

# Nivel 2 React

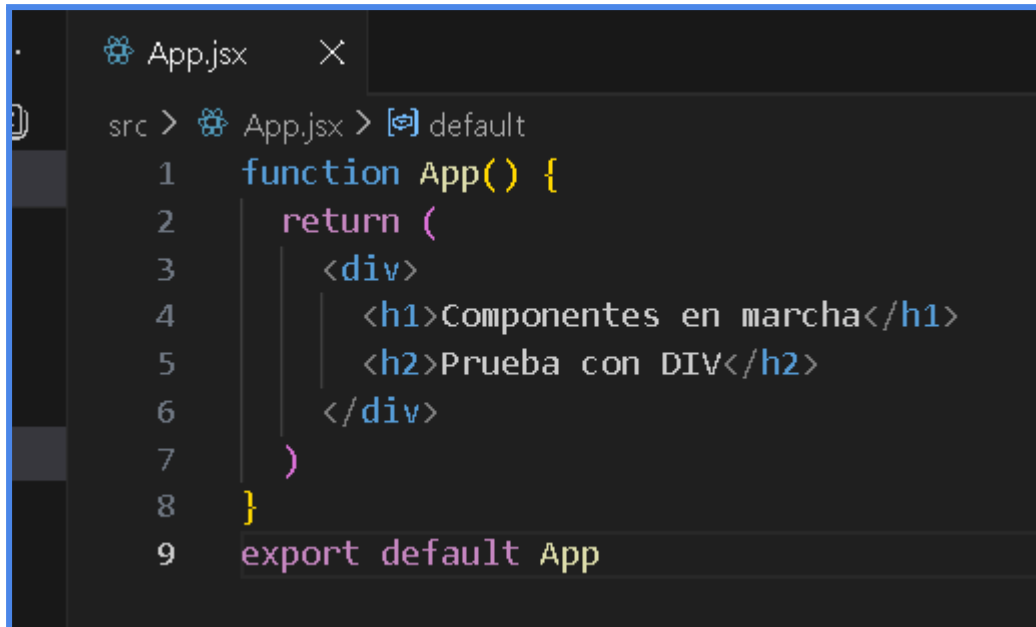
DAW2-DWEC

## Índice

1. Ejecuta el proyecto.....	1
2. Devuelve múltiples elementos con Fragment.....	1
3. Renderizado: ¿dónde se “engancha” React?.....	3
4. Comentarios en React.....	3
5. Composición: varios componentes dentro de uno.....	3
Preguntas extra para el PDF.....	5

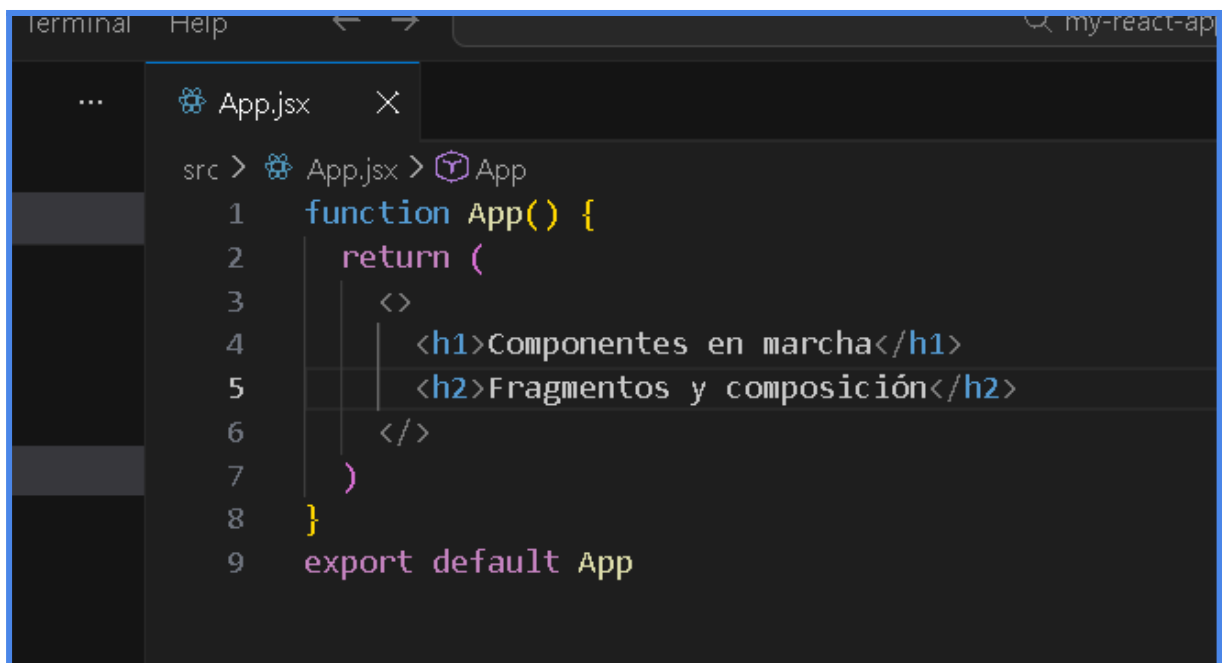
1. Ejecuta el proyecto

2. Devuelve múltiples elementos con Fragment



```
App.jsx X
src > App.jsx > [default]
1  function App() {
2      return (
3          <div>
4              <h1>Componentes en marcha</h1>
5              <h2>Prueba con DIV</h2>
6          </div>
7      )
8  }
9  export default App
```

“Figura 01, div que envuelve todo, elaboración propia”



```
Terminal Help my-react-ap
App.jsx X
src > App.jsx > App
1  function App() {
2      return (
3          <>
4              <h1>Componentes en marcha</h1>
5              <h2>Fragmentos y composición</h2>
6          </>
7      )
8  }
9  export default App
```

“Figura 02, fragmento vacío, elaboración propia”

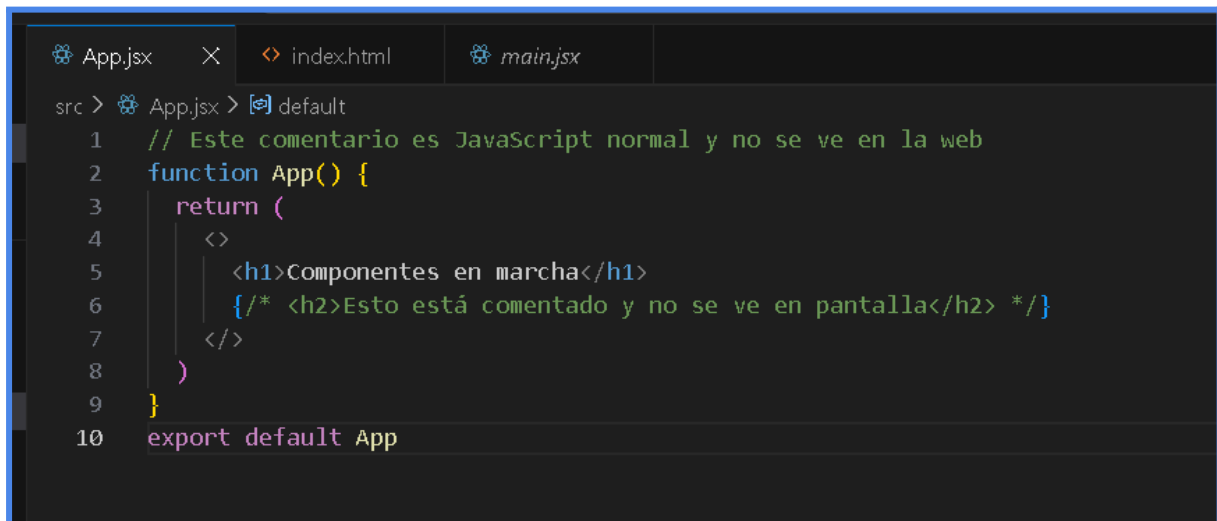


“Figura 03, visual en navegador, elaboración propia”

### 3. Renderizado: ¿dónde se “engancha” React?

- `document.getElementById('root')` lo que hace es buscar en el archivo HTML (`index.html`) el contenedor vacío con el identificador "root" porque actúa como el punto de montaje donde React mete todo el código JavaScript que se convierte en interfaz visual.

## 4. Comentarios en React



```
App.jsx | index.html | main.jsx
src > App.jsx > default
1 // Este comentario es JavaScript normal y no se ve en la web
2 function App() {
3   return (
4     <>
5       <h1>Componentes en marcha</h1>
6       {/* <h2>Esto está comentado y no se ve en pantalla</h2> */}
7     </>
8   )
9 }
10 export default App
```

“Figura 04, comentarios, elaboración propia”



“Figura 05, visual en navegador, elaboración propia”

## 5. Composición: varios componentes dentro de uno



```
src > App.jsx > ParentComponent
1  function UserComponent() {
2    |   return <h2>Soy el componente de Usuario</h2>
3    | }
4
5  function ProfileComponent() {
6    |   return <h2>Soy el componente de Perfil</h2>
7    | }
8
9  function FeedComponent() {
10   |   return <h2>Soy el muro de noticias (Feed)</h2>
11   | }
12
13  function ParentComponent() {
14    |   return (
15    |     <>
16    |       <h3>--- Inicio del Panel ---</h3>
17    |       <UserComponent />
18    |       <ProfileComponent />
19    |       <FeedComponent />
20    |       <h3>--- Fin del Panel ---</h3>
21    |     </>
22    |   )
23  }
24
25  export default function App() {
26    |   return (
27    |     <>
28    |       <h1>Bienvenido a la Tarea 2</h1>
29    |       <ParentComponent />
30    |     </>
31    |   )
32  }
```

“Figura 06, componentes, elaboración propia”

# **Bienvenido a la Tarea 2**

**--- Inicio del Panel ---**

**Soy el componente de Usuario**

**Soy el componente de Perfil**

**Soy el muro de noticias (Feed)**

**--- Fin del Panel ---**

“Figura 07, visual del navegador, elaboración propia”

## **Preguntas extra para el PDF**

### **1. ¿Qué es un componente en React?**

- Es una unidad independiente y reutilizable de la interfaz de usuario. Técnicamente, es una función de JavaScript que devuelve elementos JSX (lo que se ve en pantalla) y puede contener su propia lógica y estructura.

### **2. ¿Por qué usamos Fragmentos (<>...</>) en lugar de un <div>?**

- Usamos Fragmentos para agrupar varios elementos hijos sin añadir un nodo extra al DOM (el HTML real del navegador). Usar demasiados <div> innecesarios puede hacer que la estructura HTML se vea mal y con ello romper estilos CSS (como Flexbox o Grid) que dependen de la relación entre “padre” e “hijo”.

### **3. ¿Qué papel tienen index.html y main.jsx en el renderizado?**

- index.html: Proporciona la estructura base del navegador y contiene el contenedor vacío (`<div id="root">`) donde se encuentra la aplicación.
- main.jsx: Es el punto de entrada de JavaScript. Importa la librería React y utiliza `createRoot` para tomar el control del div root del HTML y renderizar dentro de él el componente principal (`<App />`).

#### **4. Explica con tus palabras qué significa “componer componentes”.**

- Básicamente es ir montando la web juntando piezas pequeñas. En vez de hacerlo todo de golpe, se crean componentes sencillos, como botones o menús, y se van metiendo dentro de otros más grandes.