LA FACULTÉ DES SCIENCES EL JADIDA

24/04/2025

RAPPORT De Mini Projet

Gestion de la bibliothèque

RÉALISÉ PAR:

JADOR YASSINE

ENCADRANT:

Mr.Mzili Toufik

SOMMAIRE

Indroduction

- Ce projet consiste en une application web/mobile permettant aux utilisateurs de gérer leurs emprunts de livres dans une bibliothèque. L'application offre une interface intuitive pour s'inscrire, se connecter, consulter les livres disponibles, emprunter des ouvrages et suivre ses prêts en cours.

A. Technologies

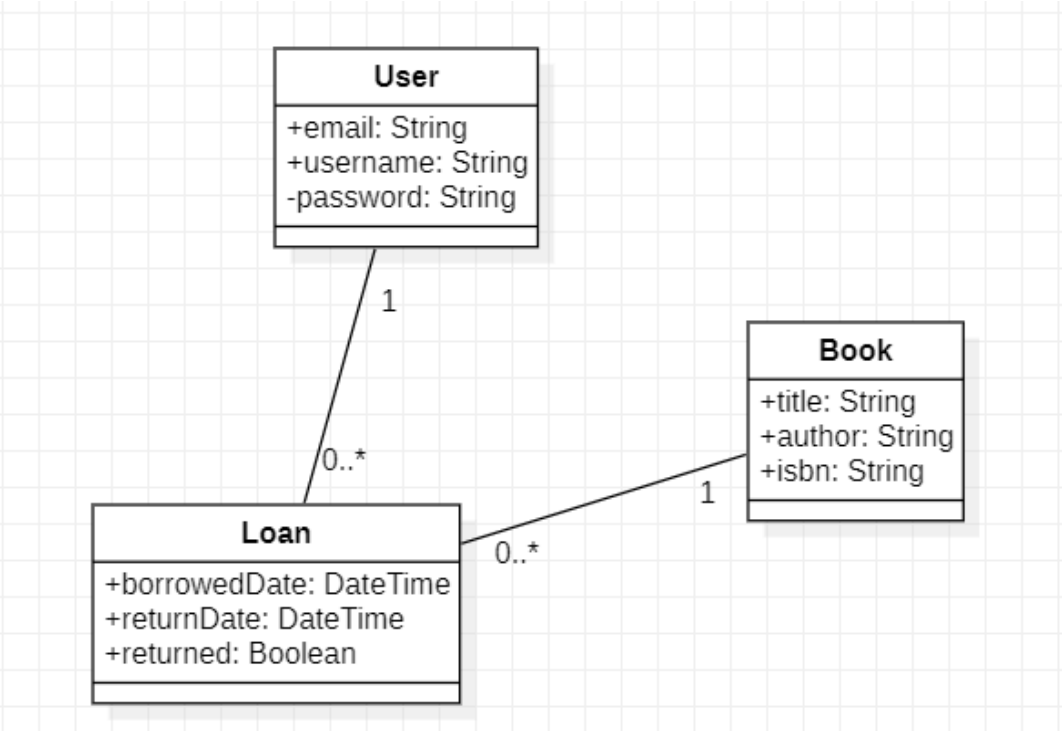
Technologies : React Native, Expo, NodeJS,ExpressJS, MongoDB
IDE : Visual studio code.
Logiciel de gestion de version : Git.
UML (conception) :StarUML.

Objectifs de Projet

- Simplifier la gestion des emprunts de livres.
- Permettre aux utilisateurs de s'inscrire et de se connecter via un système d'authentification sécurisé.
- Afficher une liste dynamique des livres disponibles avec possibilité d'emprunt.
- Fournir un tableau de bord pour visualiser les emprunts en cours.
- Utiliser une architecture client-serveur avec des technologies modernes (Node.js, React Native).

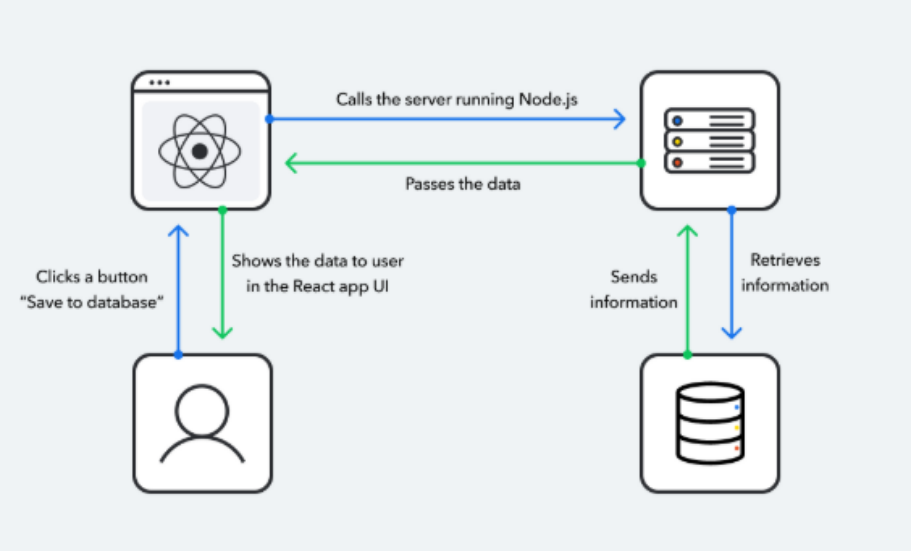
Conception et Architecture

- **Diagramme de Classe**



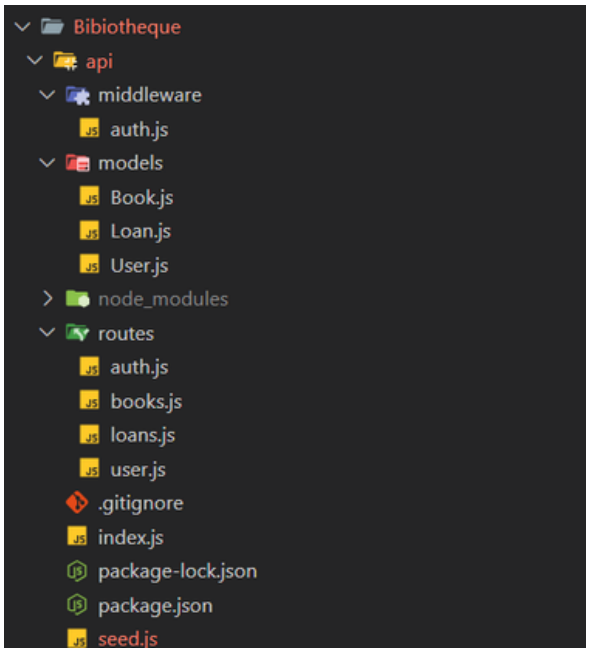
Ce modèle de données (Diagramme de Classe) permet de gérer les prêts de livres dans une application. Il comprend trois entités : **User** (utilisateur), **Book** (livre), et **Loan** (prêt). Chaque utilisateur peut emprunter plusieurs livres, et chaque livre peut être emprunté plusieurs fois. L'entité **Loan** enregistre les informations du prêt, comme les dates et le statut de retour. Ce schéma assure un suivi clair et structuré des emprunts entre utilisateurs et livres.

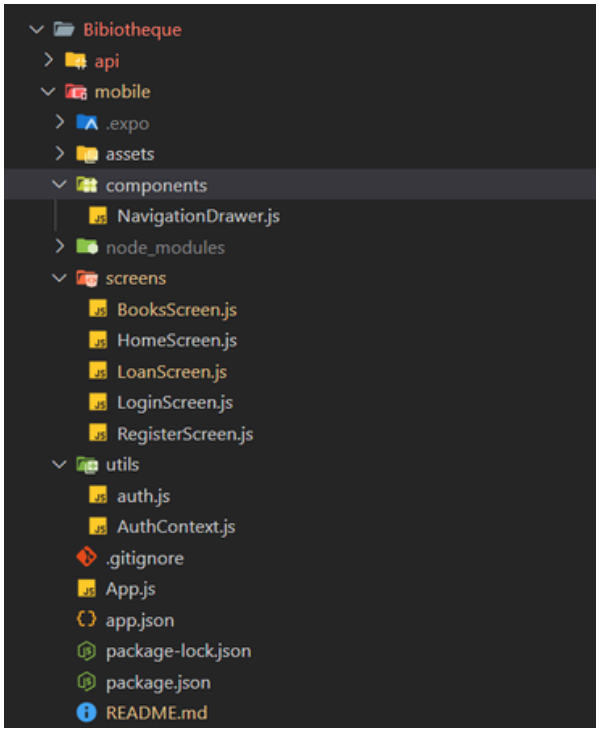
- **Architecture**



Utiliser **React Native** avec **Node.js** (Express) et **MongoDB** constitue une stack moderne et cohérente pour le développement d'applications mobiles complètes. Côté backend, Node.js avec Express offre un environnement rapide, non bloquant et léger pour construire des API RESTful performantes. MongoDB, en tant que base de données NoSQL, s'intègre naturellement avec JavaScript via les objets JSON, facilitant les échanges de données entre le frontend et le backend. Cette stack est idéale pour les projets agiles, évolutifs, et en temps réel comme les apps sociales, les marketplaces ou les services géolocalisés, grâce à sa flexibilité, sa rapidité de développement et son écosystème riche.

A. Structure du Projet



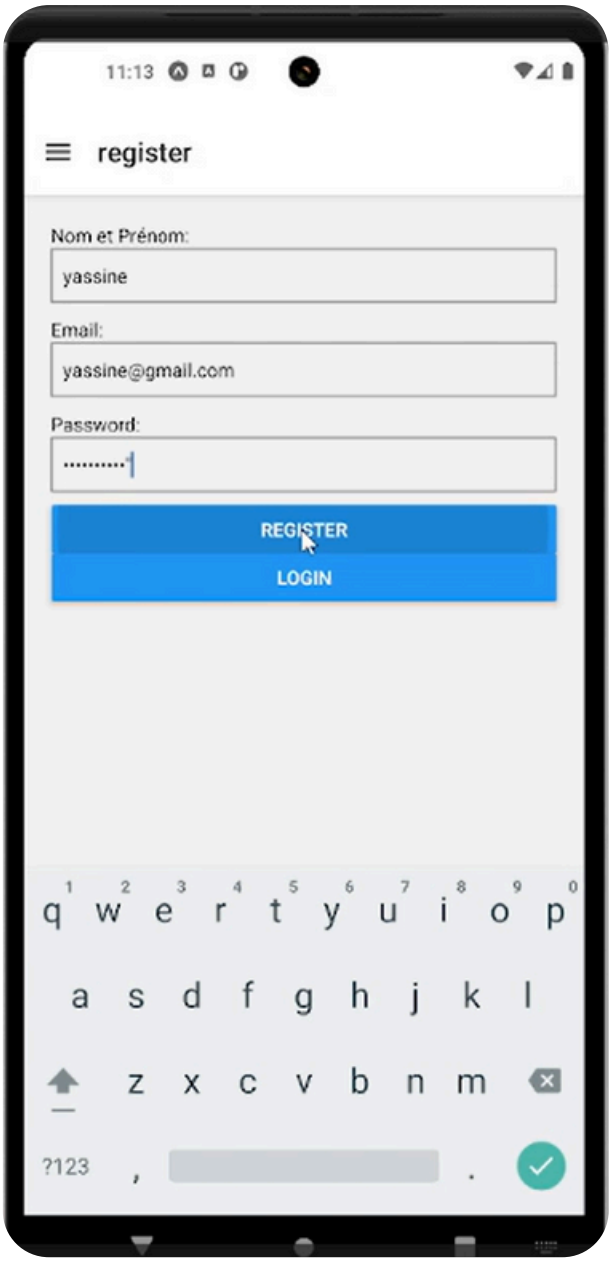


- Backend (API) :
  - Dossiers :
    - models : Définit les schémas de données (User, Book, Loan).
    - routes : Gère les endpoints (authentification, livres, emprunts).
    - middleware : Contrôle l'authentification (auth.js).
  - Fichiers Principaux :
    - index.js : Point d'entrée de l'API.
    - seed.js : Peuplement initial de la base de données.
- Frontend (Mobile) :
  - Écrans :
    - LoginScreen.js, RegisterScreen.js : Authentification.
    - BooksScreen.js : Liste des livres.
    - LoanScreen.js : Gestion des emprunts.
    - HomeScreen.js : Accueil.
  - Contexte d'Authentification :
    - AuthContext.js : Gère l'état global de l'utilisateur.

Fonctionnalités Détaillées

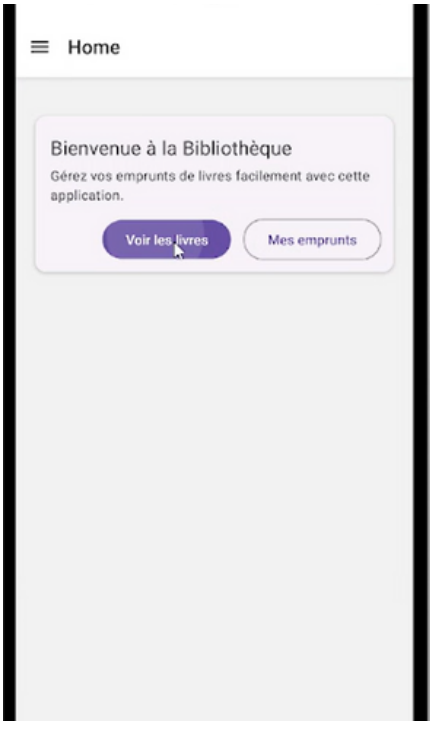
A. Authentification

- Inscription :
  - L'utilisateur saisit son nom, email et mot de passe.
  - Validation des champs et enregistrement dans la base de données.
- Connexion :
  - Vérification des identifiants via un middleware JWT (auth.js).
  - Redirection vers l'accueil après authentification.



- **Connexion**
- **Inscription**

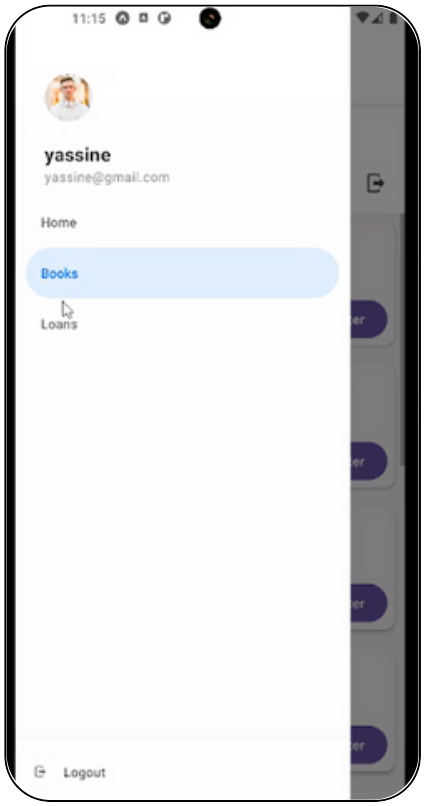
B. Page D'accueil



- Description de l'interface :
  - En-tête : En haut, on voit une barre d'application avec un menu hamburger à gauche, le titre Home, et des icônes système en haut (batterie, heure, etc.).
  - Carte de bienvenue :
    - Titre : Bienvenue à la Bibliothèque.
    - Texte : Gérez vos emprunts de livres facilement avec cette application.
  - Deux boutons d'action :
    - "Voir les livres" (bouton principal violet)
    - "Mes emprunts" (bouton secondaire, contour violet)

C. Navigation

- Menu latéral ou onglets pour accéder à :
  - Accueil : Tableau de Bord des Emprunts si utilisateur est connecter sinon Dialogue (Alert) pour demander a l'utilisateur de s'authentifier.
  - Livres : Catalogue complet.
  - Emprunts : Suivi des prêts.
- Déconnexion : Fermeture de la session.

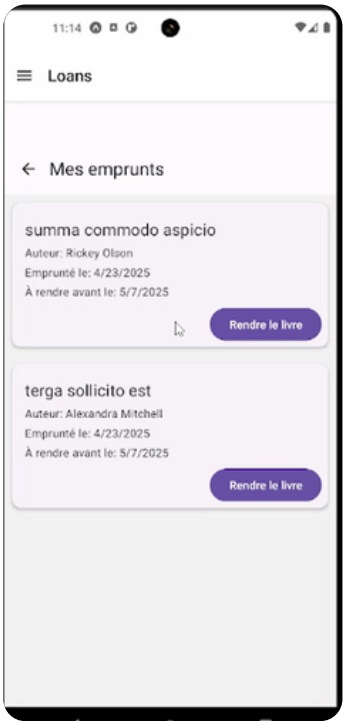


D. Tableau de Bord des Emprunts

Mes Emprunts :

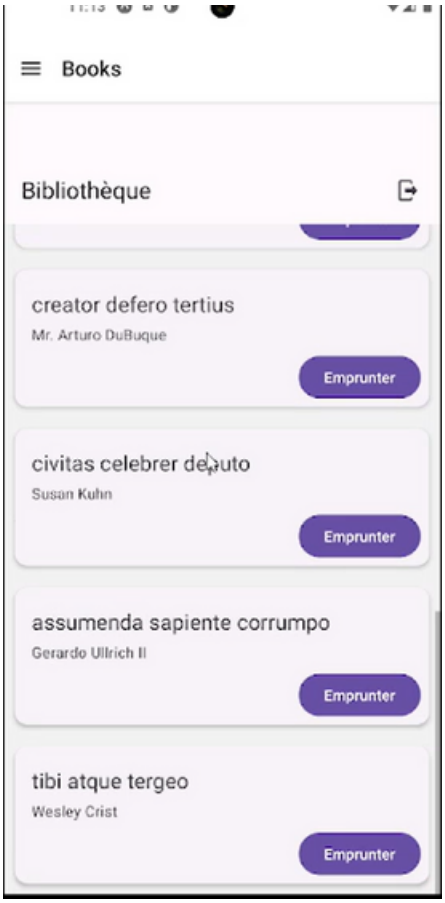
- Liste des livres empruntés avec un botton pour chaque case pour rendre le livre.

Message d'absence d'emprunt si aucun prêt n'est actif



E. Gestion des Livres

- Liste des Livres :
  - Affichage des titres, auteurs et bouton “**Emprunter**”.
  - Exemple : “creator defero tertius” par Arturo DuBuque.
- Emprunt d’un Livre :
  - L’utilisateur clique sur “Emprunter” pour ajouter un livre à ses prêts.
- Mise à jour en temps réel via l’API.



Explication Technique

- Backend (API)
  - Modèles de Données :
    - `User : { nom, email, password }.`
    - `Book : { titre, auteur, disponibilité }.`
    - `Loan : { userId, bookId, dateEmprunt }.`
  - Endpoints :
    - `POST /user/register : Création de compte.`
    - `POST /user/login : Génération de token JWT.`
    - `GET /books/disponibles : les livres disponible.`
    - `POST /books: Ajouter un livre.`
    - `PATCH /books/:id: Mettre à jour un livre.`
    - `DELETE /books/:id: Supprimer un livre.`
    - `GET /loans/my-loans : Lister les emprunts.`
    - `POST /loans : Emprunter un livre.`
    - `PATCH/loans/:id/return : Rendre un livre.`

```
// Modèles
const User = require('./models/User');
const Book = require('./models/Book');
const Loan = require('./models/Loan');

// Routes
app.use('/auth', require('./routes/auth'));
app.use('/books', require('./routes/books'));
app.use('/loans', require('./routes/loans'));      You, yesterday • Added
app.use('/user', require('./routes/user'));

const PORT = 3000;
```

- /auth
  - Ce préfixe est utilisé pour toutes les routes liées à l'authentification des utilisateurs, comme la connexion **(/auth/login)**, l'inscription (/auth/register), ou la réinitialisation de mot de passe.
  - Exemple : POST /auth/login pour se connecter.
- /books
  - Ce préfixe regroupe les routes liées à la gestion des livres, comme l'ajout, la suppression, ou la recherche de livres.
  - Exemple : GET /books pour lister tous les livres.
- /loans
  - Ce préfixe est dédié aux routes concernant les emprunts de livres, comme la création d'un emprunt ou la restitution.
  - Exemple : POST /loans pour emprunter un livre.
- /user
  - Ce préfixe concerne les routes relatives à la gestion du profil utilisateur, comme la mise à jour des informations ou la consultation des emprunts en cours.
  - Exemple : GET /user/profile pour voir son profil.

Conclusion

Cette application répond aux besoins de gestion de bibliothèque en offrant une solution moderne et intuitive. Les technologies choisies (**Node.js, React Native)** garantissent évolutivité et performance. Les améliorations futures pourraient inclure des notifications de retour de livres ou un système de réservation.