«Talento Tech»

Desarrollo Web 3

Clase 07

Buenos Aires Ciudad

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Clase N° 7 | Navegación en React

Temario:

- Creación de rutas y navegación usando la última versión de React Router DOM 6

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Creación de rutas y navegación usando la última versión de React Router DOM 6

La creación de rutas y navegación es crucial en muchas aplicaciones **React** para facilitar la visualización y navegación entre diferentes vistas. Con la última versión de **React Router DOM 6**, se han introducido algunos cambios importantes. Vamos a explorar cómo implementar rutas y navegación utilizando **React Router DOM 6**.

Instalación de React Router DOM 6

Primero, asegúrate de tener instalada la última versión de React Router DOM:

```
npm install react-router-dom@^6.0.
```

Creación de un Router y Definición de Rutas:

Ahora, crearemos un componente principal llamado “App” que actuará como el contenedor principal para nuestras rutas. Definiremos algunas rutas básicas para ilustrar cómo funciona.

```
// App.jsx
import React from 'react';
import { BrowserRouter as Router, Routes, Route, Link } from 'react-router-dom';
import Home from './Home';
import About from './About';
import Contact from './Contact';

const App = () => {
  return (
    <Router>
      <nav>
        <ul>
```

```
        <li><Link to="/">Home</Link></li>
        <li><Link to="/about">About</Link></li>
        <li><Link to="/contact">Contact</Link></li>
      </ul>
    </nav>

    <Routes>
      <Route path="/" element={<Home />} />
      <Route path="/about" element={<About />} />
      <Route path="/contact" element={<Contact />} />
    </Routes>
  </Router>
 );
};

export default App;
```

En este ejemplo:

Importamos BrowserRouter como Router, y Routes, Route, y Link de react-router-dom.
Creamos un componente “App” que utiliza el componente Router para envolver nuestras rutas.
Utilizamos el componente “Link” para crear enlaces de navegación.
Definimos rutas utilizando los componentes Routes y Route. Cada Route tiene una path (ruta) y un element que representa el componente que se debe renderizar cuando la ruta coincida.

Creación de componentes para las rutas definidas

Cada uno de estos componentes será una sección de nuestra web.
Habrá un **Home, About y un Contact.**

Buenos Aires aprende
Agencia de Habilidades para el Futuro

```jsx
// Home.jsx
import React from 'react';

const Home = () => {
  return (
    <div>
      <h2>Home</h2>
      <p>Bienvenido a la página de inicio.</p>
    </div>
  );
};

export default Home;
```

```jsx
// About.jsx
import React from 'react';

const About = () => {
  return (
    <div>
      <h2>About</h2>
      <p>Esta es la página de acerca de nosotros.</p>
    </div>
  );
};

export default About;
```

```jsx
// Contact.jsx
import React from 'react';
```

```jsx
const Contact = () => {
  return (
    <div>
      <h2>Contact</h2>
      <p>Contáctanos en contact@example.com</p>
    </div>
  );
};

export default Contact;
```

Navegación y Uso de Rutas

Al correr la aplicación, podrás ver la barra de navegación con enlaces a "Home", "About", y "Contact". Al hacer clic en estos enlaces, los componentes asociados se renderizarán en la aplicación. (sumar imagen que muestre el resultado de esta acción?)

Este es un ejemplo básico de cómo implementar rutas y navegación con React Router DOM 6. Puedes expandir este patrón para manejar rutas más complejas y mejorar la experiencia de navegación en tu aplicación. ¡Esperamos que esto te ayude a comenzar con React Router DOM 6!

Desafío #3 Parte 1

Creación de Aplicación de Gestión de Usuarios

Misión:
Crear una aplicación web utilizando **React** que permita realizar operaciones **CRUD** (Crear, Leer, Actualizar, Eliminar) en una lista de usuarios proveniente de una **API** simulada en **MockAPI.io.** Además, se deberá implementar la navegación entre las diferentes secciones de la aplicación utilizando la última versión de React Router DOM (versión 6). Lo iremos creando durante el transcurso de los Desafíos restantes (#3 y #4) y la Clase 10.

Primera Parte: Configuración y CRUD básico

En esta fase, se centrarán en configurar el proyecto, realizar las operaciones CRUD básicas y mostrar la lista de usuarios.

1. Configuración Inicial:

Utiliza Create React App para iniciar un nuevo proyecto.

```
npx create-react-app user-management-app
cd user-management-app
```

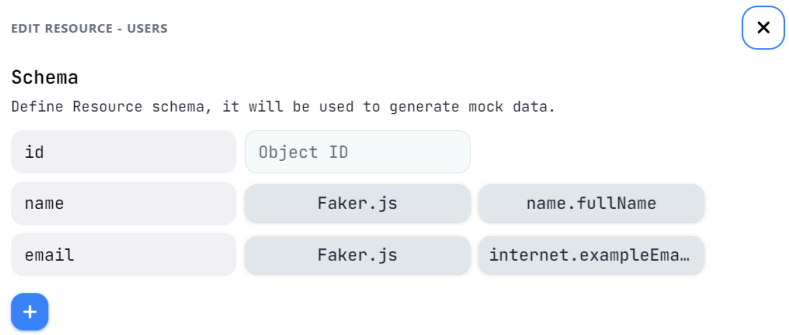
Instala React Router DOM 6

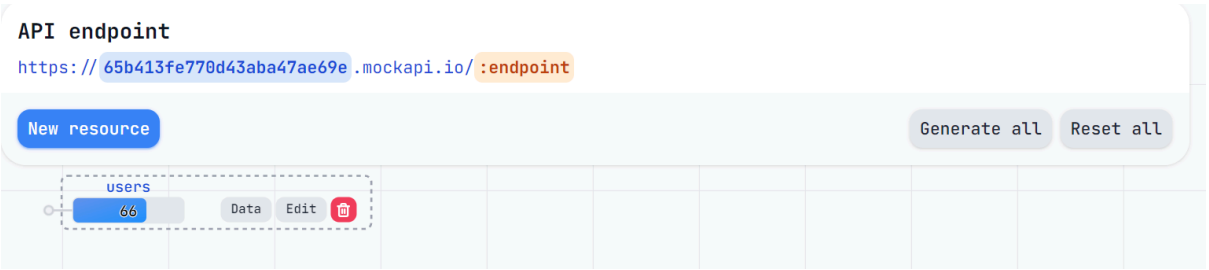
```
npm install react-router-dom
```

2. Integración con MockAPI.io:

Registra una cuenta en MockAPI.io y crea una nueva API para gestionar usuarios. La colección de datos se llamara 'users' y tendra 3 tipos de datos. ID, name y email. Seleccionar los valores segun la imagen de ejemplo mas abajo.

Una vez teniendo lo mismo que veas en las imagenes damos al boton 'Create'





4.Crear Usuarios:

a) Implementa la funcionalidad para agregar nuevos usuarios. Vamos a crear un componente 'UserForm.jsx' para el formulario de creación.

```
import React, { useState } from 'react';

const UserForm = ({ addUser }) => {
  const [name, setName] = useState('');
  const [email, setEmail] = useState('');

  const handleSubmit = () => {
    // Validar datos antes de agregar
    const newUser = { name, email };

    // Llamar a la función desde las props para agregar usuario
```

```
    addUser(newUser);
  };

  return (
    <div>
      <h2>Agregar Usuario</h2>
      <label>Nombre: </label>
      <input type="text" value={name} onChange={(e) => setName(e.target.value)}
/>
      <br />
      <label>Email: </label>
      <input type="text" value={email} onChange={(e) => setEmail(e.target.value)} />
      <br />
      <button onClick={handleSubmit}>Agregar</button>
    </div>
  );
};

export default UserForm;
```

b) En 'App.jsx' implementa el formulario para agregar nuevos usuarios.

```
// En el componente principal
import React, { useState } from 'react';
import UserForm from './UserForm';

const App = () => {
  const [users, setUsers] = useState([]);

  const addUser = (newUser) => {
    // Agregar nuevo usuario al estado
     try {
    const response = await fetch('URL_DE_TU_API', {
```

```
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(newUser),
    });

    if (response.ok) {
      // Obtener la respuesta y agregar usuario al estado
      const data = await response.json();
      setUsers([...users, data]);
    } else {
      console.error('Error al agregar usuario');
    }
  } catch (error) {
    console.error('Error en la solicitud: ', error);
  }
  };

  return (
    <div>
      <h1>Lista de Usuarios</h1>
      <UserForm addUser={addUser} />
      {/* Mostrar lista de usuarios */}
      <ul>
        {users.map((user, index) => (
          <li key={index}>{user.name} - {user.email}</li>
        ))}
      </ul>
    </div>
  );
};

export default App;
```

5. Listar Usuarios:

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Crea un componente UserList.jsx para mostrar la lista de usuarios.

```jsx
import React, { useEffect, useState } from 'react';

const UserList = () => {
  const [users, setUsers] = useState([]);

  useEffect(() => {
    // Llamar a la función para obtener la lista de usuarios
    fetchUsers();
  }, []);

  const fetchUsers = async () => {
    try {
      const response = await fetch('URL_DE_TU_API');
      const data = await response.json();
      setUsers(data);
    } catch (error) {
      console.error('Error en la solicitud: ', error);
    }
  };

  return (
    <div>
      <h1>Lista de Usuarios</h1>
      {/* Mostrar lista de usuarios */}
      <ul>
        {users.map((user) => (
          <li key={user.id}>{user.name} - {user.email}</li>
        ))}
      </ul>
    </div>
  );
};

export default UserList;
```

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Esta es la primera parte del proyecto, donde se configura la aplicación, se implementa el **CRUD** básico y se muestra la lista de usuarios. En las siguientes partes, se abordará la navegación y la actualización de usuarios.

Hasta este punto, deberás tener una aplicación funcional que pueda agregar usuarios y mostrar una lista de ellos. Corre el comando 'npm run dev'. ¡Exitos! 😉


Buenos Aires
aprende
Buenos
Aires
Ciudad