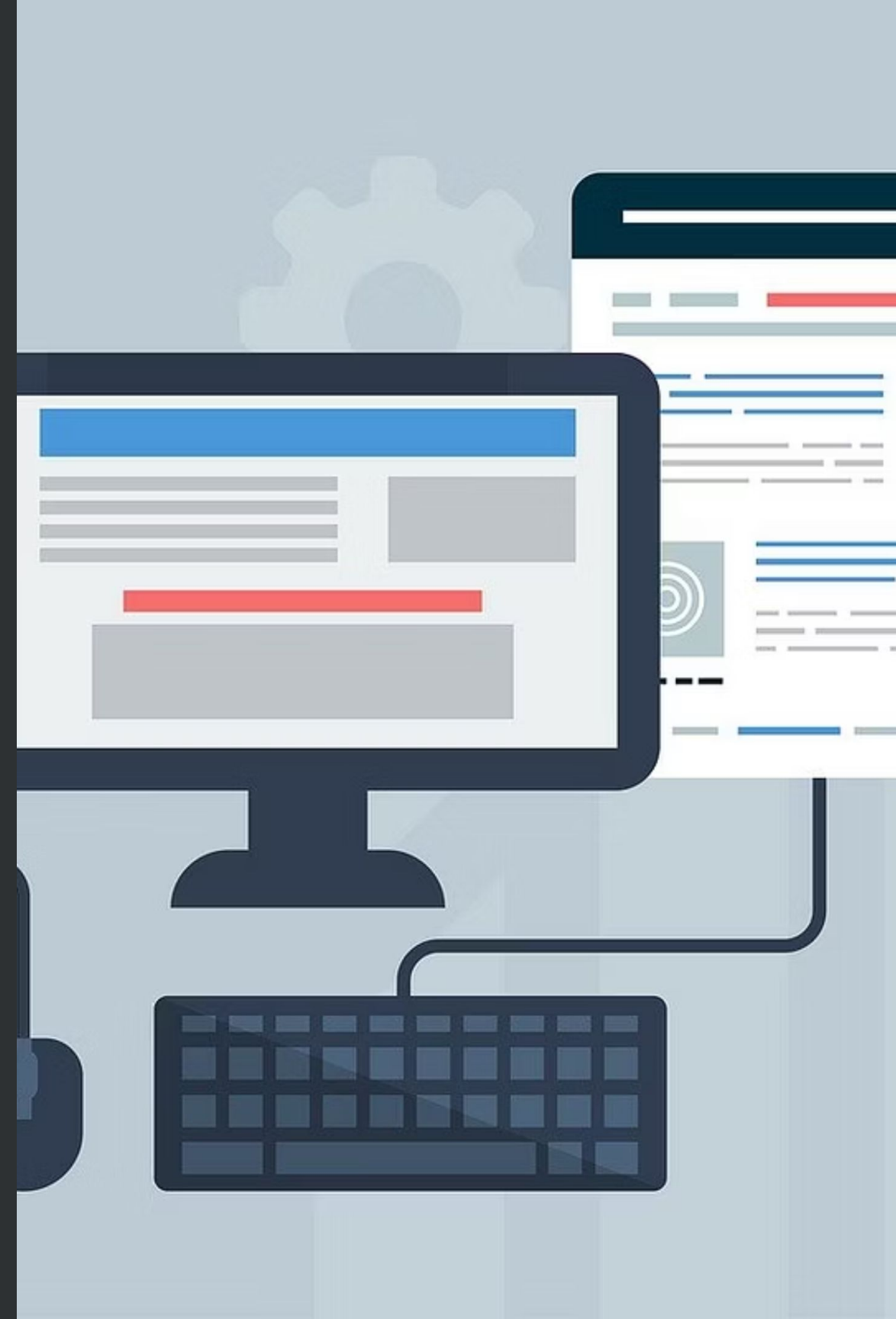


# BLOC 3 – Livecampus

Réalisé par Jules ARTAUD - MDV P2026.2

Présentation technique et démonstration fonctionnelle



# Sommaire

1 Contexte de l'application

2 Fonctionnalités de l'application

3 Modélisation de l'application

- Architecture
- Cas d'usage principaux

4 Les composants utilisés

5 Modélisation de la base de données

6 Dépendances Clés

7 Les enjeux de sécurité

8 Procédure de tests mise en place

9 L'accès au code source

10 Démonstration fonctionnelle

11 Bilan et perspective d'amélioration



# Contexte de l'application

La librairie "XYZ", existant depuis 10 ans, a identifié la nécessité de se moderniser en vue d'une meilleure gestion de ses utilisateurs et de son inventaire de livres. Cette modernisation est envisagée par le développement d'une application web.

# Les fonctionnalités de l'application

Déjà présent :

- Connexion / Inscription
- Création / Modification / Suppression / Détail des livres
- Déconnexion
- Visualisation de son profil

A développer :

Un système complet pour gérer les emprunts et les retours des livres de la librairie.

Chaque emprunt est limité à 30 jours. Après cette période, des rappels automatiques sont envoyés à l'utilisateur.

Les utilisateurs peuvent voir tous les livres qu'ils ont empruntés et la date de retour prévue.

Les utilisateurs doivent avoir la possibilité de signaler un retour de livre.

# Modélisation de l'application

## Architecture

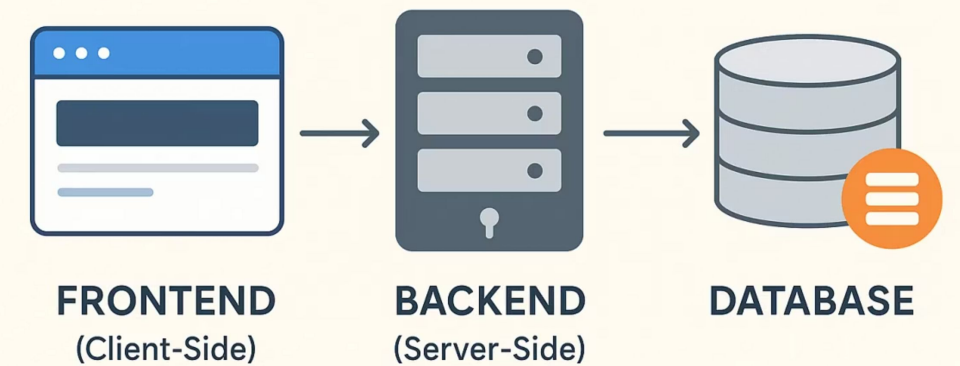
Frontend

Interface utilisateur moderne

Backend

API REST

Base de données relationnel



# Modélisation de l'application

## Cas d'usage principaux

01

---

### Authentification Utilisateur

Inscription, connexion sécurisée avec JWT et gestion des sessions

03

---

### Interface Dynamique

Mise à jour en temps réel et interactions utilisateur fluides

02

---

### Gestion des Livres et Emprunts

CRUD complet avec validation côté client et serveur

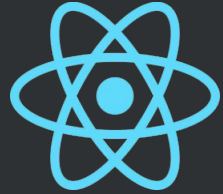
04

---

### Sécurité

Protection contre les vulnérabilités

# Les composants utilisés



## React Frontend

- Écosystème mature et communauté active
- Composants réutilisables et maintenables



## Express.js Backend

- Simplicité et rapidité
- Énorme écosystème
- Flexibilité



## MySQL Base de données

- Très répandu
- Relationnel

# Modélisation Base de Données

## Création de la table : EMPRUNTS

**id\_emprunt** : *clé primaire* de la table, de type `int` et auto-incrémentée. Elle identifie de manière unique chaque emprunt enregistré.

**id\_utilisateur** : *clé étrangère* (FK) référencée sur la table `UTILISATEURS(id)`. Ce champ indique quel utilisateur a effectué l'emprunt. La relation est de type **plusieurs emprunts pour un utilisateur**.

**id\_livre** : *clé étrangère* (FK) référencée sur la table `LIVRES(id)`. Ce champ indique quel livre a été emprunté. La relation est de type **plusieurs emprunts pour un même livre**.

**date\_emprunt** : de type `date`. Indique la date à laquelle l'utilisateur a emprunté le livre.

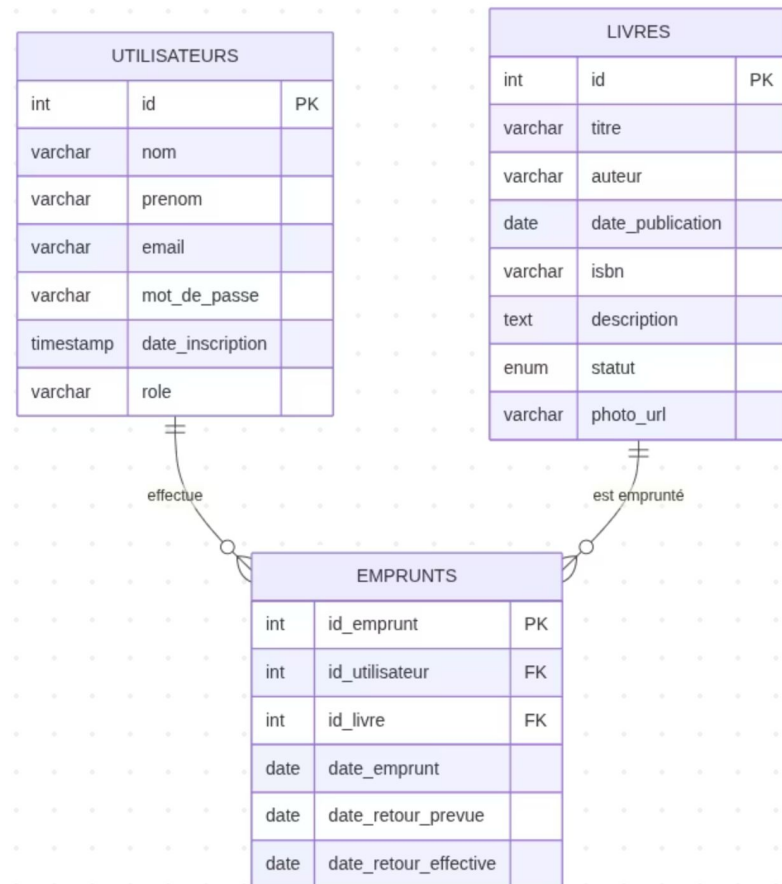
**date\_retour\_prevue** : de type `date`. Indique la date prévue pour le retour du livre.

**date\_retour\_effective** : de type `date`, peut être `NULL`. Indique la date réelle de retour du livre. Si le livre n'est pas encore retourné, ce champ reste vide.

## Relations :

Un utilisateur peut avoir plusieurs emprunts (1:N).

Un livre peut être emprunté plusieurs fois dans le temps (1:N).





# Dépendances Clés Backend

## Sécurité & Authentication

**bcrypt** pour hashage des mots de passe, **jsonwebtoken** pour l'authentification, **cors** pour la sécurité cross-origin, **csrf** pour protéger les formulaires contre les attaques CSRF (Cross-Site Request Forgery).



## Base de Données

**mysql2** pour la connexion MySQL optimisée, **dotenv** pour la gestion sécurisée des variables d'environnement



## Tests

**jest** pour la création et l'exécution de tests unitaires et d'intégration, **supertest** pour tester les endpoints HTTP de l'API et vérifier le comportement des routes.

## Rappels automatisés

**node-cron** pour planifier et exécuter des tâches récurrentes automatiquement (jobs cron) côté serveur, **node-mailer** pour envoyer des emails, comme des notifications ou des rappels

# Dépendances Clés Frontend



react

Bibliothèque principale pour construire l'interface utilisateur en composants réactifs.



react-dom

Permet d'afficher les composants React dans le navigateur.



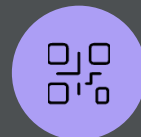
react-router /  
react-router-dom

Gère la navigation entre les pages de l'application sans rechargement.



vite

Outil moderne pour le développement et le build rapide des applications React.



eslint

Analyse le code pour détecter les erreurs et améliorer la qualité du code.



cypress

Permet de réaliser des tests E2E.

# Les enjeux de sécurité

Hachage des mots de passe.

Validation des données (client/serveur)

Protection contre les injections (SQL/XSS)

CSRF – Cross-Site Request Forgery

Utilisation de JSON Web Tokens (JWT)

Gestion des permissions sur les routes

Utilisation des variables d'environnement.

Implémentation de Content Security Policy (CSP)

```
Running: emprunts.cy.js (1 of 1)

Gestion des emprunts
  ✓ se connecte et affiche la liste de ses emprunts (1208ms)

1 passing (3s)

(Results)

Tests:      1
Passing:    1
Failing:    0
Pending:    0
Skipped:    0
Screenshots: 0
Video:      false
Duration:   2 seconds
Spec Ran:   emprunts.cy.js

PASS tests/emprunts.test.js
Emprunts API
  ✓ refuse un emprunt de plus de 30 jours (41 ms)
  ✓ refuse un emprunt si le livre est indisponible (31 ms)
  ✓ accepte un emprunt valide (14 ms)
  ✓ retourne la liste des emprunts (12 ms)
  ✓ retourne un livre (16 ms)

Test Suites: 1 passed, 1 total
Tests:       5 passed, 5 total
Snapshots:   0 total
Time:        0.555 s
```

# Procédure de Tests Mise en Place

1

Tests d'Intégration (jest + supertest)

Supertest pour tester les endpoints API avec base de données de test

2

Tests End-to-End (cypress)

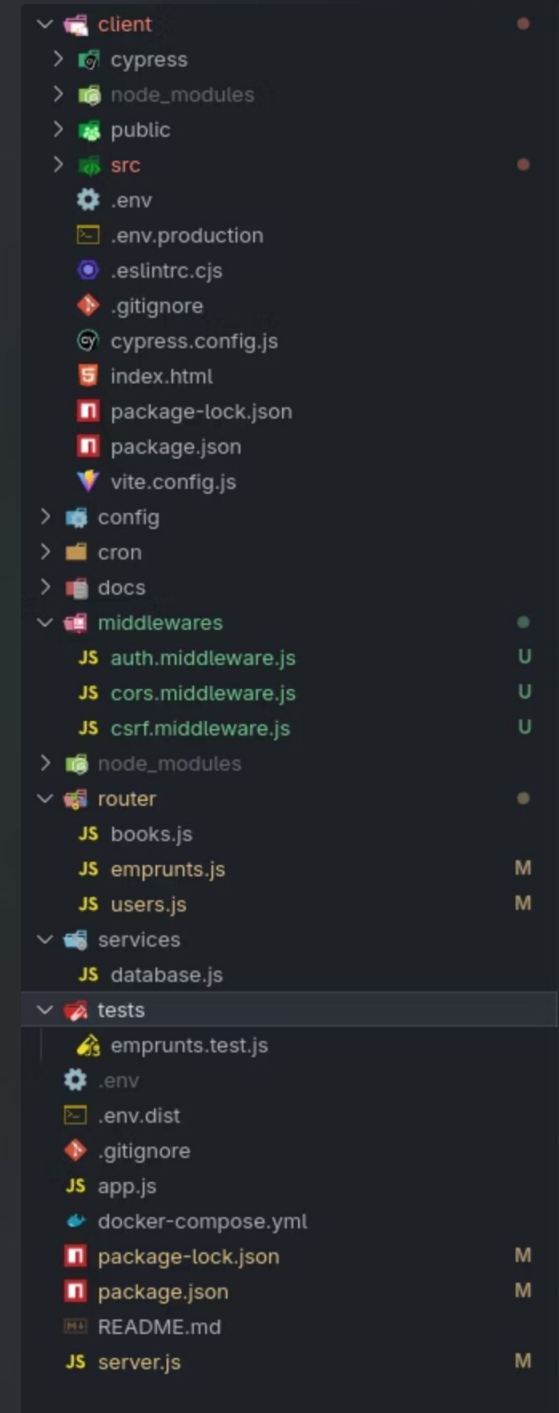
Validation complète des parcours utilisateur

# Code Source

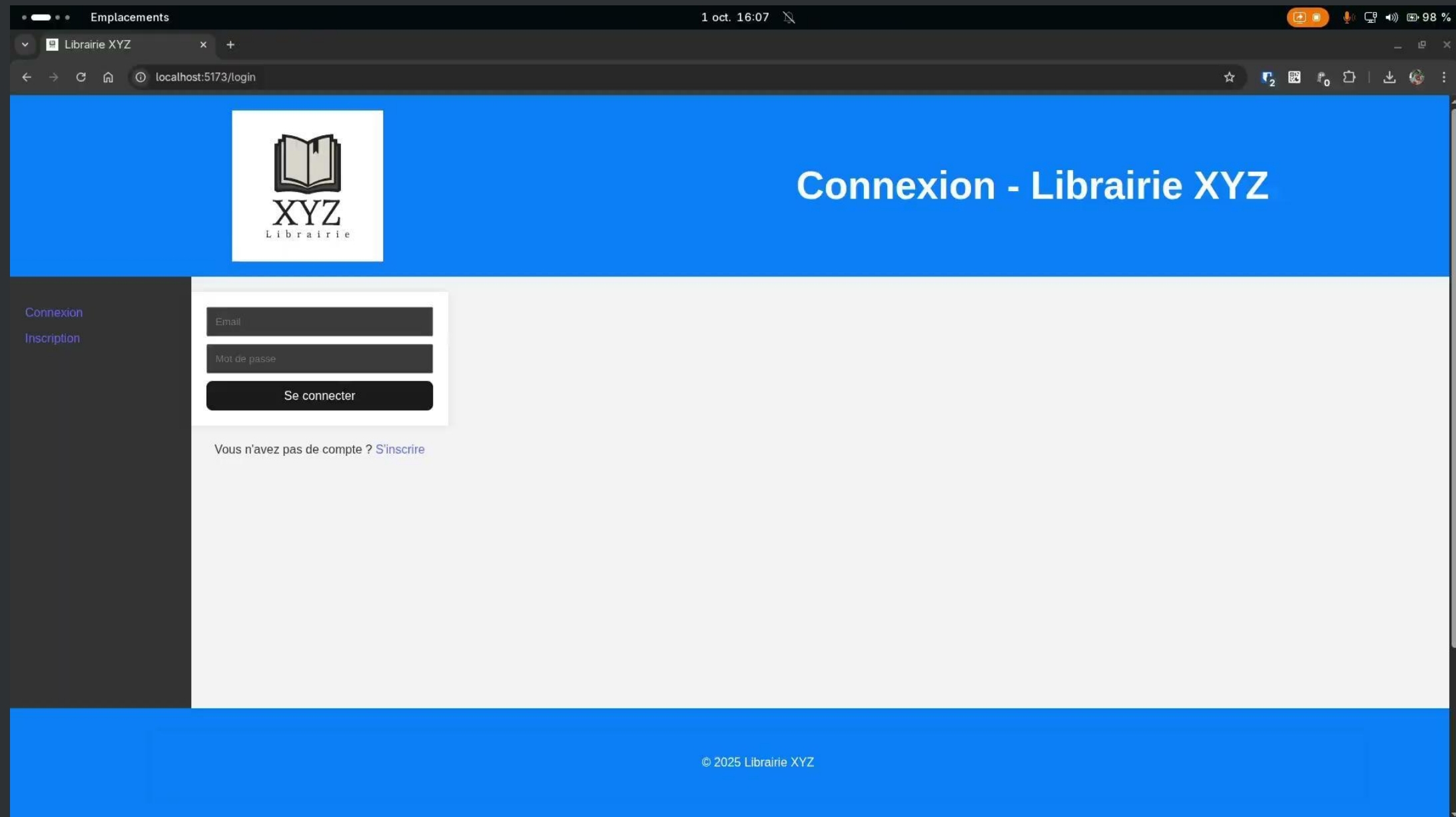
Accès au Code

Repository GitHub Public :

<https://github.com/julesartd/BC3SJ1-JAVASCRIPT>



# Démonstration fonctionnelle -1



# Démonstration fonctionnelle -2

Emplacements

1 oct. 16:08

98 %

Librairie XYZ

localhost:5173/books

accueil - Librairie XYZ

Bonjour john@smith.com

admin

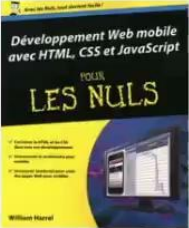
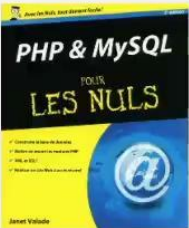
[Voir la liste des livres](#)

[Mon profil](#)

[Mes emprunts](#)

Déconnexion

Liste des Livres - Librairie XYZ

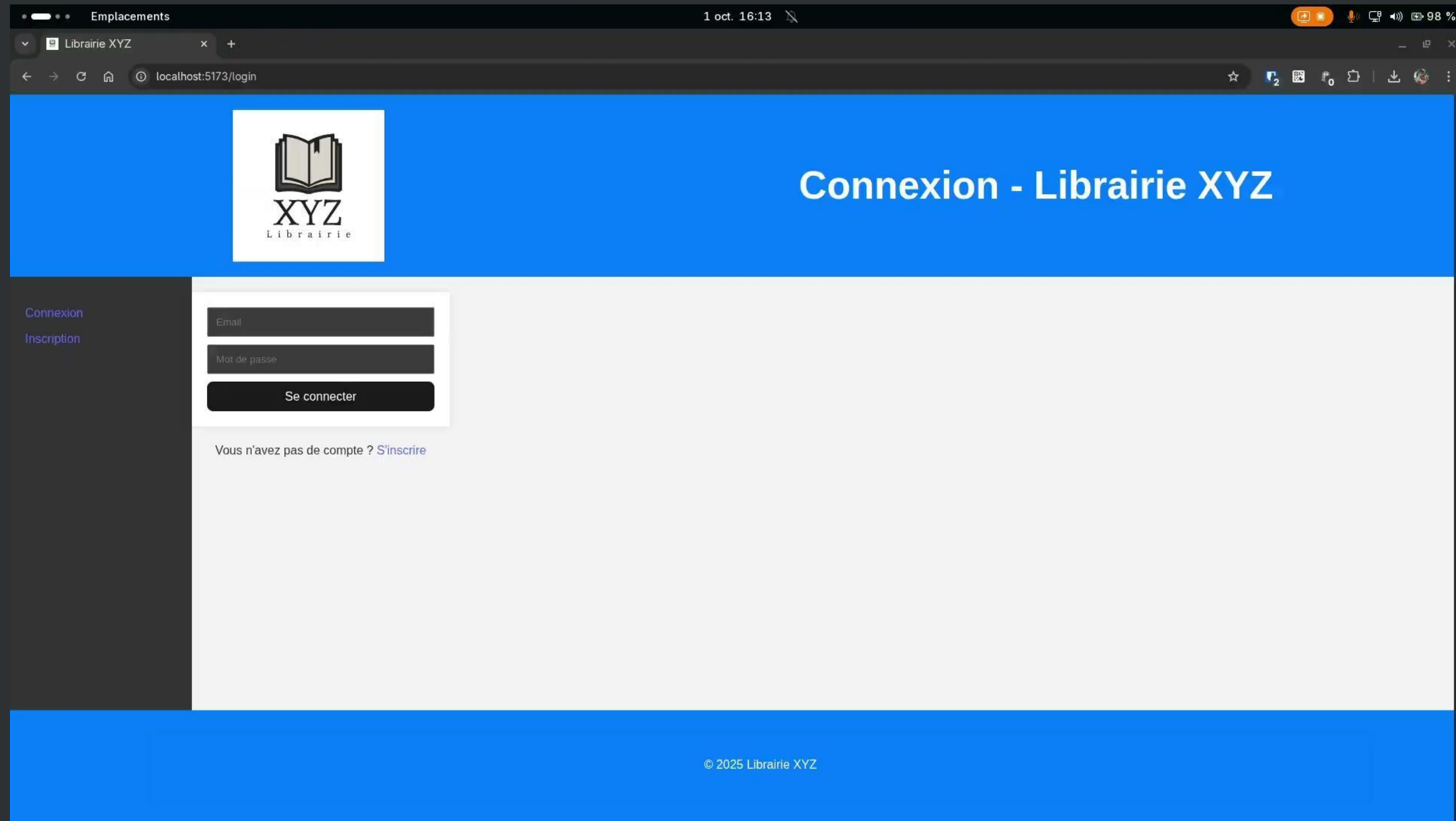
Image	Titre	Auteur	Date de publication	Statut	Détails	Action
	Developpement Web mobile avec HTML, CSS et JavaScript Pour les Nuls	William HARREL	2023-11-08T23:00:00.000Z	emprunté	<a href="#">Voir les détails</a>	-
	PHP et MySql pour les Nuls	Janet VALADE	2023-11-13T23:00:00.000Z	emprunté	<a href="#">Voir les détails</a>	-

Ajouter un livre

Retour à l'accueil

© 2025 Librairie XYZ

# Démonstration fonctionnelle -3





# Bilan & Perspectives d'Amélioration – 1



- Utiliser des variables d'environnement pour tous les secrets et identifiants sensibles et pour gérer plusieurs environnements
- Rendre la configuration CORS dynamique et adaptée à la production
- Ajouter le middleware CSRF sur toutes les routes qui modifient des données
- Valider toutes les entrées utilisateur côté client et côté serveur.
- Vérifier la protection des routes et des permissions pour chaque action.
- Tests automatisés plus complets

# Bilan & Perspectives d'Amélioration – 2



- Séparer en plusieurs fichiers les logiques du backend (models/controllers/views... MVC)
- UI / UX - Améliorer le style pour que ce soit intuitif pour les utilisateurs
- Interface mobile responsive avancée
- Analyse de code SonarQube
- Monitoring et Logs (Sentry, Grafana)

**Des questions ?**