TP3 - Authentification, sécurité, middleware et API

👋 Introduction

Ce TP est dans la continuité du TP2 et suppose que ce dernier est terminé. Vous allez utiliser une authentification plus sécurisée et protéger les routes de votre application.

🔐 Authentification

1. Mise en place

Dans le précèdent TP vous avez crée un système d'authentification mais celui-ci n'est pas sécurisé et un peu trop permissif. Vous allez ajouter la librairie NextAuth.js
https://next-auth.js.org/

Vous conserverez votre page de login mais vous allez changer la méthode d'authentification. Suivez la documentation de NextAuth.js afin de créer une authentification par login/mot de passe.

2. Configurer le provider

Suivez les instructions afin de configurer un « credential » provider. L'idée étant d'aller chercher un utilisateur et de renvoyer lors de son authentification le clé API de TheMovieDatabase. Vous pouvez utiliser une constante ou une base de données (neon.tech).

3. Middleware

La documentation de NextJS vous donne des informations sur la mise en place d'un middleware afin de protéger les routes de votre application. Ce que j'attends de vous :

- Si un utilisateur essaye d'aller sur « /« est qu'il n'est pas authentifié, il doit être redirigé vers la page « /login »

- Si un utilisateur essayer d'aller « /« est qu'il est authentifié, il doit être redirigé vers la page « /dashboard »
- Si un utilisateur est authentifié est qu'il essaye d'accéder à la page « /login » il est redirigé vers « /dashboard »
- Si un utilisateur n'est pas authentifié et qu'il essaye d'aller sur la page « / dashboard » il est redirigé vers « /login »
- Toute les pages sous « /dashboard » sont protégées (seuls les utilisateurs authentifiés peuvent y accéder)

4. Protection de l'API

Toutes les endroits de votre API doivent être authentifiés. Il n'est plus possible de faire les appels s'il n'y a pas d'authentification. Adaptez votre API afin de prendre en compte cette protection. La clé API doit être récupéré depuis les informations de connexion de l'utilisateur (cookie, token ,etc).

🖥️ Appels API côté client

Il est temps de faire vos appels API côté client. Pour cela, vous allez créer une répertoire « repository » dans lequel vous allez créer vos requêtes vers votre API.
Si vous souhaitez mettre en place la « clean architecture » avec les contextes REACT vous pouvez. Sinon on verra plus tard…

Créez toutes vos requêtes avec en utilisant « fetch » et en vous assurant du type de l'objet retourné.

⚙️ Les hooks REACT

« Clean archi » ou non, je ne veux pas voir vos appels API dans vos composants visuels. Il est préférable de voir un hook personnalisé qui laissera votre composant « front » tranquille et facilement évolutif tout en permettant de conserver une séparation entre « présentation » et « logique métier ». Créer les hooks personnalisés utilisant ReactQuery afin de rendre votre code « propre » et facilement compréhensible.
https://tanstack.com/query/latest