

# P\_294 Front-end

---



Samuel Sallaku, Romain Schertenleib, Bastien Segalen – CID2B  
Section Informatique - Vennes

40p  
G. Charmier

## Table des matières

<b>1</b>	<b>INTRODUCTION .....</b>	<b>3</b>
1.1	TITRE.....	3
1.2	DESCRIPTION.....	3
1.3	MATÉRIEL ET LOGICIELS À DISPOSITION .....	3
1.4	PRÉREQUIS .....	3
1.5	LES POINTS SUIVANTS SERONT ÉVALUÉS .....	3
<b>2</b>	<b>ANALYSE.....</b>	<b>3</b>
2.1	PLANIFICATION DES TÂCHES .....	3
2.2	MCD/MLD .....	3
2.2.1	<i>Explication de la structure</i> .....	5
2.2.2	<i>Expliquer les cardinalités</i> .....	5
2.3	STRUCTURE DU CODE.....	6
2.4	SCHÉMA.....	8
<b>3</b>	<b>RÉALISATION .....</b>	<b>8</b>
3.1	SÉCURITÉ.....	8
3.2	EXPLICATION DES FONCTIONNALITÉS TECHNIQUES & LEUR IMPLÉMENTATION.....	9
3.3	ECO-CONCEPTION .....	10
<b>4</b>	<b>TESTS .....</b>	<b>10</b>
4.1	TESTS RÉALISÉS .....	10
<b>5</b>	<b>CONCLUSION.....</b>	<b>11</b>
5.1	ORGANISATION DU CODE (GITHUB) .....	11
5.2	CONCLUSION GÉNÉRALE .....	11
5.3	CONCLUSION PERSONNELLE.....	12
5.4	CRITIQUE CONSTRUCTIVE.....	12
<b>6</b>	<b>WEBOGRAPHIE .....</b>	<b>13</b>
6.1	WEBOGRAPHIE.....	13

# 1 INTRODUCTION

## 1.1 Titre

Réaliser le front-end d'un site de livres

## 1.2 Description

Projet en lien avec le module C294, donc réaliser le front-end d'une application web permettant de faire le CRUD des livres, une interface où on peut voir un livre et mettre une appréciation ainsi qu'un commentaire, puis la possibilité de visualiser le profil d'un utilisateur.

## 1.3 Matériel et logiciels à disposition

- 1x PC ETML
- GitHub
- VS Code
- MS Teams
- Internet

## 1.4 Prérequis

- Connaissances de base en C294
- Connaissances de base en C295
- Connaissances de base en I293

## 1.5 Les points suivants seront évalués

- Le rapport
- Le journal de travail
- Le code

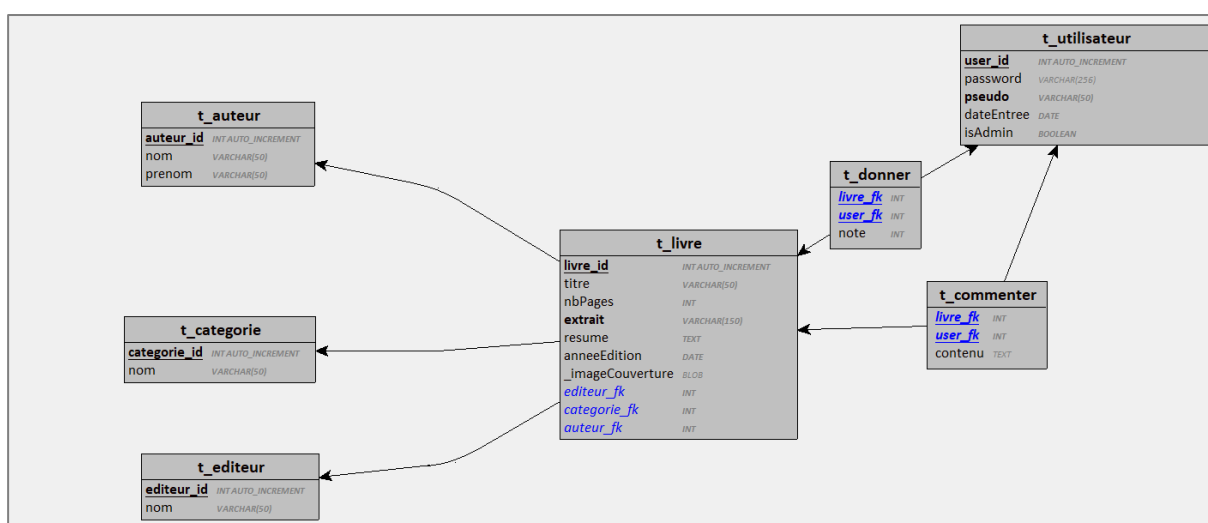
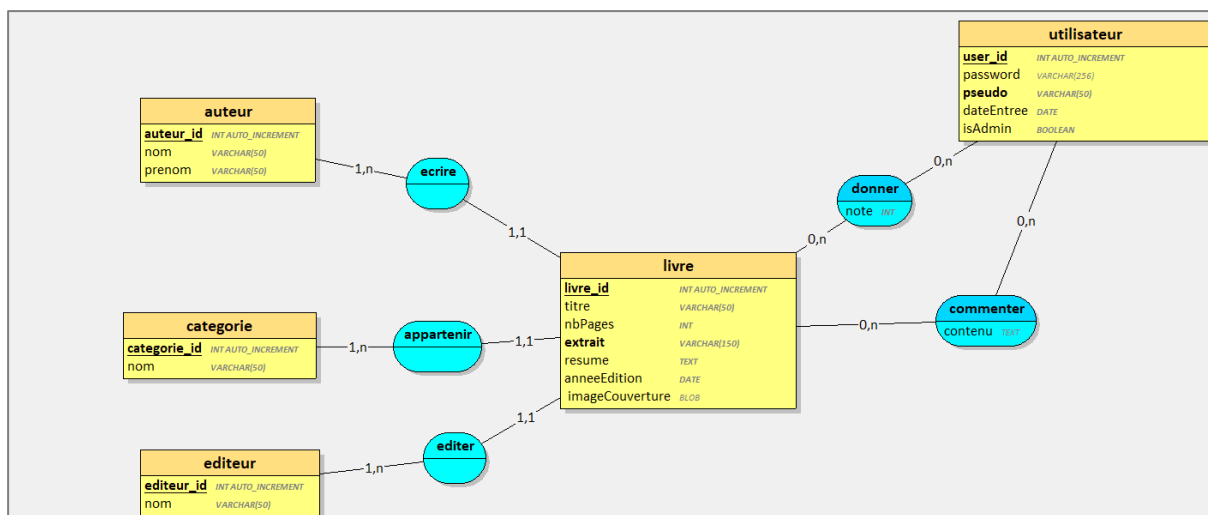
# 2 ANALYSE

## 2.1 Planification des tâches

Consultez notre GitHub Projects

Lien : <https://github.com/users/po77qxd/projects/1/views/1>

## 2.2 MCD/MLD

**T\_auteur :****Auteur\_id** : ID de l'auteur**Nom** : nom de l'auteur**Prénom** : prénom de l'auteur**T\_categorie :****Categorie\_id** : ID de la catégorie**Nom** : nom de la catégorie**T\_editeur :****Editeur\_id** : ID de l'auteur**Nom** : nom de l'éditeur**T\_livre :****Livre\_id** : ID du livre**Titre** : titre du livre**NbPages** : nombre de pages du livre**Extrait** : un extrait du livre

**Résumé** : un petit résumé du livre

**AnnéeEdition** : l'année d'édition

**\_imageCouverture** : Une image de couverture du livre

**Editeur\_fk** : clé étrangère de l'éditeur

**Catégorie\_fk** : clé étrangère de la catégorie

**Auteur\_fk** : clé étrangère de l'auteur

#### T utilisateur :

**User\_id** : Id de l'utilisateur

**Password** : mot de passe de l'utilisateur

**Pseudo** : Nom d'utilisateur

**DateEntree** : date de création de l'utilisateur

**IsAdmin** : rôle pour les utilisateurs pour savoir s'ils sont admin ou pas

#### T donner :

**Livre\_fk** : ID et clé étrangère venant de la table t\_livre

**User\_fk** : ID et clé étrangère venant de la table t\_user

**Note** : note qui sera donnée au livre (appréciation)

#### T commenter :

**Livre\_fk** : ID et clé étrangère venant de la table t\_livre

**User\_fk** : ID et clé étrangère venant de la table t\_user

**Contenu** : contenu du commentaire qui sera fait par rapport à un livre

## 2.2.1 Explication de la structure

La tables t\_livre se trouve au centre de la page car il s'agit de la table principale. Le centre de l'application

Pourquoi avoir sorti les tables :

- T\_auteur
- T\_categorie
- T\_editeur

Ces tables sont trop importantes pour juste être dans la table T\_livre c'est pourquoi elles sont dehors.

## 2.2.2 Expliquer les cardinalités

**Ecrire :**

1, N / un auteur écrit minimum 1 livre et au maximum un nombre de livre infini

1,1 / un livre est écrit par 1 et unique auteur

**Appartenir :**

1, N / une catégorie a minimum 1 livre et au maximum un nombre de livre infini

1,1 / un livre a une seule catégorie

**Éditer :**

1, N / un éditeur édite minimum 1 livre et au maximum un nombre de livre infini

1,1 / un livre est édité par 1 et unique éditeur

**Donner :**

0, n / un utilisateur donne entre 0 et un nombre infini de note

0, n / un livre a entre 0 et un nombre infini de note

**Commenter :**

0, n / un utilisateur donne entre 0 et un nombre infini de commentaire

0, n / un livre a entre 0 et un nombre indéfini de commentaire

## 2.3 Structure du code

La structure de base a été ajoutée par le framework Vue.js lors de la création du projet. C'est cela qui a généré l'organisation actuelle du projet.

La partie du code que nous avons modifiée se trouve dans le dossier src et est divisée en plusieurs parties :

- 1) Assets  
Dans assets se trouve le code de base des pages Vue. Nous n'avons pas touché à ce code-là, car la structure de base des pages nous convenait déjà très bien. Elle remplissait les besoins de notre application sans devoir y faire de modifications.
- 2) Components  
Notre application n'utilise pas de composants pour le moment, donc il n'y a rien d'utilisé ni de présent dans cette partie-là.  
Cependant, si dans le futur notre application a besoin de composants réutilisables, c'est ici que nous les implémenterons. Ce dossier est prévu pour stocker les petits blocs d'interface qu'on pourrait intégrer dans plusieurs pages.
- 3) Router  
Dans ce fichier se trouve la création de toutes les routes du frontend avec leur path, leur nom, et le component lié à la vue, qui sont dans notre cas des fichiers du dossier Views.

Liste des routes :

Nom	Path	Component	Description
Home	/	HomeView	Page d'accueil du site avec les derniers livre ajouter
addbook	/addBook	addBook	Page pour ajouter les livres
Bibliothèque	/ Bibliothèque	BibliothèqueView	Page de Bibliothèque avec tous les livres déjà ajouter trier par catégorie
bookDetail	/bookDetail/ :id	bookDetail.vue	Page de Détail du livre avec toutes les informations disponibles et si on est connecté la

			possibilité d'ajouter une note et un commentaire
Connexion	/connexion	ConnectionView.vue	Page de connexion à son compte
Inscription	/inscription	RegisterView.vue	Page pour crée un compte sur le site
Profile	/profile	UserView.vue	Page d'information avec toutes les informations d'user
userBibliotheque	/ userBibliotheque	userBibliotheque.vue	Bibliothèque de l'utilisateur avec tous les livres qu'il a ajouté et la possibilité de le modifier et de la supprimer
modifybook	/modifyBook/ :id	ModifyBooks.vue	Page pour modifier le livre
viewUser	/viewUser/ :id	viewUser.vue	Page pour que l'utilisateur connecter puis voir ces informations perso
logout	/logout	Logout.vue	Page avec rien juste pour le bon fonctionnement du logout

#### 4) Services

Dans services se trouve la connexion avec le backend. Cette connexion a été faite avec Axios, et on importe des méthodes contenant les routes du backend. Grâce à cela, on peut appeler ces méthodes dans le code pour interagir avec le backend.

Nous avons divisé cette connexion en 4 fichiers, chacun ayant un thème spécifique et étant utilisé dans des pages bien précises.

Par exemple :

Le fichier AddBookService.js contient toutes les connexions avec le backend utiles pour ajouter un utilisateur ou un livre.

#### 5) Views

Dans Views se trouve le code des pages utilisées dans le site web.

Ces pages sont divisées en 3 parties :

- 1) **Script** : utilisé pour appeler les méthodes liées aux routes du backend
- 2) **Template** : le HTML dynamique utilisé pour afficher les éléments de la page, y compris ceux importés depuis le backend
- 3) **Style scoped** : le CSS lié uniquement à cette page, ce qui permet de ne pas affecter les autres composants

#### 6) App.vue

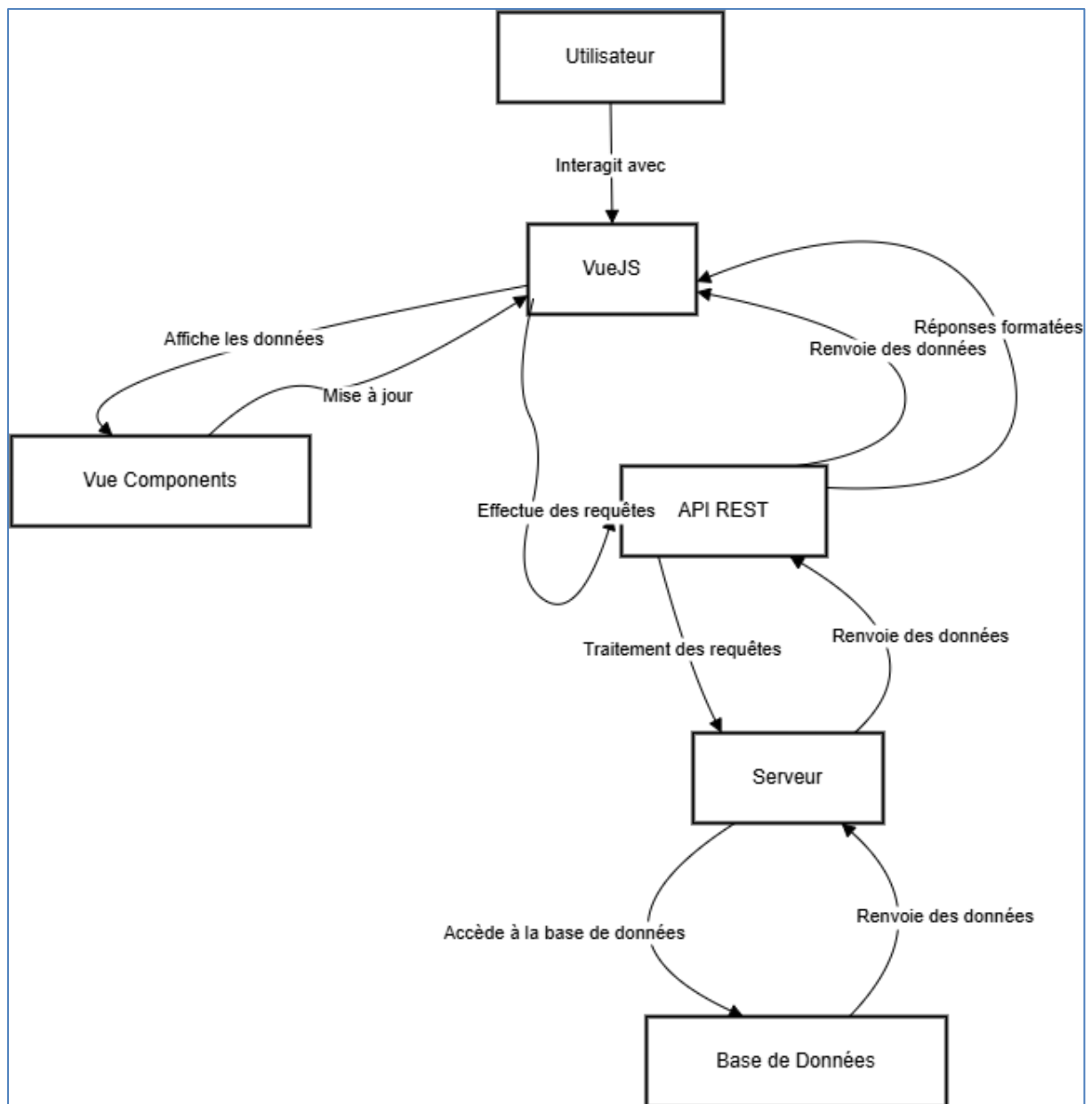
C'est la page de base du projet. Dans notre cas, elle est utilisée pour afficher le **header** du site.

#### 7) Main.js

C'est le point d'entrée principal de l'application frontend Vue.js.

C'est ce fichier qui initialise et lance l'application.

## 2.4 Schéma



## 3 RÉALISATION

### 3.1 Sécurité

Pour la sécurité, nous n'avons pas implémenté quelque chose de spécifique mais, vous avons des routes protégées qui, si nous ne sommes pas connectés, va nous rediriger vers la page de connexion.

Ceci a été fait en créant une route dans l'API REST, qui s'appelle « /api/auth/check », ce qui va vérifier s'il y a une connexion, ou bien des cookies.



Et dans le `index.js` de `Vuejs`, cette route est appelée et dans le tableau des routes on écrit « `meta : {requiresAuth : true}` » pour que cette route soit protégée.

Pour les admins, afin d'éviter que tout le monde puisse faire un CRUD sur tous les livres, nous avons mis en place dans la DB, un champ booléen qui s'appelle `isAdmin`, ce qui va nous permettre de savoir si un user est admin ou pas, et donc si c'est `true` alors l'utilisateur peut donc faire le CRUD sur les livres.

## 3.2 Explication des fonctionnalités techniques & leur implémentation

### 7.4 Fonctionnalités requises (d'un point de vue du client)

Le frontend de l'application devra avoir :

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

PS : le pied-de page du site doit faire mention de la personne qui a créé l'application ainsi que le moyen de la contacter.

- Page d'accueil : cette page est la page d'accueil où chaque visiteur va voir en premier, donc les 5 derniers livres ajoutés, classés par date et heure, ainsi qu'une explication du site et son utilisation
- La liste des ouvrages : cette page, autrement appelée la bibliothèque, comprendra tous les livres existants dans la DB classés par catégorie. Avec ces livres, nous avons l'image de couverture et le titre. Il y a également l'auteur puis l'utilisateur qui a créé ce livre, avec un lien qui nous ramène à la page de l'utilisateur sélectionné si on clique.
- Page d'ajout d'un livre : c'est une page protégée, qui nous permet de créer un nouveau livre. Il est possible d'écrire du texte sur le titre, nb de pages, extrait, résumé, l'année d'édition. Le reste des champs sont des champs où on doit choisir parmi ceux qui sont déjà existants. La raison est que, le manque de temps pour implémenter ceci, et des routes manquantes dans l'API REST car par exemple une route `POST /api/authors` n'était pas demandée ou elle ne faisait pas partie des fonctionnalités requises.
- Page de modification d'un livre : c'est aussi une page protégée, permettant aux utilisateurs connectés de modifier leurs propres livres créés. Dès que l'utilisateur accède à son profile, il peut voir la page où la liste de ses livres s'affichera. Il y a un bouton modifier, et aussi un bouton supprimer. Quand on clique sur le bouton modifier

sur le livre désiré, ceci va nous ramener à la page de modification où les informations existantes seront là à part l'image de couverture.

- Page de suppression d'un livre : cette page n'existe pas, il y a seulement un bouton dans la page « Mes livres » qui s'appelle supprimer en rouge, qui permet la suppression de ses livres.
- Page de détail d'un livre : Sur cette page, nous avons les détails principaux d'un livre comme l'image de couverture, le titre, le nombre de pages, l'extrait, le résumé, l'année d'édition, puis la date de publication sur le site. Ensuite, plus bas nous avons les appréciations ainsi que les commentaires. Il est possible de laisser une appréciation, ce qui va s'ajouter à la moyenne générale de toutes les appréciations sur le site web. Quand on envoie un commentaire, ceci s'affiche mais il n'y a pas d'indication sur qui l'a posté. Mais ceci n'est pas possible de se faire si nous ne sommes pas connectés.
- Pour les administrateurs, nous avons mis l'option de supprimer ou bien de modifier tous les livres sur le site web, même si ces livres n'appartiennent pas à l'admin. Ceci est fait par une condition dans le code qui, s'il détecte que l'utilisateur connecté a le champ booléen « isAdmin » en true, alors il a la possibilité de supprimer et modifier des livres.

### 3.3 Eco-conception

Dans le cadre de ce projet, l'éco-conception n'a pas été directement faite dès le début du développement. Mais cette partie reste essentielle dans une démarche de développement responsable et durable.

Pour un éventuel prochain projet, certaines bonnes pratiques pourraient être mises en place pour améliorer l'impact environnemental de l'application, telles que la réduction des appels API non essentiels, ou encore l'optimisation des fichiers statiques.

Ce projet peut donc servir de base à une future démarche d'éco-conception, afin d'optimiser les performances et de réduire l'impact écologique de l'application Passion Lecture.

## 4 TESTS

### 4.1 Tests réalisés

#### ○ Register

Tester la création d'un utilisateur en effectuant la manœuvre sur le site, puis vérifier dans la base de données si l'utilisateur a bien été ajouté.

Des tests dont aussi été faits sans mettre de nom, sans mot de passe, ou avec une confirmation de mot de passe différente du mot de passe principal.

Tous les tests ont réussi.

#### ○ Modifier

Tester la modification d'un livre, faire la manœuvre sur le site et vérifier dans la base de données si le livre a été modifié

Quelques modifications ont dû être apportés sur le formulaire pour que tout soit parfait

- **Ajouter**  
Tester l'ajout d'un livre, faire la manœuvre sur le site et vérifier dans la base de données si le livre a été modifié.  
Quelques modifications ont dû être apportés sur le formulaire pour que tout soit parfait
- **Supprimer**  
Tester la suppression d'un livre en cliquant sur le bouton **supprimer**, puis vérifier dans la base de données que le livre a bien été supprimé.
- **Connexion**  
Si l'utilisateur existe dans la base de données et qu'il remplit correctement le nom d'utilisateur et le mot de passe, il est connecté et reçoit un **token**.
- **Déconnexion**  
Quand l'utilisateur est connecté et qu'il clique sur le bouton **déconnexion**, il est redirigé vers la page d'accueil page et son **token** est supprimé.
- **Note**  
Quand une note est créée, elle apparaît bien dans la base de données et s'affiche sur la page de détails du livre.
- **Commentaire**  
Quand un commentaire est créé, il est bien enregistré dans la base de données et s'affiche sur la page de détails du livre.
- **Page détails**  
Quand on clique sur un livre dans la bibliothèque, on arrive bien sur la page de détails de ce livre. De plus, toutes les informations apparaissent correctement et sont au bon format.

## 5 CONCLUSION

### 5.1 Organisation du code (GitHub)

Le lien du repo GitHub : <https://github.com/po77axd/SSS-passionLecture>

Nous avons utilisé GitHub pour collaborer. Quand on veut faire une nouvelle fonctionnalité, on crée une Branch nommé « DEV-nomFonctionnalité », par exemple DEV-pageUser.

Chacun travaillait sur une fonctionnalité différente et sur une branche différente qui est liée à la fonctionnalité dont on va développer, afin d'éviter d'avoir des conflits dans la branche main.

### 5.2 Conclusion générale

Ce projet Vue.js, fait en collaboration avec 3 membres de l'équipe, a été une expérience qui nous a beaucoup plu. Travailler à trois nous a permis de mieux répartir les tâches, de partager nos connaissances, et de progresser ensemble tout au long du développement. Chacun a pu contribuer avec ses compétences et apprendre grâce aux autres, ce qui a favorisé le travail constructif et motivant.

Au-delà du développement de l'interface utilisateur, le lien direct avec le projet que nous avons eu le trimestre précédent — la création d'une API REST — a été particulièrement bien et utile. Ces projets nous ont permis de mieux comprendre les échanges entre le front-end et le back-end, et d'acquérir une vision plus complète du développement d'une application web moderne.

En somme, ce projet nous a permis de consolider nos bases en JavaScript et en Vue.js ainsi qu'en Express.js/Node.js, de découvrir les bonnes pratiques du développement web, et de travailler efficacement en équipe. Nous en ressortons avec de nouvelles compétences, une meilleure compréhension des outils utilisés, et l'envie de poursuivre dans cette voie avec des projets encore plus ambitieux.

### 5.3 Conclusion personnelle

**Romain :** J'ai pris beaucoup de plaisir à faire ce projet, car c'était la continuation d'un projet déjà réalisé. Il y avait donc un lien intéressant : au lieu de travailler sur quelque chose fait par quelqu'un d'autre, on a travaillé directement sur notre propre ancien travail.

De plus, l'équipe a bien fonctionné, il n'y a pas eu de conflits internes, donc les conditions de travail étaient bonnes.

Et pour finir, ce projet m'a permis d'approfondir certaines choses que je n'avais jusqu'ici comprises que de manière superficielle dans le module.

**Bastien :** Ce projet était intéressant car j'ai appris comment réaliser le frontend d'une application.

J'ai pu apprendre comment faire des appels à l'API en utilisant Axios. J'ai appris comment utiliser VueJS pour faire le frontend dynamique. J'ai aussi appris comment utiliser l'api composition avec les routes frontend. J'ai donc beaucoup amélioré mes compétences en frontend et ce projet me sera très utile pour mon futur.

**Samuel :** Ce projet m'a beaucoup plu, aussi bien par son aspect technique que par la manière dont il m'a permis de progresser. J'ai trouvé très intéressant de travailler avec Vue.js, un framework que je ne connaissais pas du tout avant. J'ai apprécié sa simplicité, sa flexibilité et la manière dont il facilite la création d'interfaces dynamiques et réactives.

J'ai également découvert l'importance de la communication entre le front-end et le back-end, notamment à travers l'utilisation d'une **API REST**. Comprendre comment récupérer, afficher et gérer des données depuis une API m'a permis de mieux comprendre l'architecture d'une application web complète.

Enfin, travailler en équipe sur ce projet a été une expérience très positive. Cela m'a permis de mieux m'organiser, de collaborer efficacement et travailler dans une bonne ambiance. Je ressors de ce projet avec de nouvelles compétences, de la motivation, et l'envie de continuer à apprendre dans ce domaine.

### 5.4 Critique Constructive

Même si le projet s'est globalement bien déroulé et nous a permis d'atteindre de nombreux objectifs, certaines fonctionnalités n'ont malheureusement pas pu être finies

comme nous avons voulu. Par exemple, lors de la création d'un livre sur le formulaire, il n'est actuellement pas possible d'ajouter ou de créer un nouvel auteur directement. Cela limite l'expérience utilisateur et montre qu'il reste encore des axes d'amélioration à explorer.

Ces limitations sont principalement dues au temps imparti et à la complexité technique de certaines interactions entre le front-end et l'API REST. Cela nous a permis de prendre conscience de l'importance de bien anticiper les besoins fonctionnels dès la phase de conception, et de mieux planifier les priorités de développement.

Malgré cela, cette expérience reste très formatrice : elle nous a permis d'identifier nos points forts, mais aussi les aspects sur lesquels nous devons encore progresser. Ce sont justement ces difficultés qui enrichissent l'apprentissage et donnent envie d'aller plus loin dans les projets à venir.

## 6 WEBOGRAPHIE

### 6.1 Webographie

Repo GitHub

<https://github.com/po77qxd/SSS-passionLecture>

Projet GitHub

<https://github.com/users/po77qxd/projects/1>