Render Time

Render when, where, how, what, and why.

Phil Hawksworth, Netlify

Oh, hello

Phil Hawksworth

Developer Experience, Netlify

Oh, hello

@PhilHawksworth

Developer Experience, Netlify

Oh, hello

@PhilHawksworth@indieweb.social

Developer Experience, Netlify

NAMING IS HARD

Phil Hawksworth • hawksworx.com

Developer Experience, Netlify

A slight change of pace
after Evan

Some foundations for
talks to come

Rendering

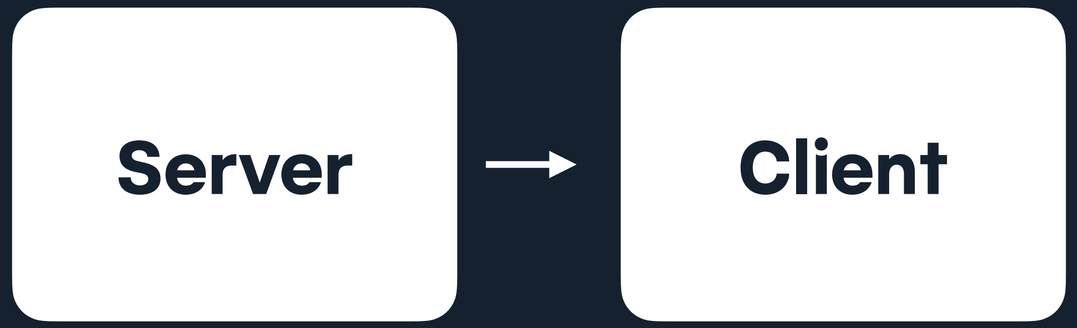
HTML / DOM

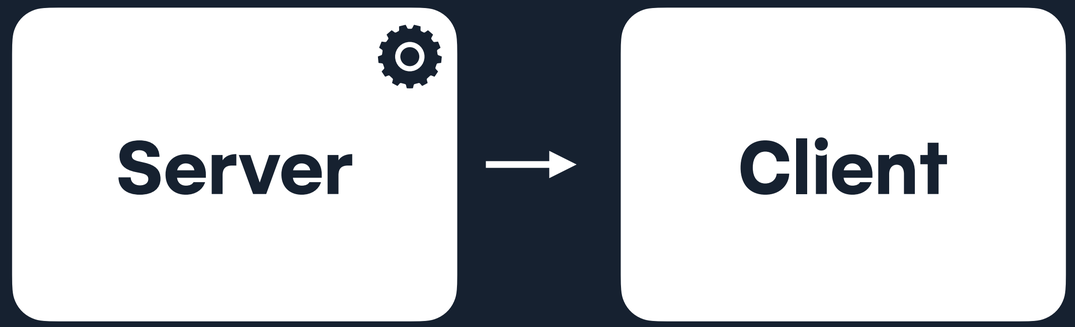
“...something something rendering on the server...”

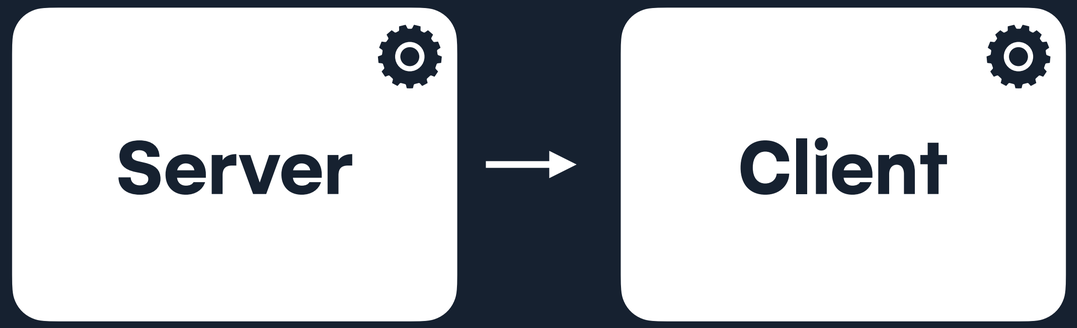
— Phil Hawksworth, 2013

“...ha ha ha do you just mean serving HTML?”

— The person listening to Phil Hawksworth, 2013


Server
Client

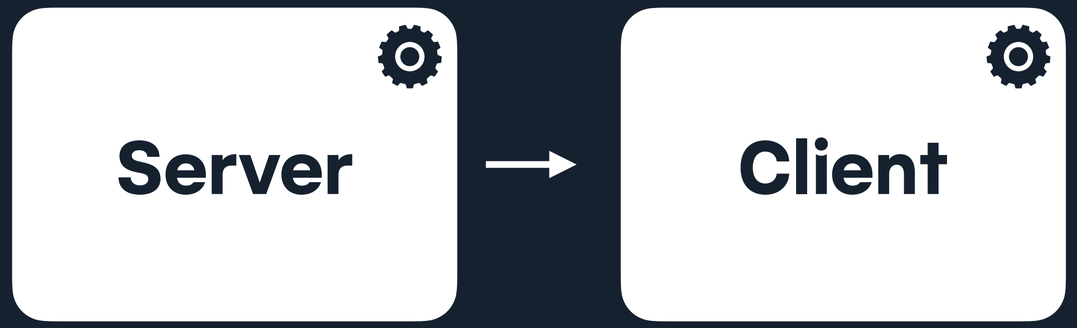

Server
Client

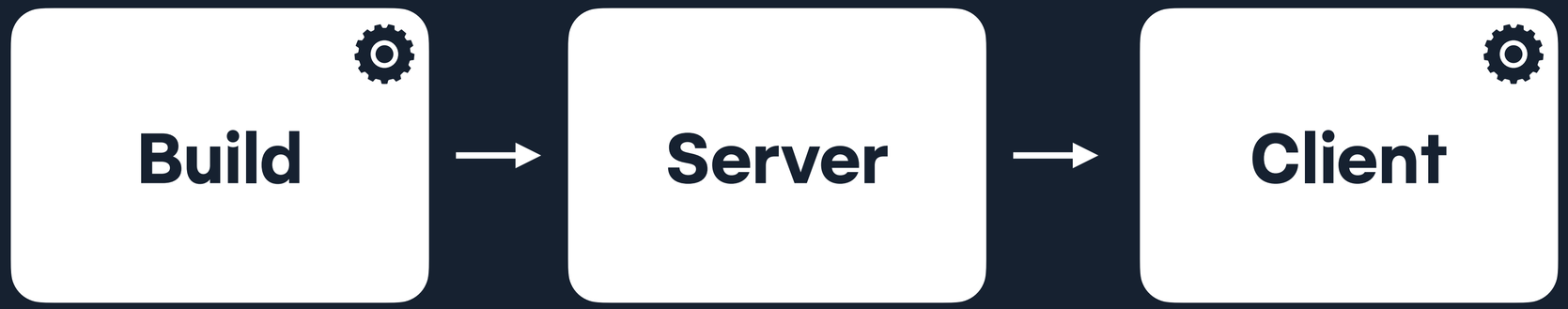

Server
Client

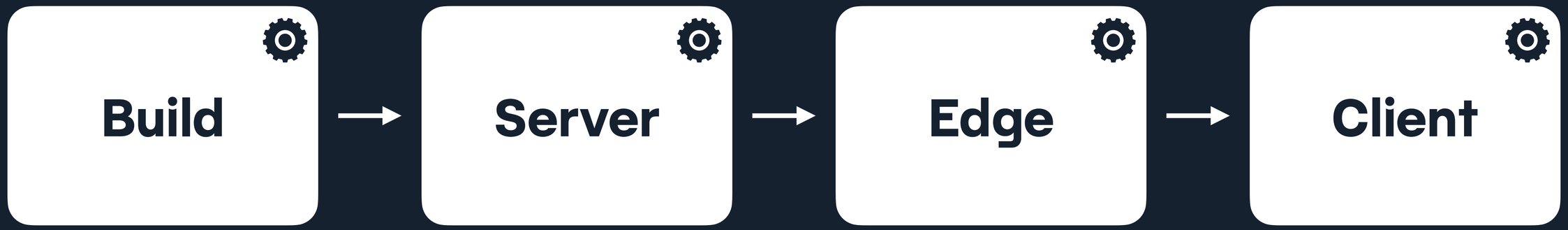
Robustness

The rule of least power

https://www.w3.org/2001/tag/doc/leastPower.html


Server
Client


Build
Server
Client


Build
Server
Edge
Client

On-demand Rendering

Decide what rendering strategy at the route level: SSR, SSG, CSR, ISR, ESR, SWR. Build any kind of website or web application with optimized performance in mind.

“by thunder, there are a lot of ways to approach rendering!”

— Phil Hawksworth, May 2023

Confusion

A place

for rendering

A time and a place

for rendering

What / Where / When / How / Why

A few mins

/ Into and twaddle

15 mins

/ Examining rendering
/ Questioning

10 mins

/ Applying techniques
/ Example
/ Lessons

SECTION 1

Examining rendering

What even do these acronyms mean?

NAMING IS HARD

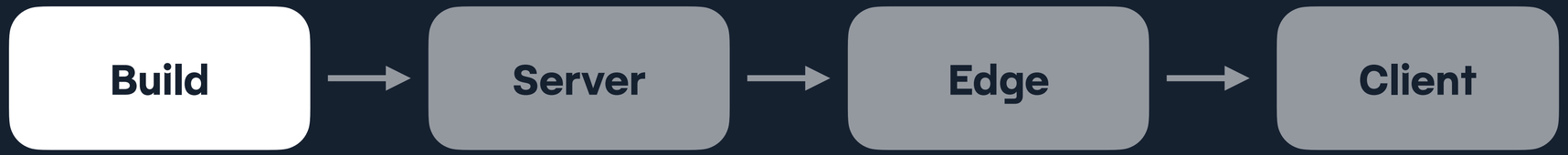
Terminology


G / ISR / ODB / SPA / MPA / ESR
PA / MPA / SSR / CSR / SSG / D
/ SPA / MPA / SSR / CSR / SSG
/ SSR / CSR / SSG / DPR / DSG
SSR / CSR / SSG / DPR / DSG /
ESR / ISR / ODB / SPA / MPA /
NAMING IS HARD

RENDERING

SSG

Static Site Generation



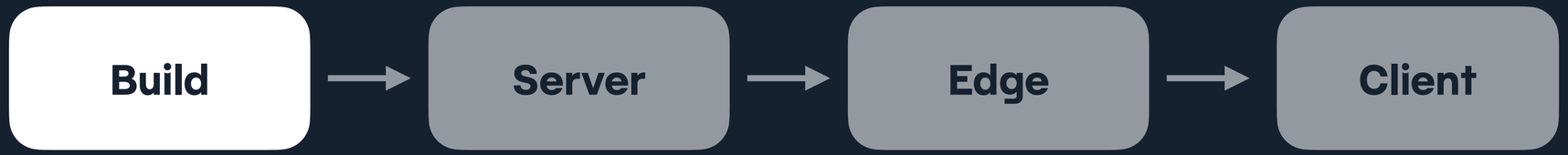
CI/CD

Continuous Integration / Continuous Deployment

RENDERING

SSG

Static Site Generation



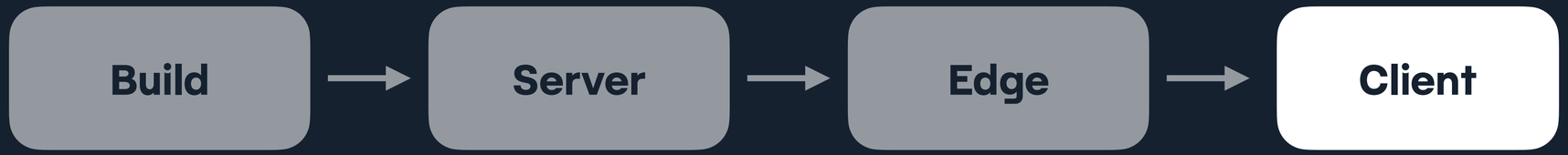
TRADITIONAL STACK

VARIOUS INCREASINGLY POPULAR PROVIDERS

RENDERING

CSR

Client side rendering



When...

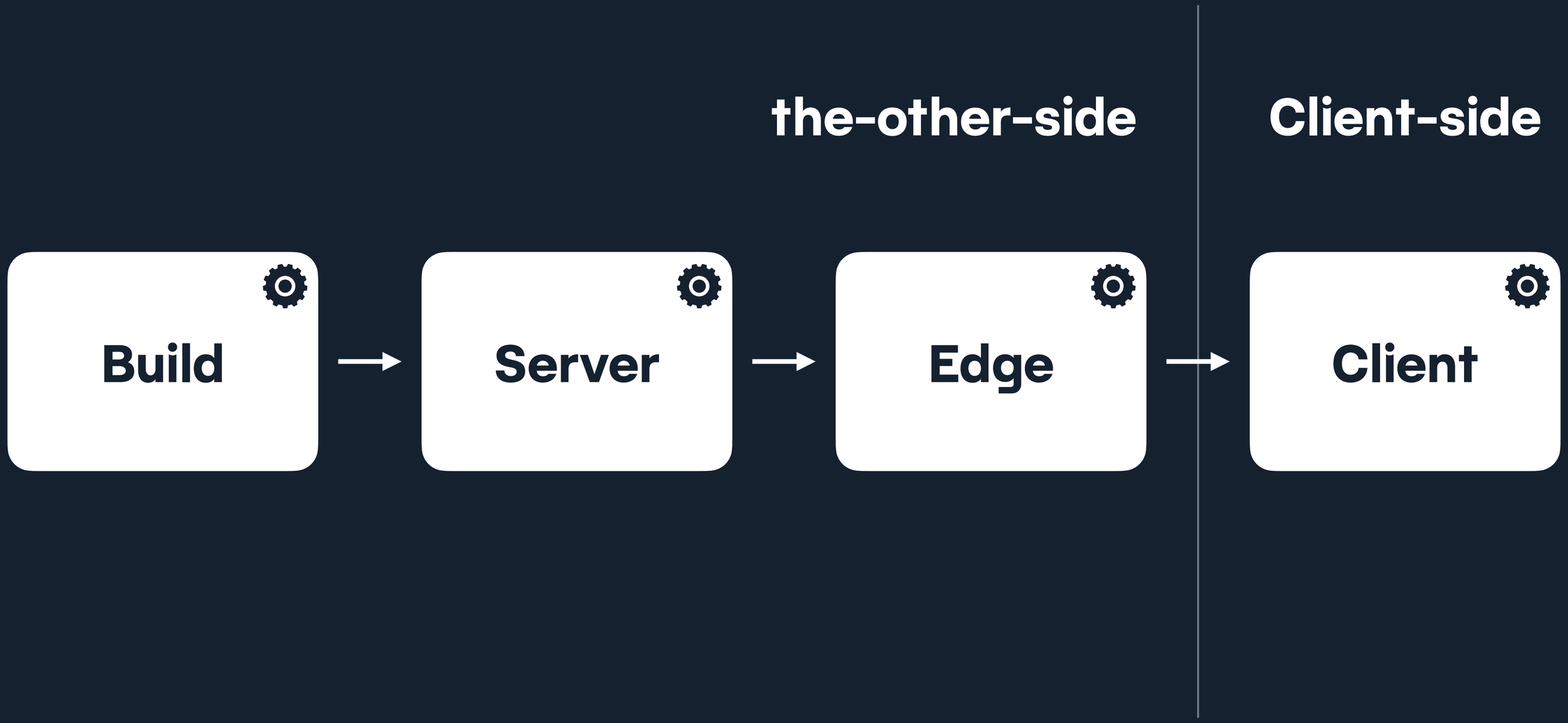
the assets
have arrived

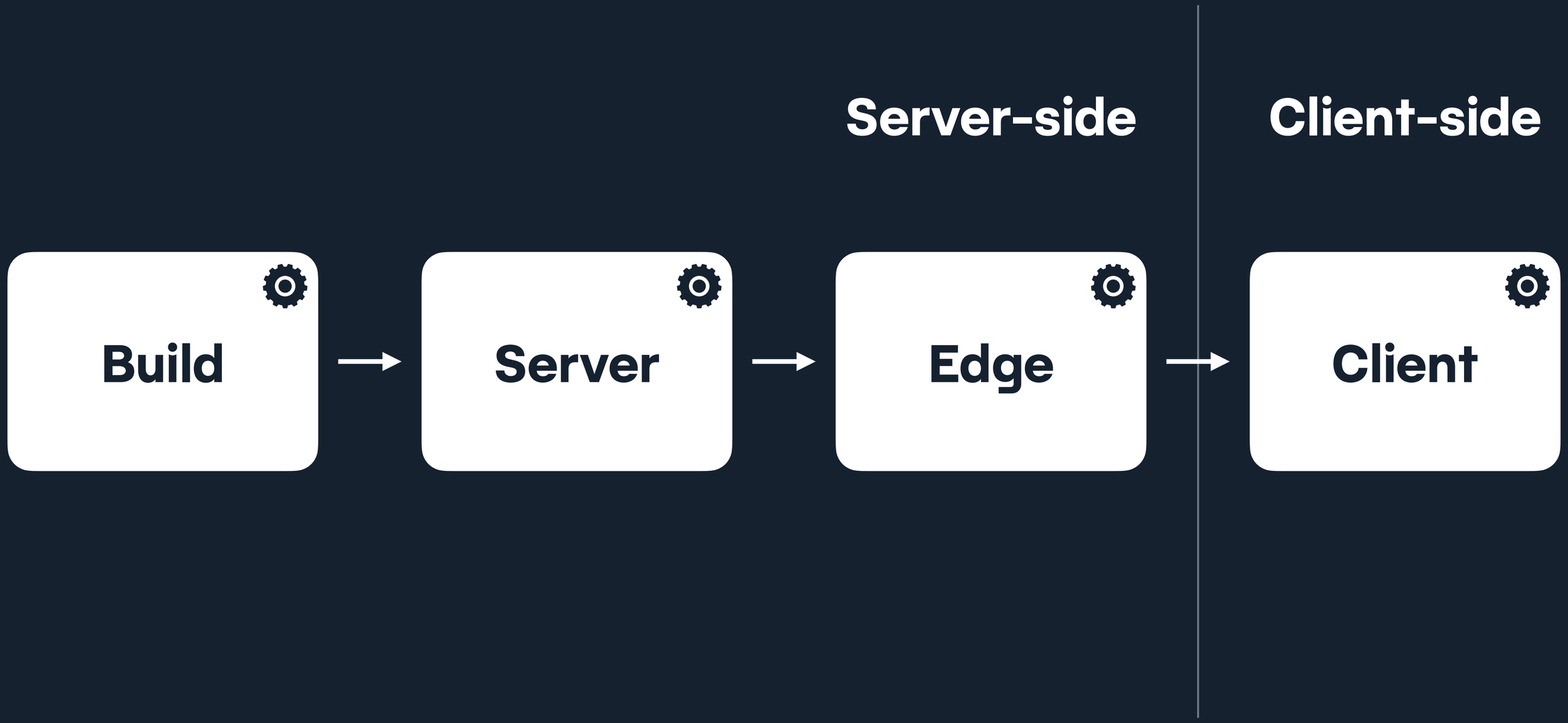
the user
interacts

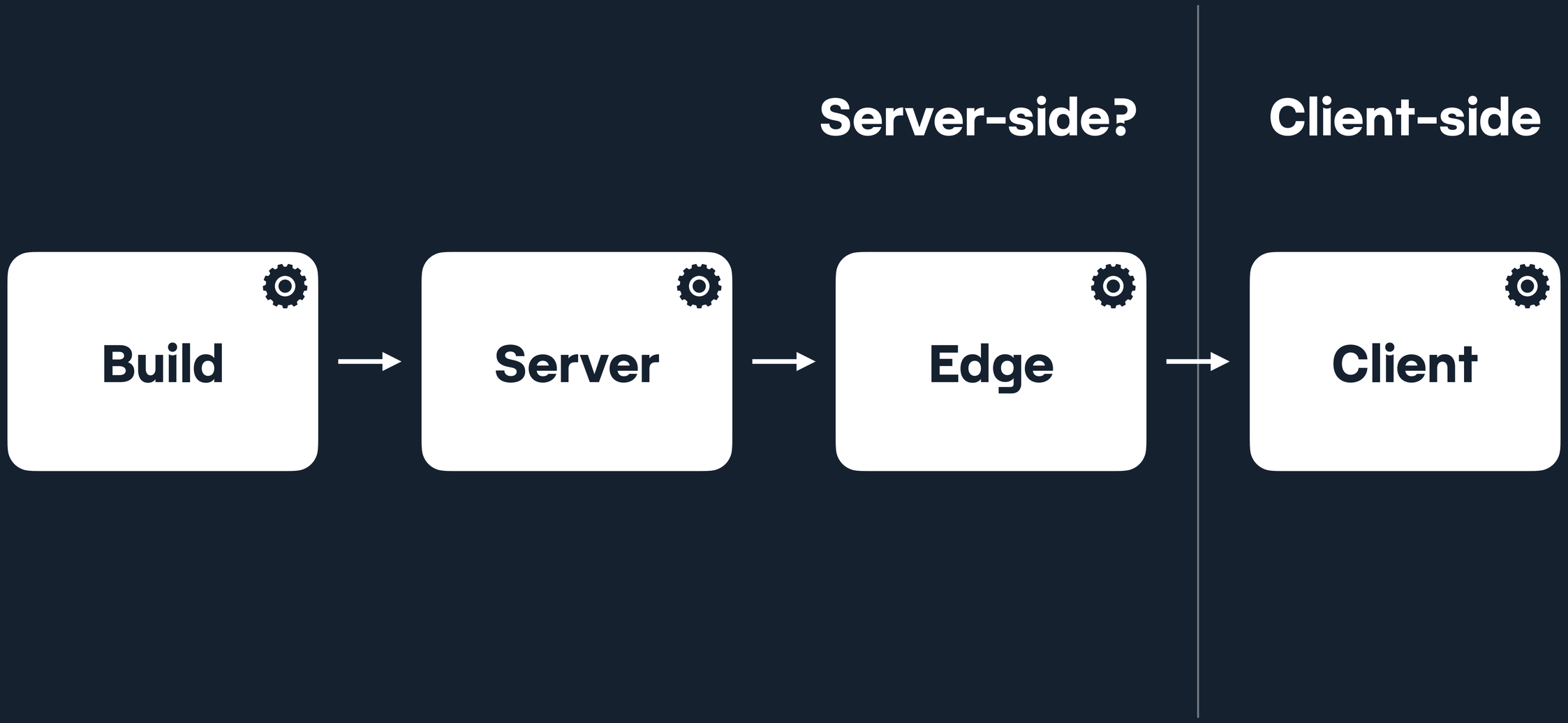
RENDERING

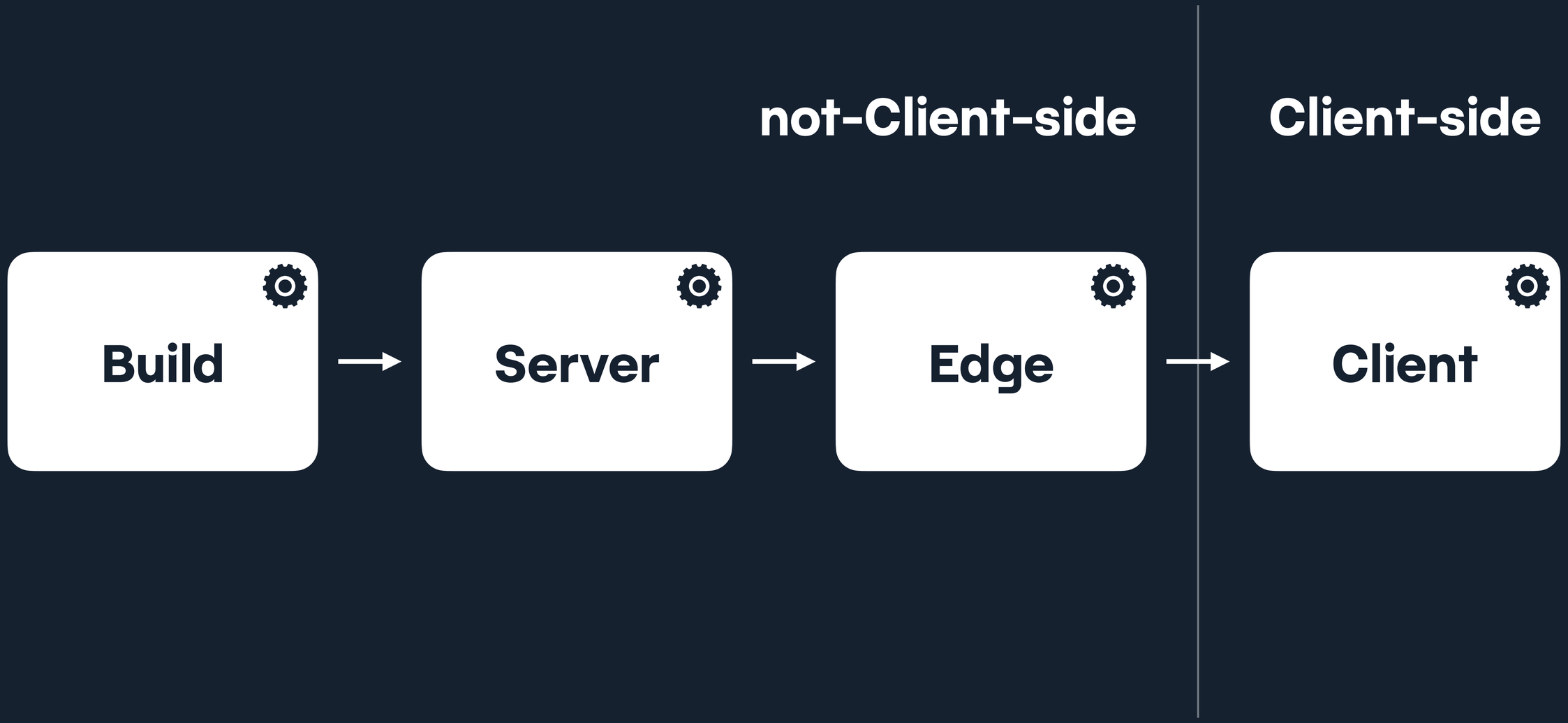
SSR

Server side rendering


the-other-side
Client-side
Build
Server
Edge
Client


Server-side
Client-side
Build
Server
Edge
Client


Server-side?
Client-side
Build
Server
Edge
Client


not-Client-side
Client-side
Build
Server
Edge
Client

RENDERING

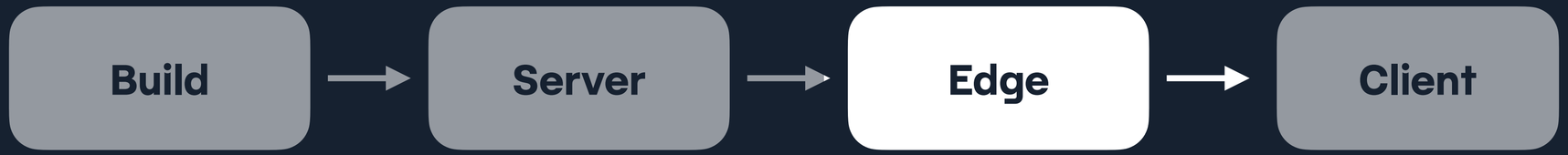
SSR

Server side rendering

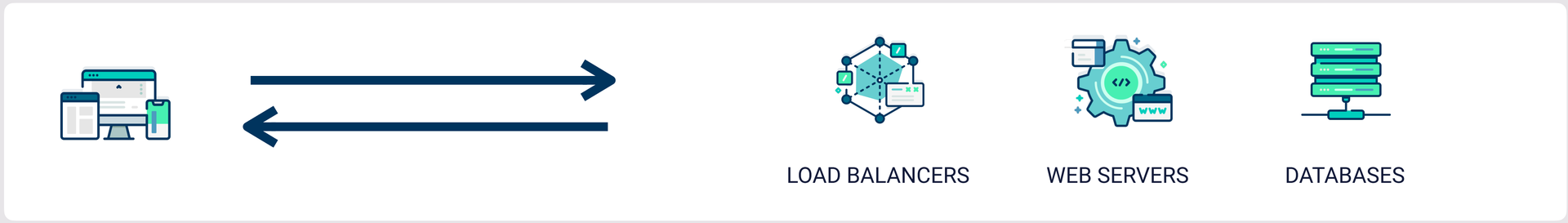
RENDERING

ESR

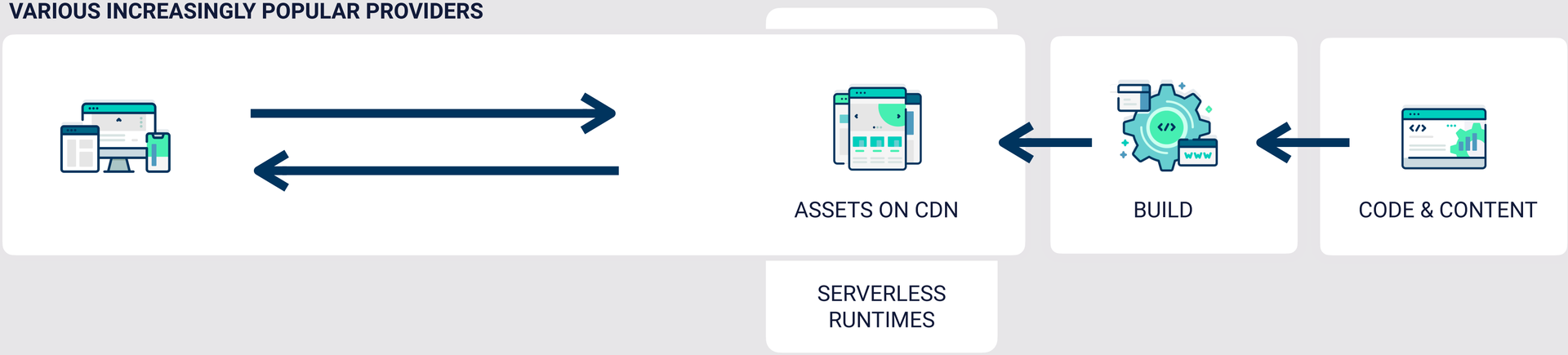
Edge side rendering



TRADITIONAL STACK



VARIOUS INCREASINGLY POPULAR PROVIDERS



“so which one is best, Phil?”

— Some of you, possibly, May 2023

“It depends”

— Phil Hawksworth, May 2023

“Yeah. Great. Thanks.”

— Some of you, possibly, May 2023

“It depends on what?”

— Some of you, hopefully, May 2023

“I’m glad you ask”

— Me again, May 2023

The rule of least power

https://www.w3.org/2001/tag/doc/leastPower.html

https://unsplash.com/photos/qG6QtyOaOGQ

Serverside, doesn't have
to mean **Serverful**.

If I can do things in advance, I will.
If I can't, but can do things serverless, I will.
If I can't, but can add a server, I will.

Increments

DPR / ODB / DSG / ISR / FFS

Increments

DPR / ODB / DSG / ISR

RENDERING

DPR

Distributed Persistent Rendering

RENDERING

ODB

On-demand Builders

RENDERING

DSG

Deferred Static Generation

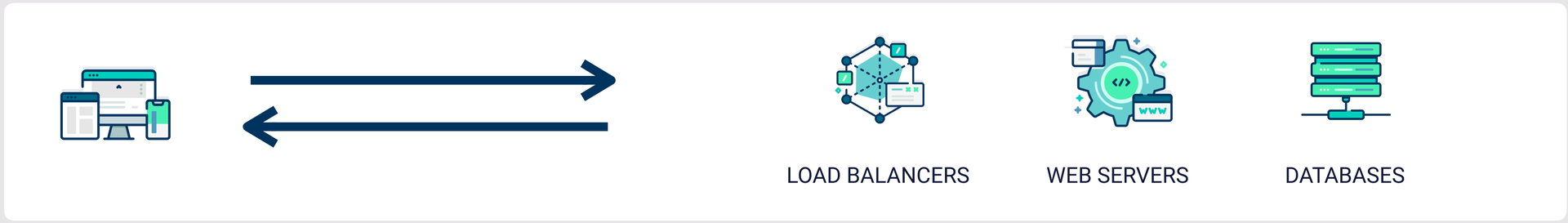
DGIITBWUTFTIRTGIAATWWBE

NAMING IS HARD

Don’t generate it in the build.

Wait until the first time it’s requested, then generate it and add it to what was built earlier.

TRADITIONAL STACK



VARIOUS INCREASINGLY POPULAR PROVIDERS

Increments

DPR / ODB / DSG / ISR

RENDERING

ISR

Incremental Static Regeneration

RENDERING

ISR

Incremental Static Regeneration

SWR

Stale While Revalidate

“Which is better, ISR or DPR”?

— Some of you, possibly, May 2023

“It depends”

Deciding demands questions

What are the requirements?

SECTION 2

Example

Applying different rendering techniques

EXAMPLE

Social posts stash

Self-hosting thousands of posts

Some requirements

A URL for each of the 24,000 tweets

Index pages listing each tweet with its URL

Ability to search the tweets

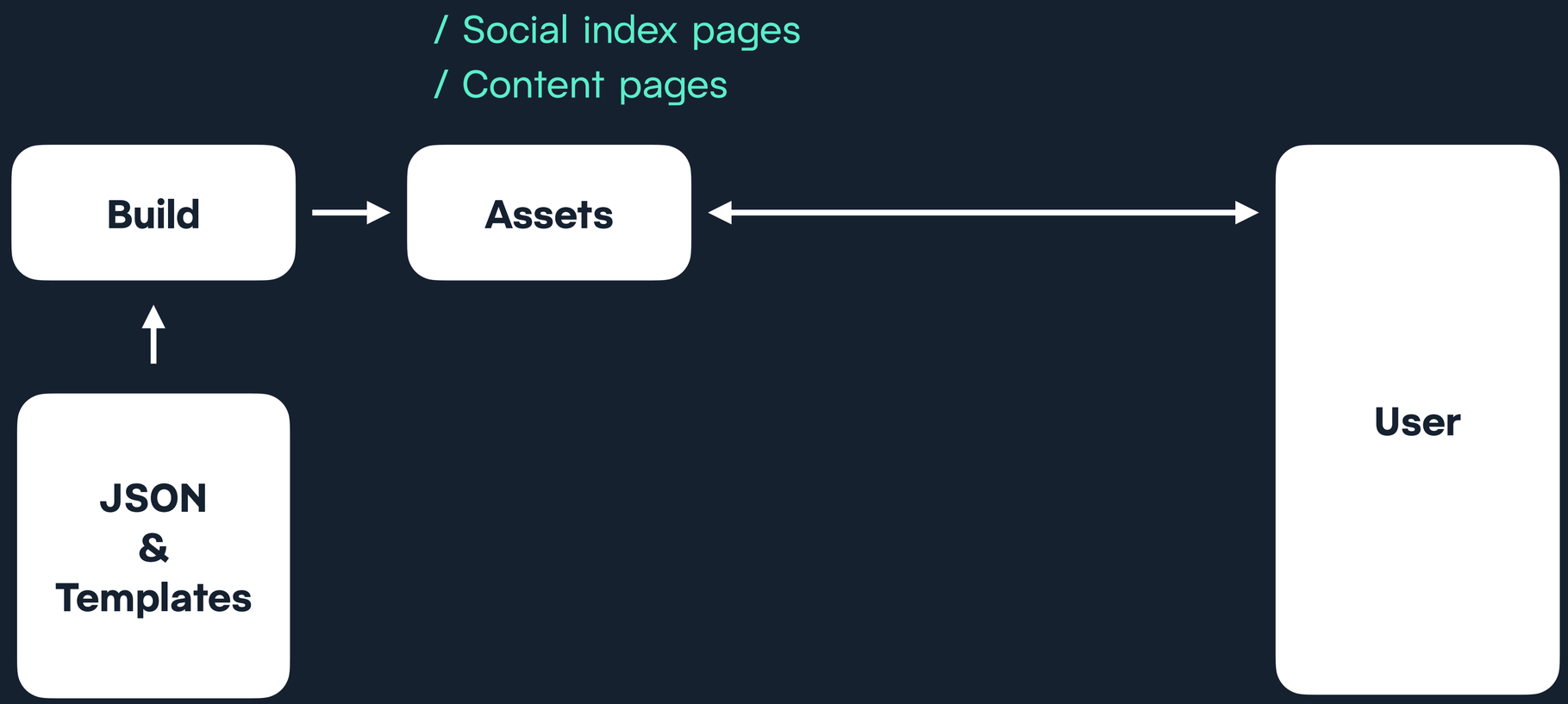
Retain a reasonable build time

Avoid client-side rendering if possible

A logical model that fits in my head

SSG ODB ESR


JSON


/ Social index pages
/ Content pages
Build
Assets
User
JSON
&
Templates

https://www.hawksworx.com

hawksworx blog speaking about search

HAWKSWORX

/hɔːkswəːks/

The blog, playground, and generally a home on the web for Phil Hawksworth, a web nerd, amateur comedian, and cat botherer.

The source code of this site is available on **GitHub** and is hosted and updated by **Netlify** automatically after each **code commit**

Subscribe to a feed of blog posts on this site.

www.hawksworx.com

You can usually find me on Mastodon (I've stopped posting on Twitter now, but you can find an archive of my tweets on this site).

You can also sometimes find me at web development conferences, where I might be talking about development techniques. I blog here less frequently than I'd like, but manage it a little more often on the Netlify blog.

Wherever you find me, online or in the real world, please do come and say a friendly hello.

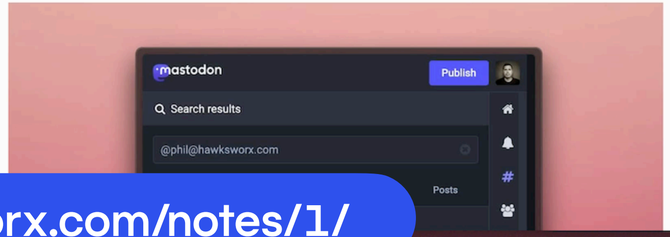
Recently on the blog

December 12th 2022

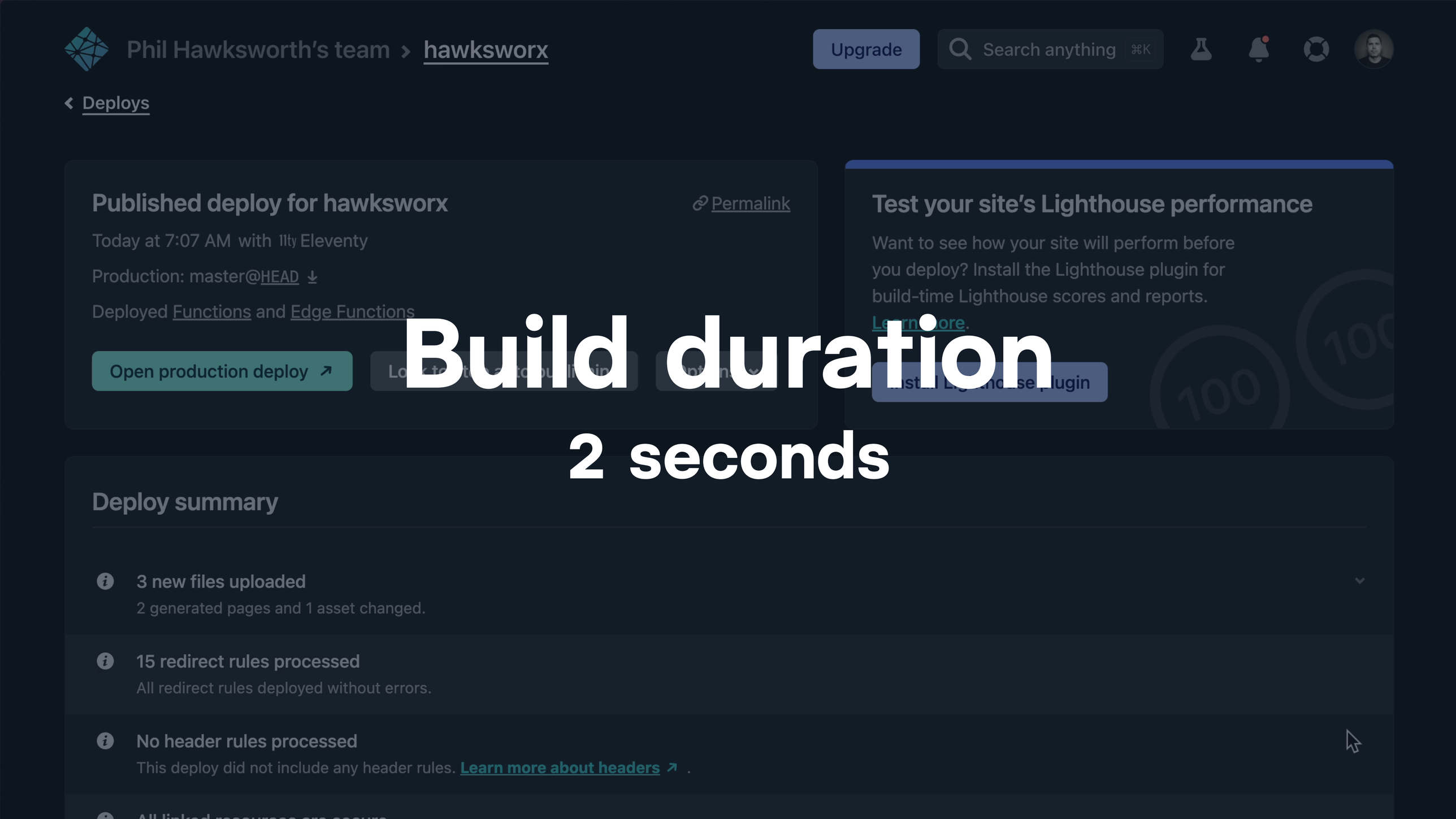
Alias your mastodon account to your own domain with Netlify redirect

#Mastodon , #Fediverse , #Netlify , #Indieweb

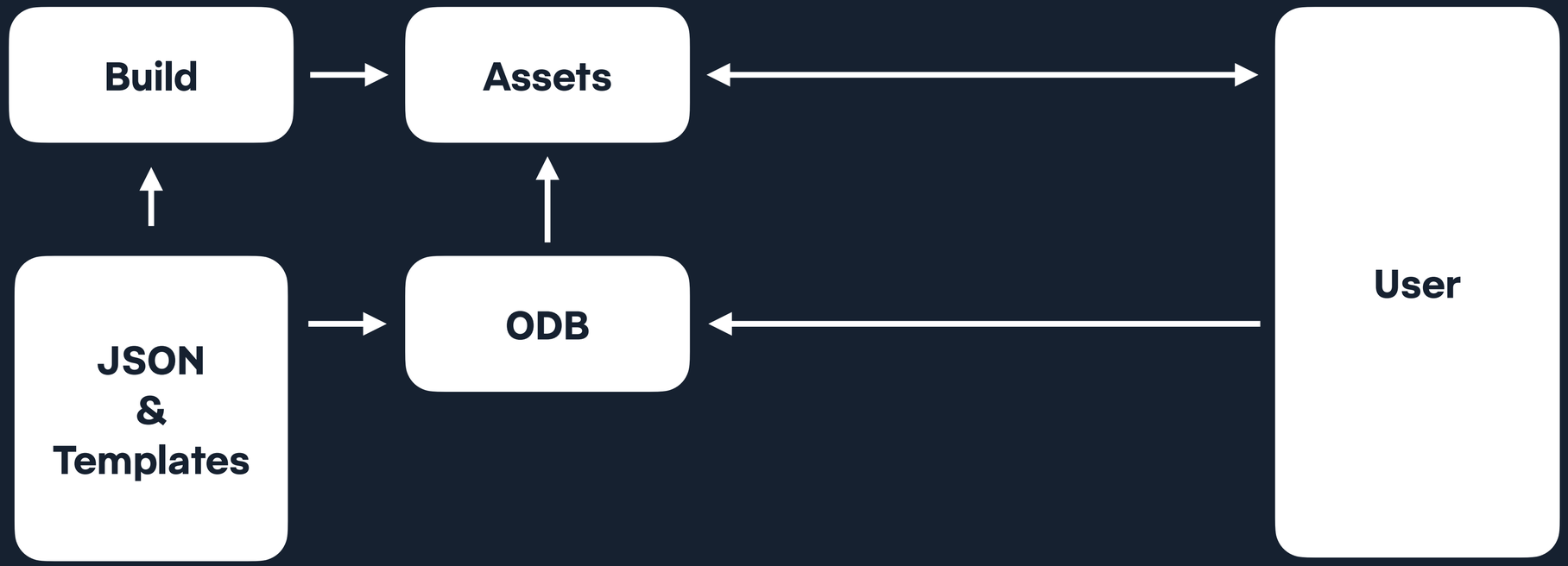
Did you know you could use your own domain for your mastodon username without hosting your own instance? It can be done with a single Netlify redirect rule.



www.hawksworx.com/notes/1/


Phil Hawksworth's team › hawksworx
Upgrade
Search anything ⌘K
Deploys
Published deploy for hawksworx
Permalink
Today at 7:07 AM with Eleventy
Production: master@HEAD
Deployed Functions and Edge Functions
Open production deploy
Test your site's Lighthouse performance
Want to see how your site will perform before you deploy? Install the Lighthouse plugin for build-time Lighthouse scores and reports.
Learn more.
Install Lighthouse plugin
Build duration
2 seconds
Deploy summary
3 new files uploaded
2 generated pages and 1 asset changed.
15 redirect rules processed
All redirect rules deployed without errors.
No header rules processed
This deploy did not include any header rules. Learn more about headers .

/ Post page view
/ Social index pages
/ Content pages



https://www.hawksworx.com/notes/955/

hawksworx blog speaking about search

NOTES - PAGE 955

Newest | Previous | Next | Oldest

Search the notes

The archive of what I posted on Twitter, which I now self host due to a lack of trust in Twitter and some other reasons.

I'll soon begin refelcting all my Mastodon posts here too. I'm happier self-hosting or maintaining an archive of my content on URLs that I can own.

There are tools to help you do this too. Such as this one from the makers of Eleventy.

Phil Hawksworth @philhawksworth • July 23rd 2007

Hacking a rails application - prototypetastic

Permalink | Twitter

Phil Hawksworth @philhawksworth • July 23rd 2007

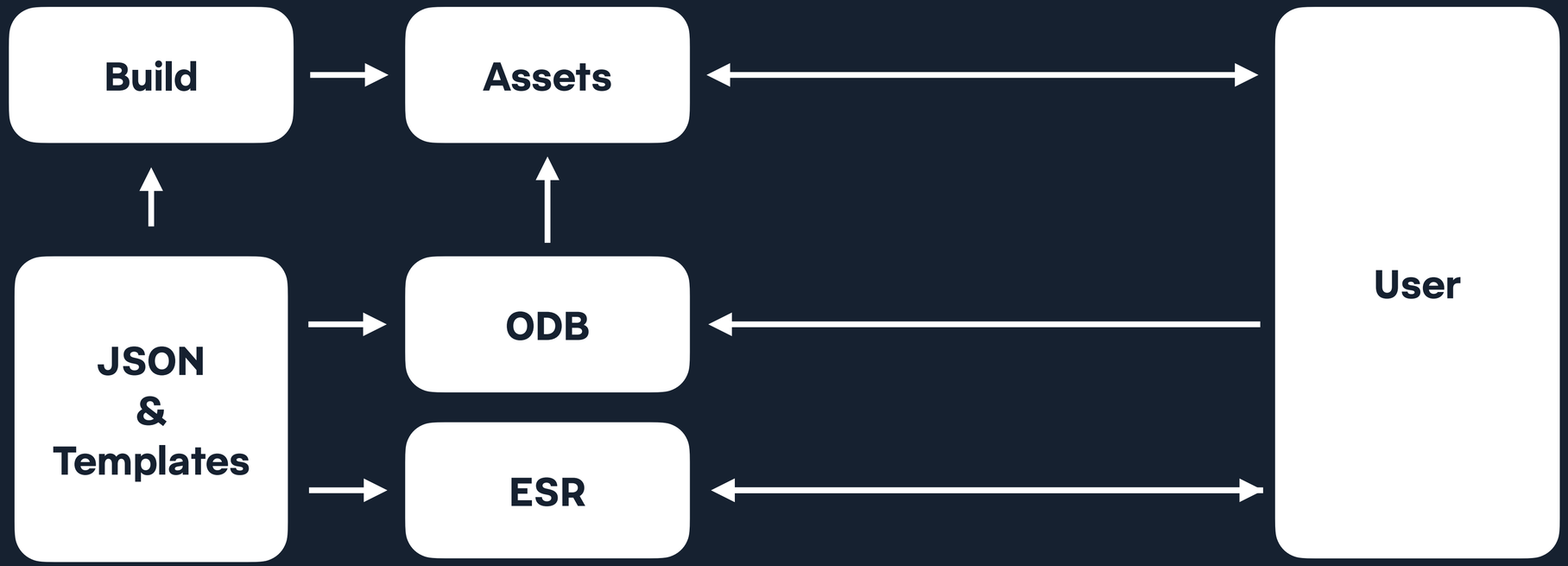
Making content changes directly on my live web server... easy does it!

Permalink | Twitter

Phil Hawksworth @philhawksworth • July 23rd 2007

www.hawksworx.com/note/mstdn/109913367394738833

/ Post page view
/ Social index pages
/ Content pages



/ Search results page

https://www.hawksworx.com/notes/1/

hawksworx blog speaking about search

NOTES

Newest | Previous | Next | Oldest

Search the notes

The archive of what I posted on Twitter, which I now self host due to a lack of trust in Twitter and some other reasons.

I'll soon begin refelcting all my Mastodon posts here too. I'm happier self-hosting or maintaining an archive of my content on URLs that I can own.

There are tools to help you do this too. Such as this one from the makers of Eleventy.

Phil Hawksworth @philhawksworth •
February 24th 2023

Quietly making a resolution to get my use of exclamation marks under control!

Dammit.

Permalink | Mastodon

Phil Hawksworth @philhawksworth •
February 23rd 2023

Started my day by making the *perfect* piece of hot buttered toast.
This made me happy.
End of message.

Permalink | Mastodon

www.hawksworx.com/notes/search/?str=render


Voilà
Ai Artist


Voilà
Ai Artist

The rule of least power

https://www.w3.org/2001/tag/doc/leastPower.html

Vitepress

Generates pages

SSG / CSR

Vitepress

Generates pages

SSG / CSR + ODB / ESR

At build time

At request time

Source archive of data into Vitepress

tweets.data.js

```
export default {
  load() {
    return { LOTS OF JSON }
  }
}
```

Source archive of data into Vitepress

tweets.data.js

```
const tweets = require(“./tweets.json");

export default {
  load() {
    return tweets
  }
}
```

ODB function — render a tweet

netlify/functions/view-tweet.js

```
const { builder } = require('@netlify/functions');
const notes = require('../../tweets.json');
const pageTemplate = require('./page/template.js');

const handler = async(event) => {
    const id = event.path.split("tweet/")[1];
    const note = notes[id];
    return {
      statusCode: 200,
      headers: {"Content-Type": "text/html"},
      body: pageTemplate(note)
    }
}
exports.handler = builder(handler);
```

ODB function — render a tweet

netlify/functions/view-tweet.js

```
const { builder } = require('@netlify/functions');
const notes = require('../../tweets.json');
const pageTemplate = require('./page/template.js');

const handler = async(event) => {
    const id = event.path.split("tweet/")[1];
    const note = notes[id];
    return {
      statusCode: 200,
      headers: {"Content-Type": "text/html"},
      body: pageTemplate(note)
    }
}
exports.handler = builder(handler);
```

ODB function — render a tweet

netlify/functions/view-tweet.js

```
const { builder } = require('@netlify/functions');
const notes = require('../../tweets.json');
const pageTemplate = require('./page/template.js');

const handler = async(event) => {
    const id = event.path.split("tweet/")[1];
    const note = notes[id];
    return {
      statusCode: 200,
      headers: {"Content-Type": "text/html"},
      body: pageTemplate(note)
    }
}
exports.handler = builder(handler);
```

ODB function — render a tweet

netlify/functions/view-tweet.js

```
const { builder } = require('@netlify/functions');
const notes = require('../../tweets.json');
const pageTemplate = require('./page/template.js');

const handler = async(event) => {
    const id = event.path.split("tweet/")[1];
    const note = notes[id];
    return {
      statusCode: 200,
      headers: {"Content-Type": "text/html"},
      body: pageTemplate(note)
    }
}
exports.handler = builder(handler);
```

ODB function — render a tweet

netlify/functions/view-tweet.js

```
const { builder } = require('@netlify/functions');
const notes = require('../../tweets.json');
const pageTemplate = require('./page/template.js');

const handler = async(event) => {
    const id = event.path.split("tweet/")[1];
    const note = notes[id];
    return {
      statusCode: 200,
      headers: {"Content-Type": "text/html"},
      body: pageTemplate(note)
    }
}
exports.handler = builder(handler);
```

ODB function — render a tweet

netlify/functions/view-tweet.js

```
const { builder } = require('@netlify/functions');
const notes = require('../../tweets.json');
const pageTemplate = require('./page/template.js');

const handler = async(event) => {
    const id = event.path.split("tweet/")[1];
    const note = notes[id];
    return {
      statusCode: 200,
      headers: {"Content-Type": "text/html"},
      body: pageTemplate(note)
    }
}
exports.handler = builder(handler);
```

ODB function — render a tweet

netlify/functions/view-tweet.js

```
const { builder } = require('@netlify/functions');
const notes = require('../../tweets.json');
const pageTemplate = require('./page/template.js');

const handler = async(event) => {
    const id = event.path.split("tweet/")[1];
    const note = notes[id];
    return {
      statusCode: 200,
      headers: {"Content-Type": "text/html"},
      body: pageTemplate(note)
    }
}
exports.handler = builder(handler);
```

Nuxt

Supports several rendering patterns

SSG / SSR / CSR / ESR

Nuxt’s route-based rendering configuration

nuxt.config.ts

```ts
export default defineNuxtConfig({
  routeRules: {

    // Homepage pre-rendered at build time
    '/': { prerender: true },

    // Product page generated on-demand, revalidates in background
    '/products/**': { swr: true },

    // Blog post generated on-demand once until next deploy
    '/blog/**': { isr: true },

    // Admin dashboard renders only on client-side
    '/admin/**': { ssr: false },
  }
})
```

SECTION 3

Lessons

Amid the confusion, what did we learn?

There is no one right way

Do work ahead of time if you can

You can mix and match rendering methods

Never choose an approach until you understand the requirements

Not all of Phil's
tweets are pure gold

NAMING IS HARD

For more

netlify.com/blog/tutorials

nuxt.com/docs/guide/concepts/rendering

twitter.com/philhawksworth

Phil Hawksworth, Netlify

Thank you!

Grab me for questions (or to say hello)

Phil Hawksworth, Netlify