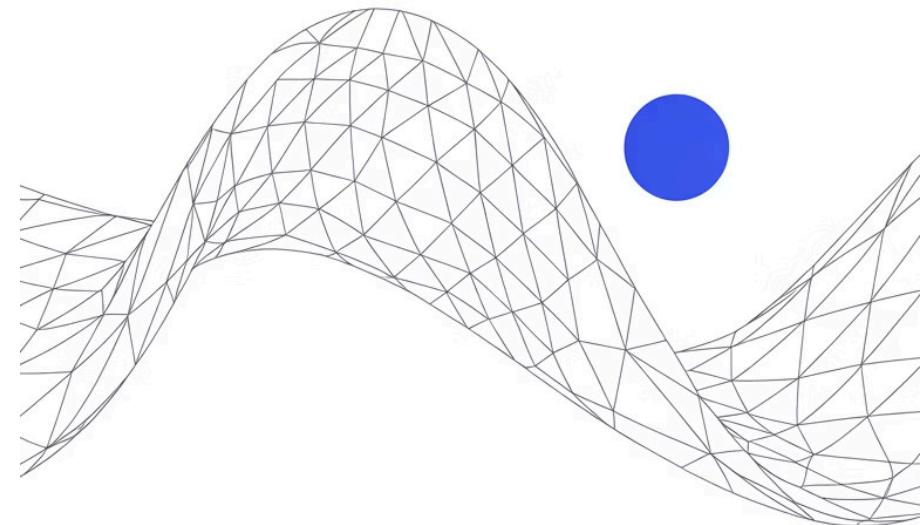


Curso de Desarrollo Frontend con React

Rutas y Navegación en React con React Router



Seamless navigation for React applications

Prmeeesitivinvation for react palq. losicitiov for withe tooetin ch
toe tappiinpilationonde: fng stloee u intciing rodicte cewang foor
racinatcear pder theivery irneç. Even: toating furiat:ttoiiwad tany piyice
wec ornes tayiner tout for erpuir coimies.

¿Qué Aprenderás Hoy?



1 Fundamentos de React Router

Conceptos básicos y diferencias con el routing tradicional

2 Componentes Esenciales

BrowserRouter, Routes, Route y navegación

3 Parámetros Dinámicos

Manejo de URLs con useParams

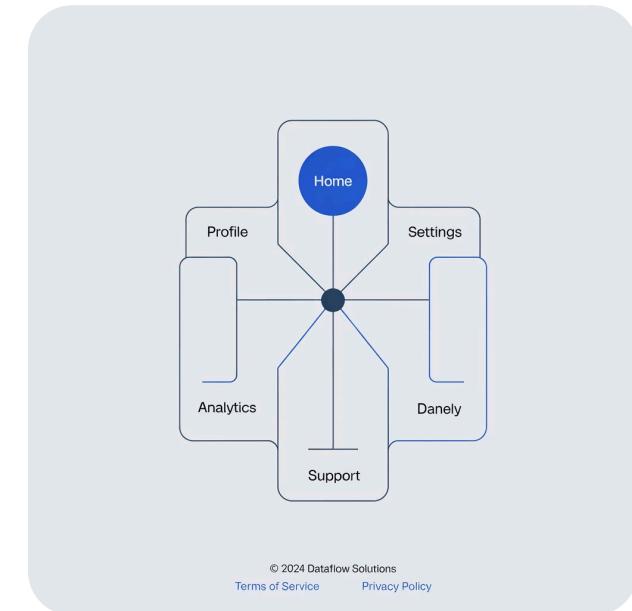
4 Proyecto Práctico

Aplicación completa paso a paso

¿Qué es React Router?

React Router es la **biblioteca estándar** para manejar rutas en aplicaciones React. Permite crear **Single Page Applications (SPA)** con múltiples vistas sin recargar la página completa.

Es esencial para cualquier aplicación React que necesite más de una pantalla o vista.



¿Por Qué Necesitamos React Router?

Experiencia de Usuario

Navegación rápida sin recargas de página completa

URLs Significativas

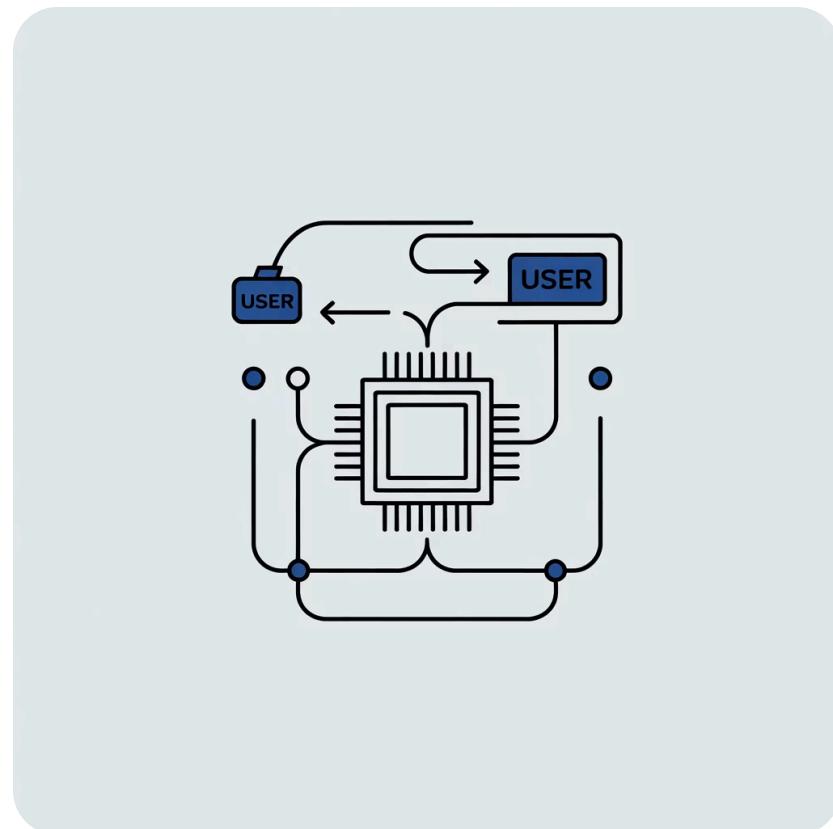
Cada vista tiene su propia URL que se puede compartir

Historial del Navegador

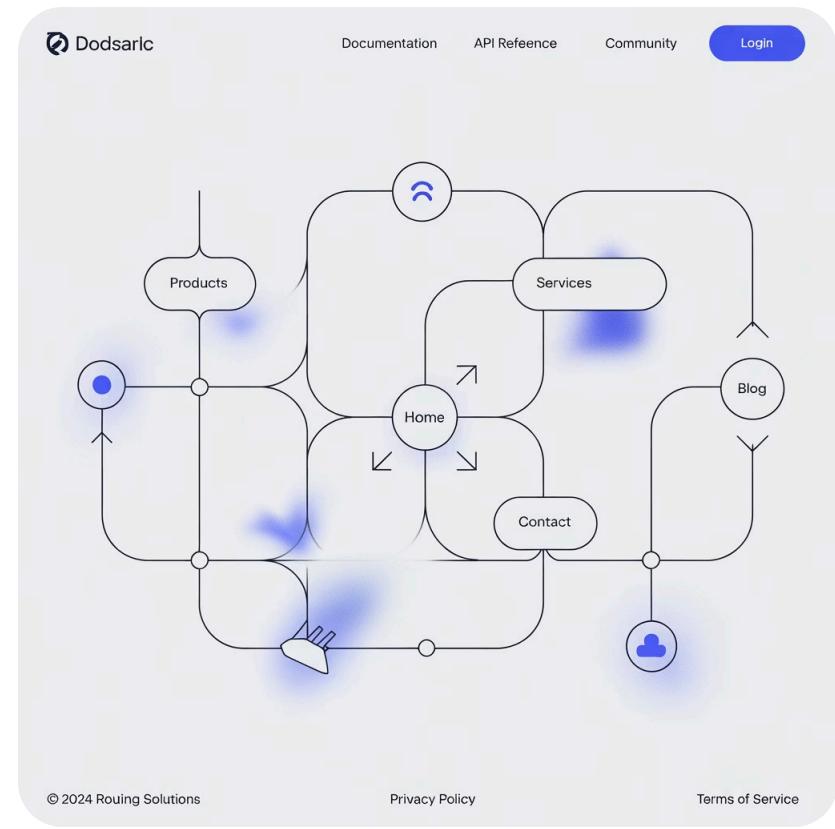
Los botones atrás/adelante funcionan correctamente

Routing Tradicional vs React Router

Routing Tradicional



React Router (SPA)

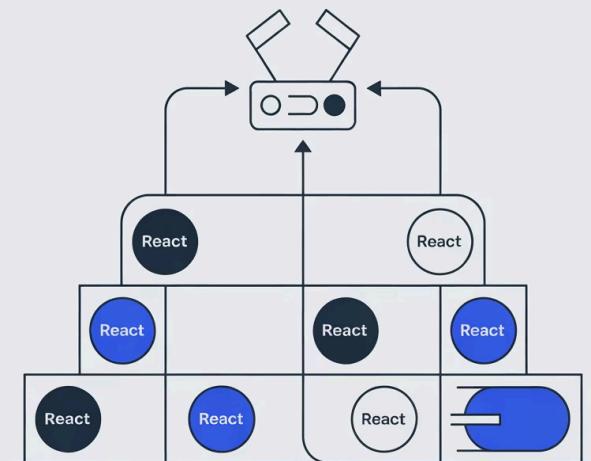


- Cada URL es una solicitud al servidor
- Recarga completa de la página
- Pérdida de estado de la aplicación
- Tiempo de carga más lento

- JavaScript maneja las rutas
- Sin recarga de página
- Preserva el estado de la aplicación
- Navegación instantánea

Conceptos Fundamentales

BrowserRouter: El Contenedor Principal



BrowserRouter es el componente raíz que habilita el routing en toda tu aplicación.

Se coloca en el nivel más alto de tu aplicación, típicamente en `App.js` o `index.js`.

Utiliza la **HTML5 History API** para mantener la UI sincronizada con la URL.

Routes y Route: Definiendo las Rutas

Routes

Contenedor que agrupa todas las rutas posibles de tu aplicación

Route

Define una ruta específica con su path y el componente a renderizar

Juntos forman el **sistema de routing** que determina qué componente mostrar según la URL actual.

Ejemplo Básico de Estructura

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

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

}
```

Rutas Absolutas vs Relativas

Rutas Absolutas

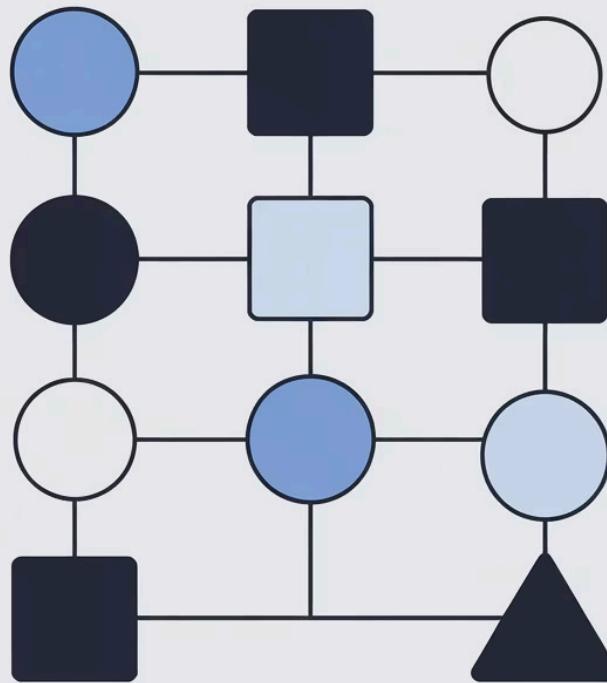
Comienzan con "/" y definen el path completo desde la raíz

/usuarios , /productos/detalle

Rutas Relativas

No comienzan con "/" y se basan en la ruta actual
perfil , configuracion

React router



¿Cómo se Renderizan los Componentes?

React Router compara la URL actual con los **paths definidos** en tus Routes y renderiza el primer componente que coincida.

Solo se renderiza **un componente a la vez**, creando una experiencia de navegación fluida.

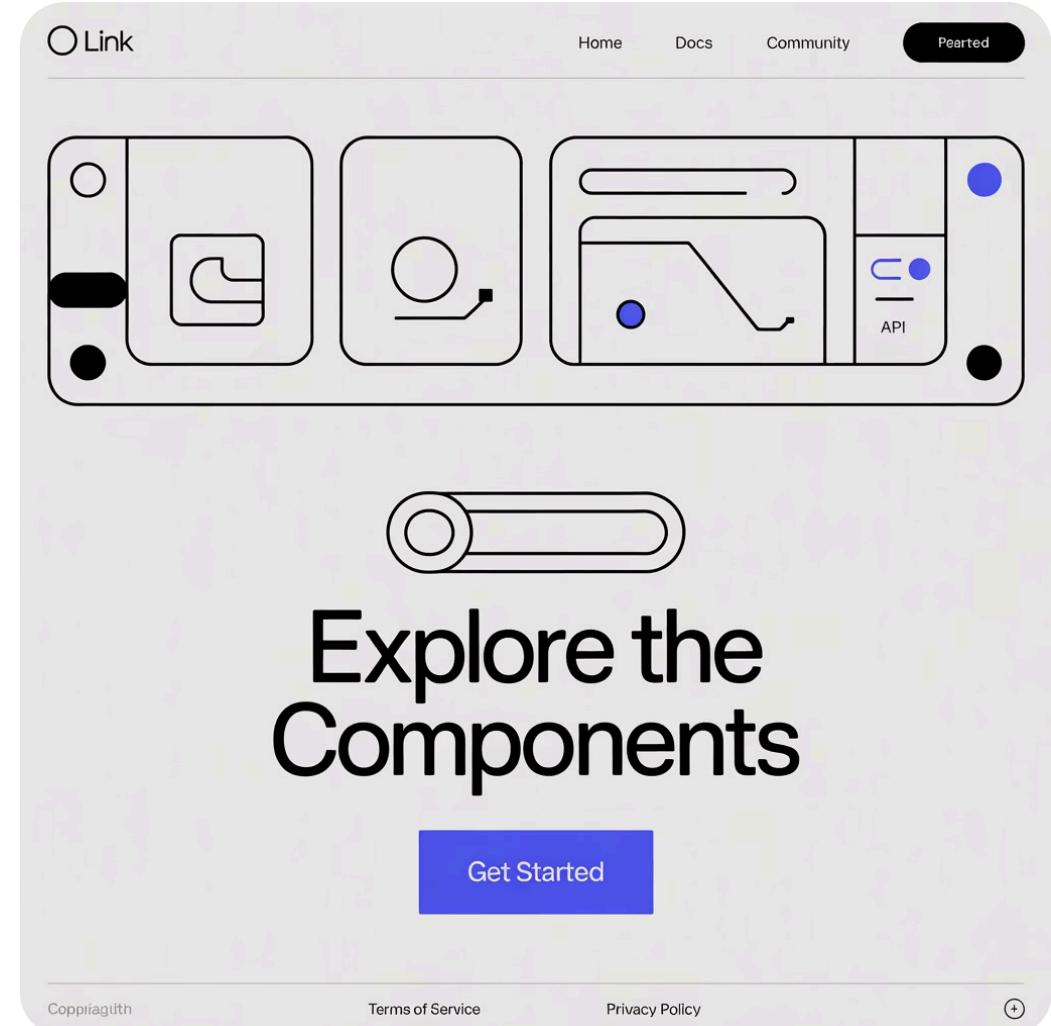
Navegación en React Router

Link: Navegación Básica

El componente **Link** crea enlaces que navegan entre rutas sin recargar la página.

Es el reemplazo de React Router para las etiquetas `<a>` tradicionales.

```
<Link to="/perfil">  
  Mi Perfil  
</Link>
```



NavLink: Navegación con Estilos Activos

NavLink extiende Link añadiendo la capacidad de mostrar estilos diferentes cuando la ruta está activa.

```
<NavLink  
  to="/productos"  
  className={({ isActive }) =>  
    isActive ? "nav-active" : "nav-inactive"  
  }  
>  
  Productos  
</NavLink>
```

Perfecto para **menús de navegación** donde quieras resaltar la página actual.

Diferencias: Link vs NavLink

Link

- Navegación básica
- No detecta estado activo
- Más liviano
- Ideal para enlaces simples

NavLink

- Navegación con estado activo
- Aplica clases CSS automáticamente
- Perfecto para menús
- Mayor control visual

Ejemplo Práctico: Barra de Navegación

```
import { NavLink } from 'react-router';

function Navbar() {
  return (
    <nav>
      <NavLink to="/" end>Inicio</NavLink>
      <NavLink to="/productos">Productos</NavLink>
      <NavLink to="/contacto">Contacto</NavLink>
    </nav>
  );
}
```

El atributo `end` asegura que "/" solo esté activo en la página de inicio, no en subrutas.

Para instalarlo, usa: `npm install react-router`

Parámetros en Rutas

Unlock data URL Parameters

Visualize and manage url parameters effectively

Learn more,



¿Qué son los Parámetros de Ruta?

Los **parámetros de ruta** permiten crear URLs dinámicas que pueden cambiar según los datos.

Por ejemplo: /usuario/123, /producto/laptop-gaming

Son esenciales para mostrar contenido específico basado en la URL.

Definiendo Rutas con Parámetros

Sintaxis de Parámetros

```
<Route
  path="/usuario/:id"
  element={<Perfil />}
/>

<Route
  path="/producto/:nombre"
  element={<Detalle />}
/>
```

URLs que Coincidan

- /usuario/123
- /usuario/ana-lopez
- /producto/laptop
- /producto/mouse-gaming

Hook useParams: Accediendo a los Parámetros

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

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

  return (
    <div>
      <h2>Perfil del Usuario</h2>
      <p>ID del usuario: {id}</p>
    </div>
  );
}
```

useParams() devuelve un objeto con todos los parámetros de la URL actual.

Múltiples Parámetros

Definición de Ruta

```
<Route  
  path="/tienda/:categoria/:producto"  
  element={<Producto />}  
/>
```

Uso en Componente

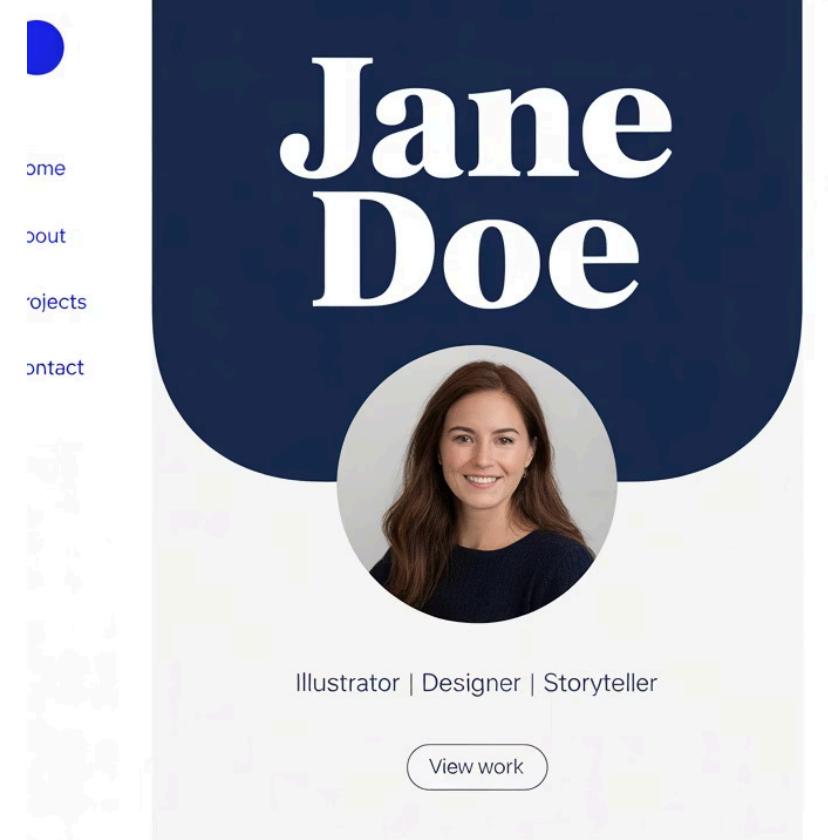
```
function Producto() {  
  const { categoria, producto } = useParams();  
  
  return (  
    <div>  
      <h2>{categoria}</h2>  
      <h3>{producto}</h3>  
    </div>  
  );  
}
```

Proyecto Práctico

Aplicación de Ejemplo: Portfolio Personal

Vamos a crear una aplicación con **tres páginas**: Inicio, Perfil y Detalles de proyecto.

Incluirá navegación, parámetros dinámicos y buenas prácticas.



Paso 1: Instalación y Configuración

```
# Crear proyecto con Vite
```

```
npm create vite@latest portfolio-react -- --template react  
cd portfolio-react  
npm install
```

```
# Instalar React Router
```

```
npm install react-router
```

```
# Estructura de carpetas
```

```
src/  
  components/  
    Navbar.jsx  
    Home.jsx  
    Perfil.jsx  
    Detalles.jsx  
  App.jsx  
  main.jsx
```

Paso 2: Configurar BrowserRouter

```
// App.jsx
import { BrowserRouter, Routes, Route } from 'react-router';
import Navbar from './components/Navbar';
import Home from './components/Home';
import Perfil from './components/Perfil';
import Detalles from './components/Detalles';

function App() {
  return (
    <BrowserRouter>
      <Navbar />
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/perfil/:id" element={<Perfil />} />
        <Route path="/detalles/:proyecto" element={<Detalles />} />
      </Routes>
    </BrowserRouter>
  );
}

export default App;
```

Paso 3: Crear la Barra de Navegación

```
// Navbar.jsx
import { NavLink } from 'react-router';

function Navbar() {
  return (
    <nav style={{ padding: '1rem', backgroundColor: '#8C98CA' }}>
      <NavLink
        to="/"
        style={({ isActive }) => ({
          color: isActive ? 'white' : 'lightgray',
        })}
        end
      >
        Inicio
      </NavLink>
      <NavLink to="/perfil/123">Mi Perfil</NavLink>
      <NavLink to="/detalles/proyecto-web">Proyectos</NavLink>
    </nav>
  );
}

export default Navbar;
```

Paso 4: Componente Home

```
// Home.jsx
import { Link } from 'react-router';

function Home() {
  return (
    <div style={{ padding: '2rem' }}>
      <h1>Bienvenido a Mi Portfolio</h1>
      <p>Desarrollador Frontend especializado en React</p>
      <div>
        <Link to="/perfil/ana-developer">Ver mi perfil</Link>
        <br />
        <Link to="/detalles/ecommerce-react">Ver mis proyectos</Link>
      </div>
    </div>
  );
}

export default Home;
```

Paso 5: Componente Perfil con useParams

```
// Perfil.jsx
import { useParams } from 'react-router';

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

  return (
    <div style={{ padding: '2rem' }}>
      <h1>Perfil de Usuario</h1>
      <p>ID del usuario: <strong>{id}</strong></p>
      <div>
        <h3>Información Personal</h3>
        <p>Nombre: {id.replace('-', ' ')})</p>
        <p>Especialidad: React Developer</p>
      </div>
    </div>
  );
}

export default Perfil;
```

Paso 6: Componente Detalles

```
// Detalles.jsx
import { useParams, Link } from 'react-router';

function Detalles() {
  const { proyecto } = useParams();

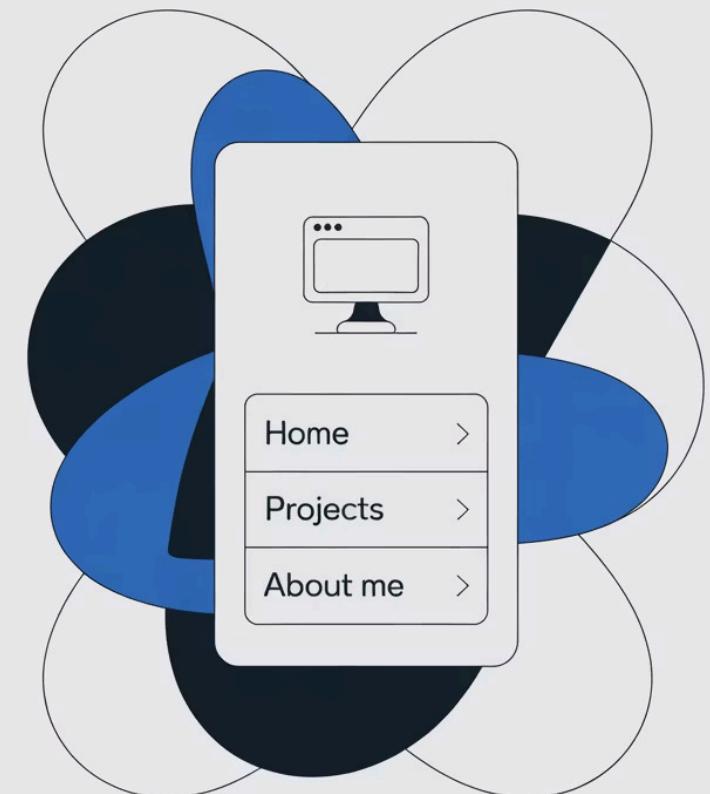
  const proyectos = {
    'ecommerce-react': 'Tienda Online con React y Redux',
    'portfolio-personal': 'Portfolio Personal Responsivo',
    'blog-gatsby': 'Blog Estático con Gatsby'
  };

  return (
    <div style={{ padding: '2rem' }}>
      <h1>Detalles del Proyecto</h1>
      <h2>{proyectos[proyecto] || 'Proyecto no encontrado'}</h2>
      <Link to="/">Volver al inicio</Link>
    </div>
  );
}

export default Detalles;
```

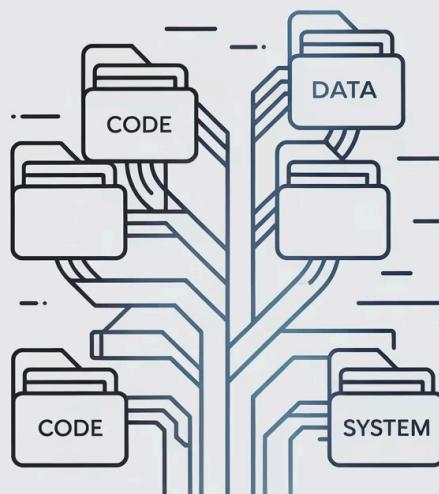
Resultado Final

- Navegación entre múltiples páginas
- Parámetros dinámicos en las URLs
- Enlaces que funcionan sin recargar la página
- Indicadores visuales de la página activa



Buenas Prácticas

Organización de Rutas en Archivo Separado



Para aplicaciones grandes, es recomendable crear un archivo `routes.js` separado.

Esto mejora la **mantenibilidad** y hace que el código sea más **legible**.

Centraliza la lógica de routing en un solo lugar.

Ejemplo: routes.js

```
// routes.js

import Home from './components/Home';
import Perfil from './components/Perfil';
import Detalles from './components/Detalles';

// Instalación: npm install react-router

export const routes = [
  {
    path: '/',
    element: Home,
    exact: true
  },
  {
    path: '/perfil/:id',
    element: Perfil
  },
  {
    path: '/detalles/:proyecto',
    element: Detalles
  }
];
```

Rutas Anidadas para Aplicaciones Complejas



Dashboard Principal

/dashboard - Layout general con sidebar



Secciones Anidadas

/dashboard/usuarios - Gestión de usuarios



Detalles Específicos

/dashboard/usuarios/123 - Usuario específico

Evita Rutas Duplicadas o Confusas

Rutas Confusas

- /user y /usuario
- /productos y /items
- Parámetros ambiguos: /:id

Rutas Claras

- Nomenclatura consistente
- Parámetros descriptivos: /:userId
- Jerarquía lógica clara

Manejo de Rutas 404

```
<Routes>
  <Route path="/" element={<Home />} />
  <Route path="/perfil/:id" element={<Perfil />} />
  <Route path="/detalles/:proyecto" element={<Detalles />} />

  {/* Ruta catch-all para 404 */}
  <Route path="*" element={<NotFound />} />
</Routes>
```

La ruta con `path="*"` captura todas las URLs que no coincidan con rutas anteriores.

Lazy Loading para Mejor Performance

```
import { lazy, Suspense } from 'react';

const Home = lazy(() => import('./components/Home'));
const Perfil = lazy(() => import('./components/Perfil'));

function App() {
  return (
    <div>
      <h1>React.lazy</h1>
      <Suspense fallback={<p>Cargando...</p>}>
        <Home />
        <Perfil />
      </Suspense>
    </div>
  );
}

export default App;
```

Pregunta de Reflexión

React Router vs Condicionales Simples

¿Qué ventajas ofrece el uso de React Router frente a manejar el enrutamiento solo con condicionales (if, switch) en React?



URLs Reales

Cada vista tiene una URL única y compatible



Historial

Funciona con los botones atrás/adelante del navegador



Bookmarks

Los usuarios pueden guardar páginas específicas



Performance

Lazy loading y code splitting automático

#EDCOUNIANDES

<https://educacioncontinua.uniandes.edu.co/>

Contacto: educacion.continua@uniandes.edu.co

© - Derechos Reservados: La presente obra, y en general todos sus contenidos, se encuentran protegidos por las normas internacionales y nacionales vigentes sobre propiedad Intelectual, por lo tanto su utilización parcial o total, reproducción, comunicación pública, transformación, distribución, alquiler, préstamo público e importación, total o parcial, en todo o en parte, en formato impreso o digital y en cualquier formato conocido o por conocer, se encuentran prohibidos, y solo serán lícitos en la medida en que se cuente con la autorización previa y expresa por escrito de la Universidad de los Andes.