**METODOS Y ATAJOS EN REACT**

**CREAR PROYECTO REACT =>**

**Estos comandos se pueden ingresar desde la consola de windows o desde la consola de Visual Code:**

**Desde la carpera ProyectosReact =>create-react-app primer-app-react**

**ahora para crear Proyecto React => npx create-react-app my-app**

**Desde el Proyecto =>**

**npm install --save react-router-dom => este paquete/libreria permite crear rutas, vinculos**

**npm install react-router-dom@5.2.0 mantiene el SWITCH ROUTE**

**npm install react-bootstrap bootstrap => este paquete/libreria corresponde a bootstrap**

**En la index.js => de esta forma podemos usar Bootstrap en la aplicación, gestionando la importación.**

**import 'bootstrap/dist/css/bootstrap.min.css';**

**npm install axios => librería que permite la comunicación cliente-servidor. "axios": "^0.21.1", se verifica desde el package.json**

**npm install fetch => librería que permite la comunicación cliente-servidor. "fetch": "^1.1.0",**

**useForm => para la validación de formularios previamente hay que instalar**

**npm install react-hook-form**

**Ejecutar => npm start**

**Estructura componente básico en REACT =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import '../assets/css/grillaAlumno.css'

const GrillaAlumno = (props) => {

    const [dato,setDato] = useState(null)

    useEffect(() => {

    },[])

    if(dato === null){

        return null

    }else{

        return(

            <Fragment>

                <Navigation></Navigation>

                <br></br>

                <div className="body">

                <Alert variant="success">

                <Alert.Heading className="alertTitle">GRILLA ALUMNOS</Alert.Heading>

                <br></br>

                <br></br>

                </Alert>

                </div>

            </Fragment>

        )

    }

}

export default GrillaAlumno;

**Componente Grilla con Table =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import '../assets/css/grillaAlumno.css'

const GrillaAlumno = (props) => {

    const [dato,setDato] = useState(null)

    useEffect(() => {

        getAlumnos()

    },[])

    //Metodo para obtener datos API =>

    const getAlumnos = async () => {

        try{

            const response = await axios("http://179.43.113.170:8082/test/td/alumnos.json", {

                method:"GET",

                params:{

                }

            })

            const resJson = await response.data

            console.log("DATOS API => ", resJson)

            setDato(resJson)

        }catch(error){

            console.log(error)

        }

    }

    //Metodo para calcular el promedio Notas =>

    const calcularPromedio = (array) => {

        let promedio = 0

        let suma = 0

        for(let i = 0; i < array.length; i++){

            suma += array[i]

        }

        return suma / array.length

    }

    if(dato === null){

        return null

    }else{

        return(

            <Fragment>

                <Navigation></Navigation>

                <br></br>

                <div className="body">

                <Alert variant="success">

                <Alert.Heading className="alertTitle">GRILLA ALUMNOS</Alert.Heading>

                <br></br>

                <br></br>

                <Table className="tabla" striped bordered hover variant="dark">

                <thead>

                        <tr>

                            <th className="celda">N°</th>

                            <th className="celda">Alumno</th>

                            <th className="celda">Legajo</th>

                            <th className="celda">Fecha Nacimiento</th>

                            <th className="celda">Promedio Notas</th>

                            <th className="celda">Acciones</th>

                        </tr>

                </thead>

                <tbody>

                    {dato.map((alumno,i) => (

                        <tr key={i}>

                            <td className="celda">{i+1}</td>

                            <td className="celda">{alumno.alumno}</td>

                            <td className="celda">{alumno.legajo}</td>

                            <td className="celda">{alumno.fechaNacimiento}</td>

                            <td className="celda">{ calcularPromedio(alumno.notas) }</td>

                            <td className="celda">

                                <Button variant="link" href={`/detalleAlumno/${alumno.legajo}`}>VER DETALLE</Button>&nbsp;&nbsp;

                            </td>

                        </tr>

                    ))}

                </tbody>

                </Table>

                </Alert>

                </div>

            </Fragment>

        )

    }

}

export default GrillaAlumno;

**Componente Grilla con Implementación de Búsqueda Navegación =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import '../assets/css/grillaAlumno.css'

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

const GrillaAlumno = (props) => {

    // Obtengo los datos pasados por search URL =>

    let {search} = useLocation();

    let query = new URLSearchParams(search)

    const [dato,setDato] = useState(null)

    // PAGINA RECIVE ESTA EN ESCUCHA DE PARAMS => VER PROYECTO ALUMNOS HOOKS

    const [nomUrl, setNomUrl] = useState(query.get("nombre"))

    const [legUrl, setLegUrl] = useState(query.get("legajo"))

    useEffect(() => {

        //De esta forma esta a la escucha de las modificaciones en los search URL =>

        setNomUrl(query.get("nombre"))

        setLegUrl(query.get("legajo"))

        getAlumnos()

    },[query.get("legajo"), query.get("nombre")])

    //Metodo para obtener datos API =>

    const getAlumnos = async () => {

        const urlNombre = query.get("nombre")

        const urlLegajo = query.get("legajo")

        if((urlNombre === null || urlNombre === undefined) && (urlLegajo === null || urlLegajo === undefined)){

            try{

                const response = await axios("http://179.43.113.170:8082/test/td/alumnos.json", {

                    method:"GET",

                    params:{

                    }

                })

                const resJson = await response.data

                console.log("DATOS API => ", resJson)

                setDato(resJson)

            }catch(error){

                console.log(error)

            }

        }else if((urlNombre !== null || urlNombre !== undefined) && (urlLegajo === null || urlLegajo === undefined)){

            try{

                const response = await axios("http://179.43.113.170:8082/test/td/alumnos.json", {

                    method:"GET",

                    params:{

                    }

                })

                const resJson = await response.data

                console.log("DATOS API => ", resJson)

                //Filtramos la busqueda para obtener el dato buscado por urlNombre =>

                const encontrado = resJson.filter((alumno) => (alumno.alumno).toString() === (urlNombre).toString())

                setDato(encontrado)

            }catch(error){

                console.log(error)

            }

        }else if((urlNombre === null || urlNombre === undefined) && (urlLegajo !== null || urlLegajo !== undefined)){

            try{

                const response = await axios("http://179.43.113.170:8082/test/td/alumnos.json", {

                    method:"GET",

                    params:{

                    }

                })

                const resJson = await response.data

                console.log("DATOS API => ", resJson)

                //Filtramos la busqueda para obtener el dato buscado por urlNombre =>

                const encontrado = resJson.filter((alumno) => (alumno.legajo).toString() === (urlLegajo).toString())

                setDato(encontrado)

            }catch(error){

                console.log(error)

            }

        }

    }

    //Metodo para calcular el promedio Notas =>

    const calcularPromedio = (array) => {

        let promedio = 0

        let suma = 0

        for(let i = 0; i < array.length; i++){

            suma += array[i]

        }

        return suma / array.length

    }

    if(dato === null){

        return null

    }else{

        return(

            <Fragment>

                <Navigation></Navigation>

                <br></br>

                <div className="body">

                <Alert variant="success">

                <Alert.Heading className="alertTitle">GRILLA ALUMNOS</Alert.Heading>

                <br></br>

                <br></br>

                <Table className="tabla" striped bordered hover variant="dark">

                <thead>

                        <tr>

                            <th className="celda">N°</th>

                            <th className="celda">Alumno</th>

                            <th className="celda">Legajo</th>

                            <th className="celda">Fecha Nacimiento</th>

                            <th className="celda">Promedio Notas</th>

                            <th className="celda">Acciones</th>

                        </tr>

                </thead>

                <tbody>

                    {dato.map((alumno,i) => (

                        <tr key={i}>

                            <td className="celda">{i+1}</td>

                            <td className="celda">{alumno.alumno}</td>

                            <td className="celda">{alumno.legajo}</td>

                            <td className="celda">{alumno.fechaNacimiento}</td>

                            <td className="celda">{ calcularPromedio(alumno.notas) }</td>

                            <td className="celda">

                                <Button variant="link" href={`/detalleAlumno/${alumno.legajo}`}>VER DETALLE</Button>&nbsp;&nbsp;

                            </td>

                        </tr>

                    ))}

                </tbody>

                </Table>

                </Alert>

                </div>

            </Fragment>

        )

    }

}

export default GrillaAlumno;

**Componente DetalleAlgo con Row y Col =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import Image from 'react-bootstrap/Image'

import '../assets/css/detalle.css'

import moment from 'moment';

const DetalleDino = (props) => {

    const [dato,setDato] = useState(null);

    useEffect(() => {

        getDino()

    },[])

    const getDino = async () => {

        try{

            const idDino = props.match.params.id;

            console.log("ID\_DINO => ", idDino)

            const response = await axios(`http://localhost:8080/Dinosaurios/DinosaurioServlet`, {

                method:"GET",

                params:{

                    action:"buscarOne",

                    idDino:idDino,

                }

            })

            const resJson = await response.data

            console.log("DATOS API => ", resJson)

            setDato(resJson)

        }catch(error){

            console.log("Error => ", error)

        }

    }

    if(dato === null){

        return null

    }else{

        return(

            <Fragment>

                <Navigation></Navigation>

                <Container>

                <br></br>

                <Alert variant="success" className="body">

                <Alert.Heading className="alertTitle">DETALLE DINOSAURIO</Alert.Heading>

                <br></br>

                <br></br>

                <Row>

                    <Col>

                    <h3>{(dato.nombre).toUpperCase()}</h3>

                    </Col>

                    <Col>

                    </Col>

                </Row>

                <br></br>

                <Row>

                    <Col>

                    <Image className="fotoDetalle" src={require(`../assets/img/${dato.imagen}`)} rounded />

                    </Col>

                    <Col>

                    <Row>

                        <Col>

                            <h5>FUERZA:</h5>

                            <br></br>

                            <h5>FECHA ALTA:</h5>

                            <br></br>

                            <h5>FECHA BAJA:</h5>

                            <br></br>

                            <h5>ESTADO:</h5>

                            <br></br>

                            <h5>INFO:</h5>

                            <br></br>

                            </Col>

                            <Col>

                            <h5>{dato.fuerza}</h5>

                            <br></br>

                            <h5>{moment(dato.fechaAlta).subtract(1,'M').format('YYYY-MM-DD')}</h5>

                            <br></br>

                            <h5>{moment(dato.fechaBaja).subtract(1,'M').format('YYYY-MM-DD')}</h5>

                            <br></br>

                            <h5>{dato.estado}</h5>

                            <br></br>

                            <h5>{dato.info}</h5>

                            <br></br>

                            </Col>

                        </Row>

                    </Col>

                </Row>

                <br></br>

                <Row>

                    <Col>

                    <Button variant="danger" size="lg" href="/grillaDino">VOLVER</Button>

                    </Col>

                </Row>

                </Alert>

                </Container>

            </Fragment>

        )

    }

}

export default DetalleDino;

**COMPONENTE NAVIGATION CON BUSQUEDA =>**

import React, {Component, Fragment, useState, useEffect} from 'react';

import Button from 'react-bootstrap/Button';

import Navbar from 'react-bootstrap/Navbar';

import Nav from 'react-bootstrap/Nav';

import FormControl from 'react-bootstrap/FormControl';

import Form from 'react-bootstrap/Form';

import Container from "react-bootstrap/Container";

import Label from "react-bootstrap/FormLabel";

import { useHistory } from "react-router-dom"

const Navigation = () => {

    //HISTORY REDIRECCIONAR POR PAGINA =>

    let history = useHistory()

    const [dato, setDato] = useState({

        legajo:null,

        nombre:null,

    })

    useEffect(() => {

    },[])

    //ESTE METODO SE PUEDE USAR PARA CAPTURAR LA INFORMACION INGRESADA EN EL FORM:

    const handleInputChange = (event) => {

        setDato({

            ...dato,

            [event.target.name] : event.target.value

        })

    }

    //Metodo para accionar el evento onClink =>

    const obtenerLegajo = () => {

        if(dato.legajo === '' || dato.legajo === null){

            alert("DEBE INGRESAR DATO A LA BUSQUEDA DE LEGAJO")

        }else{

            console.log("OBTENER LEGAJO => ", dato.legajo)

            //Redirecciono y paso los datos a traves de un search =>

            history.push({

                pathname: '/',

                search: '',

                search: `?legajo=${dato.legajo}`,

            })

        }

    }

    //Metodo para accionar el evento onClink =>

    const obtenerNombre = () => {

        if(dato.nombre === '' || dato.nombre === null){

            alert("DEBE INGRESAR DATO A LA BUSQUEDA DE NOMBRE")

        }else{

            console.log("OBTENER NOMBRE => ", dato.nombre)

            //Redirecciono y paso los datos a traves de un search =>

            history.push({

                pathname: '/',

                search: '',

                search: `?nombre=${dato.nombre}`,

            })

        }

    }

    return (

        <Fragment>

        <Navbar bg="primary" variant="dark">

            <Navbar.Brand href="/">HOME</Navbar.Brand>

            <Nav className="me-auto">

            <Nav.Link href="/grillaDino">MEJOR ALUMNO</Nav.Link>

            </Nav>

            <Form className="d-flex">

                <Form.Label>LEGAJO:</Form.Label>&nbsp;&nbsp;

                <FormControl

                    type="search"

                    placeholder="Search"

                    className="me-2"

                    aria-label="Search"

                    name="legajo"

                    onChange={handleInputChange}

                />&nbsp;&nbsp;

                <Button variant="outline-success" onClick={obtenerLegajo}>SEARCH</Button>&nbsp;&nbsp;

            </Form>

            <Form className="d-flex">

                <Form.Label>NOMBRE:</Form.Label>&nbsp;&nbsp;

                <FormControl

                    type="search"

                    placeholder="Search"

                    className="me-2"

                    aria-label="Search"

                    name="nombre"

                    onChange={handleInputChange}

                />&nbsp;&nbsp;

                <Button variant="outline-success" onClick={obtenerNombre}>SEARCH</Button>&nbsp;&nbsp;

            </Form>

        </Navbar>

        </Fragment>

    );

}

export default Navigation;

**COMPONENTE APP.js (ROUTES) =>**

import { Fragment } from 'react';

import './App.css';

import { Switch, Route} from 'react-router-dom';

import Home from './components/Home'

import GrillaArticulo from './components/GrillaArticulo'

import DetalleArticulo from './components/DetalleArticulo'

import ArticuloBarato from './components/ArticuloBarato'

const App = () => {

    return(

      <Fragment>

          <Switch>

            <Route exact path="/" component={GrillaArticulo} ></Route>

            <Route exact path="/detalleArticulo/:id" component={DetalleArticulo} ></Route>

            <Route exact path="/masBarato" component={ArticuloBarato} ></Route>

          </Switch>

      </Fragment>

    );

}

export default App;

**COMPONENTE INDEX.js EJECUTABLE =>**

import React from 'react';

import ReactDOM from 'react-dom';

import './index.css';

import App from './App';

import 'bootstrap/dist/css/bootstrap.min.css';

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

import reportWebVitals from './reportWebVitals';

ReactDOM.render(

  <React.StrictMode>

    <BrowserRouter>

      <App />

    </BrowserRouter>

  </React.StrictMode>,

  document.getElementById('root')

);

reportWebVitals();

**COMPONENTE TARJETA =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import Card from 'react-bootstrap/Card'

import '../assets/css/tarjeta.css'

const Tarjeta = (props) => {

    const[dato,setDato] = useState(null)

    useEffect(() => {

    },[])

    return (

        <Fragment>

            <Container>

            <br></br>

            <Card style={{ width: '21rem' }}>

            <Card.Img className="imagenCard" variant="top" src={props.imagen} />

            <Card.Body>

                <Card.Title>{ props.signo }</Card.Title>

                <Card.Text></Card.Text>

                <Card.Text></Card.Text>

                <Card.Text></Card.Text>

                <Button variant="primary" href={`/detalleZodiaco/${props.id}`}>VER DETALLE</Button>

            </Card.Body>

            </Card>

            </Container>

        </Fragment>

    );

}

export default Tarjeta;

**COMPONENTE HOME CON TARJETA Y BUSQUEDA =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import Tarjeta from "../components/Tarjeta"

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

import "../assets/css/home.css"

const Home = (props) => {

    //Obtengo los datos pasados por search URL =>

    let {search} = useLocation();

    let query = new URLSearchParams(search)

    const[dato,setDato] = useState(null)

    //Creamos el Hooks y queda a espera de modificar en useEffect =>

    const[urlMes, setUrlMes] = useState(query.get("mes"))

    useEffect(() => {

        getZodiacos()

        //De esta forma esta a la escucha de las modificaciones en los search URL =>

        setUrlMes(query.get("mes"))

    },[query.get("mes")])

    const getZodiacos = async () => {

        const mesURL = query.get("mes")

        if(mesURL === null || mesURL === undefined){

            try{

                const response = await axios("http://179.43.113.170:8082/test/tb/zodiaco.json", {

                    method:"GET",

                    params:{

                    }

                })

                const resJson = await response.data

                console.log("DATOS API => ", resJson)

                setDato(resJson)

            }catch(error){

                console.log("Error => ", error)

            }

        }else if(mesURL !== null || mesURL !== undefined){

            try{

                const response = await axios("http://179.43.113.170:8082/test/tb/zodiaco.json", {

                    method:"GET",

                    params:{

                    }

                })

                const resJson = await response.data

                console.log("DATOS API => ", resJson)

                //Filtramos la busqueda por array dentro de otro array =>

                const encontrado = resJson.filter((zodiaco) => {

                    return zodiaco.meses.includes(parseInt(mesURL))

                })

                setDato(encontrado)

            }catch(error){

                console.log("Error => ", error)

            }

        }

    }

    if(dato === null){

        return null

    }else{

        const cards = dato.map((zodiaco,i) => {

            return(

                <Tarjeta

                    imagen={zodiaco.imagen}

                    signo={zodiaco.signo}

                    id={zodiaco.id}

                />

            )

        })

        return(

            <Fragment>

                <Navigation></Navigation>

                <br></br>

                <Container className="body">

                <Alert variant="success">

                <Alert.Heading className="alertTitle">GRILLA ZODIACOS</Alert.Heading>

                <br></br>

                <br></br>

                <Row xs={1} md={3} className="g-4">

                    {cards}

                </Row>

                </Alert>

                </Container>

            </Fragment>

        )

    }

}

export default Home;

**Uso de AXIOS para consultar Servlet y Base de Datos =>**

 const getDino = async () => {

        try{

            //Importante ver el LocalHost 8080 0 3000 que se abre Proyecto Web Netbeans =>

            const response = await axios("http://localhost:8080/Dinosaurios/DinosaurioServlet?", {

                method:"GET",

                params:{

                    action:"buscarAll"

                }

            })

            const resJson = await response.data;

            console.log("DATOS API => ", resJson)

            setDato(resJson)

        }catch(error){

            console.log(error)

        }

    }

**Uso de AXIOS para consultar Servlet y Base de Datos Obteniendo Parametro URL =>**

const getDino = async () => {

        try{

            const idDino = props.match.params.id;

            console.log("ID\_DINO => ", idDino)

            const response = await axios(`http://localhost:8080/Dinosaurios/DinosaurioServlet`, {

                method:"GET",

                params:{

                    action:"buscarOne",

                    idDino:idDino,

                }

            })

            const resJson = await response.data

            console.log("DATOS API => ", resJson)

            setDato(resJson)

        }catch(error){

            console.log("Error => ", error)

        }

    }

**Uso de AXIOS para consultar API REST =>**

try{

 const response = await axios("http://179.43.113.170:8082/test/td/alumnos.json", {

         method:"GET",

         params:{

         }

 })

 const resJson = await response.data

 console.log("DATOS API => ", resJson)

 setDato(resJson)

}catch(error){

 console.log(error)

}

**Uso de FETCH para consultar Servlet y Base de Datos =>**

const getDino = async () => {

        try{

            //Importante ver el LocalHost 8080 0 3000 que se abre Proyecto Web Netbeans =>

            const response = await fetch("http://localhost:8080/Dinosaurios/DinosaurioServlet?action=buscarAll", {

                method:"GET",

            })

            const resJson = await response.json();

            console.log("DATOS API => ", resJson)

            setDato(resJson)

        }catch(error){

            console.log(error)

        }

    }

**Uso de FETCH consulta Servlet Base de Datos Obteniendo Parametro URL =>**

const getDino = async () => {

        try{

            const idDino = props.match.params.id;

            const response = await fetch(`http://localhost:8080/Dinosaurios/DinosaurioServlet?action=buscarOne&idDino=${idDino}`, {

                method:"GET",

            })

            const resJson = response.json()

            console.log("DATOS API => ", resJson)

            setDato(resJson)

        }catch(error){

            console.log("Error => ", error)

        }

    }

**Batalla de Dinosaurios =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

const BatallaDino = (props) => {

    const [dato, setDato] = useState(null);

    const [dino,setDino] = useState({

        dino1:'',

        dino2:'',

    })

    useEffect(() => {

        getDinos()

    },[])

    //Metodo que se ejecuta en los input onChange, permite detectar el ingreso de datos:

    const handleInputChange = (event) => {

        setDino({

            ...dino,

            [event.target.name] : event.target.value

        })

    }

    //Metodo para obtener los datos a traves de la BD =>

    const getDinos = async () => {

        try{

            const response = await fetch("http://localhost:8080/Dinosaurios/DinosaurioServlet?action=buscarAll", {

                method:"GET",

            })

            const resJson = await response.json()

            console.log("DATOS API => ", resJson)

            setDato(resJson)

        }catch(error){

            console.log("Error => ", error)

        }

    }

    const batallaDino = () => {

        console.log("DINO 1 => ", dino.dino1)

        console.log("DINO 2 => ", dino.dino2)

        let dino1 = null

        let dino2 = null

        for(let i = 0; i < dato.length; i++){

            //Obtengo el objeto dino1 =>

            if(dino.dino1 === dato[i].nombre){

                dino1 = dato[i]

            }

            //Obtengo el objeto dino2 =>

            if(dino.dino2 === dato[i].nombre){

                dino2 = dato[i]

            }

        }

        //Definicion de la Batalla =>

        if(dino1.fuerza > dino2.fuerza){

            document.getElementById("ganador").innerHTML = "GANADOR " + (dino1.nombre).toUpperCase()

        }else if(dino2.fuerza > dino1.fuerza){

            document.getElementById("ganador").innerHTML = "GANADOR " + (dino2.nombre).toUpperCase()

        }else{

            document.getElementById("ganador").innerHTML = "EMPATE"

        }

    }

    if(dato === null){

        return null

    }else{

        return(

            <Fragment>

                <Navigation></Navigation>

                <Container>

                <br></br>

                <Alert variant="success" className="body">

                <Alert.Heading className="alertTitle">BATALLA DINOSAURIOS</Alert.Heading>

                <br></br>

                <br></br>

                <Table className="tabla" striped bordered hover variant="dark">

                <tbody>

                    <tr>

                        <td className='celda'>DINO\_1</td>

                        <td className='celda'>

                            <select name="dino1" onChange={handleInputChange}>

                                {dato.map((dino,i) => (

                                    <option key={i} value={dino.nombre}>{dino.nombre}</option>

                                ))}

                            </select>

                        </td>

                        <td className='celda'>VS</td>

                        <td className='celda'>DINO2</td>

                        <td className='celda'>

                            <select name="dino2" onChange={handleInputChange}>

                                {dato.map((dino,i) => (

                                    <option key={i} value={dino.nombre}>{dino.nombre}</option>

                                ))}

                            </select>

                        </td>

                    </tr>

                    <tr>

                        <td className='celda' colSpan={5}><Button variant="danger" size="lg" onClick={batallaDino}>BATALLA</Button></td>

                    </tr>

                    <tr>

                        <td className='celda' colSpan={5}><h3 id="ganador" name="ganador"></h3></td>

                    </tr>

                </tbody>

                </Table>

                </Alert>

                </Container>

            </Fragment>

        )

    }

}

export default BatallaDino;

**Grilla con implementación de la Busqueda de Navigation =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import Tarjeta from "./Tarjeta"

import '../assets/css/grillaDino.css'

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

const GrillaDino  = (props) => {

    //Obtengo los datos pasados por search URL =>

    let {search} = useLocation();

    let query = new URLSearchParams(search)

    const [nomUrl, setNomUrl] = useState(query.get("nombre"))

    const [dato, setDato] = useState(null)

    useEffect(() => {

        getDino()

        //De esta forma esta a la escucha de las modificaciones en los search URL =>

        setNomUrl(query.get("nombre"))

    },[query.get("nombre")])

    const getDino = async () => {

        try{

            const nomURL = query.get("nombre")

            if(nomURL === null || nomURL === undefined){

                //Importante ver el LocalHost 8080 0 3000 que se abre Proyecto Web Netbeans =>

                const response = await fetch("http://localhost:8080/Dinosaurios/DinosaurioServlet?action=buscarAll", {

                    method:"GET",

                })

                const resJson = await response.json();

                console.log("DATOS API => ", resJson)

                setDato(resJson)

            }else{

                  //Importante ver el LocalHost 8080 0 3000 que se abre Proyecto Web Netbeans =>

                  const response = await fetch("http://localhost:8080/Dinosaurios/DinosaurioServlet?action=buscarAll", {

                    method:"GET",

                })

                const resJson = await response.json();

                console.log("DATOS API => ", resJson)

                const encontrado = resJson.filter(function (element){

                    return (element.nombre).toString() === (nomURL).toString()

               })

               setDato(encontrado)

            }

        }catch(error){

            console.log(error)

        }

    }

    if(dato === null){

        return null

    }else{

        //reutilizar el componente tarjeta y le paso los datos =>

        const card = dato.map((dino, i) => {

            return (

              <Tarjeta

                key = { dino.idDino }

                id = { dino.idDino }

                nombre = { dino.nombre}

                imagen = { dino.imagen }

              ></Tarjeta>

            );

        });

        return (

            <Fragment>

                <Navigation></Navigation>

                <Container>

                <br></br>

                <Alert variant="success" className="body">

                <Alert.Heading className="alertTitle">DINOSAURIOS</Alert.Heading>

                <br></br>

                <br></br>

                <Row xs={1} md={3} className="g-4">

                    { card }

                </Row>

                </Alert>

                </Container>

            </Fragment>

        );

    }

}

export default GrillaDino;

**USO DE CRUD (CREATE-READ-UPDATE-DELETE) en React =>**

**Admin (Read All/BuscarAll) Principal (Tabla de Datos) =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import '../assets/css/admin.css'

import moment from 'moment';

const AdminPrincipal = (props) => {

    const [dato, setDato] = useState(null);

    useEffect(() => {

        getDinos()

    },[])

    const getDinos = async () => {

        try{

            const response = await fetch("http://localhost:8080/Dinosaurios/DinosaurioServlet?action=buscarAll",{

                method:"GET",

            })

            const resJson = await response.json()

            console.log("DATOS API => ", resJson)

            setDato(resJson)

        }catch(error){

            console.log("Error => ", error)

        }

    }

    if(dato === null){

        return null

    }else{

        return(

            <Fragment>

            <Navigation></Navigation>

            <br></br>

            <Alert variant="success">

            <div className="body">

            <Alert.Heading className="alertTitle">ADMIN PRINCIPAL</Alert.Heading>

            <br></br>

            <br></br>

            <Table className="tabla" striped bordered hover variant="dark">

                <thead>

                        <tr>

                            <th className="celda">Id</th>

                            <th className="celda">Imagen</th>

                            <th className="celda">Info</th>

                            <th className="celda">Fuerza</th>

                            <th className="celda">Fecha Alta</th>

                            <th className="celda">Fecha Baja</th>

                            <th className="celda">Estado</th>

                            <th className="celda">Acciones</th>

                        </tr>

                </thead>

                <tbody>

                    {dato.map((dino,i) => (

                        <tr key={i}>

                            <td className="celda">{dino.idDino}</td>

                            <td className="celda">{dino.imagen}</td>

                            <td className="celda">{dino.info}</td>

                            <td className="celda">{dino.fuerza}</td>

                            <td className="celda">{moment(dino.fechaAlta).subtract(1,'M').format('YYYY-MM-DD')}</td>

                            <td className="celda">{moment(dino.fechaBaja).subtract(1,'M').format('YYYY-MM-DD')}</td>

                            <td className="celda">{dino.estado}</td>

                            <td className="celda">

                                <Button variant="warning" size="sm" href={`/actualizar/${dino.idDino}`}>ACTUALIZAR</Button>&nbsp;&nbsp;

                                <Button variant="danger" size="sm" href={`/eliminarLog/${dino.idDino}`}>ELIMINAR</Button>

                            </td>

                        </tr>

                    ))}

                </tbody>

            </Table>

            <Button variant="success" size='lg' href="/insertar">INSERTAR</Button>&nbsp;&nbsp;

            <Button variant="danger" size='lg' href="/">VOLVER</Button>

            </div>

            </Alert>

            </Fragment>

        )

    }

}

export default AdminPrincipal;

**Registrar/Insert =>**

**Se debe agregar la librería react hook form (validar formularios)**

**npm install react-hook-form**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import Form from "react-bootstrap/Form";

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

import moment from 'moment';

import '../assets/css/registrar.css'

//Form Insertar Datos =>

const Insertar = (props) => {

    //react-hook-form =>

    const {register, formState: { errors }, handleSubmit} = useForm()

    const [dato, setDato] = useState({

        nombre:'',

        imagen:'',

        info:'',

        fuerza:'',

    })

    useEffect(() => {

    },[])

    //Metodo para obtener los datos ingresados en el form =>

    const handleInputChange = (event) => {

        setDato({

            ...dato,

            [event.target.name] : event.target.value

        })

    }

    //Metodo para gestionar el envio de datos al Servlet y BD =>

    const enviarDatos = (dato, event) => {

        insertar(dato)

        event.preventDefault();

        event.target.reset();

    }

    //Metodo para insertar Datos =>

    const insertar = async (dato) => {

        try{

            const response = await axios("http://localhost:8080/Dinosaurios/DinosaurioServlet", {

                method:"GET",

                params:{

                    action:"insertar",

                    nombre:dato.nombre,

                    imagen:dato.imagen,

                    info:dato.info,

                    fuerza:dato.fuerza,

                    fechaAlta:moment().format('YYYY-MM-DD'),

                    fechaBaja:moment("1900-01-01").format('YYYY-MM-DD'),

                    estado:"activo",

                }

            })

            const resJson = await response.data

            console.log("DATOS API => ", resJson)

            alert("Datos Insertados con Exito en la Base de Datos.")

        }catch(error){

            console.log(error)

            alert("Error los datos no fueron insertados con exito.")

        }

    }

    //Validacion Personalizada que no permite que se ingresen 2 nombres iguales a la BD =>

    const validarNombre = async (nombre) => {

        try{

            const response = await fetch("http://localhost:8080/Dinosaurios/DinosaurioServlet?action=buscarAll", {

                method:"GET"

            })

            const resJson = await response.json()

            const listaDino = resJson

            let validar = true

            for(let i = 0; i < listaDino.length; i++){

                if((listaDino[i].nombre).toString() === (nombre).toString()){

                    validar = false;

                    break;

                }

            }

            //Si el dato.nombre ingresado en form es igual a algun nombre de la BD retorna falsa y no permite la validacion =>

            return validar;

        }catch(error){

            console.log("Error =>", error)

        }

    }

    if(dato === null){

        return null

    }else{

        return(

            <Fragment>

                <Navigation></Navigation>

                <br></br>

                <Alert variant="success">

                <div className="body">

                <Alert.Heading className="titulo">FORMULARIO ADMIN REGISTRO DINOSAURIO</Alert.Heading>

                <br></br>

                <br></br>

                <Form onSubmit={handleSubmit(enviarDatos)}>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Nombre: </label>

                    </Col>

                    <Col>

                        <input

                            type="text"

                            name="nombre"

                            onChange={handleInputChange}

                            placeholder="Ingrese el Nombre"

                            className="form-control my-2"

                            {...register("nombre", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                                validate:{

                                    validate1:validarNombre,

                                }

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                        <span className="text-danger text-small d-block mb-2">

                        {errors.nombre && errors.nombre.message}

                        </span>

                        <span className="text-danger text-small d-block mb-2">

                        {

                            errors.nombre && errors.nombre.type === "validate1" && (

                                <div className="error">El nombre ingresado ya existe</div>

                            )

                        }

                        </span>

                    </Col>

                </Row>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Imagen: </label>

                    </Col>

                    <Col>

                        <input

                            type="text"

                            name="imagen"

                            onChange={handleInputChange}

                            placeholder="Ingrese la imagen"

                            className="form-control my-2"

                            {...register("imagen", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                        <span className="text-danger text-small d-block mb-2">

                        {errors.imagen && errors.imagen.message}

                        </span>

                    </Col>

                </Row>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Info: </label>

                    </Col>

                    <Col>

                        <input

                            type="text"

                            name="info"

                            onChange={handleInputChange}

                            placeholder="Ingrese la info"

                            className="form-control my-2"

                            {...register("info", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                        <span className="text-danger text-small d-block mb-2">

                        {errors.info && errors.info.message}

                        </span>

                    </Col>

                </Row>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Fuerza: </label>

                    </Col>

                    <Col>

                        <input

                            type="number"

                            name="fuerza"

                            onChange={handleInputChange}

                            placeholder="Ingrese la fuerza"

                            className="form-control my-2"

                            min="1"

                            max="10"

                            {...register("fuerza", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                        <span className="text-danger text-small d-block mb-2">

                        {errors.fuerza && errors.fuerza.message}

                        </span>

                    </Col>

                </Row>

                <Row>

                    <Col className='boton'>

                        <br></br>

                        <br></br>

                        <Button type="submit" variant="primary" size="lg">REGISTER</Button>&nbsp;&nbsp;

                        <Button type="button" variant="danger" size="lg" href={`/admin`}>VOLVER</Button>

                    </Col>

                </Row>

                </Form>

                </div>

                </Alert>

            </Fragment>

        )

    }

}

export default Insertar;

**Update/Actualizar y (Read One/BuscarOne) =>**

**Se debe agregar la librería react hook form (validar formularios)**

**npm install react-hook-form**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import Form from "react-bootstrap/Form";

import moment from 'moment';

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

const Actualizar = (props) => {

    //Validar formulario con Libreria useForm =>

    const {register, formState: { errors }, handleSubmit, setValue} = useForm({

    })

    const [dato,setDato] = useState({

        nombre:'',

        imagen:'',

        info:'',

        fuerza:'',

        fechaAlta:'',

        fechaBaja:'',

        estado:''

    });

    useEffect(() => {

        getOneDino()

    },[])

    //Metodo para obtener los datos ingresados en el form =>

    const handleInputChange = (event) => {

        setDato({

            ...dato,

            [event.target.name] : event.target.value

        })

    }

    //Metodo para gestionar el envio de datos al Servlet y BD =>

    const enviarDatos = (dato, event) => {

        actualizar(dato)

        event.preventDefault();

        event.target.reset();

    }

    //Metodo para actualizar Datos Servlet - Base de datos =>

    const actualizar = async (dato) => {

        try{

            const idDino = props.match.params.id

            const response = await axios("http://localhost:8080/Dinosaurios/DinosaurioServlet", {

                method:"GET",

                params:{

                    action:"actualizar",

                    idDino:idDino,

                    nombre:dato.nombre,

                    imagen:dato.imagen,

                    info:dato.info,

                    fuerza:dato.fuerza,

                    fechaAlta:dato.fechaAlta,

                    fechaBaja:dato.fechaBaja,

                    estado:dato.estado,

                }

            })

            const resJson = await response.data

            console.log("DATOS API => ", resJson)

            alert("Datos actualizados con exito.")

        }catch(error){

            console.log("Error => ", error)

            alert("No es posible gestionar actualizacion de datos.")

        }

    }

    //Metodo para Obtener One Dino desde el Servlet Base de Datos =>

    const getOneDino = async () => {

        try{

            const idDino = props.match.params.id

            const response = await axios("http://localhost:8080/Dinosaurios/DinosaurioServlet", {

                method:"GET",

                params:{

                    action:"buscarOne",

                    idDino:idDino,

                }

            })

            const resJson = await response.data

            console.log("DATOS API => ", resJson)

            setDato(resJson)

            //Pasar los datos obtenidos al form con setValue =>

            setValue('nombre', resJson.nombre)

            setValue('imagen', resJson.imagen)

            setValue('info', resJson.info)

            setValue('fuerza', resJson.fuerza)

            setValue('fechaAlta', moment(resJson.fechaAlta).subtract(1, 'M').format('YYYY-MM-DD'))

            setValue('fechaBaja', moment(resJson.fechaBaja).subtract(1, 'M').format('YYYY-MM-DD'))

            setValue('estado', resJson.estado)

        }catch(error){

            console.log("Error => ", error)

        }

    }

    //Metodo validar estado (activo-inactivo) =>

    const validarEstado = (estado) => {

        let validar = false

        if(estado === "activo"){

            validar = true;

        }else if(estado === "inactivo"){

            validar = true;

        }

        return validar;

    }

    //Metodo para validar que el nombre ingresado ya no se encuentre en la base de datos como "Activo" =>

    const validarNombre = async (nombre) => {

        try{

            let validar = true

            const idDino = props.match.params.id

            const response = await fetch("http://localhost:8080/Dinosaurios/DinosaurioServlet?action=buscarAll", {

                method:"GET"

            })

            const resJson = await response.json()

            console.log("DATOS API => ", resJson)

            for(let i = 0; i < resJson.length; i++){

                //Si el idDino es distinto al id pasado por parametro:

                if((idDino).toString() !== (resJson[i].idDino).toString()){

                    if(nombre === resJson[i].toString()){

                        validar = false;

                        break;

                    }

                }

            }

        }catch(error){

            console.log("Error => ", error)

        }

    }

    if(dato === null){

        return null

    }else{

        return(

            <Fragment>

            <Container>

            <Alert variant="success" className="body">

                <br></br>

                <br></br>

                <Alert.Heading className="titulo">FORMULARIO ADMIN ACTUALIZACION DINOSAURIO</Alert.Heading>

                <br></br>

                <br></br>

                <Form onSubmit={handleSubmit(enviarDatos)}>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Nombre: </label>

                    </Col>

                    <Col>

                        <input

                            type="text"

                            name="nombre"

                            onChange={handleInputChange}

                            placeholder="Ingrese el Nombre"

                            className="form-control my-2"

                            {...register("nombre", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                                validate:{

                                    validate1:validarNombre,

                                }

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                            <span className="text-danger text-small d-block mb-2">

                            {errors.nombre && errors.nombre.message}

                            </span>

                            <span className="text-danger text-small d-block mb-2">

                            {

                                errors.nombre && errors.nombre.type === "validate1" && (

                                    <div className="error">El nombre ingresado ya existe.</div>

                                )

                            }

                            </span>

                    </Col>

                </Row>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Imagen: </label>

                    </Col>

                    <Col>

                        <input

                            type="text"

                            name="imagen"

                            onChange={handleInputChange}

                            placeholder="Ingrese la Imagen"

                            className="form-control my-2"

                            {...register("imagen", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                            <span className="text-danger text-small d-block mb-2">

                            {errors.imagen && errors.imagen.message}

                            </span>

                    </Col>

                </Row>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Info: </label>

                    </Col>

                    <Col>

                        <input

                            type="text"

                            name="info"

                            onChange={handleInputChange}

                            placeholder="Ingrese la Info"

                            className="form-control my-2"

                            {...register("info", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                            <span className="text-danger text-small d-block mb-2">

                            {errors.info && errors.info.message}

                            </span>

                    </Col>

                </Row>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Fuerza: </label>

                    </Col>

                    <Col>

                        <input

                            type="number"

                            name="fuerza"

                            onChange={handleInputChange}

                            placeholder="Ingrese la Fuerza"

                            className="form-control my-2"

                            min="1"

                            max="10"

                            {...register("fuerza", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                            <span className="text-danger text-small d-block mb-2">

                            {errors.fuerza && errors.fuerza.message}

                            </span>

                    </Col>

                </Row>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Fecha de Alta: </label>

                    </Col>

                    <Col>

                        <input

                            type="date"

                            name="fechaAlta"

                            onChange={handleInputChange}

                            placeholder="Ingrese la Fecha de Alta 2020-11-05"

                            className="form-control my-2"

                            {...register("fechaAlta", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                            <span className="text-danger text-small d-block mb-2">

                            {errors.fechaAlta && errors.fechaAlta.message}

                            </span>

                    </Col>

                </Row>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Fecha de Baja: </label>

                    </Col>

                    <Col>

                        <input

                            type="date"

                            name="fechaBaja"

                            onChange={handleInputChange}

                            placeholder="Ingrese la Fecha de Baja 2020-11-05"

                            className="form-control my-2"

                            {...register("fechaBaja", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                            <span className="text-danger text-small d-block mb-2">

                            {errors.fechaBaja && errors.fechaBaja.message}

                            </span>

                    </Col>

                </Row>

                <Row>

                    <Col className="col-md-3">

                        <label className="my-2">Estado: </label>

                    </Col>

                    <Col>

                        <input

                            type="text"

                            name="estado"

                            onChange={handleInputChange}

                            placeholder="Ingrese el estado (activo-inactivo)"

                            className="form-control my-2"

                            {...register("estado", {

                                required:{

                                    value: true,

                                    message: 'Campo Obligatorio'

                                },

                                validate:{

                                    validate1:validarEstado,

                                }

                            })}

                        >

                        </input>

                    </Col>

                    <Col className="col-md-3">

                            <span className="text-danger text-small d-block mb-2">

                            {errors.estado && errors.estado.message}

                            </span>

                            <span className="text-danger text-small d-block mb-2">

                            {

                                errors.estado && errors.estado.type === "validate1" && (

                                    <div className="error">El estado es incorrecto.</div>

                                )

                            }

                            </span>

                    </Col>

                </Row>

                <Row>

                    <Col ClassName='boton'>

                        <br></br>

                        <br></br>

                        <Button type="submit" variant="primary" size="lg">ACTUALIZAR</Button>&nbsp;&nbsp;

                        <Button type="button" href={`/admin`} variant="danger" size="lg">VOLVER</Button>

                    </Col>

                </Row>

                </Form>

                <br></br>

                <br></br>

            </Alert>

            </Container>

            </Fragment>

        )

    }

}

export default Actualizar;

**Delete Logic/Eliminar Logic (Se actualiza el registro que es eliminado, eliminador logico) =>**

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

import Container from "react-bootstrap/Container";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import axios from "axios";

import Button from "react-bootstrap/Button";

import Alert from "react-bootstrap/Alert";

import Table from 'react-bootstrap/Table'

import Navigation from "./Navigation";

import { useHistory } from "react-router-dom"

import moment from 'moment';

const EliminarLogic = (props) => {

    const [dato,setDato] = useState(null);

    //HISTORY REDIRECCIONAR POR PAGINA =>

    let history = useHistory()

    useEffect(() => {

        eliminarLogic()

    },[])

    //Metodo para eliminar logicamente registro x servlet =>

    const eliminarLogic = async () => {

        try{

            const idDino = props.match.params.id;

            const response = await axios("http://localhost:8080/Dinosaurios/DinosaurioServlet", {

                method:"GET",

                params:{

                    action:"eliminarLogico",

                    idDino:idDino,

                    fechaBaja:moment().format('YYYY-MM-DD'),

                }

            })

            const resJson = await response.data

            console.log("DATOS API => ", resJson)

            history.push("/admin")

        }catch(error){

            console.log("Error => ", error)

        }

    }

    if(dato === null){

        return null

    }else{

        return(

            <Fragment>

                <Navigation></Navigation>

            </Fragment>

        )

    }

}

export default EliminarLogic;

**Recorrer con map un arreglo y pintar datos (Tabla) =>**

<tbody>

   {dato.map((planeta,i) => (

     <tr key={i}>

         <td className="celda">{planeta.nombre}</td>

         <td className="celda">{planeta.diametro}</td>

     </tr>

   ))}

</tbody>

**Pintar en Componente Tarjeta reutilizando el componente =>**

 const card = dato.map((planeta,i) => {

   return(

     <Tarjeta

       key={planeta.codigo}

       imagen={planeta.urlImg}

       nombre={planeta.nombre}

       codigo={planeta.codigo}

     />

 )

})

 <Row xs={1} md={3} className="g-4">

     { card }

 </Row>

**Obtener una lista de un array y pintarlo =>**

//Obtener lista de notas con metodo map =>

const listaNotas = dato.notas.map((nota, i) => {

 return(

   <li key={i}>{nota}</li>

 )

})

**OBTENER VALOR DE PARAMETRO URL versión 5.2.0 de react-router-dom =>**

const idDino = props.match.params.id;

**IMPORTAR IMAGEN DESDE ASSETS/IMG =>**

import Image from 'react-bootstrap/Image'

<Card.Img className="imagenCard" variant="top" src={require(`..assets/img/${props.imagen}`)} />

<Image className="fotoDetalle" src={require(`../assets/img/${dato.imagen}`)} rounded />

**Uso de Librería Moment JS para trabajar con Fechas =>**

**npm install moment --save**

<h5>{moment(dato.fechaAlta).subtract(1,'M').format('YYYY-MM-DD')}</h5>

**Redirigir página desde código JavaScript =>**

import { useHistory } from "react-router-dom"

//HISTORY REDIRECCIONAR POR PAGINA =>

    let history = useHistory()

history.push("/admin")

 //Redireccionamos al componente detalle =>

 history.push(`/detalleAlumno/${legajo}`)

 //Redirecciono y paso los datos a traves de un search =>

history.push({

    pathname: '/admin',

    search: '',

 })

**DIRECCIONAR A OTRA PAGINA DESDE HTML =>**

**import { Redirect } from 'react-router'**

**<Redirect to='/somewhere'/>**

**PASAR Y OBTENER PARAMETROS FORMA 2 DESDE SEARCH (RECOMENDADO) =>**

**NO HACE FALTA MODIFICAR LA ROUTE EN APP.js COMO PARAMETROS OPCIONALES**

**Se envían los datos desde el componente NAVIGATION.js =>**

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

**PAGINA RECIVE ESTA EN ESCUCHA DE PARAMS => VER PROYECTO ALUMNOS HOOKS (Ver Líneas de Código GrillaDino con Busqueda)**

//Obtengo los datos pasados por search URL =>

    let {search} = useLocation();

    let query = new URLSearchParams(search)

    const [nomUrl, setNomUrl] = useState(query.get("nombre"))

    const [dato, setDato] = useState(null)

    useEffect(() => {

        getDino()

        //De esta forma esta a la escucha de las modificaciones en los search URL =>

        setNomUrl(query.get("nombre"))

    },[query.get("nombre")])

**Recorrer con un map la colección de datos dentro de un Select =>**

<select id="dino2" name="dino2" onChange={handleInputChange}>

     {dato.map((dino,i) => (

         <option key={i} value={dino.nombre}>{dino.nombre}</option>

     ))}

</select>

**Busquedas con el método Filter() devuelve un array =>**

**METODO FILTER =>**

**El método Filter crea un nuevo arreglo, el cual no busca modificar el valor de los elementos para el nuevo arreglo,**

**únicamente retorna aquellos elementos que cumplan con las condiciones del filtro, los cuales serán almacenados en otro arreglo.**

**const array = [**

**{ id: 1, username: "Mapper"},**

**{ id: 2, username: "Finder"},**

**{ id: 3, username: "Eaching"}**

**];**

**const filtered = array.filter(function(element){**

**return element.id > 2;**

**});**

**//Output: [{ id: "3", username: "Eaching"}]**

**Búsqueda para obtener coincidencia por el nombre =>**

//Filtramos la busqueda para obtener el dato buscado por urlNombre =>

const encontrado = resJson.filter((alumno) => (alumno.legajo).toString() === (urlLegajo).toString())

const encontrado = resJson.filter(function (element){

     return (element.nombre).toString() === (nomURL).toString()

})

**OTRA BUSQUEDA CON FILTER =>**

**//Aplico el elemento de Busqueda filter devuelve array a diferencia de find que devuelve un unico elemento =>**

**const encontrado = resJson.filter(function (element){**

**var matcher = new RegExp(desURL, 'i')**

**return matcher.test([element.descripcion].join())**

**})**

**setDato(encontrado)**

//Filtramos con el metodo Filter buscando el dato descripcion buscador =>

const encontrado = resJson.filter((planeta) => {

     var matcher = new RegExp(urlDescripcion, 'i')

     return matcher.test([planeta.descripcion].join())

})

setDato(encontrado)

**OTRA BUSQUEDA CON FILTER EN NUMEROSOS CAMPOS EJ NOMBRE-BIOGRAFIA-ARRAY PODERES =>**

**//Aplicamos la busqueda en nombre y bio =>**

**const encontrado = response.filter(function (element){**

**var matcher = new RegExp(busquedaURL, 'i')**

**return matcher.test([element.nombre, element.bio, element.poderes].join())**

**})**

**OTRA BUSQUEDA CON FILTER PASANDO UN ARRAY DENTRO DE UN ARRAY =>**

**//Filter utilizado para buscar array dentro de array =>**

**const lista = resJson.filter(function (element){**

**return element.meses.includes(parseInt(mesUrl))**

**})**

//Filtramos la busqueda por array dentro de otro array =>

 const encontrado = resJson.filter((zodiaco) => {

   return zodiaco.meses.includes(parseInt(mesURL))

 })

 setDato(encontrado)

**METODO DE BUSQUEDA FIND =>**

**//Utilizamos la funcion find() para obtener el planeta x codigo buscado =>**

 //Busco el alumno asociado al legajo con metodo find =>

const encontrado = resJson.find((alumno) => (alumno.legajo).toString() === (legajo).toString())

**METODOS PARA ORDENAR DE FORMA ASCENDENTE Y DESCENDENTE =>**

**Gestionamos el metodo sort() para ordenar objetos en forma descendente (Mayor a Menor) =>**

 //Ordenar los datos con metodo sort descendente (Mayor a Menor) =>

const listaOrdenada = resJson.sort(function (a, b){

return (b.diametro - a.diametro)

})

setDato(listaOrdenada)

**Gestionamos el metodo sort() para ordenar objetos en forma ascendente (Menor a Mayor) =>**

//Ordenar los datos con metodo sort ascendente (Menor a Mayor) =>

const listaOrdenada = resJson.sort(function (a, b){

return (a.diametro - b.diametro)

})

setDato(listaOrdenada)

**ordenar(){**

**console.log("ENTRO ORDENAR=>")**

**//Si es impar**

**if(this.contador % 2 !== 0){**

**console.log("ENTRO IMPAR ASCENDENTE=>")**

**this.listaFacturas = this.listaFacturas.sort(function (a, b){**

**return (a.importe - b.importe)**

**})**

**this.contador++;**

**}else{**

**console.log("ENTRO PAR DESCENDENTE=>")**

**this.listaFacturas = this.listaFacturas.sort(function (a, b){**

**return (b.importe - a.importe)**

**})**

**this.contador++;**

**}**

**}**

**Metodo para calcular Promedio =>**

 //Metodo para calcular el promedio Notas =>

    const calcularPromedio = (array) => {

        let promedio = 0

        let suma = 0

        for(let i = 0; i < array.length; i++){

            suma += array[i]

        }

        return suma / array.length

    }

**Metodo para calcular mejor Promedio =>**

//Metodo para calcular el mejorPromedio =>

    const calcularMejorPromedio = (listaAlumnos) => {

        let legajo = 0

        let mejor = calcularPromedio(listaAlumnos[0].notas)

        let promedio = 0

        for(let i = 0; i < listaAlumnos.length; i++){

            promedio = calcularPromedio(listaAlumnos[i].notas)

            if(mejor < promedio){

                mejor = promedio;

                legajo = listaAlumnos[i].legajo

            }

        }

        return legajo

    }

**NAVIGATION CON COMBO BOX DE BUSQUEDA =>**

import React, {Component, Fragment, useState, useEffect} from 'react';

import Button from 'react-bootstrap/Button';

import Navbar from 'react-bootstrap/Navbar';

import Nav from 'react-bootstrap/Nav';

import FormControl from 'react-bootstrap/FormControl';

import Form from 'react-bootstrap/Form';

import Container from "react-bootstrap/Container";

import Label from "react-bootstrap/FormLabel";

import { useHistory } from "react-router-dom"

const Navigation = (props) => {

    //HISTORY REDIRECCIONAR POR PAGINA =>

    let history = useHistory()

    const[dato,setDato] = useState({

        mes:null,

    })

    useEffect(() => {

    },[])

    //ESTE METODO SE PUEDE USAR PARA CAPTURAR LA INFORMACION INGRESADA EN EL FORM:

    const handleInputChange = (event) => {

        setDato({

            ...dato,

            [event.target.name] : event.target.value

        })

    }

    const obtenerMes = () => {

        if(dato.mes !== null || dato.mes !== ""){

            console.log("DATO SELECCIONADO COMBO => ", dato.mes)

            //Redirecciono y paso los datos a traves de un search =>

            history.push({

                pathname: '/',

                search: '',

                search: `?mes=${dato.mes}`,

            })

        }else{

            alert("Error. Debe seleccionar una opcion de mes en el combo.")

        }

    }

    return (

        <Fragment>

        <Navbar bg="primary" variant="dark">

            <Navbar.Brand href="/">HOME</Navbar.Brand>

            <Nav className="me-auto">

            <Nav.Link href="/agrupado">AGRUPADO</Nav.Link>

            </Nav>

            <Form className="d-flex">

                <Form.Label>MES:</Form.Label>&nbsp;&nbsp;

                <select name="mes" onChange={handleInputChange}>

                    <option value="">Seleccione un Mes</option>

                    <option value="1">Enero</option>

                    <option value="2">Febrero</option>

                    <option value="3">Marzo</option>

                    <option value="4">Abril</option>

                    <option value="5">Mayo</option>

                    <option value="6">Junio</option>

                    <option value="7">Julio</option>

                    <option value="8">Agosto</option>

                    <option value="9">Septiembre</option>

                    <option value="10">Octubre</option>

                    <option value="11">Noviembre</option>

                    <option value="12">Diciembre</option>

                </select>&nbsp;&nbsp;

                <Button variant="outline-success" onClick={obtenerMes}>SEARCH</Button>

            </Form>

        </Navbar>

        </Fragment>

    );

}

export default Navigation;

**OTRA FORMA DE COLOCAR LA LOGICA DESPUES DE ELSE =>**

 if(dato === null){

        return null

 }else{

        const id = props.match.params.id

        const encontrado = dato.find((zodiaco) => {

            return(

                (zodiaco.id).toString() === (id).toString()

            )

  })

        console.log("ENCONTRADO => ", encontrado)

**SELECCIONAR MESES DENTRO DE UN ARRAY X NOMBRE =>**

const meses = dato.meses.map((mes,i) => {

            if(mes === 1){

                return(

                    <li key={i}>Enero</li>

                )

            }

            if(mes === 2){

                return(

                    <li key={i}>Febrero</li>

                )

            }

            if(mes === 3){

                return(

                    <li key={i}>Marzo</li>

                )

            }

//Etc los siguientes nombre

))}