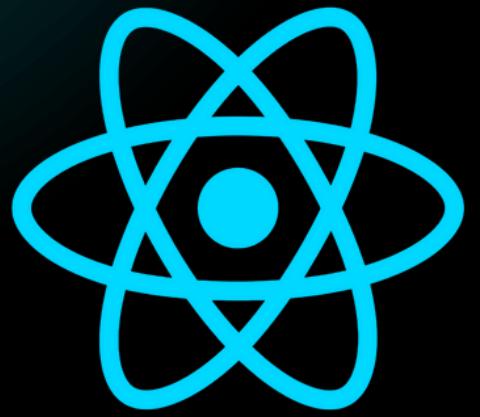


**REACT**



# LA BIBLIOTECA PARA INTERFACES DE USUARIO EN JAVASCRIPT

Presentado por

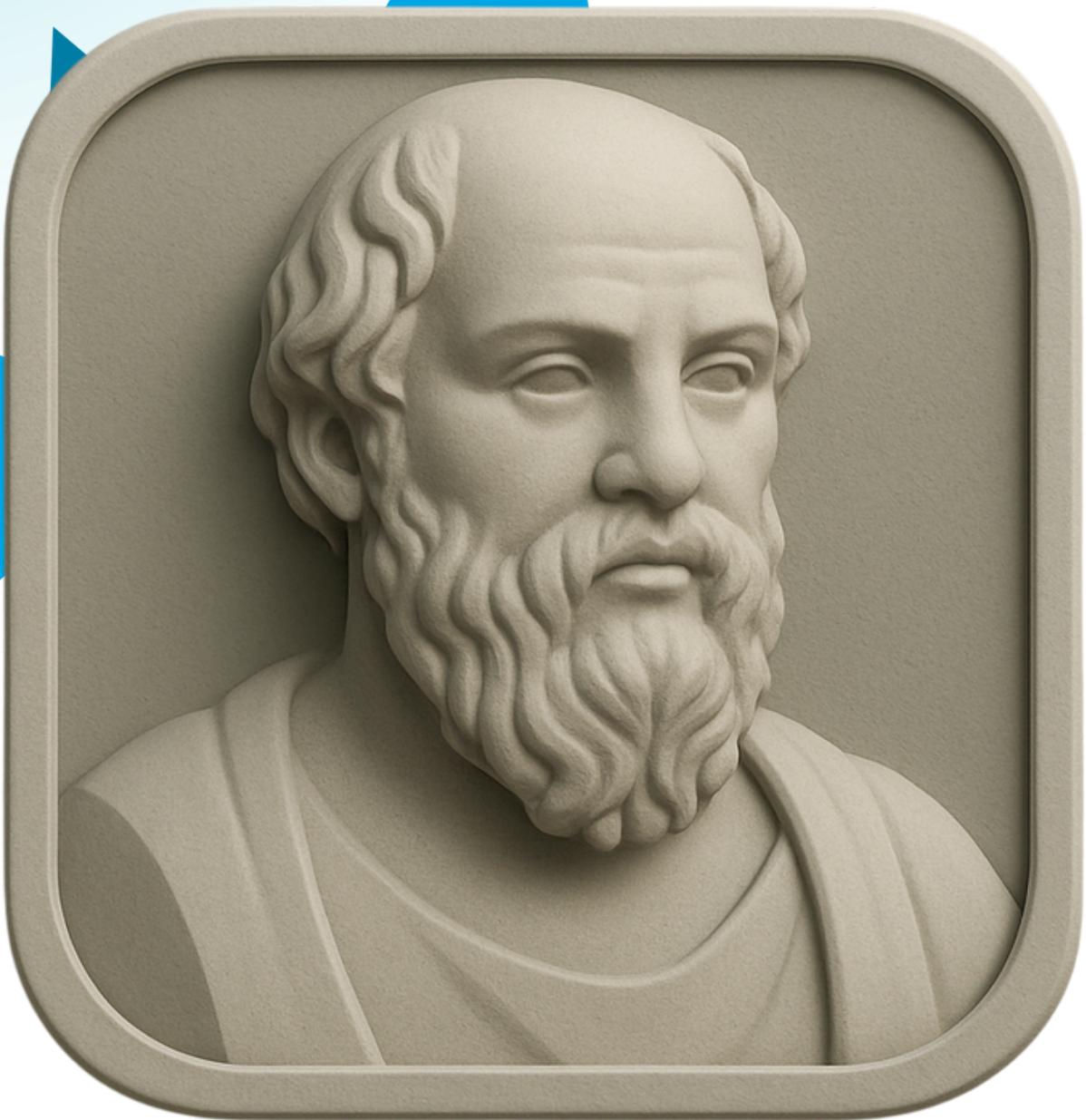
Keyla Dayana Arboleda Mina

Carlos Andres Cifuentes Montaño

Darío Restrepo Landázury

Jose Fernando Sinisterra Ibargüen

# FILOSOFÍA



Se caracteriza por la composición de componentes, los cuales son escritos por distintas personas deben trabajar bien en conjunto.



# ORIGEN



React fue creado en 2011 por **Jordan Walker** en **Facebook**, inspirado por **XHP, una librería de PHP**. Surgió para resolver la creciente complejidad de las interfaces web dinámicas, que se volvían difíciles de mantener y propensas a errores. Su solución **se basó en componentes reutilizables, JSX para escribir HTML dentro de JavaScript y un Virtual DOM**, mejorando **velocidad** y **organización** del código.

# EVOLUCIÓN

## 2013: Nacimiento

- Lanzamiento inicial por Facebook.
- Introducción del concepto de Virtual DOM.

## 2015: Expansión

- Lanzamiento de React Native para desarrollo móvil.
- Adopción masiva en la comunidad frontend.

# EVOLUCIÓN

## 2017: Mejoras de rendimiento

- React Fiber (reescritura del núcleo).
- Renderizado asíncrono y priorización de tareas.

## 2019: Revolución con Hooks

- useState, useEffect y otros hooks.
- Permitió usar estado en componentes funcionales.

# EVOLUCIÓN

## 2022: React 18

- Renderizado concurrente.
- Mejoras en SSR (Server-Side Rendering).
- Nuevos hooks como. useTransition.

## 2023-2024: Server Components

- Ejecución de componentes en el servidor.
- Mejor rendimiento y SEO.

# VENTAJAS



## Rendimiento Optimizado

- Virtual DOM para actualizaciones eficientes.
- Reconciliación inteligente de cambios.

## Gran Ecosistema

- Herramientas como Redux, Next.js, React Router
- Amplia comunidad de desarrollo.

## Componentes Reutilizables

- Arquitectura modular.
- Fácil mantenimiento y escalabilidad.

# VENTAJAS

## Flexibilidad

- Usable en proyectos pequeños y grandes.
- Funciona con otras librerías fácilmente.

## Multiplataforma

- React para web.
- React Native para móvil.
- React 360 para realidad virtual.

## Fácil Aprendizaje (si ya se sabe JS y ES6)

- Curva de aprendizaje más suave que otros frameworks.
- Buena documentación y recursos.



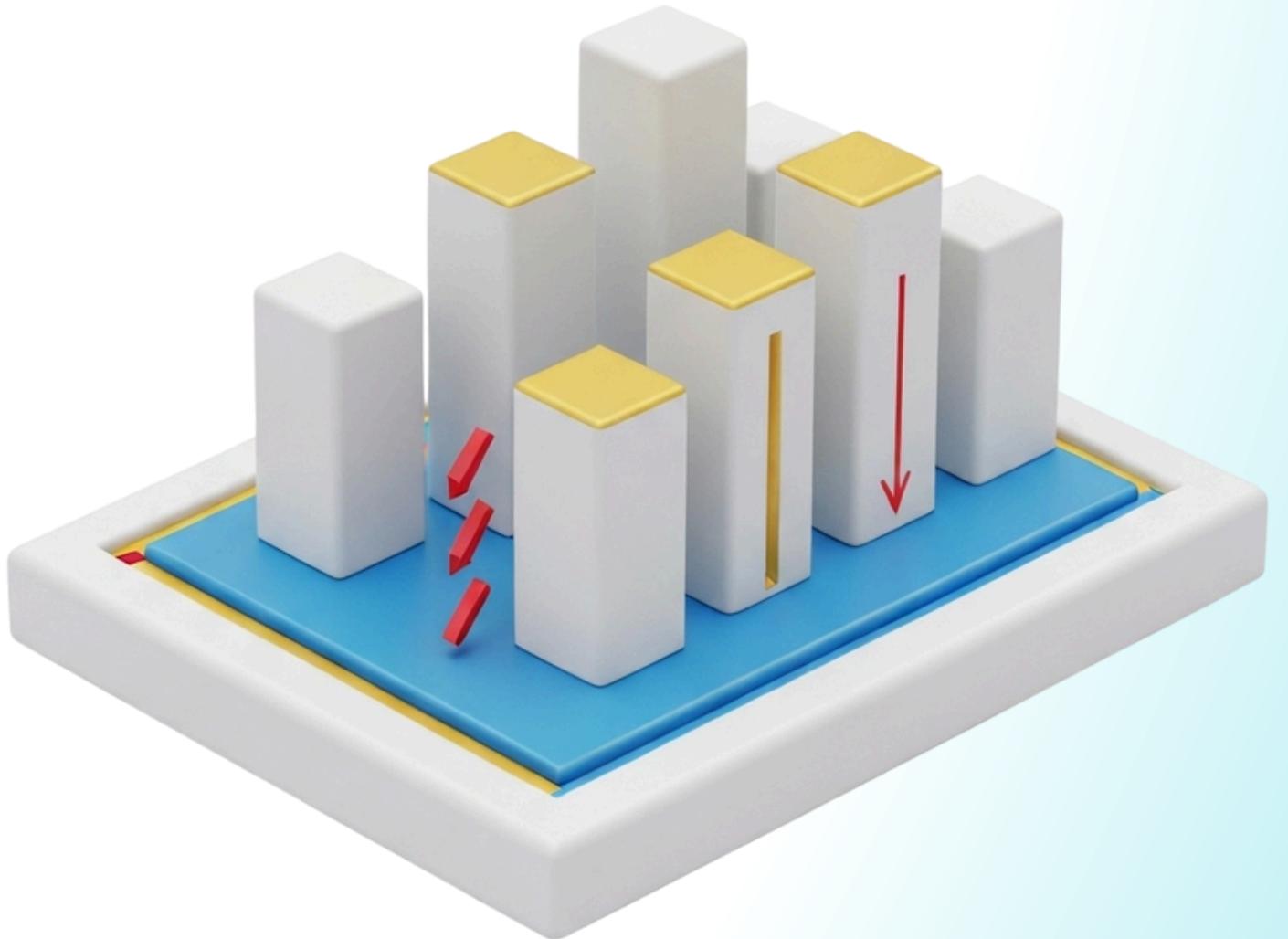
# DESVENTAJAS



- **Curva de aprendizaje:** entender JSX, estados, hooks y ciclos de vida puede ser complejo para principiantes.
- **Ecosistema variable:** cambia rápido, exige aprendizaje constante; prácticas y herramientas que se actualizan con frecuencia.
- **Adaptación a JSX:** mezclar JavaScript y HTML puede ser confuso al inicio si se está acostumbrado a tener el HTML y JS separados.

# DESVENTAJAS

- **SEO sin SSR:** en **SPA** (*Single Page Applications*), los buscadores pueden indexar mal; **Next.js** u otras herramientas que pueden mitigarlo.
- **No apto para grandes proyectos:** al ser solo **UI**, requiere integrar **Redux, Router** u otras herramientas, aumentando su complejidad.

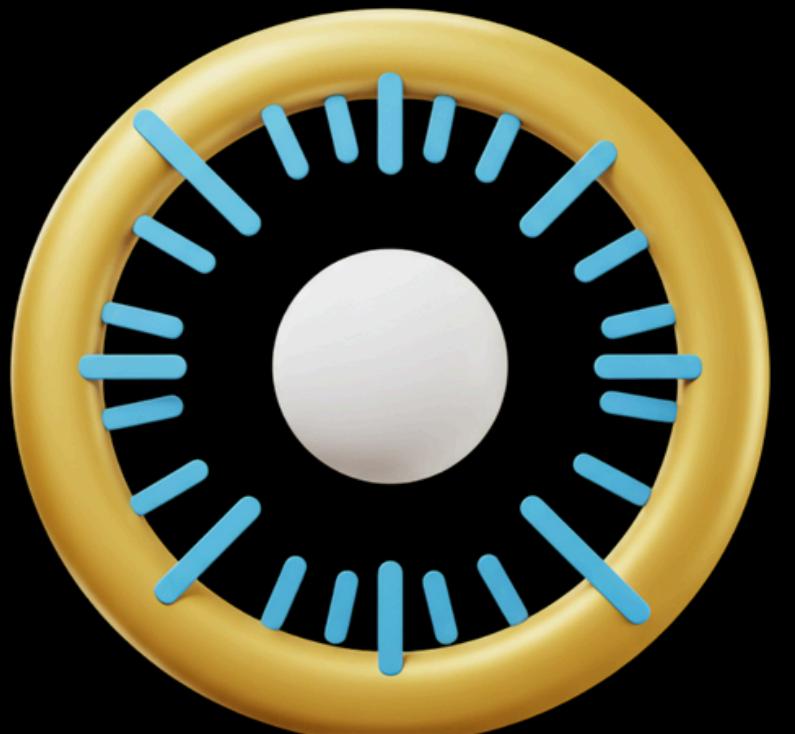


# ENFOQUE

Se basa en que **la UI es una función del estado**. En lugar de indicar pasos específicos para actualizar la pantalla, **solo describes cómo debe verse según el estado actual de los datos**, y React se encarga de actualizarla eficientemente. Hay 4 pilares que sostienen este enfoque:

Programación  
declarativa

Componentes  
reutilizables



Flujo de datos  
unidireccional

Actualización  
con Virtual DOM.

# COMPARACIÓN CON OTROS FRAMEWORKS

	REACT	ANGULAR	VUE
TIPO	Es una biblioteca centrada en las vistas ( <b>Componentes</b> ).	Es un framework completo.	Es un framework progresivo.
CURVA DE APRENDIZAJE	Media	Alta	Baja-media
VENTAJAS PRINCIPALES	Gran ecosistema; flexibilidad para elegir librerías; fácil integración incremental.	Solución integral; convenciones sólidas; ideal para equipos grandes y apps empresariales.	Sintaxis intuitiva; fácil de adoptar; buen equilibrio entre simplicidad y capacidad.

# COMPARACIÓN CON OTROS FRAMEWORKS

	<b>REACT</b>	<b>ANGULAR</b>	<b>VUE</b>
<b>LIMITACIONES PRINCIPALES</b>	No es “todo incluido”: hay que ensamblar.	Verboso; curva pronunciada; mayor peso inicial.	Ecosistema algo más pequeño que en ciertos nichos empresariales.
<b>CASOS DE USO RECOMENDADOS</b>	SPAs, apps que requieren migración incremental, proyectos que necesitan librerías específicas.	Aplicaciones empresariales a gran escala, proyectos con normas estrictas y uso intensivo de TypeScript.	SPAs y proyectos rápidos de prototipado o equipos pequeños/medios que valoran la productividad.
<b>TOOLING/META-FRAMEWORK</b>	Vite, Create React App; Next.js (SSR/SSG/ISR)	Angular CLI, Angular Universal (SSR)	Vite, Vue CLI; Nuxt (SSR/SSG)

The screenshot shows a development environment with two main windows: a terminal window on the left and a browser window on the right.

**Terminal Window:**

- Shows the command `vite` being run.
- Output message: **VITE v7.1.2 ready in 279 ms**
- Network information:
  - Local: <http://localhost:5173/>
  - Network: use `--host` to expose
  - press `h + enter` to show help

**Browser Window:**

- Title bar: Prueba de React
- Address bar: localhost:5173
- Content area: Displays the text "Buenas noches, les hablo desde React" in a large, bold font.

**VS Code Editor (Left Side):**

- File tabs: vite.config.js, main.jsx (selected), index.html
- Code content:

```
src > main.jsx > ...
1 import {createRoot} from 'react-dom/client';
2
3 const root = createRoot(document.getElementById('app'));
4
5 root.render(<h1>Buenas noches, les hablo desde React</h1>);
```

```
const despedida =  
ReactDOM.createRoot(document.getElementById('despedida'));  
root.render(<h1>Gracias por su atención!</h1>);
```

VER  
REPOSITORIO