



Is Remix your next
framework

Focused on web standards and modern web app UX, you're simply going to build better websites

Remix is a full stack web framework that lets you focus on the user interface and work back through web standards to deliver a fast, slick, and resilient user experience. People are gonna love using your stuff.

[Read the Docs](#)[Get Started](#)

```
export async function loader({ request }) {
  return getProjects();
}

export async function action({ request }) {
  const form = await request.formData();
  return createProject({ title: form.get("title") });
}

export default function Projects() {
  const projects = useLoaderData();
  const { state } = useTransition();
  const busy = state === "submitting";

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

Reaching the goal



Performance



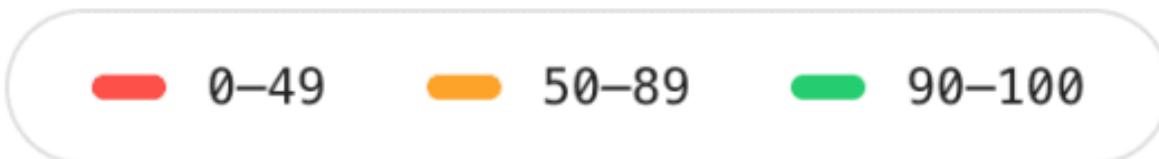
Accessibility



Best
Practices



SEO



SAME OLD PROBLEMS

NEW SOLUTIONS



SSR



SSG



ISR

Remix vs NextJS

Search Page, Cached, Virginia, Cable

Remix Rewrite

Remix Port

Next.js

0.0

0.0

0.0

<https://remix.run/blog/remix-vs-next>
<https://demo.vercel.store/>


```
4 export default function Projects() {
5   const projects = useLoaderData();
6   const { state } = useTransition();
7   const busy = state === "submitting";
8
9   return (
10     <div>
11       {projects.map((project) => (
12         <Link to={project.slug} key={project.id}>
13           {project.title}
14         </Link>
15       ))}
16       <Form method="post">
17         <input name="title" />
18         <button type="submit" disabled={busy}>
19           {busy ? "Saving..." : "Save"}
20         </button>
21       </Form>😊
22     </div>
23   )
24 }
25 }
```

Using what we know

**Without
the hard parts**

React without the hard parts



**Learn Remix,
Accidentally Learn the Web**

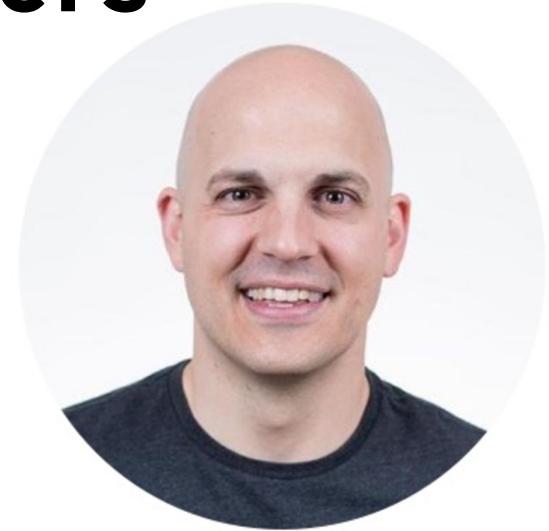
Remix is driven by community leaders



Kent C Dodds



Ryan Florence



Michael Jackson

Remix is about

- Developer friendliness
- Developer productivity
- Performance & speed
- Using web standard

A photograph of a person rappelling down a rope in a dark, underwater cave. The person is hanging vertically, facing away from the camera. The water is a deep greenish-blue, and the cave walls are rocky and textured. The lighting is dim, coming from above, which creates a dramatic effect.

Lets dive in Remix

ROUTING

Defining routes

```
1 app
2   └── root.jsx
3   └── routes
4     ├── accounts.jsx
5     ├── dashboard.jsx
6     ├── expenses.jsx
7     ├── index.jsx
8     ├── reports.jsx
9     └── sales
10    ├── customers.jsx
11    ├── deposits.jsx
12    ├── index.jsx
13    └── invoices
14      ├── $invoiceId.jsx
15      └── index.jsx
16      ├── invoices.jsx
17      └── subscriptions.jsx
18      └── sales.jsx
```

/sales/invoices/102000

- root.jsx
- routes/sales.jsx
- routes/sales/invoices.jsx
- routes/sales/invoices/\$invoiceId.jsx

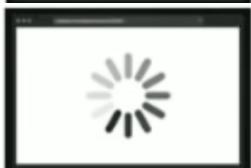
```
<Root>
  <Sales>
    <Invoices>
      <InvoiceId />
    </Invoices>
  </Sales>
</Root>
```

DATA LOADING

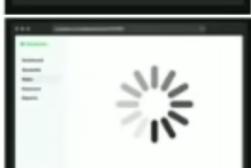
How can we describe performance



Time to First Byte (TTFB)



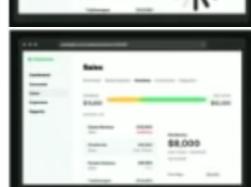
First Contentful Paint (FCP)



Cumulative Layout Shift (CLS)

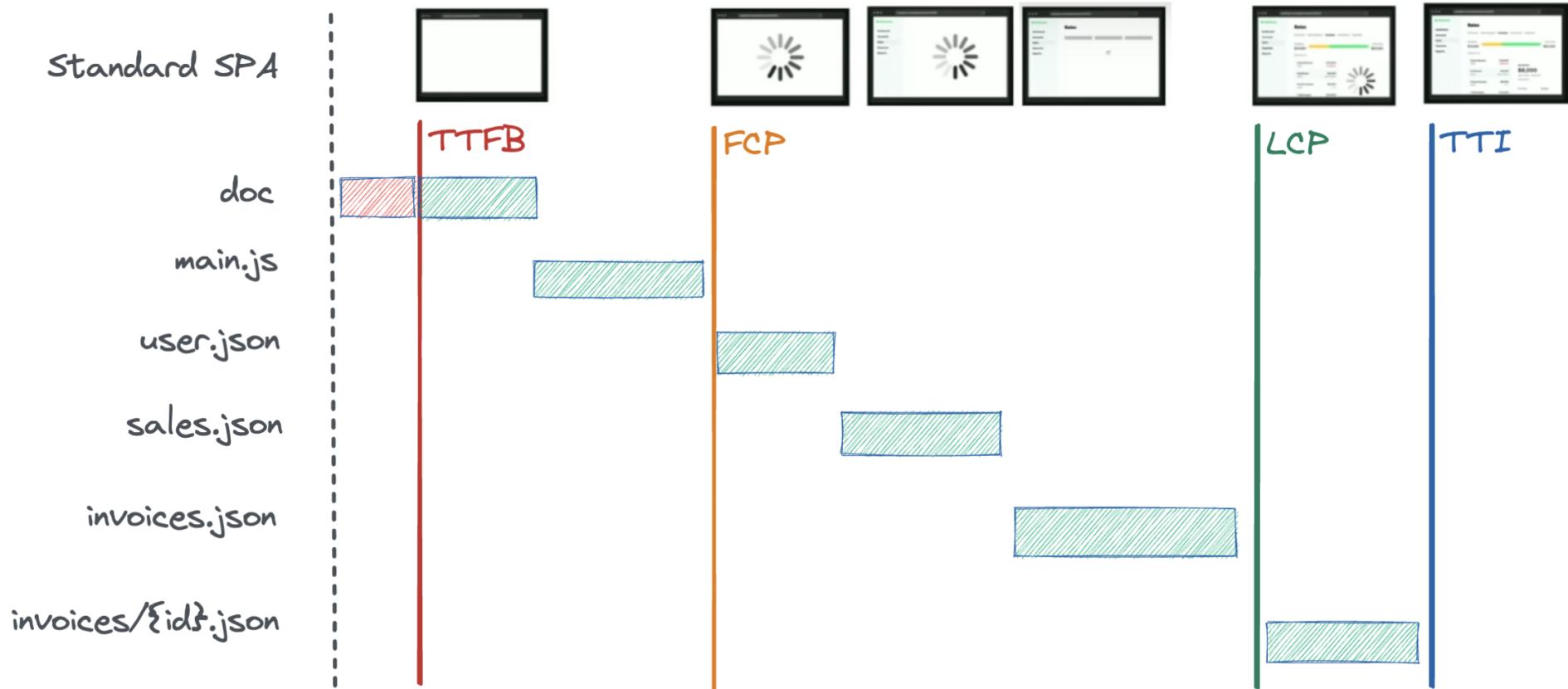


Largest Contentful Paint (LCP)

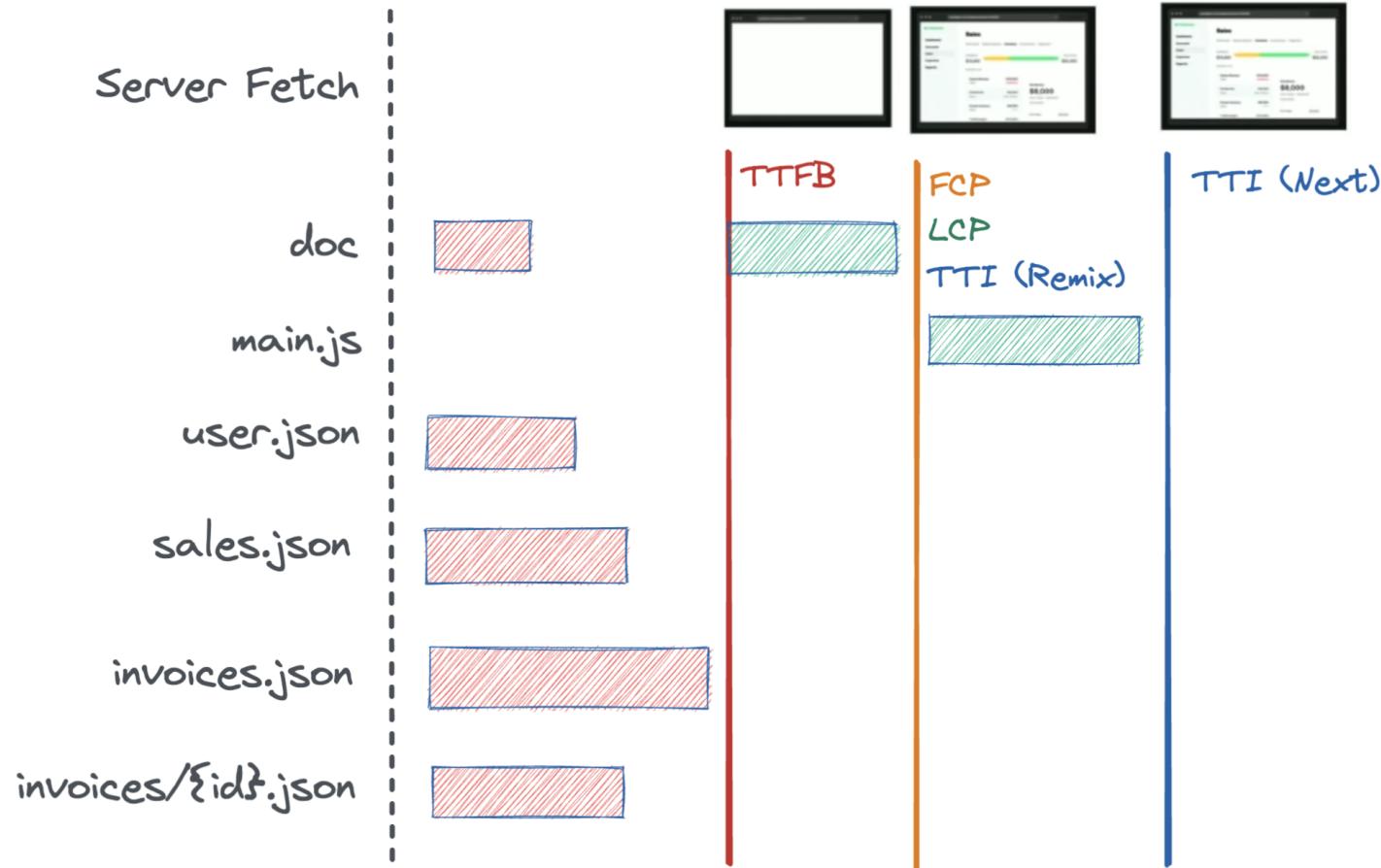


Time To Interactive (TTI)

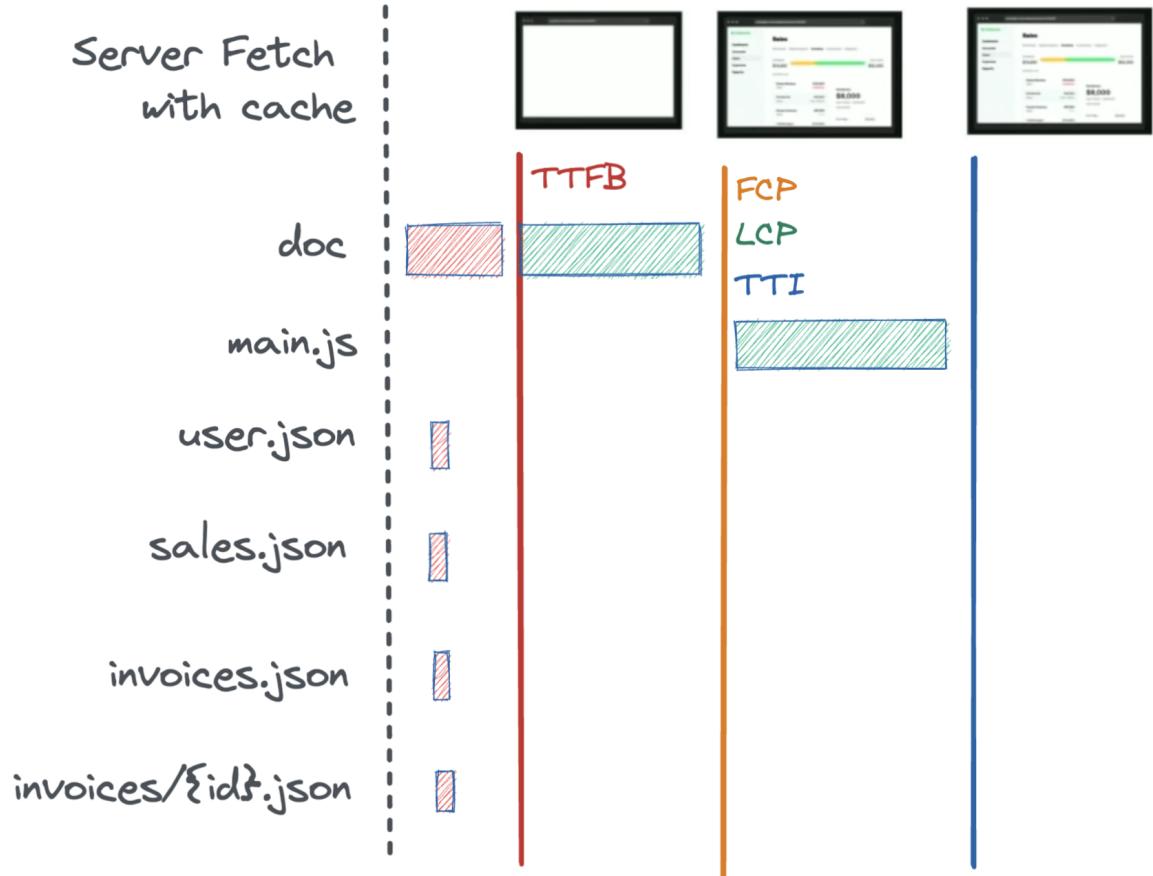
Rendering of a standard SPA



Server side rendering



Server side rendering with caching



Route with data fetching

```
1 import type { LoaderFunction } from "@remix-run/node"; // or cloudflare/deno
2 import { json } from "@remix-run/node"; // or cloudflare/deno
3 import { useLoaderData } from "@remix-run/react";
4 import { db } from "~/db.server";
5
6 export const loader: LoaderFunction = async ({ params }) => {
7   return json(
8     await db.product.findMany({
9       where: { categoryId: params.categoryId },
10      })
11    );
12  };
13
14 export default function ProductCategory() {
15   const products = useLoaderData();
16   return (
17     <div>
18       <p>{products.length} Products</p>
19       {/* ... */}
20     </div>
21   );
22 }
```

FORM ACTIONS

React Form

```
1 function MyForm() {
2   const [inputs, setInputs] = useState({});
3
4   const handleChange = (event) => {
5     const name = event.target.name;
6     const value = event.target.value;
7     setInputs(values => ({...values, [name]: value}))
8   }
9
10  const handleSubmit = (event) => {
11    event.preventDefault();
12    console.log(inputs);
13  };
14
15  return (
16    <form onSubmit={handleSubmit}>
17      <label>Enter your name:
18        <input type="text"
19          value={inputs.username || ""}
20          onChange={handleChange}
21        />
22      </label>
23    </form>
24  )
25 }
```

The **power** of Remix

GET /users

```
<a href="/users">Users</a>
```

GET /users?name=peter

```
<form action="/users">Users</form>
```

Query (form get)

```
1 import type { LoaderArgs } from "@remix-run/node"; // or cloudflare/deno
2 import { redirect } from "@remix-run/node"; // or cloudflare/deno
3
4 export const loader = async ({ request }: LoaderArgs) => {
5   const url = new URL(request.url);
6   const search = new URLSearchParams(url.search);
7   const query = search.get('query') || '';
8   const data = [....];
9   return json({ query, data });
10 };
11
12 export default function Users() {
13   const { data, query } = useLoaderData();
14   return (
15     <h1>Search page</h1>
16     <form method='get'>
17       <input type='text' name='query' defaultValue={query}/>
18     </form>
19   )
20 }
```

Mutation (form post)

```
1 import type { ActionArgs } from "@remix-run/node"; // or cloudflare/deno
2 import { redirect } from "@remix-run/node"; // or cloudflare/deno
3
4 export const action = async ({ request }: ActionArgs) => {
5   const formData = await request.formData();
6   const user = await db.save.user(formData);
7   return redirect(`users/${user.id}`);
8 };
9
10 export default function NewUser() {
11   return (
12     <form method="post" action="/search">
13       <input name="name" type="text" />
14       <input name="email" type="email" />
15       <button type="submit">Submit</button>
16     </form>
17   )
18 }
```

Form validation - action

```
1 // Zod validation schema
2 export const formSchema = z.object({
3   name: z.string(),
4   email: z.string().email().optional()
5 });
6
7 export const action = async ({ request }) => {
8   const formPayload = Object.fromEntries(await request.formData());
9   try {
10     const formValues = formSchema.parse(formPayload);
11     const user = await db.save.user(formValues);
12     return redirect(`users/${user.id}`);
13   } catch (error) {
14     return {
15       formPayload,
16       errors: (error as ZodError).flatten().fieldErrors,
17     };
18   }
}
```

Form validation - handle errors

```
1 export default function NewUser() {
2   const actionData = useActionData();
3   return (
4     <form method="get" action="/search">
5       <input name="name" type="text" />
6       {actionData?.errors.name && (
7         <span className="text-red-600">{actionData?.errors.name[0]}</span>
8       )}
9       <input name=email" type="email" />
10      {actionData?.email.name && (
11        <span className="text-red-600">{actionData?.errors.email[0]}</span>
12      )}
13      <button type="submit">Submit</button>
14    </form>
15  )
16}
```

CACHING

Caching

```
1 // data cache
2 export const loader: LoaderFunction = async ({ params }) => {
3   const products = await db.product.findMany({
4     where: { categoryId: params.categoryId },
5   })
6   return json(products, {
7     "Cache-Control": "public, s-maxage=600",
8   });
9 }
```

```
1 // route cache
2 export let headers: HeadersFunction = () => {
3   return {
4     "Cache-Control": "public, s-maxage=600",
5   };
6 }
```

ERROR HANDLING

Global Error handling

```
1 // root.aspx
2 export function ErrorBoundary({ error }) {
3   return (
4     <html>
5       <head>
6         <title>Oh no!</title>
7         <Meta />
8         <Links />
9       </head>
10      <body className='m-4'>
11        <h1 className='text-2xl'>Something went wrong!</h1>
12        <p>{error.message}</p>
13        <Scripts />
14      </body>
15    </html>
16  );
17 }
```

Route Error handling

```
1 // my route component
2 export default () => {
3   return (
4     <div>...</div>
5   )
6 }
7
8 // Error Boundary Component
9 export function ErrorBoundary({ error }) {
10   return (
11     <div>
12       <h1>Error</h1>
13       <p>{error.message}</p>
14       <p>The stack trace is:</p>
15       <pre>{error.stack}</pre>
16     </div>
17   );
18 }
```

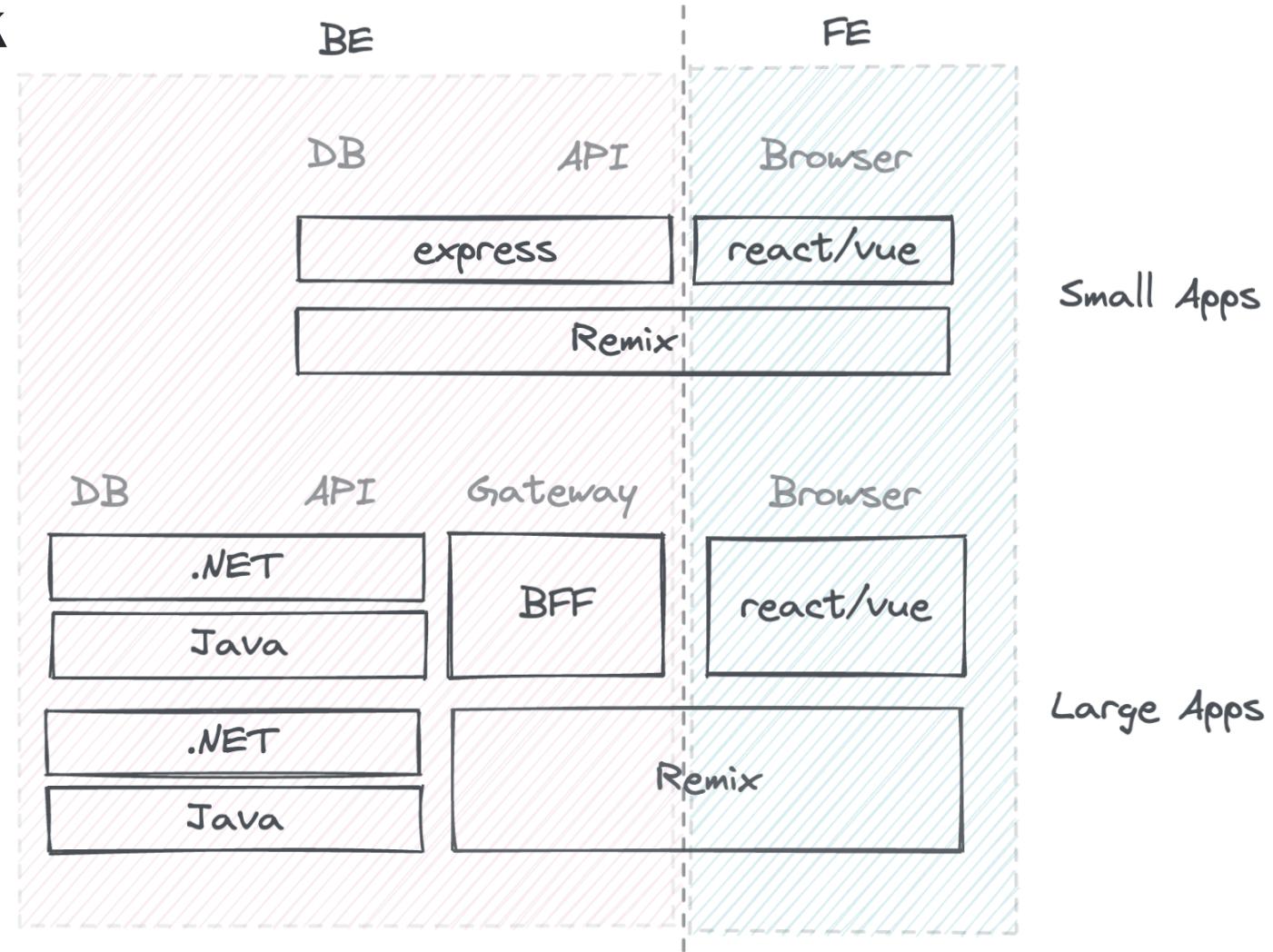
Catch Error (http status errors)

```
1 // We can throw a 404 error
2 export const loader = async ({ params }: LoaderArgs) => {
3   const id = params.id;
4   const category = await getDb().categories.findOne({ _id: new SafeObjectId(id) });
5   if (!category) {
6     throw json({ status: 404, statusText: 'Not Found' });
7   }
8
9   return json({ category });
10};
```

```
1 // Status Catch Boundary
2 export function CatchBoundary() {
3   const { data } = useCatch();
4   return (
5     <div>
6       <p>Resources not found [{data.status}]</p>
7     </div>
8   );
9 }
```

STACK

Software Stack



Software Spectrum

When BE is
.NET

React / Vue / Svelte

Electron / tauri / React Native

Nuxt / Next / SvelteKit

Remix

Astro

Type:	Static Site	Mostly Static w/ updates	Mostly Static but Often updated	Dynamic web app	Dynamic App w/ native reqs
Example:	Landing Page	Corporate Site / Portals	ECommerce	Business Application	Video Editing

A photograph of a young man with dark hair and a beard, wearing a light blue button-down shirt and a dark blue patterned tie. He is seated on a light-colored couch, gesturing with his hands as if he is in the middle of a conversation or presentation. The background shows vertical blinds on a window.

**So less code
faster UX
better DX**

Is Remix your next framework?





Want to learn more?

- [Remix Guide](#)
- [Remix Conf Europ](#)
- [awesome-remix](#)
- [React-Router 6.4](#)
- [Learn Remix](#)