Documentation Technique : Application de Gestion de Bibliothèque

1. Titre et Résumé du Projet

Nom du Projet :

Application de Gestion de Bibliothèque

Résumé :

Cette application web permet de gérer une bibliothèque en ligne. Elle offre des fonctionnalités CRUD sur les livres (ajout, modification, suppression, consultation), un filtrage avancé, de la pagination, ainsi que l'affichage de statistiques (livres par catégorie, par année, disponibilité). Le frontend est développé en Angular, le backend en Node.js (Express) et la base de données est gérée par PostgreSQL. L'application est conçue pour être légère, flexible, et facilement extensible.

2. Technologies Utilisées

Frontend :

- **Framework :** Angular 15 (composants standalone, TypeScript)
- **Langages :** TypeScript, HTML, CSS
- **UI :** Composants personnalisés basés sur Angular

Backend :

- **Plateforme :** Node.js
- **Framework :** Express.js (API RESTful)
- **ORM :** Sequelize (intégration avec PostgreSQL)
- **Validation :** Joi (validation côté backend)

Base de Données :

- **SGBD :** PostgreSQL
- **Table books :**
  - `id` SERIAL PRIMARY KEY
  - `title` VARCHAR(255) NOT NULL
  - `author` VARCHAR(255) NOT NULL

— `publishedYear` INTEGER
— `category` VARCHAR(255)
— `isbn` VARCHAR(20)
— `available` BOOLEAN DEFAULT TRUE

Outils Complémentaires :

— **Postman** : Tests manuels des endpoints de l'API
— **Git** : Gestion de version
— **VS Code** : IDE

3. Fonctionnalités du Projet

Gestion des Livres (CRUD)

— Ajouter un livre (avec validation).
— Modifier un livre existant.
— Supprimer un livre par son ID.
— Consulter un livre.

Recherche Avancée et Filtres

— Filtrage par titre, auteur, catégorie.
— Combinaison de critères.

Visualisation des Statistiques

— Livres par catégorie.
— Livres par année.
— Disponibilité des livres.

4. Architecture de l'Application

Schéma de l'Architecture (Description)

Client (Navigateur) → Frontend Angular (http ://localhost :4200) communique via HTTP/JSON Backend Node.js/Express (http ://localhost :3000/api) → Base PostgreSQL

Le frontend interroge le backend via des requêtes HTTP. Le backend interagit avec PostgreSQL via Sequelize et renvoie des données JSON au frontend. Le frontend met à jour la vue en fonction des données reçues (liste de livres, statistiques, etc.).

Organisation des Dossiers

Frontend : (`./frontend/src/app/`)

```
app/
 components/          # Composants Angular (ex: BookListComponent, BookFormComponent)
 models/              # Interfaces TypeScript pour les données
 services/            # Services Angular (BookService, StatisticsService) pour appels
 app.component.ts     # Composant racine
 app.component.html   # Template du composant racine
 app.component.css    # Styles du composant racine
 app.config.ts        # Fourniture globale de HttpClient et autres
 app.routes.ts        # Définition des routes Angular
```

- Le dossier `components` contient les composants du frontend. - Le dossier `models` stocke les interfaces décrivant la forme des données. - Le dossier `services` gère la logique de communication avec le backend.

Backend : (`./backend/src/`)

```
src/
 config/         # Configuration de la BD
 controllers/    # Logique métier (ex: BookController.js)
 database/       # Espace pour init DB (actuellement vide)
 middlewares/    # Middlewares Express (validation, authentification - vide pour l'ins
 models/         # Modèles Sequelize (ex: Book.js)
 routes/         # Routes API (ex: bookRoutes.js)
 utils/          # Fonctions utilitaires (actuellement vide)
 app.js          # Point d'entrée Express
```

Cette organisation permet une séparation claire des responsabilités, facilitant maintenance et évolution.

5. API Documentation

Base URL

```
http://localhost:3000/api
```

Endpoints Principaux

Méthode	Endpoint	Description
POST	/books	Créer un nouveau livre
GET	/books	Obtenir tous les livres (paginés, filtrables)
GET	/books/ :id	Obtenir un livre par ID
PUT	/books/ :id	Mettre à jour un livre
DELETE	/books/ :id	Supprimer un livre
GET	/books/statistics	Récupérer les statistiques

6. Installation et Configuration

Prérequis

— Node.js 16+

— PostgreSQL 13+

Étapes d'Installation

1. Cloner le dépôt :

```
git clone https://github.com/JustBarry6/library-management.git
```

2. Installer les dépendances backend et frontend :

```
cd backend
npm install
cd ../frontend
npm install
```

3. Configurer PostgreSQL et créer la base de données 'library$_m$*anagement*'.

4. Créer un fichier '.env' dans 'backend/' :

```
DB_HOST=localhost
DB_USER=postgres
DB_PASSWORD=your_password
DB_NAME=library_management
DB_PORT=5432
```

5. Démarrer le backend :

```
cd backend
npm start
```

6. Démarrer le frontend :

```
cd ../frontend
ng serve
```

7. Accéder à l'application via `http://localhost:4200`.

7. Guide d'Utilisation

— Rendez-vous sur `http://localhost:4200/books` pour voir la liste des livres.
— Utilisez le formulaire de recherche pour filtrer par titre, auteur, ou catégorie.
— Cliquez sur "Ajouter un livre" pour insérer un nouveau livre.
— Cliquez sur un livre dans la liste pour le consulter.
— Les statistiques sont affichées sur la page principale, montrant le nombre de livres par catégorie, année et disponibilité.

8. Qualité, Performance et Bonnes Pratiques

— **Pagination :** Améliore les performances et l'expérience utilisateur.
— **Validation des données (Joi) :** Assure l'intégrité des données.
— **Séparation des responsabilités :** Code organisé selon les principes SOLID.