Application "Let's Play" Client

Présentation

Dans cet exercice, vous devez développer la partie client d'une application de jeux en ligne appelée "Let's Play" en utilisant ReactJS. L'application se connectera à une API existante pour gérer l'authentification, la gestion des utilisateurs et des points.

L'objectif de cet exercice est de créer une interface utilisateur interactive et conviviale qui permettra aux utilisateurs de se connecter, de s'inscrire, de gérer leur profil, de visualiser les utilisateurs enregistrés, d'ajouter ou de supprimer des points, et de réinitialiser leur propre pointage.

Vous pouvez accéder à l'API à l'adresse : https://api.joeleprof.com/letsplay
Vous pouvez accéder à la documentation à l'adresse : https://api.joeleprof.com

Consignes :

- Ce TP se fait individuel ou en équipe de deux.
- La date de remise sera communiquée sur Omnivox. Les retards ne sont acceptés pas , mais entraîneront une pénalité de 10 points par journée de retard.
- Pour la remise, assurez-vous de supprimer le dossier "node_modules" et le fichier "package-lock.json" de votre projet.
- *(Seulement si vous être en équipe)* Seule une personne de l'équipe doit effectuer la remise via Omnivox.
- Vous devez également fournir un rapport disponible sur google form : https://docs.google.com/forms/d/e/1FAIpQLSdUtOV5hbuOEe2CchJz7BGPWW53F7TFt9ujTKc5dsSEQbNKyw/viewform?usp=sf_link
- *(Seulement si vous être en équipe)* Notez que je ne corrigerai pas le TP si je n'ai pas reçu de rapport de votre part.
- *(Seulement si vous être en équipe)* Vous devrez héberger votre application ReactJs en ligne.

Authentification (40 points)

Page de connexion (20 points)

Afficher un formulaire de connexion avec des champs pour l'adresse e-mail et le mot de passe. (4 points)

Permettre aux utilisateurs de se connecter en utilisant l'API " **POST /auth/login** " en envoyant les informations d'identification. (8 points)

Gérer les erreurs de connexion et afficher des messages appropriés. (8 points)

- Paramètres de la requête :
  - email (string, obligatoire) : Adresse e-mail de l'utilisateur.
  - password (string, obligatoire) : Mot de passe de l'utilisateur.
- Réponse réussie :
  - Statut : 200 OK
  - Corps de la réponse : Token d'authentification pour l'utilisateur connecté.
- Réponses d'erreur possibles :
  - 400 Bad Request : Si des paramètres sont manquants ou invalides.
  - 401 Unauthorized : Si l'adresse e-mail ou le mot de passe est incorrect.

Page d'inscription (20 points)

Afficher un formulaire d'inscription avec des champs pour l'adresse e-mail, le mot de passe et le nom d'utilisateur. (4 points)

Permettre aux nouveaux utilisateurs de s'inscrire en utilisant l'API " **POST /auth/register** " en envoyant les informations requises. (8 points)

Gérer les erreurs d'inscription et afficher des messages appropriés. (8 points)

- Paramètres de la requête :
  - email (string, obligatoire) : Adresse e-mail de l'utilisateur.
  - password (string, obligatoire) : Mot de passe de l'utilisateur.
  - username (string, obligatoire) : Nom de l'utilisateur.
- Réponse réussie :
  - Statut : 201 Created
  - Corps de la réponse : Token d'authentification pour l'utilisateur nouvellement inscrit.
- Réponses d'erreur possibles :
  - 400 Bad Request : Si des paramètres sont manquants ou invalides.
  - 409 Conflict : Si l'adresse e-mail est déjà utilisée par un autre utilisateur.

Gestion des utilisateurs (30 points)

- Afficher la liste des utilisateurs enregistrés en utilisant l'API "/users". (6 points)
- Permettre aux utilisateurs avec des droits d'accès administrateur de modifier ou de supprimer un utilisateur en utilisant les API "/users/:id" (PUT et DELETE). (12 points)
- Gérer les erreurs de modification et de suppression et afficher des messages appropriés. (12 points)

Liste des utilisateurs

- Endpoint: **GET /users**
- Paramètres de la requête :
  - **Authorization** (string, obligatoire) : Token d'authentification de l'utilisateur.
- Réponse réussie :
  - Statut : 200 OK
  - Corps de la réponse : Liste des utilisateurs sauf l'administrateur.
- Réponses d'erreur possibles :
  - 401 Unauthorized : Si l'utilisateur n'est pas authentifié ou si le token est invalide

Modifier un utilisateur (Admin seulement)

- Endpoint: **PUT /users/:id**
- Paramètres de la requête :
  - id (string, obligatoire) : Identifiant de l'utilisateur à modifier.
  - username (string, facultatif) : Nouveau nom de l'utilisateur.
  - email (string, facultatif) : Nouvelle adresse e-mail de l'utilisateur.
  - **Authorization** (string, obligatoire) : Token d'authentification de l'utilisateur.
- Réponse réussie :
  - Statut : 200 OK
  - Corps de la réponse : Utilisateur modifié.
- Réponses d'erreur possibles :
  - 401 Unauthorized : Si l'utilisateur n'est pas authentifié, si le token est invalide ou n'a pas les droits d'accès administrateur.
  - 404 Not Found : Si l'utilisateur spécifié n'existe pas.

Supprimer un utilisateur (Admin seulement)

- Endpoint: DELETE /users/:id
- Paramètres de la requête :
  - id (string, obligatoire) : Identifiant de l'utilisateur à supprimer.
  - Authorization (string, obligatoire) : Token d'authentification de l'utilisateur.
- Réponse réussie :
  - Statut : 204 No Content
- Réponses d'erreur possibles :
  - 401 Unauthorized : Si l'utilisateur n'est pas authentifié ou n'a pas les droits d'accès administrateur.
  - 404 Not Found : Si l'utilisateur spécifié n'existe pas.

Gestion des points (30 points)

- Permettre à l'utilisateur d'ajouter des points en utilisant l'API "/game/wins" en spécifiant la quantité de points à ajouter. (15 points)
- Permettre à l'utilisateur de supprimer des points en utilisant l'API "/game/losts" en spécifiant la quantité de points à supprimer. (15points)

Incrémenter le pointage d'un utilisateur

- Endpoint: **PUT /game/wins**
- Description: Cette route permet d'incrémenter le pointage d'un utilisateur.
- Paramètres de la requête :
  - **Authorization** (string, obligatoire) : Token d'authentification de l'utilisateur.
- Réponse réussie :
  - Statut : 200 OK
  - Corps de la réponse : Pointage mis à jour de l'utilisateur.
- Réponses d'erreur possibles :
  - 401 Unauthorized : Si l'utilisateur n'est pas authentifié ou si le token est invalide.

Décrémenter le pointage d'un utilisateur

- Endpoint: **PUT /game/lost**
- Description: Cette route permet de décrémenter le pointage d'un utilisateur.
- Paramètres de la requête :
  - **Authorization** (string, obligatoire) : Token d'authentification de l'utilisateur.
- Réponse réussie :
  - Statut : 200 OK
  - Corps de la réponse : Pointage mis à jour de l'utilisateur.
- Réponses d'erreur possibles :
  - 401 Unauthorized : Si l'utilisateur n'est pas authentifié ou si le token est invalide.

Utilisateur actuel (20 points)

- Afficher les informations de l'utilisateur actuellement connecté en utilisant l'API "/me". (4 points)
- Permettre à l'utilisateur de modifier ses informations personnelles en utilisant l'API "/me" (PUT). (8 points)
- Permettre à l'utilisateur de supprimer son compte en utilisant l'API "/me" (DELETE). (4 points)
- Permettre à l'utilisateur de réinitialiser son pointage en utilisant l'API "/me/reset-score". (4 points)

Obtenir les informations de l'utilisateur actuel

- Endpoint: **GET /me**
- Description: Cette route permet d'obtenir les informations de l'utilisateur actuellement authentifié.
- Réponse réussie :
    - Statut : 200 OK
    - Corps de la réponse : Informations de l'utilisateur.
- Réponses d'erreur possibles :
    - 401 Unauthorized : Si l'utilisateur n'est pas authentifié ou si le token est invalide.

Modifier les informations de l'utilisateur actuel

- Endpoint: **PUT /me**
- Description: Cette route permet de modifier les informations de l'utilisateur actuellement authentifié.
- Paramètres de la requête :
    - username (string, facultatif) : Nouveau nom de l'utilisateur.
    - email (string, facultatif) : Nouvelle adresse e-mail de l'utilisateur.
- Réponse réussie :
    - Statut : 200 OK
    - Corps de la réponse : Utilisateur modifié.
- Réponses d'erreur possibles :
    - 401 Unauthorized : Si l'utilisateur n'est pas authentifié ou si le token est invalide.
    - 404 Not Found : Si l'utilisateur actuel n'existe pas.

Supprimer l'utilisateur actuel

- Endpoint: **DELETE /me**
- Description: Cette route permet de supprimer l'utilisateur actuellement authentifié.
- Réponse réussie :
  - Statut : 204 No Content
- Réponses d'erreur possibles :
  - 401 Unauthorized : Si l'utilisateur n'est pas authentifié ou si le token est invalide.
  - 404 Not Found : Si l'utilisateur actuel n'existe pas.

Remettre à zéro le pointage de l'utilisateur

- Endpoint: **PUT /me/reset-score**
- Description: Cette route permet à l'utilisateur actuel de remettre à zéro son pointage.
- Paramètres de la requête :
  - **Authorization** (string, obligatoire) : Token d'authentification de l'utilisateur.
- Réponse réussie :
  - Statut : 200 OK
  - Corps de la réponse : Message indiquant que le pointage a été remis à zéro.
- Réponses d'erreur possibles :
  - 401 Unauthorized : Si l'utilisateur n'est pas authentifié ou si le token est invalide.