant la collaboration, le feedback client et des cycles de livraison rapides. Cette méthode a émergé dans le domaine du développement logiciel au début des années 2000, visant à permettre aux équipes de s'adapter efficacement aux fluctuations du marché et aux demandes évolutives des clients.

Dans un cadre Agile, une partie de la planification et de la conception est réalisée préalablement, mais le développement se déroule par petites itérations et implique une étroite coopération avec toutes les parties prenantes. Les ajustements sont continuellement intégrés, favorisant la livraison de versions utilisables du produit à un rythme souvent plus rapide que les approches traditionnelles en cascade. Cette approche offre de multiples avantages, notamment la possibilité de rectifier rapidement tout écart entre le logiciel produit et les attentes du client.

L'Agilité représente un ensemble de méthodologies plutôt qu'une seule approche de développement. Elle englobe les principes du Scrum, de l'Extreme Programming (XP), de Test driven development (TDD), de Processus Unifie Agile (AUP), Lean Software developmen (LSD) et d'autres pratiques éprouvées utilisées par les développeurs par le passé. Le résultat de cette convergence d'expertises est matérialisé dans le Manifeste Agile, qui se compose de douze principes articulés autour de quatre valeurs fondamentales [10].

Tableau : 1 Les quatre valeurs fondamentales du Manifeste Agile

Les individus et leurs interactions plutôt que les processus et les outils	**Des logiciels opérationnels** plus qu'une documentation exhaustive	**La collaboration avec les clients** plus que la négociation contractuelle	**L'adaptation au changement** plus que le suivi d'un plan

II.2. Le principe de la méthode Agile

La méthodologie Agile peut se décliner sous différentes formes et adopter un langage spécifique. Voici quelques variantes de méthodes Agile

II.2.1. Test driven development (TDD)

Le développement piloté par les tests, aussi connu sous le nom de "Test Driven Development" (TDD), est une approche de développement qui intègre étroitement l'écriture de tests unitaires, la programmation et la révision du code[11].

Avant même de commencer à écrire du code, des tests unitaires sont rédigés pour définir le comportement attendu des différentes fonctionnalités. Ensuite, le développeur écrit le code Minimal nécessaire pour que les tests passent, sachant que le code est susceptible d'échouer au départ.

À mesure que le développement progresse, le code source est simplifié autant que possible tout en maintenant la réussite des tests. De nouveaux tests sont constamment ajoutés et exécutés automatiquement tout au long du processus de développement.

Le TDD est toujours accompagné d'outils d'automatisation des tests unitaires spécifiques au langage de programmation utilisé, ce qui permet une vérification continue et une rétroaction immédiate sur la qualité du code.

II.2.2. Processus Unifie Agile (Agile UP ou AUP)

Le Processus Unifié Agile (Agile Unified Process) est une version simplifiée du Rational Unified Process (RUP). Il s'agit d'une méthode de développement d'applications destinée aux entreprises, qui intègre les pratiques agiles telles que le Test-Driven Development (TDD), le Model-Driven Development (MDD) et la gestion du changement[12].

Ce processus est divisé en quatre phases :

- Lancement : durant cette phase, le projet est délimité, les architectures potentielles du système sont définies, les parties prenantes sont impliquées et une estimation des coûts est obtenue.

- Conception : cette phase consiste à définir l'architecture du système, c'est-à-dire à concevoir la structure et les composants du système.
- Réalisation : pendant cette phase, le logiciel est développé de manière itérative et incrémentale, en se concentrant sur les fonctionnalités par ordre de priorité.
- Livraison : une fois le développement terminé, le système est validé et déployé, prêt à être utilisé par les utilisateurs finaux.

Ce processus combine donc les principes fondamentaux du développement agile avec une approche itérative et incrémentale, permettant une flexibilité et une adaptation aux besoins changeants du projet tout en assurant une gestion efficace des coûts et des ressources.

II.2.3. Scrum

Scrum, une méthodologie de gestion de projet Agile, vise à améliorer la productivité des équipes agiles, même en travail à distance, tout en permettant une optimisation du produit grâce à des retours réguliers des utilisateurs finaux.

Inspirée du monde du rugby, où "scrum" signifie "mêlée", cette méthode implique des réunions fréquentes des équipes agiles pour s'assurer que le projet progresse efficacement, prêtes à ajuster leur trajectoire au besoin. Ainsi, Scrum offre une approche dynamique et collaborative de la

gestion de projet, assurant au client un équilibre optimal entre l'investissement initial et le produit final livré.

Scrum est largement adoptée par les équipes de développement car elle met en avant les valeurs fondamentales de l'Agile Manifeste : la collaboration avec le client, l'acceptation du changement, l'interaction humaine, et la focalisation sur des logiciels opérationnels.

Bien que son vocabulaire spécifique puisse être intimidant pour les novices, Scrum n'est pas aussi complexe qu'il n'y paraît au premier abord. Il suffit de mettre de côté les notions préconçues sur la gestion de projet et de se familiariser progressivement avec ses concepts.

De plus, scrum fait partie des cinq méthodes agiles les plus répandues, offrant également des possibilités de mise à l'échelle pour déployer l'agilité à l'échelle de l'entreprise.

Dans cette méthodologie, le travail se fait par itérations courtes, permettant une approche plus claire et précise pour la réalisation de projets complexes.

La méthode agile scrum vise à établir un cadre de travail clair et précis à travers des itérations courtes, facilitant ainsi la gestion de projets complexes.

Ce cadre repose sur trois principes essentiels[13] :

- **Transparence** : il est crucial que tous les membres de l'équipe aient accès aux informations pertinentes concernant le produit à développer.
- **Inspection** : des évaluations régulières sont indispensables pour évaluer l'avancement du projet et effectuer des ajustements si nécessaire.
- **Adaptation** : lorsque les évaluations révèlent des écarts par rapport aux résultats attendus, il est nécessaire de mettre en œuvre de nouvelles mesures pour rectifier le tir et garantir le succès du projet.

Scrum implique trois principaux rôles :

- **Le Product Owner**, ou « Directeur de produit », qui communique les objectifs principaux des clients et des utilisateurs finaux, coordonne leur implication, et assure la cohérence avec les autres propriétaires de produits.
- **Le Scrum Master**, membre de l'équipe, cherche à maximiser la productivité de l'équipe en l'aidant à travailler de manière autonome et à s'améliorer continuellement.
- **L'équipe opérationnelle**, idéalement composée de moins de dix personnes et caractérisée par son absence de hiérarchie interne, travaillant de manière autonome.

D'autres termes clés associés à Scrum incluent :

- **Le product backlog** : document contenant les exigences initiales du client, mais qui évolue tout au long du projet en fonction des besoins changeants.
- **Le sprint backlog** : ensemble des tâches à réaliser définies au début de chaque sprint par l'équipe de développement.
- **User story** : les fonctionnalités décrites par le client.
- **La "mêlée"** (ou "scrum") est une réunion quotidienne de suivi durant le sprint.

La figure ci-dessous illustre les rôles principaux dans la méthodologie Scrum.

Figure 3: Méthode Scrum Agile [14]

II.2.4. Programmation Extrême (XP)

La Programmation Extrême (Extreme Programming en anglais) est une approche de développement qui se caractérise par des cycles de travail très courts et une forte collaboration entre tous les membres de l'équipe projet. La planification des tâches est flexible et les estimations de charge sont simplifiées. Les fonctionnalités sont livrées régulièrement et soumises à des tests et validations à l'aide de prototypes opérationnels, assurant ainsi l'alignement entre les besoins du client et les réalisations concrètes[14].

II.2.5. Lean Software development

Le Lean Software Development, ou développement de logiciel Lean, repose sur sept grands principes[15]:

- Éliminer les gaspillages tels que les travaux incomplets, les processus inutiles, les fonctionnalités non essentielles, les changements d'équipe et les retards.
- Favoriser l'apprentissage en multipliant les sources d'acquisition de connaissances et en synchronisant les équipes.
- Reporter les décisions à un moment opportun pour éviter les discussions prolongées et les décisions irrévocables jusqu'à ce qu'elles soient vraiment nécessaires.
- Livrer rapidement et régulièrement pour obtenir un retour client rapide.
- Responsabiliser les équipes en favorisant leur autonomie et leur leadership.
- Mettre la qualité au cœur du projet, de la conception à la réalisation.
- Optimiser le système dans son ensemble en mettant en place des mesures de performance complètes et en gérant les différentes interactions et dépendances.

Avec la méthode Lean, la qualité est véritablement placée au centre de la gestion du projet, en optimisant notamment l'ensemble des processus d'apprentissage, de prise de décision, de livraison et de mesure de performances.

II.3. Avantages et Inconvénients de la méthode Agile

Bien que la méthode Agile soit largement adoptée et offre de nombreux avantages, elle comporte également quelques inconvénients[16].

II.3.1. Les avantages

Les avantages de la méthode agile sont nombreux :

- **Adaptabilité aux changements** : L'approche agile permet de s'adapter facilement aux changements de besoins et de priorités du client grâce à des itérations courtes et des processus flexibles.
- **Livraison rapide de valeur** : Les équipes Agile livrent des fonctionnalités opérationnelles à des intervalles courts, ce qui permet au client de bénéficier rapidement de la valeur ajoutée du produit.
- **Meilleure collaboration** : L'agilité favorise une collaboration étroite entre les membres de l'équipe, ainsi qu'avec le client, ce qui conduit à une meilleure compréhension des besoins et des attentes.
- **Feedback continu** : Les cycles de développement courts permettent d'obtenir un feedback régulier du client, ce qui permet d'ajuster et d'améliorer le produit de manière itérative.
- **Réduction des risques** : En raison de sa nature itérative et incrémentale, la méthode agile permet de détecter et de résoudre les problèmes rapidement, ce qui réduit les risques liés au développement du projet.
- **Motivation de l'équipe** : Les équipes Agile ont souvent plus d'autonomie et de responsabilité, ce qui peut conduire à une plus grande motivation et à une meilleure satisfaction au travail.
- **Qualité accrue** : Avec des tests continus et une attention constante à la qualité, la méthode agile favorise la création de produits de haute qualité.
- **Transparence** : Les processus agiles sont généralement transparents, ce qui permet à toutes les parties prenantes d'avoir une visibilité sur l'avancement du projet et sur les obstacles rencontrés.

En résumé, la méthode agile offre une approche dynamique et collaborative qui permet de répondre efficacement aux besoins changeants des clients tout en favorisant la qualité, la transparence et la satisfaction de l'équipe.

II.3.2. Les inconvénients

- La méthode Agile favorise le dialogue et la collaboration, ce qui réduit souvent la quantité de documentation. Cependant, cela peut poser problème lorsqu'il y a un changement d'équipe projet, car le transfert des connaissances peut être difficile.

- De plus, la réussite de l'approche Agile repose sur une implication active du client, ce qui n'est pas toujours réalisable dans toutes les situations, notamment lorsque le client manque de temps ou d'intérêt pour s'impliquer pleinement.
- Les entreprises avec une structure hiérarchique forte peuvent également rencontrer des défis avec l'approche Agile en raison de son fonctionnement collaboratif.
- Bien que l'approche Agile offre un contrôle des coûts plus précis, elle rend difficile l'estimation du budget global du projet en raison de sa nature flexible. Cette flexibilité a un coût supplémentaire que le client doit être prêt à supporter.

En résumé, l'approche Agile offre une plus grande flexibilité et une meilleure visibilité dans la gestion du projet, ce qui attire de plus en plus d'adeptes à une époque où la personnalisation est essentielle. Elle convient particulièrement aux projets qui nécessitent une livraison rapide ou dont les exigences évoluent rapidement. Cette approche permet d'ajuster et d'optimiser le projet sans entraver l'organisation des équipes, ce qui la rend attrayante pour de nombreux types de projets.

III. L’approche DevOps

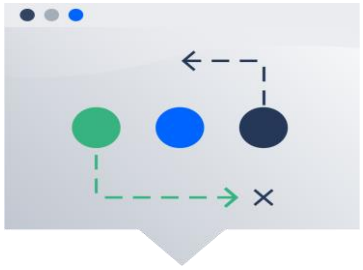
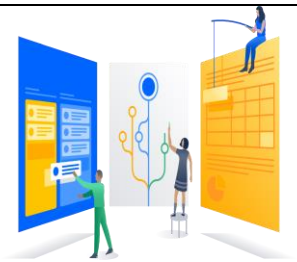
II.1. Définition

DevOps représente une approche du développement logiciel visant à accélérer et à renforcer la fiabilité des processus de développement, de test et de livraison en intégrant des principes et des pratiques issus de l'Agilité, tels que l'automatisation accrue et une collaboration renforcée entre les équipes de développement et opérationnelles. Bien que les processus de développement, de test et de déploiement soient des éléments communs à la fois dans Agile et DevOps, la méthodologie Agile traditionnelle ne prend pas en compte les opérations, aspect central de DevOps.

L'objectif principal de DevOps est de favoriser la collaboration entre les développeurs qui conçoivent les applications logicielles et les équipes opérationnelles chargées de les déployer et de les maintenir en production. En outre, DevOps s'attache à développer et gérer l'infrastructure sur laquelle ces applications sont déployées. Cette approche remplace le modèle antérieur où les équipes de développement livraient simplement leurs applications aux équipes opérationnelles pour le déploiement et la gestion, avec peu de visibilité sur le processus de développement. Dans un environnement DevOps, les développeurs et les équipes opérationnelles travaillent en collaboration tout au long du cycle de vie des applications, du développement à la gestion opérationnelle.

Deux cadres de référence courants pour comprendre DevOps sont les "Trois Voies" et "CALMS" (Culture, Automatisation, Lean, Mesure et Partage). La composante "culture" met l'accent sur le changement culturel nécessaire pour harmoniser les équipes de développement et opérationnelles. L'automatisation contribue à accélérer les processus et à garantir une qualité accrue. Les principes Lean encouragent l'amélioration continue et la prise en compte des erreurs comme un moyen d'apprentissage. La "mesure" fait référence à la pratique consistant à évaluer les résultats pour améliorer les processus. Enfin, le "partage" souligne l'importance de l'effort collectif dans DevOps et l'adoption de bonnes pratiques [17]

Tableau 2 : Les Trois Voies DevOps

		
Approche systémique Comprendre que les apps logicielles sont des systèmes complexes	**Amplification des boucles de feedback** Améliorer la communication bidirectionnelle entre les membres de l'équipe	**Virage culturel** Culture de l'expérimentation et de l'apprentissage continus

II.2. Les principes de l'approche DevOps

Pour tirer pleinement parti de DevOps, les équipes doivent suivre plusieurs principes fondamentaux qui définissent cette approche. Parmi ces principes clés, nous avons[18] :

- Automatisation : ce principe occupe une place centrale dans DevOps. Il englobe l'automatisation des différentes étapes du processus de développement, de tests, de déploiement et de gestion de l'infrastructure. L'utilisation d'outils d'automatisation permet de réduire les erreurs humaines et d'accélérer la livraison des logiciels.
- Collaboration : la collaboration favorise une meilleure compréhension des besoins de chacun au sein de l'équipe et permet de résoudre les problèmes plus rapidement en travaillant ensemble de manière transparente.

- Intégration Continue (CI) : la CI implique l'exécution automatisée de tests à chaque intégration de code afin de garantir que les modifications apportées au code restent fonctionnelles. Cela permet une intégration fréquente et sans heurts des modifications dans le code existant.
- Livraison Continue (CD) : en combinant la CD avec la CI, les équipes peuvent publier plus fréquemment des mises à jour de l'application, ce qui permet une diffusion plus rapide et efficace des nouvelles fonctionnalités et correctifs.
- Infrastructure as Code (IaC) : l'IaC consiste à gérer l'infrastructure de manière programmable, ce qui facilite la reproductibilité, la mise à l'échelle et la gestion de l'infrastructure de manière cohérente et efficace.
- Surveillance et retour d'information : devOps met l'accent sur la surveillance en temps réel des applications en production. La collecte et l'analyse des données permettent d'identifier rapidement les problèmes et d'améliorer les performances de manière proactive.
- Sécurité : les équipes travaillent ensemble pour identifier et résoudre les vulnérabilités dès le stade du développement, intégrant ainsi la sécurité dès le début du processus pour garantir la robustesse de l'application.
- Évolutivité : devOps permet de créer des applications évolutives qui peuvent s'adapter aux besoins changeants de l'entreprise. Cette approche favorise la flexibilité et la capacité à évoluer avec les demandes du marché et de l'organisation.

La figure ci-dessous illustre les différentes phases de l'approche DevOps, mettant en lumière la progression fluide et continue des activités tout au long du cycle de vie du développement et du déploiement des logiciels.

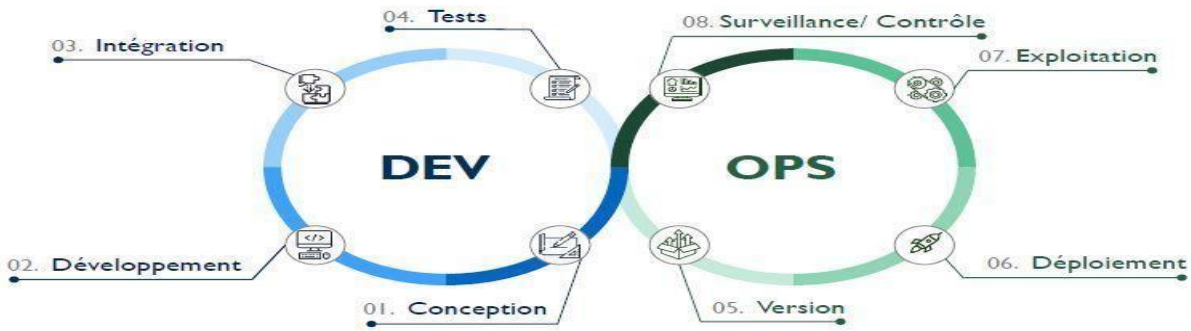


Figure 4:Méthode DevOps

II.3. Les Outils et Technologies utilisé par DevOps

Le choix des outils dépend largement des préférences spécifiques de l'entreprise ou de l'organisation, ainsi que des besoins et des objectifs du projet en cours. Parmi les outils les plus couramment utilisés dans le domaine du développement logiciel, on trouve :

- **Jenkins**[19] : C'est une plateforme d'automatisation open source autonome. Jenkins est très polyvalent et peut être configuré pour automatiser diverses tâches liées à la construction, aux tests et à la livraison ou au déploiement de logiciels.
- **Docker**[20] : Docker est une plateforme ouverte conçue pour développer, expédier et exécuter des applications de manière efficace. Il permet de séparer les applications de l'infrastructure sous-jacente, facilitant ainsi le déploiement rapide des logiciels. Grâce à Docker, la gestion de l'infrastructure peut être réalisée de manière similaire à celle des applications.
- **Kubernetes**[21] : Il s'agit d'un moteur d'orchestration de conteneurs open source qui automatise le déploiement, la mise à l'échelle et la gestion des applications conteneurisées. Kubernetes, hébergé par la Cloud Native Computing Foundation, offre une solution robuste pour gérer des environnements complexes de conteneurs.
- **Git**[22] : Git est un outil de développement collaboratif largement utilisé pour la gestion des versions du code source. Il permet à une équipe de développeurs de suivre et de gérer les changements apportés au code au fil du temps, en conservant un historique détaillé de chaque modification dans une base de données dédiée etc.

Ces outils ne représentent qu'une fraction de l'éventail d'outils disponibles pour les équipes de développement logiciel. Le choix des outils dépendra des exigences spécifiques du projet, de la taille de l'équipe, des compétences techniques disponibles et d'autres facteurs pertinents. En fin de compte, l'objectif principal est de sélectionner les outils qui favoriseront l'efficacité, la collaboration et la qualité du produit final.

II.4. Méthodes DevOps

Pour accélérer et améliorer le développement ainsi que le déploiement de leurs produits, les entreprises peuvent adopter différentes approches et techniques dans le domaine du développement de logiciels, notamment le DevOps. Parmi les méthodes les plus fréquemment utilisées se trouvent Scrum, Kanban et Agile.

Scrum est une méthode de travail qui définit comment les membres de l'équipe doivent collaborer pour accélérer les projets de développement et d'assurance qualité. Les pratiques

Scrum incluent l'utilisation de workflows spécifiques, d'une terminologie particulière et de rôles définis au sein de l'équipe.

II.5. Avantages de DevOps

Pour les adeptes des pratiques DevOps, les avantages commerciaux et techniques sont manifestes, contribuant majoritairement à accroître la satisfaction des clients[23] :

- **Accélération et amélioration de la livraison des produits :** en adoptant DevOps, les équipes parviennent à réduire significativement les délais de mise sur le marché, permettant ainsi de livrer des produits et des fonctionnalités plus rapidement. Cela favorise une réactivité accrue aux besoins changeants du marché et des clients.
- **Résolution des problèmes plus rapidement et réduction de la complexité** : grâce à l'intégration continue et à la livraison continue, les équipes peuvent identifier et corriger les problèmes plus rapidement, réduisant ainsi la complexité des déploiements et des maintenances. Cela garantit une meilleure stabilité et une fiabilité accrue des systèmes.
- **Production d'une plus grande évolutivité et une disponibilité sans précédent** : devOps permet aux organisations de concevoir des systèmes plus évolutifs et résilients, capables de s'adapter aux pics de charge et aux demandes fluctuantes des utilisateurs. Cela se traduit par une disponibilité sans précédent des services et des applications pour les utilisateurs finaux.
- **Augmentation de la stabilité de l'environnement d'exploitation** : en automatisant les processus de déploiement et de gestion de l'infrastructure, DevOps réduit les risques d'erreurs humaines et les temps d'indisponibilité, améliorant ainsi la stabilité globale de l'environnement d'exploitation.
- **Meilleure utilisation des ressources** : devOps permet une optimisation des ressources informatiques en automatisant les tâches répétitives et en utilisant efficacement les capacités disponibles, ce qui entraîne une réduction des coûts opérationnels et une utilisation plus efficiente des ressources matérielles et logicielles.
- **Automatisation accrue** : l'automatisation des processus de développement, de tests, de déploiement et de gestion de l'infrastructure permet aux équipes de se concentrer sur des tâches à plus forte valeur ajoutée. Cela réduit les délais et les efforts nécessaires pour livrer des fonctionnalités de haute qualité.

- **Meilleure visibilité sur les résultats du système** : les outils de surveillance et de suivi utilisés dans le cadre de DevOps fournissent une visibilité en temps réel sur les performances et la disponibilité des systèmes, permettant aux équipes de prendre des décisions plus éclairées et de réagir rapidement aux incidents.

En somme, en adoptant les pratiques DevOps, les organisations peuvent non seulement améliorer leur efficacité opérationnelle et leur agilité, mais aussi offrir des produits et des services de meilleure qualité, répondant ainsi aux attentes et aux besoins de leurs clients de manière plus efficace et fiable.

II.6. Inconvénients de DevOps

Bien que DevOps offre de nombreux avantages en termes d'efficacité, de collaboration et de rapidité de déploiement, il comporte également des défis et des inconvénients potentiels. Voici quelques-uns des inconvénients couramment évoqués, expliqués de manière détaillée [23]:

- **Complexité initiale :** la transition vers DevOps peut être initialement complexe, en particulier pour les organisations qui passent d'un modèle de développement traditionnel à celui de DevOps. Intégrer de nouvelles pratiques, outils et processus peut être laborieux et nécessiter du temps et de la formation pour l'ensemble de l'équipe. Les équipes peuvent rencontrer des difficultés à s'adapter à ces changements, ce qui peut entraîner des retards dans la mise en œuvre de DevOps.
- **Investissement en ressources :** mettre en place une infrastructure et des processus DevOps peut nécessiter un investissement initial important en termes de temps, de ressources humaines et de ressources financières. Cela inclut l'achat ou le développement d'outils et de technologies appropriés, ainsi que la formation du personnel sur les nouvelles pratiques et les nouveaux processus. Les organisations doivent être prêtes à consacrer ces ressources pour réussir leur transition vers DevOps.
- **Changement culturel :** devOps nécessite souvent un changement culturel au sein de l'organisation. Les équipes de développement et d'exploitation doivent travailler ensemble de manière collaborative, ce qui peut être un défi dans les organisations où ces équipes ont historiquement des objectifs, des priorités et des processus différents. Ce changement culturel peut rencontrer de la résistance, nécessitant un leadership fort et une communication efficace pour surmonter les obstacles au changement.

En résumé, de nombreuses méthodes DevOps, qui visent à rationaliser le développement et le déploiement des logiciels, reposent sur des modèles agiles tels que la programmation Lean.

L'évolution de DevOps découle de plusieurs mouvements visant à harmoniser les activités des développeurs et des équipes chargées des opérations, afin de favoriser une collaboration plus étroite et une livraison de logiciels plus efficace.

IV. Adaptation de la méthode Scrum Agile

Dans le cadre de notre projet avec l'UFR des Sciences de la Santé de l'Université, nous avons rapidement reconnu l'importance d'une collaboration étroite avec le client dès le début. Les profils variés des utilisateurs de cette UFR ont souligné la nécessité de prendre en compte des données spécifiques dans l'application que nous développions. Dès les premières réunions, une diversité de perspectives a émergé, mettant en lumière la nécessité d'adopter une approche agile.

Caractéristique	**Méthode en Cascade**	**Méthode Agile**	**Méthode DevOps**
Philosophie	Séquentielle, linéaire	Itérative, incrémentale	Collaboration et intégration continue
Flexibilité	Faible	Élevée	Élevée
Délais de livraison	Longs	Courts	Très courts
Gestion des changements	Difficile, coûteuse	Facilitée, intégrée	Facilitée, intégrée
Communication	Limitée, descendante	Fréquente, ascendante	Collaborative, multidirectionnelle
Risques	Élevés	Réduits	Réduits
Qualité	Souvent moins contrôlée	Priorisée, améliorée	Priorisée, améliorée
Client	Impliqué en début et fin	Impliqué tout au long	Impliqué tout au long
Itération	Non récurrente	Récurrente	Continuelle

Tableau 1: Tableau comparatif des méthodes

Ce tableau offre une vue comparative générale entre ces trois méthodes, mais il est important de noter que chaque projet et organisation peut adapter ces méthodes en fonction de leurs

besoins spécifiques. Dans le cadre de notre projet nous avons bases sur quatres points pour choisir Agile.

- **Collaboration et équipe** : Agile met l'accent sur la collaboration entre les développeurs et l'équipe de gestion de produit.
- **Cycle de vie du projet :** Agile se concentre sur le cycle de vie du développement, de l'idéation à la complétion du code.
- **Méthodologie de travail** : Agile favorise le développement itératif et les petits lots.
- **Gestion des tâches** : Agile structure les tâches planifiées des développeurs.

Donc le choix porté sur Scrum repose sur de nombreux avantages parmi lesquels nous pouvons citer :

- Implication du client dans le processus de développement ;
- Plus grande autonomie des développeurs ;
- Meilleure collaboration entre les équipes ;
- Meilleure gestion globale des risques

Ainsi, nous avons choisi la méthode SCRUM Agile comme cadre de travail.
Par ailleurs, nous rappelons que cette méthodologie n'a pas été adoptée entièrement, mais nous avons seulement fait appel à ses pratiques.
Dans cette configuration, le chef du Service Pédagogique a assumé le rôle de Product Owner, tandis que notre équipe de développement, composée de trois membres, était encadrée par un Scrum Master. En tant que développeur au sein de l'équipe, j'ai travaillé en binôme avec mon collègue.

Conclusion

Cette section a examiné différentes approches de développement et processus. Notre choix s'est porté sur les méthodologies agiles, avec une emphase particulière sur Scrum, qui sera notre cadre de travail tout au long de ce projet. Ce choix découle de la nécessité d'impliquer étroitement le client dans le processus de développement.

Bien que le choix de la méthodologie soit crucial, nous reconnaissons également l'importance de la communication et de la collaboration au sein de l'équipe pour la réussite du projet. Dans le chapitre suivant, nous explorerons la spécification des besoins, un élément essentiel pour démarrer efficacement le développement.

Chapitre III : Spécification des besoins

Introduction

Ce chapitre se concentre sur la définition des besoins. Tout d'abord, nous identifions les utilisateurs cibles et structurons le système en sous-modules pour simplifier notre approche. Ensuite, nous introduisons les outils et langages de modélisation que nous utilisons. Enfin, nous abordons les exigences fonctionnelles de l'application, les décrivant à l'aide de diagrammes de cas d'utilisation. Ces diagrammes offrent une vue d'ensemble détaillée du fonctionnement de l'application du point de vue de l'utilisateur.

I. Identification des acteurs du système

Dans un système, qu'il soit informatique, social, économique ou de tout autre type, un acteur est une entité ou une composante qui interagit avec ce système de différentes manières. Ces acteurs peuvent être des individus, des groupes, des organisations, des entités ou des éléments qui ont un impact sur le fonctionnement, les performances ou les résultats du système.

I.1. Définition d'un acteur de système

Dans le contexte d'un système informatique, les acteurs peuvent prendre diverses formes, telles que des utilisateurs, des administrateurs, des applications externes ou même des périphériques matériels, car ils ont des interactions directes avec le système[24].

Cependant, comprendre les acteurs d'un système revêt une importance capitale pour analyser son fonctionnement, identifier les relations et les dépendances, et concevoir des solutions efficaces. Les acteurs peuvent jouer des rôles variés, avoir des besoins différents et exercer des niveaux d'influence distincts sur le système, ce qui les rend cruciaux pour une compréhension holistique du système dans son ensemble.

I.2. Les utilisateurs du système

Dans notre système, différents acteurs occupent des rôles distincts :

- **Le profil chef de départements** : il gère tout ce qui est inscription des étudiants dans leur service de stages, faire la rotation pour le passage d'un service à un autre.

- **Le profil maitre de stage :** il est chargé de gérer l'attribution des notes de stages aux étudiants.
- **Le profil secrétaire :** il est chargé de gérer le suivie des etudiants des leurs stages.
- **Le profil responsable pédagogique :** il gère tout ce qui est inscription des étudiants dans la base, dans les hôpitaux, dans les spécialités et la rotation semestrielle.

II. Identification des modules du système

Nous avons fragmenté l'application pour notre cote en cinq sous-modules pour clarifier au mieux ses exigences fonctionnelles :

Sous-module	Descriptions
Inscription	Dans ce sous-module, nous allons gérer l'importation des étudiants dans la base.
Pointage	Dans ce module, nous allons gérer le suivi des étudiants, il nous permettra d'avoir une traçabilité sur les étudiants qui suivent régulièrement les stages.
Evaluation	Dans ce sous-module, nous allons gérer les évaluations de stage des étudiants concernant l'attribution des notes.
Processus d'inscription des étudiants dans le stage	Ce sous-module, est compose de quatre (04), sous-module la gestion de répartition des étudiants dans les hôpitaux, gestion de répartition des étudiants dans les spécialités, gestion de répartition des étudiants dans les services, gestion des rotations.
Gestion des fichiers	Dans ce sous-module, nous allons gérer la génération les résultats de stage individuel, et par promo

Tableau 2: Liste des modules et leurs descriptions

III. L'outil de modélisation

Un outil de modélisation est un logiciel ou une application utilisée pour créer des représentations visuelles de systèmes, processus, données ou concepts dans le but de les analyser, de les comprendre, ou de les communiquer à d'autres personnes. Ces outils permettent de créer différents types de modèles, tels que des diagrammes, des schémas, des cartes conceptuelles, des organigrammes, etc. Ils offrent souvent une variété de fonctionnalités pour créer, éditer, organiser et partager des modèles de manière efficace[25].

PowerAMC représente l'outil de modélisation que nous avons adopté pour élaborer les organigrammes et les diagrammes de cas d'utilisation, ainsi que tous les autres schémas (tels que les diagrammes de classes, d'activités, de séquences, etc.) présentés dans ce document. Dans les paragraphes suivants, nous allons le décrire brièvement avant d'expliquer les raisons de notre sélection.

III.1. Présentation de PowerAMC

POWER AMC est parmi les pionniers des outils de modélisation de données, qu'il s'agisse de MERISE, UML ou d'autres méthodologies, en offrant une approche graphique pour leur élaboration et une implémentation automatisée quel que soit le Système de Gestion de Base de Données (SGBD) utilisé. Il permet également la modélisation des processus métiers. En connectant la modélisation des données à celle des processus, il offre aux entreprises équipées de POWER AMC / AMC Designor la possibilité de créer un référentiel centralisé pour leurs développements et leurs processus, qu'ils soient informatisés ou non[26].

POWER AMC est un atout majeur pour tout nouveau projet d'entreprise, car il permet une identification précise des processus, des personnes et/ou des données impactés. Cela se traduit par une estimation et une maîtrise accrues des coûts.

III.2. Justification du choix de PowerAMC

PowerAMC est un logiciel de modélisation qui est disponible en deux versions lancées simultanément : PowerAMC, basé sur la méthodologie Merise et proposé en français, et PowerDesigner, fondé sur la méthodologie IE et disponible en anglais. En plus de ces deux versions, PowerAMC permet de créer des diagrammes visuels, de générer des modèles (voir Annexe 1) et des codes (voir Annexe 2). Son interface conviviale en facilite l'utilisation.

En résumé, d'un point de vue technique, SAP PowerDesigner offre une vaste gamme de modélisations standardisées et graphiques, dont la très populaire méthodologie Merise, ainsi

que d'autres telles que MCD, MOO, MPM, MSX, MPM, MPD, MFI, MGX, MTM et MAE, toutes disponibles en français.

IV. Langage de modélisation UML

IV.1. Définition

Le langage UML (Unified Modeling Language), ou langage de modélisation unifié, a été conçu pour être un langage visuel partagé, doté d'une richesse sémantique et syntaxique. Son objectif principal est de faciliter l'architecture, la conception et la mise en œuvre de systèmes logiciels complexes, tant au niveau de leur structure que de leur comportement. Bien que principalement utilisé dans le développement logiciel, l'UML trouve également des applications dans d'autres domaines, comme la modélisation des flux de processus industriels[27].

UML présente des similitudes avec les schémas utilisés dans d'autres disciplines et se compose d'une variété de types de diagrammes. Globalement, les diagrammes UML servent à définir les limites, la structure et le comportement du système ainsi que des objets qui le composent.

Il est important de noter que l'UML n'est pas un langage de programmation à proprement parler. Cependant, il existe des outils capables de traduire les diagrammes UML en code dans différents langages de programmation. En fin de compte, l'UML est étroitement lié à l'approche de l'analyse et de la conception orientées objet, fournissant ainsi un cadre puissant pour la modélisation et la compréhension des systèmes logiciels.

IV.2. Les types de diagramme UML

Pour les néophytes, le nombre de diagrammes UML peut paraître infini. En vérité, les normes en identifient 13 types, eux-mêmes répartis en deux groupes, tels que décrits ci-dessous[28].

- D'une part, les diagrammes UML structurels décrivent la structure d'un système et les relations entre ses éléments.

  Ils comprennent six diagrammes : le diagramme de classes, le diagramme de composants, le diagramme de déploiement, le diagramme de structure composite, le diagramme d'objets et le diagramme de paquetages[29].

- D'autre part, les diagrammes UML comportementaux illustrent comment le système interagit avec lui-même, les utilisateurs et les autres systèmes. Ils comprennent sept

diagrammes : le diagramme de cas d'utilisation, le diagramme de temps, le diagramme d'aperçu des interactions, le diagramme de communication, le diagramme de séquence, le diagramme d'activités et le modèle de diagramme états-transitions[30].

En plus de ces deux groupes, il y a également les diagrammes de profil, introduits récemment dans UML 2.0, qui sont utilisés comme mécanisme d'extension pour adapter les modèles UML à des domaines et des plateformes spécifiques.

En résumé, les diagrammes UML sont des outils de communication essentiels pour simplifier la conception et les fonctionnalités d'une application, mais il est crucial qu'ils soient interprétés de la même manière par toutes les parties concernées.

V. Les diagrammes de cas d'utilisation

V.1. Définition

Les diagrammes de cas d'utilisation décrivent simplement comment les différentes personnes ou systèmes utilisent un système donné. Ils sont très utiles pour comprendre les interactions entre les utilisateurs (appelés "acteurs" dans le langage de modélisation) et le système lui-même. En utilisant des formes visuelles simples comme des boîtes et des flèches, ces diagrammes fournissent une vue d'ensemble claire des différentes actions que les utilisateurs peuvent effectuer et de la manière dont ces actions affectent le système[31].

En résumé, les diagrammes de cas d'utilisation sont des outils efficaces pour communiquer le fonctionnement d'un système à un large public, même à ceux qui ne sont pas familiarisés avec les détails techniques.

Dans cette section, nous allons présenter quelques diagrammes de cas d'utilisation et leurs descriptions.

V.2. Gestion des inscriptions

La figure 5 ci-dessous illustre le diagramme de cas d'utilisation pour la gestion des inscriptions. Ce diagramme met en évidence les échanges d'informations entre le responsable pédagogique (RP) et le système lorsqu'il s'agit de gérer la gestion des inscriptions. Il offre une vue d'ensemble claire des actions que le responsable pédagogique peut entreprendre pour assurer la gestion des inscriptions des étudiants. En somme, ce diagramme permet de visualiser de manière succincte les tâches et responsabilités du responsable pédagogique dans le processus de gestion des inscriptions de stage des étudiants.

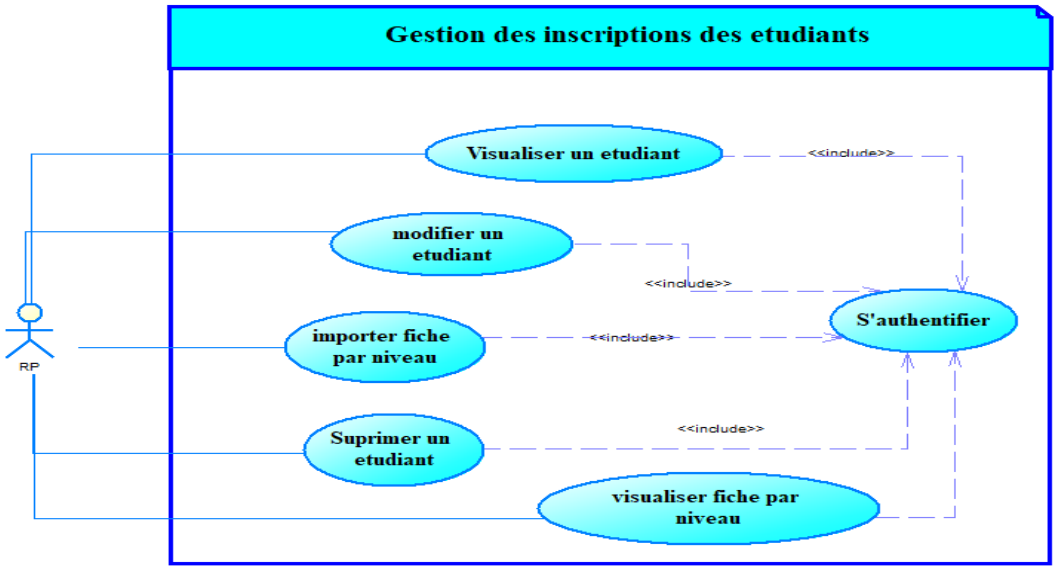


Figure 5: Diagramme de cas d'utilisation pour la gestion des Inscription

- **Description détaillée des cas d'utilisations de la *figure 5* :**

Nom du cas	**importer fiche par niveau**
Acteur principale	Responsable Pédagogique
Objectif	Charger les étudiants dans la base
Pre -condition	Aucune.
Contraintes	Le niveau sélectionné doit correspondre au niveau qui est dans le fichier sélectionné
Scenario normal	-S'authentifier, -Etudiant (Button) -Import (Button) Sélectionner le niveau que l'on voulait importer (par recherche / à partir de la liste déroulante de niveau) -vérifications des informations. -vérifications réussie - import étudiants

Scenario d'échec	-S'authentifier, -Etudiant (Button) -Importer (Button) Sélectionner le niveau que l'on voulait importer (par recherche / à partir de la liste déroulante de niveau) -Choisir fichier (Button). -vérifications des informations. -vérifications échoue. -Erreur : le niveau sélectionné et le niveau qui est dans le fichier ne correspond pas ou bien les informations qui sont dans le ficher ne respectent pas le bon formant -Affichage du Message d'erreur.
Post-Condition	Etudiants importer avec succès

Tableau 3: Description détaillée Du cas « Importer la fiche par niveau »

V.3. Gestion de répartition des étudiants dans les hôpitaux

La figure 6 ci-dessous illustre le diagramme de cas d'utilisation pour la gestion de répartition des étudiants dans les hôpitaux, et spécialités.

Ce diagramme met en évidence les échanges d'informations entre le responsable pédagogique et le système lorsqu'il s'agit de gérer la gestion de répartition des étudiants dans les hôpitaux, et spécialités. Il offre une vue d'ensemble claire des actions que le chef de département peut entreprendre pour assurer la gestion de répartition des étudiants dans les hôpitaux, et spécialités des. En somme, ce diagramme permet de visualiser de manière succincte les tâches et responsabilités du responsable pédagogique dans le processus de gestion des la gestion de répartition des étudiants dans les hôpitaux, et spécialités.

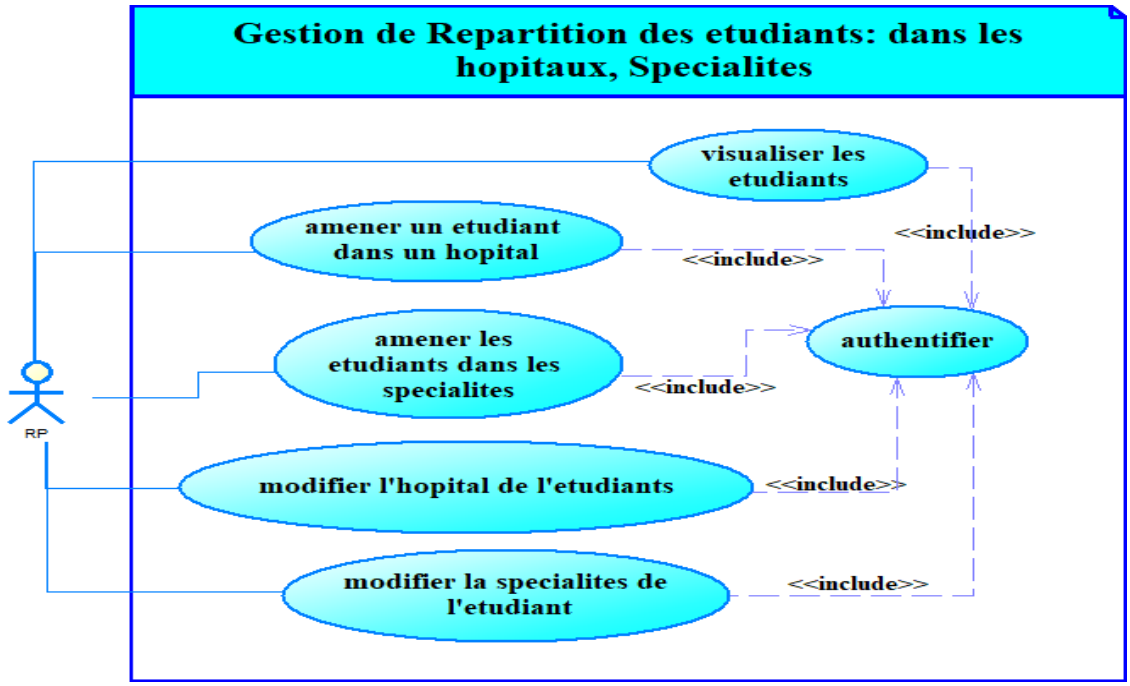


Figure 6: Diagramme de cas d'utilisation pour Gestion de Répartition des étudiants dans les hôpitaux, et Spécialités

- **Description détaillée des cas d'utilisations de la *figure 6* :**

Nom du cas	**Amener les étudiants dans les hôpitaux**
Acteur principale	Responsable Pédagogique
Objectif	Orienter les étudiants dans leurs sites de stage
Pre -condition	S'authentifier, charger les étudiants dans la base
Contraintes	L'orientation doit se faire par niveau et par hôpital
Scenario normal	-Etudiant (Button). -filtrer le niveau a oriente. -sélectionner le nombre d'étudiant qu'on voulait orienter. -Sélectionner l'hôpital concerne (par recherche / à partir de la liste des hôpitaux). -Inscrire (Button). -vérifications des informations. -vérification réussie. -Affichage du Message succès.
	-Etudiant (Button). -filtrer le niveau a oriente. -sélectionner le nombre d'étudiant qu'on voulait orienter. -Sélectionner l'hôpital concerne (par recherche / à partir de la

Scenario d'échec	liste des hôpitaux). -Inscrire (Button). Vérification des informations. -vérification échoué. -Affichage du Message Erreur.
Postcondition	Liste étudiants affichée.

Tableau 4: Description détaillée Du cas « Amener les étudiants dans les hôpitaux »

Nom du cas	**Amener les étudiants dans les spécialités**
Acteur principale	Responsable Pédagogique
Objectif	Orienter les étudiants dans les spécialités
Pre -condition	Il faut amener les étudiants dans les hôpitaux d'abord
Contraintes	L'orientation doit se faire par niveau et par hôpital
Scenario normal	-hôpitaux (Button). -choisir l'hôpital. -Répartition (Button) -Sélectionner niveau concerne (par recherche / à partir de la liste des niveau). -Inscrire (Button). -vérifications des informations. -vérification réussie. -Affichage du Message succès.
Scenario d'échec	hôpitaux (Button). -choisir l'hôpital. -Répartition (Button) -Sélectionner niveau concerne (par recherche / à partir de la liste des niveau). -Inscrire (Button). -vérifications des informations. -vérification échoué. -Affichage du Message erreur.
Post-Condition	Etudiants orientes

Tableau 5: Description détaillée Du cas « Amener les étudiants dans les spécialités »

V.4. Gestion de la répartition des étudiants dans les services

La figure 7 ci-dessous illustre le diagramme de cas d'utilisation pour la gestion la gestion de répartition des étudiants dans les services. Ce diagramme met en évidence les échanges d'informations entre le chef du département et le système lorsqu'il s'agit de gérer la gestion de répartition des étudiants dans les services. Il offre une vue d'ensemble claire des actions que le chef du département peut entreprendre pour assurer la gestion de répartition des étudiants dans les services.

En somme, ce diagramme permet de visualiser de manière succincte les tâches et responsabilités du chef du département dans le processus de gestion de répartition des étudiants dans les services.

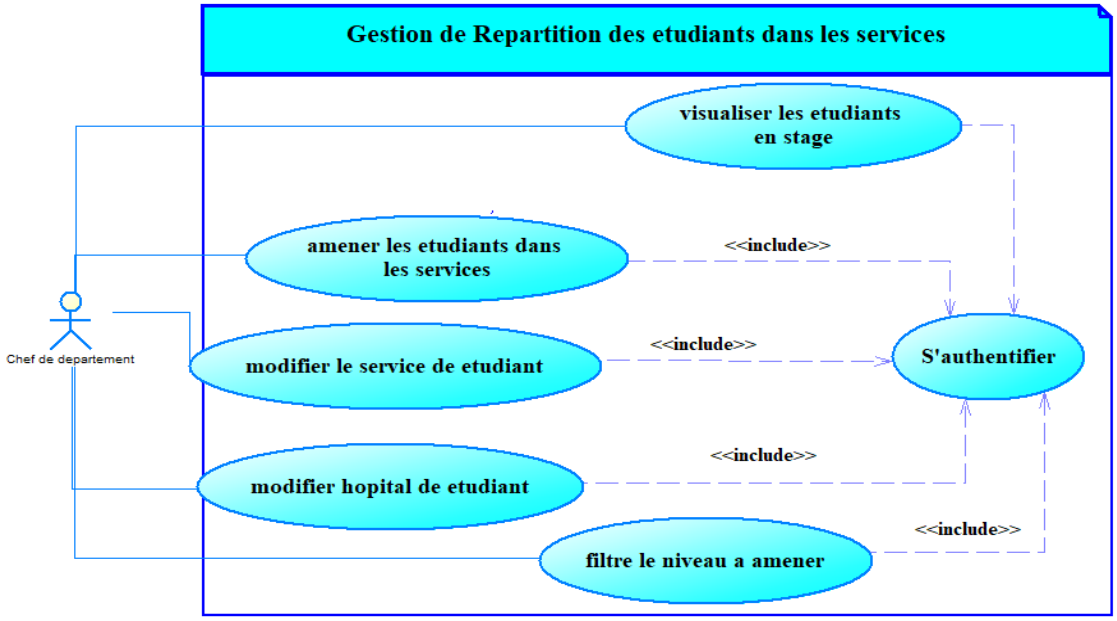


Figure 7: Diagramme de cas d'utilisation pour Gestion de Répartition des étudiants dans les services

- **Description détaillée des cas d'utilisations de la *figure 7* :**

Nom du cas	**Amener les étudiants dans les services**

Acteur principale	Chef de département
Objectif	Orienter les étudiants dans leurs services de stage
Pre -condition	S'authentifier, amener les étudiants dans les hôpitaux et spécialités d'abord
Contraintes	On peut orienter que les étudiants qui sont dans un niveau dont le stage est ouvert
Scenario normal	-Etudiant (Button). -filtrer le niveau a oriente. -sélectionner le nombre d'étudiant qu'on voulait orienter dans le service. -Sélectionner l'hôpital concerne (par recherche / à partir de la liste des hôpitaux). -Inscrire (Button). -vérifications des informations. -vérification réussie. -Affichage du Message succès.
Scenario d'échec	-Etudiant (Button). -filtrer le niveau a oriente. -sélectionner le nombre d'étudiant qu'on voulait orienter dans le service. -Sélectionner l'hôpital concerne (par recherche / à partir de la liste des hôpitaux). -Inscrire (Button). Vérification des informations. -vérification échoué. -Affichage du Message Erreur.
Post-Condition	Etudiants orientes.

Tableau 6: Description détaillée Du cas « Amener les étudiants dans les service »

Nom du cas	**Modifier service / hôpital étudiant.**
Acteur principal	Chef de département.
Objectif	Modifier un service / un hôpital de l'étudiant
Pré condition	S'authentifier, sélectionné l'étudiant concerné.
Contraintes	il faut faire la réaffectation d'un nouveau service au hôpital a l'étudiant que l'on désire modifier son service ou hôpital.

scénario normal	-Modifier (Botton). -Vérification (si l'étudiant n'est pas affecté à un service). -Vérification réussie. -Confirmer la modification. -Affichage du message de succès.
Scénario Alternative	-Modifier (Botton). -Annuler la
Scénario d'échec	Aucun.

Tableau 7: Description détaillée Du cas « modifier service ou hôpital étudiant »

V.5. Gestion des Pointages

La figure 8 ci-dessous illustre le diagramme de cas d'utilisation pour la gestion des pointages. Ce diagramme met en évidence les échanges d'informations entre le secrétaire et le système lorsqu'il s'agit de gérer la gestion des pointages. Il offre une vue d'ensemble claire des actions que le secrétaire peut entreprendre pour assurer la gestion des pointages.

En somme, ce diagramme permet de visualiser de manière succincte les tâches et responsabilités du chef du maitre de stage dans le processus de gestion des pointages.

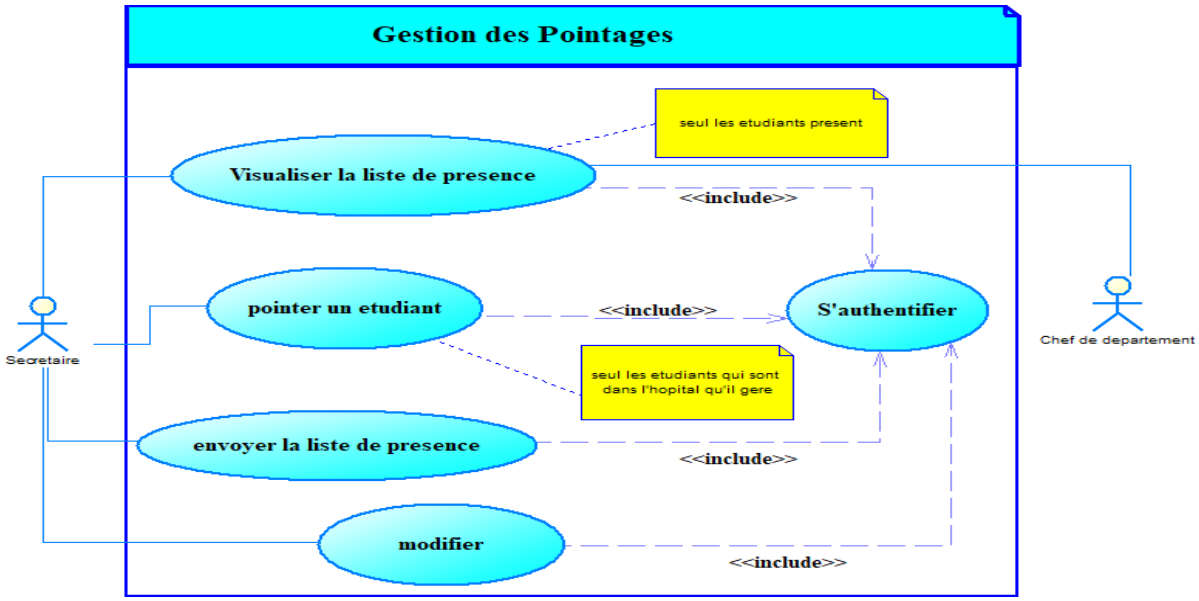


Figure 8: Diagramme de cas d'utilisation pour Gestion des Pointages

- **Description détaillée des cas d'utilisations de la *figure 8* :**

Nom du cas	**Pointer un étudiant.**
Acteur principal	Secrétaire.
Objectif	Le suivie de la présence des étudiants dans les stages
Pré condition	S'authentifier, sélectionné l'étudiant concerné.
Contraintes	Il faut que l'étudiant soit présent a l'arrive et au départ avant de marquer l'étudiant.
Scénario normal	-Etudiant (Botton). -Arriver (Botton). - départ (Botton)
Scénario d'échec	Aucun.

Tableau 8: Description détaillée Du cas «Pointer étudiant»

V.6. Gestion des Evaluations

La figure 9 ci-dessous illustre le diagramme de cas d'utilisation pour la gestion des évaluations. Ce diagramme met en évidence les échanges d'informations entre le maitre de stage et le système lorsqu'il s'agit de gérer la gestion des évaluations. Il offre une vue d'ensemble claire des actions que le maitre de stage peut entreprendre pour assurer la gestion des évaluations.

En somme, ce diagramme permet de visualiser de manière succincte les tâches et responsabilités du maitre de stage dans le processus de gestion des évaluations.

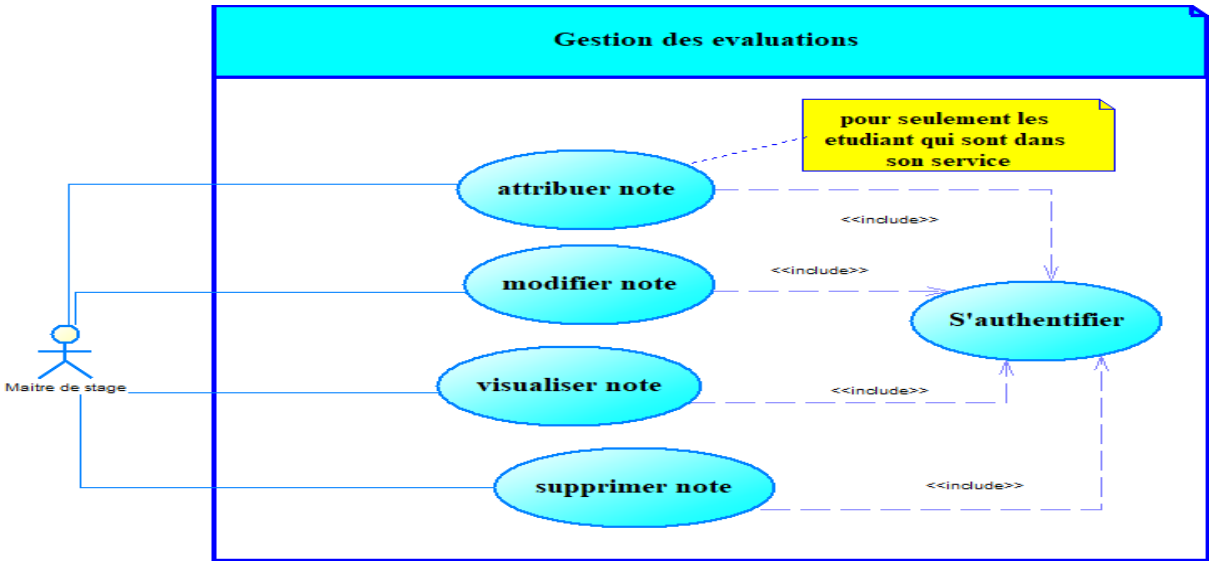


Figure 9: Diagramme de cas d'utilisation pour Gestion des Pointages

- **Description détaillée des cas d'utilisations de la *figure 9* :**

Nom du cas	**Attribuer une note a un étudiant.**
Acteur principal	Maitre de stage.
Objectif	Pour évaluer l'étudiant
Pré condition	S'authentifier. Etudiant (Botton). sélectionné, le niveau le service du maitre de stage et l'hôpital.
Contraintes	Note ne doit pas inferieur a 0 ou supérieur a 20
Scénario normal	-Evaluer (Botton). -Saisir la note. -Enregistrer (Botton). -Vérification des informations -Vérification réussie. - Afficher le message de succès
Scénario d'échec	-Evaluer (Botton). -Saisir la note. -Enregistrer (Botton). -Vérification des informations -Vérification échoué. - Afficher le message de d'erreur.
Post condition	Résultat afficher.

Tableau 9: Description détaillée Du cas « Attribuer une note a un étudiant »

V.7. Gestion des Rotations

La figure 10 ci-dessous illustre le diagramme de cas d'utilisation pour la gestion des rotations. Ce diagramme met en évidence les échanges d'informations entre le chef de département, le responsable pédagogique et le système lorsqu'il s'agit de gérer la gestion des rotations. Il offre une vue d'ensemble claire des actions que le chef de département, le responsable pédagogique peuvent entreprendre pour assurer la gestion des rotations.

En somme, ce diagramme permet de visualiser de manière succincte les tâches et responsabilités du le chef de département, du responsable pédagogique dans le processus de gestion des rotations.

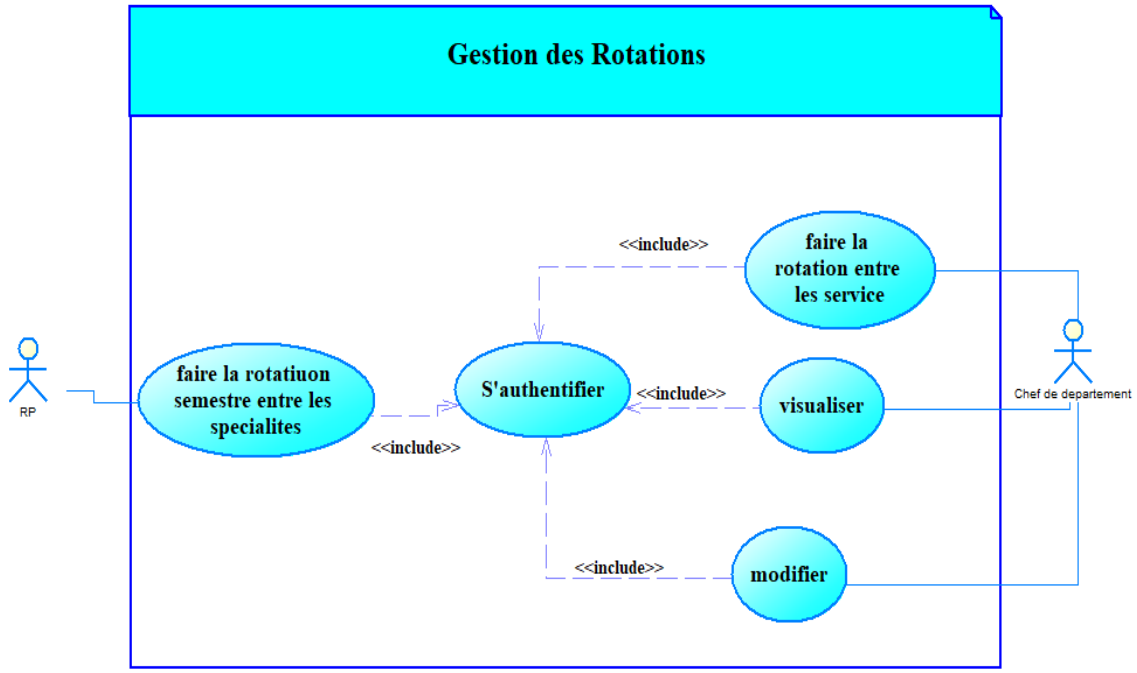


Figure 10: Diagramme de cas d'utilisation pour Gestion des Rotation

- **Description détaillée des cas d'utilisations de la *figure 10***

Nom du cas	**Faire la rotation entre les services.**
Acteur principal	**Chef de département.**

Objectif	Pour faire passe l'étudiant d'un service a l'autre
Pré condition	S'authentifier. Ouverture du service.
Contraintes	Un étudiant ne doit pas passer deux fois dans le même service
Scénario normal	-Etudiant (Button). -filtrer le niveau a oriente. -sélectionner le nombre d'étudiant qu'on voulait orienter dans le service. -Sélectionner l'hôpital concerne (par recherche / à partir de la liste des hôpitaux). -Inscrire (Button). Vérification des informations. -vérification réussie. -Affichage du Message succès.
Scénario d'échec	-Etudiant (Button). -filtrer le niveau a oriente. -sélectionner le nombre d'étudiant qu'on voulait orienter dans le service. -Sélectionner l'hôpital concerne (par recherche / à partir de la liste des hôpitaux). -Inscrire (Button). Vérification des informations. -vérification échoué. -Affichage du Message Erreur. -Vérification échoué. - Afficher le message de d'erreur.
Post condition	Résultat afficher.

Tableau 10: Description détaillée Du cas « Faire la rotation entre les service »

Nom du cas	**Faire la rotation entre les spécialités.**
Acteur principal	Responsable Pédagogique.
Objectif	Pour faire passe l'étudiant d'une spécialité a l'autre
Pré condition	S'authentifier. Démarrer le semestre 2.
Contraintes	Aucune

Scénario normal	-hôpitaux (Button). - rotation (Button).
Scénario d'échec	Aucune
Post condition	**Résultat afficher.**

Tableau 11: Description détaillée Du cas « Faire la rotation entre les services »

V.8. Gestion des Fichiers

La figure 11 ci-dessous illustre le diagramme de cas d'utilisation pour la gestion des rotations. Ce diagramme met en évidence les échanges d'informations entre le chef de département, le responsable pédagogique et le système lorsqu'il s'agit de gérer la gestion des rotations. Il offre une vue d'ensemble claire des actions que le chef de département, le responsable pédagogique peuvent entreprendre pour assurer la gestion des rotations.

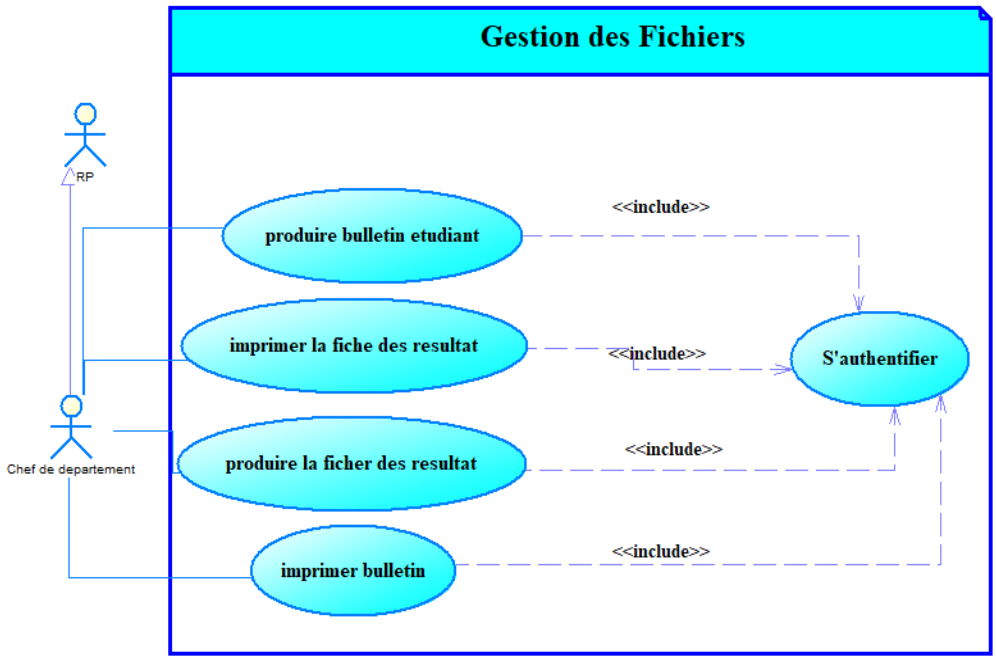


Figure 11: Diagramme de cas d'utilisation pour Gestion des fichiers

- **Description détaillée des cas d'utilisations de la *figure 11* :**

Nom du cas	**Faire la rotation entre les services.**
Acteur principal	Chef de département, Responsable pédagogique
Objectif	Pour avoir les résultats sous formes papier
Pré condition	S'authentifier.
Contraintes	Aucune
Scénario normal	-historique (Button). -sélectionner le matricule d'étudiant qu'on voulait produire son bulletin. -Saisir l'année et le semestre qu'on voulait produire son bulletin. -Produire (Button). Vérification des informations. -vérification réussie. -Affiche le bulletin.
Scénario d'échec	-historique (Button). -sélectionner le matricule d'étudiant qu'on voulait produire son bulletin. -Saisir l'année et le semestre qu'on voulait produire son bulletin. -Produire (Button). -Vérification des informations. -vérification échoué. - Afficher le message de d'erreur.
Post condition	Résultat Afficher.

Tableau 12: Description détaillée Du cas « Produire Bulletin »

VI. Les modules et leurs cas d'utilisations

Le tableau suivant répertorie les cas d'utilisation de chaque sous-module.

Sous-Module	**Cas d'utilisation**
Inscription	Importer fiche par niveau, supprimer un étudiant, modifier un étudiant, visualiser un étudiant, visualiser la fiche par niveau,

	ajouter un étudiant
Processus d'orientation des étudiants dans le stage	Gestion des de répartition dans les hôpitaux, gestion des répartitions dans les spécialités, gestion de répartition dans les services, gestion des rotation, ajouter un étudiant dans un hôpital / spécialité / service
Pointage	Pointer un étudiant, envoyer la liste de présence, modifier, visualiser
Evaluation	Attribuer note, modifier note, visualiser note, supprimer note
Gestion des Fichiers	Produire bulletin, imprimer bulletin, produire fiche des résultats, imprimer fiche des résultats

Tableau 13: Les sous-modules et leurs cas d'utilisations

Conclusion

Ce chapitre a permis de classifier les intervenants de notre système en deux catégories : les administrateurs et les autres membres du personnel de la GDS.

Ensuite, nous avons organisé les fonctionnalités nécessaires de notre futur système en modules et sous-modules, que nous avons représentés à l'aide de diagrammes décrivant les actions possibles.

Enfin, nous avons compilé un récapitulatif des actions possibles pour chaque module et sous-module, avant de présenter les outils et le langage utilisés pour cette modélisation. Après avoir identifié les intervenants et leurs actions possibles, nous aborderons l'analyse et la conception dans le prochain chapitre.

Chapitre IV : Analyse des besoins et conception

Introduction

La phase d'analyse et de conception représente une étape essentielle dans le processus de développement d'un système, visant à structurer les premières étapes de manière à répondre au mieux aux besoins du client.

Ce chapitre se focalise sur l'analyse et les différents aspects de la conception, notamment la conception générale et détaillée.

Dans la partie dédiée à l'analyse, nous examinerons les diagrammes d'activités ainsi que certains diagrammes de séquences. Pour ce qui est de la conception générale, nous exposerons l'architecture physique choisie pour notre application.

Quant à la conception détaillée, elle sera illustrée par certains diagrammes de classes.

Par ailleurs, en annexe nous présentons le dictionnaire de données

I. Étude des Exigences

Dans cette partie, nous allons vous présenter deux diagrammes d'activités : l'un portant sur la gestion de répartition des étudiant dans les hôpitaux et l'autre sur la gestion de répartition des étudiants dans les services.

De plus, nous aurons le plaisir de vous exposer trois diagrammes de séquences intéressants : l'un dédié à l'attribution de note, et les autres respectivement axés sur le pointage d'un étudiant et l'autre sur la rotation entre les services.

I.1. Les diagrammes d'activités

I.1.1. Définition d'un diagramme d'activité

Dans le cadre du langage UML, un diagramme d'activité offre une représentation du comportement d'un système en détaillant la séquence des actions d'un processus. Ces diagrammes, similaires aux organigrammes de traitement de l'information, mettent en évidence les liaisons entre les différentes actions au sein d'une activité. Toutefois, ils permettent également de visualiser les flux parallèles simultanés et les remplacements de flux[32].

Dans la conception des diagrammes d'activité, l'utilisation de nœuds d'activité et de transitions permet de modéliser le cheminement des commandes et des données entre les différentes actions.

Les diagrammes d'activité s'avèrent utiles à diverses étapes d'un projet :

- En amont du projet, ils servent à esquisser les principaux flux de travail.
- Durant la phase de spécification des exigences, ils permettent de représenter graphiquement les séquences d'événements décrites dans les cas d'utilisation.
- Pendant les phases d'analyse et de conception, leur utilisation facilite la définition du comportement des opérations.

I.1.2. Diagramme d'activité répartition dans les hôpitaux

Pour gérer la gestion de répartition des étudiants dans les hôpitaux, l'utilisateur doit s'authentifier en tant que Responsable Pédagogique pour avoir accès à cette interface. La figure ci-dessous (***figure 12***) renseigne les étapes à suivre.

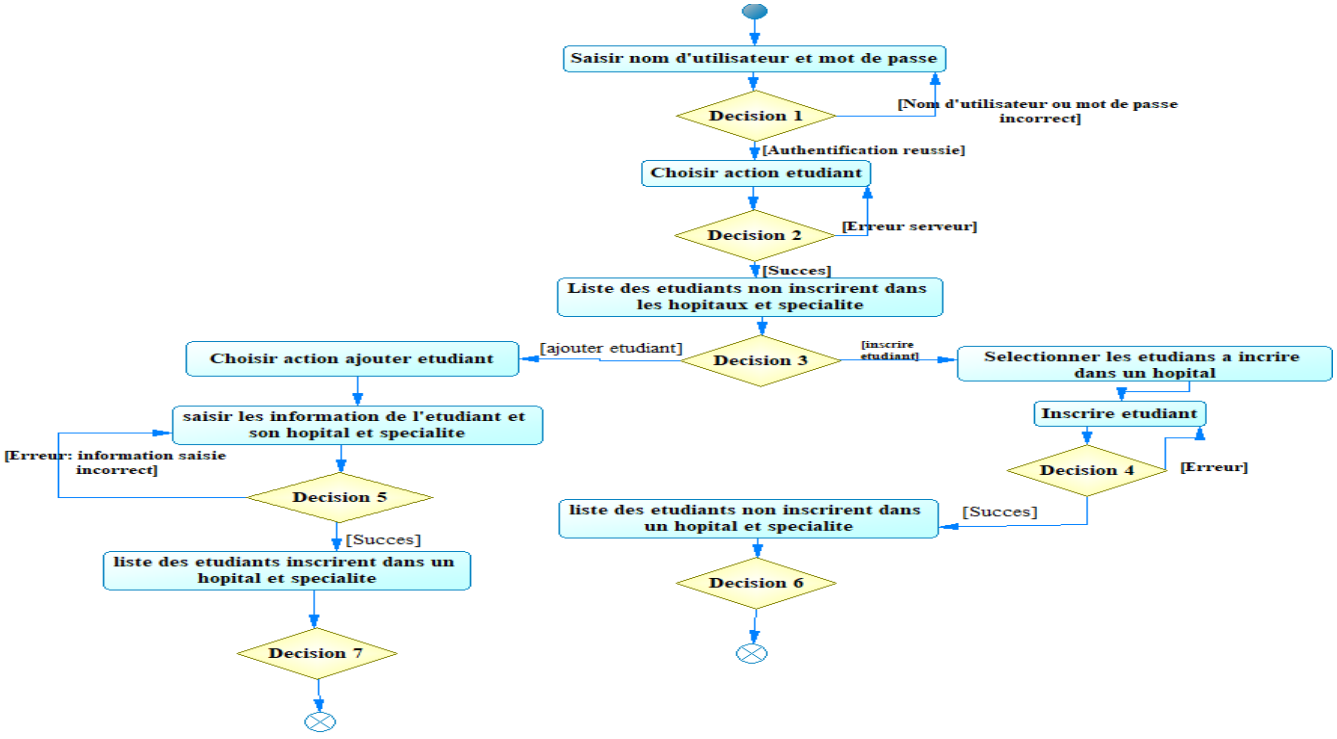


Figure 12: Diagramme d'activités pour la répartition des étudiants dans les hôpitaux.

La figure ci-dessus représente le diagramme d'activités permettant de visualiser les actions faites pour la gestion des répartitions dans les hôpitaux, et spécialistes.

I.1.3 Diagramme d'activité répartition dans les services

Pour gérer la gestion de répartition des étudiants dans les services, l'utilisateur doit s'authentifier en tant que chef de département pour avoir accès à cette interface. La figure ci-dessous (***figure 13***) renseigne les étapes à suivre.

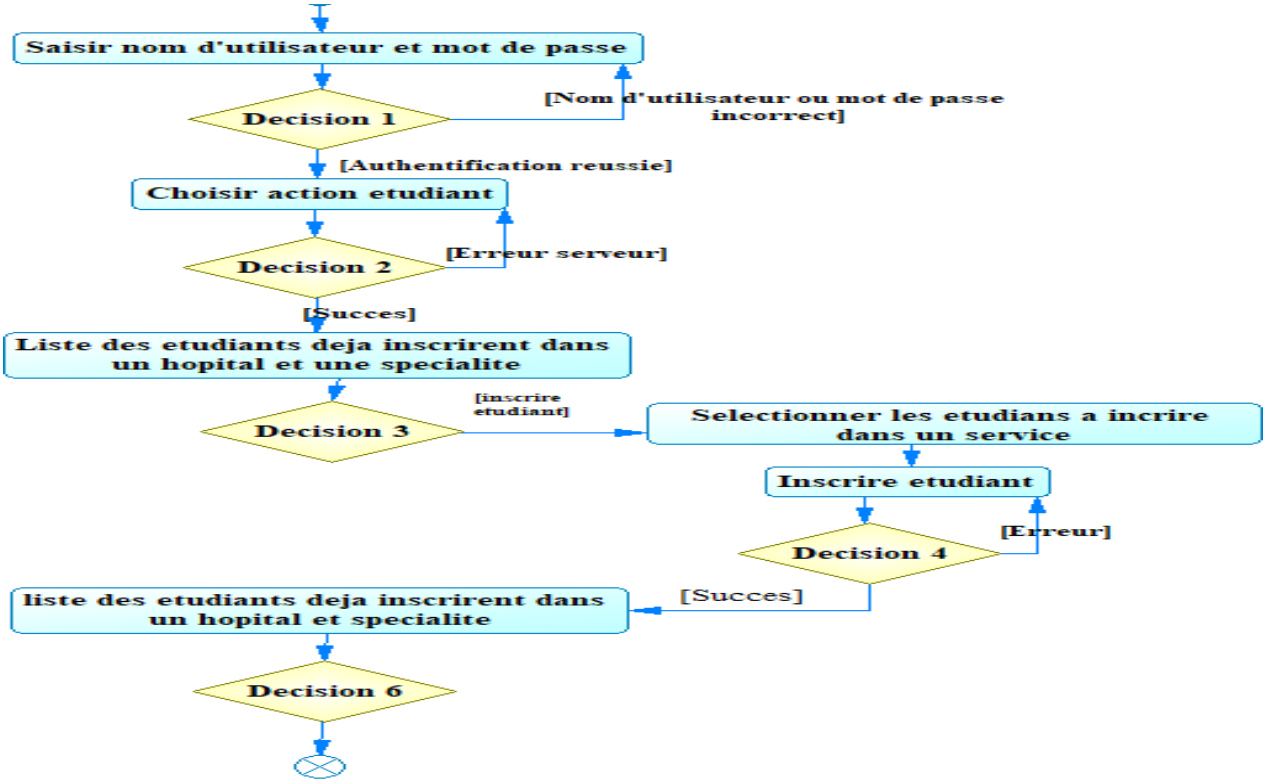


Figure 13: Diagramme d'activités pour la répartition des étudiants dans les services

La figure ci-dessus représente le diagramme d'activités permettant de visualiser les actions faites pour la gestion des répartitions dans les services.

I.2. Les diagrammes de séquences

I.2.1. Définition d'un diagramme de séquence

Un diagramme de séquence représente graphiquement les interactions entre plusieurs objets en indiquant comment ils fonctionnent ensemble et dans quel ordre. Ces schémas sont essentiels pour les développeurs de logiciels et les gestionnaires d'entreprise, car ils permettent d'analyser les besoins d'un système en développement ou de documenter des

processus déjà existants. Ils sont parfois désignés sous le nom de diagrammes d'événements ou de scénarios d'événements[33].

I.2.2. Diagramme de séquence d'attribution de notes

La figure ci-dessous (***figure ##***) est le diagramme de séquences du processus d'attribution des notes de stage des étudiants.

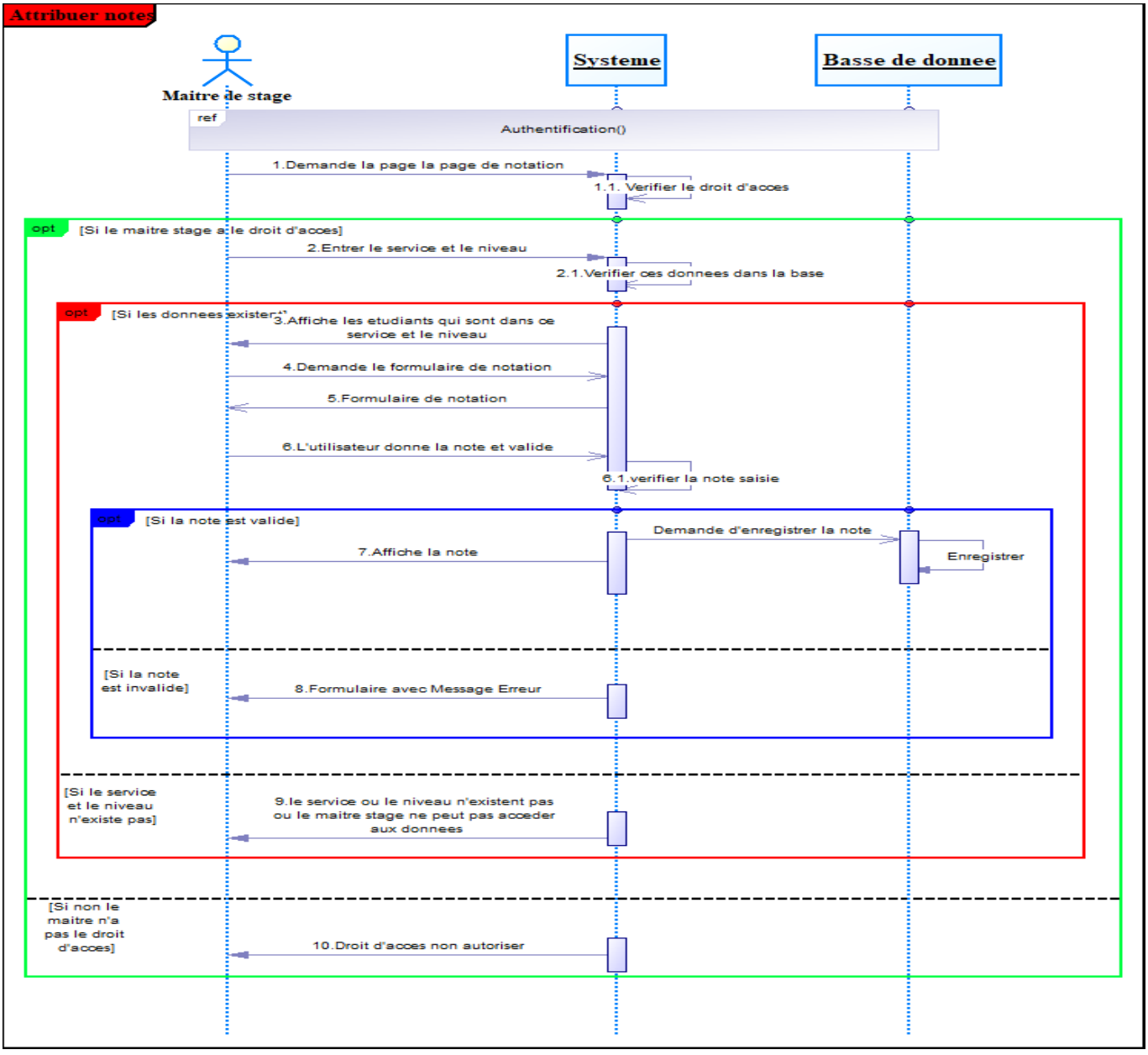


Figure 14: Diagramme de séquences d'attribution de note

Le tableau ci-après (***tableau 13***) représente la description détaillée qui accompagne le schéma ci-dessus :

Scénario principal	
0-Authentification	
1-Demande la page de notation des étudiants	1.1-Le système vérifie les droits d'accès du maitre de stage
Scénario secondaire 1 – 1	
Si le maitre de stage a le droit d'accès	2- Entrez le service et le niveau qu'il voulait évaluer
Scénario secondaire 2 – 1	
	2.1-Le système vérifie l'existence des informations dans la base de données.
Si l'information existe dans la base de données	3-Le système affiche les étudiants qui sont dans ce service et le niveau
4- Demande de formulaire de notation	5-Le système affiche : le formulaire de notation.
6-L'utilisateur donne des notes et valide	6.1-Le système vérifie les informations saisies
Scénario secondaire 3 – 1	
Si les notes saisies sont valides	7-Le système enregistre affiche : la note crée
Scénario secondaire 3 – 2	
Si la note saisie n'est pas valide	Le système affiche : le formulaire avec les messages d'erreur
Scénario secondaire 2 – 2	
Si le service ou le niveau n'existe pas dans la base de données	Le système affiche : service ou niveau non trouve
Scénario secondaire 1 – 2	
Si le maitre de stage n'a pas le droit d'accès	8- Le système n'affiche pas les étudiants mais l'erreur accès non autoriser

Tableau 14: Description détaillée du diagramme de séquence d'attribution de notes

I.2.3. Diagramme de séquence pour le pointage

La figure ci-dessous (***figure 15***) est le diagramme de séquences du processus de pointage des étudiants.

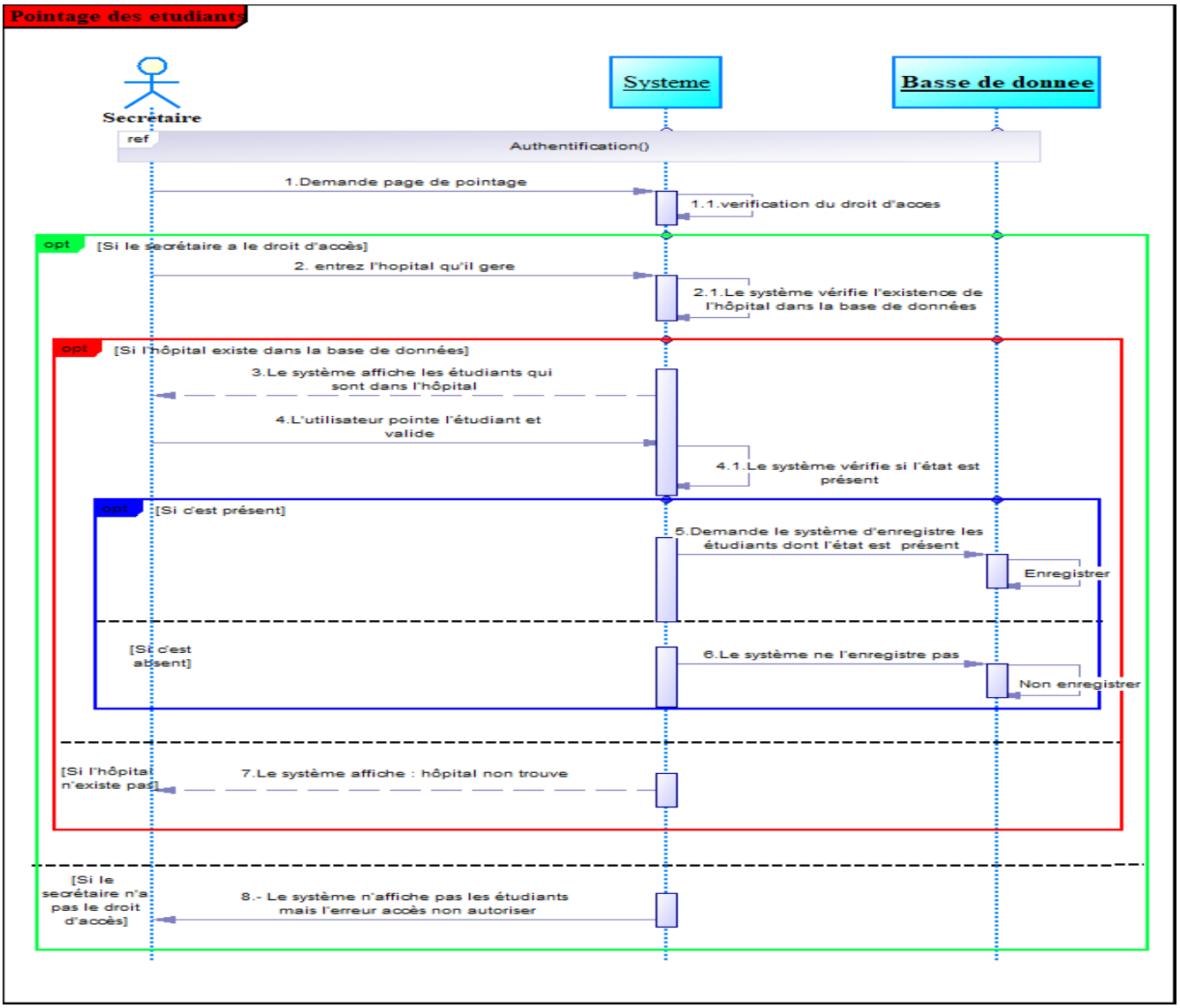


Figure 15: Diagramme de séquences pour le Pointage

Le tableau ci-après (***tableau 14***) représente la description détaillée qui accompagne le schéma ci-dessus :

Scénario principal	
0-Authentification	
1-Demande la page de pointage	1.1-Le système vérifie les droits d'accès du secrétaire
Scénario secondaire 1 – 1	
Si le secrétaire a le droit d'accès	2- Entrez l'hôpital qu'il gère
Scénario secondaire 2 – 1	
	2.1-Le système vérifie l'existence de l'hôpital dans la base de données.
Si l'hôpital existe dans la base de données	3-Le système affiche les étudiants qui sont dans l'hôpital

4-L'utilisateur pointe l'étudiant et valide	4.1-Le système vérifie si l'état est présent
Scénario secondaire 3 – 1	
Si c'est présent	5-Le système enregistre les étudiants dont l'état est présent
Scénario secondaire 3 – 2	
Si c'est absent	6-Le système ne l'enregistre pas
Scénario secondaire 2 – 2	
Si l'hôpital n'existe pas	7-Le système affiche : hôpital non trouve
Scénario secondaire 1 – 2	
Si le secrétaire n'a pas le droit d'accès	8- Le système n'affiche pas les étudiants mais l'erreur accès non autoriser

Tableau 15: Description détaillée du diagramme de séquence pour le Pointage

I.2.4. Diagramme de séquence pour la rotation

La figure ci-dessous (***figure 16***) est le diagramme de séquences du processus de pointage des étudiants.

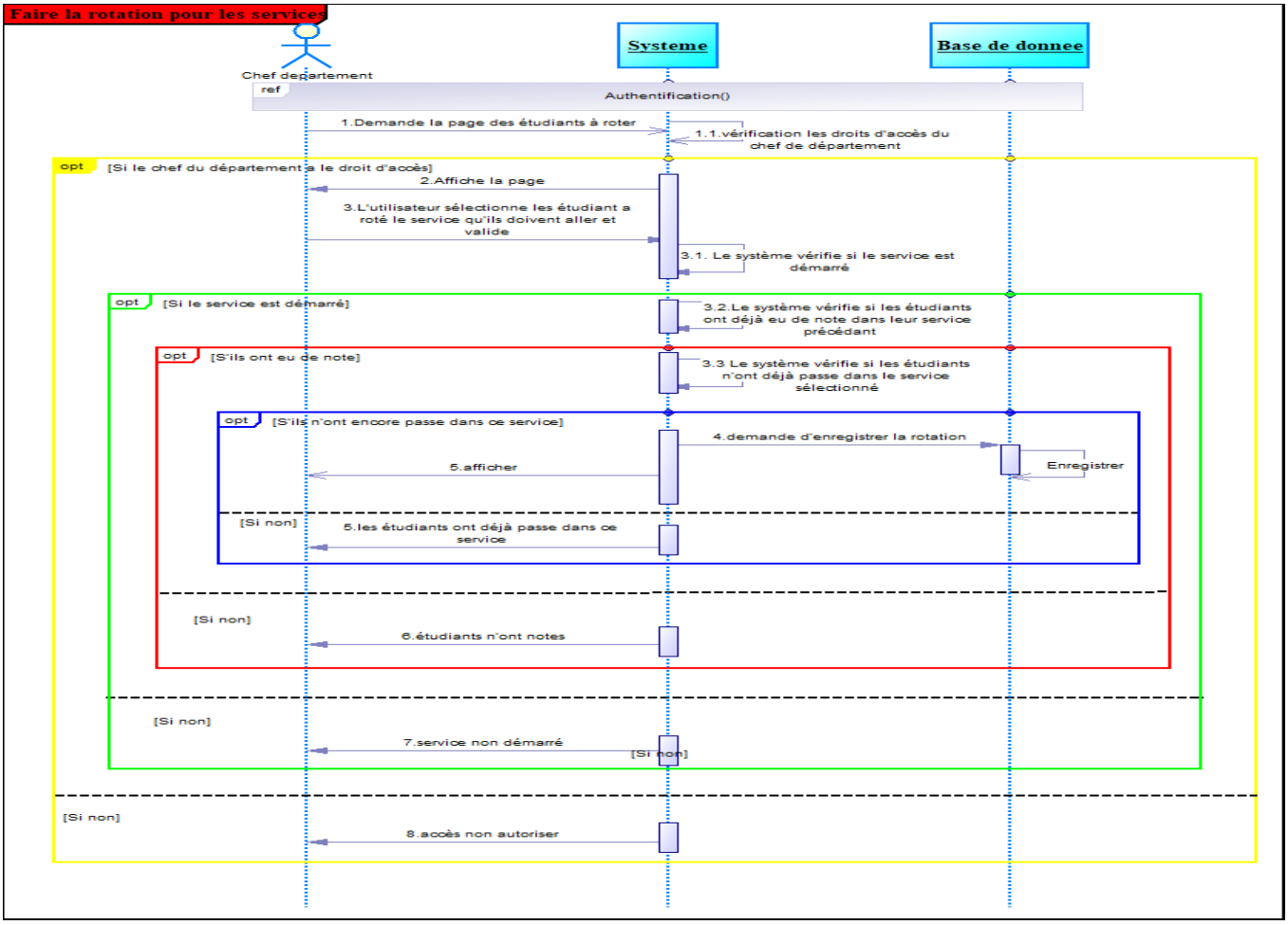


Figure 16: Diagramme de séquences pour la rotation

Le tableau ci-après (***tableau 15***) représente la description détaillée qui accompagne le schéma ci-dessus :

Scénario principal	
0-Authentification	
1-Demande la page des étudiants à roter	1.1-Le système vérifie les droits d'accès du chef de département
Scénario secondaire 1 – 1	
Si le chef du département a le droit d'accès	2- affiche la page
Scénario secondaire 2 – 1	
3-L'utilisateur sélectionne les étudiant a roté le service qu'ils doivent aller et valide	3.1-Le système vérifie si le service est démarré
Scénario secondaire 3 – 1	
Si le service est démarré	3.2-Le système vérifie si les étudiants ont déjà eu de note dans leur service précédant
Scénario secondaire 4 – 1	
S'ils ont eu de note	3.3-Le système vérifie si les étudiants n'ont déjà passe dans le service sélectionné
Scénario secondaire 5 – 1	
S'ils n'ont encore passe dans ce service	4-Le système demande d'enregistrer la rotation : affiche les étudiants avec leur nouveaux service
Scénario secondaire 4– 2	
Si les étudiants ont passé dans le service	5-Le système affiche un message d'erreur : les étudiants ont déjà passe dans ce service
Scénario secondaire 3– 2	
Si les étudiants n'ont encore de note pour le service précédant	6- Le système affiche l'erreur étudiants n'ont notes
Scénario secondaire 2 – 2	
Si le service n'est pas ouvert	7- Le système affiche l'erreur service non démarré
Scénario secondaire 1– 2	
Si le chef du département n'a pas le droit d'accès	8-Le système n'affiche pas les étudiants mais l'erreur accès non autoriser

Tableau 16: Description détaillée du diagramme de séquence pour la Rotation

II. Conception générale

La conception générale de l'architecture, également appelée conception architecturale, se réfère au processus de création et de planification d'un système ou d'une structure dans le domaine de l'architecture, que ce soit pour des bâtiments physiques ou des systèmes logiciels. Cette phase initiale est cruciale car elle définit les principes fondamentaux, les objectifs, les contraintes et les grandes lignes du projet.[34]

II.1. Architecture physique de l'application

L'architecture physique d'une application se réfère à la manière dont les composants logiciels sont déployés sur une infrastructure matérielle spécifique, incluant les aspects relatifs au matériel, à la mise en réseau, au stockage des données, et autres éléments nécessaires pour assurer le fonctionnement de l'application[35].

Les différents modèles d'architectures physiques pour le développement définissent les structures de déploiement des applications en tenant compte des exigences en termes de performances, d'évolutivité, de sécurité et de disponibilité. Quelques-unes de ces architectures sont présentées ci-après :

II.1.1. L'architecture a deux niveaux ou 2-tiers

L'architecture à deux niveaux, aussi connue sous le nom d'architecture 2-tiers, décrit les systèmes client/serveur où le client demande une ressource et le serveur la fournit directement, sans recourir à une autre application pour fournir le service. En résumé, dans ce modèle, l'interface utilisateur se trouve sur le poste client tandis que la base de données est hébergée sur le serveur. La logique de traitement peut être répartie entre les deux parties. Les ordinateurs clients sont généralement connectés aux serveurs de base de données via un réseau local.

Dans ce schéma, l'utilisateur contrôle le poste client, qui effectue une grande partie du traitement de l'application et demande des informations ou des traitements SQL à un ou plusieurs serveurs. Une partie de la logique de gestion réside sur le serveur sous forme de procédures stockées. L'aspect central du serveur est sa capacité à répondre, de préférence simultanément, aux demandes de plusieurs clients. Cette architecture est particulièrement adaptée aux environnements de calcul distribué lorsque le nombre d'utilisateurs reste limité, généralement à moins d'une centaine[36], cependant il existe :

- D'une part une limite tenant au fait que la connexion est maintenue en permanence entre le client et le serveur, même si aucun travail n'est effectué,
- D'autre part les procédures d'accès aux données étant spécifiques aux moteurs de base de données, la flexibilité et le choix d'une.

L 'architecture à deux niveaux fonctionne selon le schéma illustré a la **figure 17** ci-dessous

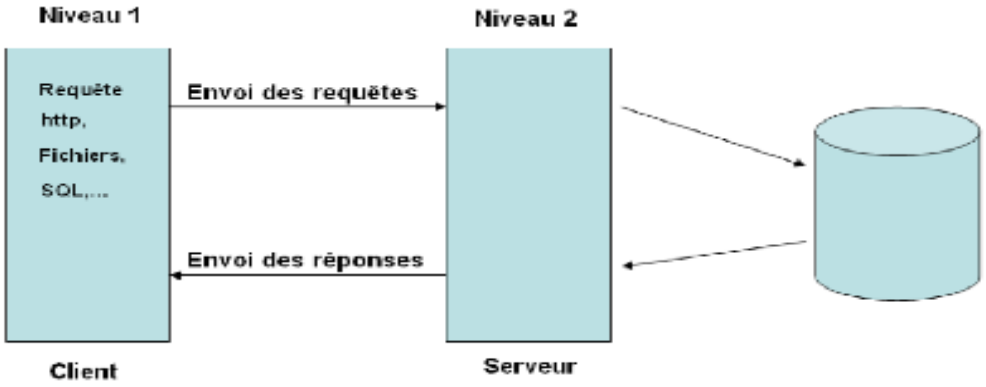


Figure 17: Architecture 2-tiers

II.1.2. L'architecture a trois niveaux ou 3-tiers

L'architecture 3-tiers est un modèle très connu et répandu. C'est une spécialisation du modèle multi-tiers qui propose de découper l'architecture logique en 3 couches[37].

Ce modèle d'architecture se décompose en 3 couches logiques bien distinctes qui ont chacune un rôle bien défini :

- La **couche de présentation** correspond à l'interface utilisateur. Son rôle est d'afficher les données et de permettre à l'utilisateur final d'interagir avec ces dernières.
- La **couche métier** est en charge d'appliquer et de respecter les règles métiers (ou actes de gestion). Avec ce modèle d'architecture, la logique applicative et la sécurité sont implémentées dans cette couche.
- La **couche d'accès aux données**, quant à elle, assure la persistance des données qui doivent être conservées.

Avec ce modèle, chaque couche dialogue avec une autre au travers d'un contrat d'interface. Par convention, la couche de données est la couche la plus basse et la couche de présentation la couche la plus haute. Chaque couche expose alors des services à la couche supérieure.

L 'architecture à trois niveaux fonctionne selon le schéma illustré a la **figure 18** ci-dessous

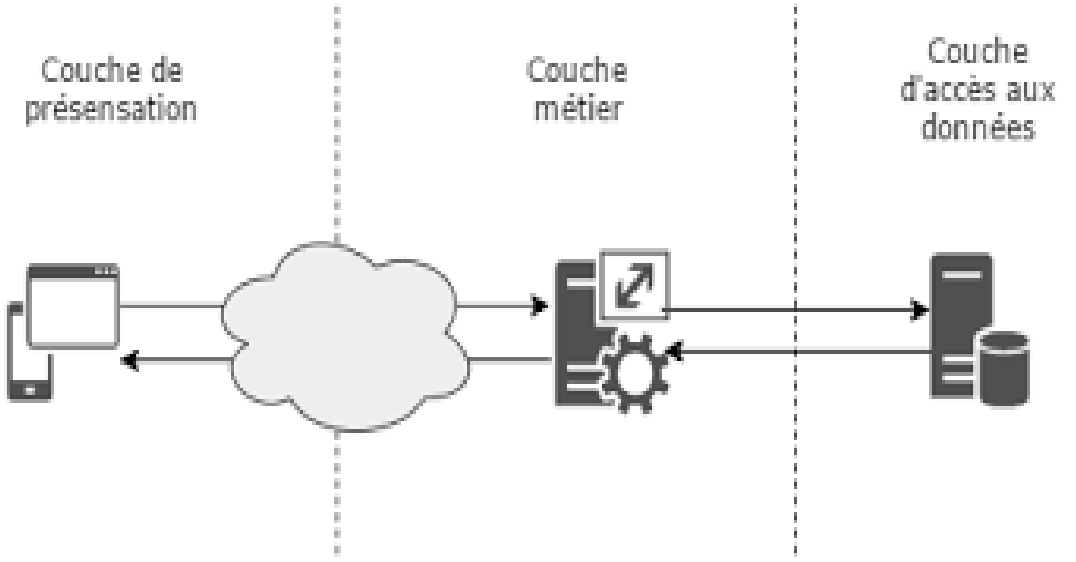


Figure 18: Architecture 3-tiers

Ce découpage permet de bien séparer les responsabilités. Chaque couche peut ainsi évoluer sans impacter les autres.

II.2. Justification du choix de l'architecture physique

Le choix de l'architecture est crucial pour garantir les avantages dans la maintenance et le développement d'une application. Il est souvent déterminé par les besoins spécifiques de l'application et les contraintes du projet. Nous avons opté pour une architecture physique en 3-tiers pour les raisons suivantes :

- **Séparation des responsabilités** : Chaque couche a des tâches spécifiques, ce qui facilite la maintenance et les mises à jour.
- **Réutilisabilité** : La logique métier peut être utilisée par différentes interfaces utilisateur, favorisant ainsi la réutilisation du code.
- **Évolutivité** : Chaque couche peut être mise à l'échelle indépendamment des autres, ce qui permet une croissance flexible de l'application.
- **Flexibilité** : Les technologies utilisées dans chaque couche peuvent être sélectionnées indépendamment les unes des autres, à condition de respecter les interfaces définies.

- **Performances** : Les clients peuvent se concentrer sur l'interface utilisateur et la réactivité en déléguant les tâches lourdes de calcul ou de traitement des données au serveur, ce qui améliore les performances globales de l'application.

II.3. Architecture logique de l'application

L'architecture logique d'une application est la manière dont les différents modules, couches ou composants sont structurés et interconnectés pour garantir un fonctionnement global cohérent et efficace. Elle joue le rôle de guide principal dans le développement de l'application et fournit un cadre pour sa mise en œuvre technique ultérieure. Le choix de l'architecture logicielle est une décision capitale qui influence la conception, le développement, la maintenance et l'évolutivité de l'application[38].

Il existe plusieurs architectures logicielles, chacune avec ses propres avantages, inconvénients et cas d'utilisation appropriés. Parmi les architectures logicielles, nous avons :

II.3.1. L'architecture monolithique

Une architecture monolithique est le modèle classique et unifié de conception d'un programme informatique.

Dans ce contexte, le terme "monolithique" implique une structure unifiée, formée d'un seul bloc. Un logiciel monolithique est conçu pour être autonome ; ses composants sont étroitement interconnectés et interdépendants, plutôt que d'être associés de manière flexible comme dans le cas des programmes modulaires. Dans cette architecture intégrée de manière étroite, chaque composant et ses dépendances doivent être présents pour permettre l'exécution ou la compilation du code[39].

La figure ci-après (***figure 19***) est une illustration l'architecture monolithique.

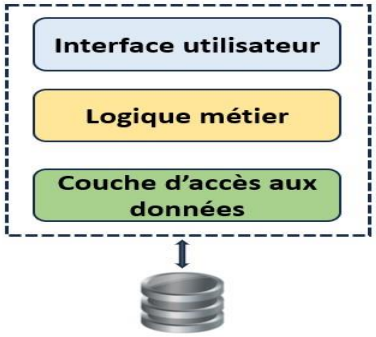


Figure 19: Exemple architecture monolithique

II.3.2. L'architecture orientée services (SOA)

L'architecture orientée services (SOA pour Service Oriented Architecture en anglais) est un modèle de conception logicielle où les services sont les composants fondamentaux et les unités de déploiement. Dans SOA, les services sont des unités autonomes, distribuées et réutilisables qui peuvent être exposées et appelées via des interfaces standardisées sur un réseau. Ces services fournissent des fonctionnalités bien définies et sont conçus pour être indépendants les uns des autres, ce qui permet une flexibilité, une évolutivité et une réutilisation accrues dans le développement et l'intégration des systèmes logiciels. En utilisant SOA, les organisations peuvent construire des applications en assemblant des services existants plutôt qu'en développant des solutions à partir de zéro, ce qui favorise une architecture modulaire et une agilité accrue dans le paysage des applications[40].

La figure ci-dessous (***figure 20***) est une illustration l'architecture SOA

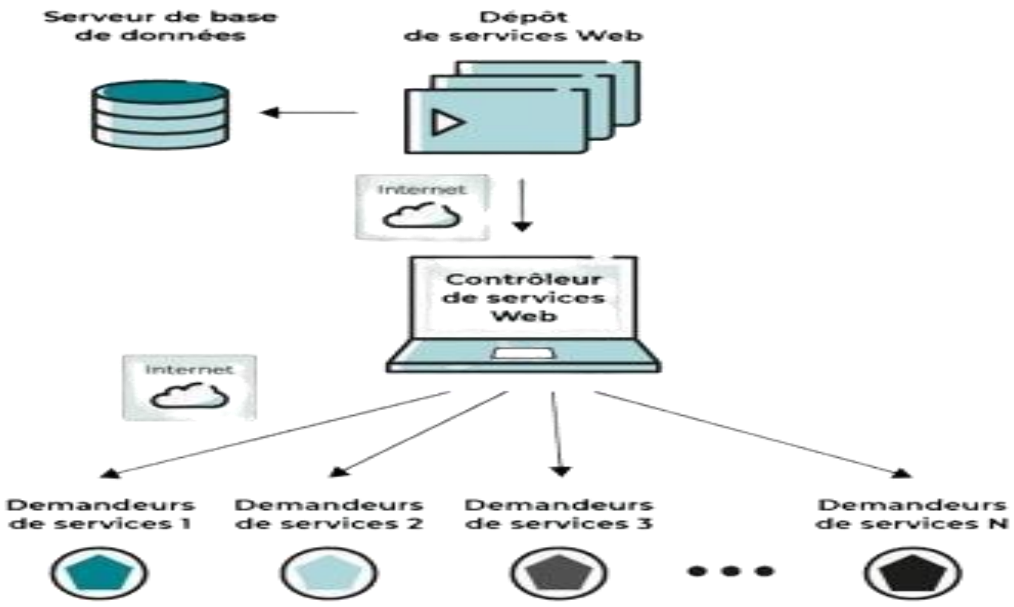


Figure 20: Exemple architecture orienté service [https://openclassrooms.com/fr/courses]

II.3.3. L'architecture micro services

L'idée de cette architecture est de **diviser et de décomposer l'application en plusieurs petits (micro) services indépendants**, l'objectif est d'avoir une série de services modulaires.

Chaque module prend en charge un objectif métier spécifique et utilise une interface simple et bien définie pour communiquer avec d'autres modules[41].

La figure ci-après (***figure 21)*** est une illustration l'architecture micro services.

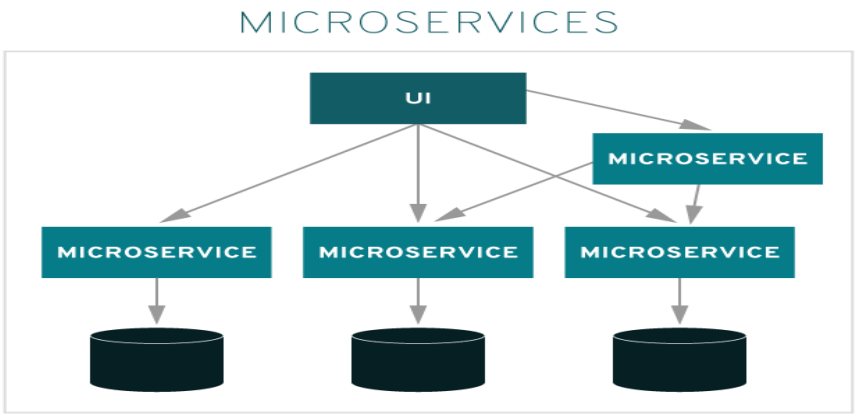


Figure 21: Exemple d'architecture micro services

II.4. Justification du choix de l'architecture logique

Nous avons envisagé l'architecture micro services comme une option prometteuse. Cependant, étant donné que notre application actuelle se concentre uniquement sur la gestion des stages, nous avons décidé de développer ce module sous forme de services tout en adoptant une architecture monolithique pour chaque aspect de l'application. Nous avons choisi d'utiliser le modèle en couches pour le Back-end et le modèle MVC pour le Front-end.

- L'approche en couches permet une structuration claire du code en utilisant différentes couches, ce qui simplifie la maintenance et l'évolution de l'application. Il est important de noter que ces couches peuvent être subdivisées ou combinées en fonction des besoins spécifiques de notre application[42].

La figure ci-après (***figure 22***) est une représentation l'architecture en couche.

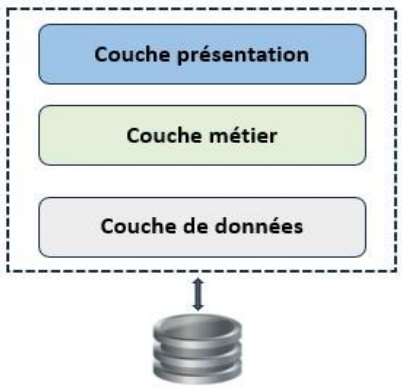


Figure 22: Architecture en couches

- Le modèle MVC offre une architecture bien organisée et modulaire qui encourage le développement, la maintenance et l'amélioration des applications logicielles. Cette approche nous permet de travailler de manière plus efficace et de concevoir des applications robustes et facilement compréhensibles.

La figure ci-après (***figure 23***) est une illustration l'architecture MVC.

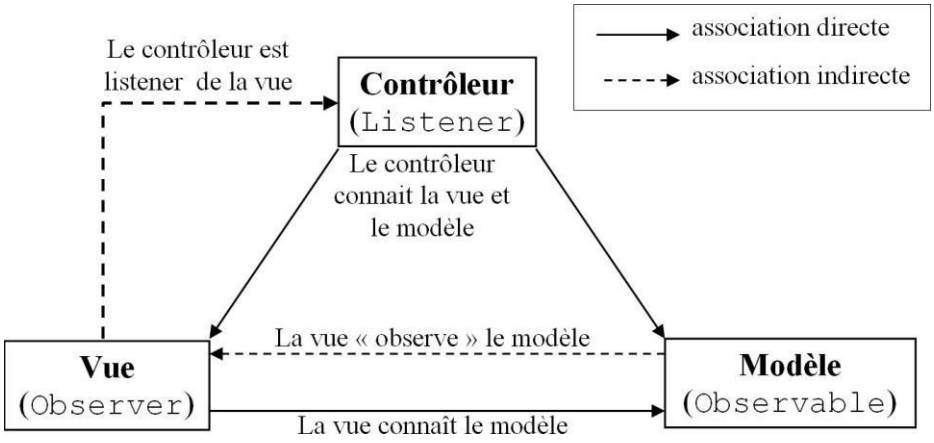


Figure 23: Architecture MVC

- **Le modèle :** Le modèle est une représentation des données de l'application et est généralement constitué d'objets.
- **La Vue :** L'interface utilisateur de l'application est illustrée dans la vue. Les vues sont définies à l'aide de modèles qui décrivent comment les données du modèle doivent être affichées.
- **Le contrôleur :** Les composants remplissent le rôle de contrôleur. La logique de présentation et la gestion des événements visuels sont définies par les composants.

La figure ci-dessous (***figure 24***) représente l'architecture logique de notre application (vue d'ensemble).

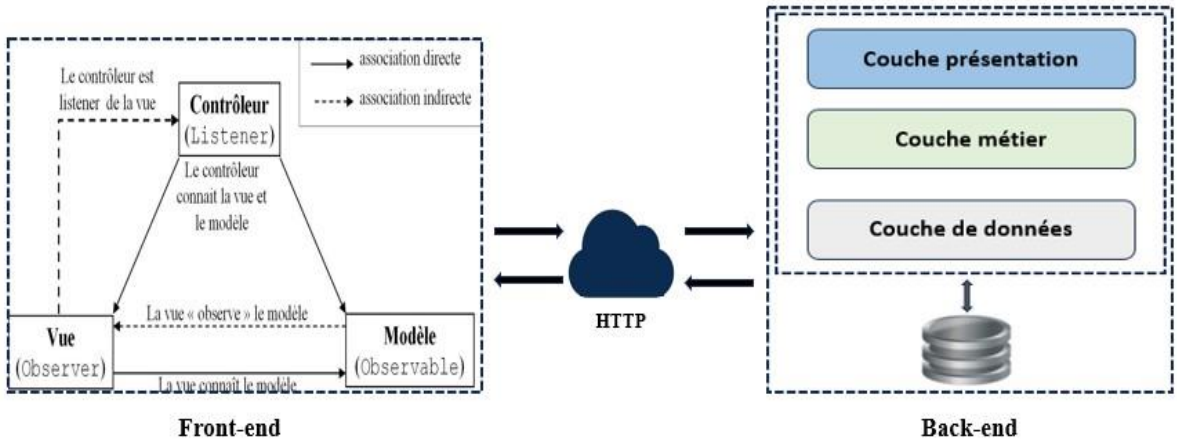


Figure 24: Architecture de notre application

III. Conception détaillée

Dans cette section, nous allons présenter quelques diagrammes de classes (Pointage, évaluation et répartition des étudiants) avant de terminer par le dictionnaire des données qui sera présenté en annexe. Le reste des diagrammes de classes (informations personnelles et informations professionnelles) est consigné aussi en Annexe.

III.1. Définition d'un diagramme de classe

Un diagramme de classes fournit une vue globale d'un système en présentant ses classes, interfaces et collaborations, et les relations entre elles. Les diagrammes de classes sont statiques : ils affichent ce qui interagit mais pas ce qui se passe pendant l'interaction.
En notation UML, une classe est représentée sous la forme d'un rectangle divisé en plusieurs parties : le nom de la classe, les attributs (champs), les opérations (méthodes) at autres[43].

III.2. Diagramme de classe des Pointages

La figure ci-dessous (***figure 25***) présente le diagramme de classes du sous-module Pointage. Nous représentons toutes les classes représentant dans le processus du pointage.

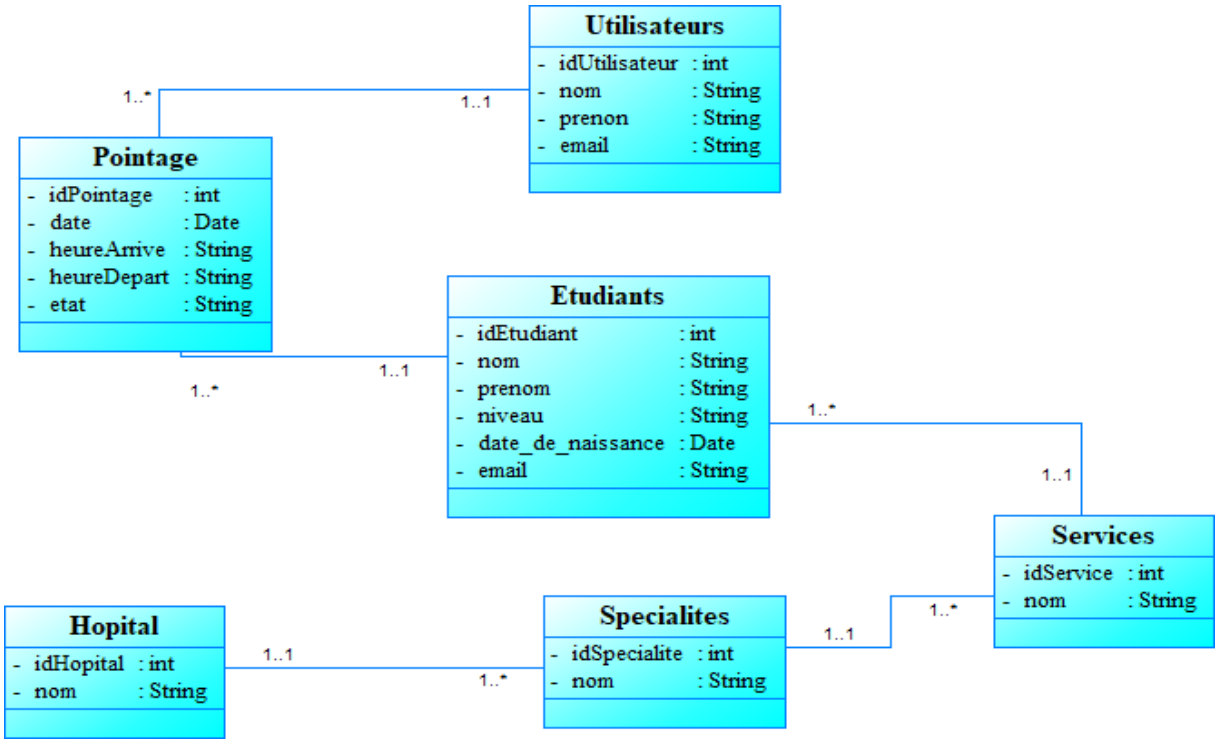


Figure 25: Diagramme de classe des Pointages

Le tableau ci-après nous donne la description détaillée du diagramme de classes ci-dessus.

Classe	**Description de la classe**
Utilisateur	Cette classe regroupe les informations communes des utilisateurs
Pointage	Cette classe permet de regrouper tous les étudiants pointer durant les stages
Etudiants	Cette classe permet de regrouper les informations sur les étudiants qui sont sur la base plus particulièrement ceux qui ont en stage
Hôpital	Cette classe permet de regrouper les informations sur les hôpitaux où se trouves les étudiants
Spécialités	Cette classe permet de regrouper toutes les spécialités de chaque hôpital.
Service	Cette classe permet de regrouper les services des étudiants.

Tableau 17: Description du diagramme de classes des pointages

III.3. Diagramme de classe des Évaluations

La figure ci-dessous (***figure 26***) présente le diagramme de classes du sous-module Evaluation.

Nous représentons toutes les classes représentant dans le processus d'Evaluation.

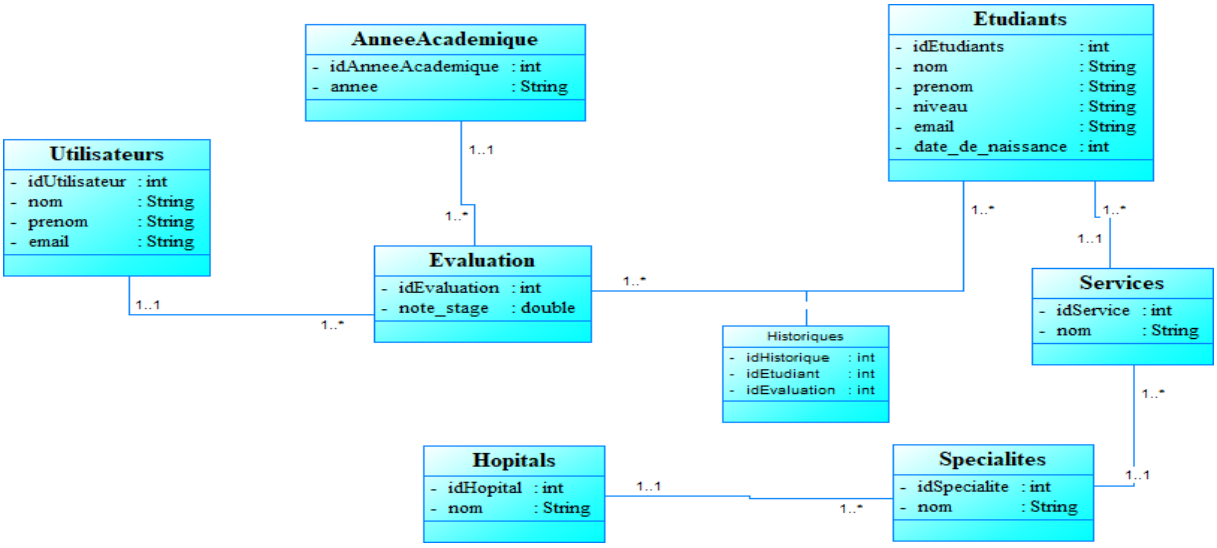


Figure 26: Diagramme de classe des Évaluations

Le tableau ci-après nous donne la description détaillée du diagramme de classes ci-dessus.

Classe	**Description de la classe**
Utilisateur	Cette classe regroupe les informations communes des utilisateurs
Évaluations	Cette classe permet de regrouper toutes les évaluations des étudiants durant les stages
Étudiants	Cette classe permet de regrouper les informations sur les étudiants qui sont sur la base plus particulièrement ceux qui ont en stage et qu'on doit évaluer
Hôpital	Cette classe permet de regrouper les informations sur les hôpitaux où se trouves les étudiants à évaluer
Spécialités	Cette classe permet de regrouper toutes les spécialités de chaque hôpital.
Service	Cette classe permet de regrouper les services des étudiants à évaluer.
Historique	Cette classe permet de regrouper l'association de l'étudiant et ses notes durant le stage
Année Académique	Cette classe permet de regrouper les années académique

Tableau 18: Description du diagramme de classes des Évaluations

Conclusion

Ce chapitre a commencé l'analyse des besoins en montrant les diagrammes d'activités et les diagrammes de séquences. Ensuite, il a proposé des architectures physique et logique dans la conception générale. Enfin, dans la conception détaillée, il est présenté et décrit les différents diagrammes de classes pour les différents modules que nous avons présenté dans les chapitres précédents. Ces phases nous a permis d'avoir un modèle de base de données pour entamer le chapitre implémentation de l'application dans la partie suivante

Chapitre V Implémentation, test et déploiement

Introduction

Dans cette section, nous débuterons par une introduction des outils employés pour le développement de notre application.

Ensuite, nous aborderons les étapes d'implémentation, de test, de déploiement, et enfin, nous examinerons quelques interfaces clés de l'application.

I. Présentation des outils de développement

Puisque nous avons opté pour une architecture client-serveur, avec le client représentant le front-end et le serveur le back-end, nous avons sélectionné deux frameworks : React pour le front-end et Express Node.js pour le back-end. En ce qui concerne la base de données, notre choix s'est porté sur MySQL.

Dans cette partie, nous présenterons en détail les frameworks React et Express Node.js, ainsi que l'API REST utilisée pour mettre en place les APIs du back-end. Nous expliquerons également notre choix pour l'Environnement de Développement Intégré (IDE), à savoir Visual Studio Code, ainsi que pour le système de gestion de base de données (SGBD) MySQL.

I.1. Présentation de React

ReactJS figure parmi les bibliothèques JavaScript les plus prisées pour le développement d'applications web et mobiles. Conçu par Facebook en 2013, React propose une collection d'éléments de code JavaScript réutilisables appelés composants, qui sont utilisés pour construire l'interface utilisateur (UI).

Il est essentiel de noter que ReactJS ne constitue pas un framework JavaScript à proprement parler. En effet, il se concentre uniquement sur le rendu des composants de l'interface utilisateur (UI). De ce fait, de nombreux développeurs JavaScript associent React à d'autres frameworks tels qu'Angular et Vue pour développer des fonctionnalités plus complexes[44].

I.1.1. Fonctionnalité de react

React possède un ensemble de fonctionnalités fondamentales qui le distinguent des autres bibliothèques JavaScript. Les sections suivantes détailleront ces fonctionnalités et expliqueront leur contribution au développement d'applications web et mobiles[45] :

JSX : JSX est une extension de syntaxe JavaScript utilisée pour créer des éléments React. Les développeurs l'utilisent pour intégrer du code HTML dans des objets JavaScript, simplifiant ainsi les structures de code complexes grâce à la prise en charge des expressions JavaScript valides et l'intégration de fonctions[46].

DOM Virtuel : Le Document Object Model (DOM) représente une page web dans une structure arborescente de données. ReactJS stocke les arbres Virtual DOM en mémoire, ce qui lui permet d'appliquer des mises à jour spécifiques à des parties de l'arbre de données, plus rapidement que le rendu complet du DOM. Le processus de comparaison entre les nouveaux arbres DOM virtuels et les précédents, pour implémenter rapidement les changements dans le DOM réel, est appelé "diffing"[47].

Composants et Props : ReactJS divise l'interface utilisateur en composants réutilisables isolés, fonctionnant de manière similaire aux fonctions JavaScript et acceptant des entrées appelées propriétés ou props[48].

Gestion de l'état : se réfère à la pratique de la gestion des états des applications React. Cela inclut le stockage des données dans des bibliothèques tierces de gestion de l'état et le déclenchement du re-rendu chaque fois que les données sont modifiées. Redux et Recoil sont deux des bibliothèques de gestion de l'état les plus populaires[49].

Redux : dispose d'un magasin centralisé pour maintenir l'arborescence d'état d'une application, ce qui réduit les incohérences des données et simplifie la maintenance grâce à une architecture stricte[50].

Recoil : une bibliothèque de gestion d'état publiée par Facebook, utilise des fonctions pures appelées sélecteurs pour calculer des données à partir d'atomes modifiables. Recoil est plus adapté aux débutants que Redux en raison de ses concepts plus simples[51].

Navigation par programmation : React Router, la bibliothèque standard de React pour le routage, permet une navigation sécurisée entre les composants sans nécessiter de cliquer sur des liens. L'utilisation de composants comme Redirect et history.push() permet de contrôler l'apparence des applications React sans dépendre des liens[52].

En résumé, ces fonctionnalités de React offrent aux développeurs une grande flexibilité et facilitent le développement d'applications web et mobiles réactives et dynamiques.

I.1.2. Justification du choix de react comme framework front-end

Les frameworks front-end sont désormais aussi cruciaux que leurs homologues côté serveur dans le développement d'applications. Les développeurs sont confrontés au défi de sélectionner le bon framework pour assurer des performances optimales et une évolutivité

efficace. Avec une multitude d'options disponibles, telles que Vue[53], Ink[54], Angular[55], React, etc., faire le choix approprié peut s'avérer être un véritable défi.
Nous avons opté pour React comme framework front-end en raison de ses fonctionnalités riches ainsi que d'autres avantages qu'il offre.

- **Facilité d'utilisation :** React repose sur du JavaScript simple et une approche basée sur les composants, ce qui permet aux développeurs ayant des connaissances en JavaScript de l'apprendre rapidement. En quelques jours seulement, il est possible de commencer à développer des applications web avec React.
- **Support des composants réutilisables :** React permet de réutiliser facilement des composants développés dans d'autres applications, grâce à sa nature open source. Cela réduit le temps de développement des applications web complexes. De plus, la capacité d'imbriquer des composants les uns dans les autres simplifie la création de fonctionnalités complexes sans alourdir le code, tout en facilitant leur maintenance grâce à la modularité des composants.
- **Facilité d'écriture des composants :** L'intégration de JSX rend l'écriture des composants React plus facile, permettant aux utilisateurs de combiner des objets JavaScript avec du HTML. Cette approche simplifie également le rendu de multiples fonctions tout en maintenant un code léger et performant.
- **Hautes performances :** Grâce au Virtual DOM, React met à jour l'arbre DOM de manière efficace, évitant ainsi le re-rendu excessif qui peut affecter les performances. De plus, la liaison de données unidirectionnelle de React simplifie le processus de débogage en réduisant les risques d'erreurs lors de la modification des composants enfants sans affecter la structure parente.
- **Optimisé pour le référencement :** React améliore l'optimisation des moteurs de recherche (SEO) des applications web en augmentant leurs performances. L'implémentation du DOM virtuel contribue à des vitesses de page plus rapides, tandis que le rendu côté serveur aide les moteurs de recherche à naviguer dans les applications web, résolvant ainsi les problèmes liés à l'exploration des sites lourds en JavaScript.

Pour résumer, la sélection de React dépend des besoins spécifiques de notre projet, de la complexité de l'application et des préférences du maître de stage. React propose une solution globale pour le développement d'applications modernes, surtout pour les projets nécessitant une architecture robuste et une maintenance à long terme.

I.2. Présentation du Framework Express

Express JS est un Framework web léger, open-source et basé sur Node.js. Il fournit une couche d'abstraction simple pour gérer les requêtes HTTP et les routes d'une manière élégante. Grâce à son approche minimaliste, il n'impose pas de structure rigide, ce qui permet aux développeurs d'avoir une grande flexibilité dans la création d'applications web[56].

I.2.1. Comment fonctionne express

Étant basé sur le modèle client-serveur, Express.js fonctionne de manière similaire à d'autres frameworks populaires comme Laravel. Lorsqu'un utilisateur envoie une requête depuis son navigateur web en tapant l'adresse d'un site, le navigateur envoie une requête HTTP à l'application ou au serveur, qui est souvent hébergé dans le cloud dans le cas des applications créées avec Express.js[57].

Le serveur reçoit alors cette requête via l'une de ses routes et la traite en utilisant le contrôleur correspondant à la route demandée. Une fois le traitement terminé, le serveur renvoie une réponse au client en utilisant le protocole HTTP, qui assure la communication aller-retour.

Cette réponse peut prendre différentes formes : elle peut être un simple texte, une page HTML dynamique que le navigateur affichera, ou des données JSON qui seront manipulées par les développeurs frontend pour afficher des informations sur la page web.

I.2.2. Justification du choix d'express

Les frameworks sont désormais essentiels au développement d'applications, nécessitant des choix judicieux pour assurer des performances optimales et une évolutivité efficiente. Avec une multitude d'options disponibles telles que Django, Laravel, Ruby on Rails, etc., la sélection appropriée peut représenter un défi de taille pour les développeurs.

Nous avons porté notre choix sur Express.js pour les raisons suivantes :

- **Facilité d'utilisation** : Express offre une syntaxe simple et intuitive pour créer des routes et gérer les requêtes, ce qui rend le processus de développement plus rapide.
- **Rapidité** : Étant donné qu'Express est un framework minimaliste, il est très performant et adapté aux applications nécessitant des temps de réponse rapides.
- **Robustesse** : Même s'il est léger, Express fournit un ensemble de fonctionnalités solides qui facilitent le développement d'applications web complexes.

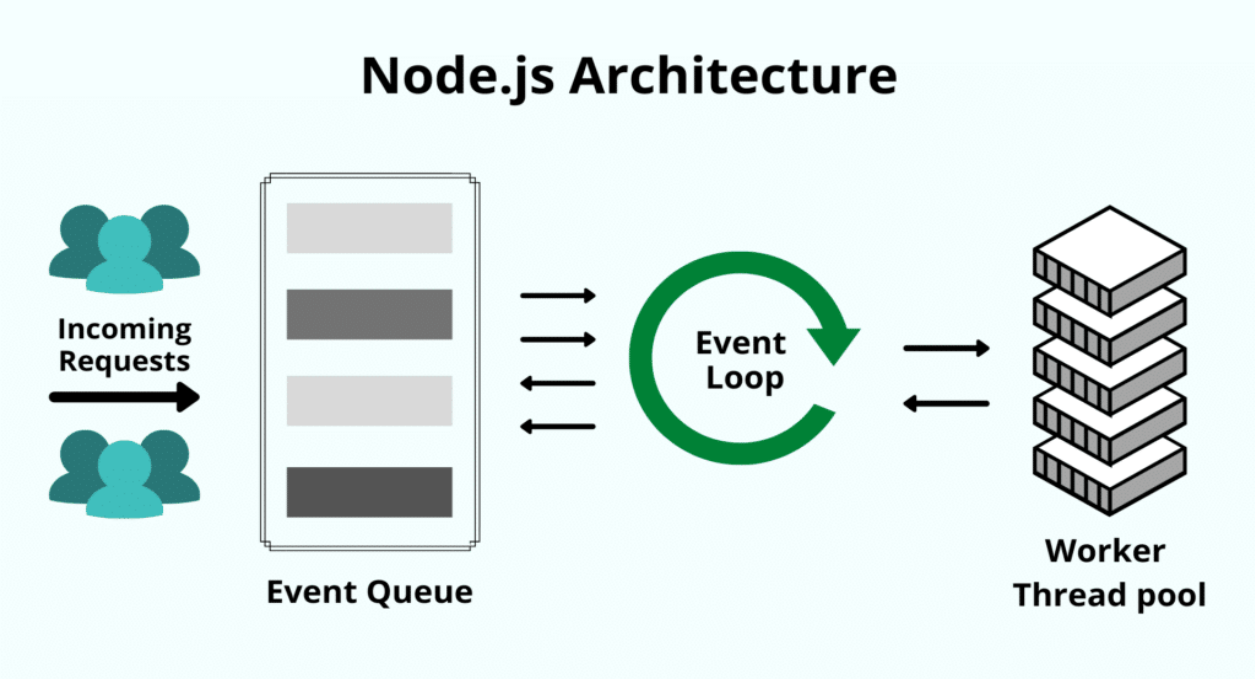
- **Grande communauté** : Express JS bénéficie d'une communauté active, ce qui signifie que vous trouverez un grand nombre de modules complémentaires (middleware) et de ressources pour vous aider à résoudre vos problèmes.

I.3. Présentation de Node.js

Pendant ses deux premières décennies, JavaScript était principalement utilisé pour les scripts côté client, limité à la balise <script>. Cette restriction obligeait les développeurs à jongler entre plusieurs langages et frameworks pour gérer à la fois le frontend et le backend de leurs applications. Cependant, l'avènement de Node.js a changé la donne. Node.js est un environnement d'exécution open-source, mono-thread et multi-plateforme conçu pour exécuter des programmes JavaScript. Il fonctionne avec le moteur d'exécution JavaScript V8 et adopte une architecture d'E/S non bloquante et pilotée par les événements, ce qui le rend particulièrement adapté aux applications en temps réel et hautement évolutives côté serveur et réseau[58].

I.3.1. Architecture de node.js et comment il fonctionne

Node.js adopte l'architecture "Single Threaded Event Loop" pour gérer plusieurs clients simultanément. Pour comprendre cette différence par rapport à d'autres runtimes, il est utile de comparer avec la gestion des clients concurrents multi-threads dans des langages comme Java. Dans un modèle requête-réponse multi-thread, plusieurs clients envoient des requêtes au serveur, et celui-ci traite chacune d'elles avant de renvoyer la réponse. Cependant, cette approche utilise plusieurs threads pour gérer les appels simultanés. Ces threads sont généralement organisés dans un pool de threads, et chaque fois qu'une requête est reçue, un thread individuel est attribué à son traitement[58].



Node.js fonctionne selon un processus distinct. Voyons chaque étape en détail :

- Node.js maintient un pool limité de threads pour répondre aux requêtes.
- Lorsqu'une requête est reçue, Node.js l'ajoute à une file d'attente.
- Ensuite, la "boucle d'événement" single-thread, qui est le composant central, entre en jeu. Cette boucle attend indéfiniment les requêtes.
- Lorsqu'une requête est récupérée de la file d'attente, la boucle d'événement vérifie si elle nécessite une opération d'entrée/sortie (E/S) bloquante. Si ce n'est pas le cas, elle traite la requête et envoie une réponse.
- Si la requête implique une opération bloquante, la boucle d'événement assigne un thread du pool de threads internes, appelé le groupe de worker, pour traiter la requête. Le nombre de threads disponibles est limité.
- Une fois la tâche bloquante traitée, la boucle d'événement suit les requêtes bloquantes et les place dans la file d'attente. Cela maintient la nature non bloquante de Node.js.
- En utilisant moins de threads, Node.js consomme moins de ressources et de mémoire, ce qui permet une exécution plus rapide des tâches. Ainsi, pour les applications en temps réel, Node.js est souvent privilégié malgré son architecture single-thread.

I.3.2. Justification du choix de node.js

La technologie Node.js s'impose comme une solution back-end alternative aux langage PHP, Java et Python pour vos serveur web. La sélection appropriée peut représenter un défi de taille pour les développeurs.

Nous avons porté notre choix sur le Node.js pour les raisons suivantes :

- **Node.js est un excellent choix pour les API Rest** : API **:** Les Application Programming Interfaces. Les API Rest sont utilisés pour les échanges de données entre serveurs et clients. L'un des avantages de la technologie Node.js est la possibilité d'exécuter plusieurs requêtes vers le serveur simultanément, car c'est un système « single thread » non bloquant. donc Node.js est capable de gérer une multitude de requêtes sans attendre que la requête lancée précédemment soit terminée.
- **Excellentes performances** : Avec l'avantage de traiter une multitude d'opérations dans sa boucle d'événements, le **Node.js** offre des performances incroyables. De plus, Node.js est léger et la technologie est basée sur le très puissant moteur V8 de Google Chrome**,** ce qui offre des performances incroyables aux applications web qui doivent chercher des informations et exécuter une multitude de requêtes.
- **Beaucoup de flexibilité** : Un grand écosystème de librairies open-source offre la possibilité aux développeurs d'ajouter les modules nécessaires pour une application web **Node.js**. Un développeur qui maîtrise le **JavaScript** peut concevoir le backend et le frontend d'une application web. La très grande communauté **JavaScript** (le langage de programmation le plus utilisé sur le web) travaille constamment sur la création de nouveaux modules. Des entreprises telles que **Netflix, PayPal et IBM** utilisent la technologie **Node.js.**
- **Un bon stack technique** : En utilisant du **JavaScript** côté serveur en backend et côté interface frontend avec les différents frameworks frontend tels que le **React.js, Next.js, Vues.Js ou Angular.js**, votre application web utilise du JavaScript partout.
- **Réduire le temps de développement** : Le temps de développement peut être considérablement réduit avec le **Node.js**. Le code JavaScript peut être partagé sur le frontend et le backend par un seul développeur grâce au Node.js.

I.4. API REST (Representational State Transfer)

I.4.1. Présentation de l'API REST

Une API REST (Representational State Transfer, en français Transfert d'État Représentatif), également connue sous le nom d'API RESTful, est une interface de programmation d'application (API) conçue conformément aux principes architecturaux de REST. Elle facilite l'interaction avec les services web basés sur l'architecture REST. Cette architecture a été conceptualisée par l'informaticien Roy Fielding[59].

I.4.2. Propriétés

L'API REST est basée sur 7 propriétés principales :

- Performance : Interaction simple entre les composants ;
- Évolutivité : Supporte une large variété de composants ;
- Simplicité : Entre les interfaces ;
- Modification : Peut être modifié sans impacter les clients ;
- Visibilité : Communication claire entre les composantes ;
- Confiance : Reprise sur panne.

I.4.3. Caractéristiques

L'architecture RESTful a plusieurs caractéristiques :

- Les requêtes sont client-serveur (séparation client-serveur) ;
- Les requêtes sont stateless (sans état) ;
- Une interface uniforme est utilisée par les clients et les serveurs. Une interface générique permet l'accès à toutes les ressources. : les méthodes HTTP : GET, POST, PUT, DELETE, HEAD ou OPTION ;
- Les clients accèdent à des ressources nommées. Le système comprend des ressources nommées en utilisant des URL, comme des URL HTTP (mais ne se limitant pas à des URL HTTP). Pour faire simple, une ressource est représentée par une URL ;
- Possibilité de mise en cache ;
- Etc

I.5. Le système de gestion de base de données

Un système de gestion de base de données (SGBD) est un logiciel conçu pour permettre à un ordinateur de stocker, récupérer, ajouter, supprimer et modifier des données. Il assure la gestion de tous les aspects fondamentaux d'une base de données, incluant la manipulation des données, tels que l'authentification des utilisateurs et l'insertion ou l'extraction des données. Le SGBD est responsable de la définition du schéma de données ou de la structure dans laquelle les données sont organisées. Il gère trois composants essentiels : les données elles-mêmes, le moteur de base de données qui permet d'accéder, de verrouiller et de modifier les données, ainsi que le schéma de base de données, qui décrit la structure logique de la base de données. Ces trois éléments contribuent de manière cruciale à garantir la cohérence, la sécurité, l'intégrité des données et l'uniformité des procédures administratives. Il existe différentes SGBD : relationnelles (Oracle, MySQL, PostgreSQL, etc.) et non relationnelles (MongoDB, ArangoDB, etc.)[60].

Parmi ces bases de données, nous avons choisi le SGBD relationnelles MySQL que nous allons présenter avant de justifier son choix.

I.5.1. Présentation et Utilisation du SGBD MySQL

MySQL est un système de gestion de bases de données relationnelles (SGBDR) qui agit comme un serveur fournissant un accès multi-utilisateur à plusieurs bases de données.

MySQL est largement reconnu comme la base de données open source la plus répandue dans le monde. Il est utilisé dans divers contextes, tels que les applications web, le traitement et le stockage de données à grande échelle, et toute autre situation nécessitant une haute disponibilité, fiabilité et performance. Développé par MySQL AB, MySQL a été initialement publié en 1995 sous la licence publique générale GNU. Il est écrit en C et C++ et est compatible avec plusieurs systèmes d'exploitation, notamment Windows, Linux/Unix, Mac OS X, Solaris (SPARC), et bien d'autres[61].

I.5.2. Justification du choix de MySQL comme SGBD

MySQL est l'une des SGBD les plus utilisées aujourd'hui, il présente beaucoup d'avantages qui peuvent justifier notre choix. Voici quelques-uns de ses avantages :

- MySQL est un logiciel gratuit et open-source, accessible à tous pour une utilisation et une modification sans frais de licence, adapté aussi bien aux projets personnels que commerciaux.

- Il offre une interface utilisateur intuitive et des commandes SQL simples, facilitant sa manipulation même pour les débutants.
- Conçu pour gérer des bases de données de grande taille, MySQL assure des performances exceptionnelles, capable de traiter rapidement des millions d'enregistrements et de supporter des charges de travail élevées avec de nombreuses connexions simultanées.
- Sa grande communauté de développeurs et d'utilisateurs contribue activement à son évolution et sa maintenance, offrant ainsi une abondance de ressources pour résoudre les problèmes et améliorer les fonctionnalités.

En résumé, MySQL est un SGBD relationnelle puissante, extensible et open source qui, en raison de sa flexibilité et de ses fonctionnalités avancées, est adaptée à une variété de cas d'utilisation.

La figure ci-dessous (***figure 27***) est une illustration de l'architecture logique globale de notre application

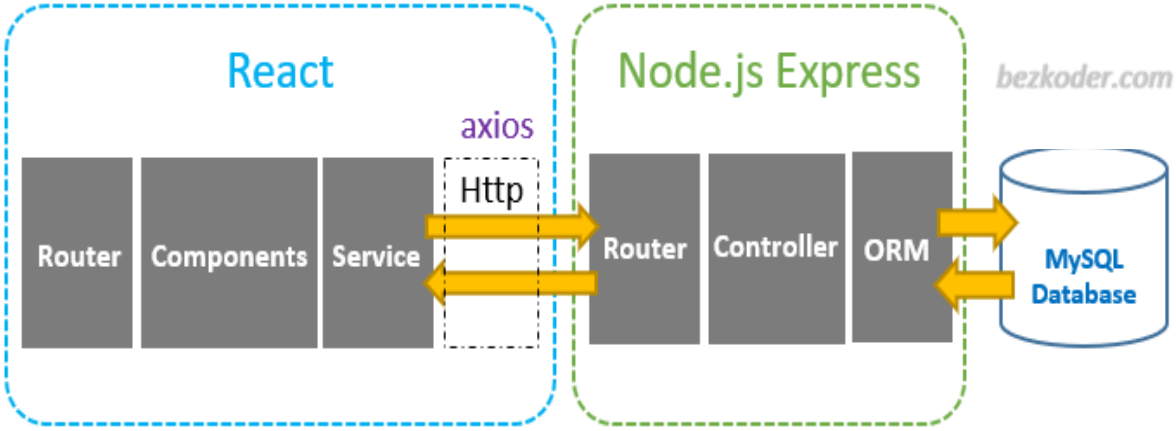


Figure 27: Architecture logique générale de l'application

I.6.Environnement de développement intégré (IDE)

Un environnement de développement intégré (IDE) est un logiciel de création d'applications qui regroupe divers outils de développement au sein d'une interface utilisateur graphique (GUI) unique. Les composants typiques d'un IDE incluent[62] :

- **Éditeur de code source :** Un outil de rédaction de code qui facilite l'écriture du logiciel en offrant des fonctionnalités telles que la coloration syntaxique, l'auto-complétions basée sur le langage et la détection de bugs pendant la saisie.

- **Utilitaires d'automatisation locale :** Des outils permettant d'automatiser des tâches courantes lors de la création d'une version locale du logiciel, telles que la compilation du code source en code binaire, la mise en paquet du code binaire et l'exécution de tests automatisés.
- **Débogueur :** Un programme qui facilite le test et le débogage des autres programmes en identifiant les bugs dans le code source original.

Parmi les IDE, nous pouvons citer Visual Studio, IntelliJ IDEA, PyCharm, Eclipse, etc.
Nous avons opté pour **Visual Studio Code** pour des raisons que nous allons présenter après sa définition.

I.6.1. Présentation de Visual Studio Code

Visual Studio Code est un éditeur de code simplifié, qui est gratuit et développé en open source par Microsoft. Il fonctionne sous Windows, mac OS et Linux. Il fournit aux développeurs à la fois un environnement de développement intégré avec des outils permettant de faire avancer les projets techniques, de l'édition, à la construction, jusqu'au débogage[63].

I.6.2. Justification du choix de l'IDE Visual Studio Code

Les fonctionnalités proposées par Visual Studio Code sont nombreuses ce qui nous a permis de le choisir comme IDE. On retrouve notamment :

- **La prise en charge de plusieurs de langages de programmations**, telles que C, C#, C++, CSS, HTML, Java, JavaScript, JSON, Markdown, PHP, Powershell, Python, TypeScript, YAML…,
- **IntelliSense**, une fonction de complétion intelligente du code,
- **Un débogueur intégré** pour accélérer votre boucle d'édition, de compilation et de suppression des bugs,
- **Une interface d'édition**, qui intègre des raccourcis clavier, des sélections multiples, un enregistrement automatique de votre travail, une fonction rechercher/remplacer, le formatage du code source…,
- **Peek,** une fonction qui permet de parcourir rapidement le code source et de naviguer entre les fichiers,
- **Les commandes Git intégrées** ainsi que la gestion du contrôle des sources (SCM).

Visual Studio Code permet également aux développeurs de créer et d'utiliser des extensions grâce à son API, afin de personnaliser leur utilisation de l'outil. Il est livré avec un support pour JavaScript, TypeScript et Node.js.

Plusieurs versions de Visual Studio Code sont disponibles :

- **Pour Windows** (versions 7, 8, 10 et 11) : 64 bits, 32 bits et ARM
- **Pour Linux** (Debian, Ubuntu, Red Hat, Fedora, SUSE) : 64 bits, ARM et ARM 64,
- **Pour Mac** (version 10.11 de macOS et ultérieures) : .zip (Universal, Intel Chip et Apple Silicon).

II. Implémentations de l'applications

II.1. Implémentation du Back-end

Durant l'implémentation de notre back-end, nous avons régulièrement tenu des réunions avec certains membres de l'urf 2S plus particulièrement le responsable pédagogique de l'UFR pour la mise en place d'un cahier de charges. Après la mise en place du cahier de charge, nous avons commencé la conception et l'implémentation du back-end. Nous organisons des réunions d'équipe une fois par semaine pour la validation d'un sprint et le début d'un autre sprint.

En effet, comme le backend était unique pour le PGS, nous avons dû utiliser un outil de travail collaboratif pour que chaque membre de l'équipe qui travaillait sur son module puisse apporter des modifications le concernant sans faire obstruction aux autres. Pour ce travail collaboratif, nous avons choisi de travailler avec Git pour des raisons que nous allons expliquer après sa présentation avant de passer à la description de la manière dont nous avons travaillé avec Git.

II.1.1. Définition de Git

Git est un système de contrôle de version qui a été inventé et développé par **Linus Torvalds**, également connu pour l'invention du noyau Linux, en 2005.

Il s'agit d'un outil de développement qui aide une équipe de développeurs à **gérer les changements apportés au code source** au fil du temps.

Les logiciels de contrôle de version gardent une trace de chaque changement apporté au code dans un type spécial de base de données.

Git est le plus connu des **VCS (versionning control system),** c'est un projet open source très puissant qui est utilisé par l'ensemble de la communauté des développeurs[64].

II.1.2. Choix de Git comme outil de travail collaboratif

Les logiciels de gestion de projet Git simplifient la collaboration entre les développeurs, que ce soit dans le domaine du développement informatique ou du Big Data. Il permet aux différentes personnes travaillant sur le même projet de partager des informations cohérentes. Ainsi, chaque participant peut consulter les modifications apportées aux différentes versions du projet, ainsi que les tâches terminées et celles restant à accomplir. Voici quelques autres avantages considérables qui ont motivé notre choix :

- Enregistrement de l'historique des modifications d'un ensemble de fichiers.
- Possibilité de revenir à des versions antérieures d'un ou plusieurs fichiers.
- Identification des modifications susceptibles d'avoir causé des erreurs.
- Partage et récupération des modifications effectuées par d'autres collaborateurs.
- Proposition, discussion et examen de modifications sans altérer la dernière version existante.
- Identification des auteurs et des dates des modifications.

II.1.3. Méthode de travail avec Git

Dans cette section, nous n'allons pas entrer dans les détails techniques, mais nous allons simplement nous concentrer sur la structure de notre processus de développement. Nous avons commencé par créer un dépôt local sur le serveur de l'Université, puis nous avons créé une branche dédiée à chaque membre de l'équipe.

Ensuite, nous avons procédé à l'implémentation des classes en commun, chacun travaillant de son côté selon son sprint. Une fois les tâches terminées, chaque membre a effectué un commit dans sa propre branche, soumis son travail pour validation, puis poussé ses modifications dans la branche respective avant de passer à la prochaine étape de développement. Après avoir achevé l'implémentation des classes en commun, chacun a poursuivi son travail individuel.

Enfin, le Scrum Master a fusionné le contenu des différentes branches dans la branche principale avant de procéder au déploiement. Pour une exploration plus approfondie de Git, vous pouvez vous référer à d'autres ressources[65].

II.2. Implémentation du Front-end

Durant l'implémentation de notre front-end, nous avons presque procédé de la même manière à celle du back-end (avec un nouveau dépôt pour le front-end).

Parce que comme c'est le même front-end avec ses sprints et à la fin de ces sprints, nous avons fusionne les branches du fait que tous les modules du PGS ont le même front-end.
En ce qui concerne le design, nous avons téléchargé un template adéquat au Framework React de notre choix et nous l'avons adapté à nos besoins.

III. Test

III.1. Test du back-end

Pour évaluer les services de notre back-end, nous avons utilisé Insomnia, un outil polyvalent et puissant qui simplifie la gestion des API, la documentation, le développement et les tests. Insomnia est devenu indispensable dans le domaine du développement logiciel moderne pour les développeurs, les testeurs et les équipes travaillant sur des projets liés aux API.
Grâce à Insomnia, il est possible d'effectuer des requêtes HTTP via une interface graphique. Les méthodes HTTP les plus couramment utilisées sont les suivantes, chacune ayant sa propre fonction :

- GET : Ces requêtes sont similaires à celles effectuées par un navigateur lorsque vous entrez une URL dans la barre de navigation. Elles visent à récupérer une page ou des données.
- POST : Ces requêtes sont utilisées pour envoyer des informations vers le serveur, contenues dans le corps de la requête.
- PUT : Ces requêtes remplacent une ressource par de nouvelles données, également fournies dans le corps de la requête. Elles sont utilisées pour mettre à jour des données, à condition de fournir la ressource mise à jour dans son intégralité.
- PATCH : Ces requêtes sont similaires aux requêtes PUT, mais elles modifient uniquement l'élément envoyé dans le corps de la requête, permettant une mise à jour partielle de la ressource.
- DELETE : Comme son nom l'indique, cette requête est utilisée pour supprimer une ressource.

III.2. Test du front-end

Pour cette étape, nous avons créé un compte administrateur pour tester les fonctionnalités de notre front-end.

V. Présentation de quelques interfaces de l'application

V.1. La page d'authentification

Cette page demande deux éléments clés : l'identifiant et le mot de passe. Seuls les utilisateurs disposant d'un compte ont accès. Cependant, les comptes des chefs de département, des responsables de Pédagogique, des secrétaires et des maitres de stages sont gérés par l'administrateur. (Voir figure 28).

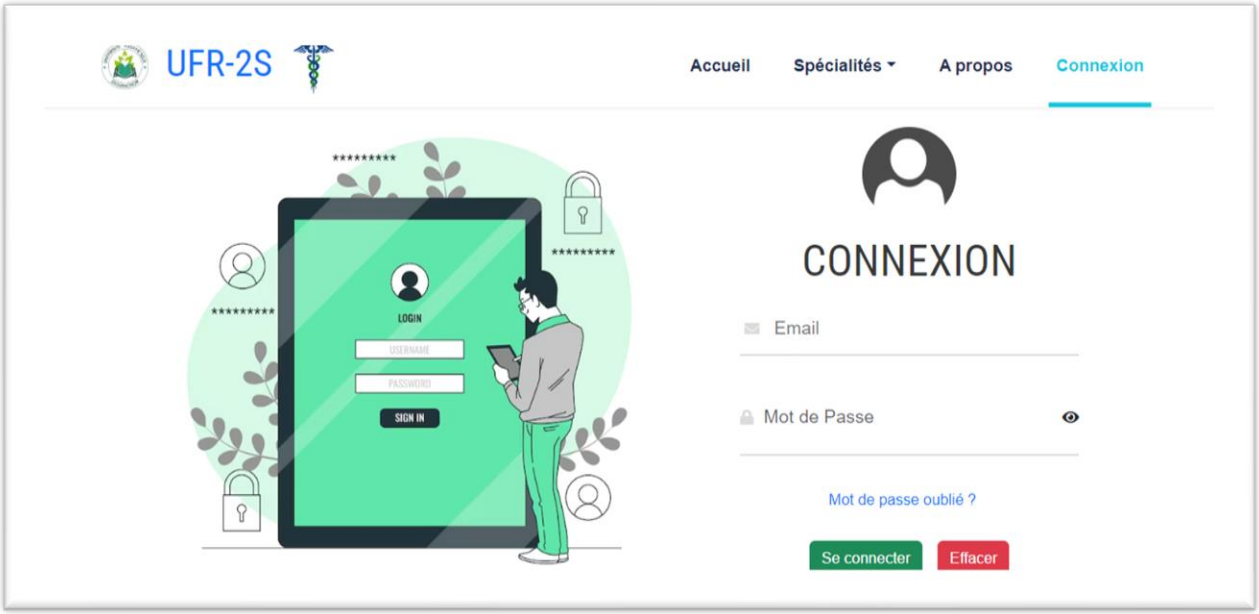


Figure 28: page de connexion

V.2. Espace responsable Pédagogique

V.2.1. La page du formulaire d'importation des étudiants

C'est la page qui contient le formulaire qui permet au responsable pédagogique d'importer la liste des étudiants qui font les stages dans la base (Voir figure 29).

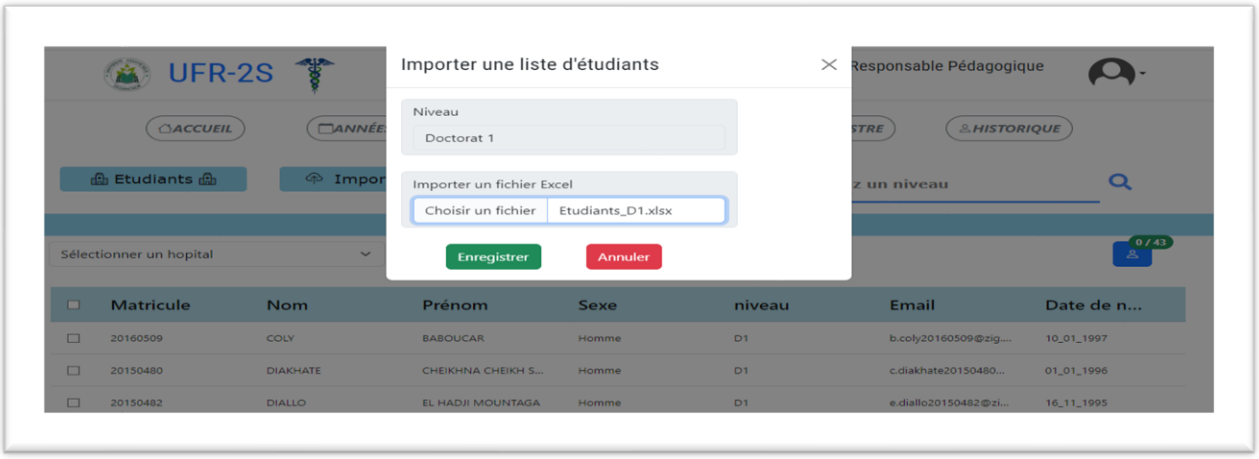


Figure 29:Page de la liste étudiant charger dans la base avec formulaire d'importation

V.2.2. La page des étudiants non-inscrits

Elle énumère tous les étudiants non-orienté et importer par le responsable pédagogique (voir figure 30).

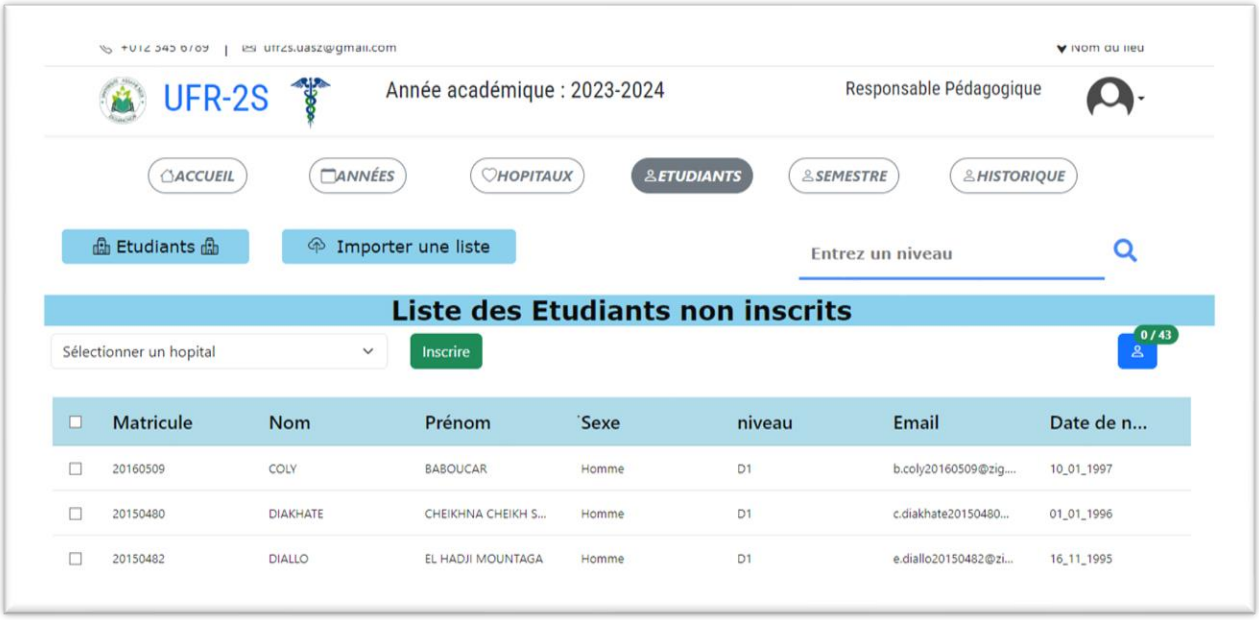


Figure 30: Liste des étudiants non inscrit

V.3. Espace chef de département

V.3.1. La page de la liste des étudiants dans une spécialité

Elle énumère tous les étudiants orientés dans un hôpital et dans une spécialité (voir figure 31).

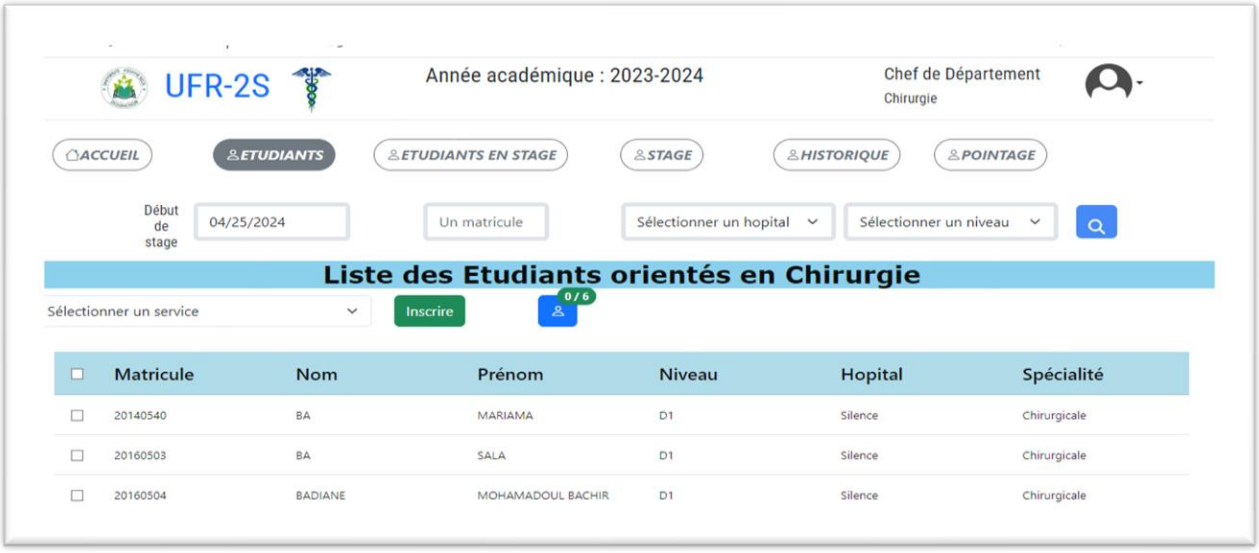


Figure 31: listes des étudiants d'une spécialité

V.3.2. La page de la liste des étudiants en stage

Elle énumère tous les étudiants orientés dans un hôpital et dans une spécialité et dans un service (voir figure 32).

Figure 32: Liste des étudiants en stage

V.4. Espace Secrétaire

V.4.1. La page d'accueil du Secrétaire

C'est une page qui regroupe l'ensemble des actions qui sont sous la directive du secrétaire (Voir figure 33).

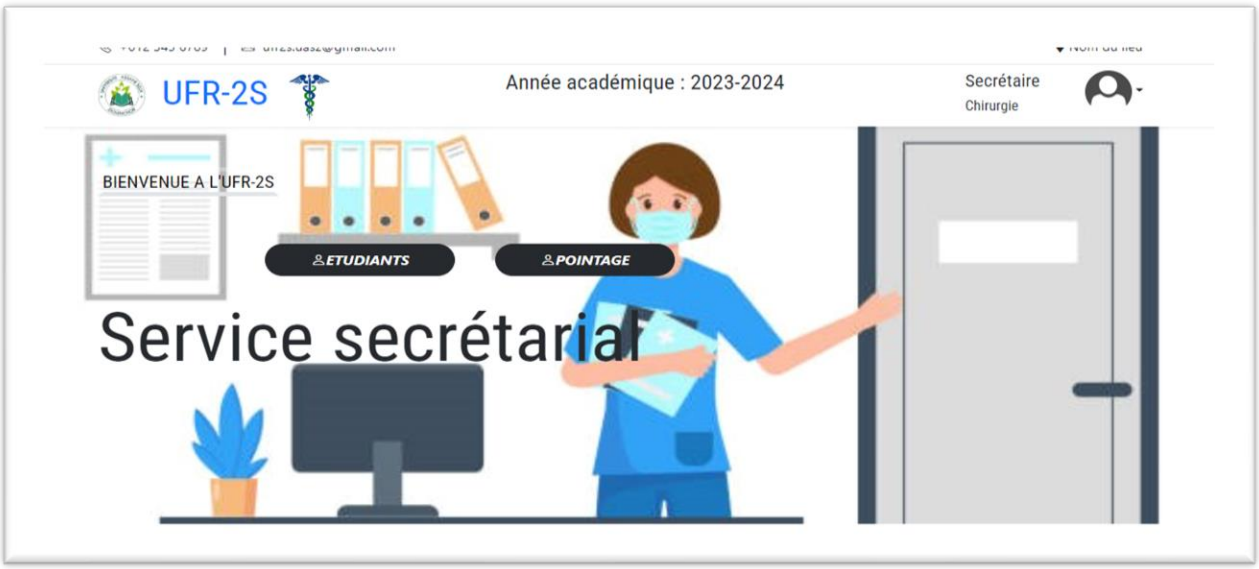


Figure 33:Page d'accueil du secrétaire

V.4.2. La page de la liste des étudiants pointes

Elle énumère tous les étudiants orientés dans un hôpital et dans une spécialité et dans un service et pointer par ce le secrétaire responsable de l'hôpital (voir figure 34).

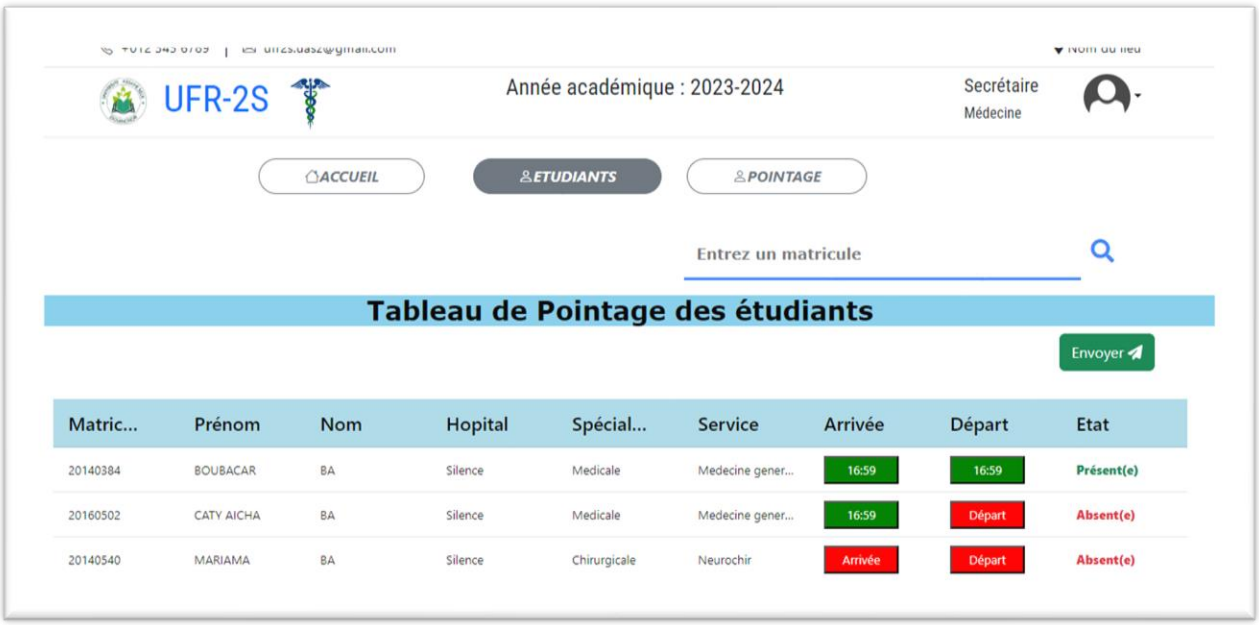


Figure 34:liste des étudiants à pointer

V.5. Espace Maitre de Stage

V.5.1. La page d'accueil du Maitre de Stage

C'est une page qui regroupe l'ensemble des actions qui sont sous la directive du Maitre de stage (Voir figure 35).

Figure 35:Page d'accueil du maitre de stage

V.5.2. La page des étudiants notes

Elle énumère tous les étudiants orientés dans un hôpital et dans une spécialité et dans un service et noter par ce le maitre de stage responsable du service (voir figure 36).

Figure 36:Liste des étudiants à noter

Conclusion

Ce chapitre a montré de manière détaillée les étapes qui nous ont permis de réaliser notre application. Nous avons commencé par lister les technologies utilisées avant de montrer l'installation et la configuration du Framework CodeIgniter. Nous avons ensuite présenté le processus de l'implémentation de la base de données puis de l'application. Enfin, nous avons présenté quelques interfaces de l'application. Le dernier chapitre de ce document est une conclusion générale résumant l'ensemble du travail accompli et des perspectives envisagées.

Conclusion générale

Dans le cadre de notre cursus de Master en Génie Logiciel, nous avons eu l'opportunité de réaliser un stage au sein de la Direction du Service Pédagogique de l'UFR 2S de l'Université Assane Seck de Ziguinchor (UASZ), dans le cadre du Projet de Gestion des Stages des Étudiants (PGSE). Notre mission principale était de concevoir une plateforme web permettant une gestion efficace pour l'exploitation de l'environnement de gestion des stages hospitaliers de l'UFR 2S qui se base sur un module : le module de la gestion des étudiants, englobant tous les aspects liés aux étudiants : orientation dans les hôpitaux, spécialités et services, suivi de la présence, notes obtenues, rotations entre spécialités et services, et génération des résultats pour le compte de la Direction du Service Pédagogique (SP) de l'UFR 2S.

La gestion des stages des étudiants est essentiellement un ensemble exhaustif de processus nécessaires pour assurer la bonne marche des stages. Malheureusement, sous sa forme de gestion sur papier, il présente plusieurs inconvénients et limites qui entravent sa fonctionnalité et rendent difficile l'exploitation des données à des fins de recherche. Pour pallier ces défauts, nous avons proposé d'informatiser la gestion des stages des étudiants en le structurant de manière plus efficace et en utilisant les outils modernes des technologies de l'information.

Notre travail a débuté par une présentation des structures impliquées (UASZ, L'UFR et SP), suivie d'une mise en lumière des problèmes associés au gestion des stages des étudiants sur papier. Nous avons ensuite opté pour la méthodologie Agile pour notre processus de conception, en raison de sa compatibilité avec notre méthode de travail, qui nécessitait une forte interaction avec le client et une collaboration étroite au sein de l'équipe.

Ensuite, nous avons procédé à la modélisation et à la conception de l'application en identifiant les besoins, les acteurs du système, et en divisant l'application en sous-modules avec leurs fonctionnalités respectives, grâce à des diagrammes de cas d'utilisation. Nous avons présenté les architectures choisies pour la réalisation de l'application, ainsi que des diagrammes de séquences, d'activités et de classes pour détailler notre conception.

L'implémentation de l'application a été réalisée en utilisant Express Node.js pour le back-end et React pour le front-end. Nous avons également présenté quelques interfaces de l'application. Pendant ce processus, nous avons mis à profit nos connaissances théoriques et pratiques acquises lors de notre formation, en utilisant des outils tels que l'UML, les frameworks Express et React, ainsi que des outils de développement comme VS Code et PowerAMC.

Ce stage nous a permis d'acquérir une expérience professionnelle significative et de renforcer notre capacité à travailler en équipe. Nous sommes fiers d'avoir atteint nos objectifs en créant une gestion des stages informatisé qui facilite les tâches du personnel du service pédagogique, et ceux de l'hôpital, notamment en ce qui concerne la répartition des étudiants, les rotations des étudiants entre les spécialités et services, les évaluations, les pointages.

Pour l'avenir, plusieurs perspectives s'ouvrent à nous, notamment l'établissement de statistiques pour faciliter la gestion des bilans des stages, la liaison de la base de l'application avec la base de l'université, de veiller plus sur la sécurité, etc.

Annexes

Annexe : dictionnaire de données

Utilisateur		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
Nom	Varchar (255)	Non
Prenom	Varchar (255)	Non
Sexe	Varchar (255)	Non
Email	Varchar (255)	Non
Spécialité	Varchar (255)	Non
Etudiant		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
Matricule	Varchar (255)	Non
Nom	Varchar (255)	Non
Prenom	Varchar (255)	Non
Sexe	Varchar (255)	Non
niveau	Varchar (255)	Non
Email	Varchar (255)	Non
dateDeNaissance	Varchar (255)	Non
hopitalSiteId	Bigint (20)	Non
SpecialiteId	Bigint (20)	Non
ServiceId	Bigint (20)	Non
Hôpital		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
nonHôpital	Varchar (255)	Non
etat	Varchar (255)	Non
Spécialité		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint	Non
nomSpecialite	Varchar (255)	Non
Affilier		
Colonne	**Type**	**Null**
Id (Primaire)	Bigin t(20)	Non
hopitalSiteId	Bigint (20)	Non
utilisateurId	Bigint (20)	Non
Anneeacademique		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
annee	Varchar (255)	Non
utilisateurId	Bigint (20)	Non
Services		
Colonne	**Type**	**Null**

Id (Primaire)	Bigint (20)	Non
NomService	Varchar (255)	Non
specialiteId	Bigint (20)	Non
Comptes		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
login	Varchar (255)	Non
Mot_de_passe	Varchar (255)	Non
etat	Varchar (255)	Non
profil	Varchar (255)	Non
photo	Varchar (255)	Non
utilisateurId	Bigint (20)	Non
Evaluation		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
note	double	Non
utilisateurId	Bigint (20)	Non
anneeAcademiqueId	Bigint (20)	Non
serviceId	Bigint (20)	Non
moduleId	Bigint (20)	Non
Historiques		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
etudiantId	Bigint (20)	Oui
evaluationId	Bigint (20)	Oui
date	Varchar (255)	Oui
type	Varchar (255)	Oui
hopital	Varchar (255)	Oui
specialite	Varchar (255)	Oui
service	Varchar (255)	Oui
semestre	Varchar (255)	Oui
duree	Varchar (255)	Oui
annee	Varchar (255)	Oui
Modules		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
utilisateurId	Bigint (20)	Oui
date	Varchar (255)	Oui
type	Varchar (255)	Oui
hopital	Varchar (255)	Oui
specialite	Varchar (255)	Oui
service	Varchar (255)	Oui
semestre	Varchar (255)	Oui
duree	Varchar (255)	Oui
Pointage		
Colonne	**Type**	**Null**
Id (Prfimaire)	Bigint (20)	Non
heureArrive	Varchar (255)	Non

heureDepart	Varchar (255)	Non
etat	Varchar (255)	Non
utilisateurId	Bigint (20)	Non
etudiantId	Bigint (20)	Non
datePointage	date	Non
Stage		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
semestre	Varchar (255)	Non
dateDebut	Varchar (255)	Non
dateFin	Varchar (255)	Non
annee	Varchar (255)	Non
utilisateurId	Bigint (20)	Non
Logs		
Colonne	**Type**	**Null**
Id (Primaire)	Bigint (20)	Non
dateDeLog	date	Oui
heureDeConn	Varchar (255)	Oui
heureDeDeconn	Varchar (255)	Oui
utilisateurId	Bigint (20)	Oui

Bibliographie

[1] T. Collonvillé, « Elaboration de processus de développements logiciels spécifiques et orientés modèles: application aux systèmes à évenements discrets ».

[2] « Le modèle en cascade (waterfall model) », IONOS Digital Guide. Consulté le: 23 février 2024. [En ligne]. Disponible sur: https://www.ionos.fr/digitalguide/sites-internet/developpement-web/modele-en-cascade/

[3] « Les 5 étapes du processus de planification stratégique ». Consulté le: 15 avril 2024. [En ligne]. Disponible sur: https://www.lucidchart.com/blog/fr/les-5-etapes-du-processus-de-planification-strategique

[4] « La phase d'analyse fonctionnelle – Société canadienne d'analyse de la valeur ». Consulté le: 15 avril 2024. [En ligne]. Disponible sur: https://www.valueanalysis.ca/functionanalysis.php?lang=fr

[5] F. Bouchaouir, Y. Dentinger, et O. Englender, « Chapitre 3. Phase de conception », in *Gestion de projet*, vol. 4è éditio, in Lire Agir, vol. 4è éditio. , Paris: Vuibert, 2017, p. 141-164. Consulté le: 15 avril 2024. [En ligne]. Disponible sur: https://www.cairn.info/gestion-de-projet--9782311622133-p-141.htm

[6] « Phase de mise en oeuvre d'un projet ». Consulté le: 15 avril 2024. [En ligne]. Disponible sur: https://web.maths.unsw.edu.au/~lafaye/CCM/projet/phase-mise-en-oeuvre.htm

[7] M. H. Chahine, « [ISTQB] le processus de test », La taverne du testeur. Consulté le: 15 avril 2024. [En ligne]. Disponible sur: https://latavernedutesteur.fr/2023/10/16/istqb-le-processus-de-test/

[8] « Phase de déploiement | AppMaster ». Consulté le: 15 avril 2024. [En ligne]. Disponible sur: https://appmaster.io/fr/glossary/phase-de-deploiement

[9] « Phase de maintenance ». Consulté le: 15 avril 2024. [En ligne]. Disponible sur: https://help.hcltechsw.com/commerce/9.1.0/fr/admin/concepts/cpm_launchphase.html

[10] Atlassian, « Présentation d'Agile », Atlassian. Consulté le: 23 février 2024. [En ligne]. Disponible sur: https://www.atlassian.com/fr/agile

[11] *Test Driven Development*. Consulté le: 26 février 2024. [En ligne]. Disponible sur: https://books.google.com/books/about/Test_Driven_Development.html?hl=fr&id=zNnPEAAAQBAJ

[12] « Qu'est-ce que l'agile UP (AUP) ? - My Agile Partner Scrum ». Consulté le: 26 février 2024. [En ligne]. Disponible sur: https://blog.myagilepartner.fr/index.php/2017/09/16/quest-ce-que-lagile/

[13] « Présentation des méthodes agiles et Scrum. » Consulté le: 23 février 2024. [En ligne]. Disponible sur: https://ineumann.developpez.com/tutoriels/alm/agile_scrum/

[14] « 03_d-xp.pdf ». Consulté le: 26 février 2024. [En ligne]. Disponible sur: http://mfworld42.free.fr/cnam/NFE103_METHODOLOGIES%20AVANCEES%20D'INFORMATISATION/16_R%E9visions%20et%20divers%20polys/03_d-xp.pdf

[15] jc-QualityStreet, « Lean Software Development: 7 principes fondateurs », Blog Agile depuis 2007. Consulté le: 23 février 2024. [En ligne]. Disponible sur: https://www.qualitystreet.fr/2008/08/27/lean-software-development-7-principes-fondateurs/

[16] « Méthode Agile : avantages et limites du mode agile ». Consulté le: 23 février 2024. [En ligne]. Disponible sur: https://blog-gestion-de-projet.com/avantages-et-inconvenients-de-la-methode-agile/

[17] « DevOps c'est quoi ? Définition DevOps et concepts », France. Consulté le: 23 février 2024. [En ligne]. Disponible sur: https://www.qrpinternational.fr/blog/gestion-des-services-informatiques/devops-cest-quoi-definition-devops/

[18] « Approche DevOps : définition, concepts et principes - Glossaire Syloé », Syloe. Consulté le: 15 avril 2024. [En ligne]. Disponible sur: https://www.syloe.com/glossaire/approche-devops/
[19] « Jenkins », Jenkins. Consulté le: 1 mars 2024. [En ligne]. Disponible sur: https://www.jenkins.io/
[20] « Docker Desktop: The #1 Containerization Tool for Developers | Docker ». Consulté le: 1 mars 2024. [En ligne]. Disponible sur: https://www.docker.com/products/docker-desktop/
[21] « Qu'est-ce-que Kubernetes ? » Consulté le: 1 mars 2024. [En ligne]. Disponible sur: https://kubernetes.io/fr/docs/concepts/overview/what-is-kubernetes/
[22] « Git ». Consulté le: 1 mars 2024. [En ligne]. Disponible sur: https://git-scm.com/
[23] O. F. & Consulting, « Quels sont les avantages de l'approche DevOps ? », Oo2 Formations & Consulting. Consulté le: 1 mars 2024. [En ligne]. Disponible sur: https://www.oo2.fr/actualites/quels-sont-les-avantages-de-l-approche-devops
[24] « SyBooks Online (Archive) ». Consulté le: 1 mars 2024. [En ligne]. Disponible sur: https://infocenter-archive.sybase.com/help/index.jsp?topic=/com.sybase.stf.poweramc.docs_12.1.0/html/dcgu/dcgup168.htm
[25] « La modélisation ». Consulté le: 26 février 2024. [En ligne]. Disponible sur: http://www.jnlog.com/model1_fr.htm
[26] « modelisation_orientee_objet.pdf ». Consulté le: 26 février 2024. [En ligne]. Disponible sur: https://infocenter-archive.sybase.com/help/topic/com.sybase.infocenter.poweramc.15.1/doc/pdf/modelisation_orientee_objet.pdf
[27] « Qu'est-ce que le langage UML (langage de modélisation unifié) ? », Lucidchart. Consulté le: 27 février 2024. [En ligne]. Disponible sur: https://www.lucidchart.com/pages/fr/langage-uml
[28] « Les types de diagrammes UML | Blog Lucidchart ». Consulté le: 27 février 2024. [En ligne]. Disponible sur: https://www.lucidchart.com/blog/fr/types-de-diagrammes-UML
[29] « SyBooks Online ». Consulté le: 16 avril 2024. [En ligne]. Disponible sur: https://infocenter.sybase.com/help/index.jsp?topic=/com.sybase.infocenter.dc31018.1610/doc/html/rad1232632577723.html
[30] « Diagrammes Comportementaux — UML SysML ». Consulté le: 16 avril 2024. [En ligne]. Disponible sur: https://www.uml-sysml.org/diagrammes-uml-et-sysml/diagramme-uml/diagrammes-comportementaux/
[31] « IBM Documentation ». Consulté le: 15 avril 2024. [En ligne]. Disponible sur: https://www.ibm.com/docs/fr/was-zos/9.0.5?topic=servers-testing-production-phases
[32] « Qu'est-ce qu'un diagramme d'activité UML ? », Lucidchart. Consulté le: 1 mars 2024. [En ligne]. Disponible sur: https://www.lucidchart.com/pages/fr/diagramme-dactivite-uml
[33] « Qu'est-ce qu'un diagramme de séquence UML ? », Lucidchart. Consulté le: 5 mars 2024. [En ligne]. Disponible sur: https://www.lucidchart.com/pages/fr/diagramme-de-sequence-uml
[34] « conception générale - Définitions, synonymes, prononciation, exemples | Dico en ligne Le Robert ». Consulté le: 6 mars 2024. [En ligne]. Disponible sur: https://dictionnaire.lerobert.com/definition/conception-generale
[35] « Architecture physique », *Wikipédia*. 13 décembre 2022. Consulté le: 6 mars 2024. [En ligne]. Disponible sur: https://fr.wikipedia.org/w/index.php?title=Architecture_physique&oldid=199456801
[36] « Memoire Online - Application web pour la gestion de la bibliothèque - Emna Guermazi », Memoire Online. Consulté le: 7 mars 2024. [En ligne]. Disponible sur: https://www.memoireonline.com/04/11/4453/m_Application-web-pour-la-gestion-de-la-bibliotheque7.html
[37] SeB, « L'architecture 3-tiers à l'heure du serverless », Le weblogue de SeB. Consulté le: 7 mars 2024. [En ligne]. Disponible sur: https://blog.lecacheur.com/2017/01/26/larchitecture-3-tiers-a-lheure-du-serverless/

[38] jamesmontemagno, « Architecture logique et architecture physique - .NET ». Consulté le: 17 avril 2024. [En ligne]. Disponible sur: https://learn.microsoft.com/fr-fr/dotnet/architecture/microservices/architect-microservice-container-applications/logical-versus-physical-architecture
[39] « Que signifie architecture monolithique? - Definition IT de LeMagIT », LeMagIT. Consulté le: 7 mars 2024. [En ligne]. Disponible sur: https://www.lemagit.fr/definition/architecture-monolithique
[40] « Apprenez l'architecture orientée services », OpenClassrooms. Consulté le: 7 mars 2024. [En ligne]. Disponible sur: https://openclassrooms.com/fr/courses/7210131-definissez-votre-architecture-logicielle-grace-aux-standards-reconnus/7371141-apprenez-larchitecture-orientee-services
[41] « Les architectures microservices - LAB 5COM | Conseil en développement web & mobile ». Consulté le: 7 mars 2024. [En ligne]. Disponible sur: https://lab5com.fr/
[42] « L'Architecture En Couches, Toujours La Norme? », InfoQ. Consulté le: 7 mars 2024. [En ligne]. Disponible sur: https://www.infoq.com/fr/articles/architecture-couches/
[43] « Définition des diagrammes de classes UML 1.5 — RAD Studio ». Consulté le: 8 mars 2024. [En ligne]. Disponible sur: http://docwiki.embarcadero.com/RADStudio/Alexandria/fr/D%C3%A9finition_des_diagrammes_de_classes_UML_1.5#D.C3.A9finition
[44] « Qu'est-ce que React.js ? Un regard sur la bibliothèque JavaScript populaire », Kinsta®. Consulté le: 17 avril 2024. [En ligne]. Disponible sur: https://kinsta.com/fr/base-de-connaissances/qu-est-react-js/
[45] L. com Mbiya Jean Claude, « Qu'est-ce que React et comment fonctionne-t-il réellement ? », Letecode. Consulté le: 11 mars 2024. [En ligne]. Disponible sur: https://letecode.com/quest-ce-que-react-et-comment-fonctionne-t-il-reellement
[46] « Introduction à JSX – React ». Consulté le: 17 avril 2024. [En ligne]. Disponible sur: https://fr.legacy.reactjs.org/docs/introducing-jsx.html
[47] « DOM virtuel et autres détails – React ». Consulté le: 17 avril 2024. [En ligne]. Disponible sur: https://fr.legacy.reactjs.org/docs/faq-internals.html
[48] « Composants et props – React ». Consulté le: 17 avril 2024. [En ligne]. Disponible sur: https://fr.legacy.reactjs.org/docs/components-and-props.html
[49] « Qu'est-ce que la gestion de l'État ? : r/Frontend ». Consulté le: 17 avril 2024. [En ligne]. Disponible sur: https://www.reddit.com/r/Frontend/comments/17kyo0v/questce_que_la_gestion_de_l%C3%A9tat/fr/?rdt=35388
[50] « Quel est l'intérêt de Redux dans un projet React ? » Consulté le: 17 avril 2024. [En ligne]. Disponible sur: https://oclock.io/quel-est-linteret-de-redux-dans-un-projet-react
[51] « Commencer | Recoil ». Consulté le: 17 avril 2024. [En ligne]. Disponible sur: https://recoiljs.org/fr/docs/introduction/getting-started/
[52] L. Renzetti, « Gérer facilement la navigation avec React Native - Blog ». Consulté le: 17 avril 2024. [En ligne]. Disponible sur: https://www.kaliop.com/fr/gerer-facilement-la-navigation-avec-react-native/
[53] « Apprendre VueJS », Grafikart.fr. Consulté le: 18 avril 2024. [En ligne]. Disponible sur: https://grafikart.fr/tutoriels/vuejs
[54] « Ink ». Consulté le: 18 avril 2024. [En ligne]. Disponible sur: https://www.ink-formation.com/
[55] « Angular ». Consulté le: 18 avril 2024. [En ligne]. Disponible sur: https://angular.io/
[56] « (4) Le Framework Web Express JS pour Node.js | LinkedIn ». Consulté le: 12 mars 2024. [En ligne]. Disponible sur: https://www.linkedin.com/pulse/le-framework-web-express-js-pour-nodejs-eric-venturino/?originalSubdomain=fr
[57] « Qu'est-ce qu'Express.js ? Tout ce que vous devez savoir », Kinsta®. Consulté le: 11 mars 2024. [En ligne]. Disponible sur: https://kinsta.com/fr/base-de-connaissances/qu-est-express-js/

[58] « Qu'est-ce que Node.js et pourquoi l'utiliser ? », Kinsta®. Consulté le: 11 mars 2024. [En ligne]. Disponible sur: https://kinsta.com/fr/base-de-connaissances/qu-est-ce-que-node-js/
[59] « API REST : définition ». Consulté le: 12 mars 2024. [En ligne]. Disponible sur: https://www.redhat.com/fr/topics/api/what-is-a-rest-api
[60] « Qu'est-ce qu'un système de gestion de base de données », Oracle France. Consulté le: 12 mars 2024. [En ligne]. Disponible sur: https://www.oracle.com/fr/database/systeme-gestion-base-de-donnees-sgbd-definition/
[61] « Qu'est ce que le MySQL ? IA School », IA School. Consulté le: 12 mars 2024. [En ligne]. Disponible sur: https://www.intelligence-artificielle-school.com/ecole/technologies/quest-ce-que-mysql-tout-savoir-sur-cette-base-de-donnees/
[62] « Qu'est-ce qu'un environnement de développement intégré (IDE) ? » Consulté le: 12 mars 2024. [En ligne]. Disponible sur: https://www.redhat.com/fr/topics/middleware/what-is-ide
[63] « Visual Studio Code : l'éditeur de code gratuit et complet de Microsoft », BDM | Tools. Consulté le: 12 mars 2024. [En ligne]. Disponible sur: https://www.blogdumoderateur.com/tools/visual-studio-code/
[64] « Qu'est-ce que Git ? » Consulté le: 12 mars 2024. [En ligne]. Disponible sur: https://www.next-decision.fr/wiki/qu-est-ce-que-git
[65] sphinx, « Ma méthode de travail avec Git et GitHub », Bidouilleux d'Web. Consulté le: 12 mars 2024. [En ligne]. Disponible sur: https://tech.mozfr.org/post/2016/04/16/Ma-methode-de-ravail-avec-Git-et-GitHub