Kontent.ai

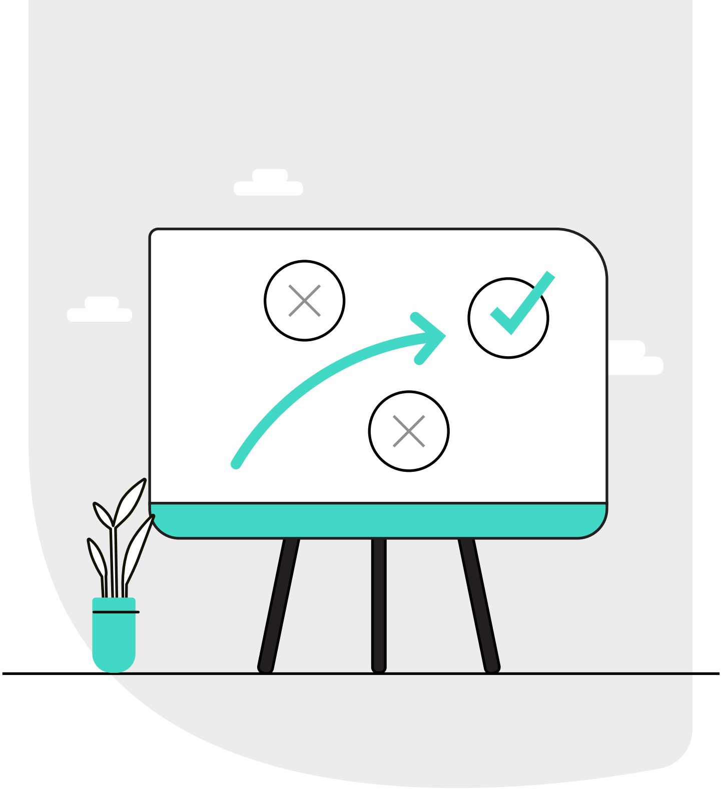
A step-by-step guide

How to choose Headless CMS for your organization.

We’ve seen a major shift in the content management system (CMS) industry in recent years. As companies go through digital transformation, they look for ways to become more agile and deliver modern digital experiences. They want to embrace new technologies and methodologies, such as cloud, microservices, front-end frameworks, or DevOps. At some point, they inevitably realize their legacy content management solutions hold them back. If you’re one of them, this guide will help you decide which CMS option is best for you.

Table of contents.

1



Start with a business case

It’s tempting to jump right into the shopping mood. But sooner or later, you will need to clarify why you need a new CMS and justify the investment. Below, you can find some of the main reasons why organizations replace their legacy CMS:

Supporting growth through higher agility

Many industries are being disrupted by fast-moving start-ups that leverage digital channels to enable new business models or by situations like a global pandemic that pushes businesses to become digital-first overnight. Businesses need to increase their agility so that they can launch digital products faster and quickly iterate on customer feedback.

Modernizing technology to provide a competitive digital experience

In recent years, we've witnessed a wave of new technologies, including cloud, APIs, microservices, front-end frameworks, artificial intelligence, Progressive Web Applications, and others. Many organizations realize their legacy technologies hold them back. They want to make their new investments using future-proof technologies to provide a modern digital experience for their customers.

Delivering personalized omnichannel experiences

Today's customers want to engage with brands not only through websites but also through mobile, social media, chat, voice, in-store experiences, and other channels. Legacy CMSs were designed for managing website content, and they do not enable modern digital products.

Making their content operations more efficient

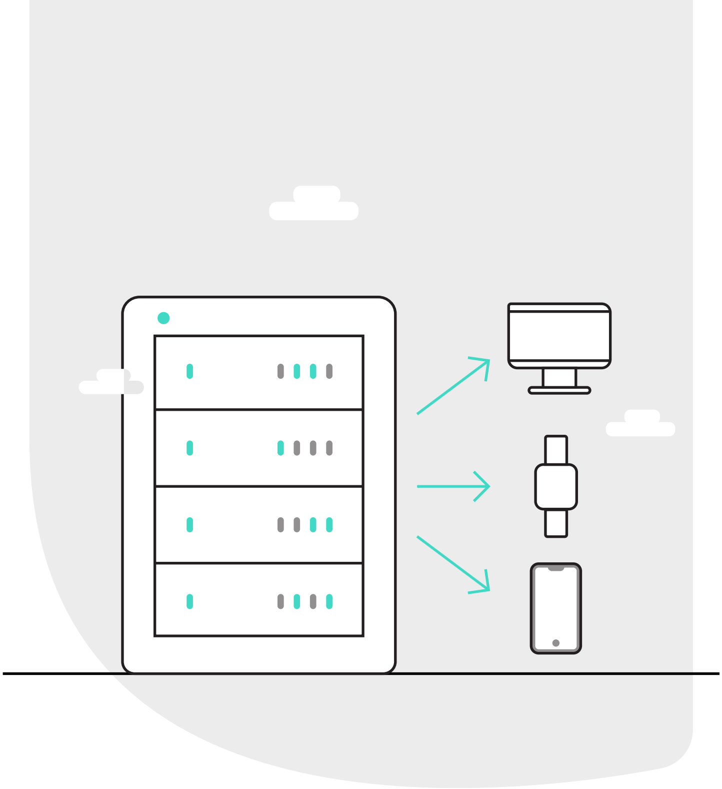
Many organizations manage their content in silos. They often use multiple disconnected tools with duplicated content, and they struggle to collaborate across multiple functions or departments. Publishing new content and keeping it in sync takes lots of time. All that impacts their agility, productivity, and brand consistency.

Providing the best retail experience

Retailers realize they need to provide a unique experience if they do not want to compete just on price. An engaging product-related content represents a key element of such an experience. However, most commerce platforms do not provide a robust content management functionality. Headless CMS represents an ideal central hub for product content that can be used not only for online stores but also for in-store experiences, such as digital signage or kiosks.

What's the opportunity for your organization, and how do you articulate the need for investment?

2



Understand the headless approach

If you're familiar with traditional CMSs, it's important that you understand how they're different from headless ones. Moving to the headless model requires a major shift in how you think about your solution.

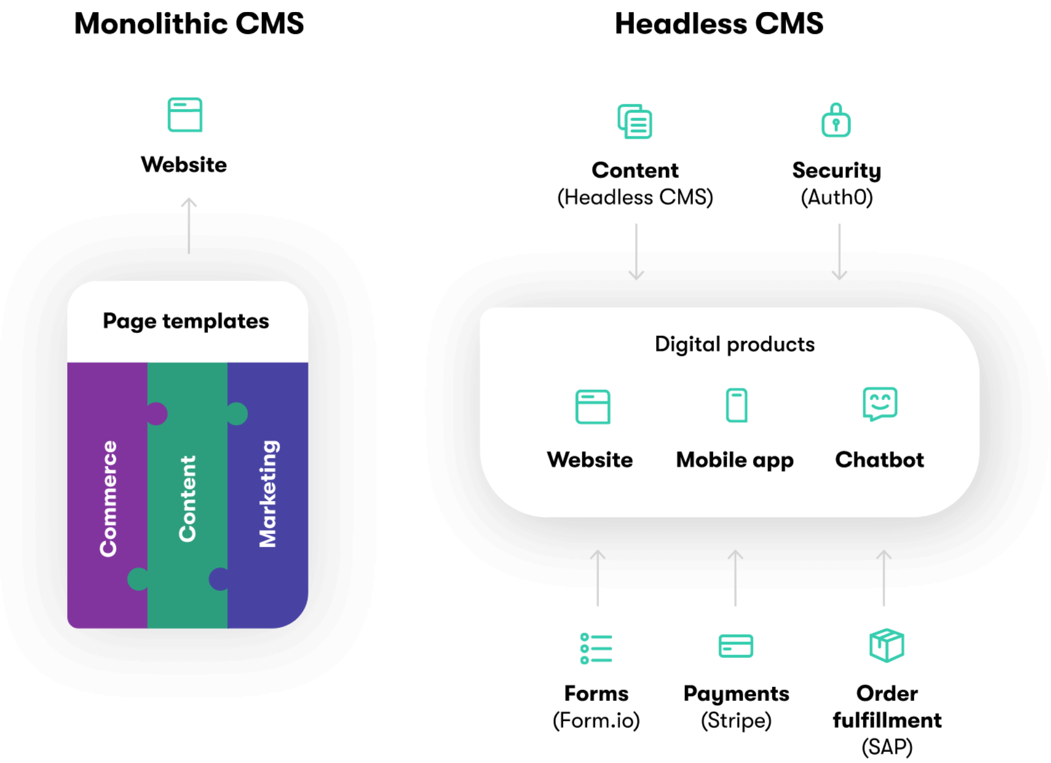
Monolith vs. microservices architecture

Traditional CMSs often combined content management and its presentation in a single coupled solution. They control how content is presented on the website, often using proprietary templating engines. On top of that, they tend to add lots of additional marketing or commerce capabilities. The result is one big monolith in which the CMS plays a central role.

Headless CMSs manage content and provide it via API. That means the content can be delivered to any digital channel (website, mobile, online store, chatbot, etc.). This approach is popular among developers who want to use progressive front-end frameworks and build mobile applications or other modern experiences.

Headless CMSs fit the microservices architecture perfectly. In this model, you build your digital product by connecting specialized services from different vendors through their APIs. The CMS becomes just another service in your digital experience stack.

In the microservices architecture, you build your digital experience stack around your digital product rather than building your product around a specific CMS. That means you don't have to throw away your digital product when you decide to switch to another CMS.



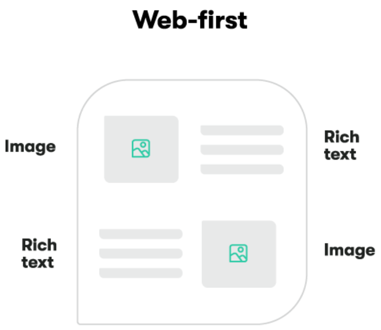
In the microservices architecture, you build your digital experience stack around your digital product rather than building your product around a specific CMS. That means you don't have to throw away your digital product when you decide to switch to another CMS.

Web-first vs. content-first approach

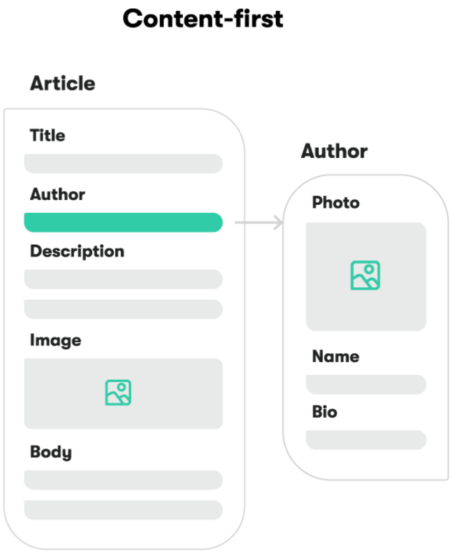
Traditional CMSs were designed for the web. They're built around web-specific concepts, such as a website, sitemap, and pages. They're focused on giving business users a high level of control over design, which may be empowering, but too much control may lead to brand inconsistency.

On the other hand, headless CMSs are designed for the content-first approach. This approach starts with content modeling that helps you define a proper structure of your content so that it can be reused across various channels and devices. That provides more flexibility but also requires a change in how you think about and work with content.

There are, however, elegant solutions on how to make the transition from web-first to content-first easier for the users, such as Kontent.ai's Web Spotlight.


Web-first

SINGLE-PURPOSE CONTENT


Content-first

REUSABLE CONTENT

Hybrid CMSs carry on the traditional mindset

Most of the traditional CMS vendors now offer some headless capabilities. Nevertheless, just having an additional API layer doesn't make them equal to headless-native, API-first solutions. Most of these hybrid CMSs are still built on the monolithic approach and organize content around the web-first model.

Their REST APIs may address certain needs such as delivering content to a mobile app or enabling a partial use of front-end frameworks, but the rest of the solution makes them suitable primarily for traditional website projects.

Since the separation of CMS and custom code remains limited, most hybrid solutions cannot provide the same benefits of a true SaaS model as pure headless CMSs.

Overall, the hybrid approach may be a good choice for organizations that look to continue using the traditional web-first approach with a monolithic CMS architecture and want to avoid the change of mindset.

Self-hosted model, managed hosting, PaaS vs. SaaS

Different products provide a varying level of cloud service:

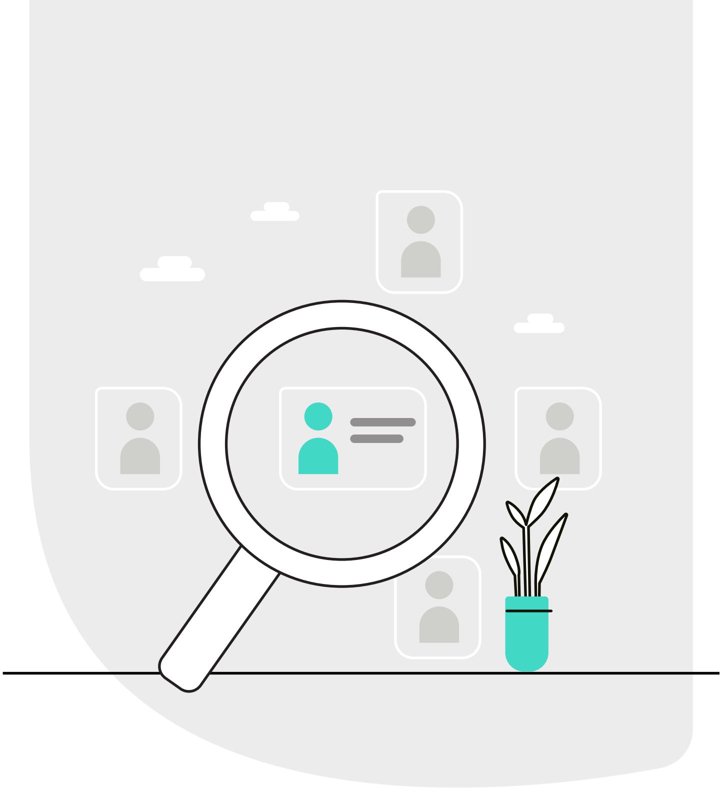
Self-hosted model is common among open-source products in both traditional and headless worlds. You download the code and install it on your server on-premises or in the cloud. This approach is suitable for environments where you need to have complete control over the CMS, and you have an internal IT team that takes care of hosting, security, availability, upgrades, hotfixes, etc.

Managed hosting is often provided by traditional CMS vendors who try to move their products to the cloud. They take care of hosting, security, and availability. When it comes to upgrades and hotfixes, they're often conducted by customers and then approved and deployed by the vendor since the CMS is tightly connected with custom code.

Platform as a Service (PaaS) is also a common model among traditional CMS vendors. Unlike managed hosting, PaaS should be fully automated, and you shouldn't depend on the involvement of the vendor. Beware: some vendors talk about PaaS, while they, in fact, provide managed hosting. Ask the vendor how the deployment, upgrades, and hotfixes work and make sure they're fully self-service and, ideally, can be scripted as part of your continuous delivery pipeline.

Software as a Service (SaaS) means that you only use the CMS as a service via its user interface and API. You don't have to worry about any element of running or upgrading the CMS, and you're always on the latest version. This model is only possible when there's a clearly defined API between the CMS and your code.

3



Identify the needs of your key stakeholders

Although the selection process is often initiated by IT, it’s important that you consider the needs of all personas who will get in touch with the new CMS. This includes:

- **Technical roles** – enterprise architects, front-end and back-end developers, DevOps team, system administrators, security specialists, etc.
- **Business roles** – marketers, content writers, subject matter experts, etc.

In the following paragraphs, we will take a look at key considerations for technical and business teams.

Key headless CMS criteria for technical roles

Here are some of the key technical considerations for developers:

Deployment model

With Headless CMS, you typically decide between two options: self-hosted and SaaS. SaaS allows you to focus on your project without worrying about running the CMS. On the other hand, a self-hosted CMS gives you full control over the application lifecycle (upgrades and related changes in API or UI) and your data.

Data center location

If you go with a SaaS solution, consider the location of your data. Some SaaS vendors let you choose the data center location, which may impact your compliance with data protection regulations (such as GDPR) as well as the API response time (see below).

API response time

In the headless CMS model, the API response time is critical for the speed of your application and for the user experience you deliver. Does the vendor provide a global Content Delivery Network (CDN) for both APIs and digital assets? Where is the data center located, and what latency can you expect?

Isolation model

If the solution is provided as a multi-tenant SaaS, you are sharing resources with other tenants. Does the vendor have restrictions in place to avoid one customer impacting the performance of the solution for the others? Can you get a dedicated instance?

Continuous integration

Despite the decoupling of content and code, your content, content model, and various metadata remain in the CMS. As such, they need to be part of your automated scripts for continuous integration and continuous deployment. Does the CMS provide an API to script any action in the system? Can you create multiple environments for development, QA, staging, etc., that are independent and allow moving content and settings between them?

SDKs

All headless vendors provide a REST API or GraphQL API. However, SDKs and sample applications may significantly speed up your development. Is there an SDK for your favorite technology?

Customization and extensibility

Can you customize or extend the solution with custom field types? Does the solution allow you to bring your own code into the CMS UI?

Integrations

What integrations does the vendor provide? How can you connect your existing solutions? Does the solution provide granular webhooks for key activities in the system? Some tools require advanced (first-class) integration to fully leverage their benefits, so make sure the solution integrates smoothly with the rest of your technology stack.

Security and data protection

How does the vendor ensure the security of the system? Are they ISO and SOC 2 certified? Are they able to comply with your security standards and with security requirements in your industry? How do they comply with data protection regulations, such as GDPR?

Documentation

What's the quality of the documentation? Does it cover all APIs as well as customization and integration scenarios?

Key CMS attributes for content teams

Adoption of the CMS by your content team is critical for the success of the whole project.

Many IT teams make the mistake of selecting a CMS based on technical criteria and then putting the CMS in front of the actual users without considering their input in the selection and implementation phase. And they are surprised by getting pushback from users whose needs aren’t met.

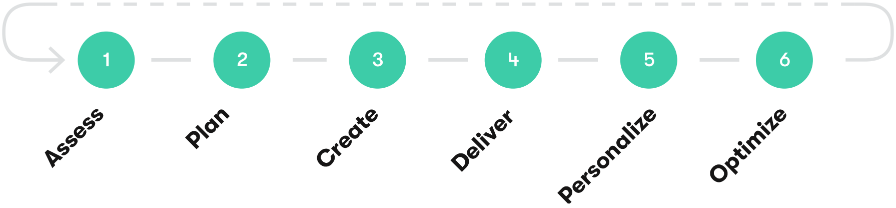
CMS should be a bridge between IT and content teams. These teams need to build the bridge together.

Involve your content teams at the very beginning—at the point when you start building the business case. After all, it’s them who should provide important inputs in terms of business needs, benefits, and objectives.

Below, you can find some key considerations for content teams. However, be aware that checking all the boxes isn’t enough. It’s important that you validate the CMS based on the actual tasks your users will be performing and make sure all the features “click” together and that your editors get a seamless experience that will make them happy and highly productive.

A good way to think about content-related tasks is starting with the content lifecycle. For each step of the content lifecycle, identify what you do with content today. Also, consider your long-term vision of how you want to work with content. Think about which tasks you want to perform in your CMS and what needs to be integrated with other systems.

Content lifecycle:



Assessment: Evaluate the performance of the content you have. Which topics get the highest interest? Which format works best in terms of driving conversions? What content do you need in each step of the customer lifecycle?

Planning: Based on the previous assessment, you know what to do—now put that into a content plan and assign tasks to content authors.

Creation: Write your content and collaborate with others. Think of your editorial and approval workflow. If it's relevant, consider your translation process and how you can automate it.

Delivery: What channels do you use today? How is it going to change in the future? Is it just web or also mobile or chatbot? Do you need to integrate your content with your commerce platform or marketing tools? How do you promote your content?

Personalization: Think of how you create content variants and how you ensure the delivery of personalized experiences.

Optimization: How do you analyze the engagement of your audience? Are you A/B testing your content?

For each task that you identify, consider how you will perform it in the new CMS. Can you do it out of the box? Or do you need to customize the solution or integrate other systems?

Evaluate each task based on:

- Ease of use
- Time to complete it
- Integration with adjacent tools versus manual copy-pasting and synchronization
- Empowerment of the business users (can they perform that task without a developer?)
- Duplication of content and work across multiple systems

Mind the business user gap

Comparing the business user needs to specific headless CMS products in the previous step, you have most likely identified that most headless CMSs do not meet the needs of business users. Most headless CMSs were created by developers to meet the needs of developers. As a result, business users often struggle to adopt headless CMSs:

1. They miss proper content governance throughout the content lifecycle and struggle to collaborate on content in the CMS.

2. They focus on managing content independently of the presentation, which is generally the right approach, but it requires a major change in your mindset.

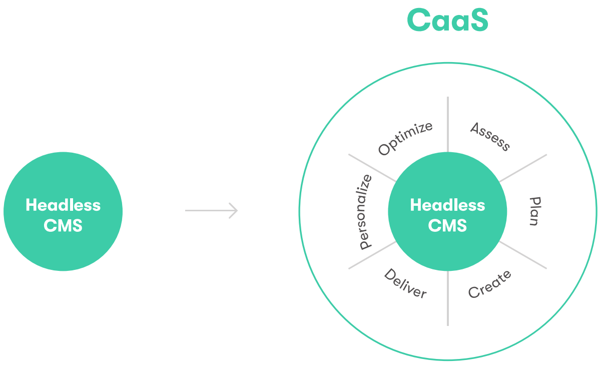
3. They rely heavily on developers when they want to create new pages and control the layout.

Fortunately, there's a solution to all these challenges. You just need to go beyond a pure headless CMS.

Content as a Service (CaaS) is a new approach to working with content. It's built on top of the headless CMS architecture and provides all its technical benefits.

At the same time, it provides business users with rich functionality to collaborate on content across its entire content lifecycle.

As such, Content as a Service enables the headless CMS to become a centralized content hub for enterprise organizations.



Other criteria to consider

Here are some additional criteria that are important for your selection:

Support and customer success management

Does the vendor provide 24x7 support? How many support engineers do they have, and how quickly are they able to respond and resolve your requests? Do they provide a dedicated Customer Success Manager? Do they have a well-defined onboarding process? Switching to a headless model requires a switch in mindset—can they help you instill best practices into your organization, such as proper content modeling? Do they provide consulting services and training?

System availability and SLAs

Does the vendor provide a transparent status page with a history of incidents? Do they offer an SLA? What guarantees do they offer?

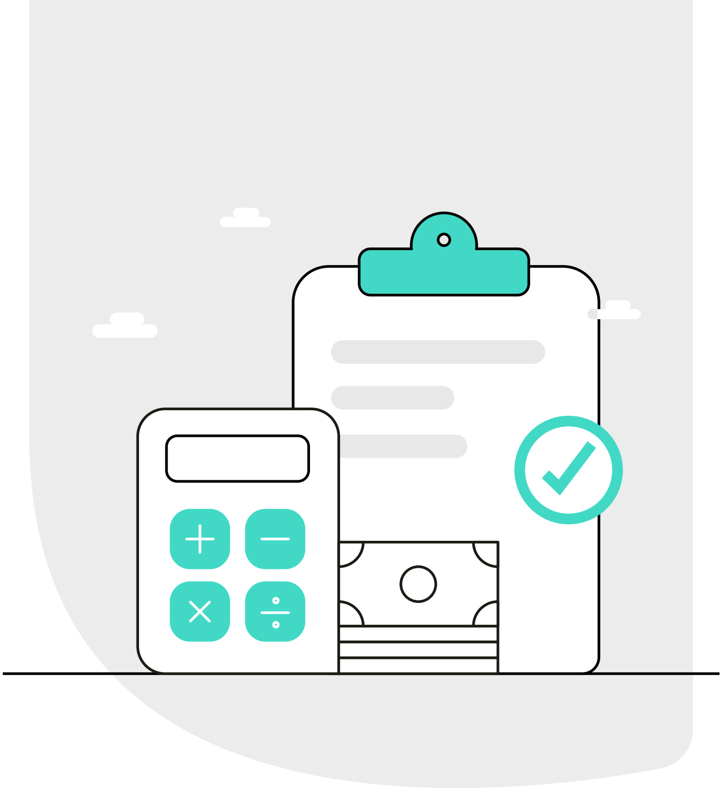
Vendor intangibles

What’s the vendor’s background? How big is the company, and how long have they been in the business? How big is their R&D and support team? Are they recognized in the industry by key analyst firms, such as Gartner and Forrester? Do they have references from customers with similar use cases? How do their customers rate them on G2, Gartner Peer Insights, and similar review websites?

Roadmap

As the headless CMS model is fairly new, the products are still quickly evolving. Vendors invest in different areas. While some are strongly focused on the developer experience, others are evolving towards meeting the needs of business users. What’s the vendor’s vision, and how does it align with your needs? Do they provide a transparent roadmap? What has been their velocity in delivering on that vision and roadmap in the past twelve months?

4



Set your budget

When thinking about your budget for switching to a new CMS, it's important to consider the Total Cost of Ownership. This includes:

- CMS costs (see below)
- Implementation costs for the initial project
- Migration costs for existing content
- Ongoing application development costs
- Hosting costs for your website or application
- Maintenance costs (upgrades, hotfixes, etc.)

When thinking about your budget for switching to a new CMS, it's important to consider the Total Cost of Ownership. This includes:

Free open-source — $0

These are typically self-hosted solutions. In many cases, the open-source model is part of the vendor's business model and the organization behind it expects to make money by providing either a premium version with additional features, a hosted version, or paid support. Be aware of what is included in the free edition. Also, consider the costs of hosting the solution by yourself, including upgrades, maintenance, security, performance, availability, etc.

Examples: Strapi, Squidex, Directus

Low-cost solutions — $20 to $500 per month

These products are usually built by individuals or small start-up teams. They may provide an easy way to get started for individual projects but may not keep up if you decide to deploy the CMS across a midsized or large organization. Some of these vendors face challenges when it comes to scaling their business—as their customer base grows, they lack the capacity to handle support and feature requirements while their business model isn't strong enough to support team expansion.

These products are often very developer driven. If you go this path, make sure you evaluate these products together with your business users and let them evaluate their usability in real-world scenarios. Also, validate the vendor's ability to deliver quality service and support.

Examples: Prismic, Graph CMS, Butter CMS, Storyblok, Strapi, Squidex, Directus

Enterprise solutions — $999 to more than $10,000 per month

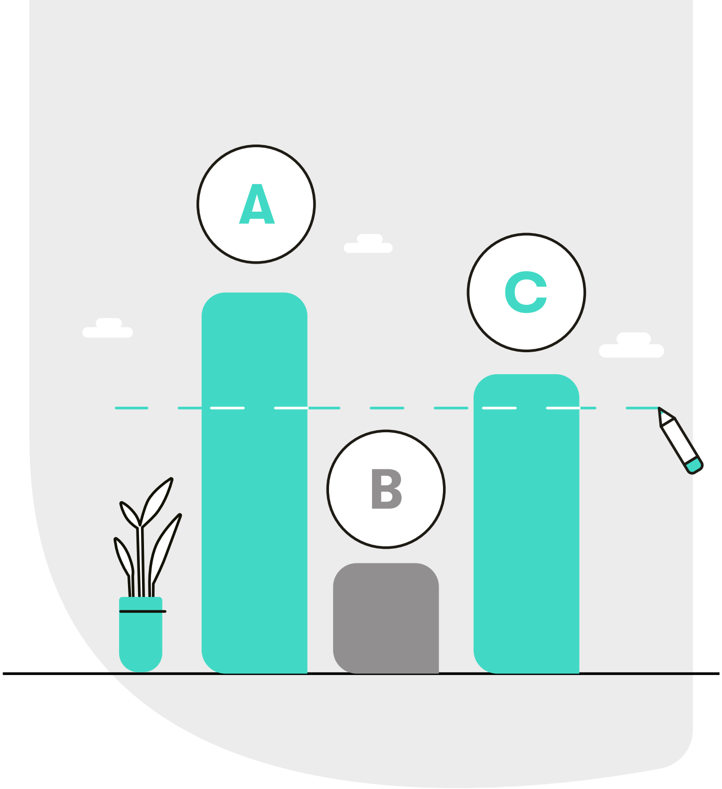
There are very few vendors in the pure headless CMS/CaaS market that are ready to support enterprise-level projects. The word "enterprise" is tricky, so let's define what we mean by that and what are some of the key characteristics of an enterprise solution:

- **Product maturity**
  - Advanced governance and workflow capabilities
  - Versioning and archiving of content
  - Strong multilingual support and ability to integrate with external translation services
  - Collaboration for larger teams – simultaneous editing (avoid overwriting of content), comments, tasks
  - Continuous Integration/Continuous Deployment – support for multiple environments, ability to roll back to a previous version
- **Scalability**
  - Ability to handle hundreds of thousands of content items in a single repository (both on the database level and in the user interface)
  - Ability to scale to large amounts of traffic
  - Ability to support a large number of concurrent users
  - Option to have a dedicated environment rather a shared one or ability to isolate customers in the shared environments
- **Security**
  - Flexible roles and permissions
  - Single sign-on (SSO)
  - Audit log
  - Security certifications, such as ISO 27001 and SOC 2

- **SLAs**
- **24x7 support backed by a well-staffed team**
- **Consulting services**
- **A dedicated Customer Success Manager**
- **Training and certification for both developers and business users**
- **A solution partner program with certified local partners around the world**
- **Integrations with commonly used enterprise platforms**

Examples of enterprise-level headless CMSs: Kontent.ai, Contentful, Contentstack

5



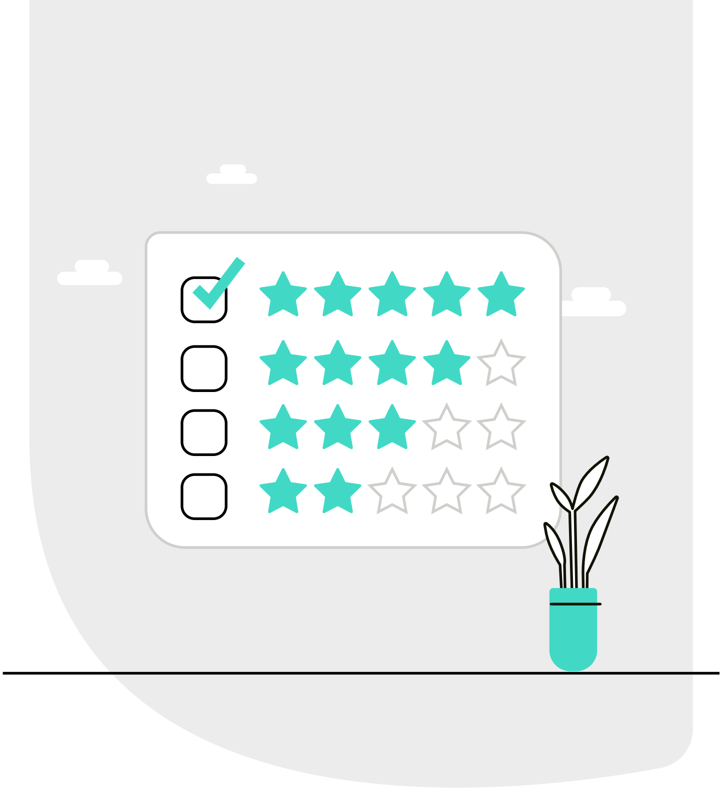
Shortlist the vendors

Now that you understand the differences between vendors, you should pick the segment of the CMS and define your long list of vendors you would consider.

Then, identify your key requirements and compare them to their feature set. You may run an RFI process or just make your own shortlist based on the publicly available information.

Once you have your shortlist, you can build your RFP document with more detailed requirements.

6



Evaluate the products

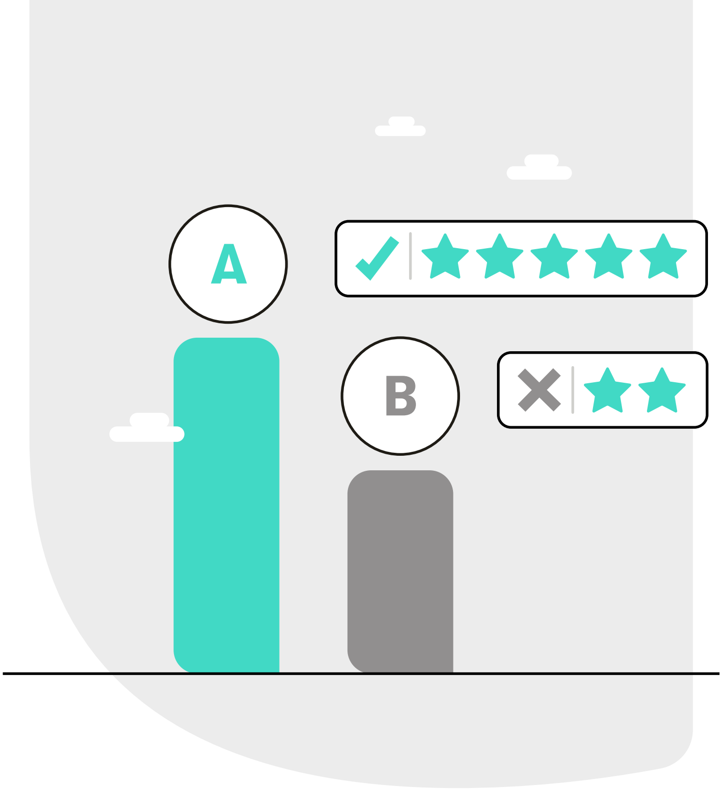
When you take a closer look at headless CMS products in the market, you will find out that many of them may check the same boxes on your list of requirements. It's important that you go beyond YES and NO. You need to validate how well the shortlisted products meet the needs of your developers as well as business users.

The best way to conduct a proper evaluation is by creating a simple proof-of-concept project. It will give you clear answers to questions like:

- How flexible is the available API?
- How good are the developer training and documentation?
- How quickly can I get technical support from the vendor?
- How does the solution support key use cases of our business users?
- How does the solution integrate within our existing ecosystem?
- How easily can the business users start using the CMS, and how do they like it?

That will help you decide which products are your preferred ones and move on to get their proposal.

7



Evaluate the vendors and their proposals

At this point, when you understand the differences between products, it’s time to get a better sense of vendors and what you can get for your money.

Make sure you know what you pay for

With most headless CMS vendors, you will encounter a mix of the following pricing metrics:

- **Functionality**
- **Users**
- **Projects**
- **Content items**
- **API calls**
- **Bandwidth**
- **Storage**
- **Environments**
- **Roles**

It's important that you look beyond the base price because some pricing models may grow faster than others as you continue adopting the product. Create the most likely scenarios for the upcoming 12-24 months and think about future projects you will use the CMS for. Consider especially the number of users, content types, content items, projects, and environments that you need now and in the future, as these are often the key drivers for vendors' pricing.

Consider the implementation costs and time to value

While the CMS costs are important, it's usually a relatively smaller part of the whole investment. With enterprise-level CMSs, the implementation typically costs 5 to 10 times of the software investment in the first year.

What can make a major difference is the use of vendor's training, onboarding, and consulting services. Or you may pick one of the vendor's certified partners who already have extensive experience with the given CMS. While the costs of such services may not be low, they can help you avoid typical pitfalls and achieve your results much faster. At the time when talent and time are scarce, this may be money well spent.

Training

Training is an important step to ensure the adoption of the new CMS and make your users efficient when using the solution. You may benefit from a standardized training provided by the vendor, but you also need to provide training and guidance for the specifics of your implementation, such as which content types should be used, your naming conventions, usage of taxonomies and metadata, as well as content planning, creation, approval, and publishing process. Ask the vendor about training options available and how much they cost. Also, consider your internal costs of preparing and delivering training.

Onboarding and Customer Success management

Deploying a new CMS may become a fairly complex process. Having someone who would guide you can significantly increase your chances to succeed. Ask the vendor what onboarding services they provide and how they work with their clients to ensure their success.

Support

Ask the vendor about support included in the cost of the solution:

- Do they provide 24x7?
- Is it included in the cost of your plan, or is it extra?
- Is there any limit of number of requests or number of users who can submit those requests?
- What channels can you use (chat/email/phone)?
- What's the typical/guaranteed response time?

Also, verify if the vendor is actually able to deliver the level of support they promise. Delivering quality 24x7 support requires a support team of at least 10 dedicated support engineers.

Consulting services

Similar to onboarding, leveraging vendor's consulting services may help you increase your chances of succeeding with your implementation. A good consultant will help you with the solution architecture, content modeling, as well as performance and security best practices. The ROI of investment into such consulting engagement is often 10x of its cost as you avoid common mistakes, shorten your implementation time, and make sure you use the system in the right way.

SLAs

If the quality of service is critical for you, you should dig deeper into the Service Level Agreement provided by the vendor:

What quality of service does the vendor promise?

Be cautious if the vendor promises 100% uptime—this rarely happens, and the vendors claiming 100% uptime often hope that you won't hold them accountable.

What happens if the promise is not met?

Does the vendor provide a refund or discount?

How does the vendor handle planned downtime?

Is the planned downtime considered as downtime, or is it excluded?

Ask the vendor about SLAs, their cost as well as the compensation when it's not met. The following figure shows the cumulative downtime according to SLA level:

SLA	Downtime per week	Downtime per month	Downtime per year
99%	1.68 hours	7.2 hours	3.65 days
99.9%	10.1 minutes	43.2 minutes	8.76 hours
99.95%	5 minutes	21.6 minutes	4.38 hours
99.99%	1 minute	4.32 minutes	52.56 minutes
99.999%	6 seconds	25.9 seconds	5.26 minutes

Company culture fit

Choosing the right technology partner for your business does not only depend on ticking all the feature boxes. After all, it’s the people behind every product that deliver the value to you.

Try to learn about the company culture and its employees’ wellbeing. It shows how each vendor treats the most critical asset of their business and has a direct impact on the long-term quality and reliability of the provided service.

Kontent.ai

About Kontent.ai.

Enable marketers. Free up developers. Unify teams. Kontent.ai is the headless CMS where modern digital experiences are made.

In the content hub, marketers plan and publish content that resonates. Intuitive, organized workspaces make it easier to collaborate and land on the best ideas in real-time. Secure permissions govern enterprise content workflows while streamlining tasks for specific authors.

Well-structured content is delivered by API. The complete decoupling of content production and presentation gives developers the freedom to use the technologies they prefer to create engaging, personalized experiences that look and feel great on any channel.

Seamless workflows and control meet independence and agility. This is unified content management at scale.

SCHEDULE YOUR DEMO →