React router (navigation multi pages)

Pour mettre en place un système de navigation entre plusieurs pages dans une application React, il est conseillé d’utiliser le routeur React comme React Router. Voici étape par étape comment intégrer React Router dans vos application existante (en partant de votre fichier App.js) :

**1. Installer React Router**

Dans vos projet React, ouvre un terminal à la racine et tapez :

bash

npm install react-router-dom@6

**2. Créer les pages composants**

Pour commencer, créez plusieurs fichiers composants pour représenter les différentes pages de vos application. Par exemple :

* Home.js (page d’accueil)
* About.js (page À propos)
* Profile.js (page profil, vous avez déjà un composant Profile que vous pouvez réutiliser)

## 3. Configurer React Router dans App.js

Modifie vos fichier App.js pour importer les composants du routeur et définir les routes :

Voici l’import :



On importe également ces pages qui sont dans un dossier pages pour être bien organisé,

Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement

Ensuite on met en place le router et les routes,

Une image contenant texte, capture d’écran, Police, logiciel

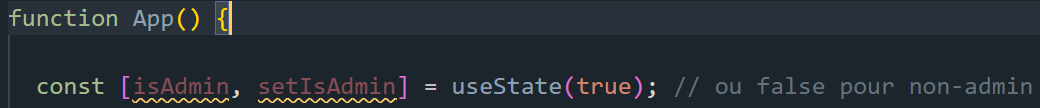
Description générée automatiquement

## 4. Détails supplémentaires

* Le composant <Router> englobe toute l’application ou au moins la zone où vous voulez activer la navigation.
* Le composant <Routes> contient les différentes routes possibles de votre app.
* Chaque <Route> a un path (URL) et un element qui est le composant à afficher.
* Les liens sont créés avec <Link to="..."> et remplacent les balises <a>. Ils permettent la navigation sans rechargement complet de la page.

Une fois cette structure en place, vous pourrez naviguer entre vos pages en cliquant sur les liens, et React Router prendra en charge le rendu du composant correspondant à l’URL.Si besoin, il est possible d’ajouter plus de pages et routes, ou gérer des routes dynamiques (ex : /profile/:id).

Imaginons que l’on veut protéger une page pour un admin, alors faisons cela, on va ajouter un state admin dans le App.js,



On crée un composant de protection appelé PrivateRoute.js dans le folder Components, et on et cela en place :

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Ensuite on utilise PrivateRoute dans le App.js, par exemple pour la route profile, on l’importe bien évidemment, et on modifie la route profile,

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Evidemment avec un système de connexion/inscription et définition d’un rôle user, avant la question, pour définir un rôle admin a un user cela se fait dans l’api backend de cette manière.

## 1. Au niveau backend (API)

* Lors de la création ou mise à jour d'un utilisateur, on attribue un champ rôle, souvent une propriété telle que role ou isAdmin.
* Exemple d’objet utilisateur stocké en base de données :

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

* Le serveur doit vérifier ce rôle lors des requêtes protégées.

## 2. Lors de la connexion

* Le backend renvoie, avec le jeton d’authentification (par exemple JWT), les informations du profil utilisateur, incluant son rôle.

Une image contenant texte, Police, capture d’écran, conception

Description générée automatiquement

## 3. Côté React

* Dans le contexte d’authentification (AuthContext ou AuthProvider), on stocke ces informations utilisateur, dont le rôle.

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

Puis pour protéger des zones admin, on vérifie dans le composant ou la route :

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

## 4. Contrôle d’accès dans les routes privées

* Le composant PrivateRoute (ou ProtectedRoute) utilisera la vérification du rôle.

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

En résumé : Le rôle admin est défini et géré côté backend avec les données utilisateur.L’information sur le rôle est envoyée à React à la connexion et stockée en état d’authentification.React utilise cette information pour conditionner l’accès aux routes et composants sensibles.