Application TODO list

L'application de gestion de tâches permettra aux utilisateurs de s'inscrire, de se connecter, de créer, de lire, de mettre à jour et de supprimer des tâches. Chaque utilisateur aura ses propres tâches, et l'application devra gérer les opérations CRUD de manière sécurisée et efficace.

1. Initialisation des projets

Back-end (Node.js, Express, MySQL)

1. **Créer le projet Node.js** :
    - Initialisez un nouveau projet Node.js et installez les dépendances nécessaires (Express et MySQL).
    - Configurez la connexion à la base de données MySQL.
2. **Configurer la base de données** :
    - Créez une base de données MySQL nommée task_manager.
    - Créez les tables users et tasks avec les champs appropriés pour stocker les informations des utilisateurs et des tâches.

Front-end (React)

1. **Créer le projet React** :
    - Initialisez un nouveau projet React avec create-react-app.
    - Installez Axios pour gérer les appels API.

2. Développement du Back-end

1. **Configurer Express et MySQL** :
    - Configurez le serveur Express pour gérer les requêtes HTTP et connectez-vous à la base de données MySQL.
    - Créez les routes nécessaires pour les opérations CRUD sur les tâches et la gestion des utilisateurs.
2. **Gestion de l'authentification** :
    - Implémentez des routes pour l'inscription et la connexion des utilisateurs.
    - Utilisez des requêtes SQL pour vérifier les informations d'identification des utilisateurs lors de la connexion.

3. **Routes API pour les tâches** :
    - Créez des routes pour permettre aux utilisateurs de créer, lire, mettre à jour et supprimer des tâches.
    - Assurez-vous que chaque utilisateur ne peut accéder qu'à ses propres tâches en utilisant des paramètres de requête ou des en-têtes HTTP pour identifier l'utilisateur.

3. Développement du Front-end

1. **Création des composants React** :
    - Créez des composants pour l'inscription, la connexion, l'ajout, la modification, la suppression et la visualisation des tâches.
    - Utilisez des hooks comme useState et useEffect pour gérer l'état et les effets secondaires.
2. **Gestion de l'état avec React Hooks** :
    - Utilisez useState pour gérer l'état local des composants (par exemple, les informations de connexion, les tâches).
    - Utilisez useEffect pour effectuer des appels API et mettre à jour l'état en conséquence (par exemple, récupérer les tâches après la connexion).
3. **Appels API** :
    - Utilisez Axios pour faire des appels API à votre serveur Express.
    - Gérez les réponses et les erreurs des appels API pour mettre à jour l'interface utilisateur en conséquence.

4. Tests

1. **Tests** :
    - Testez manuellement l'application pour vous assurer que toutes les fonctionnalités fonctionnent correctement.
    - Utilisez des outils comme Postman pour tester les routes API.