

Introducción al desarrollo Web con Ant Design

Local Storage

Para crear la sesión del usuario es necesario almacenar el token en algún lado, en el caso del desarrollo web, vamos a almacenar el token en el Local Storage, Para comenzar, debemos de crear el espacio para almacenar el token, lo haremos en el directorio de src/utills/constants.js crear STORAGE y almacenar TOKEN

```
src > utills > JS constants.js > [e] ENV
1  export const ENV = {
2    STORAGE: {
3      TOKEN: "token",
4    }
5  }
```

El segundo paso es crear las funciones para almacenar, obtener y eliminar el token, creamos un archivo en el directorio src/services/token.js

```
src > services > JS token.js > ...
1  import { ENV } from "../utils/constants";
2
3  // Función para almacenar el token en el local storage
4  const setToken = (token) => {
5    localStorage.setItem(ENV.STORAGE.TOKEN, token);
6  }
7
8  // Función para obtener el token del local storage
9  const getToken = () => {
10   return localStorage.getItem(ENV.STORAGE.TOKEN);
11 }
12 //Función para eliminar el token del local storage
13 const removeToken = () => {
14   localStorage.removeItem(ENV.STORAGE.TOKEN);
15 }
16 export const storageController = {
17   setToken,
18   getToken,
19   removeToken
20 }
21
```

Prueba que el token se está almacenando correctamente, en **AuthContext** crea la función de login (No olvides importar **storageController**) corrige data para poder usar la función fuera del contexto.

```
src > context > AuthContext.jsx > [e] AuthProvider > [e] data
1  import React, { useState, useEffect, createContext } from 'react';
2  import { storageController } from '../services/token';
3
4  export const AuthContext = createContext();
5
6  export const AuthProvider = (props) => {
7    const { children } = props;
8
9    const login = async (token) => {
10     try {
11       console.log('Obteniendo', token);
12       await storageController.setToken(token);
13     } catch (error) {
14       console.error(error);
15     }
16   }
17   const data = {
18     user: 'Ana',
19     login,
20     logout: () => console.log('logout'),
21     updateUser: () => console.log('updateUser'),
22   };
23
24   return (
25     <AuthContext.Provider value={data}>
26       {children}
27     </AuthContext.Provider>
28   );
29 };
```

```
const { login } = useAuthData
```

top ▾ Niveles predeterminados ▾ Sin problemas

- 5 mensajes [vite] connecting... [client:469](#)
- 5 mensajes ... [vite] connected. [client:574](#)
- 1 error ▶ Warning: [antd: Menu] `children` is deprecated. Please use `items` instead. [chunk-ZLTZVGJ.js?v=f9ac28bc:1691](#)
- Sin adverten... 2 Token --> : [AuthContext.jsx:29](#)
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY2NWUzNmQ5MDU1MjdlYWIXNTYyNTlhMCIsImhhdCI6MTcxNzQ2MzYyMywiZXBhIjoxNzE3NTUwMDIzfkQ._C09iZ63bSggJPtysPA7UV0nWUy4a4kkI8_y5YHlffQ
- 2 mensajes ... ▶ |