

# BiblioSolidaire

TPI

Elio Magnenat

Chef de projet : Dimitrios Lymberis

Expert 1: Süeleyman Ceran

Expert 2: Pascal Benzonana

30.04.2025 – 26.05.2025

Table des matières

[BiblioSolidaire 1](file:///C:\Users\pa13ynf\Documents\TPI\UwAmp\www\TPI\Documentation\R-TPI-elimagnenat-Rapport.docx#_Toc198823239)

[1 Analyse du cahier des charges 3](#_Toc198823240)

[1.1 Contexte 3](#_Toc198823241)

[1.2 Objectifs 3](#_Toc198823242)

[1.3 Enjeux et défis du projet 4](#_Toc198823243)

[1.4 Technologies utilisées 4](#_Toc198823244)

[1.5 Structure du rapport 4](#_Toc198823245)

[2 Conception 6](#_Toc198823246)

[2.1 Gestion de projet 6](#_Toc198823247)

[2.2 Planification initiale 6](#_Toc198823248)

[2.2.1 Documentation 6](#_Toc198823249)

[2.2.2 Analyse 6](#_Toc198823250)

[2.2.3 Conception 7](#_Toc198823251)

[2.2.4 Réalisation 7](#_Toc198823252)

[2.2.5 Test 7](#_Toc198823253)

[2.3 Architecture de l’application 7](#_Toc198823254)

[2.4 Base de données 8](#_Toc198823255)

[2.5 Maquette du site 9](#_Toc198823256)

[2.5.1 Structure globale 10](#_Toc198823257)

[2.5.2 Importation/Exportation 10](#_Toc198823258)

[2.5.3 Catalogue 11](#_Toc198823259)

[2.5.4 Formulaire d’ajout d’un livre 12](#_Toc198823260)

[2.5.5 Formulaire de modification d’un livre 14](#_Toc198823261)

[2.5.6 Suppression d’un livre 14](#_Toc198823262)

[2.5.7 Page détail des livres 15](#_Toc198823263)

[2.5.8 Page liste des élèves 16](#_Toc198823264)

[2.5.9 Ajouter un élève 16](#_Toc198823265)

[2.5.10 Modifier un élève 17](#_Toc198823266)

[2.5.11 Page détail d’un élève 19](#_Toc198823267)

[2.5.12 Formulaire d’emprunt 19](#_Toc198823268)

[2.5.13 Formulaire de rendu 20](#_Toc198823269)

[3 Développement et implémentation 21](#_Toc198823270)

[3.1 Mise en place de l’environnement 21](#_Toc198823271)

[3.2 Base de données 21](#_Toc198823272)

[3.3 Développement 23](#_Toc198823273)

[3.3.1 Structure Globale 23](#_Toc198823274)

[3.3.2 Page du catalogue 24](#_Toc198823275)

[3.3.3 Formulaire d’ajout d’un livre 26](#_Toc198823276)

[3.3.4 Formulaire de modification de livres 28](#_Toc198823277)

[3.3.5 Page détail d’un livre 29](#_Toc198823278)

[3.3.6 Liste des élèves 30](#_Toc198823279)

[3.3.7 Formulaire d’ajout d’un élève 31](#_Toc198823280)

[3.3.8 Formulaire de modification d’un élève 32](#_Toc198823281)

[3.3.9 Page détail d’un élève 33](#_Toc198823282)

[3.3.10 Formulaire d’emprunt d’un livre 34](#_Toc198823283)

[3.3.11 Formulaire de rendu d’un livre 35](#_Toc198823284)

[3.3.12 Import/Export d’un livre 36](#_Toc198823285)

[4 Tests 38](#_Toc198823286)

[4.1 Plan des tests à effectuer 38](#_Toc198823287)

[4.1.1 Environnement de test 38](#_Toc198823288)

[4.1.2 Scénario de test 38](#_Toc198823289)

[4.2 Résultats des tests 38](#_Toc198823290)

[5 Conclusion 38](#_Toc198823291)

[6 Annexes 38](#_Toc198823292)

[6.1 Résumé du rapport du TPI / version succincte de la documentation 38](#_Toc198823293)

[6.2 Webographie 38](#_Toc198823294)

[6.3 Glossaire 38](#_Toc198823295)

[6.4 Journal de travail 38](#_Toc198823296)

[6.5 Manuel d'Installation 38](#_Toc198823297)

[6.6 Manuel d'Utilisation 38](#_Toc198823298)

# Analyse du cahier des charges

Ce chapitre contient une analyse détaillée du cahier des charges en commençant par une mise en contexte du projet suivi d’une définition des objectifs et des enjeux de ce développement avant d’introduire les technologies qui seront utilisées. Et finalement je conclurai cette analyse en présentant la structure globale de ce document.

## Contexte

Ce projet a été réalisé dans le cadre de mon Travail Pratique Individuel (TPI) à l’ETML. C’est un développement web basé sur un cahier des charges précis qui demande des compétences en programmation, base de données et gestion de projet.

Il consiste à réaliser une application web autonome de gestion de bibliothèque pour zones sans internet. Pour ce faire je dois mettre en place une interface utilisateur simple, intuitive et responsive, ainsi qu’un backend intégrant de nombreuses fonctionnalité que je décrirai plus tard dans ce document.

## Objectifs

L’objectif principal du projet est de permettre à l’association AfricanPuzzle de doter des bibliothèques scolaires, situées dans des zones rurales au Bénin, d’un outil informatique local facilitant la gestion quotidienne de la bibliothèque.

Plus précisément l’application doit permettre :

* Au bibliothécaire de gérer le catalogue de livres (ajout, modification, suppression, gestion des emprunts)
* À un superviseur hors du bénin, grâce à un système d’import et d’export, de suivre l’activité de la bibliothèque.

Pour répondre à ces besoins, l’application doit être le plus simple possible d’utilisation.

## Enjeux et défis du projet

Sur le plan technique l’un des principaux défis est de concevoir une application web totalement fonctionnelle sans internet ce qui implique qu’au lieu de charger des dépendances via des url externe je devrai télécharger tous les fichiers et les mettre dans mon projet ce qui aura comme conséquence de l’alourdir. Il faudra donc minimiser les dépendances pour réussir à respecter l’un des autres défis qui demande que l’application soit utilisable sur des PC peu performant.  
  
Sur le plan fonctionnel, l’interface doit permettre à des utilisateurs souvent peu familiers avec l’informatique d’utiliser efficacement l’outil ce qui demande qu’elle soit épurée et intuitive.  
  
Et enfin sur le plan méthodologique et organisationnelle ce travail demande beaucoup de rigueur tant dans la réalisation que dans la planification pour respecter les délais impartis.

## Technologies utilisées

Le développement de BiblioSolidaire repose sur différentes technologies combinées permettant son utilisation hors-ligne :

**Frontend**

* HTML/CSS pour la structure et le style de l’interface
* Javascript pour certaines animations et interactions
* Bootstrap 4 pour une interface responsive et rapide à mettre en place
* Font Awesome pour l’affichage d’icônes

**Backend**

* PHP 8 utilisé pour la logique métier

**Base de données**

* MySQL avec phpMyAdmin

**Autres outils**

* DataTables pour l’affichage et la gestion des listes
* GitHub pour le versioning
* Uwamp qui servira au déploiement local de l’application et pourra être utilisé sur les postes des bibliothécaires pour utiliser l’application sans connexion internet

Pour gagner du temps dans le développement et le choix du design le thème gratuit SB Admin 2 a été utilisé.

## Structure du rapport

Ce document est structuré comme ceci :

* **Chapitre 1 : Analyse du cahier des charges** – Présentation des besoins, du contexte et des contraintes.
* **Chapitre 2 : Conception** – Explication des choix techniques et de la structure de la db.
* **Chapitre 3 : Développement et implémentation** – Description des fonctionnalités et du code.
* **Chapitre 4 : Analyse du cahier des charges** – Présentation des besoins, du contexte et des contraintes.
* **Chapitre 5 : Tests et validation** – Plan de test et résultats.
* **Chapitre 6 : Annexes** – Liens et documents utiles

# Conception

Ce chapitre conséquent présente l’ensemble des choix conceptuels qui ont dû être fait pour créer le site BiblioSolidaire.

## Gestion de projet

L’un des premiers choix à faire a été celui de la méthode à utiliser pour organiser ce travail. La méthode des 6 pas a été celle retenue car elle est simple à mettre en place et tout à fait efficace pour un projet réalisé par une seule et même personne.  
  
Le projet a donc été divisé en 6 parties :

* **S’informer** : cette phase contient principalement la lecture du cahier des charges et la discussion des points techniques avec le chef de projet.
* **Planifier** : définir chaque tâche et les organiser dans le temps imparti.
* **Décider** : faire des choix techniques et définir des maquettes.
* **Réaliser** : Créer le site et chacune de ses fonctionnalités.
* **Contrôler** : tester que tout soit présent et fonctionnel.
* **Evaluer** : Faire un bilan du projet comprendre ce qui a fonctionné ou non et définir de potentiels améliorations.

## Planification initiale

Pour réaliser ce projet du temps imparti, le travail a été organisé en plusieurs phases, chacune contenant des tâches plus spécifiques.

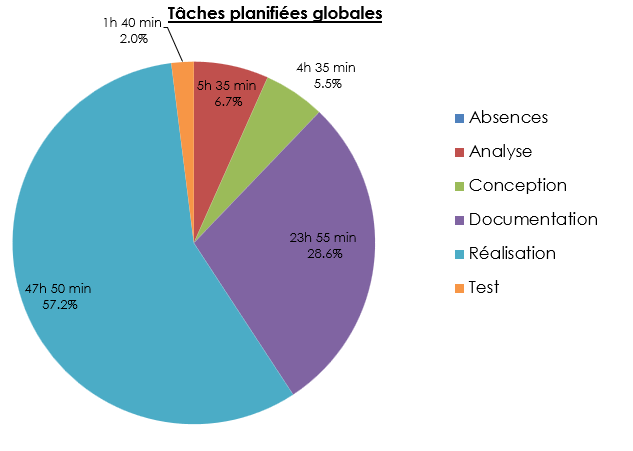


Figure 1 planification initiale

### Documentation

Cette tâche très conséquente qu’est la documentation (estimée à 23h55) sera répartie tout au long du projet avec une grosse partie d’analyse et de description faite au début, mais aussi un suivi tout au long du projet des fonctionnalités réalisées et finalement avec la documentation des tests et une analyse/bilan du projet à la fin du temps imparti.

### Analyse

Cette partie contient la planification du travail ainsi que l’analyse du cahier des charges. Elle a été estimée à 5h35 avec une grosse partie du temps destiné à la planification (4h).

### Conception

Regroupant la création des maquettes et de la base de données, cette phases est estimé à 4h35 avec cette fois ci les maquettes qui devrait prendre plus de temps car au moment de leur création il y a toute la partie réflexion et définition du fonctionnement de l’application.

### Réalisation

Cette phase devrait occuper plus de la moitié de la durée du projet avec une estimation à 47h50. Elle contiendra le développement du site de A à Z et donc de nombreuses tâches plus ou moins conséquente comme la création du footer qui devrait être très rapide (estimé à 30 minutes) et la gestion de l’exportation (estimé à 1,5 jours).

### Test

Cette dernière phase estimée à 1h40 consistera comme son nom l’indique à contrôler le bon fonctionnement de chacune des fonctionnalités du site.

## Architecture de l’application

Le projet est basé sur une architecture MCV (Modèle -Vue – Contrôleur) qui m’a permis d’organiser proprement mon code en séparant les logiques. Cette séparation facilite la compréhension du code pour un développeur qui reprendrai le site et permet de diminuer la redondance de code.



Figure 2 MVC

1. L’utilisateur demande une page.
2. Le contrôleur, si besoin, fait appel au modèle pour récupérer les informations requises depuis la base de données.
3. Le contrôleur va appeler la vue qui correspond à la demande de l’utilisateur et lui fournit les données nécessaires.
4. La page HTML demandée est retournée.

Ce concept est structuré dans mon code de cette manière :

**Fichier racine**

index.php se situe à la racine du projet et est utilisé comme point d’entrée de l’application.

**Fichier Routes**

Situé dans le dossier routes il s’occupe de rediriger les requêtes de l’utilisateur vers le bon contrôleur en fonction des paramètres de l’URL ($\_GET['controller']).

**Dossier Controller**

Comme son nom l’indique ce dossier contient tous les différents contrôleurs de l’application et notamment le controller.php qui instancie la classe parente abstraite dont héritent tous les autres contrôleurs.

**Dossier Model**

Ce dossier contient notamment le fichier database.php qui centralise la gestion des connexions à la base de données ainsi que toutes les requêtes SQL qui seront utilisées et appelés par les repository présents eux aussi dans ce dossier.

**Dossier View**

Regroupe tous les fichiers contenant de l’html comme le footer, la nav et le header mais aussi les différentes pages.

**Dossier ressources**

Contient toutes les images du site, mes fichiers css et js mais aussi tous les fichiers nécessaires au thème SB Admin 2.

## Base de données

Ce site devant stocker des données tel que les livres, les utilisateurs et les emprunts il est impératif d’avoir une base de donnée.

Une image contenant texte, diagramme, capture d’écran, Plan

Le contenu généré par l’IA peut être incorrect.**MCD**

Figure 3 MCD

**MLD**

Une image contenant texte, capture d’écran, diagramme, ligne

Le contenu généré par l’IA peut être incorrect.

Figure 4 MLD

Cette base de données composée de 5 tables est en réalité légèrement simplifiée de ce qu’il faudrait faire pour avoir une application la plus proprement réalisée mais dans un souci de gestion du temps imparti j’ai décidé d’utiliser cette structure pour mon TPI. Une version améliorée sera proposée plus loin dans ce document. Dans cette version, nous pouvons retrouver la table des prêts qui fait le lien entre les tables livre et étudiant en stockant les fk mais aussi les dates nécessaires à la gestion des prêts ainsi qu’un potentiel commentaire sur l’état du livre au moment du rendu. Deux petites tables sous-jacentes à celle des livres permettent une meilleur gestion des statuts ainsi que des catégories.

## Maquette du site

Les maquettes ont été fait sur Figma et ont été conçue en tenant compte des besoins fonctionnels décrits dans le cahier des charges.

### Structure globale



Figure 5 Maquette structure globale

Cette structure a été conçue en se basant sur le thème SB Admin 2 et en gardant à l’esprit qu’il faut une interface la plus simple et la plus épurée possible.

La navigation repose sur un bandeau latéral fixe à gauche qui permet de naviguer entre les deux pages principales : le catalogue (gestion des livres) et les élèves (gestion des adhérents).

Le header contient uniquement le tire du site ainsi qu’un bouton permettant de gérer l’importation et l’exportation.

Le footer rappelle l’origine du projet en mettant en avant le logo ainsi que le slogan de l’association African Puzzle.

Et finalement la zone centrale de l’écran sera réservée à l’affichage du contenu.

### Importation/Exportation



Figure 6 Maquette Importer/Exporter

Les trois petits points situés dans le header aura comme fonction, une fois cliqué, de déplier les deux options possible : Importer ou Exporter les données de l’applications.

Si l’utilisateur choisi « exporter », un fichier contenant l’entièreté des données de l’application sera automatiquement téléchargé sur son poste.

En revanche si l’utilisateur choisi « importer », l’explorateur de fichier s’ouvrira pour qu’il puisse sélectionner son fichier stocké localement. Une fois le fichier choisi, un message lui sera affiché pour s’assurer qu’il comprend que toutes les données actuelles de l’application seront écrasées.

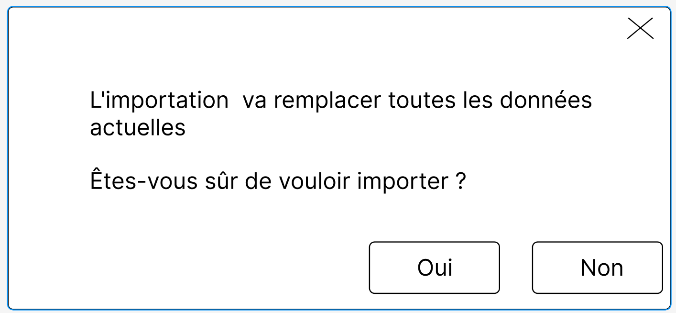


Figure 7 Maquette confirmer importer

### Catalogue

La page catalogue sera la page ouverte par défaut à l’arrivée sur le site et elle affichera dans une liste tous les ouvrages avec les colonnes qui peuvent être utile pour la recherche.



Figure 8 Maquette Catalogue

Un bouton en haut à droite permet d’ajouter un livre en étant redirigé vers le formulaire de création de livre.  
  
La barre de recherche située juste en dessous du bouton d’ajout permet de rechercher parmi toutes les colonnes pour trouver un livre ou voir les retards par exemple en cherchant retard.  
  
Chacune des colonnes ont un bouton avec une flèche vers le haut et une vers le bas qui permet de changer l’ordre d’affichage des données. Par exemple si l’on clique sur le bouton de tri pour la colonne Titre, les ouvrages seront ordonnés alphabétiquement ou dans le sens inverse si l’on clique à nouveau dessus.

La colonne Statut peut comporter plusieurs valeurs comme « En rayon » si le livre peut être emprunté, « Emprunté » si le livre est actuellement emprunté, « En retard » si le livre est emprunté et que la date de retour est dépassée ou encore « Retiré » si le livre n’est plus disponible mais que l’on souhaite en garder une trace.

La dernière colonne permet d’effectuer différentes actions de gestion qui seront décrites dans leurs chapitres respectifs.

En bas du tableau, des boutons permettent de changer de page lorsqu’il y a trop de données pour être affichées sur une page.

### Formulaire d’ajout d’un livre

Après avoir cliqué sur le bouton « Ajouter un livre » présent sur la page catalogue, l’utilisateur sera redirigé vers ce formulaire :



Figure 9 Maquette Ajout d'un livre

Il pourra y remplir tous ces champs assez classiques et choisir une photo du livre via l’explorateur de fichier grâce au champs « Photo ».

Le champ Catégorie est un menu déroulant permettant de choisir parmi les différentes catégories d’ouvrages.

Le champ remarque permet par exemple de mettre un commentaire sur son état.

Pour permettre au bibliothécaire de facilement retrouver un ouvrage, les champs référence et emplacement font respectivement référence à l’étagère et au rayon de l’étagère.

### Formulaire de modification d’un livre



Figure 10 Maquette Modifier un livre

Ce formulaire est accessible en cliquant sur le bouton de modification présent dans la colonne action du catalogue.

  
Il est identique à celui de création et peut-être validé au moyen du bouton « Enregistrer les modifications » qui renverra à nouveau vers la page catalogue.

### Suppression d’un livre

La suppression se fait par le biais du bouton poubelle situé dans la colonne action du catalogue.



Cette poubelle a pour fonction de retirer un livre des rayons en lui mettant le statut retiré mais s’il a déjà été retiré, la poubelle aura pour effet de complètement supprimer le livre ainsi que son activité.

### Page détail des livres



Figure 11 Maquette Détail d'un livre

Il est possible d’avoir une vue plus détaillée d’un livre en cliquant sur le bouton « œil » présent lui aussi dans la colonne action du catalogue.



Sur cette page tous les champs du livre seront visibles y compris l’image et une potentielle remarque sur son état.   
  
En dessous de ce détail on retrouvera l’historique des personnes qui l’ont emprunté avec un potentiel commentaire de retour si le livre a été abimé par exemple. On peut aussi voir les dates d’emprunt, de rendu prévu ainsi que celle de rendu réel.  
  
Les identifiant des élèves seront cliquables pour accéder au détail de leurs profils.

Et finalement nous pourront retourner consulter le catalogue grâce au bouton « retour au catalogue ».

### Page liste des élèves



Figure 12 Maquette Liste des élèves

La liste des élèves est très similaire à celle des livres à la seule différence des noms de colonnes et des actions possible.  
  
Le contenu de la colonne identifiant est créé automatiquement en prenant la première lette du prénom, la première lettre du nom suivi de la dernière lettre du nom.  
  
Le champ date de validation définie la durée d’activation du compte avant que l’utilisateur doivent se présenter à nouveau vers le bibliothécaire pour renouveler son compte.

### Ajouter un élève

Un élève peut être créé en cliquant sur le bouton ajouter un élève présent sur la liste des élèves.



Figure 13 Maquette Ajout d'un élève

L’utilisateur peut valider l’ajout en cliquant sur le bouton ajouter à condition qu’au minimum les champs Nom, Prénom, Date d’entrée, Date de validation et photo soient remplis. L’utilisateur sera ensuite automatiquement redirigé vers la page catalogue.

### Modifier un élève

Un élève peut être modifier en cliquant sur le bouton prévu à cet effet dans la colonne actions de la page liste des élèves.





Figure 14 Maquette Modifier un élève

### Page détail d’un élève



Figure 15 Maquette Page détail d'un élève

La page détail d’un élève est très similaire à celle des livres à quelques différences près. Premièrement un bouton situé en haut à droite permet d’imprimer la fiche de l’élève. Et deuxièmement l’historique en bas est celui des livres qu’il a empruntés avec toujours le nom du livre qui est un lien de redirection vers sa page de détail.

### Formulaire d’emprunt



Figure 16 Maquette Formulaire d'emprunt

Ce formulaire très basique est accessible via ce bouton dans la colonne action du catalogue :  
Une image contenant Bleu électrique, logo, bleu, symbole

Le contenu généré par l’IA peut être incorrect.

Il permet au bibliothécaire de saisir l’emprunt d’un livre en sélectionnant l’élève dans un menu déroulant ainsi que la date à laquelle l’emprunt est fait (par défaut date du jour) et la date à laquelle il devrait l’avoir rendu (par défaut 30 jours).

### Formulaire de rendu



Figure 17 Maquette Formulaire de rendu

Accessible en cliquant dans la colonne action du catalogue sur le bouton :  


Le champ date de rendu sera automatiquement rempli avec la date du jour mais pourra être modifié si besoin. Le champ commentaire permet de garder une trace sur l’état du livre au moment du rendu.

# Développement et implémentation

## Mise en place de l’environnement

Uwamp a été téléchargé sur le poste avec la version 8.0.1 de php qui est une version stable et compatible avec mes dépendances. J’ai laissé la version par défaut de MySQL sur Uwamp qui est la 5.7.11 car elle est suffisante pour l’utilisation prévue.  
  
Un repos git TPI a été créé dans le dossier www de Uwamp pour permettre le versionning de mon développement.  
  
La structure de dossier nécessaire au MVC décrit précédemment a été créé dans le repos TPI.

Le thème SB Admin 2 a été ajouté à l’intérieur et toutes ses dépendances ont été téléchargée.

## Base de données

La mise en place de la base de données sur phpMyAdmin a été assez rapide grâce au schéma MLD déjà fait lors de la conception.

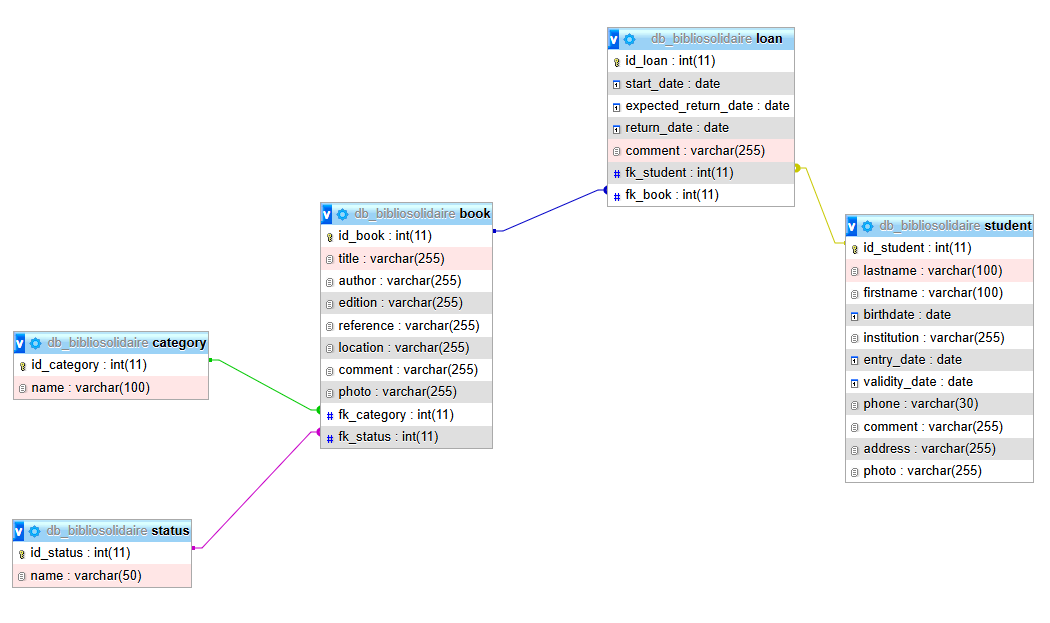


Figure 18 MPD Base de données

J’ai choisi de mettre des varchar de taille 100 pour tout ce qui est nom/prénom ce qui couvre sans problème les noms les plus longs, une taille de 30 pour les numéros de téléphone ce qui est suffisant pour les numéros au Bénin, une taille de 255 pour tous les champs plus grands afin d’être large et finalement des champs date pour les différentes dates.  
  
Pour des questions de sécurité j’ai créé un compte utilisateur « bibliosolidaire » qui sera utilisé dans le code et qui a uniquement des droits sur les données mais pas sur la structure ou la gestion des droits.

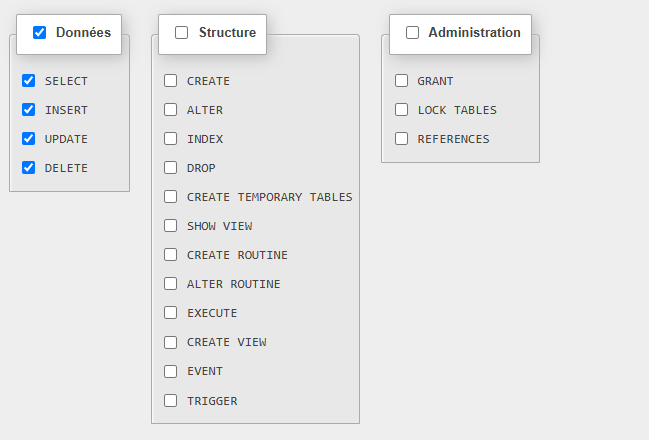


Figure 19 Droits dans la db

## Développement

Dans cette section, les différentes parties du site seront présentée, expliquée et comparée aux maquettes.

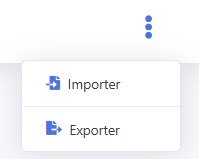
### Structure Globale



Figure 20 Réalisation - Structure Globale

Le titre et le bouton d’options pour importer/exporter sont bien disposé dans le header.  
  
La navigation contient les deux pages principales ainsi que le logo.

Et finalement le pied-de page contient comme prévu le logo african puzzle ainsi que le slogan.

Lorsque l’on clique sur le bouton d’options en haut à droite, un menu déroulant s’ouvre avec les options importer et exporter.

La zone vide au milieu sera remplie par des cards qui accueilleront les différents contenus du site comme les tables, les formulaires, etc.

### Page du catalogue

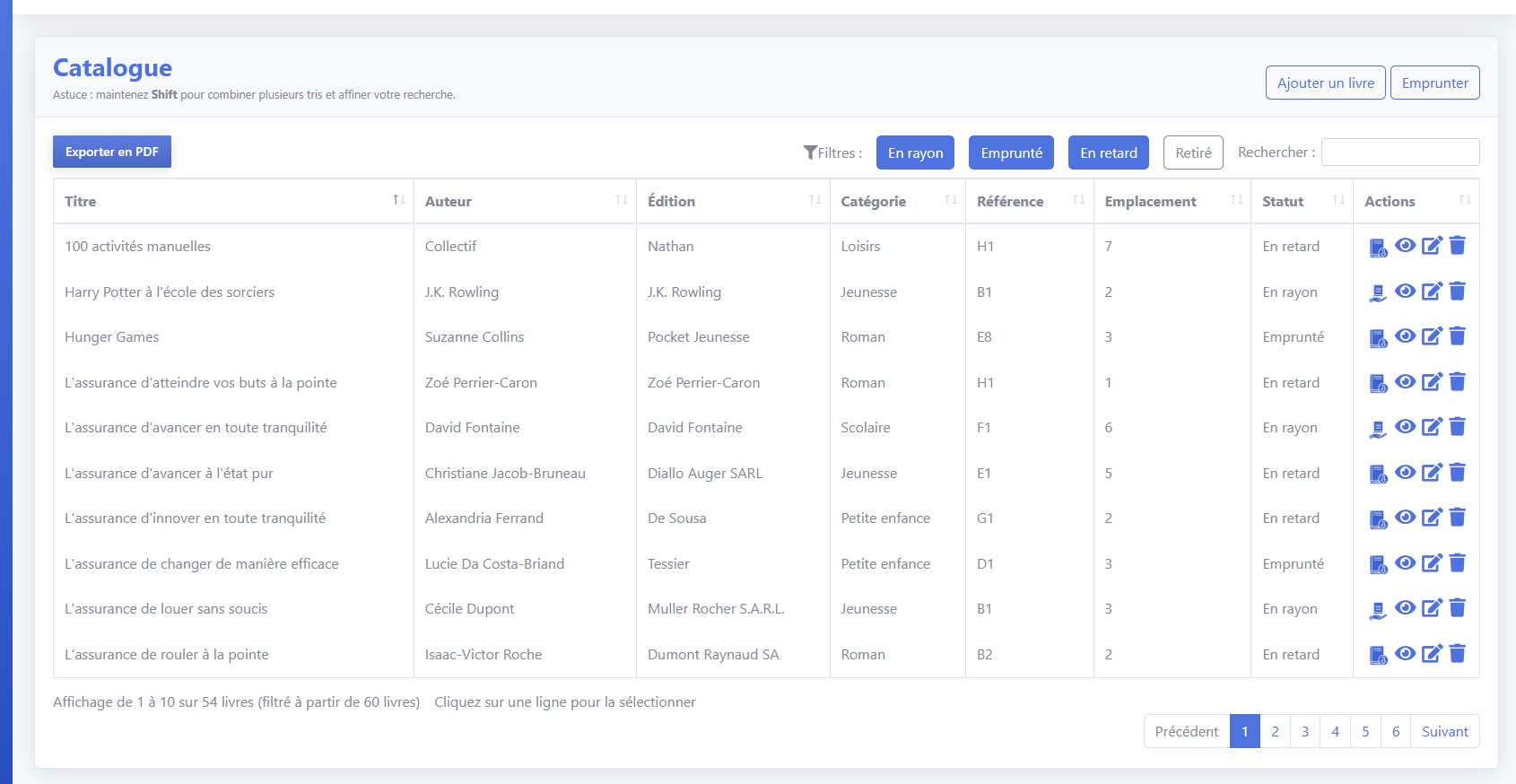
Afin d’améliorer l’interface de gestion du catalogue, il a été choisi d’utiliser la bibliothèque JavaScript « Datatables » qui a permis de gérer de nombreuses fonctionnalités.  


Figure 21 Réalisation – Catalogue

#### Fonctionnalités principales

* **Affichage dynamique** des données envoyées par le contrôleur grâce à une boucle foreach.
* **Recherche instantanée** par mots clés sur l’ensemble des colonnes.
* **Tri des colonnes** (croissant/décroissant) et possibilité de les combiner avec shift comme indiqué dans « Astuce ».
* **Pagination automatique** avec navigation intégrée.
* **Filtrage par statut** avec les boutons situé à côté de la barre de recherche.
* **Sélection de plusieurs livres** en cliquant sur l’élément du tableau qui permet d’effectuer un emprunt groupé.
* **Export PDF** de la table sans la colonne d’actions et uniquement avec les données actuellement dans la table (par exemple sur la capture d’écran les livres retirés ne seraient pas exportés).

Toutes ces fonctionnalités ont été intégrées en activant différents paramètres prévus par datatable sauf la fonction de filtrage par statut qu’il a fallu un peu personnaliser. Pour celle-ci, il a fallu injecter la recherche comme si elle était effectuée depuis la barre prévue à cet effet mais seulement sur la colonne 6 (Statut). Lorsque les filtres sont sélectionnés, ils sont assemblés avec un ou « | » pour que plusieurs filtres puissent être appliqués en même temps.

#### Colonne Actions

Différentes actions sont possibles depuis la dernière colonne de la table :



- Détail : permet d’accéder à la fiche complète d’un livre.



- Modifier : permet d’accéder au formulaire de modification d’un livre.



- Supprimer : permet de supprimer un livre de la db. (Un message de confirmation s’affiche pour s’assurer que le clic est volontaire.)



- Emprunter : permet d’accéder au formulaire d’emprunt d’un livre



- Rendre : permet d’accéder au formulaire de rendu d’un livre.

#### Export PDF

En cliquant sur le bouton exporter en PDF, un fichier Catalogue\_Bibliosolidaire.pdf sera téléchargé sur la machine du bibliothécaire.

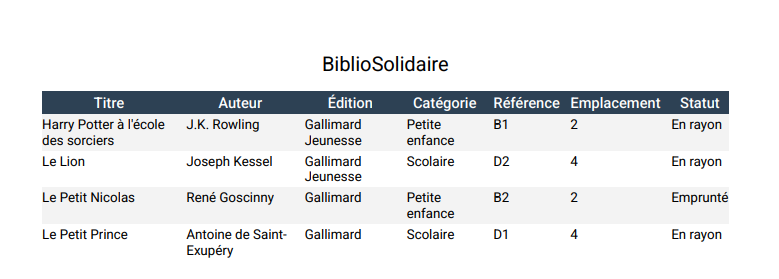


Figure 22 Réalisation – Extrait PDF du catalogue

La requête SQL qui récupère tous les livres va combiner le contenu de la table book avec celui des tables category et status pour directement avoir un tableau contenant dans les colonnes category et status les valeurs et non pas uniquement leurs id.

SELECT

book.title,

book.author,

book.edition,

category.name,

book.reference,

book.location,

status.name

FROM

book

JOIN

category ON book.fk\_category = category.id\_category

JOIN

status ON book.fk\_status = status.id\_status

ORDER BY   
 book.title ASC

### Formulaire d’ajout d’un livre

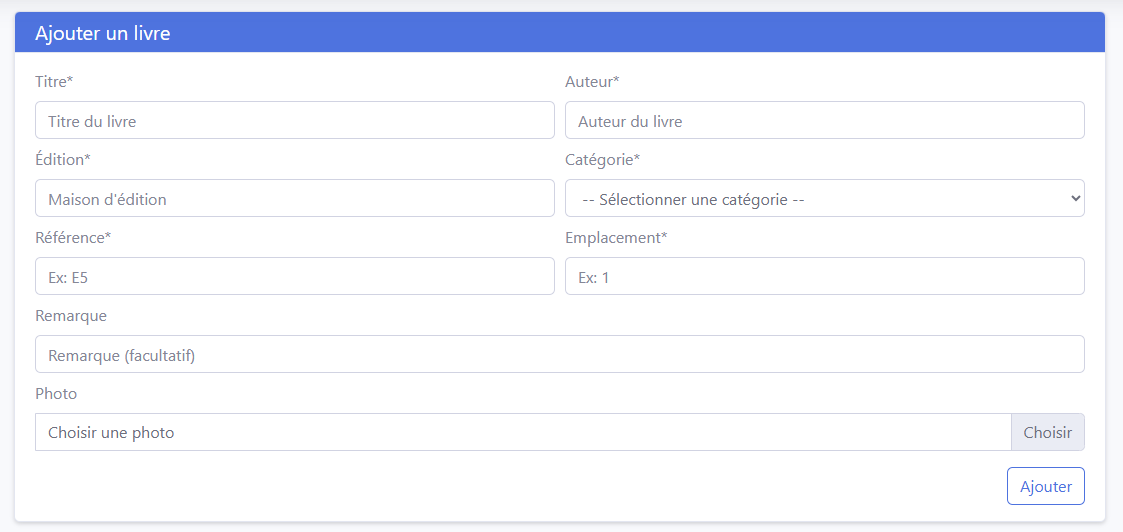


Figure 23 Réalisation – Formulaire ajout de livres

A l’origine le formulaire a été réalisé sur les bases de la maquette mais après discussion il a été réorganisé en deux colonnes pour les champs les plus courts et sur toute la largeur pour les plus conséquent. Cette modification permet au formulaire d’être plus compact alors que celui prévu lors de la conception nécessitait de scroller pour être rempli.

#### Validation

Pour qu’un livre soit ajouté il faut que tous les champs soient remplis sauf les champs Remarque et Photo qui eux sont optionnels. Les informations obligatoires sont indiquées par des astérisques, un contrôle en javascript est effectué à la soumission et des messages d’erreurs s’affichent si besoin.



Figure 24 Réalisation - Message d'erreur pour le champ Titre

#### Photo

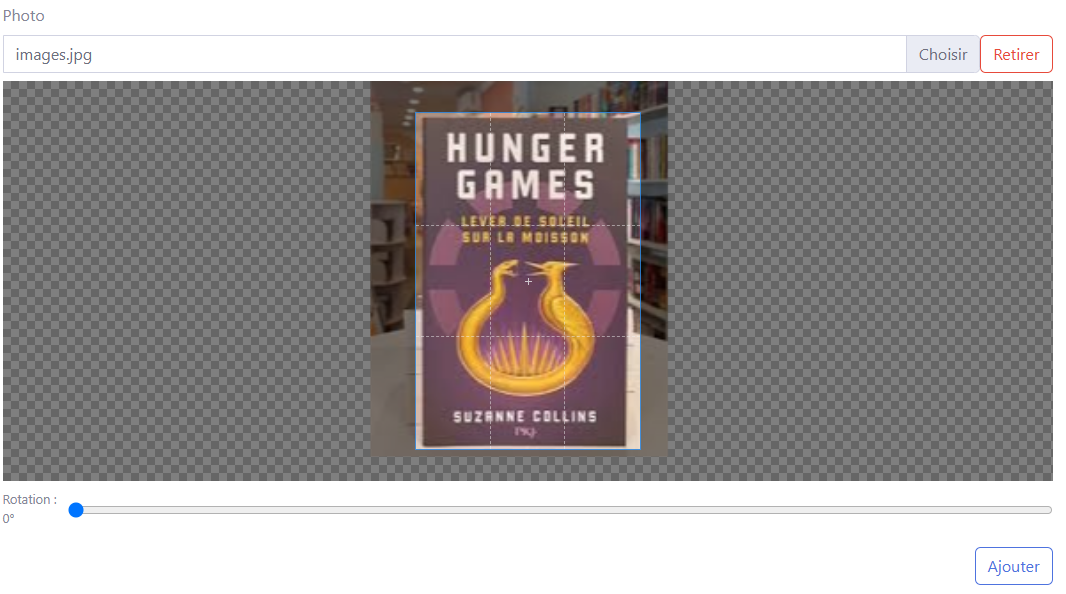


Figure 25 - Réalisation - Cropper photo de livre

Lorsqu’une image est sélectionnée dans le champ photo, elle sera affichée dans un cropper pour permettre au bibliothécaire d’ajuster sa photo dans le cadre en appliquant des rotations, en zoomant avec la molette et en déplaçant l’image avec la souris. Le cadre est fixe pour imposer la forme d’image autorisée et les formats attendus sont .png, .jpg, .jpeg, .webp et .gif. Toutes ces fonctionnalités ont été mise en place à l’aide de la bibliothèque javascript « Cropper.js ».  
  
Différents paramètre de l’objet Cropper ont été définis pour permettre cette modification. Voici les principaux :

* dragMode: 'move' // Permet de déplacer l’image et empêche la création d’un nouveau cadre.
* cropBoxMovable: false, // L’utilisateur ne peut pas déplacer le cadre.
* cropBoxResizable: false, // L’utilisateur ne peut pas redimensionner le cadre.
* zoomable: true, // Active le zoom avec la molette.
* rotatable: true, // Permet la rotation de l’image.

Les images sont stockées dans un dossier du site et leur nom est enregistré dans la table book pour pouvoir les retrouver.

#### Modal de confirmation



Figure 26 - Réalisation - Modal de confirmation création de livre

Si le formulaire est correctement rempli, une modal de confirmation s’ouvre avec un message indiquant au bibliothécaire que le livre a bien été ajouté. Toutes les actions effectuées sur le site sont confirmées par cette même modal en changeant juste le message. C’est une simple modal Bootstrap dans laquelle j’affiche un texte en fonction de la situation grâce à des sessions.

### Formulaire de modification de livres

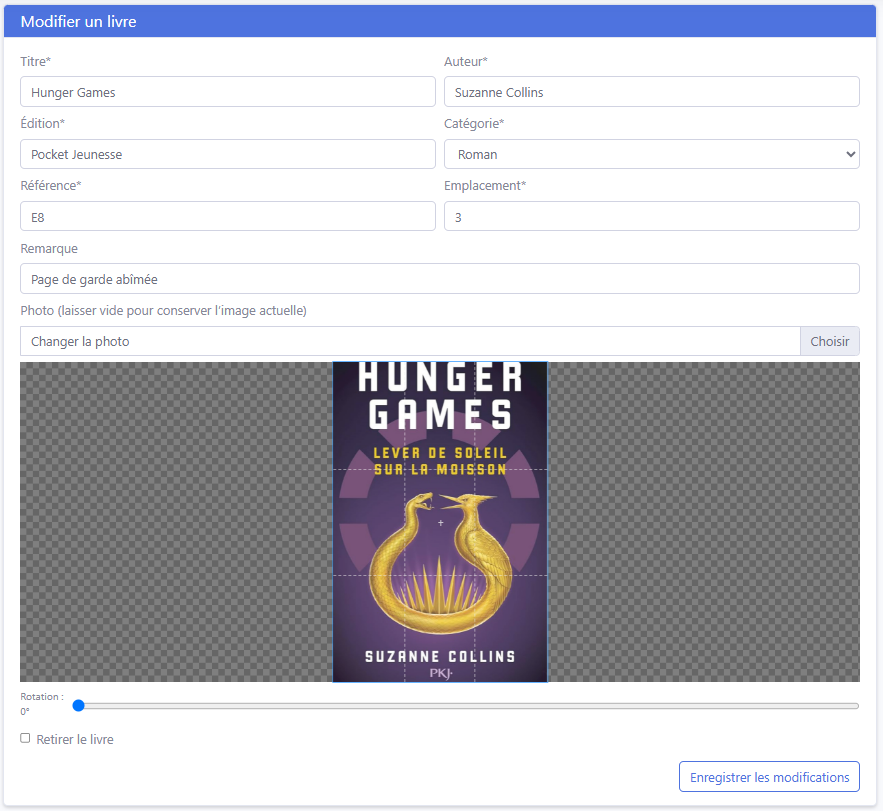


Figure 27 Réalisation - Modification des livres

Ce formulaire identique a celui de création a cependant été légèrement modifié par rapport à ce qui été prévu dans la conception déjà au niveau de sa structure comme expliqué précédemment mais aussi car un champ a été retiré parce qu’après réflexion il est risqué de laisser l’utilisateur modifier le statut depuis la modification car si le faisait ça pourrait par exemple fermer un prêt sans ouvrir le formulaire de rendu. Ce champ a donc été retiré lors de la réalisation pour ne pas avoir à gérer l’ouverture d’un autre formulaire depuis celui-ci. Une case à cocher a aussi été ajoutée pour permettre de retirer un livre des rayons sans le supprimer définitivement et sans qu’il disparaisse de l’historique des élèves.

### Page détail d’un livre

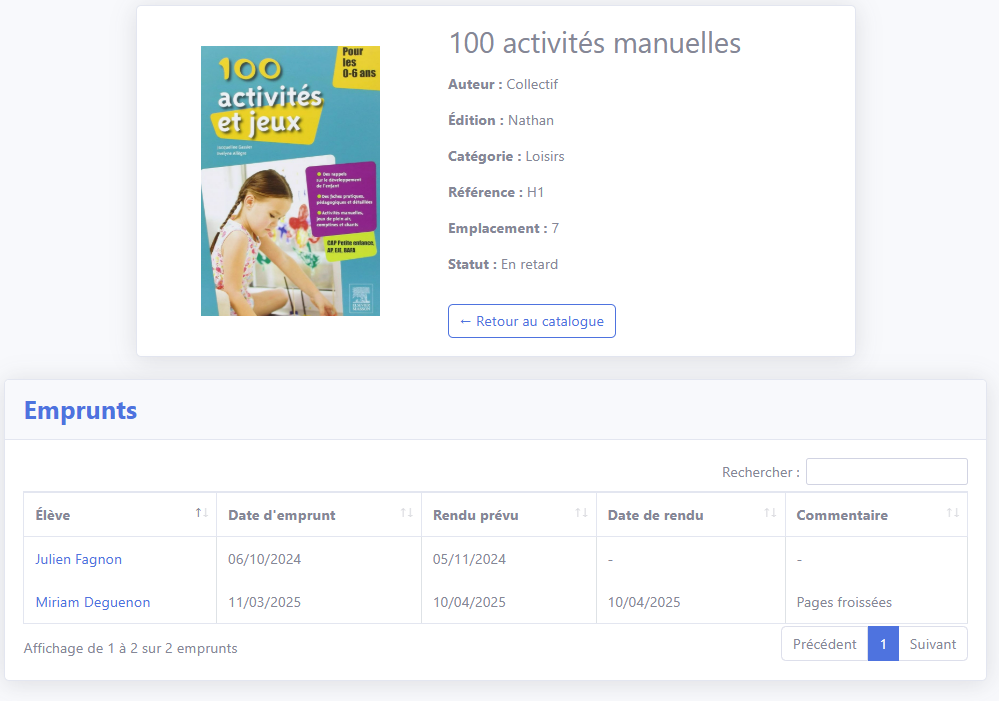


Figure 28 - Réalisation - Page détail d'un livre

La première partie de cette page affiche toutes les informations du livre, y compris sa photo. Un bouton est présent pour retourner au catalogue.

La deuxième partie affiche l’historique emprunts pour ce livre ainsi que les dates de l’emprunt, du rendu prévu et un commentaire sur l’état du livre si besoin. Cette table peut par exemple être utile si le bibliothécaire veut savoir qui a emprunté ce livre et peu cliquer sur le nom de l’élève pour accéder à sa fiche.

### Liste des élèves

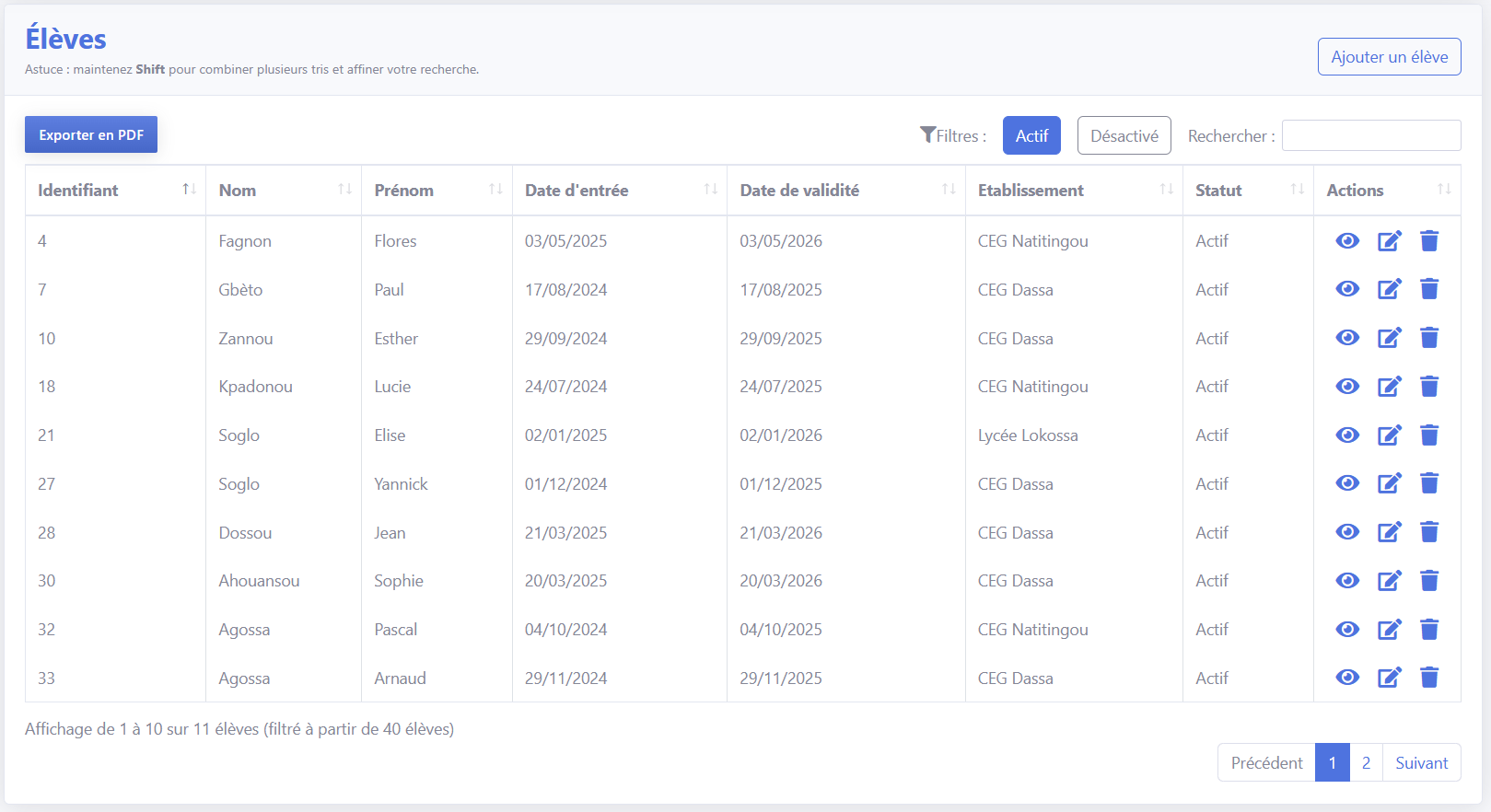


Figure 29 Réalisation - Liste des élèves

La page liste des élèves a la même structure que celle du catalogue en utilisant également la bibliothèque DataTables pour gérer l’affichage des données et d’autres fonctionnalités.

#### Fonctionnalités principales

* **Affichage** des données
* **Recherche instantanée** par mots clés sur toutes les colonnes.
* **Tri des colonnes** (croissant/décroissant) et possibilité de les combiner avec shift comme indiqué dans « Astuce ».
* **Pagination automatique** avec navigation intégrée.
* **Filtrage par statut** avec les boutons situé à côté de la barre de recherche.
* **Export PDF** de la table sans la colonne d’actions et uniquement avec les données actuellement dans la table (par exemple sur la capture d’écran les élèves désactivés ne seraient pas exportés).

Le fonctionnement du filtrage par statut est le même que celui dans le catalogue à la seule différence que les statuts ne proviennent pas de la db mais sont déduis grâce à la date de validité du compte et directement affiché dans la vue.

Les 3 actions possible dans cette table sont trois actions aussi présente dans le catalogue. Le bouton détail permet d’accéder à la fiche complète de l’élève, modifier permet d’accéder au formulaire de modification et supprimer affiche un message de confirmation avant de supprimer définitivement l’élève.

L’export au format PDF de la liste n’était pas demandé mais ça ne coûte rien de le mettre alors il a été ajouté sur cette liste aussi.

Un fichier Liste\_Eleves\_BiblioSolidaire.pdf est téléchargé sur le PC du bibliothécaire lorsqu’il clique sur le bouton « Exporter en PDF ».

### Formulaire d’ajout d’un élève

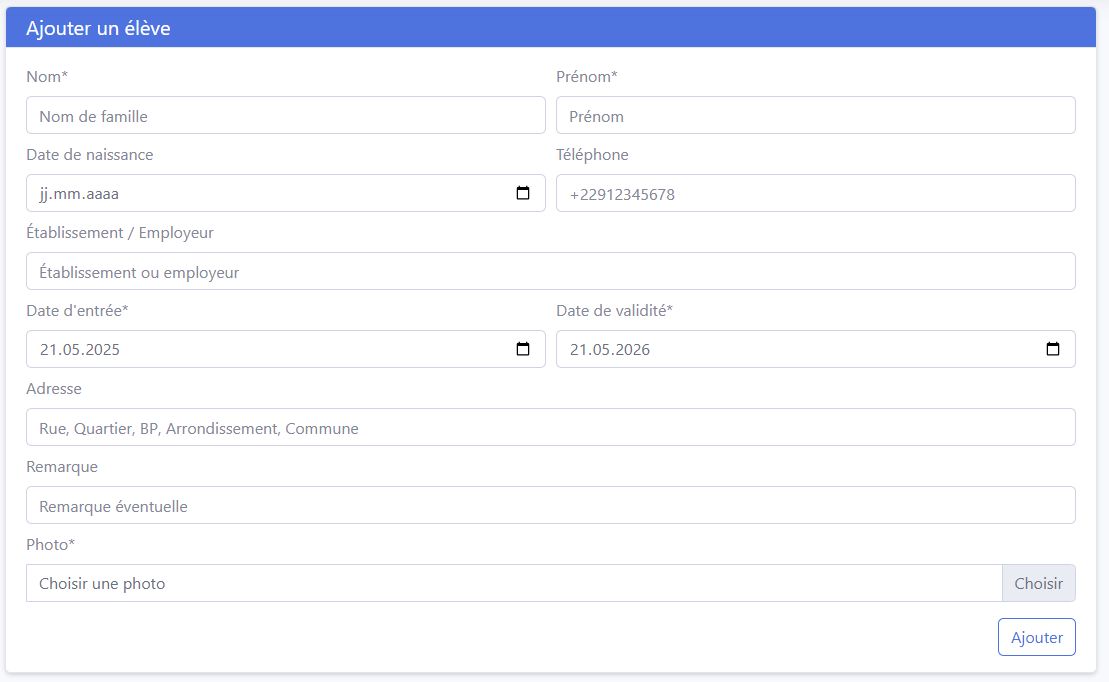


Figure 30 Réalisation – Formulaire ajout d'un élève

Formulaire un peu plus long que celui des livres mais les seuls champs obligatoires son nom, prénom, date d’entrée, date de validité et la photo (indiqué par les astérisques). Le champ date d’entrée est par défaut rempli à la date du jour et la date de validité est prévue un an plus tard par défaut.

Comme les formulaires précédents sa structure a été revue en suivant la logique de deux colonnes pour les champs courts et la largeur pleine pour les plus long.

Les images sont stockées de la même manière que pour les livres mais dans autre dossier et l’on peut aussi les modifier depuis le formulaire avec Cropper.js comme décrit précédemment.

### Formulaire de modification d’un élève

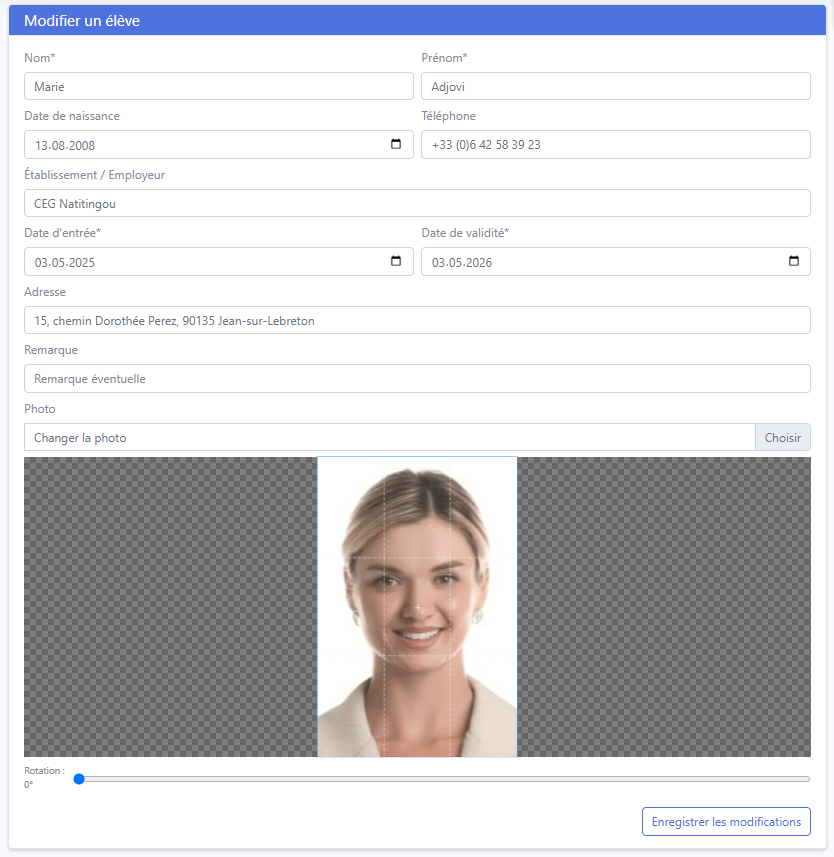


Figure 31 Réalisation – Formulaire modification d'un élève

Le formulaire est identique à celui de l’ajout et les informations sont simplement récupéré dans la base de données grâce à son identifiant pour préremplir le formulaire. Lorsque l’image est modifiée une fonction va supprimer l’ancienne dans le dossier de stockage pour ne pas alourdir l’application avec des fichiers inutiles.

### Page détail d’un élève

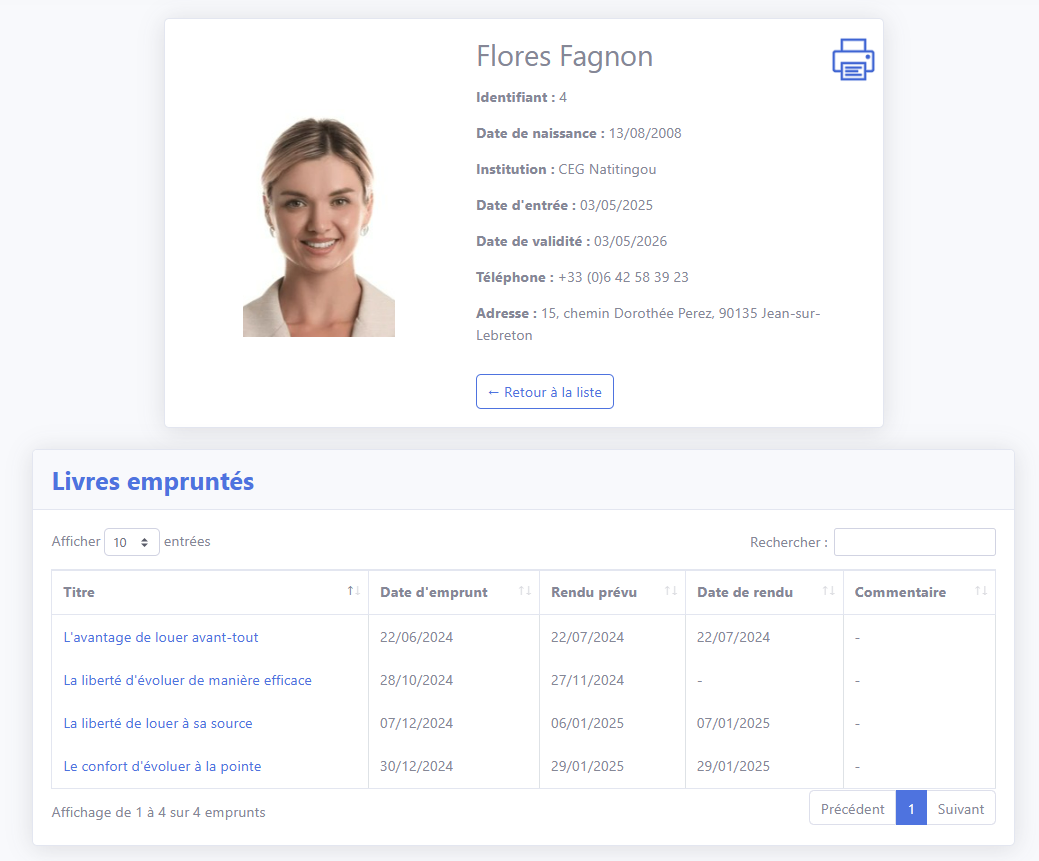


Figure 32 Réalisation - Page détail d'un élève

La page détail d’un élève contient sa fiche avec toutes ses informations affichées comme pour le livre. Le bouton de retour vers la liste est aussi présent mais un bouton supplémentaire a été ajouté en haut à droite de la card qui permet d’imprimer cette fiche.

En dessous comme pour les livres se trouve une table listant tous les livres que l’utilisateur a emprunté. Les titres sont cliquables pour atteindre la page détail de l’ouvrage.

Grâce à cette table, le bibliothécaire pourra par exemple décider de ne pas renouveler un compte parce qu’il pourra voir que l’élève a rendu de nombreux livre dans un piteux état.

#### Impression d’une fiche élève

Lorsqu’un compte est créé le bibliothécaire pourra se rendre sur la page détail du nouvel élève pour imprimer sa fiche. Une fois imprimée est plastifiée l’élève devra se présenter avec à chaque fois qu’il voudra emprunter un livre.

Pour effectuer cette impression la fonction onclick="window.print()" a été utilisée pour ouvrir le menu d’impression avec par défaut tout le contenu de la page. Comme l’objectif est de n’avoir qu’une carte avec les informations de l’élève j’ai ajouté du css dans @media print {} pour à la fois masquer tous les éléments indésirables comme le header, la navigation, les boutons, etc. mais aussi redimensionner à une taille de carte standard qui rentre dans un porte-monnaie un fois découpé et plastifié.

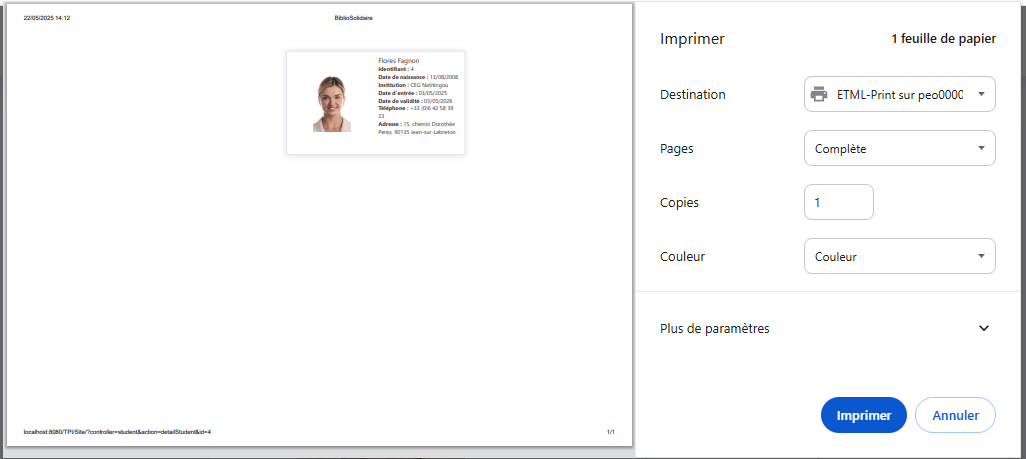


Figure 33 - Réalisation - Fenêtre d'impressions d'une fiche élève

### Formulaire d’emprunt d’un livre



Figure 34 Réalisation – Formulaire d’emprunt d'un livre

Le formulaire est rapide à remplir et ne contient que quelques restrictions. Il faut qu’un élève soit sélectionné, que les dates soient indiquées et que la date de retour soit après la date d’emprunt. Par défaut la date d’emprunt correspond à la date du jour et la date de rendu théorique est un mois plus tard. Le champ élève est un menu déroulant dans lequel nous pouvons chercher via l’id dont le bibliothécaire aura pris connaissance grâce à la carte de l’élève.

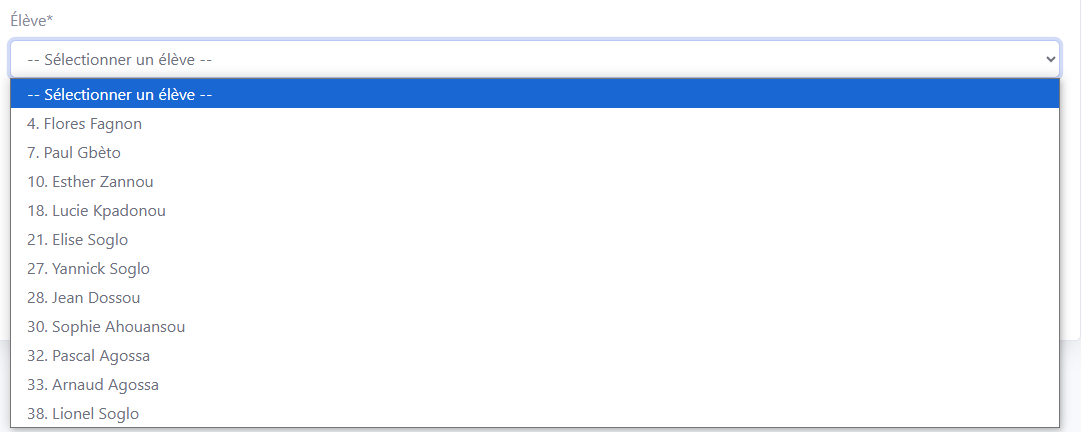


Figure 35 - Réalisation- Menu déroulant de sélection d'un élève

### Formulaire de rendu d’un livre

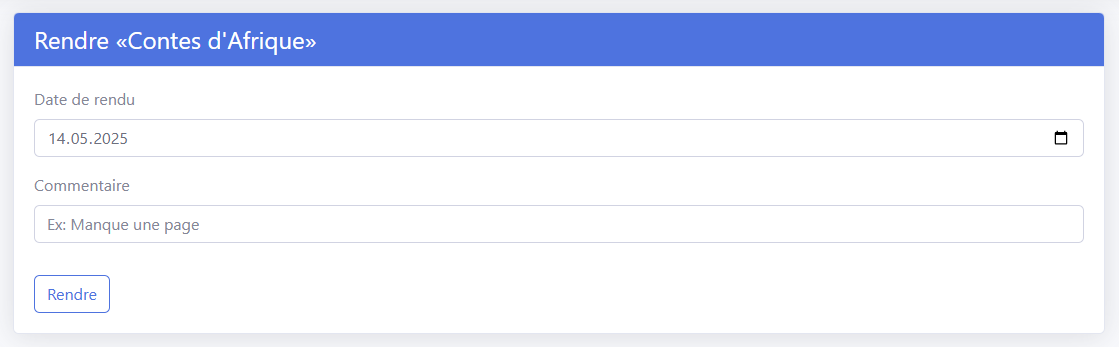


Figure 36 Réalisation – Formulaire de rendu d'un livre

Encore plus simple que le formulaire d’emprunt, ce formulaire contient uniquement la date de rendu qui est par défaut la date du jour ainsi qu’un champ commentaire optionnel. Ce commentaire sera visible dans l’historique des emprunts de ce livre et dans l’historique de l’utilisateur.

### Import/Export d’un livre

L’une des fonctionnalités les plus importantes était de permettre au bibliothécaire d’exporter les données pour les envoyer à un superviseur qui lui les importera dans la même application.

#### Fonction d’export

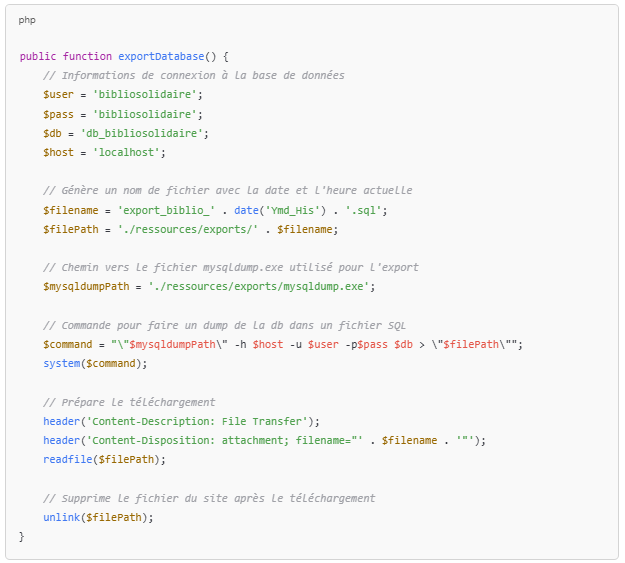


Figure 37 - Réalisation - Code de la fonction d’export

Plusieurs solutions ont été envisagées comme de faire un gros SELECT dans la base de données pour récupérer toutes les données dans un tableau et ensuite les insérer dans un fichier csv. Mais il est vrai que ça aurait été une perte de temps étant donné que depuis phpMyAdmin une fonction de dump de toute la base de données existe déjà. Pour effectuer le dump de la base de données il est nécessaire d’utiliser le fichier mysqldump.exe qui est dans les dossiers de Uwamp. Pour faciliter son accès il a été déplacé dans un des dossiers du site (./ressources/exports/).

Explication des différents paramètres de la commande d’export :

* "$mysqldumpPath" : chemin vers le fichier mysqldump.exe (outil d’export MySQL).
* -h $host : spécifie l’hôte de la base de données (localhost dans notre cas).
* -u $user : nom d’utilisateur pour se connecter à MySQL.
* -p$pass : mot de passe de connexion à MySQL
* $db : nom de la base de données à exporter.
* > "$filePath" : place l’export dans un fichier sql

#### Fonction d’import

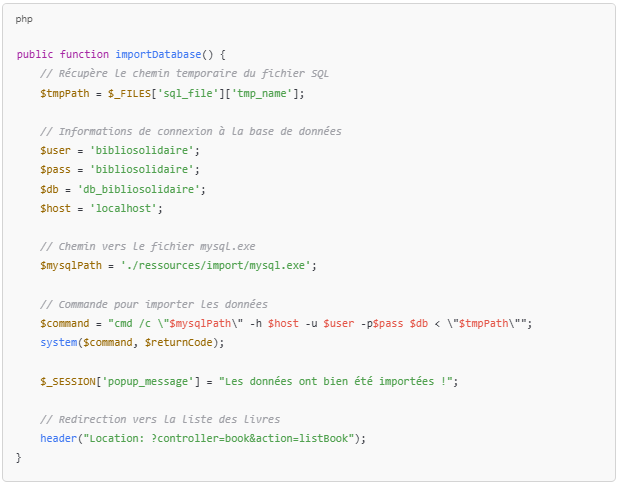


Figure 38 - Réalisation - Code de la fonction d’import

Pour importer le dump effectuer grâce à la fonction d’export il a fallu cette fois déplacer le fichier mysql.exe qui permet d’effectuer des commandes comme celle d’import dans la base de données.

Explication des différents paramètres de la commande d’export :

* cmd /c : exécute la commande dans le cmd
* "$mysqlPath" : chemin vers le fichier mysql.exe.
* -h $host : hôte de la base de données (localhost dans notre cas).
* -u $user : utilisateur MySQL.
* -p$pass : mot de passe MySQL.
* $db : nom de la base de données où il faut importer.
* < "$tmpPath" : récupère le contenu du fichier sql pour l’importer dans la db.

# Tests

## Plan des tests à effectuer

Ce chapitre décri un plan de test élaboré pour garantir une solution fonctionnelle qui respecte les exigences définies dans le cahier des charges.

### Environnement de test

Les tests seront effectués sur un serveur local (Uwamp) parce que c’est exactement ce qui sera utilisé par le bibliothécaire et dans un souci de compatibilité, chacun des tests sera effectué sur les navigateurs Chrome, Edge et Firefox.

### Scénario de test

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Fonctionnalité** | **Action/Test** | **Résultat attendu** |
| T001 |  |  |  |
| T002 |  |  |  |
| T003 |  |  |  |
| T004 |  |  |  |
| T005 |  |  |  |

## Résultats des tests

# Conclusion

# Annexes

## Résumé du rapport du TPI / version succincte de la documentation

## Webographie

## Glossaire

## Journal de travail

## Manuel d'Installation

## Manuel d'Utilisation