Go full-stack with a framework

React is a library. It lets you put components together, but it doesn't prescribe how to do routing and data fetching. To build an entire app with React, we recommend a full-stack React framework like Next.js or Remix.

Can I use React without a framework?

Hide Details

You can definitely use React without a framework—that's how you'd use React for a part of your page. **However, if you're building a new app or a site fully with React, we recommend using a framework.**

Here's why.

Even if you don't need routing or data fetching at first, you'll likely want to add some libraries for them. As your JavaScript bundle grows with every new feature, you might have to figure out how to split code for every route individually. As your data fetching needs get more complex, you are likely to encounter server-client network waterfalls that make your app feel very slow. As your audience includes more users with poor network conditions and low-end devices, you might need to generate HTML from your components to display content early—either on the server, or during the build time. Changing your setup to run some of your code on the server or during the build can be very tricky.

"Fullstack Architektur-Vision"

https://react.dev/learn/start-a-new-react-project#which-features-make-up-the-react-teams-full-stack-architecture-vision

React "Fullstack Architektur-Vision"

https://react.dev/learn/start-a-new-react-project#which-features-make-up-the-react-teams-full-stack-architecture-vision

- **React Server Components** (RSC):
  - Komponenten, die auf dem Server und im Build gerendert werden können
  - Data Fetching "integriert"

React "Fullstack Architektur-Vision"

https://react.dev/learn/start-a-new-react-project#which-features-make-up-the-react-teams-full-stack-architecture-vision

- **React Server Components** (RSC):
  - Komponenten, die auf dem Server und im Build gerendert werden können
  - Data Fetching "integriert"

- **Suspense**:
  - Platzhalter für "langsame" Teile einer Seite
  - Mit Streaming können diese Teile einer Seite "nachgeliefert" werden, sobald sie gerendert sind

React empfiehlt "Fullstack-Framework"

- **Server Components** erfordern Rendern auf dem Server oder im Build

- Dazu braucht man ein "**Fullstack-Framework**"

React empfiehlt "Fullstack-Framework"

- **Server Components** erfordern Rendern auf dem Server oder im Build
- Dazu braucht man ein "**Fullstack-Framework**"
- "**Framework**" ist verharmlosend, weil es sich in der Regel um einen kompletten Stack samt Build-Tools und Laufzeitumgebung handelt

React empfiehlt "Fullstack-Framework"

- **Server Components** erfordern Rendern auf dem Server oder im Build
- Dazu braucht man ein "**Fullstack-Framework**"
- "**Framework**" ist verharmlosend, weil es sich in der Regel um einen kompletten Stack samt Build-Tools und Laufzeitumgebung handelt
- Deswegen werden solche Frameworks auch als "**Meta-Frameworks**" bezeichnet (=> Sammlung von Frameworks)

React empfiehlt "Fullstack-Framework"

- **Next.js** entspricht den Vorstellungen des React-Teams

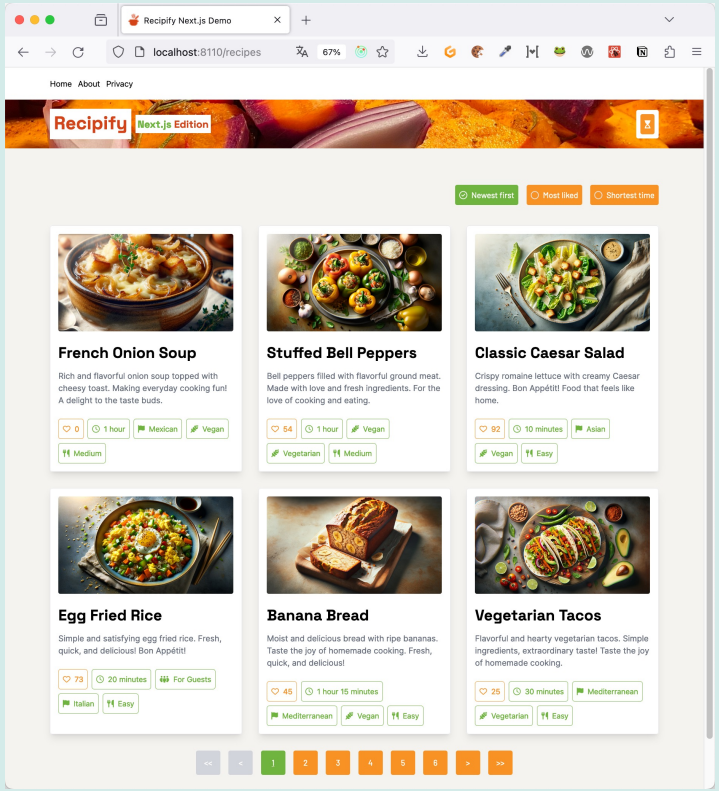
React empfiehlt "Fullstack-Framework"

- **Next.js** entspricht den Vorstellungen des React-Teams
- **Remix** unterstützt noch keine RSC, hat aber ähnliche Features
  - RSC-Support für Version 3 angekündigt

React empfiehlt "Fullstack-Framework"

- **Next.js** ist **React** und **mehr**
  - Man kann damit eine ganze Anwendung samt Backend bauen
  - Caching
  - Statische Seiten generieren
  - API Routes
  - Frontend ist dann ein Teil der Anwendung

Let's cook @ localhost:8110



HTTPS://GITHUB.COM/NILSHARTMANN/NEXTJS-WORKSHOP