DOCUMENT DE SPECIFICATION FONCTIONNELLES DETAILLEE

Projet : Création D’une Application De Gestion D’établissement Scolaire

**Cas : ENSITECH**

# Introduction

1. **Objectif du document**

Le présent document de spécifications fonctionnelles détaillées a pour objectif de définir de manière exhaustive les exigences fonctionnelles du logiciel de gestion destiné à l'école d’enseignement supérieur ENSITECH. Ce système sur mesure vise à optimiser la gestion globale de l'établissement, en particulier la gestion des étudiants, des cours et du corps professoral.

1. Objectifs principaux du système :

* **Amélioration de la gestion des étudiants :** Le système permettra de créer, de suivre et de mettre à jour les profils individuels des étudiants, en incluant des informations telles que leurs coordonnées, leur statut scolaire, et les détails de leur progression académique.
* **Optimisation de la gestion des cours :** Les responsables pourront associer chaque étudiant à ses cours respectifs, facilitant ainsi la planification et l'organisation des horaires et des ressources pédagogiques.
* **Centralisation des informations scolaires :** En intégrant toutes les données pertinentes dans une base de données unique, le système garantira une accessibilité rapide et sécurisée aux informations cruciales concernant les étudiants et leurs parcours académiques.
* **Facilitation de la gestion du corps enseignant :** Le logiciel permettra de gérer efficacement les informations relatives aux enseignants, y compris leurs disponibilités, leurs matières enseignées, et leurs contacts professionnels.
* **Sécurisation de l'accès et des données :** L'accès au système sera sécurisé, avec des rôles définis pour garantir que seul le directeur de l'école et le responsable des études auront accès aux fonctionnalités sensibles du système dans un premier temps.

1. Résultats attendus :

* **Amélioration de l'efficacité administrative :** Réduction des tâches administratives manuelles grâce à l'automatisation des processus de gestion des étudiants et des cours.
* **Amélioration de la satisfaction des étudiants :** Grâce à une gestion optimisée des parcours scolaires, les étudiants bénéficieront d'une expérience plus fluide et personnalisée au sein de l'établissement.
* **Renforcement de la transparence et de la traçabilité :** Le système permettra un suivi précis des interactions entre les étudiants, les enseignants et l'administration, garantissant ainsi une meilleure traçabilité des activités académiques.

1. **Portée du projet**
2. Fonctionnalités pour la gestion des étudiants

* **Création d'étudiants :** Permet aux administrateurs de créer de nouveaux profils d'étudiants en saisissant des informations telles que nom, prénom, adresse email, adresse physique, numéro de téléphone et date de naissance.
* **Association des cours :** Permet d'associer chaque étudiant à ses cours respectifs, en enregistrant le thème du cours et le nombre d'heures correspondant.
* **Consultation des informations étudiantes :** Offre la possibilité de consulter les détails complets des étudiants, y compris leur progression académique, leurs inscriptions précédentes et actuelles, et leurs données personnelles.
* **Modification des informations étudiantes :** Autorise la mise à jour des informations des étudiants, telles que les coordonnées ou le statut scolaire, en cas de besoin.
* **Suppression d'étudiants :** Permet de supprimer un profil étudiant lorsque cela est nécessaire, en respectant les politiques et procédures de l'école.
* **Liste complète des étudiants :** Fournit une fonctionnalité permettant de visualiser la liste complète de tous les étudiants inscrits à l'école.

1. Fonctionnalités pour la gestion des enseignants

* **Création d'enseignants :** Permet aux administrateurs de créer de nouveaux profils d'enseignants en saisissant des informations telles que nom, prénom, adresse email, adresse physique, numéro de téléphone et matière enseignée.
* **Consultation des informations enseignantes :** Offre la possibilité de consulter les détails complets des enseignants, y compris leurs matières enseignées et leurs disponibilités.
* **Modification des informations enseignantes :** Autorise la mise à jour des informations des enseignants, telles que les coordonnées ou les matières enseignées.

1. Autres fonctionnalités

* **Sécurité et accès :** Le système garantit un accès sécurisé avec des rôles définis (directeur, responsable des études) pour contrôler l'accès aux fonctionnalités sensibles.
* **Automatisation des processus :** Automatise les tâches administratives liées à la gestion des étudiants et des enseignants pour améliorer l'efficacité globale de l'école.

1. Parties prenantes impliquées

* **Directeur de l'école :** Responsable ultime du système de gestion, s'assurant que les objectifs stratégiques de l'école sont atteints à travers l'application.
* **Responsable des études :** Gère les opérations quotidiennes liées à la gestion des étudiants et des cours, utilisant le système pour optimiser les processus académiques.
* **Administrateurs système :** Responsables de la configuration, du maintien et de la sécurité du système, garantissant un fonctionnement sans faille et sécurisé.
* **Enseignants :** Utilisent le système pour gérer leurs horaires, les matières enseignées et les interactions avec les étudiants.
* **Étudiants :** Bénéficient du système pour consulter leurs informations académiques, s'inscrire aux cours et interagir avec l'école.

# Description générale du système

1. **Contexte du système**

Le système de gestion proposé pour ENSITECH sera déployé au sein d'une école d’enseignement supérieur spécialisée dans les métiers du Numérique. Cette institution vise à offrir des programmes éducatifs de haute qualité dans le domaine numérique, en préparant les étudiants à relever les défis technologiques actuels et futurs.

1. Environnement institutionnel

ENSITECH opère dans un environnement éducatif dynamique et en constante évolution, où la technologie joue un rôle crucial dans l'enseignement et la gestion académique. L'école se concentre sur l'innovation pédagogique et la préparation professionnelle, cherchant à offrir une expérience d'apprentissage enrichissante et adaptée aux besoins du marché du travail numérique.

1. Utilisateurs principaux et leurs besoins

* **Directeur de l'école :**
  + Besoin de surveiller et de gérer efficacement l'ensemble des opérations académiques et administratives de l'école.
  + Requiert des outils pour évaluer les performances des étudiants, suivre la progression des cours et prendre des décisions stratégiques basées sur des données.
* **Responsable des études :**
  + Responsable de la gestion quotidienne des programmes d'études et de l'organisation des cours.
  + Doit pouvoir planifier les horaires, attribuer les cours aux enseignants et assurer le suivi des inscriptions des étudiants.
* **Enseignants :**
  + Ont besoin d'un accès facile aux informations des étudiants inscrits à leurs cours, y compris les notes, les présences et les progrès académiques.
  + Recherchent des outils pour communiquer avec les étudiants, planifier les cours et évaluer les performances académiques.
* **Étudiants :**
  + Ont besoin d'accéder à leurs horaires de cours, de gérer leurs inscriptions et de consulter leurs résultats académiques.
  + Recherchent une plateforme conviviale pour interagir avec l'école, soumettre des devoirs en ligne et accéder aux ressources pédagogiques.

1. Impact attendu sur les utilisateurs

L'introduction du système de gestion aura plusieurs impacts positifs sur les utilisateurs :

* **Pour le directeur et le responsable des études :**
  + Amélioration de la gestion globale de l'école avec des processus automatisés et une visibilité accrue sur les performances académiques.
  + Optimisation des ressources et prise de décisions stratégiques éclairées basées sur des données en temps réel.
* **Pour les enseignants :**
  + Simplification des tâches administratives liées à la gestion des cours et des étudiants.
  + Augmentation de l'efficacité pédagogique grâce à un accès rapide aux informations pertinentes sur les étudiants et les cours.
* **Pour les étudiants :**
  + Amélioration de l'expérience d'apprentissage avec un accès facile aux informations académiques et aux outils d'interaction avec l'école.
  + Renforcement de l'engagement des étudiants grâce à une communication efficace et à des services en ligne conviviaux.

1. **Fonctionnalités principales** 
   * **Gestion des Étudiants**

* Création, modification, et suppression des profils étudiants avec des informations détaillées (nom, prénom, adresse email, adresse, numéro de téléphone, date de naissance).
* Attribution automatique d'un identifiant unique à chaque étudiant.
* Association des étudiants à des cours spécifiques.
  + **Gestion des Cours**
* Création et modification des cours avec un thème défini et un nombre d'heures.
* Attribution des cours à des enseignants spécifiques.
  + **Gestion des Enseignants**
* Création, modification, et suppression des profils enseignants avec des informations complètes (nom, prénom, adresse email, adresse, numéro de téléphone, matière enseignée).
* Attribution automatique d'un identifiant unique à chaque enseignant.
  + **Gestion Académique**
* Consultation des informations détaillées sur les étudiants inscrits à chaque cours.
* Suivi des notes, des présences, et des progrès académiques des étudiants.
* Attribution et gestion des horaires de cours.
  + **Administration des Rôles et des Autorisations**
* Attribution de rôles spécifiques aux utilisateurs (directeur, responsable des études, enseignants).
* Contrôle des autorisations d'accès basé sur les rôles définis.
  + **Authentification et Sécurité**
* Mécanisme d'authentification sécurisé pour tous les utilisateurs.
* Gestion des sessions et des accès basés sur des mécanismes de sécurité avancés.
  + **Interface Utilisateur Conviviale**
* Interface intuitive et conviviale pour une navigation aisée.
* Prise en charge de différents dispositifs (ordinateurs de bureau, tablettes, smartphones).
  + **Rapports et Statistiques**
* Génération de rapports sur les performances académiques des étudiants.
* Analyse statistique des données pour des prises de décisions informées.
  + **Communication Interne**
* Outils de communication intégrés pour faciliter les échanges entre les enseignants, les étudiants et l'administration.
* Notifications automatisées pour les événements académiques importants.
  + **Sécurité et Confidentialité**
* Protection des données sensibles des étudiants et des enseignants.
* Respect des normes de sécurité et de confidentialité des données (GDPR, par exemple).

# Modèles conceptuels

1. **Modèle des cas d'utilisation**
2. Acteurs Principaux

* Directeur
* Responsable des Études
* Enseignant
* Étudiant

1. Cas d'Utilisation Principaux

* **Gérer les Étudiants**
  + **Créer Étudiant**
  + **Modifier Étudiant**
  + **Supprimer Étudiant**
  + **Consulter Informations Étudiant**
  + **Associer Étudiant à un Cours**
  + **Lister l’ensemble des Etudiants**
* **Gérer les Enseignants**
  + Créer Enseignant
  + Modifier Enseignant
  + Supprimer Enseignant
  + Consulter Informations Enseignant
  + Lister l’ensemble des Enseignants
* **Gérer les Cours**
  + Créer Cours
  + Modifier Cours
  + Consulter Informations Cours
  + Attribuer Enseignant à un Cours
  + Lister l’ensemble des cours
* **Gérer l'Administration**
  + Gérer Rôles et Autorisations
  + Générer Rapports Académiques
* **Authentification et Sécurité**
  + S'Authentifier
  + Gérer Sessions Utilisateur
* **Communication**
  + Envoyer Notifications
  + Consulter Messages

1. Diagramme de cas d’utilisation :

Voir Diagrammedecasd’utilisation\_Ensitech1

1. **Modèle des classes**
2. Classes Principales

* **Étudiant**
* **Enseignant**
* **Cours**
* **Directeur**
* **ResponsableDesEtudes**
* **Utilisateur** (classe parente pour la gestion des utilisateurs)

1. Attributs et Méthodes
2. **Étudiant**
   * Attributs :
     + identifiant: int
     + nom: String
     + prenom: String
     + adresseEmail: String
     + adresse: String
     + numeroTelephone: String
     + dateNaissance: Date
   * Méthodes :
     + consulterCours()
3. **Enseignant**
   * Attributs :
     + identifiant: int
     + nom: String
     + prenom: String
     + adresseEmail: String
     + adresse: String
     + numeroTelephone: String
     + matiereEnseignee: String
   * Méthodes :
     + attribuerCours()
     + consulterCours()
4. **Cours**
   * Attributs :
     + identifiant: int
     + theme: String
     + nombreHeures: int
   * Méthodes :
     + ajouterEtudiant()
     + retirerEtudiant()
     + consulterEtudiants()
5. **Directeur**
   * Hérite d’Utilisateur
   * Méthodes :
     + gererEtudiants()
     + gererEnseignants()
     + gererCours()
     + genererRapports()
6. **ResponsableDesEtudes**
   * Hérite d’Utilisateur
   * Méthodes :
     + gererEtudiants()
     + gererCours()
7. **Utilisateur**
   * Attributs :
     + identifiant: int
     + nom: String
     + prenom: String
     + adresseEmail: String
     + adresse: String
     + numeroTelephone: String
   * Méthodes :
     + sAuthentifier()
8. Relations

* **Directeur** et **ResponsableDesEtudes** héritent de la classe **Utilisateur**.
* Un **Étudiant** peut être associé à plusieurs **Cours**, et un **Cours** peut inclure plusieurs **Étudiants**.
* Un **Enseignant** peut être attribué à plusieurs **Cours**, mais chaque **Cours** est attribué à un seul **Enseignant**.

1. Diagramme de Classe

Voir Diagramme Classe Ensitech2

1. Spécifications des exigences fonctionnelles
2. **Description textuelle des cas d'utilisation :**
3. Cas d'utilisation 1 : **Créer un Étudiant**
4. **Objectif**

Permettre au directeur ou au responsable des études d'ajouter un nouvel étudiant dans le système.

1. **Acteurs concernés**

* Directeur
* Responsable des études

1. **Préconditions**

* L'utilisateur doit être authentifié en tant que directeur ou responsable des études.

1. **Postconditions**

* Un nouvel étudiant est ajouté au système avec un identifiant unique et ses informations personnelles.

1. **Scénario Nominal**
2. L'utilisateur se connecte au système.
3. L'utilisateur accède à la section "Gestion des étudiants".
4. L'utilisateur clique sur "Ajouter un étudiant".
5. Le système affiche un formulaire pour saisir les informations de l'étudiant.
6. L'utilisateur remplit le formulaire avec les informations suivantes :
   * Nom
   * Prénom
   * Adresse email
   * Adresse
   * Numéro de téléphone
   * Date de naissance
7. L'utilisateur soumet le formulaire.
8. Le système valide les informations et crée un nouvel étudiant avec un identifiant unique.
9. Le système confirme la création de l'étudiant et affiche les détails de l'étudiant nouvellement créé.
10. **Scénarios Alternatifs**

* **SA1 : Échec de validation des informations**
  + Étape 7 : Si les informations sont invalides ou manquantes, le système affiche un message d'erreur indiquant les champs à corriger.
  + L'utilisateur corrige les informations et soumet à nouveau le formulaire.

1. Cas d'utilisation 2 : **Associer un Cours à un Étudiant**
2. **Objectif**

* Permettre au directeur ou au responsable des études d'associer un cours à un étudiant.

1. **Acteurs concernés**

* Directeur
* Responsable des études

1. **Préconditions**

* L'utilisateur doit être authentifié en tant que directeur ou responsable des études.
* L'étudiant doit exister dans le système.
* Le cours doit exister dans le système.

1. **Postconditions**

* Le cours est associé à l'étudiant dans le système.

1. **Scénario Nominal**
2. L'utilisateur se connecte au système.
3. L'utilisateur accède à la section "Gestion des étudiants".
4. L'utilisateur sélectionne un étudiant.
5. L'utilisateur accède à la section "Cours associés".
6. L'utilisateur clique sur "Associer un cours".
7. Le système affiche une liste de cours disponibles.
8. L'utilisateur sélectionne un cours et clique sur "Associer".
9. Le système associe le cours à l'étudiant.
10. Le système confirme l'association et affiche les cours associés à l'étudiant.
11. **Scénarios Alternatifs**

* **SA1 : Étudiant ou cours inexistant**
  + Étape 3 ou 6 : Si l'étudiant ou le cours n'existe pas dans le système, le système affiche un message d'erreur.
  + L'utilisateur retourne à la liste des étudiants ou des cours pour effectuer une nouvelle sélection.

1. Cas d'utilisation 3 : **Lire les informations d'un Étudiant**
2. **Objectif**

Permettre au directeur ou au responsable des études de consulter les informations d'un étudiant.

1. **Acteurs concernés**

* Directeur
* Responsable des études

1. **Préconditions**

* L'utilisateur doit être authentifié en tant que directeur ou responsable des études.

1. **Postconditions**

* Les informations de l'étudiant sont affichées à l'utilisateur.

1. **Scénario Nominal**
2. L'utilisateur se connecte au système.
3. L'utilisateur accède à la section "Gestion des étudiants".
4. L'utilisateur sélectionne un étudiant.
5. Le système affiche les informations détaillées de l'étudiant sélectionné.
6. **Scénarios Alternatifs**

* **SA1 : Étudiant inexistant**
  + Étape 3 : Si l'étudiant n'existe pas dans le système, le système affiche un message d'erreur.
  + L'utilisateur retourne à la liste des étudiants pour effectuer une nouvelle sélection.

1. Cas d'utilisation 4 : **Modifier les informations d'un Étudiant**
2. **Objectif**

* Permettre au directeur ou au responsable des études de modifier les informations d'un étudiant.

1. **Acteurs concernés**

* Directeur
* Responsable des études

1. **Préconditions**

* L'utilisateur doit être authentifié en tant que directeur ou responsable des études.
* L'étudiant doit exister dans le système.

1. **Postconditions**

* Les informations de l'étudiant sont mises à jour dans le système.

1. **Scénario Nominal**
2. L'utilisateur se connecte au système.
3. L'utilisateur accède à la section "Gestion des étudiants".
4. L'utilisateur sélectionne un étudiant.
5. Le système affiche les informations détaillées de l'étudiant sélectionné.
6. L'utilisateur clique sur "Modifier".
7. Le système affiche un formulaire pré-rempli avec les informations de l'étudiant.
8. L'utilisateur modifie les informations souhaitées.
9. L'utilisateur soumet le formulaire.
10. Le système valide les informations et met à jour les informations de l'étudiant.
11. Le système confirme la mise à jour et affiche les nouvelles informations de l'étudiant.
12. **Scénarios Alternatifs**

* **SA1 : Échec de validation des informations**
  + Étape 9 : Si les informations modifiées sont invalides ou manquantes, le système affiche un message d'erreur indiquant les champs à corriger.
  + L'utilisateur corrige les informations et soumet à nouveau le formulaire.

1. Cas d'utilisation 5 : **Supprimer un Étudiant**
2. **Objectif**

* Permettre au directeur ou au responsable des études de supprimer un étudiant du système.

1. **Acteurs concernés**

* Directeur
* Responsable des études

1. **Préconditions**

* L'utilisateur doit être authentifié en tant que directeur ou responsable des études.
* L'étudiant doit exister dans le système.

1. **Postconditions**

* L'étudiant est supprimé du système.

1. **Scénario Nominal**
2. L'utilisateur se connecte au système.
3. L'utilisateur accède à la section "Gestion des étudiants".
4. L'utilisateur sélectionne un étudiant.
5. Le système affiche les informations détaillées de l'étudiant sélectionné.
6. L'utilisateur clique sur "Supprimer".
7. Le système demande une confirmation de suppression.
8. L'utilisateur confirme la suppression.
9. Le système supprime l'étudiant du système.
10. Le système confirme la suppression et retourne à la liste des étudiants.
11. **Scénarios Alternatifs**

* **SA1 : Annulation de la suppression**
  + Étape 7 : Si l'utilisateur annule la suppression, le système retourne à la vue des informations de l'étudiant sans effectuer de modifications.

1. Cas d'utilisation 6 : **Lister l'ensemble des Étudiants**
2. **Objectif**

* Permettre au directeur ou au responsable des études de consulter la liste complète des étudiants inscrits.

1. **Acteurs concernés**

* Directeur
* Responsable des études

1. **Préconditions**

* L'utilisateur doit être authentifié en tant que directeur ou responsable des études.

1. **Postconditions**

* La liste complète des étudiants est affichée à l'utilisateur.

1. **Scénario Nominal**
2. L'utilisateur se connecte au système.
3. L'utilisateur accède à la section "Gestion des étudiants".
4. Le système affiche la liste complète des étudiants inscrits avec leurs informations principales (nom, prénom, identifiant, email).
5. **Scénarios Alternatifs**

* Aucun scénario alternatif pour ce cas d'utilisation.

1. **Exigences spécifiques**
2. **Fonctionnalité 1 : Créer un Étudiant**
3. **Entrées Requises**

* Nom
* Prénom
* Adresse email
* Adresse
* Numéro de téléphone
* Date de naissance

1. **Sorties Attendues**

* Confirmation de création de l'étudiant avec un identifiant unique
* Affichage des détails de l'étudiant nouvellement créé

1. **Comportement du Système**
2. L'utilisateur authentifié accède au formulaire de création d'un étudiant.
3. L'utilisateur saisit les informations requises et soumet le formulaire.
4. Le système vérifie que toutes les informations obligatoires sont présentes et valides.
5. Si les informations sont valides, le système attribue un identifiant unique à l'étudiant et enregistre les informations dans la base de données.
6. Le système affiche une confirmation de la création et les détails de l'étudiant.
7. Si les informations sont invalides ou incomplètes, le système affiche un message d'erreur et invite l'utilisateur à corriger les informations.
8. **Fonctionnalité 2 : Associer un Cours à un Étudiant**
9. **Entrées Requises**

* Identifiant de l'étudiant
* Identifiant du cours

1. **Sorties Attendues**

* Confirmation de l'association du cours à l'étudiant
* Affichage des cours associés à l'étudiant

1. **Comportement du Système**
2. L'utilisateur authentifié sélectionne un étudiant et accède à la section des cours associés.
3. L'utilisateur sélectionne un cours à associer à l'étudiant.
4. Le système vérifie que l'étudiant et le cours existent dans la base de données.
5. Si les informations sont valides, le système associe le cours à l'étudiant et enregistre cette association.
6. Le système affiche une confirmation de l'association et met à jour la liste des cours associés à l'étudiant.
7. Si l'étudiant ou le cours n'existe pas, le système affiche un message d'erreur.
8. **Fonctionnalité 3 : Lire les informations d'un Étudiant**
9. **Entrées Requises**

* Identifiant de l'étudiant

1. **Sorties Attendues**

* Affichage des informations détaillées de l'étudiant

1. **Comportement du Système**
2. L'utilisateur authentifié sélectionne un étudiant à partir de la liste des étudiants.
3. Le système récupère les informations détaillées de l'étudiant depuis la base de données.
4. Le système affiche les informations de l'étudiant à l'utilisateur.
5. **Fonctionnalité 4 : Modifier les informations d'un Étudiant**
6. **Entrées Requises**

* Identifiant de l'étudiant
* Informations à modifier (Nom, Prénom, Adresse email, Adresse, Numéro de téléphone, Date de naissance)

1. **Sorties Attendues**

* Confirmation de la mise à jour des informations de l'étudiant
* Affichage des nouvelles informations de l'étudiant

1. **Comportement du Système**
2. L'utilisateur authentifié sélectionne un étudiant à partir de la liste des étudiants et accède aux informations détaillées.
3. L'utilisateur clique sur "Modifier" pour accéder au formulaire pré-rempli.
4. L'utilisateur modifie les informations souhaitées et soumet le formulaire.
5. Le système vérifie que les nouvelles informations sont valides.
6. Si les informations sont valides, le système met à jour les informations de l'étudiant dans la base de données.
7. Le système affiche une confirmation de la mise à jour et les nouvelles informations de l'étudiant.
8. Si les informations sont invalides, le système affiche un message d'erreur et invite l'utilisateur à corriger les informations.
9. **Fonctionnalité 5 : Supprimer un Étudiant**
10. **Entrées Requises**

* Identifiant de l'étudiant

1. **Sorties Attendues**

* Confirmation de la suppression de l'étudiant
* Mise à jour de la liste des étudiants

1. **Comportement du Système**
2. L'utilisateur authentifié sélectionne un étudiant à partir de la liste des étudiants et accède aux informations détaillées.
3. L'utilisateur clique sur "Supprimer".
4. Le système demande une confirmation de suppression.
5. L'utilisateur confirme la suppression.
6. Le système supprime l'étudiant de la base de données.
7. Le système affiche une confirmation de la suppression et met à jour la liste des étudiants.
8. **Fonctionnalité 6 : Lister l'ensemble des Étudiants**
9. **Entrées Requises**

* Aucune (la liste est générée pour tous les étudiants)

1. **Sorties Attendues**

* Affichage de la liste complète des étudiants inscrits avec leurs informations principales (nom, prénom, identifiant, email)

1. **Comportement du Système**
2. L'utilisateur authentifié accède à la section "Gestion des étudiants".
3. Le système récupère la liste complète des étudiants depuis la base de données.
4. Le système affiche la liste des étudiants avec leurs informations principales.
5. Interfaces utilisateur
6. **Conception de l'interface utilisateur**

**1. Page de Connexion**

1. **Description**

La page de connexion permet aux utilisateurs autorisés (directeur et responsable des études) d'accéder au système. Elle doit être simple et sécurisée.

1. **Éléments de l'interface**

* **Champ d'identifiant** : Zone de texte pour saisir l'identifiant de l'utilisateur.
* **Champ de mot de passe** : Zone de texte pour saisir le mot de passe de l'utilisateur.
* **Bouton de connexion** : Bouton pour soumettre les informations de connexion.
* **Lien de mot de passe oublié** : Lien pour réinitialiser le mot de passe en cas d'oubli.

**2. Tableau de Bord Principal**

1. **Description**

Le tableau de bord principal présente un aperçu des informations clés, telles que le nombre total d'étudiants, de cours, et les notifications importantes.

1. **Éléments de l'interface**

* **Menu de navigation** : Liens vers les différentes sections du système (Gestion des étudiants, Gestion des enseignants, Gestion des cours, etc.).
* **Cartes d'aperçu** : Petits panneaux affichant des statistiques clés (nombre d'étudiants, nombre d'enseignants, etc.).
* **Zone de notification** : Panneau affichant les notifications récentes et importantes.

**3. Page de Gestion des Étudiants**

1. **Description**

Cette page permet de gérer les informations des étudiants : créer, lire, modifier et supprimer des profils d'étudiants.

**Éléments de l'interface**

* **Tableau des étudiants** : Tableau listant tous les étudiants avec des colonnes pour l'identifiant, le nom, le prénom, et les actions (voir, modifier, supprimer).
* **Bouton "Ajouter un étudiant"** : Bouton pour accéder au formulaire de création d'un nouvel étudiant.
* **Actions sur chaque ligne** : Icônes ou boutons pour voir, modifier, ou supprimer un étudiant spécifique.

**4. Formulaire de Création/Mise à Jour d'un Étudiant**

1. **Description**

Ce formulaire permet de saisir ou de modifier les informations d'un étudiant.

1. **Éléments de l'interface**

* **Champs de texte** : Champs pour l'identifiant, le nom, le prénom, l'adresse email, l'adresse, le numéro de téléphone, et la date de naissance.
* **Bouton de soumission** : Bouton pour soumettre le formulaire et enregistrer les informations.
* **Bouton d'annulation** : Bouton pour annuler et retourner à la page précédente sans enregistrer les modifications.

**5. Page de Gestion des Cours**

1. **Description**

Cette page permet de gérer les cours proposés par l'école : créer, lire, modifier et supprimer des cours.

1. **Éléments de l'interface**

* **Tableau des cours** : Tableau listant tous les cours avec des colonnes pour le thème, le nombre d'heures, et les actions (voir, modifier, supprimer).
* **Bouton "Ajouter un cours"** : Bouton pour accéder au formulaire de création d'un nouveau cours.
* **Actions sur chaque ligne** : Icônes ou boutons pour voir, modifier, ou supprimer un cours spécifique.

**6. Formulaire de Création/Mise à Jour d'un Cours**

1. **Description**

Ce formulaire permet de saisir ou de modifier les informations d'un cours.

1. **Éléments de l'interface**

* **Champs de texte** : Champs pour le thème du cours et le nombre d'heures.
* **Bouton de soumission** : Bouton pour soumettre le formulaire et enregistrer les informations.
* **Bouton d'annulation** : Bouton pour annuler et retourner à la page précédente sans enregistrer les modifications.

**7. Page de Gestion des Enseignants**

1. **Description**

Cette page permet de gérer les informations des enseignants : créer, lire, modifier et supprimer des profils d'enseignants.

1. **Éléments de l'interface**

* **Tableau des enseignants** : Tableau listant tous les enseignants avec des colonnes pour l'identifiant, le nom, le prénom, la matière enseignée, et les actions (voir, modifier, supprimer).
* **Bouton "Ajouter un enseignant"** : Bouton pour accéder au formulaire de création d'un nouvel enseignant.
* **Actions sur chaque ligne** : Icônes ou boutons pour voir, modifier, ou supprimer un enseignant spécifique.

**8. Formulaire de Création/Mise à Jour d'un Enseignant**

1. **Description**

Ce formulaire permet de saisir ou de modifier les informations d'un enseignant.

1. **Éléments de l'interface**

* **Champs de texte** : Champs pour l'identifiant, le nom, le prénom, l'adresse email, l'adresse, le numéro de téléphone, et la matière enseignée.
* **Bouton de soumission** : Bouton pour soumettre le formulaire et enregistrer les informations.
* **Bouton d'annulation** : Bouton pour annuler et retourner à la page précédente sans enregistrer les modifications.

# Exigences non fonctionnelles

1. **Performances du système :**

1. **Capacité de Charge**

1. **Description**

Le système doit pouvoir gérer efficacement un grand nombre de connexions simultanées sans dégradation des performances.

1. **Exigences spécifiques**

* **Nombre maximum d'utilisateurs simultanés** : Le système doit pouvoir supporter jusqu'à 500 utilisateurs simultanés sans perte de performance.
* **Nombre maximum de requêtes par seconde** : Le système doit pouvoir traiter jusqu'à 100 requêtes par seconde.

**2. Temps de Réponse**

1. **Description**

Le système doit répondre rapidement aux actions des utilisateurs pour garantir une expérience utilisateur positive.

1. **Exigences spécifiques**

* **Temps de réponse pour l'affichage des pages** : Les pages principales (tableau de bord, gestion des étudiants, etc.) doivent se charger en moins de 2 secondes.
* **Temps de réponse pour les actions CRUD (Créer, Lire, Mettre à jour, Supprimer)** : Les opérations de création, lecture, mise à jour et suppression doivent être complétées en moins de 1 seconde.

**3. Disponibilité**

1. **Description**

Le système doit être disponible et fonctionnel à tout moment pour éviter les interruptions de service.

1. **Exigences spécifiques**

* **Taux de disponibilité** : Le système doit avoir un taux de disponibilité de 99,9%, soit une indisponibilité maximale de 8,76 heures par an.

**4. Évolutivité**

1. **Description**

Le système doit pouvoir évoluer pour supporter une augmentation du nombre d'utilisateurs et de données sans nécessiter de refonte majeure.

1. **Exigences spécifiques**

* **Évolutivité horizontale** : Le système doit être conçu pour permettre l'ajout de serveurs supplémentaires pour gérer une charge accrue.
* **Évolutivité verticale** : Le système doit permettre l'amélioration des performances des serveurs existants (augmentation de la mémoire, du CPU, etc.) pour gérer une charge accrue.

**5. Fiabilité**

1. **Description**

Le système doit être fiable et capable de fonctionner correctement sous différentes conditions sans erreur.

1. **Exigences spécifiques**

* **Taux d'erreur** : Le taux d'erreur du système doit être inférieur à 0,1%.
* **Récupération en cas de panne** : En cas de panne, le système doit pouvoir se rétablir et reprendre les opérations normales en moins de 5 minutes.

**6. Sécurité**

1. **Description**

Le système doit garantir la sécurité des données et des opérations pour protéger les informations sensibles des étudiants, des enseignants et de l'école.

1. **Exigences spécifiques**

* **Temps de réponse pour les vérifications de sécurité** : Les vérifications de sécurité (authentification, autorisation) doivent être effectuées en moins de 500 millisecondes.
* **Résistance aux attaques** : Le système doit être résistant aux attaques courantes (DDoS, injection SQL, XSS, etc.) et capable de les détecter et de les bloquer en temps réel.

**7. Maintenance**

1. **Description**

Le système doit être conçu pour permettre une maintenance aisée et minimiser les interruptions de service pendant les opérations de maintenance.

1. **Exigences spécifiques**

* **Temps de maintenance** : Les opérations de maintenance planifiées doivent être effectuées en dehors des heures de pointe et doivent être complétées en moins de 2 heures.
* **Impact sur les utilisateurs** : Les interruptions de service pendant les opérations de maintenance doivent être minimales et communiquées à l'avance aux utilisateurs.

1. **Sécurité :**

**1. Authentification**

**Description**

L'authentification garantit que seuls les utilisateurs autorisés peuvent accéder au système.

**Exigences spécifiques**

* **Méthode d'authentification principale** : Utilisation d'un nom d'utilisateur et d'un mot de passe pour accéder au système.
* **Authentification à deux facteurs (2FA)** : Implémentation de 2FA pour ajouter une couche de sécurité supplémentaire, en utilisant des applications comme Google Authenticator ou des messages SMS.
* **Gestion des mots de passe** : Les mots de passe doivent être cryptés et stockés de manière sécurisée. Les utilisateurs doivent être obligés de choisir des mots de passe forts (minimum 8 caractères, comprenant des lettres majuscules, des lettres minuscules, des chiffres et des caractères spéciaux).
* **Réinitialisation de mot de passe** : Un processus sécurisé pour la réinitialisation des mots de passe, incluant la vérification par e-mail ou SMS.

**2. Autorisation**

**Description**

L'autorisation contrôle les actions que les utilisateurs peuvent effectuer une fois authentifiés.

**Exigences spécifiques**

* **Gestion des rôles** : Mise en place d'un système de gestion des rôles (par exemple, directeur, responsable des études) pour contrôler l'accès aux différentes fonctionnalités du système.
* **Contrôle d'accès basé sur les rôles (RBAC)** : Chaque utilisateur doit avoir un rôle spécifique qui détermine ses droits d'accès. Par exemple, seuls le directeur et le responsable des études peuvent créer, modifier ou supprimer des étudiants.
* **Permissions granulaires** : Définir des permissions granulaires pour des actions spécifiques, telles que la création de nouveaux cours, la modification des informations des enseignants, etc.

**3. Protection des Données Sensibles**

**Description**

La protection des données sensibles vise à garantir que les informations personnelles et académiques sont sécurisées contre les accès non autorisés et les fuites de données.

**Exigences spécifiques**

* **Chiffrement des données en transit** : Utilisation de protocoles SSL/TLS pour sécuriser les données échangées entre le client et le serveur.
* **Chiffrement des données au repos** : Les données sensibles stockées dans la base de données doivent être chiffrées à l'aide d'algorithmes de chiffrement forts (par exemple, AES-256).
* **Accès sécurisé aux bases de données** : Restreindre l'accès aux bases de données aux seuls administrateurs autorisés et mettre en place des audits réguliers des accès.
* **Journalisation et surveillance** : Enregistrement de toutes les tentatives de connexion, les actions des utilisateurs, et les modifications des données sensibles pour permettre un audit de sécurité.
* **Sauvegardes régulières** : Mise en place de sauvegardes régulières et sécurisées des données pour garantir la récupération en cas de perte de données.

**4. Prévention et Détection des Intrusions**

**Description**

Mettre en place des mécanismes pour prévenir, détecter et répondre aux tentatives d'intrusion et aux attaques.

**Exigences spécifiques**

* **Pare-feu et protection DDoS** : Utilisation de pare-feu pour protéger le système contre les attaques externes et mise en place de mesures de protection contre les attaques par déni de service distribué (DDoS).
* **Détection des intrusions** : Mise en place de systèmes de détection et de prévention des intrusions (IDS/IPS) pour surveiller et réagir aux activités suspectes.
* **Mises à jour et correctifs** : Application régulière des mises à jour de sécurité et des correctifs pour toutes les composantes du système (serveurs, logiciels, bibliothèques).

**5. Formation et Sensibilisation**

**Description**

Assurer que tous les utilisateurs sont formés et conscients des meilleures pratiques en matière de sécurité.

**Exigences spécifiques**

* **Formation des utilisateurs** : Formation régulière des utilisateurs sur les bonnes pratiques de sécurité, y compris la gestion des mots de passe, la détection des tentatives de phishing, et l'utilisation sécurisée du système.
* **Politiques de sécurité** : Élaboration et diffusion de politiques de sécurité claires et compréhensibles pour tous les utilisateurs.

1. **Fiabilité :**

**1. Tolérance aux Pannes**

**Description**

La tolérance aux pannes vise à garantir que le système peut continuer à fonctionner correctement même en cas de défaillance de certaines de ses composantes.

**Exigences spécifiques**

* **Redondance des Composants** : Mise en place de systèmes redondants pour les composants critiques tels que les serveurs, les bases de données et les systèmes de stockage. Chaque composant critique doit avoir un double ou un miroir pour prendre le relais en cas de panne.
* **Failover Automatique** : Mise en œuvre de mécanismes de basculement automatique pour garantir la continuité du service sans interruption perceptible pour les utilisateurs en cas de défaillance matérielle ou logicielle.
* **Sauvegardes Régulières** : Réalisation de sauvegardes régulières et automatiques des données critiques pour permettre une récupération rapide et complète en cas de perte de données ou de défaillance du système.
* **Monitoring et Alertes** : Mise en place d’un système de surveillance en temps réel pour détecter les anomalies et les pannes, avec des alertes automatiques envoyées aux administrateurs pour une intervention rapide.
* **Tests de Résilience** : Effectuer des tests réguliers de tolérance aux pannes, y compris des tests de scénario de défaillance, pour s'assurer que le système peut gérer les pannes et se rétablir rapidement.

**2. Disponibilité du Système**

**Description**

L'objectif est de garantir que le système est accessible et opérationnel pour les utilisateurs à tout moment, minimisant les temps d'arrêt et maximisant la disponibilité.

**Exigences spécifiques**

* **Haute Disponibilité (HA)** : Conception du système pour assurer une disponibilité de 99,9% ou plus, impliquant l'utilisation de clusters de serveurs, des configurations de répartition de charge et des systèmes de basculement automatique.
* **Maintenance Planifiée** : Planification des maintenances régulières en dehors des heures de pointe pour minimiser l'impact sur les utilisateurs, avec des notifications préalables aux utilisateurs concernant les périodes de maintenance.
* **SLA (Service Level Agreement)** : Définition d’un SLA avec des objectifs clairs en matière de temps de disponibilité et de récupération, et des pénalités pour non-respect de ces objectifs.
* **Load Balancing** : Utilisation de mécanismes de répartition de charge pour assurer une distribution équilibrée des demandes utilisateur sur plusieurs serveurs, évitant ainsi les surcharges et garantissant des temps de réponse optimaux.
* **Capacité Évolutive (Scalabilité)** : Conception du système pour permettre une augmentation dynamique des ressources en réponse à une augmentation de la charge utilisateur, assurant ainsi la disponibilité et la performance même en période de forte demande.

**Mécanismes Additionnels**

* **Virtualisation et Conteneurisation** : Utilisation de technologies de virtualisation et de conteneurisation (comme Docker et Kubernetes) pour faciliter la gestion des ressources et améliorer la tolérance aux pannes et la disponibilité.
* **Réplication Géographique** : Mise en œuvre de la réplication des données dans plusieurs centres de données géographiquement dispersés pour assurer la continuité des services en cas de catastrophe naturelle ou d'autres incidents majeurs.

1. **Interfaces système**

**1. Système de Gestion des Inscriptions**

**Description**

L'intégration avec un système de gestion des inscriptions permettra de synchroniser les informations des nouveaux étudiants et de maintenir à jour les données des inscrits.

**Exigences spécifiques**

* **API RESTful** : Utiliser une API RESTful pour faciliter la communication entre le système de gestion des inscriptions et le système ENSITECH.
* **Authentification OAuth** : Mettre en place un système d'authentification OAuth pour sécuriser les échanges de données entre les systèmes.
* **Synchronisation en Temps Réel** : Assurer que les informations des étudiants sont synchronisées en temps réel pour éviter les incohérences de données.
* **Gestion des Erreurs** : Implémenter des mécanismes de gestion des erreurs pour traiter les cas où les données ne peuvent pas être synchronisées, avec des notifications automatiques aux administrateurs.

**2. Système de Gestion des Paiements**

**Description**

L'intégration avec un système de gestion des paiements est essentielle pour traiter les frais de scolarité, les paiements des cours et d'autres transactions financières.

**Exigences spécifiques**

* **API de Paiement** : Utiliser les API fournies par les systèmes de paiement (par exemple, PayPal, Stripe, ou des services de paiement mobile locaux).
* **Sécurité des Transactions** : Assurer que toutes les transactions sont sécurisées en utilisant des protocoles SSL/TLS et le chiffrement des données sensibles.
* **Confirmation de Paiement** : Implémenter des fonctionnalités de confirmation de paiement en temps réel pour informer les étudiants et le personnel administratif.
* **Comptabilité Intégrée** : Synchroniser les données des paiements avec le système comptable pour une gestion financière transparente et précise.

**3. Système de Gestion des Notes et des Evaluations**

**Description**

L'intégration avec un système de gestion des notes et des évaluations permettra une mise à jour automatique des résultats académiques des étudiants.

**Exigences spécifiques**

* **Importation/Exportation des Données** : Assurer que les données des notes et des évaluations peuvent être importées et exportées facilement entre les systèmes.
* **API RESTful** : Utiliser des API RESTful pour faciliter l'intégration et la communication entre les systèmes.
* **Mises à Jour Automatiques** : Mettre en place des processus pour mettre à jour automatiquement les notes des étudiants dès qu'elles sont disponibles.
* **Notifications Automatiques** : Envoyer des notifications automatiques aux étudiants et aux enseignants lorsque les notes sont mises à jour ou disponibles.

**4. Système de Gestion des Bibliothèques**

**Description**

L'intégration avec un système de gestion des bibliothèques permettra aux étudiants et aux enseignants d'accéder facilement aux ressources pédagogiques.

**Exigences spécifiques**

* **Accès SSO (Single Sign-On)** : Implémenter un accès SSO pour permettre aux utilisateurs de se connecter une seule fois et d'accéder à plusieurs systèmes sans devoir se reconnecter.
* **Recherche Intégrée** : Fournir une fonctionnalité de recherche intégrée pour permettre aux utilisateurs de rechercher des ressources pédagogiques directement depuis le système ENSITECH.
* **Gestion des Emprunts** : Synchroniser les données des emprunts de livres et de ressources pédagogiques pour une gestion efficace des emprunts et des retours.

**5. Système de Messagerie et de Collaboration**

**Description**

L'intégration avec un système de messagerie et de collaboration (par exemple, Microsoft Teams, Slack, ou Google Workspace) facilitera la communication entre les étudiants, les enseignants et le personnel administratif.

**Exigences spécifiques**

* **Notifications en Temps Réel** : Assurer que les notifications importantes (comme les mises à jour de cours, les résultats d'examens, etc.) sont envoyées en temps réel via les systèmes de messagerie.
* **Calendriers Intégrés** : Synchroniser les calendriers des cours et des événements académiques avec les calendriers des systèmes de collaboration.
* **Accès aux Documents** : Permettre l'accès et le partage de documents pédagogiques et administratifs directement via les systèmes de collaboration.

**6. Exigences Générales d'Intégration**

**Interopérabilité**

* Assurer que les systèmes peuvent échanger des données de manière fluide et cohérente, en utilisant des standards ouverts et des API bien documentées.

**Sécurité des Données**

* Mettre en place des mesures de sécurité rigoureuses pour protéger les données sensibles lors des échanges entre systèmes, en utilisant des protocoles de chiffrement et des méthodes d'authentification sécurisées.

**Scalabilité**

* Garantir que les intégrations peuvent évoluer pour gérer une augmentation du nombre d'utilisateurs et de transactions sans compromettre les performances du système.

**Fiabilité**

* Implémenter des mécanismes de reprise sur erreur et de journalisation pour assurer la fiabilité des intégrations et permettre une surveillance et un dépannage efficaces.

**Maintenance et Support**

* Prévoir des processus de maintenance réguliers pour vérifier et mettre à jour les intégrations, ainsi que des supports techniques pour gérer les problèmes et les évolutions des systèmes intégrés.

# Contraintes techniques

1. **Environnement technologique :**

**1. Langages de Programmation**

**Backend**

* **Python** : Utilisé pour sa simplicité, sa robustesse et son large éventail de bibliothèques, notamment Django ou Flask pour le développement rapide d'applications web.
* **Java** : Choisi pour ses performances, sa scalabilité et sa sécurité, avec des frameworks comme Spring Boot.

**Frontend**

* **JavaScript** : Utilisé avec des frameworks modernes tels que React.js ou Angular pour la création d'interfaces utilisateur réactives et interactives.
* **TypeScript** : Superset de JavaScript, apporte des avantages de typage statique, améliorant la qualité et la maintenabilité du code.

**2. Bases de Données**

**Relationnelles**

* **PostgreSQL** : Préféré pour sa fiabilité, ses performances et ses capacités avancées de gestion de données.

**NoSQL**

* **MongoDB** : Utilisé pour les données non structurées et pour les scénarios nécessitant une scalabilité horizontale.

**3. Serveurs et Hébergement**

**Serveurs Web**

* **Nginx** : Utilisé comme serveur web et reverse proxy pour sa performance et sa faible utilisation de ressources.
* **Apache** : Option alternative, offrant une grande flexibilité et une large communauté de support.

**Hébergement Cloud**

* **Amazon Web Services (AWS)** : Fournit une infrastructure scalable et flexible, avec des services comme EC2, S3 et RDS pour le déploiement et la gestion des applications.
* **Microsoft Azure** : Alternative à AWS, offrant des services similaires pour l'hébergement et la gestion des applications cloud.
* **Google Cloud Platform (GCP)** : Fournit des services cloud compétitifs avec une excellente infrastructure pour le déploiement des applications.

**4. Sécurité**

**Authentification et Autorisation**

* **OAuth 2.0** : Standard pour l'authentification et l'autorisation, garantissant une gestion sécurisée des accès.
* **JWT (JSON Web Tokens)** : Utilisé pour sécuriser les sessions utilisateur et faciliter la gestion des permissions.

**Protection des Données**

* **SSL/TLS** : Protocoles de chiffrement pour sécuriser les communications entre le client et le serveur.
* **Chiffrement des Données** : Utilisation de bibliothèques comme OpenSSL pour le chiffrement des données sensibles au repos et en transit.

**5. DevOps et CI/CD**

**Gestion du Code Source**

* **Git** : Utilisation de plateformes comme GitHub ou GitLab pour la gestion du code source et la collaboration.

**Intégration et Déploiement Continus**

* **Jenkins** : Serveur open source pour l'automatisation des processus de build, test et déploiement.
* **Travis CI** : Alternative pour l'intégration continue, particulièrement avec GitHub.

**Conteneurisation et Orchestration**

* **Docker** : Pour créer, déployer et gérer des conteneurs, facilitant la portabilité des applications.
* **Kubernetes** : Pour l'orchestration des conteneurs, assurant la scalabilité et la gestion automatique des déploiements.

**6. Frameworks et Bibliothèques**

**Backend Frameworks**

* **Django** : Framework Python complet pour le développement rapide d'applications web sécurisées et évolutives.
* **Spring Boot** : Framework Java facilitant la création de microservices et d'applications web robustes.

**Frontend Frameworks**

* **React.js** : Bibliothèque JavaScript pour la création d'interfaces utilisateur dynamiques et performantes.
* **Angular** : Framework complet pour le développement d'applications web riches et modulaires.

**Bibliothèques de Tests**

* **JUnit** : Pour les tests unitaires en Java.
* **pytest** : Pour les tests unitaires en Python.
* **Selenium** : Pour les tests automatisés des interfaces utilisateur.

**7. Monitoring et Log Management**

**Surveillance**

* **Prometheus** : Pour la surveillance des applications et des infrastructures, offrant des capacités de monitoring et d'alerte.
* **Grafana** : Pour la visualisation des données de surveillance, avec des tableaux de bord interactifs.

**Gestion des Logs**

* **ELK Stack (Elasticsearch, Logstash, Kibana)** : Pour la collecte, la recherche et l'analyse des logs en temps réel.
* **Splunk** : Alternative puissante pour la gestion et l'analyse des logs.

1. **Normes et standards :**

**Normes de Sécurité**

**ISO/IEC 27001**

* **Description** : Norme internationale sur la gestion de la sécurité de l'information.
* **Application** : Implémenter un Système de Management de la Sécurité de l'Information (SMSI) pour protéger les données sensibles et garantir leur confidentialité, intégrité et disponibilité.

**OWASP Top Ten**

* **Description** : Liste des 10 principales vulnérabilités de sécurité des applications web identifiées par l'Open Web Application Security Project (OWASP).
* **Application** : Effectuer des revues de sécurité régulières et intégrer les meilleures pratiques OWASP dans le cycle de développement pour prévenir les vulnérabilités courantes comme les injections SQL, les failles XSS, etc.

**PCI DSS**

* **Description** : Norme de sécurité pour les entreprises qui gèrent des cartes de crédit.
* **Application** : Si le système implique des transactions par carte de crédit, respecter les exigences de PCI DSS pour sécuriser les données de paiement.

**Normes de Confidentialité**

**GDPR (General Data Protection Regulation)**

* **Description** : Règlement de l'Union Européenne sur la protection des données personnelles des citoyens européens.
* **Application** : Assurer que toutes les données personnelles des étudiants, enseignants et autres utilisateurs sont traitées conformément aux principes de protection des données (transparence, minimisation des données, etc.) et aux droits des individus (droit d'accès, de rectification, de suppression, etc.).

**HIPAA (Health Insurance Portability and Accountability Act)**

* **Description** : Législation américaine pour la protection des informations de santé.
* **Application** : Si le système gère des données de santé des utilisateurs, garantir que ces informations sont protégées conformément aux exigences HIPAA.

**Normes de Développement**

**ISO/IEC 25010**

* **Description** : Norme pour l'évaluation de la qualité des logiciels.
* **Application** : Assurer que le système respecte les caractéristiques de qualité telles que la fonctionnalité, la fiabilité, l'utilisabilité, l'efficacité, la maintenabilité et la portabilité.

**IEEE 830**

* **Description** : Norme pour la spécification des exigences logicielles.
* **Application** : Structurer et rédiger les spécifications des exigences logicielles conformément à cette norme pour garantir la clarté, la cohérence et la complétude.

**Normes d'Interopérabilité**

**RESTful API Design**

* **Description** : Principes de conception pour les API RESTful.
* **Application** : Développer les API du système en suivant les meilleures pratiques RESTful pour assurer l'interopérabilité et la facilité d'intégration avec d'autres systèmes.

**XML/JSON Standards**

* **Description** : Formats de données pour l'échange d'informations.
* **Application** : Utiliser des formats standardisés comme XML et JSON pour l'échange de données entre le système ENSITECH et d'autres systèmes.

1. Validation et vérification
2. **Critères de validation**

**Critères de Validation du Système**

**Tests Fonctionnels**

1. **Création d'Étudiant**
   * **Objectif** : Vérifier que le système permet la création d'un nouvel étudiant avec toutes les informations requises.
   * **Critères de Validation** :
     + L'étudiant est créé avec succès dans la base de données.
     + Toutes les données obligatoires sont saisies correctement.
     + Confirmation de la génération d'un identifiant unique pour l'étudiant.
2. **Association d'un Cours à un Étudiant**
   * **Objectif** : Tester la fonctionnalité permettant d'associer un cours à un étudiant inscrit.
   * **Critères de Validation** :
     + Sélection correcte du cours parmi ceux disponibles dans le système.
     + Vérification que l'étudiant est lié au cours dans la base de données.
     + Assurance que l'association respecte les règles de cohérence établies (par exemple, nombre maximum de cours par étudiant).
3. **Lecture des Informations d'un Étudiant**
   * **Objectif** : S'assurer que le système permet d'accéder aux informations complètes d'un étudiant.
   * **Critères de Validation** :
     + Affichage correct des données personnelles de l'étudiant (nom, prénom, email, etc.).
     + Accès sécurisé aux informations sensibles, telles que la date de naissance et l'adresse.
4. **Modification des Informations d'un Étudiant**
   * **Objectif** : Tester la capacité du système à mettre à jour les détails d'un étudiant.
   * **Critères de Validation** :
     + Modification réussie des informations sans perte de données.
     + Enregistrement de l'historique des modifications effectuées.
     + Validation des restrictions d'accès basées sur les rôles (seuls les administrateurs peuvent effectuer des modifications).
5. **Suppression d'un Étudiant**
   * **Objectif** : Vérifier que le système permet la suppression sécurisée d'un étudiant.
   * **Critères de Validation** :
     + Confirmation de la suppression définitive des données de l'étudiant de la base de données.
     + Respect des protocoles de sécurité pour éviter toute récupération non autorisée des données.
6. **Liste de tous les Étudiants**
   * **Objectif** : Assurer que le système peut lister tous les étudiants inscrits.
   * **Critères de Validation** :
     + Affichage complet et paginé des étudiants enregistrés.
     + Fonctionnalité de recherche pour trouver des étudiants spécifiques selon différents critères (nom, identifiant, etc.).
     + Temps de réponse acceptable même avec une grande quantité de données.

**Tests d'Acceptation Utilisateur (UAT)**

Les tests d'acceptation utilisateur visent à confirmer que le système répond aux besoins des utilisateurs finaux, notamment le directeur et le responsable des études à ENSITECH. Voici quelques critères typiques pour les tests UAT :

1. **Facilité d'utilisation**
   * Les utilisateurs peuvent naviguer facilement dans l'interface utilisateur.
   * Les actions principales sont intuitives et bien guidées.
2. **Performance**
   * Temps de réponse rapide lors de l'accès aux données et de la modification des informations.
   * Système capable de gérer simultanément plusieurs utilisateurs sans dégradation significative des performances.
3. **Sécurité**
   * Les mécanismes d'authentification et d'autorisation fonctionnent correctement.
   * Les données confidentielles sont protégées contre l'accès non autorisé.
4. **Fiabilité**
   * Le système est stable et n'a pas de dysfonctionnements fréquents.
   * Les erreurs sont correctement gérées et rapportées aux utilisateurs de manière claire.
5. **Critères de vérification :**
6. **Création d'Étudiant**
   * **Critères de Vérification** :
     + L'étudiant est créé avec succès dans la base de données.
     + Toutes les informations obligatoires sont saisies correctement.
     + Vérification de la génération d'un identifiant unique pour l'étudiant.
     + Le processus de création respecte les règles de validation définies (par exemple, format d'email valide, longueur des champs correcte).
7. **Association d'un Cours à un Étudiant**
   * **Critères de Vérification** :
     + Sélection correcte du cours parmi ceux disponibles dans le système.
     + Vérification que l'étudiant est lié au cours dans la base de données.
     + Respect des restrictions d'association définies (par exemple, nombre maximum de cours par étudiant).
     + Les modifications sont enregistrées de manière appropriée dans la base de données.
8. **Lecture des Informations d'un Étudiant**
   * **Critères de Vérification** :
     + Affichage correct des données personnelles de l'étudiant.
     + Accès sécurisé aux informations sensibles (comme la date de naissance).
     + Les informations sont présentées de manière claire et dans un format facile à comprendre.
9. **Modification des Informations d'un Étudiant**
   * **Critères de Vérification** :
     + Les modifications apportées aux informations d'un étudiant sont enregistrées correctement.
     + Les restrictions d'accès et les permissions sont respectées (par exemple, seuls les administrateurs peuvent modifier certaines informations).
     + Le système gère les erreurs de validation de manière appropriée et informe l'utilisateur des corrections nécessaires.
10. **Suppression d'un Étudiant**
    * **Critères de Vérification** :
      + Confirmation que l'étudiant est supprimé de manière définitive de la base de données.
      + Les données associées à l'étudiant sont supprimées conformément aux règles de sécurité et de confidentialité.
      + Le processus de suppression est sécurisé et nécessite une confirmation pour éviter les suppressions accidentelles.
11. **Liste de tous les Étudiants**
    * **Critères de Vérification** :
      + Affichage correct et paginé de tous les étudiants enregistrés.
      + Fonctionnalité de recherche fonctionne correctement pour trouver des étudiants spécifiques.
      + Le temps de réponse est raisonnable même avec une grande quantité de données.

**Méthodologie de Vérification**

Pour chaque fonctionnalité, la vérification doit inclure les étapes suivantes :

* **Tests Unitaires** : Vérification des composants individuels pour s'assurer qu'ils fonctionnent comme prévu.
* **Tests d'Intégration** : Validation des interactions entre les différents modules du système.
* **Tests de Système** : Vérification globale du système pour confirmer que toutes les fonctionnalités interagissent correctement.
* **Tests de Performance** : Mesure du temps de réponse et de la capacité de charge pour garantir que le système fonctionne efficacement sous charge.
* **Tests de Sécurité** : Validation des mécanismes d'authentification, d'autorisation et de protection des données.

1. Annexes
2. **Diagrammes supplémentaires** 
   1. **Diagrammes de Séquence**

* **Diagrammes de Séquence d'Inscription d'Étudiant** : Illustrer le flux d'actions entre l'acteur (par exemple, le responsable des études) et le système pour l'inscription d'un nouvel étudiant.
* **Diagrammes de Séquence de Modification d'Informations Étudiant** : Décrire comment les informations d'un étudiant sont modifiées à travers les interactions avec le système.
  1. **Diagrammes d'État**
* **Diagrammes d'État d'Étudiant** : Définir les différents états qu'un étudiant peut traverser, par exemple, "Nouveau", "Inscrit", "En Cours", "Diplômé", etc.
* **Diagrammes d'État de Cours** : Modéliser les états qu'un cours peut avoir, comme "Disponible", "En Cours", "Terminé", etc.
  1. **Diagrammes de Collaboration**
* **Diagrammes de Collaboration entre Étudiant et Cours** : Illustrer la communication entre un étudiant et un cours pour montrer comment les informations sont échangées.
  1. **Diagrammes de Package**
* **Diagrammes de Package pour les Modules du Système** : Structurer les différents composants du système en packages pour montrer leur organisation logique.
  1. **Diagrammes de Déploiement**
* **Diagrammes de Déploiement du Système ENSITECH** : Représenter comment les différents composants du système (comme la base de données, le serveur d'application, etc.) sont déployés physiquement.

1. **Glossaire des termes**

**Termes Techniques**

1. **Système de Gestion de l'École** : Le logiciel sur mesure développé pour gérer les opérations académiques et administratives de l'école ENSITECH.
2. **Base de Données** : Système de stockage structuré des informations relatives aux étudiants, aux cours, aux enseignants, etc.
3. **Interface Utilisateur (UI)** : La partie visible et interactive du logiciel à travers laquelle les utilisateurs interagissent avec le système.
4. **Backend** : La partie du logiciel qui gère le traitement des données, généralement située côté serveur.
5. **Frontend** : La partie du logiciel avec laquelle les utilisateurs interagissent directement, souvent à travers une interface utilisateur graphique.
6. **Authentification et Autorisation** : Les processus par lesquels le système vérifie l'identité des utilisateurs et leur accorde l'accès aux fonctionnalités appropriées.
7. **API (Interface de Programmation Applicative)** : Ensemble de protocoles et de définitions permettant aux logiciels de communiquer entre eux.

**Acronymes**

1. **UML** : Unified Modeling Language, langage de modélisation unifié utilisé pour visualiser, spécifier, concevoir et documenter les artefacts d'un système logiciel.
2. **UI** : User Interface, interface utilisateur.
3. **UX** : User Experience, l'expérience utilisateur globale lors de l'utilisation du logiciel.
4. **DBMS** : Database Management System, système de gestion de base de données.
5. **API** : Application Programming Interface, interface de programmation d'application.
6. **HTML** : HyperText Markup Language, langage de balisage utilisé pour créer des pages web.
7. **CSS** : Cascading Style Sheets, langage utilisé pour définir la présentation des documents HTML.

**Exemples de Définitions**

* **Backend** : La partie du système qui traite les données, logique métier et interactions avec la base de données, sans interagir directement avec l'utilisateur final.
* **API RESTful** : Une API qui suit les principes de l'architecture REST (Representational State Transfer), utilisée pour la communication entre systèmes distribués.
* **Interface Utilisateur (UI)** : L'ensemble des éléments visibles et interactifs avec lesquels l'utilisateur interagit pour utiliser le système.
* **Base de Données Relationnelle** : Un type de base de données structurée basée sur des tables relationnelles, utilisé pour stocker des données conformément à un modèle de données relationnel.