

[Open in app](#)**Medium** Search

★ Get unlimited access to the best of Medium for less than \$1/week. [Become a member](#)



Top 9 React JS Frameworks Developers Are Using in 2025

Top 9 React JS Frameworks Developers Are Using in 2025

7 min read · Apr 15, 2025



Brilworks Software

Follow



Listen



Share

... More

One of the most widely used options for creating contemporary web and mobile interfaces is React. However, developers frequently resort to specialised tools that augment React's capabilities as projects become more complex and user expectations rise. React frameworks can help with that.

The React ecosystem has greatly matured by 2025. There is probably a React framework designed especially for that use case, whether you're creating a highly

customised user interface, a native mobile application, or a server-rendered website. Developers now have more options than ever before thanks to robust full-stack solutions like Next.js and RedwoodJS, mobile-focused tools like React Native and Expo, and even UI component libraries like Material-UI and React Bootstrap.

But how can you choose the best framework for your project when there are so many options?

This guide examines the most popular React frameworks for 2025, their ideal uses, and how to choose the one that best suits your development requirements. This breakdown will help you make sense of the situation whether you're a tech lead, frontend developer, or startup founder organising your next major release.

What is React?

A JavaScript library called React is used to create user interfaces, particularly dynamic ones that change rapidly in response to user input. React, developed by Facebook (now Meta), revolutionised UI development by dissecting interfaces into manageable, reusable parts.

The concept of a virtual DOM, or an in-memory representation of the actual DOM, is central to React. React updates just the necessary portions of the page whenever something changes, rather than the entire page. Because of this, React apps are quick and effective even as they get more complicated.

Since React only concentrates on the view layer, it is frequently referred to as the “V” in MVC (Model-View-Controller). It excludes server-side rendering, data management, and routing tools on its own. The surrounding ecosystem has thrived as a result. To fill in the gaps, developers have created React frameworks over time, which provide additional structure, pre-built features, and tools to help transform React into a comprehensive development solution.

Want to dive deeper into how React powers the front end? You might like our full guide on [React for Front-End Development](#).

Is React JS a Framework?

New developers frequently ask, “Is React JS a framework?” Not exactly, is the short answer.

Technically speaking, React is a JavaScript library for creating user interfaces, with a focus on controlling the view layer of applications. It focusses on effectively rendering user interface elements and updating them in response to data changes. Nevertheless, because of the way it works in actual projects, a lot of developers call it a React JS framework.

Why is there so much confusion? Because React acts like a complete framework when it is used with tools like Next.js, Gatsby, or RedwoodJS. These tools add features you would normally expect from a traditional framework, such as server-side rendering, data fetching, routing, and backend integration, to React's capabilities.

Therefore, even though React is a library, developers can create full-scale apps with the help of the larger React framework ecosystem. Because of this, despite the technical difference, people frequently use the terms interchangeably.

Top React Frameworks in 2025

1. Next.JS

Let's begin with Next.js, which is at the top of the list for a reason. For developers looking to create production-ready web apps with built-in speed, scalability, and performance, this React JS framework has emerged as the go-to option. Next.js takes care of a lot of things that you would otherwise have to set up by hand, whether you're creating a large eCommerce site or a personal blog.

It's more than just a front-end tool; it provides you with server-side rendering, routing, API endpoints, and even SEO optimisation in a convenient package. For this reason, it's frequently the first suggestion for React-built websites.

Key Features:

- 1. Hybrid rendering (SSR + SSG):** You can choose between server-side rendering and static generation on a per-page basis. This gives you control over performance and SEO, depending on the page's needs.
- 2. File-based routing system:** Simply create a file inside the `pages/` directory, and it becomes a route. No need to configure routing manually — just name your files, and you're good to go.
- 3. Image and font optimization:** Next.js automatically optimizes images and fonts for faster loading times, which helps with both user experience and Core Web

Vitals (aka Google's ranking factors).

4. b You can create serverless backend functions right inside your project. This makes it easy to handle things like form submissions, user auth, or API calls without setting up a separate backend.
5. Automatic code splitting: Next.js only loads the JavaScript needed for the page being viewed, which significantly reduces load times and improves performance.

Best for:

Web applications that require a combination of static and dynamic content, quick load times, and SEO optimisation. Blogs, SaaS platforms, eCommerce websites, and pretty much anything else you would build for the web are all perfect for it.

2. React Native

This might be the most popular among the React frameworks. React Native is at the top when talking about cross-platform development. A significant component of the React ecosystem, React Native is particularly useful for creating mobile applications. By bringing React's power to iOS and Android, developers can create apps with JavaScript and React components and still provide a native-like user experience. It significantly saves time and effort when you can build one app for both Android and iOS.

You can build once and deploy everywhere, eliminating the need to maintain two distinct codebases for two platforms. It's among the most effective methods for developing cross-platform mobile applications without sacrificing functionality or aesthetics.

Also read: [React Native for Beginners](#)

Key features:

1. **Cross-platform development:** Use one codebase for both Android and iOS, helping teams save time and resources while keeping updates consistent across platforms.
2. **Truly native UI:** React Native bridges to native components, so your app doesn't feel like a web page wrapped in a mobile shell — it feels like the real thing.
3. **Fast Refresh:** Developers can instantly see their code changes reflected in the app, speeding up development and testing without restarting the whole app.

4. **Easy integration with native code:** Need to access device features like camera, Bluetooth, or location? React Native makes it straightforward to plug into native modules when needed.
5. **Strong community support:** With a massive developer base and mature ecosystem, there are plenty of libraries, tools, and guides to lean on as your app grows.

Best for

Constructing mobile applications that must function well on both Android and iOS. Perfect for internal tools, startups, MVPs, and even consumer apps of production quality.

3. Gatsby

The focus of the React-based Gatsby framework is speed, specifically lightning-fast static websites and applications. It is especially well-liked by developers creating documentation portals, blogs, and marketing websites where SEO and performance are crucial.

Gatsby is special because it pre-renders everything at build time, so users will experience a remarkably smooth and lightning-fast website. It is a good option for projects with a lot of content because it is also well integrated with GraphQL and has a robust plugin ecosystem.

Key features:

1. **Static site generation (SSG):** Gatsby builds your site as static HTML, CSS, and JS at compile time. This leads to lightning-fast load times and great SEO out of the box.
2. **Rich plugin ecosystem:** From sourcing data via CMSs like WordPress or headless systems like Contentful, to adding features like image optimization or analytics — there's likely a plugin for it.
3. **Built-in GraphQL data layer:** Gatsby pulls in data from multiple sources and serves it through a unified GraphQL API. This gives you more flexibility in managing content and APIs.
4. **Automatic performance optimization:** Things like image compression, lazy loading, and code splitting are handled automatically, so you don't have to tweak everything manually.

5. PWA and offline support: Gatsby makes it easy to build a progressive web app that works offline and feels like a native experience.

Best for:

Static websites, marketing pages, personal blogs, and documentation sites where speed, SEO, and smooth performance are key priorities.

4. Expo

Expo is frequently referred to as the simplest way to get started with React Native, and it truly does live up to that claim. It is a platform and framework that builds upon React Native and provides developers with a collection of tools, services, and libraries to make the process of creating mobile apps even easier.

Expo can save a great deal of time if you're working under pressure or are new to mobile development. You can create, test, and launch apps with little setup, and you don't have to jump straight into native code.

Key features:

- 1. Out-of-the-box toolchain:** Expo comes with a full development environment right out of the box; you don't need to install Xcode or Android Studio to get started.
- 2. Expo Go for live testing:** Without requiring a build process, you can quickly preview your app on a physical device by scanning a QR code. ideal for quick iterations.
- 3. Managed and bare workflows:** Start simple with the managed workflow (no native code), and switch to the bare workflow later if you need full control.
- 4. Built-in APIs for common tasks:** You can use Expo's prebuilt APIs to access the camera, location, notifications, haptics, and more without having to create your own native modules.
- 5. Easy deployment with EAS:** Expo Application Services (EAS) let you build and submit apps to the App Store or Google Play without touching native code.

Best for:

Developers who want to build React Native apps quickly and painlessly, especially for MVPs, prototypes, or apps where speed and simplicity matter more than deep customization.

Read the full [article here](#).

Originally published at <https://www.brilworks.com>.

React

Reactjs

JavaScript



Follow

Written by Brilworks Software

372 followers · 8 following

<https://www.brilworks.com/>

No responses yet




Vaddeamarnath

What are your thoughts?

More from Brilworks Software

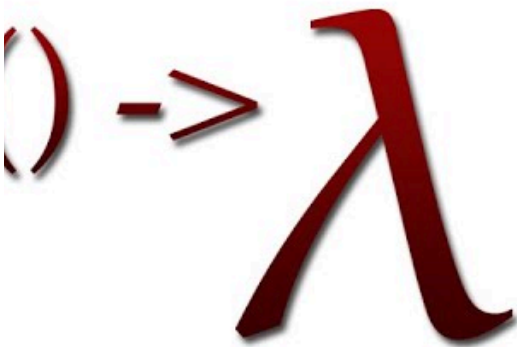



 Brilworks Software

Path to Become AWS Partner

Amazon Web Services (AWS): A Brief Overview

Oct 5, 2023  482  2



 Brilworks Software


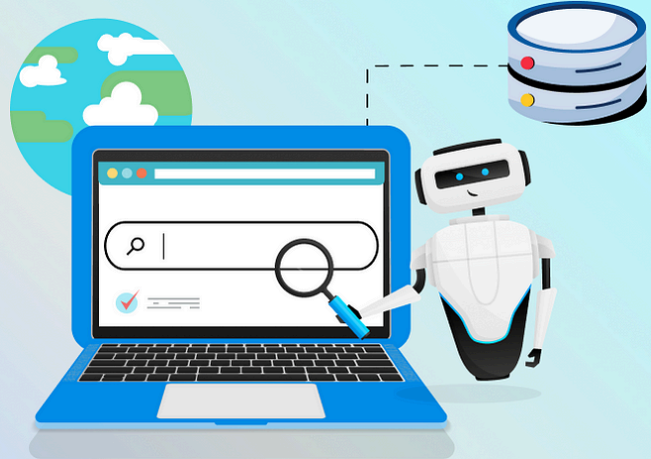
Java: Predicate & Function

Java 8 is not new anymore. Still, I see so many of us unaware of some very good features introduced after Java 8. We will cover some of the...

Oct 20, 2022 🖱️ 1.1K 💬 1



6 Best Practices For Using AI for Database Querying

 Brilworks Software


6 Best Practices For Using AI for Database Querying

We are currently in the AI age, and it is changing how we interact with computers and databases. We are progressing towards building...

Mar 3 🖱️ 160



What's Changed in Java Versions: New Features and Deprecations

 Brilworks Software

What's Changed in Java Versions: New Features and Deprecation

Java has been around for over two decades, powering robust applications across industries. It has consistently ranked as one of the most...

Jan 6 🖱 252



See all from Brilworks Software

Recommended from Medium



JS In JavaScript in Plain English by Imran Farooq

10 React Anti Patterns to Avoid as a React Developer

React is a powerful library, but even experienced developers can fall into common traps that lead to messy, inefficient, or...

★ May 10



REACT 19

New features and Updates



FE In Front-end World by Komal Raut

React 19 Memoization: No More useMemo & useCallback?

As React developers, we have used useMemo and useCallback for years to keep our applications efficient and avoid uncalled re-renders. It is...

★ Feb 25 🖱 534 💬 8



Optimization

92

Performance

96

Accessibility

100

Best Practices

100

SEO

92

Performance

Values are estimated and may vary. The [performance score is calculated](#) directly from these metrics. [See calculator.](#)



FS In Full Stack Forge by Daniel Scott

How I Optimized My React Application to Handle 3M Traffic

10 Must Know Tips for Every React Developer (So You Don't Have to Suffer Like I Did!) 😊

★ Mar 16 🖱 560 💬 13



TypeScript Bites

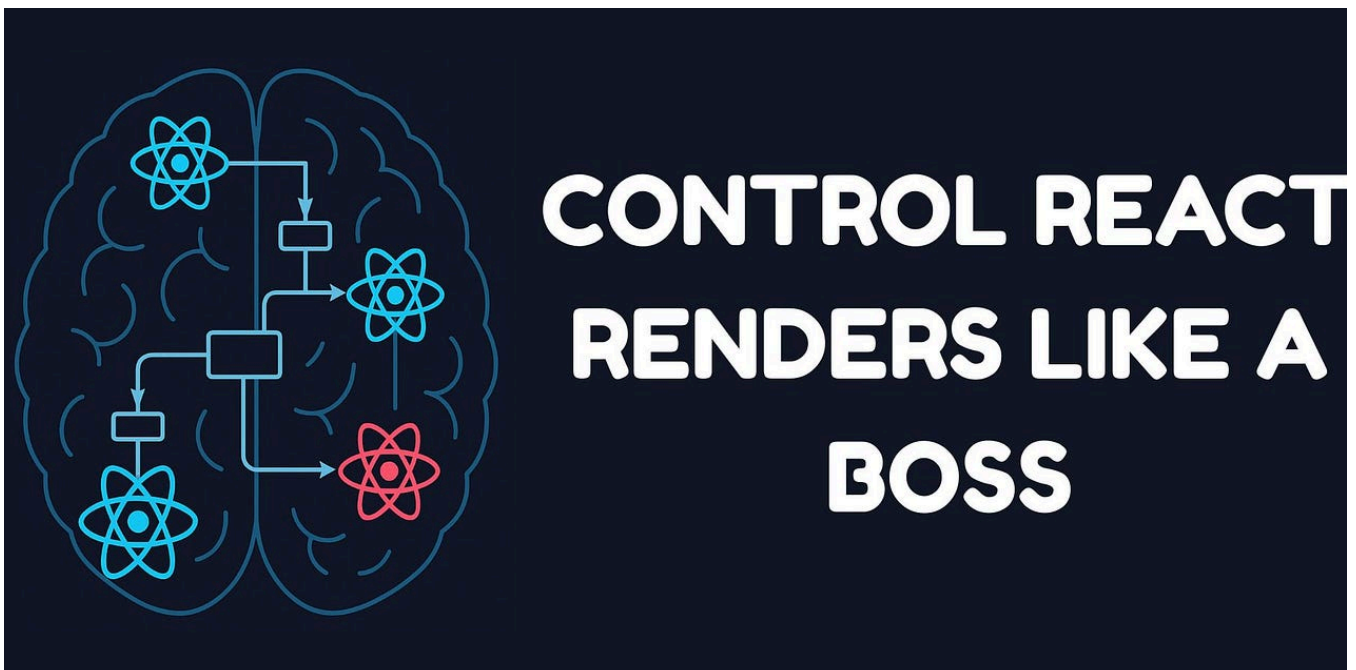
Type Narrowing & Type Guards – Master in, *typeof*,

 In Level Up Coding by Chamith Madusanka

TypeScript Bites #3: Type Narrowing & Type Guards—Use in, *typeof*, and *instanceof* Like a Pro

TypeScript uses type guards to figure out the exact type at runtime, so you can safely access type-specific properties and methods.

May 5 🖱 4

 In The Syntax Diaries by Amaresh Adak

Stop Chasing React Re-Renders—Here's How Top Engineers Control Them Like a Boss

Most devs waste time battling unnecessary re-renders. Real React architects never let the problem start in the first place—and here's...

★ May 6 🖱 71 💬 3



Fikret Sefa Çakır



React Roadmap 2025: The Power of useCallback in React

If you're building React apps that need performance boosts, especially when passing callbacks to child components, useCallback will be your...

Apr 26 🖱 11



See more recommendations