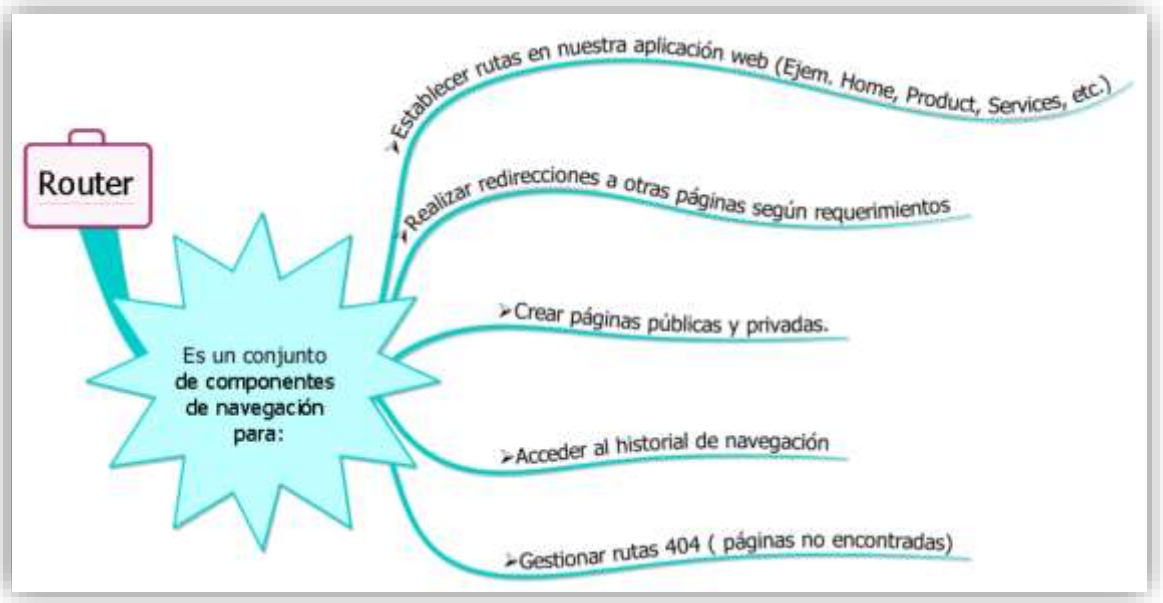
Actividad de Aprendizaje N° 9A

Enrutadores y Estilos en React

1. Router en React

¿Qué es un Router?



Instalar Router en React

> **Nota:** Si instaló su proyecto con *Create React App* o *Vite*... estos *NO INCLUYEN el componente Router*.

Para *instalar React Router*, ejecutar el comando NPM *(desde el directorio raíz de la aplicación)*:

```
>npm install -D react-router-dom
```

Para *instalar React Router*, la más reciente versión ejecutar el comando NPM con el parámetro @latest:

```
>npm install -D react-router-dom@latest
```

Estructura de directorio

Para crear una aplicación con varias páginas, es necesario organizar los directorios donde se guardarán estas páginas. En el directorio ***src***, crearemos un directorio con el nombre de ***pages***:

Ejemplo:

Crear una aplicación con 4 páginas (Home, Blogs, Contact y NoPage), las 3 primeras deberán tener su Link y la cuarta es para devolver cuando se ingrese una url con una página no encontrada.

1. Creando el Componente de Navegación

pages/Layout.jsx

```
import {Outlet, Link} from "react-router-dom";
const Layout = function() {
    return (
        <>
            <nav>
                <ul>
                    <li>
                        <Link to="/">Home</Link>
                    </li>
                    <li>
                        <Link to="/blogs">Blogs</Link>
                    </li>
                    <li>
                        <Link to="/contact">Contact</Link>
                    </li>
                </ul>
            </nav>
            <Outlet />
        </>
    )
};

export default Layout;
```

2. Creando los componentes de página

pages/Home.jsx

```
const Home = function() {
    return <h1>Home la pagina</h1>
};

export default Home;
```

pages/Blogs.jsx

```
const Blogs = function () {
    return <h1>Blogs me aqui</h1>
}

export default Blogs;
```

pages/Contact.jsx

```
const Contact = function() {
    return <h1>Contact me aquiii</h1>
}

export default Contact;
```

pages/NoPage.jsx

```
const NoPage = function() {
    return <h1>404 Página NO encontrada</h1>
}

export default NoPage;
```

3. Modificando ./App.jsx, responder a los Routers,
   Se define <BrowserRouter><Routes>
   <Route> para Layout y los otros.

/App.jsx

```
import { BrowserRouter, Routes, Route } from "react-router-dom";
import Layout from "./pages/Layout";
import Home from "./pages/Home";
import Blogs from "./pages/Blogs";
import Contact from "./pages/Contact";
import NoPage from "./pages/NoPage";
import './App.css'


function App() {
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element={<Layout />}>
          <Route index element={<Home />} />
          <Route path="blogs" element={<Blogs />} />
          <Route path="contact" element={<Contact />} />
          <Route path="*" element={<NoPage />} />
        </Route>
      </Routes>
    </BrowserRouter>
  );
}
```

4. El main.jsx

```
import React, { Component, StrictMode } from 'react'
import ReactDOM from 'react-dom/client'
import App from './App'
import './index.css'

ReactDOM.createRoot(document.getElementById('root')).render
  (
    <StrictMode>
      <App />
    </StrictMode>
  )
```

2. Estilos CSS en React

Hay muchas formas de diseñar React con CSS, las tres formas más comunes son:

- estilo en línea
- hojas de estilo CSS
- Módulos CSS

Estilos Inline:

Agregar la propiedad:

```
style={ objetoConEstilosCSS }
```

Los estilo en línea, deben ser definidos en un objeto de JavaScript:

{ color: “red” }

Ejemplo

Crear un tag h1 con estilo color rojo.

```
const NoPage = function() {
    return <h1 style={{color: "red"}}>404 Página NO encontrada</h1>
}

export default NoPage;
```

Ejemplo

Crear un tag h1 con un objeto, con varios estilos definidos en un const.

```
const Contact = function() {
    const myStyle={
        color: "orange",
        backgroundColor: "black",
        padding: "30px",
        fontFamily: "forte"
    };
    return <h1 style={myStyle}>Contact me aqui</h1>
}

export default Contact;
```

Estilos con Stylesheet:

Escriba su estilo CSS en un archivo separado con la extensión *.css, los archivos importados contienen propiedades de los tags generales de html.

./App.css

```
body {
  background-color: #282c34;
  color: white;
  padding: 40px;
  font-family: Sans-Serif;
  text-align: center;
}
```

Importar en el componente

```
import './App.css'
const Blogs = function () {
    return <h1 >Blogs me aqui</h1>
}

export default Blogs;
```

Estilos en Módulos:

Escriba su estilo CSS en un archivo separado con la extensión *.module.css, los archivos importados contienen estilos de clase también id.

../css/esti.module.css

```
.clase1{
    background-color: blueviolet;
    color: aquamarine;
    padding: 50px;
}
```

```
import estilos from '../css/esti.module.css'
const Blogs = function () {
    return <h1 className={estilos.clase1}>Blogs me aqui</h1>
}

export default Blogs;
```

Componentes Estilos:

Existen muchas librerías CSS que recodificaron sus estilos en componentes.

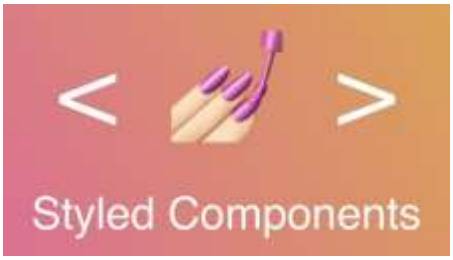
¿Qué significa eso?

Aplicar un estilo con:

```
 <ComponenteEstilo>
       'Aquí HTML
</ComponenteEstilo>
```

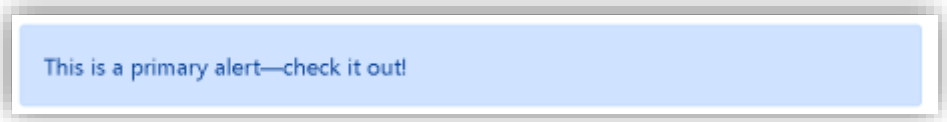
Librería de Estilos: Styled Components

```
import styled from "styled-components";

const BotonXL = styled.button`
  font-size: 1.5em;
  font-weight: bold;
  color: #fff;
  background-color: #00f;
  border-radius: 5px;
  padding: 10px;
  margin: 10px;
`;

function Componente1() {
  return <BotonXL>botón 1</BotonXL>;
}
```



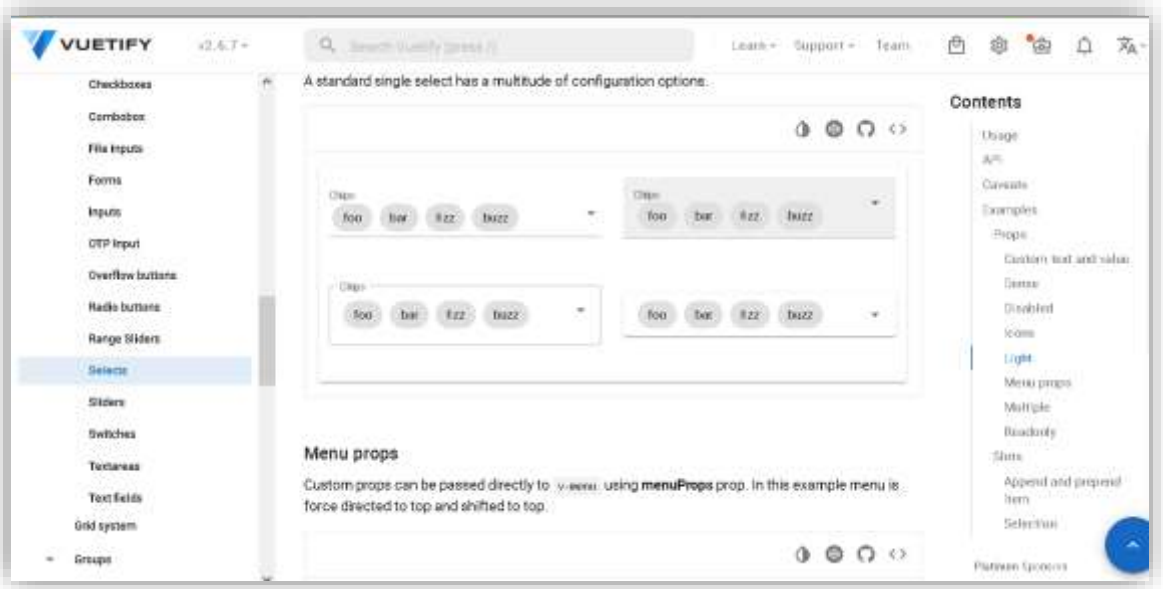
Librería de Estilos: React Bootstrap

https://react-bootstrap.github.io/

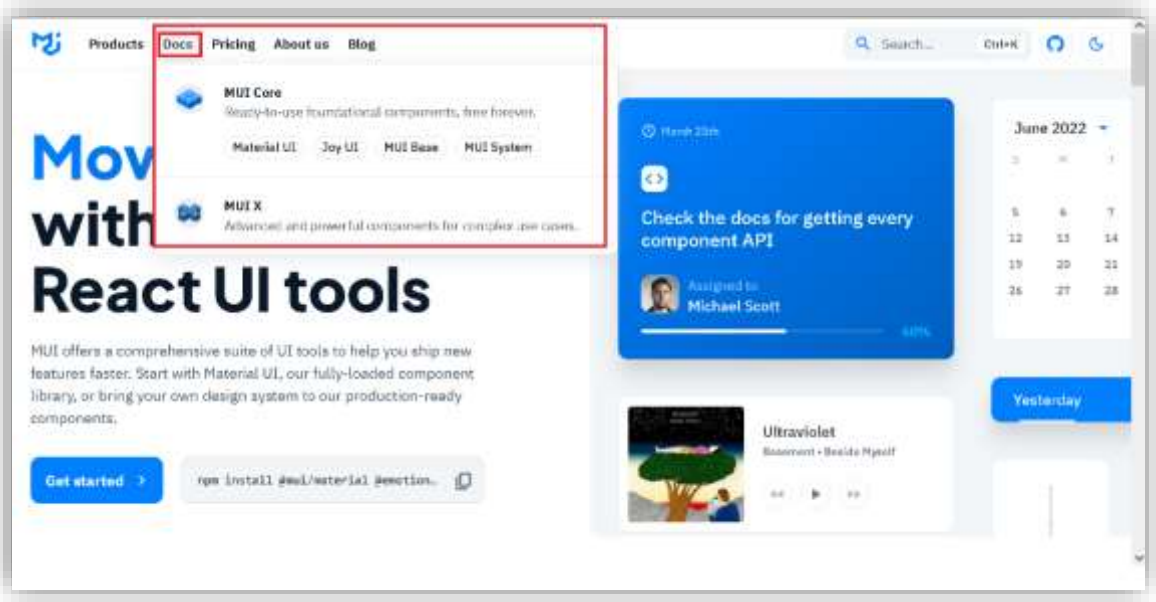



```
<Alert key={variant} variant={variant}>
  This is a {variant} alert—check it out!
</Alert>
```

Librería de Estilos: Vuetify solo para VUE JS



Librería de Estilos: Material UI para React



3. Librería tailWind CSS en React

Instalación

Instale tailwindcss y 2 dependencias a través de npm

Luego ejecute el comando npx init para generar tanto tailwind.config.js como postcss.config.js.

```
>npm install -D tailwindcss postcss autoprefixer

>npx tailwindcss init -p
```

Nota: debe estar en la carpeta del proyecto Vite.

Configuración

Configurar los path de plantilla

Agregar los paths de todos los archivos que utilizarán tailWind en el archivo tailwind.config.js.

```
module.exports = {
  content: [
    './index.html',
    './src/**/*.jsx'
  ],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

Configurar las directivas Tailwind a su CSS

Agregue las directivas @tailwind para cada una de las capas de Tailwind a su archivo ./src/index.css.

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

Listo para Utilizar

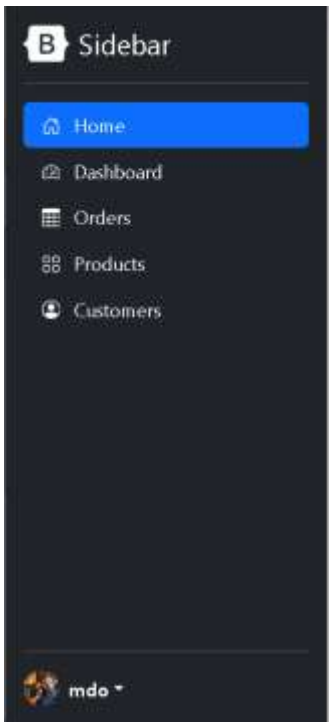
Con className.

Evaluación de competencia

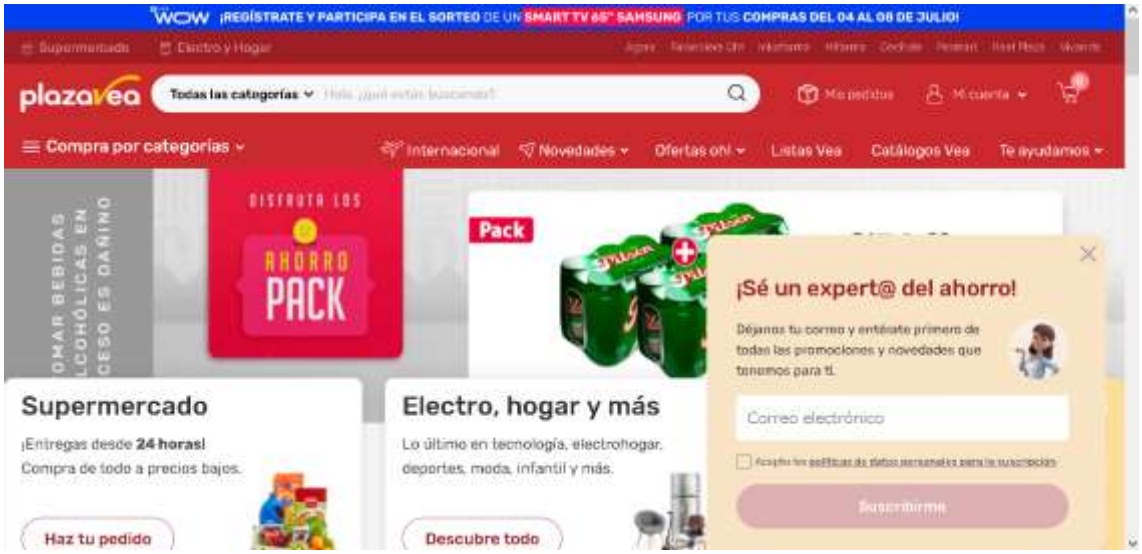
1. Desarrolle la siguiente interfaz en un componente Header en React e inserte 4 fonts, utilizando un fontawesome u otro.

2. Desarrolle la siguiente interface (un carrusel) en un componente Carrusel en React utilizando las mismas imágenes de la página https://www.imf.org/es/home .

3. Desarrolle la siguiente interfaz en un componente SideBar en React, utilizando el framework tailwind.



4. Desarrolle el componente <nav> de la siguiente interfaz que contenga *Compra por categoría, internacional, novevades, ofertas oh, Listas Vea, Catalogo Vea Te ayudamos* utilizando Router de React.



5. Desarrolle el componente <modal> de la siguiente interfaz que contenga *se un experto en ahorros*