ALLOY TECHNOLOGY

A primer on decoupled, headless, Jamstack & more

The journey from monolithic to composable 2024

alloy

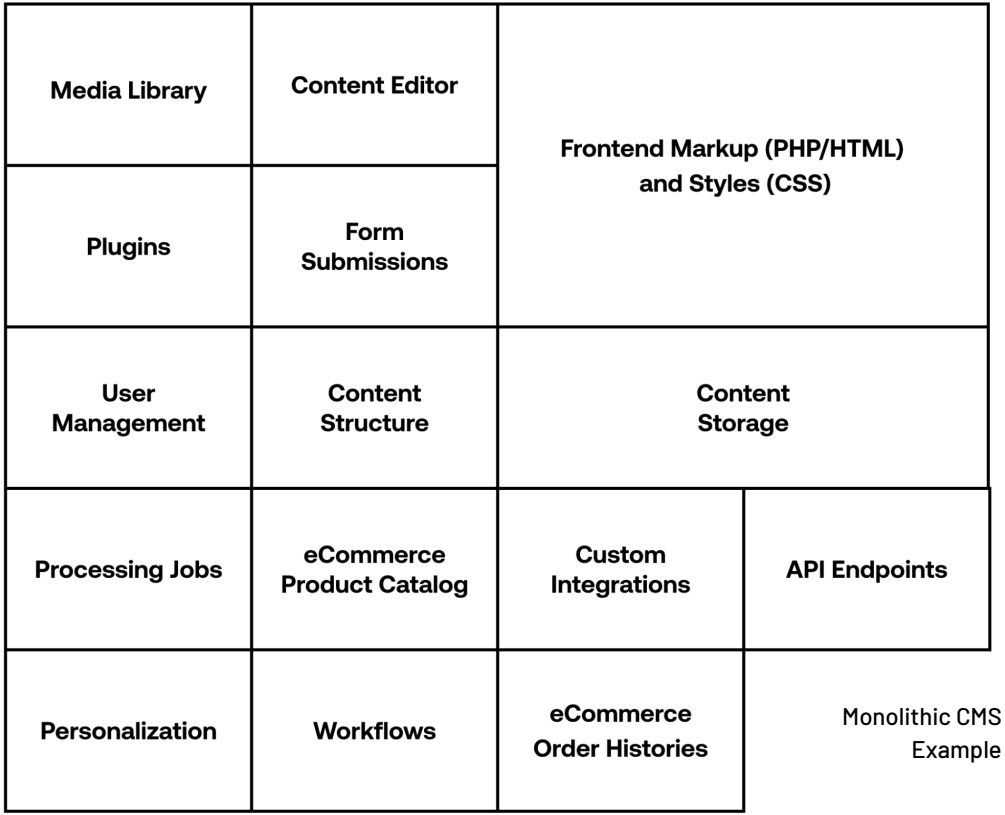
+Contents

What is a monolithic architecture?

Monolithic architectures

A **monolithic architecture** refers to a traditional programming model in which all elements of a system are interwoven and interdependent — a singular and all-encompassing platform.

As such, a **monolithic CMS** is a single platform that handles all aspects of a website, from content management by backend authors to serving pages to website visitors and everything in between.



Monolithic CMS Example

Examples: WordPress, Drupal, Magento, Sitecore

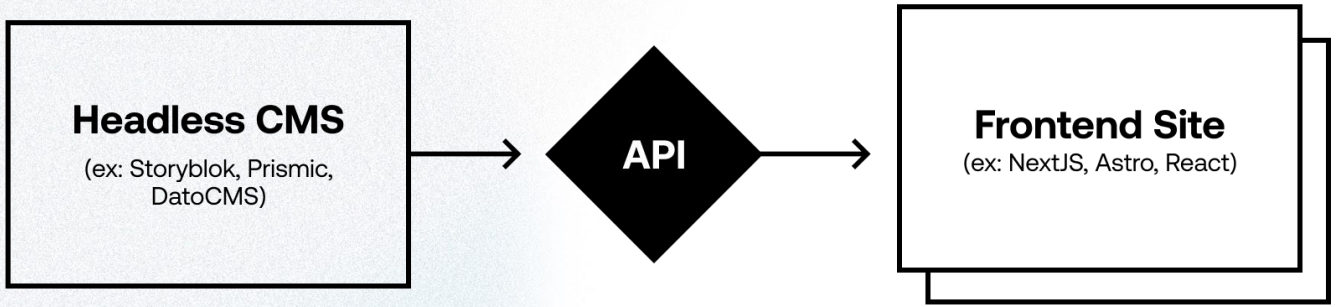
What is a decoupled architecture?

Decoupled architectures

A **decoupled architecture** refers to a modern programming model in which elements of a system are well separated by responsibility and integrated via strongly defined APIs—a set of platforms that are used in combination.

As such, a **decoupled CMS** is a platform that handles content management but not how that content is rendered. Instead, it surfaces content via an API (e.g., REST or GraphQL) to other systems, such as a front-end website or build process.

DECOUPLED CMS EXAMPLE



Examples: Storyblok, DatoCMS, Hygraph, Contentstack, Contentful, Sanity.io, Prismic, ButterCMS

Separation of concerns

*Seh·pr·**ay**·shn of kuhn·**surnz***

A design principle for separating a system into distinct sections by responsibility.

Decoupling is nothing new

HTML

Defines the **elements** on the page

```
<h1>Page Title</h1>
```

CSS

Defines the **Styles** of the elements on the page

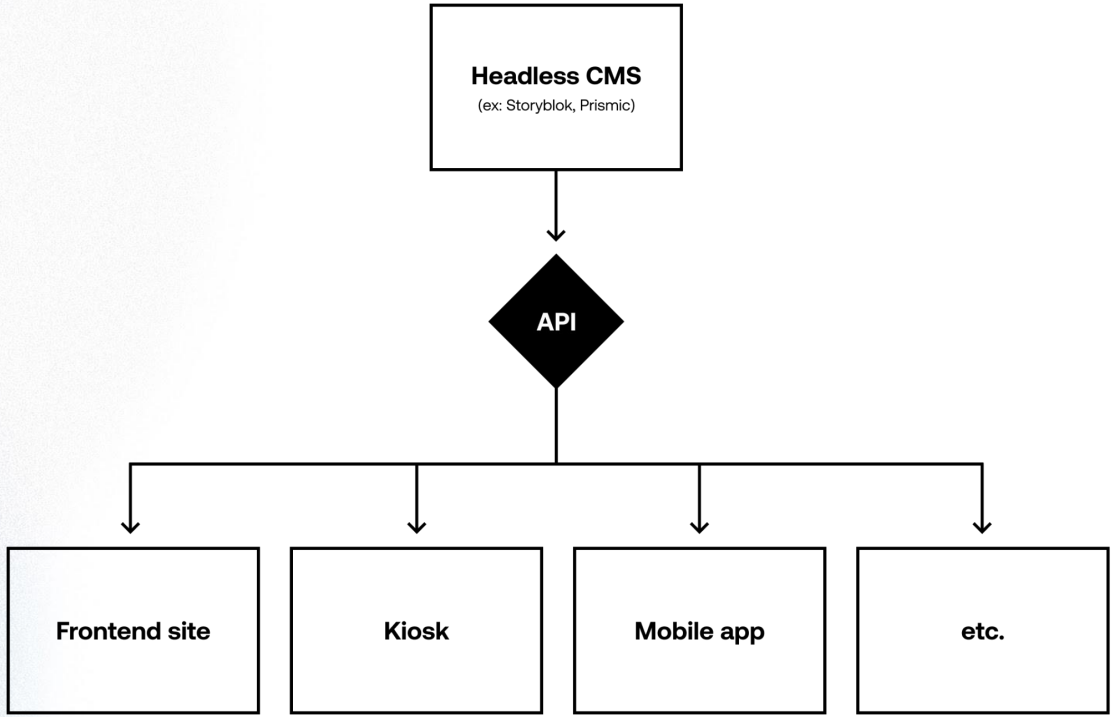
```
h1  {
  font-size: 32px;
}
```

What about “headless” and “composable?”

Decoupled & frontend agnostic

"Headless" and "decoupled" are terms frequently used interchangeably, but there is some nuance.

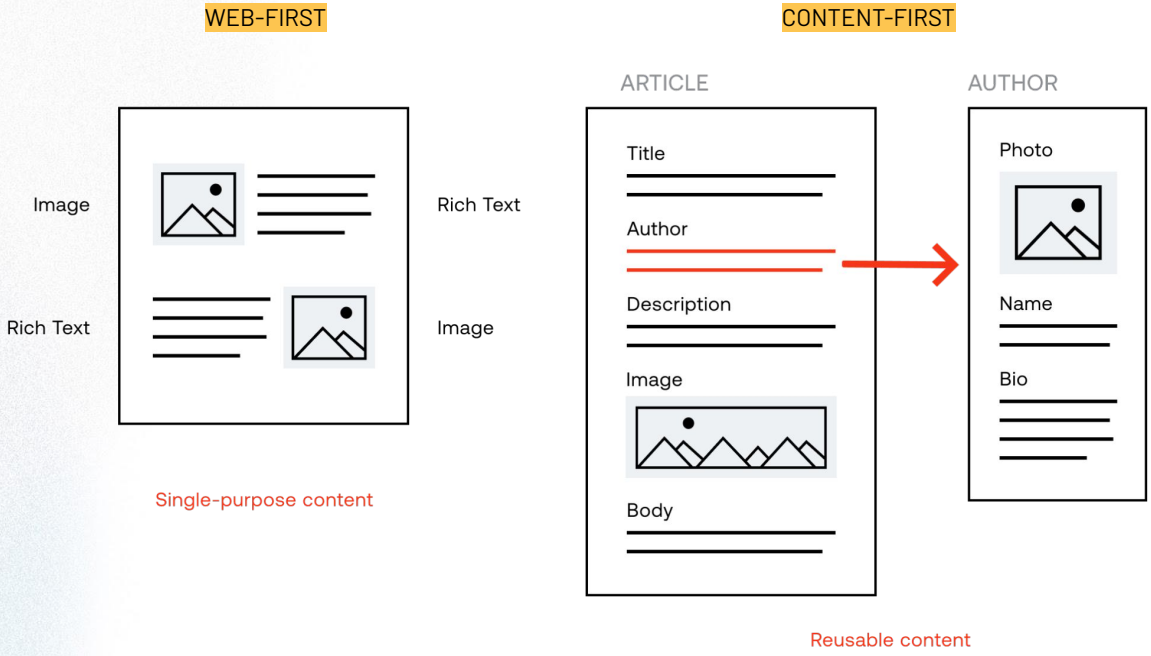
A **headless CMS** is a content management system where content is not tied to a specific user interface. Content is structured and can be displayed on various platforms such as web pages, kiosks or mobile apps. **It's similar to a decoupled CMS**, but the content is also independent of any specific front-end used to display it.



Decoupled & frontend agnostic

"Headless" and "decoupled" are terms frequently used interchangeably, but there is some nuance.

Another way to think of headless is a CMS that is **less focused on building web pages and more focused on building content for multi-channel platforms** (e.g., a website or a digital display). This adds a layer of abstraction and complexity to content creation and offers a unified way to maintain content across channels.



Interchangeable & focused services

Compose your solution out of best-in-breed SaaS solutions.

A **composable** architecture is a technology pattern based on using components with standardized interfaces so that each component is swappable and replaceable as needed to meet evolving business needs.

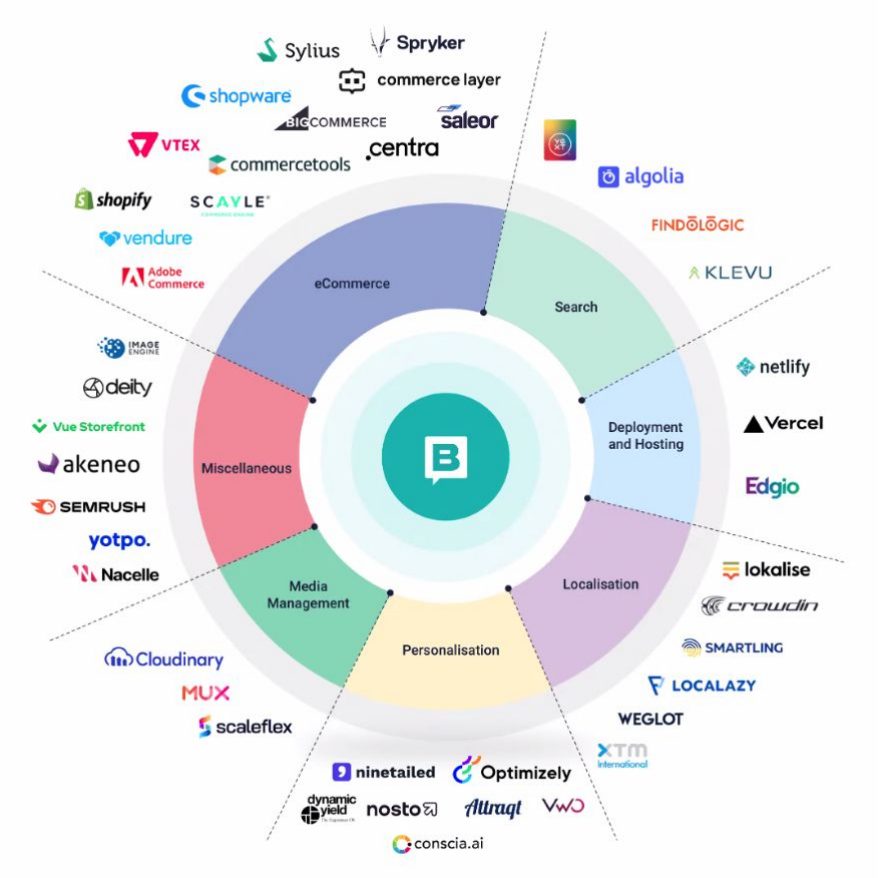


Image Attribution: Storyblok

What are Jamstack and the MACH Alliance?

JavaScript, APIs & markup

Jamstack is a JavaScript-focused movement.

Jamstack is an acronym for a "JAM" technology stack - **JavaScript, API, Markup**. The term was coined in 2015 with core principles of decoupling and pre-rendering*.

- **JavaScript** — One coding language for all parts of the stack (no PHP, C#, etc.)
- **APIs** — Common definitions for integrating between decoupled parts
- **Markup** — Front-end tech for how we define what renders on the page

***Pre-rendering** — The process of pre-generating all webpages as static HTML/CSS/JS during a build process and not at request time.

Microservices, API-first, cloud-native SaaS & headless

The **MACH Alliance** is an industry body that advocates for open technology ecosystems that give enterprises confidence they are using best-in-class vendors that can deliver future-proof technology. Members meet requirements and largely focus on being **composable** solutions.

- **Microservices based** — Individual business technologies that are independently developed, deployed, and managed (ex: SaaS platforms)
- **API-first** — Common definitions for integrating between decoupled parts
- **Cloud Native SaaS** — Cloud platforms built for elastic scalability
- **Headless** — Decoupled, framework agnostic, front-end presentation layers

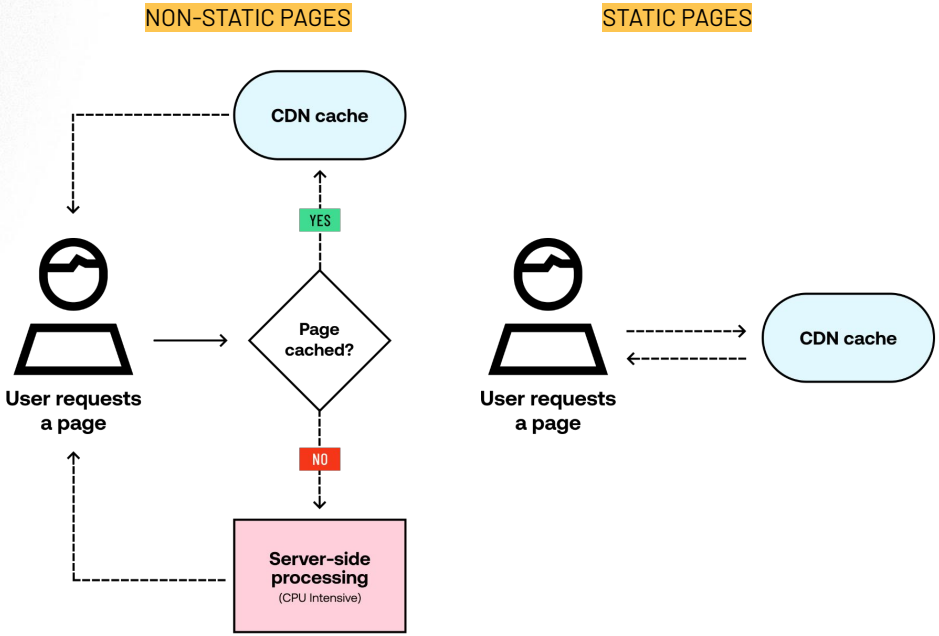
Why should you care?

The fastest page loads possible

Where possible, static pages can be pre-generated, meaning there is zero server-side processing overhead to render a page — it's just files being thrown over the wire.

Decoupling content editing from presentation gives us full control over how pages are rendered. Meaning we can leverage front-end stacks like NextJS or Astro — platforms that have great performance benefits:

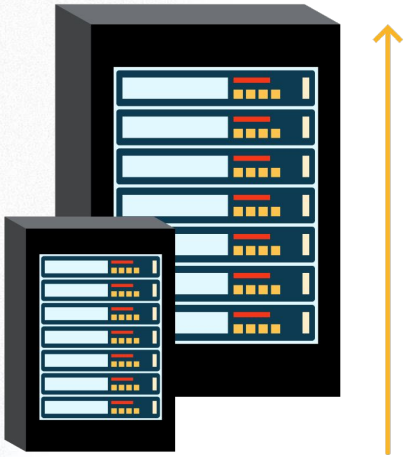
- Support for various rendering strategies — static pages can be built and served statically, and dynamic pages can be rendered realtime server-side
- Advanced CSS bundling and page serving optimizations
- Custom Image components (in some cases developed in tandem with Google for the best PageSpeed Insight scores)



Infinite horizontal scalability

Scale horizontally as needed and deploy to numerous points of presence.

Monolithic architectures typically have to scale vertically by adding RAM and CPU processing power to the server. Modern tech stacks are based on Node images, which can scale horizontally quickly, allowing for “infinite horizontal scalability”.



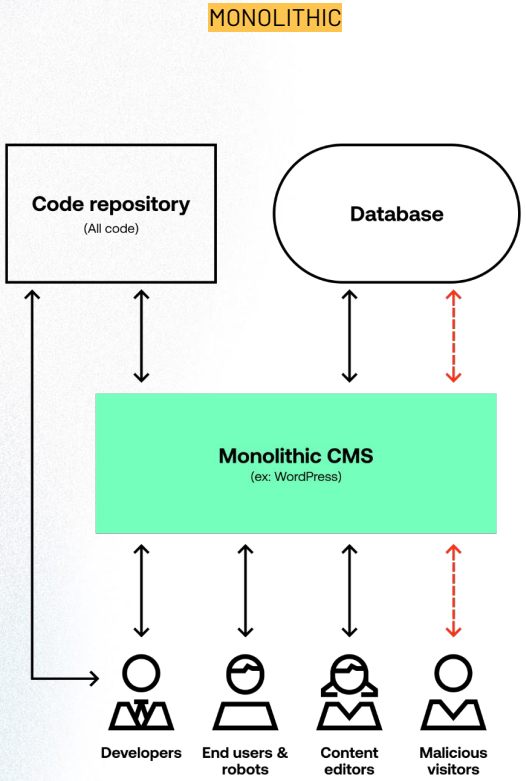
Vertical Scaling
(SCALING UP)

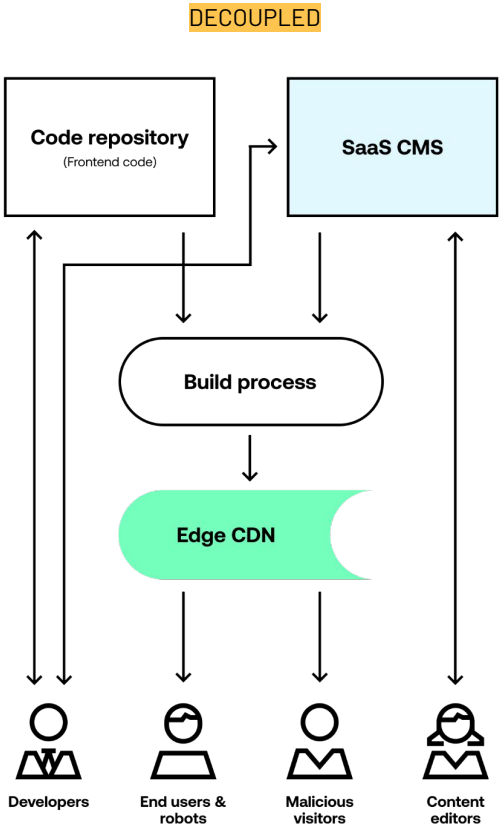
Horizontal Scaling
(SCALING OUT)

More secure in every way

Stop serving your developers, editors, and visitors from the same server — better limit service access and reduce attack vectors.

With a decoupled approach, you can stop having the same site serve both end users and content creators. Now, content creators will visit the CMS, but end users will not. Malicious visitors will find it less obvious what to attack, and compromising the CMS won't compromise the public live site — and vice versa.





Want to know more?

The Alloy development team has the deep expertise to help evaluate your needs so you can make the best decisions for your next website build.

Check out our latest case study on how we built a decoupled architecture for a software testing powerhouse.

Or even better, reach out and let's discuss!