

React Framework Showdown 🥊

DISCLAIMER 🤪

Every framework has its use cases
and a lot of work goes into making
web better 🕸

GOAL 🤔

Help you assess the **pros and cons** of each framework so you can **feel empowered** to make that decision yourself

OATH TO BE UNBIASED 🤘



HI, I'M ANKITA! 🙌



Ankita Kulkarni

**Educator | Senior Engineering Leader |
Developer**

bit.ly/reactrallylinks

[X @kulkarniankita9](https://twitter.com/kulkarniankita9)

[@kulkarniankita](https://www.youtube.com/@kulkarniankita)

SLIDES & MORE...



Slides and code snippets at:
bit.ly/reactrallylinks

RAISE YOUR HAND 🙋

How many of you have used
Next.js before?

RAISE YOUR HAND 🙋

How many of you have used
Remix before?

RAISE YOUR HAND 🙋

**How many of you have used
Astro before?**

We'll be comparing Astro, Remix and Next.js

Your tool choice 🛠️ matters **much less** than **your skill at using the tool** to accomplish your desired outcome (**a great user experience**) 👉.

- Kent C Dodds

WHAT IS A GOOD DEVELOPER EXPERIENCE?

What is Remix?

WHAT MAKES REMIX SPECIAL

It unifies the client and server with web standards so you can think less about code and more about your product.

When you google

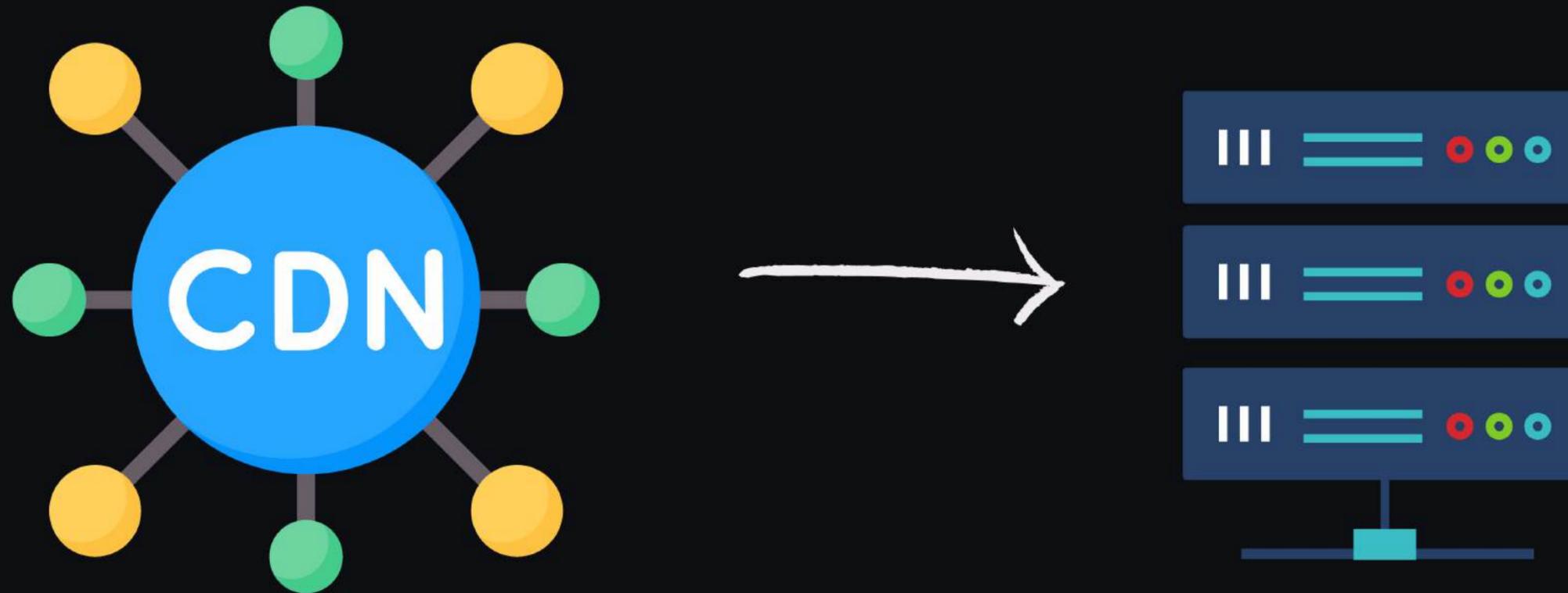
You go to MDN docs vs docs created by Remix

The screenshot shows the MDN web docs page for the `<form>` HTML element. The page title is "`<form>`: The Form element". The left sidebar lists various HTML elements, with `<form>` highlighted. The main content area includes a description: "The `<form>` HTML element represents a document section containing interactive controls for submitting information." Below this is a "Try it" section with a code editor and a live preview. The code editor shows the following HTML code:

```
1 <form action="" method="get" class="form-example">
2   <div class="form-example">
3     <label for="name">Enter your name: </label>
4     <input type="text" name="name" id="name"
5       required />
6   </div>
7   <div class="form-example">
8     <label for="email">Enter your email:
9     </label>
10    <input type="email" name="email" id="email"
11      required />
12  </div>
13  <div class="form-example">
14    <input type="submit" value="Subscribe!" />
15  </div>
16 </form>
```

The live preview shows a form with two text input fields and a "Subscribe!" button. The first field is labeled "Enter your name:" and the second is labeled "Enter your email:". There is a "RESET" button in the top right corner of the demo area.

Flexibility with Remix



When you bet on Remix, **you bet on web standards**, Vite eco-system etc.

What is Next.js? 🚀

NEXT.JS

It is a framework built on top of React that gives you the flexibility of building scalable apps by allowing you to render content on the server

WHAT MAKES NEXT.JS SPECIAL?

Data fetching mechanisms

Custom components

Server Components

Automatic Code Splitting

A layer on top

BUILDING STATIC & DYNAMIC CONTENT

EASY SELL UPLOAD >

OUR TOP PRODUCTS

You can pay to boost your products here.



MUSHROOM ORANGE LAMP
WHETHER YOU'RE LOOKING TO AD...
\$49



JEEP WRANGLER
EMBARK ON YOUR NEXT ADVENTU...
\$7900



SLICK SNEAKERS PREMIUM
LIMITED EDITION DESIGNS, EACH...
\$4500

ALL PRODUCTS



GRAY COUCH
TRANSFORM YOUR LIVING SPACE WIT...
\$800



RED NIKE SHOES
DON'T JUST WEAR SHOES - MAKE A S...
\$98



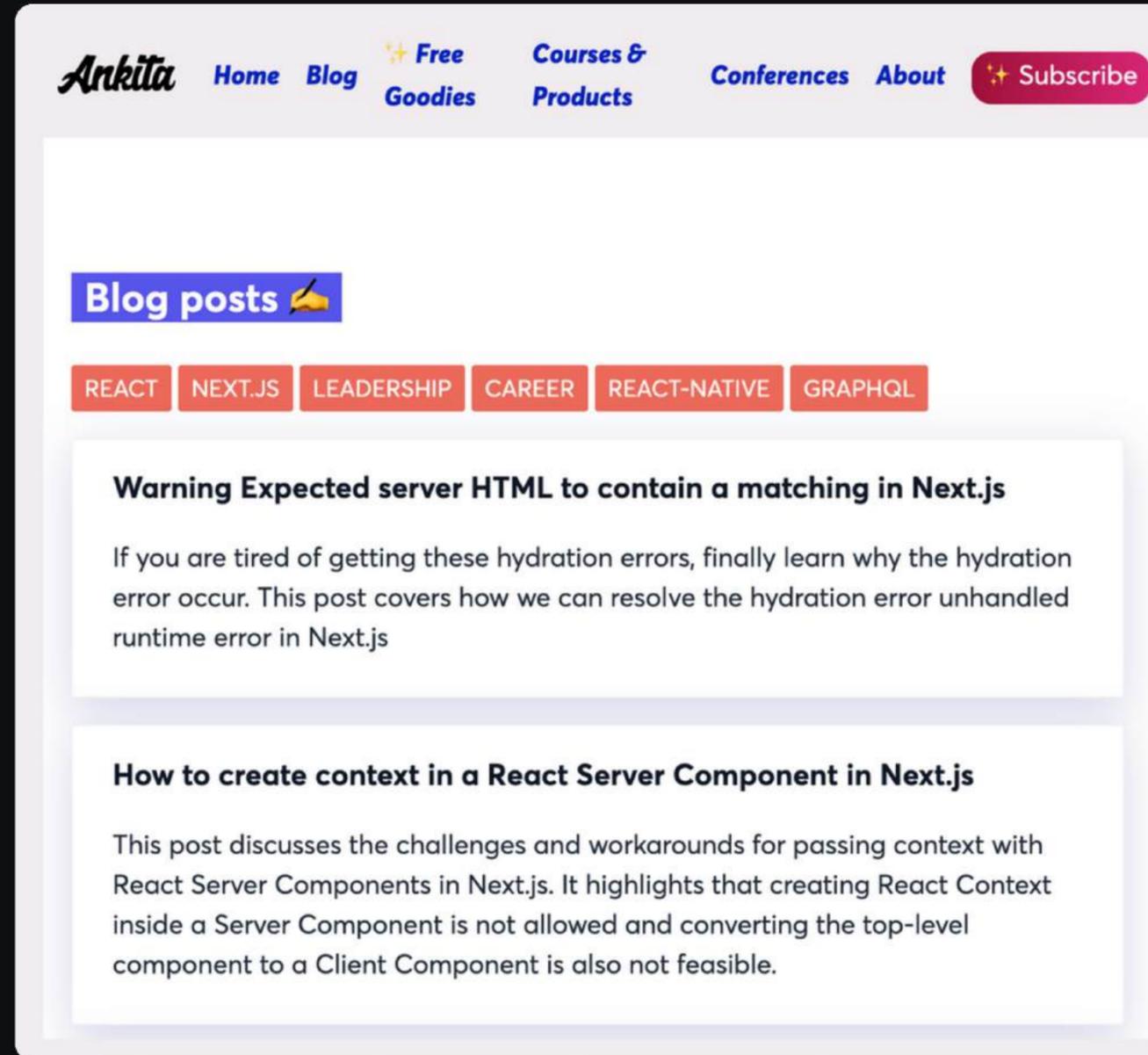
RAINBOW SILICONE HEAT INSULATION PAD
SOFT TABLE MAT, HEAT RESISTANT SL...
\$10



MUSHROOM ORANGE LAMP
WHETHER YOU'RE LOOKING TO ADD ...
\$49

What is Astro?

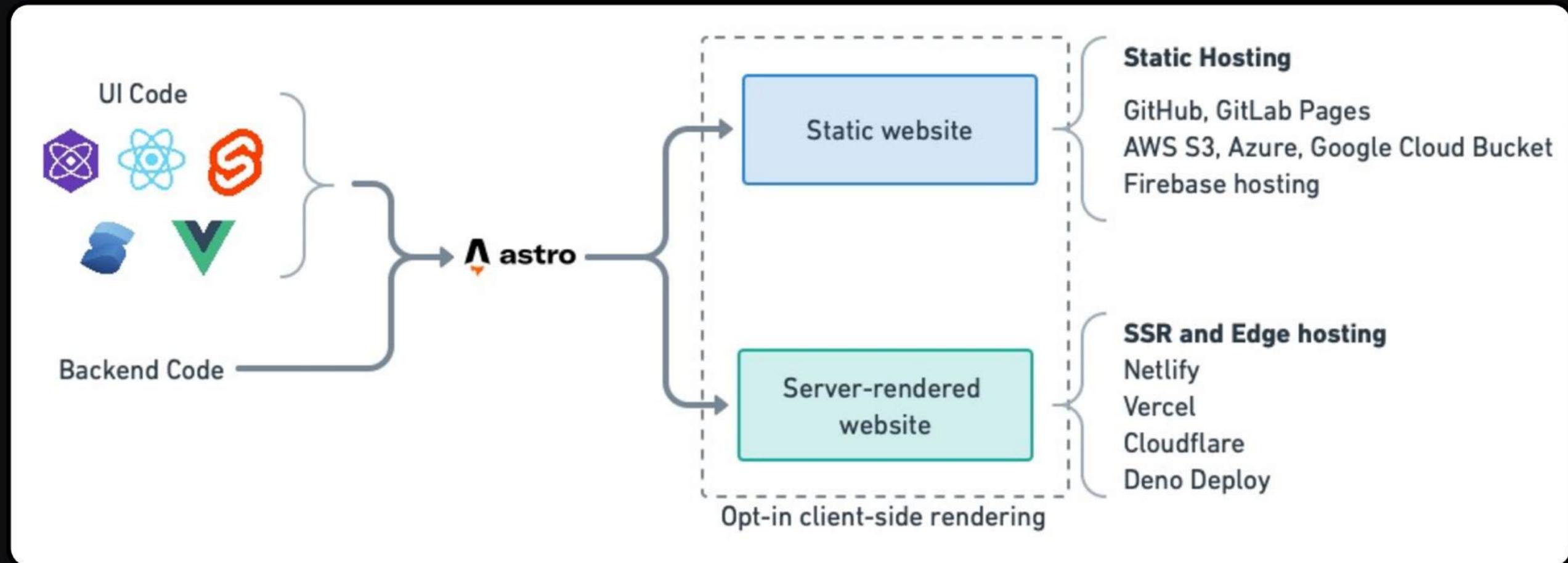
BUILDING STATIC CONTENT



The screenshot shows the top navigation bar of the Ankita website. The navigation items are: Home, Blog, Free Goodies (with a star icon), Courses & Products, Conferences, About, and a Subscribe button (with a star icon). Below the navigation is a 'Blog posts' section with a pencil icon. There are six category tags: REACT, NEXT.JS, LEADERSHIP, CAREER, REACT-NATIVE, and GRAPHQL. Two blog post previews are visible:

- Warning Expected server HTML to contain a matching in Next.js**
If you are tired of getting these hydration errors, finally learn why the hydration error occur. This post covers how we can resolve the hydration error unhandled runtime error in Next.js
- How to create context in a React Server Component in Next.js**
This post discusses the challenges and workarounds for passing context with React Server Components in Next.js. It highlights that creating React Context inside a Server Component is not allowed and converting the top-level component to a Client Component is also not feasible.

ASTRO



[Reference](#)

Astro works with any framework

UI Frameworks



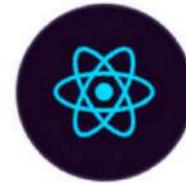
@astrojs/alpinejs



@astrojs/lit



@astrojs/preact



@astrojs/react



@astrojs/solid-js



@astrojs/svelte



@astrojs/vue

You don't need React or Vue to get started

```
about.astro
---
import BaseLayout from "../layouts/BaseLayout.astro";

const pageTitle = "About Me";
const skills = ["HTML", "CSS", "JavaScript", "React", "Astro", "Writing Docs"];

const skillColor = "navy";
---

<style define:vars={{ skillColor, fontWeight, textCase }}>
  .skill {
    color: var(--skillColor);
  }
</style>

<BaseLayout pageTitle={pageTitle}>
  <h2>... and my new Astro site!</h2>

  <p>My skills are:</p><p></p><ul>
    {skills.map((skill) => <li class="skill">{skill}</li>)}
  </ul>
</BaseLayout>
```

But you can get React if you'd like



6 PILLARS

Adoption

Developer
Experience

Performance

Routing

Data Fetching

Deployment

Adoption

bit.ly/reactrallylinks

 [@kulkarniankita9](https://twitter.com/kulkarniankita9)

 [@kulkarniankita](https://www.youtube.com/@kulkarniankita)

**Learning Curve
with Next.js**



**Learning Curve
with Remix**

THINGS TO LEARN

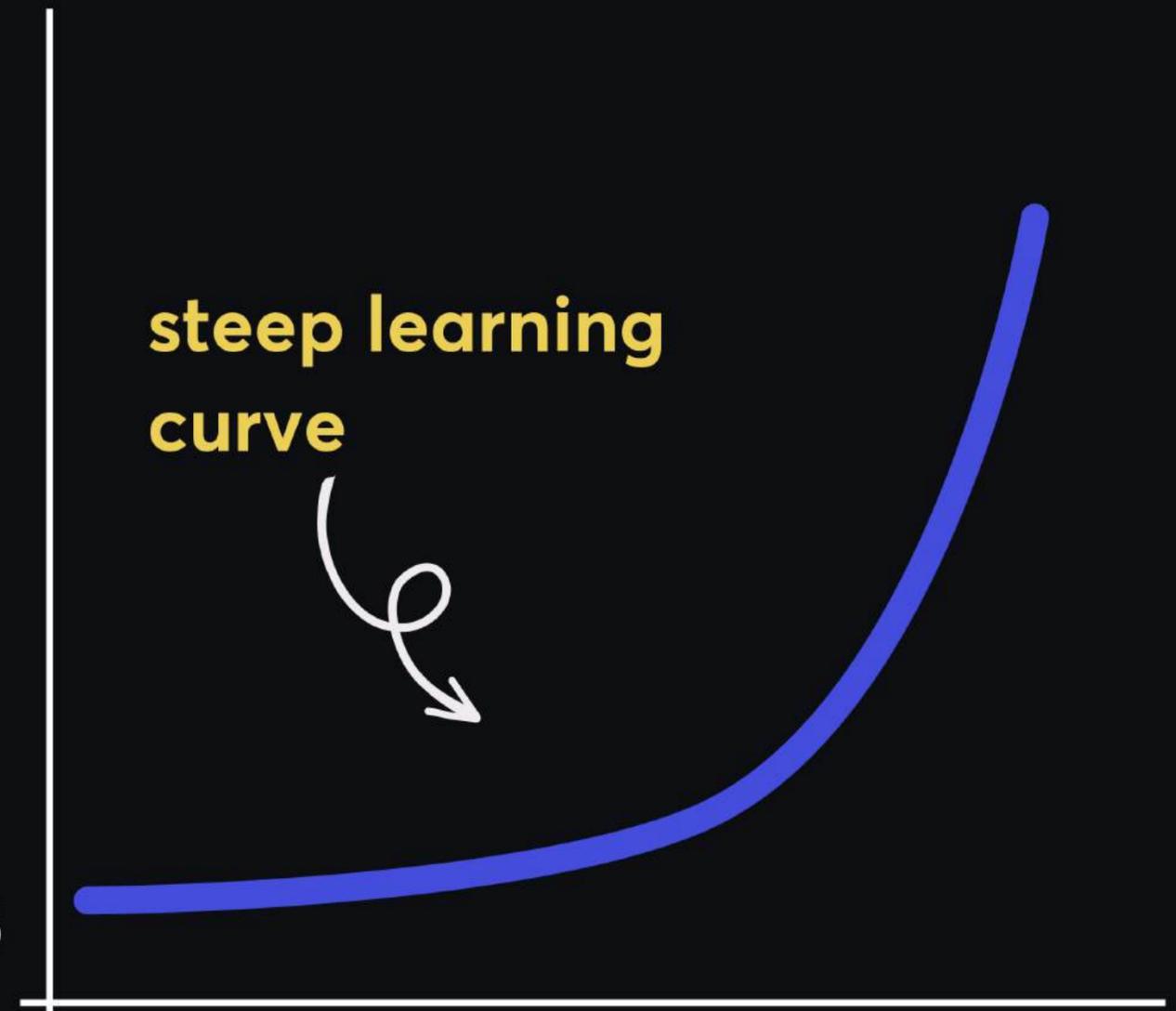
App & Pages Router

Server Actions

Server Components

Different behaviours in versions

Different rendering methods



ASTRO



UNTIL YOU INTRODUCE SSR



Course Platform Build Story 📖

Make your docs shine with Starlight

Everything you need to build a stellar documentation website. Fast, accessible, and easy-to-use.

[Get started](#) →

[View on GitHub](#) 



REMIX

React Router V7 🤝 Remix

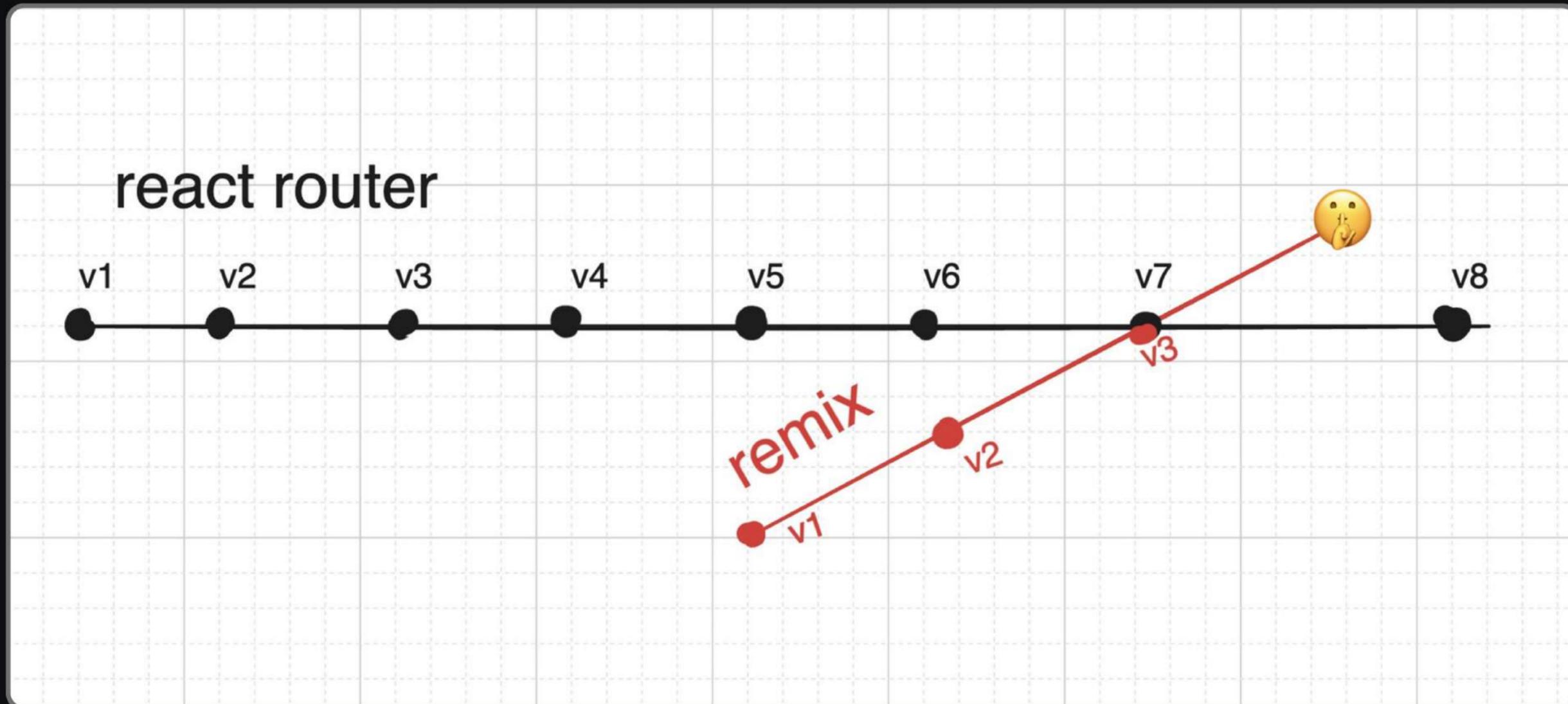
There are millions of projects using React Router, many built on top of Create React App (CRA). These days CRA is no longer recommended and [the React docs recommend using a framework](#). Since Remix has always been effectively **"React Router: The Framework"**, we wanted to create a bridge for all these React Router projects to be able to upgrade to Remix.

Turns out we made that bridge a little too well, specifically with the introduction of [our Vite plugin](#) and [SPA Mode](#). We found ourselves looking at Remix, then looking at React Router, then looking back at Remix, and we could no longer meaningful tell the difference.



So we're remixing React Router (again).

REMIX



Ecosystem
around Next.js

 7 yrs ago



Ecosystem
around Remix

 3 yrs ago

Routing

bit.ly/reactrallylinks

[X](#) [@kulkarniankita9](#)

[▶](#) [@kulkarniankita](#)

Next.js



Pages Router

Before Next.js 13



App Router

After Next.js 13

Next.js



Pages Router

Before Next.js 13

```
└─ pages
  └─ blog
     JS index.js
  └─ conferences
     JS [id].js
     JS index.js
```

Next.js

```

├── app
│   ├── (course-platform)
│   │   ├── watch / [lessonId]
│   │   │   ├── page.tsx
│   │   │   └── layout.tsx
│   │   └── fonts
│   │       ├── layout.tsx
│   │       └── page.tsx

```



App Router

After Next.js 13

Remix



Pages Router

Before Next.js 13

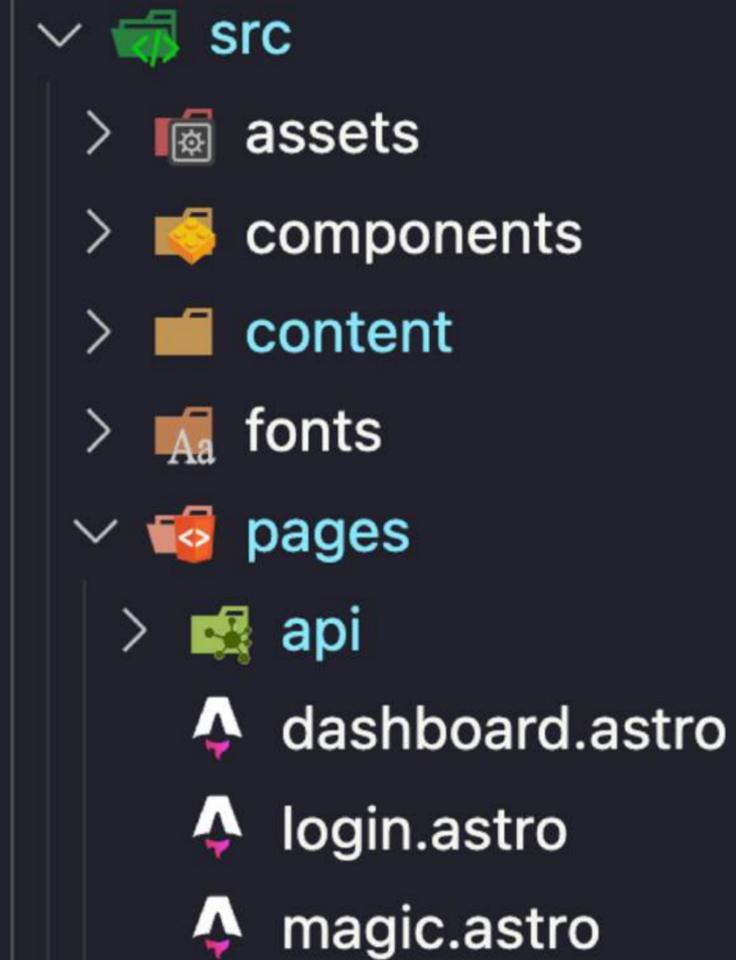
```
└─ app
  └─ routes
     ├── _index.tsx
     ├── about.tsx
     ├── concerts._index.tsx
     └── concerts.$city.tsx
```

Astro



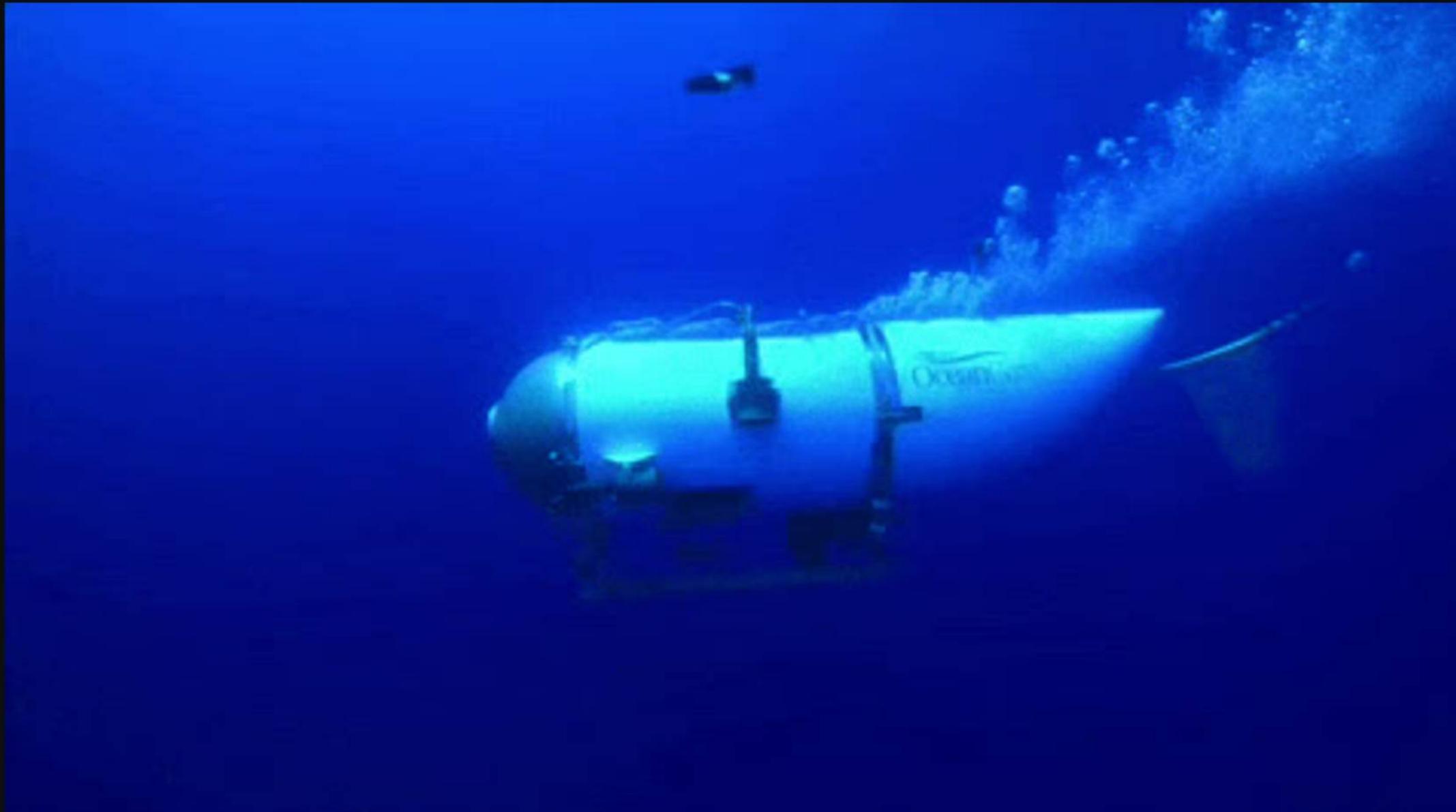
Pages Router

Before Next.js 13



Developer Experience

NEXT.JS 13 APP ROUTER



Things are way better in Next 14

← → 1 of 1 error ● Next.js is up to date ✕

Unhandled Runtime Error

Error: Hydration failed because the initial UI does not match what was rendered on the server.
See more info here: <https://nextjs.org/docs/messages/react-hydration-error>

In HTML, <div> cannot be a descendant of <p>.
This will cause a hydration error.

```

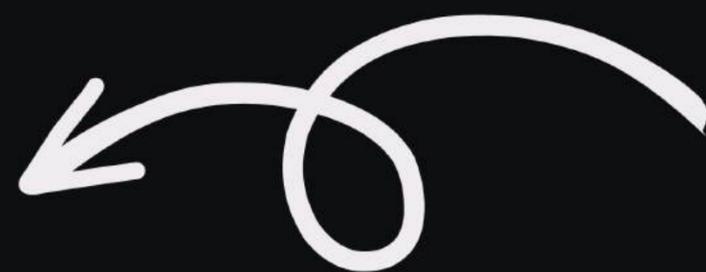
  ✓ <Home>
    <main>
      <p>
        <div>

```

Call Stack

>  React

Remix is way better



so fast 🔥

Remix itself is
just a Vite
plugin 🙌



TS vite.config.ts

```
import { vitePlugin as remix } from "@remix-run/dev";
import { defineConfig } from "vite";
import tsconfigPaths from "vite-tsconfig-paths";

export default defineConfig({
  plugins: [
    remix({
      ignoredRouteFiles: ["**/*.css"],
    }),
    tsconfigPaths(),
  ],
});
```

Astro is just incredible from onboarding, to docs, helping developers etc.

Performance

bit.ly/reactrallylinks

[X @kulkarniankita9](#)

[@kulkarniankita](#)

OUR TOP PRODUCTS

You can pay to boost your products here.



MUSHROOM ORANGE LAMP

WHETHER YOU'RE LOOKING TO AD...

\$49



JEEP WRANGLER

EMBARK ON YOUR NEXT ADVENTU...

\$7900



SLICK SNEAKERS PREMIUM

★ LIMITED EDITION DESIGNS: EACH...

\$4500

ALL PRODUCTS



GRAY COUCH

TRANSFORM YOUR LIVING SPACE WIT...

\$800



RED NIKE SHOES

DON'T JUST WEAR SHOES – MAKE A S...

\$98



RAINBOW SILICONE HEAT INSULATION PAD

SOFT TABLE MAT, HEAT RESISTANT SI...

\$10



MUSHROOM ORANGE LAMP

WHETHER YOU'RE LOOKING TO ADD ...

\$49

E-COMMERCE APP

<Image /> in Next.js

```
import { CardHeader } from '@components/ui/card';
import Image from 'next/image';

<CardHeader className="relative h-80">
  <Image
    src={recipe.image}
    alt={recipe.name}
    fill={true}
    className="bg-cover rounded-md shadow-xl"
    sizes="(max-width: 768px) 100vw, (max-width: 1200px) 50vw, 33vw"
  />
</CardHeader>;
```

Shadcn UI Library Card Header component

fill causes the image to fill the parent element

On smaller devices, image size would be 33vw

Without "sizes", it will be 100vw which can be 3x the size & cost you a slow render cycle

There is more...

Image Component 🌟

Script 📄

Link 🔗

Fonts 🖋️

Granular caching controls 🚀

Automatic code splitting ⚙️

Link Pre-fetching

All 3 frameworks enable instant transitions

Data Fetching

NEXT.JS HAS EXTENDED FETCH

Static Site Generation

`fetch('https://...', { cache: 'force-cache' })` → 🙄😞 Caches by default

Incremental Static ReGeneration

`fetch('https://...', { next: { revalidate: 3600 } })` → 🕒 Caches every 1 hour

Server-side Rendering

`fetch('https://...', { cache: 'no-store' })` → 🆕 No caching, fetches everytime!

they are changing this in Next.js 15 🎉

NEXT.JS HAS EXTENDED FETCH

Static Site Generation

`fetch('https://...', { cache: 'force-cache' })` → ~~🌟 Caches by default~~

Incremental Static ReGeneration

`fetch('https://...', { next: { revalidate: 3600 } })` → 🕒 Caches every 1 hour

Server-side Rendering

`fetch('https://...', { cache: 'no-store' })` → NEW No caching, fetches everytime!

new
default

they are changing this in Next.js 15

NEXT.JS

Server components for fetch ★



Server actions for mutations

page.tsx

```
async function getData() {
  const res = await fetch('https://nextjscourse.dev/invoices',
    { ...authHeaders });
  return res.json();
}

export default async function Page() {
  const { invoices } = await getData()
  return <main>Num of invoices:{invoices.length}</main>
}
```

Next.js

page.tsx

```
export default function Page() {
  async function invoiceAction(formData: FormData) {
    'use server'
    const amount = formData.get('amount');
    await createInvoiceInDb({ amount }); //mutate data
    revalidateData('/invoices'); // invalidate cache
  }
  return <form action={invoiceAction}> ... </form>
}
```

REMIX



 **MJ** 
@mjackson · [Follow](#)

Replying to @mjackson

You may be thinking: that's cheating, Michael! You should've built an abstraction around `wget` and taunted it as a new feature of Remix. 🙄

But `wget` is *the original* static site generator! With logging, cookies, base URL paths, progress... it's pretty nice! 🙌

9/12

12:50 AM · Oct 28, 2022

 23  Reply  Copy link

● ● ● TS routes/projects.ts

Component

```
// Component
export default function Projects() {
  const projects = useLoaderData<typeof loader>();
  const actionData = useActionData<typeof action>();

  return (
    <div>
      {projects.map((project) => (
        <Link key={project.slug} to={project.slug}>
          {project.title}
        </Link>
      ))}
      <Form method="post">
        <input name="title" />
        <button type="submit">Create New Project</button>
      </Form>
      {actionData?.errors ? (
        <ErrorMessage errors={actionData.errors} />
      ) : null}
    </div>
  );
}
```

● ● ● TS routes/projects.ts

Loader

```
// Loaders only run on the server and provide
// data to your component on GET requests
export async function loader() {
  return json(await db.projects.findAll());
}

// Previous Project Component
```

● ● ● TS routes/projects.ts

Actions

```
// Actions only run on the server and handle POST
// PUT, PATCH, and DELETE. They can also provide data
// to the component
export async function action({
  request,
}: ActionFunctionArgs) {
  const form = await request.formData();
  const errors = validate(form);
  if (errors) {
    return json({ errors });
  }
  await createProject({ title: form.get("title") });
  return json({ ok: true });
}
```

TS routes/projects.ts

Component

```
// Component
export default function Projects() {
  const projects = useLoaderData<typeof loader>();
  const actionData = useActionData<typeof action>();

  return (
    <div>
      {projects.map((project) => (
        <Link key={project.slug} to={project.slug}>
          {project.title}
        </Link>
      ))}
      <Form method="post">
        <input name="title" />
        <button type="submit">Create New Project</button>
      </Form>
      {actionData?.errors ? (
        <ErrorMessage errors={actionData.errors} />
      ) : null}
    </div>
  );
}
```

TS routes/projects.ts

Loader

```
// Loaders only run on the server and provide
// data to your component on GET requests
export async function loader() {
  return json(await db.projects.findAll());
}

// Previous Project Component
```

TS routes/projects.ts

Actions

```
// Actions only run on the server and handle POST
// PUT, PATCH, and DELETE. They can also provide data
// to the component
export async function action({
  request,
}: ActionFunctionArgs) {
  const form = await request.formData();
  const errors = validate(form);
  if (errors) {
    return json({ errors });
  }
  await createProject({ title: form.get("title") });
  return json({ ok: true });
}
```

● ● ● TS routes/projects.ts

Component

```
// Component
export default function Projects() {
  const projects = useLoaderData<typeof loader>();
  const actionData = useActionData<typeof action>();

  return (
    <div>
      {projects.map((project) => (
        <Link key={project.slug} to={project.slug}>
          {project.title}
        </Link>
      ))}
      <Form method="post">
        <input name="title" />
        <button type="submit">Create New Project</button>
      </Form>
      {actionData?.errors ? (
        <ErrorMessage errors={actionData.errors} />
      ) : null}
    </div>
  );
}
```

● ● ● TS routes/projects.ts

Loader

```
// Loaders only run on the server and provide
// data to your component on GET requests
export async function loader() {
  return json(await db.projects.findAll());
}

// Previous Project Component
```

● ● ● TS routes/projects.ts

Actions

```
// Actions only run on the server and handle POST
// PUT, PATCH, and DELETE. They can also provide data
// to the component
export async function action({
  request,
}: ActionFunctionArgs) {
  const form = await request.formData();
  const errors = validate(form);
  if (errors) {
    return json({ errors });
  }
  await createProject({ title: form.get("title") });
  return json({ ok: true });
}
```

TS routes/projects.ts

Component

```
// Component
export default function Projects() {
  const projects = useLoaderData<typeof loader>();
  const actionData = useActionData<typeof action>();

  return (
    <div>
      {projects.map((project) => (
        <Link key={project.slug} to={project.slug}>
          {project.title}
        </Link>
      ))}
      <Form method="post">
        <input name="title" />
        <button type="submit">Create New Project</button>
      </Form>
      {actionData?.errors ? (
        <ErrorMessage errors={actionData.errors} />
      ) : null}
    </div>
  );
}
```

TS routes/projects.ts

Loader

```
// Loaders only run on the server and provide
// data to your component on GET requests
export async function loader() {
  return json(await db.projects.findAll());
}

// Previous Project Component
```

TS routes/projects.ts

Actions

```
// Actions only run on the server and handle POST
// PUT, PATCH, and DELETE. They can also provide data
// to the component
export async function action({
  request,
}: ActionFunctionArgs) {
  const form = await request.formData();
  const errors = validate(form);
  if (errors) {
    return json({ errors });
  }
  await createProject({ title: form.get("title") });
  return json({ ok: true });
}
```

ASTRO



Rebuilt my (next.js) blog using [@astrodotbuild](#) out of curiosity...holy shit the difference in bundle size.

Home route: 138kb -> 7.6kb

"All posts": 570kb -> 100kb (85kb was images)

Seriously considering moving my static sites to this



From [astro.t3.gg](#)

ASTRO

Uses fetch Web API's

```
---
import Contact from '../components/Contact.jsx';

const response = await fetch('https://randomuser.me/api/');
const data = await response.json();
const randomUser = data.results[0];
---
<!-- Data fetched at build can be rendered in HTML -->
<h1>User</h1>
<h2>{randomUser.name.first} {randomUser.name.last}</h2>

<!-- Data fetched at build can be passed to components as props -->
<Contact client:load email={randomUser.email} />
```

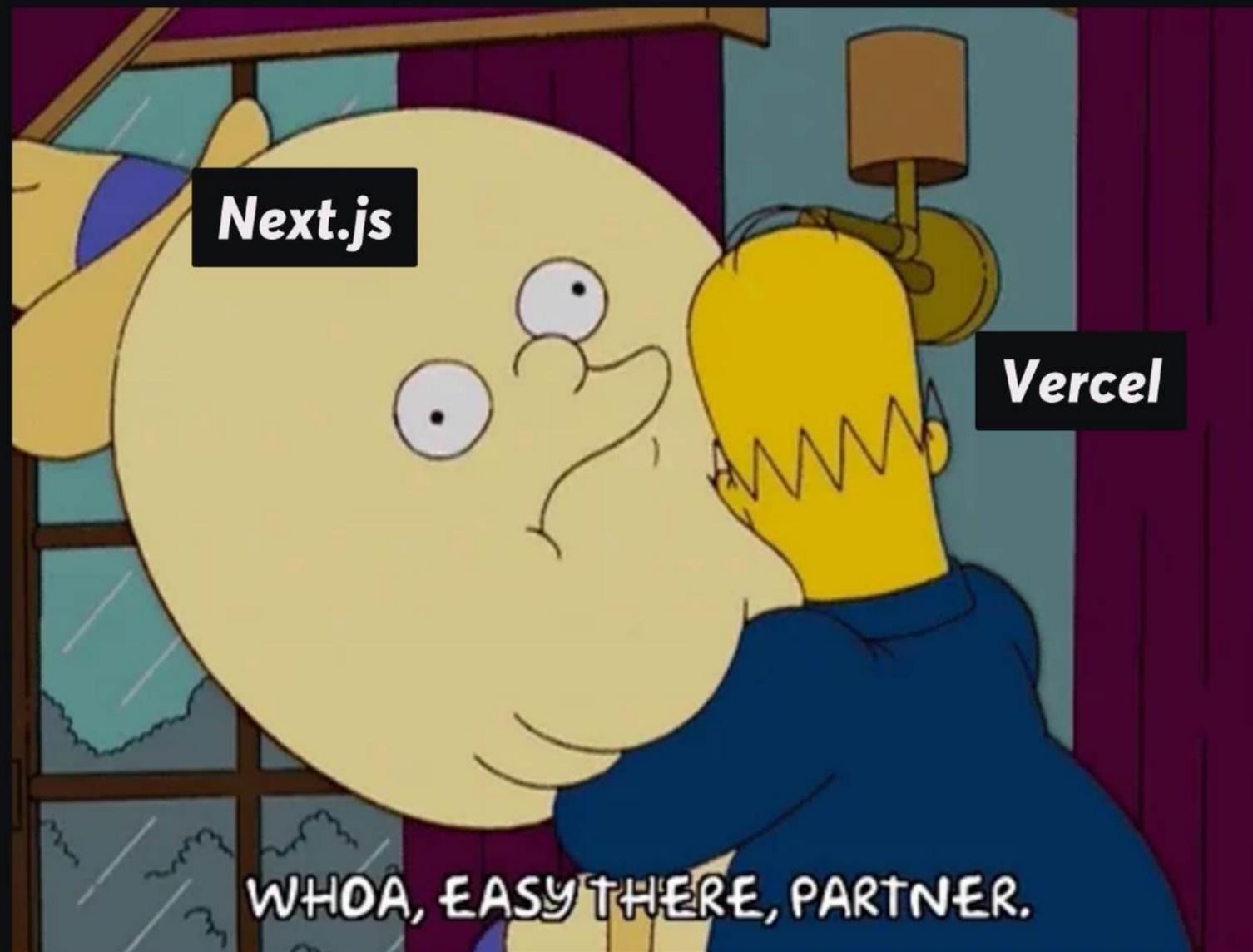
Deployment

bit.ly/reactrallylinks

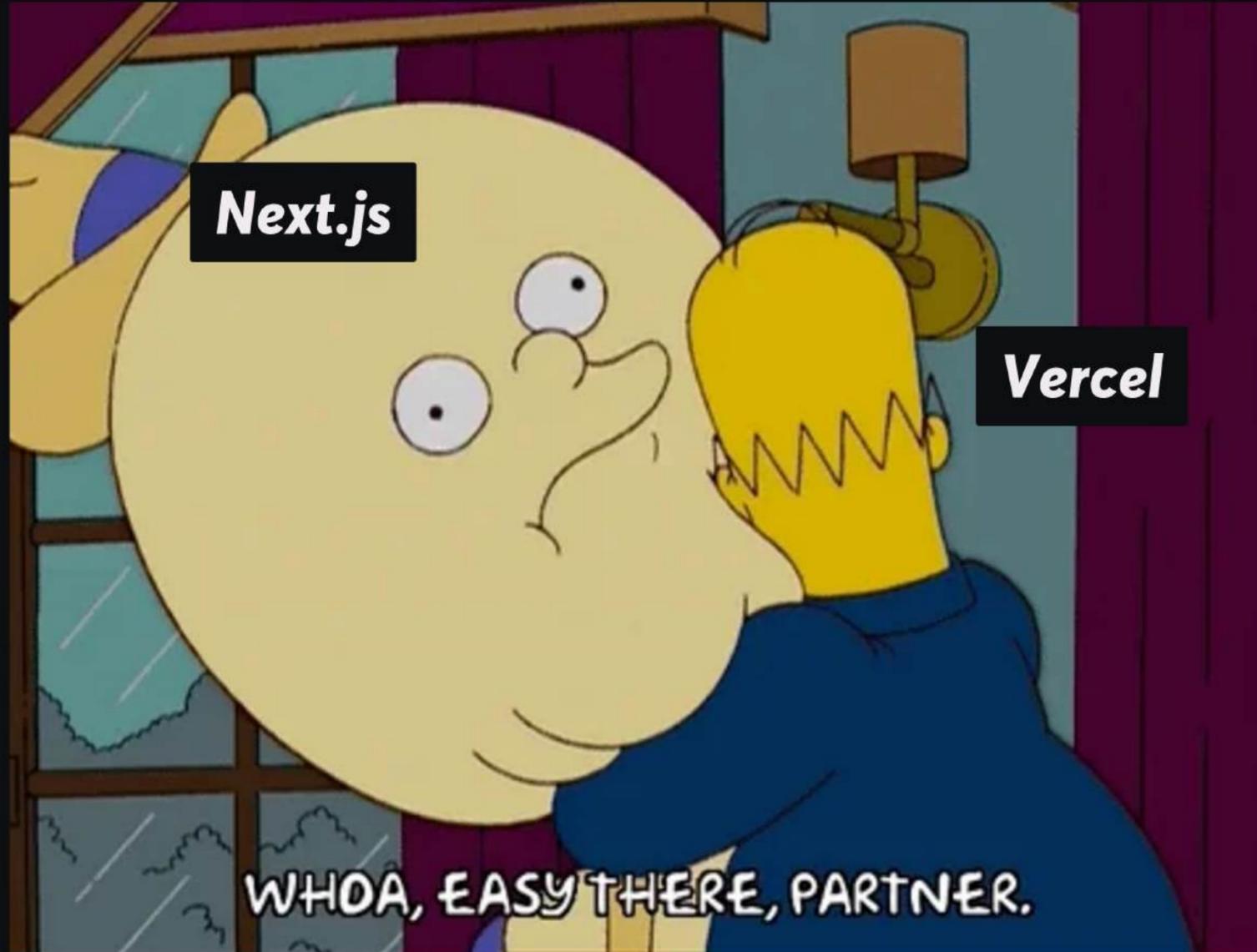
 [@kulkarniankita9](https://twitter.com/kulkarniankita9)

 [@kulkarniankita](https://www.youtube.com/@kulkarniankita)

NEXT.JS



NEXT.JS



tightly coupled 



Marc Lou ✓
@marc_louvion



Vercel: our pricing is too high
Also Vercel: let's double your monthly bill

Did I miss something?



Hey there,

I wanted to let you know we're **updating Vercel's pricing later this month.**

Our community's feedback has been clear: bandwidth and functions are too expensive. So we're making our infrastructure pricing more granular and giving you more ways to optimize.

Based on the way your application uses our infrastructure, your monthly bill for team **Marc Lou** is projected to go from **\$43 to \$86** on July 24th.

To allow time for optimization, we will provide your team with a **100% discount on new metrics** (Fast Origin Transfer and Edge Requests) for 3 months on new pricing.

AM · Apr 5, 2024 · **218.1K** Views

**But, all my products are on
Vercel...**

IT JUST WORKS!!

REMIX: CAN RUN ANYWHERE



cloudflare

fly.io

netlify

vercel

others

ASTRO

Static

Dynamic



ASTRO

Static



Dynamic



CHOOSE FRAMEWORK WHEN

CHOOSE NEXT.JS WHEN

01

**BUILD-IN OPTIMIZATIONS, USE
SERVER COMPONENTS**

02

**NEED GRANULAR CACHING
CONTROLS**

03

NO CUSTOM WEBPACK BUILDS

CHOOSE REMIX WHEN

01

LOTS OF DYNAMIC PAGES

02

SUPPORT FOR WEB APIS

03

**NO LIMITATION FOR
INTEGRATION TO CLOUD
PLATFORMS**

CHOOSE ASTRO WHEN

01

BUILDING CONTENT-DRIVEN WEBSITES

02

LOADS FAST & GREAT SEO

03

EASILY ACCESSIBLE TO ALL DEVELOPERS IRRESPECTIVE OF LIBRARIES



NEXT.JS

ADOPTION



DEVELOPER EXPERIENCE



PERFORMANCE



ROUTING



DATA FETCHING



DEPLOYMENT



REMIX

ADOPTION



DEVELOPER EXPERIENCE



PERFORMANCE



ROUTING



DATA FETCHING



DEPLOYMENT



ASTRO

ADOPTION



DEVELOPER EXPERIENCE



PERFORMANCE



ROUTING



DATA FETCHING



DEPLOYMENT



WINNER 

Remix

WINNER 

it depends....

BIT.LY/REACTRALLYLINKS

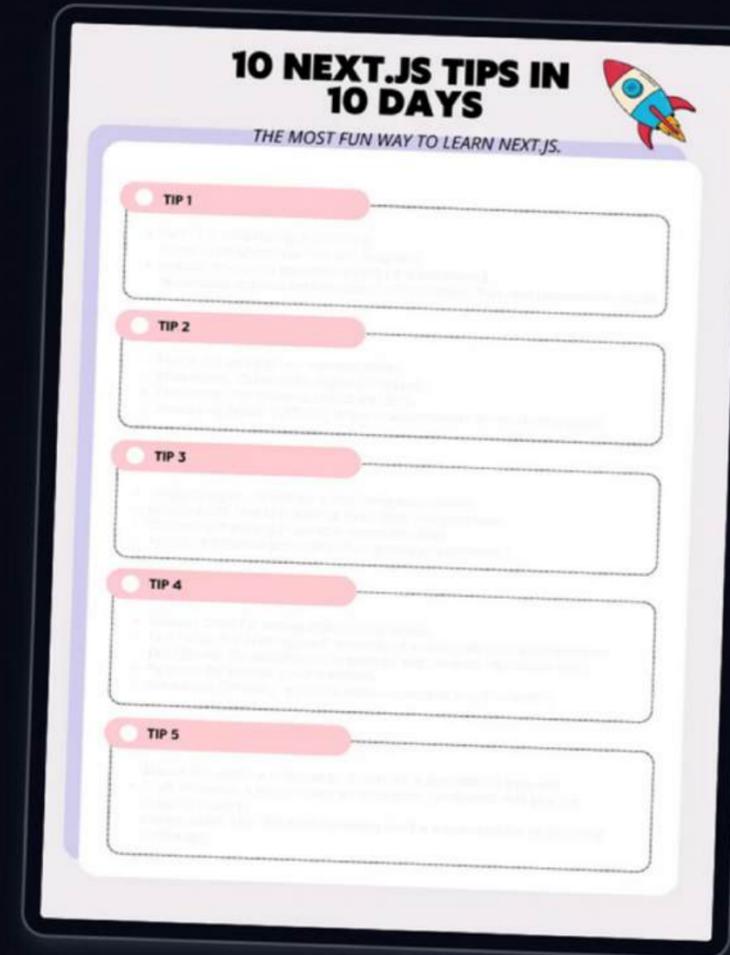
🔥 FREE: Email Course

Level up your Next.js skills w/ 10 digestible tips

*The most fun way to learn Next.js. 10 easy-to-follow tips
in 10 days!*

🚀 Get Lesson 1

📁 Bonus: FREE access to Next.js code repository with examples.



*Plus, get a downloadable guide at
the end of the course with all 10
tips.*

Thank you! 🙌🙌

ALL MY SLIDES ARE AT
[BIT.LY/REACTRALLYLINKS](https://bit.ly/reactrallylinks)

