FullStack Developer

React – react-router-dom

¿Qué es React Router DOM?

React Router DOM es una librería que nos permite manejar el enrutamiento en aplicaciones React de manera sencilla y eficiente. Nos ayuda a definir qué componentes se deben mostrar cuando la URL cambia, sin recargar toda la página. Esto es clave para crear aplicaciones de una sola página (SPA, *Single Page Applications*).

Conceptos clave de React Router DOM:

1.BrowserRouter: Es el componente que envolvemos alrededor de nuestra aplicación para habilitar el uso de rutas. Internamente, utiliza la API de historia del navegador (history API) para mantener la interfaz sincronizada con la URL.

```
import { BrowserRouter as Router } from 'react-router-dom';

function App() {
  return (
    <Router>
      {/* Aquí irán las rutas */}
    </Router>
  );
}
```

Routes y **Route**: Estos componentes son los que nos permiten definir las rutas y los componentes que queremos renderizar. Routes es el contenedor donde ponemos nuestras rutas, y Route define una ruta específica.

```
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';

function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about" element={<About />} />
      </Routes>
    </Router>
  );
}
```

En este ejemplo, cuando la URL es /, se renderiza el componente Home, y cuando la URL es /about, se renderiza About.

Link: Para navegar dentro de la aplicación, en lugar de usar etiquetas <a> (que recargan la página), usamos el componente **Link**, que permite cambiar la URL sin refrescar la página.

```
import { Link } from 'react-router-dom';

function NavBar() {
  return (
    <nav>
      <Link to="/">Home</Link>
      <Link to="/about">About</Link>
    </nav>
  );
}
```

useNavigate: Es un hook que nos permite navegar programáticamente dentro de la aplicación. Es útil cuando queremos redirigir al usuario desde el código, por ejemplo, después de un formulario exitoso.

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

function GoDashboard() {
  const navigate = useNavigate();

  const handleGo = () => {
    // Lógica de redirección
    navigate('/dashboard'); // Redirigir
  };

  return <button onClick={handleGo}>Go to Dashboard</button>;
}

export default GoDashboard;
```

useParams: Este hook nos permite acceder a los parámetros dinámicos que pasamos en las rutas. Por ejemplo, si tenemos una ruta que incluye un parámetro como /user/:id, useParams nos da acceso a ese id.

```
import { useParams } from 'react-router-dom';

function UserProfile() {
    const { id } = useParams();

    return <h1>Perfil de usuario {id}</h1>;
}
```

Ventajas de usar React Router DOM:

1.Navegación sin recarga: Nos permite cambiar de vistas sin recargar la página, lo que mejora la experiencia del usuario en aplicaciones de una sola página.
2.Rutas dinámicas: Podemos crear rutas que cambian dinámicamente en función de parámetros (como id en un perfil de usuario).
3.Historial del navegador: React Router DOM maneja el historial del navegador para que la navegación funcione correctamente (como el botón de atrás y adelante del navegador).
4.Control programático: Con hooks como useNavigate, podemos controlar la navegación desde cualquier lugar de la aplicación.

Resumen:

React Router DOM es una herramienta fundamental para crear aplicaciones React con múltiples páginas o vistas sin recargar la página. Nos permite manejar rutas dinámicas, usar parámetros en la URL, y controlar la navegación de manera sencilla con componentes y hooks. Además, es muy flexible y fácil de integrar en cualquier proyecto de React.

Vamos a crear un proyecto en React que utilice **React Router DOM** y **Bootstrap**

Paso 1: Crear el proyecto de React con Vite
Primero, necesitamos crear el proyecto de React utilizando Vite. Para esto, abrimos la terminal y ejecutamos el siguiente comando:

```
npm create vite@latest .
```

Paso 2: Instalar dependencias
Una vez creado el proyecto, debemos instalar las dependencias necesarias. Primero, instalamos **React Router DOM** y **Bootstrap**:

```
npm install react-router-dom bootstrap
```

Paso 3: Configurar Bootstrap

Para agregar Bootstrap a nuestro proyecto, necesitamos importarlo en nuestro archivo src/App.jsx:

```
import 'bootstrap/dist/css/bootstrap.min.css'
import 'bootstrap/dist/js/bootstrap.bundle.min.js'

function App() {

  return (
```

Paso 4: Creamos un Layout

Vamos a crear un componente Layout que contendrá la barra de navegación (y eventualmente el pie de página o cualquier otra estructura común). El contenido específico de cada ruta se va a renderizar dentro de este layout:

```
import Footer from "./components/Footer"
import NavBar from "./components/NavBar"

function Layout({ children }) {
    return (
        <>
            <NavBar/>
            {children}
            <Footer/>
        </>
    )
}

export default Layout
```

Paso 5: Utilizar el Layout en App.jsx

Vamos a envolver cada ruta en el Layout, para que siempre se muestre la estructura común (navegación y pie de página):

```jsx
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import Home from './components/Home';
import About from './components/About';
import Contact from './components/Contact';
import Layout from './components/Layout';
import 'bootstrap/dist/css/bootstrap.min.css'
import 'bootstrap/dist/js/bootstrap.bundle.min.js'

function App() {
  return (
    <Router>
      <Layout>
        <Routes>
          <Route path="/" element={<Home/>} />
          <Route path="/about" element={<About/>} />
          <Route path="/contact" element={<Contact/>} />
        </Routes>
      </Layout>
    </Router>
  );
}

export default App;
```

Paso 5: Utilizar el Layout en App.jsx

Vamos a envolver cada ruta en el Layout, para que siempre se muestre la estructura común (navegación y pie de página):

```jsx
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import Home from './components/Home';
import About from './components/About';
import Contact from './components/Contact';
import Layout from './components/Layout';
import 'bootstrap/dist/css/bootstrap.min.css'
import 'bootstrap/dist/js/bootstrap.bundle.min.js'

function App() {
  return (
    <Router>
      <Layout>
        <Routes>
          <Route path="/" element={<Home/>} />
          <Route path="/about" element={<About/>} />
          <Route path="/contact" element={<Contact/>} />
        </Routes>
      </Layout>
    </Router>
  );
}

export default App;
```

Paso 6: Crear los componentes de las páginas
Al igual que antes, creamos los componentes para cada una de las páginas. Creamos una carpeta llamada components dentro de src y agregamos los archivos Home.jsx, About.jsx, y Contact.jsx.

(Esto queda como ejercicio realizar cada componente necesario)

Paso 7: Ejecutar el proyecto
Con todo configurado, podemos ejecutar el proyecto utilizando el comando:

```
npm run dev
```

Resultado esperado:

•Verás una barra de navegación estilizada con **Bootstrap** en la parte superior, con enlaces a "Home", "About", y "Contact".
•Al hacer clic en estos enlaces, las rutas cambiarán sin recargar la página, gracias a **React Router DOM**.
•Las páginas estarán estilizadas de forma básica con Bootstrap.

Resumen:

1.Creamos un proyecto de React utilizando Vite.
2.Instalamos **React Router DOM** y **Bootstrap**.
3.Configuramos el enrutamiento con BrowserRouter, Routes, y Route.
4.Creamos componentes para las páginas Home, About y Contact.
5.Ejecutamos el proyecto con npm run dev.
¡Con este setup tenés una aplicación React optimizada y moderna usando Vite, Bootstrap para los estilos, y React Router DOM para la navegación!