



Kontent.ai



How to compare CMS options.

**Understanding the role of CMS architecture
so you implement the right solution**

Michael Andrews, Kontent.ai

Table of contents.


About Kontent.ai.	3
Why this guide?	4
Prepare to transform your content operations.	5

Part 1: How CMSs vary.

1 The landscape: Content management categories.	8
2 Page-oriented CMSs.	10
3 Modified page-oriented CMSs that have been updated to be “head-optional”.	12
4 Headless-native CMSs.	14

Part 2: CMS Capabilities compared.

1 Understanding content platform capabilities.	17
2 Comparing content structure and organization.	18
3 Comparing authoring tools.	22
4 Comparing content delivery capabilities.	25
5 Comparing enhancement options.	28
6 Comparing CMS architectures.	31
7 Deciding what you need.	33



About Kontent.ai

Kontent.ai is the leading modular content platform that enables marketers and developers to plan, create, and deliver content at scale.

Business teams collaborate daily in the intuitive authoring experience, designed for governed content production and management, real-time reviewing and approvals, and modular content reuse. In parallel, developers leverage the total flexibility of the headless solution, best-of-breed technologies, and composable AI to deliver digital experiences via API to any channel.

With hubs in New York, London, Amsterdam, Brno, and Sydney, Kontent.ai helps companies like Zurich Insurance, Algolia, Vogue, AC Milan, and Oxford University unlock the full potential of their content and connect with customers worldwide.

Why this guide?

Your choice of a content management system (CMS) will affect everyone who relies on content for business results. Despite the importance of CMSs, it can be difficult to know what elements influence the outcomes that they can deliver.

Understand the factors that contribute to success. While the individual features of a CMS each play a role, how these features work together – or not – will determine what outcomes you can achieve. Many organizations make the mistake of focusing on CMS features individually instead of examining how they are designed to work together as a platform.

Content management systems have earned a reputation – not undeserved – for being difficult to navigate. This guide provides a non-technical introduction to significant factors in content management that will influence the outcomes your organization can achieve.

Clarify your decision parameters. Once you get past the jumble of CMS features and jargon, what factors are most critical, and which choice will be in the best long-term interests of your content operations? This guide will explore the factors necessary to ensure:

- Your content is maintainable and future-ready
- Your CMS setup promotes operational efficiency
- Your setup can adapt to evolving needs

This guide will explain the differences in CMS product categories so you can:

- Appreciate how features that are used or implemented will influence outcomes
- Look beyond the presence of individual features to see how they interact
- Understand the capabilities necessary to adapt to evolving requirements

Prepare to transform your content operations.

Think about how content fits in the digital transformation of your organization. Most professionals believe that content operations should become more connected and unified. Enterprises need to respond to a changing competitive environment.

Table: How content operations issues are changing

Issue	Before	Now
Scalability	Teams worked independently on specific websites and apps with little coordination.	Enterprises need to address and coordinate a growing range of diverse content. They can't afford for their content operations to be siloed.
Omnichannel	Most content was intended for websites.	Organizations need to support a range of channels.
Ease of use	The choice was between simple tools that did little but required no training or complex tools that had more capabilities but required extensive training.	Organizations expect ease of use without sacrificing sophistication.
Agility	Organizations expected time-consuming workflows and difficulty when making operational changes.	Improving the speed of publishing and avoiding time-intensive handovers are priorities.
Adaptability to future needs	Old CMSs would need to be replaced every few years because they weren't easily upgraded.	Cloud-based platforms can continuously upgrade. Enterprises want to be able to add enhancements to take advantage of the latest and best technologies, quickly and without disruption.

Dependability	Because CMSs would routinely need to be replaced (re-platformed), the content would also be reworked or replaced to fit website redesigns.	Organizations cannot afford the disruption of having to redo content when making design or technology changes.
---------------	--	--

Each of these issues suggests a need for greater flexibility in how enterprises work with content. But flexibility isn't a discrete feature of a CMS. It depends on many factors, as we will see.



Part 1:

How CMSs vary.

1 The landscape: Content management categories.

The backbone of content operations is a content management system (CMS), which supplies a range of capabilities to support the creation, management, and delivery of content. CMSs come in many configurations, which can sometimes make them difficult to compare.

Content management systems can be divided into three broad categories:

1. **Page-oriented CMSs** designed to output HTML web pages
2. **Modified page-oriented CMSs** that let publishers remove or switch off functionality used to generate the display of pages (we'll refer to these as "head-optional CMSs" –because the use of internal functionality that presents the content, known as the "head," is optional)
3. **Headless CMSs**, which can deliver all kinds of digital content to any channel, but unlike other CMSs, don't render the content in a user interface for a specific channel such as a website (we'll refer to them as "headless-native CMSs" because the body of content is managed separately from the how that content is presented in user interfaces)

Key differences to notice

The category of CMS you choose will influence how your organization works with content. Each category has an opinion about how content should be managed. They vary in terms of who decides major decisions:

- **Prescriptive:** the vendor decides most core choices
- **Exception-driven:** the vendor decides the default configuration but publishers can override settings on a case-by-case basis
- **Open choice:** the publisher decides how they want to configure their setup

Table: How categories decide configuration

CMS category	Opinion about content management settings	Configuration
Page-oriented	Prescriptive: choices predefined	Fixed
Head-optional	Options as exceptions	Partly flexible
Headless-native	Open choice: publisher chooses among many options	Flexible

CMSs differ in whether features are fixed or flexible. Many CMSs seem to offer equivalent features, but on closer look, they vary as to whether the feature is fixed or flexible. While it may be possible to modify a fixed feature, it will require more effort than if the features provided or that can be added are flexible by design.

Look beyond the features. Capabilities are more than a checklist of features. The configuration of your CMS will determine whether the product will meet your needs. CMS products will vary in how readily they can be adapted to your organization's specific requirements. Many CMSs permit a degree of customization of default settings or workarounds of them, but certain CMSs allow for a much higher degree of configuration.

2 Page-oriented CMSs.

In the past, most all CMSs were page-oriented, designed to output HTML for web pages. Page-oriented CMSs encompass a range of products, such as:

- Web CMSs (WCMSs)
- Digital Experience Platforms (DXPs)
- Component CMSs (CCMSs) designed to publish XML documents to websites

While these CMS products are often considered distinct categories, they share a number of common traits.

Basic features of page-oriented CMSs

Even though page-oriented CMSs can differ widely in their specific features, they embrace a similar approach to managing content. They typically positioned themselves as “all-in-one” solutions that will handle everything a web publisher would need to do. The vendor has decided on most settings, defining generic functionality that it believes is sufficient for most organizations. These CMSs aren’t designed to support flexible configuration and can be difficult or expensive to modify.

A focus on pages and documents. As reflected in its name, a page-oriented CMS is geared to building web pages and articles. This orientation influences how the content gets created and how it is delivered.

The output destination shapes how the content is composed. In a page-oriented CMS, layout templates strongly influence how the content must be created and how it can be used. Authors create content for a specific screen component or part of a web page.

The CMS vendor supplies predefined page types. The CMS vendor supplies default page structures used to generate the output. If the default page types don’t match what the publisher wants to express, the publisher must modify existing pages or custom-develop new page types. Flexibility may be limited.

Website oriented. The page-oriented CMS is not designed to support diverse publishing scenarios. The CMS is set up to publish content to the web channel. If customers need to

access content outside of a web browser (for example, in a mobile app or a chatbot), the publisher will need to use other tools or else needs to customize the delivery capabilities provided by the CMS. For example, with one CMS, you may need to create a “custom REST endpoint” to deliver your content to a mobile app (we’ll discuss APIs and their significance shortly). With another one, you may get REST API functionality that would allow you to deliver content to non-website channels, but it’s hard to tailor your content for other channels because the CMS’s authoring UI is based on predefined page types and is geared to building web pages. These CMSs need major revamping or overhauling to support publishing to non-website channels.



3 Modified page-oriented CMSs that have been updated to be “head-optional”.

Many older page-oriented CMSs have announced updated features that provide some of the capabilities associated with a headless-native CMS, the most recent generation of CMSs. These head-optional CMSs are built with functionality to generate web pages for websites, but optionally, the internal functionality that generates the display of these web pages (the “head”) can be removed or disabled if the publisher wants to utilize external user interface code instead.

Some vendors assert that their head-optional versions (also called “hybrid CMSs”) offer the same capabilities as headless-native CMSs without requiring their customers to change their CMS. In many cases, these CMSs are legacy products built on top of an architectural foundation that is ten or fifteen years old, although the branding may have changed.

Although modified page-oriented CMSs may offer headless content delivery as an optional feature or operating mode, these CMSs do not have the same architecture and capabilities as a headless-native CMS.

Basic features of head-optional CMSs

Design customization possibilities. While the CMS internally defines the layout and presentation of web pages, publishers may be able to override this setup for some or all of their content.

Partitioned mode of operation. Although the default settings rely on page-building templates, users can elect to bypass these defaults to create structured content. When organizations decide to develop customized content structures that have different characteristics than the predefined page types, they will need to manage them separately using different operational

procedures. In some cases, CMS workflow support functionality may not be available for custom-structured content.

Need a separate tool to create reusable content pieces. The built-in editor of modified page-oriented CMSs does not support creating modular content. Employees who need to create reusable content pieces must use a separate editor, often developed by a third party.

Partitioned content: a mixture of predefined page types and user-defined structure. The content's structure is often predefined, at least in part, by the CMS instead of by the publishing organization. Some structural elements may not be relevant to the publishing organization.

API delivery added. Modified CMSs often provide add-on or plug-in APIs to enable content to be delivered to touchpoints other than websites. What the API can deliver will depend on the profile of the source content. Because modified CMSs may store content in diverse ways, they may require more than one API to deliver content, and not all content can be delivered to all destinations.

Predefined enhancement choices. Opportunities to enhance capabilities with external third-party tools depend on the underlying CMS architecture. When publishers want to supplement the internal capabilities of their CMS with external tools, their integration choices may be limited to a short list of vendor-provided or approved options.

4 Headless-native CMSs.

The most recent generation of CMS — known as headless — has developed over the past decade to bring greater flexibility to content management. We'll refer to CMSs that are purpose-built to be headless as “headless-native” CMSs to distinguish them from updated page-oriented CMSs that have incorporated some headless features, the “head-optional” CMSs discussed previously.

In contrast to page-oriented CMSs, headless-native CMSs don't use internal templates to define the content. Once authors create the content, it's ready for delivery without needing a template inside the CMS to first “build” a web page. Instead, headless-native CMSs rely on external code to create user interfaces for different channels and touchpoints. This arrangement simplifies how the content is developed and stored, improving its agility and availability.

Basic features of headless-native CMSs

Content is independent of layout. A headless-native CMS doesn't manage the presentation layer. As a result, the internal organization of the content in the CMS, which shapes how authors create and use the content, doesn't have to be bound up with how content is presented to customers on their devices. The content can be defined independently of a specific UI design, what's known as “decoupling”. When that's done, the redesign of a website or app won't necessitate a change to the content – existing content can continue to be used in the new UI design. Decoupling content from UI design makes the content more flexible.

Publishers decide their content's structure and organization. Instead of relying on a template supplied by the vendor, the publishing organization decides how to create and structure their content to reflect their business priorities. They can define what parts of their content are important to manage (to reuse, personalize, update easily, or optimize). This gives them greater control over the details within their content.

Any kind of content can be modular. When using a headless-native CMS, publishers can structure their content to be flexible and platform-independent.

API delivery by default. All content created and managed in a headless-native CMS is delivered to websites, apps, and other consumer touchpoints using an API.

Content can be published anywhere. By using an API, the same content can be published to multiple places and channels.

May have limited preview of content. Some headless-native CMSs don't provide the ability for authors to preview what the content will look like before it is published.

Open integrations. Headless-native CMSs generally support the ability to integrate other vendors' tools and apps to support content operations. These CMSs are designed to connect to third-party applications readily, providing a wider range of enhancement options than possible with other CMS architectures.

Part 2: CMS Capabilities compared.

1 Understanding content platform capabilities.

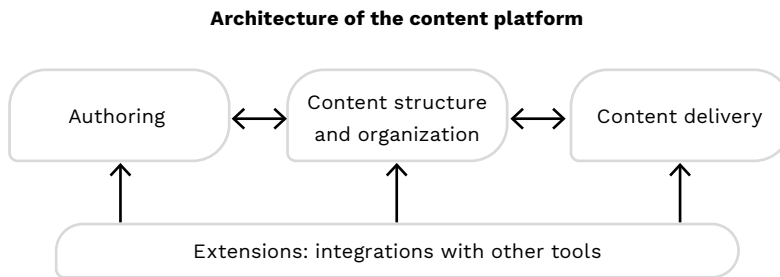
Your content operations depend on a range of capabilities that need to work together. The CMS will provide three internal capabilities governing:

1. The structure and organization of content
2. The authoring environment supporting the creation of content
3. The mode of delivery

The CMS may also provide opportunities to enhance capabilities through integration with other tools.

How these capabilities fit together defines your content platform's overall architecture.

Diagram: relationship of capabilities within a content platform



Part 2 of this guide will compare the capabilities of the three CMS categories (page-based, head-optional CMSs, and headless-native CMSs) in each of the five capability areas (Content structure and organization, Authoring tools, Delivery capabilities, Enhancement options, and Content platform architecture).

2 Comparing content structure and organization.

Why content structure and organization are important

How content gets structured in the CMS influences the types of content and experiences your organization can create and deliver.

A basic difference in CMSs is whether they rely on templates that fix the content's structure or whether they enable the publisher to decide the content structure themselves. This can influence the diversity of content the CMS supports.

- **Web pages:** All CMSs can produce web pages, but some CMSs will only support fixed page layouts, while others support modular flexibility, which allows the composition of web pages to change as needed.
- **Non-website channels:** Only some CMSs are designed to address content for touchpoints other than websites.

Key differences to notice

Many vendors will say they support “structured content” – a term that lacks an agreed definition. What's important is whether the content structure is:

- **Fixed** (set by the vendor and designed for a specific channel) or
- **Flexible** (decided by the publisher and channel-independent).

The characteristics of content flexibility involve several aspects, summarized below.

Table: Differences between fixed and flexible structure

Fixed structure	Flexible structure
Defined by page types or templates; the content may be organized in folders	Defined by content types that are related (linked to each other) within a content model
Decided by the vendor – modification is limited by the constraints of the CMS	Decided by the publisher, who can choose how to structure their content
Channel specific (content is designed to be outputted to a specific channel)	Channel independent (content can be reused in many channels)
Provides little or no ability to connect different items together	Provides a modular structure that allows authors to compose larger items by connecting them together in a range of combinations
Can be a “black box”: authors and developers must decipher how the structure is put together	Is transparent: both authors and developers can perceive the content’s structure and have a common understanding of it

When a CMS supports a channel-independent structure, the content can be combined in diverse ways and be reused on different websites and other channels. A page or UI-dependent structure doesn’t allow this flexibility.

Be aware that the ability to publish to more than one channel does not mean the content is truly channel independent. A CMS may let the publisher create content for more than one channel, but not all content will necessarily be modular and able to be used flexibly. CMSs can vary in the extent that the same content can be reused in different scenarios.

Full flexibility versus selective flexibility. Some CMSs will offer selective flexibility, which limits what types of content can be reused. The content’s structure is not fully flexible when content reuse is limited to:

- Parts of web pages
- Content in specific UI components
- “Content fragments”
- “Information snippets”

By contrast, a CMS that provides a unified enterprise content model will support content that is fully modular, reusable, and ready for any channel. With an enterprise content model, publishers have the flexibility to plan what opportunities authors can target and how it is provided to readers. Content is stored in a format-neutral manner that makes it future-proof.

Capabilities comparison

Table: Comparison of content structure and organization approaches

	Page-based CMS	Head-optional	Headless-native
Templates define fixed content structure focused on web pages	✔	✔	
Allows the option to define the structure of some content	*	✔	✔
CMS organizes content by content type	*	*	✔
Supports flexible structure for all kinds of content		*	✔
Can define relationships between content pieces		*	✔
All content is structured by a unified enterprise content model			✔
Content can be future proof (platform independent)			✔

✔ Available by default * Varies: depends on the CMS



Considerations: Questions to ask about a CMS's content structure and organization approach

- Who sets the structure or schema for the content?
- Can any kind of content be structured the way the publisher wants?
- How much effort is involved to define the content's structure in a CMS?
- Is the structure easy for authors to understand?

3 Comparing authoring tools.

Why the authoring tool is important

How will you build your content, and what can you do with that content once it's created? The tool that authors use influences many aspects of the content process:

- **Productivity:** Does the tool help authors develop content in a strategic way?
- **Quality:** Does the authoring tool promote good content practices?
- **Flexibility:** Can authors develop a diverse range of content? Can they connect different kinds of content together?
- **Learnability:** How much training is needed to use the authoring tool?
- **Maintainability:** Does everyone use the same authoring tool, or do different groups use different tools?

Kinds of authoring tools. Writers may use various kinds of authoring tools:

1. Page editors focused on page styling, which resemble word processors
2. Page editors that manipulate page components such as paragraph blocks
3. Text editors that require writers to add code (such as Markdown) to their writing
4. Structured content editors that break content into sections according to its meaning and purpose

Key differences to notice

Authoring tools differ in whether they emphasize:

- **How the content is constructed** (and what parts are needed) or
- **How the content will look** once published.

Some authoring tools provide a preview of a web page before it is published. This can be helpful when authors will be responsible for deciding how parts of the content are ordered within a page in cases where this is not done automatically. Page editors generally offer a preview; other tools may not offer a preview.

Authoring tools are associated with the CMS in various ways. They may be a:

- **Default, built-in tool** that works with how the CMS is set up
- **Alternative tool integration** that replaces the built-in tool if the default tool is not adequate
- **Offline tool** that is used instead of the in-built tool to develop the content

Capabilities comparison

Table: Comparison of authoring capabilities

	Page-based CMS	Head-optional	Headless-native
Default authoring environment supports the creation of structured modular content			✓
All CMS users access the same authoring environment	✓		✓
Employees will need to use more than one authoring tool		✓	
Authors can preview templated content for web pages	✓	✓	
Authors can preview modular content for web pages			*

✓ Available by default * Varies: depends on the CMS

Considerations: Questions to ask about a CMS's authoring tools

- Do all staff use the same authoring environment, or are different tools needed depending on the type of content created?
- Is support for modular content authoring built-in to the CMS?
- Does creating modular content require a separate add-on tool? Does the add-on tool require additional resources such as training, maintenance, or hosting?

4 Comparing content delivery capabilities.

Why content delivery is important

APIs deliver content, connecting the content that's stored in the CMS to various UIs where customers interact with the content. They provide the ability to search, retrieve, and deliver selected content to different destinations. An API query can specify the subject matter of the content as well as which parts of content are desired.

Most CMSs now have APIs, but not all APIs offer equivalent flexibility.

An API's delivery flexibility will determine:

1. **The delivery scope:** the types of content available from an API
2. **The delivery details:** the granularity of information that can be specified in a query

APIs can vary widely in what they can retrieve and deliver, as their utility can depend on the structure and organization of content stored in the CMS. APIs may provide access to various dimensions, depending on the CMS:

- Web pages
- Parts of web pages
- Content for specific UI components
- Structured data for apps
- Modular (channel independent) content that can be of any content type

Is all content globally available? If not all content can be delivered using an API, that will limit where the content can be delivered. It will be difficult or impossible to provide certain kinds of content in specific channels. Even if you can work around limitations, the content's formatting might make it hard to process in some channels.

An API-first approach offers the greatest flexibility, giving the publisher control over to which user interfaces and channels to deliver the content and allowing them to refresh any details

presented to users immediately. These APIs can deliver finely targeted content to customers. Publishers are not limited to preconfigured modules defined by the CMS (such as page parts, UI components, or app screens). They can decide to which frontend UI to deliver the content that would provide the best experience for customers.

Key differences to notice

A CMS’s delivery capabilities will rely on one of two approaches:

- 1. **API-first** (the content that’s available is *independent* of how it is stored, managed, and processed in the CMS) which is generally a feature of headless-native CMSs
- 2. **Code-first** (the content that’s available is *dependent* on how it is stored, managed, and/or processed by the CMS) which is common in page-oriented and head-optional CMSs

Capabilities comparison

Table: Comparison of delivery capabilities

	Page-based CMS	Head-optional	Headless-native
Can deliver any content to any destination		*	☑
API is an add-on (Code-first)	*	☑	
API is native capability (API-first)			☑
General purpose APIs (supports delivery of all kinds of content)		*	☑
Special purpose APIs (only valid for some content)	*	*	

☑ Available by default * Varies: depends on the CMS. May be subject to limitations.



Considerations: Questions to ask about a CMS's delivery capabilities

- Check with developers about the ease and precision of using the available API – this influences how flexible the content will be and how agilely you can put your content to use in different channels and scenarios
- Is the API oriented toward retrieving whole content items (less precise) or specific information in fields within content items (more precise)?
- Assess what needs to happen for the content, once created, to be ready for delivery. What dependencies are there? Is the content useful as it is (more agile), or does it need to be filtered or transformed to become useful when delivered in the UI (less agile)?
- Do developers see the content model structure when they create an API query? Is there a one-to-one correspondence between how the content is stored and which parts are delivered (more unified and agile), or does the query need to extract parts that are useful from within denser pieces (less agile)?

5 Comparing enhancement options.

Why the ability to enhance the CMS is important

A range of new tools is available that can enhance different dimensions of content operations. Instead of a “one size fits all” approach, publishers recognize their needs are not identical to other organizations. Sometimes teams within the same organization will have different needs as well. Enhancement options allow publishers to customize their setup to match their requirements. They can target what they need specifically and make it available where it will be most useful in their organization. This allows them to spend money wisely on features they can most use.

Publishers can extend the capabilities of their CMS in numerous dimensions:

- Authoring assistance and editorial compliance
- Content classification, assessment, and automation
- Content personalization and optimization
- Content recommendations
- Delivery integrations
- Advanced search tools and enhancements
- Translation and localization
- AI and ML insights

The extension approach will influence how easy it is to add or change a tool and the variety of options available. Maintaining the flexibility to choose the best tool for your needs is important as:

- New or better tools become available
- Existing tool capabilities are upgraded
- Your business priorities and operational maturity evolves

Key differences to notice

Integrations depend on the platform architecture. Vendors offer different approaches to how customers can integrate enhancements, which vary in their flexibility:

- 1. **Custom plug-ins**, where the tool needs a special connector to work with the CMS
- 2. **Integration frameworks** defined by the vendor that allow certain tools to be added after custom code development
- 3. **Vendor suites**, where the vendor supplies additional functionality at additional cost
- 4. **Composable platforms** that allow a wide range of tools having open APIs to be integrated using a “low code” approach

Capabilities comparison

Table: Comparison of integration and enhancement opportunities

	Page-based CMS	Head- optional	Headless- native
Options depend on vendor's integration access	✓	✓	
Vendor promotes specific enhancements, including their own	✓	✓	
Publisher can choose their own integrations			✓

✓ Available by default * Varies: depends on the CMS

Considerations: Questions to ask about a CMS's integrations and enhancements options

- Can your organization choose enhancements from different vendors?
- How will enhancements work together? Are they platform-independent, or do they mash-up products whose compatibility depends on specific versions of the CMS or of the tools involved?
- Can you swap out tools if you want to?

6 Comparing CMS architectures.

Why CMS architecture is important

The architecture determines how specific capabilities (authoring, management, and delivery) fit together.

Organizations will want their content management capabilities to follow a common design approach to ensure they are cohesive. If they aren't well matched, content operations can be glitchy and difficult to maintain or change. Some vendors have added or acquired authoring or delivery capabilities that weren't originally part of their CMS. In the worst-case scenario, a solution may be a Frankenstein-like collection of features that have been cobbled together at different times from disparate sources. This situation arises when one vendor has bought out other vendors and merged their tools under a single product umbrella.

The end-to-end experience of content operations, from authoring to delivery, should be unified, not mashed together. Content staff and developers should have a common mental picture of the CMS's architecture and how it works.

If these connections aren't seamless, content operations will become fragmented and get bogged down. Internal handoffs won't be smooth, and delays will occur routinely. Maintenance costs will be higher, which can take away from spending on content creation and user experience.

Key differences to notice

Architectures will vary according to whether they offer:

- **A unified content platform**, where all content uses a common platform for authoring, management, and delivery
- **Grafted content operations**, where disparate systems are used to create, manage, and deliver different kinds of content

Capabilities comparison

Table: Comparison of architectures

	Page-based CMS	Head-optional	Headless-native
Unified content platform	✓		✓
Partitioned content management		✓	
Alignment between authoring and delivery			✓
Full content portability			✓
Single view of the content			✓

✓ Available by default * Varies: depends on the CMS

Considerations: Questions to ask about a CMS’s architecture

- Does the solution provide a unified experience for all employees, or does it involve a hodgepodge of features that will be exposed to different users?
- If teams use disparate tools, how will this impact training and maintenance?
- Is there a unified content model for how to store and manage all enterprise content, or is there a partitioned content model where different kinds of content are managed in different ways?
- Can the content be migrated to and exchanged with other systems easily (no special scripting is required)?
- Will the solution make the content future-proof, so that existing content can be used in new channels, integrated with new tools and capabilities, and presented in new design frameworks?

7 Deciding what you need.

Your choice of content architecture will shape the outcomes you achieve with your content. Make sure your decisions support your organization's readiness to grow and mature its operations and to adapt to new requirements.

Action plan

Have an action plan for building the right platform:

1. Prioritize your critical (long-term) needs and don't get distracted by minor features
2. Decide the appropriate CMS category and evaluate how easily you'd be able to shape these products to meet your needs
3. Build and tailor platform capabilities to support your content operations

Prioritize your needs: look beyond features. Outcomes depend on more than features: they depend on the CMS architecture. Many CMSs advertise they include or support common features. But how they provide these capabilities can be radically different. Differences in CMS architecture can impact both the experience of using the CMS and the results it can provide.

Decide the appropriate CMS category. Enterprises face a decision whether to stick with a legacy CMS product and hope it will add features and improvements that will keep up with changing requirements or whether to switch to a headless-native CMS. Increasingly, enterprises are choosing headless-native CMSs because of their extensibility, flexibility, and capacity to adapt to future requirements.

While not all enterprises are currently pursuing omnichannel publishing, a CMS's omnichannel support will be a strong indicator of how future-ready a platform will be. Whatever your current priorities, you'll want to avoid becoming locked in to decisions you can't change in the future. Check that the CMS's authoring environment, content model, and API delivery are aligned to support omnichannel publishing.

Think beyond the CMS: shape your platform so it will support your needs. You'll want to customize and extend the capabilities of your CMS. Watch out for dependencies such as vendor plugins and bolt-on tools that can be fragile and prone to break when other changes occur. Aim for seamless integrations, where capabilities work together without needing special maintenance or oversight. The more steps and internal processes that are necessary, the less agile operations will be.

A headless-native CMS can provide a robust foundation to support your content operations. It should provide a composable architecture that allows your organization to add capabilities that will be tailored to fit your requirements.

Using a headless-native CMS does not, by itself, provide a modular platform that can adapt to changing needs. Some organizations purchase a headless-native CMS and use it like a traditional page-oriented CMS. They replicate their current processes and miss out on the benefits of modularity. It's up to the publisher to activate the modular potential of a headless CMS. The modularity realized will depend on how the CMS is implemented.

While headless-native CMSs as a category support modularity, specific products will vary in how convenient and user-friendly they are in allowing business users to tailor their setups. You should expect to adapt and extend available capabilities to your needs. Make sure the solution you purchase enables your organization to configure your setup flexibly.

Validate that your CMS provides the foundation for modularity. Does it allow for coherent modular coordination of capabilities, or is it a hodgepodge of fragments?

Modularity provides flexibility. It supports operational agility. Your organization can take charge instead of being dependent on your CMS's vendor to give you the configuration you need.

A modular platform enhances the reliability of your content operations. It can adapt as your operations need to scale, address new channels, and deliver new experiences.

Is your platform modular?

Confirm that your platform provides:

- A modular approach to content creation, allowing authors to create incremental modules that can be reused?
- A modular approach to content publishing, providing the flexibility to combine modules in a multitude of ways?
- A modular approach to delivering content, delivering only what's needed where it's needed?
- A modular approach to enhancing capabilities that let you choose what you want?
- A modular architecture that fits together seamlessly?



Kontent.ai

[Talk to us](#)

START YOUR FREE
30-DAY TRIAL.



in

