**Règles de gestion**

Chaque utilisateur doit s'inscrire sur la plateforme avec des informations personnelles valides (nom, e-mail, etc.). Un processus de validation par e-mail sera utilisé pour confirmer l’inscription.

Tous les utilisateurs doivent se connecter avec des identifiants uniques (e-mail et mot de passe). Un système de réinitialisation de mot de passe doit être prévu en cas d’oubli.

Les utilisateurs doivent être classés en rôles spécifiques (étudiant, enseignant, administrateur), chacun ayant des droits et des privilèges spécifiques sur la plateforme (ex : accès à certaines fonctionnalités, permissions d'édition, etc.).

Chaque utilisateur doit pouvoir consulter et modifier son profil, à l'exception des informations sensibles (e-mail, rôle, etc.), qui ne peuvent être changées que par l'administrateur.

Les enseignants doivent pouvoir créer des cours, ajouter des modules et des leçons, ainsi que les éditer ou les supprimer.

Les cours peuvent être affectés à des étudiants spécifiques ou à des groupes d’étudiants. Seuls les enseignants et administrateurs peuvent gérer les affectations.

Les étudiants doivent avoir accès uniquement aux cours auxquels ils sont inscrits. Les enseignants doivent pouvoir accéder à tous les cours qu'ils gèrent.

Les contenus pédagogiques doivent être accessibles selon la progression de l’étudiant, c’est-à-dire qu’il doit avoir terminé un module pour accéder au suivant, à moins que l'enseignant ne décide d'ouvrir tous les modules à l'avance.

Les enseignants doivent pouvoir créer des évaluations sous forme de quiz, d’examens ou de projets, avec des critères d’évaluation prédéfinis (points, pourcentage, etc.). L’évaluation doit avoir de délais.

Après soumission des évaluations par les étudiants, les enseignants doivent pouvoir corriger et fournir un feedback personnalisé à chaque étudiant.

Les performances des étudiants doivent être suivies tout au long du cours. Chaque étudiant doit pouvoir consulter un tableau de bord récapitulatif de ses progrès (notes, devoirs soumis, progression dans les modules).

Les résultats des évaluations doivent être consultables par les étudiants uniquement après leur publication par l'enseignant. Les enseignants ont un accès complet aux résultats de leurs cours, tandis que les administrateurs peuvent consulter tous les résultats.

Les étudiants doivent recevoir des notifications automatiques pour les échéances de devoirs, d’examens ou de projets à venir.

Une fois les évaluations corrigées, les étudiants doivent recevoir une notification avec un lien vers leurs résultats et les feedbacks associés.

Les enseignants doivent pouvoir envoyer des messages ou des notifications groupées aux étudiants inscrits à leurs cours (ex : rappel de cours en direct, modification de planning).

Les informations personnelles des utilisateurs doivent être protégées conformément aux réglementations en vigueur (ex : RGPD). Les données sensibles (mot de passe, résultats, etc.) doivent être cryptées et stockées de manière sécurisée.

Les données de la plateforme (cours, résultats des étudiants, profils utilisateurs) doivent être sauvegardées régulièrement pour éviter toute perte de données en cas de problème technique.

Les administrateurs doivent avoir un accès complet à toutes les fonctionnalités de la plateforme, tandis que les enseignants et étudiants ont des accès limités en fonction de leur rôle.

Un forum ou un système de discussion doit être mis en place pour permettre aux étudiants d’échanger entre eux ou avec les enseignants sur les sujets des cours.

Les utilisateurs doivent pouvoir contacter un service de support en cas de problème technique (bug, difficulté à accéder à un cours, etc.). Ce système doit inclure une réponse sous 24h aux demandes.

Les étudiants doivent pouvoir évaluer les cours qu'ils suivent, donnant ainsi un retour d'information aux enseignants et aux administrateurs sur la qualité des contenus.

Seuls les administrateurs peuvent gérer les paramètres globaux de la plateforme (ex : ajout de nouveaux enseignants, modifications des paramètres de sécurité).

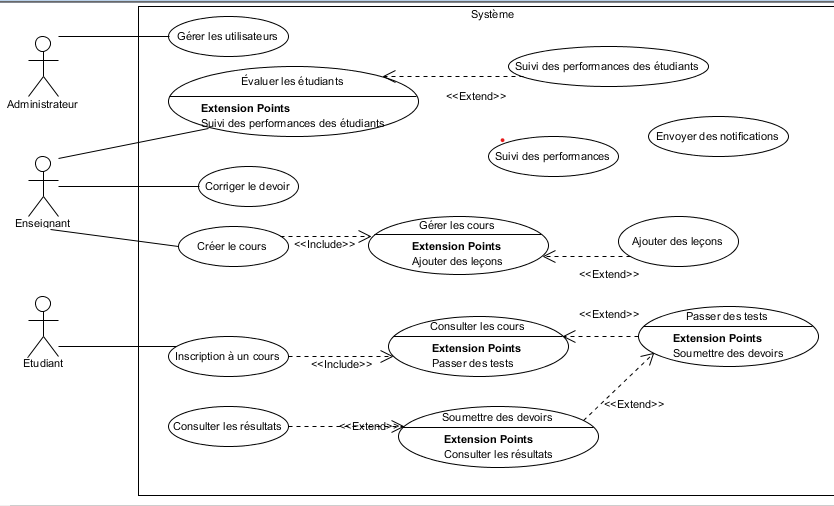
Une fois qu’un cours est terminé, il doit être archivé. Les étudiants peuvent continuer à consulter les contenus, mais ne peuvent plus interagir activement avec le cours (ex : soumettre des devoirs).

**Diagramme de cas d'utilisation**

Les besoins étant énuméré dans le chapitre précèdent, nous allons maintenant concevoir le diagramme de cas d’utilisation correspondant à ces besoins.

**Acteurs principaux :**

|  |  |
| --- | --- |
| Acteurs | Cas d’utilisation |
| **Administrateur** | Gère le système dans son ensemble (cours, utilisateurs). |
| **Enseignant** | Crée et gère les cours, les leçons, les évaluations, et suit les performances des étudiants. |
| **Étudiant** | S'inscrit aux cours, consulte le contenu, soumet des devoirs et passe des tests. |



**Description détaillée des cas d'utilisation :**

**Gérer les utilisateurs**

|  |  |
| --- | --- |
| **Acteur** | Administrateur |
| **Description** | L'administrateur peut créer des comptes pour les utilisateurs (enseignants ou étudiants), les modifier ou les supprimer. |
| **Précondition** | L'administrateur doit être connecté. |
| **Post-condition** | Le système met à jour la liste des utilisateurs. |

**Créer un cours**

|  |  |
| --- | --- |
| **Acteur** | Enseignant |
| **Description** | L'enseignant peut créer un nouveau cours en entrant des informations telles que le nom, la description, et en définissant les modules ou les chapitres. |
| **Précondition** | L'enseignant doit être authentifié et disposer des droits appropriés. |
| **Post-condition** | Le cours est disponible pour les inscriptions des étudiants. |

**Gérer les cours**

|  |  |
| --- | --- |
| **Acteur** | Enseignant |
| **Description** | L'enseignant peut modifier ou supprimer des cours existants. Il peut aussi ajouter ou supprimer des modules de cours. |
| **Précondition** | L'enseignant doit être connecté et le cours doit lui appartenir. |
| **Post-condition** | Les modifications sont sauvegardées dans le système. |

**Inscription à un cours**

|  |  |
| --- | --- |
| **Acteur** | Étudiant |
| **Description** | Un étudiant peut parcourir les cours disponibles et s'inscrire à ceux qui l'intéressent. |
| **Précondition** | L'étudiant doit être authentifié. |
| **Post-condition** | L'étudiant est inscrit dans le cours et peut accéder au contenu. |

**Consulter les cours**

|  |  |
| --- | --- |
| **Acteur** | Étudiant |
| **Description** | L'étudiant peut consulter les leçons, vidéos, et autres contenus des cours auxquels il est inscrit. |
| **Précondition** | L'étudiant doit être inscrit à un cours. |
| **Post-condition** | Le contenu est accessible à l'étudiant. |

**Ajouter des leçons**

|  |  |
| --- | --- |
| **Acteur** | Enseignant |
| **Description** | L'enseignant peut ajouter des leçons à un cours en publiant des documents, vidéos, ou fichiers. |
| **Précondition** | L'enseignant doit être connecté et avoir créé un cours. |
| **Post-condition** | La leçon est ajoutée au cours et devient disponible pour les étudiants. |

**Passer des tests**

|  |  |
| --- | --- |
| **Acteur** | Étudiant |
| **Description** | L'étudiant peut passer des tests créés par les enseignants pour évaluer sa compréhension du cours. |
| **Précondition** | L'étudiant doit être inscrit à un cours et les tests doivent être disponibles. |
| **Post-condition** | Les résultats du test sont enregistrés. |

**Soumettre des devoirs**

|  |  |
| --- | --- |
| **Acteur** | Étudiant |
| **Description** | L'étudiant soumet des devoirs ou des projets pour évaluation. |
| **Précondition** | Le devoir doit être disponible pour soumission. |
| **Post-condition** | Le devoir est envoyé et en attente d’évaluation. |

**Évaluer les étudiants**

|  |  |
| --- | --- |
| **Acteur** | Enseignant |
| **Description** | L'enseignant évalue les devoirs soumis ou les tests passés par les étudiants et attribue des notes. |
| **Précondition** | L'enseignant doit avoir accès aux devoirs ou tests soumis. |
| **Post-condition** | Les notes sont enregistrées et accessibles aux étudiants. |

**Suivi des performances des étudiants**

|  |  |
| --- | --- |
| **Acteur** | Enseignant |
| **Description** | L'enseignant peut suivre la progression des étudiants grâce à des rapports, notamment sur les tests et les devoirs. |
| **Précondition** | L'enseignant doit être connecté. |
| **Post-condition** | Le système affiche des statistiques sur les performances des étudiants. |

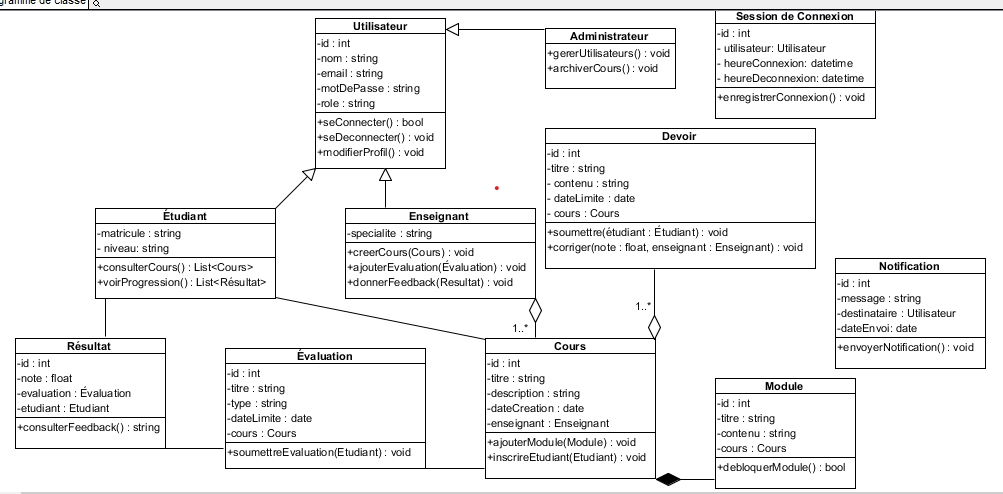
**Consulter les résultats**

|  |  |
| --- | --- |
| **Acteur** | Étudiant |
| **Description** | L'étudiant peut consulter ses résultats pour chaque cours (notes obtenues, tests réussis, etc.). |
| **Précondition** | L'étudiant doit avoir passé des tests ou soumis des devoirs. |
| **Post-condition** | Les résultats sont visibles. |

**Diagramme de Classe**

**Classes Principales :**

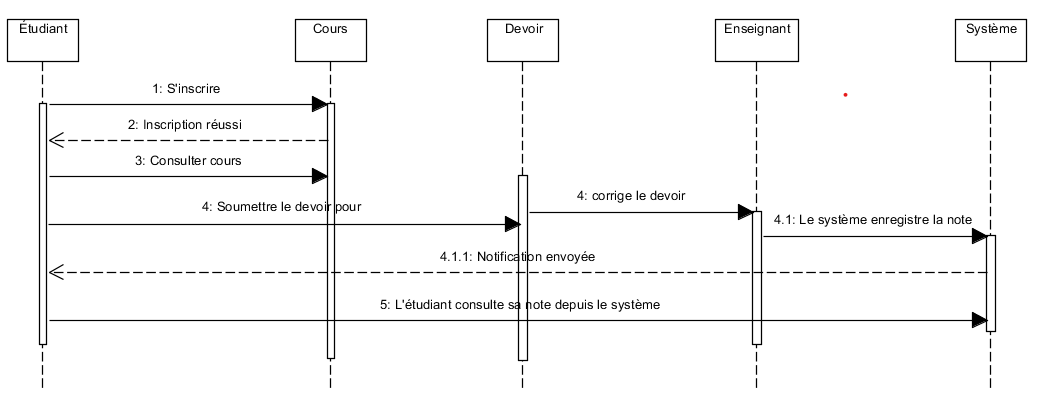
1. **Utilisateur**
2. **Étudiant** (hérite de Utilisateur)
3. **Enseignant** (hérite de Utilisateur)
4. **Administrateur** (hérite de Utilisateur)
5. **Cours**
6. **Module**
7. **Évaluation**
8. **Résultat**
9. **Notification**
10. **Session de Connexion**



**Diagramme de séquence**

**Flux pour le cas où un étudiant soumet un devoir**

1. **Étudiant :** s’inscrire au cours.
2. **Étudiant :** Consulter cours.
3. **Étudiant :** soumet un devoir.
4. **Enseignant :** Corrige le devoir.
5. **Système :** Le système enregistre la note.
6. **Système**: Notification envoyée aux étudiants
7. **Étudiant :** L'étudiant consulte sa note depuis le système



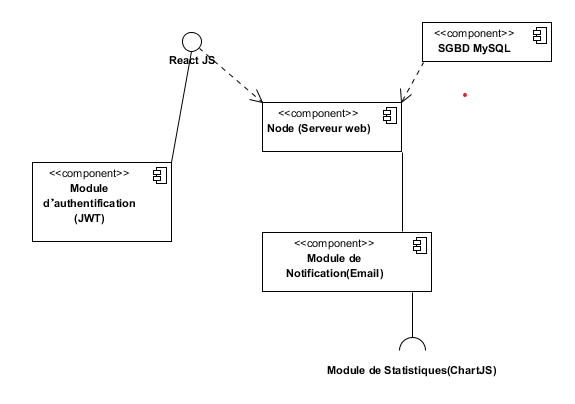
**Description du processus :**

1. **Connexion de l'étudiant** : L'étudiant s'authentifie via le système. Si la connexion échoue, une erreur est affichée.
2. **Accès aux cours et leçons** : Après connexion, l’étudiant peut consulter ses cours, leçons et devoirs.
3. **Soumission de devoir** : Si l'étudiant soumet son devoir à temps, le système l'enregistre, sinon une erreur de soumission est affichée.
4. **Notification** : Une notification est envoyée à l’enseignant pour l’informer de la soumission.
5. **Correction du devoir** : L'enseignant corrige le devoir et attribue une note, qui est enregistrée dans le système.
6. **Suivi des performances** : L'étudiant est notifié de sa note, et les statistiques de performance sont mises à jour.

**Diagramme de composants**

#### **Composants principaux :**

1. **Interface utilisateur (UI)**
   * Composant : Application Web Frontend
     + Technologies : Next.js / React
     + Rôle : Gérer l’interface utilisateur et la navigation.
     + Dépendance : API Backend (via requêtes HTTP).
     + Fonctionnalités : Authentification, consultation de cours, soumission de devoirs, consultation des résultats.
2. **API Backend**
   * Composant : Serveur Backend
     + Technologies : Node.js / Express (ou Laravel)
     + Rôle : Gérer la logique métier et les requêtes.
     + Dépendance : Base de données.
     + Fonctionnalités : Authentification, gestion des utilisateurs, gestion des cours et des évaluations.
3. **Système d'authentification**
   * Composant : Module d’authentification (JWT)
     + Fonction : Authentifier les utilisateurs (étudiants, enseignants, administrateurs).
     + Intégré avec : Frontend et Backend.
4. **Base de Données**
   * Composant : Base de données MySQL / Prisma ORM
     + Rôle : Stocker les informations sur les utilisateurs, cours, devoirs, résultats, notifications.
     + Dépendance : Serveur Backend (ORM gérant les transactions).
5. **Service de Notifications**
   * Composant : Module de Notification
     + Fonction : Envoi de notifications par email ou sur l’interface (ex. remise de notes, alertes sur les échéances).
6. **Module de gestion des statistiques**
   * Composant : Module de Statistiques
     + Fonction : Calculer et afficher les performances des étudiants via graphiques.



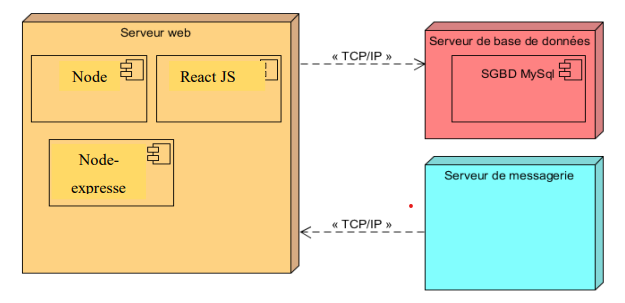
### ****Description du Flux :****

1. **Utilisateur (étudiant/enseignant)** accède à l'application via l’interface frontend.
2. Le **frontend** envoie des requêtes API au **backend** pour :
   * L'authentification.
   * La récupération de cours et devoirs.
   * L'enregistrement des résultats.
3. Le **backend** utilise l'**ORM Prisma** pour interagir avec la **base de données**.
4. Lorsqu’un devoir est corrigé, une notification est envoyée par le **module de notification** à l’étudiant.
5. Le **module de statistiques** collecte les données de performance et les affiche sur l'interface utilisateur.

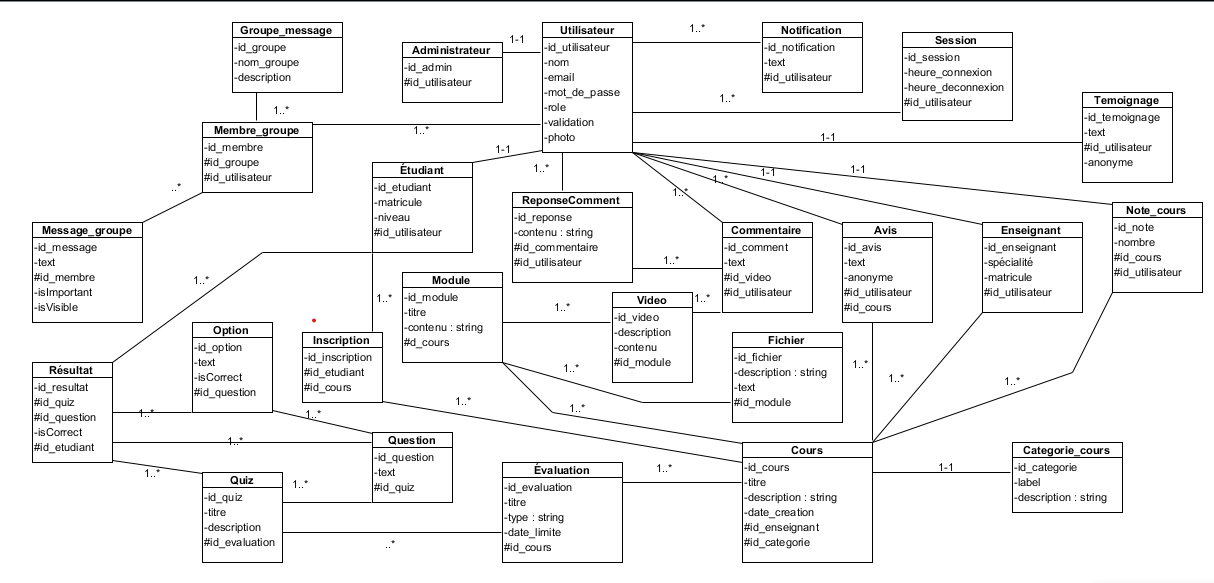
**Diagramme de déploiement**

Un diagramme de déploiement décrit la disposition physique des ressources matérielles qui composent le système et montre la répartition des composants sur ces matériels. Chaque ressource étant matérialisée par un nœud, le diagramme de déploiement précise comment les composants sont répartis sur les nœuds et quelles sont les connexions entre les composants ou les nœuds.

En d’autres termes ce diagramme décrit l’architecture logicielle du système mis en place.



**MCD**

****

**MLD**

* UTILISATEUR (id, nom, prenom, email, mot\_passe, role, validation, #id\_photo)
* PHOTO (id, nom, contenu)
* ENSEIGNAT (id, specialite, matricule, #id\_utilisateur)
* ETUDIANT (id, matricule, niveau, #id\_utilisateur)
* ADMINISTRATEUR (id, #id\_utilisateur)
* CATEGORIE (id, label, description)
* COURS (id, nom, description, #id\_enseignant, #id\_categorie)
* MODULE (id, nom, contenu, #id\_cours)
* VIDEO (id, nom, contenu, description, #id\_module)
* FICHIER (id, nom, contenu, description, #id\_module)
* COMMENTAIRE (id, text, #id\_video, #id\_utilisateur)
* REPONSE\_COMMENT (id, text, #id\_comment, #id\_utilisateur)
* DEVOIR (id, titre, type, contenu, date\_limite, #id\_cours)
* INSCRIR (#id\_etudiant, #id\_cours)
* EVALUATION (id, titre, type, date\_limite, #id\_cours)
* QUIZ (id, titre, description, #id\_evaluation)
* QUESTION (id, text, #id\_quiz)
* OPTION (id, text, isCorrect, #id\_question)
* RESULTAT (id, #id\_quiz, #id\_question, #id\_etudiant, isCorrect)
* AVIS (id, text, isAnonyme, #id\_utilisateur, #id\_cours)
* GROUPE\_MESSAGE (id, nom, description)
* MESSAGE\_GROUPE (id, #id\_groupe, #id\_utilisateur)
* MESSAGE (id, text, #id\_membre, isVisible, isImportant)
* NOTIFICATION (id, text, #id\_utilisateur)
* TEMOIGNAGE (id, text, isAnonyme, #id\_utilisatreur)
* SESSION (id, heure\_connexion, heure\_deconnexion, #id\_utilisateur)