



# Learning Next.js by Converting a WordPress Blog

A mission for all out performance!

**Presented by: Thomas Desmond**





# Thomas Desmond

Developer Advocate @ Sitecore

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

[@ThomasJDesmond](#)

[www.TheTomBomb.com](http://www.TheTomBomb.com)



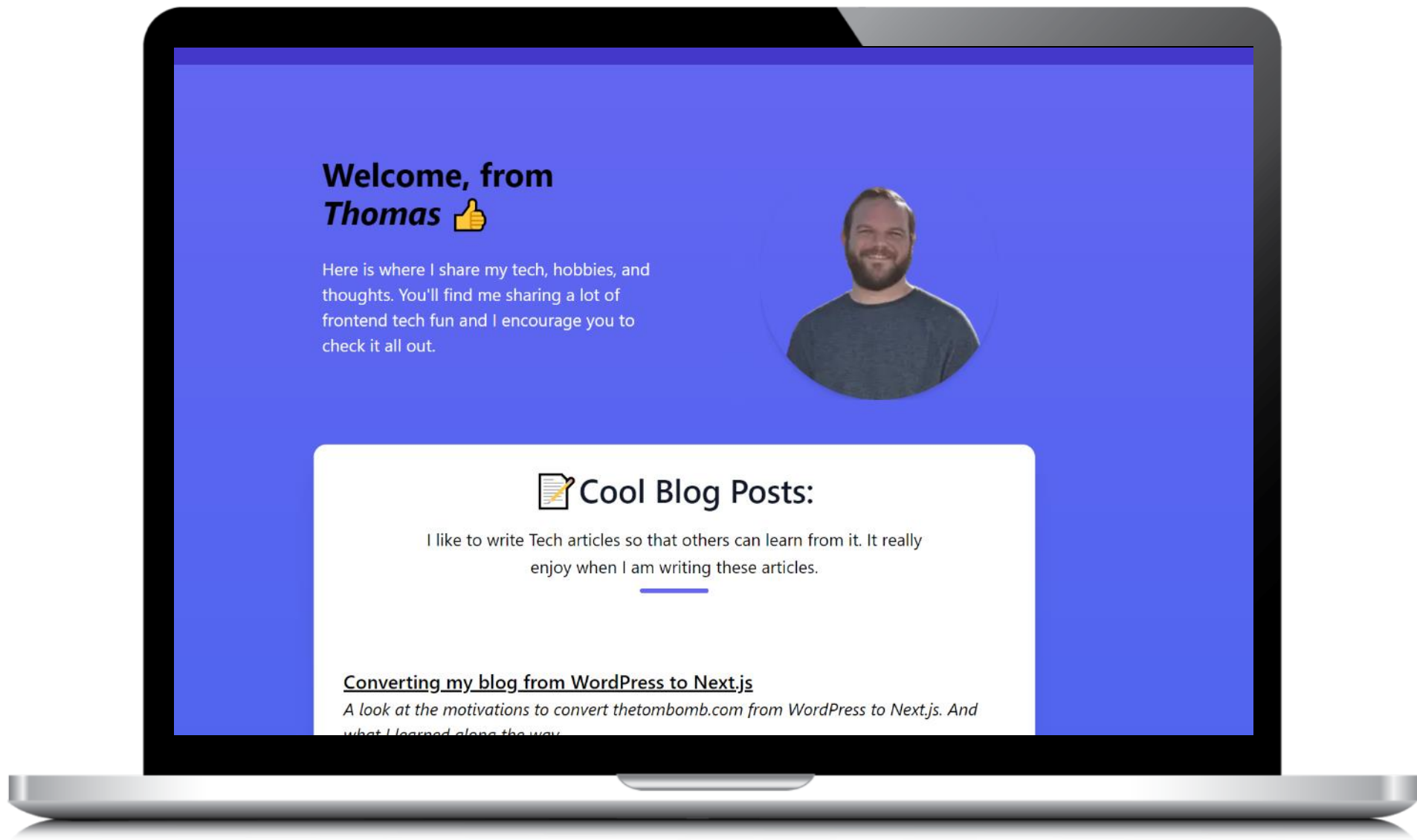


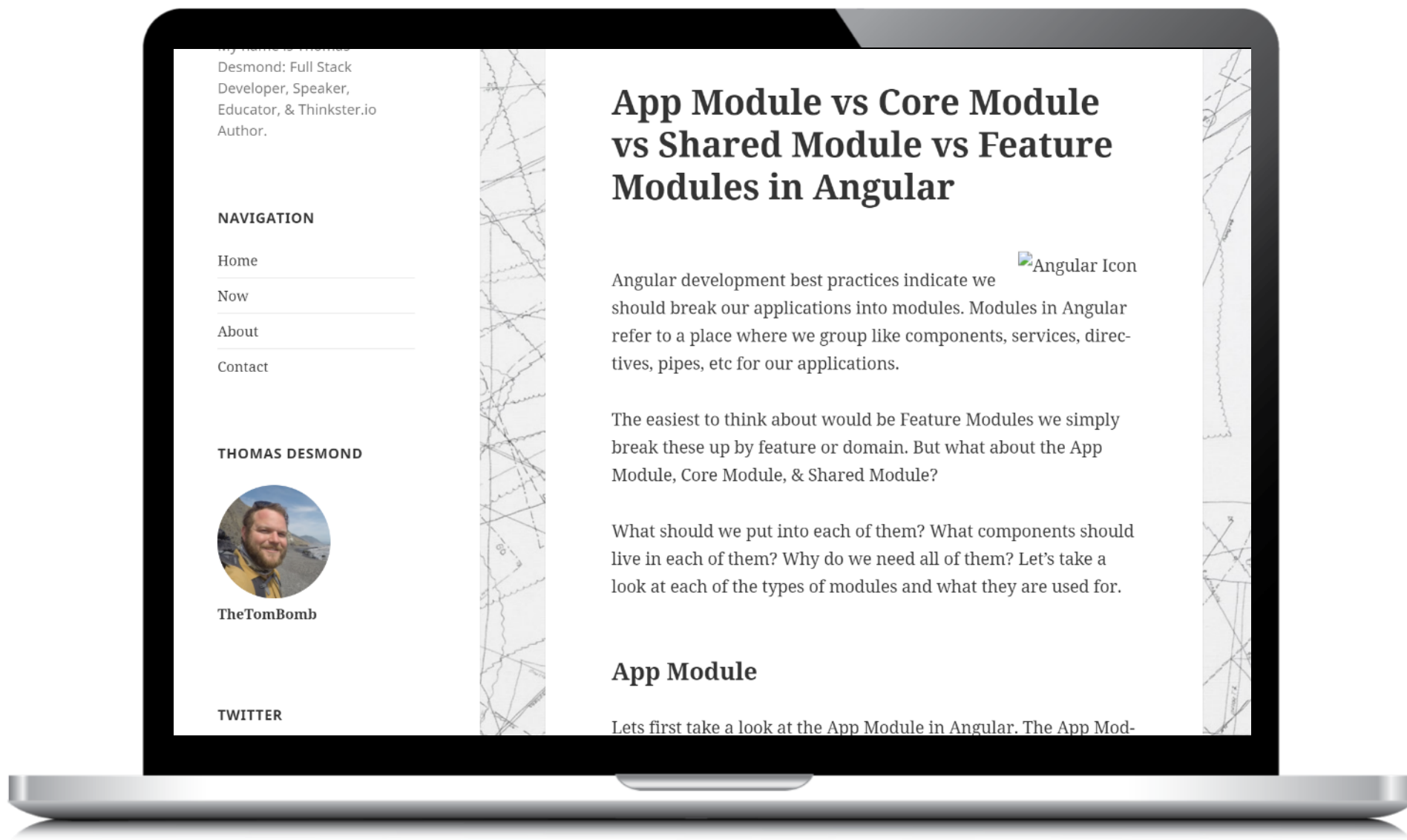
# Learning Next.js by Converting a WordPress Blog

A mission for all out performance!

**Presented by: Thomas Desmond**







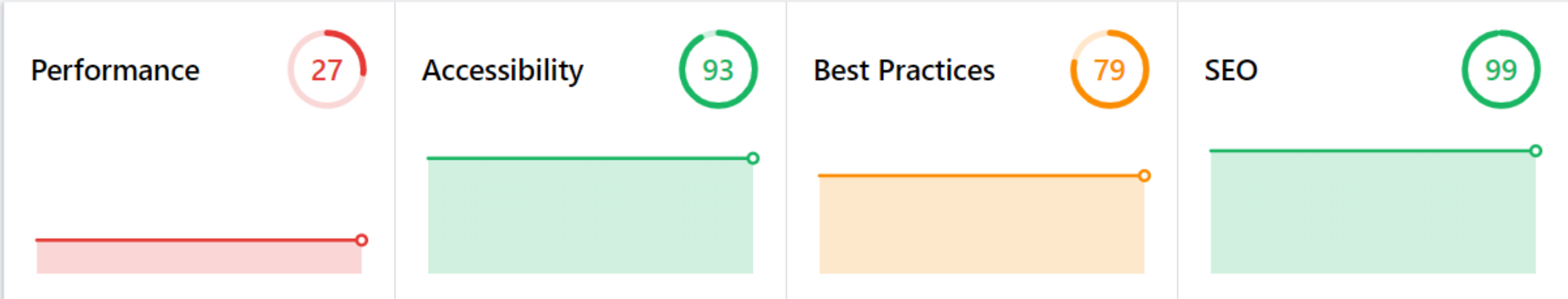
# Performance!

🌐 <http://thetombomb.com>

SWITCH URL

RUN AUDIT

Last audit: Feb 18, 1:30 PM [View Report](#)



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](#).



# Looking at all the options

~~NEXT~~.JS

The Gatsby logo, consisting of a purple circle with a white stylized 'G' inside, followed by the word "Gatsby" in black.

The Vue.js logo, featuring a green and blue stylized 'V' followed by the text "Vue.js" in blue.

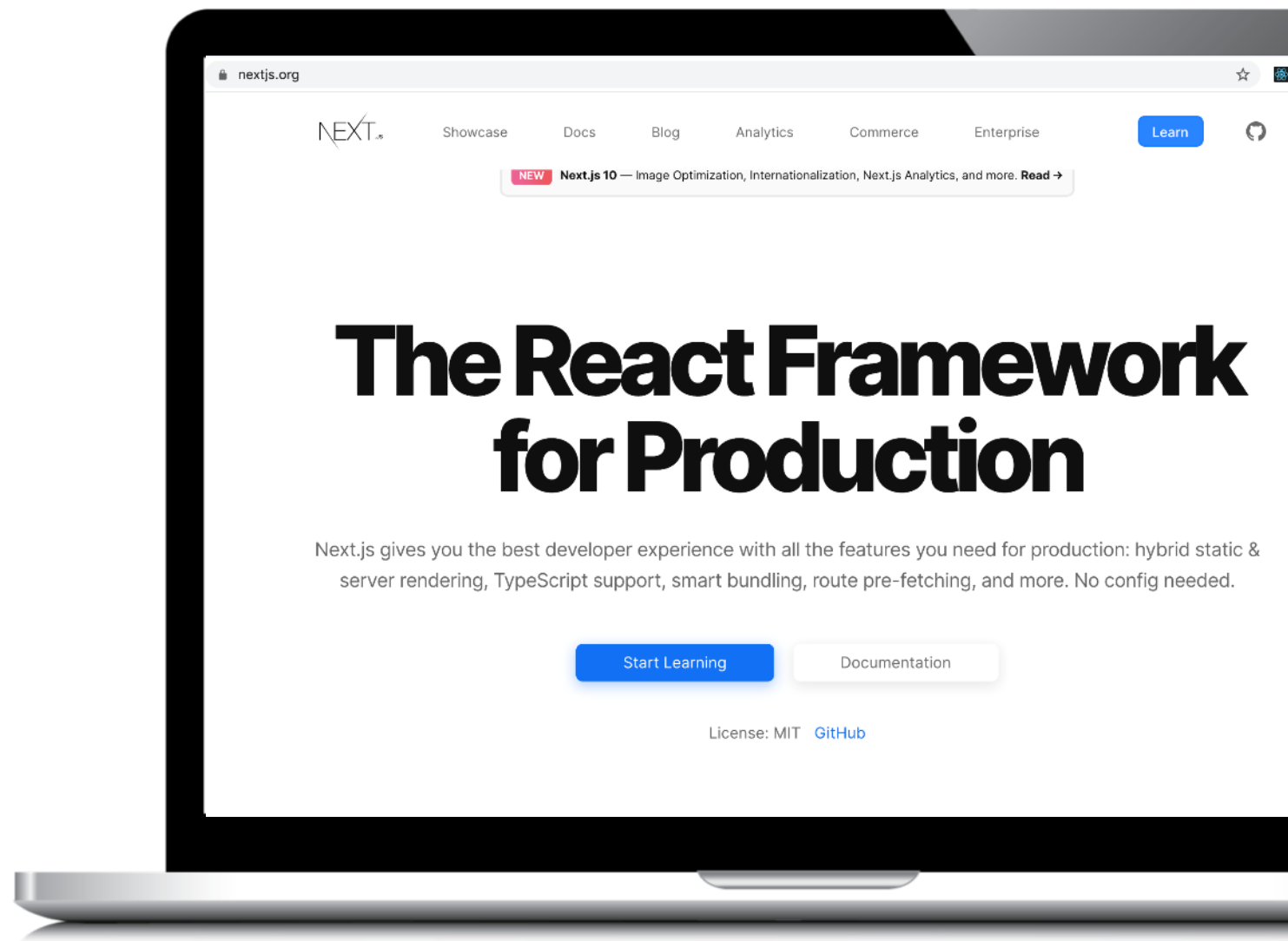
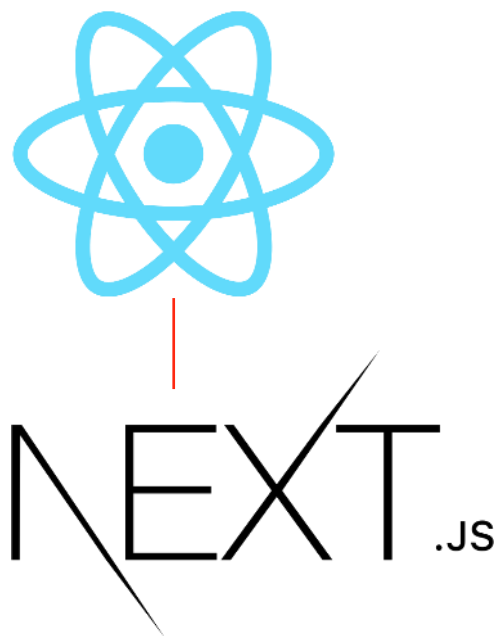
The React logo, featuring a blue stylized atom icon followed by the word "React" in blue.

The Scully logo, featuring a green shield-like shape with a white stylized 'S' inside, followed by the word "scully" in black with a green underline.

The Hugo logo, featuring a pink hexagon with the word "HUGO" in white capital letters and the tagline "A Fast & Modern Static Website Engine" in small text below it.

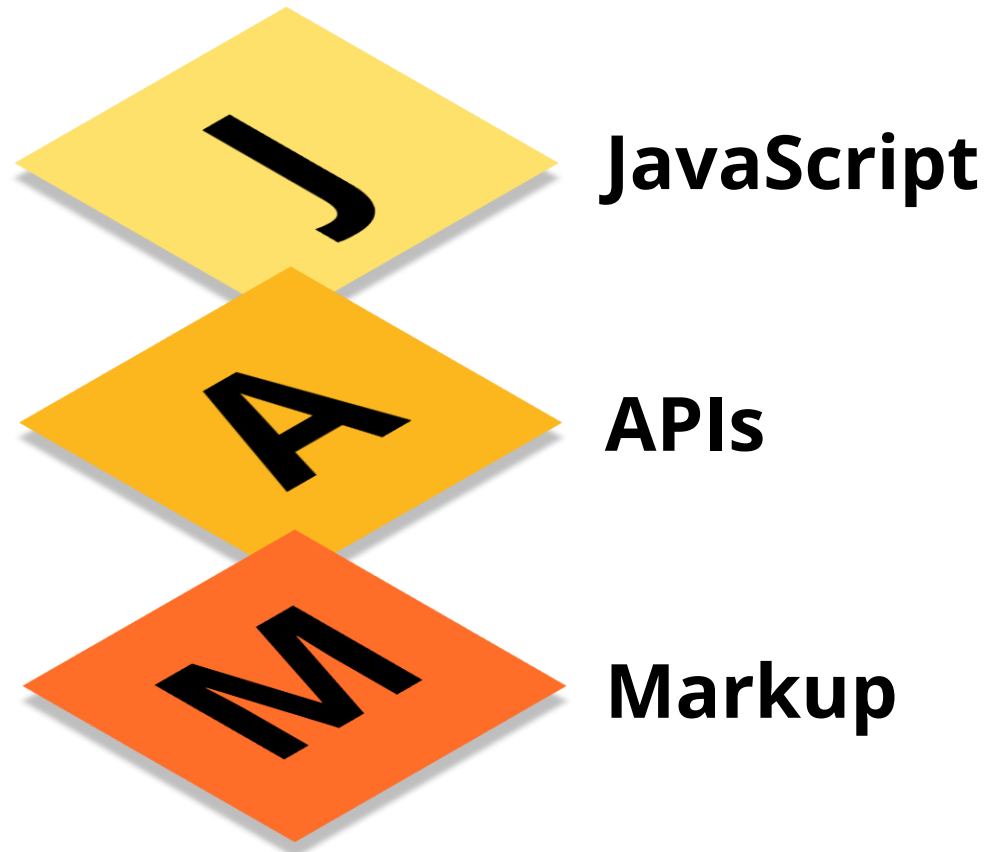


# What is Next.js



# JAMSTACK

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



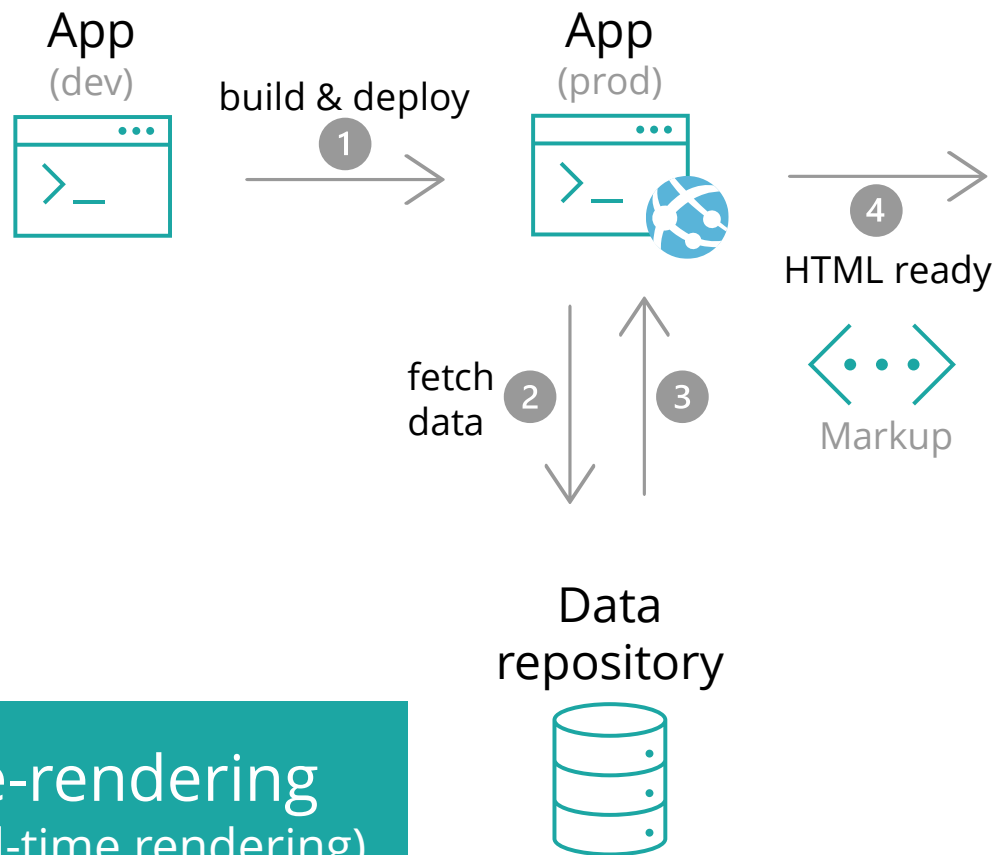
It's a high-level  
classification of an  
**architectural  
approach.**



## **Jamstack anchors:**

- Pre-rendering
- Fast delivery
- Rehydration

## Build & Deploy Time



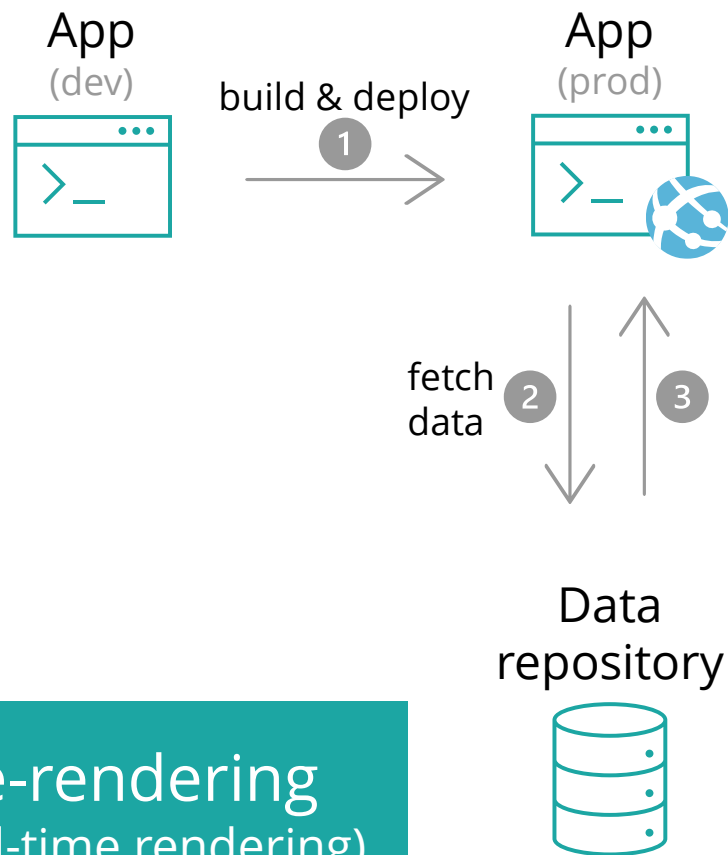
Pre-rendering  
(build-time rendering)

## Request Time

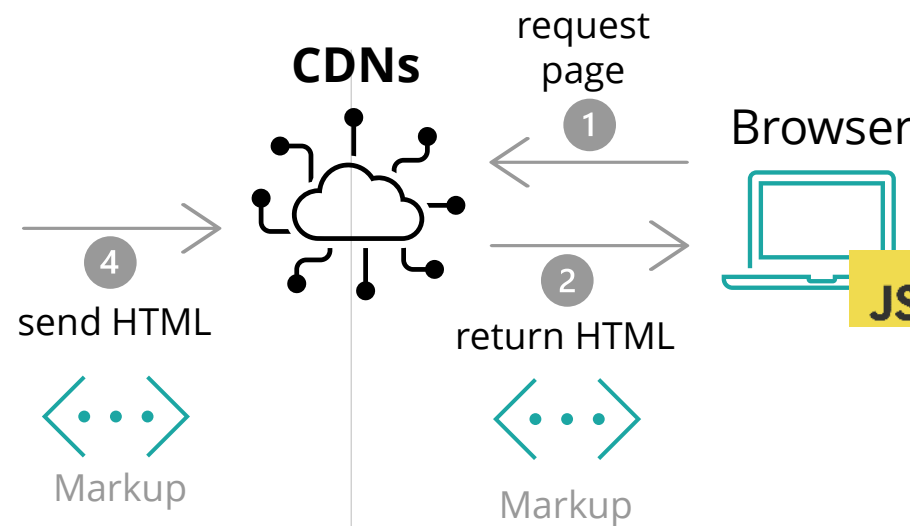




## Build & Deploy Time

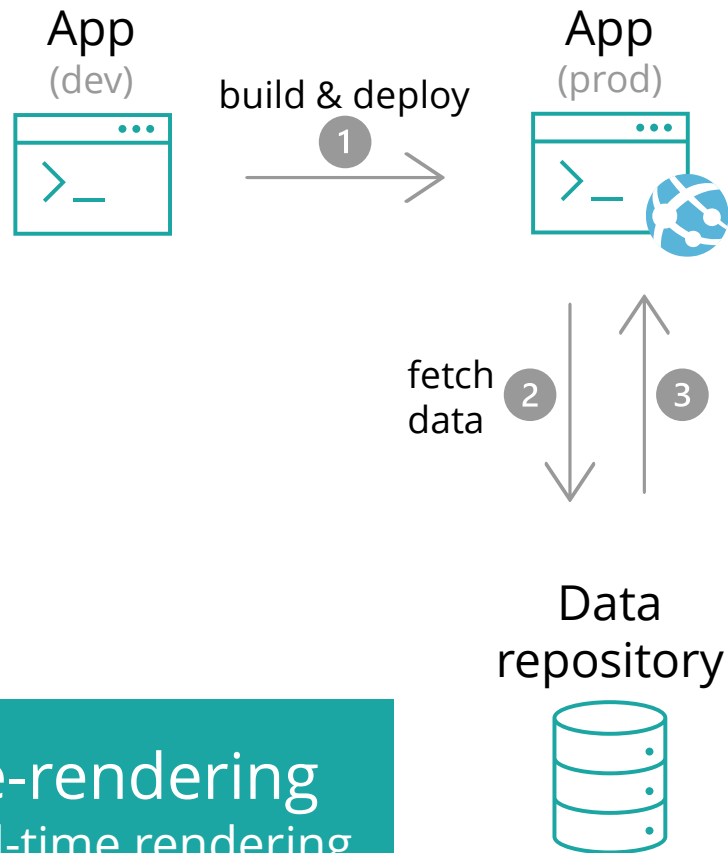


## Request Time

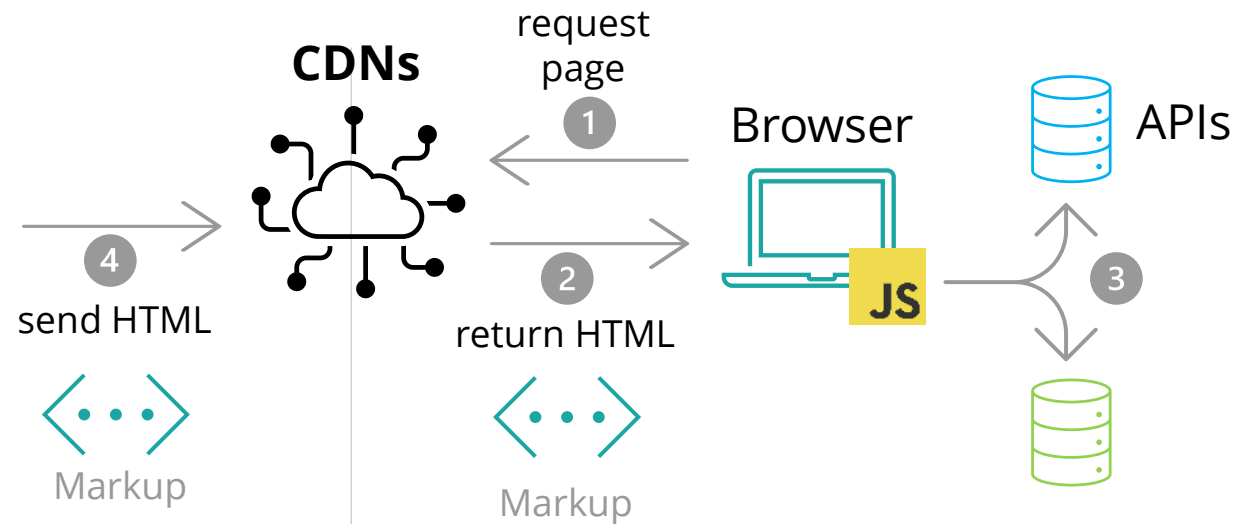


Pre-rendering  
(build-time rendering)

## Build & Deploy Time



Pre-rendering  
build-time rendering



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

## 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

# Code Time

Let's deploy some code in  
Next.js

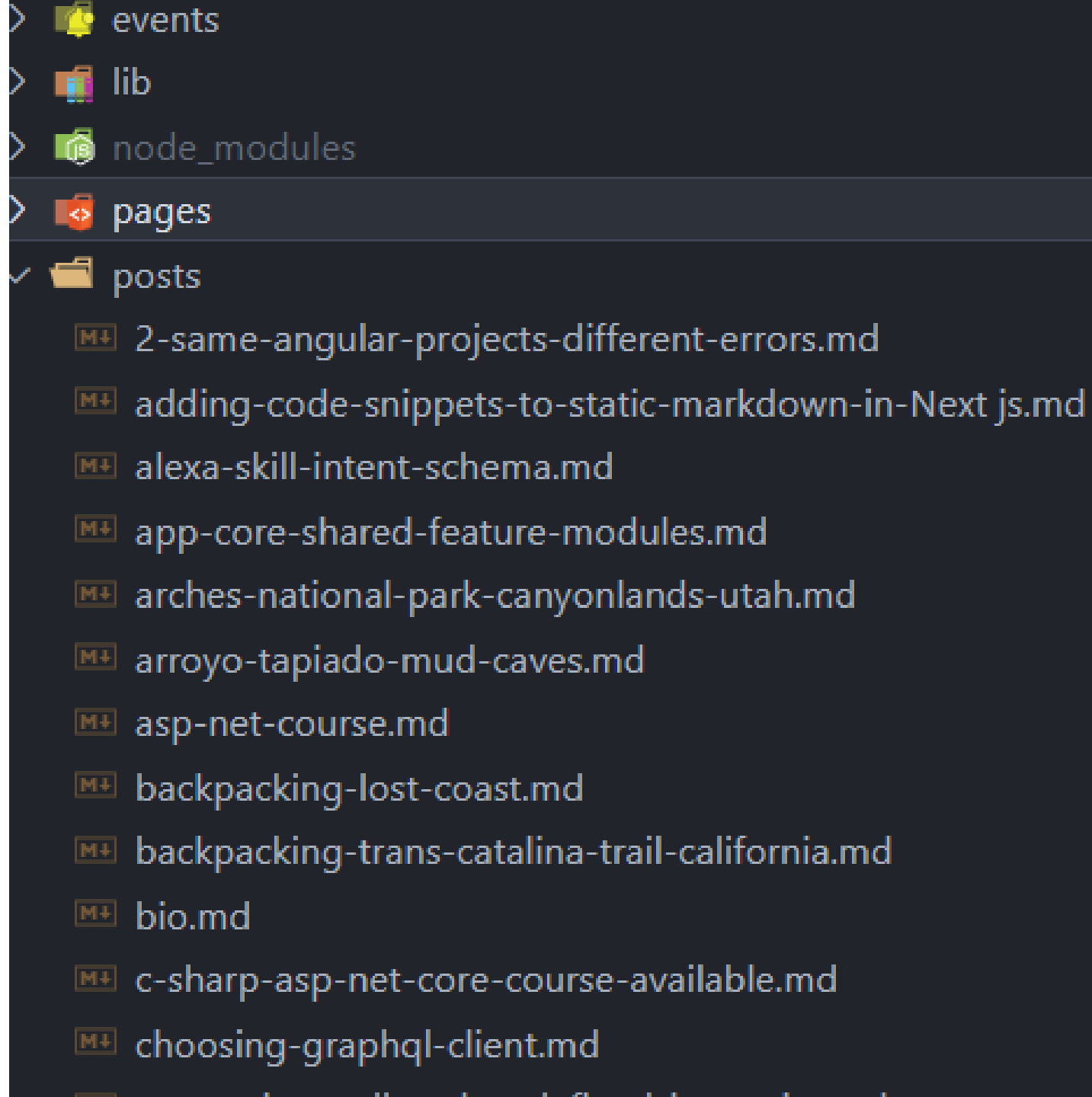




## getStaticProps() in index.js

```
9
10 export async function getStaticProps() {
11   const topPosts = getAllPostsByCategory("topPost");
12   const hobbyPosts = getAllPostsByCategory("hobby");
13
14   return {
15     props: {
16       topPosts: topPosts,
17       hobbyPosts: hobbyPosts,
18     },
19   };
20 }
21
22 export default function Home({ topPosts, hobbyPosts }) {
23   return (
24     <>
25       <HomeLayout>
26         <Head>
```

# Local Directory of Posts



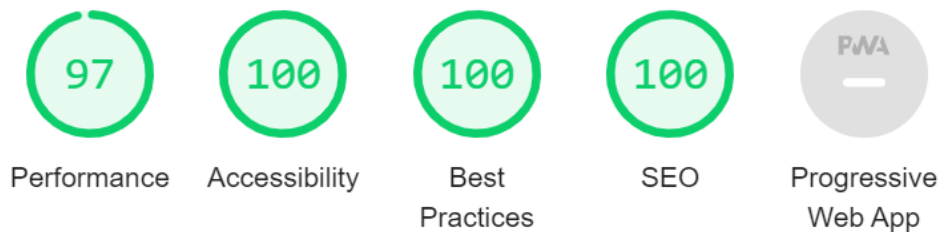
## CodeBlock

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.

```
1 // Don't forget to import codeblock at the top of your file
2 import CodeBlock from "../../components/codeblock"
3
4 <ReactMarkdown source={postData.markdown} renderers={{ code: CodeBlock
```

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

# Was it worth it?



▲ 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.**



## 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



# Keep It Simple

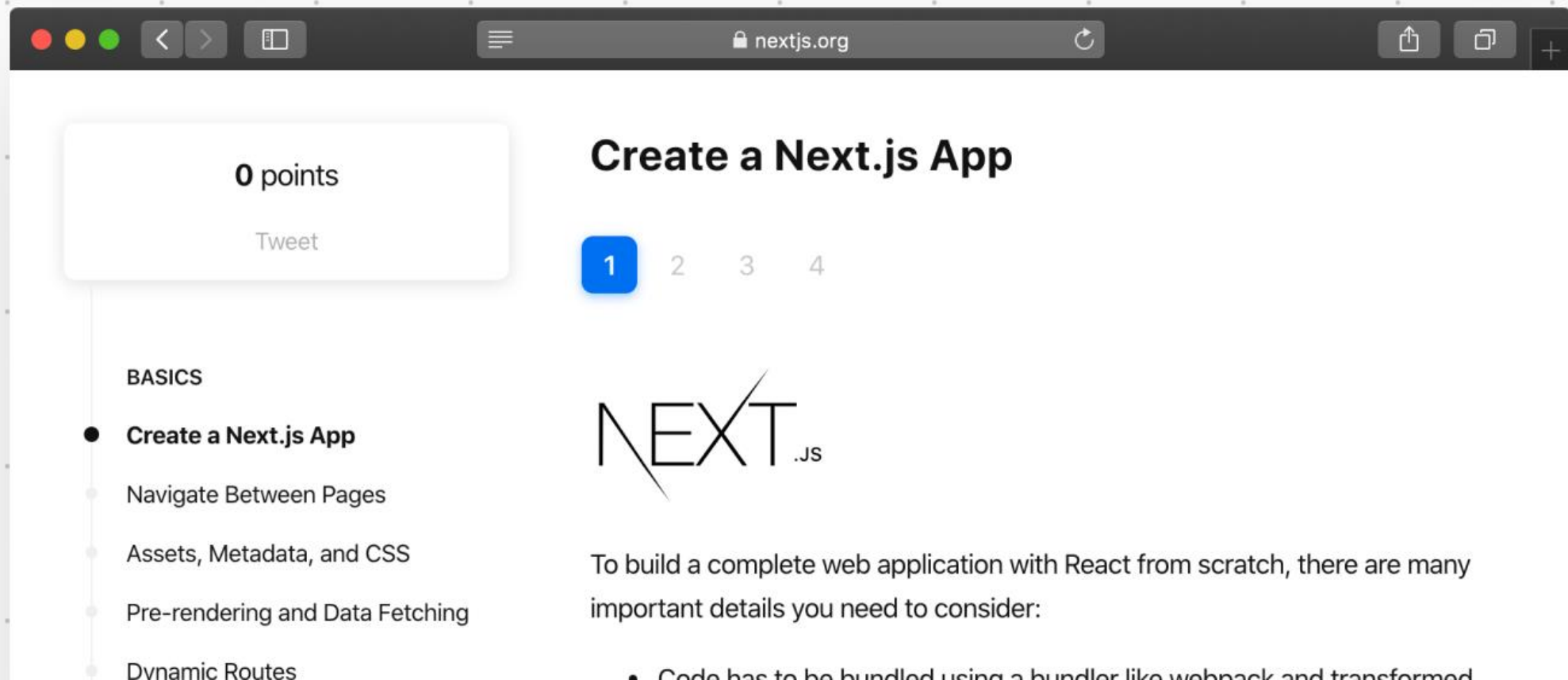


# Headless WordPress

- [Colby Fayock: Headless WordPress with Next.js tutorial](#)



# nextjs.org/learn



The screenshot shows a web browser window displaying the Next.js Learn page. The browser's address bar shows 'nextjs.org'. The page content includes a '0 points' badge with a 'Tweet' button, a progress indicator with steps 1, 2, 3, and 4 (step 1 is active), and a sidebar menu. The sidebar menu lists 'BASICS' with sub-items: 'Create a Next.js App' (selected), 'Navigate Between Pages', 'Assets, Metadata, and CSS', 'Pre-rendering and Data Fetching', and 'Dynamic Routes'. The main content area features the 'Create a Next.js App' title, the Next.js logo, and introductory text about building web applications with React.


0 points  
Tweet

1 2 3 4

**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:

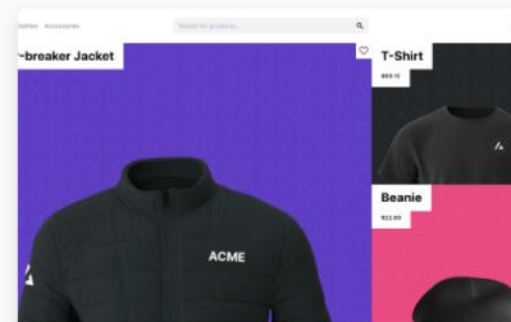
- Code has to be bundled using a bundler like webpack and transformed

# Next.js templates and more



## Next.js

A Next.js app and a Serverless Function API.



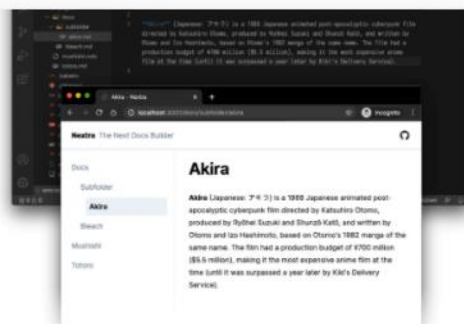
## Commerce Starter Kit

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



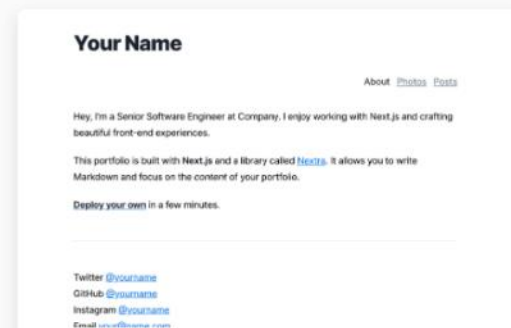
## Virtual Event Starter Kit

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.





# Thomas Desmond

Developer Advocate @ Sitecore

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

[@ThomasJDesmond](#)

[www.TheTomBomb.com](http://www.TheTomBomb.com)