«Talento Tech»

React JS

Clase 03

Buenos Aires Ciudad

Clase N° 3 | Layout en React

Índice:

- Creación de la estructura básica de la aplicación.
- Desarrollo de los primeros componentes reutilizables (Header, Footer, Nav, Main, Gallery).
- Visualización de los componentes en el navegador.
- TalentoLab

Objetivos de la Clase:

- Aprender a estructurar un proyecto React desde cero.
- Crear componentes reutilizables que representen partes fundamentales de una interfaz.
- Comprender cómo ensamblar y renderizar estos componentes en el navegador.

Creación de la Estructura Básica

Crear un nuevo proyecto React con Vite

1. Abre una terminal y ejecuta el siguiente comando para crear el proyecto:
   npm create vite@latest mi-proyecto-react --template react
   Aquí `mi-proyecto-react` es el nombre del proyecto.
   Puedes cambiarlo por otro.

2. Accede al directorio del proyecto recién creado:
   cd mi-proyecto-react

3. Instala las dependencias necesarias:
   npm install

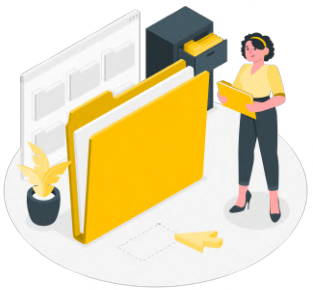
4. Inicia el servidor de desarrollo:
   npm run dev
   Esto abrirá la aplicación predeterminada de React en tu navegador en `http://localhost:5173`.

En el directorio `src`, vamos a crear una carpeta llamada `components` donde organizaremos nuestros componentes. Dentro de esta carpeta, crearemos los archivos de cada componente:

- `Header.jsx`
- `Nav.jsx`
- `Main.jsx`
- `Gallery.jsx`
- `Footer.jsx`

Además, crearemos una carpeta llamada `styles` en `src` para almacenar los estilos.
Estructura del proyecto:

```
src/
├── components/
│   ├── Header.jsx
│   ├── Nav.jsx
│   ├── Main.jsx
│   ├── Gallery.jsx
│   └── Footer.jsx
├── styles/
│   └── style.css
├── App.jsx
└── main.jsx
```



Creación de Componentes Reutilizables

Header.jsx

Este componente representa la cabecera del sitio web. Su propósito es mostrar un título principal.

```
import React from 'react';

function Header() {
    return (
        <header style={{ backgroundColor: "#4CAF50", padding: "10px",
textAlign: "center", color: "white" }}>
            <h1>Bienvenidos a mi App React</h1>
        </header>
    );
}

export default Header;
```

Este componente usa estilos en línea para establecer el fondo, el color del texto y la alineación. Es la parte superior visible de nuestra página.

Nav.jsx

El componente `Nav` actúa como un menú de navegación simple con enlaces simulados.

```
import React from 'react';

function Nav() {
    return (
        <nav style={{ backgroundColor: "#333", color: "white", padding:
"10px" }}>
            <ul style={{ listStyle: "none", display: "flex",
justifyContent: "space-around", margin: 0 }}>
                <li><a href="#" style={{ color: "white",
textDecoration: "none" }}>Inicio</a></li>
                <li><a href="#" style={{ color: "white",
textDecoration: "none" }}>Acerca de</a></li>
```

```
                <li><a href="#" style={{ color: "white",
textDecoration: "none" }}>Contacto</a></li>
            </ul>
        </nav>
    );
}

export default Nav;
```

Cada enlace del menú está diseñado con estilos en línea para que luzcan uniformes y profesionales.

Main.jsx

Este componente se usa para el contenido principal de la página.

```
import React from 'react';

function Main() {
    return (
        <main style={{ padding: "20px" }}>
            <h2>Contenido Principal</h2>
            <p>Este es un ejemplo de contenido dentro del área
principal.</p>
        </main>
    );
}

export default Main;
```

Es el lugar donde se colocará información central, como texto, imágenes u otros componentes.

Gallery.jsx

El componente `Gallery` muestra una lista de imágenes.

```
import React from 'react';

function Gallery() {
    const images = [
        "https://via.placeholder.com/150",
        "https://via.placeholder.com/150/0000FF",
        "https://via.placeholder.com/150/FF0000"
    ];

    return (
        <section style={{ display: "flex", gap: "10px", justifyContent:
"center", marginTop: "20px" }}>
            {images.map((src, index) => (
                <img key={index} src={src} alt={`Imagen ${index + 1}`}
style={{ width: "150px", height: "150px" }} />
            ))}
        </section>
    );
}

export default Gallery;
```

Las imágenes se generan dinámicamente desde un array, lo que hace al componente flexible y escalable.

Footer.jsx

Este componente actúa como el pie de página del sitio.

```
import React from 'react';

function Footer() {
    return (
        <footer style={{ backgroundColor: "#f1f1f1", padding: "10px",
textAlign: "center", marginTop: "20px" }}>
            <p>&copy; 2024 - Mi Aplicación React</p>
        </footer>
    );
}
```

```
export default Footer;
```

Agrega un mensaje al final de la página con estilo simple.

Ensamblaje de Componentes en App.jsx

En `App.jsx`, juntamos todos los componentes:

`App.jsx` es el contenedor principal que organiza y muestra los diferentes componentes.

```
import React from 'react';
import Header from './components/Header';
import Nav from './components/Nav';
import Main from './components/Main';
import Gallery from './components/Gallery';
import Footer from './components/Footer';

function App() {
    return (
        <div>
            <Header />
            <Nav />
            <Main />
            <Gallery />
            <Footer />
        </div>
    );
}
export default App;
```

Visualización en el Navegador

Ejecuta el servidor con:

npm run dev

Deberías ver todos los componentes ensamblados correctamente en el navegador.

Reflexión Final

Aprendimos a estructurar un proyecto React desde cero, enfocándonos en la creación de componentes reutilizables que representan las partes fundamentales de una interfaz de usuario, como el Header, Nav, Main, Gallery y Footer. Estos componentes nos permiten modularizar el código y mejorar su mantenibilidad, además de fomentar la reutilización en diferentes contextos.

La práctica de ensamblar estos componentes en App.jsx no sólo consolidó conceptos básicos de React, sino que también nos mostró cómo trabajar de manera colaborativa en proyectos más grandes. Este enfoque facilita la organización del proyecto y establece una base sólida para desarrollar aplicaciones más complejas en el futuro.

Segunda fase del proceso para sumarte a Talento Lab

En este segundo ejercicio práctico te desafiamos a demostrar tu creatividad y habilidades técnicas para estructurar y diseñar una interfaz dinámica y funcional utilizando React. Prepárate para este reto, que evaluará tu capacidad para trabajar con componentes, props y estilos.

Ejercicio Práctico:

Crea una página dinámica para TalentoLab

1. **Crea un componente `EquipoTalentoLab`:**

   Este componente debe recibir como prop un array de objetos, donde cada objeto represente a un miembro del equipo.

   - **Propiedades de cada objeto:** `nombre`, `rol`, e imagen.

- El componente debe mostrar una tarjeta para cada miembro con su foto, nombre y rol.

Ejemplo del array: `const equipo = [`

```
        { nombre: 'Silvia', rol: 'Product Owner', imagen:
'https://via.placeholder.com/100' },

        { nombre: 'Luis', rol: 'Diseñador UX/UI', imagen:
'https://via.placeholder.com/100' },

        { nombre: 'Matías', rol: 'Desarrollador', imagen:
'https://via.placeholder.com/100' },

        { nombre: 'Sabrina', rol: 'Desarrolladora', imagen:
'https://via.placeholder.com/100' },

      ];
```

Diseño sugerido: Cada tarjeta debe incluir:

- La imagen del miembro (con estilos para redondear bordes).
- Su nombre destacado.
- Una descripción breve del rol que desempeña.

---

2. **Crea un componente TarjetaProyecto:**
   Este componente debe recibir las siguientes props:
   - titulo: Nombre del proyecto.
   - descripcion: Detalles del proyecto.
   - botonTexto: Texto de un botón que invite a saber más.

   Ejemplo de uso:

```
      <TarjetaProyecto

          titulo="Plataforma de Gestión"

          descripcion="Una herramienta para optimizar la gestión de
equipos."
```

```
        botonTexto="Explorar proyecto"

    />
```

3. **Interactividad:**
    - Al hacer clic en el botón, muestra un mensaje en la consola que diga: Explorando: [titulo del proyecto].

---

3. **Crea un componente GaleriaIntereses:**
   Este componente debe recibir un array de temas como prop y mostrar un botón para cada uno.
    - **Interactividad:** Al hacer clic en un botón, cambia su color de fondo dinámicamente.

   Array de ejemplo:

```
    const intereses = ['React', 'JavaScript', 'APIs', 'Diseño UX',
'Node.js'];
```

Cómo ensamblar todo:

1. En App.jsx, importa los componentes creados y úsalos para construir la página:
    - Muestra la lista del equipo con EquipoTalentoLab.
    - Destaca los proyectos utilizando TarjetaProyecto.
    - Agrega una galería interactiva con GaleriaIntereses.
2. Usa estilos para que la página sea atractiva y organizada. Considera aplicar flexbox o grid para el diseño de las secciones.

Objetivo del ejercicio práctico:

Construir una interfaz que muestre:

- La **lista del equipo** con sus roles e imágenes.
- Una **sección de proyectos** con tarjetas interactivas.
- Una **galería de intereses** con botones que cambien dinámicamente.

Este desafío pone a prueba tu capacidad para reutilizar componentes, manejar props y diseñar interfaces dinámicas. ¡Sorpréndenos con tu solución y demuestra por qué TechLab es tu lugar ideal! 🚀

Materiales y Recursos Adicionales:

- ★ Documentación oficial de React - Componentes
- ★ Vite - Getting Started
- ★ Placeholder.com
- ★ CSS Tricks - A Complete Guide to Flexbox
- ★ CSS Tricks - A Complete Guide to Grid

Preguntas para Reflexionar:

- ¿Por qué es útil dividir una aplicación en componentes en React?
- ¿Qué papel juegan las props en la personalización de los componentes?
- ¿Cómo ayuda Vite en el desarrollo de aplicaciones React?

Próximos Pasos:

- Gestión del estado local con useState.
- Manejo de eventos (clics, formularios).
- Primer ejercicio práctico: agregar productos al carrito.

Buenos Aires aprende

Agencia de Habilidades para el Futuro

Buenos Aires Ciudad