

UNIVERSITÉ HASSAN II DE CASABLANCA

Faculté des Sciences – Département d'Informatique



PROJET DE FIN D'ÉTUDES

**Implémentation d'une solution web interactive pour le
Laboratoire d'Informatique et Systèmes (LIS)**

Réalisé par :

Anas Ouizzane

Fatiha Doula

Abdelouahid Aitsi

Encadré par :Pr. Tayed Ouaderhman

Filière :Systèmes d'Information Décisionnels (SID)

Année universitaire :2024/2025

Remerciements

Nous exprimons notre profonde gratitude à toutes les personnes qui ont contribué, de près ou de loin, à la réussite de cette plateforme pour le Laboratoire d'Informatique et Systèmes (LIS).

Nous tenons d'abord à remercier chaleureusement **Pr. Tayed Ouaderhman**, notre encadrant, pour ses conseils avisés, sa disponibilité et sa confiance tout au long du projet. Ses orientations méthodiques ont été décisives pour la conception et la mise en œuvre de notre solution.

Nos remerciements s'adressent également aux **membres du jury**, qui ont accepté d'évaluer notre travail et dont les observations enrichiront la version finale de ce rapport.

Nous remercions enfin nos **familles** et nos **amis** pour leur soutien moral inconditionnel, leur patience et leurs encouragements constants, indispensables à la bonne poursuite de cette aventure académique et humaine.

Grâce à l'implication et à l'appui de chacun, nous avons transformé notre idée initiale en un outil concret qui, nous l'espérons, contribuera durablement au rayonnement du laboratoire.

INTRODUCTION GÉNÉRALE:

À l'ère de la révolution numérique, enseignement supérieur, recherche scientifique et communication institutionnelle se digitalisent à grande vitesse. Les laboratoires universitaires, pôles d'innovation par excellence, doivent donc moderniser la gestion de leurs équipes, publications et événements afin de rester compétitifs et visibles.

Or, la coexistence de tableurs, dossiers partagés et outils disparates fragmente l'information : articles, annonces, listes d'équipes et calendriers d'événements se trouvent dispersés, ce qui ralentit la recherche de données fiables et nuit à la valorisation des travaux.

Pour remédier à cette situation, notre projet de fin d'études consiste à concevoir **une plateforme web unifiée** pour le Laboratoire d'Informatique et de Sciences (LIS). Cette solution offrira :

- **Centralisation** : une base unique regroupant membres, articles, actualités et événements ;
- **Validation sécurisée** : un flux clair — *membre* → *responsable* → *directeur* — garantissant traçabilité et fiabilité ;
- **Accès facilité aux ressources** : moteur de recherche transversal, filtres par équipe ou thématique, et téléchargements directs (PDF, jeux de données, supports de conférence) depuis un même tableau de bord ;

- **Diffusion efficace** : un site vitrine public couplé à des espaces privés adaptés aux rôles internes.

Ainsi, le LIS disposera d'un socle numérique robuste, améliorant l'efficacité quotidienne et renforçant la visibilité externe du laboratoire.

Ce document est structuré pour refléter les étapes clés du projet :

1. **Analyse de l'existant** : état des lieux, besoins fonctionnels et non fonctionnels ;
2. **Conception** : choix d'architecture, modélisation UML et schéma de données ;
3. **Implémentation** : technologies retenues (React, Node.js/Express, MongoDB), structure du code et modules développés ;
4. **Tests et validation** : scénarios de test, résultats et ajustements ;

Cette progression guide le lecteur pas à pas, depuis le contexte initial jusqu'aux réalisations concrètes et aux perspectives futures du projet.

Table des matières

Introduction générale.....	2
Chapitre 1 – Contexte général du projet.....	6
1. Introduction.....	7
2. Problématique.....	7
3. Objectifs et missions.....	8
4. Identification des besoins.....	9
4.1 Besoins fonctionnels.....	9
4.2 Besoins non fonctionnels.....	11
5. Planification du proje.....	12
6. Conclusion.....	13
Chapitre 2 – Conception et modélisation.....	14
1. Introduction.....	15
2. Identification des acteurs.....	16
3. Diagramme de cas d'utilisation.....	17
4. Diagramme de classes.....	20
5. Modèle conceptuel de données (MCD)	22
6. Diagrammes de séquence.....	22
6.1 Authentification.....	24

6.2 Ajouter un utilisateur.....	25
6.3 Ajouter un membre à l'équipe.....	26
6.4 Publication d'un article.....	28
7. Conclusion.....	29
Chapitre 3 – Implémentation.....	30
1. Introduction.....	31
2. Backend.....	31
2.1 Architecture MVC.....	32
2.2 Base de données MongoDB.....	33
3. Frontend.....	34
4. Outils et environnement de développement.....	35
5. Conclusion.....	37
Chapitre 4 – Réalisation.....	38
Introduction	39
Aperçu des interfaces (captures d'écran).....	
Conclusion générale.....	52

Chapitre 1:

Contexte général du projet

1. Introduction

Le Laboratoire d’Informatique et de Sciences (LIS), rattaché à la Faculté des sciences aïn chock - Casablanca, rassemble plusieurs équipes de recherche, d’enseignement et d’innovation. Comme la plupart des structures académiques marocaines, il souhaite s’inscrire pleinement dans la dynamique nationale de transformation numérique afin de valoriser ses productions scientifiques, fluidifier ses processus internes et accroître sa visibilité.

2. Problématique

Aujourd’hui, l’information du laboratoire est éclatée :

- articles, actualités et événements circulent entre tableurs, dossiers partagés, réseaux sociaux et e-mails ;
- chaque acteur (directeur, responsables d’équipe, membres) utilise ses propres outils et nomenclatures ;
- aucune traçabilité fiable n’existe pour savoir qui a soumis quoi, quand et avec quelle version.

Cette fragmentation de l’information, couplée à des solutions de recherche peu intuitives, complique l’accès aux ressources et ralentit la valorisation

des travaux . Les chercheurs et étudiants se heurtent ainsi à des délais d'obtention ou de validation de données qui freinent leurs projets.

3. Objectifs et missions du projet

Objectif global

Mettre en place pour le Laboratoire d'Informatique et de Sciences (LIS) une plateforme web unique qui centralise l'ensemble des ressources scientifiques et administratives, fluidifie les processus internes et renforce la visibilité externe du laboratoire.

Missions concrètes pour atteindre cet objectif :

- Centralisation de l'information**

Rassembler, dans une même base de données, les fiches membres, articles, actualités et événements, afin d'en finir avec la dispersion entre tableurs, dossiers partagés et réseaux sociaux ; principe déjà éprouvé sur d'autres plateformes académiques où le stockage centralisé simplifie la recherche et la consultation des documents .

- Mise en place d'un circuit de validation à trois niveaux**

Formaliser le flux *Membre* → *Responsable* → *Directeur* : chaque contenu soumis est vérifié, puis approuvé ou refusé, garantissant traçabilité et fiabilité avant toute mise en ligne, à l'image des workflows décrits dans les rapports de projets similaires .

- **Accès facilité aux ressources**

Offrir un moteur de recherche transversal et des filtres par équipe, auteur ou thème, ainsi que le téléchargement direct de fichiers validés pour encourager la réutilisation et la citation des travaux du LIS .

- **Diffusion et communication**

Déployer un site vitrine public, optimisé pour le référencement, afin de promouvoir les activités du laboratoire, tandis que des tableaux de bord privés fournissent aux acteurs internes un suivi en temps réel des soumissions, validations et statistiques de consultation.

4. Identification des besoins

L'objectif de cette section consiste à présenter les différentes exigences fonctionnelles du projet.

4.1 Besoins fonctionnels

- **Authentification et rôles** — chaque utilisateur (Directeur, Responsable, Membre) dispose d'un compte protégé par e-mail / mot de passe ; le Visiteur accède librement aux pages publiques. Ce principe d'accès contrôlé s'inspire des exemples d'applications analysés, où l'authentification constitue la première exigence métier .

- **Gestion des utilisateurs (Directeur)**

- créer, modifier et supprimer les comptes ;

- associer un rôle, une spécialité, un avatar ;
- empêcher tout doublon d'adresse e-mail.

- **Module Équipes**

- créer une équipe, désigner un leader, téléverser un logo ;
- ajouter ou retirer des membres existants ;
- afficher publiquement chaque équipe sous forme de carte responsive.

- **Module Articles**

- soumission par un Membre (statut *En attente*) ;
- validation ou refus par le Responsable ;
- publication finale par le Directeur ;
- archivage des versions jusqu'à la mise en ligne.

- **Actualités & Événements** — mêmes étapes de workflow que pour les articles ; prise en charge d'une image, d'un lieu, d'une date et d'un lien de diffusion.
- **Recherche et filtres** — moteur de recherche transversal (mots-clés, équipe, auteur, période) avec filtres dynamiques, suivant le besoin d'accès rapide à l'information déjà souligné dans les projets de référence .
- **Notifications** — alertes e-mail et in-app déclenchées lors de chaque soumission, validation, refus ou changement de rôle.
- **Téléversement sécurisé** — dépôt d'images et de PDF avec contrôle du type et de la taille ; mise à disposition publique uniquement après validation.
- **Tableaux de bord** — vue personnalisée pour chaque rôle : files d'attente, statistiques de contribution, indicateurs de fréquentation.

4.2 Besoins non fonctionnels

- **Sécurité** — chiffrement TLS, hashage des mots de passe, gestion fine des autorisations ; déconnexion automatique après inactivité, conformément aux recommandations relevées dans les rapports d'étude .
- **Performance** — Afin de garantir une réactivité optimale même en période de forte charge, le système doit bénéficier d'une accélération côté serveur. Cela inclut :
 - l'utilisation d'un serveur Node.js non bloquant (event loop),

- l'optimisation des requêtes vers la base de données MongoDB (indexation, pagination, projections),
 - le caching stratégique des réponses fréquentes,
 - et la compression des réponses HTTP.
 - Ces optimisations assurent un temps de réponse rapide même avec une montée en charge, et contribuent à respecter l'objectif de performances fixé à moins de 300 ms par requête API.
-
- **Ergonomie & accessibilité** — interface responsive, navigation claire, respect des bonnes pratiques ergonomiques extrapolées des documents étudiés .
 - **Compatibilité** — prise en charge des principaux navigateurs (Chrome, Firefox, Edge) et des écrans mobiles, exigence déjà mise en avant pour la plateforme de thèses .
 - **Maintenabilité** — code modulaire, documentation interne, suites de tests automatisés et pipeline CI/CD.

4. Planification du projet

Le diagramme de Gantt ci-dessous offre une vue d'ensemble du calendrier du projet : il aligne chronologiquement les principales tâches depuis l'analyse des besoins jusqu'à la préparation de la soutenance, en matérialisant pour chacune la date de début, la durée et les chevauchements éventuels.

La progression horizontale des barres permet d'identifier les phases menées en parallèle (ex. développement front-end et back-end) et les points charnières qui conditionnent la suite (validation UML/MCD, intégration front-back, tests). Ce visuel constitue donc la référence temporelle qui a guidé le pilotage du PFE.

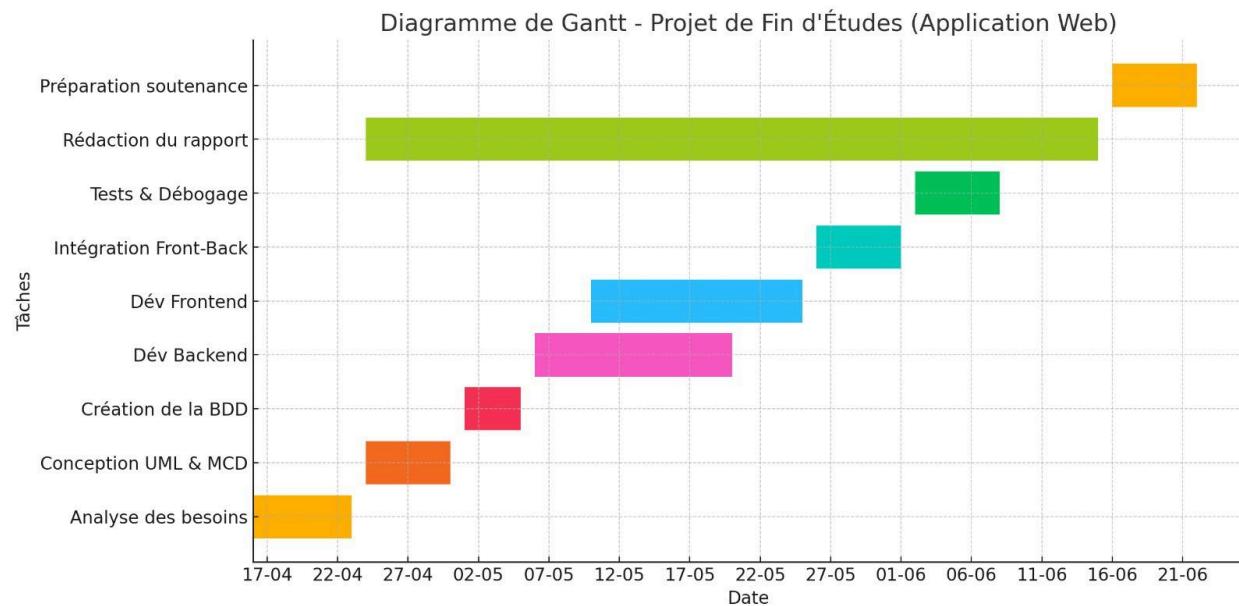


Figure 1: Diagramme de Gantt

5. Conclusion

Après avoir dressé l'état des lieux, identifié les problèmes de fragmentation de l'information et défini les besoins fonctionnels et non fonctionnels, ce chapitre a permis de préciser le périmètre exact du projet. Nous disposons désormais d'un cadre clair : centraliser les données du laboratoire, instaurer un circuit de validation rigoureux et offrir une interface accessible aux différents rôles. Ces constats guident l'ensemble des étapes suivantes et justifient les choix technologiques et méthodologiques adoptés.

Chapitre 2:

Conception et modélisation du
projet

1. Introduction

Pour transformer des exigences encore conceptuelles en une architecture logicielle solide, il nous faut un langage visuel partagé : l'**UML (Unified Modeling Language)**. Dans ce chapitre, nous mobilisons ses principaux diagrammes afin de couvrir chaque facette du système :

- **Diagrammes de cas d'utilisation** : ils cadrent les fonctionnalités et montrent, côté métier, comment chaque acteur interagit avec l'application.
- **Diagrammes de séquence** : ils déroulent, étape par étape, les échanges entre acteurs et composants, révélant la logique interne et les points de synchronisation.
- **Diagramme de classes** : il décrit la structure statique du domaine — entités, attributs et associations — et sert de passerelle directe vers le code.
- **Modèle conceptuel des données (MCD)** : il affine la vision des classes orientées objet en un schéma orienté données, préparant ainsi la base MongoDB.

En combinant ces vues, l'UML joue trois rôles essentiels :

1. **Clarifier** : il rend des exigences parfois abstraites parfaitement lisibles par tous.
2. **Aligner** : il garantit que développeurs, décideurs et utilisateurs partagent la même compréhension du périmètre.
3. **Guider** : il sert de feuille de route technique pour le développement, les tests et la maintenance, limitant les risques de dérive par rapport aux objectifs initiaux.

Grâce à cette approche, chaque partie prenante dispose d'une référence claire et consensuelle ; les futures implémentations resteront donc cohérentes, traçables et évolutives.

2. Identification des acteurs

Dans la modélisation UML, **un acteur** représente toute entité externe (personne, groupe ou système) qui interagit avec l'application ; il est défini par ses objectifs, non par sa fonction interne au logiciel.

Les acteurs décrits ci-dessous incarnent donc les rôles clés qui exploitent ou administrent la plateforme du LIS.

Visiteur – Internaute non authentifié. Il consulte librement actualités, événements, équipes et articles déjà publiés afin de s'informer sur les activités et les travaux du laboratoire.

Membre – Chercheur, doctorant ou étudiant inscrit. Depuis son tableau de bord, il soumet de nouveaux articles (statut *en attente*), suit l'avancement de ses contributions et met à jour son profil pour accroître sa visibilité scientifique.

Responsable – Leader d'équipe désigné par le directeur. Dans son espace privé, il valide ou refuse les articles de son groupe, gère l'appartenance des membres (ajout / retrait) et consulte les statistiques de production pour garantir la qualité et dynamiser la recherche de son équipe.

Directeur – Administrateur global du laboratoire. Il dispose des droits les plus étendus : création, modification et suppression des comptes, approbation finale des contenus et supervision des indicateurs. Son rôle est d'assurer la gouvernance, la cohérence et la visibilité externe du LIS.

3. Diagramme de Cas d'Utilisation

Un **diagramme de cas d'utilisation** (Use Case Diagram) décrit, de façon visuelle et très synthétique, les fonctionnalités majeures qu'un système offre à ses utilisateurs et la manière dont ces derniers interagissent avec lui.

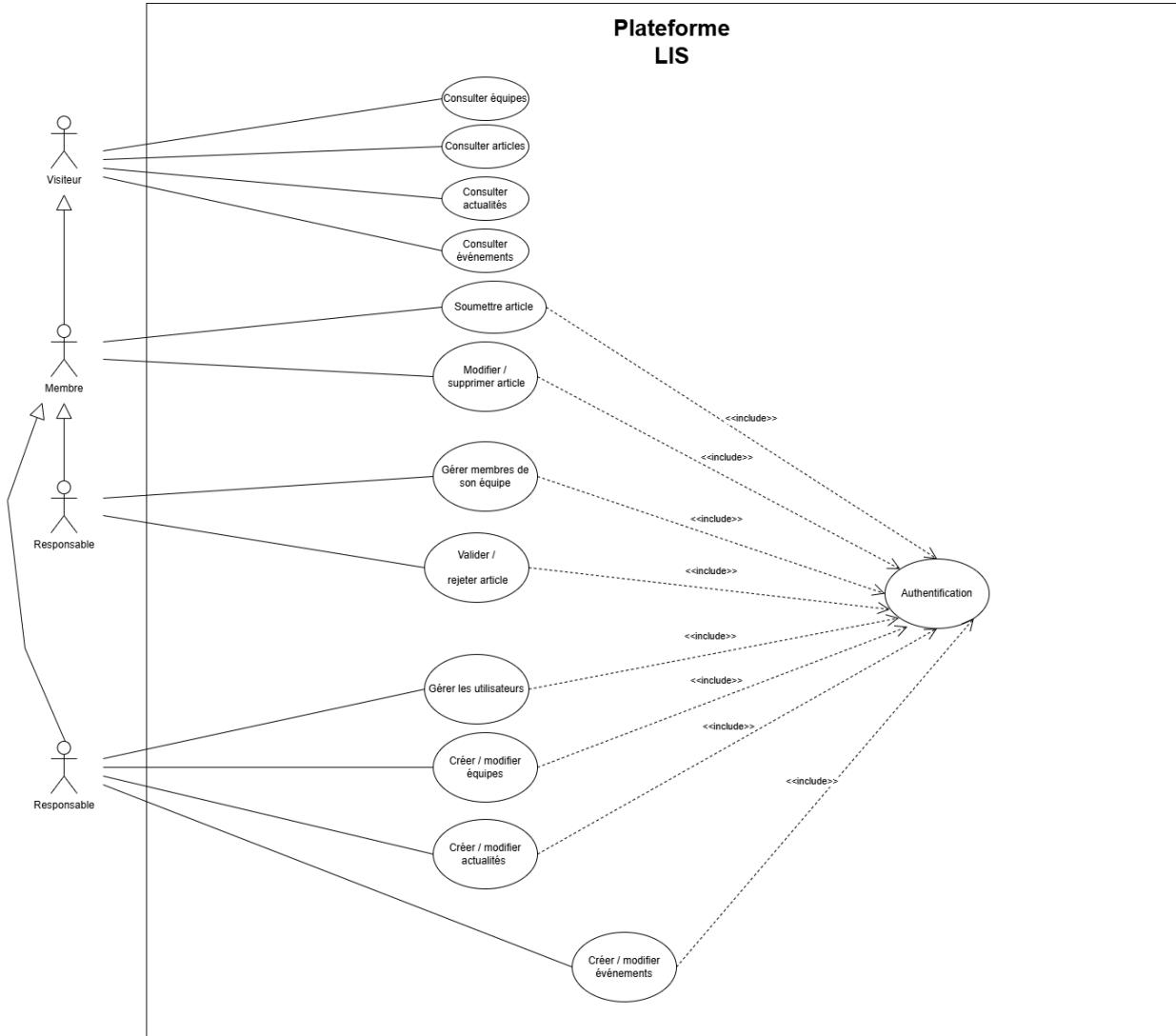


Figure 2: diagramme de cas d'utilisation

Le diagramme de cas d'utilisation constitue un outil fondamental de modélisation fonctionnelle. Il permet de représenter, de manière synthétique et lisible, les **fonctionnalités principales du système** ainsi que les **rôles des différents utilisateurs** de la plateforme.

Dans notre projet, ce diagramme met en évidence les différentes **interactions possibles entre les acteurs et la plateforme LIS**. Chaque

acteur dispose d'un ensemble de cas d'utilisation qui reflètent ses **droits et responsabilités** spécifiques :

- Le **Visiteur** peut uniquement consulter les pages publiques telles que les équipes, les actualités, les événements et les articles validés. Il ne peut ni modifier, ni interagir avec les contenus.
- Le **Membre** peut soumettre des articles, suivre leur statut de validation, modifier son profil, mais n'a aucun pouvoir de validation ou de publication.
- Le **Responsable** a la possibilité de gérer les membres de son équipe et de valider ou refuser les articles qui lui sont soumis. Il joue un rôle de filtre avant la validation finale.
- Le **Directeur** possède les droits les plus étendus : il peut gérer tous les utilisateurs, créer et modifier des équipes, approuver les publications et consulter les statistiques globales de la plateforme.

Chaque **cas d'utilisation** correspond à une **fonctionnalité métier essentielle**, par exemple :

- « Créer un utilisateur »
- « Soumettre un article »
- « Valider une publication »
- « Gérer une équipe »
- « Consulter les statistiques »

Une relation particulière, appelée <<include>>, est utilisée dans le diagramme pour indiquer qu'un **cas d'utilisation est inclus automatiquement dans un autre**. Cela signifie que certaines actions nécessitent obligatoirement l'exécution préalable d'un processus commun.

Par exemple, des actions comme « Soumettre un article », « Valider une publication » ou « Gérer une équipe » incluent toutes le cas « Authentification ». Cela signifie qu'un utilisateur doit être connecté avant d'accéder à ces fonctionnalités. L'inclusion explicite de ce cas d'utilisation permet d'**éviter la redondance et de mettre en évidence les étapes obligatoires** dans l'enchaînement logique des actions.

3. Diagramme de classe

Le **diagramme de classes** traduit en image la structure logique d'un logiciel : il montre quelles classes existent, quelles données elles possèdent (attributs) et quels comportements elles offrent (méthodes). Chaque ligne entre deux classes exprime un type de lien :

- **Association** : simple relation métier
- **Héritage** : une classe fille hérite du code et des responsabilités de la classe mère.
- **Agrégation / Composition** : relation contenant-contenu, plus ou moins forte selon que les parties peuvent vivre sans l'ensemble.

- **Dépendance** : une classe en utilise une autre de façon ponctuelle (paramètre, retour de méthode...).

En phase de conception, ce diagramme sert à fixer le **langage commun** de l'équipe : il clarifie les entités clés, leurs responsabilités et leurs interactions, avant même d'écrire la première ligne de code.

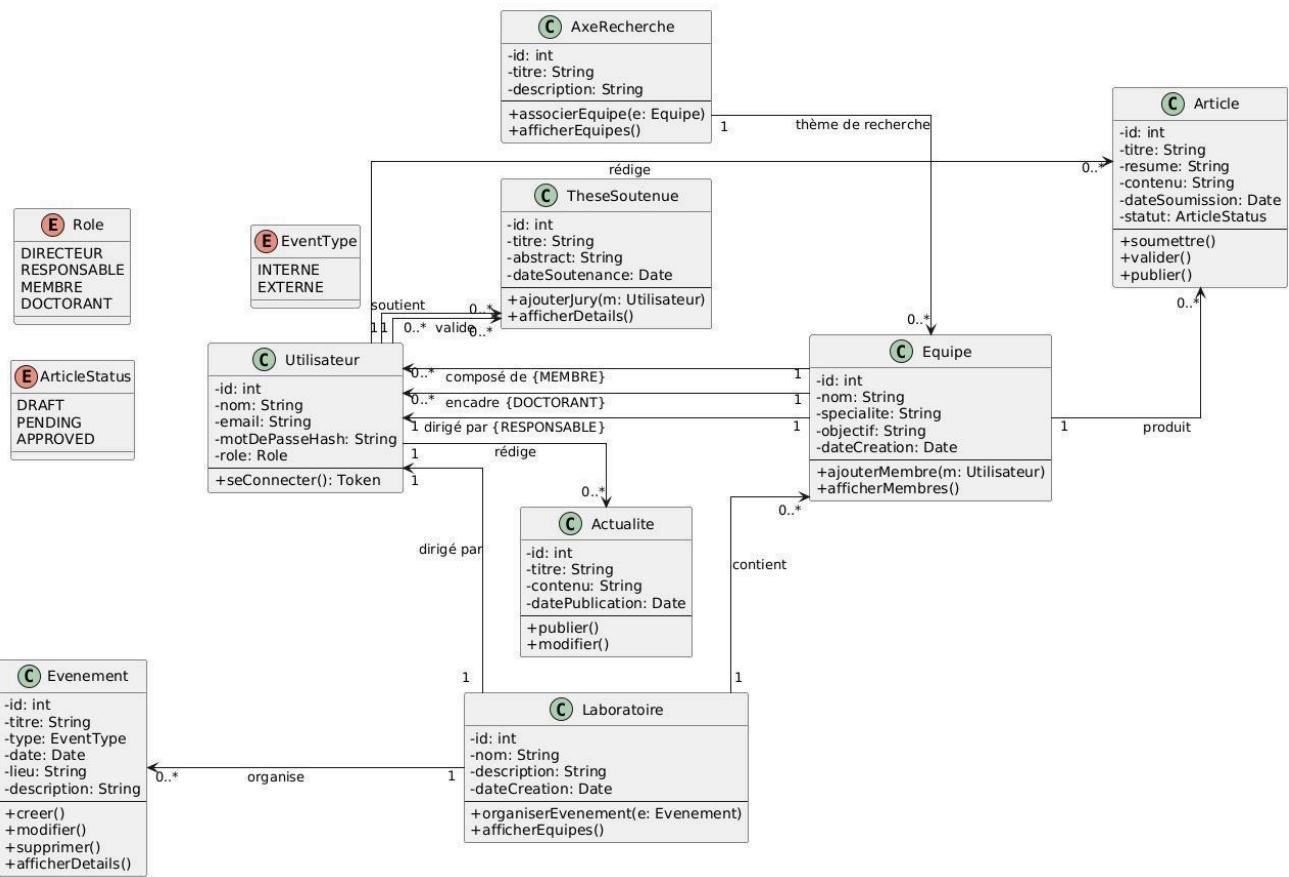


Figure 3: diagramme de classes

4. Modèle conceptuel de donnée

Le **Modèle Conceptuel de Données (MCD)** est la représentation la plus abstraite d'une base de données ; il décrit ce que l'on veut stocker, sans se préoccuper de comment cela sera implémenté. On l'utilise principalement dans la méthode Merise (mais l'idée existe aussi dans d'autres démarches).

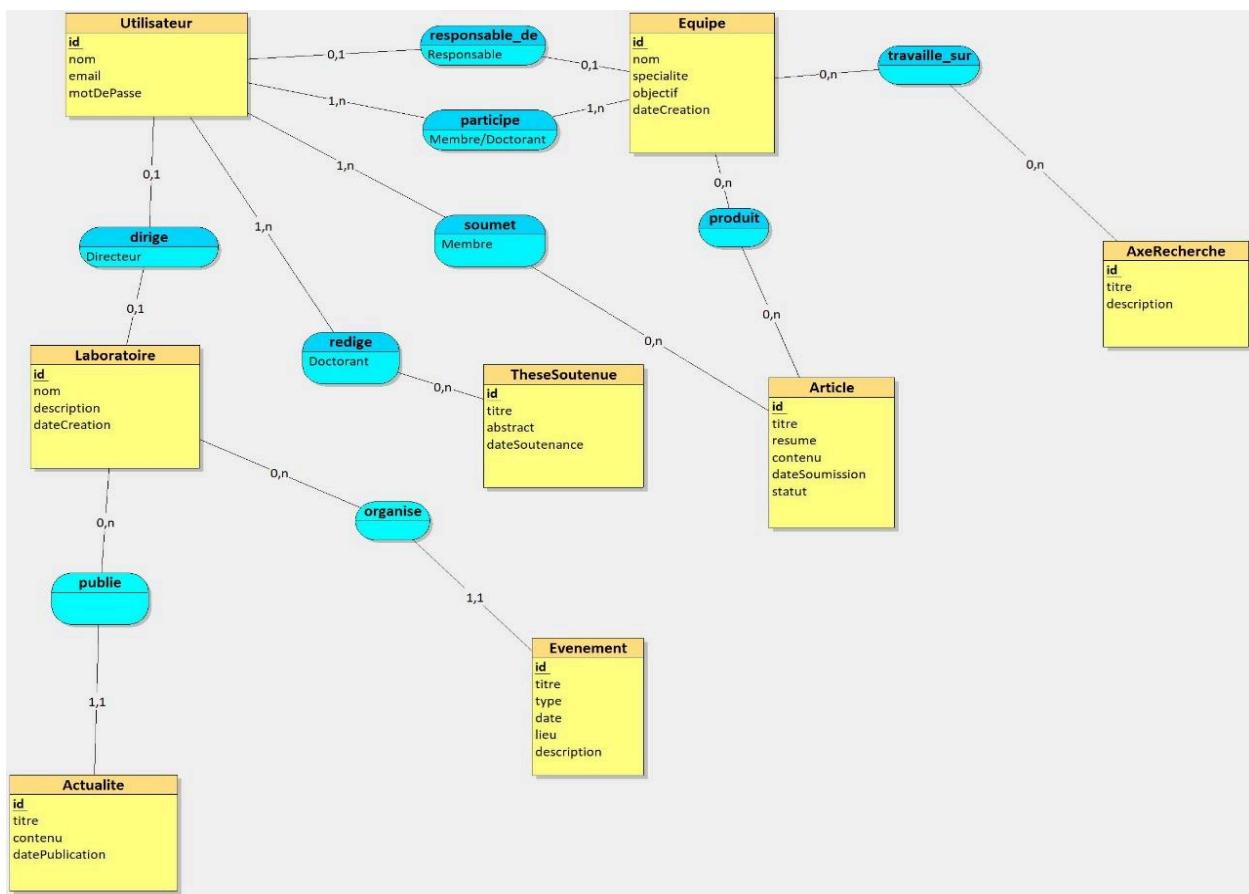


Figure 4: Modèle Conceptuel de Données (MCD)

4. Diagramme de séquence

Un **diagramme de séquence** met en scène, dans UML, l'enchaînement détaillé d'un scénario. Il dispose les participants (acteurs, objets, services)

en colonnes verticales appelées lignes de vie ; l'axe vertical figure l'écoulement du temps, tandis que l'horizontale matérialise les messages. Lorsqu'un participant exécute une action, un rectangle d'activation se superpose à sa ligne ; quand il émet ou reçoit une requête, une flèche part de sa colonne vers celle du destinataire. On peut aussi indiquer l'instant où un objet apparaît (flèche « create ») ou disparaît (croix « destroy »).

Au-delà de la simple figuration graphique, ce type de diagramme sert à exposer **la chronologie experte d'un cas d'utilisation** : il dévoile l'ordre exact des appels, les dépendances temporelles entre participants et la circulation des données ou des confirmations. Grâce à cette vue pas-à-pas, toutes les parties prenantes – des analystes fonctionnels aux développeurs – disposent d'une référence visuelle pour vérifier que la logique métier est respectée avant de passer à l'implémentation. Le diagramme aide ainsi à détecter les points sensibles : appels synchrones susceptibles de bloquer un processus, boucles d'attente, messages concurrents, créations ou destructions prématurées d'objets. Il devient un support privilégié pour les revues de conception, la rédaction de tests et la communication avec les équipes extérieures (interfaces, micro-services ou partenaires).

4.1 Diagramme de séquence « authentification »

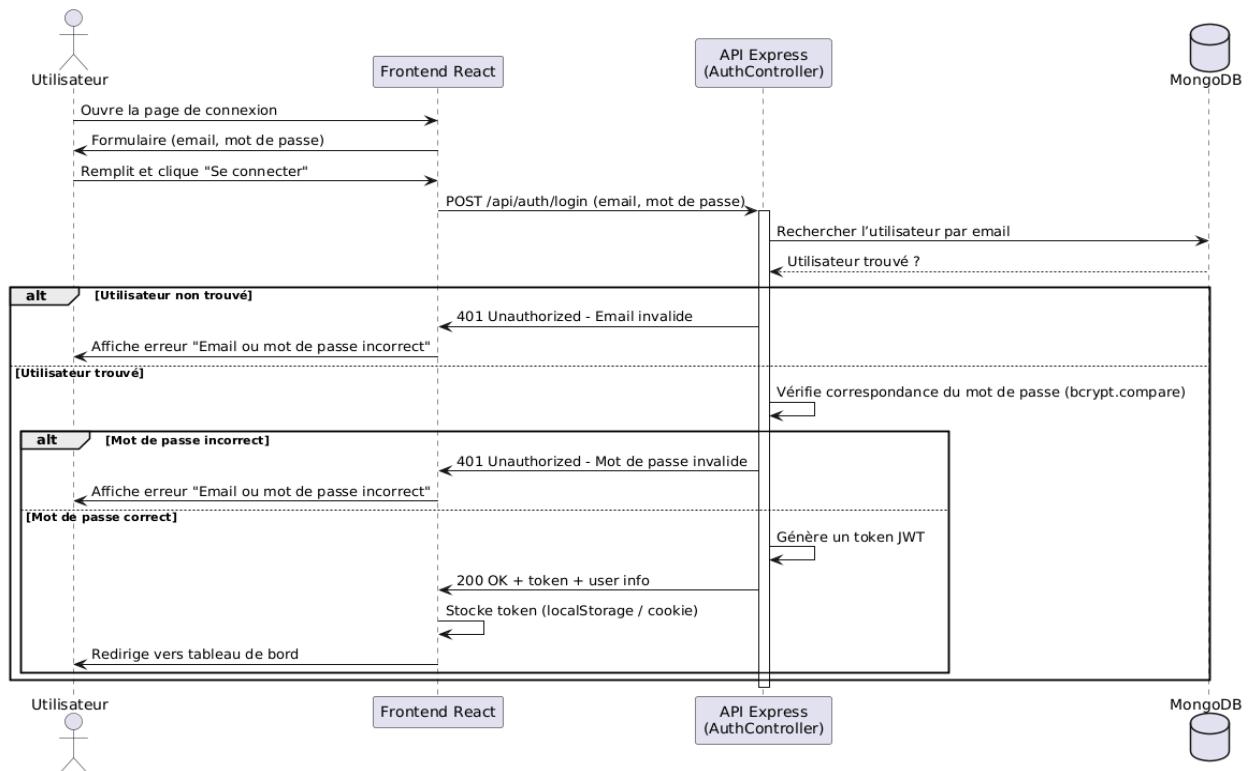


Figure 5: Diagramme de séquence « authentification »

Le diagramme de séquence « **Authentification** » illustre l'enchaînement des échanges lors d'une connexion : l'utilisateur soumet son email et son mot de passe via le frontend React, lequel envoie la requête à l'API Express. L'API interroge MongoDB ; si le compte est introuvable ou si le mot de passe est incorrect, elle renvoie un code 401 et le frontend affiche l'erreur. Sinon, l'API génère un JWT, renvoie un 200 avec le jeton et les infos utilisateur ; le frontend stocke le token et redirige l'utilisateur vers son tableau de bord.

4.2 Diagramme de séquence « Ajouter un utilisateur »

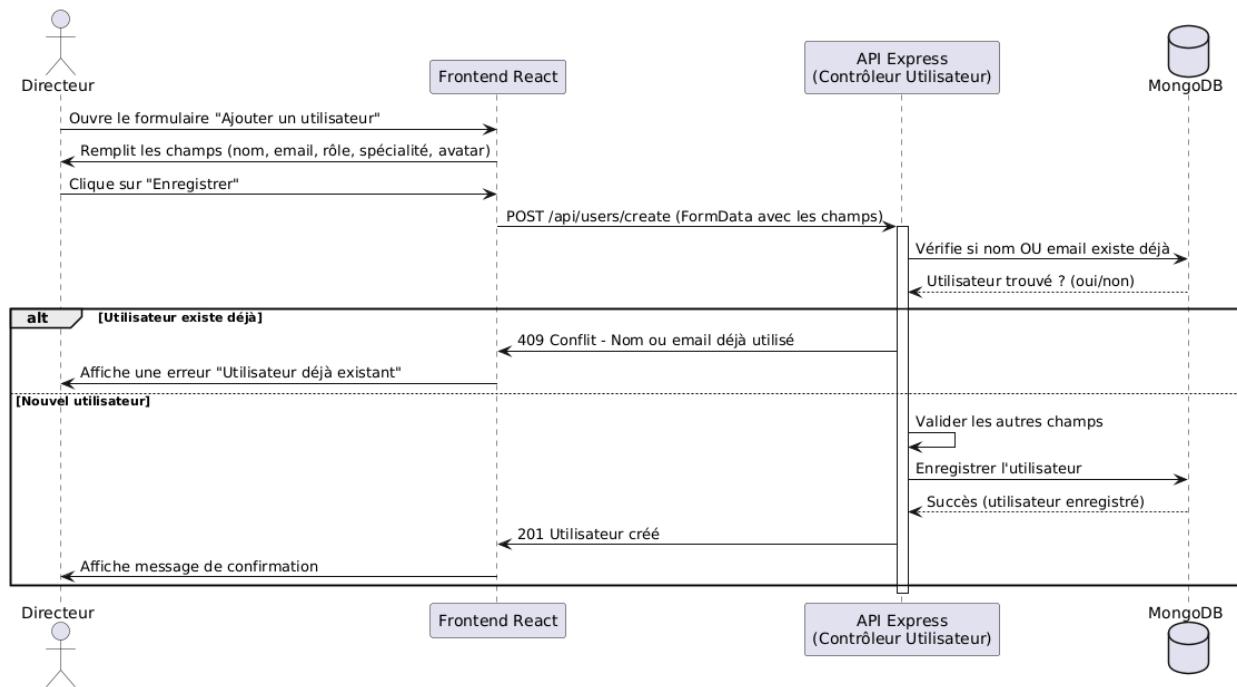


Figure 6: Diagramme de séquence « Ajouter un utilisateur »

Le diagramme de séquence « Ajouter un utilisateur » retrace la création d'un compte par le Directeur. D'abord, celui-ci ouvre le formulaire d'ajout sur le frontend React, saisit le nom, l'email, le rôle, la spécialité et l'avatar, puis clique sur Enregistrer. Le frontend envoie alors une requête POST /api/users/create au contrôleur Utilisateur de l'API Express. L'API interroge MongoDB pour vérifier si le nom ou l'email existe déjà : si c'est le cas, elle renvoie un 409 Conflit, et le frontend affiche l'erreur « Utilisateur déjà existant ». Si aucun doublon n'est trouvé, l'API valide les autres champs, enregistre le nouvel utilisateur dans la base et renvoie un **201 Utilisateur créé **. Le frontend affiche alors un message de confirmation au Directeur.

Ainsi, le diagramme illustre clairement le flux principal de création ainsi que la branche alternative de refus en cas de doublon.

4.3 Diagramme de séquence « Ajouter un membre à l'équipe »

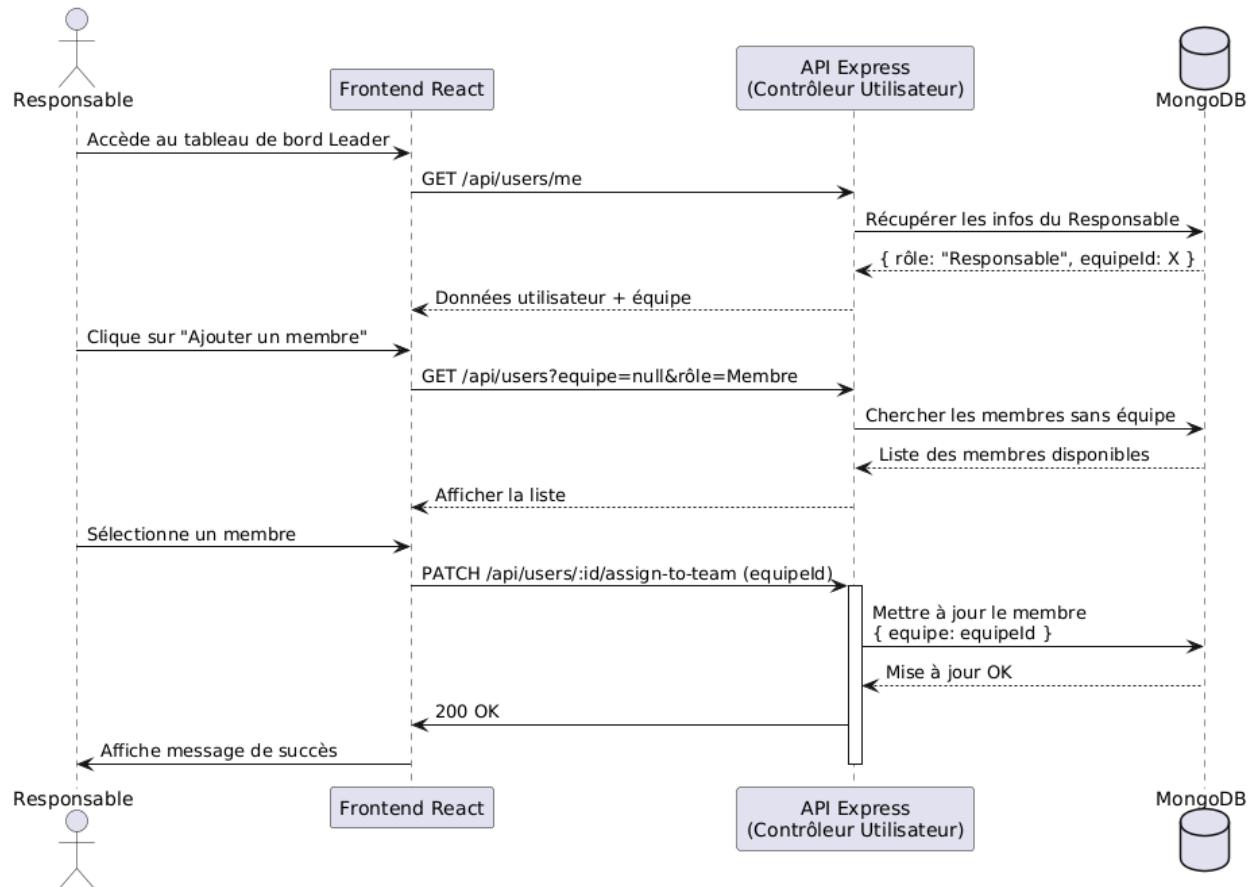


Figure 7: Diagramme de séquence « Ajouter un membre à l'équipe »

Le diagramme « **Ajouter un membre à l'équipe** » montre comment un Responsable rattache un Membre disponible à son groupe :

1. Dès qu'il ouvre son tableau de bord, l'interface récupère son profil afin de connaître l'identifiant de son équipe.

2. Lorsqu'il clique sur « Ajouter un membre », l'application demande à la base de données la liste des utilisateurs qui ne sont encore liés à aucune équipe.
3. Cette liste s'affiche à l'écran. Le Responsable choisit la personne à intégrer.
4. L'application met alors à jour la fiche du Membre en y inscrivant l'identifiant de l'équipe du Responsable.
5. Une confirmation apparaît : le rattachement a bien été effectué.

Le diagramme met ainsi en lumière trois temps forts : obtention du contexte du Responsable, sélection d'un Membre libre, puis enregistrement de ce Membre dans l'équipe.

4.4 Diagramme de séquence « Publication d'un article »

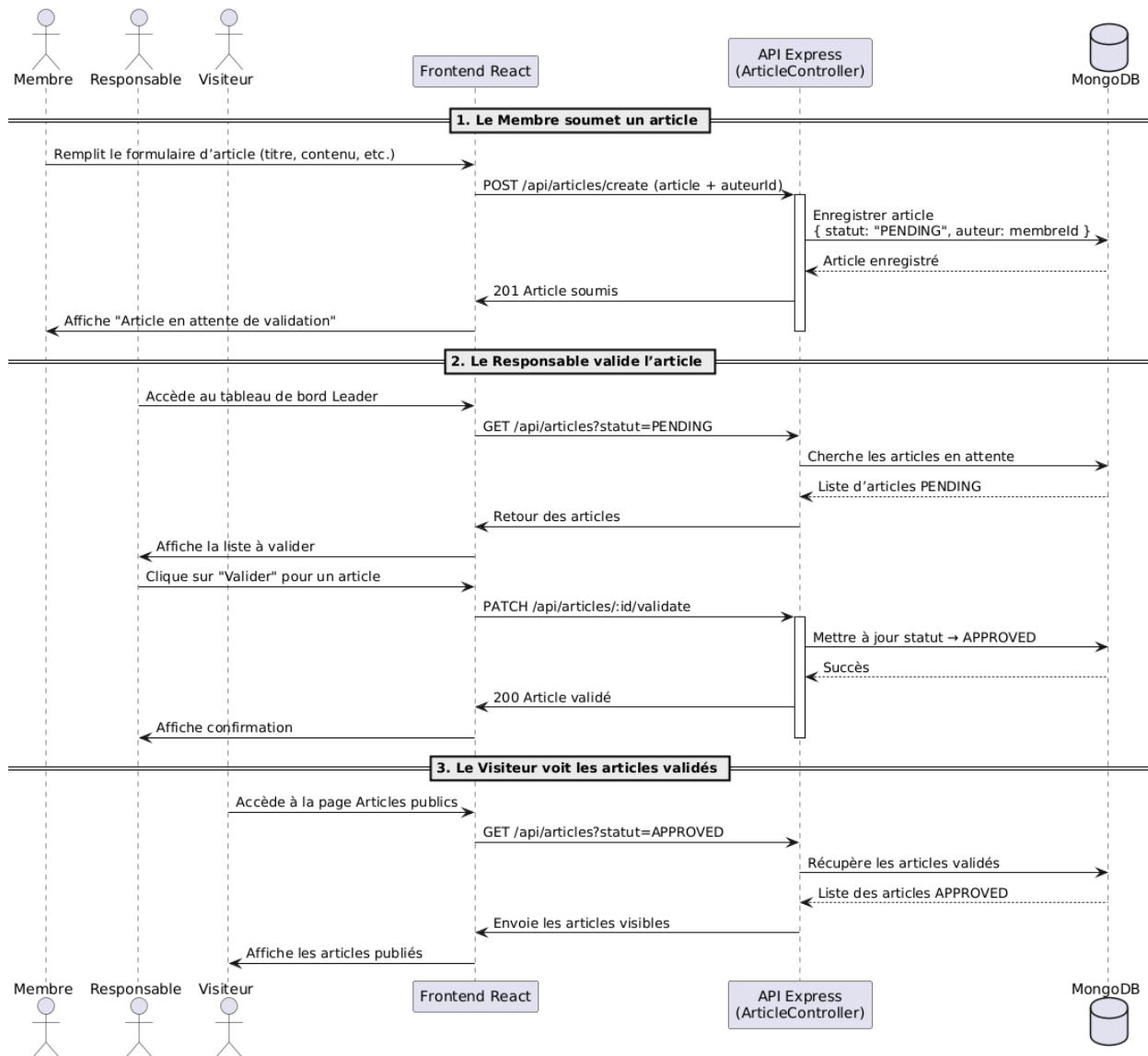


Figure 8: Diagramme de séquence « Publication d'un article »

Le diagramme « Publication d'un article » illustre le cycle complet de publication d'un article.

- Soumission** : un Membre remplit le formulaire, envoie son texte au système ; l'article est enregistré avec le statut « En attente de

validation », et le Membre voit aussitôt un message confirmant que son contenu reste à approuver.

2. **Validation** : dans son tableau de bord, le Responsable affiche la liste des articles en attente, choisit celui du Membre et déclenche l'action « Valider ». Le système change alors son statut en « Approuvé » et renvoie une confirmation.

3. **Consultation publique** : un Visiteur qui ouvre la page des articles ne reçoit que ceux marqués « Approuvé » ; la liste visible ne contient donc que les contenus déjà validés par un Responsable. Ainsi, le diagramme met en évidence la transition de l'article depuis la création jusqu'à la mise en ligne, en passant par la validation obligatoire.

5. Conclusion

La phase de conception a posé les fondations du système grâce aux différents diagrammes UML et au MCD. Les cas d'utilisation ont éclairé les fonctionnalités attendues, les diagrammes de séquence ont détaillé le déroulement des scénarios clés, tandis que le diagramme de classes et le modèle conceptuel de données ont fixé la structure interne. Cette modélisation garantit une compréhension partagée entre les parties prenantes et sert de feuille de route fiable pour le développement, les tests et la maintenance.

Chapitre 3:

Implémentation

1. Introduction

Après avoir validé les modèles conceptuels et les diagrammes de conception, ce chapitre présente la réalisation concrète de la plateforme **LIS**. L'application s'appuie sur une **architecture full JavaScript** : un backend **Node.js / Express** expose une API REST sécurisée (JWT, contrôle d'accès par rôles), tandis qu'un frontend **React** assure une interface réactive pour les Directeurs, Responsables, Membres et Visiteurs. Les données sont persistées dans **MongoDB**, dont la souplesse du schéma facilite l'évolution des modules : actualités, événements, équipes, articles, thèses, et gestion des utilisateurs. Nous décrirons successivement : la configuration du serveur Express, la mise en place des modèles Mongoose, l'implémentation des principaux endpoints, puis l'intégration côté React (routes, contextes d'authentification, composants dédiés). Enfin, nous terminerons par les scripts de déploiement et les tests qui valident la stabilité et la sécurité de la plateforme.

Backend:



Node.js, créé par Ryan Dahl en 2009, s'appuie sur le moteur V8 pour exécuter JavaScript côté serveur. Sa boucle d'événements non bloquante gère de nombreuses requêtes simultanées, idéale

pour des API rapides. Grâce à l'unification du langage entre frontend et backend et à l'écosystème riche de NPM (JWT, uploads, tests), il accélère le développement tout en garantissant la fiabilité.

Express est un micro-framework web pour Node.js, conçu par TJ Holowaychuk en 2010. Minime et non prescriptif, il fournit un routage clair et une architecture de middlewares chaînables qui permettent d'ajouter authentification, validation ou journalisation à la demande, sans alourdir le cœur de l'application. Cette souplesse accélère la création d'API REST et de services web réactifs tout en laissant au développeur la liberté de structurer son projet selon ses besoins.

Pour assurer une organisation claire et maintenable du code backend, nous avons adopté l'architecture **MVC** (*Modèle – Vue – Contrôleur*). Ce modèle de structuration est largement utilisé dans les projets web modernes car il permet de séparer les responsabilités en trois couches distinctes :

- **Le Modèle** représente la structure des données. Il définit les entités principales de l'application, comme les utilisateurs, les équipes ou les articles. C'est à ce niveau que sont décrits les attributs, les relations entre entités et les règles de validation des données.
- **Le Contrôleur** joue le rôle d'intermédiaire entre les demandes du client et le modèle de données. Il reçoit les requêtes, applique la logique métier (comme la vérification des rôles ou des doublons), interagit avec le modèle, puis renvoie une réponse adaptée.

- **La Vue**, dans le contexte d'une API REST, correspond aux **réponses JSON** envoyées au frontend. Ce ne sont pas des pages HTML, mais des données structurées que le frontend utilise pour afficher dynamiquement les informations à l'écran.

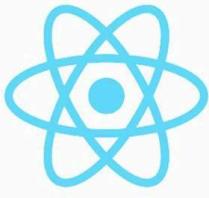
L'utilisation du modèle MVC facilite la lisibilité du code, permet de modifier une partie du système sans impacter les autres, et rend l'application plus évolutive. Grâce à cette organisation, nous avons pu gérer efficacement des modules complexes tels que les utilisateurs, les validations ou encore les tableaux de bord, tout en assurant une séparation claire entre les données, la logique métier et les interfaces d'échange.



Pour la gestion de la base de données, nous avons opté pour **MongoDB**, une base de données NoSQL orientée documents.

Grâce à son modèle flexible basé sur les documents JSON, MongoDB nous a permis de stocker et manipuler les données de manière intuitive et adaptée aux besoins évolutifs de notre application. L'intégration avec Mongoose, un ODM pour Node.js, nous a facilité la définition de schémas, la validation des données et les opérations complexes. Cette solution s'est révélée particulièrement efficace pour la manipulation rapide des articles, utilisateurs, équipes et événements, tout en assurant des performances optimales et une grande évolutivité.

Frentend:



React

Pour le développement de l'interface utilisateur, nous avons utilisé React, une bibliothèque JavaScript créée par Meta (Facebook). Grâce à son système de composants réutilisables et à sa gestion efficace du DOM virtuel, React nous a permis de construire une interface dynamique, fluide et facile à maintenir. La séparation entre la logique métier, l'affichage et les états nous a aidés à structurer notre code de manière claire et modulaire. De plus, React s'intègre parfaitement avec notre backend en Node.js, facilitant la communication avec l'API et le rendu des données en temps réel.

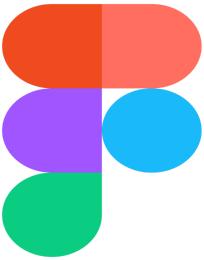


Pour le côté design de notre interface, nous avons utilisé **Tailwind CSS**, un framework utilitaire moderne qui permet de styliser directement les composants via des classes prédéfinies. Contrairement aux approches classiques basées sur des fichiers CSS séparés, Tailwind favorise une écriture rapide, lisible et cohérente du style directement dans le code JSX. Grâce à sa flexibilité, nous avons pu créer une interface responsive, épurée et uniforme sans réécrire de règles CSS personnalisées. Ce choix a considérablement simplifié la mise en page et accéléré notre travail de développement front-end.

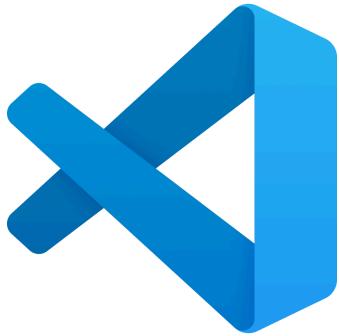


Pour accélérer le développement et optimiser les performances du frontend, nous avons utilisé **Vite** comme outil de build. Conçu par **Evan You**, créateur

de Vue.js, Vite offre un environnement de développement rapide grâce au rechargement à chaud (Hot Module Replacement) et à une compilation ultra-rapide. Contrairement aux bundlers traditionnels, Vite exploite les modules ES natifs du navigateur, ce qui réduit considérablement le temps de démarrage. Son intégration fluide avec React nous a permis de bénéficier d'une expérience de développement moderne, efficace et légère.



Pour la phase de conception visuelle, nous avons utilisé **Figma**, un outil en ligne de design d'interface très répandu dans le développement web. Il nous a permis de créer des maquettes interactives et de définir l'apparence générale de la plateforme avant d'entamer l'implémentation. Grâce à sa collaboration en temps réel, chaque membre de l'équipe pouvait consulter, modifier ou commenter les éléments graphiques, ce qui a facilité les échanges et accéléré les décisions.



Pour le développement du projet, nous avons utilisé **Visual Studio Code (VS Code)**, un éditeur de code moderne, léger et extensible développé par **Microsoft**. Il offre une prise en charge complète de JavaScript, Node.js et React, ainsi qu'un large éventail d'extensions utiles telles que Prettier, ESLint, et les snippets pour React. Son terminal intégré, son débogueur puissant et ses fonctionnalités de Git intégrées nous ont permis de travailler efficacement tout au long du projet. Grâce à son interface intuitive et à ses outils de productivité, VS Code a largement facilité la phase d'implémentation, aussi bien côté frontend que backend.



Pour la gestion du code source et le travail en équipe, nous avons utilisé **GitHub**, une plateforme de versionnage basée sur Git. Elle nous a permis de centraliser l'ensemble du projet, de suivre l'historique des modifications et de collaborer efficacement via des branches, des pull requests et des commits bien structurés. Chaque membre de l'équipe pouvait ainsi développer une partie spécifique du projet sans perturber le travail des autres. GitHub nous a également offert une solution fiable pour sauvegarder notre code et assurer la traçabilité de chaque évolution tout au long du développement de la plateforme.



Pour la modélisation des différentes structures de notre application, nous avons utilisé **StarUML**, un outil de conception UML puissant et flexible. Il nous a permis de créer des diagrammes de cas d'utilisation, de classes, de séquence et de base de données de manière claire et professionnelle. Grâce à son interface intuitive et à sa prise en charge de plusieurs langages de modélisation, StarUML nous a aidés à structurer efficacement notre système dès la phase d'analyse. Ces schémas ont servi de base solide pour la phase de développement, facilitant la communication entre les membres de l'équipe et assurant une cohérence tout au long du projet.



Pour la modélisation conceptuelle des données, nous avons utilisé Looping, un logiciel spécialisé dans la création de MCD (Modèles Conceptuels de Données). Il nous

a permis de représenter les entités, les associations, les cardinalités ainsi que les relations réflexives de manière claire et structurée. Grâce à ses fonctionnalités graphiques intuitives, Looping facilite la conception logique des bases de données tout en respectant les règles de modélisation. Cet outil nous a aidés à poser une base solide pour le modèle relationnel, servant ainsi de référence lors de l'implémentation de la base de données dans MongoDB.

2. Conclusion

L'implémentation confirme la pertinence de l'architecture full JavaScript choisie. Le couple Node.js / Express fournit une API performante et extensible, tandis que React et Tailwind CSS offrent une interface fluide et responsive. L'utilisation de MongoDB, associée à Mongoose, a facilité l'évolution du schéma de données au fil des sprints. Les premiers tests valident la robustesse de l'authentification JWT, la cohérence du pattern MVC et la bonne communication entre frontend et backend, plaçant la plateforme sur des bases techniques solides.

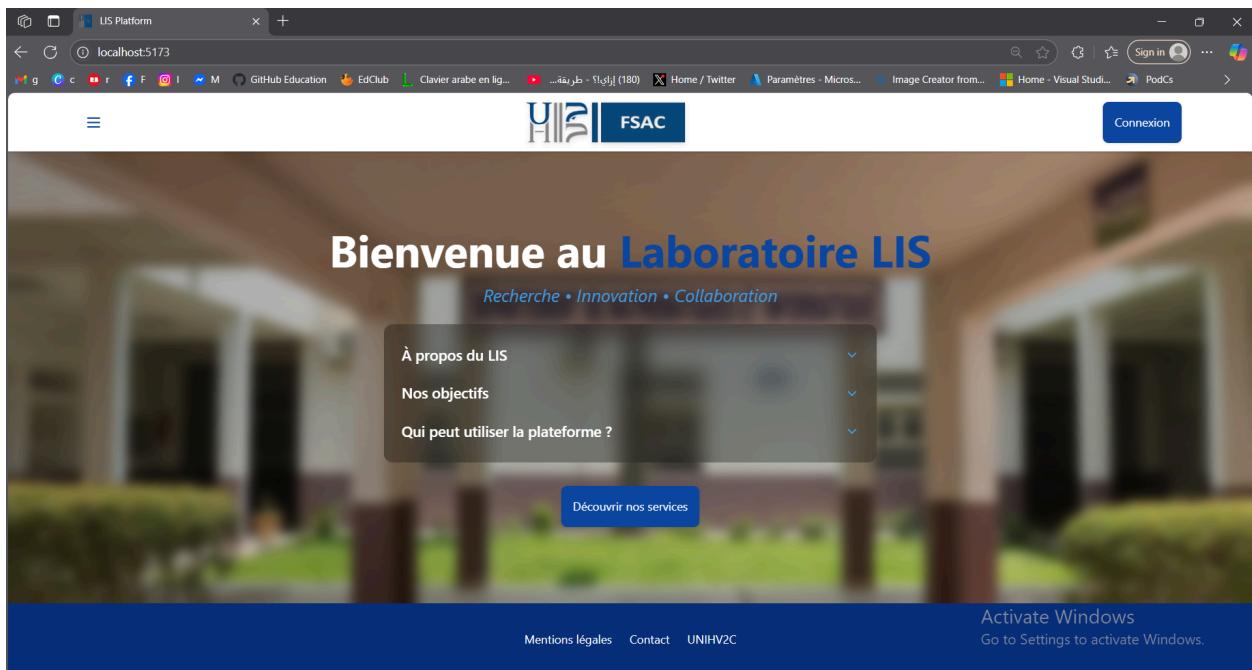
Chapitre 4:

Réalisation

Introduction

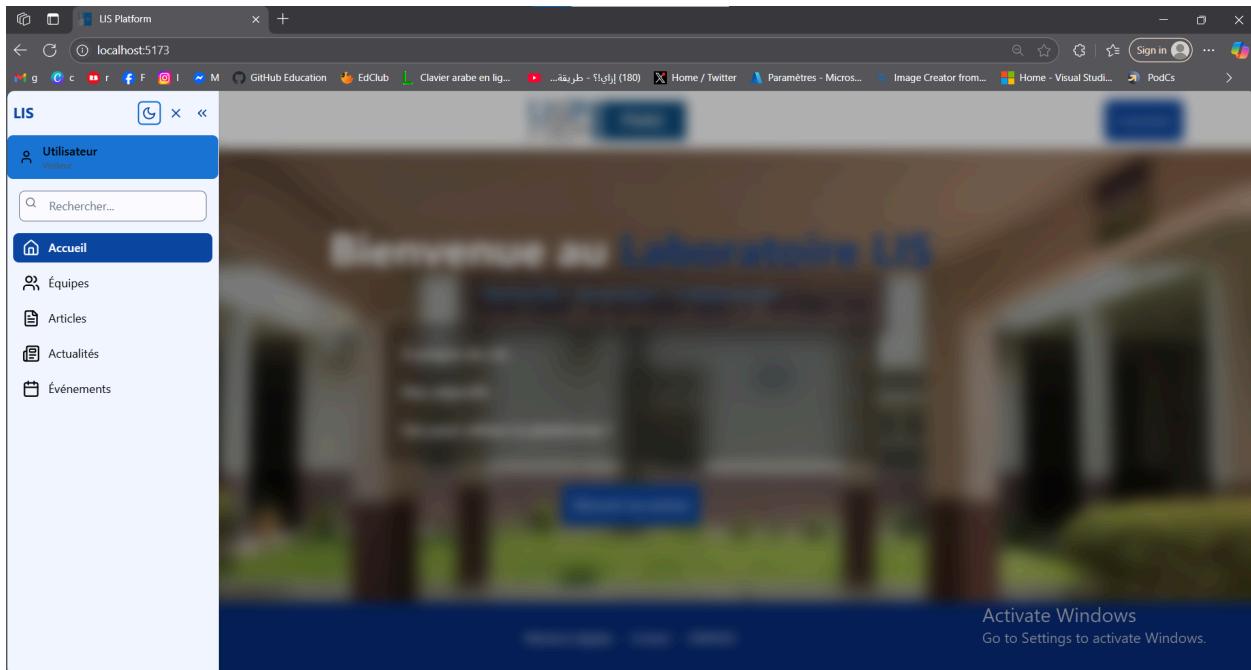
Ce chapitre présente le résultat final de notre plateforme à travers des captures d'écran des différentes pages. Chaque vue illustre l'interface utilisateur et les fonctionnalités développées, mettant en évidence la structure, l'ergonomie et le respect des objectifs définis lors de la conception.

Page d'accueil :



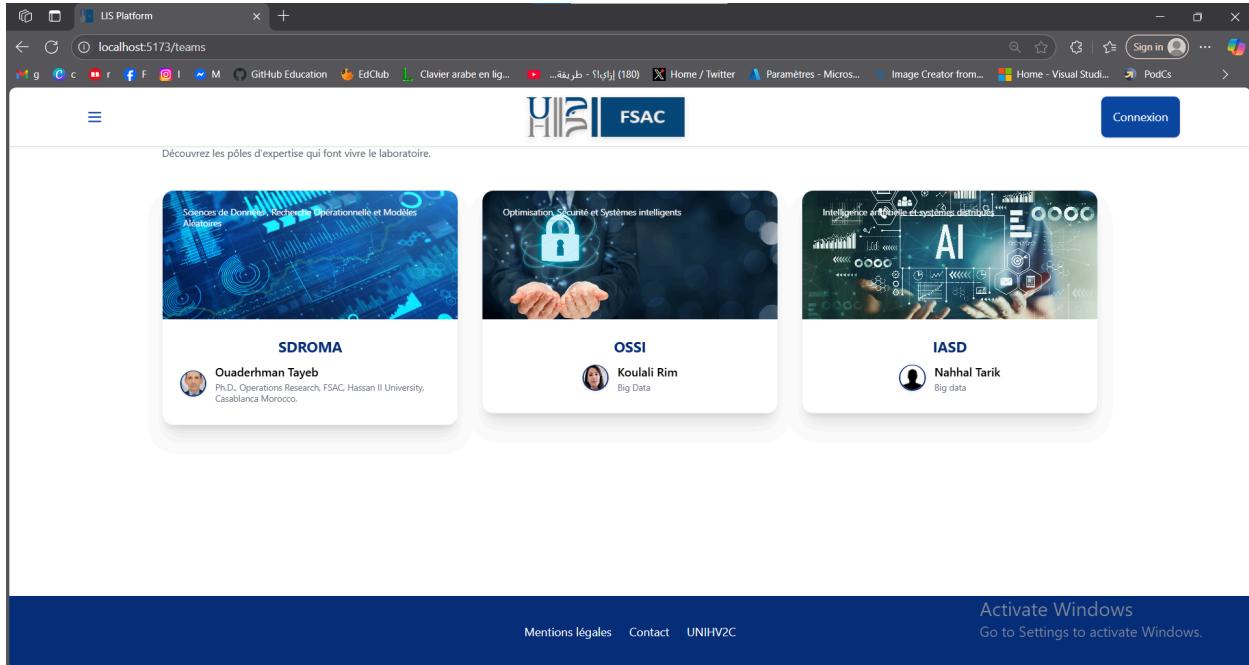
La page d'accueil constitue la première interface de contact avec la plateforme. Elle met en avant le nom du laboratoire, accompagné d'un slogan axé sur la recherche, l'innovation et la collaboration. L'utilisateur y

trouve des sections informatives telles que « À propos du LIS », « Nos objectifs » et « Qui peut utiliser la plateforme ? », accessibles sous forme de menus déroulants. Un bouton central permet d'accéder rapidement aux services proposés, dans une mise en page simple, moderne et accessible.



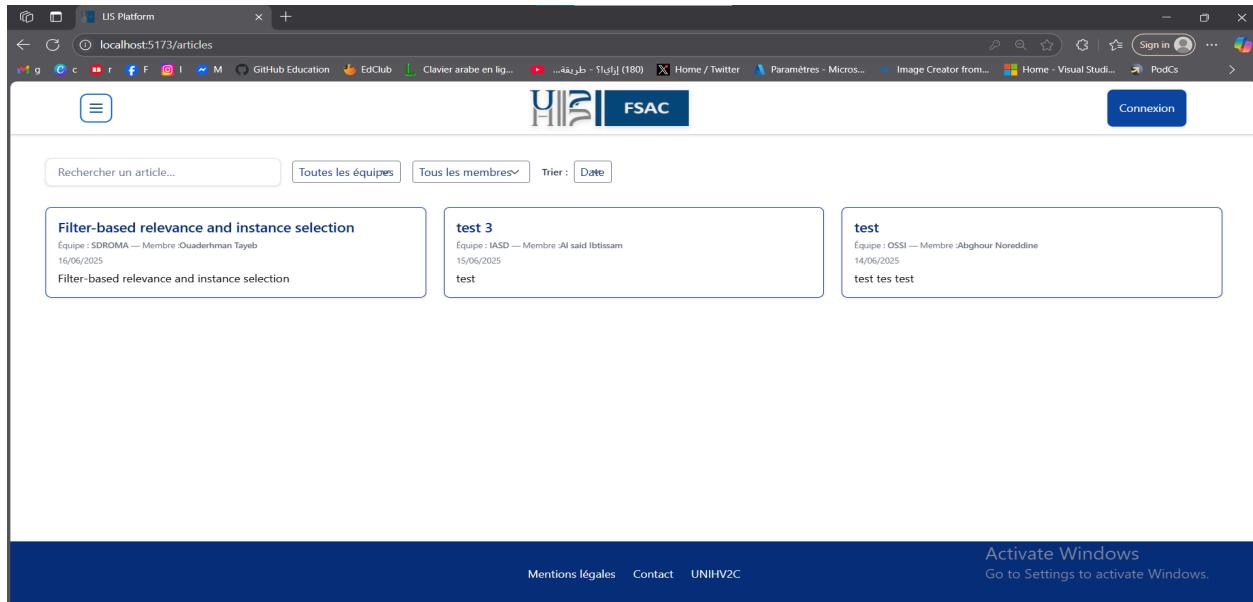
La **barre latérale** de la plateforme permet une navigation claire et structurée entre les différentes sections publiques du site. Elle affiche le rôle de l'utilisateur connecté (ici Visiteur) et propose un accès rapide aux pages principales : Accueil, Équipes, Articles, Actualités et Événements. Un champ de recherche est également intégré pour faciliter l'exploration du contenu. L'interface est pensée pour être simple, intuitive et accessible à tout moment depuis n'importe quelle page.

Page Équipes:



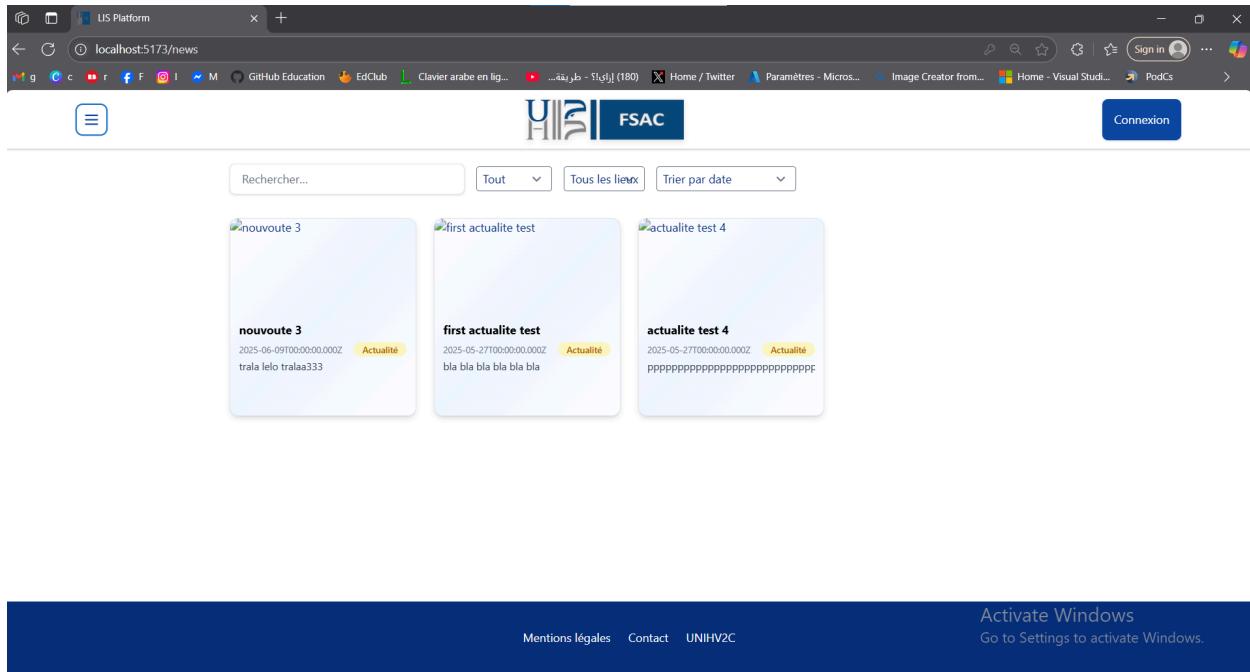
La page Équipes présente les différents pôles de recherche qui composent le laboratoire LIS. Chaque équipe est affichée sous forme de carte visuelle contenant son nom, le nom du responsable, sa spécialité ainsi qu'une image représentative. Cette section permet aux visiteurs d'identifier rapidement les domaines d'expertise du laboratoire et les chercheurs qui les encadrent. L'interface est conçue pour être claire, attractive et facile à parcourir.

Page Articles:



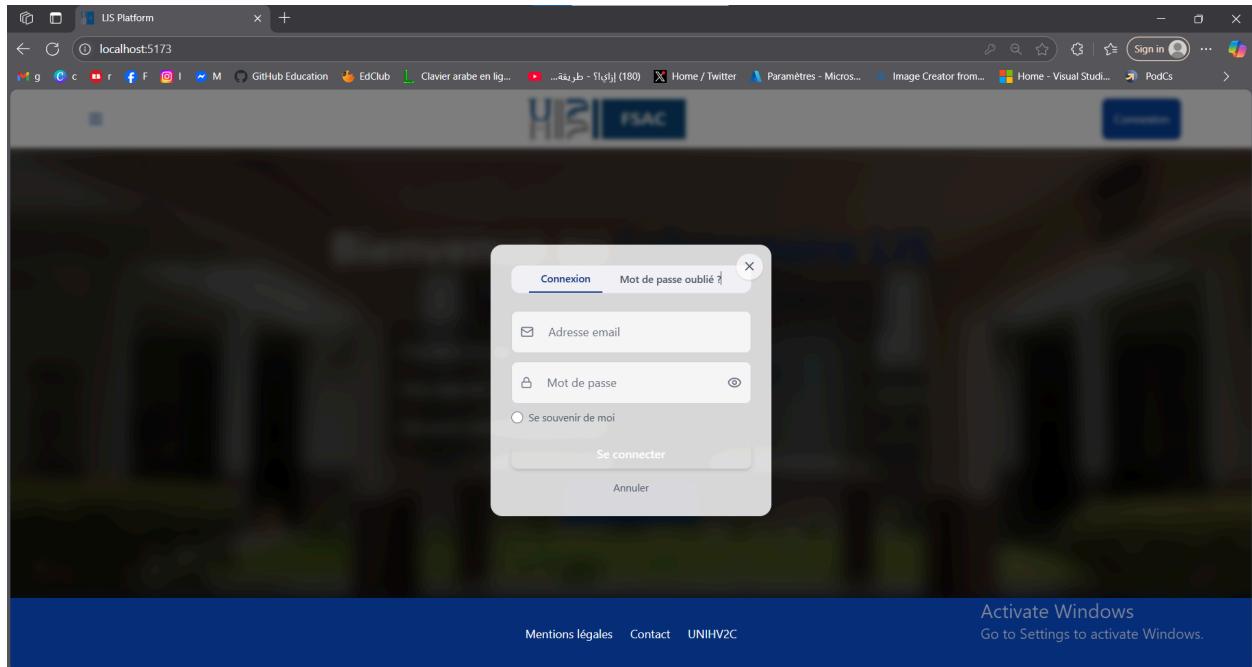
La page Articles affiche l'ensemble des publications validées par les membres du laboratoire. Chaque carte présente le titre de l'article, la date de publication, l'équipe concernée ainsi que le nom de l'auteur. Un champ de recherche et des filtres permettent de trier les articles par équipe, par membre ou par date, offrant ainsi une navigation simple et rapide. Cette interface vise à valoriser les travaux réalisés au sein du LIS tout en facilitant leur consultation par les visiteurs.

Page Actualités:



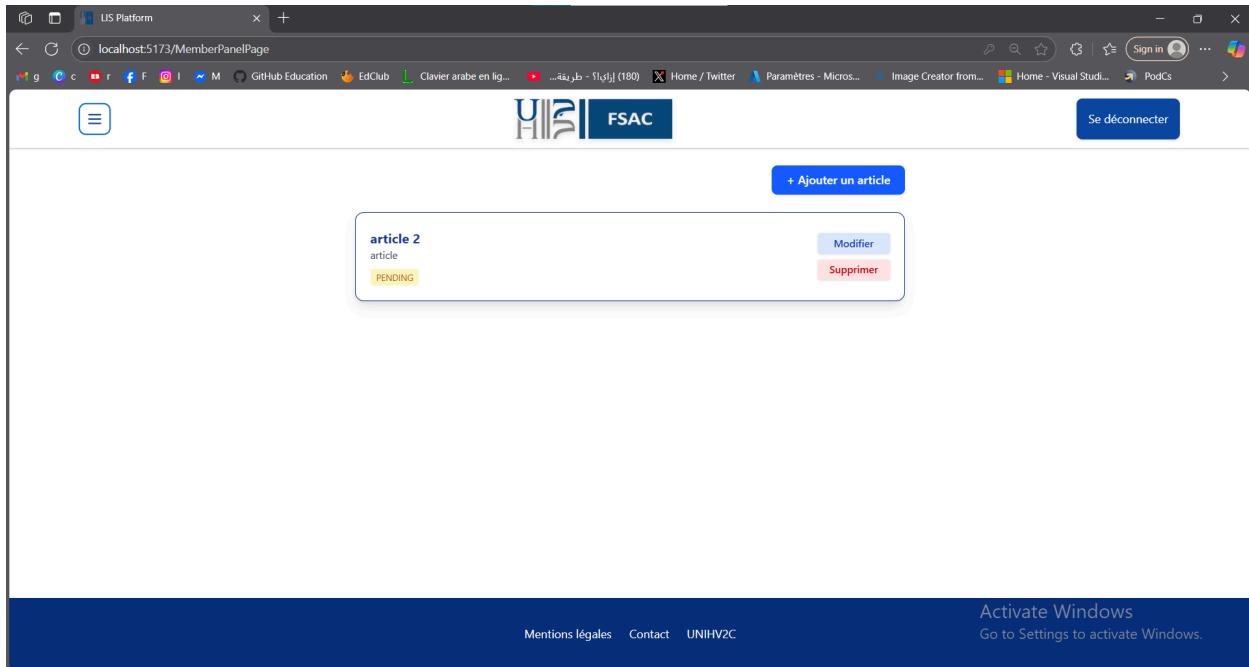
La page Actualités permet d'afficher les dernières informations liées aux activités du laboratoire. Chaque carte présente un titre, une date, une courte description, et une étiquette visuelle pour indiquer le type de contenu. Un moteur de recherche et des filtres permettent de trier les actualités par lieu ou par date. Cette section offre aux visiteurs un aperçu clair et actualisé des nouveautés et événements marquants du LIS.

Page Authentification:



La page Authentification permet aux utilisateurs autorisés d'accéder à leur espace personnel selon leur rôle (Directeur, Responsable ou Membre). Elle propose un formulaire clair et sécurisé avec les champs email et mot de passe, une option « Se souvenir de moi » ainsi qu'un lien pour réinitialiser le mot de passe en cas d'oubli. Son design sobre et centré assure une expérience utilisateur simple et rapide dès la connexion.

Le tableau de bord Membre:



Le tableau de bord Membre permet à chaque utilisateur connecté de gérer ses propres articles. Depuis cette interface simple et épurée, le membre peut consulter l'état de ses publications, soumettre de nouveaux articles, les modifier ou les supprimer tant qu'ils ne sont pas encore validés. Chaque article est affiché avec son titre, un résumé et un badge indiquant son statut (ici "PENDING"). Ce tableau de bord offre un espace personnel clair et fonctionnel pour suivre l'activité de publication au sein du laboratoire.

Le tableau de bord Responsable:

The screenshot shows a web browser window titled 'UIS Platform' with the URL 'localhost:5173/leader-dashboard'. The page features a header with the UNIHV2C logo and 'FSAC'. On the left, a sidebar has a 'Membres de l'équipe' section with a search bar and two entries: 'Nahhal Tarik' (big data, Rôle : RESPONSABLE) and 'Al said Ibtissam' (SCIENCES DE L'INGÉNIERIE, Rôle : MEMBRE). A 'Retirer' button is next to Al said's entry. Below this is a navigation bar with 'Articles en attente' (selected) and 'Articles publiés', followed by a search bar and a 'Valider' button. At the bottom, there are links for 'Mentions légales', 'Contact', and 'UNIHV2C', and a dark blue footer bar with the text 'Activate Windows Go to Settings to activate Windows.'

Le tableau de bord Responsable offre un espace de gestion complet pour superviser son équipe et valider les articles. Il permet d'afficher les membres de l'équipe, d'en ajouter ou retirer facilement, et de consulter leurs informations. Dans la section inférieure, les articles en attente sont listés avec leur statut, et le Responsable peut les valider d'un simple clic. Une navigation par onglets permet également de basculer entre les articles en attente et ceux déjà publiés. L'interface a été pensée pour être à la fois intuitive, réactive et conforme aux responsabilités de ce rôle.

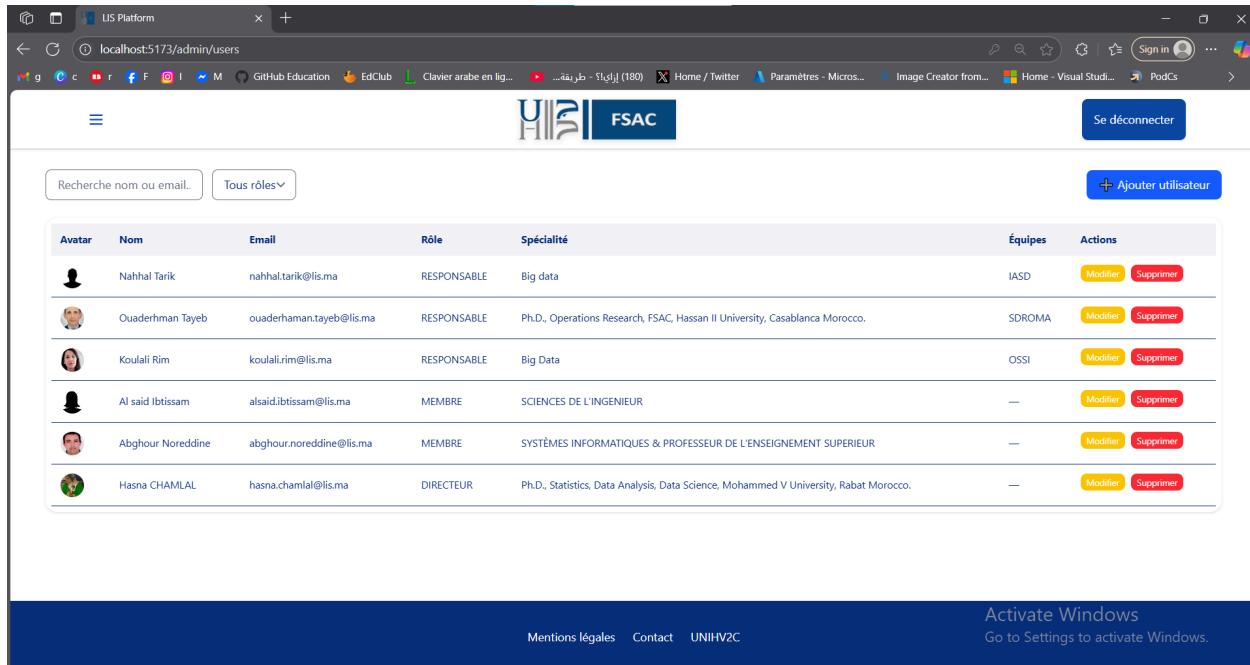
Le tableau de bord Directeur:

The screenshot shows the Director Dashboard interface. At the top, there are four summary boxes: 'Membres' (4), 'Équipes' (3), 'Articles' (17), and 'Événements' (7). Below these are two main sections: 'Articles récents' and 'Événements à venir'. The 'Articles récents' section lists five articles with their status (APPROVED or PENDING) and a preview of the content. The 'Événements à venir' section lists five events with their dates. At the bottom, there is a table for 'Équipes' (Teams) with columns for Team, Speciality, Leader, and Members. The table contains three entries. At the very bottom, there are links for 'Mentions légales', 'Contact', and 'UNIHV2C', along with an 'Activate Windows' message.

Équipe	Spécialité	Leader	Membres
Équipe Systèmes Embarqués	IoT, capteurs intelligents, temps réel	2	2
Équipe Data Science	Big Data, ML, Deep Learning	3	1
Équipe Sécurité Réseaux	Chiffrement, sécurité IoT, VPN	4	1

Le tableau de bord Directeur offre une vue d'ensemble complète de l'activité sur la plateforme. Il affiche des indicateurs clés comme le nombre total de membres, d'équipes, d'articles et d'événements. On y trouve une liste des derniers articles publiés avec leur statut (validé ou en attente), les événements à venir avec leurs dates, ainsi qu'un récapitulatif des équipes, leurs spécialités, leurs responsables et le nombre de membres associés. Ce tableau de bord centralise les informations essentielles et permet au Directeur de suivre facilement l'évolution du laboratoire.

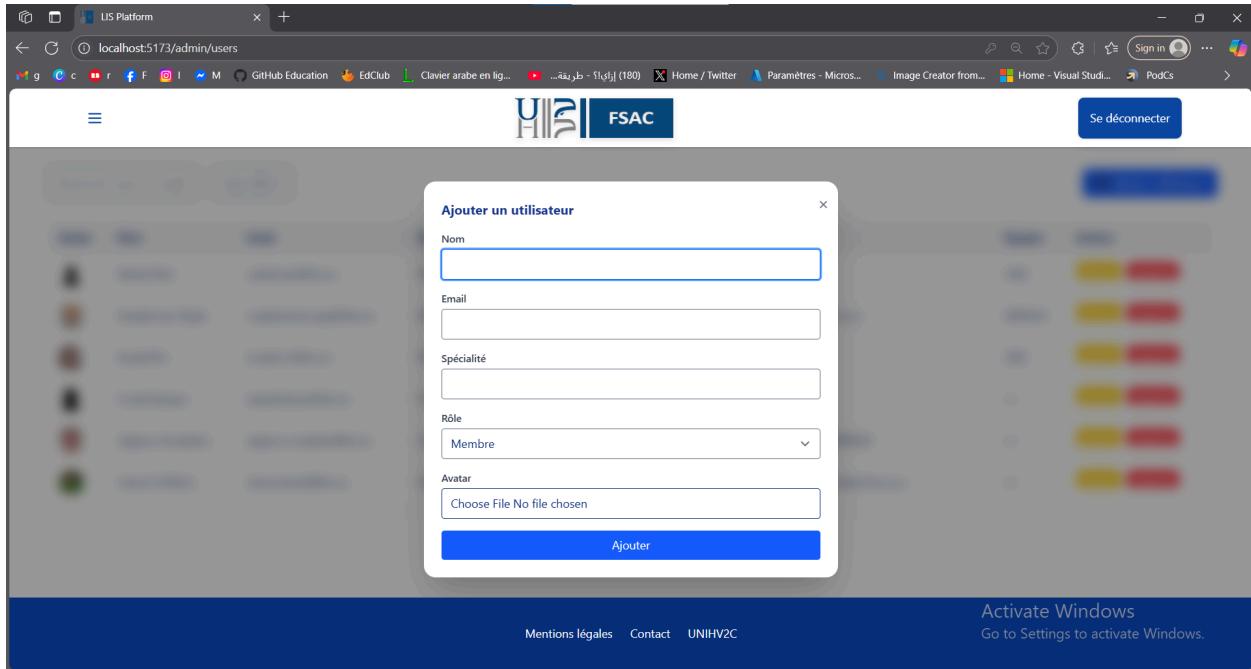
La page Gestion des utilisateurs:



Avatar	Nom	Email	Rôle	Spécialité	Équipes	Actions
	Nahhal Tarik	nahhal.tarik@lis.ma	RESPONSABLE	Big data	IASD	<button>Modifier</button> <button>Supprimer</button>
	Ouaderhaman Tayeb	ouaderhaman.tayeb@lis.ma	RESPONSABLE	Ph.D., Operations Research, FSAC, Hassan II University, Casablanca Morocco.	SDROMA	<button>Modifier</button> <button>Supprimer</button>
	Koulali Rim	koulali.rim@lis.ma	RESPONSABLE	Big Data	OSSI	<button>Modifier</button> <button>Supprimer</button>
	AI said Ibtissam	alsaid.ibtissam@lis.ma	MEMBRE	SCIENCES DE L'INGENIEUR	—	<button>Modifier</button> <button>Supprimer</button>
	Abghour Noredinne	abghour.noreddine@lis.ma	MEMBRE	SYSTÈMES INFORMATIQUES & PROFESSEUR DE L'ENSEIGNEMENT SUPERIEUR	—	<button>Modifier</button> <button>Supprimer</button>
	Hasna CHAMLAL	hasna.chamlal@lis.ma	DIRECTEUR	Ph.D., Statistics, Data Analysis, Data Science, Mohammed V University, Rabat Morocco.	—	<button>Modifier</button> <button>Supprimer</button>

Activate Windows
Go to Settings to activate Windows.

La page Gestion des utilisateurs, accessible au Directeur, permet d'administrer tous les comptes de la plateforme. Elle affiche sous forme de tableau les informations clés de chaque utilisateur : nom, email, rôle, spécialité et appartenance à une équipe. Le Directeur peut ajouter un nouvel utilisateur via un bouton dédié, modifier les informations existantes ou supprimer un compte. Un champ de recherche et un filtre par rôle facilitent la gestion ciblée. Cette interface assure un contrôle complet et centralisé de l'ensemble des utilisateurs du système.



Le formulaire **Ajouter un utilisateur**, accessible depuis l'espace Directeur, permet de créer un nouveau compte en renseignant les informations essentielles : nom, adresse email, spécialité, rôle (Directeur, Responsable ou Membre) ainsi qu'un avatar. Ce module simple et intuitif garantit une saisie rapide et centralisée des données, tout en assurant la cohérence des informations utilisateur dans l'ensemble du système.

La page Gestion des équipes:

The screenshot shows a web application interface titled 'Gestion des équipes'. At the top, there is a header with the logo 'UFR FSAC' and a search bar labeled 'Rechercher nom ou spécialité...'. A blue button '+ Ajouter équipe' is located in the top right corner. Below the header is a table listing three research teams:

Nom	Spécialité	Leader	# Membres	Actions
SDROMA	Sciences de Données , Recherche Opérationnelle et Modèles Aléatoires	Ouaderhman Tayeb	2	<button>Modifier</button> <button>Supprimer</button>
OSSI	Optimisation, Sécurité et Systèmes intelligents	Koulali Rim	2	<button>Modifier</button> <button>Supprimer</button>
IASD	Intelligence artificielle et systèmes distribués	Nahhal Tarik	2	<button>Modifier</button> <button>Supprimer</button>

At the bottom of the page, there is a dark footer bar with links to 'Mentions légales', 'Contact', and 'UNIHIV2C'. On the right side of the footer, there is an 'Activate Windows' message: 'Go to Settings to activate Windows.'

La page Gestion des équipes, accessible au Directeur, permet d'administrer toutes les équipes du laboratoire. Chaque ligne du tableau affiche le nom de l'équipe, sa spécialité, le responsable (leader), ainsi que le nombre de membres associés. Des actions sont disponibles pour modifier ou supprimer une équipe, et un bouton permet d'en créer une nouvelle. Un champ de recherche facilite la navigation et le filtrage par nom ou spécialité. Cette interface centralise la structuration des pôles de recherche de manière simple et efficace.

The screenshot shows a web-based administrative interface for managing teams. On the left, there's a list of existing teams with columns for 'Nom' (Name) and 'Spécialité' (Specialization). Three teams are listed: SDROMA (Sciences de Données, Recherche O...), OSSI (Optimisation, Sécurité et Systèmes i...), and IASD (Intelligence artificielle et systèmes d...). On the right, a modal window titled 'Ajouter une équipe' (Add team) is open, prompting the user to enter information for a new team. The fields include:

- Nom de l'équipe** (Team name): An input field.
- Spécialité** (Specialization): An input field.
- Description**: An input field.
- Image (optionnelle)** (Optional image): A file upload input field with the placeholder "Choose File No file chosen".
- Responsable** (Responsible): A dropdown menu showing "Nahhal Tarik".

Below the modal, there's a large blue button labeled "Ajouter" (Add). To the right of the modal, a preview area shows three entries with "Modifier" and "Supprimer" buttons for each. At the bottom of the page, there are links for "Mentions légales", "Contact", and "UNIHV2C", along with an "Activate Windows" message.

Le formulaire **Ajouter une équipe**, accessible au Directeur, permet de créer une nouvelle entité de recherche au sein du laboratoire. Il comporte plusieurs champs à renseigner : le nom de l'équipe, sa spécialité, une description, une image représentative (facultative) et le choix du responsable. Cette interface offre un moyen rapide et structuré d'enrichir l'organisation scientifique de la plateforme, tout en garantissant une saisie complète et cohérente des données.

Conclusion

Ce projet de fin d'études a constitué une expérience complète, alliant modélisation, conception, développement et déploiement d'une solution web au service du Laboratoire d'Informatique et de Systèmes (LIS). Face à une problématique réelle de fragmentation de l'information et de manque de visibilité, nous avons proposé une plateforme unifiée, moderne et performante, adaptée aux besoins spécifiques du laboratoire.

Grâce à une analyse approfondie des besoins fonctionnels et non fonctionnels, nous avons défini une architecture robuste s'appuyant sur des technologies actuelles telles que React, Node.js/Express, MongoDB et Tailwind CSS. L'ensemble a été pensé pour garantir :

- une gestion fluide des utilisateurs, équipes, articles, actualités et événements,
- une sécurité renforcée via JWT, contrôles d'accès et validations multi-niveaux,
- une performance optimisée pour répondre aux exigences d'un usage en temps réel,
- et une interface ergonomique, accessible à tous les rôles utilisateurs.

Les différents diagrammes UML, le MCD, et les séquences implémentées ont servi de base solide pour assurer la cohérence technique du projet. Le travail collaboratif, facilité par GitHub, ainsi que les phases de test et de validation, ont permis d'assurer la qualité et la stabilité du produit final.

Au-delà des compétences techniques mobilisées, ce projet nous a permis de renforcer notre capacité à travailler en équipe, à gérer un cycle de développement complet, et à répondre de manière concrète à des enjeux organisationnels réels. Nous espérons que cette plateforme contribuera efficacement à la transformation numérique du LIS, et qu'elle pourra évoluer pour intégrer de nouvelles fonctionnalités à l'avenir.