



Agencia de Habilidades
para el Futuro

<Talento
Tech/>

ReactJS

Clase 03 | Layout en React

¡Les damos la bienvenida!



Vamos a comenzar a grabar la clase.

Índice

JXS y Componentes

- Repaso de JSX y componentes funcionales.
- Creación de componentes más complejos.
- Bienvenida a TechLab
- Situación Inicial en TechLab

Layout en React

- **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**

React Hooks y Eventos

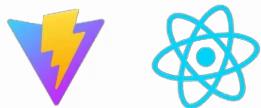
- Introducción al manejo del estado local con useState.
- Comprensión y manejo de eventos en React (clics, formularios).
- Proyecto final Talento Lab

Objetivos de la Clase

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

Creación de la Estructura Básica

Creación de un Nuevo Proyecto React



Vite + React

Crear Proyecto con Vite

Ejecuta: `npm create vite@latest mi-proyecto-react --template react`

1

Instalar Dependencias

Instala las dependencias necesarias: `npm install`

2

Acceder al Directorio

Navega al directorio del proyecto: `cd mi-proyecto-react`

3

Iniciar Servidor

Inicia el servidor de desarrollo: `npm run dev`

4

Carpetas

Carpetas

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:

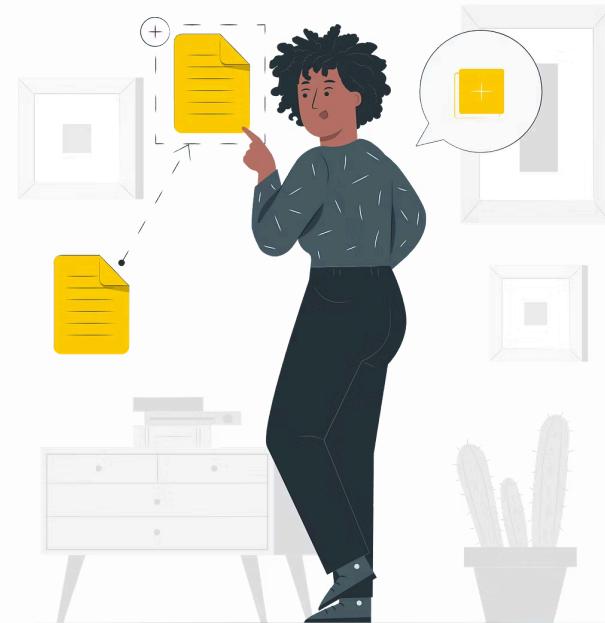
Header.jsx

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

Componente Header

Propósito

El componente Header representa la cabecera del sitio web y muestra un título principal.

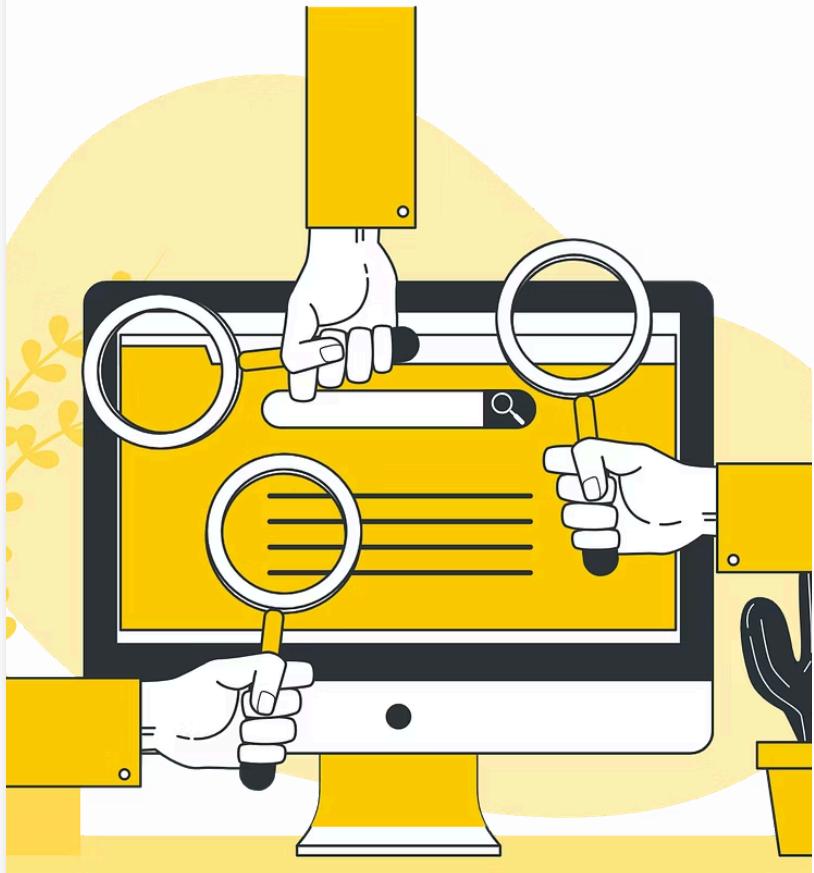
Implementación

Se crea un componente funcional que retorna un elemento header con estilos en línea para el fondo, color de texto y alineación.

Header.jsx

Ejemplo de uso:

```
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>
    );
}
```



Componente Nav

1 Definición

Crear componente funcional Nav

2 Estructura

Implementar menú de navegación simple

3 Estilos

Aplicar estilos en línea para uniformidad

Nav.jsx

Ejemplo de uso:

```
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;
```

```
</ul>
</nav>
);

}

export default Nav;
```

Componente Main

Contenido Principal

Área central de la página

Estructura Flexible

Diseñado para albergar contenido diverso

Estilos Básicos

Main.jsx

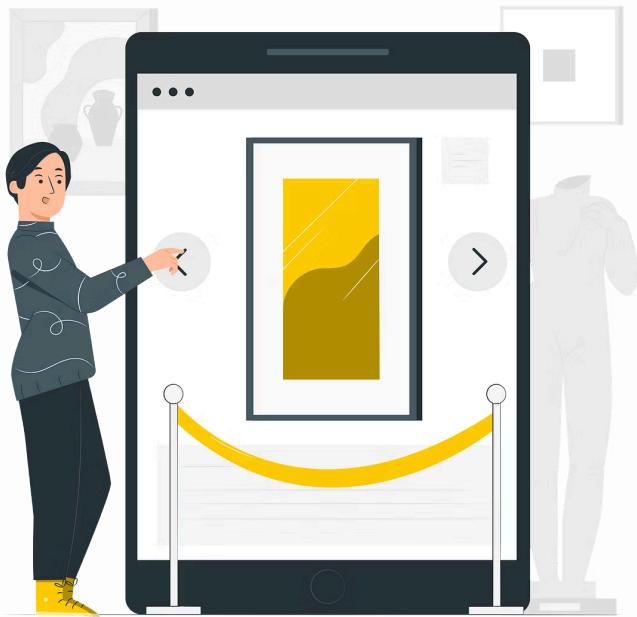
Ejemplo de uso:

```
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;
```

Componente Gallery



Funcionalidad

El componente Gallery muestra una lista de imágenes de forma dinámica. Utiliza un array de URLs de imágenes para generar los elementos visuales.

Implementación

Se utiliza el método map para iterar sobre el array de imágenes y crear elementos img para cada una. Los estilos en línea se aplican para crear una disposición flexible y centrada.

Gallery.jsx

Ejemplo de uso:

```
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: "50px" }}>
      {images.map((image) => (
        <img alt={image} key={image} style={{ width: "100px", height: "100px" }} />
      ))}
    </section>
  );
}

export default Gallery;
```

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

export default Gallery;
```

Componente Footer

Diseño Simple

El Footer utiliza estilos en línea para un diseño sencillo y efectivo.

Contenido Estático

Incluye un mensaje de copyright y el año actual.

Posicionamiento

Se añade un margen superior para separarlo del contenido principal.

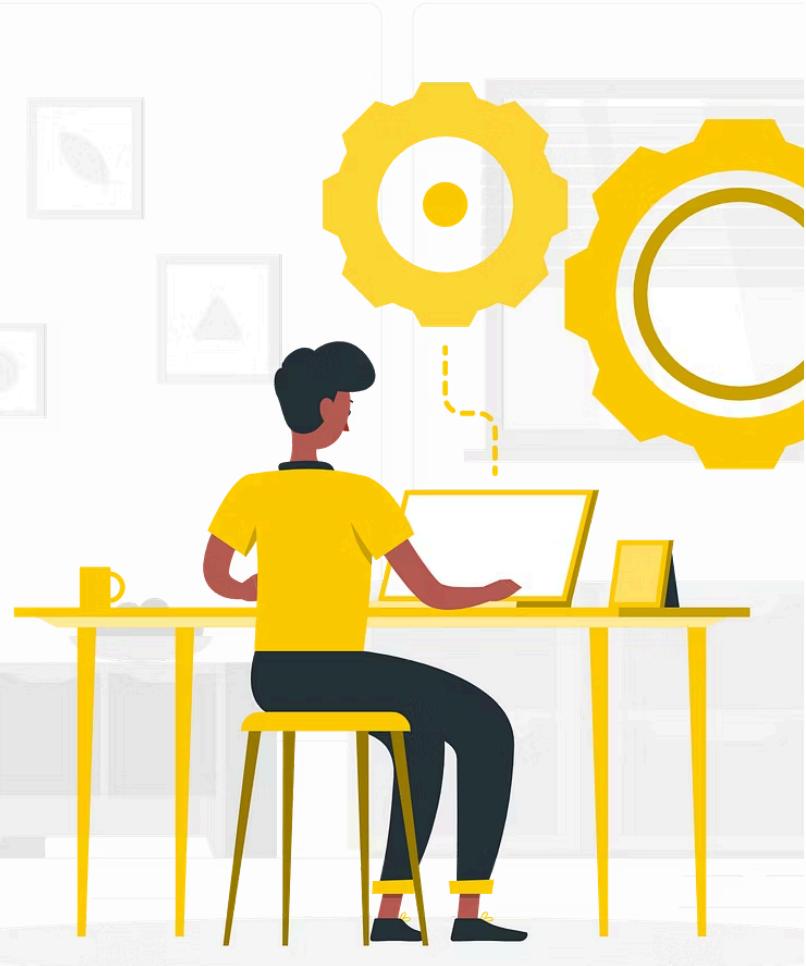
Footer.jsx

Ejemplo de uso:

```
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;
```



Ensamblaje en App.jsx

- 1 Importar Componentes**
Incluir todos los componentes creados
- 2 Estructurar JSX**
Organizar componentes en el return
- 3 Exportar App**
Hacer disponible el componente principal

App.jsx

Ejemplo de uso:

```
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

Visualización en el Navegador



Iniciar Servidor

Ejecuta `npm run dev` en la terminal para iniciar el servidor de desarrollo.

Acceder a la Aplicación

Abre <http://localhost:5173> en tu navegador para ver la aplicación en funcionamiento.

Verificar Componentes

Asegúrate de que todos los componentes se rendericen correctamente y estén ensamblados como se esperaba.

Reflexión final

Modularidad

La creación de componentes reutilizables permite una mejor organización del código y facilita el mantenimiento de la aplicación.

Escalabilidad

El enfoque basado en componentes hace que sea más sencillo expandir y modificar la aplicación en el futuro.





Preguntas para Reflexionar

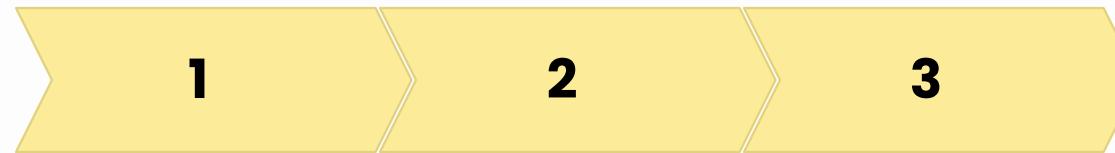
- 1** ¿Por qué es útil dividir una aplicación en componentes en React?

- 2** ¿Qué papel juegan las props en la personalización de los componentes?

- 3** ¿Cómo ayuda Vite en el desarrollo de aplicaciones React?



Próximos Pasos en React



Estado Local

Aprender sobre la gestión del estado local con useState

Manejo de Eventos

Profundizar en el manejo de eventos como clics y formularios

Ejercicio Práctico

Realizar el primer ejercicio de agregar productos al carrito

Recursos Adicionales

1

[Documentación oficial de React - Componentes](#)

2

[Vite - Getting Started](#)

3

[Placeholder.com](#)

4

[CSS Tricks - A Complete Guide to Flexbox](#)

5

[CSS Tricks - A Complete Guide to Grid](#)



Ejercicios Prácticos

Segunda fase del proceso para sumarte a TalentoLab



Silvia

Product Owner

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

Obligatorio

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.



Ejercicio Práctico

Obligatorio

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' },  
];
```

Ejercicio Práctico

Obligatorio

Crea un componente TarjetaProyecto:



Este componente debe recibir las siguientes props:

- **Título:** Nombre del proyecto.
- **Descripcion:** Detalles del proyecto.
- **BotónTexto:** 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"  
/>
```

Ejercicio Práctico

Obligatorio

Interactividad:



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

Ejercicio Práctico

Obligatorio

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'];
```



Ejercicio Práctico

Obligatorio

¿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.

Ejercicio Práctico

Obligatorio



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! 

