Rapport de Projet – LivresGourmands.net

Technologie Web – Laravel et MySQL

Étudiant : LABLE Komlanvi Yoan

Date : 18 avril 2025

---

1. Introduction

Le projet **LivresGourmands.net** est une application web développée dans le cadre d'un cours en technologies web. Il s'agit d'une plateforme permettant la gestion de livres de cuisine, avec plusieurs rôles utilisateurs distincts. Chaque rôle a accès à des fonctionnalités spécifiques selon ses responsabilités.

Le projet repose sur le Framework **Laravel** (PHP) pour le backend, **MySQL** comme système de gestion de base de données, et **Bootstrap** pour le développement de l'interface utilisateur.

---

2. Objectifs du projet

L'objectif était de mettre en place une application web selon le modèle **MVC (Modèle-Vue-Contrôleur)**, intégrant une gestion complète des utilisateurs, des rôles, de la sécurité et des différents modules de données. L'application devait permettre :

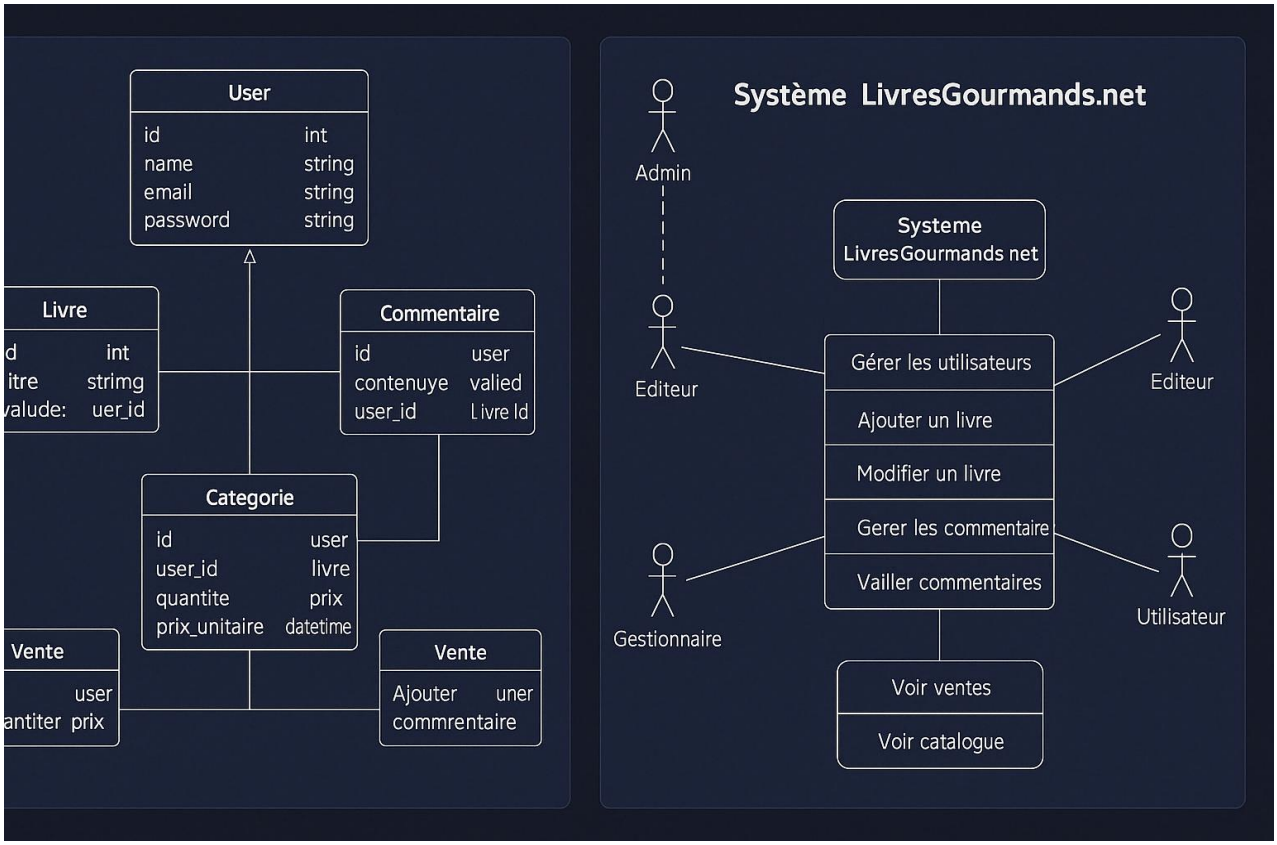
- La gestion des livres par des éditeurs
- La gestion des utilisateurs par un administrateur
- Le suivi des ventes par un gestionnaire
- La modération des commentaires par les éditeurs
- L'affichage et la classification des livres par catégorie

---

3. Technologies utilisées

- **Laravel 12**: framework PHP pour architecture backend (MVC)
- **PHP 8.3** : langage principal du serveur

- **MySQL** : base de données relationnelle
- **Bootstrap 5** : Framework CSS pour la mise en page
- **Vite** : gestionnaire d'assets (CSS, JS)
- **Plan UML** : remplacé par une modélisation **manuelle** des diagrammes UML au format image insérée dans le rapport



---

4. Description des fonctionnalités

4.1 Rôle : Administrateur

- Ajout d'utilisateurs avec choix du rôle (admin, éditeur, gestionnaire)
- Liste des utilisateurs enregistrés
- Suppression possible des utilisateurs
- Redirection automatique selon le rôle

4.2 Rôle : Éditeur

- Ajout d'un livre (titre, auteur, description, prix, niveau d'expertise, catégorie, image)
- Liste des livres ajoutés avec possibilité de suppression
- Gestion dynamique des catégories (ajout et suppression)
- Possibilité de commenter un livre
- Interface de validation ou de rejet des commentaires

4.3 Rôle : Gestionnaire

- Visualisation des ventes avec détails utilisateur et livre
- Gestion des stocks (modification du stock de chaque livre)

---

5. Modèle de données UML

5.1 Diagramme de classes

Le diagramme de classes a été **construit manuellement** et intégré au rapport en tant qu'image. Il modélise les entités principales suivantes :

- **User**: id, name, email, password, role_id
- **Role** : id, nom
- **Livre** : id, titre, description, auteur, niveau_expertise, category_id, prix, stock, image
- **Categorie** : id, nom
- **Commentaire** : id, contenu, valide, user_id, livre_id
- **Vente** : id, user_id, livre_id, quantite, prix_unitaire, date_vente

Les relations entre entités sont représentées par des associations classiques :

- Un User appartient à un Role
- Un Livre appartient à une Categorie
- Un Commentaire est lié à un User et à un Livre
- Une Vente est liée à un User (le client) et un Livre (vendu)

5.2 Diagramme de cas d'utilisation

Ce diagramme a également été **dessiné manuellement** pour illustrer les interactions clés avec le système :

Acteurs :

- Administrateur
- Éditeur
- Gestionnaire
- Utilisateur connecté

Principales interactions :

- L'administrateur gère les utilisateurs et les rôles
- L'éditeur gère les livres, catégories, et commentaires
- Le gestionnaire consulte les ventes et modifie les stocks
- L'utilisateur commente les livres

---

6. Interface utilisateur

L'interface a été conçue en HTML/Blade pur avec Bootstrap, sans thème préconçu. Chaque rôle dispose d'un affichage distinct :

- **Administrateur** : formulaire de création de comptes avec choix du rôle
- **Éditeur** : interface simple d'ajout de livres, gestion de commentaires, catégories
- **Gestionnaire** : tableau des ventes et formulaire de mise à jour des stocks

La navigation est conditionnée selon le rôle grâce à des directives @if(Auth::user()->role_id === ...).

---

7. Sécurité et gestion des rôles

L'application utilise le système **Laravel Breeze** pour l'authentification.
Chaque utilisateur possède un rôle (role_id) lié à une table roles.
Des **middlewares** sont utilisés pour restreindre l'accès aux routes selon le rôle.

Chaque contrôleur est responsable de la logique liée à son rôle et redirige l'utilisateur après connexion vers la page appropriée. Les routes sont également protégées par des groupes avec middleware(['auth']).

---

8. Conclusion

Ce projet a permis de mettre en œuvre toutes les compétences acquises en programmation web côté serveur :

- Utilisation du pattern MVC
- Gestion d'une base de données relationnelle
- Création d'une interface dynamique
- Validation de formulaires
- Envoi de fichiers et sécurité d'accès via les rôles

Le travail effectué démontre une bonne structure d'un projet professionnel.