TP de positionnement

Développement d'une API d'Émargement avec Node.js et, Express

Contexte

Vous êtes chargé(e) de développer une API REST permettant de gérer une application d'émargement pour des formations. Cette application permettra :

- De gérer des utilisateurs (formateurs et étudiants).
- De créer et gérer des sessions de cours.
- De permettre aux étudiants de signer via un système sécurisé d'authentification par JWT.

Objectifs

1. Développer une API REST en Node.js avec Express.
2. Implémenter une gestion des utilisateurs (inscription et connexion avec JWT).
3. Protéger les routes sensibles avec un middleware d'authentification basé sur JWT.
4. Permettre la création et la gestion des sessions de cours (CRUD).
5. Implémenter une fonctionnalité d’émargement (ajout de la présence d’un étudiant à une session).

Méthode d’évaluation

Pour cet exercice pratique, vous devrez créer un dépôt sur github.com afin de permettre l’évaluation de votre travail. Il vous faudra également ajouter l’utilisateur *hellodamien* aux collaborateurs de votre dépôt pour lui donner accès et mentionner l’URL de votre dépôt ci-dessous.

URL du dépôt sur github.com : ...

Si cela est nécessaire, vous pouvez également documenter votre application à l’aide d’un fichier README.md.

---

1. Initialisation du projet

- Créer un projet Node.js avec npm init.
- Installer les dépendances nécessaires :

  express body-parser jsonwebtoken bcrypt mysql2 zod dotenv
- Installer les dépendances pour le développement :

nodemon

2. Modèle de données

L'API doit gérer trois entités principales :

- **Utilisateur** :
  - id (int) : Identifiant unique.
  - name (string) : Nom de l'utilisateur.
  - email (string) : Adresse email (unique).
  - password (string) : Mot de passe (haché).
  - role (string) : formateur ou etudiant.
- **Session** :
  - id (int) : Identifiant unique.
  - title (string) : Titre de la session (ex. : "Cours Node.js").
  - date (date) : Date de la session.
  - formateur_id (int) : ID du formateur responsable.
- **Émargement** :
  - id (int) : Identifiant unique.
  - session_id (int) : ID de la session.
  - etudiant_id (int) : ID de l'étudiant.
  - status (bool) : Statut de présence (true ou false).

Fonctionnalités

Votre API doit inclure les routes suivantes :

1. Gestion des utilisateurs :

- **POST /auth/signup** : Inscription d’un utilisateur avec hachage du mot de passe.
- **POST /auth/login** : Connexion et génération d’un token JWT.

2. Gestion des sessions de cours :

- **POST /sessions** *(formateur uniquement)* : Création d’une session.
- **GET /sessions** : Liste de toutes les sessions.
- **GET /sessions/:id** : Détails d’une session spécifique.
- **PUT /sessions/:id** *(formateur uniquement)* : Modification d’une session.
- **DELETE /sessions/:id** *(formateur uniquement)* : Suppression d’une session.

3. Gestion des émargements :

- **POST /sessions/:id/emargement** *(étudiant uniquement)* : Émargement à une session (ajoute une ligne dans la table d'émargement).
- **GET /sessions/:id/emargement** *(formateur uniquement)* : Liste des étudiants émargés pour une session.

4. Mécanismes de sécurité

- Implémentez un middleware pour protéger les routes avec JWT (authentification).
- Assurez-vous que :
  - Seuls les formateurs peuvent créer, modifier ou supprimer des sessions.
  - Seuls les étudiants peuvent s’émarger à une session.

5. Exigences techniques

- L'API doit être développée en Node.js avec Express.
- Utilisez mysql2/promise pour interagir avec une base de données MySQL.
- Les mots de passe des utilisateurs doivent être hachés avec bcrypt.
- Les tokens JWT doivent inclure l'id et le role de l'utilisateur.