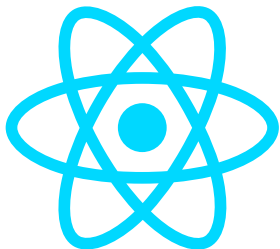


REACT VS VUE



VS



2025

Jonathan Pérez

Si tecleas "React vs Vue" en cualquier foro, te llueven opiniones. Fanáticos declarando una guerra santa, gurús predicando con gráficas de rendimiento y puristas discutiendo sobre la pureza del código. Es agotador. Y lo peor: la mayoría no ha tomado la decisión en un proyecto real en los últimos dos años.

La verdad incómoda es que en 2025, **ninguno de los dos perdió**. La "batalla" no es por el cetro, sino por el tipo de desarrollador y el proyecto que tienes entre manos.

Después de años de usarlos, de apagar fuegos a las 2 de la mañana y de celebrar despliegues exitosos, te digo:

la elección no es religión, es pragmatismo.

Olvidemos el marketing. Hablemos de sensaciones en los dedos, de la fatiga mental a las 4 PM y de cómo duermes por la noche sabiendo que tienes que escalar ese feature.

React: El Caos Creativo (y Poderoso)

Imagina un taller de un artista genial pero desordenado. Las herramientas están por todas partes, tienes la libertad absoluta de montar tu propio torno de alfarero o traer una sierra de otro planeta. Pero si no eres ordenado, acabas enterrado en serrín y con proyectos a medio terminar.

A Favor (Pros):

- **El Ecosistema es un Continente:** Lo que necesites, existe. Una librería para animaciones micro-interactivas? Tres opciones. Un hook para gestionar el estado de tu cafetera inteligente? Probablemente.
-

Esta es su mayor ventaja. No estás eligiendo un framework, estás eligiendo una economía entera de soluciones.

- **JSX: Esa Relación Amor/Odio:** Escribir HTML dentro de tu JavaScript parece una herejía hasta que te acostumbras. Entonces, algo hace click. La lógica y la UI viven juntas, en el mismo componente. Es increíblemente intuitivo para construir interfaces dinámicas y complejas. Te da una expresividad brutal.
 - **Demanda Laboral y Transferibilidad:** React es un estándar de facto. Aprenderlo profundamente te abre puertas no solo en React, sino en el ecosistema JavaScript moderno
-

El mental model que aprendes (componentes, estado, hooks) es tremendamente valioso y transferible.

En Contra (Contras):

- **La Fatiga de Decisión:** React no te dice cómo hacer las cosas. ¿Estado global? Redux, Zustand, Context API, Jotai? ¿Ruteo? React Router, Next.js. ¿Estilos? CSS-in-JS, módulos CSS, Tailwind... Tú eres el arquitecto jefe. Es poderoso, pero agotador. Puedes pasar dos días decidiendo la pila tecnológica para un proyecto nuevo.
 - **La Maldición de la Flexibilidad:** Un proyecto React de 6 meses puede parecer irreconocible al lado de uno de 2 años.
-

Sin convenciones estrictas, cada equipo inventa su propia arquitectura. Esto puede hacer que el onboarding de un nuevo desarrollador sea una pesadilla si el código no es excepcionalmente claro.

- **La Complejidad que Crece Orgánicamente:** Los hooks son maravillosos, pero es fácil terminar con un `useEffect` anidado en otro `useEffect` que depende de un estado que se actualiza en otro lado. Gestionar efectos secundarios de forma elegante requiere oficio.
-

¿Cuándo Elegir React?

- **Proyectos a gran escala y muy dinámicos:** Aplicaciones web complejas como dashboards administrativos, herramientas de productividad empresarial (como Asana o Trello) o grandes marketplaces. Donde la lógica de negocio es intrincada.
 - **Equipos experimentados y grandes:** Donde tienes arquitectos senior que pueden establecer y hacer cumplir buenas convenciones. Donde la fuerza del ecosistema es una necesidad.
 - **Cuando priorizas la empleabilidad o el crecimiento a largo plazo:** Es una apuesta segura para tu CV.
-

Vue: La Elegancia Pragmática (y Productiva)

Ahora imagina un taller de diseño suizo. Todas las herramientas tienen su lugar, brillantes y organizadas. El manual de instrucciones es claro y conciso. Puedes empezar a crear algo valioso en la primera hora, sin perder tiempo montando el taller.

A Favor (Pros):

- **La Curva de Aprendizaje es una Cuesta Suave:** Vue es, sin duda, el más fácil de aprender y enseñar. Su documentación es ejemplar. La separación entre `<template>`, `<script>` y `<style>` es familiar y reconfortante para desarrolladores que vienen
-

de HTML/CSS/JS clásico. La productividad desde el minuto uno es altísima.

- **La Complejidad que Crece**
Orgánicamente: Los hooks son maravillosos, pero es fácil terminar con un `useEffect` anidado en otro `useEffect` que depende de un estado que se actualiza en otro lado. Gestionar efectos secundarios de forma elegante requiere oficio.
 - **La Reactividad es Mágica (y Sencilla):**
Cambias un valor en el estado y la UI se actualiza. Punto. No necesitas pensar en `setState` o hooks de dependencias al principio. Es intuitivo y poderoso, lo que reduce drásticamente los bugs relacionados con el estado en proyectos pequeños/medianos.
-

En Contra (Contras):

- **El Ecosistema, Aunque Sólido, es Más Pequeño:** Para necesidades ultra-específicas o de nicho, a veces no hay una librería Vue-first tan pulida como las que hay en React. Aunque normalmente siempre hay una alternativa o puedes usar librerías vanilla, no tienes la abrumadora cantidad de opciones.
 - **La Flexibilidad "Controlada":** Vue es menos opinado que Angular, pero más que React. Si quieres salirte de su manera de hacer las cosas, a veces puedes sentir que estás luchando contra el framework, en lugar de que el framework te ayude.
-

-
- **La Percepción en el Mercado (aunque cambia):** Aún existe en algunas empresas la idea de que Vue es para "proyectos pequeños". Esto es falso (see: GitLab, Nintendo, Louis Vuitton), pero es un sesgo que persiste en algunas mentes de managers.
-

¿Cuándo Elegir Vue?

- **Startups y Proyectos que Necesitan Velocidad:** Cuando necesitas un MVP estable y funcional para ayer. Vue te permite moverte rápido sin sacrificar estructura.
 - **Equipos Pequeños o Multidisciplinares:** Es perfecto si hay desarrolladores back-end o juniors que necesitan ponerse al día rápidamente. La coherencia del código está casi garantizada.
 - **Proyectos de Complejidad Media:** Portales de contenido, aplicaciones SaaS elegantes, sitios de e-commerce interactivos. Donde la experiencia de desarrollo fluida es tan importante como el resultado final.
-

El Veredicto: No Elijas el "Mejor", Elige el "Mejor Para"

- **¿Tienes un equipo senior, un proyecto enorme y complejo, y necesitas el poder bruto del ecosistema? → React.**
- **¿Necesitas ir rápido, tener un código coherente desde el día uno, y tu equipo valora la simplicidad y la elegancia? → Vue.**

Al final, ambos te llevarán al mismo destino: una aplicación moderna y funcional. React es el 4x4 todoterreno que puedes tunear hasta el infinito. Vue es el sedan de lujo que te lleva con comodidad, seguridad y sin esfuerzo.

La verdadera batalla no está entre librerías,
sino en tu capacidad para dominar la que
elijas.

Así que elige, y construye algo increíble.

Eso es lo que importa al final.
