Rapport de Projet : Développement de la Plateforme E-learning Backend

1. Introduction

Nous travaillons sur le développement d'une plateforme d'apprentissage en ligne avec un accent particulier sur le développement backend utilisant Node.js et PostgreSQL. Le projet implique la gestion de diverses entités telles que `Utilisateur`, `Formations`, `Cours`, et `Modules`. La plateforme est conçue pour gérer les opérations CRUD (Create, Read, Update, Delete) pour ces entités via des routes API RESTful.

2. Configuration de la Base de Données

Nous utilisons PostgreSQL comme système de gestion de base de données pour ce projet. La base de données est structurée pour gérer les tables suivantes :

- **Utilisateur** : Gère les données des utilisateurs, incluant les informations d'identification et les données personnelles.
- **Formations** : Gère les programmes de formation ou les cours disponibles.
- **Cours** : Gère les cours individuels au sein d'un programme de formation, incluant les détails et les objectifs des cours.
- **Modules** : Gère le contenu de chaque cours, y compris les leçons et les sujets abordés.
- **Apprenant (Learner/Student)** : Gère les informations spécifiques aux apprenants, y compris leur progression et leur inscription aux formations.
- **Administrateur (Admin)** : Gère les données relatives aux administrateurs du système, y compris les privilèges d'accès et les responsabilités.
- **Formation (Enrollment)** : Gère les inscriptions des utilisateurs aux programmes de formation, y compris les dates et les statuts des inscriptions.
- **Paiement (Payment)** : Gère les informations de paiement liées aux inscriptions aux formations, y compris les montants et les méthodes de paiement.
- **Certification (Certification)** : Gère les certificats délivrés aux utilisateurs après la réussite des formations, incluant les détails de la certification et les dates de délivrance.
- **Editeur (Editor)** : Gère les informations des éditeurs qui créent ou modifient le contenu des cours et des articles.
- **Article (Article for Blog)** : Gère les articles de blog publiés sur la plateforme, incluant les titres, les contenus, et les auteurs.

- **Session (User Session)** : Gère les sessions des utilisateurs, y compris les informations de connexion, les durées de session, et les activités des utilisateurs.

Les détails de connexion à la base de données sont les suivants :

- **Nom d'utilisateur :** `postgres`
- **Mot de passe :** `1234Kamal`
- **Port :** `5432`
- **Nom de la Base de Données :** `elearning`

3. Configuration du Serveur

Nous avons configuré un serveur utilisant Node.js et Express.js. Le serveur écoute sur le port `3000` et gère diverses routes API pour interagir avec la base de données.

4. Routes Implémentées

4.1 Routes Utilisateur

- **GET /utilisateurs :** Récupère tous les utilisateurs.
- **GET /utilisateurs/**

  : Récupère un utilisateur par ID.

- **POST /utilisateurs :** Ajoute un nouvel utilisateur.
- **PUT /utilisateurs/**

  : Met à jour un utilisateur existant.

- **DELETE /utilisateurs/**

  : Supprime un utilisateur.

4.2 Routes Formations

- **GET /formations :** Récupère tous les programmes de formation.
- **GET /formations/**

  : Récupère un programme de formation par ID.

- **POST /formations :** Ajoute un nouveau programme de formation.
- **PUT /formations/**

  : Met à jour un programme de formation existant.

- **DELETE /formations/**

  : Supprime un programme de formation.

4.3 Routes Cours

- **GET /cours :** Récupère tous les cours.
- **GET /cours/**

  : Récupère un cours par ID.

- **POST /cours :** Ajoute un nouveau cours.
- **PUT /cours/**

  : Met à jour un cours existant.

- **DELETE /cours/**

: Supprime un cours.

4.4 Routes Modules

- **GET /modules :** Récupère tous les modules.
- **GET /modules/**

  : Récupère un module par ID.

- **POST /modules :** Ajoute un nouveau module.
- **PUT /modules/**

  : Met à jour un module existant.

- **DELETE /modules/**

  : Supprime un module.

5. État Actuel

- **Configuration de la Base de Données :** Complète, avec des relations entre les tables telles que des relations un-à-plusieurs entre `Utilisateurs` et `Cours`, ainsi qu'entre `Cours` et `Modules`.
- **Développement de l'API :** Les opérations CRUD pour `Utilisateur`, `Formations`, `Cours`, et `Modules` sont implémentées et fonctionnelles.
- **Tests :** Les routes API ont été testées avec Postman pour s'assurer qu'elles gèrent les entrées attendues et renvoient les résultats corrects. La plateforme est prête pour un développement et une intégration plus poussés avec le frontend.

6. Prochaines Étapes

- **Intégration :** Commencer l'intégration du backend avec le frontend de la plateforme.
- **Tests :** Effectuer des tests plus approfondis, y compris les cas limites et la gestion des erreurs.
- **Documentation :** Compléter la documentation détaillée pour toutes les routes et fonctionnalités afin de garantir que le projet soit maintenable.