Creacion de Back

Instalación

npm init -y //Crea los package

npm i express mysql2 morgan //Modulos Morgan muestra los envios en consola

mpm i nodemon

Creacion del scrip de nodemon

{

  "name": "server",

  "version": "1.0.0",

  "description": "",

  "main": "index.js",

  "scripts": {

    "corre":"nodemon server/index.js",

    "test": "echo \"Error: no test specified\" && exit 1"

  },

  "keywords": [],

  "author": "",

  "license": "ISC",

  "dependencies": {

    "express": "^4.18.2",

    "morgan": "^1.10.0",

    "mysql2": "^3.6.1",

    "nodemon": "^3.0.1"

  }

}

Ejecucion

npm run corre

Codio del index

Importación

Se debe crear el tipo para hacer importaciones tipo react

{

  "name": "server",

  "version": "1.0.0",

  "description": "",

  "main": "index.js",

  "type": "module",

  "scripts": {

    "corre":"nodemon server/index.js",

    "test": "echo \"Error: no test specified\" && exit 1"

  },

  "keywords": [],

  "author": "",

  "license": "ISC",

  "dependencies": {

    "express": "^4.18.2",

    "morgan": "^1.10.0",

    "mysql2": "^3.6.1",

    "nodemon": "^3.0.1"

  }

}

Index

import express  from "express";//Modulo express

import { Port } from "./config.js";//importación del puerto

const app=express();//Asinar variable

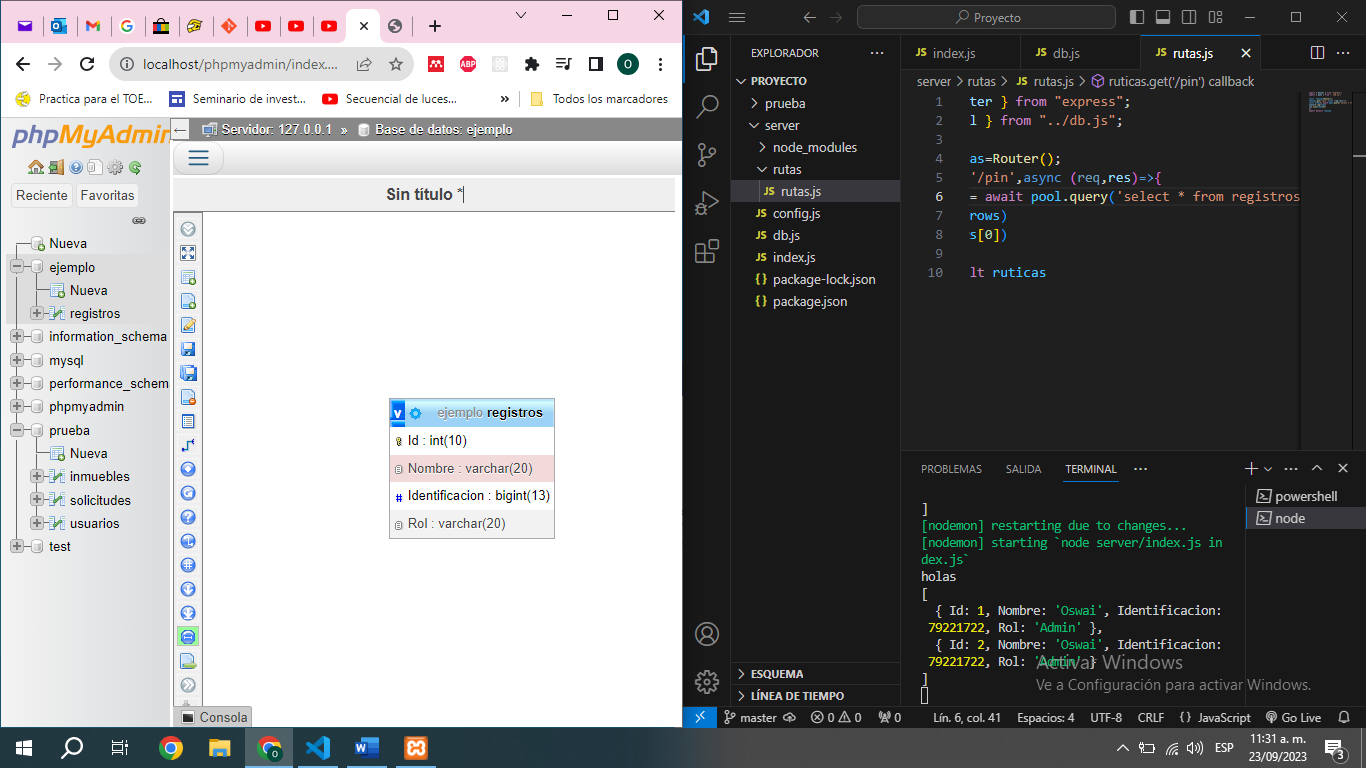
app.listen(Port)//Escuchando el puerto

console.log("holas");

Confi

export const Port =8000;//# del puerto

Preparando la bbdd



Acceso a la BBDD (archivo db)

import {createPool} from 'mysql2/promise';

export const pool=createPool({

    host:"localhost",

    port:3306,

    user:"root",

    password:'',

    database:'ejemplo'

})

Generando las rutas (archivo rutas)

import { Router } from "express";

import { pool } from "../db.js";//Exporta la bbdd

const  ruticas=Router();

ruticas.get('/pin',async (req,res)=>{//el acceso lo hace con el /

const [rows]= await pool.query('select \* from registros' )//Consulta el array es para que no me muestre toda la confg del Dom

console.log(rows)

res.json(rows[0])//respuesta en nformata json

})

export default ruticas

Index

import express  from "express";

import { Port } from "./config.js";

import ruticas from "./rutas/rutas.js";//traemos las rutas

const app=express();

app.use(ruticas);//usamos las rutas

app.listen(Port)

console.log("holas");

Creando las rutas

Creamos las tareas(Fichero tareas)

import { Router } from "express";

const router=Router();

router.get('/tarea')

router.post('/tarea')//crea tarea

router.put('/tarea/:id')//parametro cualquiera

router.get('/tarea:id')//Parametro unico

router.delete('/tarea/:id')//Elimina tarea

export default router

Creamos el controlador de tareas(Fichero ControladorTareas)

export const getTareas=(req,res)=>{

    res.send('obteniendo  tareas')

}

export const getTarea=(req,res)=>{

    res.send('obteniendo una tareas')

}

export const createTareas=(req,res)=>{

    res.send('creando tareas')

}

export const updateTareas=(req,res)=>{

    res.send('actualizando tareas')

}

export const deleteTareas=(req,res)=>{

    res.send('eliminando tareas')

}

Exportamos a las tareas

import { Router } from "express";

import { getTarea,getTareas,createTareas,updateTareas,deleteTareas } from "../Controlador/ControladorTareas.js";

const router=Router();

router.get('/tarea',getTareas)

router.post('/tarea',createTareas)//crea tarea

router.put('/tarea/:id',createTareas)//parametro cualquiera

router.get('/tarea:id',getTarea)//Parametro unico

router.delete('/tarea/:id',deleteTareas)//Elimina tarea

export default router

Añadimos al index

import express  from "express";

import { Port } from "./config.js";

import ruticas from "./rutas/rutas.js";

import tareas from "./rutas/tareas.js"

const app=express();

app.use(ruticas);

app.use(tareas);

app.listen(Port)

console.log("holas");

La consulta será <http://localhost:8000/tarea>

Realizando las conexiones

Insertando(contralodarTareas)

import {pool} from '../db.js'

export const getTareas=(req,res)=>{

    res.send('obteniendo  tareas')

}

export const getTarea=(req,res)=>{

    res.send('obteniendo una tareas')

}

export const createTareas= async (req,res)=>{

    const {Nombre,Identificacion,Rol}= req.body

    const [result]=await pool.query('Insert into registros(Nombre,Identificacion,Rol) values (?,?,?)',[Nombre,Identificacion,Rol])

    res.json({

       Id:result.insertId, Nombre,Identificacion,Rol

    })

}

export const updateTareas=(req,res)=>{

    res.send('actualizando tareas')

}

export const deleteTareas=(req,res)=>{

    res.send('eliminando tareas')

}

Obteniendo tareas

import {pool} from '../db.js'

export const getTareas= async (req,res)=>{

   const [result] = await pool.query('select \* from registro')

    res.json(result)

}

export const getTarea= async (req,res)=>{

    const [result] = await pool.query('select \* from registro where id=?',[req.params.Id])//Requerimiento de parametro en este caso el ID

    res.json(result)

}

export const createTareas= async (req,res)=>{

    const {Nombre,Identificacion,Rol}= req.body

    const [result]=await pool.query('Insert into registros(Nombre,Identificacion,Rol) values (?,?,?)',[Nombre,Identificacion,Rol])

    res.json({

       Id:result.insertId, Nombre,Identificacion,Rol

    })

}

export const updateTareas=(req,res)=>{

    res.send('actualizando tareas')

}

export const deleteTareas=(req,res)=>{

    res.send('eliminando tareas')

}

Actualizar y borrar

import {pool} from '../db.js'

export const getTareas= async (req,res)=>{

   const [result] = await pool.query('select \* from registro')

    res.json(result)

}

export const getTarea= async (req,res)=>{

    const [result] = await pool.query('select \* from registro where id=?',[req.params.Id])//Requerimiento de parametro en este caso el ID

    res.json(result)

}

export const createTareas= async (req,res)=>{

    const {Nombre,Identificacion,Rol}= req.body

    const [result]=await pool.query('Insert into registros(Nombre,Identificacion,Rol) values (?,?,?)',[Nombre,Identificacion,Rol])

    res.json({

       Id:result.insertId, Nombre,Identificacion,Rol

    })

}

export const updateTareas=async (req,res)=>{

    const [result]=await pool.query('update  registro set ? where id=?',[req.body,req.params.Id])

    res.json(result)

}

export const deleteTareas=async (req,res)=>{

    const [result]=await pool.query('delete  from registro where id=?',[req.params.Id])

    res.json(result)

}

Combatiendo el error de no encontrar registros

Manejo de excepciones

Try(intentar)

Catch(Captura de error)

import {pool} from '../db.js'

export const getTareas= async (req,res)=>{

   try{

    const [result] = await pool.query('select \* from registro')

    res.json(result)

   }

   catch(error){

    return res.status(500).json({mensaje:error.mensaje})

   }

}

export const getTarea= async (req,res)=>{

try{

    const [result] = await pool.query('select \* from registro where id=?',[req.params.Id])//Requerimiento de parametro en este caso el ID

    res.json(result)

}

catch(error){

    return res.status(500).json({mensaje:error.mensaje})

   }

}

export const createTareas= async (req,res)=>{

    try{

        const {Nombre,Identificacion,Rol}= req.body

    const [result]=await pool.query('Insert into registros(Nombre,Identificacion,Rol) values (?,?,?)',[Nombre,Identificacion,Rol])

    res.json({

       Id:result.insertId, Nombre,Identificacion,Rol

    })

    }

    catch(error){

        return res.status(500).json({mensaje:error.mensaje})

       }

}

export const updateTareas=async (req,res)=>{

  try{

    const [result]=await pool.query('update  registro set ? where id=?',[req.body,req.params.Id])

    res.json(result)

  }

  catch(error){

    return res.status(500).json({mensaje:error.mensaje})

   }

}

export const deleteTareas=async (req,res)=>{

    try{

        const [result]=await pool.query('delete  from registro where id=?',[req.params.Id])

    res.json(result)

    }

    catch(error){

        return res.status(500).json({mensaje:error.mensaje})

       }

}

Index

import express  from "express";

import { Port } from "./config.js";

import ruticas from "./rutas/rutas.js";

import tareas from "./rutas/tareas.js"

const app=express();

app.use(express.json())

app.use(ruticas);

app.use(tareas);

app.listen(Port)

console.log("holas");

Frontend

Limpiar el react(Solo dejamos el App y el index)

App

function App() {

  return (

    <div >

      Holas

    </div>

  );

}

export default App;

Index

import React from 'react';

import ReactDOM from 'react-dom/client';

import App from './App';

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

root.render(

    <App />

);

Importante instalar el react-router-dom

Generamos el enrutamiento

Index

import React from 'react';

import ReactDOM from 'react-dom/client';

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

import App from './App';

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

root.render(

  <BrowserRouter>

    <App />

  </BrowserRouter>

);

App

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

function App() {

  return (

    <Routes>

      <Route path="/" />

    </Routes>

  );

}

export default App;

Creamos la carpeta pages y el archivo de tareas

Tarea(Fichero tareas)

function Tarea(){

return(

    <div>

        JAJJAJAJ

    </div>

)

}

export default Tarea

App

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

import Tarea from "./pages/tareas.js";

function App() {

  return (// la ruta / es la raiz

    <Routes>

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

    </Routes>

  );

}

export default App;

Creando mas page

Nofound

function NO(){

    return(

        <div>

        Pagina no encontrada

    </div>

    )

}

export default NO

Form

function Formu(){

    return(

        <div>

        ppppppppp

    </div>

    )

}

export default Formu

App

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

import Tarea from "./pages/tareas.js";

import Formu from "./pages/Form.js";

import NO from "./pages/Nofound.js";

function App() {

  return (// la ruta / es la raiz

    <Routes>

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

      <Route path="/new" element={<Formu/>} />

      <Route path="\*" element={<NO/>} />//El \* hace referencia a todas las pages no creadas

    </Routes>

  );

}

export default App;

Creando Navegabilidad

Creamos la carpeta components y creamos el nav

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

function Navegar(){

    return(

        <div>

            <h1>

                Mi pagina

            </h1>

            <ul>

                <li>

                    <Link to="/">Home</Link>

                </li>

                <li>

                    <Link to="/new">Crear usuario</Link>

                </li>

            </ul>

        </div>

    )

}

export default Navegar

App

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

import Tarea from "./pages/tareas.js";

import Formu from "./pages/Form.js";

import NO from "./pages/Nofound.js";

import Navegar from "./components/nav.js"

function App() {

  return (// la ruta / es la raiz

  <> // este tag genera el orden en que muestra los componentes

  <Navegar/>

    <Routes>

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

      <Route path="/new" element={<Formu/>} />

      <Route path="\*" element={<NO/>} />

    </Routes>

  </>

  );

}

Creando formularios

Utilizaremos formik

npm install formik –save

Creamos el formulario

import { Form,Formik } from "formik"

function Formu(){

    return(

    <div>

       <Formik

       initialValues={{

        Nombre:"",

        Identificacion:"",

        Rol:""

       }}

       >

        <Form>

            <label >Nombre</label>

            <input type="text" name="Nombre"/>

            <label>Identificacion</label>

            <input type="text" name="Identificacion"/>

            <label>Rol</label>

            <select name="Rol">

                <option value="Usuario">Usuario</option>

                <option value="Administrador">Administrador</option>

            </select>

            <button>Enviar</button>

        </Form>

       </Formik>

    </div>

    )

}

export default Formu

Capturando los datos del formulario se utilizan cosas de JSX

import { Form,Formik } from "formik"

function Formu(){

    return(

    <div>

       <Formik

       initialValues={{

        Nombre:"",

        Identificacion:"",

        Rol:""

       }}

       onSubmit={(values)=>{

        console.log(values)

       }}

       >

       {({handleChange,handleSubmit})=>(

         <Form onSubmit={handleSubmit}>

         <label >Nombre</label>

         <input type="text" name="Nombre" onChange={handleChange}/>

         <label>Identificacion</label>

         <input type="text" name="Identificacion" onChange={handleChange}/>

         <label>Rol</label>

         <select name="Rol" onChange={handleChange}>

             <option value="Usuario" onChange={handleChange}>Usuario</option>

             <option value="Administrador" onChange={handleChange}>Administrador</option>

         </select>

         <button type="submit">Enviar</button>

     </Form>

       )}

       </Formik>

    </div>

    )

}

export default Formu

Comunicando con el backend

Instalamos axios (Permite la comunicación Http)

Creamos la carpeta Api y dentro las rutas

Apirutas

import  axios from 'axios';

export const Crea=async (task)=>{

    await axios.post('http://localhost:8000/tarea',task)

}

Form

import { Form,Formik } from "formik";

import {Crea} from "../Api/Apirutas.js"

function Formu(){

    return(

    <div>

       <Formik

       initialValues={{

        Nombre:"",

        Identificacion:"",

        Rol:""

       }}

       onSubmit={async (values)=>{

        console.log(values)

        try{//envio de datos

            const respuest= await Crea(values)

            console.log(respuest)

        }

        catch(error){

            console.error(error)

        }

       }}

       >

       {({handleChange,handleSubmit})=>(

         <Form onSubmit={handleSubmit}>

         <label >Nombre</label>

         <input type="text" name="Nombre" onChange={handleChange}/>

         <label>Identificacion</label>

         <input type="text" name="Identificacion" onChange={handleChange}/>

         <label>Rol</label>

         <select name="Rol" onChange={handleChange}>

             <option value="Usuario" onChange={handleChange}>Usuario</option>

             <option value="Administrador" onChange={handleChange}>Administrador</option>

         </select>

         <button type="submit">Enviar</button>

     </Form>

       )}

       </Formik>

    </div>

    )

}

export default Formu

El error es debido a que tenemos dos servidores necesitamos inatlar Cors en el back

npm i cors

lo aplicamos en el index

import express  from "express";

import { Port } from "./config.js";

import ruticas from "./rutas/rutas.js";

import tareas from "./rutas/tareas.js"

import cors from 'cors'

const app=express();

app.use(express.json())

app.use(cors({ // Le decimos que solo se comunique con este servidor

    origin:'http://localhost:3000'

}));

app.use(ruticas);

app.use(tareas);

app.listen(Port)

console.log("holas");

Limpiando el formulario

import { Form,Formik } from "formik";

import {crea} from "../Api/Apirutas.js"

function Formu(){

    return(

    <div>

       <Formik

       initialValues={{

        Nombre:"",

        Identificacion:"",

        Rol:""

       }}

       onSubmit={async (values,actions)=>{

        console.log(values)

        try{

            const respuest= await crea (values)

            console.log(respuest)

            actions.resetForm()

        }

        catch(error){

            console.error(error)

        }

       }}

       >

       {({handleChange,handleSubmit,values})=>(

         <Form onSubmit={handleSubmit}>

         <label >Nombre</label>

         <input type="text" name="Nombre" onChange={handleChange} value={values.Nombre}/>

         <label>Identificacion</label>

         <input type="text" name="Identificacion" onChange={handleChange} value={values.Identificacion}/>

         <label>Rol</label>

         <select name="Rol" onChange={handleChange} value={values.Rol}>

             <option value="Usuario" onChange={handleChange} >Usuario</option>

             <option value="Administrador" onChange={handleChange}>Administrador</option>

         </select>

         <button type="submit">Enviar</button>

     </Form>

       )}

       </Formik>

    </div>

    )

}

export default Formu

Esperar hasta que el registro se envie para que no cree varios

import { Form,Formik } from "formik";

import {crea} from "../Api/Apirutas.js"

function Formu(){

    return(

    <div>

       <Formik

       initialValues={{

        Nombre:"",

        Identificacion:"",

        Rol:""

       }}

       onSubmit={async (values,actions)=>{

        console.log(values)

        try{

            const respuest= await crea (values)

            console.log(respuest)

            actions.resetForm()

        }

        catch(error){

            console.error(error)

        }

       }}

       >

       {({handleChange,handleSubmit,values,isSubmitting})=>(

         <Form onSubmit={handleSubmit}>

         <label >Nombre</label>

         <input type="text" name="Nombre" onChange={handleChange} value={values.Nombre}/>

         <label>Identificacion</label>

         <input type="text" name="Identificacion" onChange={handleChange} value={values.Identificacion}/>

         <label>Rol</label>

         <select name="Rol" onChange={handleChange} value={values.Rol}>

             <option value="Usuario" onChange={handleChange} >Usuario</option>

             <option value="Administrador" onChange={handleChange}>Administrador</option>

         </select>

         <button type="submit" disabled={isSubmitting}>

           {isSubmitting ? "Enviando...": "Enviar"} // es como un condicional

            </button>

     </Form>

       )}

       </Formik>

    </div>

    )

}

export default Formu

Listar los registros

Fichero tareas

import { useEffect,useState } from "react"

import { getTareas } from "../Api/Apirutas.js"

import TareaTarjeta from "../components/task.js"

function Tarea(){

    const [taks,setTask]=useState([])

    useEffect(()=>{//Apenas se carge inicia esta parte

        async function Load(){

            const respuesta=await getTareas()

            setTask(respuesta.data)

        }

        Load()

    },[])

return(

    <div>

        <h1> Los registros son</h1>

        {

        taks.map(t1=>(

         <TareaTarjeta

         t1={t1}

         key={t1.id}

         />

        )

        )

        }

    </div>

)

}

export default Tarea

Creamos la ruta

Fichero APIrutas

import  axios from 'axios';

export const crea=async (task)=>{

    await axios.post('http://localhost:8000/tarea',task)

}

export const getTareas=async ()=>

    await axios.get('http://localhost:8000/tarea');

Creamos el componente que nos muetre la información

Fichero task

function TareaTarjeta({t1}){

    return(

        <div>

        <h2>{t1.Nombre}</h2>

        <p>{t1.Identificacion}</p>

        <p>{t1.Rol}</p>

        <button>Borrar</button>

        <button>Editar</button>

        </div>

    )

}

export default TareaTarjeta

Elimminando archivos

Le asignamos valor al botón Borrar

import { getBorrar } from "../Api/Apirutas"

function TareaTarjeta({t1}){

const handleDelete=async(Id)=>{

    try{

        const respuesta=await getBorrar(Id)

        console.log(respuesta)

    }

    catch(error){

        console.error(error);

    }

}

return(

        <div>

        <h2>{t1.Nombre}</h2>

        <p>{t1.Identificacion}</p>

        <p>{t1.Rol}</p>

        <button onClick={()=>handleDelete(t1.Id)}>Borrar</button>

        <button>Editar</button>

        </div>

    )

}

export default TareaTarjeta

Creamos la ruta

import  axios from 'axios';

export const crea=async (task)=>{

    await axios.post('http://localhost:8000/tarea',task)

}

export const getTareas=async ()=>

    await axios.get('http://localhost:8000/tarea');

    export const getBorrar=async (Id)=>

    await axios.delete(`http://localhost:8000/tarea/${Id}`);

Y renderizamos cuando no hay registros

Fichero tareas

import { useEffect,useState } from "react"

import { getTareas } from "../Api/Apirutas.js"

import TareaTarjeta from "../components/task.js"

function Tarea(){

    const [taks,setTask]=useState([])

    useEffect(()=>{//Apenas se carge inicia esta parte

        async function Load(){

            const respuesta=await getTareas()

            setTask(respuesta.data)

        }

        Load()

    },[])

    function renderMin(){

        if (taks.length=== 0)return <h1>No hay registros</h1>

      return taks.map((t1)=>(<TareaTarjeta t1={t1} key={t1.id}/>) )

    }

return(

    <div>

        <h1> Los registros son</h1>

        {renderMin()}

    </div>

)

}

export default Tarea

Creando contexto

Creamos el contexto (Context.js)

import { createContext } from "react"

export const Mcontext=createContext()

Modificamos el componente task (task)

import { getBorrar } from "../Api/Apirutas"

import { useTare,Tcontext } from "../Context/TareasContext"

function TareaTarjeta({t1}){

const {handleDelete}=useTare()

return(

        <div>

        <h2>{t1.Nombre}</h2>

        <p>{t1.Identificacion}</p>

        <p>{t1.Rol}</p>

        <button onClick={()=>handleDelete(t1.Id)}>Borrar</button>

        <button>Editar</button>

        </div>

    )

}

export default TareaTarjeta

Modificamos las tareas(tareas)

import { useEffect,useState } from "react"

import { getTareas } from "../Api/Apirutas.js"

import TareaTarjeta from "../components/task.js"

import {useTare} from "../Context/TareasContext.js"

function Tarea(){

    const {taks,Load} =useTare()

    useEffect(()=>{//Apenas se carge inicia esta parte

        Load()

    },[])

    function renderMin(){

        if (taks.length=== 0)return <h1>No hay registros</h1>

      return taks.map((t1)=>(<TareaTarjeta t1={t1} key={t1.id}/>) )

    }

return(

    <div>

        <h1> Los registros son</h1>

        {renderMin()}

    </div>

)

}

export default Tarea

Modificamos el Api

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

import Tarea from "./pages/tareas.js";

import Formu from "./pages/Form.js";

import NO from "./pages/Nofound.js";

import Navegar from "./components/nav.js"

import { Tcontext } from "./Context/TareasContext.js";

function App() {

  return (// la ruta / es la raiz

  <Tcontext>

  <Navegar/>

    <Routes>

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

      <Route path="/new" element={<Formu/>} />

      <Route path="\*" element={<NO/>} />

    </Routes>

  </Tcontext>

  );

}

export default App;

Modificamos el Formulario(Form)

import { Form,Formik } from "formik";

import { useTare } from "../Context/TareasContext";

function Formu(){

const {createTarea}=useTare()

    return(

    <div>

       <Formik

       initialValues={{

        Nombre:"",

        Identificacion:"",

        Rol:""

       }}

       onSubmit={async (values,actions)=>{

        console.log(values)

        createTarea(values)

        actions.resetForm()

       }}

       >

       {({handleChange,handleSubmit,values,isSubmitting})=>(

         <Form onSubmit={handleSubmit}>

         <label >Nombre</label>

         <input type="text" name="Nombre" onChange={handleChange} value={values.Nombre}/>

         <label>Identificacion</label>

         <input type="text" name="Identificacion" onChange={handleChange} value={values.Identificacion}/>

         <label>Rol</label>

         <select name="Rol" onChange={handleChange} value={values.Rol}>

             <option value="Usuario" onChange={handleChange} >Usuario</option>

             <option value="Administrador" onChange={handleChange}>Administrador</option>

         </select>

         <button type="submit" disabled={isSubmitting}>

           {isSubmitting ? "Enviando...": "Enviar"}

            </button>

     </Form>

       )}

       </Formik>

    </div>

    )

}

export default Formu

Creamos las tareas del contexto(TareasContext)

import {  useContext, useState } from "react";

import { getTareas,getBorrar,crea } from "../Api/Apirutas.js";

import { Mcontext } from "./Context.js";

export const useTare=()=>{

    const contexto=useContext(Mcontext)

    if(!contexto){

        throw new Error("Falla en el contexto")

    }

    return contexto

}

export const createTarea=async(taks)=>{

    try{

        const respuest= await crea (taks)

        console.log(respuest)

    }

    catch(error){

        console.error(error)

    }

}

export const Tcontext=({children})=>{

    const [taks,setTask]=useState([])

    async function Load(){

        const respuesta=await getTareas()

        setTask(respuesta.data)

    }

    const handleDelete=async(Id)=>{

        try{

            const respuesta=await getBorrar(Id)

            setTask(taks.filter(t1=>t1.Id!==Id));

        }

        catch(error){

            console.error(error);

        }

    }

     return <Mcontext.Provider value={{taks,Load,handleDelete,createTarea}}>

        {children}

    </Mcontext.Provider>

}