



ESPACE UNIVERSITAIRE REGIONALE DE L'OCEAN INDIEN

(Excellence – Professionnalisme – Dépassement de soi)

MEMOIRE DE FIN D'ETUDES POUR L'OBTENTION DU
DIPLOME D'INGENIORAT

« PLATEFORME DE COMMUNICATION, DE
GESTION ET DE PARTAGE DE DOCUMENTS
DANS UN ETABLISSEMENT »

Présenté par : RAKOTOARINTSIFA Loïc Judicaël Harrison.

Encadreur : Mr RANDRIAMIHAJARISON M. Jimmy

Filière : Electronique – Télécommunication – Informatique.

Niveau : Master II.

Année : 2023.

REMERCIEMENTS

Ce travail est le fruit de la combinaison d'efforts de plusieurs personnes. Je remercie tout d'abord le Tout puissant qui, par sa grâce m'a permis d'arriver au bout de mes efforts en me donnant la santé, la force, le courage et en me faisant entourer des merveilleuses personnes que je tiens à remercier. Je remercie :

- **Madame Henriette Marie RASOANOMENJANAHARY**, Directeur Général de l'Espace Universitaire Régional de l'Océan Indien, de m'avoir permis de poursuivre mes études supérieures dans son honorable établissement.
- **Monsieur RANDRIAMIHAJARISON M. Jimmy**, ce travail ne serait pas aussi riche et n'aurait pas pu voir le jour sans l'aide et la qualité de son encadrement exceptionnel, pour sa patience, sa rigueur et sa disponibilité durant la préparation de ce mémoire.
- Je remercie également **tout le comité éducatif et les professeurs de l'EUROI**, pour leur générosité et la qualité de l'enseignement qu'ils m'ont prodigué au cours de ces cinq années passées dans cet établissement.

Un grand merci qui s'adresse à toute ma famille et particulièrement à mes chers parents qui m'ont soutenu moralement, physiquement, spirituellement et surtout financièrement durant ma formation entière jusqu'à la réalisation de ce mémoire de fin d'études.

De même, un grand merci à toutes les personnes qui ont contribué, de loin ou de près, à l'achèvement de ce travail ; pour leurs conseils et leurs directives. Les discussions et les critiques ont été largement appréciés.

AVANT – PROPOS

L'Espace Universitaire Régional de l'Océan Indien connu sous le nom de l'EUROI, est une institution qui forme des jeunes bacheliers dans les domaines suivants : Télécommunication, Informatique et Electronique. Elle a pour devise « Excellence – Professionnalisme – Dépassement de soi ».

Tout étudiant au niveau M2 de formation au sein de l'établissement doit obligatoirement présenter un mémoire de fin d'études réalisé à la fin de la formation, qui lui permettra d'acquérir un diplôme d'ingéniorat.

La réalisation qui suit est une plateforme d'application web pour faciliter la communication des utilisateurs dans un établissement scolaire. Elle utilise le langage de programmation JavaScript ainsi que ses Framework et librairie.

Ainsi, ce travail représentera les acquis théoriques et pratiques que nous avons reçus durant ces années d'études et de formation dans cet établissement.

SOMMAIRE

REMERCIEMENTS

AVANT-PROPOS

SOMMAIRE

LISTE DES ACRONYMES

LISTE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION

PARTIE I : CONCEPTION THEORIQUE

CHAPITRE I : PRESENTATION DU THEME

CHAPITRE II : ETUDE THEORIQUE DU THEME

CHAPITRE III : UNE APPLICATION WEB

PARTIE II : CONCEPTION PRATIQUE

CHAPITRE IV : ETUDE PRATIQUE DU PROJET

PARTIE III : ESSAIS ET RESULTATS

CHAPITRES V : ESSAIS ET RESULTATS

- **RESULTATS DES SIMULATIONS**
- **BILAN ET RECOMMANDATION**

CONCLUSION

REFERENCES BIBLIOGRAPHIQUE ET WEBOGRAPHIQUE

TABLE DES MATIERES

LISTE DES ACRONYMES

AJAX : Asynchronous Javascript And XML.
API : Application Programming Interface.
CMD : Command.
CMS : Content Management System.
CSS : Cascading Style Sheet.
DNS : Domain Name System.
DOM : Document Object Model.
FTP : File Transfer Protocol.
HTML : HyperText Markup Language.
HTTP : HyperText Transfer Protocol.
HTTPS : HyperText Transfer Protocol Secure.
IDE : Integrated Development Environment.
IoT : Internet of Things.
IP : Internet Protocol.
JSON : JavaScript Object Notation.
JSX : Javascript Syntaxe eXtension.
MERN : MongoDB Express React Node.
MSI : Medium Scale Integration.
MYSQL : My Structured Query Language.
NoSQL : Not only Structured Query Language.
NPM : Node Package Manager.
ORM : Object Relational Mapping.
PHP : PHP hypertext Processor.
REST : REpresentational State Transfer.
SAAS : Sotfware As A Service.
SEO : Search Engine Optimization.
SGBD : Système de Gestion de Base de Données.
SMTP : Simple Mail Transfer Protocol.
SOAP : Simple Object Access Protocol.
UI : User Interface.
UML : Unified Modeling Language.

URL : Uniform Resource Locator.

UX : User eXperience.

VSCODE : Visual Studio Code.

WWW : World Wide Web.

XML : eXtensible Markup Language.

LISTES DES FIGURES

Figure 1 : Architecture de la plateforme	6
Figure 2 : Fonctionnement du HTTP (source : HOSTINGER)	18
Figure 3 : Exemple de code HTML.....	23
Figure 4 : Squelette d'une page web	25
Figure 5 : Diagramme de cas d'utilisation.....	31
Figure 6 : Diagramme de classe.....	32
Figure 7 : Diagramme de cas d'utilisation.....	33
Figure 8 : Icone de Visual Studio Code	38
Figure 9 : IDE Visual Studio Code.....	39
Figure 10 : Commande d'affichage de version de Node.js.....	44
Figure 11 : Commande pour créer une application React	44
Figure 12 : Commande pour lancer l'application React.....	45
Figure 13 : Première page de démarrage d'une nouvelle application React.....	45
Figure 14 : Télécharger l'installateur de Node.js	48
Figure 15 : Commande pour afficher la version de Node.js.....	49
Figure 16 : Commande pour afficher la version de npm.....	49
Figure 17 : Commande pour créer une application Node.js	54
Figure 18 : Commande pour installer Express.js	54
Figure 19 : Commande pour utiliser Express.js dans une application Node.js.....	54
Figure 20 : Commande pour instancier Express.js	54
Figure 21 : Commande pour voir la version de mongodb installé.	59
Figure 22 : Commande pour créer un nouveau projet Strapi.	63
Figure 23 : Commande pour installer les dépendances nécessaires de Strapi.	64
Figure 24 : Commande pour démarrer le serveur de Strapi.	64
Figure 25 : Page d'authentification de la plateforme.	67
Figure 26 : Page de tableau de bord de la plateforme.	68
Figure 27 : Contenu du tableau de bord de la plateforme pour un administrateur.	70
Figure 28 : Contenu du tableau de bord de la plateforme pour un enseignant ou étudiant.	70
Figure 29 : Espace utilisateurs de la plateforme.	71
Figure 30 : Dialogue montrant les informations d'un utilisateur.	72
Figure 31 : Page de création d'un administrateur de la plateforme.	73
Figure 32 : Page de profil de la plateforme.	74
Figure 33 : Paramètres dans la plateforme.	74

Figure 34 : Mes documents dans la plateforme.	75
Figure 35 : Statistique de la plateforme.	76
Figure 36 : Page à propos de l'établissement dans la plateforme.	76
Figure 37 : Messagerie dans la plateforme.	77
Figure 38 : Notifications dans la plateforme.	77
Figure 39 : Communiqué dans la plateforme.	78

INTRODUCTION

Dans le monde de l'éducation, où l'organisation et la gestion des données jouent un rôle crucial, il est essentiel d'adopter des outils modernes et efficaces pour assurer le bon fonctionnement d'un établissement scolaire. C'est dans cette optique qu'une nouvelle application de gestion de données voit le jour, offrant une solution complète et adaptée aux besoins spécifiques des établissements scolaires.

Alors que les piles de papier et les classeurs encombrants étaient autrefois la norme, nous assistons à une transition vers un monde où l'information est stockée, partagée et gérée de manière électronique.

L'utilisation des documents physiques peuvent prendre beaucoup de temps et entraîner des erreurs ou des omissions. De plus, la collecte et le traitement des données peuvent être lents et laborieux. Cela peut poser des problèmes de gestion des données. La recherche d'informations spécifiques peut être difficile, en particulier lorsque de nombreux formulaires sont remplis. Le stockage et la sauvegarde des fiches d'inscription papier peuvent également être complexes, et il peut être difficile de garantir la confidentialité et la sécurité des données. L'utilisation de document physique implique des coûts liés à l'impression, la distribution et le stockage.

En outre, l'utilisation de papier contribue à la déforestation et à la consommation d'énergie nécessaire à la production et au transport du papier. Cela peut également entraîner une production accrue de déchets, car les fiches d'inscription peuvent être jetées une fois les données transférées dans un système informatique.

C'est vers cette direction que notre projet se focalise sur un thème qui s'intitule : « Plateforme de communication, de gestion et de partage de documents dans un établissement scolaire »

Ce présent mémoire se divise en trois grandes parties. Dans la première partie nous allons dévoiler les études théoriques du thème, suivi de la deuxième partie où nous allons illustrer la conception pratique. Et en dernière partie, les résultats et les recommandations seront mise en valeur suivi d'une conclusion générale.

Partie 1 : Conception théorique

CHAPITRE I : PRESENTATION DU THEME

Ce mémoire de fin d'études est le fruit de tout ce qu'on nous a enseigné pendant ces cinq années d'études au sein de l'EUROI. Suivi d'une réalisation qui a pour but d'exprimer la maîtrise des enseignements acquis ainsi que les diverses expériences professionnelles obtenues au sein du monde professionnel.

1.1.Choix de l'objet de la recherche

De nos jours, l'homme utilise beaucoup les réseaux sociaux, notamment Facebook. Cette dernière a permis à l'homme de communiquer plus facilement, de partager des informations, etc. Or l'utilisation de Facebook n'est pas professionnelle dans certain domaine surtout dans le milieu professionnel. D'où la conception de cette plateforme de communication, de gestion et de partage de documents au sein d'un établissement scolaire.

1.2.Objectif

L'objectif principal du projet est de faire communiquer les utilisateurs entre eux, de partager des informations ou documents dans une seule plateforme accessible à tous ses utilisateurs. En outre, on souligne aussi les différents objectifs spécifiques suivants :

- La centralisation des données : Cela permet d'avoir une vue d'ensemble des données, d'éviter la dispersion et la redondance des informations, et de faciliter la recherche et l'accès aux données.
- Amélioration de la collaboration : Ce projet vise à favoriser la collaboration et l'échange d'informations au sein de l'organisation. Elle offre des fonctionnalités de partage de fichiers, de discussion en temps réel et de travail collaboratif, permettant aux membres de l'équipe de travailler efficacement ensemble, quel que soit leur emplacement géographique.
- Optimisation de la communication interne : Elle offre des outils tels que des messageries instantanées, des fichiers partagés, des notifications, etc., permettant aux membres de l'équipe de communiquer rapidement, de coordonner les tâches et de rester informés des mises à jour importantes.
- La sécurité des données : L'un des objectifs clés est aussi de garantir la sécurité des informations. Elle doit offrir des mesures de sécurité avancées telles que les

autorisations d'accès personnalisées, les sauvegardes régulières, etc., pour protéger les données contre les accès non autorisés, les pertes ou les dommages.

- Amélioration de l'efficacité opérationnelle : Le projet vise aussi à optimiser les processus opérationnels en automatisant les tâches, en rationalisant les flux de travail et en réduisant les erreurs humaines. Elle permet également de suivre et d'analyser les données, fournissant ainsi des informations précieuses pour prendre des décisions éclairées et améliorer les performances de l'organisation.
- Facilite l'accès aux informations : Une plateforme de gestion des données vise à rendre les informations facilement accessibles à tous les membres de l'organisation. Elle offre une interface conviviale et intuitive, des fonctionnalités de recherche avancées et une organisation structurée des données, permettant aux utilisateurs de trouver rapidement les informations dont ils ont besoin.

1.3.Intérêts de l'objet de la recherche

D'après les objectifs du projet, nous avons identifié les intérêts suivants :

- Il suffit de quelques clics pour trouver des informations concernant les utilisateurs de la plateforme (administrateurs, professeurs, étudiants).
- Un tableau de communiqués pour les nouvelles à partager au sein de l'établissement visible par tous les utilisateurs n'importe quand et n'importe où.
- Possibilité d'envoi de mail dans la plateforme.
- Facilité de partage des documents/support de cours entre professeurs et étudiants.
- Facilité de discussion entre utilisateurs.
- Facilité de téléchargement de documents utiles comme les notes ou les différents certificats.
- Différentes accessibilités en fonction du rôle de chaque utilisateur.

1.4.Architecture de la plateforme

L'architecture d'une plateforme web se réfère à la structure, à la conception et à l'organisation globale d'un système ou d'une application web. Elle définit comment les différents composants d'une plateforme web interagissent entre eux pour offrir les fonctionnalités et les services attendus. Observons les principales composantes d'une architecture de plateforme web :

Front-end : C'est la partie visible de la plateforme web, qui est accessible par les utilisateurs via leur navigateur. Elle est généralement développée en utilisant des technologies de développement web comme HTML, CSS et JavaScript, et elle est responsable de l'interface utilisateur et de l'expérience utilisateur de la plateforme.

Back-end : C'est la partie invisible de la plateforme web, qui gère les fonctionnalités et les données en arrière-plan. Elle est généralement développée en utilisant des langages de programmation comme Python, PHP, Ruby, Java, ou Node.js, et elle est responsable de la logique métier, de l'authentification, de l'autorisation, de la gestion des bases de données, de l'intégration de services externes, et de la gestion des requêtes et des réponses entre le front-end et les autres composants.

Base de données : C'est le composant qui stocke les données de la plateforme web, comme les informations des utilisateurs, les données des produits, les paramètres de configuration, etc. Elle peut être basée sur différents systèmes de gestion de base de données (SGBD) comme MySQL, PostgreSQL, MongoDB, ou d'autres, en fonction des besoins de la plateforme.

Services externes : Ce sont des services tiers utilisés par la plateforme web pour étendre ses fonctionnalités, comme les API de paiement, les services de messagerie, les services de géolocalisation, les services de stockage en ligne, etc. Ils sont intégrés à la plateforme web via des API (interfaces de programmation d'application) pour permettre l'échange de données et la communication avec ces services externes.

Sécurité : C'est un aspect essentiel de l'architecture d'une plateforme web pour garantir la protection des données sensibles des utilisateurs et la sécurisation des transactions. Cela peut inclure la mise en œuvre de protocoles de sécurité tels que HTTPS, la gestion des authentifications et des autorisations, la protection contre les attaques de sécurité, la validation des données d'entrée, la gestion des erreurs, etc.

Scalabilité : C'est la capacité d'une plateforme web à gérer efficacement une augmentation du nombre d'utilisateurs et de la charge de travail. Une architecture scalable doit permettre l'ajout de ressources (comme des serveurs) pour répondre à la demande croissante sans compromettre les performances et la disponibilité de la plateforme.

Concernant le projet, l'architecture correspondante est la suivante :

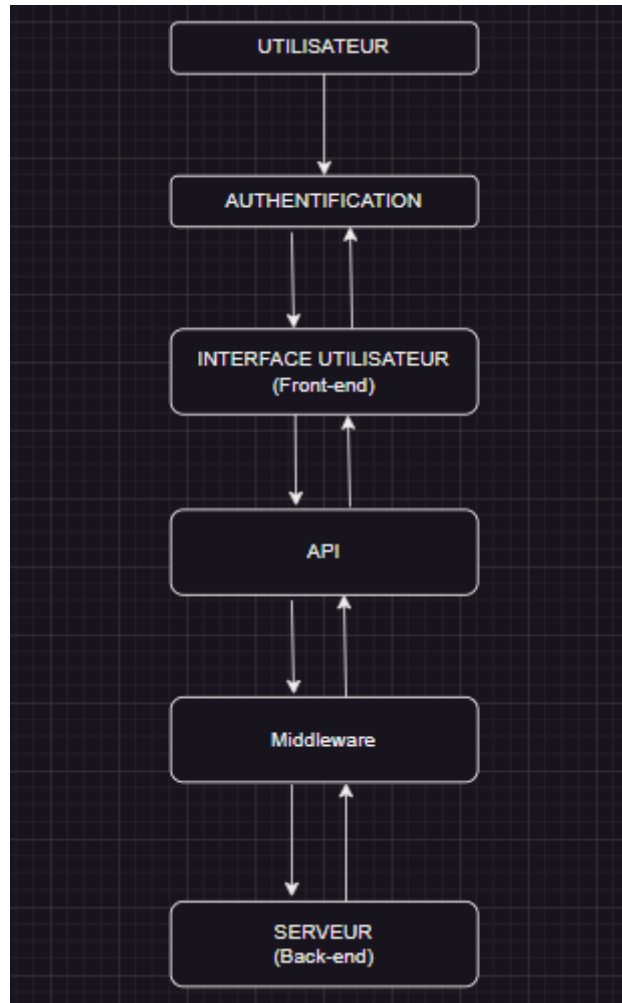


Figure 1 : Architecture de la plateforme

1.5.Explication de l'architecture

1.5.1. Authentification

L'authentification est le processus de vérification de l'identité ou de l'authenticité d'une personne, d'un appareil ou d'une entité avant d'accorder l'accès ou les autorisations à certaines ressources, systèmes ou services. Il s'agit d'une mesure de sécurité utilisée pour s'assurer que seules les personnes ou entités autorisées peuvent accéder à des informations protégées, des comptes ou des systèmes, tout en empêchant les accès non autorisés ou les violations de données.

L'authentification implique généralement la présentation de justificatifs tels que des noms d'utilisateur et des mots de passe, des jetons de sécurité, des données biométriques ou

d'autres preuves d'identité. Ces justificatifs sont vérifiés par rapport à une source de confiance, telle qu'une base de données ou un serveur d'authentification, pour déterminer si l'entité qui tente d'accéder au système ou à la ressource est effectivement le propriétaire légitime des justificatifs.

L'authentification est un élément crucial de la sécurité de l'information et est utilisée dans divers contextes, notamment l'authentification des utilisateurs pour les comptes en ligne, le contrôle d'accès aux installations physiques ou aux réseaux informatiques, et l'authentification des appareils dans l'écosystème de l'Internet des objets (IoT). Différentes méthodes et techniques d'authentification peuvent être utilisées en fonction du niveau de sécurité requis et du cas d'utilisation spécifique.

Bien évidemment, la plateforme dispose d'une interface d'authentification pour permettre seulement aux utilisateurs vérifiés d'accéder à ses ressources et pour garantir la sécurité des données.

1.5.2. Interface Utilisateur (IU)

Une interface utilisateur est un terme utilisé dans le domaine de l'informatique pour désigner le moyen par lequel un utilisateur interagit avec un système informatique, qu'il s'agisse d'un logiciel, d'un site web, d'une application ou d'un autre type de système. Une IU est conçue pour permettre à un utilisateur d'interagir avec un système informatique de manière conviviale et efficace.

Une IU peut inclure divers éléments, tels que des éléments visuels (comme des boutons, des menus déroulants, des icônes), des éléments interactifs (comme des formulaires, des boîtes de dialogue, des curseurs) et des éléments de navigation (comme des menus, des onglets, des liens). L'objectif d'une IU est de fournir à l'utilisateur une expérience de navigation fluide et intuitive, lui permettant d'accomplir ses tâches de manière efficace et agréable.

Les IU peuvent varier considérablement en fonction du type de système informatique et du public cible. Les principes de conception d'une IU peuvent inclure la facilité d'utilisation, la cohérence, la clarté, l'efficacité et l'accessibilité, entre autres.

En somme, une interface utilisateur est la partie du système informatique avec laquelle les utilisateurs interagissent pour accomplir leurs tâches et obtenir les résultats souhaités. Une IU bien conçue peut améliorer l'expérience de l'utilisateur, rendre l'utilisation d'un système plus facile et plus efficace, et contribuer à la satisfaction de l'utilisateur.

1.5.3 API (Application Programming Interface)

Une API (Application Programming Interface) est un ensemble de règles et de conventions qui permettent à différents logiciels de communiquer et d'interagir les uns avec les autres. En d'autres termes, c'est une interface qui définit comment les différents composants logiciels peuvent interagir et échanger des informations.

Une API agit comme un pont entre différentes applications, permettant à celles-ci de partager des données, d'accéder à des fonctionnalités ou de communiquer entre elles de manière structurée. Elle définit les méthodes de communication, les formats de données, les conventions d'appel et les autorisations nécessaires pour interagir avec une application spécifique.

Les APIs sont largement utilisées dans le développement de logiciels et sont essentielles pour la création d'applications interconnectées et pour l'intégration de services tiers. Par exemple, de nombreuses plateformes de médias sociaux, services de paiement en ligne, services de cartographie, services de messagerie et autres fournisseurs de services proposent des APIs pour permettre aux développeurs d'accéder à leurs fonctionnalités et d'intégrer leurs services dans leurs propres applications.

Les APIs peuvent être de différents types, notamment les APIs RESTful (Representational State Transfer), les APIs SOAP (Simple Object Access Protocol), les APIs GraphQL, les APIs de bases de données, les APIs de messagerie, les APIs de médias sociaux, et bien d'autres. Elles sont utilisées dans une large gamme d'applications, du développement d'applications web et mobiles à l'Internet des objets (IoT), en passant par les services cloud, les applications d'entreprise, les systèmes intégrés et plus encore.

Autrement dit, une API est une interface qui permet à différentes applications de communiquer et d'interagir entre elles de manière structurée, en définissant les règles et les conventions de communication. Elle est un élément clé de l'intégration et de l'interopérabilité des applications logicielles.

1.5.4. Middleware

Un middleware est un concept informatique qui fait référence à un logiciel ou à un composant logiciel qui agit comme une couche intermédiaire entre différentes applications, systèmes ou composants dans un environnement informatique. Il peut être utilisé dans différents contextes, tels que le développement de logiciels, la gestion de réseaux, les systèmes d'exploitation, les serveurs web et les applications client-serveur.

En général, le middleware sert d'intermédiaire pour faciliter la communication, la coordination et l'intégration entre les différentes parties d'un système informatique. Il peut gérer les interactions entre les composants, les protocoles de communication, la sécurité, la gestion des erreurs, la transformation des données, la gestion des transactions, la mise en cache et d'autres fonctionnalités nécessaires pour la mise en œuvre d'une architecture logicielle complexe.

Un exemple courant de middleware est le middleware de serveur web, qui intervient entre le serveur web et les applications web pour gérer des tâches telles que la gestion des sessions utilisateur, la gestion des requêtes HTTP, la gestion des cookies, la mise en cache et d'autres fonctionnalités liées à la communication entre le client et le serveur. Un autre exemple est le middleware de gestion des transactions utilisé dans les systèmes de bases de données pour garantir l'intégrité des transactions et la cohérence des données.

Le middleware peut être développé en interne, acheté auprès de fournisseurs tiers ou utilisé en tant que service dans le cloud. Il peut être essentiel pour permettre l'intégration et l'interopérabilité entre les différents composants d'un système informatique complexe et faciliter le développement et la gestion des applications.

1.5.5. Serveur (back-end)

Un serveur back-end est une composante d'un système informatique qui gère la logique de traitement des données et la communication avec les clients ou les utilisateurs d'une application ou d'un site web. Il est responsable de la gestion des données, du traitement des requêtes, de l'authentification et de l'autorisation des utilisateurs, et du logique métier de l'application.

Le serveur back-end traite les requêtes envoyées par les clients du côté front-end, effectue les opérations nécessaires sur les données, et renvoie les réponses appropriées. Il peut communiquer avec d'autres composantes du système, tels que les bases de données, les services externes, ou les autres serveurs, pour récupérer ou stocker des données nécessaires au fonctionnement de l'application.

Le serveur back-end est souvent développé en utilisant des langages de programmation tels que Java, C#, Python, JavaScript, Ruby, ou PHP, et peut s'appuyer sur des Framework ou des bibliothèques pour faciliter le développement et la gestion des tâches courantes, tels que la gestion des requêtes HTTP, la gestion des bases de données, l'authentification et l'autorisation des utilisateurs, etc. Dans notre cas, le serveur back-end est développé par le langage de programmation JavaScript avec le Framework Node.js.

Le serveur back-end est essentiel pour assurer le bon fonctionnement d'une application ou d'un site web et pour gérer les opérations complexes liées au logique métier de l'application, tandis que le client ou l'interface utilisateur front-end est responsable de la présentation des données et de l'interaction avec les utilisateurs. Les deux côtés (front-end et back-end) travaillent ensemble pour fournir une expérience utilisateur complète et fonctionnelle.

1.6. Gestion des utilisateurs

La gestion des utilisateurs dans une plateforme fait référence à la manière dont les comptes d'utilisateurs sont créés, gérés, modifiés et supprimés dans le cadre d'une application ou d'un système en ligne. Cela peut inclure la gestion des informations d'identification, des autorisations d'accès, des rôles et des responsabilités, ainsi que la sécurité des données des utilisateurs.

Création de comptes d'utilisateurs : La plateforme permet la création de comptes d'utilisateurs de manière simple et sécurisée. Cela peut inclure la collecte des informations d'identification nécessaires, telles que les noms d'utilisateur, les mots de passe et les adresses e-mail valides, ainsi que la vérification de ces informations pour éviter les comptes frauduleux.

Authentification et autorisation : La plateforme met en place un système d'authentification pour vérifier l'identité des utilisateurs et garantir qu'ils ont l'autorisation d'accéder aux fonctionnalités et aux données appropriées. Cela peut inclure l'utilisation de méthodes d'authentification à plusieurs facteurs pour renforcer la sécurité.

Gestion des autorisations : La plateforme permet la gestion fine des autorisations d'accès des utilisateurs en fonction de leurs rôles et de leurs responsabilités. Cela inclut la définition de différents niveaux d'autorisation pour différents types d'utilisateurs, ainsi que la gestion des autorisations spécifiques pour les fonctionnalités et les données sensibles.

Gestion des profils d'utilisateurs : La plateforme permet aux utilisateurs de gérer leurs profils, y compris les informations personnelles, les préférences de communication, les préférences de confidentialité, etc. Les utilisateurs peuvent également modifier et mettre à jour leurs informations de profil en toute sécurité.

Gestion des comptes inactifs ou supprimés : La plateforme possède des politiques en place pour gérer les comptes inactifs ou supprimés, y compris la désactivation ou la suppression sécurisée des comptes inutilisés ou indésirables, ainsi que la gestion des données associées à ces comptes.

La gestion des utilisateurs dans une plateforme est un aspect essentiel de la sécurité et de la convivialité d'une application ou d'un système en ligne. Dans notre cas, seuls les administrateurs peuvent ajouter des nouveaux utilisateurs.

1.7. Fonctionnalités dans l'application

Tout d'abord, il y a la fonctionnalité d'authentification, qui est primordial pour garantir la sécurité des données.

Ensuite, l'application possède les fonctionnalités suivantes :

1.7.1 Gestion des utilisateurs

L'application permet la gestion des utilisateurs tels que la création, les mises à jour, lecture et la suppression des utilisateurs. Cela dit, seuls les administrateurs en ont le droit et l'accès.

Un rôle spécifique est attribué à chaque utilisateur et leurs accessibilités dans la plateforme dépendent de ce même rôle. Ces rôles sont les suivants : Administrateur, Professeur, Etudiant.

La plateforme possède des interfaces/formulaire pour la création, la lecture, la mise à jour et la suppression des données d'utilisateur.

A chaque utilisateur crée, la plateforme envoie un mail à cette dernière. Cet e-mail est une forme de salutation de bienvenue dans la plateforme mais aussi il contient un mot de passe par défaut pour permettre au destinataire de se connecter à la plateforme.

1.7.2 Gestion des communiqués

L'application enregistre une fonction d'ajout et affichage des communiqués. Il s'agit d'un tableau virtuel où seront exposés tous les communiqués/nouvelles dans l'établissement scolaire.

Son fonctionnement est la suivante : un administrateur ajoute une information à partager, que ce soit un fichier ou une image ou simplement du texte, ensuite il le publie. Après cette action, chaque utilisateur reçoit une alerte de nouveau communiquée et ils peuvent directement les consulter dans l'onglet dédié spécialement pour ces informations.

Il est à noter que seuls les administrateurs ont le droit et privilège d'ajouter ces nouvelles/communiqués.

1.7.3 Envoi de mail à partir de la plateforme

Autre que les mails envoyés à chaque nouvel utilisateur, l'application offre une possibilité d'envoyer des mails à partir de la même plateforme. Cela est possible par la librairie du Framework Node.js qui est nodemailer.

L'application possède 500 mails journaliers gratuits à disposition. Il suffit de se diriger vers l'onglet spécifique pour l'envoi des mails.

1.7.4 Communication et partage de documents entre utilisateurs

La plateforme permet de se communiquer entre utilisateurs. Les utilisateurs peuvent s'envoyer des messages entre eux, les étudiants entre étudiants, ou étudiants entre professeurs, ou professeurs entre professeurs ou encore avec des administrateurs.

Autre que les messages de type textes, ils peuvent aussi envoyer des fichiers ou des images.

1.7.5 Téléchargement de document

L'application permet la fonction de téléchargement de document. Autre que les fichiers partagés qui sont téléchargeables, on peut aussi télécharger des documents internes. Par exemple, les lettres d'introduction, les disciplines internes de l'établissement scolaire, les certificats de scolarités ainsi que les notes et les résultats des tests.

CHAPITRE II : ETUDE THEORIQUE DU THEME

2.1. Application Web

Une application web est une application informatique qui est exécutée dans un navigateur web, tel que Google Chrome, Mozilla Firefox ou Microsoft Edge. Contrairement aux applications traditionnelles qui doivent être installées sur un ordinateur ou un appareil mobile, une application web ne nécessite qu'un accès à internet et un navigateur web pour être utilisée. En cas d'utilisation local, il suffit d'avoir un navigateur pour exécuter l'application.

Les applications web peuvent avoir différentes fonctionnalités et peuvent être utilisées pour diverses tâches, telles que la gestion de projet, la communication, la comptabilité, la planification et bien plus encore. Les applications web peuvent être développées pour être utilisées sur des ordinateurs de bureau, des ordinateurs portables, des tablettes et des smartphones, et peuvent être accessibles à partir de différents systèmes d'exploitation.

2.2. Avantages d'une application web

Les applications web offrent de nombreux avantages par rapport aux applications traditionnelles installées localement. Quelques-uns des avantages clés d'une application web sont présentés ci-dessous :

Accessibilité : Les applications web peuvent être facilement accessibles à partir de n'importe quel appareil disposant d'une connexion Internet et d'un navigateur web. Les utilisateurs peuvent y accéder à partir de leur ordinateur, smartphone, tablette ou tout autre appareil compatible, ce qui offre une grande flexibilité et une portabilité accrue.

Pas besoin d'installation : Contrairement aux applications traditionnelles, les applications web ne nécessitent pas d'installation préalable sur l'appareil de l'utilisateur. Cela facilite l'accès et la mise à jour de l'application, car les utilisateurs n'ont pas besoin de télécharger ni d'installer de logiciel supplémentaire.

Mises à jour transparentes : Les mises à jour des applications web peuvent être déployées côté serveur, ce qui signifie que les utilisateurs bénéficient automatiquement des dernières fonctionnalités et des correctifs de sécurité sans avoir à effectuer de mises à jour manuelles. Cela garantit une expérience utilisateur à jour et réduit les problèmes de compatibilité liés aux différentes versions de logiciels.

Coût réduit : Le développement et la maintenance d'une application web peuvent être plus rentables par rapport aux applications traditionnelles. Les mises à jour et les améliorations

peuvent être déployées pour tous les utilisateurs simultanément, ce qui permet d'économiser des coûts liés à la distribution et à l'installation de mises à jour sur chaque appareil utilisateur individuellement.

Collaboration en temps réel : Les applications web permettent une collaboration en temps réel entre les utilisateurs, facilitant ainsi le travail d'équipe et la communication. Plusieurs utilisateurs peuvent accéder à l'application simultanément et travailler ensemble sur des documents ou des projets, ce qui améliore l'efficacité et la productivité.

Sauvegarde et récupération des données : Les applications web peuvent offrir des fonctionnalités de sauvegarde automatique des données sur des serveurs distants. Cela garantit la sécurité des données et facilite leur récupération en cas de problème technique ou de perte de l'appareil utilisateur.

Évolutivité et extensibilité : Les applications web sont plus facilement extensibles et évolutives, car elles peuvent être hébergées sur des serveurs distants capables de gérer une charge croissante. Il est plus facile d'ajouter de nouvelles fonctionnalités, d'optimiser les performances et de faire évoluer l'application en fonction des besoins changeants de l'entreprise ou des utilisateurs.

Plateforme indépendante : Les applications web sont généralement développées pour être compatibles avec différents systèmes d'exploitation, navigateurs web et appareils. Cela signifie que les utilisateurs peuvent accéder à l'application quel que soit leur environnement technologique, ce qui offre une plus grande flexibilité et un plus large public cible.

2.3. Inconvénients d'une application web

Bien que les applications web offrent de nombreux avantages, elles présentent également certains inconvénients potentiels.

Dépendance à Internet : Les applications web nécessitent une connexion Internet stable pour fonctionner. Si la connexion Internet est lente, instable ou indisponible, cela peut entraîner une expérience utilisateur médiocre voire une incapacité d'accéder à l'application. Les applications web peuvent parfois être plus lentes que les applications locales, en raison de la latence du réseau et de la nécessité de communiquer avec un serveur distant. Les temps de chargement des pages peuvent être plus longs, en particulier pour les applications avec une grande quantité de données ou une complexité élevée.

Contraintes de fonctionnalités : Certaines fonctionnalités avancées et spécifiques du système d'exploitation ou du matériel de l'appareil de l'utilisateur peuvent être difficiles à

implémenter ou indisponibles dans une application web. Cela peut limiter les possibilités d'interaction avec les périphériques locaux ou les fonctionnalités spécifiques de l'appareil.

Sécurité : Les applications web peuvent présenter des risques de sécurité, notamment en raison des vulnérabilités potentielles des navigateurs web, des attaques par injection de code et des problèmes liés à la gestion des données sensibles. Des mesures de sécurité adéquates doivent être mises en place pour protéger les données et prévenir les cyberattaques.

Dépendance aux mises à jour du navigateur : Les applications web peuvent dépendre des fonctionnalités et des performances des navigateurs web utilisés par les utilisateurs. Par conséquent, des problèmes de compatibilité peuvent survenir si les utilisateurs n'utilisent pas les versions les plus récentes ou les navigateurs pris en charge par l'application.

Intégration système limitée : L'intégration avec d'autres systèmes ou logiciels peut être plus complexe dans une application web, en particulier si des protocoles ou des interfaces spécifiques sont nécessaires. Des défis peuvent se poser lors de l'intégration avec des systèmes existants ou des applications tierces.

2.4. Une plateforme web

Une plateforme web est un ensemble de technologies, de logiciels et d'infrastructures qui permettent à des utilisateurs de créer, de partager, de stocker et d'accéder à des informations et des services en ligne.

Les plateformes web sont souvent constituées d'un ensemble d'applications et de services qui sont intégrés de manière transparente pour permettre aux utilisateurs d'interagir facilement avec elles. Elles peuvent être utilisées pour de nombreuses applications différentes, allant des réseaux sociaux aux systèmes de gestion de contenu, en passant par les plateformes de commerce électronique et les applications de productivité.

Les plateformes web sont des environnements en ligne qui permettent aux utilisateurs de créer, de partager, de stocker et d'accéder à du contenu et des services. Ces plateformes peuvent prendre différentes formes, allant des réseaux sociaux aux plateformes de commerce électronique, en passant par les blogs, les forums de discussion et les applications de productivité.

Les plateformes web peuvent être utilisées à des fins personnelles ou professionnelles, et elles offrent de nombreux avantages, tels que la facilité d'utilisation, l'accessibilité à partir de n'importe quel appareil connecté à Internet, la possibilité de collaborer avec d'autres utilisateurs en temps réel et la capacité à stocker des données et des fichiers en toute sécurité dans le cloud.

Les plateformes web ont également transformé de nombreux aspects de la vie quotidienne et des affaires, en offrant des moyens innovants de communication, de marketing, de vente, de gestion des projets et de collaboration à distance. Cependant, elles soulèvent également des préoccupations en matière de confidentialité des données, de sécurité et de dépendance envers les fournisseurs de services en ligne.

En fin de compte, les plateformes web ont transformé notre façon de travailler, de communiquer, de partager des informations et de consommer des produits et des services en ligne, et elles continuent d'évoluer pour répondre aux besoins changeants des utilisateurs et des entreprises.

2.5. Différence entre une application web et une plateforme web

Une plateforme web est un environnement en ligne qui offre une variété de services et de fonctionnalités, généralement sous la forme de logiciels en tant que service ou Software as a Service en anglais (SaaS). Les plateformes web peuvent inclure des outils pour la gestion de projets, la communication en ligne, la création de sites web, le stockage en ligne et la gestion des relations clients. Les utilisateurs peuvent accéder à ces services via un navigateur web sans avoir à télécharger ou installer de logiciel sur leur ordinateur.

Une application web, quant à elle, est un programme informatique qui s'exécute sur un navigateur web. Les applications web peuvent être des sites web interactifs, des applications de commerce électronique, des applications de réseaux sociaux, des jeux en ligne, des applications de productivité et plus encore. Contrairement à une plateforme web, une application web est généralement conçue pour répondre à un besoin spécifique et offre souvent des fonctionnalités avancées qui ne sont pas disponibles sur un site web standard.

En résumé, une plateforme web est une collection de services et de fonctionnalités en ligne, tandis qu'une application web est un programme informatique qui s'exécute sur un navigateur web et offre une fonctionnalité spécifique.

2.6. Un navigateur web

Un navigateur web est un logiciel conçu pour permettre aux utilisateurs d'accéder et de naviguer sur Internet. Il s'agit d'une application qui interprète le code HTML, CSS et JavaScript des sites web pour les afficher de manière conviviale pour l'utilisateur.

Les navigateurs web permettent également aux utilisateurs de naviguer sur Internet en suivant des liens hypertextes, de saisir des requêtes de recherche, de remplir des formulaires en ligne, de visionner des vidéos et d'interagir avec des applications web. Les navigateurs les plus populaires incluent Google Chrome, Mozilla Firefox, Safari, Microsoft Edge et Opera.

En résumé, un navigateur web est une application logicielle qui permet aux utilisateurs de naviguer sur Internet et d'accéder aux pages web.

2.7. Un serveur web

En termes simples, un serveur web est un ordinateur qui stocke, traite et fournit des fichiers de sites internet aux navigateurs web.

Les serveurs web se composent de matériel et de logiciels qui utilisent le protocole HTTP (HyperText Transfer Protocol). Il s'agit ici de répondre aux requêtes des utilisateurs web effectuées via le World Wide Web (www).

Grâce à ce processus, les serveurs web chargent et délivrent la page web demandée au navigateur de l'utilisateur.

Les serveurs web emploient également le protocole SMTP (Simple Mail Transfer Protocol) et le protocole FTP (File Transfer Protocol) pour traiter les fichiers pour les courriers électroniques ou le stockage selon le besoin et l'orientation de l'application.

2.8. Composant matériels et logiciels d'un serveur web

Côté matériel, un serveur web se connecte à Internet. Cela lui permet d'échanger des données ou des fichiers avec d'autres appareils également connectés. Ces données peuvent se présenter sous différentes formes, telles que des fichiers HTML, des images, des fichiers JavaScript ou des feuilles de style CSS. Le matériel du serveur web stocke en même temps le logiciel de serveur web.

Le logiciel de serveur web contrôle la manière dont les utilisateurs web accèdent aux fichiers hébergés. Il contient plusieurs composants, hébergeant au moins un serveur HTTP. Un serveur HTTP est un logiciel capable de comprendre les requêtes HTTP et les URLs.

2.9. Fonctionnement d'un serveur web

Les serveurs web suivent un modèle client-serveur. Dans cette structure, un programme, également appelé client, demande une ressource ou un service à un autre programme, le serveur.



Figure 2 : Fonctionnement du HTTP (source : HOSTINGER)

Pour traiter les requêtes des clients web, les serveurs web suivent quelques étapes nécessaires :

- Lorsqu'un internaute souhaite charger le contenu d'un site web, son navigateur web demande l'accès via Internet. C'est ce qu'on appelle une requête HTTP. Le navigateur web recherche l'adresse IP du site internet demandé en traduisant l'URL des pages web via le Système de Noms de Domaines (DNS) ou en cherchant dans son cache. Ce processus localise le serveur web sur lequel les fichiers du site internet sont hébergés.
- Le serveur web reçoit la requête HTTP et la traite via son serveur HTTP. Une fois que son serveur HTTP accepte la requête, il effectue une recherche dans les fichiers du serveur afin d'obtenir les données requises du site internet.
- Après cela, le serveur web renvoie les fichiers du site au navigateur web qui a envoyé la demande. Ensuite, l'internaute voit le contenu des pages web.

Cependant, si le serveur HTTP ne parvient pas à trouver ou à traiter les fichiers demandés sur le serveur, il répond au navigateur web par un message d'erreur.

2.10. Serveur web statique ou serveur web dynamique

Un serveur web peut être classé en deux catégories principales : statique et dynamique, en fonction de la manière dont il génère et renvoie les pages web aux clients.

2.10.1. Serveur web statique

Un serveur web statique renvoie des pages web qui restent les mêmes pour tous les utilisateurs et dans toutes les circonstances. Les pages web statiques sont préalablement créées

et stockées sur le serveur, et lorsque le serveur reçoit une demande d'un client, il renvoie simplement la version préexistante de la page demandée.

Les serveurs web statiques sont généralement utilisés lorsque le contenu d'un site web ne change pas fréquemment ou lorsqu'il n'y a pas besoin d'interactions complexes avec les utilisateurs. Ils sont plus simples à configurer et à gérer, et offrent généralement des performances élevées car ils n'ont pas besoin de générer dynamiquement les pages à la volée.

2.10.2. Serveur web dynamique

Un serveur web dynamique génère les pages web en temps réel en fonction des requêtes des clients et des données présentes dans une base de données ou d'autres sources de données. Les pages web dynamiques sont générées au moment de la demande, et elles peuvent varier en fonction des informations spécifiques à chaque utilisateur, des paramètres ou des actions effectuées sur le site.

Les serveurs web dynamiques sont couramment utilisés dans des scénarios où le contenu doit être personnalisé pour chaque utilisateur, où des interactions complexes sont nécessaires, ou lorsque le contenu du site est mis à jour fréquemment.

Les serveurs web dynamiques nécessitent souvent une configuration et une gestion plus complexes en raison de la nécessité de gérer les requêtes, de se connecter à des bases de données et de gérer l'état de session des utilisateurs. Ils peuvent également nécessiter des ressources de serveur supplémentaires pour générer les pages à la volée, ce qui peut affecter les performances dans les cas de forte charge.

2.11. Fonctionnalités du serveur web

Les serveurs web offre différentes fonctionnalités pour permettre la gestion et la diffusion de sites web.

- **Hébergement de sites web :** Le serveur web est responsable de l'hébergement des fichiers et du contenu d'un site web. Il stocke les fichiers HTML, CSS, JavaScript, images et autres ressources nécessaires pour afficher le site.
- **Traitement des requêtes HTTP :** Le serveur web est capable de recevoir des requêtes HTTP (Hyper-Text Transfer Protocol) des clients (navigateurs web) et de les traiter. Il peut interpréter les demandes, récupérer les fichiers demandés et les renvoyer au client en tant que réponse HTTP.

- **Journalisation des fichiers** : Les fichiers journaux documentent tous les événements ou activités que les serveurs web effectuent, tels que les requêtes, la sécurité et les erreurs. Chaque fois qu'un serveur web reçoit une nouvelle requête, une ligne de texte est ajoutée au journal.
- **Authentification** : De nombreux serveurs offrent cette fonctionnalité avant d'autoriser un accès partiel ou complet aux ressources d'un site web. Les fonctions d'authentification impliquent souvent des demandes d'autorisation. Cela se produit lorsqu'un nom d'utilisateur et un mot de passe sont requis.
- **Limitation de la bande passante** : La bande passante d'un serveur web est la quantité de données qu'il peut transférer ou traiter à un moment donné. La limitation de la bande passante contrôle la vitesse des réponses pour s'assurer qu'un réseau n'est pas saturé et peut livrer les fichiers sans problèmes.
- **Espace de stockage** : Il fait référence à la quantité d'espace disque disponible pour stocker des fichiers. Il détermine par conséquent si un serveur web peut héberger un site internet avec des caractéristiques spécifiques.
- **Langage de programmation** : Le langage de programmation d'un serveur web est le type de code utilisé pour développer des programmes exécutés par un serveur. Il est également connu sous le nom de langages de script côté serveur.
- **Disponibilité** : La disponibilité du serveur suit la durée pendant laquelle un serveur web fonctionnel et peut traiter les demandes ou fournir des fichiers.

CHAPITRE III : UNE APPLICATION WEB

La création d'une application web nécessite plusieurs étapes. Ci-dessus un aperçu des étapes générales pour créer une application web :

1. **Définir les objectifs et les fonctionnalités de l'application web** : Avant de commencer à construire une application web, vous devez définir clairement les objectifs et les fonctionnalités de l'application. Cela vous aidera à déterminer les technologies nécessaires pour construire l'application.
2. **Concevoir l'interface utilisateur** : La conception de l'interface utilisateur est une étape importante de la création d'une application web. Elle doit être facile à utiliser et intuitive pour les utilisateurs.
3. **Choisir une technologie de développement web** : Il existe de nombreuses technologies de développement web disponibles pour créer une application web, comme les langages de programmation (par exemple, JavaScript, Python, Ruby), les Framework (par exemple, React, Angular, Vue) et les bases de données.
4. **Développer l'application web** : Cette étape consiste à écrire du code pour créer l'application web en utilisant la technologie choisie. Vous devrez également créer une base de données pour stocker les informations de l'application.
5. **Tester l'application web** : Il est important de tester l'application web pour s'assurer qu'elle fonctionne correctement et qu'elle est exempte d'erreurs. Vous pouvez effectuer des tests manuels et automatisés pour identifier les bogues et les problèmes de performance.
6. **Déployer l'application web** : Lorsque vous êtes satisfait de l'application, vous pouvez la déployer sur un serveur pour qu'elle soit accessible aux utilisateurs. Il existe de nombreux services d'hébergement web disponibles pour déployer une application web.

3.1. Langage de programmation

3.1.1. HTML

L'HTML, abréviation de HyperText Markup Language, est un langage de balisage utilisé pour créer et structurer des pages web. Il s'agit d'un langage de base pour le développement web, permettant de décrire la structure et le contenu d'une page web en utilisant des balises et des attributs.

L'HTML est utilisé pour créer des pages web statiques et interactives, en définissant la structure du contenu, tels que les titres, les paragraphes, les listes, les images, les liens, etc.

Les balises contiennent des éléments, éventuellement enrichis avec des attributs. La page web est construite depuis un fichier HTML, élément par élément.

Pour écrire en HTML, il faut insérer des éléments et des attributs dans des balises, en utilisant la syntaxe imposée. Un paragraphe de texte, par exemple, constitue un élément à écrire entre les balises < >.

Les éléments HTML, « tags » en anglais, désignent les différents types de contenus qui structurent la page web. Un paragraphe de texte, une image et une liste numérotée, par exemple, sont trois types de contenus distincts, chacun constitue donc un élément HTML spécifique. Pour indiquer au navigateur le type de contenu à afficher, il faut utiliser le nom de l'élément correspondant.

En langage HTML, la plupart des éléments s'écrivent, dans l'ordre :

- Une balise ouvrante : elle indique que l'élément commence. Elle s'écrit avec un chevron ouvrant et un chevron fermant, à l'intérieur duquel renseigner le nom de l'élément : < nom de l'élément >. La balise ouvrante n'est pas visible à l'écran.
- Le contenu : le contenu de l'élément s'affiche à l'écran.
- Une balise fermante : elle clôture l'élément. Elle s'écrit comme la balise ouvrante, en ajoutant une barre oblique après le chevron ouvrant : < /nom de l'élément >.

Pour insérer un élément paragraphe, par exemple, il faut écrire : <p> texte du paragraphe< /p>. Seul le contenu, c'est-à-dire la partie de l'élément « texte du paragraphe », est affiché par le navigateur, le reste du code est invisible. À noter que les éléments HTML ne sont pas sensibles à la casse, c'est-à-dire que l'emploi de majuscules et de minuscules est indifférent. L'élément paragraphe s'écrit indistinctement < p > et < P >. Pour des raisons de lisibilité et de cohérence, le développeur néanmoins a intérêt à utiliser systématiquement les minuscules.

Code HTML :

```
<p>Le <strong>mot</strong> est en  
gras.</p>
```

Résultat dans le navigateur :

Le **mot** est en gras.

Figure 3 : Exemple de code HTML

Le langage HTML qualifie certains éléments :

- Les éléments HTML dits « vides » n'utilisent pas la syntaxe balise d'ouverture, contenu, balise de fermeture : ils se composent d'une balise unique. C'est notamment le cas des éléments vides image « img » et saut de ligne « br ». Pour sauter une ligne à la fin d'une phrase, par exemple, il suffit d'insérer l'élément < br > sans contenu ni balise de fermeture.
- Le langage HTML distingue les éléments de bloc des éléments en ligne. Les éléments de bloc structurent la page : chacun représente une portion de contenu, séparée des autres. Les éléments en ligne donnent des informations sur une partie d'un contenu. Illustration : l'élément de bloc paragraphe < p > inclut l'élément en ligne lien hypertexte < a >, pour insérer une partie du texte cliquable à l'intérieur du paragraphe.

Il est possible d'imbriquer des éléments les uns dans les autres : un élément est inséré dans un autre élément, pour augmenter le niveau d'information à fournir au navigateur à propos d'un contenu. Exemple : pour mettre un mot en gras dans un paragraphe, il faut imbriquer l'élément en ligne < strong > dans l'élément de bloc < p >. Le langage HTML est le suivant : < p > Le < strong >mot< /strong > est en gras. < /p >.

Le langage HTML suffit à coder une page web. En pratique néanmoins, d'autres langages informatiques lui sont souvent associés : les feuilles de style en cascade CSS permettent de personnaliser de manière rapide et avancée la mise en forme de la page web, JavaScript permet d'ajouter du contenu dynamique.

3.1.1.1. Ecriture des attributs HTML

Comme les éléments, les attributs HTML donnent de l'information au navigateur pour afficher le contenu. Mais à la différence de l'élément :

- L'attribut n'est pas structurant : il ne détermine pas un type de contenu à afficher, il décrit le contenu.
- L'attribut ne se suffit pas à lui-même : il est toujours l'attribut d'un élément. À noter que l'attribut est majoritairement facultatif. Certains éléments néanmoins ne fonctionnent qu'accompagnés d'un attribut. C'est le cas notamment de l'élément lien hypertexte `< a >` qui ne fonctionne qu'à condition de renseigner l'attribut `href`.
- L'attribut s'écrit différemment : il a un nom et une valeur. Le nom de l'attribut est renseigné après le nom de l'élément, et suivi du signe égal ; la valeur de l'attribut est renseignée entre guillemets : `< élément attribut="" > contenu < /élément >`. Un élément peut inclure plusieurs attributs.

À noter que le langage HTML utilise des caractères spéciaux : ils sont part intégrante de la syntaxe du code. Le chevron « `<` », par exemple, est interprété comme du code et ne paraît donc pas à l'écran. Pour utiliser des caractères spéciaux dans du texte à afficher à l'écran, il faut les remplacer par des références spéciales, entourées des caractères « `&` » et « `;` ». Pour insérer « `<` » afin de symboliser « inférieur à », par exemple, il faut écrire « `<` ».

3.1.1.2. Forme d'une page HTML

Le fichier HTML doit se présenter sous la forme suivante :

- La 1ère ligne du fichier mentionne le type de document comme suit : `< !DOCTYPE html >`. Cette information n'est pas visible à l'écran.
- La 2ème ligne indique le langage de balisage utilisé, avec l'élément racine : `< html >`. Cet élément englobe tout le code, il doit être fermé à la fin du fichier, avec la balise `< /html >`. Ces indications n'apparaissent pas à l'écran.
- La 3ème ligne insère l'élément `< head >` : toutes les informations contenues à l'intérieur des balises `< head >` informations `< /head >` sont invisibles par l'internaute. Il s'agit d'informations à destination du navigateur : meta description, title et CSS interne, notamment.

- L'élément < body > est inséré en suivant : c'est cet élément qui englobe le contenu à afficher sur la page web. Texte, images, tableau HTML, vidéos, formulaires ou encore boutons : tous les éléments sont insérés entre les balises < body > contenu à afficher < /body >.

```
1  <!DOCTYPE html>
2
3  <html>
4
5  <head>
6
7      <title></title>
8
9  </head>
10
11 <body>
12
13 </body>
14
15 </html>
```

Figure 4 : Squelette d'une page web

3.1.2. CSS

CSS, ou "Cascading Style Sheets" (Feuilles de style en cascade en français), est un langage de programmation utilisé pour décrire la présentation et le style d'un document HTML. Il est largement utilisé dans le développement web pour contrôler l'apparence des pages web et permettre la séparation des préoccupations entre la structure (HTML) et la présentation (CSS) d'un site web.

Avec CSS, les développeurs web peuvent définir des règles de style qui déterminent la façon dont les éléments HTML sont affichés sur une page web. Cela inclut des propriétés comme la couleur, la police, la taille, la mise en page, les marges, les bordures et bien d'autres. CSS offre également la possibilité de créer des animations, des transitions et d'autres effets visuels pour améliorer l'expérience utilisateur sur un site web.

Une des principales caractéristiques de CSS est sa capacité à appliquer des styles en cascade, ce qui signifie que les règles de style peuvent être définies à différents niveaux (tel que le style inline, le style interne ou le style externe) et se combiner pour déterminer le style final d'un élément. Cela permet une gestion efficace et modulaire du style d'un site web.

En somme, CSS est un langage de programmation utilisé pour définir la présentation et le style des pages web, offrant ainsi un contrôle précis sur l'apparence des éléments HTML et permettant de créer des designs attractifs et modernes pour les sites web.

Comme pour HTML, CSS a commencé petit et grandit au fil des versions. La première version a vu le jour vers 1996. Il s'agissait de CSS1, suivie de CSS2 qui était la plus populaire jusqu'à l'apparition de HTML5 qui a intégré de nouvelles fonctionnalités qui ont emmené à développer la version 3 de CSS (dite CSS3) qui est le standard le plus utilisé actuellement. On utilise même le terme HTML5/CSS3 pour désigner les deux technologies qui vont alors ensemble.

CSS est un langage coté client, c'est à dire que sa syntaxe est comprise par le navigateur qui l'exécute avec le HTML. Ceci peut conduire à des problèmes de compatibilité, puisque parfois, les navigateurs ne comprennent pas toujours certains codes CSS de la même manière. Heureusement, ce sont des cas rares qui disparaissent petit à petit.

- **Framework CSS**

Un Framework CSS (Cascading Style Sheets) est une bibliothèque préparée de code CSS qui offre une collection de styles et d'outils normalisés pour simplifier le développement web. Les Framework CSS sont conçus pour simplifier le processus de création de pages web cohérentes, réactives et esthétiquement attrayantes en fournissant des classes CSS prêtes à l'emploi et des composants pouvant être facilement utilisés dans différents projets web.

En d'autres termes, un Framework CSS est un ensemble de fichiers CSS pré-écrits et organisés qui fournissent un ensemble de styles, de mises en page, de composants et d'utilitaires pour faciliter la conception et la mise en page d'un site web. Les développeurs web peuvent utiliser ces Framework pour accélérer leur processus de développement en réutilisant du code CSS existant, en adoptant des conventions de codage cohérentes et en bénéficiant de fonctionnalités prêtes à l'emploi pour créer des sites web rapidement et efficacement. Les Framework CSS sont généralement conçus pour être flexibles, réactifs et personnalisables, ce qui permet aux développeurs de les adapter aux besoins spécifiques de leurs projets.

Voici quelque exemple de Framework CSS :

- Bootstrap
- Foundation
- Materialize

- Skeleton
- Tailwind
- UIKit
- Bulma
- Semantic UI
- Animate css
- Material UI

3.1.3. JavaScript

Jusqu'ici nous avons vu les langages HTML et CSS. Le premier est un langage de description de pages Web et se base sur les balises et le deuxième est un langage de style qui repose sur des déclarations simples sous la forme de directives.

JavaScript quant à lui, est bel et bien un langage de programmation. Pour être plus précis, il s'agit d'un langage de script mais repose sur les mêmes principes qu'un langage de programmation évolué. D'ailleurs il est inspiré des fameux langages C et Java (et d'autres langages peu connus).

Les fonctions JavaScript peuvent permettre d'améliorer l'expérience utilisateur d'un site web, de la mise à jour des flux de médias sociaux à l'affichage d'animations et de cartes interactives. En tant que langage de script côté client, c'est l'une des principales technologies du web. Lors de la navigation sur Internet, à tout moment vous pouvez par exemple voir un carrousel d'images, un menu déroulant « Cliquer pour afficher » ou le changement dynamique de la couleur des éléments d'une page web. Tout cela est possible grâce à JavaScript.

3.1.3.1. Utilités du JavaScript

Auparavant, les pages web étaient statiques, similaires aux pages d'un livre. Une page statique affichait principalement des informations dans une mise en page fixe et ne présentait pas toutes les fonctionnalités d'un site web moderne. JavaScript est apparu comme une technologie côté navigateur conçue pour dynamiser les applications web. Grâce à JavaScript, les navigateurs pouvaient répondre aux interactions des utilisateurs et modifier la mise en page du contenu sur la page web.

À mesure que le langage a évolué, les développeurs JavaScript ont établi des bibliothèques, des Framework et des pratiques de programmation et ont commencé à utiliser

JavaScript en dehors des navigateurs web. Aujourd'hui, vous pouvez utiliser JavaScript pour le développement côté client et côté serveur.

3.1.3.2.1. Fonctionnement du JavaScript

JavaScript est globalement catégorisé comme un langage de script ou langage interprété. Le code JavaScript est interprété, c'est-à-dire qu'il est directement transposé en un code de langage machine sous-jacent par un moteur JavaScript. Pour les autres langages de programmation, un compilateur compile l'intégralité du code en un code machine lors d'une étape distincte. Ainsi, tous les langages de script sont des langages de programmation, mais tous les langages de programmation ne sont pas des langages de script.

3.1.3.3. Moteur JavaScript

Un moteur JavaScript est un programme informatique qui exécute du code JavaScript. Les premiers moteurs JavaScript n'étaient que de simples interprètes, alors que tous les moteurs modernes utilisent la compilation à la volée ou à l'exécution pour améliorer leurs performances.

3.1.3.4. JavaScript côté client

JavaScript côté client fait référence à la manière dont JavaScript fonctionne sur votre navigateur. Dans le cas présent, le moteur JavaScript se trouve dans le code du navigateur. Les principaux navigateurs web sont équipés de leur propre moteur JavaScript intégré.

Les développeurs d'applications web rédigent le code JavaScript avec différentes fonctions associées à différents événements, comme le clic de souris ou le survol de souris. Ces fonctions apportent des modifications au HTML et au CSS.

Voici un aperçu du fonctionnement de JavaScript côté client :

1. Le navigateur charge une page web lorsque vous la consultez.
2. Lors du chargement, le navigateur transforme la page et tous ses éléments, comme les boutons, les étiquettes et les listes déroulantes, en une structure de données appelée Document Object Model (DOM, modèle d'objets de document).
3. Le moteur JavaScript du navigateur convertit le code JavaScript en bytecode. Ce code joue le rôle d'intermédiaire entre la syntaxe JavaScript et la machine.

4. Plusieurs éléments, comme le clic d'une souris sur un bouton, déclenchent l'exécution du bloc de code JavaScript associé. Le moteur interprète ensuite le bytecode et apporte les modifications au DOM.
5. Le navigateur affiche le nouveau DOM.

3.1.3.5.JavaScript côté serveur

JavaScript côté serveur fait référence à l'utilisation du langage de codage dans une logique de serveur dorsal. Dans le cas présent, le moteur JavaScript se trouve directement sur le serveur. Une fonction JavaScript côté serveur peut accéder à la base de données, effectuer des opérations logiques différentes et réagir à plusieurs événements déclenchés par le système d'exploitation du serveur. L'avantage principal de la rédaction de scripts côté serveur est que vous pouvez grandement personnaliser la réponse du site web selon vos exigences, vos droits d'accès et les demandes d'informations provenant du site web.

3.1.3.6.Framework JavaScript

Un Framework JavaScript est une bibliothèque ou un ensemble d'outils pré-écrits qui fournissent une structure et des directives pour construire des applications web en utilisant le langage de programmation JavaScript. Les Framework JavaScript simplifient le processus de développement en fournissant des fonctions, des classes et des API prêtes à l'emploi qui peuvent être utilisées pour gérer des tâches courantes, telles que la manipulation du DOM, la liaison de données, le routage et les requêtes AJAX, entre autres.

Il existe de nombreux Framework JavaScript disponibles, chacun avec ses propres fonctionnalités uniques et ses cas d'utilisation. Voici quelques exemples de Framework JavaScript populaires :

- **React** : Développé par Facebook, React est une bibliothèque d'interface utilisateur largement utilisée pour construire des interfaces utilisateur dans les applications web. Il utilise une architecture basée sur les composants et fournit un DOM virtuel pour un rendu efficace des éléments d'interface utilisateur.
- **Angular** : Développé par Google, Angular est un Framework complet pour les applications web qui suit une approche déclarative pour construire des applications web dynamiques. Il utilise un ensemble de composants, de directives, de services et de modules pour créer des applications web complexes.

- **Vue.js** : Un Framework progressif pour la construction d'interfaces utilisateur, Vue.js permet aux développeurs d'adopter progressivement ses fonctionnalités au besoin. Il offre une architecture flexible et évolutive pour la construction d'applications web modernes.
- **Ember.js** : Un Framework avec des opinions pour la construction d'applications web ambitieuses, Ember.js suit l'approche "convention plutôt que configuration". Il fournit un ensemble de conventions et de meilleures pratiques pour structurer des applications web complexes.
- **Backbone.js** : Un Framework léger pour la construction d'applications monopages, Backbone.js fournit un ensemble minimaliste d'outils pour gérer les événements, les modèles, les vues et les collections. Il est souvent utilisé en combinaison avec d'autres bibliothèques pour construire des applications web complexes.

Ce ne sont que quelques exemples parmi les nombreux Framework JavaScript disponibles. Le choix d'un Framework dépend des besoins spécifiques de votre application web, de votre familiarité avec le Framework et des préférences de votre équipe de développement. Il est important de bien se renseigner et de comprendre les fonctionnalités, les avantages et les limitations des différents Framework avant d'en choisir un pour votre projet.

3.2. UML (Unified Modeling Language)

UML, ou Unified Modeling Language, est un langage de modélisation graphique utilisé pour visualiser, spécifier, concevoir et documenter les systèmes logiciels. Il fournit une notation standardisée pour représenter visuellement les différentes parties d'un système, telles que les classes, les objets, les relations, les processus et les flux de données.

UML est largement utilisé dans le domaine du développement logiciel, notamment pour la modélisation des systèmes orientés objet. Il offre une représentation visuelle abstraite qui facilite la compréhension et la communication entre les développeurs, les concepteurs et les parties prenantes.

Le langage UML comprend plusieurs types de diagrammes, tels que le diagramme de classes, le diagramme d'objets, le diagramme de séquence, le diagramme d'activité, le diagramme de composants, le diagramme de déploiement, et bien d'autres. Chaque type de diagramme représente un aspect spécifique du système logiciel et permet de visualiser ses différentes dimensions et interactions.

Les diagrammes utilisés dans le projet sont :

3.2.1. Diagramme de cas d'utilisation

Il permet de décrire les fonctionnalités du système du point de vue des utilisateurs (acteurs) et des scénarios d'utilisation. Il peut être utilisé pour identifier les principales fonctionnalités de l'application web et les interactions entre les utilisateurs et le système.

Dans notre cas, ci-dessous le diagramme de cas d'utilisation utilisé :

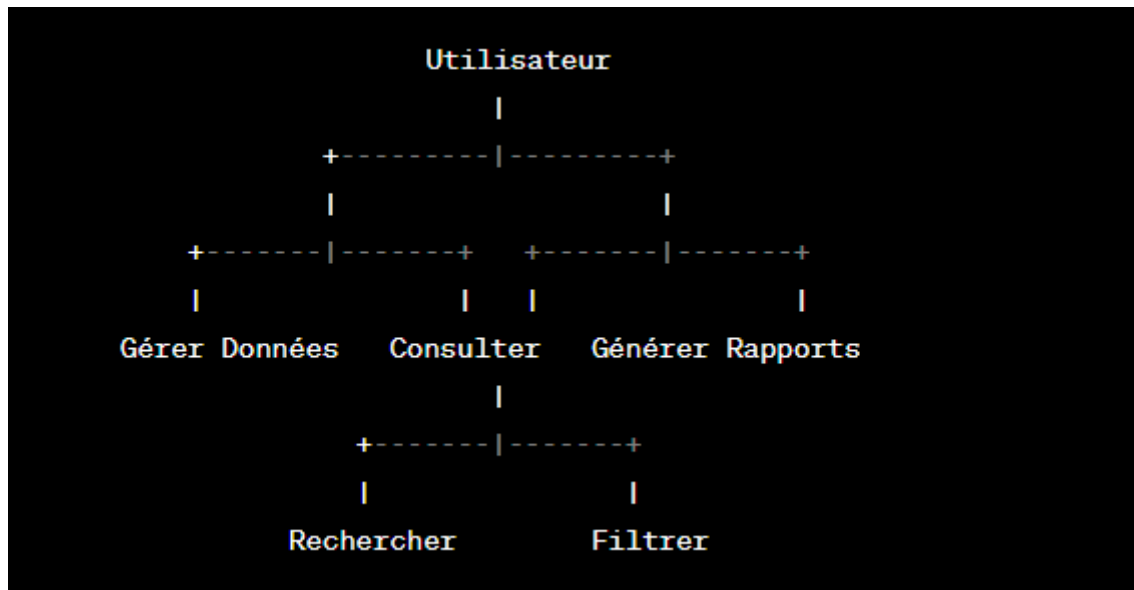


Figure 5 : Diagramme de cas d'utilisation

L'acteur principal est l'"Utilisateur" qui interagit avec l'application de gestion de données. Le diagramme de cas d'utilisation met en évidence les principales fonctionnalités offertes par l'application.

"Gérer Données" : Permet à l'utilisateur d'ajouter, modifier ou supprimer des données dans l'application.

"Consulter" : Permet à l'utilisateur de visualiser les données existantes dans l'application.

"Générer Rapports" : Permet à l'utilisateur de générer des rapports basés sur les données disponibles.

"Rechercher" : Permet à l'utilisateur de rechercher des données spécifiques dans l'application.

"Filtrer" : Permet à l'utilisateur d'appliquer des filtres pour affiner les résultats lors de la consultation ou de la recherche de données.

Ce diagramme de cas d'utilisation fournit une vue globale des fonctionnalités principales de l'application de gestion de données et des interactions entre l'utilisateur et le système. Il peut servir de point de départ pour spécifier plus en détail chaque cas d'utilisation et pour définir les scénarios d'utilisation spécifiques.

3.2.2. Diagramme de classe

Il permet de représenter la structure des classes et des objets dans le système. Pour une application web, cela peut inclure les modèles de données, les contrôleurs, les vues, les services, les composants d'interface utilisateur, etc. Ce diagramme aide à visualiser les relations entre les différentes classes et à comprendre la structure du système.

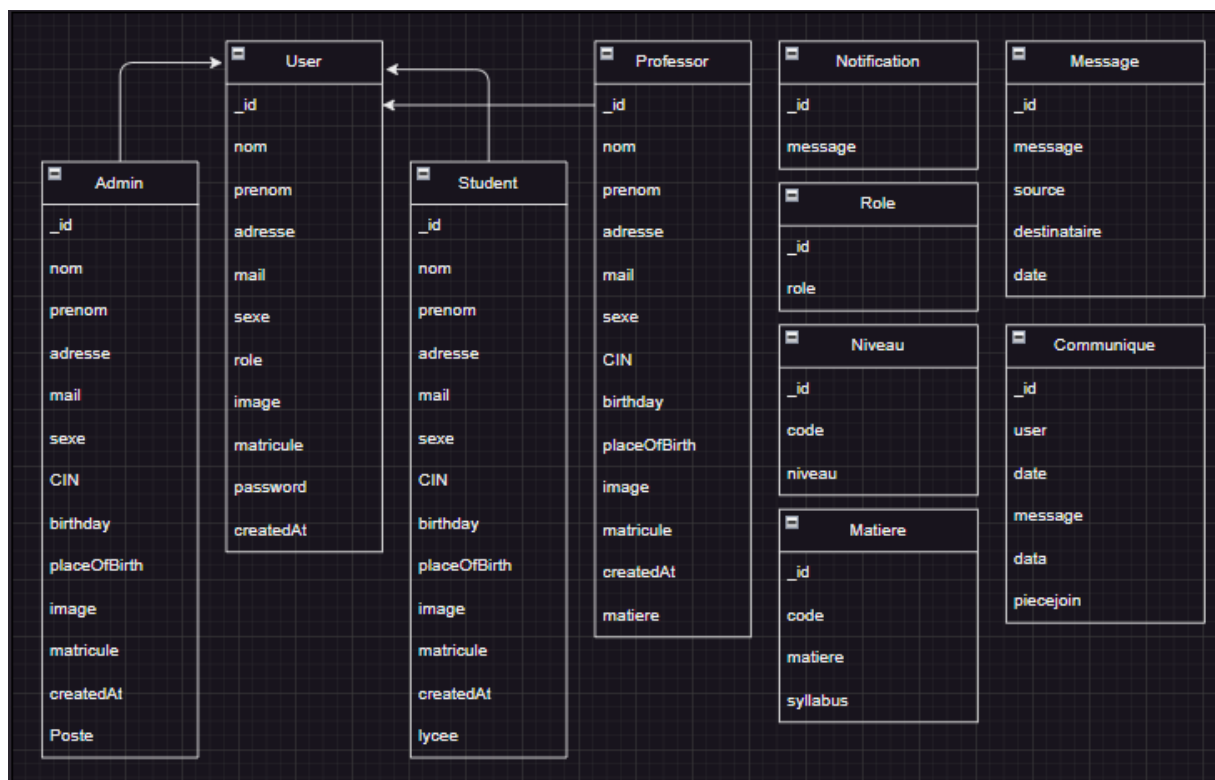


Figure 6 : Diagramme de classe

La figure ci-dessous nous montre qu'il y a dix modèles en tout pour l'application. Chaque modèle possède leurs propres attributs et un identifiant unique.

3.2.3. Diagramme de composants

Il montre les différents composants logiciels de l'application web et les dépendances entre eux. Cela peut inclure les bibliothèques, les Framework, les services externes, les API, etc.

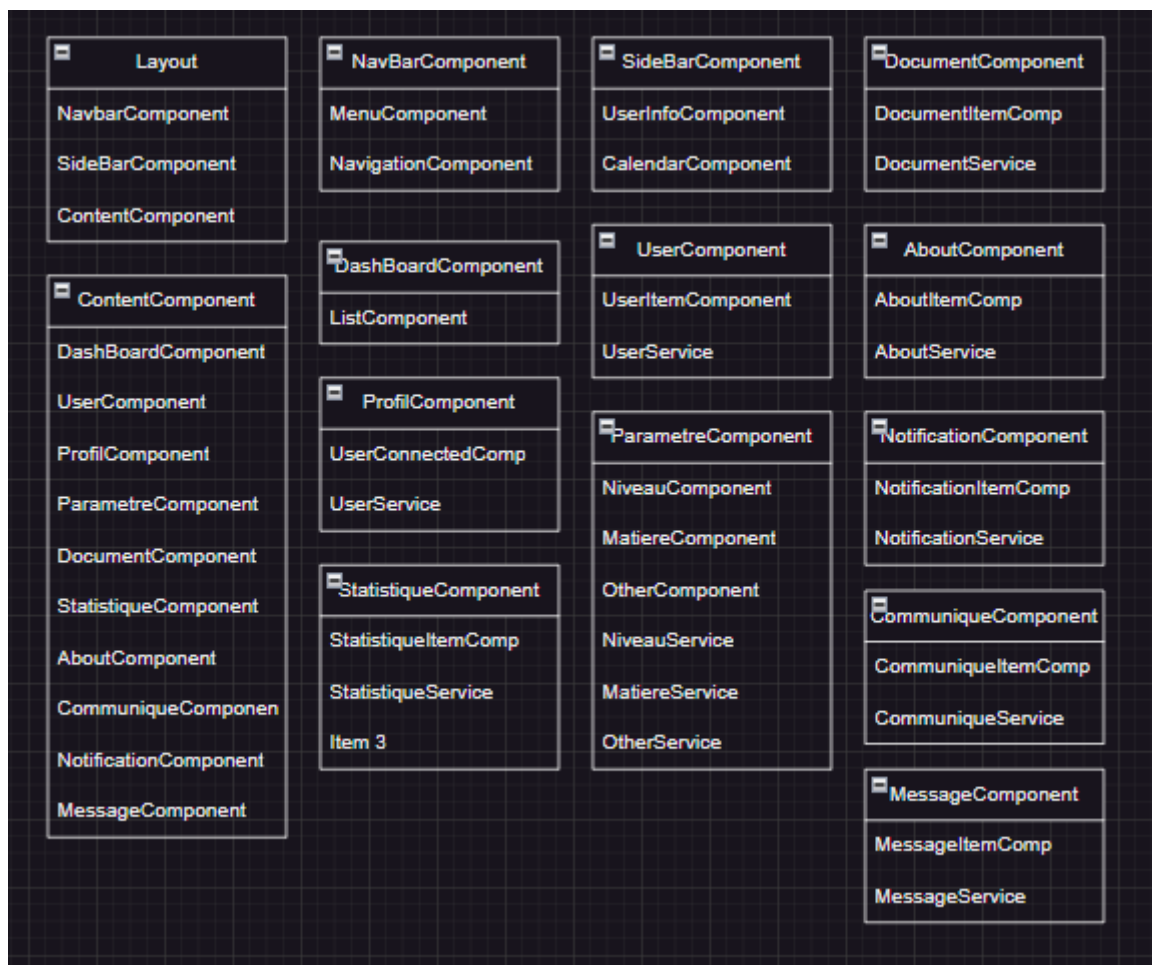


Figure 7 : Diagramme de cas d'utilisation

- « Layout »: Ce composant sert de conteneur principal pour l'application. Il englobe les composants « NavBarComponent », « SideBarComponent » et du contenu principal « ContentComponent ».

- « NavBarComponent » : Ce conteneur représente l'en-tête de l'application et inclus des sous composants tels que : « MenuComponent » et « NavigationComponent ».
- « MenuComponent » : ce composant contient les éléments de navigation.
- « NavigationComponent » : Ce conteneur contient les composants suivants : « MessageComponent », « NotificationComponent », « CommuniquerComponent » ainsi que « LogoutComponent ».
- « SideBarComponent » : Ce conteneur représente la barre latérale de l'application qui contient les éléments « UserInfoComponent » et « CalendarComponent ».
- « ContentComponent » : Ce conteneur représente la zone principale de contenu de l'application. Il peut inclure des sous-composants tels que « DashBoardComponent », « UserComponent », « ProfilComponent », « ParametreComponent », « DocumentComponent », « StatistiqueComponent », « AboutComponent », « CommuniquerComponent », « NotificationComponent » et « MessageComponent ».
- « DashBoardComponent » : Ce conteneur englobe le composant « ListComponent ».
- « UserComponent » : Ce conteneur contient le composant « UserInterfaceComp » pour afficher les détails. Il peut également interagir avec le service UserService pour récupérer les données des utilisateurs.
- « ProfilComponent » : Ce conteneur contient le composant « UserConnectedComp » pour afficher les détails. Il peut également interagir avec le service UserService pour récupérer les données des utilisateurs.
- « ParametreComponent » : Ce conteneur contient les composants « NiveauComponent », « MatiereComponent » ainsi que « OtherComponent ». Il peut également interagir avec les services NiveauService, MatiereService et OtherService pour récupérer les données nécessaires.
- « DocumentComponent » : Ce conteneur contient le composant « DocumentItemComp ». Il utilise le service DocumentService pour récupérer les données.
- « StatistiqueComponent » : Ce conteneur contient le composant « StatistiqueItemComp » et utilise le service StatistiqueService pour récupérer les données.
- « AboutComponent » : Ce conteneur contient le composant « AboutItemComp » et utilise le service AboutService pour récupérer les données.

- « NotificationComponent » : Ce conteneur contient le composant « NotificationItemComp » et utilise le service NotificationService pour récupérer les données.
- « MessageComponent » : Ce conteneur contient le composant « MessageItemComp » et utilise le service MessageService pour récupérer les données.
- « CommuniquerComponent » : Ce conteneur contient le composant « CommuniquerItemComp » et utilise le service CommuniquerService pour récupérer les données.

Partie 2 : Conception Pratique

CHAPITRE IV : ETUDE PRATIQUE DU PROJET

4.1. Outils utilisés

Pour créer une application web, nous aurons besoin de plusieurs outils pour différents aspects du développement. A savoir, un environnement de développement intégré (IDE), un langage de programmation, un système de gestion de version et des Framework.

4.1.1. Environnement de développement intégré (IDE)

Un IDE (Environnement de Développement Intégré) est un logiciel qui offre un ensemble d'outils pour faciliter le développement de logiciels, y compris les applications web. Un IDE regroupe généralement plusieurs fonctionnalités essentielles pour le développement, telles que l'édition de code, la gestion de fichiers, le débogage, la compilation ou l'exécution de code, la gestion de version, la complétion automatique de code, et bien plus encore. Les IDE sont conçus pour aider les développeurs à être plus productifs et à créer des applications de manière plus efficace.

Voici quelques exemples d'IDE populaires utilisés pour le développement d'applications web :

- **Visual Studio Code** : Il s'agit d'un IDE gratuit et open source développée par Microsoft, qui offre une interface utilisateur conviviale, une grande variété d'extensions et une intégration étroite avec de nombreux langages de programmation couramment utilisés pour le développement web.
- **Sublime Text** : C'est un éditeur de texte avancé qui offre de nombreuses fonctionnalités pour le développement web, comme la coloration syntaxique, la complétion automatique de code et la gestion de projet.
- **JetBrains WebStorm** : C'est un IDE payant spécifiquement conçu pour le développement web, offrant une gamme complète d'outils pour le développement front-end et back-end, ainsi que pour le débogage et le test d'applications web.
- **Eclipse** : C'est un IDE populaire utilisé principalement pour le développement Java, mais il offre également des fonctionnalités pour d'autres langages de programmation couramment utilisés dans le développement web.

- **Atom** : C'est un éditeur de texte open source conçu pour les développeurs web, offrant une interface personnalisable et de nombreuses extensions pour faciliter le développement web.

Il existe de nombreux autres IDE disponibles, et le choix d'un IDE dépend généralement des préférences personnelles du développeur, du langage de programmation utilisé et des besoins spécifiques du projet.

Dans notre cas, on s'intéresse beaucoup à Visual Studio Code pour rédiger et compiler notre code.

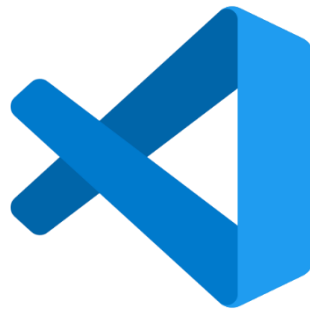


Figure 8 : Icône de Visual Studio Code

4.1.2. Téléchargement et installation de Visual Studio Code

a) Téléchargement

Pour télécharger Visual studio code, il suffit de se rendre sur cette page : <https://code.visualstudio.com/download>

Puis téléchargez la version pour Windows User Installer en fonction de votre processeur : 64-bit, 32-bit ou ARM.

b) Installation

Une fois le téléchargement fini, suivez les instructions suivante :

- Exécutez l'installateur.
- Acceptez la licence.

- Choisissez le répertoire d'installation.
- Choisissez les fonctionnalités supplémentaires à ajouter.
- Cliquez sur « install » pour finaliser l'installation.

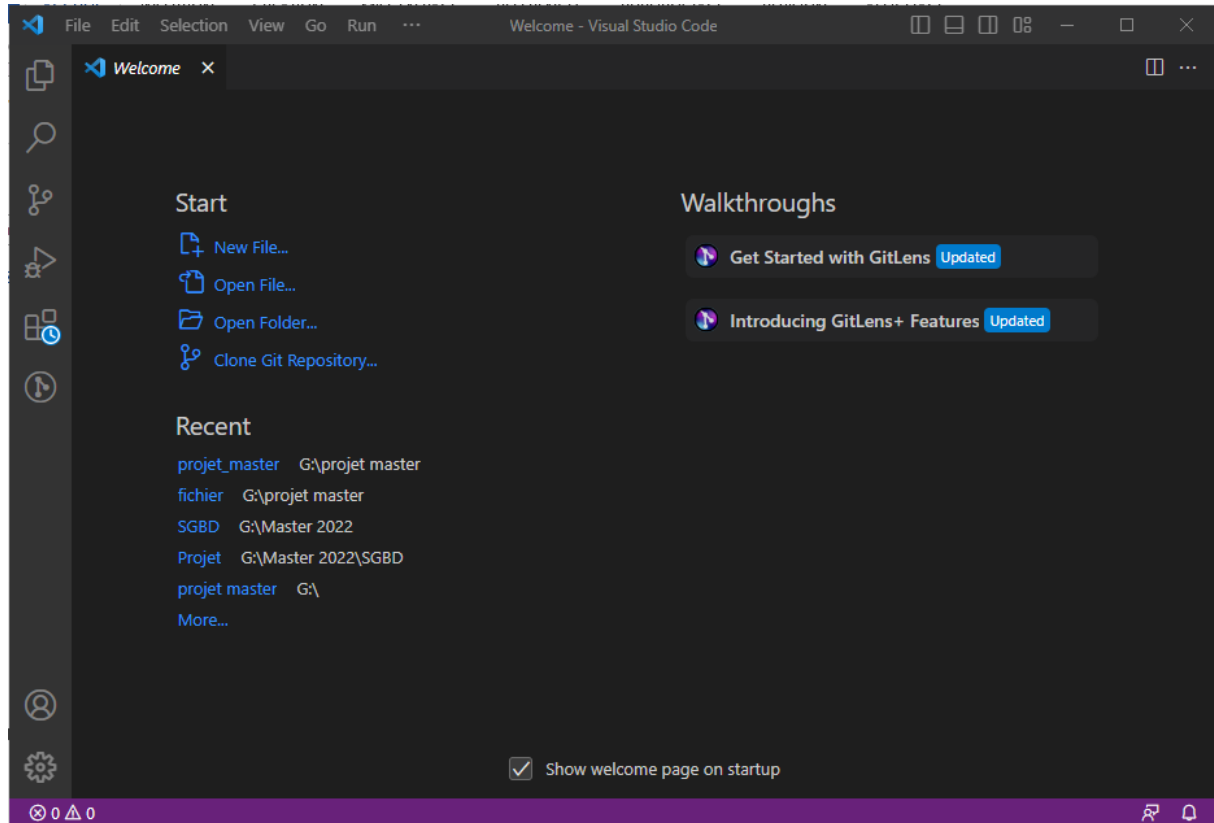


Figure 9 : IDE Visual Studio Code

4.1.3. Pourquoi choisir Visual Studio Code

Visual Studio Code (VS Code) est un éditeur de code source gratuit et open-source développé par Microsoft. Il est devenu populaire en raison de plusieurs facteurs qui le distinguent des autres éditeurs de code. Les raisons pour lesquelles Visual Studio Code est souvent préféré :

Léger et rapide : VS Code est conçu pour être léger et réactif, ce qui signifie qu'il se lance rapidement et offre des performances fluides, même lorsqu'il est utilisé sur des machines moins puissantes. Il est optimisé pour une utilisation efficace des ressources système.

Large compatibilité : VS Code est compatible avec de nombreux langages de programmation courants, tels que JavaScript, Python, Java, C++, HTML, CSS, etc. Il prend

également en charge une grande variété de Framework, de bibliothèques et d'outils de développement.

Extension et personnalisation : VS Code offre une vaste gamme d'extensions qui permettent aux développeurs de personnaliser leur environnement de développement en fonction de leurs besoins spécifiques. Ces extensions ajoutent des fonctionnalités supplémentaires telles que la coloration syntaxique, les linters, les débogueurs, les outils de gestion de version, etc.

Intégration avec Git : VS Code offre une intégration transparente avec Git, le système de contrôle de version le plus couramment utilisé. Cela facilite le suivi des modifications du code, la gestion des branches et des conflits, et la collaboration avec d'autres développeurs.

Débogage puissant : VS Code dispose d'un système de débogage intégré qui facilite le processus de recherche et de résolution des erreurs dans le code. Les développeurs peuvent placer des points d'arrêt, inspecter les variables, suivre l'exécution pas à pas et analyser le flux d'exécution pour identifier les problèmes plus rapidement.

Terminal intégré : VS Code dispose d'un terminal intégré qui permet aux développeurs d'exécuter des commandes directement à partir de l'éditeur sans avoir besoin d'ouvrir une fenêtre de terminal séparée. Cela facilite l'exécution de scripts, de commandes de construction et d'autres tâches liées au développement.

Prise en charge de la communauté : VS Code bénéficie d'une communauté active et en croissance, ce qui se traduit par un large éventail de ressources, de tutoriels, de plugins et de thèmes disponibles en ligne. Les développeurs peuvent trouver facilement de l'aide et des solutions à leurs problèmes grâce à la communauté.

Ces caractéristiques et avantages font de Visual Studio Code un choix populaire parmi les développeurs et les programmeurs, car il leur offre un environnement de développement efficace, polyvalent et personnalisable pour travailler sur leurs projets.

4.1.4. Langage de programmation

La plateforme est une application MERN, c'est-à-dire que cette dernière est entièrement basée sur le langage de programmation JavaScript. Nous allons alors utiliser ces différents Framework, à savoir, React pour le côté client (interface utilisateur) et Node.js pour le côté serveur (back-end).

4.2. Conception d'une application MERN

MERN est un acronyme qui fait référence à un ensemble de technologies utilisées pour développer des applications web modernes. Chaque lettre représente un outil ou un Framework spécifique :

M : MongoDB : une base de données NoSQL (Not Only SQL) orientée documents. Elle est basée sur un modèle de données flexible et est utilisée pour stocker les données dans des documents au format JSON.

E : Express : un Framework web pour Node.js, qui permet de créer des applications web côté serveur. Express facilite la gestion des requêtes HTTP, des routes, des cookies et des sessions, ainsi que la création d'APIs RESTful.

R : React : une bibliothèque JavaScript pour la création d'interfaces utilisateur interactives. React permet de créer des composants réutilisables pour construire des interfaces utilisateur modernes et réactives.

N : Node.js : un environnement d'exécution JavaScript côté serveur basé sur le moteur V8 de Google Chrome. Node.js permet d'exécuter du code JavaScript sur le serveur, ce qui permet de créer des applications web côté serveur et de gérer les requêtes et les réponses.

En combinant ces quatre technologies, MERN permet de développer des applications web à architecture complète (full-stack) en JavaScript. MongoDB est utilisé comme base de données, Express comme Framework web côté serveur, React comme bibliothèque pour l'interface utilisateur côté client, et Node.js pour exécuter le code JavaScript côté serveur. MERN est donc une pile de technologies complète pour le développement d'applications web modernes et performantes.

4.2.1. React

React.js, communément appelé simplement React, est une bibliothèque JavaScript utilisée pour construire des interfaces utilisateur. Chaque application web React est composée de composants réutilisables qui constituent des parties de l'interface utilisateur, nous pouvons avoir un composant distinct pour notre barre de navigation, un pour le pied de page, un autre pour le contenu principal, et ainsi de suite.

Le fait de disposer de ces composants réutilisables facilite le développement car nous n'avons pas à répéter le code récurrent. Il nous suffit de créer sa logique et d'importer le composant dans n'importe quelle partie du code où il est nécessaire.

React est également une application à page unique. Ainsi, au lieu d'envoyer une requête au serveur à chaque fois qu'une nouvelle page doit être rendue, le contenu de la page est chargé directement à partir des composants React. Cela conduit à un rendu plus rapide sans rechargement de la page.

Dans la plupart des cas, la syntaxe utilisée pour construire des applications React est appelée JSX (JavaScript XML), qui est une extension syntaxique de JavaScript. Cela nous permet de combiner la logique JavaScript et la logique de l'interface utilisateur d'une manière unique. Avec JSX, nous éliminons le besoin d'interagir avec le DOM en utilisant des méthodes comme `document.getElementById`, `querySelector`, et d'autres méthodes de manipulation du DOM.

Bien que l'utilisation de JSX ne soit pas obligatoire, elle facilite le développement des applications React.

a) Pourquoi choisir React

De nombreux développeurs et organisations ont choisi React plutôt que d'autres bibliothèques ou Framework.

Facile à apprendre : React est facile à apprendre et à comprendre tant que vous avez une bonne maîtrise des prérequis. React dispose d'une solide documentation et de nombreuses ressources gratuites créées par d'autres développeurs en ligne via la communauté très active de React.

Composants réutilisables : Chaque composant de React possède sa propre logique qui peut être réutilisée partout dans l'application. Cela réduit le besoin de réécrire le même morceau de code plusieurs fois.

Opportunités d'emploi : Un plus grand pourcentage d'opportunités de développement web front-end en ce moment ont React comme l'une des compétences requises. Le fait de comprendre le fonctionnement de React et de pouvoir travailler avec lui augmente donc vos chances de décrocher un emploi.

Performances améliorées : Grâce au DOM virtuel de React, le rendu des pages peut se faire plus rapidement. En utilisant une bibliothèque de routage comme React Router, vous aurez différentes pages rendues sans aucun rechargement de page.

Largement extensible : React est une bibliothèque qui assure uniquement le rendu de l'interface utilisateur de notre application. C'est au développeur de choisir les outils avec

lesquels il souhaite travailler, comme les bibliothèques de rendu de différentes pages, les bibliothèques de conception, etc.

b) Qui utilise React

React a été utilisé par de nombreux ingénieurs front-end dans des start-ups et des entreprises établies comme Facebook, Netflix, Instagram, Yahoo, Uber, The New York Times, et plus encore.

Bien que toutes les entreprises citées ci-dessus n'aient pas construit l'ensemble de leur produit à l'aide de React, certaines de leurs pages ont été construites avec React. Cela s'explique par les hautes performances, la facilité d'utilisation et l'évolutivité de React.

c) Caractéristiques de React

React possède une pléthore de fonctionnalités étonnantes qui continuent à en faire une option populaire pour les développeurs.

Quelques-unes des principales fonctionnalités de React sont énoncées ci-dessous:

- **JSX** : Il s'agit d'une extension syntaxique JavaScript qui étend les fonctionnalités de l'ES6 (ECMAScript 2015). Cela nous permet de combiner la logique et le balisage JavaScript dans un composant.
- **DOM virtuel** : Il s'agit d'une copie de l'objet DOM qui met d'abord à jour et rend à nouveau nos pages lorsque des modifications sont apportées ; il compare ensuite l'état actuel avec le DOM original pour le maintenir en phase avec les modifications. Cela permet un rendu plus rapide des pages.
- **Composants** : Les applications React sont composées de différents composants réutilisables qui ont leur propre logique et interface utilisateur respective. Cela le rend efficace pour la mise à l'échelle des applications et le maintien de performances élevées, car vous ne dupliquez pas le code aussi souvent que dans d'autres Framework.

d) Avantages de l'utilisation de React

Les avantages de l'utilisation de React :

- React est facile à apprendre et à comprendre.

- React dispose d'une communauté très active où vous pouvez contribuer et obtenir de l'aide si nécessaire.
- Il existe de nombreuses opportunités d'emploi pour les développeurs React.
- React s'accompagne d'une augmentation des performances des applications.


e) Inconvénients de l'utilisation de React

Les inconvénients de l'utilisation de React :

- Les débutants qui n'ont pas une solide compréhension de JavaScript (en particulier ES6) peuvent avoir du mal à comprendre React.
- React est dépourvu de certaines fonctionnalités courantes comme la gestion d'un état unique et le routage ; vous devrez installer et apprendre à utiliser des bibliothèques externes pour les obtenir.

f) Installation de React

Avant d'installer React, il nous faut s'assurer que Node.js est installé sur notre ordinateur. Pour vérifier il suffit d'ouvrir un terminal et exécuter le code suivant :



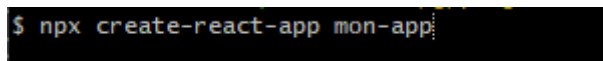
```
$ node -v
v16.14.0
```

Figure 10 : Commande d'affichage de version de Node.js

Cette commande nous affichera la version de Node.js s'il est déjà installé.

Ensuite, nous allons exécuter une autre commande pour commencer à installer React.

Tout d'abord, créez un dossier ou naviguez jusqu'à l'emplacement où vous voulez que l'application React soit installée, puis exécutez la commande ci-dessous dans votre terminal :



```
$ npx create-react-app mon-app
```

Figure 11 : Commande pour créer une application React

La commande ci-dessus va installer React dans un dossier appelé mon-app.

Une fois l'installation terminée, ouvrez votre dossier React nouvellement installé dans l'éditeur de code de votre choix. Nous utiliserons Visual Studio Code dans ce projet. Visual Studio Code est livré avec un terminal intégré. Si vous comptez utiliser un autre terminal comme Git Bash ou CMD, il vous suffit de naviguer dans le répertoire de votre application React et d'exécuter la commande ci-dessous :

```
$ npm start
```

Figure 12 : *Commande pour lancer l'application React*

Cette commande met en route votre serveur de développement. Après un petit moment, vous devriez avoir cette page ci-dessous rendue sur localhost:3000 dans votre navigateur :

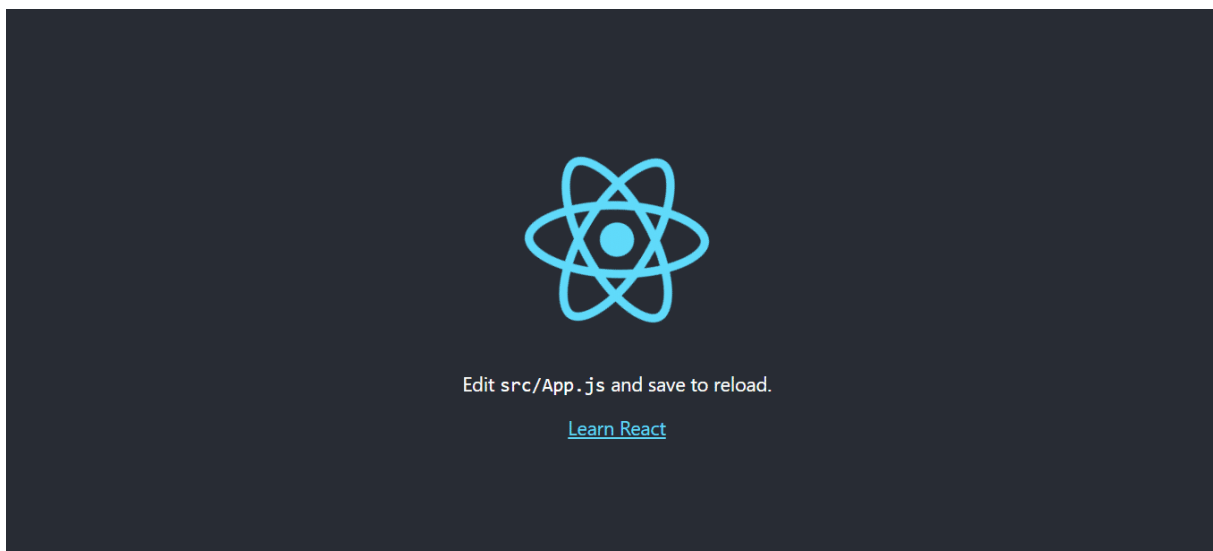


Figure 13 : *Première page de démarrage d'une nouvelle application React*

4.2.2. Node.js

a) C'est quoi Node.js

Node.js est un environnement d'exécution JavaScript asynchrone non bloquant. Un tel environnement procure au JavaScript la possibilité de s'exécuter sur le serveur (comme c'est le cas pour PHP, Python ou Perl...) afin de créer des applications back end ou natives (comme les applications mobiles).

Node.js permet de donner vie à des applications dans des environnements concurrentiels avec une forte montée en charge tout en maintenant de bonnes performances. Ces particularités lui ont permis d'être au cœur de nombreux sites et applications Web comme Netflix, Uber, Paypal ou LinkedIn.

b) Caractéristiques de Node.js

Node.js a connu une croissance rapide au cours des dernières années. Cela est dû à la vaste liste de fonctionnalités qu'il offre :

- Facile – Easy-Node.js est assez facile à prendre en main. C'est un choix incontournable pour les débutants en développement web. Grâce à de nombreux tutoriels et à une vaste communauté, il est très facile de se lancer.
- Évolutif – Il offre une grande évolutivité aux applications. Node.js, étant single-thread, est capable de gérer un grand nombre de connexions simultanées avec un débit élevé.
- Vitesse – L'exécution non bloquante des threads rend Node.js encore plus rapide et plus efficace.
- Package – Un vaste ensemble de packages Node.js open source est disponible et peut simplifier votre travail. Aujourd'hui, il y a plus d'un million de paquets dans l'écosystème NPM.
- Back end solide – Node.js est écrit en C et C++, ce qui le rend rapide et ajoute des fonctionnalités comme le support réseau.
- Multiplateforme – La prise en charge multiplateforme vous permet de créer des sites web SaaS, des applications de bureau et même des applications mobiles, le tout en utilisant Node.js.
- Maintenable – Node.js est un choix facile pour les développeurs, car le front end et le back end peuvent être gérés avec JavaScript comme un seul langage.

c) Avantage de l'utilisation de Node.js

On peut résumer les avantages de l'environnement Node.js dans les points suivants :

- Applications concurrentielles et rapides : Tirant profit de son asynchronicité, les applications faites en Node.js sont plus rapides et gèrent mieux la montée en charge.

- Adapté aux applications temps réel : Cela rejoint le premier avantage. En effet, Node.js est connu pour son adaptabilité aux applications temps réel qui nécessitent un temps de réaction minimum.
- Modules facilement accessibles via NPM : Le gestionnaire de paquet npm de Node.js donne la possibilité de télécharger des modules utiles pour l'application en cours de développement.
- Un seul est unique langage pour le font end et le back end: Le point que la plupart des développeurs trouvent intéressant c'est le fait qu'on n'a besoin que d'un seul langage qui est JavaScript et avec lequel on peut monter une application entière englobant le font end et le back end (du fullstack).
- Portabilité et facilité de déploiement : Node.js est disponible pour divers systèmes d'exploitation, notamment, Windows, Linux et MacOS, et il est facile et rapide à déployer.
- Applications en single page : Avec Node.js, il est possible de créer une application qui s'exécute entièrement à travers une seule et unique page à travers des routes (ou endpoints) virtuelles.

d) Inconvénients de l'utilisation de Node.js

Comme inconvénients, Node.js nécessite de coder tous les aspects d'un serveur manuellement. En effet, en Node.js on est amené à programmer les différents aspects qui concernent un serveur Web, comme le serveur HTTP, le port d'écoute, la gestion des requêtes et des entêtes... Ces choses que l'on trouve généralement déjà disponibles dans des environnements qui exécutent PHP ou Perl par exemple. Cependant, il existe des Framework, notamment Express.js (que l'on va aussi voir dans ce cours) qui permettent d'abstraire toutes ses opérations rendant encore Node.js plus facile à utiliser.

e) Installation de Node.js

Pour installer Node.js sur Windows, il suffit de suivre les étapes suivantes :

- **Télécharger Windows Installer**

Tout d'abord, vous devez télécharger le fichier Windows Installer (.msi) depuis le site officiel de Node.js. Ce fichier d'installation MSI contient une collection de fichiers d'installation essentiels pour installer, mettre à jour ou modifier la version existante de Node.js.

Notamment, le programme d'installation contient également le gestionnaire de paquets Node.js (npm). Cela signifie que vous n'avez pas besoin d'installer npm séparément.

Lors du téléchargement, sélectionnez la version correcte en fonction de votre système d'exploitation. Par exemple, si vous utilisez un système d'exploitation 64 bits, téléchargez la version 64 bits, et si vous utilisez la version 32 bits, téléchargez la version 32 bits.

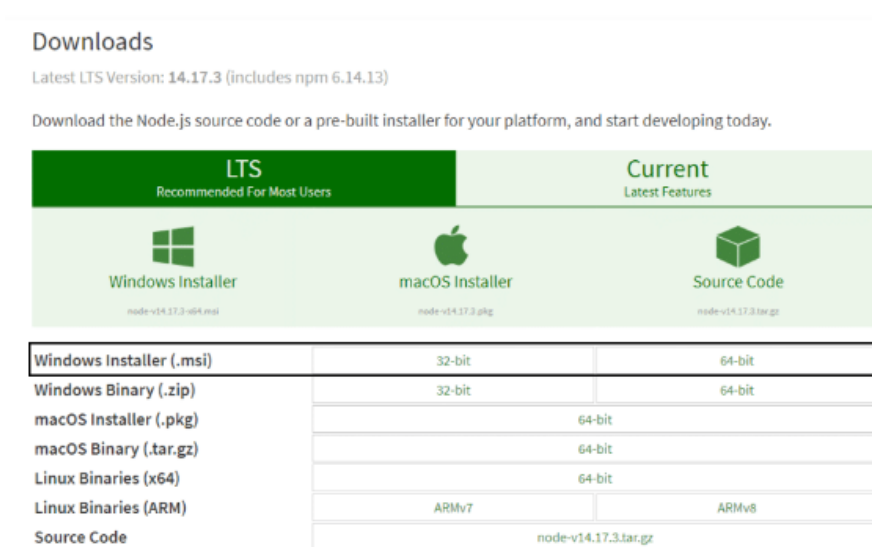


Figure 14 : Télécharger l'installateur de Node.js

- **Commencer le processus d'installation**

Une fois que vous avez ouvert et exécuté le fichier .msi, le processus d'installation commence. Mais vous devez définir quelques paramètres avant de lancer le processus d'installation.

Double-cliquez sur le fichier d'installation et exécutez-le. Le programme d'installation vous demandera d'accepter le contrat de licence de Node.js. Pour continuer, cochez la case « J'accepte » et cliquez sur Suivant :

Ensuite, sélectionnez la destination où vous voulez installer Node.js. Si vous ne voulez pas changer de répertoire, choisissez l'emplacement par défaut de Windows et cliquez à nouveau sur le bouton Suivant.

Node.js vous offre des options pour installer des outils pour les modules natifs. Si vous êtes intéressé par ces derniers, cliquez sur la case à cocher pour marquer vos préférences, ou cliquez sur Suivant pour continuer avec la valeur par défaut :


- **Lancer l'installation de Node.js sur Windows**

Cliquez sur le bouton Installer pour lancer le processus d'installation. Le système terminera l'installation en quelques secondes ou minutes et vous montrera un message de réussite. Cliquez sur le bouton Terminer pour fermer le programme d'installation de Node.js.

Le processus d'installation est donc terminé. Maintenant, vous devez vérifier si Node.js est installé avec succès ou non.

- **Vérifier l'installation de Node.js**

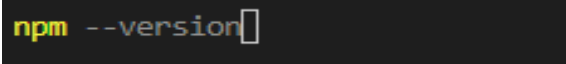
Pour vérifier l'installation et confirmer que la bonne version a été installée, ouvrez l'invite de commande votre PC et saisissez la commande suivante :



```
node --version
```

Figure 15 : Commande pour afficher la version de Node.js

Et pour vérifier la version de npm, lancez cette commande :



```
npm --version
```

Figure 16 : Commande pour afficher la version de npm

Si la version de Node.js et npm sont correctement installés, vous verrez le nom de la version dans l'invite CMD.

4.2.3. Express.js

a) C'est quoi Express.js

Express.js est un Framework web populaire pour Node.js, qui facilite le développement d'applications web et d'API. Il s'agit d'une couche d'abstraction légère construite au-dessus de Node.js, qui offre des fonctionnalités pour gérer les routes, les requêtes HTTP, les cookies, les sessions et bien plus encore.

Express.js est connu pour sa simplicité et sa flexibilité. Il permet de créer rapidement des serveurs web en fournissant des méthodes et des middlewares pour gérer les différentes étapes du traitement des requêtes HTTP. Il permet également de définir des routes pour répondre à des URL spécifiques et d'effectuer des opérations telles que la récupération de données à partir d'une base de données, le rendu de vues HTML dynamiques ou la création d'une API RESTful.

Le Framework Express.js est extrêmement populaire dans l'écosystème Node.js en raison de sa facilité d'utilisation et de sa grande communauté de développeurs. Il est utilisé par de nombreuses entreprises et applications web pour créer des serveurs robustes et performants.

b) Caractéristiques d'Express.js

Express.js possède plusieurs caractéristiques qui en font un Framework web populaire pour Node.js. Voici quelques-unes de ses principales caractéristiques :

- **Routing** : Express.js permet de définir des routes pour gérer les requêtes HTTP. Vous pouvez définir des routes pour des URL spécifiques et définir des actions à effectuer lorsque ces routes sont demandées. Cela facilite la gestion des différentes pages et fonctionnalités de votre application web.
- **Middleware** : Express.js utilise un système de middleware qui permet d'ajouter des fonctionnalités supplémentaires au flux de traitement des requêtes. Vous pouvez utiliser des middlewares pour gérer des tâches telles que l'authentification, la gestion des sessions, la journalisation, la compression des réponses, etc. Cela facilite l'ajout de fonctionnalités à votre application de manière modulaire.
- **Gestion des requêtes et des réponses** : Express.js facilite la gestion des requêtes et des réponses HTTP. Vous pouvez accéder aux données de la requête, comme les paramètres de l'URL, les en-têtes, les cookies, les données du formulaire, etc. De plus, vous pouvez

renvoyer des réponses avec différents types de contenu, tels que du texte, du JSON, des fichiers, etc.

- **Moteurs de Template :** Express.js offre la possibilité d'utiliser des moteurs de Template. Cela permet de générer des vues HTML dynamiques en utilisant des modèles réutilisables et des données provenant de votre application.
- **Gestion des erreurs :** Express.js fournit des mécanismes pour gérer les erreurs dans votre application. Vous pouvez définir des middlewares spécifiques pour gérer les erreurs, ainsi que des gestionnaires d'erreurs pour capturer et traiter les erreurs survenues pendant le traitement des requêtes.
- **Intégration de modules complémentaires :** Express.js s'intègre facilement avec de nombreux modules complémentaires de la communauté Node.js. Vous pouvez utiliser des modules pour gérer les sessions, les bases de données, l'authentification, la validation des données, etc. Cela vous permet d'étendre les fonctionnalités d'Express.js selon vos besoins spécifiques.
- **Simplicité et légèreté :** Express.js est apprécié pour sa simplicité et sa légèreté. Il offre une interface minimaliste et une syntaxe concise, ce qui facilite l'apprentissage et la mise en œuvre. Express.js se concentre sur les fonctionnalités essentielles pour la création d'applications web, ce qui le rend efficace et performant.

Ces caractéristiques font d'Express.js un choix populaire pour le développement d'applications web avec Node.js. Sa simplicité, sa flexibilité et sa grande communauté de développeurs en font un outil puissant pour la création de serveurs web et d'API.

c) Avantages de l'utilisation d'Express.js

Express.js présente plusieurs avantages qui en font un choix populaire pour le développement d'applications web avec Node.js :

- **Simplicité :** Express.js est connu pour sa simplicité et sa facilité d'apprentissage. Il offre une interface minimaliste et une syntaxe concise, ce qui permet aux développeurs de créer rapidement des applications web.
- **Flexibilité :** Express.js est un Framework très flexible qui permet aux développeurs de construire des applications web selon leurs besoins spécifiques. Il n'impose pas de structures ou de conventions rigides, ce qui permet une grande liberté de conception.

- **Middleware** : Express.js utilise le concept de middleware, qui permet d'ajouter des fonctionnalités supplémentaires au flux de traitement des requêtes. Cela permet d'ajouter des fonctionnalités telles que l'authentification, la gestion des sessions, la compression des réponses, la journalisation, etc.
- **Gestion des routes** : Express.js facilite la gestion des routes, c'est-à-dire la définition des URL et des actions à effectuer lorsque ces URL sont demandées. Il offre des méthodes simples pour définir des routes et effectuer des opérations telles que l'extraction de paramètres d'URL, la validation des données et la génération de réponses.
- **Grande communauté** : Express.js bénéficie d'une vaste communauté de développeurs et de contributeurs. Cela signifie qu'il existe de nombreuses ressources, tutoriels, modules complémentaires et exemples de code disponibles pour faciliter le développement avec Express.js.
- **Performances** : Express.js est connu pour sa légèreté et ses performances élevées. Il est conçu pour être rapide et efficace, ce qui en fait un choix adapté pour les applications web nécessitant une grande réactivité et des temps de réponse courts.
- **Intégration aisée** : Express.js s'intègre facilement avec d'autres modules et Framework de la communauté Node.js. Il est couramment utilisé avec des bases de données telles que MongoDB et des moteurs de Template.

Ces avantages font d'Express.js un choix populaire pour développer des applications web avec Node.js, en offrant à la fois une simplicité d'utilisation et une grande flexibilité pour répondre aux besoins spécifiques des développeurs.

d) Inconvénients de l'utilisation d'Express.js

Bien qu'Express.js soit un Framework populaire et largement utilisé, il présente également quelques inconvénients potentiels :

- **Minimalisme** : Bien que la simplicité soit considérée comme un avantage, certains développeurs peuvent trouver que le côté minimaliste d'Express.js limite les fonctionnalités intégrées. Cela signifie que vous devrez souvent utiliser des modules tiers pour des fonctionnalités avancées telles que l'authentification, la validation des données ou l'ORM (Object-Relational Mapping).
- **Courbe d'apprentissage** : Si vous êtes nouveau dans le développement web avec Node.js, Express.js peut présenter une courbe d'apprentissage initiale, surtout si vous

n'êtes pas familier avec la syntaxe et les concepts de Node.js. Comprendre le concept de middleware et la façon dont les routes sont gérées peut prendre un certain temps pour les débutants.

- **Structure non conventionnelle** : Express.js est très flexible et n'impose pas de structure ou de conventions strictes. Cela peut être un avantage pour certains développeurs, mais cela peut aussi conduire à des applications moins cohérentes et plus difficiles à maintenir si une structure organisationnelle solide n'est pas mise en place dès le départ.
- **Gestion des erreurs** : Express.js ne fournit pas de gestion des erreurs intégrée par défaut. Cela signifie que vous devrez gérer vous-même les erreurs dans votre application, ce qui peut être un défi, en particulier lorsqu'il s'agit de gérer les erreurs de manière uniforme dans toute l'application.
- **Performance et évolutivité** : Bien qu'Express.js soit généralement considéré comme performant, il peut ne pas être aussi performant que certains autres Framework web plus spécialisés. Pour des applications nécessitant une très haute performance ou une grande évolutivité, d'autres Framework tels que Fastify ou Koa peuvent être préférés.
- **Documentation dispersée** : Étant donné qu'Express.js est très populaire et qu'il existe de nombreux modules et extensions développés par la communauté, la documentation peut parfois être dispersée et varier en qualité. Il peut être nécessaire de consulter plusieurs sources pour trouver les informations dont vous avez besoin.

Malgré ces inconvénients, Express.js reste un choix solide pour le développement web avec Node.js, en particulier pour les applications de taille moyenne ou lorsque la flexibilité et la simplicité sont des priorités.

e) Installation d'Express.js

Pour installer Express.js, vous devez suivre les étapes suivantes :

Étape 1 : Configuration de l'environnement Node.js

Assurez-vous d'avoir Node.js installé sur votre machine. Vous pouvez le télécharger à partir du site officiel de Node.js (<https://nodejs.org>) et suivre les instructions d'installation spécifiques à votre système d'exploitation.

Étape 2 : Initialisation d'un projet Node.js

Créez un nouveau répertoire pour votre projet Express.js et ouvrez une console ou un terminal à cet emplacement. Exécutez la commande suivante pour initialiser un nouveau projet Node.js

```
npm init -y
```

Figure 17 : Commande pour créer une application Node.js

Cela va créer un fichier package.json qui contiendra les informations de configuration de votre projet.

Étape 3 : Installation d'Express.js

Dans la même console ou terminal, exécutez la commande suivante pour installer Express.js en tant que dépendance de votre projet :

```
npm install express
```

Figure 18 : Commande pour installer Express.js

Cela téléchargera les fichiers nécessaires et les ajoutera à votre répertoire de projet.

Étape 4 : Utilisation d'Express.js dans votre application

Maintenant, vous pouvez commencer à utiliser Express.js dans votre application. Créez un fichier JavaScript (par exemple, app.js) et importez Express.js en utilisant la ligne suivante :

```
const express = require('express');
```

Figure 19 : Commande pour utiliser Express.js dans une application Node.js

Vous pouvez ensuite créer une instance d'Express en ajoutant les lignes suivantes :

```
const app = express();
```

Figure 20 : Commande pour instancier Express.js

À partir de là, vous pouvez définir des routes, des middlewares, des gestionnaires d'erreurs, etc., en utilisant les méthodes et les fonctionnalités fournies par Express.js.

Notez que ce sont les étapes de base pour installer Express.js et commencer à l'utiliser. Selon les besoins de votre projet, vous pourriez également avoir besoin d'installer d'autres modules complémentaires ou d'utiliser des outils supplémentaires. Assurez-vous de consulter la documentation officielle d'Express.js pour plus d'informations sur les fonctionnalités avancées et les meilleures pratiques.

4.2.4. MongoDB

a) C'est quoi MongoDB

MongoDB est un système de gestion de base de données (SGBD) orienté documents, appartenant à la catégorie des bases de données NoSQL (Not Only SQL). Il a été conçu pour stocker et traiter de grandes quantités de données, avec une flexibilité et une évolutivité élevée.

Dans MongoDB, les données sont organisées et stockées sous forme de documents JSON (JavaScript Object Notation) dans ce qu'on appelle des collections. Chaque document peut avoir une structure différente, contrairement aux bases de données relationnelles traditionnelles où les données sont organisées en tables avec un schéma rigide.

Voici quelques caractéristiques clés de MongoDB :

- **Structure flexible** : Les documents JSON permettent une représentation flexible des données. Chaque document peut avoir des champs différents, ce qui facilite l'évolution du schéma des données au fil du temps.
- **Évolutivité horizontale** : MongoDB peut être distribué sur plusieurs serveurs, permettant une scalabilité horizontale en ajoutant simplement de nouveaux nœuds au cluster. Cela permet de gérer facilement de grandes quantités de données et une charge de travail élevée.
- **Haute performance** : MongoDB est conçu pour offrir de bonnes performances, notamment grâce à son modèle de stockage en mémoire cache et à son indexation efficace.
- **Requêtes riches** : MongoDB prend en charge des requêtes riches, y compris des requêtes ad hoc, des indexations textuelles et géospatiale, ainsi que des agrégations de données pour des opérations analytiques avancées.

- **Réplication et tolérance aux pannes** : MongoDB offre des fonctionnalités de réplication automatique, permettant la mise en place de réplicas de données pour assurer une disponibilité continue en cas de défaillance d'un serveur.

MongoDB est utilisé dans de nombreuses applications, notamment dans les environnements où la flexibilité et l'évolutivité sont primordiales, tels que les applications Web, les systèmes de gestion de contenu, les applications de suivi et de traitement de données en temps réel, etc.

b) Avantages de l'utilisation de MongoDB

MongoDB présente plusieurs avantages par rapport aux bases de données relationnelles traditionnelles :

- **Flexibilité du schéma** : Dans MongoDB, les documents peuvent avoir une structure flexible, ce qui signifie que chaque document peut avoir des champs différents. Cela permet d'ajouter, de modifier ou de supprimer des champs facilement, sans nécessiter de modification du schéma de la base de données. Cela rend MongoDB adapté aux applications dont le schéma des données évolue fréquemment.
- **Évolutivité horizontale** : MongoDB est conçu pour être facilement mis à l'échelle horizontalement. Il prend en charge la distribution des données sur plusieurs serveurs, ce qui permet d'ajouter de nouveaux nœuds au cluster pour gérer une augmentation de la charge de travail. Cela permet d'assurer une haute disponibilité et des performances élevées, même avec de grandes quantités de données.
- **Haute performance** : MongoDB offre de bonnes performances grâce à plusieurs fonctionnalités. Il utilise une mémoire cache intégrée pour accélérer l'accès aux données. De plus, MongoDB prend en charge l'indexation efficace, ce qui permet d'accélérer les requêtes en recherchant rapidement les données indexées. Il est également capable de paralléliser les opérations pour optimiser les performances.
- **Requêtes riches** : MongoDB offre une grande souplesse dans les requêtes. Il prend en charge des requêtes ad hoc, ce qui signifie que vous pouvez effectuer des requêtes sans avoir à définir un schéma rigide à l'avance. MongoDB prend également en charge l'indexation textuelle et géospatiale, ce qui permet de réaliser des recherches avancées sur des données textuelles ou basées sur des informations géographiques. De plus, il propose des fonctionnalités d'agrégation de données pour effectuer des opérations analytiques avancées.

- **Réplication et tolérance aux pannes :** MongoDB offre des fonctionnalités de réplication automatique, ce qui signifie que les données peuvent être répliquées sur plusieurs serveurs pour assurer une haute disponibilité et une tolérance aux pannes. En cas de défaillance d'un serveur, les autres serveurs répliqués peuvent prendre le relais et assurer la continuité du service.

Ces avantages font de MongoDB un choix populaire pour les applications qui nécessitent une flexibilité du schéma, une évolutivité horizontale, des performances élevées et des requêtes riches. Cependant, il convient de noter que MongoDB peut ne pas convenir à tous les cas d'utilisation, notamment lorsque les données sont hautement structurées et que des opérations de jointure complexes sont nécessaires.

c) Inconvénients de l'utilisation de MongoDB

Bien que MongoDB présente de nombreux avantages, il existe également quelques inconvénients potentiels à prendre en compte :

- **Consommation de ressources :** En raison de sa flexibilité et de sa capacité à gérer de grandes quantités de données, MongoDB peut consommer plus de ressources système que les bases de données relationnelles traditionnelles. Cela inclut l'utilisation de la mémoire, du processeur et du stockage. Par conséquent, il est important de dimensionner correctement l'infrastructure pour éviter les problèmes de performances.
- **Pas de support intégré pour les transactions multi-documents :** Jusqu'à la version 4.0 de MongoDB, il n'y avait pas de support intégré pour les transactions multi-documents. Cela signifie que les opérations de modification de plusieurs documents ne pouvaient pas être effectuées de manière atomique. Bien que la version 4.0 ait introduit le support des transactions, il est important de noter que leur utilisation peut affecter les performances.
- **Pas adapté aux cas d'utilisation nécessitant des jointures complexes :** MongoDB n'est pas aussi performant que les bases de données relationnelles pour les opérations de jointure complexes impliquant plusieurs collections. Si votre cas d'utilisation nécessite fréquemment des jointures complexes, une base de données relationnelle traditionnelle peut être plus adaptée.
- **Gestion de la cohérence des données :** Comme MongoDB permet une évolutivité horizontale avec des répliques de données, la gestion de la cohérence des données peut être complexe. Lorsqu'une écriture est effectuée sur un nœud, il faut un certain temps

pour que cette modification soit propagée à toutes les répliques. Par conséquent, si une lecture est effectuée immédiatement après une écriture, il peut y avoir une certaine latence pour que la donnée soit cohérente sur tous les nœuds.

- **Apprentissage et adoption :** Lorsque vous utilisez MongoDB, il peut y avoir une courbe d'apprentissage pour comprendre et utiliser efficacement ses fonctionnalités. Si vous êtes habitué aux bases de données relationnelles, l'adaptation à un modèle de données orienté documents peut nécessiter un temps d'apprentissage supplémentaire pour comprendre les meilleures pratiques et les bonnes stratégies de conception.

Il est important de prendre en compte ces inconvénients potentiels et d'évaluer si MongoDB convient à votre cas d'utilisation spécifique. Il est recommandé de bien comprendre les exigences de votre application et de réaliser des tests de performance et de charge pour s'assurer que MongoDB répond à vos besoins.

d) Installation de MongoDB

Pour installer MongoDB, vous pouvez suivre les étapes ci-dessous :

- Rendez-vous sur le site officiel de MongoDB à l'adresse suivante : <https://www.mongodb.com/try/download/community>
- Sélectionnez la version de MongoDB appropriée pour votre système d'exploitation. Vous pouvez choisir entre les différentes versions pour Windows, macOS et Linux.
- Téléchargez le fichier d'installation correspondant à votre système d'exploitation.

Une fois le téléchargement terminé, ouvrez le fichier d'installation et suivez les instructions spécifiques à votre système d'exploitation. Voici quelques instructions générales pour chaque système :

Windows : Exécutez le fichier d'installation (.msi) et suivez l'assistant d'installation pour configurer MongoDB sur votre système.

macOS : Ouvrez le fichier d'installation (.dmg) et faites glisser l'icône de MongoDB dans le dossier Applications.

Linux : Décompressez l'archive téléchargée et copiez les fichiers binaires de MongoDB dans un répertoire approprié. Vous devrez également configurer les variables d'environnement, comme le chemin d'accès au répertoire bin de MongoDB, dans votre fichier de configuration du profil système (par exemple, ~/.bashrc ou ~/.bash_profile).

Une fois l'installation terminée, vous pouvez démarrer MongoDB. Dans la plupart des cas, MongoDB démarrera automatiquement après l'installation. Si ce n'est pas le cas, vous pouvez démarrer MongoDB manuellement en utilisant la commande appropriée pour votre système d'exploitation.

Pour vérifier si MongoDB est installé et fonctionne correctement, ouvrez une fenêtre de terminal ou de ligne de commande et exécutez la commande suivante :



```
mongod --version
```

Figure 21 : Commande pour voir la version de mongodb installé.

Cela affichera la version de MongoDB installée sur votre système.

Félicitations ! Vous avez maintenant installé MongoDB sur votre système. Vous pouvez commencer à utiliser MongoDB en vous connectant à l'interface de ligne de commande (ou shell en anglais) ou en utilisant un pilote compatible avec le langage de programmation de votre choix.

4.3. Content Management System

Un Content Management System ou CMS en anglais, est un système de gestion de contenu. Il s'agit d'un logiciel utilisé pour créer, gérer et organiser le contenu d'un site web de manière conviviale, sans avoir besoin de compétences techniques avancées en programmation.

Un CMS permet aux utilisateurs de créer et de modifier facilement des pages web, d'ajouter du contenu, d'organiser la structure du site, de gérer des utilisateurs et des autorisations, ainsi que d'effectuer d'autres tâches liées à la gestion du contenu. Les CMS sont souvent utilisés pour la création de sites web tels que des blogs, des sites d'actualités, des sites d'entreprises, des sites de commerce électronique, etc.

4.3.1. Caractéristiques et fonctionnalités d'un CMS

Les CMS facilitent la création et la gestion de sites web de manière flexible, efficace et adaptée aux utilisateurs qui ne sont pas des experts en programmation.

Interface utilisateur conviviale : Les CMS offrent une interface intuitive et conviviale qui permet aux utilisateurs de gérer le contenu du site sans avoir à écrire de code.

Gestion de contenu : Les CMS permettent aux utilisateurs de créer et de modifier facilement des pages web, d'ajouter du texte, des images, des vidéos, des liens et d'autres types de contenu.

Gestion des utilisateurs et des autorisations : Les CMS offrent des fonctionnalités pour gérer les utilisateurs, attribuer des rôles et des autorisations spécifiques, et contrôler l'accès au contenu du site.

Modèles et thèmes : Les CMS proposent généralement une sélection de modèles prédéfinis et de thèmes personnalisables pour permettre aux utilisateurs de créer un design attrayant et cohérent pour leur site web.

Gestion des médias : Les CMS facilitent l'importation, la gestion et la publication d'images, de vidéos et d'autres fichiers multimédias sur le site.

Fonctionnalités de recherche : Les CMS intègrent souvent des fonctionnalités de recherche qui permettent aux visiteurs du site de trouver facilement le contenu souhaité.

Optimisation pour les moteurs de recherche (SEO) : Certains CMS offrent des fonctionnalités intégrées pour faciliter l'optimisation du site web pour les moteurs de recherche, telles que la gestion des balises méta, les URL conviviales, les sitemaps, etc.

Extensibilité et personnalisation : Les CMS offrent la possibilité d'étendre leurs fonctionnalités en ajoutant des plugins, des modules complémentaires ou des extensions spécifiques pour répondre aux besoins spécifiques du site.

Dans notre cas, le CMS utilisé est Strapi.

4.3.2. Strapi

Strapi est un système de gestion de contenu (CMS) headless open-source, qui permet de créer facilement et de gérer des API RESTful ou GraphQL pour fournir du contenu à des applications front-end. Contrairement aux CMS traditionnels, Strapi se concentre sur la gestion de contenu en tant qu'API, offrant ainsi une plus grande flexibilité et une séparation claire entre le contenu et la présentation.

Strapi utilise principalement JavaScript comme langage de programmation. Plus précisément, il s'appuie sur Node.js, un environnement d'exécution JavaScript côté serveur, pour exécuter son code. Strapi utilise également d'autres technologies JavaScript couramment utilisées, telles qu'Express.js, un Framework web pour Node.js, et MongoDB, une base de données NoSQL, pour stocker les données.

Du côté du front-end, Strapi peut être utilisé avec n'importe quel langage de programmation ou Framework front-end qui prend en charge les requêtes HTTP vers des API RESTful ou GraphQL. Cela peut inclure des langages et des Framework populaires tels que React, Vue.js, Angular, etc.

En résumé, pour travailler avec Strapi, il est nécessaire d'avoir des connaissances en JavaScript, Node.js, Express.js et MongoDB pour la configuration, le développement et la personnalisation du back-end. Pour le front-end, il est préférable de maîtriser le langage ou le Framework de programmation utilisé pour consommer les API de Strapi.

4.3.3. Caractéristiques et fonctionnalités clés de Strapi

Interface d'administration conviviale : Strapi propose une interface d'administration intuitive et facile à utiliser, permettant aux utilisateurs de gérer leur contenu, de créer des types de contenu personnalisés, d'ajouter et de modifier du contenu sans nécessiter de compétences techniques avancées.

Création de contenu flexible : Strapi permet aux utilisateurs de créer des modèles de données personnalisés pour structurer le contenu selon leurs besoins spécifiques. Les utilisateurs peuvent définir des champs, des relations et des validations pour chaque type de contenu.

Gestion avancée des autorisations : Strapi offre un système de gestion des autorisations robuste, permettant aux administrateurs de définir des rôles d'utilisateur et de contrôler finement les actions et les accès autorisés à chaque type de contenu ou d'API.

API RESTful et GraphQL : Strapi génère automatiquement des API RESTful et/ou GraphQL pour chaque type de contenu créé, permettant aux développeurs de récupérer et de manipuler le contenu via ces API depuis des applications front-end ou d'autres systèmes.

Extensibilité et personnalisation : Strapi offre une architecture modulaire et extensible, permettant aux développeurs de personnaliser et d'étendre les fonctionnalités en ajoutant des plugins ou en modifiant le code source selon leurs besoins spécifiques.

Gestion des médias : Strapi propose des fonctionnalités de gestion des médias, permettant aux utilisateurs de télécharger, de stocker et de gérer des fichiers multimédias tels que des images, des vidéos, etc.

Internationalisation (i18n) : Strapi prend en charge l'internationalisation, ce qui permet aux utilisateurs de créer et de gérer du contenu dans différentes langues, facilitant ainsi la création de sites multilingues.

Communauté active : Strapi bénéficie d'une communauté active qui propose des ressources, des tutoriels, des plugins et des thèmes supplémentaires, ainsi qu'un support communautaire.

Grâce à ses fonctionnalités puissantes, sa flexibilité et sa facilité d'utilisation, Strapi est une solution populaire pour les développeurs et les équipes souhaitant créer des applications front-end modernes avec un CMS headless, permettant une plus grande liberté de conception et une meilleure séparation des préoccupations entre le contenu et la présentation.

4.3.4. Pourquoi choisir Strapi

Il existe plusieurs raisons pour lesquelles vous pourriez envisager d'utiliser Strapi comme système de gestion de contenu (CMS) pour votre projet. Ci-dessus quelques-unes des principales raisons :

Flexibilité et architecture headless : Strapi est un CMS headless, ce qui signifie qu'il se concentre sur la gestion de contenu en tant qu'API et permet de séparer complètement le front-end du back-end. Cela offre une grande flexibilité dans la conception et le développement de votre application, car vous pouvez utiliser n'importe quel Framework ou langage de programmation pour construire le front-end et communiquer avec l'API de Strapi.

Facilité d'utilisation : Strapi offre une interface d'administration conviviale et intuitive qui permet aux utilisateurs non techniques de créer, gérer et organiser facilement leur contenu. Vous pouvez créer des modèles de données personnalisés, ajouter du contenu, gérer des utilisateurs et des autorisations, tout cela grâce à une interface utilisateur conviviale.

Personnalisation et extensibilité : Strapi est hautement personnalisable et extensible. Vous pouvez créer vos propres types de contenu, définir des relations complexes entre eux, et ajouter des plugins pour étendre les fonctionnalités de base de Strapi. Cela vous permet de créer une expérience de gestion de contenu sur mesure qui répond spécifiquement aux besoins de votre projet.

Développement rapide d'API : Strapi génère automatiquement des API RESTful ou GraphQL pour chaque type de contenu créé. Cela signifie que vous pouvez rapidement mettre en place une API robuste pour alimenter vos applications front-end sans avoir à écrire une grande partie du code d'infrastructure nécessaire.

Écosystème et communauté active : Strapi bénéficie d'une communauté active d'utilisateurs et de contributeurs. Vous pouvez trouver de nombreuses ressources, tutoriels,

exemples de code, plugins et thèmes développés par la communauté pour vous aider à démarrer rapidement et résoudre les problèmes rencontrés.

Performances et évolutivité : Strapi est conçu pour être performant et évolutif. Il est basé sur Node.js et utilise une architecture moderne qui permet de gérer facilement des volumes de données importants et de gérer des charges de trafic élevées.

En résumé, utiliser Strapi comme CMS nous offre une flexibilité, une facilité d'utilisation et une personnalisation importante dans la gestion de notre contenu. Cela nous permet de créer des applications web modernes, réactives et évolutives, tout en bénéficiant d'un écosystème actif et d'une communauté de soutien.

4.3.5. Installation de Strapi

Les étapes générales pour installer Strapi :

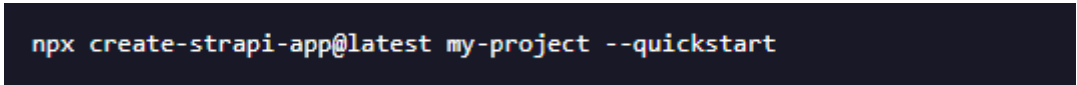
- **Prérequis**

Assurez-vous d'avoir Node.js installé sur votre machine. Vous pouvez le télécharger à partir du site officiel de Node.js (<https://nodejs.org>) et suivre les instructions d'installation pour votre système d'exploitation.

- **Création d'un nouveau projet Strapi**

Ouvrez votre terminal (ou invite de commandes) et accédez au répertoire où vous souhaitez créer votre projet Strapi.

Exécutez la commande suivante pour créer un nouveau projet Strapi



```
npx create-strapi-app@latest my-project --quickstart
```

Figure 22 : Commande pour créer un nouveau projet Strapi.

Remplacez "my-project" par le nom que vous souhaitez donner à votre projet.

- **Configuration du projet**


Après la création du projet, vous serez invité à choisir un Template de démarrage. Sélectionnez le Template qui correspond le mieux à votre projet ou choisissez "Custom" pour une configuration personnalisée.

Suivez les instructions pour configurer la base de données. Strapi prend en charge plusieurs bases de données, y compris SQLite, MySQL, PostgreSQL et MongoDB. Choisissez celle qui convient le mieux à votre projet et fournissez les informations de connexion nécessaires.

- **Installation des dépendances**

Une fois la configuration terminée, accédez au répertoire du projet Strapi (à l'aide de la commande `cd nom-du-projet`).

Exécutez la commande suivante pour installer les dépendances nécessaires

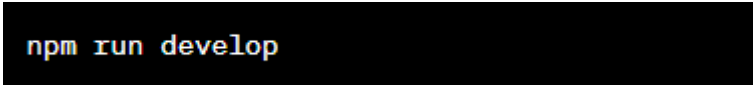


```
npm install
```

Figure 23 : Commande pour installer les dépendances nécessaires de Strapi.

- **Démarrage du serveur Strapi**

Une fois l'installation des dépendances terminée, vous pouvez démarrer le serveur Strapi en exécutant la commande suivante



```
npm run develop
```

Figure 24 : Commande pour démarrer le serveur de Strapi.

- **Accès à l'interface d'administration Strapi**

Après le démarrage du serveur, vous pouvez accéder à l'interface d'administration Strapi en ouvrant votre navigateur et en visitant l'URL <http://localhost:1337/admin>.

Suivez les instructions pour créer un compte administrateur et vous connecter à l'interface d'administration.

Ces étapes fournissent une vue d'ensemble générale de l'installation de Strapi. Pour des instructions plus détaillées ou des configurations spécifiques, consultez la documentation officielle de Strapi (<https://strapi.io/documentation>) qui fournit des informations complètes et à jour sur l'installation et la configuration de Strapi.

Partie 3 : Essais et résultats

CHAPITRE V : ESSAIS ET RESULTATS

5.1. Résultats

Dans ce chapitre, nous allons dévoiler les résultats obtenus après la conception pratique. Certes, ces résultats sont le fruit de nombreux débogages des erreurs et fonctionnalités durant l'achèvement total de l'application web.

5.1.1. La page d'authentification

L'application mobile possède une page d'authentification qui permet de s'authentifier avant d'accéder dans la plateforme. Ceci a pour but la sécurité de la plateforme en vue des documents et informations sensibles et propre à l'établissement.

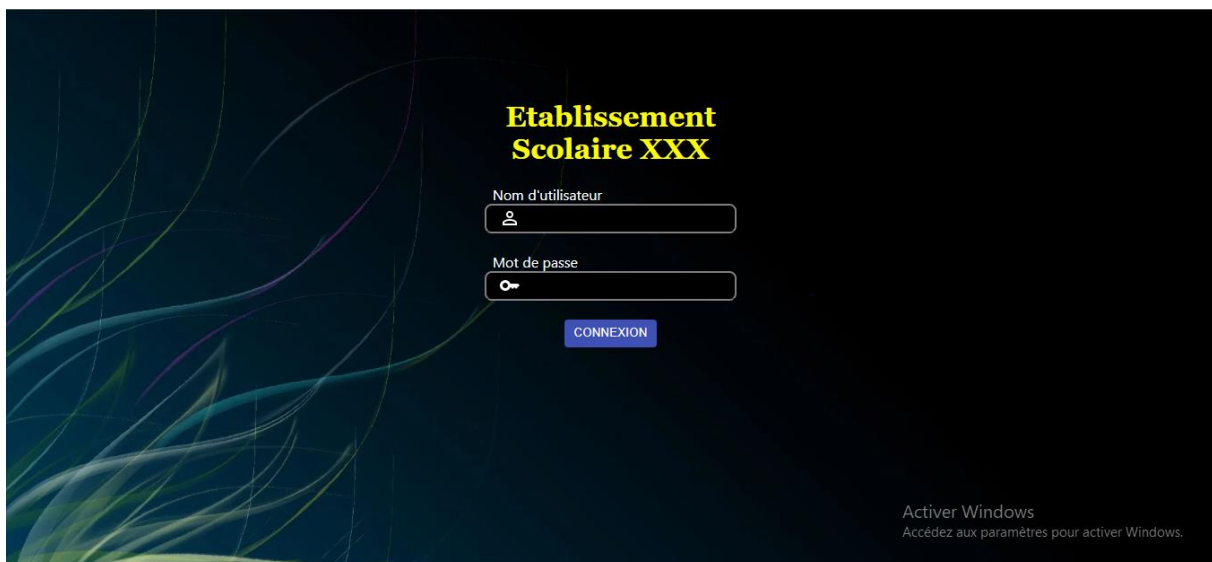


Figure 25 : Page d'authentification de la plateforme.

Ci-dessus la page d'authentification de l'application, cette page comporte deux champs à remplir qui sont : Nom d'utilisateur et Mot de passe.

Pour se connecter, il suffit alors d'entrer votre propre nom d'utilisateur et votre mot de passe.

Cette page ne contient pas une option de mot de passe oublié où de créer un nouveau compte pour plus de sécurité. Ainsi, seuls les utilisateurs inscrits par les administrateurs auront l'accès.

Le bouton « CONNEXION » déclenchera alors l'action d'authentification. Si les champs sont vides, un message d'erreur apparaîtra alors. Si vous entrez des données incorrectes, un message d'erreur surviendra aussi. Si vous entrez les données correctes, alors vous serez redirigés vers le tableau de bord de la plateforme.

5.1.2. Tableau de bord de la plateforme

Après que vous vous êtes authentifié, vous serez redirigés vers cette page intitulé « Tableau de bord ». C'est la page principale de la plateforme.

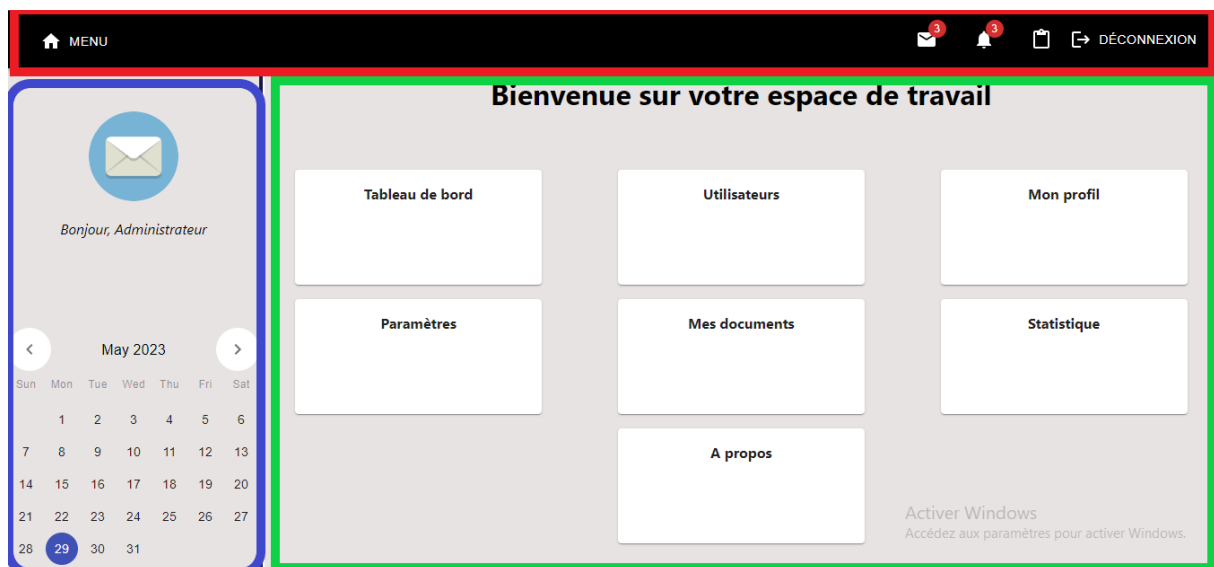


Figure 26 : Page de tableau de bord de la plateforme.

L'interface se divise en trois grandes parties :

5.1.2.1. La barre de navigation

La barre de navigation est une composante que toutes les autres pages de la plateforme héritent. Ce qui veut dire que c'est toujours cette barre de navigation qui s'affichera dans toutes les différentes pages de la plateforme. Cette partie est illustrée par la couleur rouge dans l'image ci-dessus.

Celle-ci comporte les boutons suivants :

- Menu : En appuyant sur ce bouton menu, la barre de menu latéral va s'afficher. Dans cette option, nous pouvons cliquer sur les différents pages de la plateforme à savoir le tableau de bord, utilisateurs, mon profil, paramètres, mes documents, statistique et à propos. Mais l'affichage de ces dernières dépend du rôle de l'utilisateur connecté.
- Messages : En appuyant sur ce bouton, vous serez redirigés vers la page de discussion instantané de la plateforme. Vous pouvez alors communiquer avec les autres utilisateurs de la plateforme.
- Notifications : En appuyant sur ce bouton, vous serez redirigés vers la page de notifications de la plateforme où vous serez en position de voir toutes les notifications vous concernant.
- Communiqués : En appuyant sur ce bouton, vous serez redirigés vers la page de communiqué où vous pouvez voir tous les derniers communiqués de l'établissement.
- Déconnexion : En appuyant sur ce bouton, vous serez déconnecté de la plateforme et vous ne pourrez plus revisiter cette dernière sans vous reconnecter à nouveau.

5.1.2.2. SideBar de la plateforme

Dans cette partie de la plateforme, on peut visualiser la photo de profil de l'utilisateur ainsi que son nom suivi du mot « bonjour » pour souhaiter la bienvenue à l'utilisateur connecté.

On peut aussi voir un calendrier pour une interface plus présentable et utile à l'utilisateur. Cette partie est aussi héritée pour toutes les autres pages de la plateforme. Cette partie est illustrée par la couleur bleue dans l'image ci-dessus.

5.1.2.3. Contenu de la plateforme

C'est la partie principale qui contient les différentes données de chaque page de la plateforme. Dans la page « Tableau de bord », celle-ci nous montre une phrase de « bienvenue dans votre espace de travail » ainsi différents composants nous permettant de naviguer plus facilement dans la plateforme. Ces différents composants diffèrent avec chaque rôle des utilisateurs connectés mais leurs formes restent le même. Cette partie est illustrée par la couleur verte dans l'image ci-dessus.

Pour un utilisateur avec un rôle d'administrateur, cette contenue est comme suit :

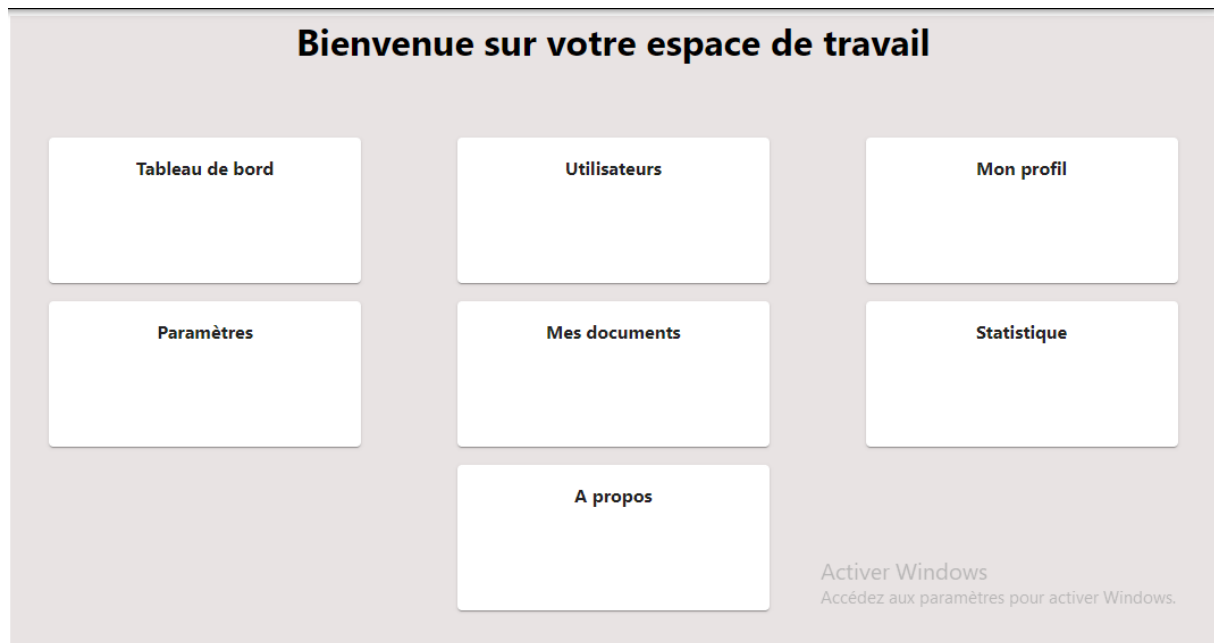


Figure 27 : Contenu du tableau de bord de la plateforme pour un administrateur.

Mais pour un utilisateur avec un rôle d'étudiant ou d'enseignant, cette interface diffère un peu comme suit :

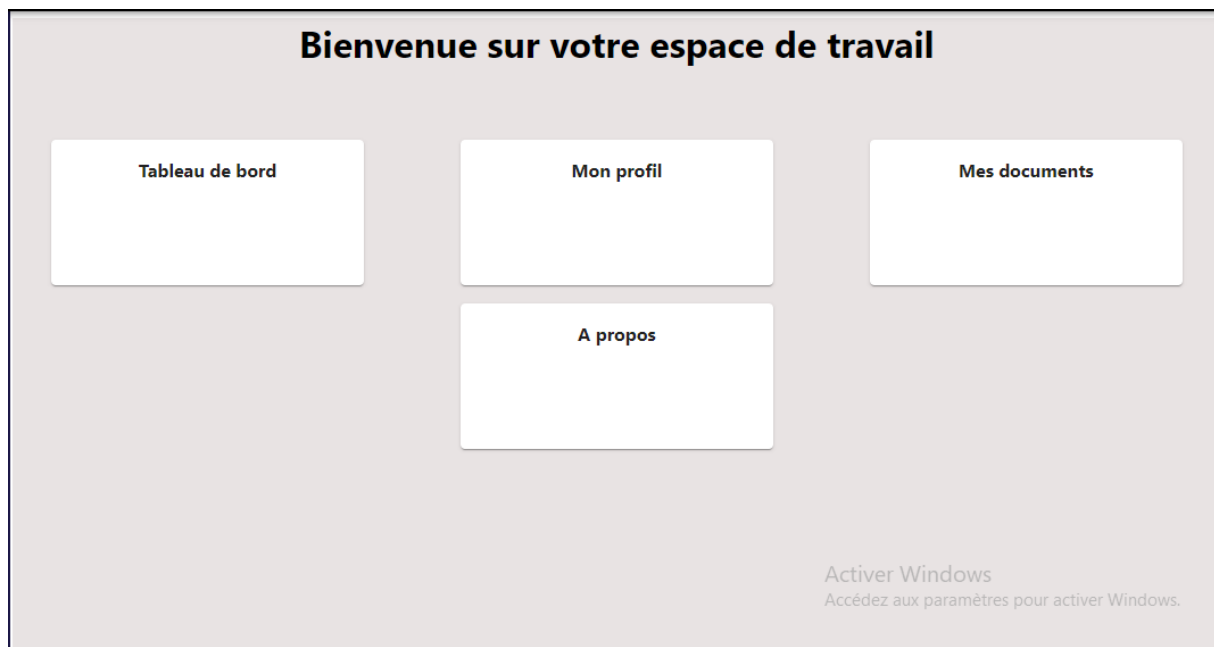


Figure 28 : Contenu du tableau de bord de la plateforme pour un enseignant ou étudiant.

5.1.3. Utilisateurs

C'est une page seulement accessible aux administrateurs. Dans cette page, les informations personnelles concernant chaque utilisateur est affichés.

C'est dans cette page qu'un administrateur peut ajouter un nouvel utilisateur que ce soit un autre administrateur, un enseignant ou un étudiant. Il peut aussi voir les informations déjà disponibles, les mettre à jour ou les supprimer.

ADMINISTRATEURS ENSEIGNANTS ETUDIANTS							
Rechercher ...		RECHERCHER		CRÉER			
Image	Nom	Prénom	Adresse	E-mail	CIN	Date de naissance	Lieu de
	Administrateur	Administrateur	Ilafy Ankadikely 213	jr-sellambin@hotmail.com	12345678	13/04/2023	Tana
	Administratrice	administratrice	ilafy	jr-sellambin@hotmail.com	1234	18/05/2023	Tana
	Nouvel admin	test	ilafy	jr-sellambin@hotmail.com	1234	25/05/2023	Tana
	test encore	encore	ipjpijpijp	jr-sellambin@hotmail.com	1234	02/06/2023	Tana
	hdffdh	hfdhdfh	fdhdf	jr-sellambin@hotmail.com	1234	24/05/2023	Tana
Activer Windows Accédez aux paramètres pour activer Windows.							

Figure 29 : Espace utilisateurs de la plateforme.

Dans cette image, il existe trois onglets qui sont : Administrateurs, Enseignants ainsi que Etudiants. Les informations des utilisateurs sont séparées conformément à ses différents rôles.

Dans chaque onglet, on peut voir :

- Un champ « Rechercher.. » : cette champ nous permet d'entrer une données concernant un utilisateur spécifique que l'on veut rechercher.

- Un bouton « RECHERCHER » : celle-ci nous permet de lancer une recherche en fonction d'une donnée entrée dans le champ de recherche.
- Un bouton « créer » : ce bouton nous ouvre une autre page pour nous permettre d'ajouter un nouvel utilisateur.

En cliquant sur un enregistrement de ce tableau, une fenêtre de dialogue s'ouvre qui nous affiche les informations correspondantes à un utilisateur sélectionnées.

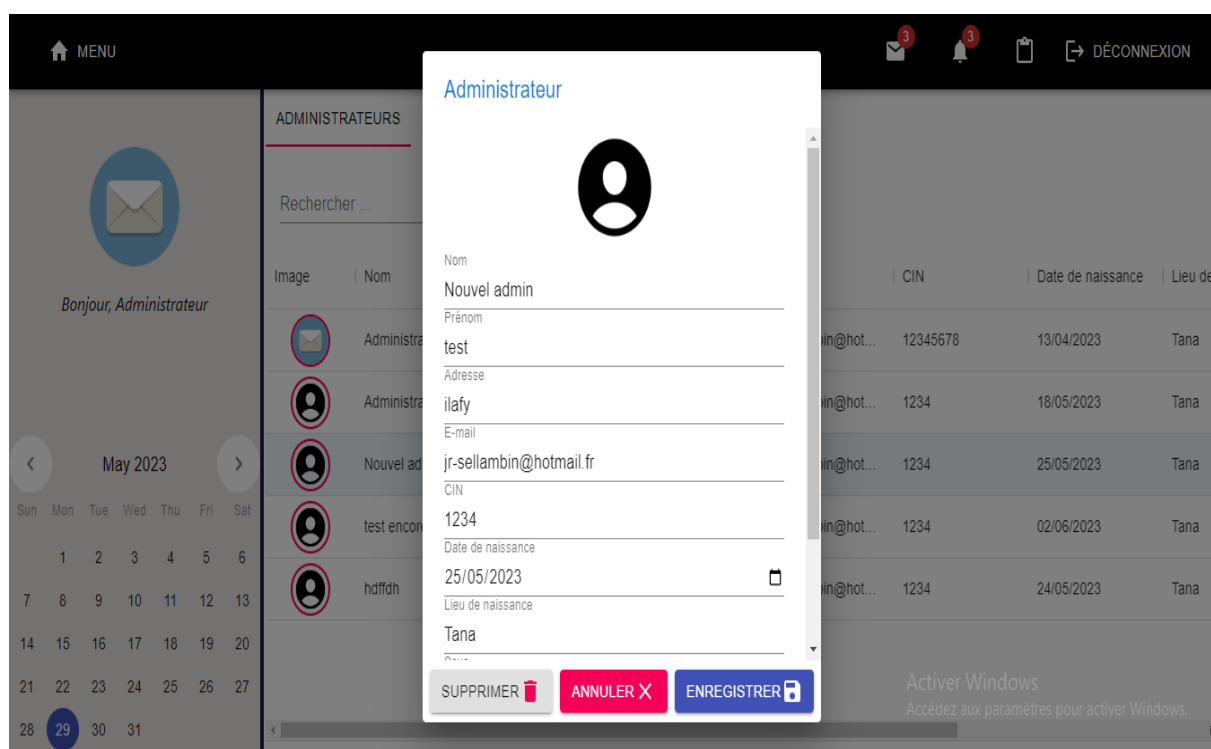


Figure 30 : Dialogue montrant les informations d'un utilisateur.

Une fois cette boîte de dialogue ouverte, on peut lire les informations concernant un utilisateur mais aussi les modifier ou les supprimer. Ceci est valable pour les onglets Enseignants et Etudiants.

The screenshot displays the 'Création Administrateur' (Administrator Creation) page. The interface includes a top navigation bar with a home icon, 'MENU', and notification icons. A left sidebar shows a calendar for May 2023 and a greeting 'Bonjour, Administrateur'. The main content area is titled 'Création Administrateur' and contains a form with the following fields: 'Nom *', 'Prénom *', 'Sexe' (with a dropdown arrow), 'Adresse *', 'Date de naissance *' (pre-filled with 'jj/mm/aaaa'), 'Lieu de naissance *', 'E-mail *', 'CIN *', and 'Poste *'. At the bottom of the form are two buttons: 'ANNULER' and 'ENREGISTRER'. A Windows watermark is visible in the bottom right corner.

Figure 31 : Page de création d'un administrateur de la plateforme.

La figure ci-dessus nous montre la page de création d'un utilisateur dans la plateforme. On peut y voir les différents champs pour les informations utiles et nécessaires pour créer un nouvel utilisateur avec un rôle d'administrateur.


Le bouton « annuler » nous permet d'annuler l'action de création et de revenir à la liste d'utilisateur.


Le bouton « Enregistrer » nous permet d'effectuer l'action d'ajout d'un nouvel utilisateur. Une fois terminer, nous serons redirigés vers la liste d'utilisateurs.


Le même principe est aussi ajouté aux onglets Enseignants et Etudiants.


5.1.4. Profil


Dans cette page, nous pouvons voir nos informations nous concernant. C'est-à-dire nos propres informations introduites lors de notre ajout dans la plateforme. Nous pouvons ainsi modifier quelques informations si besoin.


 MENU







 DÉCONNEXION




Bonjour, Administrateur

<

May 2023

>

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			



Nom

Administrateur

E-mail

jr-seliambin@hotmail.fr

CIN

12345678

Adresse

Ilafy Ankadikely 213

Adresse

Ilafy Ankadikely 213

Sexe

Homme

Prénom

Administrateur

Date de naissance

13/04/2023

Lieu de naissance

Tana

Mot de passe

Enregistrer

Activer Windows


Accédez aux paramètres pour activer Windows.



Figure 32 : Page de profil de la plateforme.

Dans la figure ci-dessus, on peut voir les différents champs ainsi que les informations correspondantes. Pour les modifier, il suffit d'entrer les nouvelles informations et de cliquer sur le bouton « Enregistrer ».

5.1.5. Paramètres

Cette page aussi est réservée seulement aux administrateurs. Dans cette page, on peut ajouter, lire, supprimer ou modifier les différents niveaux, matières, années scolaires en cours ainsi que les périodes d'examens.

 MENU



Bonjour, Administrateur

<

June 2023

>

Sun

Mon

Tue

Wed

Thu

Fri

Sat

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

NIVEAU

MATIERE

AUTRES

Liste des niveaux

Créer

Code	Niveau
L1	Licence 1
L2	Licence 2
L3	Licence 3
M1	Master 1
M2	Master 2

Activer Windows

Accédez aux paramètres pour activer Windows.

1-5 of 6 < >

Figure 33 : Paramètres dans la plateforme.

La figure ci-dessus nous affiche trois onglets concernant les niveaux, les matières et autres. Il suffit de sélectionner un enregistrement pour pouvoir lire les données, les modifier ou les supprimer.

5.1.6. Mes documents

Dans cette page, nous pouvons voir les différents documents à notre disposition. Elle se présente comme une bibliothèque numérique où sont stockés les documents communs que chaque utilisateur peut consulter à chaque instant.



Figure 34 : Mes documents dans la plateforme.

Dans la figure ci-dessus, nous pouvons voir les différents documents qui sont à notre disposition. Il suffit de cliquer dessus pour les télécharger.

5.1.7. Statistique

C'est aussi une page réservée uniquement aux administrateurs. Dans cette page sont illustrées les statistiques de notes des étudiants. Pour pouvoir analyser le taux de niveaux des étudiants de chaque promotion.

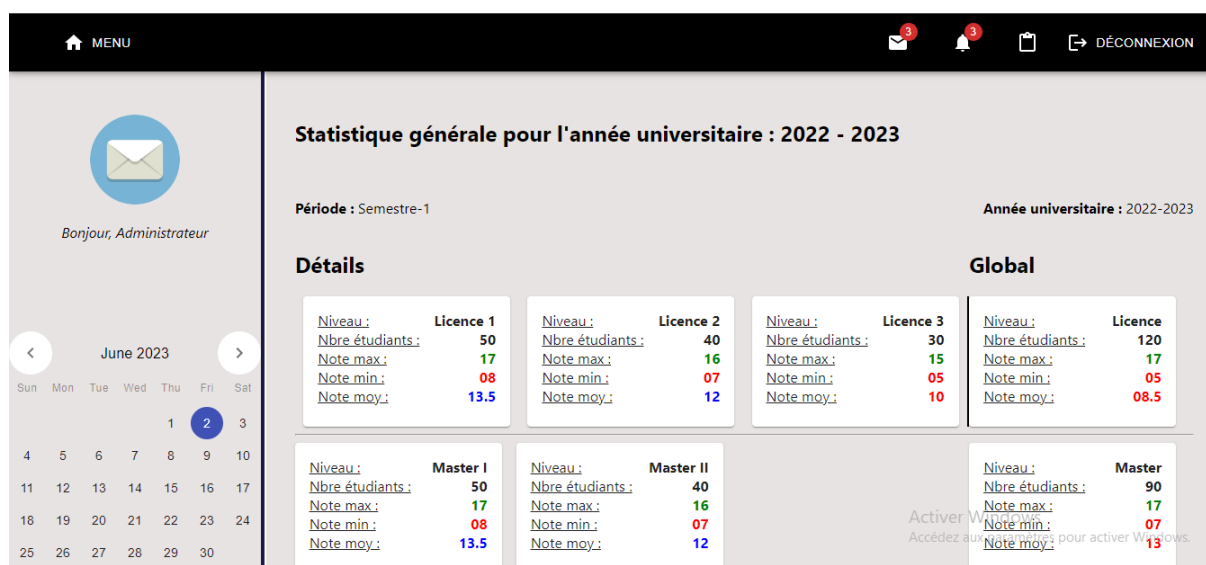


Figure 35 : Statistique de la plateforme.

La figure ci-dessus nous permet de suivre l'évolution des niveaux des étudiants des différents niveaux et différentes promotions.

5.1.8. A propos

C'est une page concernant les informations sur l'établissement scolaire en question. Il contient des images, des informations tels que : l'adresse, contact, etc. Les informations sur cette page sont totalement faciles à mettre à jour car il s'agit d'une donnée avec le CMS Strapi. Un administrateur ayant accès à l'espace admin de Strapi est en mesure de mettre à jour ces informations.



Figure 36 : Page à propos de l'établissement dans la plateforme.

5.1.9. Messagerie

Dans cette page, nous pouvons communiquer directement avec les autres utilisateurs de la plateforme. On y trouve la liste de tous les utilisateurs. Il suffit de sélectionner un utilisateur et d'écrire un message pour l'envoyer à cette dernière.

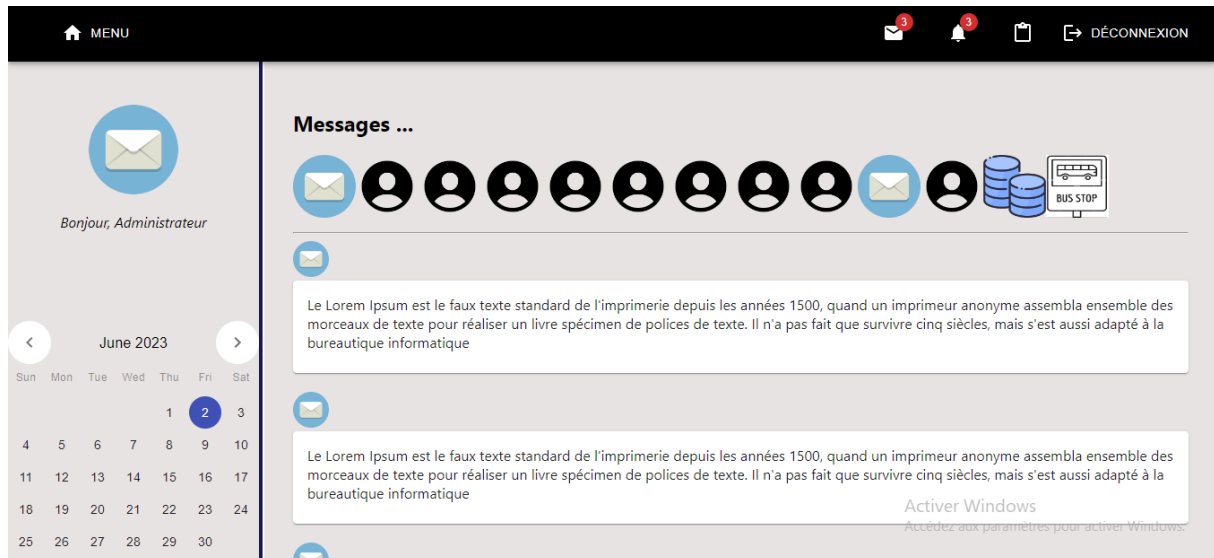


Figure 37 : Messagerie dans la plateforme.

5.1.10. Notifications

Il existe des actions qui provoquent l'ajout des notifications dans la plateforme. Ces notifications sont listées dans cette page. On peut alors lire toutes les notifications disponibles dans cette dernière.

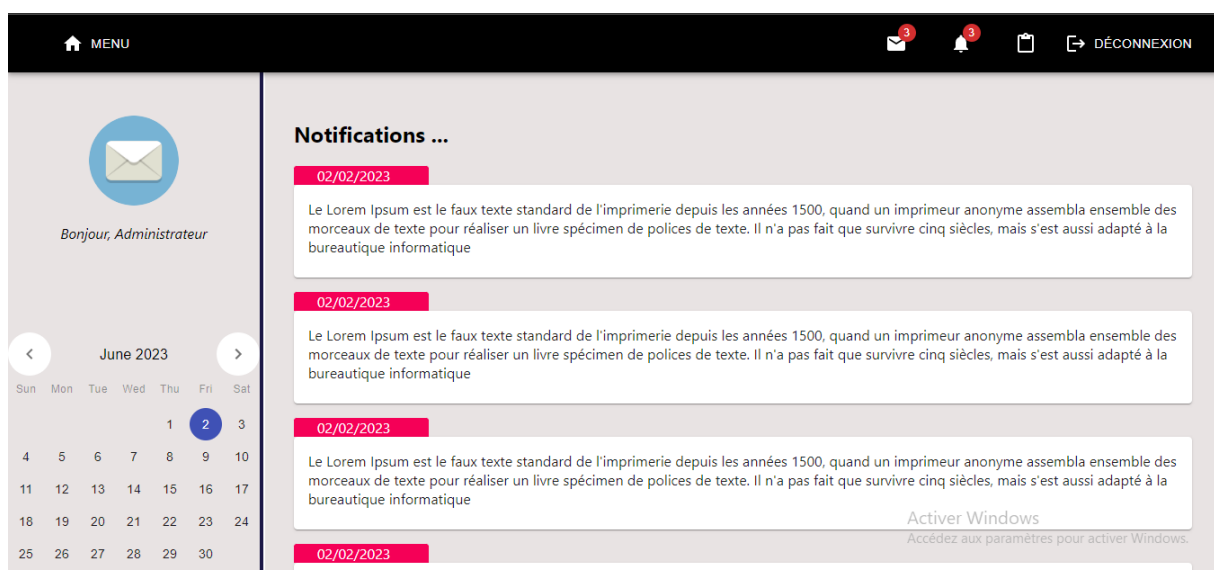


Figure 38 : Notifications dans la plateforme.

5.1.11. Communiqué

Dans cette page sont listés les différentes nouvelles ainsi que les communiqués communs dans l'établissement. Seuls les administrateurs peuvent ajouter du contenu, quant aux autres, ils ne peuvent que lire les informations. Ces informations peuvent contenir des fichiers.

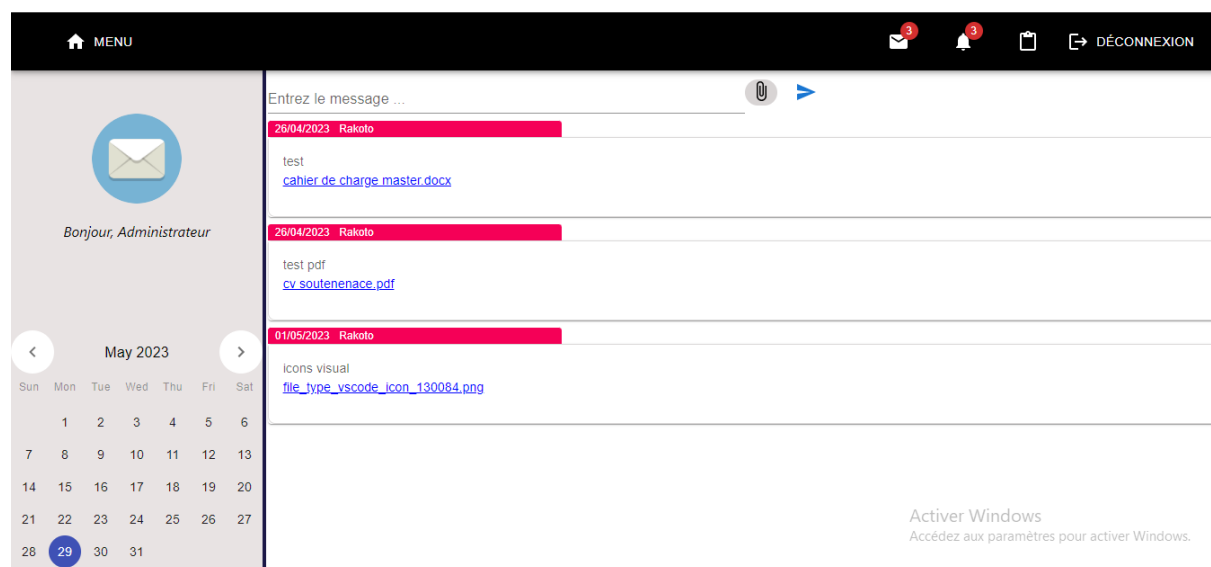


Figure 39 : Communiqué dans la plateforme.

Ce champ de texte pour entrer un message n'est visible que pour un utilisateur avec un rôle d'administrateur.

5.2. Bilan et recommandations

Concernant le projet, des recommandations sont à discuter :

Amélioration de l'interface utilisateur : On pourrait s'assurer que l'interface utilisateur de l'application est conviviale, intuitive et attrayante. Les utilisateurs doivent pouvoir naviguer facilement dans l'application, accéder aux fonctionnalités pertinentes et comprendre rapidement comment utiliser les différentes options de gestion des données.

Renforcement de la sécurité des données : Les données sont précieuses et doivent être protégées contre les accès non autorisés. Des mesures de sécurité solides, telles que l'authentification à deux facteurs, le chiffrement des données sensibles et la sauvegarde régulière des données peuvent être implémentés.

Optimisation des performances : Une telle application doit être rapide et réactive, surtout lorsque des volumes importants de données sont traités. On recommande d'optimiser les performances de l'application en utilisant des techniques telles que la mise en cache, l'indexation efficace des données et l'optimisation des requêtes.

Analyses et visualisations : Les utilisateurs peuvent bénéficier de fonctionnalités d'analyse et de visualisation intégrées pour mieux comprendre et interpréter leurs données. On recommande d'incorporer des outils d'analyse et de visualisation de données, tels que des graphiques, des tableaux de bord interactifs ou des rapports automatisés.

Documentation complète : Une documentation détaillée est essentielle pour aider les utilisateurs à comprendre le fonctionnement de l'application et à tirer le meilleur parti de ses fonctionnalités. Une documentation claire et complète, comprenant des guides d'utilisation, des tutoriels et des exemples pratiques devraient être fournies.

Mise à jour continue : On doit s'assurer que l'application soit mise à jour avec la dernière technologie en vogue et aussi des informations à jour.

Ces recommandations visent à améliorer l'expérience utilisateur, la sécurité et les performances de l'application, tout en offrant des fonctionnalités avancées pour la gestion et l'analyse des données.

CONCLUSION

En guise de conclusion, ce mémoire nous a dévoilé qu'avec la prolifération massive des données au sein des organisations, il est devenu essentiel de disposer d'une application de communication et de gestion de données efficace pour extraire, organiser, analyser et exploiter ces précieuses ressources informationnelles.

Dans ce travail, nous avons mis l'accent sur les avantages bénéfique sur l'utilisation de cette application. La convivialité de l'interface utilisateur est un élément essentiel pour garantir une adoption efficace de l'application. Des interfaces intuitives et attrayantes permettent aux utilisateurs de naviguer facilement dans l'application, d'accéder rapidement aux fonctionnalités et d'accomplir leurs tâches sans complications.

En outre, des recommandations sont aussi à discuter pour une meilleure utilisation de l'application. La sécurité des données est une préoccupation majeure dans le contexte actuel où les violations de données sont devenues courantes. Il est crucial de mettre en œuvre des mesures de sécurité solides, telles que l'authentification à deux facteurs, le chiffrement des données sensibles et la sauvegarde régulière des données, afin de protéger les informations confidentielles et de garantir la confidentialité et l'intégrité des données. Les performances de l'application sont également primordiales pour assurer une expérience utilisateur fluide. Les fonctionnalités avancées, telles que les analyses et les visualisations, apportent une réelle valeur ajoutée à cette application. Elles permettent aux utilisateurs de trouver rapidement les informations dont ils ont besoin, de prendre des décisions éclairées et d'extraire des connaissances précieuses à partir de leurs données. Enfin, une documentation complète et à jour est indispensable pour faciliter l'adoption et l'utilisation de l'application. Elle fournit aux utilisateurs les informations nécessaires pour comprendre les fonctionnalités, les processus et les bonnes pratiques liées à l'application, favorisant ainsi une utilisation optimale et une résolution autonome des problèmes éventuels.

En gardant ces aspects à l'esprit, les développeurs peuvent créer une application qui répond efficacement aux besoins des utilisateurs, facilite la gestion des données et contribue à la prise de décisions éclairées dans les domaines où la gestion de données est cruciale.

REFERENCE BIBLIOGRAPHIQUE ET WEBOGRAPHIQUE

[1] : Visité le mois de Mai 2023. Une bibliothèque JavaScript pour créer des interfaces utilisateurs. Récupéré sur : <https://fr.legacy.reactjs.org/>

[2] : Visité en mois d'Avril 2023. Node.js® is an open-source, cross-platform JavaScript runtime environment. Récupéré sur : <https://nodejs.org/en>

[3] : Visité en mois d'Avril 2023. Express, Fast, unopinionated, minimalist web framework for Node.js. Récupéré sur : <https://expressjs.com/>

[4] : Visité en mois d'Avril 2023. MONGODB, Build the next big thing. Récupéré sur : <https://www.mongodb.com/>

[5] : Visité en mois de Mai 2023. Strapi, Manage Any Content. Anywhere. Récupéré sur : <https://strapi.io/>

[6] : Eric Sarrion, React.js, Le Framework javascript de facebook, edition EYROLLES.

[7] : Alain Cazes, Joëlle Delacroix, Développez une application web, septembre 2016.

TABLE DES MATIERES

CHAPITRE I : PRESENTATION DU THEME.....	3
1.1. Choix de l'objet de la recherche.....	3
1.2. Objectif.....	3
1.3. Intérêts de l'objet de la recherche.....	4
1.4. Architecture de la plateforme.....	4
1.5. Explication de l'architecture	6
1.5.1. Authentification	6
1.5.2. Interface Utilisateur (IU)	7
1.5.3. API (Application Programming Interface).....	8
1.5.4. Middleware	8
1.5.5. Serveur (back- end)	9
1.6. Gestion des utilisateurs	10
1.7. Fonctionnalités dans l'application	11
1.7.1. Gestion des utilisateurs.....	11
1.7.2. Gestion des communiqués	11
1.7.3. Envoi de mail à partir de la plateforme	12
1.7.4. Communication et partage de documents entre utilisateurs	12
1.7.5. Téléchargement de document	12
CHAPITRE II : ETUDE THEORIQUE DU THEME.....	13
2.1. Application Web	13
2.2. Avantages d'une application web.....	13
2.3. Inconvénients d'une application web.....	14
2.4. Une plateforme web.....	15
2.5. Différence entre une application web et une plateforme web	16
2.6. Un navigateur web.....	16
2.7. Un serveur web	17
2.8. Composant matériels et logiciels d'un serveur web	17
2.9. Fonctionnement d'un serveur web.....	17
2.10. Serveur web statique ou serveur web dynamique	18
2.10.1. Serveur web statique	18
2.10.2. Serveur web dynamique.....	19
2.11. Fonctionnalités du serveur web.....	19
CHAPITRE III : UNE APPLICATION WEB	21
3.1. Langage de programmation	21

3.1.1. HTML	21
3.1.1.1. Ecriture des attributs HTML	24
3.1.1.2. Forme d'une page HTML	24
3.1.2. CSS	25
3.1.3. JavaScript	27
3.1.3.1. Utilités du JavaScript	27
3.1.3.2.1. Fonctionnement du JavaScript	28
3.1.3.3. Moteur JavaScript	28
3.1.3.4. JavaScript côté client	28
3.1.3.5. JavaScript côté serveur	29
3.1.3.6. Framework JavaScript	29
3.2. UML (Unified Modeling Language)	30
3.2.1. Diagramme de cas d'utilisation	31
3.2.2. Diagramme de classe	32
3.2.3. Diagramme de composants	33
CHAPITRE IV : ETUDE PRATIQUE DU PROJET	37
4.1. Outils utilisés	37
4.1.1. Environnement de développement intégré (IDE)	37
4.1.2. Téléchargement et installation de Visual Studio Code	38
4.1.3. Pourquoi choisir Visual Studio Code	39
4.1.4. Langage de programmation	40
4.2. Conception d'une application MERN	41
4.2.1. React	41
a) Pourquoi choisir React	42
b) Qui utilise React	43
c) Caractéristiques de React	43
d) Avantages de l'utilisation de React	43
e) Inconvénients de l'utilisation de React	44
f) Installation de React	44
4.2.2. Node.js	45
a) C'est quoi Node.js	45
b) Caractéristiques de Node.js	46
c) Avantage de l'utilisation de Node.js	46
d) Inconvénients de l'utilisation de Node.js	47
e) Installation de Node.js	47
4.2.3. Express.js	50

a) C'est quoi Express.js	50
b) Caractéristiques d'Express.js.....	50
c) Avantages de l'utilisation d'Express.js.....	51
d) Inconvénients de l'utilisation d'Express.js.....	52
e) Installation d'Express.js	53
4.2.4. MongoDB	55
a) C'est quoi MongoDB	55
b) Avantages de l'utilisation de MongoDB	56
c) Inconvénients de l'utilisation de MongoDB	57
d) Installation de MongoDB	58
4.3. Content Management System.....	59
4.3.1. Caractéristiques et fonctionnalités d'un CMS	59
4.3.2. Strapi	60
4.3.3. Caractéristiques et fonctionnalités clés de Strapi.....	61
4.3.4. Pourquoi choisir Strapi.....	62
4.3.5. Installation de Strapi.....	63
CHAPITRE V : ESSAIS ET RESULTATS	67
5.1. Résultats	67
5.1.1. La page d'authentification	67
5.1.2. Tableau de bord de la plateforme	68
5.1.2.1. La barre de navigation.....	68
5.1.2.2. SideBar de la plateforme.....	69
5.1.2.3. Contenu de la plateforme.....	69
5.1.3. Utilisateurs	71
5.1.4. Profil	73
5.1.5. Paramètres	74
5.1.6. Mes documents	75
5.1.7. Statistique.....	75
5.1.8. A propos	76
5.1.9. Messagerie.....	77
5.1.10. Notifications	77
5.1.11. Communiqué.....	78
5.2. Bilan et recommandations	78
CONCLUSION.....	80
REFERENCE BIBLIOGRAPHIQUE ET WEBOGRAPHIQUE	81