UFR Sciences et Techniques

UNIVERSITÉ DE ROUEN

Rapport du projet de langages Web

« Gestion d'une bibliothèque »

Auteurs :

Maya Abbad

Yohan Hery

Mai 2024

Table des matières

Présentation du projet

Ce projet a pour objectif de développer une application web de gestion d'une bibliothèque. Le système permet de gérer les informations sur les livres, les utilisateurs et les emprunts grâce à une interface intuitive et des fonctionnalités robustes. Les langages utilisés pour ce projet incluent HTML, CSS, JavaScript et PHP.

Le modèle de données inclut des entités pour les livres, les utilisateurs et les emprunts, permettant une gestion complète et efficace des ressources de la bibliothèque.

Méthodologie

1. Structure du projet :

Le projet est organisé de manière à séparer les préoccupations liées aux différents langages utilisés. Voici une vue d'ensemble de la structure du projet :

➢ PHP :

- **main.php** : Page principale de l'application.
- **panier.php** : Page de gestion du panier d'emprunts.
- **connect.php** : Gère les connexions des utilisateurs/administrateurs ainsi que l'inscription.
- **profil.php** : Page de profil de l'utilisateur ou de l'administrateur (selon le compte).
- **recherche.php :** Page de recherche de livres.
- **config.php** : Un fichier qui gère la connexion à la base de données.
- **affichage.php** : Page dédiée à l'interface de l'administrateur.
- **ajout_emprunt.php** et **ajout_panier.php** : Gèrent l'ajout aux tables d'emprunt et de panier.
- **error.php** : Page de traitement des erreurs.
- **footer.php** : Contient le code pour le pied de page commun.
- **form.php** : Gère le formulaire d'inscription.

➢ CSS :

- **cnx.css** : Feuille de style pour la page **connect.php.**
- **aff.css** : Feuille de style pour la page **affichage.php.**
- **main.css** : Feuille de style pour la page **main.php.**
- **panier.css** : Feuille de style pour la page **panier.php.**
- **profil.css** : Feuille de style pour la page **profil.php.**
- **rech.css** : Feuille de style pour la page **recherche.php**.
- **footer.css** : Feuille de style pour la page **footer.php.**
- **error.css** : Feuille de style pour la page **error.php.**

➢ JavaScript :

- **main.js** : Gère l'animation des carrousels de la page principale.
- D'autres balises **<script>** ont été ajoutées au fichiers PHP.

➢ Dossier imageanno :

Contient des images pour les couvertures des livres se trouvant dans la base de données.

➢ Dossier Source :

Contient des images/logos utilisés dans le CSS de différentes pages .

2. Fonctionnalités Principales :

- **Gestion des livres** : Les livres ont des attributs tels que le titre, l'auteur, l'année de parution, la catégorie, le stock, une photo de couverture ainsi qu'un ID. Les administrateurs peuvent ajouter, modifier ou supprimer des livres via une interface dédiée (affichage.php).

- **Gestion des comptes** : Les comptes utilisateurs/administrateurs ont un profil comprenant un identifiant ID, un nom, un prénom, une date de naissance, une adresse électronique, un mot de passe ainsi qu'un rôle. Les administrateurs peuvent ajouter, modifier, supprimer un compte ainsi que valider ou refuser des inscriptions faites a partir de la page de connexion. L'attributs rôle peut alors prendre une de ces valeurs :
  - **Admin** : Pour identifier un compte d'administrateur.
  - **User** : Pour un compte utilisateur.
  - **Acceptée** : Pour indiquer que l'inscription de ce compte a été approuvé par l'un des administrateurs.
  - **Refusée** : Pour indiquer que l'inscription de ce compte a été refusée.
  - **Waiting** : Pour une inscription en attente de traitement.

  De plus, un administrateur n'a pas la possibilité de supprimer un compte utilisateur si ce dernier a encore des livres non rendus.

- **Gestion des emprunts** : Les utilisateurs peuvent emprunter des livres, consulter leurs emprunts en cours, rajouter dans leurs paniers des livres à emprunter, rendre les livres ou renouveler leurs emprunts. Un emprunt est défini pour l'administrateur par le titre du livre emprunté, l'utilisateur, sa date de début, sa date limite, son nombre de prolongation ainsi qu'un statut. Le statut pouvant prendre une de ces trois valeurs :
  - **Emprunt** : Le livre est alors juste emprunté.
  - **Waiting** : Traitement d'une demande de prolongation en attente.
  - **ret_Waiting** : Retour du livre en cours de traitement.
  - **panier :** Le livre a été ajouté au panier.

  La validation d'un retour du livre entrainera la suppression de ce dernier dans la table d'emprunt de l'utilisateur. Pour la prolongation, un utilisateur n'a le droit de prolonger son emprunt qu'une seule fois et pour une durée de sept jours supplémentaires. Cette prolongation peut toutefois être refusée par l'administrateur et le message « Prolongation refusée » sera affiché pour l'utilisateur.

3. Explication détaillée de quelques traitements:

➢ La connexion :

Dans le fichier **config.php**, une fonction nommée **getConnection()** a été développée afin d'établir une connexion à une base de données MySQL en utilisant l'extension PDO. Cette fonction est ainsi appelée dans tous les pages qui requirent une connexion. En cas d'échec rencontré lors de cette connexion, une page html dédiée pour cette erreur sera affichée.

projetMYnotFinal > config.php

```php
<?php
function getConnection()
{
    // $host = 'inf-mysql.univ-rouen.fr:3306';
    // $dbname = 'heryyoh';
    // $username = 'heryyoh';
    // $password = '14032003';

    // $host = 'inf-mysql.univ-rouen.fr:3306';
    // $dbname = 'abbadmay';
    // $username = 'abbadmay';
    // $password = '26092004';

    $host = 'localhost';
    $dbname = 'maya';
    $username = 'root';
    $password = '';

    try {
        $pdo = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
        $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        return $pdo;
    } catch (PDOException $e) {
        // Afficher une page d'erreur stylisée en CSS
        echo "
        <!DOCTYPE html>
        <html lang='en'>
        <head>
            <meta charset='UTF-8'>
            <meta name='viewport' content='width=device-width, initial-scale=1.0'>
            <title>Erreur</title>
```

➢ Gestion de la sécurité :

Pour ajouter de la sécurité à notre site, nous avons utilisé différentes techniques pour gérer les problèmes suivants :

Injection SQL (SQL Injection) :

Technique : Utilisation de `htmlspecialchars` et de requêtes préparées

- `htmlspecialchars` : Cette fonction PHP convertit les caractères spéciaux en entités HTML. Par exemple, & devient `&`, < devient `<`, et ainsi de suite. Cela empêche les caractères spéciaux d'être interprétés comme du code par le navigateur ou la base de données, réduisant le risque d'injection SQL.
- Requêtes préparées : Elles permettent de séparer le code SQL de l'entrée utilisateur. Les valeurs des paramètres sont traitées comme des données et non comme des instructions SQL, empêchant ainsi les attaques d'injection SQL. Par exemple :

Cross-Site Scripting (XSS) :

Technique : Utilisation de `htmlspecialchars`

- `htmlspecialchars` : En plus de prévenir les injections SQL, cette fonction protège contre les attaques XSS en transformant les caractères spéciaux avant qu'ils ne soient affichés dans le navigateur. Par exemple, si un utilisateur entre `<script>`, cela sera affiché comme `&lt;script&gt;`, empêchant l'exécution du script malveillant.

Utilisation de la navigation via le code source pour générer des scripts sous contraintes sans y être affecté :

Technique : Encapsulation des scripts

- Encapsulation des scripts : Encapsuler le code JavaScript dans des fonctions auto-invoquées ou des modules pour limiter la portée des variables et des fonctions, réduisant ainsi les risques d'interférence.

Cross-Site Request Forgery (CSRF):

Technique : Utilisation de jetons CSRF (CSRF Tokens)

- Jetons CSRF : Générer un jeton unique pour chaque session utilisateur, puis inclure ce jeton dans chaque requête susceptible de modifier des données (comme les formulaires POST). Le serveur vérifie ce jeton pour s'assurer que la requête provient d'une source légitime.

Faille de sécurité des fichiers uploadés :

Technique : Validation et contrôle strict des fichiers uploadés

- Validation des fichiers : Vérifier l'extension des fichiers uploadés. Restreindre les types de fichiers acceptés pour éviter les fichiers exécutables.
- Stockage sécurisé : Stocker les fichiers uploadés dans un répertoire hors du répertoire web root pour empêcher l'accès direct via URL.

Contrôle d'accès insuffisant :

Technique : Implémentation d'un contrôle d'accès basé sur les rôles (RBAC)

- RBAC : Attribuer des rôles spécifiques aux utilisateurs et définir des permissions pour chaque rôle. Chaque action ou ressource est protégée par des vérifications de permissions.

Exposition des données sensibles :

Technique : Chiffrement et stockage sécurisé des données

- Chiffrement des données : Utiliser des algorithmes de chiffrement robustes pour protéger les données sensibles. Par exemple, pour les mots de passe, utiliser `password_hash` et `password_verify` en PHP.

  `$hashed_password = password_hash($password, PASSWORD_DEFAULT).`

En implémentant ces techniques, on a pu considérablement améliorer la sécurité de notre site web contre les vulnérabilités courantes.

```php
$email = htmlspecialchars($_POST['email']);
$psswd = $_POST['psswd'];
$pdo = getConnection();
$stmt = $pdo->prepare("SELECT * FROM compte WHERE email = :email");
$stmt->execute(['email' => $email]);
$user = $stmt->fetch(PDO::FETCH_ASSOC);
if ($user > 0 && password_verify($psswd, $user['psswd'])) {
```

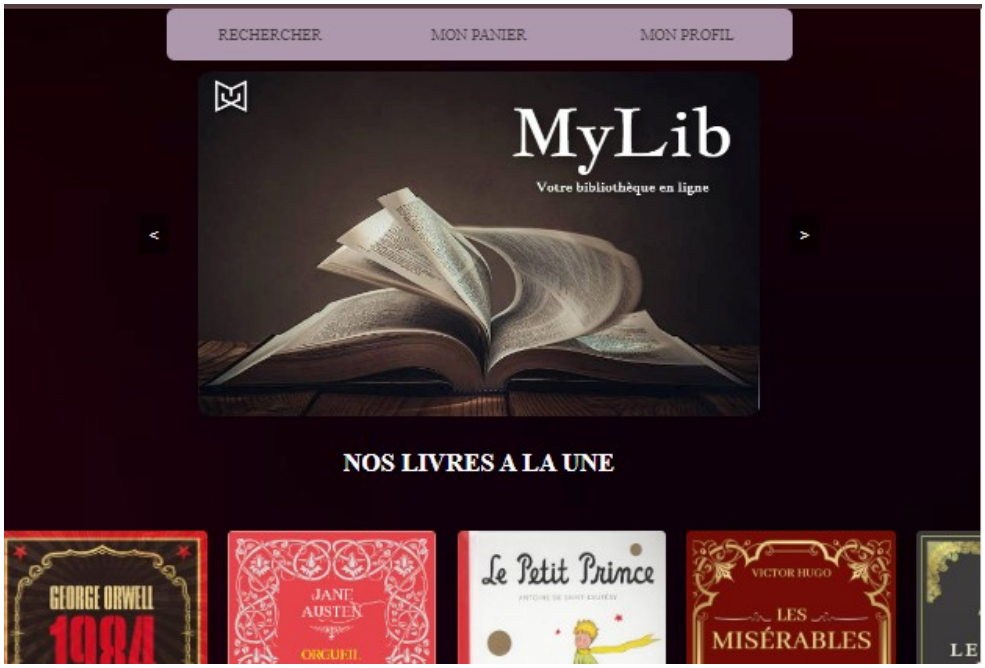
D’autres améliorations auraient pu être ajoutées, telles que :

- Content Security Policy (CSP) : Utiliser CSP pour spécifier quelles sources de scripts sont autorisées à être exécutées. Cela empêche les scripts non autorisés d'être injectés et exécutés.
- Transmission sécurisée : Utiliser HTTPS pour chiffrer les données en transit, empêchant les interceptions et les attaques de type man-in-the-middle.
- Validation des fichiers : Vérifier l'extension, le type MIME et la taille des fichiers uploadés. Restreindre les types de fichiers acceptés pour éviter les fichiers exécutables.

4. CSS et interfaces:

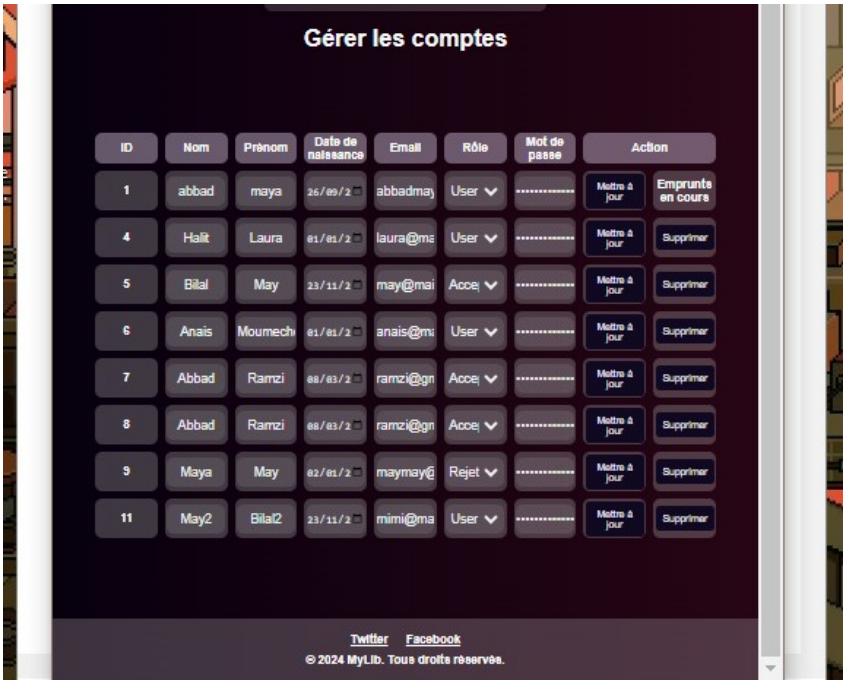
➢ Page principale « MyLib » :

Une interface bien accueillante avec plusieurs animations de carrousels.



➢ Page de gestion pour l'administrateur :

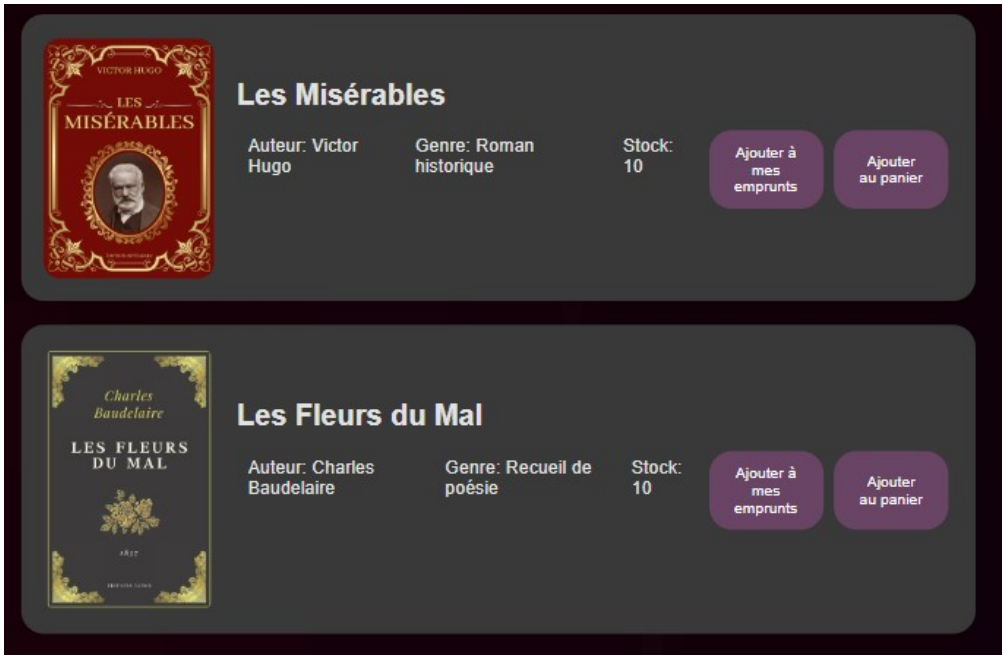
Une interface responsive et intuitive pour la gestion des comptes, emprunts et livres. Exemple sur la responsivité du tableau des comptes (la fenêtre est rétrécie en largeur au maximum) :



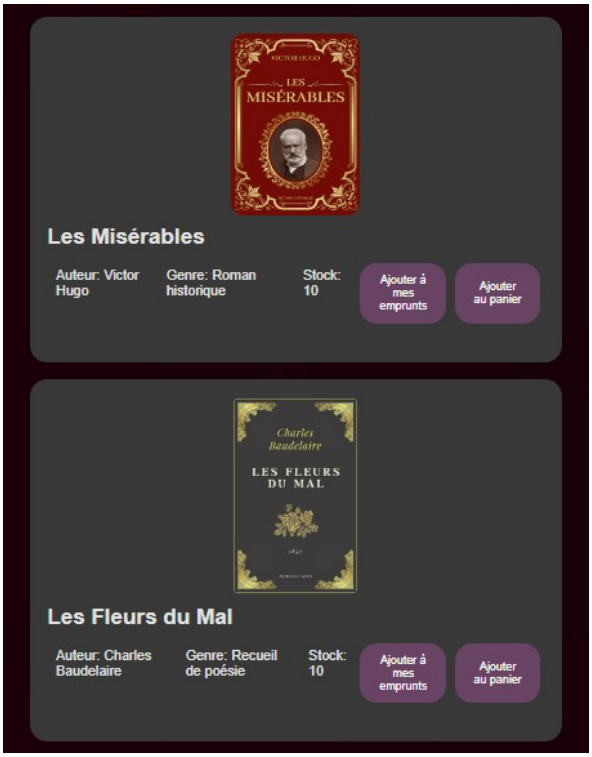
- Page de recherche :

Une interface responsive et intuitive pour rechercher des livres. Exemple sur la responsivité des livres et de leurs informations :

Fenêtre normale :



Fenêtre rétrécie :



Conclusion

La structure de ce projet web de gestion de bibliothèque est soigneusement conçue pour séparer les différentes responsabilités entre les fichiers CSS, JavaScript et PHP, facilitant ainsi la maintenance et l'évolution de l'application. Les fonctionnalités proposées permettent une gestion efficace des livres, des utilisateurs et des emprunts, tout en offrant une interface utilisateur intuitive et réactive. En utilisant les technologies enseignées en cours, ce projet constitue une application web robuste et complète pour la gestion d'une bibliothèque.