Système de Gestion de Livres et Auteurs

Cette application permet de gérer des livres et des auteurs. Elle utilise un modèle de micro services où plusieurs services interagissent via gRPC, REST, et Kafka pour envoyer des événements. Le système est composé des éléments suivants :

* **Service Book** : Gestion des livres (ajout, récupération, recherche).
* **Service Author** : Gestion des auteurs (ajout, récupération, recherche).
* **API Gateway** : Point d'entrée unique pour les utilisateurs, exposant les services REST et GraphQL.
* **Kafka** : Utilisé pour l'envoi d'événements entre les services.

1. Mes etapes pour la creation de mon projet :

* Création du service gRPC pour les livres et auteurs :

Définition du fichier .proto

**Création du modèle Mongoose**

**Créer bookMicroservice et authorMicrosservice : gérer les methodes**

* **Le GraphQL :Creation du schema**

Et resolvers : le résolveur récupère les données demandées ou fait des action (le add)

* L’apiGetway :

il est comme une porte d'entrée unique pour toutes les requêtes clients vers plusieurs microservices.

Intégration de Kafka dans les microservices :

**Producteurs Kafka** : Chaque microservice implémente une logique de producteurpour envoyer des messages à Kafka lors de la création ou la mise à jour des données.

**Consommateurs Kafka** : Les microservices consomment également des messages de Kafka pour synchroniser les données ou répondre à des événements générés par d'autres services.

**2. Composants du Système**

**2.1. Service Book (gRPC)**

Le service de gestion des livres est responsable de la gestion des informations sur les livres (ajout, récupération, recherche). Ce service communique avec les autres services via gRPC.

**Endpoints gRPC** :

* **getBook** : Récupère les informations d'un livre par son ID.
* **searchBooks** : Recherche des livres selon un critère.
* **addBook** : Ajoute un nouveau livre dans la base de données.

**2.2. Service Author (gRPC)**

Le service des auteurs permet de gérer les informations des auteurs (ajout, récupération, recherche). Comme le service Book, il utilise gRPC pour communiquer avec les autres services.

**Endpoints gRPC** :

* **getAuthor** : Récupère les informations d'un auteur par son ID.
* **searchAuthors** : Recherche des auteurs selon un critère.
* **addAuthor** : Ajoute un nouvel auteur.

**2.3. API Gateway (REST + GraphQL)**

L'API Gateway agit comme un point d'entrée unique pour les utilisateurs, en exposant les services via une API REST et GraphQL.

**Endpoints REST** :

* **GET /api/books** : Récupère une liste de livres.
* **GET /api/books/:id** : Récupère un livre par son ID.
* **POST /api/books** : Ajoute un livre.
* **GET /api/authors** : Récupère une liste d'auteurs.
* **GET /api/authors/:id** : Récupère un auteur par son ID.
* **POST /api/authors** : Ajoute un auteur.

**2.4 Kafka (système de Messagerie)**

Kafka est utilisé pour la gestion des événements au sein du système. Lorsqu'un livre est ajouté, un message est envoyé à un topic Kafka pour notifier les autres services intéressés.

**Topic Kafka** :

* **books\_topic** : Reçoit les événements relatifs à l'ajout de livres.
* **authors\_topic** : Reçoit les événements relatifs à l'ajout d'auteurs.

