**Rapport de Projet – Librairie Numérique**

**Date :** 21 octobre 2025

**Objet :** Synthèse technique et fonctionnelle du projet de plateforme de vente de livres en ligne.

**1. Synthèse du Projet**

Ce document décrit l'architecture et les fonctionnalités du projet de librairie en ligne. La plateforme est une application web développée avec **Node.js et Express**, conçue pour la vente de **livres numériques (e-books)**. Elle intègre un système d'authentification sécurisé, la gestion d'un catalogue, un processus de commande, ainsi que des fonctionnalités sociales avancées comme les listes de cadeaux et les avis clients.

La solution repose sur une base de données relationnelle **MySQL** qui garantit la cohérence et l'intégrité des données. L'ensemble du projet respecte les bonnes pratiques de sécurité, notamment la protection des endpoints sensibles et la validation systématique des entrées.

**2. Architecture Technique et Sécurité**

La pile technique a été choisie pour sa robustesse, sa performance et sa scalabilité.

* **Backend (Node.js / Express) :** La nature asynchrone de Node.js est idéale pour une application e-commerce, gérant efficacement un grand nombre de requêtes simultanées (navigation, recherche, etc.). Express fournit un cadre de travail minimaliste et puissant pour la création de l'API REST.
* **Base de Données (MySQL) :** Le choix d'un SGBD relationnel assure des transactions ACID (Atomicité, Cohérence, Isolation, Durabilité), ce qui est indispensable pour un système de commandes et de gestion des droits d'accès aux e-books.
* **Authentification et Autorisation :**
  + **JWT (JSON Web Tokens) :** Utilisés pour gérer les sessions utilisateur de manière *stateless*, ce qui simplifie l'architecture et facilite la mise à l'échelle horizontale de l'application.
  + **Hashing de mot de passe (bcrypt) :** Les mots de passe sont hachés avec un algorithme robuste, conformément aux standards de sécurité actuels.
  + **Contrôle d'accès basé sur les Rôles (RBAC) :** Le système définit plusieurs rôles (CLIENT, ADMIN, EDITOR, MANAGER). Des *middlewares* dédiés protègent les routes de l'API, garantissant que seuls les utilisateurs autorisés peuvent accéder aux fonctionnalités sensibles (ex: validation de commentaires, gestion du catalogue).

**3. Justification des choix du modèle de données**

Le modèle de données a été conçu en suivant les principes du modèle relationnel pour garantir la cohérence, la performance et la maintenabilité.

* **Normalisation :** La structure est conforme à la **Troisième Forme Normale (3NF)** pour minimiser la redondance et prévenir les anomalies de données. Chaque information (comme le titre d'un livre ou le nom d'un utilisateur) est stockée en un seul endroit et référencée via des identifiants uniques. Cela assure que les mises à jour sont simples et cohérentes à travers toute l'application.
* **Relations et Intégrité Référentielle :** L'intégrité des données est assurée par l'usage de **clés étrangères (FOREIGN KEY)**. Les stratégies de suppression (ON DELETE) sont définies pour refléter les liens logiques : CASCADE pour les données dépendantes (un article de commande n'existe pas sans sa commande) et SET NULL pour les relations non-essentielles (un livre n'est pas supprimé si sa catégorie l'est), prévenant ainsi la perte accidentelle de données.
* **Choix d’indexes :** Pour garantir la rapidité des requêtes, des indexes ont été placés stratégiquement. En plus des **clés primaires** indexées par défaut, des **indexes uniques** sont utilisés sur des colonnes critiques comme users(email) et books(isbn). Ces indexes accélèrent considérablement les opérations courantes comme l'authentification et la recherche, tout en renforçant l'unicité des données.

**4. Modélisation des Données et Fonctionnalités Clés**

La structure de la base de données est le reflet direct des fonctionnalités offertes par la plateforme.

* **Gestion du Catalogue Numérique :** La table books est au cœur du système. La présence du champ fileUrl confirme la nature numérique des produits. Le champ stock, bien que traditionnellement utilisé pour des produits physiques, offre la flexibilité de gérer des ventes de licences limitées pour certains e-books.
* **Processus d'Achat et Bibliothèque Personnelle :**
  1. Un utilisateur passe une commande (orders et order\_items).
  2. Une fois le paiement validé (status = 'PAID'), un enregistrement est créé dans la table user\_ebooks.
  3. Cette table user\_ebooks agit comme la **bibliothèque numérique permanente de l'utilisateur**, lui donnant un accès perpétuel au contenu acheté via le fileUrl. Ce mécanisme est central pour la gestion des droits numériques.
* **Fonctionnalités d'Engagement et Sociales :**
  1. **Avis et Commentaires :** Les utilisateurs peuvent noter (reviews) et commenter (comments) les livres. Le champ validated dans la table comments indique un système de modération géré par les EDITOR ou ADMIN.
  2. **Listes de Cadeaux (gift\_lists) :** Une fonctionnalité clé permettant aux utilisateurs de créer et partager des listes de souhaits via un code unique (share\_code).
  3. **Offrir un E-book (gift\_redemptions) :** Le système permet d'offrir un livre numérique à une personne non-inscrite via son email. Un token unique est généré, permettant au destinataire de réclamer son cadeau et de l'ajouter à sa propre bibliothèque.

**5. Implémentation des Règles Métier**

Les logiques de gestion critiques sont appliquées au niveau de l'API pour assurer la cohérence des opérations.

* **Droit de Publication d'Avis :** La règle "seul un client ayant acheté le livre peut laisser un avis" est implémentée par l'API. Avant de sauvegarder un nouvel avis, le système vérifie que l'utilisateur possède bien le livre en consultant la table user\_ebooks.
* **Décrémentation du Stock :** Lors de la validation d'une commande pour un livre à licences limitées, l'opération de décrémentation du stock est effectuée à l'intérieur d'une **transaction SQL**. Cela garantit que le stock n'est mis à jour que si la commande est validée avec succès, prévenant ainsi les incohérences de données.

**6. Pistes d'Évolution et Recommandations**

Pour assurer la pérennité et la performance du projet, les améliorations suivantes sont recommandées :

* **Performance :** Ajouter des **indexes** sur les clés étrangères fréquemment utilisées (ex: orders.user\_id, reviews.book\_id). Cette optimisation simple aura un impact majeur sur la rapidité des requêtes à mesure que la base de données grandira.
* **Intégrité des Données :** Mettre en place une stratégie de **"soft delete"** (suppression logique) pour les livres. Plutôt que de supprimer un enregistrement de la base, un champ is\_active serait basculé à false. Cela préserverait l'historique des commandes et des avis liés à des livres qui ne sont plus en vente.
* **Expérience Utilisateur :** Implémenter un **panier d'achat persistant** en base de données (tables panier et panier\_items comme suggéré dans les spécifications). Cela permettrait à un utilisateur de retrouver son panier en se connectant depuis un autre appareil.