Jamstack at Scale: How Development Teams Build for Audiences of Millions

A COMPREHENSIVE GUIDE

Table of Contents

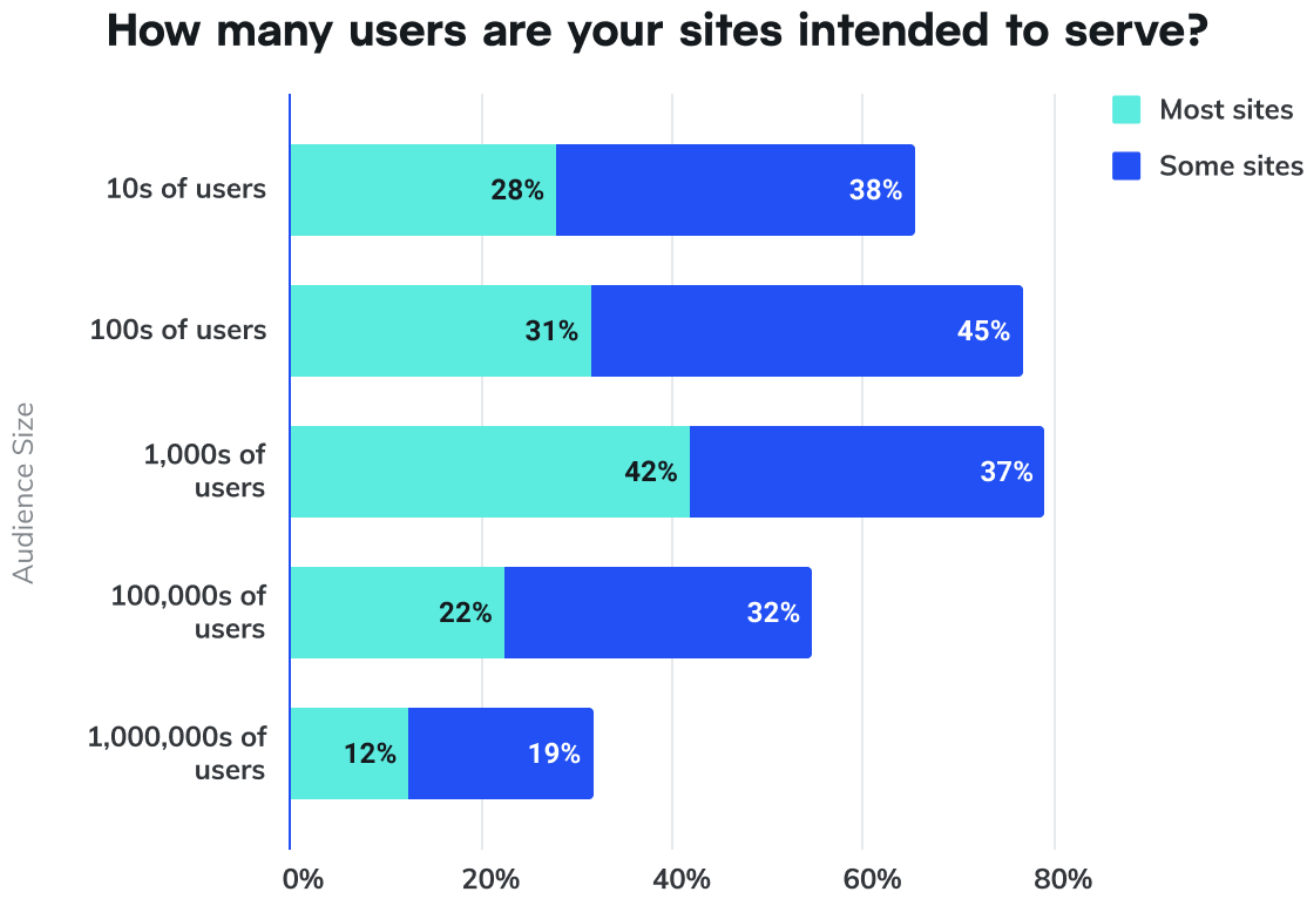
INTRODUCTION

Methodology

Introduction

Earlier this year we released the results of our Jamstack Community Survey, a comprehensive look at the current practices and preferences of Jamstack developers around the world.

Jamstack developers build sites of all kinds for all types of audiences. When analyzing this year's data, we were pleased to note that the Jamstack is used to build some of the biggest websites in the world, with audiences measured in the millions. Fully 32% of our survey respondents said they had worked on a site of this kind in the last year.



We decided to look more closely at this group and ask some questions. How do developers who work on sites with very large audiences differ from other developers? Do they have different priorities? Do they prefer a different technical stack? Do they have different levels of experience? The answers were: all of the above, and more.

Methodology

For this analysis, we compared three groups, split by how frequently they said they work on websites with audiences of millions of users. For brevity we'll be referring to those as "very large websites", though a website that's big in terms of audience doesn't necessarily imply the website is large in terms of how many pages it has. The three groups are:

- People who **never** work on very large sites
- People who **sometimes** work on very large sites
- People who **mostly** work on very large sites

We also filtered respondents to exclude students and other people not currently employed, for a more apples-to-apples comparison of job titles and professional preferences.

SECTION I

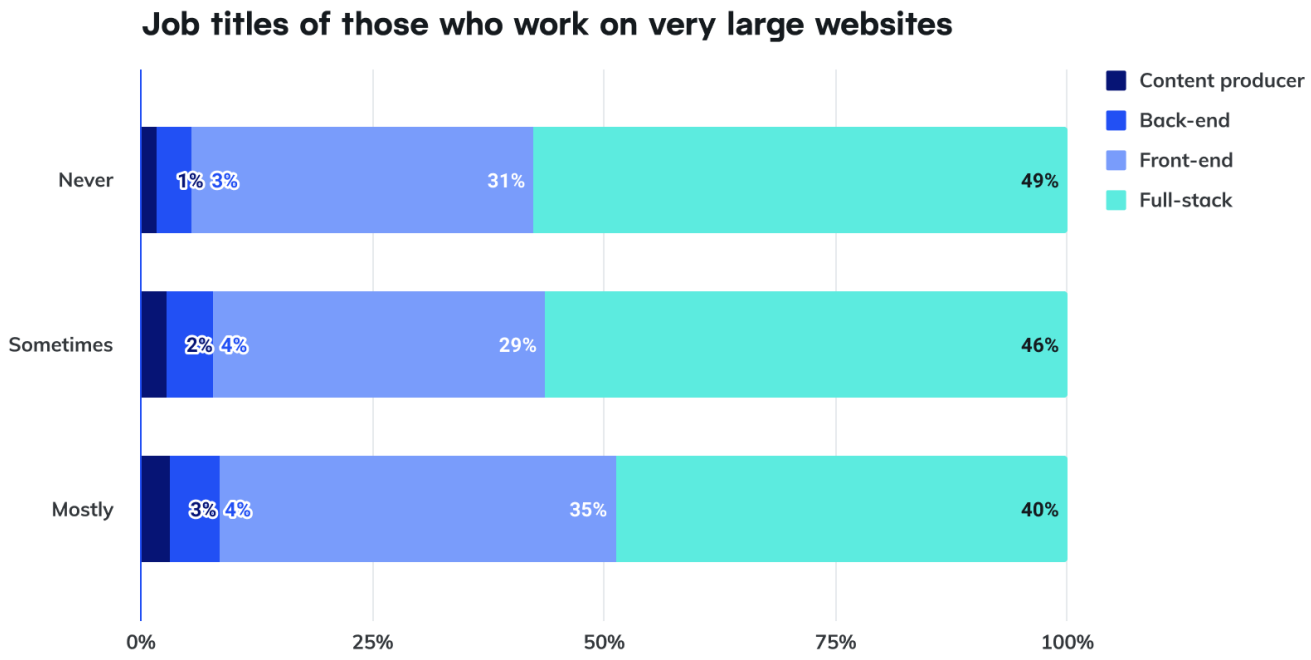
Developer Demographics

Developer Demographics

There are two ways to answer the question of how development differs at scale: how do the developers themselves differ, and how does their work differ? Let's first examine how the developers themselves are different.

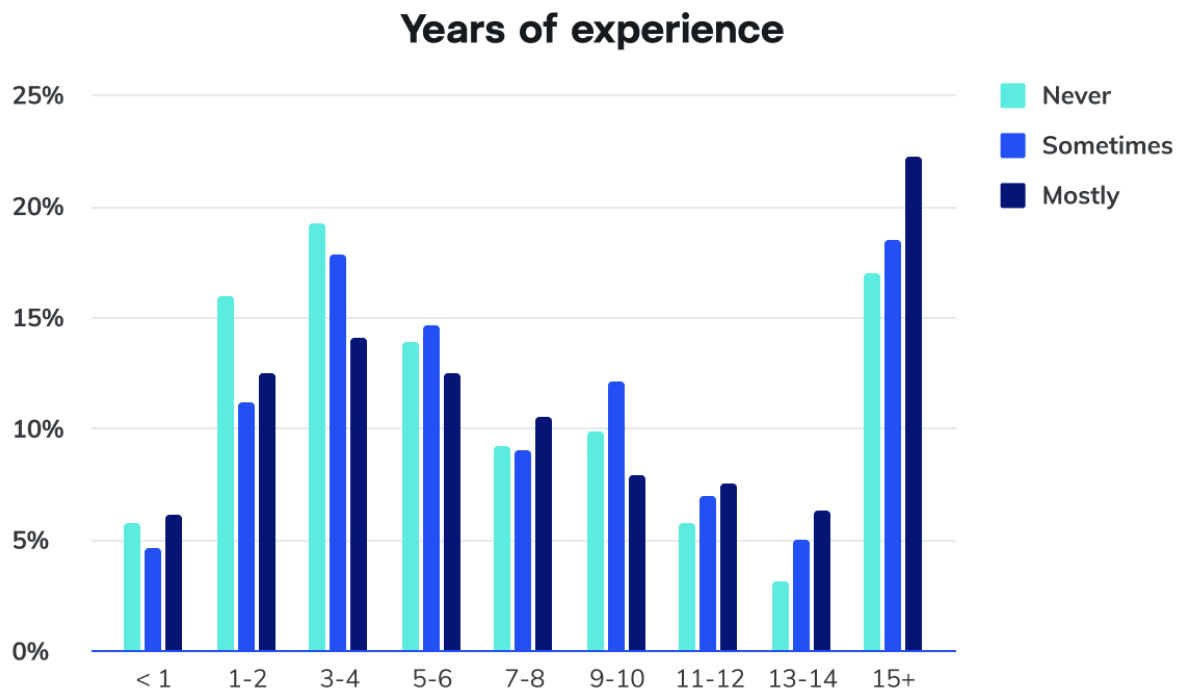
Large websites are built by teams of specialists

Our first discovery was that there is a trend in job titles across these groups:



People who work on very large sites are less likely to call themselves "full stack" developers. Instead, the more specialized "front-end" and "back-end" developers become more common, and the "content producer" job title also becomes more frequent. Unsurprisingly, large websites are built by larger teams, and larger teams have more specialized roles.

Large teams are working remotely successfully



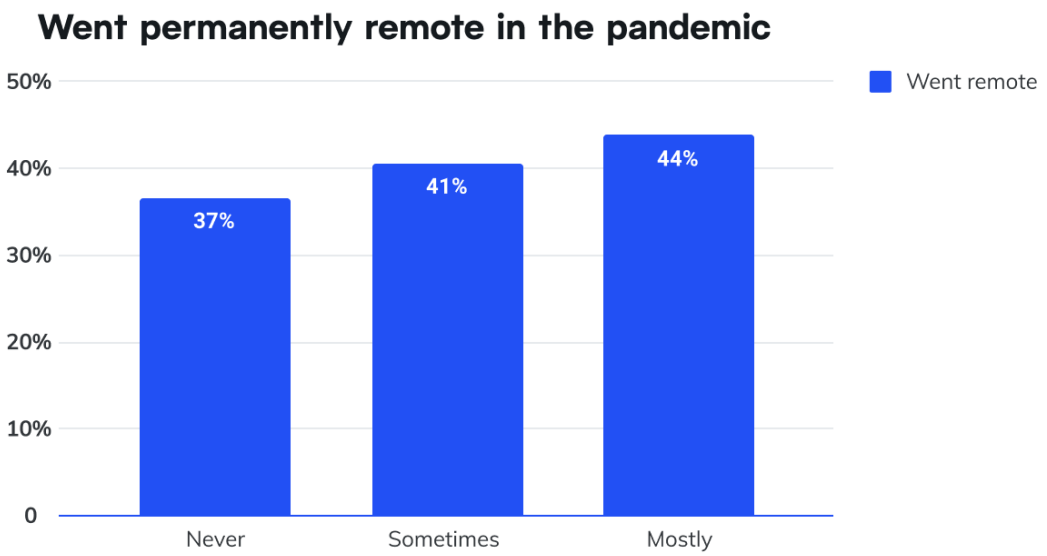
Source: Jamstack Community Survey 2021

Another intuitive finding is that larger websites are also built by more experienced people. Developers who say they “mostly” build sites for audiences of millions are less common in the groups with 5 or fewer years of experience, but in the category of developers with 15+ years of experience they are the biggest group.

We know from our earlier explorations of this data that more experienced developers were more likely to have gone permanently remote in the upheaval caused by the global pandemic. This holds true for developers working on very large sites, showing that **remote work is not a barrier to working at scale.**

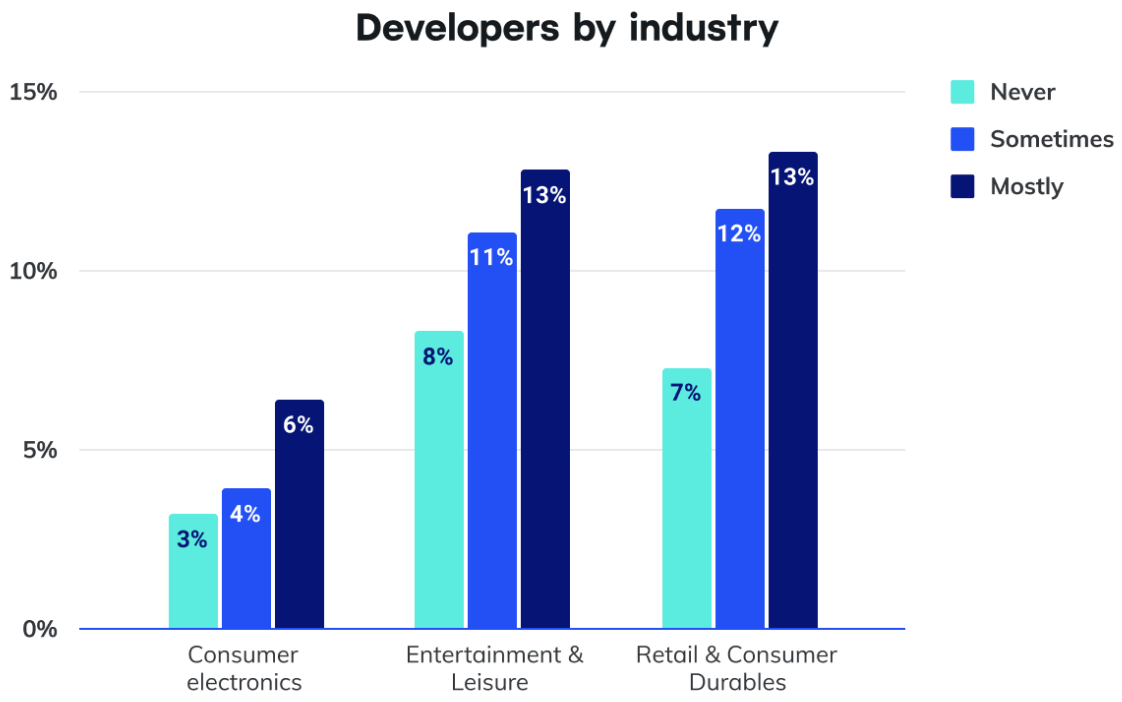
Large sites are more frequent in a handful of industries

Another question we had was: are very large sites, as defined by audience size, concentrated in a particular industry? If you were a developer wanting to work at scale, where should you go?



Source: Jamstack Community Survey 2021

There were three industries with a clear trend towards having more developers working at large scale: consumer electronics, entertainment, and retail. In all three cases, a developer working in that industry is 1.5-2x more likely to be “mostly” working on very large websites.



Source: Jamstack Community Survey 2021

SECTION II

Technical choices

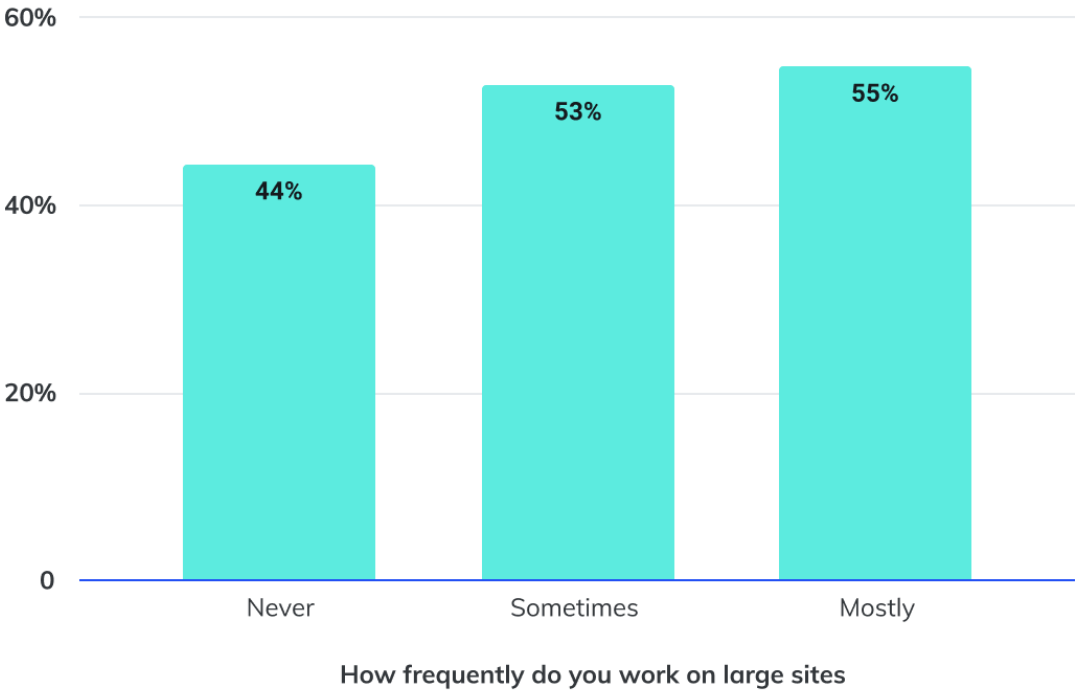
Technical choices

Developers who work at scale are different: more specialized, more experienced, and more likely to work in some industries than others. What about the sites themselves? They are also different in important ways.

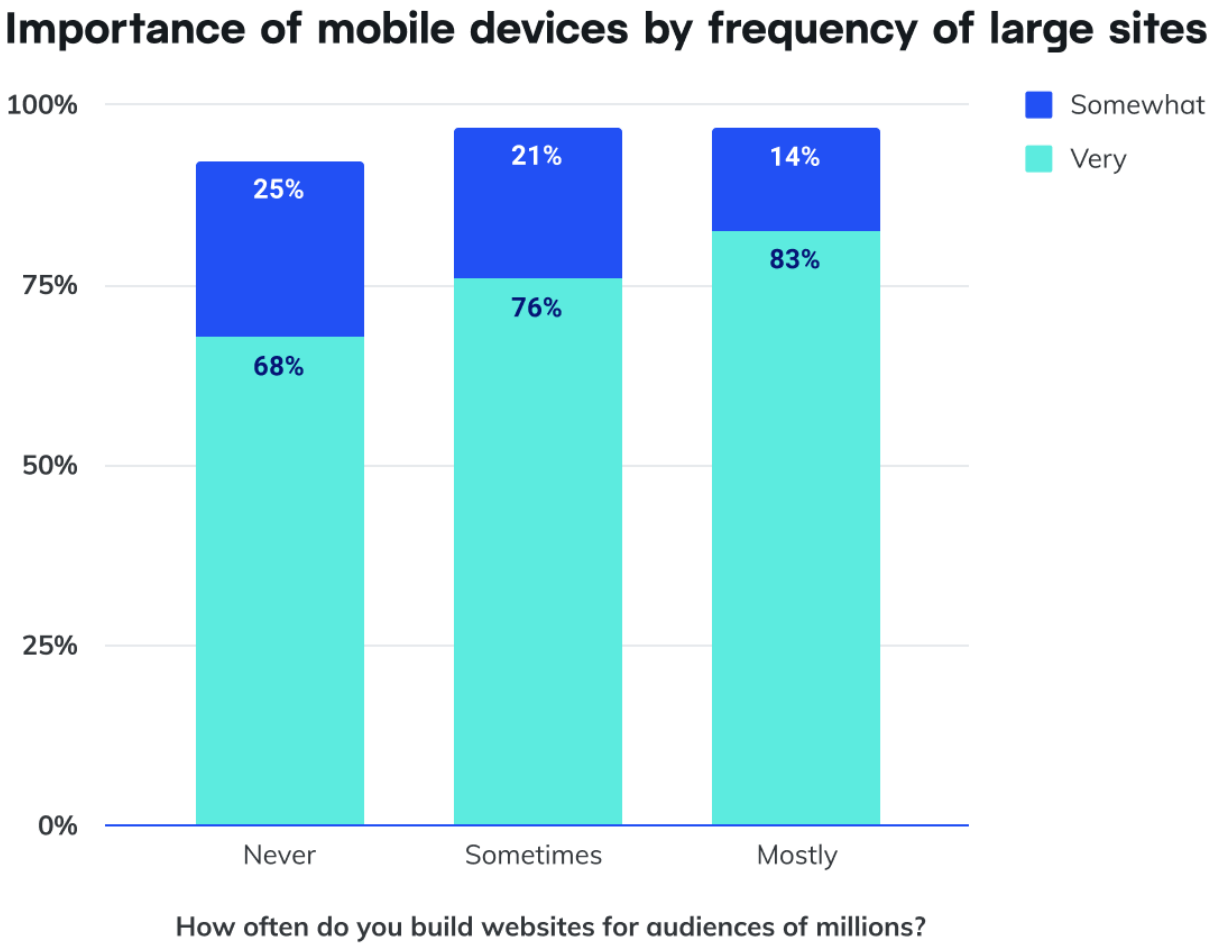
Large sites are more likely to be dynamic sites

We looked at the types of web applications built by people who work on very large sites. We discovered **the more likely you are to build sites for very large audiences, the more likely you are to be building fully dynamic sites**. We saw a similar but smaller trend for "mostly static sites with some dynamic elements" but not for "fully static sites". Our conclusion is that **the larger your audience, the more necessary it is to customize the experience to satisfy so many different people.** Given the concentration of large sites in the Retail sector, where pages are often highly customized with purchasing recommendations, this makes sense.

Jamstack developers who "frequently" build fully dynamic sites



Mobile apps were also a higher priority for the group who most frequently build large websites. Again, if you're trying to reach a very wide audience, making mobile a priority makes sense.

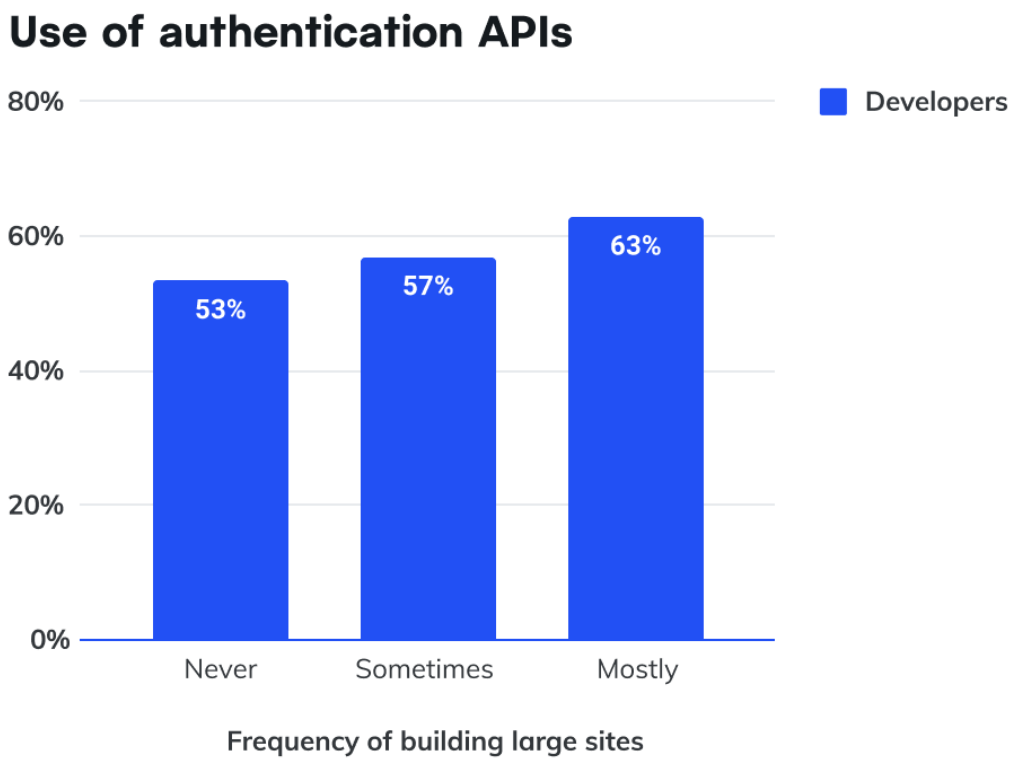


Techniques + Tooling for Large Sites

There are a number of techniques and tooling choices that show a trend when looking at how frequently users work on large sites.

Large Sites Support 3rd Party Authentication APIs

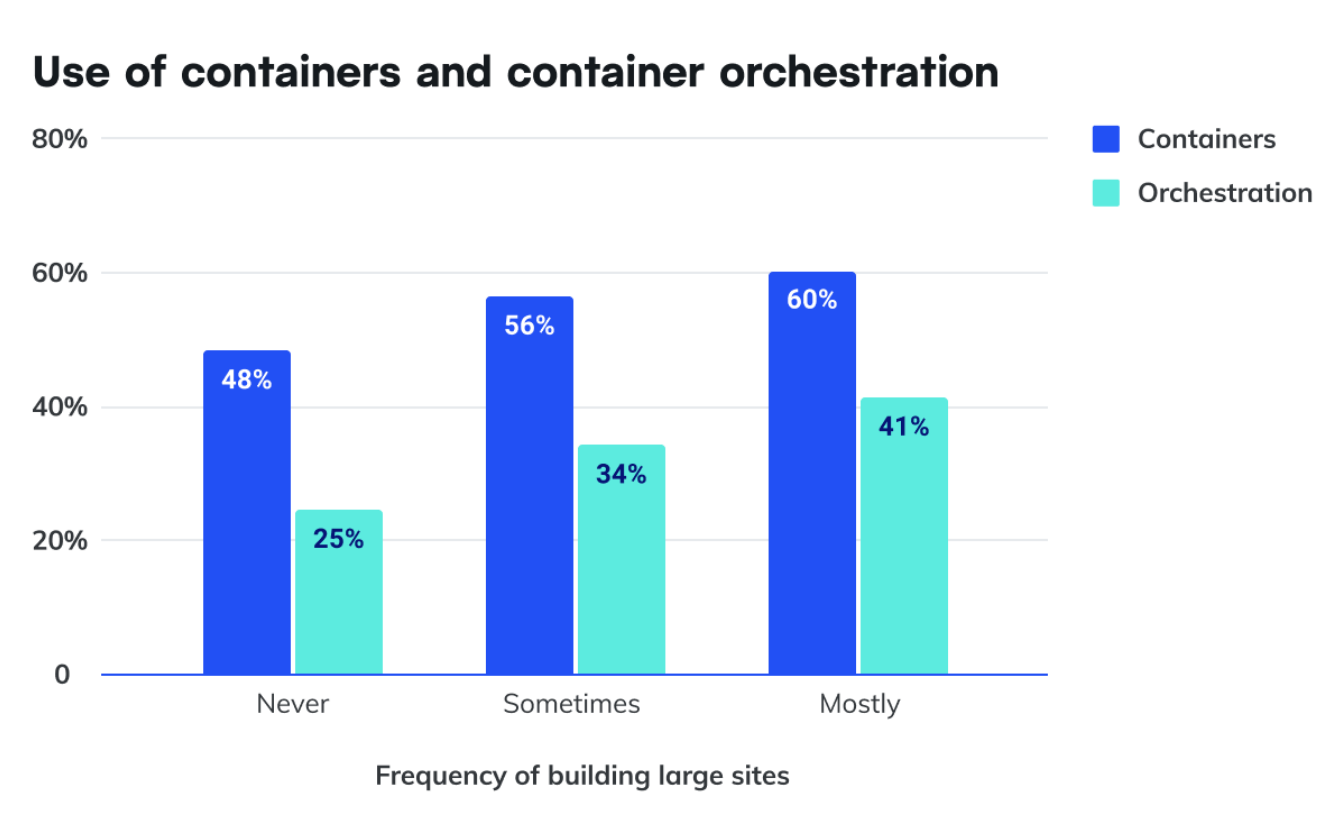
The first was authentication APIs:



One might have expected that larger sites would be more likely to “roll their own” authentication but it appears the opposite is true: **those who work with large audiences are more likely to use a third-party authentication service.** One explanation is that scaling up to large audiences requires reducing friction, so in addition to having your home-grown signup solution you might also support third-party authentication.

Large Sites Use Containers and orchestration

Two closely related tooling choices, containers and container orchestration, showed similar trends towards higher usage amongst those building larger sites.



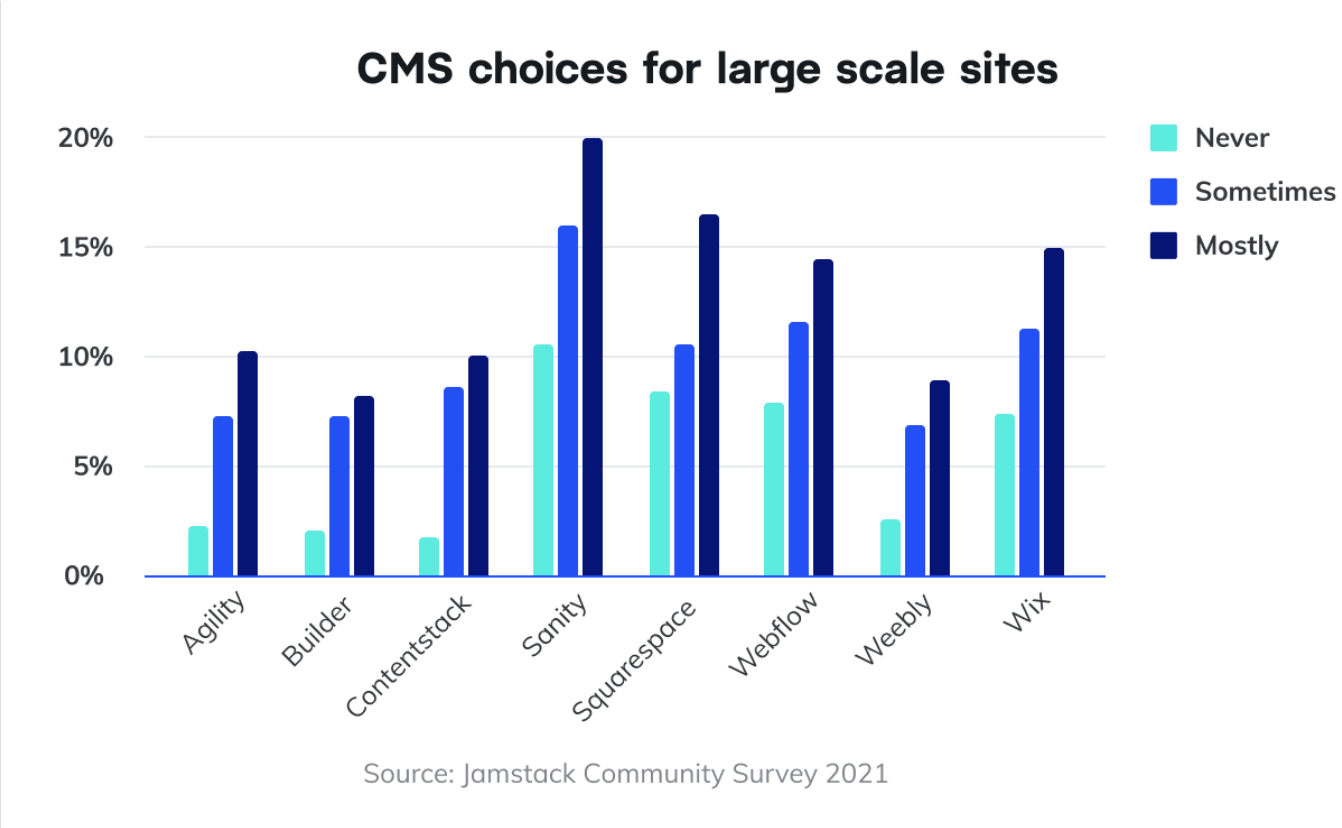
The correlation here seems pretty easy to explain: larger sites, which we already know are more likely to be partly or fully dynamic, have more complex back-ends, requiring more sophisticated tooling. We saw a similar trend in the use of microservices, which are a likely user of containers.

Sanity challenges WordPress for large sites

From our earlier explorations of survey data, we already knew overall trends in CMS preferences: WordPress is still a strong leader in the space, with Contentful and Strapi the next two most popular choices. Of particular interest to us was whether those choices changed depending on the scale of the sites being built.

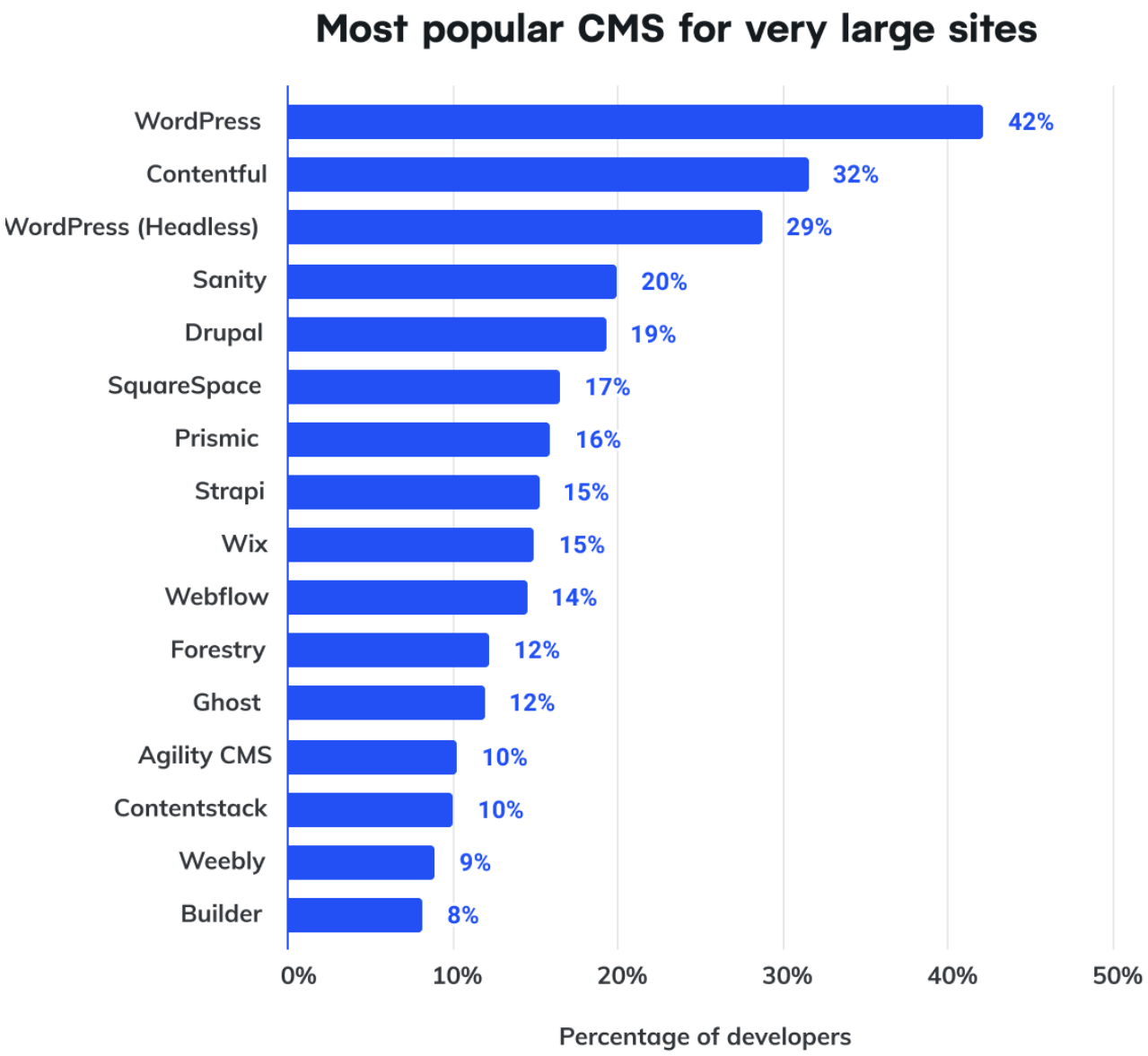
Out of 16 different CMS options we surveyed, we found 8 that showed a positive trend associated with working on larger websites. Many of these choices more than double their market share going from the group who “never” build large sites to the group who “mostly” build large sites.

Four of these choices are closely related: Squarespace, Webflow, Weebly and Wix all have in common that in addition to content management, they also have strong design and templating functions. We connected this preference to the greater presence of people whose job title is content specialist, discussed earlier. These specialists have less technical expertise coupled with a greater desire to control their output, making these choices a good fit for them.



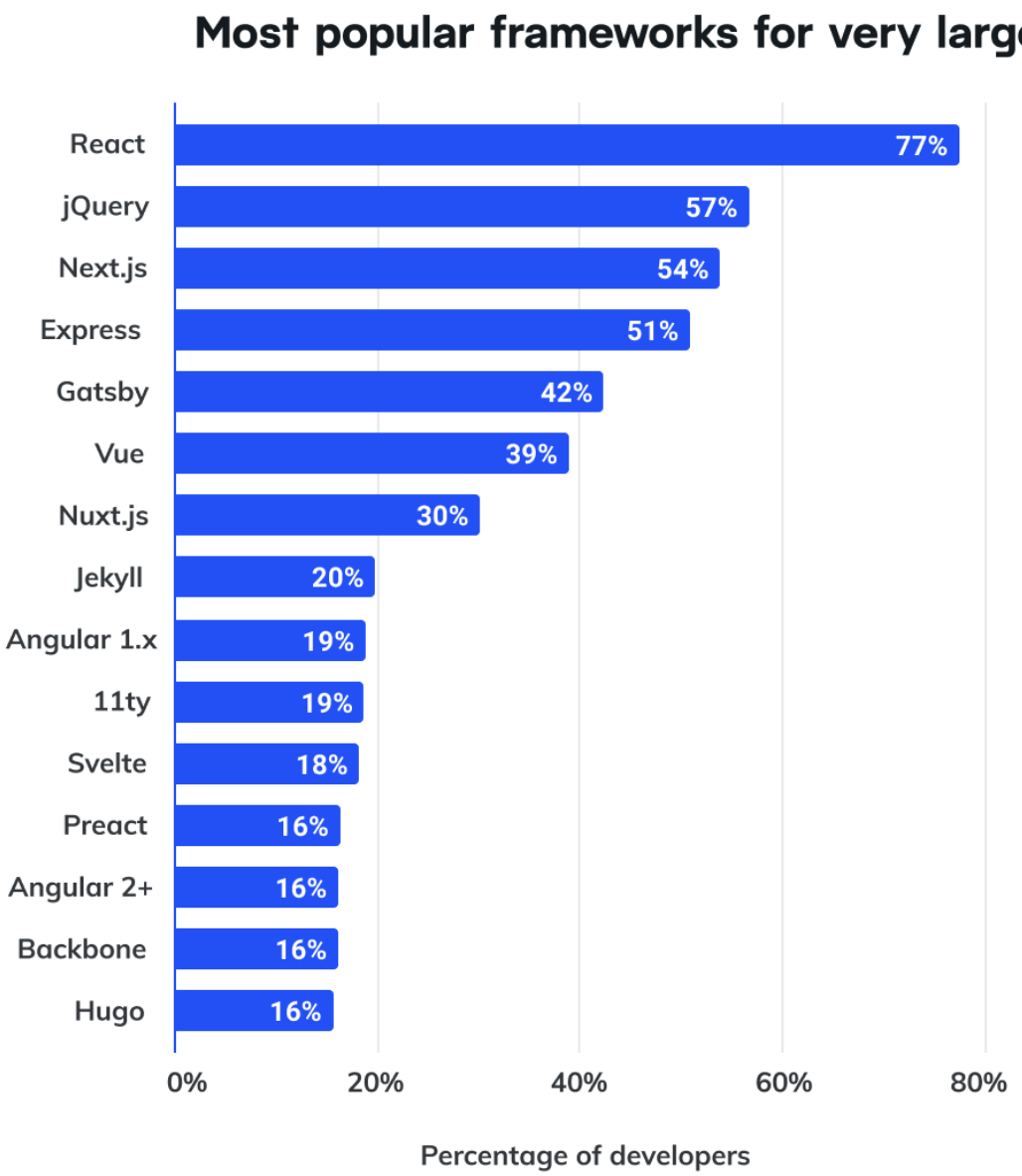
Source: Jamstack Community Survey 2021

It's important not to over-state these findings: **WordPress is still the most popular CMS choice, even amongst the largest sites**, with 42% usage. However **Sanity at 20% usage is a much stronger competitor to WordPress on large-scale websites** than it is overall (13%), which is an interesting finding that we will keep an eye on for future surveys.

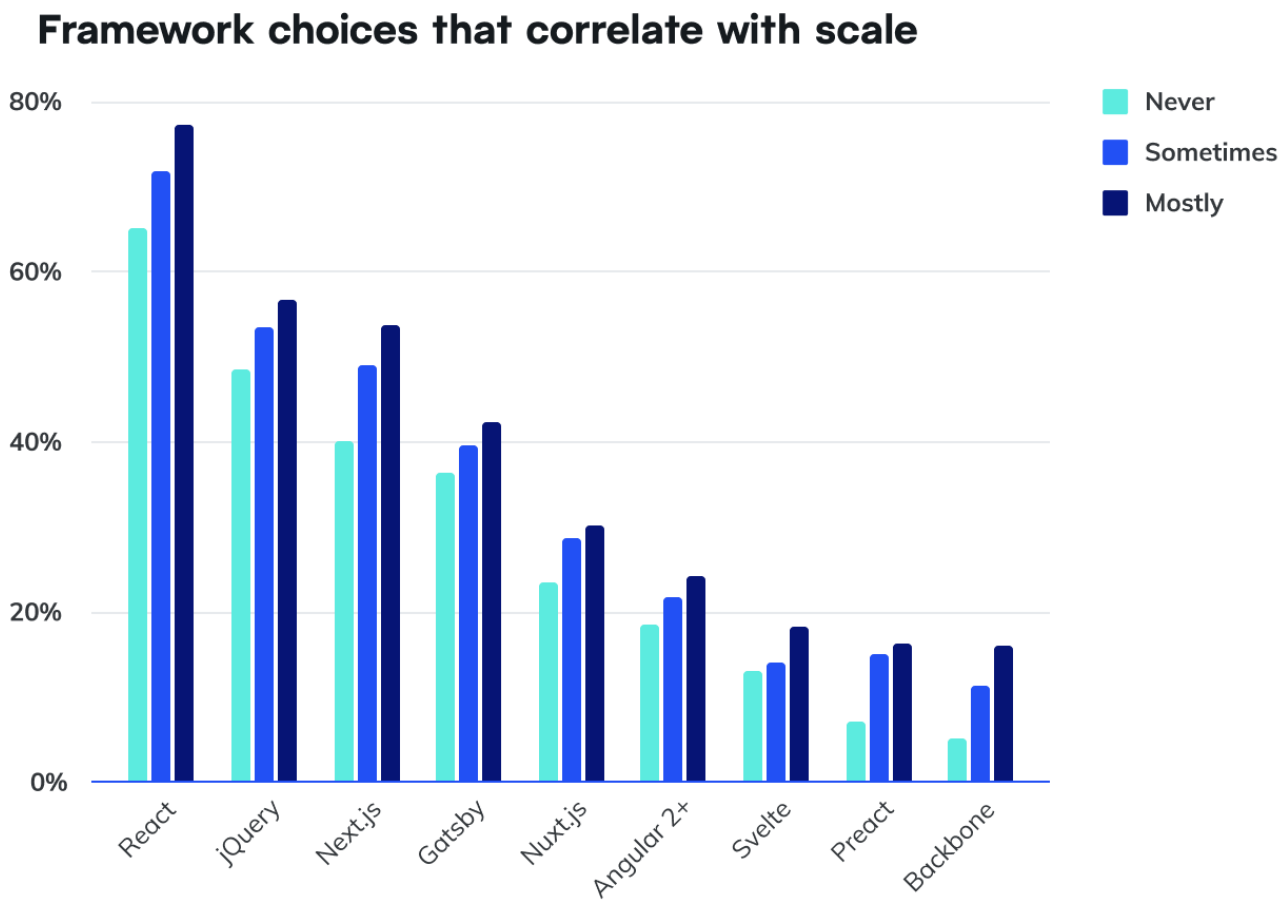


Top Frameworks include React, jQuery, and Next.js

We applied an analysis similar to the one we did for content management systems to web frameworks, looking for correlations to scale. As with CMS, the overall rankings of frameworks did change, but not by much: React remains an enormously popular choice, and jQuery is even more popular among large sites at 57% than it is in general (51%).



What about frameworks that correlate with scale? We did find some, though the change was not as dramatic as when we looked at CMS choices. The top 3 are unchanged: React, jQuery and Next.js. What's almost more interesting is who drops out of the ranking: Express usage does not correlate with increased scale, and neither does usage of Vue, Jekyll, or Angular 1.x. It's hard to find a common quality between those frameworks, so we don't have an explanation for why their popularity might not grow with increased scale.


Framework choices that correlate with scale
80%
60%
40%
20%
0%
Never
Sometimes
Mostly
React
jQuery
Next.js
Gatsby
Nuxt.js
Angular 2+
Svelte
Preact
Backbone

How to Build a Jamstack Website at Scale

It can be an illuminating exercise to take findings and “reverse” them to create advice. If we do that with these findings, you would get the following advice for “how to build a website at scale”:

You should definitely:

1. **Hire a team of experienced specialists; they’ll work remotely.**
2. **Build a site that is mostly or fully dynamic.**
3. **On the front end, you’ll want React and Next.js.**
4. **Make sure you treat mobile devices as first-class targets.**
5. **Allow users to log in with third-party services.**
6. **Deploy and scale your back-end using orchestrated containers.**

In addition to those clear majority preferences, you should also consider:

1. *Attaching WordPress, Contentful or Sanity as your CMS.*
2. *Implementing microservices as a back-end architecture*
3. *Instrumenting monitoring and analytics using an event hub service.*

Obviously this is unrealistically prescriptive; this set of choices wouldn’t work for every use case. But this configuration is nevertheless interesting as a loose definition of what might be considered “industry standard” in 2021. If you find your own large scale site departing from these choices, it doesn’t necessarily mean you’re making a bad choice, but it’s worth talking to your team and being clear about the reasons behind your choices.

Talk to a Jamstack Expert


netlify