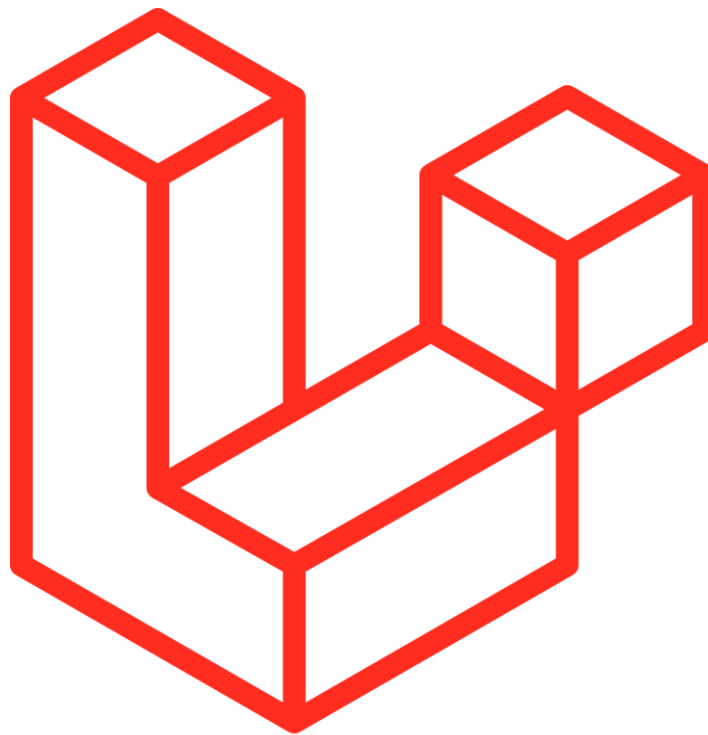


# P\_040-Web2

---



David Dieperink, Stefan Petrovic  
Robustiano Lombardo, Alexis Rojas  
CID2a  
ETML  
Gregory Charmier

# Table des matières

<b>1</b>	<b>SPÉCIFICATIONS</b>	<b>4</b>
1.1	TITRE	4
1.2	DESCRIPTION	4
1.3	MATÉRIEL ET LOGICIELS À DISPOSITION	4
1.4	PRÉREQUIS	4
1.5	CAHIER DES CHARGES	4
1.5.1	Objectifs et portée du projet (objectifs SMART)	4
1.5.2	Fonctionnalités requises (du point de vue de l'utilisateur)	4
1.5.3	Travail à réaliser par l'apprenti	5
1.5.4	Points supplémentaires	6
1.5.5	Si le temps le permet	6
1.5.6	Méthodes de validation des solutions	6
1.6	LES POINTS SUIVANTS SERONT ÉVALUÉS	6
1.7	VALIDATION ET CONDITIONS DE RÉUSSITE	6
<b>2</b>	<b>PLANIFICATION INITIALE</b>	<b>7</b>
<b>3</b>	<b>ANALYSE</b>	<b>7</b>
3.1	OPPORTUNITÉS	7
3.1.1	Difficultés potentielles	7
3.1.2	Solutions éventuelles	7
3.2	DOCUMENT D'ANALYSE ET CONCEPTION	7
3.2.1	Base de données	7
3.2.2	Mind Map	9
3.2.3	Maquette site web	9
3.2.4	Différences maquette et site web	10
3.3	CONCEPTION DES TESTS	11
3.3.1	Tests d'intégration	11
3.3.2	Tests fonctionnels	12
3.4	PLANIFICATION DÉTAILLÉE	13
<b>4</b>	<b>RÉALISATION</b>	<b>13</b>
4.1	DOSSIER DE RÉALISATION	13
4.1.1	Version des outils logiciels utilisés	13
4.1.2	Configurations spécifiques	13
4.1.3	Heroku	13
4.1.4	Notre utilisation de Laravel	14
4.1.4.1	Vues	14
4.1.4.2	Controller	15
4.1.4.3	Model	15
4.1.4.4	Migrations	17
4.1.4.5	Middleware	17
4.1.4.6	Factories / Seeds	17
4.1.5	Notre utilisation de Tailwind	18
4.1.5.1	Composants	18
4.1.5.2	CDN	19
4.1.6	Organisation – Répartition des tâches	19
4.2	MODIFICATIONS	19
<b>5</b>	<b>ECOCONCEPTION</b>	<b>20</b>
5.1	RÉDUCTION DE LA TAILLE DU DOM	20
5.2	LE SCROLL	20

5.3	ANIMATION .....	20
<b>6</b>	<b>TESTS.....</b>	<b>20</b>
6.1	DOSSIER DES TESTS .....	20
6.1.1	Tests d'intégrations.....	20
6.2	TESTS CI .....	21
<b>7</b>	<b>CONCLUSION.....</b>	<b>22</b>
7.1	BILAN DES FONCTIONNALITÉS DEMANDÉES .....	22
7.1.1	Fonctionnalités requises.....	22
7.2	BILAN DE LA PLANIFICATION .....	22
7.3	BILAN PERSONNEL.....	22
7.3.1	Avis David.....	22
7.3.2	Avis Alexis.....	22
7.3.3	Avis Stefan .....	22
7.3.4	Avis Robustiano.....	23
<b>8</b>	<b>DIVERS.....</b>	<b>23</b>
8.1	JOURNAL DE TRAVAIL.....	23
8.2	WEBOGRAPHIE .....	23
<b>9</b>	<b>ANNEXES .....</b>	<b>23</b>

# 1 SPÉCIFICATIONS

## 1.1 Titre

Passion Lecture

## 1.2 Description

Ce site est une bibliothèque en ligne, il permet de retrouver des informations sur des livres que les utilisateurs auront mis en ligne.

De plus les utilisateurs ont la possibilité de noter les livres.

## 1.3 Matériel et logiciels à disposition

- Microsoft Windows 10
- PHP Storm
- Visual Studio Code
- UwAmp
- Internet

## 1.4 Prérequis

Modules : 101, 431, 104, 302, 403, 404, 226A, 226B, 214, 133 et 151.

## 1.5 Cahier des charges

### 1.5.1 Objectifs et portée du projet (objectifs SMART)

Ce projet vise à mettre en œuvre les connaissances apprises dans les modules 133 et 151, qui se déroulent en parallèle au projet. Au final, l'application réalisée devra être exploitable et livrable. Dès lors, on attend un rendu professionnel et un soin particulier dans la documentation du projet.

### 1.5.2 Fonctionnalités requises (du point de vue de l'utilisateur)

Le site web / application aura les pages suivantes :

- 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 permettant d'ajouter une appréciation à un ouvrage (accès utilisateur).

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.

### 1.5.3 Travail à réaliser par l'apprenti

- Introduction
  - Comprend une brève explication du projet (1/2 page)
- Analyse
  - Contiendra une analyse quant à la réalisation et à la mise en page du HTML (1 page)
  - Contiendra une analyse de de la base de données à réaliser (MCD, MLD, MPD) (1 page)
  - Contiendra une analyse de la structure du code qui sera effectuée (Schéma UML, découpe du code ...) (2 pages)
- Réalisation
  - Comprend une explication de l'algorithme utilisé pour gérer l'identification (1 page)
  - Comprend une explication sur l'appréciation moyenne d'un ouvrage (1/2 page)
  - Comprend un manuel d'utilisation du site du point de vue admin ou utilisateur (comment ajouter un ouvrage, comment ajouter une appréciation) (2 pages)
- Test
  - Comprend une explication des tests réalisés (Unit Test) (1 page)
- Conclusion
  - Comprend une conclusion générale sur le projet (1/2 page)
  - Comprend une conclusion personnelle sur le projet (1/2 page)
  - Comprend une critique constructive sur la planification du projet (1/2 page)
- Webographie / Bibliographie / Glossaire

Concernant la méthode de projet, elle devra être annexée au rapport et sera imposée par le chef de projet. Pour la partie pratique, il sera rendu terminé à 100% au plus tard à la fin de l'antépénultième séquence. Néanmoins, les échéances suivantes doivent être respectées durant le projet :

- Démonstration de l'avancement de l'application : Echéance -> Séquence 4
- Démonstration de l'avancement de l'application et auto-évaluation intermédiaire : Echéance -> Séquence 8
- Démonstration de l'avancement de l'application : Echéance -> Séquence 12

Une présentation et l'auto-évaluation finale doivent être rendues à l'enseignant au début du cours de l'avant-dernière séquence. Les présentations commencent à ce moment.

Règles pour la présentation :

- min. 10 minutes de présentation

- max. 5 minutes de démonstration

#### 1.5.4 Points supplémentaires

- MVC avec Laravel
- Git / github
- Méthodologie SCRUM
- CSS avec Tailwind
- Tests unitaires + CI
- Déploiement de l'application sur HEROKU en mode Production

#### 1.5.5 Si le temps le permet ...

- Ajout d'une recherche pour retrouver un ouvrage
- Modification d'un ouvrage
- Suppression d'un ouvrage
- Ajout d'une personne
- Ajout d'un commentaire accompagnant une appréciation

#### 1.5.6 Méthodes de validation des solutions

Les tests d'intégrations seront écrits sous formes d'un tableau lors de l'analyse et ils seront réécrits lors de réalisation avec le résultat obtenu.

### 1.6 Les points suivants seront évalués

- Le rapport
- Les planifications (initiale et détaillée)
- Le journal de travail
- Le code et les commentaires
- Les documentations de mise en œuvre et d'utilisation

### 1.7 Validation et conditions de réussite

- Compréhension du travail
- Possibilité de transmettre le travail à une personne extérieure pour le terminer, le corriger ou le compléter
- Etat de fonctionnement du produit livré

## 2 PLANIFICATION INITIALE

Semaine N°	Date	N° séance	Activités	GCR
12	Mercredi 23.03.2022	1	Définition du cahier des charges	Voir comment le groupe s'organise Voir le dashboard "Scrum" dans github
12	Vendredi 25.03.2022	2		
13	Mercredi 30.03.2022	3		
13	Vendredi 01.04.2022	4		
14	Mercredi 06.04.2022	5		
14	Vendredi 08.04.2022	6		
15	Mercredi 13.04.2022	7	Démonstration de l'avancement	
15	Vendredi 15.04.2022		Jour Férié	
16	Mercredi 20.04.2022		Vacances	
16	Vendredi 22.04.2022		Vacances	
17	Mercredi 27.04.2022		Vacances	
17	Vendredi 29.04.2022		Vacances	
18	Mercredi 04.05.2022	8		
18	Vendredi 06.05.2022	9		
19	Mercredi 11.05.2022	10	Démonstration de l'avancement	
19	Vendredi 13.05.2022	11		
20	Mercredi 18.05.2022	12	Rendre le code de l'application + Rapport en fin de séance	
20	Vendredi 20.05.2022	13	Rendre la présentation en fin de séance	
21	Mercredi 25.05.2022	14	Présentation des 4 projets	
21	Vendredi 27.05.2022		Jour Férié	

## 3 ANALYSE

### 3.1 Opportunités

#### 3.1.1 Difficultés potentielles

- Comprendre le fonctionnement de Laravel + Tailwind
- Organisation du projet -> répartition des tâches à effectuer
- Mise en production du site web avec Heroku

#### 3.1.2 Solutions éventuelles

- Lire la documentation + faire l'initialisation avec les membres du groupe
- Répartition des tâches en fonctions des connaissances (au départ)

### 3.2 Document d'analyse et conception

#### 3.2.1 Base de données

Nous avons créé le modèle conceptuel de données et le modèle logique pour avoir une représentation du stockage des données du site web. Pour créer ces 2 modèles nous nous sommes mis à 4 pour prendre l'avis de tout le monde est que tout fonctionne bien lors de l'implémentation de la base de données.

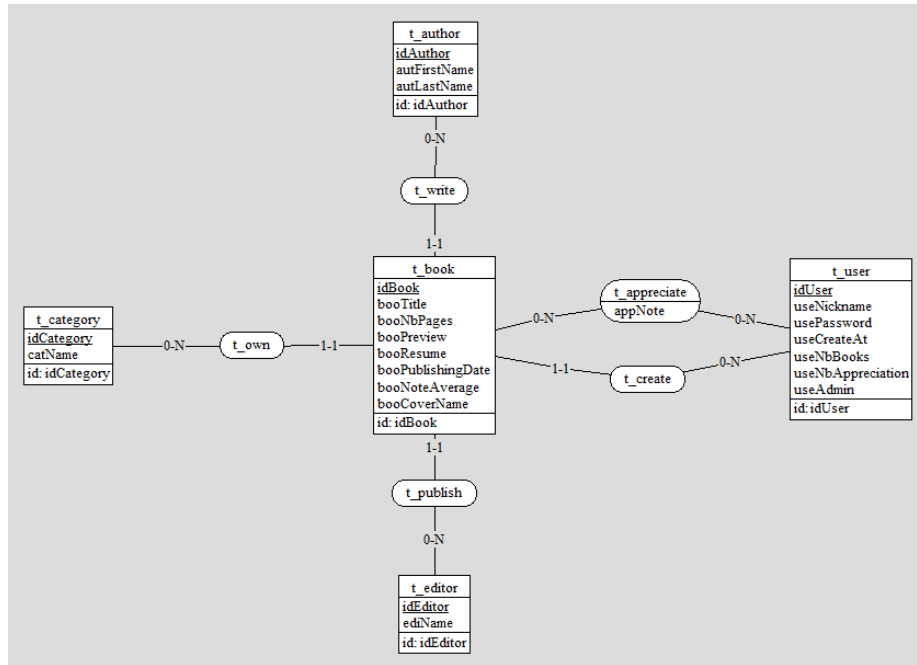


Figure 2 : MCD - Version 1

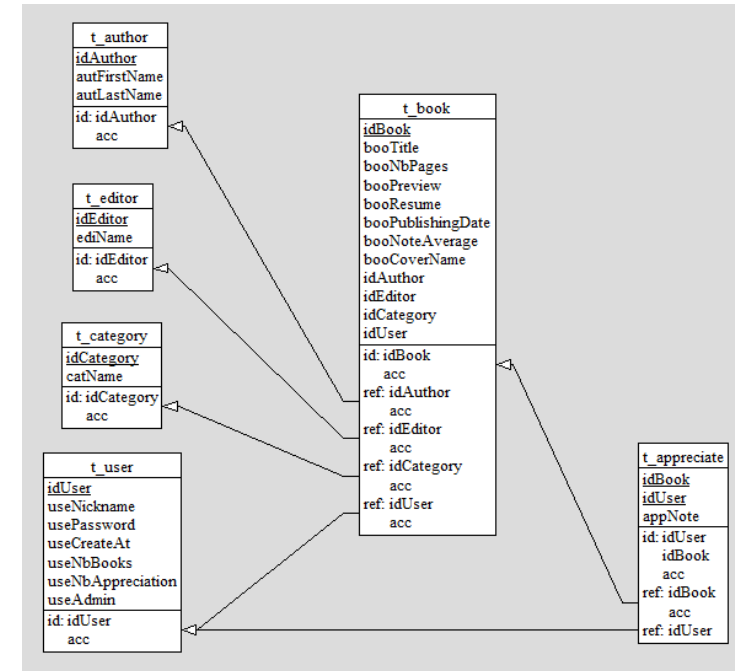
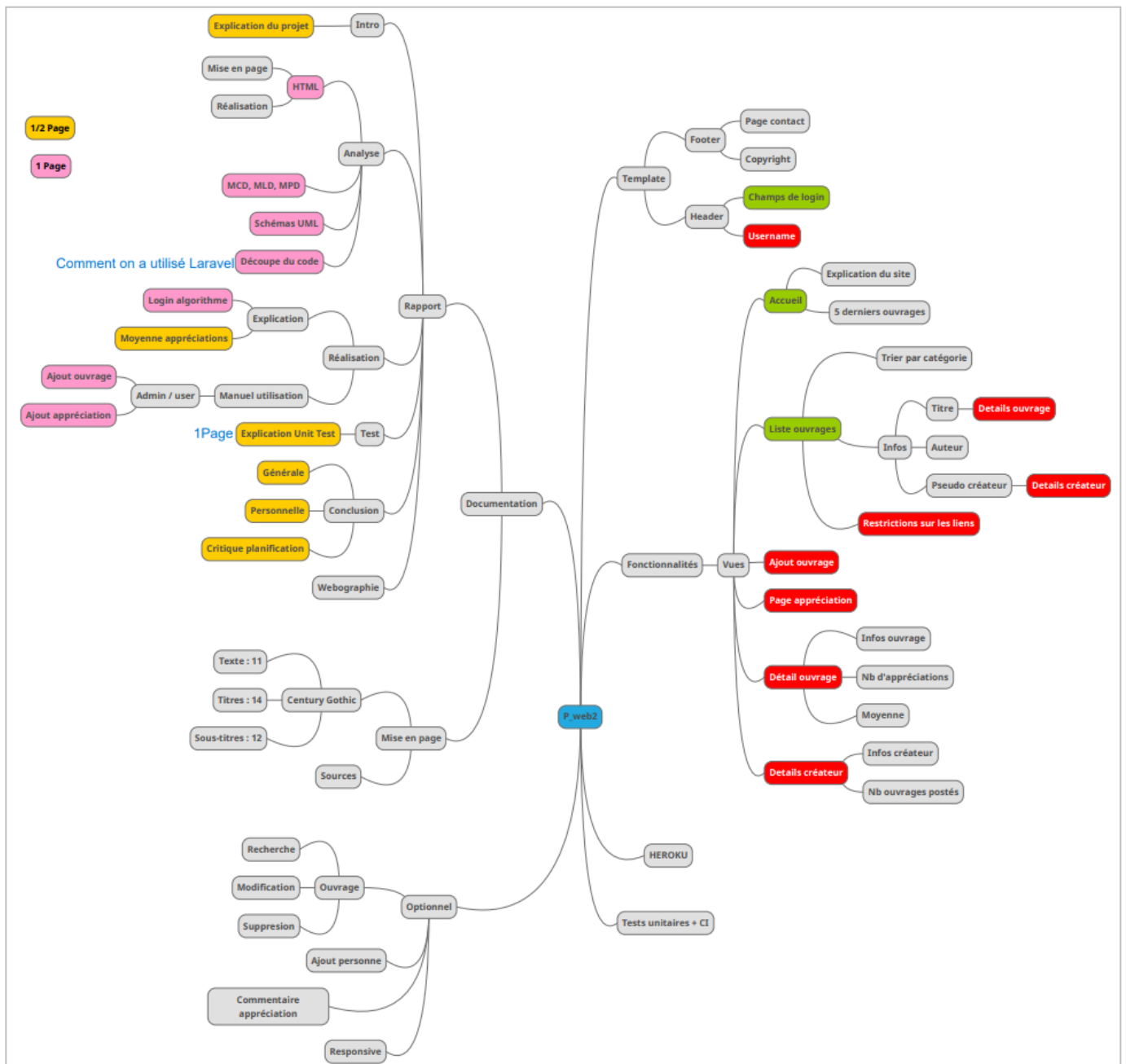


Figure 1 : MLD - Version 1



### 3.2.2 Mind Map



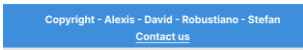

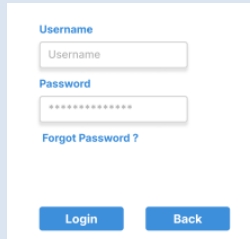
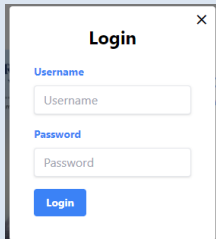
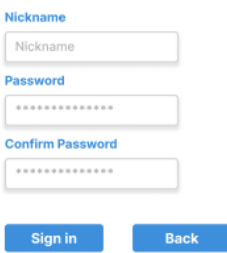
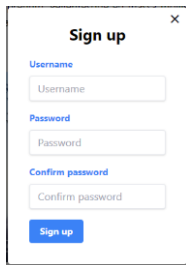


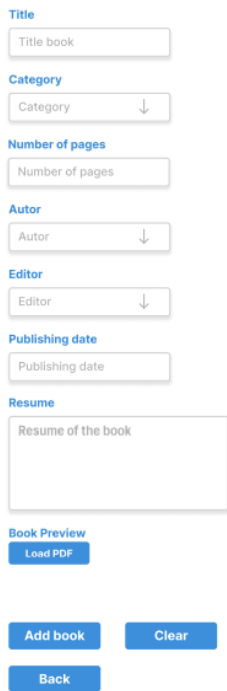
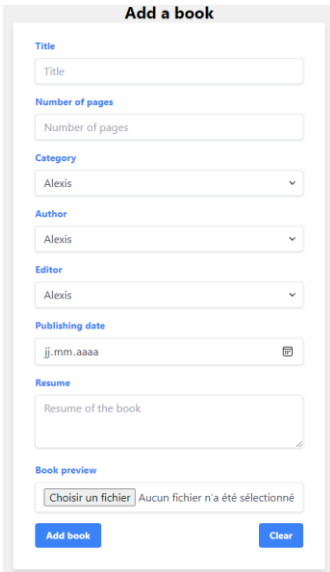


### 3.2.3 Maquette site web

Nous avons utilisé Figma pour faire notre maquette. La maquette est composée de plusieurs pages car nous avons fait chaque page de site web pour pouvoir les implémenter plus facilement. Voici le lien de la maquette :

<https://www.figma.com/file/KDZvY8uQ0cxHLIYnnxV2W4/Untitled?node-id=0%3A1>

## 3.2.4 Différences maquette et site web

Élément concerné	Maquette	Site	Raison
Carte d'affichage des ouvrages			Implémentation trop compliquée, mise en place problématique
Footer			Pas très esthétique après réflexion
Formulaire login			Le bouton back a été remplacé par une croix de fermeture.
Formulaire inscription			Le bouton back a été remplacé par une croix de fermeture.
Bouton de la barre de navigation			Problème de couleurs.
Formulaire d'ajout d'ouvrage			Retrait du bouton back et ajout d'un titre, uniformisation de la longueur des champs.

### 3.3 Conception des tests

#### 3.3.1 Tests d'intégration

Titre	Fonctionnalité	Description	Etapes	Résultat attendu
<b>Recherche livre réussie</b>	Recherche d'un livre	Recherche d'un livre avec son nom	1) Entrer le livre dans la barre de recherche 2) Appuyer sur le bouton « Search »	Le livre s'affiche lors de la recherche
<b>Recherche livre échouée</b>	Recherche d'un livre	Recherche d'un livre avec son nom	1) Entrer le nom du livre dans la barre de recherche 2) Appuyer sur le bouton « Search »	Aucun livre s'affiche car le nom est faux
<b>Filtrage livre par catégorie réussi</b>	Filtrer les livres par catégories	Filtrer l'affichage des livres par catégories	1) Choisir une catégorie dans la liste déroulante 2) Appuyer sur le bouton « Search »	Les livres de la catégories s'affichent
<b>Filtrage livre par catégorie échoué</b>	Filtrer les livres par catégories	Filtrer l'affichage des livres par catégories	1) Choisir une catégorie dans la liste déroulante 2) Appuyer sur le bouton « Search »	Les livres de la catégories ne s'affichent pas car aucun livre n'est dans la catégorie.
<b>Redirection sur la page de détail créateur</b>	Redirection sur la page detailsCreator	Redirection sur la page de détail du créateur	1) Appuyer sur le créateur du livre	La page de détail du créateur s'affiche
<b>Redirection sur la page d'ajout d'un livre</b>	Redirection sur la page addBook	Redirection sur la page d'ajout d'un livre	1) Appuyer sur le bouton « Add book »	La page pour l'ajout d'un livre s'affiche
<b>Redirection sur la page détail du livre</b>	Redirection sur la page bookDetails	Redirection sur la page de détail du livre	1) Appuyer sur le titre du livre	La page de détail du livre s'affiche

### 3.3.2 Tests fonctionnels

#### Scénario 1 : Login

Etape	Description	Remarque
<b>Arrange</b>	Informations de l'utilisateur	
<b>Act</b>	Vérifier les informations reçues avec celle de la base données	
<b>Assert</b>	Les informations reçues doivent correspondre à celle de la base de données	

Résultat : Le test fonctionnel fonctionne parfaitement.

Remarque :

#### Scénario 2 : Confirmation du mot de passe (inscription)

Etape	Description	Remarque
<b>Arrange</b>	Mot de passe	
<b>Act</b>	Vérifier que les deux champs soient les mêmes	
<b>Assert</b>	Les deux champs doivent correspondre	

Résultat : Le test fonctionnel fonctionne parfaitement.

Remarque :

#### Scénario 3 : Validation des données d'un formulaire

Etape	Description	Remarque
<b>Arrange</b>	Données du formulaire	
<b>Act</b>	Vérifier qu'elles correspondent aux critères	
<b>Assert</b>	Les données doivent correspondre aux critères demandés	

Résultat : Le test fonctionnel fonctionne parfaitement.

Remarque :

### 3.4 Planification détaillée

Aucun journal de travail n'a été demandé par le chef du projet car cela prend trop de temps et il préfère favoriser l'apprentissage de la technique.

## 4 RÉALISATION

### 4.1 Dossier de Réalisation

#### 4.1.1 Version des outils logiciels utilisés

Outils / Logiciels	Version
PHP Storm	2021.3.3.PS-213.7172.28
UwAmp	3.1.0
PHP	8.1.4
MySQL	5.7.11
Laravel	9
Tailwind	3.0.24
Git Bash	2.34
Windows	10

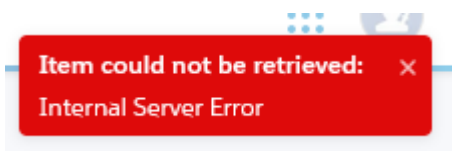
#### 4.1.2 Configurations spécifiques

Pour les configurations spécifiques nous avons créé des espaces de discussions sur le dépôt GitHub, cela permet que tous les membres du groupe puissent trouver l'information.

- Configuration de UwAmp en Host Virtual -> [lien](#)
- Configuration version de php à installer -> [lien](#)
- Configuration du fichier .env de laravel -> [lien](#)

#### 4.1.3 Heroku

La mise en production du site web grâce à Heroku n'a pas fonctionné. La liaison entre GitHub et Heroku ne fonctionne pas (18.05.2022). Lors de la connexion une erreur est générée, nous n'avons aucune idée de la provenance de celle-ci.



Suite à cela, nous avons essayé en utilisant le CLI de Heroku. Voici la marche à suivre que nous avons suivie.

1. **npm install -g heroku** cette commande installe heroku sur la machine.
2. **heroku login** cette commande permet de faire la connexion au site de heroku.
3. **heroku create** cette commande génère un projet heroku en local et sur le site d'heroku.

4. **git push heroku main** cette commande push heroku sur la branch main du dépôt GitHub.

Après cette dernière commande une erreur se produit, cela empêche donc de push la configuration heroku sur le dépôt GitHub.

```
To https://git.heroku.com/etml-p040-web2.git
! [remote rejected] main -> main (pre-receive hook declined)
error: failed to push some refs to 'https://git.heroku.com/etml-p040-web2.git'
```

Le push est rejeté, le dépôt refuse le push car la configuration de github côté serveur à un problème. [Lien](#) stackoverflow de l'erreur.

Toute la marche à suivre se trouve sur le site de [Heroku](#) et nous avons également regardé un tutoriel sur [Youtube](#) pour essayer mais rien à faire. Aucune solution n'a été trouvée.

Nous avons également regardé avec le groupe de Thomas et Damien pour savoir s'ils avaient des solutions, malheureusement ils n'en avaient pas.

#### 4.1.4 Notre utilisation de Laravel

Dans cette partie nous allons expliquer les concepts de laravel que nous avons utilisé pour le projet.

##### 4.1.4.1 Vues

Pour les vues, nous avons décidé d'utiliser des partials et des layout. Les partials nous sont très utiles, c'est des petits blocs de code que l'on peut implémenter sur toutes les pages en envoyant des informations. Par exemple pour l'affichage de nos livres nous avons fait comme ceci :

```
<div class="uppercase tracking-wide text-sm text-blue-500 font-semibold hover:under
<p class="block mt-1 text-20 leading-tight font-medium text-black">{{ $author }}</p>
<a href="/creatorDetails{{ $idUser }}" class="hover:underline">{{ $user }}</a>
<div class="flex items-center">
  <svg class="w-7 h-7 text-yellow-400" fill="currentColor" viewBox="0 0 20 20" xm
  <p class="ml-2 text-18 font-bold text-black dark:text-white">{{ $note }}</p>
```

Les champs avec les \$ sont des paramètres que le partial reçoit lorsqu'il est implémenté sur une page, les « {{ }} » représente un echo. Les informations sont envoyées comme ceci :

```
@forelse($books as $book)
  @include('partials.book-card',[
    'title'=>$book->booTitle,
    'author'=>$book->author->autFirstName,
    'user'=>$book->user->useNickname,
    'img'=>$book->booCoverName,
    'note'=>$book->booNoteAverage,
    'idUser'=>$book->idUser,
    'idBook'=>$book->idBook
  ])
@empty
  {{-- TODO: Info pas de valeurs --}}
@endforelse
```

Nous faisons un include du partial et grâce à une boucle forelse nous allons récupérer toutes les informations du livre et les envoyer au partial.

Le layout est un fichier qui est composé des éléments de base d'un site web, c'est-à-dire un header, un footer, les liens sur les fichiers css, etc. C'est une template qui est implémentée sur toutes les vues du site web pour que toutes les vues aient les mêmes paramètres. C'est le fichier qui structure les pages du site.

#### 4.1.4.2 Controller

Nous avons 3 controllers dans le projet. Le controller pour les livres, celui pour la page d'accueil et celui pour l'utilisateur.

Le controller pour les livres regroupe toutes les fonctions qui concernent les livres, par exemple la fonction pour rechercher un livre grâce à son nom.

```
public function searchBooks(Request $request){  
    $param = $request->input( key: 'booName');  
  
    $books = BookModel::where('booTitle', 'like', "%$param%")->get();  
  
    return view( view: 'bookList', ['books'=>$books]);  
}
```

Le paramètre d'entrée est le nom du livre, ensuite la variable \$books contiendra le contenu que le model « BookModel » lui retournera. Le contenu retourner sera le résultat de la requête, la requête cherche dans tous les livres quel livre contient le paramètre demandé.

#### 4.1.4.3 Model

Nous avons un model par table dans la base de données. Donc un model pour les livres, les auteurs, etc.

Pour les model et tout ce qui concerne la base de données nous avons utilisé le mappeur relationnel d'objet Eloquent qui est inclus dans Laravel. Eloquent permet d'interagir plus facilement entre les models et les tables liées.

Dans chaque model il faut définir le nom de la table, la clé primaire et les champs de la table.

```
protected $table = 't_appreciate';  
protected $primaryKey = ['idUser', 'idBook'];
```

Dans ce cas précis, la table « t\_appreciate » est une table pivot, donc elle ne possède pas sa propre clé primaire. Elle sert à faire la liaison entre la table t\_user et t\_book. Cela veut dire qu'elle possède une clé primaire composée pour faire la liaison.

```
protected $fillable = [  
    'idBook',  
    'idUser',  
    'appNote'  
];
```

Le \$fillable est le tableau qui regroupe les champs de la table.

**Nous avons dû spécifier tous les champs car nous avons utilisé les conventions de nommages de l'ETML et laravel ne les comprends pas car il utilise le « snake\_case ». Si nous avons utilisé les conventions de nommages de laravel nous n'aurions en aucun cas eu besoin de définir la clé primaire et les champs de la table.**

Il faut également savoir que chaque model possède ses propres fonctions, elles dépendent des liaisons entre les tables. Pour le model de la table t\_appreciate nous auront 2 fonctions dans le model.

```
public function user(){  
    return $this->belongsTo( related: UserModel::class, foreignKey: 'idUser', ownerKey: 'idUser');  
}
```

La fonction user() permet de faire la relation entre la table t\_appreciate et la table t\_user. Le belongsTo veut dire qu'une appréciation appartient à un utilisateur unique.

```
public function book(){  
    return $this->belongsTo( related: BookModel::class, foreignKey: 'idBook');  
}
```

La fonction book() permet de faire la relation entre la table t\_appreciate et la table t\_book. Dans ce cas-là le belongsTo veut dire qu'une appréciation appartient à un livre spécifique. Par contre un livre peut posséder plusieurs appréciations, cette fonction se trouve dans le model de la table t\_book.

```
public function appreciations(){  
    return $this->hasMany( related: AppreciateModel::class, foreignKey: 'idBook');  
}
```

Le hasMany veut dire que le livre possède potentiellement plusieurs appréciations.

Dans un autre cas de figure nous avons utilisé le hasMany. Cette fonction se trouve dans le model de la table t\_author, donc cela représente les auteurs.

```
public function books(){  
    return $this->hasMany( related: BookModel::class, foreignKey: 'idBook');  
}
```

Le hasMany veut dire qu'un auteur possède potentiellement plusieurs livres.



#### 4.1.4.4 Migrations

Les migrations nous ont servi pour créer la base de données sans utiliser un script SQL.

```
public function up()
{
    Schema::create( table: 't_user', function (Blueprint $table) {
        $table->id( column: 'idUser');
        $table->string( column: 'useNickname', length: 60);
        $table->string( column: 'usePassword', length: 255);
        $table->timestamp( column: 'useCreateAt')->default(DB::raw('CURRENT_TIMESTAMP'));
        $table->integer( column: 'useNbBooks')->default( value: 0);
        $table->integer( column: 'useNbAppreciation')->default( value: 0);
        $table->boolean( column: 'useAdmin')->default( value: 0);
    });
}
```

La migration pour la création de la table t\_user, tous les types sont spécifiés ainsi que les valeurs par défaut.

#### 4.1.4.5 Middleware

Un middleware a été créé pour sécuriser les routes du site web auxquelles seulement les utilisateurs connectés peuvent accéder.

S'il y a un utilisateur connecté, le middleware renvoie la page recherchée ; sinon, il le renvoie vers la page d'accueil et affiche le formulaire de connexion.

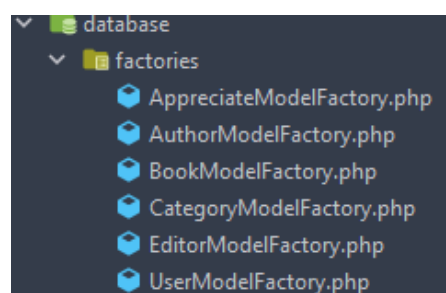
```
public function handle(Request $request, Closure $next)
{
    if (auth()->guest()) {
        return redirect( to: '/' )->withErrors([
            'userLogin' => 'You need to connect to see this page'
        ]);
    }
    return $next($request);
}
```

#### 4.1.4.6 Factories / Seeds

Des factories ont été créées pour insérer des données aléatoires dans la base de données afin de la tester.

On utilise également le DatabaseSeeder pour insérer un tas de données dans chaque table à chaque fois qu'on fait une migration.

Php artisan migrate:refresh --seed



Pour que la factory « AppreciateModelFactory » fonctionne il faut mettre en commentaire la définition des clés primaires dans le model de la table t\_appreciate. Etant donné que la clé primaire est une clé composée, c'est-à-dire qu'elle comporte 2 clés primaires, la factory génère des erreurs avec liaisons des clés primaires / étrangères.

```
/**
 * Nom des clés primaires
 * @var string[]
 */
protected $primaryKey = ['idUser', 'idBook'];
```

C'est la seule solution que nous avons trouvée pour que la factory fonctionne avec cette table. En faisant de cette manière les clés étrangères entre les table n'étaient pas présente donc les appréciations n'étaient pas liée à un utilisateur ni à un livre.

Pour toutes les autres tables les données ont été générées sans problèmes.

#### 4.1.5 Notre utilisation de Tailwind

Nous avons utilisé tailwind pour le css. Tailwind a été installé sur un PC pour pouvoir créer des composants. Le CDN a également été utilisé car nous n'avons pas eu le temps pour créer tous les composants.

##### 4.1.5.1 Composants

Les composants servent à définir le style pour un élément précis, par exemple pour un bouton, nous définissons un style comme ça lors de la création d'un bouton le style est déjà en place.

```
@layer components {
  .btnConnection {
    @apply text-white border-2 hover:bg-blue-800 focus:ring-4 focus:outline-none focus:ring-blue-300
  }
  .btnDefault {
    @apply bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4 rounded focus:outline-none;
  }
}
```

Le « .btnConnection » est le style pour les boutons qui concerne la connexion. Pour les inclure dans le php il faut faire cela.

```
<button onClick="openModal('modal-login')" id="login-btn" class="btnConnection">Login</button>
<button onClick="openModal('modal-register')" id="register-btn" class="btnConnection">Sign Up</button>
```

La class du bouton doit avoir le nom du composant.

#### 4.1.5.2 CDN

Le CDN c'est l'utilisation de tailwind avec un lien web, sans le cdn le style de notre site ne fonctionnerais pas sauf pour les éléments qui possède le style d'un composant.

```
<script src="https://cdn.tailwindcss.com"></script>
```

Pour utiliser le CDN il faut utiliser la balise script dans le layout du projet, étant donné que le layout est inclus sur toutes les pages le CDN peut être utilisé sur toutes les pages.

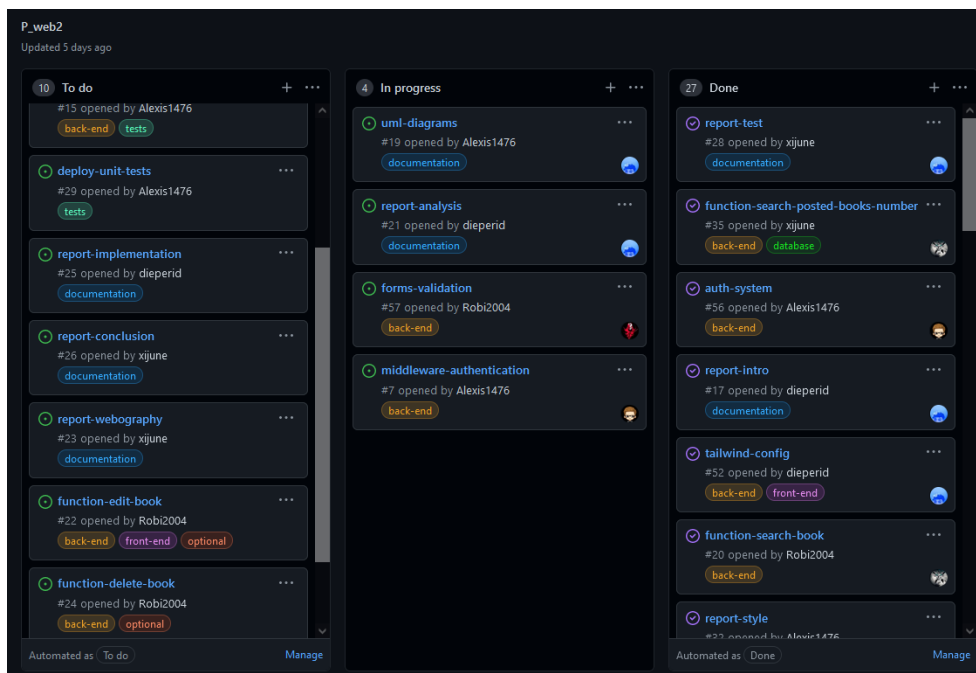
#### 4.1.6 Organisation – Répartition des tâches

En terme d'organisation nous avons réalisés un document en markdown qui sert de conventions de nommages pour le projet. Ce document a été réalisé au départ du projet pour que toute l'équipe utilise les mêmes nommages. Le document se trouve sur [GitHub](#).

Pour la répartition des tâches nous avons utilisé un trello sur GitHub. Les issues ont été créé au départ du projet.

Dès le début nous nous sommes mis à 4 pour créer un mind map qui nous a permis de nous représenter toutes les choses à faire sur le projet. Suite à cela nous avons créé à 4 le mcd et le mld pour la base de données.

Après tout cela, chaque membre du groupe a pris une tâche sur le trello qui l'intéressait et c'était comme ça jusqu'à la fin du projet.



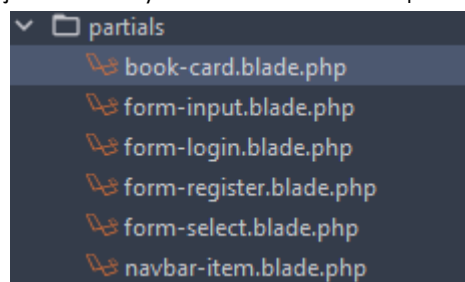
## 4.2 Modifications

Les modifications sont représentées par les commit sur le dépôt GitHub. Chaque personne a effectué des commit pour pousser ses modifications sur le dépôt distant.

## 5 ECOCONCEPTION

### 5.1 Réduction de la taille du DOM

Nous avons réduit la taille du DOM (Data Object Model). Cela réduit le temps de chargement et le temps de la latence entre les pages. Elles possèdent moins de contenu donc elles sont plus rapides à charger. Pour réduire cela nous avons créé des vues partielles, cela nous permet de réduire la répétition d'un élément dans un page de code ainsi que la taille du fichier.



### 5.2 Le Scroll

Nous avons également fait en sorte de réduire le nombre de pages ou l'utilisateur peut scroller. Les pages sont donc plus petites et moins conséquente à l'affichage. Cela aide à la réduction du DOM.

### 5.3 Animation

Les animations (gifs, carrousels, chatbots, etc.) inutiles ont été retirée. De plus nous avons limité les animations pour permettre une meilleure expérience à l'utilisateur et c'est bénéfique à l'écologie.

## 6 TESTS

### 6.1 Dossier des tests

#### 6.1.1 Tests d'intégrations

Titre	Résultat attendu	Résultat obtenu	Date
<b>Recherche livre réussie</b>	Le livre s'affiche lors de la recherche	Le livre s'affiche lors de la recherche	18.05.22
<b>Recherche livre échouée</b>	Aucun livre s'affiche car le nom est faux	Aucun livre s'affiche car le nom est faux	18.05.22
<b>Filtrage livre par catégorie réussi</b>	Les livres de la catégories s'affichent	<b>PAS IMPLEMENTE MANQUE DE TEMPS</b>	18.05.22
<b>Filtrage livre par catégorie échoué</b>	Les livres de la catégories ne s'affichent pas car aucun livre n'est dans la catégorie.	<b>PAS IMPLEMENTE MANQUE DE TEMPS</b>	18.05.22
<b>Redirection sur la page de détail créateur</b>	La page de détail du créateur s'affiche	La redirection fonctionne, les détails du créateur s'affichent.	11.05.22
<b>Redirection sur la page d'ajout d'un livre</b>	La page pour l'ajout d'un livre s'affiche	La redirection fonctionne, la page d'ajout de livre s'affiche.	11.05.22
<b>Redirection sur la page détail du livre</b>	La page de détail du livre s'affiche	La redirection fonctionne, les détails du livre s'affiche.	11.05.22

## 6.2 Tests CI

Pour l'implémentation des tests fonctionnels CI, on a utilisé les **Actions Github**.

Les [tests fonctionnels](#) sont testés à chaque fois qu'on fait un push sur la branche *main* ou *dev*.



Un fichier **yml** est créé dans le répertoire *github* à la racine du projet **Github** où se trouve la démarche pour lancer les tests fonctionnels :

1. Création du fichier « **env** »
2. Création de la base de données
3. Installation des dépendances composer
4. Génération d'une clé pour l'application
5. Mis à jour du driver **Chrome** + installation **Dusk**
6. Initialisation Chrome Driver
7. Démarrage Laravel Server + Définition variables d'environnement
8. Exécution des migrations
9. Exécution des tests **Dusk**
10. Captures d'écran en cas d'erreur
11. Log en cas d'erreur

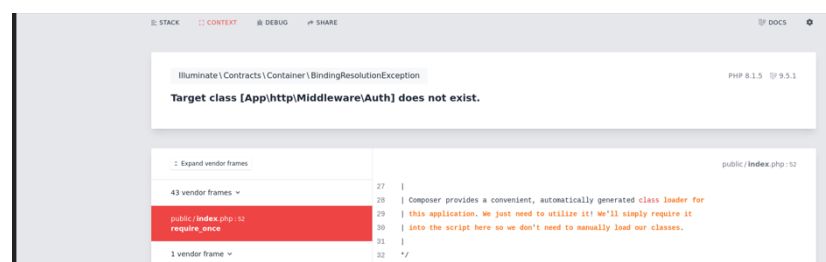
36 workflow runs			Event ▾	Status ▾	Branch ▾	Actor ▾
✓	Merge pull request #74 from Alexis1476/dev	main	2 days ago	46s	...	
CITests #36: Commit 77c27d pushed by Alexis1476						
✓	Dev suppression js inutile	dev	2 days ago	50s	...	
CITests #35: Pull request #74 opened by Alexis1476						
✓	Merge branch 'dev' of https://github.com/Alexis1476/et...	dev	2 days ago	45s	...	
CITests #34: Commit fd0af94 pushed by Alexis1476						

En cas d'erreur, un screenshot de l'erreur en question peut être téléchargé.  
Exemple :

3 Github > Actions > Runs

Artifacts	
Produced during runtime	
Name	Size
 console	3.68 KB
 screenshots	436 KB

4 Exemple de screenshot



## 7 CONCLUSION

### 7.1 Bilan des fonctionnalités demandées

Les fonctionnalités qui possèdent un « OK » en vert sont les fonctionnalités qui ont été atteintes lors du projet.

#### 7.1.1 Fonctionnalités requises

Le site web / application aura les pages suivantes :

- Une page d'accueil comprenant une explication de l'utilité du site ainsi que les cinq derniers ouvrages ajoutés (accès tout public). **OK**
- Une page comprenant la liste des ouvrages par catégorie (accès tout public avec restrictions sur les liens). **OK**
- Une page d'ajout d'un ouvrage (accès utilisateur). **OK**
- Une page permettant d'ajouter une appréciation à un ouvrage (accès utilisateur). **OK**

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. **OK**

Les fonctionnalités sont atteintes car, tous les éléments demandés figurent sur le site web.

### 7.2 Bilan de la planification

Le projet a pu être fini dans les temps. Nous n'avons pas réellement effectué de planification. Nous avons réalisé des user story qui concernaient les éléments à réaliser dans le projet. Elles ont toutes été terminées sauf les optionnelles.

### 7.3 Bilan personnel

#### 7.3.1 Avis David

Personnellement j'ai beaucoup aimé faire ce projet. Il m'a permis de voir deux framework web (Laravel et Tailwind). Sachant que je souhaite devenir développeur web fullstack cela m'a grandement aidé dans mon choix d'avenir et m'a fait prendre énormément de connaissances.

#### 7.3.2 Avis Alexis

J'ai eu beaucoup de plaisir à faire ce projet vu qu'on a utilisé deux frameworks différents (Laravel et Tailwind), ce qui m'a apporté beaucoup de connaissances et m'a permis de mettre en pratique plusieurs concepts et outils de Laravel que j'avais étudié au préalable.

Si j'avais à refaire ce projet je le ferais en respectant les conventions de nommage Laravel et en utilisant les composants Blade.

#### 7.3.3 Avis Stefan

Je trouve ce projet très utile pour la suite de ma formation, j'ai énormément appris durant ce projet. De plus, j'ai beaucoup apprécié le travail en

équipe qui m'a permis de mieux comprendre le fonctionnement de GitHub en groupe.

#### 7.3.4 Avis Robustiano

J'ai trouvé ce projet très intéressant il nous en a plus appris sur le php et surtout nous a permis d'apprendre quelques bases sur Laravel. J'ai également appris Tailwind mais aussi les tests unitaires en php. En bref, un projet très intéressant pour travailler en groupe et qui nous pousse à sortir de notre zone de confort.

## 8 DIVERS

### 8.1 Journal de travail

Le journal de travail se trouve sur [GitHub](#), c'est un trello de plus il y'a les commit log.

### 8.2 Webographie

PhpStorm : <https://www.jetbrains.com/phpstorm/>

Laravel : <https://laravel.com/>

Tailwind : <https://tailwindcss.com/>

Flowbite : <https://flowbite.com/>

## 9 ANNEXES

Laraguide : <https://www.youtube.com/c/ThibaudDauce/videos>