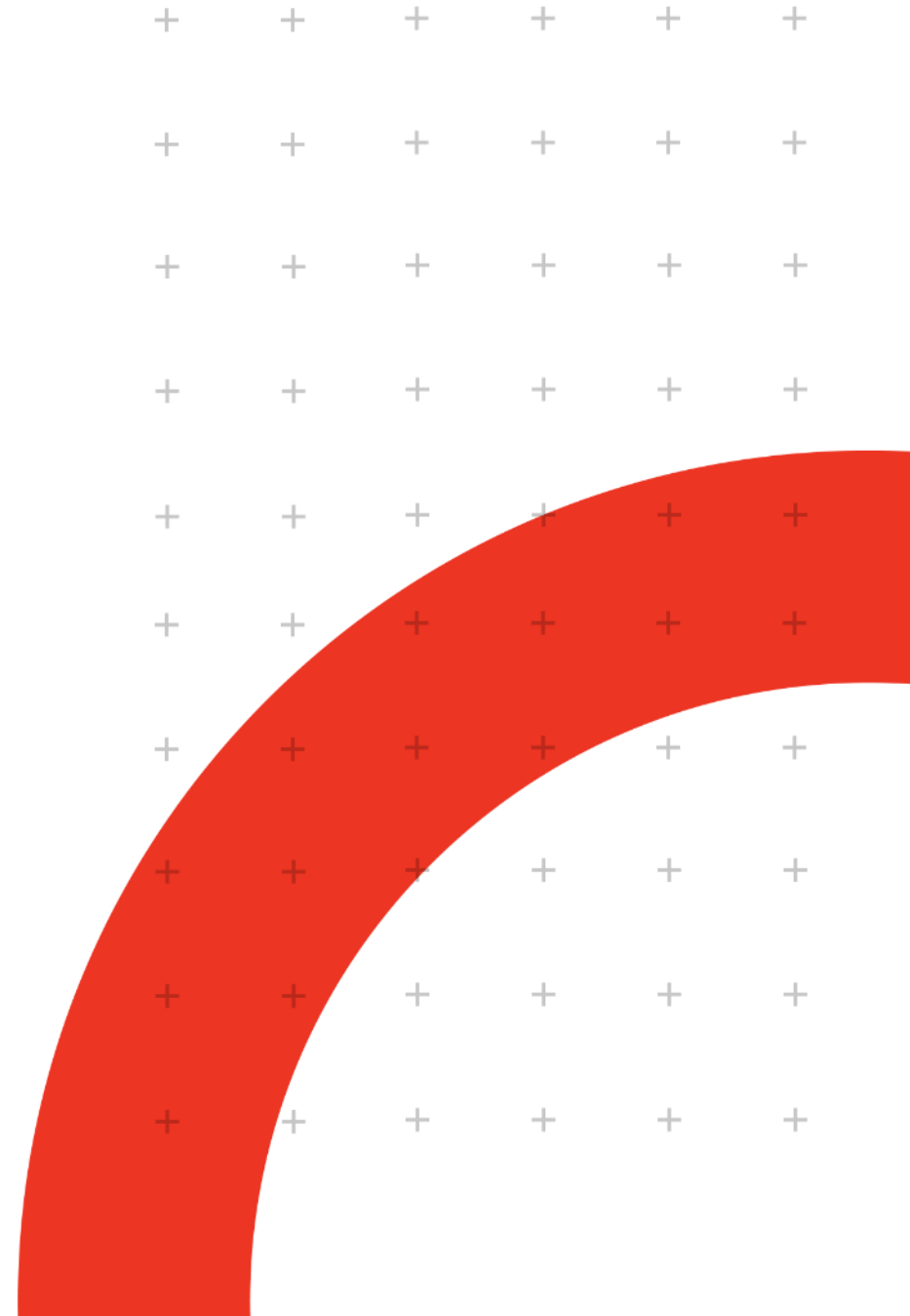




Render Here, Render There, Render Everywhere with Next.js

An introduction to rendering options
on the web

Thomas Desmond





Thomas Desmond

Developer Advocate @ Sitecore

- Frontend developer advocate
- 7+ years of development experience
- Live in San Diego, CA

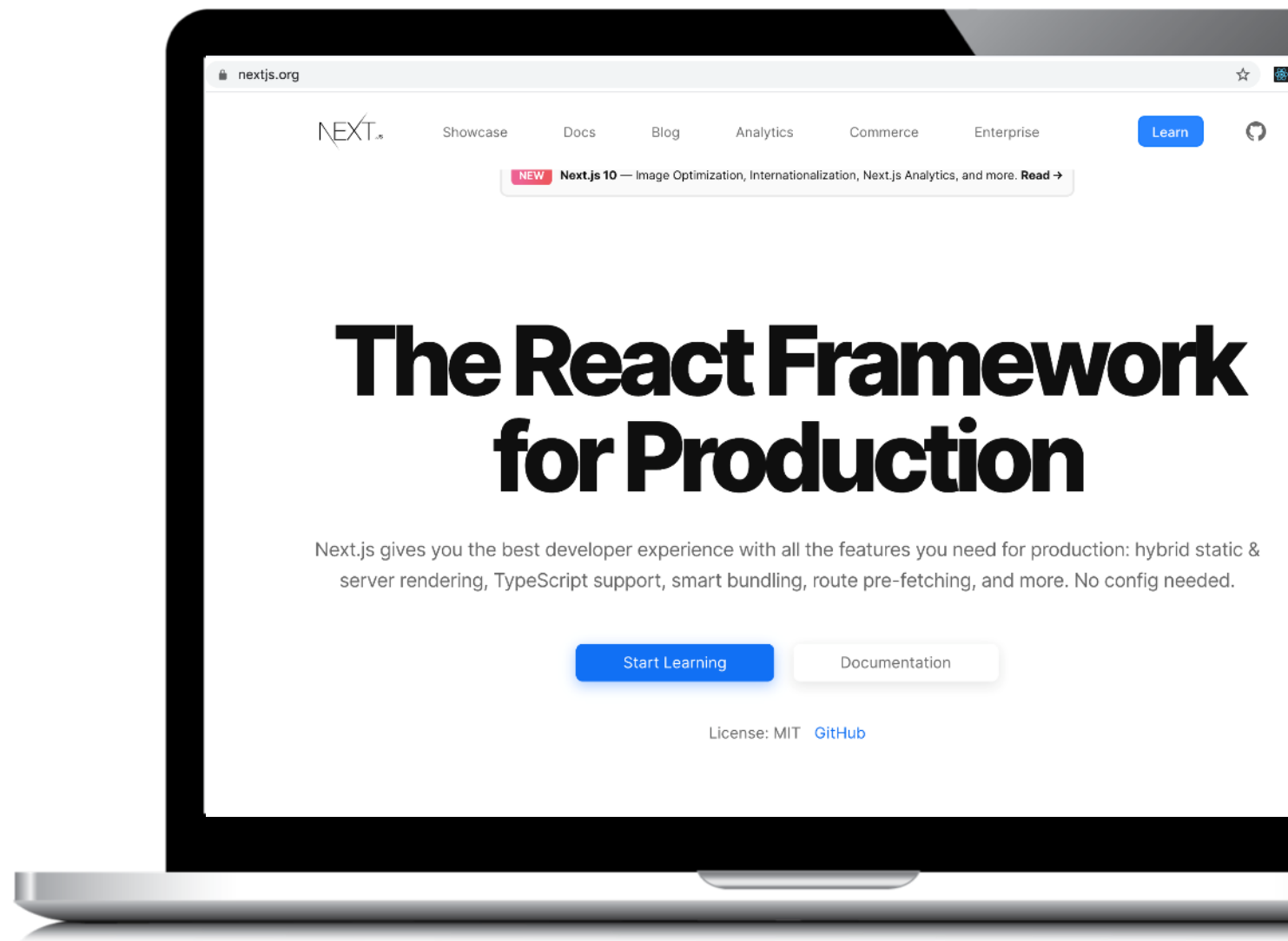
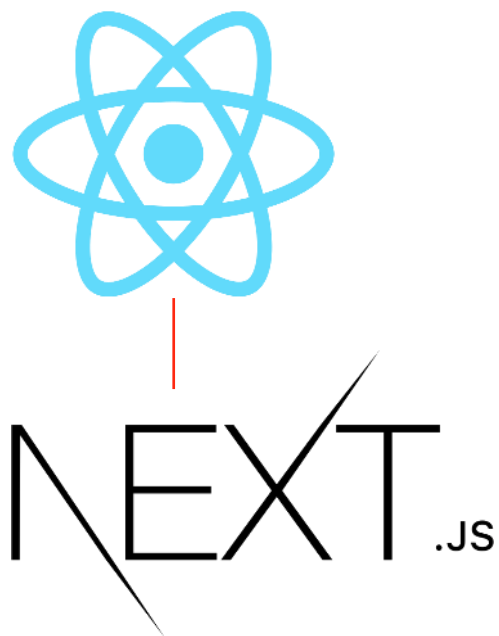
[@ThomasJDesmond](#)

www.TheTomBomb.com

What is rendering?



What is Next.js



Why do I care about rendering?



The Big Three Options

Server Side Rendering
Client Side Rendering
Static Site Generation



Server Side Rendering

- Happens each request
- Reliable powerful machine
- Maintenance/Upkeep

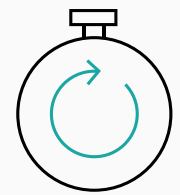


Server Side Rendering Flow

1

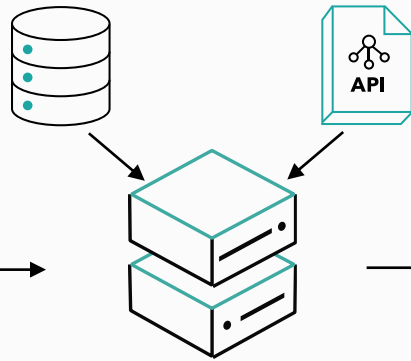


Server receives the request to view the page

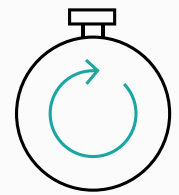


Loading

2

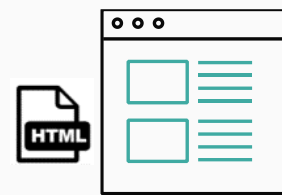


Server gathers all data (databases, API's, etc), generates HTML, sends that to browser

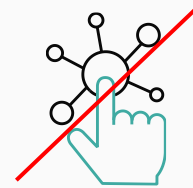


Loading

3

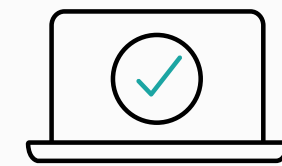


Browser displays HTML and downloads minimal JavaScript



Visible but not interactive

4



Page is now interactive and useable



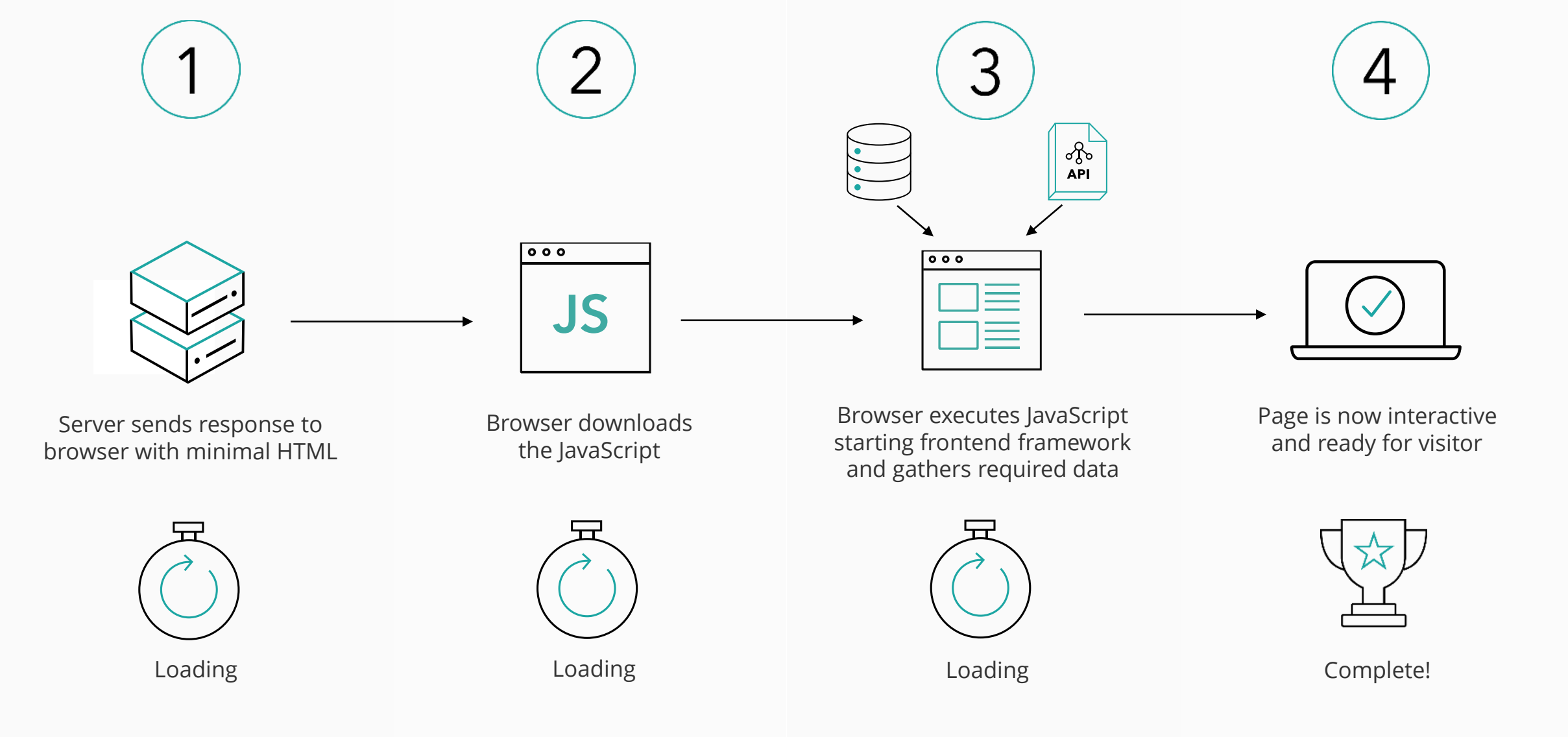
Complete!


```
export async function getServerSideProps(context) {  
  const res = await fetch(`https://...`)  
  const data = await res.json()  
  
  if (!data) {  
    return {  
      notFound: true,  
    }  
  }  
  
  return {  
    props: {}, // will be passed to the page component as props  
  }  
}
```

Client Side Rendering

- Data fetches each request
- Tiny HTML sent
- Highly flexible







Code Time

Looking at how Next.js
handles SSR and CSR

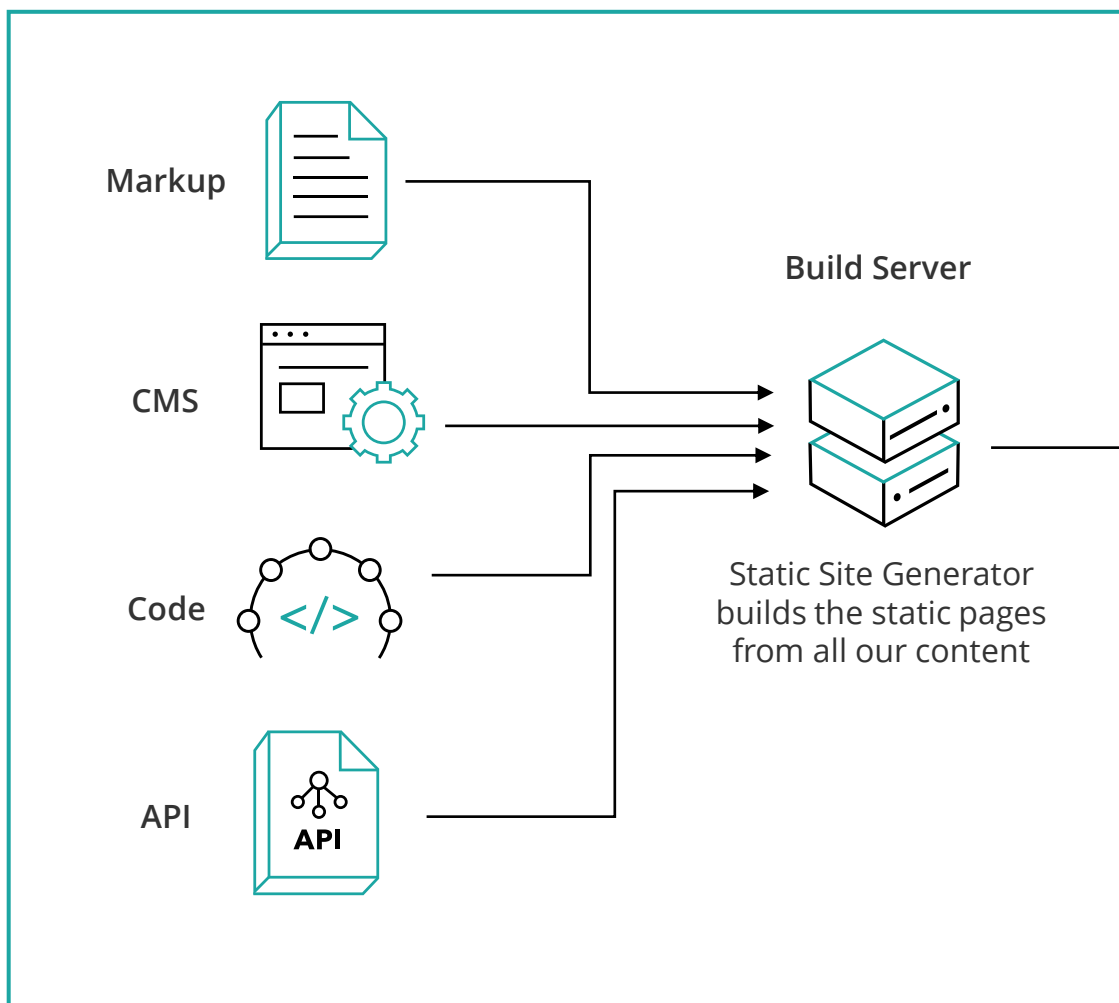
Static Site Generation

- Data fetches once at build
- Super speed and SEO
- Content that changes infrequently

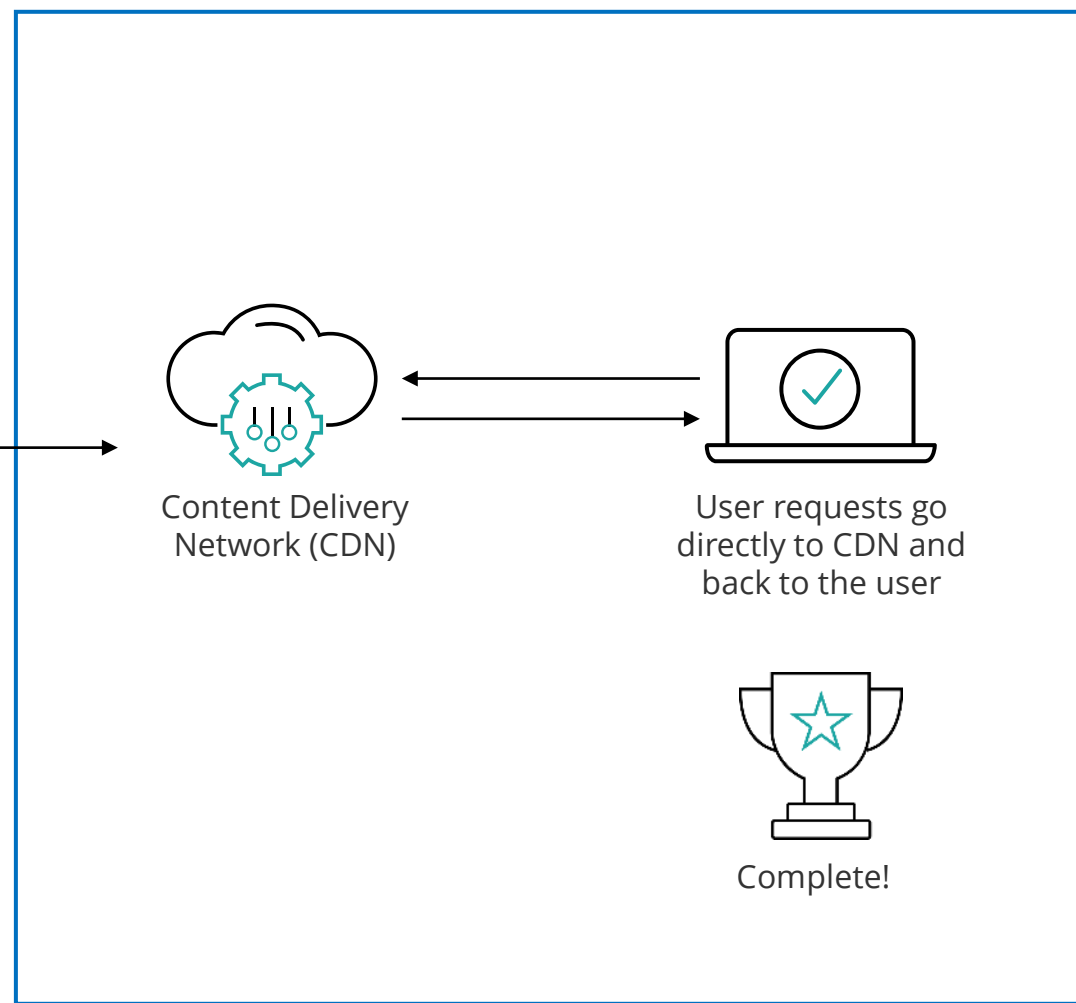


Static Site Generation (SSG) Flow

Build time



Visitor request time



getStaticProps()

```
export async function getStaticProps(context) {  
  const res = await fetch(`https://.../data`)  
  const data = await res.json()  
  
  if (!data) {  
    return {  
      notFound: true,  
    }  
  }  
  
  return {  
    props: { data }, // will be passed to the page component as props  
  }  
}
```




Code Time

Looking at how Next.js
handles SSG

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

Incremental Static Regeneration (ISR)

EDGE FUNCTIONS (BETA)

Fearlessly Dynamic

Edge Functions give you the benefits of static with the power of dynamic. Now you can personalize and experiment without sacrificing speed or performance.

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

Geolocation Edge Function Next.js

```
1 // pages/_middleware.js
2
3 export default function middleware (req, ev) {
4   console.log('Edit and run at the edge!')
5
6   return new Response({
7     ip: req.ip,
8     geo: req.geo, // this will spin the globe!
9     ua: req.ua
10  })
11 }
```

 Run

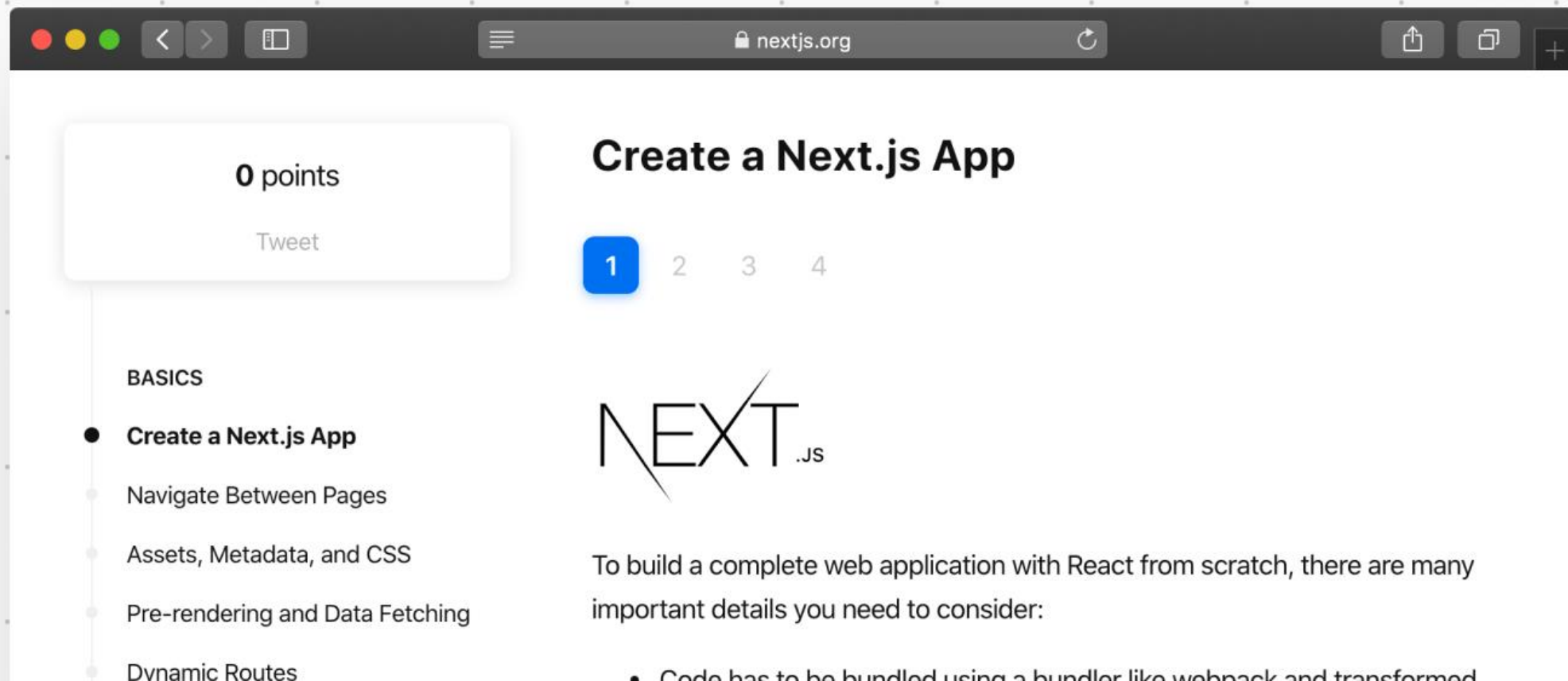
Why limit it to just one?

The Big Three Options

Server Side Rendering
Client Side Rendering
Static Site Generation



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 progress indicator showing '0 points' and a 'Tweet' button. The main heading is 'Create a Next.js App', followed by a progress bar with steps 1, 2, 3, and 4, where step 1 is highlighted. Below the progress bar is the Next.js logo. The text below the logo reads: 'To build a complete web application with React from scratch, there are many important details you need to consider:'. A sidebar on the left lists the topics: 'BASICS', 'Create a Next.js App' (selected), 'Navigate Between Pages', 'Assets, Metadata, and CSS', 'Pre-rendering and Data Fetching', and 'Dynamic Routes'.

0 points

Tweet

Create a Next.js App

1 2 3 4

NEXT.js

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

BASICS

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

Don't forget to complete an online

Render Here, Render There, Render Everywhere with Next.js

Your evaluation helps organizers build better
conferences
and helps speakers improve their sessions.

Thank you!

Thomas Desmond

Developer Advocate @ Sitecore

- Frontend developer advocate
- 7+ years of development experience
- Live in San Diego, CA

[@ThomasJDesmond](#)

www.TheTomBomb.com



Reference Links

- [Vercel Edge Functions](#)
- [Bringing Jamstack to the Enterprise \(PayPal Story\)](#)
- [Nextjs.org/learn](#)
- [Vercel](#)
- [Example Code](#)





Questions?

IN-PERSON EVENT

APRIL 5-7, 2022

<anglebrackets/>

LAS VEGAS, NV
MGM GRAND

