



الجامعة الأورومتوسطية بفاس
EUROMED UNIVERSITY OF FES
UNIVERSITÉ EUROMED DE FÈS



Projet de Semestre

Université : Université Euromed de Fès (UEMF)

Département : EIDIA - École d'Ingénierie Digitale et d'Intelligence Artificielle

Module : Programmation Web Sécurisée

Enseignant : AMAMOU Ahmed

Membres : AITAOUICHA Yassine

DAKHIL Bakr

BOUAYAD Othman

HANNOUN Haytam

EL MOURABIT Hamza

Date : 18 Janvier 2025

1. Introduction et Contexte

L'évolution de l'éducation numérique a rendu indispensable l'utilisation d'outils centralisés pour la gestion des cours, devoirs et ressources pédagogiques. Les étudiants et enseignants perdent beaucoup de temps à naviguer entre différents supports et plateformes. Notre projet vise à créer une plateforme inspirée de Google Classroom, offrant un **environnement unique pour l'apprentissage et la gestion pédagogique.**

Cette plateforme permettra notamment de gérer les cours, de publier des annonces et ressources, de créer et corriger des devoirs, et de suivre la progression des étudiants, le tout dans un cadre sécurisé et performant.

2. Objectifs du Projet

L'objectif principal est de **centraliser et simplifier toutes les interactions éducatives** entre enseignants, étudiants et administrateurs. Les objectifs détaillés incluent :

- Création et gestion complète des cours avec personnalisation visuelle.
 - Publication et organisation des ressources pédagogiques.
 - Création, soumission et correction des devoirs avec feedback détaillé.
 - Suivi des performances et génération de statistiques.
 - Notifications automatiques et gestion des alertes.
-

3. Acteurs du Système

3.1 Professeur

Le professeur est l'acteur central. Il crée des cours, publie des ressources et annonces, crée et corrige des devoirs, et suit les résultats des étudiants.

Fonctionnalités principales :

- Création et personnalisation des cours (titre, description, niveau, couleur, bannière).
- Génération automatique du code d'invitation.
- Gestion des étudiants inscrits et possibilité de retrait.
- Publication d'annonces avec pièces jointes, vidéos et liens externes.
- Gestion des devoirs : création, assignation, notation et feedback.
- Consultation de statistiques et export de notes.

Workflow :

1. Créer un cours → définir paramètres → générer code d'invitation.
2. Publier ressources et annonces → programmer ou publier immédiatement.

3. Créer un devoir → définir instructions et barème → assigner aux étudiants.
 4. Corriger devoirs → publier notes et feedback → envoyer notifications.
-

3.2 Étudiant

L'étudiant rejoint les cours via code d'invitation, consulte les ressources, soumet les devoirs et reçoit des notifications et feedbacks.

Fonctionnalités principales :

- Rejoindre et quitter un cours.
 - Accéder au fil d'actualité et aux dossiers de ressources.
 - Soumettre des devoirs via upload sécurisé (max 10 fichiers).
 - Consulter notes, moyennes et feedbacks.
 - Recevoir notifications pour deadlines et nouvelles publications.
-

3.3 Administrateur

L'administrateur supervise la plateforme, gère les utilisateurs et configure les paramètres globaux.

Responsabilités :

- Création et import massifs d'utilisateurs via CSV.
 - Supervision des cours et archivage forcé si nécessaire.
 - Gestion des statistiques et génération de rapports.
 - Configuration des paramètres généraux, notifications et limites de fichiers.
-

4. Modules Fonctionnels

4.1 Module Cours

Le module cours permet la création et la personnalisation complète des cours. Chaque cours peut être dupliqué ou archivé. Les paramètres incluent autorisation de commentaires, soumissions tardives et notifications automatiques.

Résumé des fonctionnalités :

Fonctionnalité	Description
Création de cours	Titre, description, niveau, couleur, bannière
Code d'invitation	Généré automatiquement
Gestion des membres	Ajouter/retirer étudiants
Archivage	Cours consultable mais non modifiable
Duplication	Créer un nouveau cours basé sur un existant

4.2 Module Publications et Ressources

Les enseignants peuvent publier annonces et ressources pédagogiques, organiser par dossiers, programmer des publications et ajouter des commentaires.

Résumé :

- Upload multi-fichiers (PDF, images, vidéos, liens).
 - Organisation en dossiers thématiques.
 - Publication programmée et notifications automatiques.
 - Commentaires activables/désactivables par annonce.
-

4.3 Module Devoirs

Les devoirs permettent aux enseignants de créer, assigner et corriger les travaux des étudiants.

Workflow détaillé :

1. Création du devoir : titre, instructions, barème.
2. Assignation : tous les étudiants ou sélection spécifique.
3. Soumission : validation des fichiers et stockage sécurisé.
4. Correction : notation selon rubriques JSON, feedback personnalisé.
5. Publication des résultats : notifications automatiques aux étudiants.

Exemple JSON Rubrique Devoir :

```
[  
  {"critere":"Présentation", "points":5},  
  {"critere":"Contenu", "points":10},  
  {"critere":"Respect des délais", "points":5}  
]
```

4.4 Module Notes et Suivi

Toutes les notes sont centralisées et consultables par chaque étudiant et professeur. Les administrateurs peuvent exporter des statistiques globales.

Fonctionnalités principales :

- Calcul automatique des moyennes.
 - Tableau récapitulatif par devoir et étudiant.
 - Export Excel/CSV.
 - Notifications automatiques pour nouvelles notes ou retards.
-

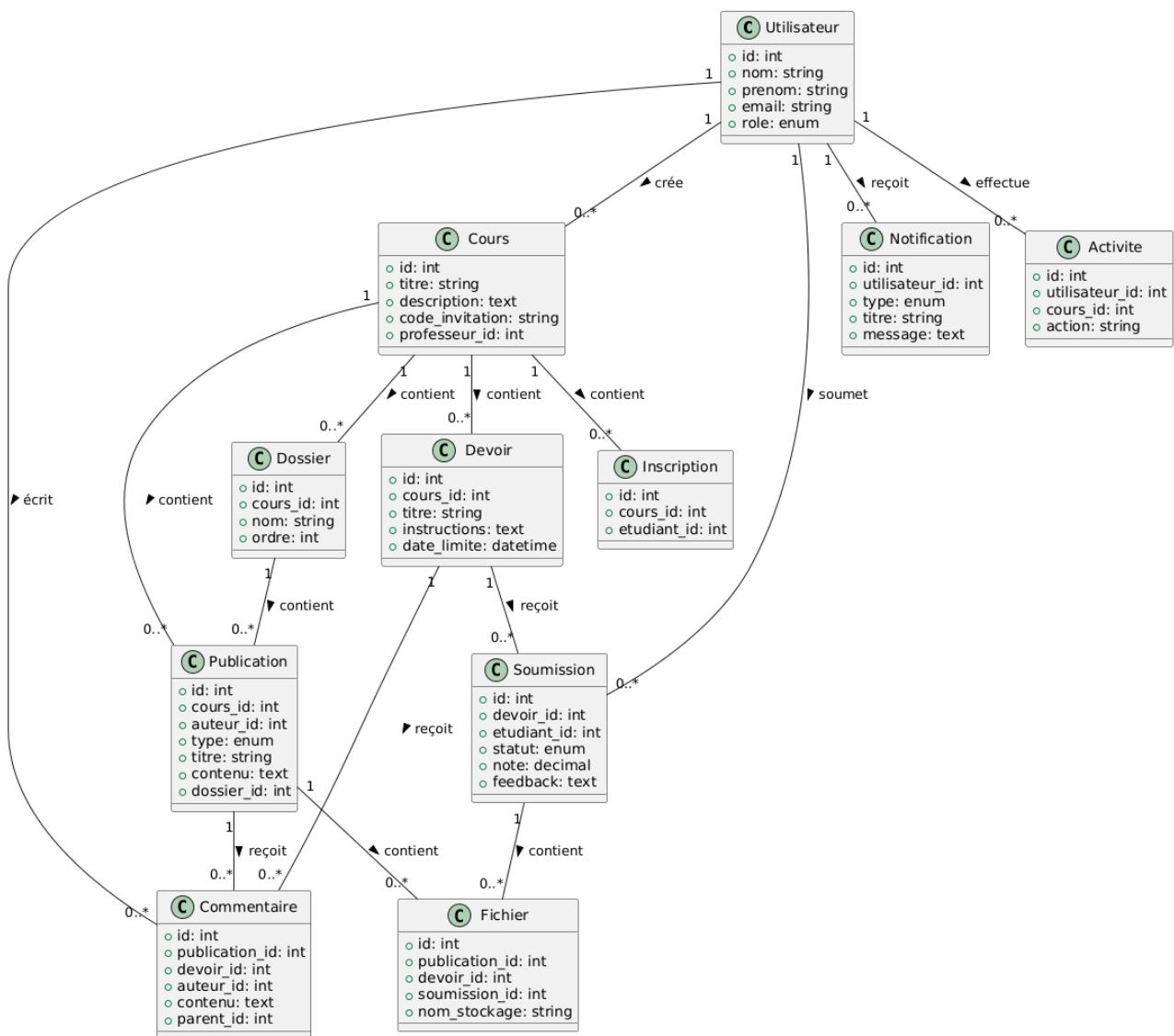
5. Sécurité et Performance

La plateforme intègre plusieurs niveaux de sécurité :

- Upload sécurisé : validation MIME, limite de taille, renommage avec hash unique, stockage hors webroot.
- Contrôle d'accès : seuls les étudiants inscrits voient les cours correspondants.
- Protection XSS : contenu HTML filtré via bibliothèques sécurisées et Blade.
- Confidentialité des notes : visibles uniquement par le professeur et l'étudiant concerné.
- Rate limiting : prévention spam sur commentaires et connexions.
- Backup quotidien avec rétention 30 jours.

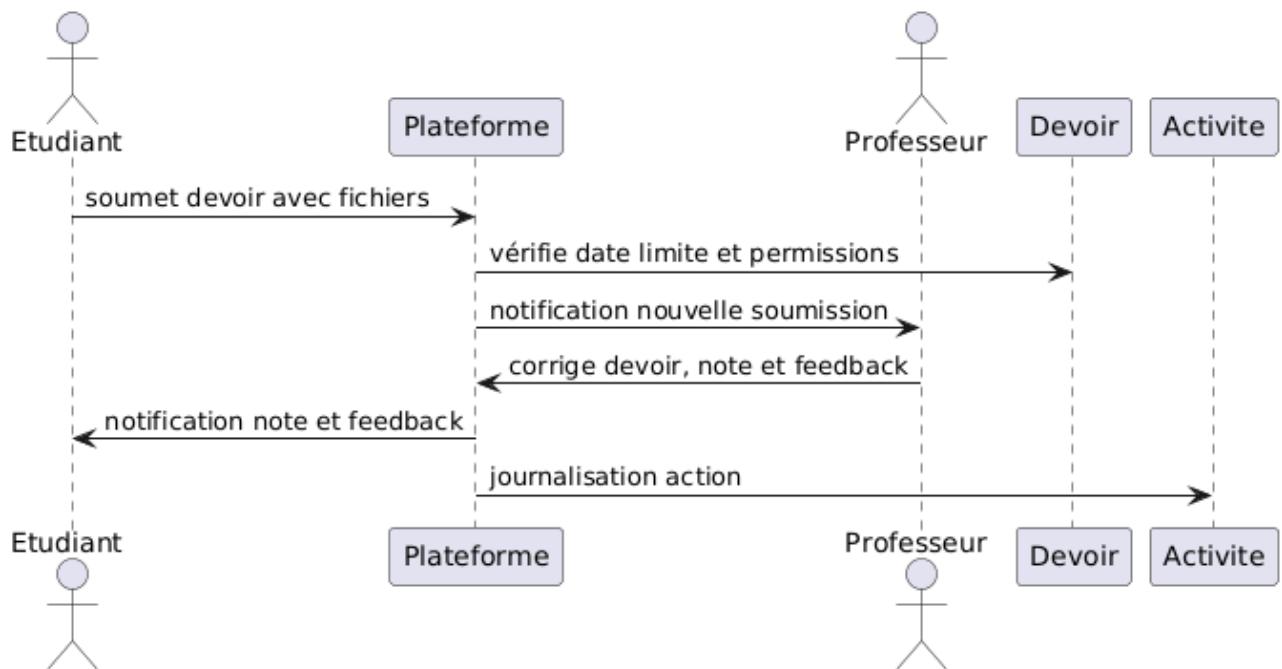
6. Diagrammes UML

1. Diagramme de Classes



- **Utilisateur** : représente les trois rôles (admin, professeur, étudiant). Les relations montrent qui crée, soumet ou corrige les devoirs, et qui reçoit les notifications.
- **Cours** : chaque cours est créé par un professeur et peut contenir plusieurs devoirs, publications et dossiers de ressources.
- **Inscription** : relie les étudiants aux cours.
- **Publication et Fichier** : illustrent la gestion des ressources et annonces, avec possibilité d'attacher plusieurs fichiers.
- **Devoir et Soumission** : permettent la création des devoirs et la réception des travaux soumis par les étudiants.
- **Commentaire et Notification** : permettent la communication et l'alerte des utilisateurs.
- **Activité** : journalisation des actions de chaque utilisateur pour la traçabilité.

2. Diagramme de Séquence – Workflow Soumission de Devoir



L'étudiant soumet son devoir via l'interface de la plateforme.

1. La plateforme vérifie la date limite et les permissions.
2. Le professeur est notifié de la nouvelle soumission.
3. Le professeur corrige, note et ajoute un feedback personnalisé.
4. La plateforme envoie la note et le feedback à l'étudiant.
5. Chaque action est journalisée dans la table **Activité** pour assurer traçabilité et audit.

5. Annexes Techniques

Devoir.php:

```
1      <?php
2
3      namespace App\Models;
4
5      use Illuminate\Database\Eloquent\Factories\HasFactory;
6      use Illuminate\Database\Eloquent\Model;
7
8  <?php
9      class Devoir extends Model
10     {
11         use HasFactory;
12
13         protected $table = 'devoirs';
14         protected $fillable = [
15             'cours_id','titre','instructions','date_limite','points_max','rubrique','autoriser_retard',
16             'penalite_retard','limite_fichiers','formats_Autorises','assigne_a','publie','date_publication'
17         ];
18
19         protected $casts = [
20             'rubrique' => 'array',
21             'formats_Autorises' => 'array',
22             'assigne_a' => 'array',
23             'autoriser_retard' => 'boolean',
24             'publie' => 'boolean',
25             'date_limite' => 'datetime',
26             'date_publication' => 'datetime',
27         ];
28
29         public function cours()
30         {
31             return $this->belongsTo(Cours::class);
32         }
33
34         public function soumissions()
35         {
36             return $this->hasMany(Soumission::class);
37         }
38
39         public function fichiers()
40         {
41             return $this->hasMany(Fichier::class);
42         }
43
44         public function commentaires()
45         {
46             return $this->hasMany(Commentaire::class);
47         }
48     }
```

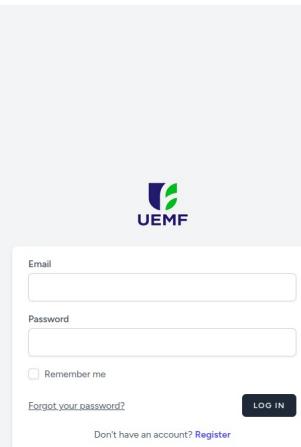
User.php:

```
1      <?php
2
3      namespace App\Models;
4
5      use Illuminate\Foundation\Auth\User as Authenticatable;
6      use Illuminate\Database\Eloquent\Factories\HasFactory;
7      use Illuminate\Notifications\Notifiable;
8      use Laravel\Sanctum\HasApiTokens;
9
10     class User extends Authenticatable
11     {
12         use HasApiTokens, HasFactory, Notifiable;
13
14         protected $table = 'utilisateurs';
15
16         protected $fillable = [
17             'nom',
18             'prenom',
19             'email',
20             'mot_de_passe',
21             'role',
22             'avatar_path',
23             'bio',
24             'preferences_notif',
25             'derniere_connexion',
26             'actif',
27         ];
28
29         protected $hidden = [
30             'mot_de_passe',
31             'remember_token',
32         ];
33
34         protected $casts = [
35             'preferences_notif' => 'array',
36             'derniere_connexion' => 'datetime',
37             'actif' => 'boolean',
38         ];
39
40         public function getAuthPassword()
41         {
42             return $this->mot_de_passe;
43         }
44     }
```

Migration de la table Utilisateur:

```
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration {
8      public function up(): void {
9          Schema::create('utilisateurs', function (Blueprint $table) {
10              $table->id();
11              $table->string('nom', 100);
12              $table->string('prenom', 100);
13              $table->string('email', 255)->unique();
14              $table->string('mot_de_passe');
15              $table->enum('role', ['admin', 'professeur', 'etudiant']);
16              $table->string('avatar_path')->nullable();
17              $table->text('bio')->nullable();
18              $table->json('preferences_notif')->nullable();
19              $table->dateTime('derniere_connexion')->nullable();
20              $table->boolean('actif')->default(true);
21              $table->timestamps(); // created_at et updated_at
22          });
23      }
24
25      public function down(): void {
26          Schema::dropIfExists('utilisateurs');
27      }
28  };
```

Page du Login:



Conclusion:

Ceprojet illustre une conception complète et réfléchie, couvrant l'ensemble des besoins pédagogiques et administratifs. Les diagrammes UML et les scénarios d'utilisation démontrent une organisation claire des modules, des flux de données et des interactions entre utilisateurs.

La structure proposée garantit une **modularité et une évolutivité**, permettant d'intégrer facilement les fonctionnalités futures, comme la gestion des devoirs, la publication de ressources ou le suivi des notes. Même si le développement reste en cours, la conception technique établie offre une base solide et sécurisée pour la mise en œuvre de la plateforme complète.