P_Web294

[1]

Site de Vennes - Lausanne

2nd semestre

Denis Romain, Herrera Gonzalo, Moreira Thomas

Chef de projet Charmier G.

[1] Image réalisé avec ChatGPT

Introduction

Pour ce projet du module P_WEB294, on a développé le frontend d'une application web qui permet de partager sa passion pour la lecture. Ce travail fait suite à un précédent projet où on avait déjà créé la base de données et mis en place le backend. Ici, l'objectif était de construire une interface utilisateur simple, agréable et fonctionnelle à l'aide de Vue.js.

Le site permet de voir une liste d'ouvrages, d'en ajouter, modifier ou supprimer, de laisser des commentaires et des notes. Certaines actions sont réservées aux utilisateurs connectés, et d'autres uniquement aux admins. L'idée, c'était de rendre l'application intuitive, rapide et bien structurée. On a aussi pris en compte quelques principes d'écoconception pour limiter l'impact du site.

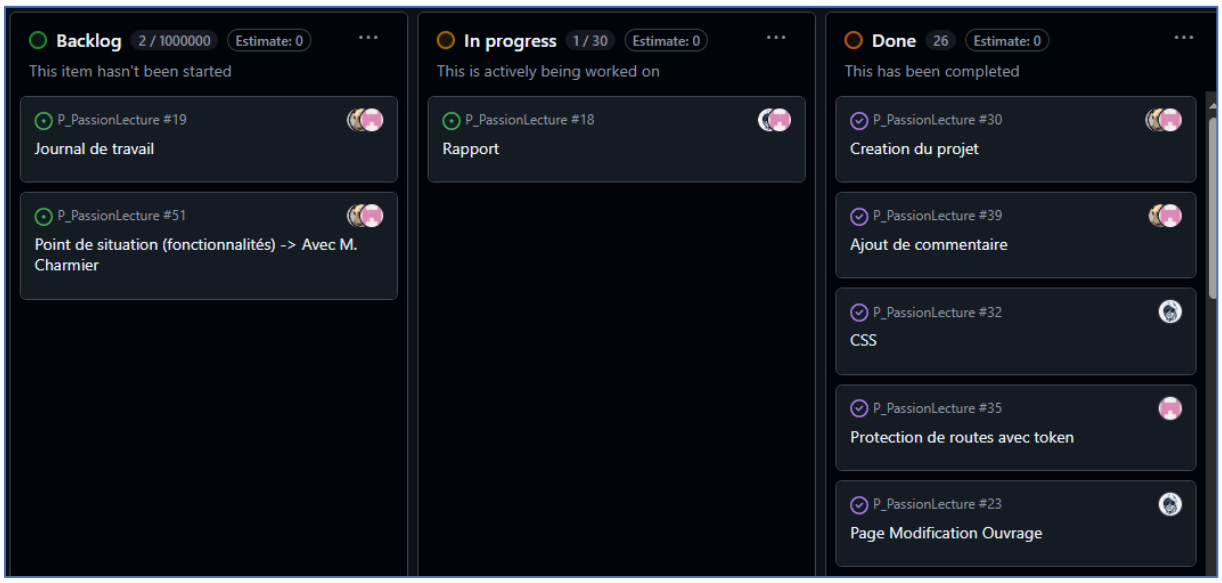
Dans cette partie du rapport, je vais expliquer comment j'ai organisé le code, mis en place les différentes pages et fonctionnalités demandées, et comment le frontend communique avec le backend déjà existant.

Analyse

Planification - Gestion de groupe

Tout au long du projet, nous avons utilisé GitHub pour organiser notre travail en équipe. Nous avons mis en place un tableau Kanban avec trois colonnes : Backlog (pour les tâches à faire), En cours (pour celles en train d'être développées) et Terminée (pour signaler qu'une tâche était finie). Cette méthode nous a permis de mieux nous répartir le travail, de suivre la progression de chacun et de rester efficaces.

Comme cette organisation a bien fonctionné, nous avons choisi de continuer à l'utiliser pour la suite du projet 294. Cela nous a aidés à garder une bonne vue d'ensemble sur l'avancement du projet, à éviter les doublons et à mieux collaborer.



Maquettes

Avant de commencer le développement, on a créé des maquettes pour avoir une idée claire de ce à quoi allait ressembler notre site. Elles nous ont servi de guide tout au long du projet.

Voici le lien Figma vers les différentes pages

Modélisation de la DB grâce au MCD et au MLD

Voici notre MCD (une copie du fichier looping est dans le répertoire du projet). Il a les associations suivantes :

1. Un livre ne peut avoir qu'une seule catégorie
2. Une catégorie peut avoir plusieurs livres
3. Un livre ne peut avoir qu'un seul éditeur
4. Un éditeur peut avoir plusieurs livres
5. Un livre ne peut avoir qu'un seul auteur
6. Un auteur peut avoir plusieurs livres
7. Un livre appartient et est créé par un utilisateur
8. Un livre peut avoir plusieurs commentaires de plusieurs utilisateurs différents
9. Un livre peut avoir plusieurs notes de plusieurs utilisateurs différents

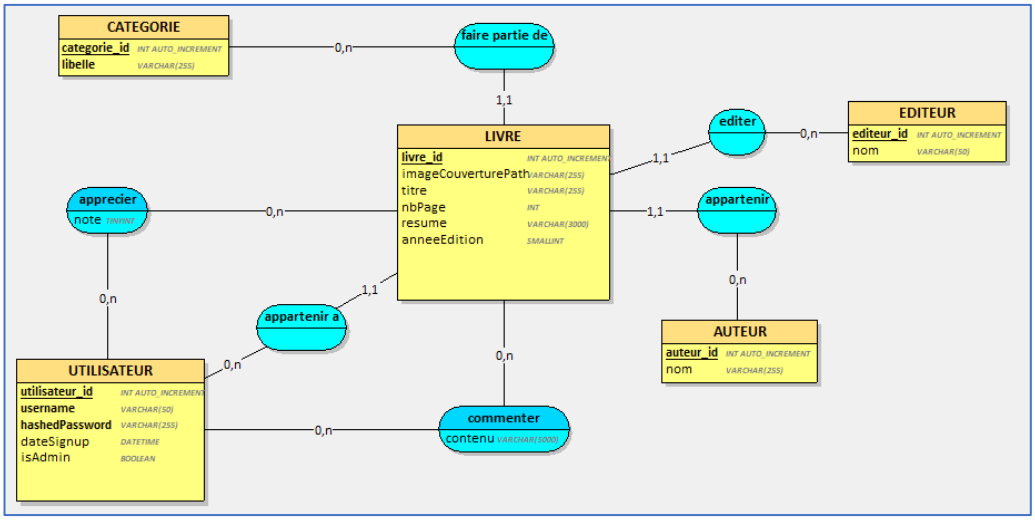


Table	Champ	Type	Description
t_livre	livre_id	Clef primaire	L'id d'un livre
t_livre	imageCouverturePath	Chaine de caractère	Le chemin de l'image de couverture
t_livre	titre	Chaine de caractère	Le titre d'un livre
t_livre	nbPage	Entier	Le nombre de page dans un livre
t_livre	resume	Chaine de caractère	Le résume du livre
t_livre	utilisateur_fk	Clef étrangère	Clef étrangère pour l'id de l'utilisateur du livre
t_livre	editeur_fk	Clef étrangère	Clef étrangère pour l'id de l'éditeur du livre
t_livre	categorie_fk	Clef étrangère	Clef étrangère pour l'id de la catégorie du livre

t_livre	auteur_fk	Clef étrangère	Clef étrangère pour l'id de l'auteur du livre
t_livre	anneeEdition	Entier	L'année d'édition
t_categorie	categorie_id	Clef primaire	L'id d'une catégorie
t_categorie	libelle	Chaine de caractère	Le nom de la catégorie
t_editeur	editeur_id	Clef primaire	L'id d'un éditeur
t_editeur	nom	Chaine de caractère	Le nom de l'éditeur
t_utilisateur	Utilisateur_id	Clef primaire	L'id d'un utilisateur
t_utilisateur	username	Chaine de caractère	Le nom entré par l'utilisateur
t_utilisateur	hashedPassword	Chaine de caractère	Le password entré par l'utilisateur, après avoir été haché
t_utilisateur	dateSignup	Date	La date l'utilisateur a créé son compte
t_utilisateur	isAdmin	Booléen	Si l'utilisateur est admin ou pas
T_apprecier	note	Entier	La note donnée par l'utilisateur a un livre
T_apprecier	Livre_fk	Clef étrangère	Cette table est avec un ID composé : ça veut dire que l'id est la réunion des fks de t_livre et t_utilisateur
T_apprecier	Utilisateur_fk	Clef étrangère	
T_laisser	contenu	Chaine de caractère	Le commentaire donné par l'utilisateur a un livre
T_ laisser	Livre_fk	Clef étrangère	Cette table est avec un ID composé : ça veut dire que l'id est la réunion des fks de t_livre et t_utilisateur
T_ laisser	Utilisateur_fk	Clef étrangère	

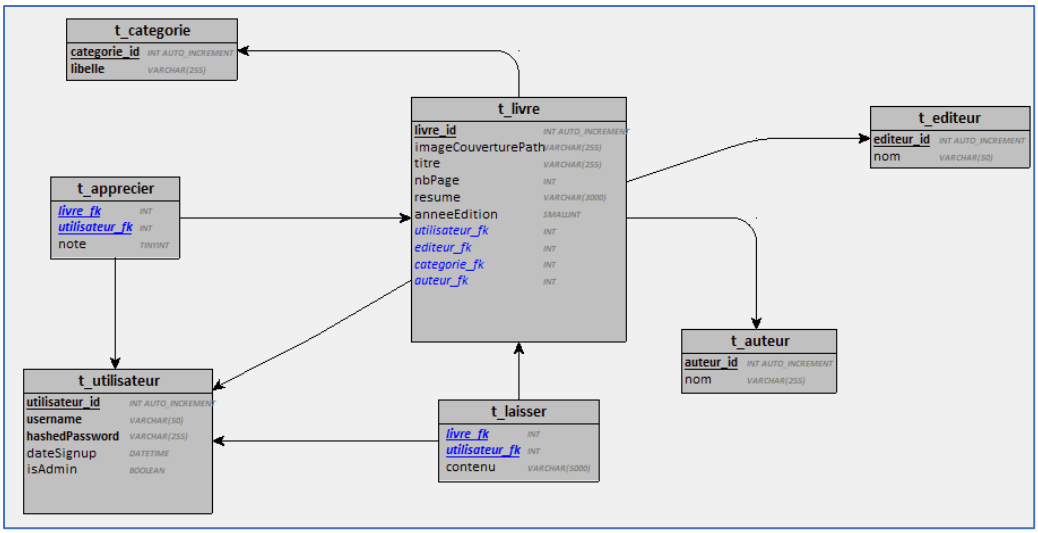
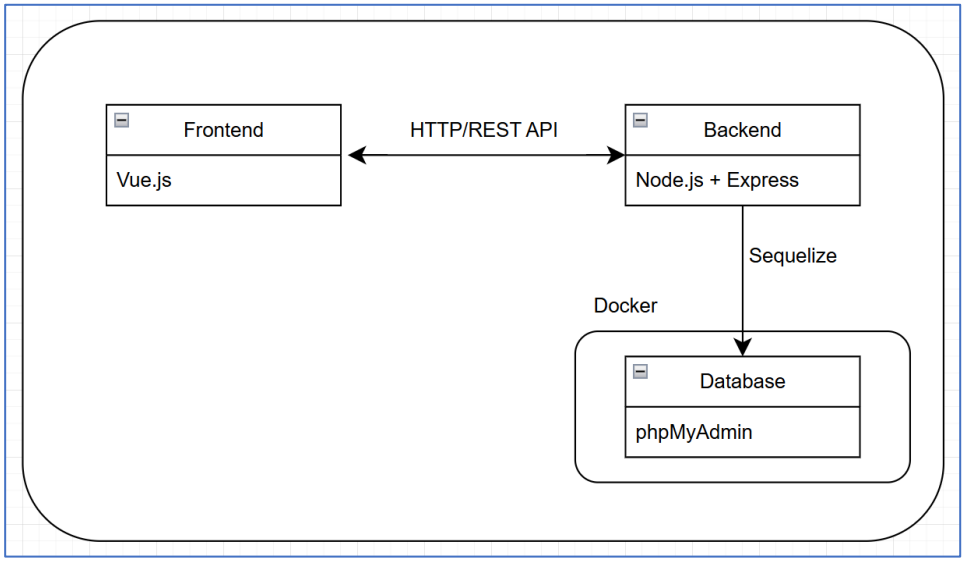


Schéma de l'architecture

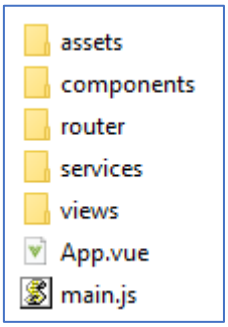


Analyse de la structure du code

Notre code a été développé avec la structure ci-dessous :

Pour organiser notre projet frontend, on a séparé le code en plusieurs dossiers, chacun avec un rôle bien précis :

- **assets/ :** contient les fichiers statiques comme le logo du site et le fichier de style CSS global.
- **components/ :** regroupe tous les composants réutilisables du site (comme les cartes de livres, le header, le footer, etc.). Cela permet d'avoir un code plus propre et modulaire.
- **router/ :** contient la configuration des routes avec Vue Router, ce qui permet de gérer la navigation entre les pages de l'application.
- **services/ :** sert à gérer la communication entre le frontend et le backend (via des appels API).



- **views/ :** contient les différentes pages du site (accueil, liste des livres, page de profil, etc.). Chaque vue représente une section principale du site.

Ecoconception Web

Fond noir et couleurs sombres : sur les écrans OLED et AMOLED, les pixels noirs sont éteints, ce qui permet de réduire la consommation énergétique et de prolonger la durée de vie des appareils.

Design épuré et minimaliste : la réduction des éléments graphiques superflus diminue la charge de calcul, entraînant une moindre utilisation des ressources système et une optimisation de la consommation d'énergie.

Moins de sollicitations, davantage de sobriété numérique : cette approche encourage une utilisation plus consciente et respectueuse de l'environnement.

Connexion du Backend au Frontend

Nous avons utilisé le fichier /services/api.js pour connecter le frontend et le backend

Ce code est le lien entre les 2. Il fait que toutes les requêtes sont fait avec l'url du backend.

Le paramètre *withCredentials sert a* transferrer les tokens et cookies

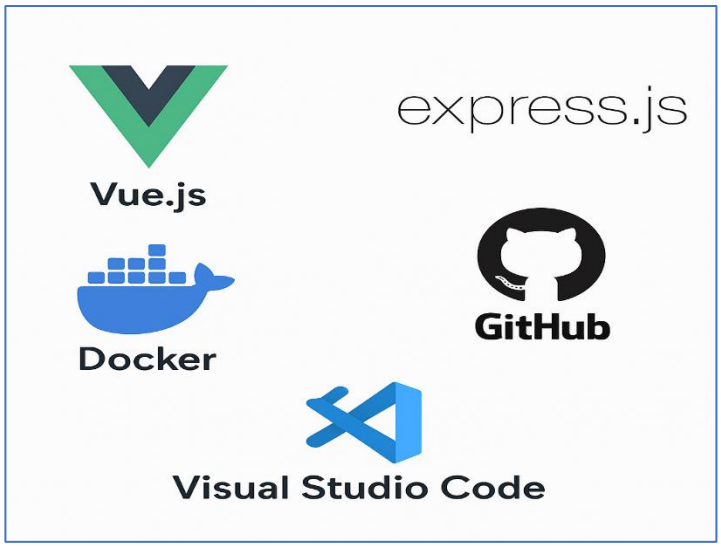
```
export const api = axios.create({
  baseURL: 'http://localhost:3000',
  withCredentials: true,
})
```

Technologie utilisée

Pour ce projet, on a utilisé plusieurs technologies qu'on avait déjà vues en cours :

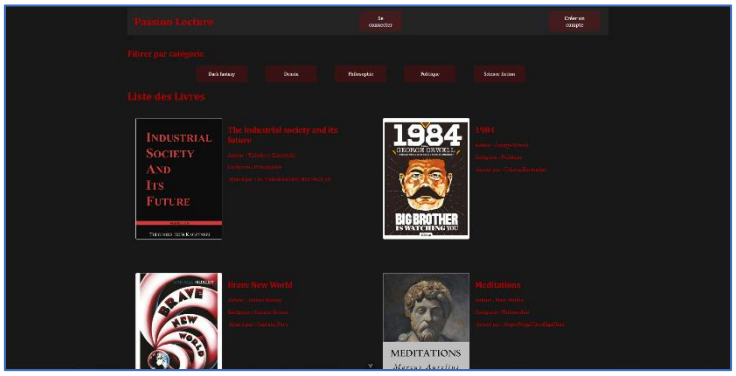
- **Vue.js** : pour développer le frontend de l'application, avec des composants dynamiques et réutilisables.
- **Express.js** : pour le backend, sous forme d'API REST. On l'avait déjà réalisé dans le module 295, donc on s'est basé dessus pour ce projet.
- **Docker** : pour faire tourner le frontend, le backend et la base de données dans des conteneurs, ce qui facilite les tests et l'organisation de l'environnement de travail.
- **GitHub** : pour suivre l'avancement du projet en équipe, versionner le code, et organiser nos tâches avec un tableau Kanban.
- **Visual Studio Code (VS Code)** : notre éditeur principal, utilisé pour écrire et organiser tout le code du projet.

Ces outils nous ont permis de créer une application complète, structurée et plus proche d'un environnement de développement professionnel.



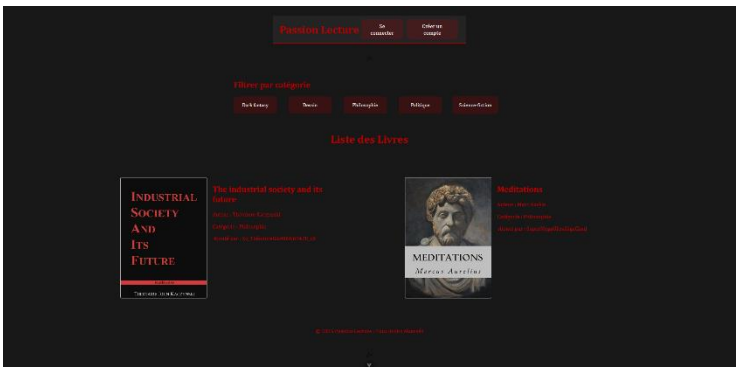
Fonctionnalité technique demandée

Une page d'accueil comprenant une explication de l'utilité du site ainsi que les cinq derniers ouvrages ajoutés (accès tout public).

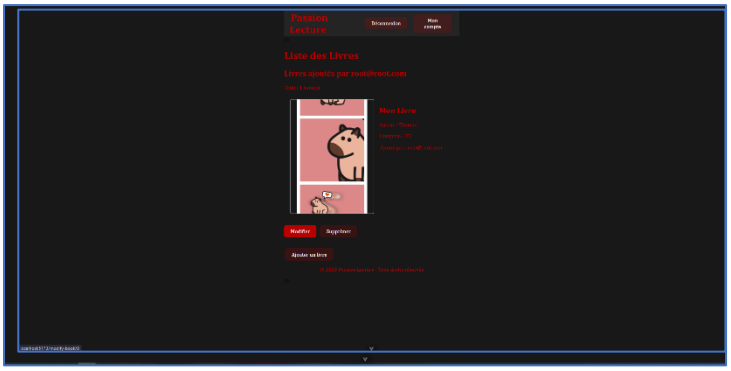


Une page comprenant la liste des ouvrages par catégorie (accès tout public avec restrictions sur les liens).

- Cliquer sur 'Philosophie'

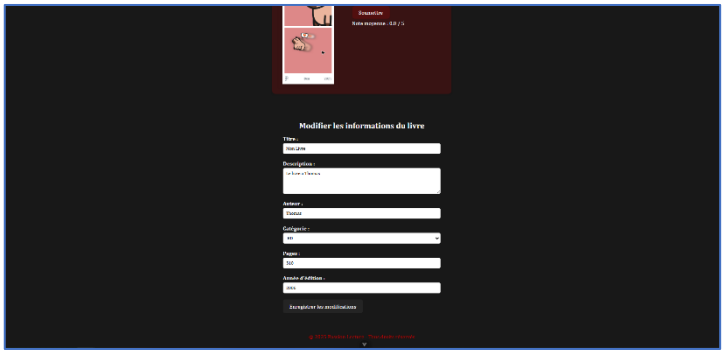


Une page d'ajout d'un ouvrage (accès utilisateur).

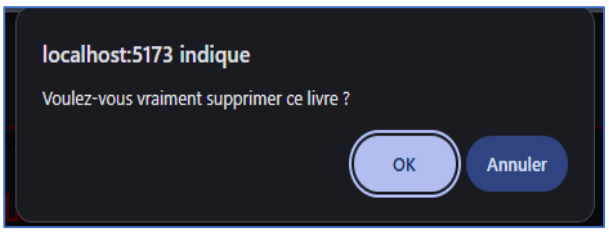


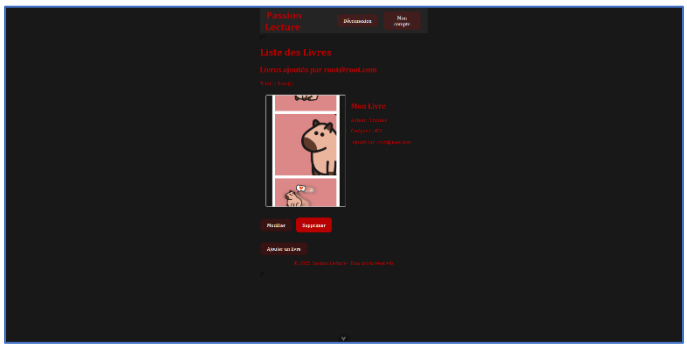
Une page de modification d'un ouvrage (accès utilisateur pour ses ouvrages)

-Il faut scroller en bas

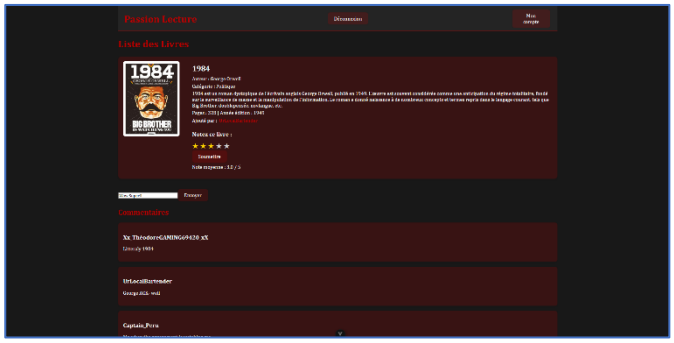


Une page de suppression d'un ouvrage (accès utilisateur pour ses ouvrages)

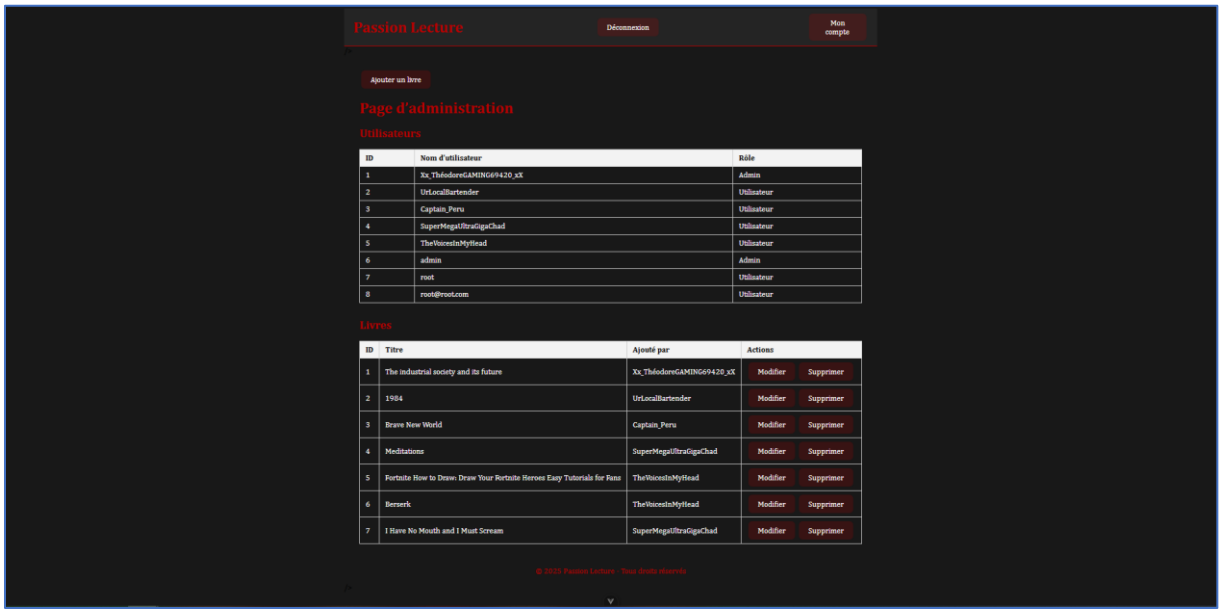




Une page (vue détail d'un livre) **permettant d'ajouter une appréciation et un commentaire à un ouvrage** (accès utilisateur).



L'utilisateur admin peut réaliser toutes les actions.



Explications des vues

AccueilView.vue
→ Vue d'accueil ou page d'entrée de l'application, celle que l'utilisateur voit en arrivant.

AddBook.vue
→ Formulaire ou interface permettant d'ajouter un nouveau livre à la base de données.

AdminView.vue
→ Interface d'administration, accessible uniquement par les administrateurs, pour gérer les utilisateurs, livres ou autres contenus.

BookView.vue
→ Vue détaillée d'un livre spécifique avec ses informations (titre, auteur, note, etc.).

CategoryView.vue
→ Affiche les livres filtrés par catégorie, comme "Romans", "Science-fiction", etc.

LoginView.vue
→ Page de connexion pour que les utilisateurs puissent se connecter à leur compte.

ModifyView.vue
→ Permet de modifier les informations d'un livre existant (réservé aux admins ou utilisateurs autorisés).

RegisterView.vue
→ Formulaire pour que les nouveaux utilisateurs puissent créer un compte.

UserView.vue
→ Profil de l'utilisateur connecté, avec ses informations personnelles et peut-être ses livres ajoutés ou ses commentaires.

Comment démarrer le projet

Pour démarrer le projet, il faut premièrement cloner le code depuis GitHub. Il faut aller dans une invite de commande et exécuter la commande suivante

Git clone https://github.com/romaindenis1/P_PassionLecture

Ensuite, il faut lancer le contenaire docker (si docker desktop est installé)

Dans l'invite de commande il faut naviguer au bon endroit avec

Cd P_PassionLecture

Cd Docker_MySQL

Docker compose up –d

Après ceci, il faut ouvrir une invite de commande a *src/Frontend* et *src/Backend* et exécuter *npm i* pour installer les dépendances du projet. Après, il faut simplement exécuter *npm start* sur les 2 invites. Après, en allant sur un navigateur web et sur localhost:5173 le site web sera là.

Protection des routes

Pour sécuriser notre application, on a mis en place une protection des routes aussi bien côté backend que côté frontend.

Côté backend

On utilise un système d'authentification basé sur JWT (JSON Web Tokens). Lorsqu'un utilisateur se connecte, un token est généré et envoyé au client. Ce token doit ensuite être envoyé dans chaque requête (dans les en-têtes) pour accéder aux routes protégées.

On a créé un middleware **auth** qui vérifie si le token est présent, valide, et s'il correspond bien à l'utilisateur. Si ce n'est pas le cas, la requête est bloquée et une erreur est renvoyée. Ce système empêche les utilisateurs non autorisés d'accéder à certaines fonctionnalités comme la modification ou la suppression de livres.

Côté frontend

On utilise Vue Router pour bloquer l'accès à certaines pages si l'utilisateur n'est pas connecté ou s'il n'a pas les droits nécessaires. Pour ça, on a ajouté des "**meta**" dans les routes (comme **requiresAuth** ou **isAdmin**), puis on utilise un **beforeEach** pour vérifier les conditions avant chaque navigation. Par exemple :

- Si un utilisateur essaie d'accéder à une page sans être connecté, il est redirigé vers /login.
- Si un utilisateur veut accéder à un espace admin sans les droits, il est renvoyé vers la page d'accueil.

Grâce à cette double vérification (frontend + backend), on s'assure que les données sensibles sont protégées et que seuls les utilisateurs autorisés peuvent accéder aux fonctionnalités importantes.

Rôles et utilisateurs

Utilisateur standard

- Peut ajouter de nouveaux livres.
- Peut modifier uniquement ses propres livres.
- Peut ajouter des commentaires sur les livres.
- Peut noter les livres.

Administrateur

- Peut modifier et supprimer n'importe quel livre, sans restriction.
- Dispose de tous les droits des utilisateurs standards.

Stratégie de tests

Pour le frontend, nous avons principalement utilisé des console.log() afin de suivre l'exécution du code et localiser les erreurs. Cette méthode simple et efficace nous a permis de vérifier l'état des variables et le déroulement de certaines fonctions pendant le développement.

Conclusion

Conclusion générale

Ce projet *Passion lecture* nous a vraiment permis de mettre en pratique et de consolider ce qu'on avait vu dans les modules précédents (295). En connectant le backend qu'on avait déjà réalisé avec le frontend qu'on a développé pour ce projet, on a pu mieux comprendre comment fonctionne une appli web de bout en bout, de la base de données jusqu'à l'interface utilisateur.

Le fait de séparer le projet en deux parties, backend et frontend, nous a aidés à mieux saisir les interactions entre les différentes couches d'une appli et à être plus organisés dans notre travail. On en a aussi profité pour prendre en main des outils comme Vue.js et Vue Router, tout en continuant à utiliser GitHub pour le versionnage et le travail en équipe.

Enfin, la dynamique de groupe a été très positive. Chacun s'est bien investi et la répartition des tâches a permis de garder un bon rythme. Cette expérience nous donne envie de continuer sur cette lancée pour les prochains projets, en utilisant ce qu'on a appris et en cherchant à améliorer encore nos méthodes de travail et de collaboration.

Conclusion Romain Denis

Ce projet était étonnamment différent de la partie backend. La structure globale des fichiers a rendu la coordination extrêmement difficile et l'absence de conflits, tant sur GitHub que dans la manière d'effectuer les tâches. Notre mauvaise planification et notre manque d'utilisation des branches Git ont entraîné à plusieurs reprises la perte de modifications et la nécessité de les refaire. Cependant, nous avons toujours surmonté ces difficultés et collaboré efficacement pour créer un projet globalement bien structuré, et je suis très satisfait du résultat.

Conclusion Gonzalo Herrera

Ce projet m'a permis de renforcer mes compétences en développement frontend avec Vue.js et de mieux comprendre l'intégration avec le backend. J'ai aussi appris à mieux organiser mon code, à collaborer efficacement en équipe et à utiliser des outils comme Docker et GitHub dans un vrai contexte de projet.

Conclusion Thomas Moreira

Ce projet m'a permis d'apprendre Vue.js et de renforcer mes connaissances en développement backend. Grâce à cette expérience, j'ai pu mieux comprendre l'intégration entre le frontend et le backend, et acquérir de nouvelles compétences que je pourrai réutiliser dans mes futurs projets.

Critique sur la planification

La gestion des tâches était clairement à revoir. Bien que nous ayons utilisé GitHub Project pour catégoriser nos tâches, la mise à jour était partielle et, parfois, pas organisée. Au lieu de systématiquement tenir à jour les tâches du GitHub Project, nous discutions justement des tâches entre nous, créant alors un suivi oral sans beaucoup de forme.

Un de nos plus gros obstacles était le méga planning fait au lancement. En effet, celles-ci étaient très grandes et larges pour vraiment viser un objectif précis. Nous aurions dû les découper en petites tâches précises afin de les suivre au fur et à mesure et de facilité le travail de chacun.

De plus, étant seulement trois dans l'équipe, l'utilisation de GitHub Project semblait un peu bizarre, car cet outil donne l'impression d'être fait pour des plus grandes équipes.

Pour le prochain projet, il faudrait donc nous discipliner à catégoriser les tâches de manière plus précise et lisible, où chacun se doit de compléter une tâche, afin de bien suivre le projet tout au long de sa création. En effet, cela facilite, indéniablement, la répartition du travail et le suivi des tâches.

Utilisation de l'IA

Nous avons utilisé l'IA uniquement pour s'informer, comprendre nos erreurs & bugs, en aucun cas pour effectuer le travail à notre place.

Annexe

Après une observation attentive et objective du travail de chacun, il apparaît clairement que certaines personnes ont contribué davantage à ce projet que d'autres. D'ailleurs, il est bien connu que dans chaque groupe, il y a toujours un ou

deux membres qui profitent du travail collectif pour obtenir la même note sans fournir d'efforts équivalents. Il serait intéressant qu'à l'avenir, une répartition officielle des responsabilités soit imposée pour éviter que les éléments les moins impliqués ne tirent encore le groupe vers le bas, comme cela a pu être constaté ici.

Webographie

romaindenis1/P_PassionLecture

Backlog · P_PassionLecture

Figma

ChatGPT