KIBO®

Everything You Ever Wanted to Know About Headless Commerce

...but were afraid to ask

Table of Contents

Introduction	3
Definitions	4
A Brief History of eCommerce (or, Why Headless is the Future)	6
The Role of APIs	9
The Business Case for Headless	11
How to Evaluate Headless eCommerce Platforms	13
How to Implement a Headless Solution (a step-by-step guide)	17
Conclusion	20

Introduction

Headless commerce is no longer the preserve of tech enthusiasts and coders. From tiny online stores to the biggest retail enterprises, it seems everyone wants to go headless. And with good reason - headless commerce offers online retailers several benefits. **To name a few:**

- **FASTER SITES**
- **GREATER DESIGN FLEXIBILITY**
- **LESS HOSTING AND DEVELOPMENT HEADACHES,**
- **RICHER DIGITAL EXPERIENCES FOR YOUR CUSTOMERS**

and, most importantly,

This last point is crucial. According to PWC, one in three consumers will walk away from a brand after just one bad customer experience.

Implemented properly, headless can transform your online business and help define your brand's future.

But while there is no doubt that headless commerce marks a major breakthrough in the world of online retail, for the uninitiated, getting your head (excuse the pun) around the technology can feel like a huge challenge.

If you're looking into headless commerce for the first time you probably have a host of questions:

- **WHAT DO ALL THE TECHNICAL TERMS ACTUALLY MEAN?**
- **HOW WILL GOING HEADLESS BENEFIT MY BUSINESS?**
- **HOW DO I EVALUATE DIFFERENT HEADLESS SOLUTIONS?**
- **WHAT'S INVOLVED IN MIGRATING TO A HEADLESS PLATFORM?**

We've produced this guide to answer these questions and more. It will give you a clear understanding of what's involved in headless commerce, the knowledge you need to evaluate different headless options, and will help you decide if headless is right for your business.

Definitions

The world of headless commerce can seem daunting with technical language. But once you have an understanding of what the words and phrases mean, the concepts are a lot easier to get your head around.

Here are some definitions of key terms:

BACKEND All the behind-the-scenes functions that are needed to make your site or storefront run smoothly: cart, product information, inventory, etc.

FRONTEND The part of your online business that your customers interact with. Not just your site, but the whole digital experience: mobile app, in-store kiosks, IoT devices, etc.

HEADLESS Headless commerce refers to the practice of separating the frontend and backend parts of your eCommerce operation. The backend functions are handled by an entirely separate layer from the customer experience frontend. As the two layers are separated, the backend is without a frontend 'head'.

API (APPLICATION PROGRAMMING INTERFACE) If you have two programs written in different languages (for example, your storefront and the backend eCommerce platform), they won't have a common frame of reference and so are unable to interact with each other. APIs bridge the gap, providing a standard in which both programs can 'talk' to each other. In this way, APIs can relay messages from one program to another, with the messages containing instructions on how to amend, update, or delete data.

VUE.JS, ANGULARJS, AND REACT.JS JavaScript is a programming language that works with HTML and CSS (the languages that all websites are written in). Programmers use JavaScript to make pages more responsive to users as it gives developers more flexibility to create interactive components of a site. It's this flexibility that has made JavaScript languages popular with frontend developers.

Vue.js, AngularJS, and React.js are JavaScript frameworks. That is, they are collections of bundled JavaScript code that developers can use as programming 'shortcuts' that provide a foundation on which to make fast, responsive and interactive front-end applications.

SOFTWARE ARCHITECTURE Like the elements of design that are used to build physical structures, software architecture refers to the layout of a system and the principles underpinning its construction. There are several different ways to configure a complex software system like headless. How it performs in different scenarios will depend on how the system is architected.

MONOLITHIC ECOMMERCE An eCommerce solution with no separation between the frontend and backend. In a monolithic architecture, the different functions and services are hard coded together, making them hard to disentangle.

MODULAR Some headless platforms offer services on a standalone basis, meaning you can take one or more services without being tied into a huge software deployment. It brings extra flexibility as a modular platform doesn't have to be implemented wholesale, but can be done in stages, according to your needs.

MICROSERVICES The building blocks of all the eCommerce services your site has. Microservices are modules broken down to the smallest unit of functionality. Using an API to expose a microservice to your frontend environment means that you only take the specific functionality you need from the backend.

COMPOSABLE In addition to separating the frontend from the backend, headless allows separation of backend services so they can be used independently of each other. If you're taking this approach to building your tech stack, you're not limited to one provider, language, or framework and so can 'compose' your own eCommerce solution out of disparate elements.

CORE COMMERCE According to Gartner, these are the functions that all eCommerce businesses will need to have in place to be able to operate, and every commerce platform should provide. Making sure your headless commerce solution has an outstanding catalog, search, basket, promotions, payment, and customer account is essential to the evaluation process.

CMS (CONTENT MANAGEMENT SYSTEM) This, as the name suggests, is a repository for storing, organizing, tagging, and arranging all the content you need for your site (plus your mobile site, devices, in-store kiosks, etc). Although CMSs provide a backend function, they are often accessed and managed via a WYSIWYG (What You See Is What You Get) page editor or page-builder. More sophisticated CMSs will come with their own page-builder, but many CMSs are standalone and will require integration with another 3rd-party page-builder.

A Brief History of eCommerce

To properly explain the significance of headless commerce and why it makes sense for today's online retailers, we need to take a look at how eCommerce has developed. Online retailing has gone from a relatively straightforward extension of brick and mortar stores into the dynamic, multichannel driver of brand identity and immersive digital experience that it is now.

A long time ago, around the year 2000, retailers were waking up to the possibilities of selling their products online.

In this distant past, the way companies thought about online retail was simplistic by today's standards.

Early eCommerce providers catered to businesses who sold their products through a desktop site with a dedicated warehouse attached to it. As such, they were monolithic in design, geared towards supporting a single website, and focused on the logistics of a simple 'warehouse A' to 'consumer B' fulfillment model.

Digital experience and the rise of eCommerce platforms

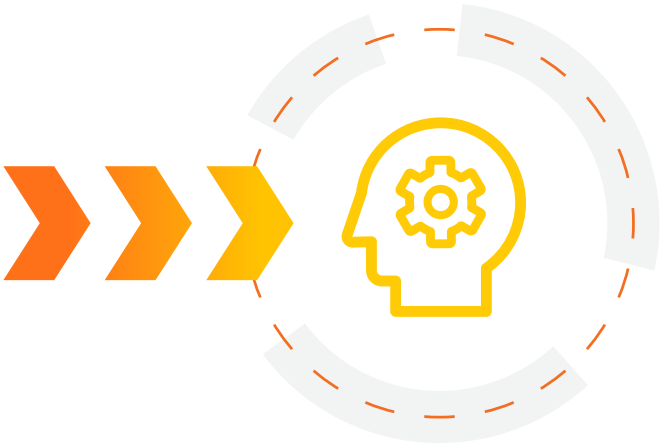
20 years later, and the online retail landscape has changed almost beyond recognition.

As brands and retailers have sought to meet an online generation where they spend most of their time, and technological innovation has continued apace, there has been a proliferation of channels. From mobile apps to in-store kiosks, brands now exist in an omnichannel world with a premium on providing outstanding digital experiences across touchpoints.

The linear model that saw a customer landing on a site and clicking through the steps to eventually make a purchase has been replaced by a desire to provide customers with a rich, cross-channel digital experience.

These experiences involve everything from personalized product recommendations to video guides to targeted reminders. And, increasingly, the boundaries between online and offline are becoming blurred, with customers expecting services like click-and-collect as standard. One report estimates that US shoppers spent $72.46 billion on click-and-collect in 2020.

A new breed of eCommerce providers has sprung up to cater to this omnichannel world where the frontend requirements are getting increasingly complex.



Frontend developers have adopted a wide range of programming languages to meet digital experience challenges. Whether it's Vue, React, or Angular, all these languages have their own ecosystem of developers, requirements, and standards.

It now matters what language your frontend experience is built on.

Enter headless commerce

The old generation of monolithic eCommerce platforms were designed with a frontend and backend that were completely dependent on one another. As the frontend layer was hard coded together with the backend, it was difficult to separate them.

As the needs of online retail have evolved to accommodate different frontend languages, monolithic providers have tried to adapt by 'sawing off' the top layer and building APIs (more on these later) to connect their backend with whatever frontend framework their customers are using.

But behind-the-scenes, the services that these providers offer are still intertwined with each other in ways that can be difficult to untangle. It means that even if you only want one service from an eCommerce provider, they may have to deploy their whole platform.

This is a problem for brands looking to headless to provide them with a more agile, composable tech stack.

Luckily, the new breed of eCommerce platforms are designed with a headless architecture, providing eCommerce services that are designed from their inception to be agnostic about the language the frontend site is built on.

The backend is a completely separate layer from the frontend, and the two communicate via APIs.

The best of these platforms are microservices-based, meaning they can offer retailers a range of services on a standalone basis. These microservices are linked to each other and to the frontend by APIs, simplifying the architecture of your overall eCommerce solution.

Key differences between headless and traditional commerce

Headless commerce emerged to meet the changing needs of an increasingly complex online retail environment. As such, headless solutions are more flexible, faster, and more agile than traditional commerce.

As an introductory rule, there are three major differences between headless commerce and a traditional, monolithic eCommerce solution:

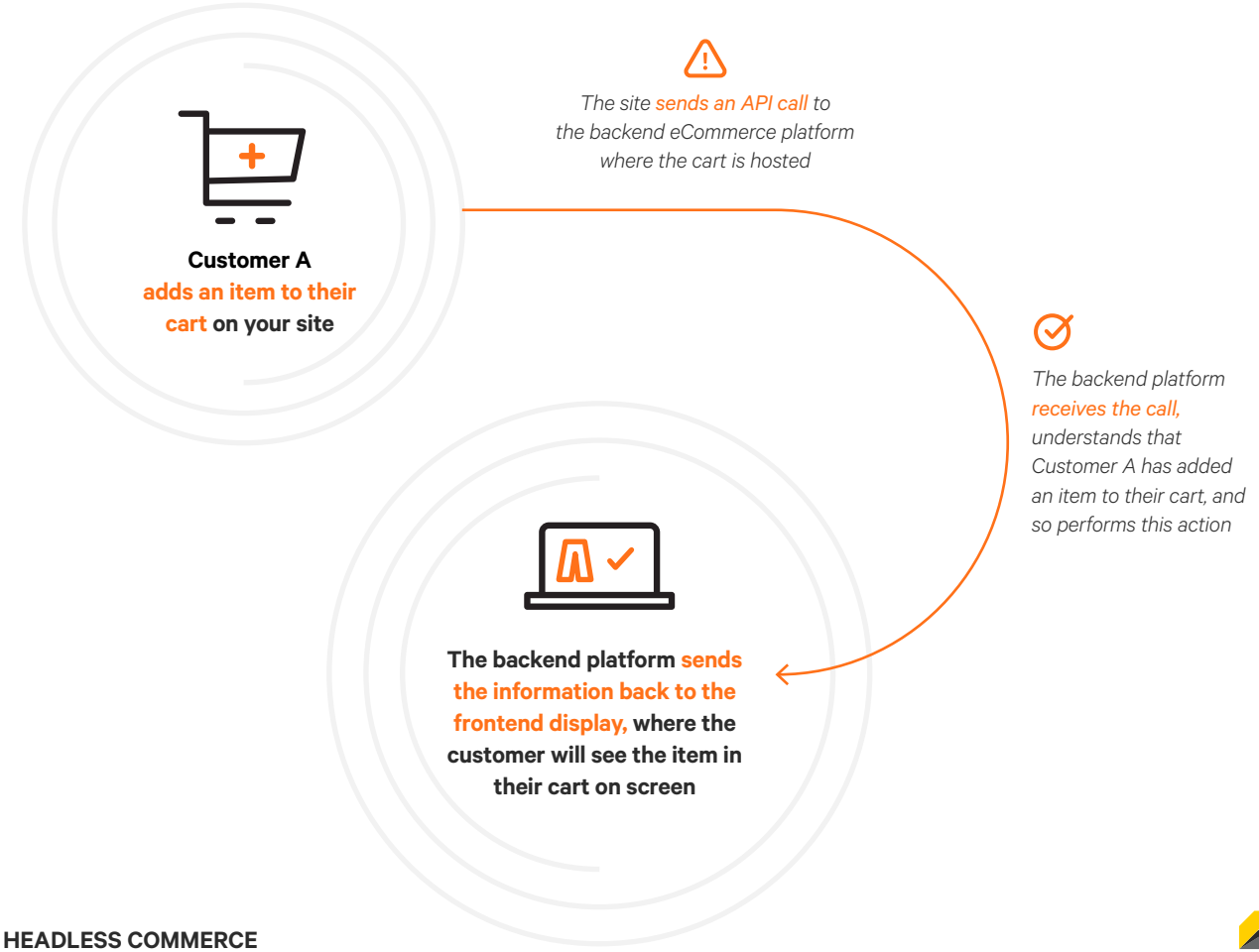
Headless Commerce	Traditional Commerce
Backend management and frontend delivery are separated	*Backend and frontend are hardcoded together and cannot be separated*
Often offered on a microservices basis	*Services are bundled and can't be offered separately*
Communication via APIs	*As the whole system is hardcoded together, APIs are not used*

Roles of APIs

APIs are a key component of headless. Their capacity to let two separate pieces of software, built-in different programming languages, talk to each other is what allows the two elements of an eCommerce solution to work together even though the frontend and the backend are separated.

The frontend layer can request data or trigger actions by making an API call to the backend, which is then able to process the request and send the relevant information back.

Let's take an example:





This is a simplified example, in the actual workings of a microservices-based platform (see the chapter on How to Evaluate Headless eCommerce Platforms), the cart function can also send API calls to other backend services, such as catalog and promotions.

It means that services are separated from each other and can be implemented on a modular basis, but can work together behind the scenes.

It makes it much easier to make changes to one part of your tech stack without affecting the whole of your eCommerce operation.

Because APIs allow programs built in different languages to talk to each other, it means you can design your digital experience in your CMS and push it out to your various channels simultaneously, regardless of what language they're built on.

How this impacts customer experience:

INSTANT UPDATES All customers experience changes immediately, rather than some (maybe using your mobile site or other device) having to wait hours for the update to filter through to them.

HIGH-QUALITY DIGITAL EXPERIENCE Updating your channels via APIs frees your marketing teams from the worry that some frontend frameworks won't support changes. They can be more inventive and experimental with content ideas, resulting in an enriched experience for your customers.

The Business Case for Headless

With all the technical details to consider, it's easy to lose sight of the reason you're considering going headless - that it makes business sense.

There's a compelling business case for an eCommerce solution built to meet the needs of an omnichannel world and capable of integrating with almost any frontend environment. A comprehensive survey of 46,000 shoppers found that omnichannel customers "spent an average of 4% more on every shopping occasion in the store and 10% more online than single-channel customers."

Advantages of headless commerce - and a note of caution

Advantages

- **FASTER SITE SPEED** The frontend can access data and services as needed from a central storage hub, reducing page-load times and delivering content to the customer significantly quicker than monolithic commerce.

- **FASTER TIME TO VALUE** You can make changes quickly and realize business gains faster. Regional variables like language and currency are easy to update while still maintaining consistency of digital experience.

  Built for omnichannel: Delivers fast and consistent digital experiences regardless of channel or device. This is especially important on mobile, where load times are still slow, making visitors more likely to bounce.

- **MORE AGILE** Separating functionality makes headless solutions more flexible and adaptable to the needs of online brands. It means that updates are easier to implement and don't require coding changes or re-deployments.

- **SIMPLER TO MANAGE** A separate backend can be hosted by your eCommerce platform provider, so you don't have to manage scaling capacity to deal with sudden surges of traffic around holidays and promotions.

- **LOWER CUSTOMER ACQUISITION COSTS AND HIGHER CONVERSION RATES** There's a premium on creating unique customer experiences that are capable of driving organic traffic to your channels and forging a durable relationship with your customers. According to PWC, 73% of consumers say CX is an important factor in making purchasing decisions



- **GREATER CONTROL** APIs allow you to orchestrate services so that you have end-to-end control over all aspects of your sales channel - from what customers see when they arrive on your site, to what fulfillment center their order is routed through, to when and how it is shipped to them.
- **LOWER TOTAL COST OF OWNERSHIP (TCO)** As the backend hosting (with all the attendant space, energy, and labor requirements) is taken off your hands, moving to a 3rd-party SaaS provider reduces the overall cost of maintaining your tech stack.
- **BEST-IN-CLASS CAPABILITIES** Faster feature innovations and the ability to choose best-of-breed solutions.

Cautions

- **HEADLESS CAN BE COMPLICATED,** even with a platform provider doing some of the work for you. Make sure you pick a vendor that is able to deliver services out-of-the-box and provide ongoing support.
- **LOTS OF HYPE AROUND THE TECHNOLOGY,** so it can be hard to know what aspects are genuinely beneficial.
- **DIFFERENT MODELS ON THE MARKET,** all with different architectures. It can be hard to know what the right one is for your brand.

As long as you're aware of these notes of caution and arm yourself with the right knowledge, you can easily leverage the many benefits of headless commerce.

How to Evaluate Headless eCommerce Platforms

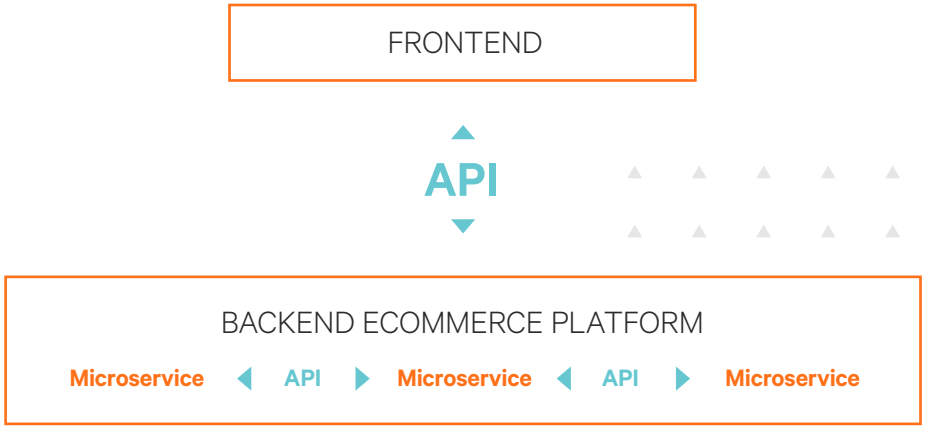
The sheer volume and technical density of the literature written on the subject of headless can be intimidating to the non-technical expert.

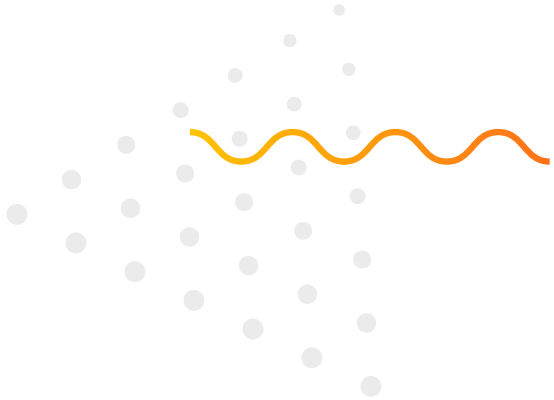
In this section we'll take a look at the mechanics of headless commerce platforms, helping orientate you as you navigate through the various options.

It might help to circle back to the definitions at the start of this guide for a refresher on some of the specialist terms used.

Things to look for in a headless solution

Because of their superior performance and modular architecture, the first specification that many online brands look for in their prospective headless solution is that it is microservices-based.





These platforms have been conceived and built to meet current use-cases. As such, they take APIs as a starting point for their whole design. These platforms are sometimes referred to as 'API-first' or having a microservices architecture. With a repurposed legacy enterprise platform, you may have to disentangle several auxiliary backend functions that you don't necessarily want, but need to be deployed behind the scenes to make it work.

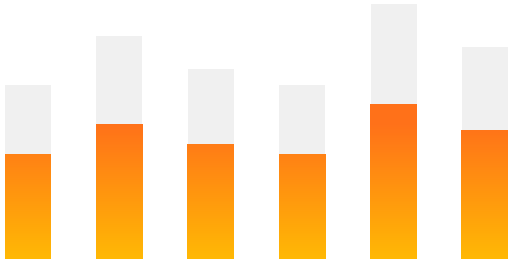
Having those monolithic functions present behind the scenes can have performance implications. They can also affect the degree to which your headless architecture is future-proofed. You may not want or need a best-of-breed modular approach across all business functions right away, but having a composable architecture will allow you to painlessly swap out services as your business and the technology evolves. A modular approach equals more flexibility.

But even if you do opt for a modular approach, you'll still have to decide whether you want a commerce platform that comes pretty much in one piece, that you can then adapt to your business needs, or whether to use modular offerings to create your own bespoke solution. As a recent Forrester report argued, "'Customize or compose' is the new 'build versus buy'".

Again, flexibility and speed are improved with a composable solution. You can easily extend services as you need them without creating a drag on your site performance (see the below section on extensibility for more).

Another big thing to look out for is whether your eCommerce solution comes packaged with a **headless CMS**. Some retailers prefer to build their tech stack themselves and use one of the many powerful headless CMSs now available (Contentstack, Contentful, Amplience, etc).

Others prefer to use their platform provider's 'in-house' CMS. Doing so can simplify your architecture and reduce hosting and management costs. If you're thinking of going this route, make sure your eCommerce platform offers a headless CMS and integrated page-builder that provides a single point of access for your teams.



But these are only the first steps.

When evaluating microservices-based solutions, you should evaluate the:

- **DEPTH OF MICROSERVICES**
- **WIDTH OF MICROSERVICES**
- **EXTENSIBILITY**

Depth

The process of constructing your tech stack is a little like building a house. Your headless platform needs to provide you with a solid foundation. But instead of timber and stone, the foundation consists of the basic services needed to make your site run smoothly and to maintain an excellent digital experience.

Much of the backend architecture needed to make these services work quickly, smoothly, and at scale is time-consuming to construct and costly to maintain.

So when comparing headless platforms, look at what services are provided 'out-of-the-box.' This is a commonly used term in eCommerce and it refers to those services and features that can be used as soon as you deploy the software.

Any services that are provided out-of-the-box are ones that your development teams don't have to build for themselves. You can let the 3rd-party eCommerce platform host and maintain them while you focus on the things you do best - creating breathtaking content and getting features to your customers faster.

Customize out-of-the-box services

Even though you should look for services that are already ready to go, with as little blood, sweat, and tears spilled over backend considerations as possible, you should still make sure you have the ability to customize these functions if you need to.

Out-of-the-box should not mean rigid and cookie-cutter

It's completely possible for a headless platform to provide hosting and maintenance for a range of services and still allow you to make customizations to them to serve your unique needs.

Breadth

As well as providing services with a robust foundation, headless platforms need to provide a wide range of services. Compare the breadth of services on offer from each platform and make sure they have what you need.

But platforms that offer a wide range of services also need to get the basics right. Gartner ranked Kibo Commerce highest for core commerce capabilities in their 2021 Critical Capabilities for Digital Commerce report, meaning they were the best ranked for providing the core functions of "catalog, search, basket, promotions, payment, customer account."

Extensibility

eCommerce platforms are built to serve a range of use-cases, and headless platforms are no different. 3rd-party providers want to serve as much of the market as possible - you won't come across a vendor that makes commerce platforms solely for farming machinery manufacturers, for example.

As such, these platforms need to design and build their services so that they can be used by many different types of online businesses, whose needs differ to a greater or lesser extent.

That approach works great for the most part - every company that sells online will have a cart function, regardless of whether they're a B2B microchip manufacturer or a DTC fashion brand.

But that may be where the similarities end. Our B2B seller might require a unique cart function that displays their product's technical specs, or notifies the customer if two products they've added to their cart are not compatible with each other.

This is a specialist function that wouldn't be necessary for the fashion brand (though they might have other, equally unique requirements) and it would be fruitless for the business to look for this function as a standard feature of an eCommerce platform.

This is where extensibility comes in.

Headless platforms should have an extensibility layer that can take the core functionality and modify, or extend it, to the unique requirements of your business.

When evaluating headless vendors, look for **managed extensibility**. This means the platform will host and manage the extension.

Managed extensibility has several benefits:

NO EXTRA COST The hosting and managing should come as part of the package.

IMPROVED SITE PERFORMANCE As the extension is hosted in the same place as the rest of the backend, there is no latency as information passes from your hosting environment to the eCommerce platform.

NO NEED TO WORRY ABOUT MANAGING AND SCALING If you take responsibility for hosting a customized service yourself, you need to make sure it can withstand the demands of sudden surges in traffic (Black Friday, for eg.) You also have the headache of looking after the day-to-day maintenance yourself.

How to Implement a Headless Solution

A step-by-step guide

The case for headless is clear but re-platforming your entire backend can be costly and time-consuming, and the complexity of the task might tempt you into thinking you can get by with the system you already have in place.

If you opt for a truly microservices-based platform, however, you can migrate to a new system at the speed that suits you. With the right platform, you can take a 'crawl, walk, run' approach to going headless.

Moving from monolithic to headless

Often, online brands using a monolithic, legacy platform have outgrown their capabilities but find that they're unable to move to a new provider all in one go.

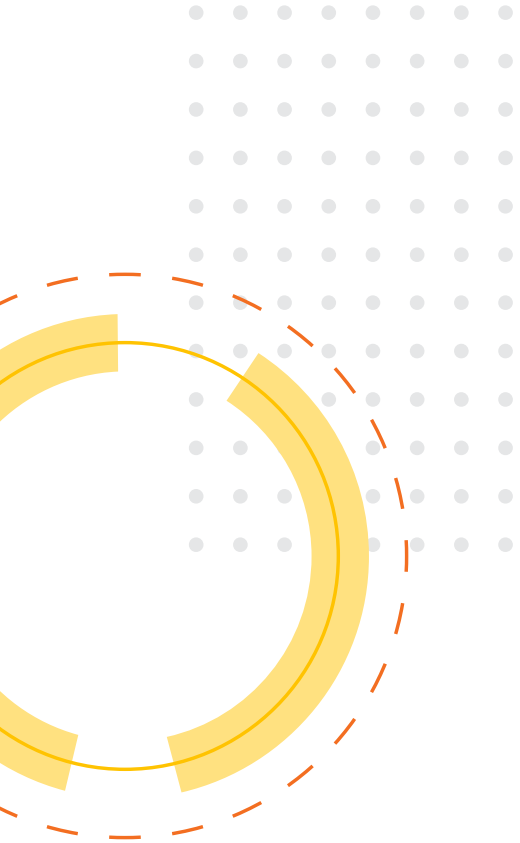
Doing a full-scale replatforming whilst maintaining a high standard of service for their customers proves impossible.

If you find yourself in this position, moving your backend functions over piecemeal can get you out of your predicament.

APIs allow services to be deployed independently of each other while linking them up on the backend, so the orchestration is already in place. This means that when you do come to implement further services, they'll work together.

Let's take an example:

1. Headless platform deploys the catalog service initially
2. Retailer ports their tech stack to the catalog service
3. Retailer's site connects with headless platform's catalog API
4. Retailer's site receives catalog functionality (pricing, promotions, merchandising, etc)
5. Headless platform then deploys iinventory service and the steps are repeated



A crawl, walk, run approach

Your provider can guide you through the process of migrating to a headless platform, holding your hand as much or as little as you need. If you already have a team of highly skilled developers who are keen to oversee much of the implementation and management of commerce functions, you can use your eCommerce platform sparingly - picking and choosing the services you want.

If you want a 3rd-party to do much of the heavy lifting for you but still want to host some eCommerce functionality yourself, a flexible backend provider will be able to accommodate your needs.

If you want to remove all the headaches associated with building and maintaining backend functions, a comprehensive headless platform will be able to offer this, along with complete developer support.

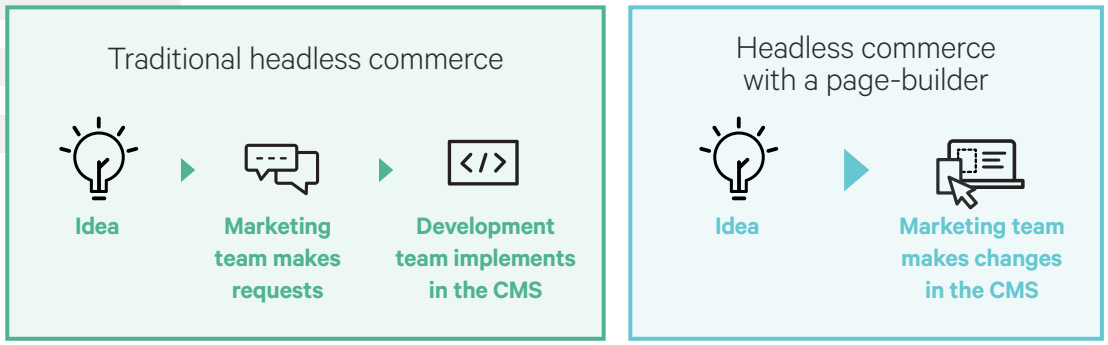
How your teams need to be set up

The key thing you need to work out when setting up your teams is how the development team and the marketing team will work together.

Exactly what the relationship is between these two departments will depend on what your tech stack looks like. As the richness of your customer experience grows, so does the volume and complexity of content in your CMS. Managing all this content so it is readily usable by your design and marketing teams quickly becomes beyond the capabilities of traditional CMSs for ambitious brands focused on the customer experience.

A headless CMS is built for omnichannel. As such, it's capable of ordering and grouping content intended for multiple channels, regions, and customer segments. Using one alongside a page-builder that non-development teams can easily access to make changes to pages simplifies your workflow.

If you don't have a WYSIWYG page-builder that can work with your headless CMS then your marketing team will be reliant on the development team to implement changes in the CMS.



What this re-arrangement of teams means:

DEVELOPER RESOURCES ARE PROTECTED Less time spent on executing content change requests means you can devote more resources to areas that will move your business forward - testing and experimenting with coding changes and site optimization, for example.

COMPREHENSIVE BUY-IN FROM TEAMS Working with page editors is a lot more user-friendly than staring at lines of code. Accessibility improves, aiding business-wide buy-in.

Conclusion

The separation of eCommerce functions that is the foundation of headless commerce is a powerful innovation and one that can unlock the creativity of your content teams. The freedom that a front-end agnostic platform provides means faster, richer, and more comprehensive digital experiences for customers.

Headless can be complicated - the language doubly so - but a thorough understanding of the concepts behind the technology arms you with the knowledge you need to make the move to a headless future.

