Unleashing the Power of Headless Architecture: Benefits and Drawbacks

Practical Logix

Description

As devices and technologies improve, customer expectations are changing towards increasingly individual and tailored user experiences. The topic of headless commerce is becoming more and more critical due to better adaptability, speed, and control options. Therefore, in this blog, we want to take a closer look at the topic.

On one hand, customers want e-commerce businesses to keep up with the new technology development of infrastructures and end devices: the performance, i.e., the loading and structure of eCommerce store offers, must take place in real-time. On the other hand, leading e-commerce platforms set the benchmark in terms of usability, product presentation, and functionality. Whether sophisticated search and filter functions, an appealing landing page, or an interactive product finder – today's online shop has to be up to date in various respects.

The Concept Behind Headless eCommerce

First, headless commerce is the idea of ??decoupling the front-end developers in e-commerce – i.e., the various digital touchpoints – from the backend infrastructure – i.e., the logic and data storage. This enables companies to combine, aggregate, and publish content and features across differentplatforms, channels and touchpoints.

A headless commerce architecture connects the presentation layer to various backend services via an API. The head, i.e., what the visitors see, is the front end. APIs are the neck, and the body represents, for example, the product catalog, the shopping cart functionality, or the payment process.

Since the front only communicates via the APIs, different front ends can be connected to the corresponding interfaces. For example, this app can be developed natively for iOS or Android. A PWA (Progressive Web App) is also a way to provide a front end – in this case, via an eCommerce platform.

Other possible touch points could be a POS system, an IoT solution, or other third-party software such as a marketplace or social media platform that also accesses the APIs.

The logic, such as checking availability, calculating a shopping cart total, or validating product configurations, takes place in the backend systems. These work independently and are usually designed for scalability and performance.

The front end can communicate with the backend interfaces in real-time, which only has to be loaded once onto the user's end device. This allows for significantly better performance and short reaction times, for example, when a product variant is selected.

Practical Logix

Key Differentiating Factors:

- Headless separates frontend and backend in development
- APIs form the basis for communication between frontend and backend
- Various touch points can be connected via the APIs
- Logic is mapped via the backend
- Backend systems are specialized

The Pros of Headless Commerce

The e-commerce platform is divided into logical components through the clear separation of frontend and backend and standardized APIs and data exchange formats such as REST or GraphQL. This brings several advantages for users from a development point of view:

1. Decoupled development and testing

Headless systems are decoupled; therefore, businesses can experiment without worrying about slowing down their online stores. A new user experience can be implemented without touching the core system, such as changes on a product detail page. Changes via A/B tests can also be carried out in parallel since the actual logic, and data structure is not interfered with.

2. Personalization

The personalization of content, product offers, prices, or other information can also be solved better with a headless approach. Content may be tailored to demographics (gender, geographic location, age, etc.), user information, and interactions while on the site.

By connecting identity management or a central user database, personalization can be carried out based on the login data. For example, a Digital Experience Platform (DXP) aggregates the various APIs based on different multipliers and personalizes the corresponding data.

3. Performance and User Experience

The customer experience, also known as user experience, comprises various factors. A key factor is performance, i.e., the speed with which page content is loaded or how a website reacts to interactions (e.g., setting filters in a list).

With a headless approach, the front end communicates with a backend, which only provides the relevant data. If, for example, the “Add to shopping cart” button is clicked or a new category is called up, the backend returns the corresponding data in a structured form.

Practical Logix

This data – usually provided as XML – is then interpreted and displayed by the front end. If the quantity of products in the shopping cart or the calculated total price changes, only this data has to be updated, and not the entire website has to be reloaded.

This approach ensures a very high loading speed and changes without time lag. In particular, there is no need to reload a website when changes are made, which in turn, has a positive impact on the user experience.

4. Efficiency in development

Headless architectures decouple visualization and logic so that different development teams can work better on a standard product: While the teams are working on both the frontend and backend code, content is already being maintained, and product data is structured in parallel.

A decoupled architecture allows organizations to make changes faster on their frontend framework without impacting the backend and vice versa. Due to the openness of the architecture, new functions and integrations can be implemented in a much shorter time. New functions can be provided as micro-services separately from the core system and implemented in the front end as the marketing team isn’t sufficiently reaching out to people.

The display layer communicates with the application layer via APIs since they are separate systems. This method can create a data cache layer between the presentation and application layers, thanks to a traditional CMS architecture. The cache layer can cache any requests for product catalogs or static content. Data loads much faster than a full-stack application, and the system can handle much more traffic.

5. Flexibility

Headless eCommerce allows for endless flexibility and customization by separating the frontend and back ends, so many tasks only require front-end developers to make desired changes. With a decoupled front end, changes are possible with significantly less effort, regardless of whether they are complex adjustments such as creating an individual checkout process or simple extensions such as adding a field to a customer account.

Cons

Of course, there are also limitations and disadvantages when building a headless architecture, especially when comparing a corresponding solution with out-of-the-box products:

1. Front-end development resources

Since the headless commerce systems initially do not have their front end, resources for development must be planned. Businesses need IT and development support to ensure the APIs communicate effectively with the backend systems. However, once the platform has been implemented, employees can change content on the websites or in the shop without needing the support of the development teams.

Practical Logix

2. Content abstraction

Editing content in a live preview environment, commonly known as WYSIWYG (what you see is what you get), is not as easy in the headless context. Content management is wholly detached from the design since website content, or product data is reusable in different channels. This requires a higher level of abstraction from editors and marketing staff.

Conclusion

In today's complex, digital world, above a specific size, there is hardly any way around a headless architecture. As soon as several digital channels are to be used, content and product data have to be played out in different touch points. Or time-to-market in developing new features is essential, headless commerce makes sense.

In the development and implementation of new functions, in particular. A large number of APIs from external systems can be used with a headless concept. Whether it's search, personalization, payment, or validation of data. A headless architecture makes it much easier to implement the interfaces of external web services and to use.

Practical Logix

Also Read: BUY VS BUILD: FINDING WHAT IS RIGHT FOR YOU

Category

1. Engineering
2. Product Strategy
3. Technology

Tags

1. Application Development
2. Architecture
3. Content Management Systems
4. Decision Making

Date
2026/11/08
Date Created
2022/08/20
Author
anand