

# Système de Gestion des Cours

**Date de dépôt :** Avant le 7 Janvier 2026

**Sujet :** Développement d'Application Web Avancé

## 1. Objectif du Projet

Conformément au cahier des charges, ce projet vise à faciliter l'interaction transparente entre les étudiants, les instructeurs et l'administration. Le système permet la gestion centralisée des cours, la soumission de projets/devoirs, et facilite la rétroaction (feedback) pédagogique.

L'architecture est fondée sur les **trois modules interconnectés** exigés :

- **Module Administrateur** : Gestion des utilisateurs et des cours.
- **Module Étudiants** : Accès aux cours et soumission des travaux.
- **Module Instructeur** : Évaluation des soumissions et communication.

## 2. Spécifications Techniques

Le projet respecte scrupuleusement les contraintes technologiques imposées :

1. **Spring Boot** : Utilisé comme socle robuste pour le back-end, assurant une configuration rapide et un déploiement efficace via le serveur Tomcat embarqué.
2. **RESTful APIs** : L'architecture est entièrement orientée services (SOA). Chaque fonctionnalité est exposée via des endpoints HTTP REST standards.
3. **Spring Data JPA** : L'accès aux données est simplifié par l'utilisation du pattern Repository et de l'ORM Hibernate (Object-Relational Mapping).
4. **Front-end / Postman** : La validation des endpoints a été réalisée pour garantir la robustesse des échanges JSON.

## 3. Implémentation des Modules

### 3.1 Module Administrateur

Ce module jette les bases du système.

- **Gestion des Comptes** : Création de comptes pour Étudiants et Instructeurs via AuthController et UtilisateurController.
- **Gestion du Programme** : Création et codification des matières (ex: "MATH101") via CourseController.
- **Sécurité** : Attribution des rôles (ADMINISTRATEUR, INSTRUCTEUR, ETUDIANT) pour cloisonner les accès.

### 3.2 Module Étudiants

Conçu pour l'usage quotidien des apprenants.

- **Consultation** : Visualisation des cours inscrits et des matériels pédagogiques.
- **Soumission** : Envoi de fichiers (PDF, etc.) pour les Projets/Devoirs via SubmissionController.
- **Feedback** : Consultation des notes et des commentaires correctifs laissés par les instructeurs.

### 3.3 Module Instructeur

Dédié au suivi pédagogique.

- **Vérification** : Accès à la liste des travaux soumis par projet.
- **Communication** : Envoi de commentaires privés sur une soumission spécifique ou publics sur le mur du cours via CommentController.
- **Évaluation** : Attribution de notes aux travaux des étudiants.

## 4. Base de Données (Conformité Schéma Entité-Relation)

Le modèle de données (entity package) a été construit pour refléter exactement les entités principales demandées :

Entité Demandée	Classe Java Implémentée	Attributs Clés & Relations
Utilisateur	Utilisateur.java	id, nom, prenom, email (Unique), motDePasse (Hashé), role (Enum)

Entité Demandée	Classe Java Implémentée	Attributs Clés & Relations
Cours	Course.java	id, titre, description, code (Unique), instructeur_id
Projet/Devoir	Project.java	id, titre, dateLimite, cours_id
SoumissionÉtudiant	Submission.java	etudiant_id, projet_id, dateSoumission, fichierUrl, note
Commentaire	Comment.java	texte, dateCommentaire, auteur_id, soumission_id / cours_id

### Choix de Conception :

- **Polymorphisme** : L'entité Comment gère à la fois les discussions générales et les feedbacks, optimisant ainsi le schéma base de données.
- **Sécurité** : Le mot de passe est systématiquement haché avant persistance.

## 5. Conclusion

---

Le système délivré répond à l'ensemble des exigences fonctionnelles et techniques. Il offre une plateforme sécurisée, modulaire et évolutive, prête à être déployée pour faciliter la gestion académique.