CRUD EN REACT CON REACT HOOK FORMS

En este proyecto implementaremos operaciones básicas de bases de datos y utilizaremos para la comunicación de los datos React Hook Forms y usaremos props que como vimos es una forma de pasar información de nuestro App.js a un componente que tenga la extensión .jsx.

1.- Creamos nuestro proyecto al que llamaremos myapp6.

2.- Instalamos en la carpeta del proyecto los react hook forms el comando <https://react-hook-form.com/get-started>

npm install react-hook-form

3.- Instalaremos una librería para crear ID aleatorios esto para poder identificar los registros de nuestras colecciones.

La documentación esta en el sitio <https://www.npmjs.com/package/uuid>

npm install uuid

4.- Editamos los archivos de nuestro proyecto para preparar nuestro App.js

import React from 'react';

function App() {

  return (

    <div><h1>Proyecto CRUD</h1></div>

  );

}

export default App;

5.- Para la interface de nuestro proyecto usaremos un archivo de un repositorio <https://taniarascia.github.io/primitive/css/main.css> lo copiamos y lo remplazamos en nuestro index.css

6.- Maquetamos nuestro proyecto y probamos que el css este funcionando con el siguiente código nota recordamos que en react las class del css se llaman className

function App() {

  return (

    <div className="container">

      <h1>Proyecto CRUD</h1>

      <div className="flex-row">

        <div className="flex-large">

          <h2>Add user</h2>

        </div>

        <div className="flex-large">

          <h2>View users</h2>

        </div>

      </div>

    </div>

  );

}

7.- Ahora vamos a crear nuestra carpeta compentes y el primero que vamos a realizar se llamara UserTable.jsx usaremos Props. Dentro pondremos un código para mostrar una tabla. Usamos las abreviaturas imr para importar y sfc para crear nuestro hook y este lo agregamos en nuesto App.js debajo del View users

import React from 'react';

const UserTable = () => {

    return (

        <table>

            <thead>

            <tr>

                <th>Name</th>

                <th>Username</th>

                <th>Actions</th>

            </tr>

            </thead>

            <tbody>

            <tr>

                <td>Name data</td>

                <td>Username data</td>

                <td>

                <button className="button muted-button">Edit</button>

                <button className="button muted-button">Delete</button>

                </td>

            </tr>

            </tbody>

        </table>

    );

}

export default UserTable;

8.- En el App.js Vamos a crear un array de objetos que serán los elementos que usaremos como ejemplo. Usaremos useState y set entonces agregaremos a nuestra librería el { useState }

import React, { useState } from 'react';

import UserTable from './components/UserTable';

function App() {

  const usersData = [

    { id: 1, name: 'Tania', username: 'floppydiskette' },

    { id: 2, name: 'Craig', username: 'siliconeidolon' },

    { id: 3, name: 'Ben', username: 'benisphere' },

  ]

  const [users, setUsers] = useState(usersData);

  return (

    <div className="container">

      <h1>Proyecto CRUD</h1>

      <div className="flex-row">

      <div className="flex-large">

          <h2>Add user</h2>

        </div>

        <div className="flex-large">

          <h2>View users</h2>

          <UserTable />

        </div>

      </div>

    </div>

  );

}

export default App;

si guardamos y checamos en la inspección nuestro componente ya están los datos.

9.- Muy importante ahora usaremos props y estos se usaran en nuestro components llamado UserTable

9.1 Agregar el props

<UserTable users={users}/>

9.2 Mandar el props

const UserTable = (props) => {

9.3 Probar que este recibiendo e imprimir en la tabla

console.log(props.users);

<tbody>

                {

                    props.users.map(user => (

                        <tr key={user.id}>

                            <td>{user.name}</td>

                            <td>{user.username}</td>

                            <td>

                            <button className="button muted-button">Edit</button>

                            <button className="button muted-button">Delete</button>

                            </td>

                        </tr>

                    ))

                }

            </tbody>

10.- Ahora si vamos a ocupar nuestros id dinámicos la librería uuid que instalamos y esto se hace importando la librería y usando la función como lo manda la documentación:

import { v4 as uuidv4 } from 'uuid';

uuidv4(); // ⇨ '9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d'

import { v4 as uuidv4 } from 'uuid';

function App() {

  const usersData = [

    { id: uuidv4(), name: 'Tania', username: 'floppydiskette' },

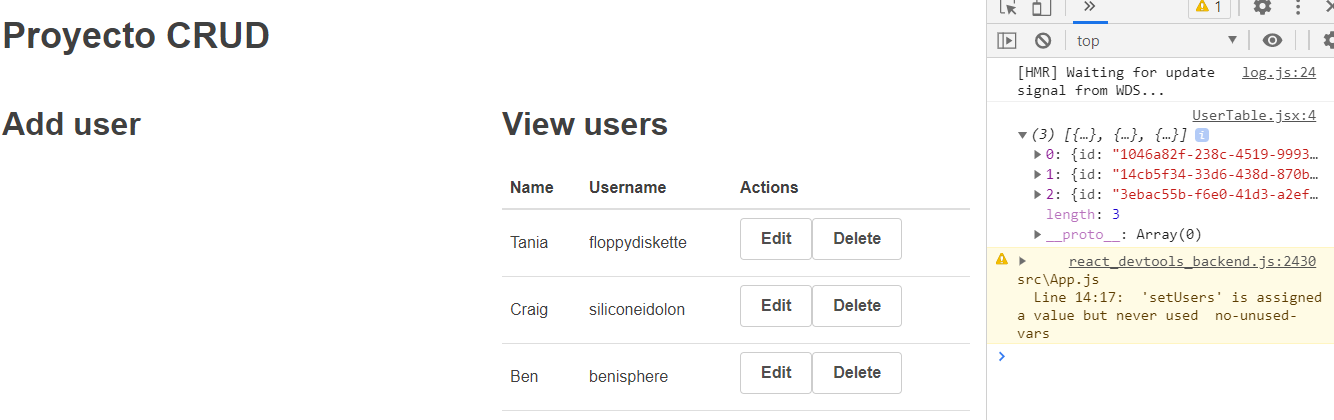
    { id: uuidv4(), name: 'Craig', username: 'siliconeidolon' },

    { id: uuidv4(), name: 'Ben', username: 'benisphere' },

  ]

  const [users, setUsers] = useState(usersData);

así se ve nuestro proyecto



11.- Agregaremos una validación para saber si hay o no hay registros es muy simple agregamos una condición en nuestra tabla y el código quedaría así

<tbody>

                {

                    props.users.length > 0 ?

                    props.users.map(user => (

                        <tr key={user.id}>

                            <td>{user.name}</td>

                            <td>{user.username}</td>

                            <td>

                            <button className="button muted-button">Edit</button>

                            <button className="button muted-button">Delete</button>

                            </td>

                        </tr>

                    )) : (

                        <tr>

                            <td colSpan={3}>No Users</td>

                        </tr>

                    )

                }

Iniciar con las operaciones CRUD

Agregar usuario

1.- Creamos una función para agregar elementos en el App.js aquí vamos a recibir como parámetro el user, vamos a utilizar la función uuidv4() para generar el id y vamos a editar nuestro state con el setUser.

//Agregar usuarios

  const addUser = (user) => {

    user.id = uuidv4()

    setUsers([

      ...users,

      user

    ])

  }

2.- Ahora vamos a crear el componente AddUserForm.jsx ponemos el código y le vamos a mandar un props que será el user id que se creara cada que se ejecute o se mande a llamar

import React from 'react';

const AddUserForm = (props) => {

    return (

        <form>

            <label>Name:</label>

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

            <label>UserName:</label>

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

            <button>Add new User</button>

        </form>

     );

}

export default AddUserForm;

Del otro lado simplemente lo importamos y mandamos el props que llamaremos adduser={addUser}

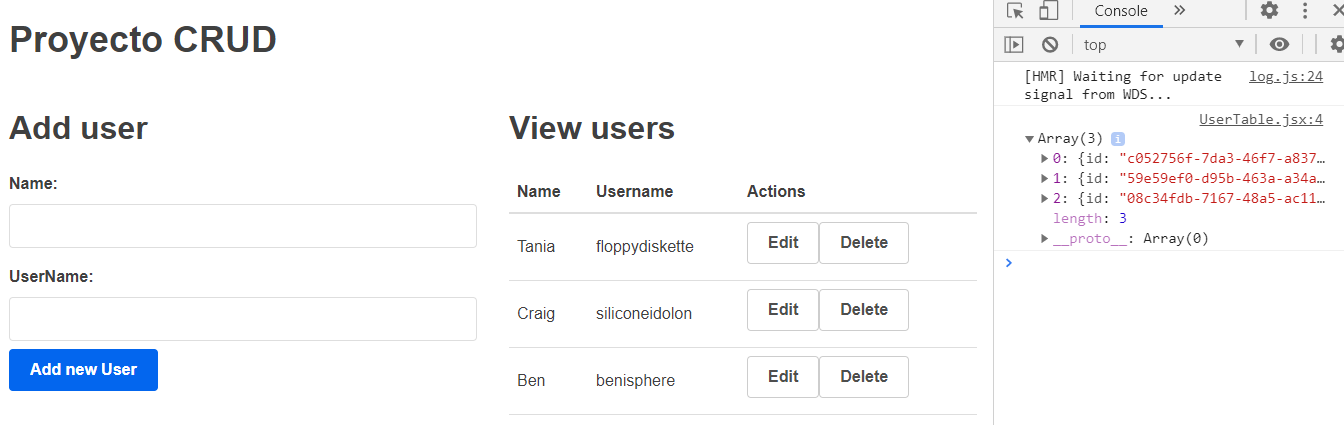
<div className="flex-large">

          <h2>Add user</h2>

          <AddUserForm addUser={addUser}/>

        </div>

Previsualizamos y ya debemos de tener incorporado nuestro formulario



Ahora en el componente AdduserForm vamos a importar la librería de los React hooks forms lo vamos a tomar del proyecto myapp6 FormsHooks ahí vemos el ejemplo y esto queda así:

import React from 'react';

import { useForm } from "react-hook-form";

const AddUserForm = (props) => {

    const {register, errors, handleSubmit} = useForm();

    const onsubmit = (data, e) => {

        console.log(data)

        //Limpiar caja

        e.targe.reset();

    }

    return (

        <form onSubmit={handleSubmit(onsubmit)}>

            <label>Name:</label>

            <input type="text" name="name"

            ref={

                register({

                    required: {

                        value: true,

                        message: 'Campo obligatorio'

                    },

                })

            }/>

            <span className="text-danger">

                    {errors?.name?.message}

            </span>

            <label>UserName:</label>

            <input type="text" name="username"

            ref={

                register({

                    required: {

                        value: true,

                        message: 'Campo obligatorio'

                    },

                })

            }/>

            <div>

                <span className="text-danger">

                    {errors?.username?.message}

                </span>

            </div>

            <button>Add new User</button>

        </form>

     );

}

export default AddUserForm;

Ahora lo mas sencillo vamos ya por fin agregar elementos a mi colección solo usamos el props,addUser(data) le mandamos la data que estamos tomando del form hook

const onsubmit = (data, e) => {

        console.log(data)

        //Vamos a agregar elemento usando el props

        props.addUser(data)

        //Limpiar caja

        e.target.reset();

    }

