

# Rapport du projet – LivreGourmand

## 1. Introduction

Ce rapport présente la conception et la mise en œuvre de l'API REST développée dans le cadre du cours Programmation Web avancée (420-WA6-AG). L'objectif de cette étape était de partir du diagramme de classes et de produire un schéma relationnel cohérent ainsi qu'une API complète en Node.js et Express, connectée à une base de données MySQL. Le système gère les utilisateurs, les produits, les paniers, les commandes, et les avis, avec authentification sécurisée par JWT et bcrypt.

## 2. Choix du modèle et normalisation

Le modèle de données a été conçu selon les principes de la normalisation (3NF) afin d'éviter la redondance et d'assurer la cohérence des données. Les principales entités sont : users, categories, ouvrages, panier, commandes, listes\_cadeaux et avis. Chaque table possède une clé primaire unique, et les relations entre entités sont établies via des clés étrangères. Les types ENUM sont utilisés pour les rôles utilisateurs et les statuts de commande.

## 3. Règles métier et logique API

Les règles métier sont implémentées directement dans l'API et au niveau de la base de données. Par exemple :

- Seuls les ouvrages avec stock > 0 sont affichés au public.
- Lorsqu'une commande est validée, une transaction décrémente le stock des ouvrages.
- Un client ne peut laisser un avis que pour un ouvrage qu'il a réellement acheté.
- Les commentaires soumis doivent être validés par un éditeur avant publication.

L'authentification utilise JWT et les mots de passe sont hachés avec bcrypt. Les routes sensibles sont protégées par un middleware vérifiant le rôle de l'utilisateur.

## 4. Limitations et améliorations possibles

Certaines fonctionnalités comme la gestion automatique des paniers expirés et les paiements simulés pourraient être améliorées. Une future version pourrait inclure un service de messagerie (ex. envoi d'e-mails de confirmation). La modularité du code facilite déjà ces extensions.

## 5. Conclusion

Le projet respecte les exigences techniques et les bonnes pratiques de développement. La séparation claire entre routes, contrôleurs et services assure une architecture maintenable et évolutive. Ce travail constitue une base solide pour le développement complet d'une application e-commerce sécurisée.