DECOUPLED DAYS 2021

presented by srijan:

Virtual Edition
July 14–15, 2021

DIAMOND SPONSORS

therefore oomph Centarro platform.sh contentful

Learning Next.js by Converting a WordPress Blog

Thomas Desmond

Today's speaker

SITECORE®

Thomas Desmond

Javascript Technical Evangelist, Sitecore

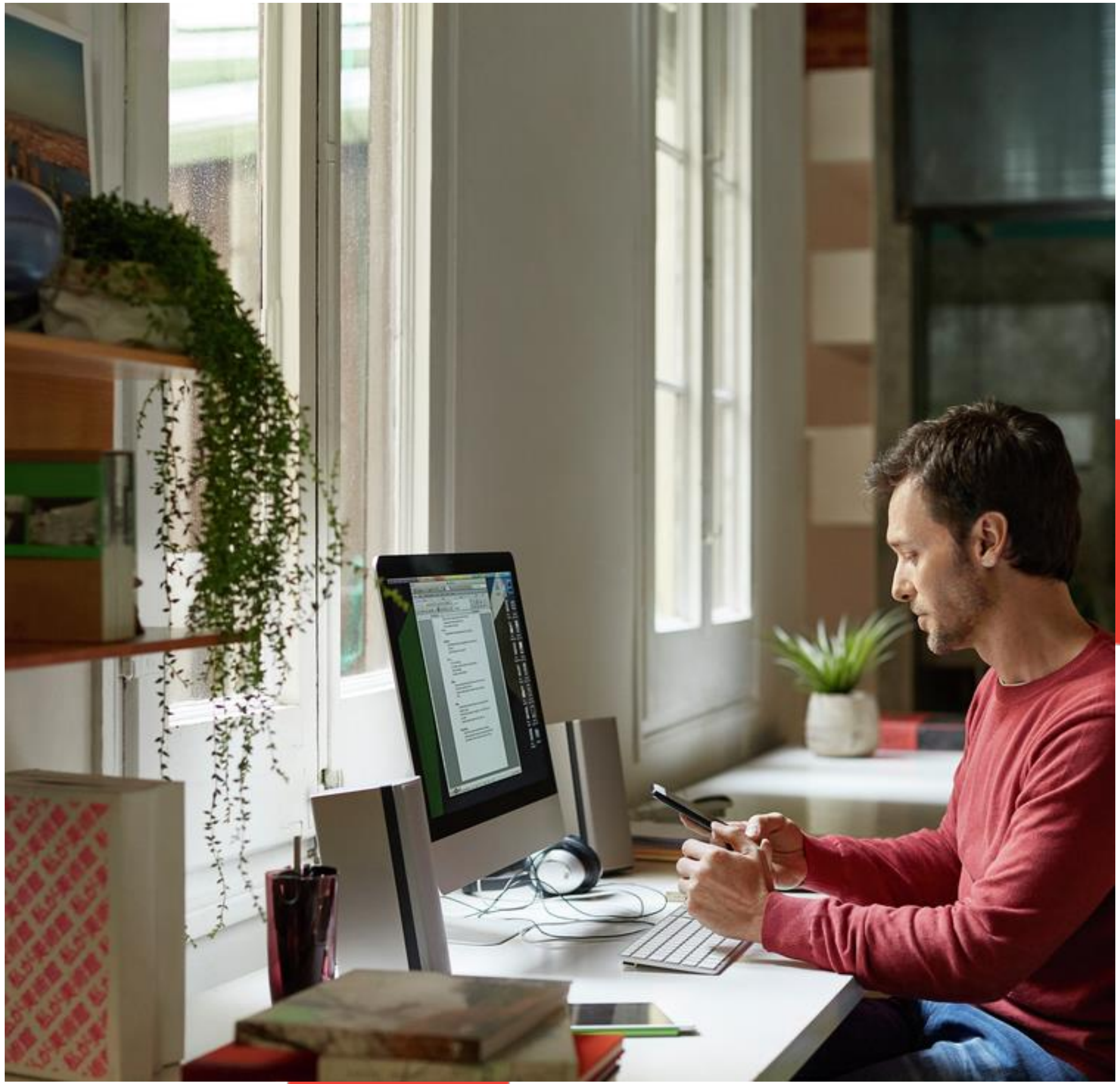
- Long time developer
- Focus on frontend advocacy
- Located in San Diego, California

@ThomasJDesmond

www.TheTomBomb.com

SITECORE®

Learning Next.js by Converting a WordPress Blog



Presented by: Thomas Desmond

Date: July 14, 2021

SITECORE®

Agenda

01 The original blog

02 Motivations for change

03 Why Next.js

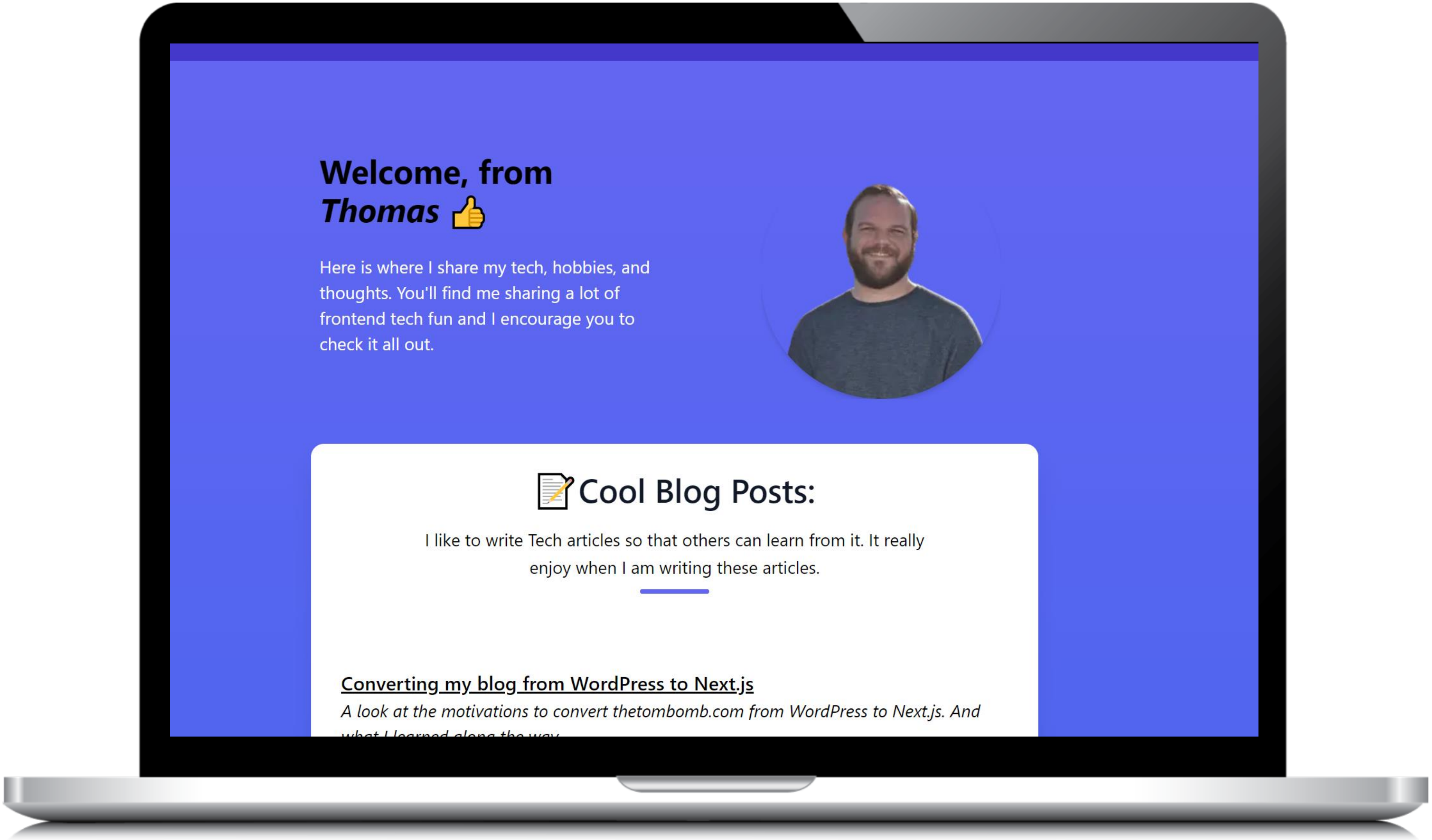
04 Jamstack Architecture

05 Next.js Code

06 Was it worth it?

07 Learn More

TheTomBomb.com in 2021



Welcome, from *Thomas* 👍

Here is where I share my tech, hobbies, and thoughts. You'll find me sharing a lot of frontend tech fun and I encourage you to check it all out.

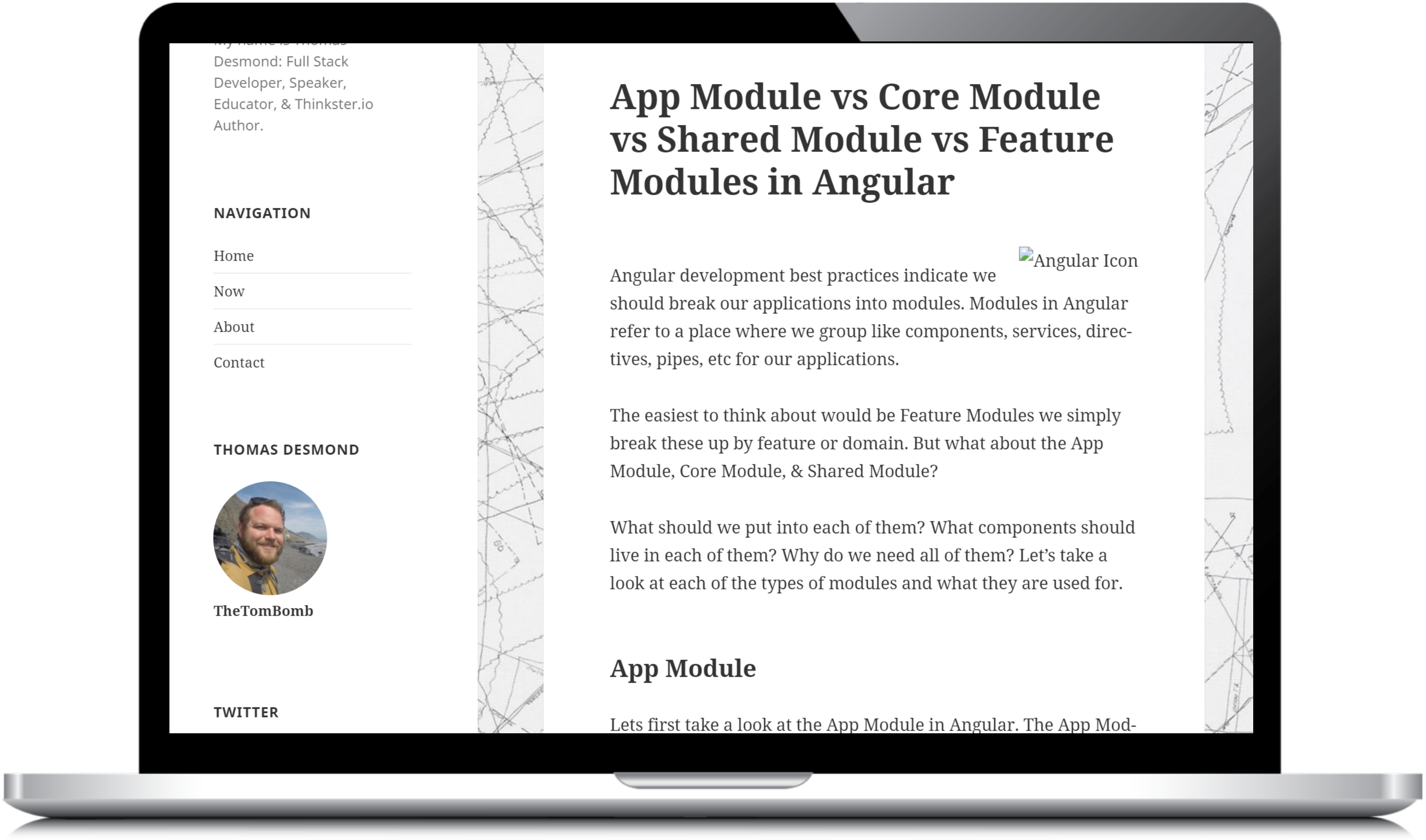
📝Cool Blog Posts:

I like to write Tech articles so that others can learn from it. It really enjoy when I am writing these articles.

Converting my blog from WordPress to Next.js

A look at the motivations to convert thetombomb.com from WordPress to Next.js. And what I learned along the way.

It all started in 2016



My name is Thomas Desmond: Full Stack Developer, Speaker, Educator, & Thinkster.io Author.

NAVIGATION

- Home
- Now
- About
- Contact

THOMAS DESMOND

TheTomBomb

TWITTER

App Module vs Core Module vs Shared Module vs Feature Modules in Angular

Angular Icon

Angular development best practices indicate we should break our applications into modules. Modules in Angular refer to a place where we group like components, services, directives, pipes, etc for our applications.

The easiest to think about would be Feature Modules we simply break these up by feature or domain. But what about the App Module, Core Module, & Shared Module?

What should we put into each of them? What components should live in each of them? Why do we need all of them? Let's take a look at each of the types of modules and what they are used for.

App Module

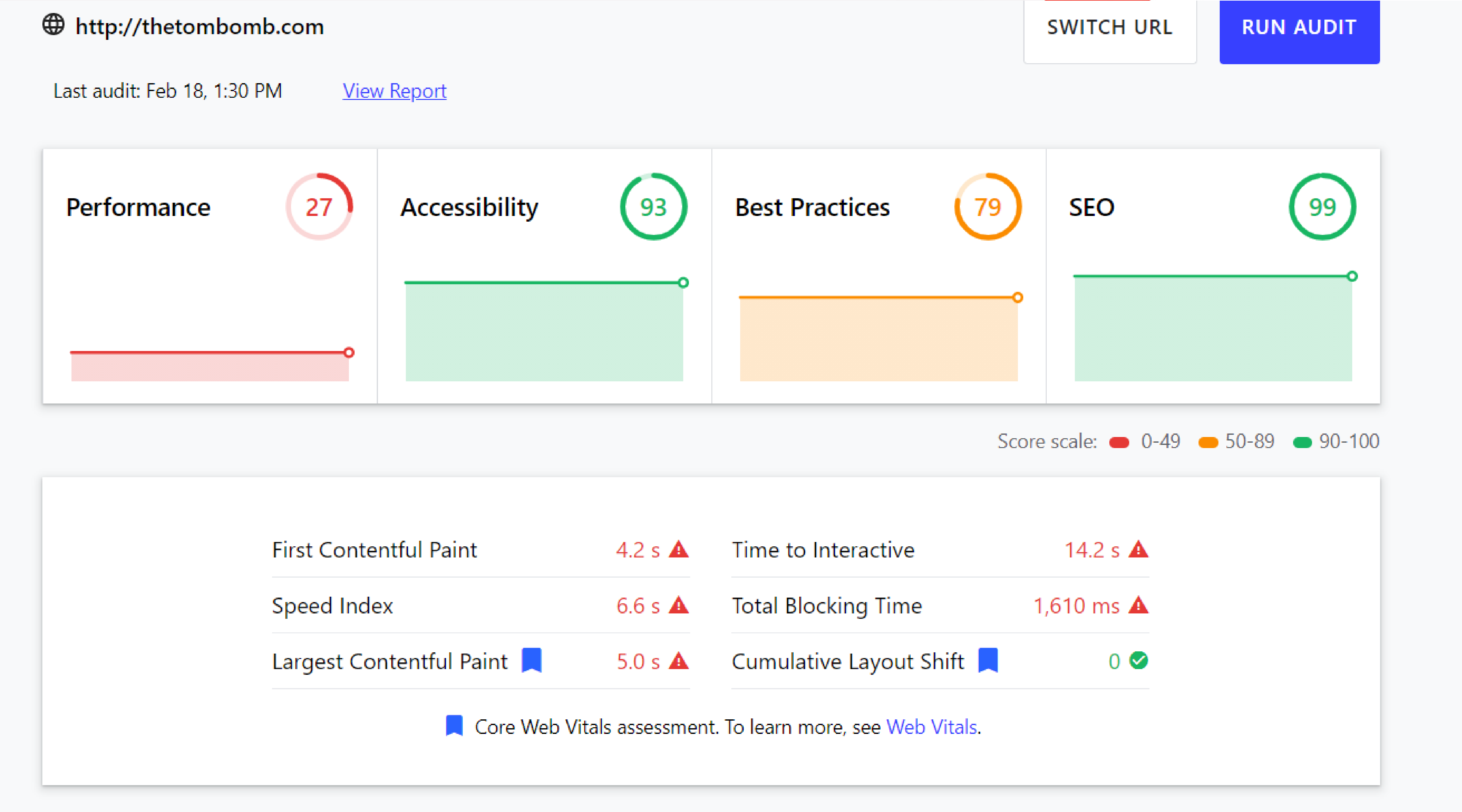
Lets first take a look at the App Module in Angular. The App Mod-

Motivation for change?

Performance!

WordPress Lighthouse Scores ☹

SITECORE®



http://thetombomb.com

Last audit: Feb 18, 1:30 PM View Report

Performance 27

Accessibility 93

Best Practices 79

SEO 99

Score scale: 🟥 0-49 🟧 50-89 🟩 90-100

First Contentful Paint	4.2 s ⚠	Time to Interactive	14.2 s ⚠
Speed Index	6.6 s ⚠	Total Blocking Time	1,610 ms ⚠
Largest Contentful Paint 🔖	5.0 s ⚠	Cumulative Layout Shift 🔖	0 ✓

🔖 Core Web Vitals assessment. To learn more, see Web Vitals.

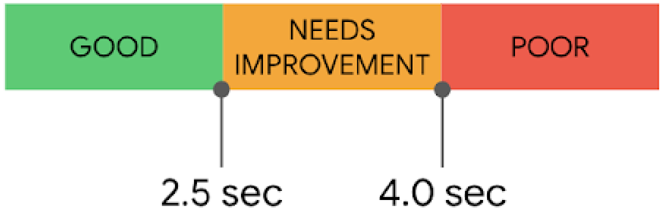
Google Core Web Vitals

SITECORE®

(loading)

LCP

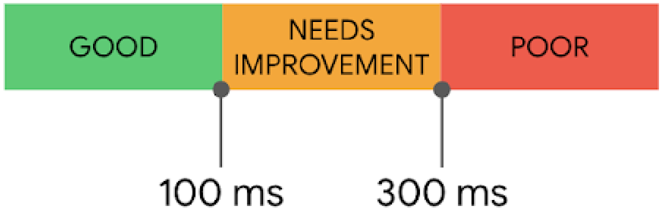
Largest Contentful Paint



(interactivity)

FID

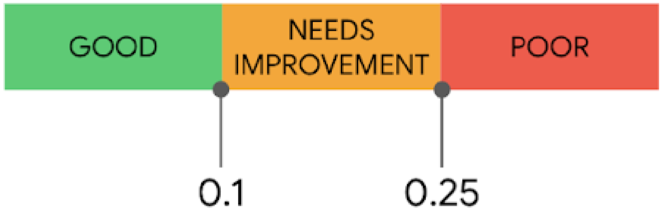
First Input Delay

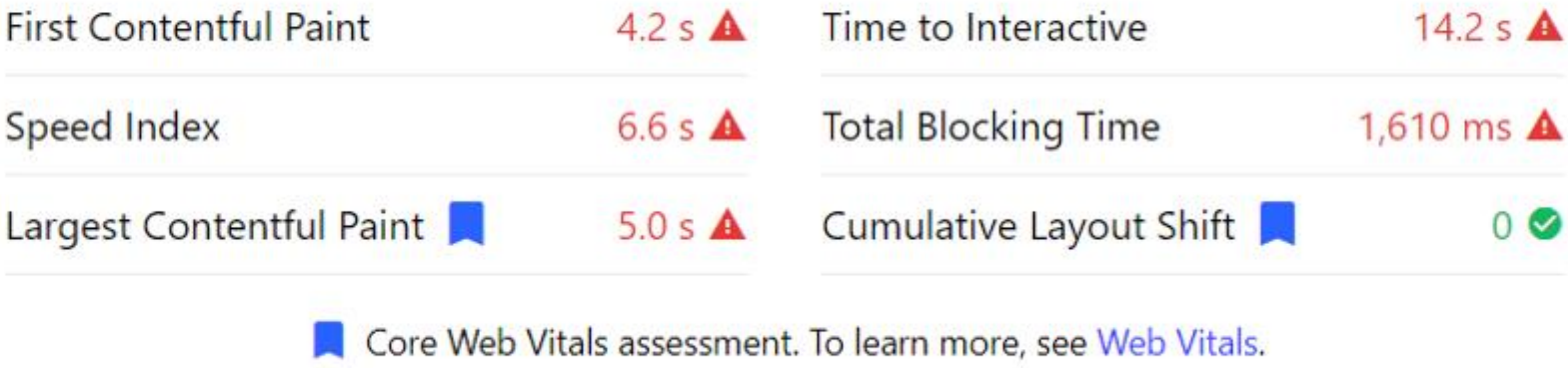


(visual stability)

CLS

Cumulative Layout Shift

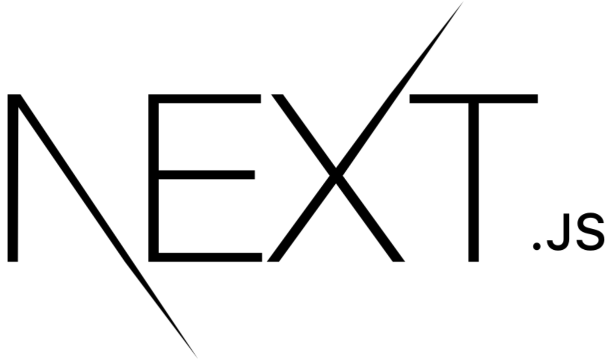




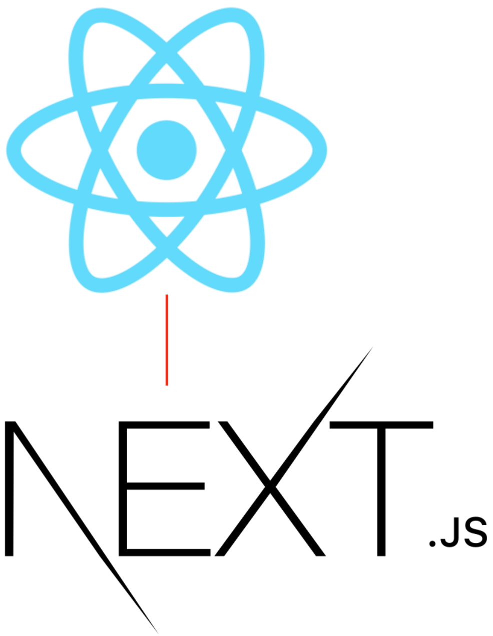
First Contentful Paint	4.2 s ⚠	Time to Interactive	14.2 s ⚠
Speed Index	6.6 s ⚠	Total Blocking Time	1,610 ms ⚠
Largest Contentful Paint 🔖	5.0 s ⚠	Cumulative Layout Shift 🔖	0 ✓

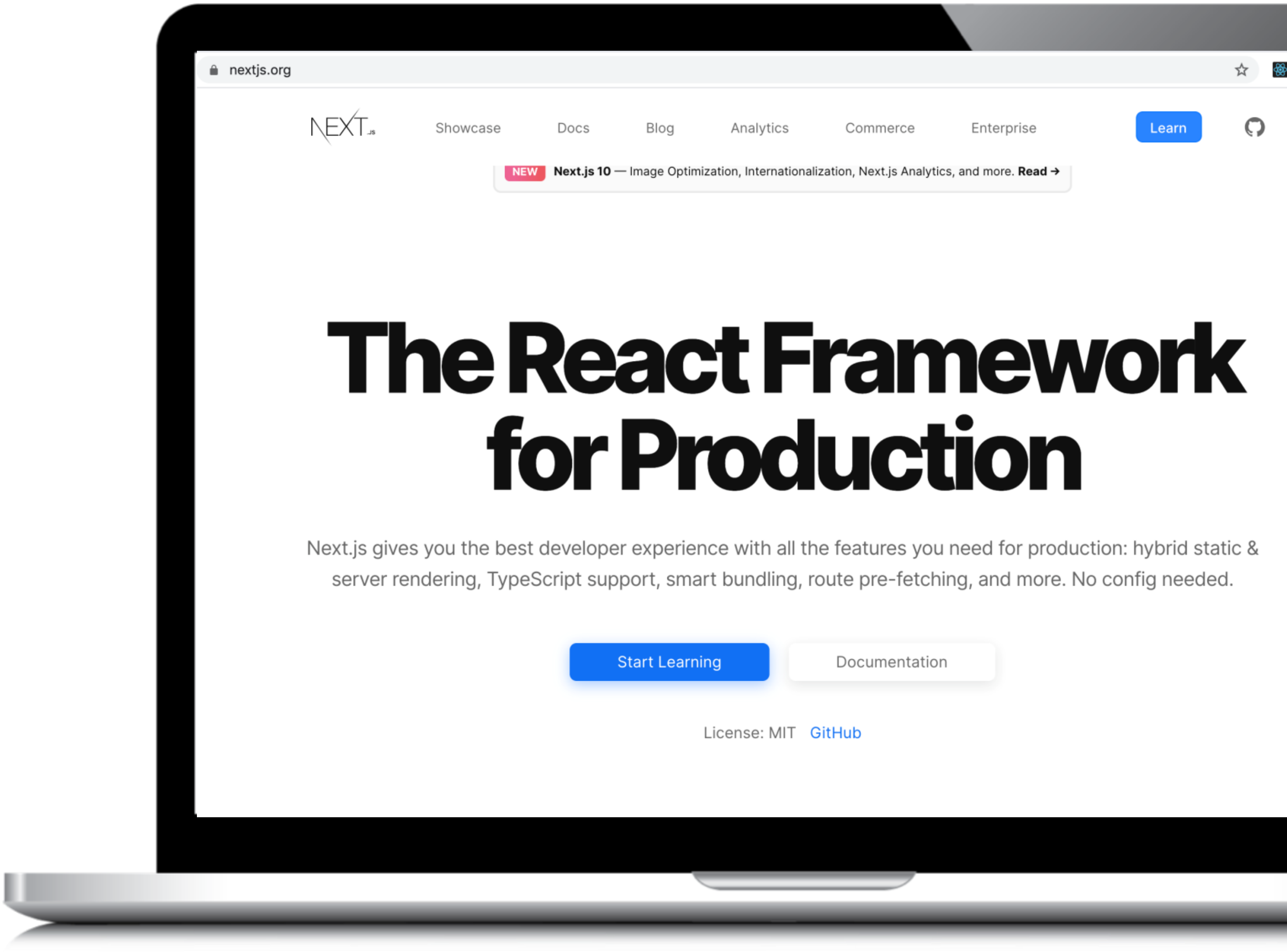
🔖 Core Web Vitals assessment. To learn more, see Web Vitals.

Looking at all the options



What is Next.js





NEW **Next.js 10** — Image Optimization, Internationalization, Next.js Analytics, and more. **Read →**

The React Framework for Production

Next.js gives you the best developer experience with all the features you need for production: hybrid static & server rendering, TypeScript support, smart bundling, route pre-fetching, and more. No config needed.

License: MIT GitHub

Jamstack

JAMSTACK

“Jamstack” is **not a new tech stack.**

JavaScript

APIs

Markup

It’s a high-level classification of an **architectural approach.**

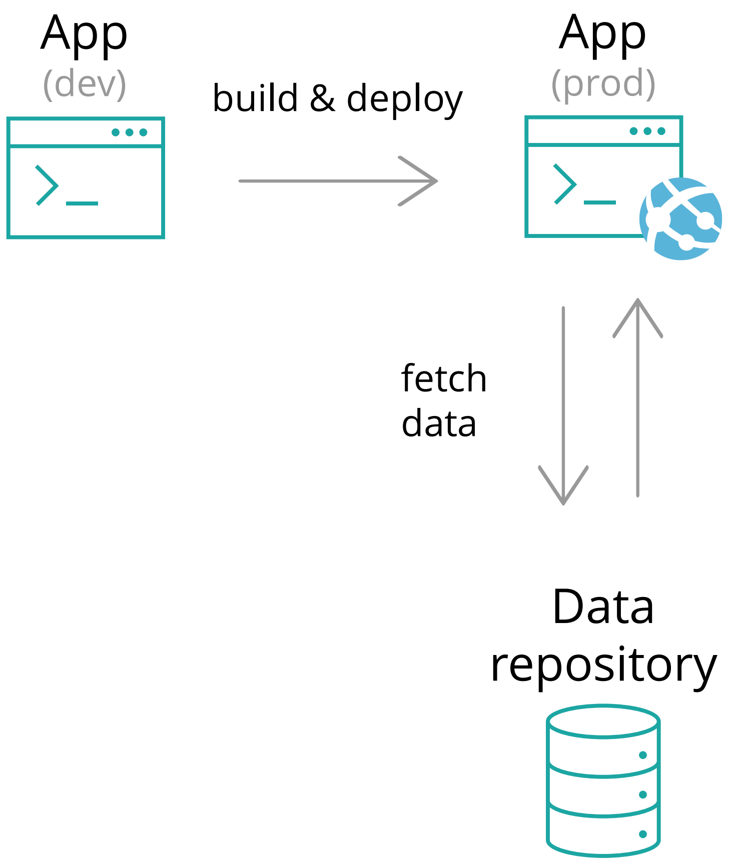
Jamstack anchors

Jamstack anchors:

- Pre-rendering
- Fast delivery
- Rehydration

Jamstack architecture

Build & Deploy Time



App
(dev)

build & deploy

App
(prod)

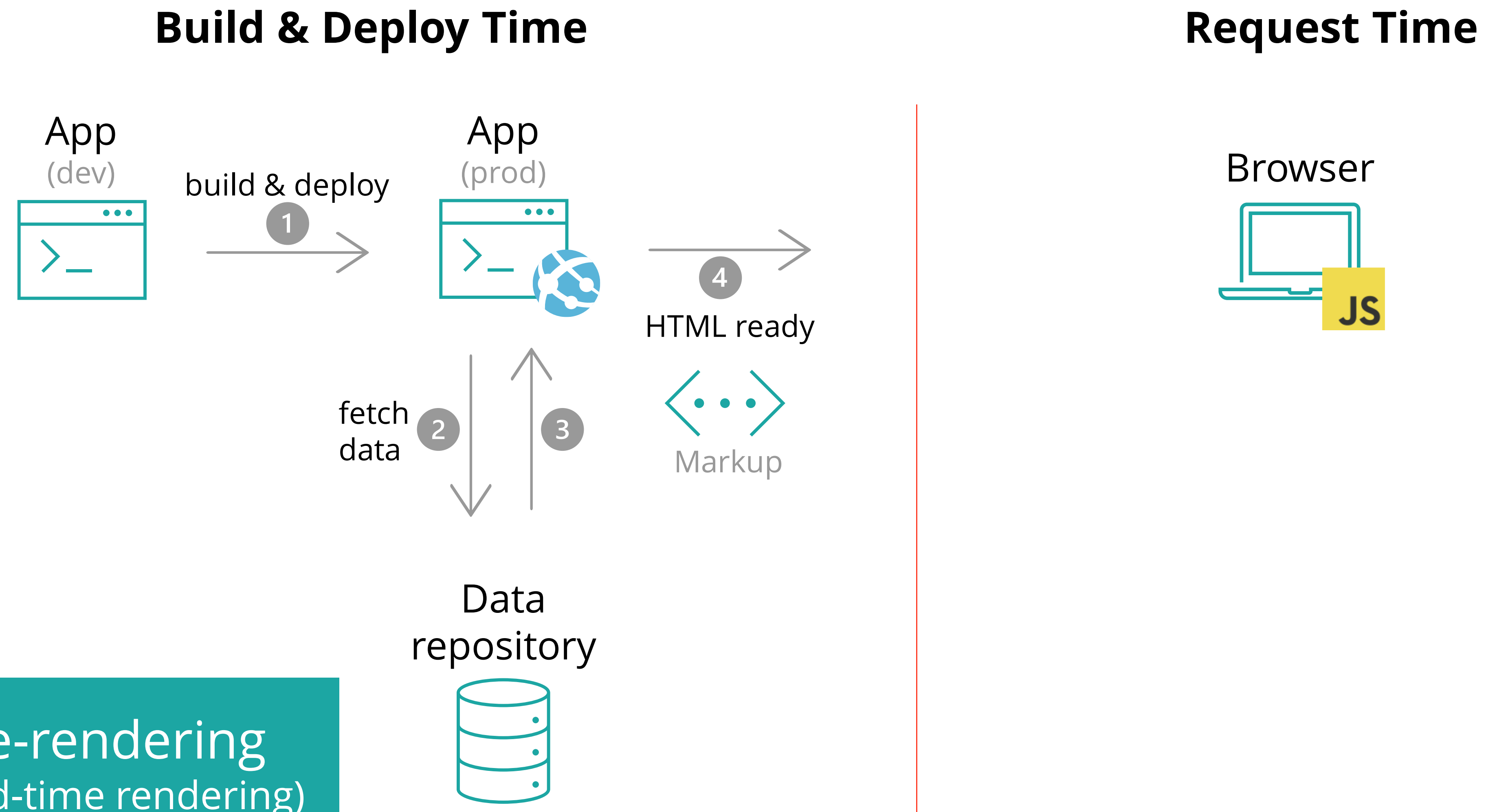
fetch
data

Data
repository

Request Time

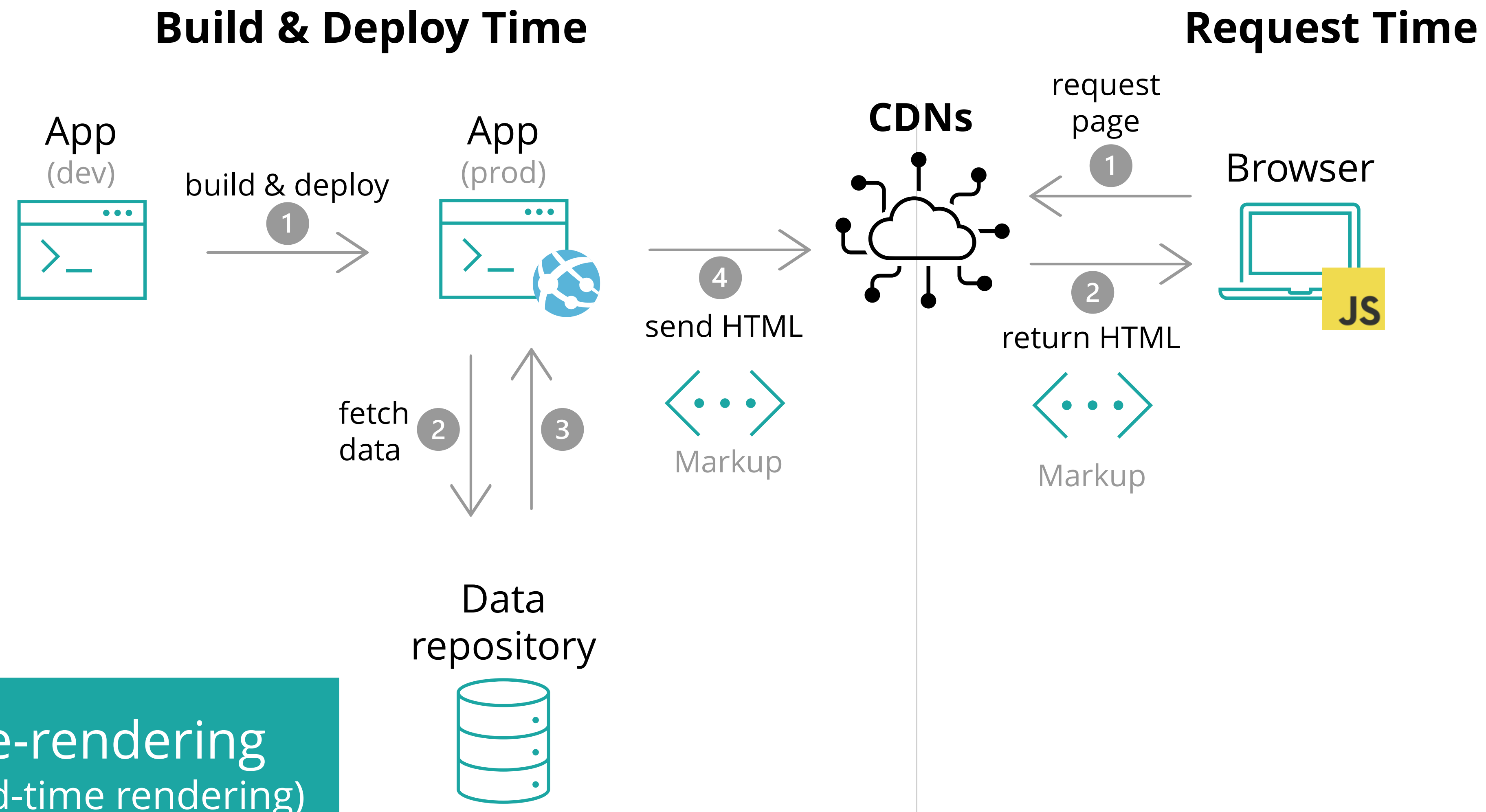
Browser

Jamstack architecture



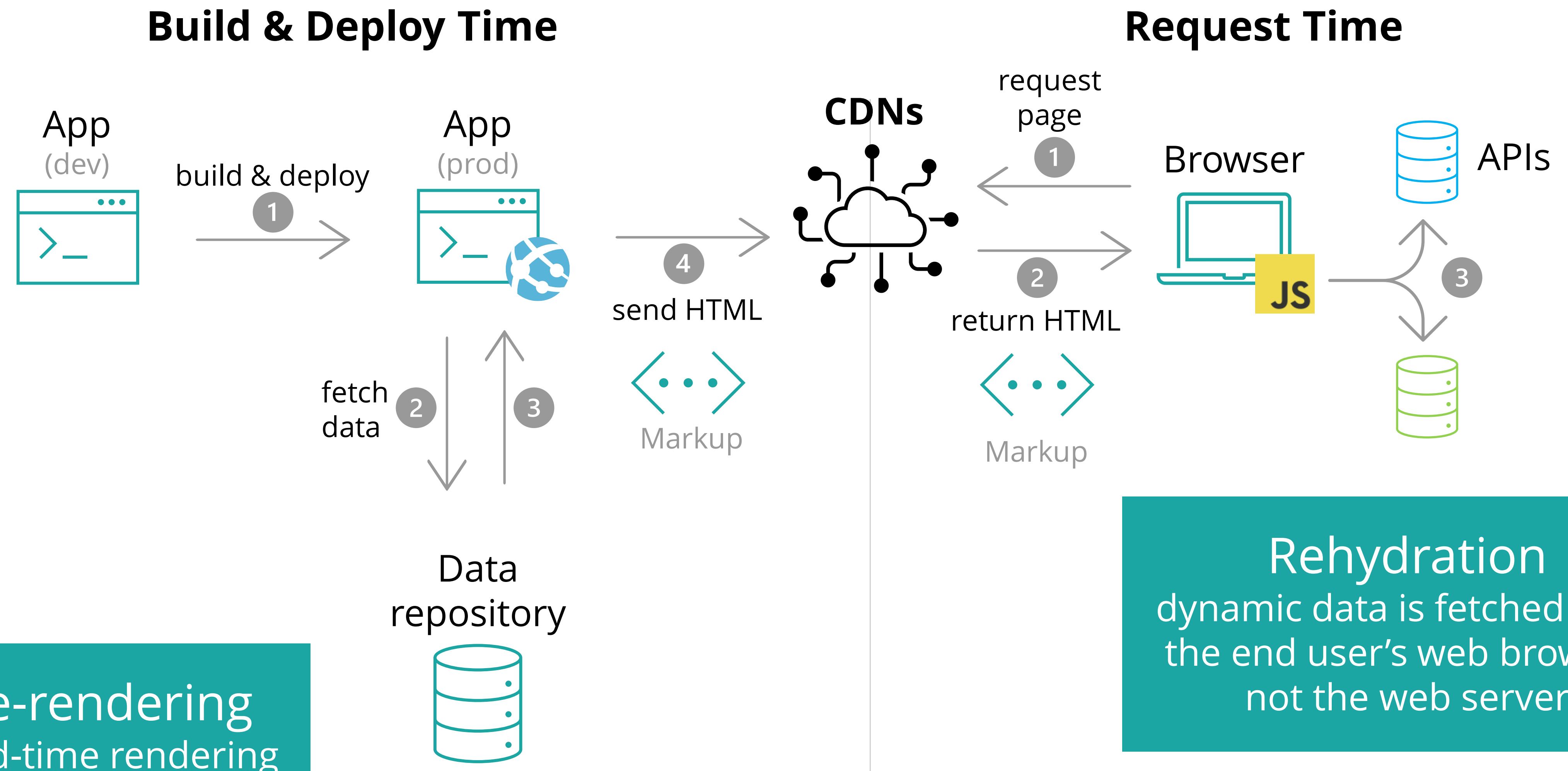
Pre-rendering
(build-time rendering)

Jamstack architecture



Pre-rendering
(build-time rendering)

Jamstack architecture



Pre-rendering
build-time rendering

Rehydration
dynamic data is fetched from the end user's web browser, not the web server

Pre-rendering in Next.js

Server-Side Rendering

getServerSideProps

Fetch data on each request

Static Generation

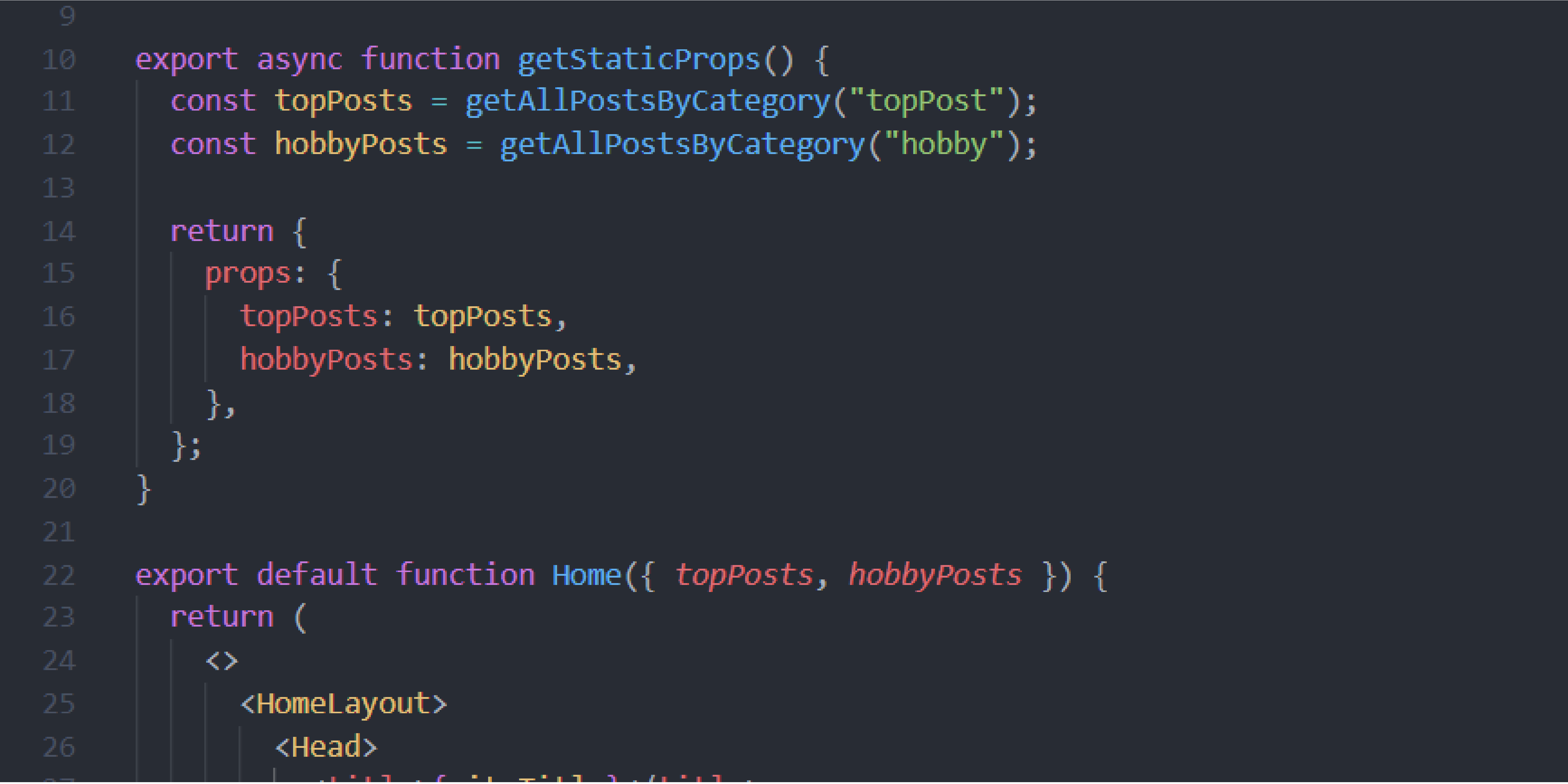
getStaticProps

Fetch data at build time

getStaticPaths

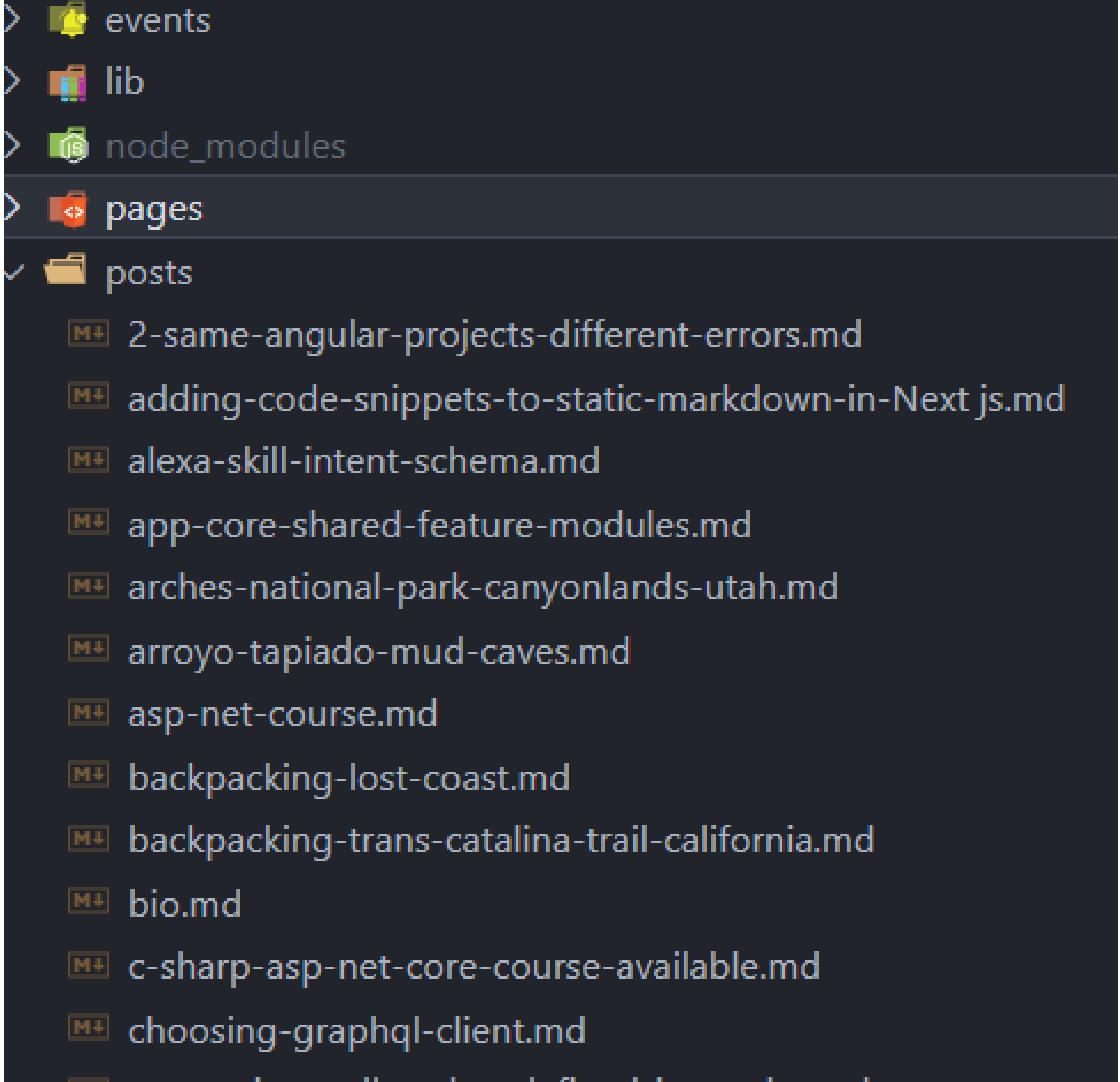
Pre-render dynamic routes based on data

getStaticProps() in index.js



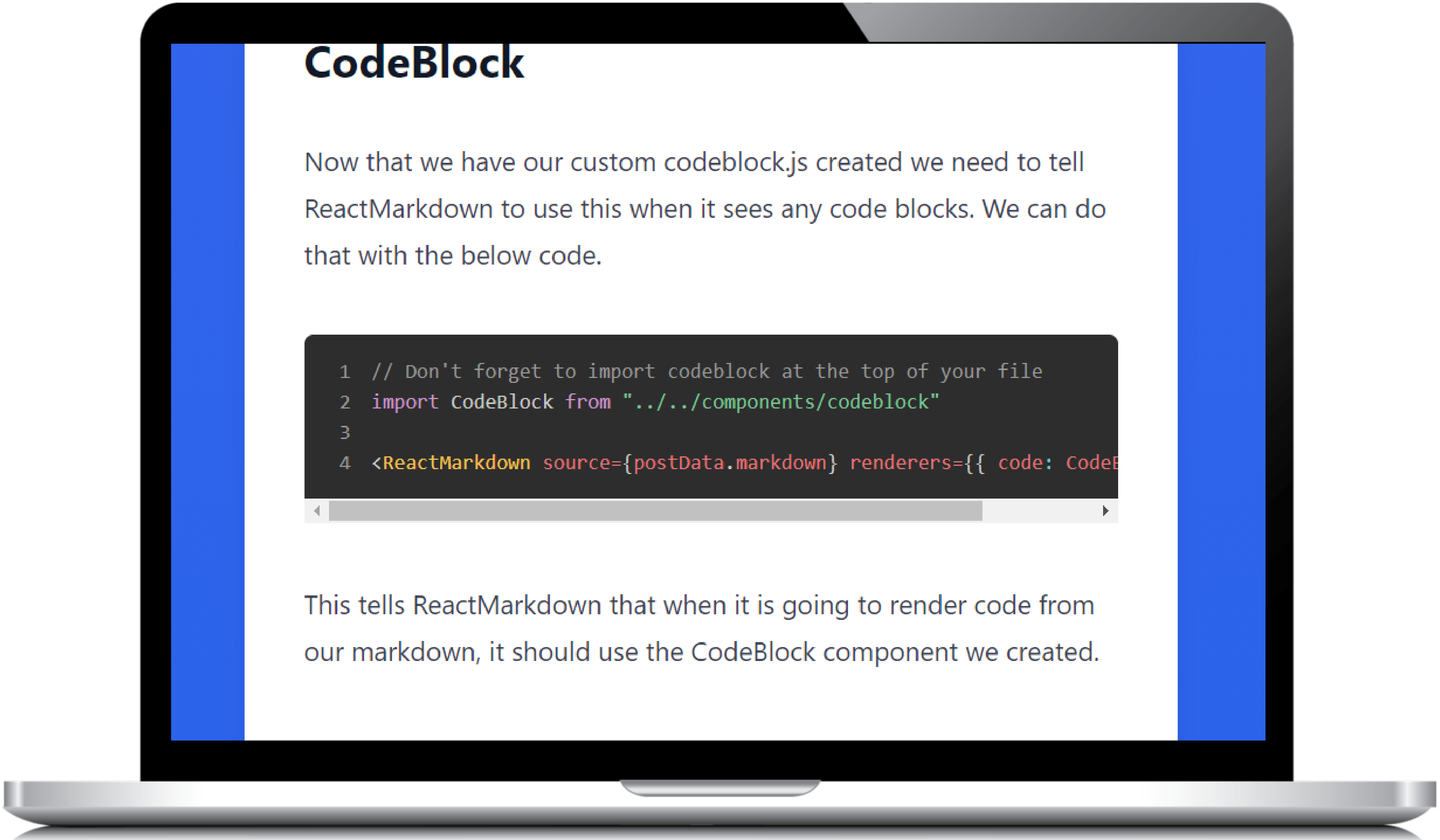
```
export async function getStaticProps() {
  const topPosts = getAllPostsByCategory("topPost");
  const hobbyPosts = getAllPostsByCategory("hobby");

  return {
    props: {
      topPosts: topPosts,
      hobbyPosts: hobbyPosts,
    },
  };
}

export default function Home({ topPosts, hobbyPosts }) {
  return (
    <>
      <HomeLayout>
        <Head>
```

Local Directory of Posts



```
> events
> lib
> node_modules
> pages
v posts
    2-same-angular-projects-different-errors.md
    adding-code-snippets-to-static-markdown-in-Next js.md
    alexa-skill-intent-schema.md
    app-core-shared-feature-modules.md
    arches-national-park-canyonlands-utah.md
    arroyo-tapiado-mud-caves.md
    asp-net-course.md
    backpacking-lost-coast.md
    backpacking-trans-catalina-trail-california.md
    bio.md
    c-sharp-asp-net-core-course-available.md
    choosing-graphql-client.md
```

Markdown to HTML



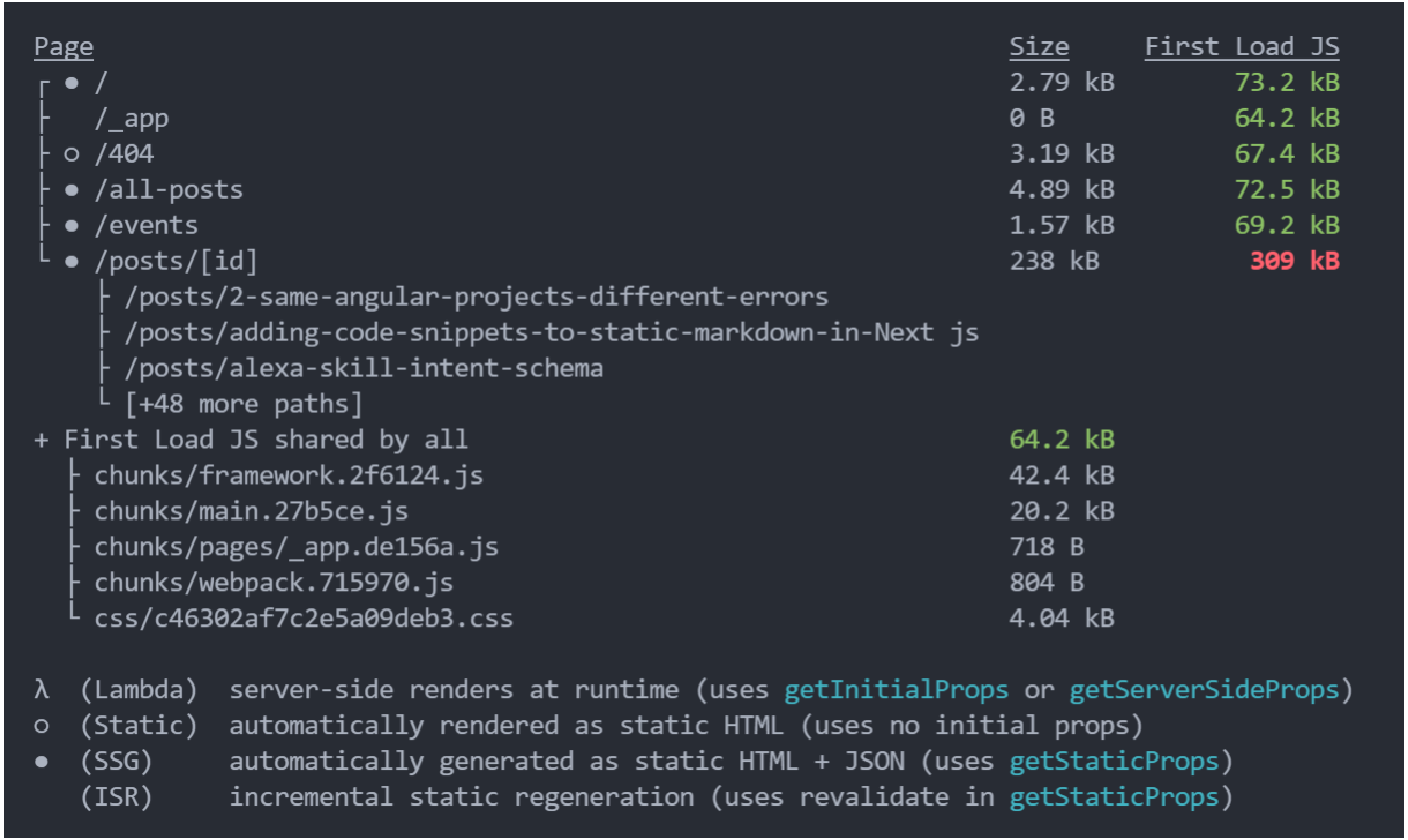
CodeBlock

Now that we have our custom codeblock.js created we need to tell ReactMarkdown to use this when it sees any code blocks. We can do that with the below code.

```
// Don't forget to import codeblock at the top of your file
import CodeBlock from "../../components/codeblock"

<ReactMarkdown source={postData.markdown} renderers={{ code: CodeB
```

This tells ReactMarkdown that when it is going to render code from our markdown, it should use the CodeBlock component we created.

‘next build’ command

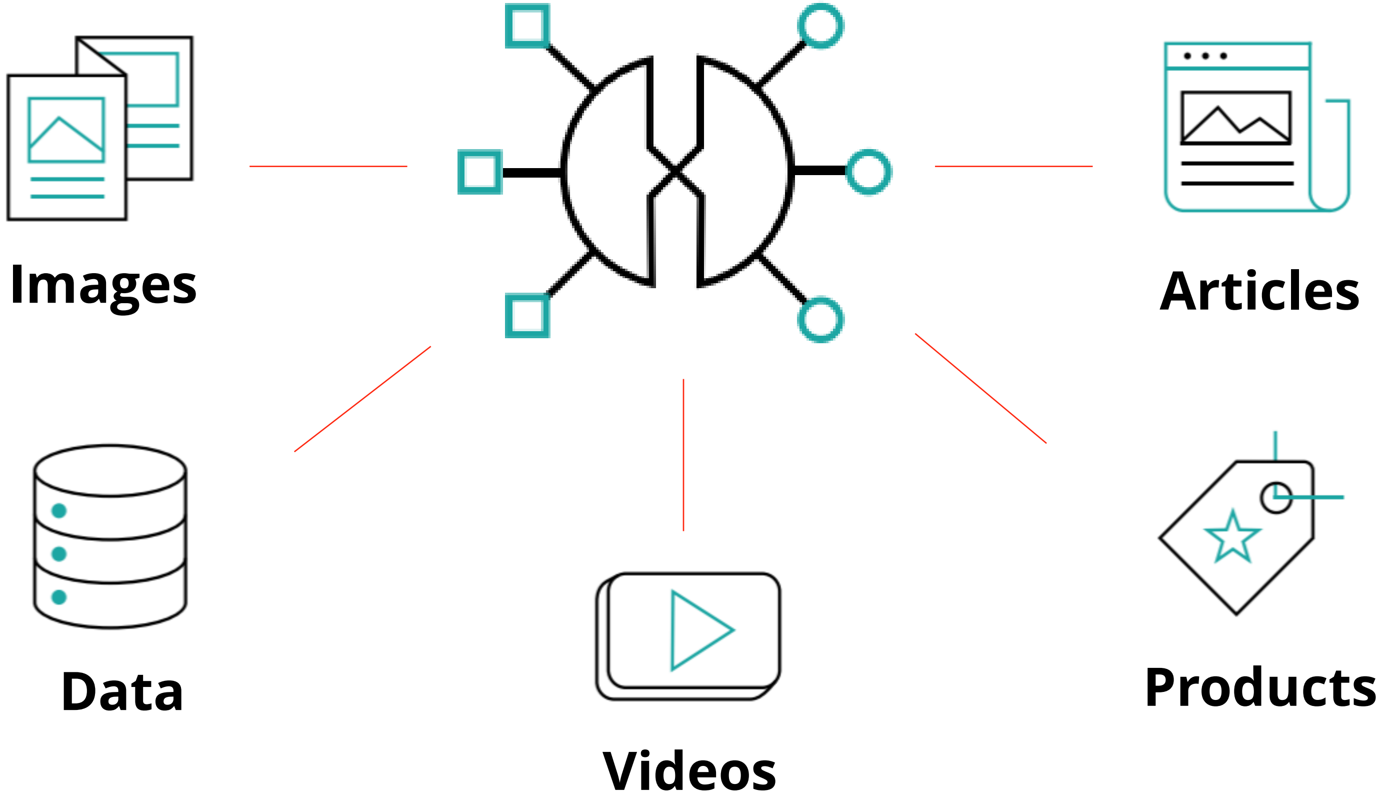


```
Page                                                              Size     First Load JS
┌ ● /                                                             2.79 kB        73.2 kB
├   /_app                                                         0 B            64.2 kB
├ ○ /404                                                          3.19 kB        67.4 kB
├ ● /all-posts                                                    4.89 kB        72.5 kB
├ ● /events                                                       1.57 kB        69.2 kB
└ ● /posts/[id]                                                   238 kB          309 kB
    ├ /posts/2-same-angular-projects-different-errors
    ├ /posts/adding-code-snippets-to-static-markdown-in-Next js
    ├ /posts/alexa-skill-intent-schema
    └ [+48 more paths]
+ First Load JS shared by all                                     64.2 kB
  ├ chunks/framework.2f6124.js                                    42.4 kB
  ├ chunks/main.27b5ce.js                                         20.2 kB
  ├ chunks/pages/_app.de156a.js                                   718 B
  ├ chunks/webpack.715970.js                                      804 B
  └ css/c46302af7c2e5a09deb3.css                                  4.04 kB

λ  (Lambda)  server-side renders at runtime (uses getInitialProps or getServerSideProps)
○  (Static)  automatically rendered as static HTML (uses no initial props)
●  (SSG)     automatically generated as static HTML + JSON (uses getStaticProps)
   (ISR)     incremental static regeneration (uses revalidate in getStaticProps)
```

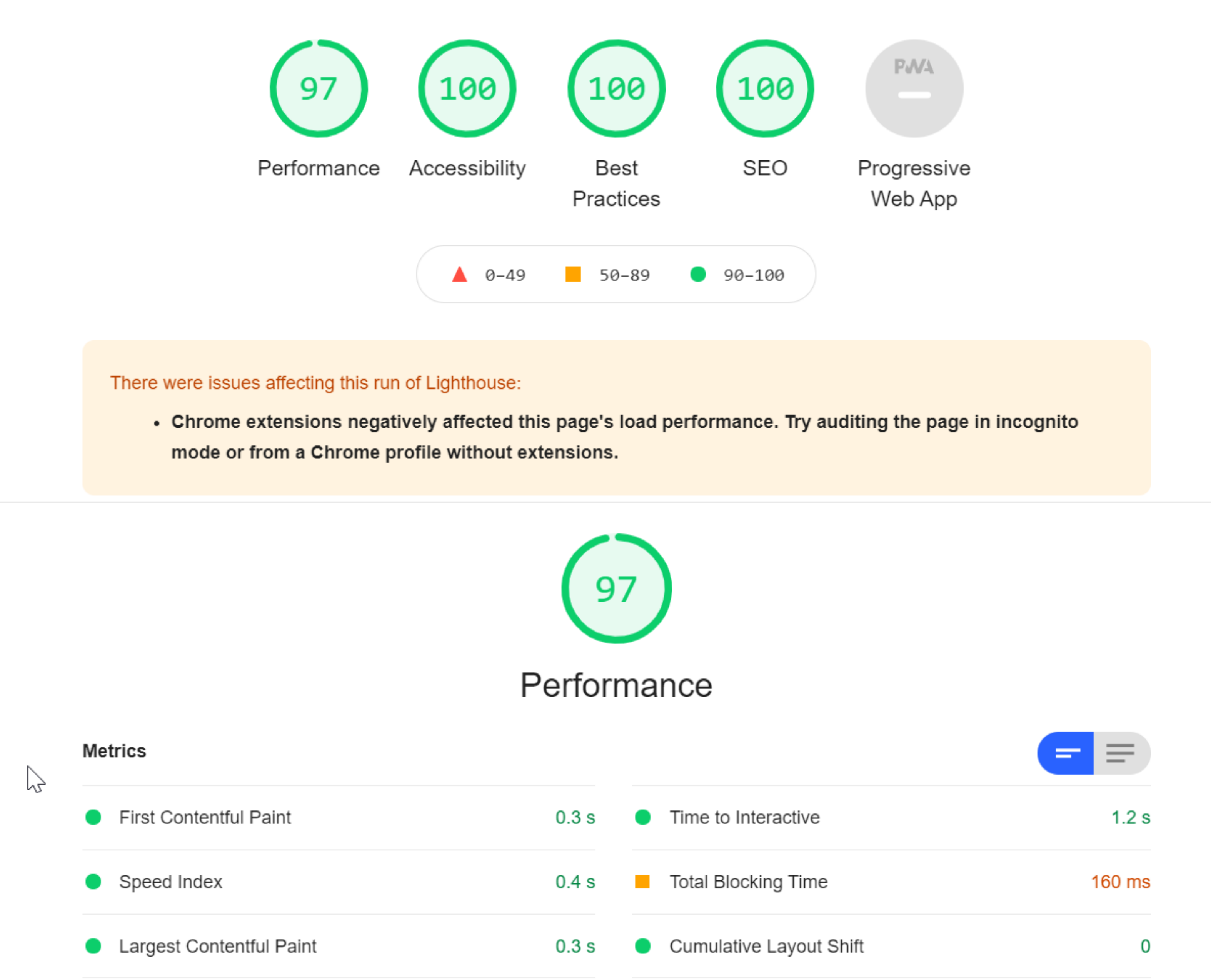
Powerful CMS for Enterprise

SITECORE

Content Management System



Was it worth it?



97 Performance

100 Accessibility

100 Best Practices

100 SEO

PWA — Progressive Web App

▲ 0–49 ■ 50–89 ● 90–100

There were issues affecting this run of Lighthouse:

- **Chrome extensions negatively affected this page's load performance. Try auditing the page in incognito mode or from a Chrome profile without extensions.**

97

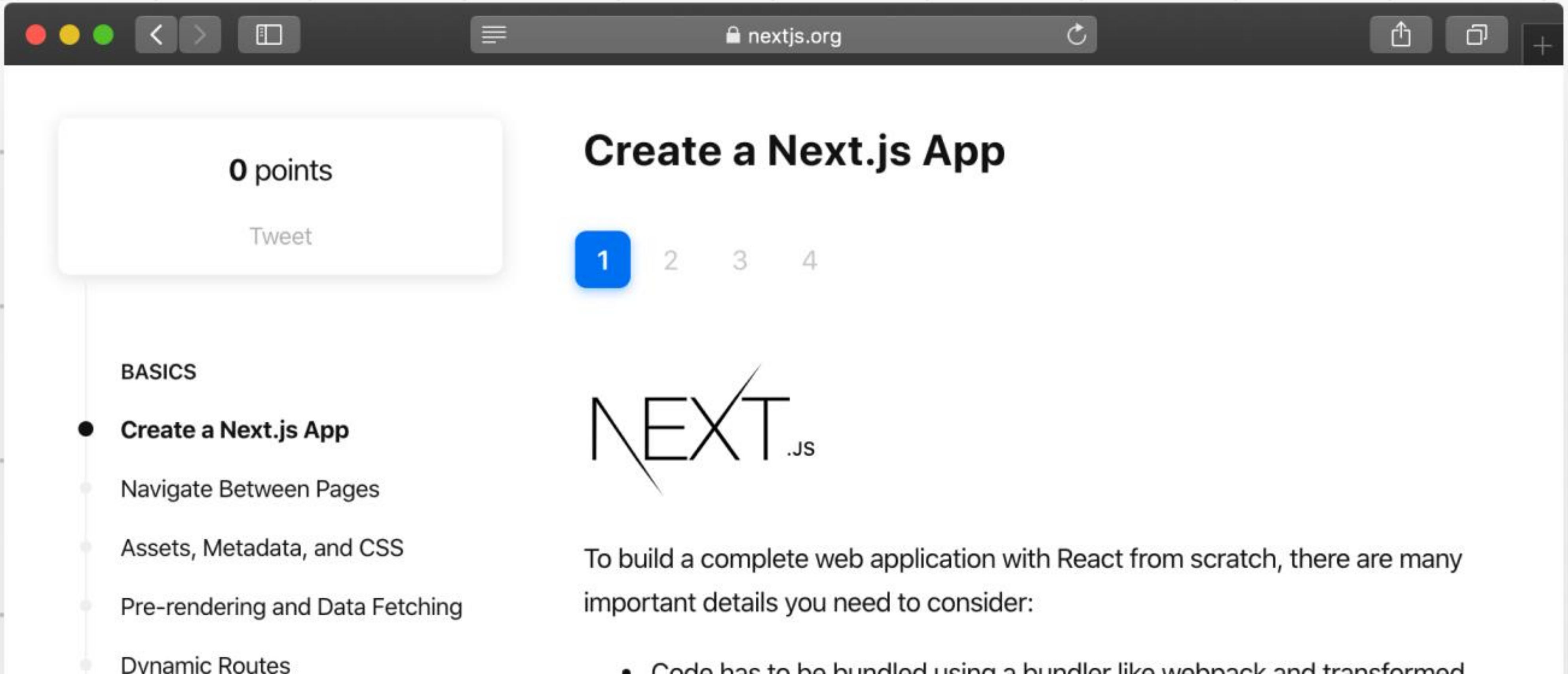
Performance

Metrics

●	First Contentful Paint	0.3 s	●	Time to Interactive	1.2 s
●	Speed Index	0.4 s	■	Total Blocking Time	160 ms
●	Largest Contentful Paint	0.3 s	●	Cumulative Layout Shift	0

Learn Next.js

nextjs.org/learn



0 points

Tweet

BASICS

- **Create a Next.js App**
- Navigate Between Pages
- Assets, Metadata, and CSS
- Pre-rendering and Data Fetching
- Dynamic Routes

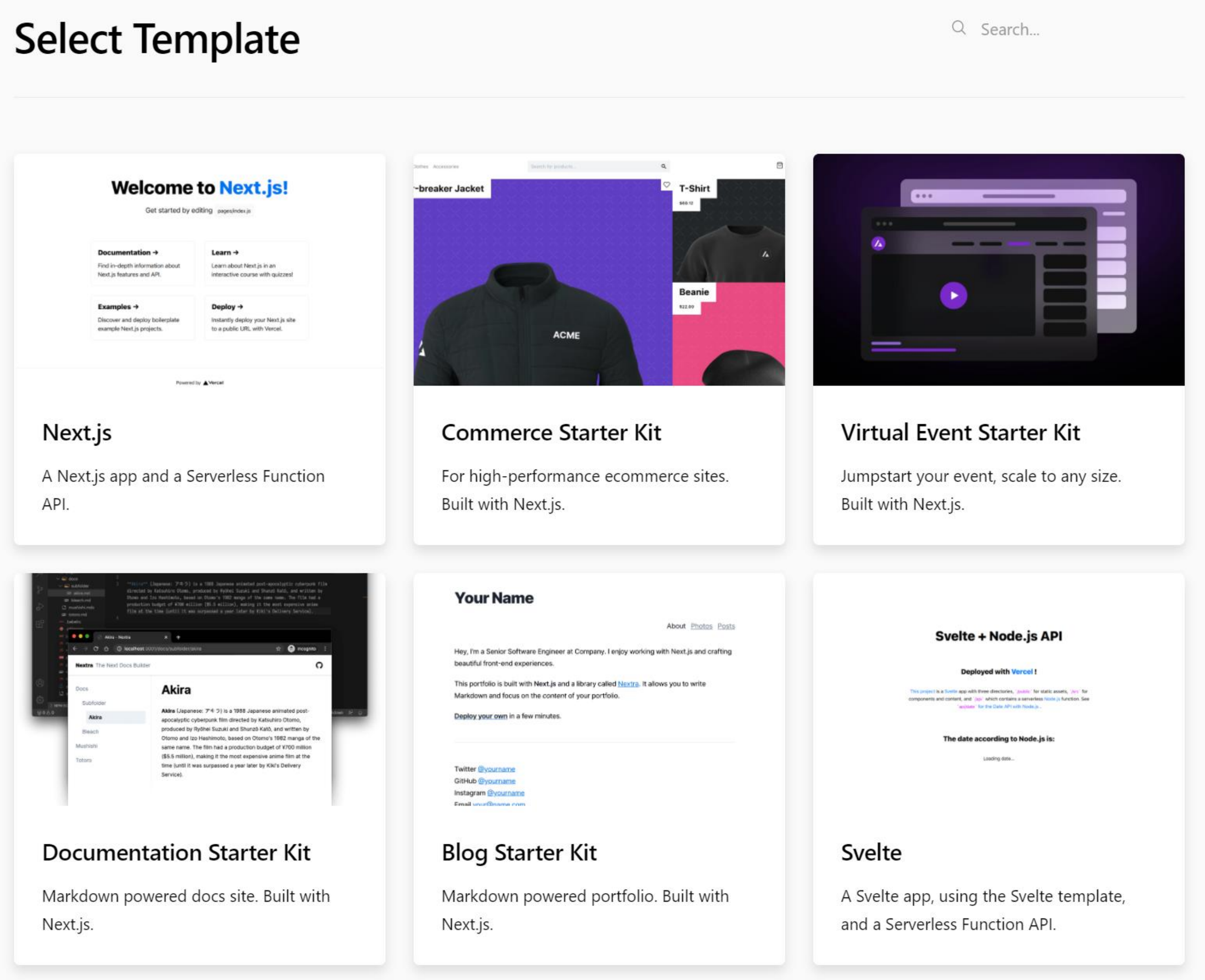
Create a Next.js App

To build a complete web application with React from scratch, there are many important details you need to consider:

- Code has to be bundled using a bundler like webpack and transformed

SITECORE

Next.js templates and more



Select Template

Next.js

A Next.js app and a Serverless Function API.

Commerce Starter Kit

For high-performance ecommerce sites. Built with Next.js.

Virtual Event Starter Kit

Jumpstart your event, scale to any size. Built with Next.js.

Documentation Starter Kit

Markdown powered docs site. Built with Next.js.

Blog Starter Kit

Markdown powered portfolio. Built with Next.js.

Svelte

A Svelte app, using the Svelte template, and a Serverless Function API.

Review

01 The original blog

02 Motivations for change

03 Why Next.js

04 Jamstack Architecture

05 Next.js Code

06 Was it worth it?

07 Where to Learn More

Today's speaker

SITECORE®

Thomas Desmond

Javascript Technical Evangelist, Sitecore

- Long time developer
- Focus on frontend advocacy
- Located in San Diego, California

@ThomasJDesmond

www.thetombomb.com

DECOUPLED
DAYS
2021

presented by srijan:

Virtual Edition
July 14–15, 2021

DIAMOND SPONSORS

therefore oomph Centarro platform.sh contentful

Thank You!

Thomas Desmond

#DecoupledDays DecoupledDays.com