The Ultimate Headless CMS Selection Checklist

Changing your CMS is a time consuming and massive strategic decision. Have you considered everything you need to know before making the switch?

Written by hygraph

Table of Contents

TL;DR

When choosing a Headless CMS, there are a lot of options and combinations that might seem overwhelming. On the technical side, the development team needs to choose a CMS that fits into their tech stack, allows them to work with their preferred frameworks, and provides an enjoyable API experience. However, on the marketing side, the options are no fewer. There are lots of considerations that need to be taken to ensure that a Headless CMS is both operationally fulfilling, and future-proof.

With the emergence of Headless CMS over their Monolithic counterparts, it's important to understand which features and capabilities matter most to you, and how you should go about selecting a Headless CMS that best suits your needs.

Considerations when choosing a Headless CMS

5

With the emergence of Headless CMS over their Monolithic counterparts, it's important to understand which features and capabilities matter most to you.

Selecting a new Headless CMS is a major decision, and not one to be taken lightly. Identifying your key problems and goals as a department or organization will help you navigate the selection process successfully.

However, given the options, and considering that opting for a CMS is an important decision to your tech-stack, identifying your key problems and goals as a department or organization will help you navigate the buying process successfully.

By knowing where your organization’s pain-points are, you will know what to look for in a Headless CMS. You will decrease the risk of buyer’s remorse, overspending, and avoid overly complex solutions.

Another decision is deciding between open source CMS or closed source CMS. Since headless is still a rather new concept, open source headless projects may require significant development investment to make them production-ready.

Commercial options are still more mature and aligned with commercial use-cases.

> **By knowing where your organization’s pain-points are, you will know what to look for in a Headless CMS.**

The Headless CMS Selection Checklist

Now that we've gone over the high-level validations required when deciding to opt for a Headless CMS, let's dive into covering the specific features you should keep an eye out for.

Content Architecture & API Design

□ **Ease of Setup**

If you can't set up the content architecture quickly and intuitively, then you may be wasting resources with your CMS.

□ **Your Content API(s)**

Consider the differences in RESTful vs. GraphQL, and ensure your CMS provides a content API that matches your needs.

□ **API Functions**

Your CMS's content API should offer you functions like ordering, filtering, and pagination, among others.

□ **API Content Management**

Your API shouldn't just deliver content via API (read/query), but allow you to add content programmatically (write/mutations).

□ **Development Environments**

Your CMS should enable a secure development process. Developers should have access to multiple Environments.

□ **Content Stages**

Your CMS should provide you with multiple versions of your content with a flexible retention period.

□ **Versioning**

Your CMS should provide you with multiple versions of your content with a flexible retention period.

8 Content Delivery & Performance

□ **Predictable Payloads**

Your API should deliver exactly what you ask for - no more, and no less. This is the default behaviour with GraphQL APIs.

□ **Content Availability**

Your content should be available whenever you need it with a minimum guaranteed uptime that suits your business needs.

□ **Content Distribution**

Your content and assets should be distributed (and cached) across several data centres close to your key markets.

□ **Effective CDN**

Your CMS and Content API should have middle-layer caching across a global CDN.

> **Your API should deliver exactly what you ask for - no more, and no less. This is the default behaviour with GraphQL APIs.**

9 General API Considerations

□ **Management API**

Your CMS should have a strong & flexible management API to manage schemas, users, and admin activities across projects.

□ **Mutation API**

A Mutation API allows your team to programmatically create, edit, and update content with GraphQL Mutations.

□ **Documentation**

A well documented API flattens your team's learning curves, costs, and overheads.

□ **Roles & Permissions**

Your CMS should allow you to set several levels of access for multiple users, and manage them on the fly.

□ **i18n and i10n**

Your Headless CMS should support localization and internationalization out of the box to help serve your markets.

□ **No hard limits**

Your content API should not dictate or throttle the complexity or size of your generated queries.

□ **Content security**

Your CMS should support Permanent Authentication Tokens (PATs) for your endpoint out of the box.

10 Editorial Features

□ **Ease of Use**

Your content editors should be able to intuitively use a Headless CMS with a seamless editorial experience.

□ **Custom Roles**

Depending on your team's needs and structure, you should be able to set custom roles for colleagues.

□ **SEO**

Your CMS should support strong SEO best practices, technically and operationally - more in our **Headless CMS and SEO guide**.

□ **Flexible Content Modelling**

Your CMS should handle all marketing related formats and templates - more in our **Headless CMS for Marketers guide**.

□ **DAM**

A good CMS allows you to handle your digital assets within the app, and serve them for optimal performance.

□ **Programmatic Asset Transformations**

For SEO and performance, a CMS should allow you to serve your assets with best practices and modern formats like WebP.

□ **Manual Asset Transformations**

Furthermore, and for ease of use, your CMS's DAM should enable you to manually edit assets to perfection.

□ **Asset Hosting**

Most CMS's DAM would be publicly accessible. Your CMS should enable you to host assets in a self-owned bucket too.

Editorial Features

□ **Rich Editor Experience**

Whether in Rich Text, Markdown, or Plain Text, your CMS's editor should offer the options your team prefer.

□ **Editorial Convenience**

Your team should be able to create, edit, and update content on the fly in multiple languages without a learning curve.

□ **Martech Integrations**

As a core component, your CMS should be able to seamlessly communicate with your other marketing APIs and tools.

□ **Omnichannel Content**

Your CMS should let you to manage content for all your devices - whether web, mobile, watches, or anything else.

□ **DXP Ready**

Your Headless CMS should work with your tech stack, not against it, when delivering modern digital experiences.

Business Flexibility

□ **Flexible Pricing**

Your CMS should scale with you and your project, and not make you suffer vendor lock-in or unfair pricing practices.

□ **Multiple Support Options**

A strong SaaS CMS allows you to find assistance via an active community, docs, support channels, or account managers.

□ **Service Level Agreements**

Depending on your needs for support response time, API uptime, or dedicated resources, SLAs should be available.

12 Security, Compliance, and Privacy

□ **Backups**

You should have the options of manual backups, point-in-time recoveries, nightly, and offsite backups.

□ **Robust Infrastructure**

Your customers and users are across the world and need constant attention. Your content should be able to match that.

□ **Auto-Scaling Features**

Based on your changing business landscape and project growth, your CMS features should scale with you.

□ **Scalable Infrastructure**

You should have the option of a strong hosted infrastructure - shared or dedicated - that doesn't let you down.

□ **Certifications**

Your CMS should ensure using ISO 27001 certified providers and data centres to match compliance needs.

□ **Data Encryption**

Your content and assets should be served over secure protocols without being compromised.

□ **Secure API**

Ensure that your CMS's API security policies include custom origin policies, IP firewalls, and regulation compliance.

□ **Compliance**

Your CMS (and other services by extension) should be GDPR compliant, and able to securely process information.

□ **Internal Protocol compliance**

Your CMS should give you the option of Single Sign-On (SSO) and Audit Logs to ensure your team's activities aren't at risk.

13 The above considerations should assist you in making an informed decision into which Headless CMS best suits your use-cases. To get a more in-depth analysis into considerations when opting for a Headless CMS, download our **Headless CMS Buyer's Guide**.

The Headless CMS Buyer's Guide

How to find the right Headless CMS for your organization.

Download the Headless CMS Buyer's Guide

About Hygraph

Hygraph is the first native GraphQL Federated Content Platform, enabling teams across the world to rapidly build and deliver tomorrow's multi-channel digital experiences at scale.

Hygraph is trusted by over 50,000 teams of all sizes:

It was designed for building a hosted GraphQL back-end that provides the tools needed for modern omni-channel content management. Hygraph integrates with any front-end technology, such as React, Angular, and Vue.

Hygraph is developed by GraphCMS GmbH, headquartered in Gießen, Germany, with an additional office in Berlin, Germany.

Get started with Hygraph by **creating a free account**, learn how our customers are **solving real-world problems**, gather information about Headless CMS from our **resources** or **academy**, or learn more about the **applications of Hygraph**.

To discuss how Hygraph can help you transform your digital projects, reach out to us on **hygraph.com/contact**.