

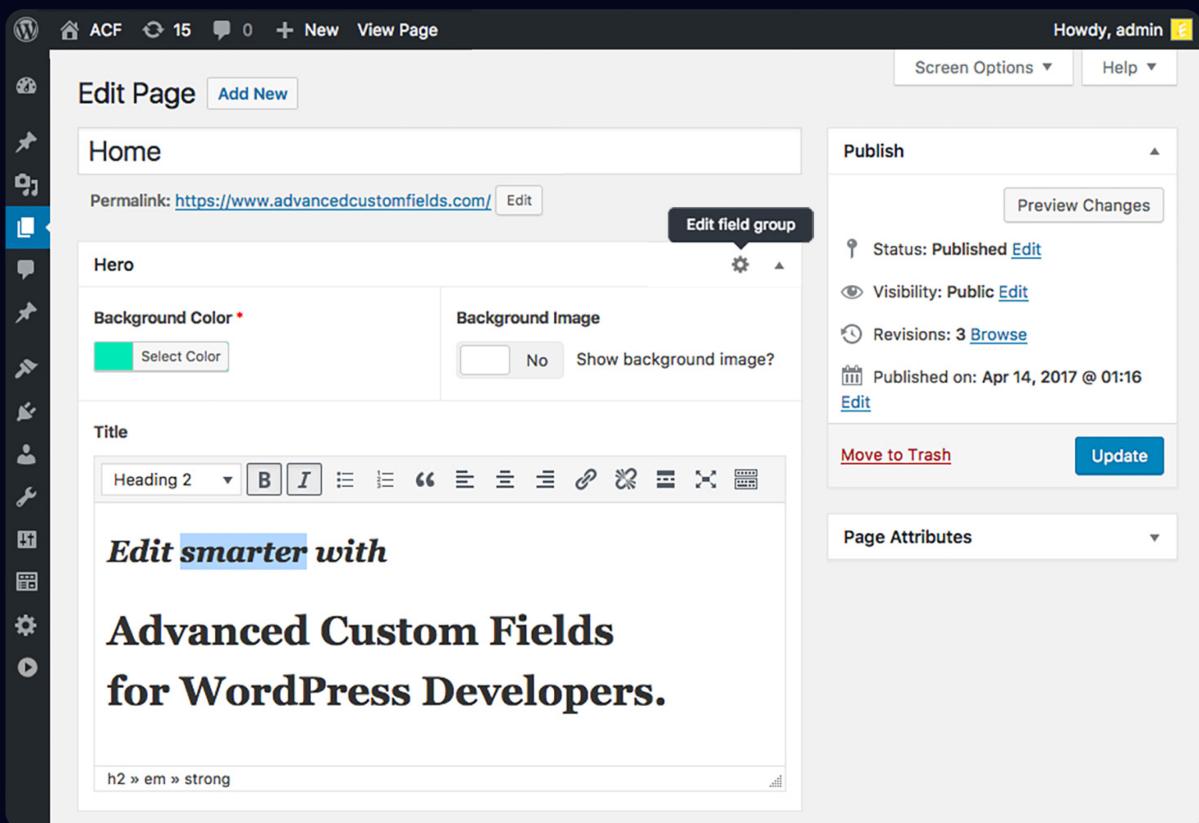
Remplacer Wordpress par Nuxt

en 5 étapes



1. Déjà, pourquoi 90% des sites utilisent WordPress

Le principal avantage de WordPress est son **back-office**, une interface qui permet aux utilisateurs de modifier le contenu du site web sans toucher au code, notamment grâce à des plugins comme ACF (Advanced Custom Fields). Cela rassure généralement les clients, car c'est une interface qu'ils connaissent, bien qu'elle commence à dater un peu.

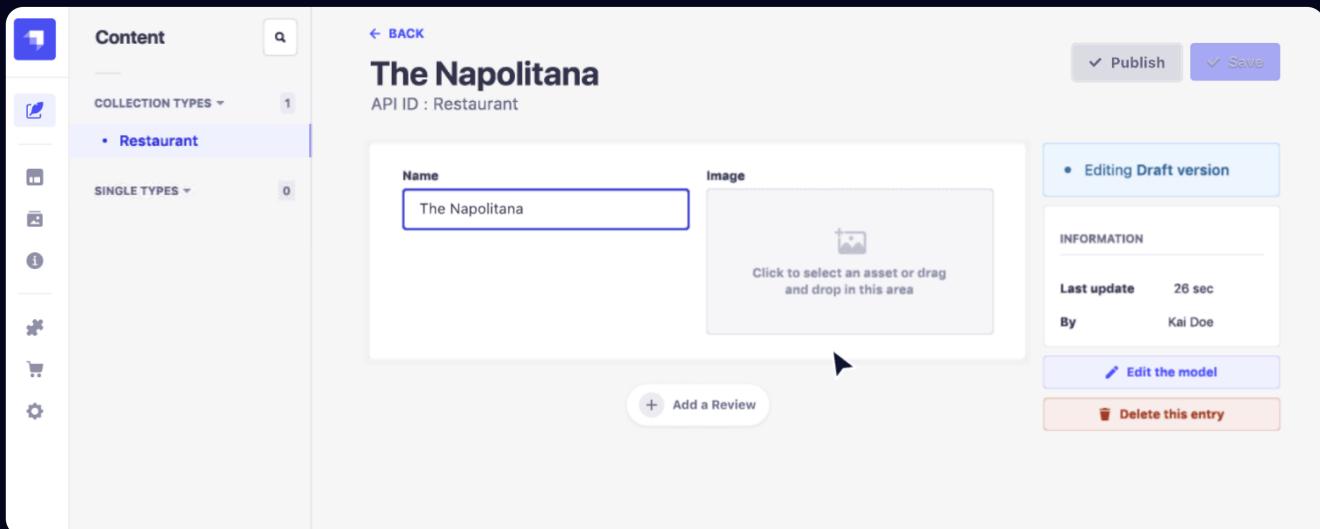


2. Pourquoi utiliser Nuxt à la place

- **Performance** : Les sites WordPress sont souvent très lents, notamment en raison des rechargements de page à chaque action et de l'utilisation excessive de plugins, qui réduisent également les performances. À l'inverse, Nuxt, en tant qu'application monopage (SPA), offre une expérience plus fluide sans recharge complet des pages.
- **Expérience développeur (DX)** : Nuxt, basé sur un framework JavaScript comme Vue, permet de créer des expériences interactives avec un code plus simple et plus léger à écrire. Cela facilite le développement d'interfaces riches et dynamiques, par exemple, si l'on souhaite créer un composant pour permettre aux utilisateurs de tester un produit directement sur une landing page.
- **Sécurité** : Les sites WordPress, étant basés sur une architecture très répandue et utilisant souvent des plugins peu maintenus, sont vulnérables à de nombreuses failles de sécurité. Nuxt est beaucoup moins exposé à ces risques en raison de son écosystème plus sécurisé et de sa structure moderne.

3. Back-office avec un CMS Headless : Strapi

Le problème de **Nuxt**, c'est qu'il n'intègre pas nativement de back-office permettant au client de gérer le contenu. C'est pourquoi il est nécessaire d'utiliser **Strapi**, un back-office indépendant qui offre toutes les fonctionnalités d'un CMS comme WordPress, avec des possibilités encore supérieures. Les données sont ensuite exposées sous forme d'**API**, que l'on peut facilement consommer dans notre application Nuxt. Ces données peuvent même être réutilisées dans d'autres applications, car le back-office et le front-end sont séparés. C'est pour cela que Strapi est qualifié de **CMS Headless**, car il est indépendant du front-end.



4. Utiliser le module Strapi avec Nuxt pour récupérer du contenu

Nuxt a développé un **module** permettant de connecter Strapi à son application avec une **configuration très légère**. Cela évite de faire des appels API complexes, en offrant la possibilité d'utiliser des fonctions comme **find** ou **findOne** pour récupérer les données, grâce au composable **useStrapi()** fourni avec le module.

```
<script setup lang="ts">
import type { Restaurant } from '~/types'

const { find } = useStrapi()

const response = await find<Restaurant>('restaurants')
</script>
```



Nuxt Strapi

5. Déployer le back-office Strapi et l'application Nuxt sur un VPS

La dernière étape sera de **déployer** les applications en ligne. Contrairement à un site WordPress où il n'y a qu'un seul projet, ici, il y en a deux. Pour cela, de la manière la plus simple et la moins coûteuse, il est possible de prendre un **VPS à 5 €** par mois chez OVH, Hostinger ou ailleurs, et de déployer les deux applications dessus.

Il faudra simplement prendre en compte les URLs dans les variables d'environnement pour la configuration de Strapi et du site web, afin qu'il soit ensuite visible pour tout le monde avec son contenu.

L'avantage, c'est qu'on peut **multiplier** les applications front-end ou même back-end sur ce même VPS et utiliser les données de contenu de Strapi dans toutes les applications. Cela est très avantageux pour une entreprise, car cela maintient une cohérence dans toutes les applications.

N'hésite pas à t'abonner si tu cherches d'autres astuces sur ces technos !

