comment utiliser axios dans un slice redux toolkit

**1.Installer Axios :** Assurez-vous d'avoir Axios installé dans votre projet. Si ce n'est pas le cas, vous pouvez l'installer en utilisant

npm install axios

**2.Importer Axios dans votre fichier slice :**

import axios from 'axios';

**3.Définir des actions asynchrones :**

Utilisez createAsyncThunk pour définir des actions asynchrones qui effectueront des appels API à l'aide d'Axios.

import { createSlice, createAsyncThunk } from '@reduxjs/toolkit';

// Créer une action asynchrone avec createAsyncThunk

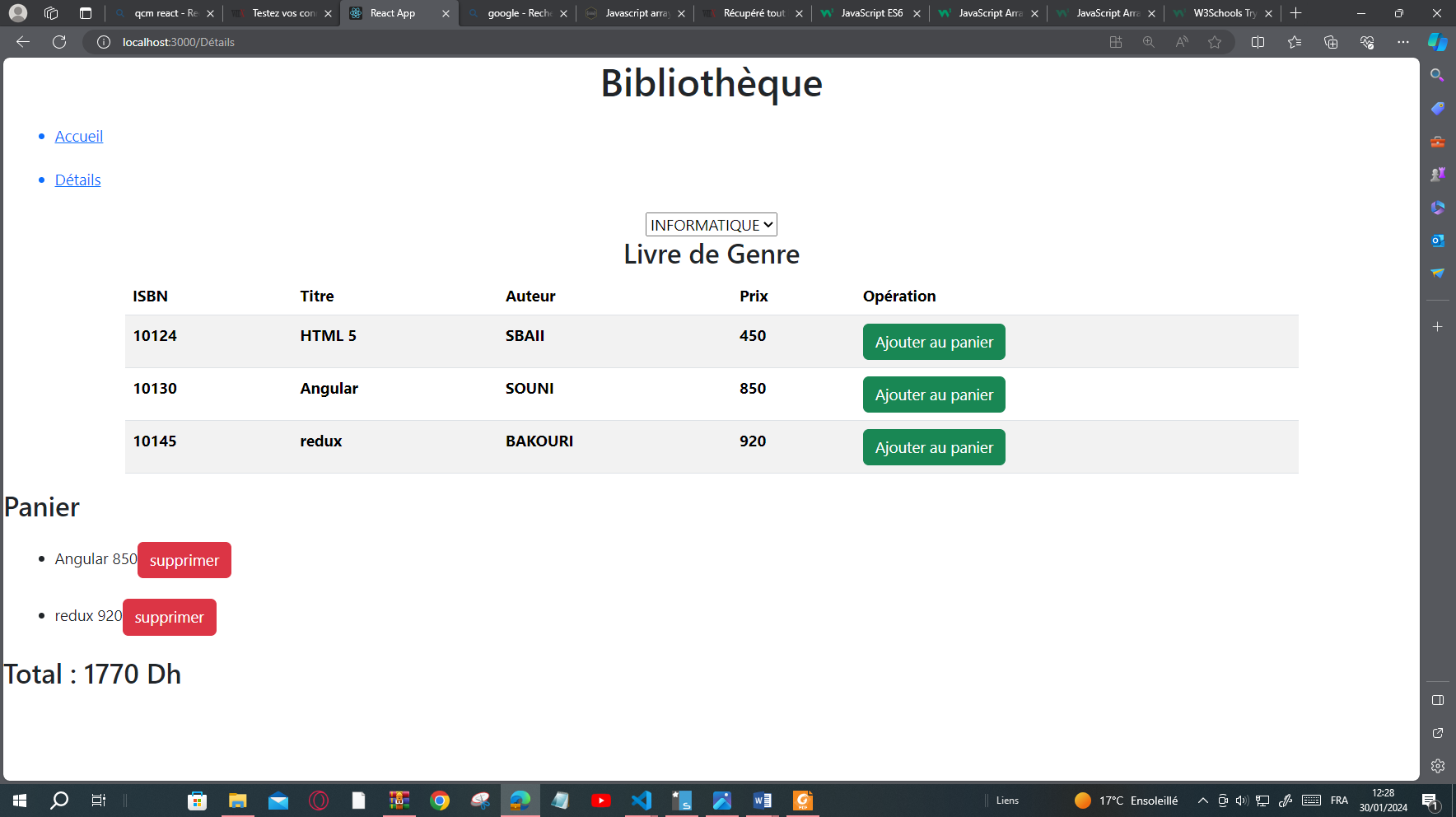
export const fetchData = createAsyncThunk('sliceName/fetchData', async () => {

const response = await axios.get('https://api.example.com/data');

return response.data;

});

TP AVEC (API)



Solution 2 avec API

Creation slice : BiblioSlice.js

import { createSlice, createAsyncThunk } from '@reduxjs/toolkit';

import axios from 'axios';

export const fetchData = createAsyncThunk('BiblioSlice/fetchData', async () => {

    const response = await axios.get('http://localhost:3030/livres');

    return response.data;

  });

const initState = {

    datalivres: [],

}

const BiblioSlice = createSlice({

    name:"BIBLIO",

    initialState:initState,

    reducers:{},

  extraReducers: (builder) => {

    builder.addCase(fetchData.fulfilled, (state, action) => {

        state.datalivres = action.payload;

      })

  },

})

export default BiblioSlice.reducer;

creation store:Store.js

import { configureStore } from "@reduxjs/toolkit";

import BiblioSlice from "./BiblioSlice";

const Store=configureStore({

reducer:{

    Biblio:BiblioSlice,

},

})

export default Store

Livre.js

import { useEffect, useState } from 'react';

import { useDispatch, useSelector } from "react-redux";

import {Link } from "react-router-dom";

import { fetchData } from './BiblioSlice';

function Livre(){

    const [genre,setGenre]=useState("INFORMATIQUE");

    const [Tpanier,setTpanier]=useState([]);

    const dispatch = useDispatch();

    const listLivre = useSelector((state) => state.Biblio.datalivres);

    useEffect(() => {

      dispatch(fetchData());

    }, [dispatch]);

    //let listLivre = useSelector(state => state.Biblio.livres );

    let listLivre2=listLivre.filter(liv=>liv.Genre==genre)

    console.log(listLivre2)

    const TGenre =  listLivre.map(liv => liv.Genre);

   let  TGenre2 = [...new Set(TGenre)]

    console.log(TGenre2)

    const totalPrix=Tpanier.reduce(function(total,liv){

        return total+=liv.Prix

        },0);

    const supp = (isbn) =>{

        const T=[...Tpanier.filter((liv)=>{return liv.ISBN!==isbn })]

        window.confirm('vous êtes sûr de supprimer ?') && setTpanier(T)

    }

    const ajouter=(isbn)=>{

        const T=[...Tpanier];

      const Liv=  listLivre.find(liv=>liv.ISBN==isbn)

        T.push(Liv)

        setTpanier(T)

    }

    return (

        <div>

            <center>

      <select onChange={(ev)=>setGenre(ev.target.value)} value={genre}>

     { TGenre2.map((gen)=><option>{gen}</option>)}

      </select>

        <h3>Livre de Genre</h3>

            <table width="60%" className=" container table table-striped">

                <thead>

                    <tr><th>ISBN</th><th>Titre</th><th>Auteur</th><th>Prix</th><th>Opération</th></tr>

                </thead>

                <tbody>

                    {

                       listLivre2.map((liv,index) =>

                           <tr key={index}>

                           <th>{liv.ISBN}</th>

                            <th>{liv.Titre}</th>

                            <th>{liv.Auteur}</th>

                            <th>{liv.Prix}</th>

                            <th>

                           <button className="btn btn-success"  onClick={()=>ajouter(liv.ISBN)} >Ajouter au panier</button>

                            </th>

                           </tr>

                            )

                    }

                </tbody>

                </table></center>

                <h3>Panier</h3>

                <ul>

               {Tpanier.map((liv) =>

                         <li>{liv.Titre} {liv.Prix}

                           <button className="btn btn-danger"  onClick={()=>supp(liv.ISBN)} >supprimer</button>

                           <br/>  </li>

                           )}

                           </ul>

          <h3>Total :  {totalPrix}</h3>

        </div>

     );

}

export default Livre;

app.js

import { BrowserRouter, Route, Routes } from "react-router-dom";

import {Link } from "react-router-dom";

import './App.css';

import Accueil from "./Accueil"

import Livre from "./Livre";

function App() {

    return (

        <BrowserRouter>

        <h1 align="center">Bibliothèque</h1>

           <ul>

                <Link end to='/'><li>Accueil</li></Link>

                <Link to='/Détails'><li>Détails</li></Link>

            </ul>

        <Routes>

            <Route path="/" element={<Accueil />} />

            <Route path="/Détails/" element={<Livre />}/>

        </Routes>

        </BrowserRouter>

     );

}

export default App;

index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import Store1  from './Store';

import { Provider } from 'react-redux';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<Provider store={Store1}>  <App/></Provider>

);

App.css

.App {

  text-align: center;

}

.App-logo {

  height: 40vmin;

  pointer-events: none;

}

@media (prefers-reduced-motion: no-preference) {

  .App-logo {

    animation: App-logo-spin infinite 20s linear;

  }

}

.App-header {

  background-color: #282c34;

  min-height: 100vh;

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: center;

  font-size: calc(10px + 2vmin);

  color: white;

}

.App-link {

  color: #61dafb;

}

@keyframes App-logo-spin {

  from {

    transform: rotate(0deg);

  }

  to {

    transform: rotate(360deg);

  }

}

 ul li {

  margin: 20px;

}

Index.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="utf-8" />

    <link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet">

    <meta name="viewport" content="width=device-width, initial-scale=1" />

    <meta name="theme-color" content="#000000" />

    <meta

      name="description"

      content="Web site created using create-react-app"

    />

    <link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

    <!--

      manifest.json provides metadata used when your web app is installed on a

      user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/

    -->

    <link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

    <!--

      Notice the use of %PUBLIC\_URL% in the tags above.

      It will be replaced with the URL of the `public` folder during the build.

      Only files inside the `public` folder can be referenced from the HTML.

      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will

      work correctly both with client-side routing and a non-root public URL.

      Learn how to configure a non-root public URL by running `npm run build`.

    -->

    <title>React App</title>

  </head>

  <body>

    <noscript>You need to enable JavaScript to run this app.</noscript>

    <div id="root"></div>

    <!--

      This HTML file is a template.

      If you open it directly in the browser, you will see an empty page.

      You can add webfonts, meta tags, or analytics to this file.

      The build step will place the bundled scripts into the <body> tag.

      To begin the development, run `npm start` or `yarn start`.

      To create a production bundle, use `npm run build` or `yarn build`.

    -->

  </body>

</html>