FRONT END FULL STACK 201 - 202 -205 -206 IFMOTICA FES

***REACT TP API***

Objectif: utilisé API Locale –créer une application qui utilise un api local (exécution des requêtes get-post-put-délete)

Soit un api local suivant :

{

    "Articles": [

      {

        "id": 1,

        "nomA": "PC",

        "pu": 2500,

        "Qt":15

      },

      {

        "id": 2,

        "nomA": "TABLET",

        "pu": 4500,

        "Qt":150

      },

      {

        "id": 3,

        "nomA": "TV",

        "pu": 12500,

        "Qt":24

      },

      {

        "id": 4,

        "nomA": "SMART PHONE",

        "pu": 3500,

        "Qt":18

      }

    ]

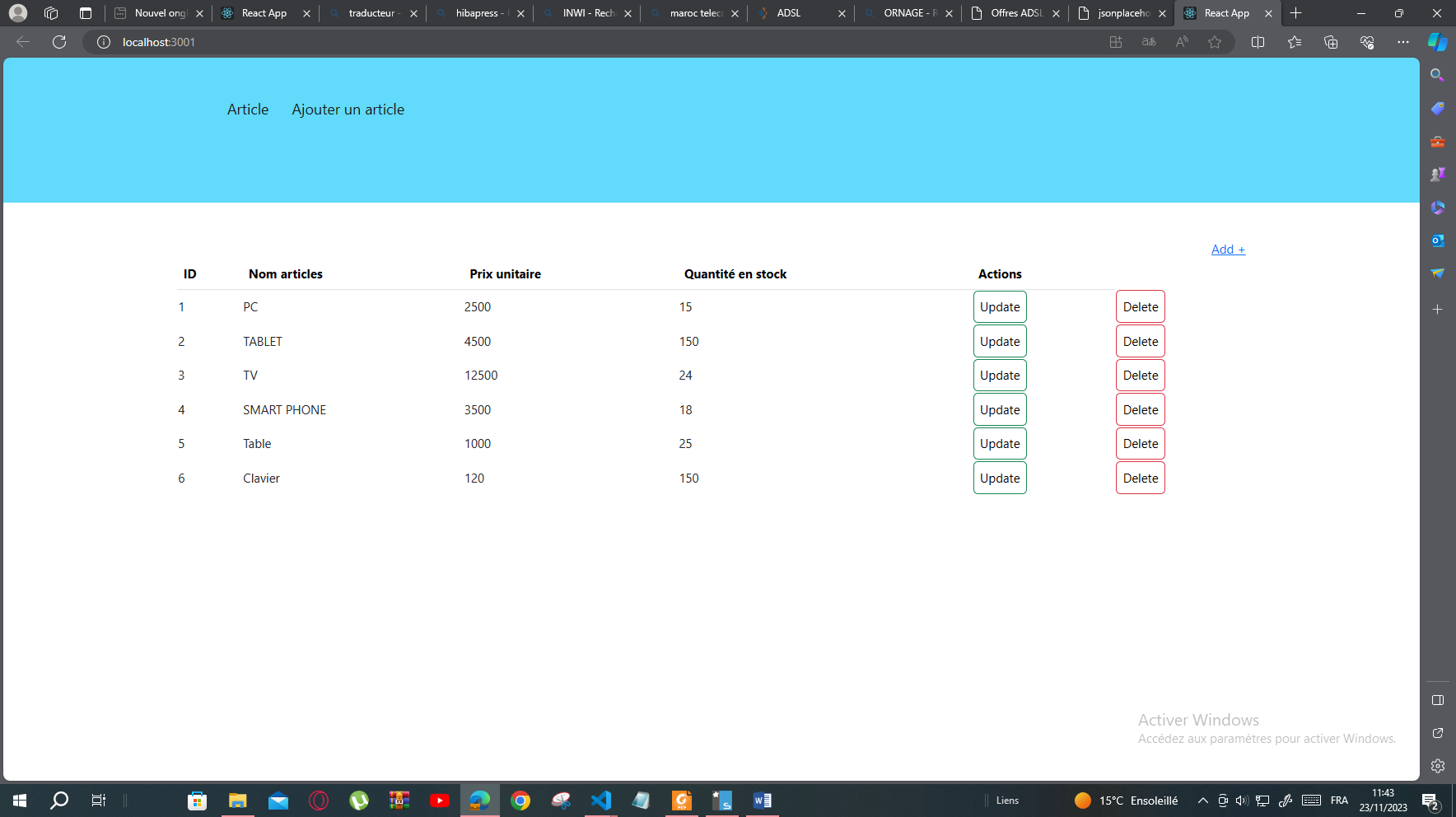
  }

Etape 1 : installer le serveur local json :

npm install json-server

Etape 1 : lancer server json local

npx json-server --watch db.json --port 3030



AppRouter.js :

import React from 'react'

import Article from './Article'

import { Routes,Link, Route, BrowserRouter } from 'react-router-dom'

import AjoutAr from './AjoutAr'

export default function AppRouter1() {

  return (

    <>

    <div class="scrollmenu">

    <div class="div1">

<Link to='/'>Article</Link>

<Link to='/create'>Ajouter un article</Link>

</div></div>

<div class="scrollmenu2">

<Routes>

<Route path='/' element={<Article/>}></Route>

<Route path='/create' element={<AjoutAr/>}></Route>

</Routes>

</div>

</>

)

}

Article .js

import React,{useState,useEffect} from 'react'

import axios from 'axios'

import {Link, useNavigate} from 'react-router-dom'

import { hasSelectionSupport } from '@testing-library/user-event/dist/utils'

export default function Article() {

    const [tarticles,setTarticles]=useState([])

    const navigation=useNavigate();

   const sup=(idsup)=>{

    const conf=window.confirm('oui/non')

    if(conf){

axios.delete('http://localhost:3030/articles/'+idsup).then((res)=>{

  alert('bien supprimé')

   navigation('/');

}).catch(err=>alert(err))

}

   }

    useEffect(()=>{

        axios.get('http://localhost:3030/articles')

        .then((res)=>{ console.log(res.data);setTarticles(res.data)})

        })

        return(

        <div className='container mt-5'>

        <div className='text-end'><Link to='create'>Add +</Link></div>

<table className='table'>

    <thead>

    <tr><th>ID</th><th>Nom articles</th><th>Prix unitaire</th><th>Quantité en stock</th><th>Actions</th></tr>

    </thead>

        {tarticles.map(Art=>{

        return(

            <tr><td>{Art.id} </td><td>{Art.nomA}</td><td>{Art.pu}</td><td>{Art.Qt}</td>

           <td><Link to='create' className='btn btn-success'>Update</Link></td>

           <td><button  className='btn btn-danger' onClick={(ev)=>sup(Art.id)}>Delete</button></td> </tr>

        )

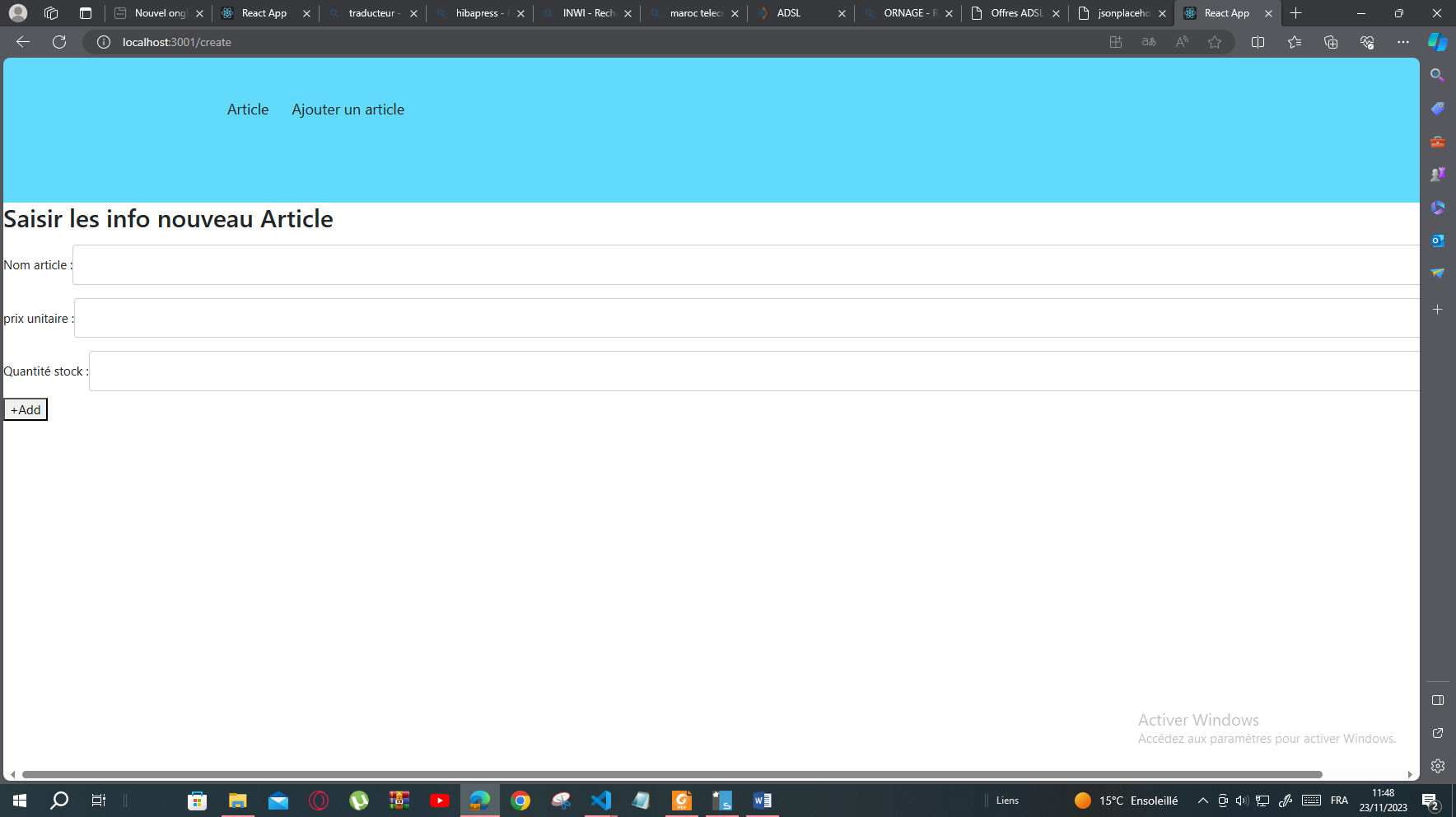
        })}

        </table>

       </div>

        )

        }



AjouteAr.js

import React,{useState,useEffect} from 'react'

import axios from 'axios'

import { useNavigate } from 'react-router-dom'

export default function AjoutAr() {

    const navigation=useNavigate();

    const [tarticles,setTarticles]=useState([])

    const [nomA,setNomA]=useState('')

    const [pu,setPu]=useState('')

    const [qt,setQt]=useState('')

    const  ajout=()=>{

        axios.post('http://localhost:3030/articles',{nomA:nomA,pu:pu,Qt:qt})

        .then((res)=>{ alert('bien ajouter')

           }).catch(err=>alert('erreur ajout'))

       setNomA('');

       setPu('');

       setQt('');

       navigation('/')

      }

  return (

    <div ><h2>Saisir les info nouveau Article</h2>

<label for="fname">  Nom article   : </label>

<input type='text' value={nomA} onChange={ev=>setNomA(ev.target.value)}></input><br></br>

<label for="fname">  prix unitaire   : </label>

<input type='text' value={pu} onChange={ev=>setPu(ev.target.value)}></input><br></br>

<label for="fname">  Quantité stock     : </label>

<input type='text' value={qt} onChange={ev=>setQt(ev.target.value)}></input><br></br>

       <button onClick={ajout}> +Add </button>

       </div>

  )

}

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>

Index.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);

  }

}

div.scrollmenu {

  background-color:#61dafb;

  overflow: auto;

  white-space: nowrap;

  position:absolute ;top:0% ;left:0%;width:100%;height:20%;float:center;

}

div.scrollmenu2 {

  background-color: #fff;

  overflow: auto;

  white-space: nowrap;

  position:absolute ;top:20% ;left:0%;width:100%;height:80%;float:center;

}

.div1{

    position:absolute ;top:20%; left:15%;width:70%;height:20%;float:center;

}

.div2{

  position:absolute ;top:10%; left:75%;width:20%;height:20%;float:right;

}

div.scrollmenu a {

  display: inline-block;

  color: #181717;

  text-align: center;

  padding: 14px;

  text-decoration: none;

 font-size: larger;

}

div.scrollmenu a:hover {

  background-color: #181717 ;

  color:#faf6f6;

}