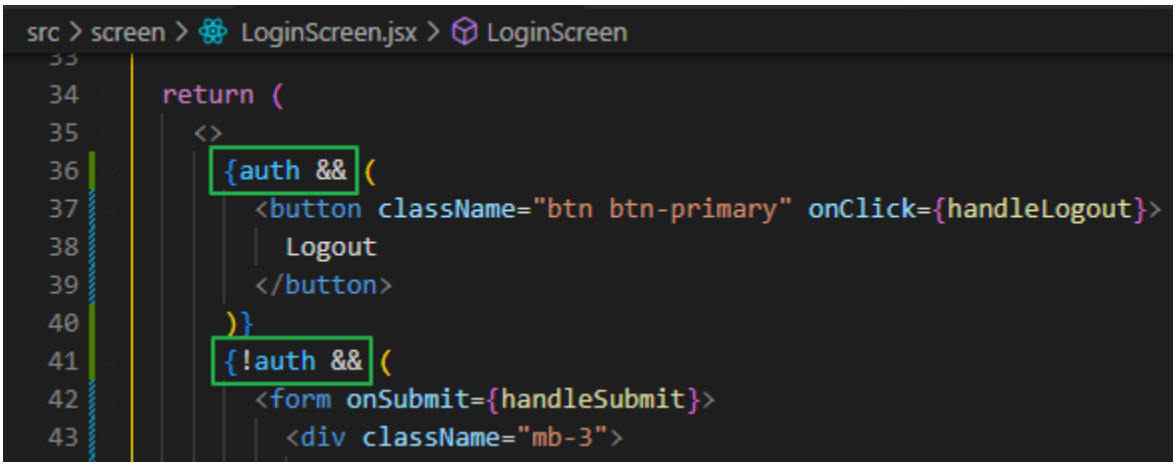
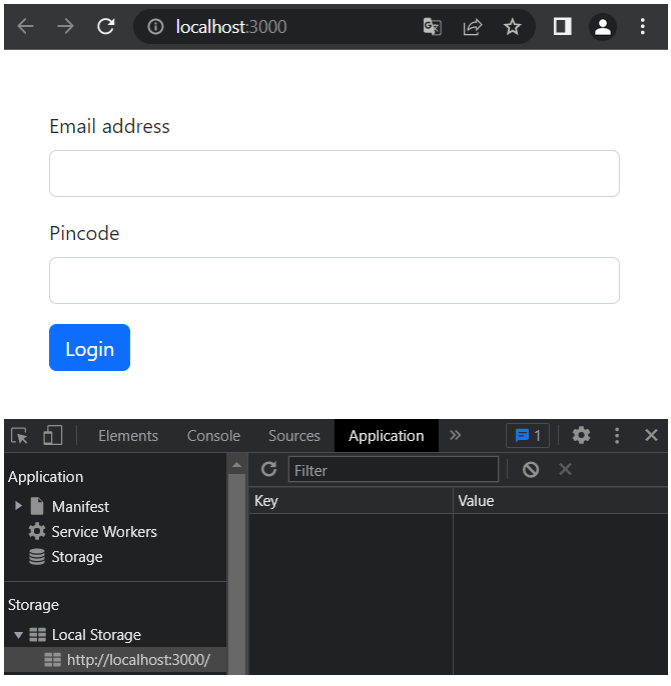
React, Node et MySql Login (Partie 5)

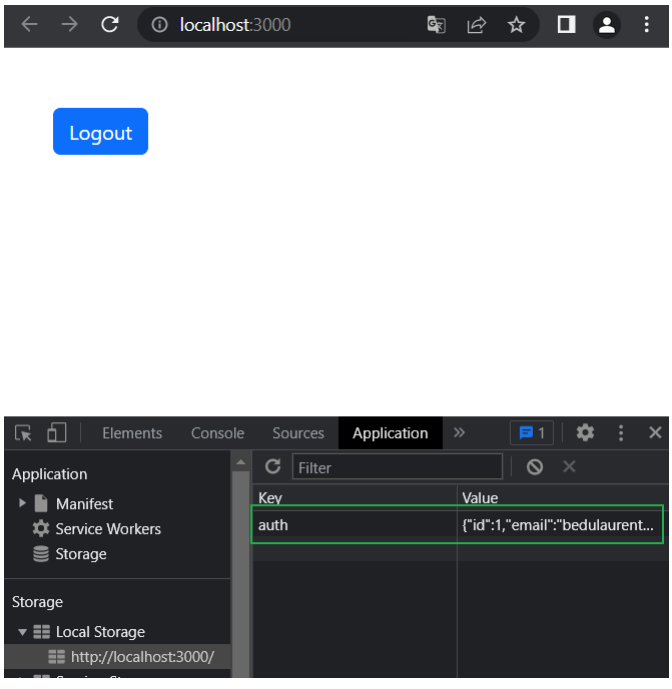
Stockage de l'info de login côté React (Version 2)

Avant de créer un context qui va nous servir à stocker les infos de login, vous pouvez tester à nouveau l'appli en ajoutant de l'affichage conditionnel
Si l'on est connecté on affiche le bouton logout, sinon le formulaire de connexion.

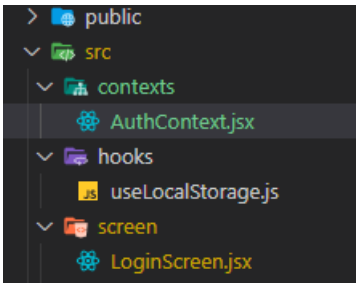


Tests :





Étape 10 : Création d'un contexte pour stocker l'info de login.



src > contexts > AuthContext.jsx > ...

```jsx
import { createContext, useState, useMemo } from "react";

const AuthContext = createContext();

const AuthProvider = ({ children }) => {

  const [auth, setAuth] = useState(null);
  const authMemo = useMemo(() => ({ auth, setAuth }), [auth]);

  return (
    <AuthContext.Provider value={authMemo}>
      {children}
    </AuthContext.Provider>
  );
};

export { AuthContext, AuthProvider };
```

Ln 3 : Création du context
Ln 5 : Déclaration du provider
Ln 7 : Création du state pour stocker les infos de login
Ln 8 : Création d'un memo pour améliorer les performances
(voir : https://fr.reactjs.org/docs/hooks-reference.html#usememo)
Ln 10 à 14 : Le composant retourne le provider du context.
Ln 17 : on exporte le context et son provider dans un objet
Pour le contexte, allez voir https://fr.reactjs.org/docs/context.html

Étape 11 : Utilisation du contexte
On englobe l'application avec le provider du contexte

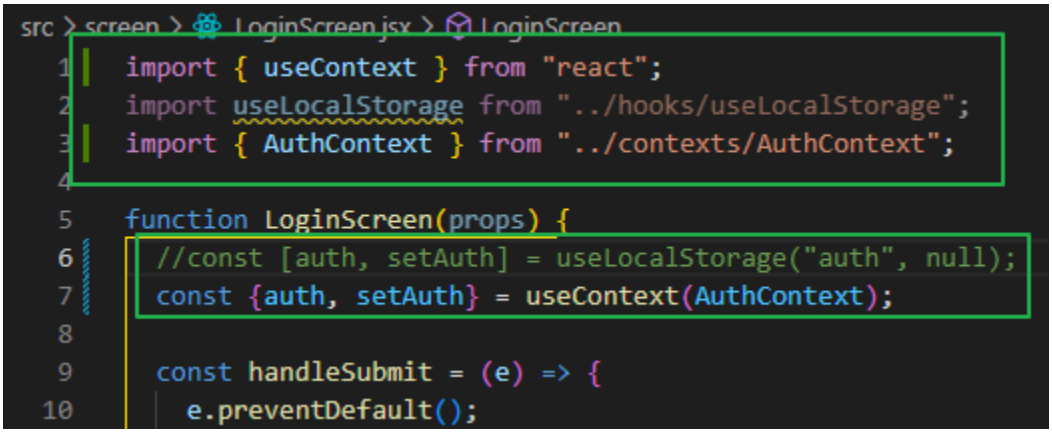
src > App.js > App

```jsx
import "bootstrap/dist/css/bootstrap.min.css";
import "bootstrap/dist/js/bootstrap.bundle.js";
import "./App.css";
import LoginScreen from "./screen/LoginScreen";
import { AuthProvider } from "./contexts/AuthContext";

function App() {
  return (
    <AuthProvider>
      <div className="App container pt-5">
        <LoginScreen/>
      </div>
    </AuthProvider>
  );
}

export default App;
```

On importe le contexte et on remplace le hook useLocalStorage par un useContext initialisé avec notre AuthContexte dans LoginScreen



Rien d'autre à faire, en testant on obtient le même comportement qu'avec useLocalStorage sauf que l'info n'est plus stockée dans le navigateur mais directement dans le contexte
Dans ce screen, l'utilisateur est bien authentifié car le bouton Logout est affiché, mais rien n'est stocké en localStorage