NEWS · SEPTEMBER 27, 2025

Better-Auth taking over management of Auth.js

By Dr. Holger Flick

I've tried a lot of authentication frameworks for React and Next.js in the last year. I wanted something that just works, is secure, and lets me move fast without wrestling with OAuth flows or rolling my own session handling.

For a long time I liked Auth.js (formerly NextAuth.js). It solved many problems out of the box and made it possible to own auth without spending months on integrations. But there were a couple of things that held me back when starting new projects: progress on the project sometimes felt slow to my eyes, and the getting-started documentation could be tricky to follow when you were just learning the library.

Then I learned about Better-Auth — and honestly, I never looked back. Better-Auth felt clearer, more opinionated where it needed to be, and focused on the primitives teams actually reuse. It removed the friction I had with Auth.js when building new apps.

The big news

Last week, there's been some news that validates that move and makes the ecosystem healthier: Auth.js is now maintained and overseen by the Better-Auth team.

What happened (short summary)

- Auth.js, the project previously known as NextAuth.js, has been handed over to the Better-Auth team for maintenance and stewardship.
- Better-Auth began as an effort to make authentication primitives easier to own and evolve. The two projects share similar goals: reduce friction around OAuth, sessions, and common auth patterns so developers can ship faster and more safely.
- The transition means existing Auth.js users can continue without disruption. Security patches and urgent fixes will be handled, and Better-Auth has published guidance for teams thinking about migration.

Why this matters

- **Consolidation**: Rather than keeping two competing ecosystems, the stewardship under Better-Auth should help the community converge on a single, more capable solution.
- **Roadmap alignment**: Better-Auth plans to bring missing capabilities (for example, stateless session support without a database) into its core, which reduces fragmentation and long-term maintenance burden for teams.
- **Continuity**: If you're already using Auth.js/NextAuth.js, the immediate work is to keep running as before — the new maintainers will keep fixing security issues and producing migration guides.

My recommendation

-

If you're starting a new project today: consider Better-Auth first. It's where the active development and direction are headed, and it's built with the kinds of primitives I keep needing.

- **If you maintain an existing Auth.js project**: you can continue operating as-is. Check Better-Auth's migration guide and roadmap if you want to plan a migration — there's no urgent need to move unless you want the new features.

Final thoughts

Auth.js helped a lot of teams ship real auth quickly. Better-Auth grew out of those same needs and now carries the torch forward. This transition should mean fewer duplicated efforts and a clearer path for teams to own their auth. I'll be watching the migration guides and trying out new Better-Auth features in upcoming projects.

Need help?

If you'd like help learning how to implement authentication in web applications, I can guide you step-by-step: choosing and configuring an auth library (Better Auth, Auth.js, or custom), designing secure session/token flows, and writing example code. I especially enjoy helping teams migrate Delphi desktop applications to the web—I'll map your existing auth and session model to a web-friendly approach, outline a migration checklist, and provide practical tips to avoid common traps.

Send details about your app and requirements and I'll suggest a concrete plan.

Source of this news

- Better-Auth announcement

INDEPENDENT · AD-FREE · READER-SUPPORTED

Free to read. Not free to make.

Every tutorial, article, and line of example code on Holger's Code is published without a paywall and without ads. This page is about what keeps it that way.

Professional work, published free

The content here is held to the same standard as professional client work: every tutorial is built as a real project, tested against current versions, documented step by step, and revisited when the frameworks move on. A single in-depth tutorial routinely takes days — and that's before the bills behind it: servers, storage, build infrastructure, licenses, and the hardware everything is verified on.

If an article or tutorial here has saved you an afternoon of debugging, taught you a technique, or helped you ship, it created real value for you. Supporting it isn't charity — it's paying for professional work you found useful, at whatever price you choose. And if you can't, keep reading anyway. That's what the free part means.

Ways to support

KO-FI

Buy me a coffee

A one-time tip — no account, no subscription, any amount.

BOOKS & COURSES

Buy the paid work

The paid tier of everything published free here — and you get something lasting in return.

SPREAD THE WORD

Share an article

Send a tutorial to a colleague, cite an article, subscribe to the RSS feed.

Nothing here goes behind a paywall, whether you chip in or not. But if something on this site earned its keep today, this is where to say so.

Support on Ko-fi —