REVISTA CIENTÍFICA
CONECTIVIDAD
Instituto Superior Tecnológico Universitario Rumiñahui

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

Soluciones de E-Commerce: Integración de Strapi y Next.js para la creación de una API Adaptable y Personalizable

E-Commerce Solutions: Strapi and Next.js Integration for the Creation of an Adaptable and Customizable API

Erick Roberto Arias Sánchez[1], Cristian David Muñoz Tenempaguay[2]

[1] Instituto Superior Tecnológico Internacional ITI, erick.arias@iti.edu.ec, Quito, Ecuador
[2] Instituto Superior Tecnológico Internacional ITI, cristian.munoz@ iti.edu.ec, Quito, Ecuador

Autor para correspondencia: erick.arias@iti.edu.ec

RESUMEN

El comercio electrónico ha evolucionado significativamente en los últimos años, requiriendo soluciones cada vez más adaptables y personalizables para satisfacer las necesidades de los usuarios. Este artículo presenta la integración de Strapi y Next.js para la creación de una API adaptable y personalizable, destacando su flexibilidad, eficiencia en la gestión de contenido y mejora en la experiencia del usuario. A través de la metodología ágil y herramientas como Pentest Tools y SonarQube, se desarrolló y evaluó una API que supera las limitaciones de las arquitecturas monolíticas tradicionales. Los resultados demuestran que esta integración ofrece una solución robusta, segura y escalable para aplicaciones de comercio electrónico, alineándose con las tendencias actuales en desarrollo web.

Palabras clave: API, Headless CMS, Strapi, Next.js, Comercio Electrónico.

ABSTRACT

E-commerce has significantly evolved in recent years, requiring increasingly adaptable and customizable solutions to meet user needs. This article presents the integration of Strapi and Next.js for the creation of an adaptable and customizable API, highlighting its flexibility, efficiency in content management, and improved user experience. Through an agile methodology and tools such as Pentest Tools and SonarQube, an API was developed and evaluated that surpasses the limitations of traditional monolithic architectures. The results show that this integration offers a robust, secure, and scalable solution for e-commerce applications, aligning with current trends in web development.

Key words: API, Headless CMS; Strapi; Next.js; E-commerce.

1. INTRODUCCIÓN

En la era digital actual, el comercio electrónico se ha consolidado como un pilar fundamental para la expansión de negocios a nivel global. La rápida evolución de las tecnologías y la creciente demanda de soluciones personalizadas han impulsado la necesidad de desarrollar APIs adaptables y personalizables, que permitan a las empresas integrar funciones y servicios de manera eficiente y escalable (Pinnis, 2022).

Se aborda la integración de Strapi y Next.js para el desarrollo de una API adaptable y personalizable, una solución innovadora que facilita la creación de aplicaciones web de comercio electrónico con un alto grado de flexibilidad y control. Strapi, como CMS sin cabeza (headless

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

CMS), proporciona una plataforma versátil para la gestión de contenido, mientras que Next.js ofrece una sólida base para el desarrollo del frontend, permitiendo la creación de interfaces de usuario optimizadas y dinámicas (Nandha Kumar & Anbumani, 2023).

La adopción de APIs personalizables en e-commerce permite no solo la gestión eficiente de contenido y usuarios, sino también la implementación de características avanzadas como la internacionalización y la seguridad de datos. Esto es crucial para competir en un mercado globalizado, ya que ofrece una mayor adaptabilidad a las cambiantes necesidades de los clientes y del mercado (León-Monar et al., 2023). La importancia de estas tecnologías se ve reforzada por su capacidad de mejorar la experiencia del usuario y optimizar las operaciones, proporcionando a los negocios las herramientas necesarias para adaptarse y crecer en un entorno digital en constante cambio (MINTEL & MPCEIP, 2021).

Este artículo tiene como objetivo presentar un análisis detallado de la implementación de esta API, destacando sus beneficios y aplicaciones prácticas en el ámbito del comercio electrónico. Además, se explorarán las consideraciones técnicas y metodológicas que sustentan su desarrollo, ofreciendo una visión integral de cómo las tecnologías modernas como Strapi y Next.js pueden ser utilizadas para impulsar la innovación en el comercio electrónico.

2. MATERIALES Y MÉTODOS

Para el desarrollo de la API adaptable y personalizable destinada a aplicaciones web de comercio electrónico, se utilizaron tecnologías modernas que ofrecen flexibilidad y escalabilidad. Las herramientas principales utilizadas en este proyecto fueron Strapi y Next.js. Strapi es un CMS sin cabeza (headless CMS) de código abierto que facilita la creación de APIs personalizadas, y Next.js es un framework basado en React que permite desarrollar aplicaciones web con una arquitectura eficiente y un enrutamiento sencillo (Pinnis, 2022; Strapi, s.f.).

Herramientas tecnológicas

Uno de los métodos más utilizados para la obtención del BC es la desagregación mediante biocatalizadores como enzimas (altamente especificas), permitiendo una alta calidad en la extracción de macromoléculas de origen proteico, lipídico o amiláceo. El uso de esta técnica permite obtener extractos solubles a partir de macromoléculas insolubles sin alterar las propiedades funcionales.

Strapi

Se eligió como CMS (Content Management System) debido a su naturaleza "headless," que permite una gestión de contenido centralizada y facilita la integración con diferentes frontends. Strapi ofrece una alta personalización y control sobre el contenido, lo que es fundamental para desarrollar una API que se pueda adaptar a las necesidades específicas de los usuarios. Además, cuenta con un plugin de internacionalización que permite la gestión de contenido multilingüe, esencial para la expansión global (Nandha Kumar & Anbumani, 2023).

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

Next.js

Se utilizó como framework para el desarrollo del frontend debido a su capacidad para generar páginas estáticas y dinámicas. Next.js permite la creación de interfaces de usuario optimizadas, mejorando la experiencia del usuario final. Su compatibilidad con Strapi facilita la integración de la API y la implementación de funcionalidades avanzadas (Next.js, s.f.).

Base de Datos PostgreSQL

Se empleó PostgreSQL como sistema de gestión de bases de datos relacional, debido a su robustez y capacidad para manejar grandes volúmenes de datos. Esto garantiza la integridad, seguridad y rendimiento del contenido gestionado por la API (PostgreSQL, 2023).

Otras herramientas

Se integraron tecnologías adicionales como Axios para realizar solicitudes HTTP de manera eficiente, Tailwind para el diseño modular y componible de la interfaz de usuario, y i18next para la internacionalización de la aplicación, adaptándola a múltiples idiomas (Axios, s.f.; Tailwind CSS, s.f.; i18next, s.f.).

Metodología

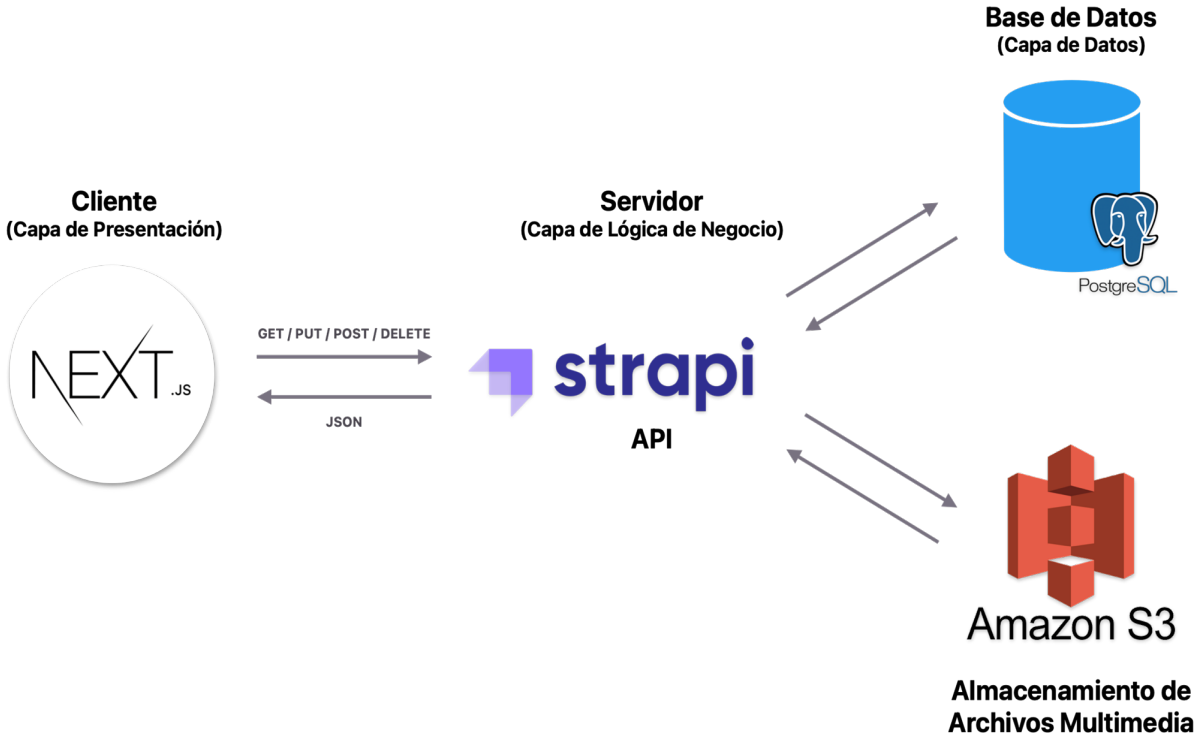
El desarrollo de la API se realizó siguiendo una metodología ágil, específicamente el marco de trabajo Scrum, que permitió iterar y mejorar continuamente el proyecto. La metodología ágil se centra en la entrega temprana y continua de software útil y facilita la colaboración estrecha entre los desarrolladores y los clientes (Singh Matharu et al., 2015).

Diseño de la arquitectura

La arquitectura del sistema se diseñó con un enfoque cliente/servidor, separando claramente el frontend y el backend. El frontend, desarrollado con Next.js, interactúa con el backend gestionado por Strapi para recuperar y mostrar datos. La separación entre el frontend y el backend permite una mayor flexibilidad y escalabilidad del sistema.

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

Figura 1. Diagrama de la Arquitectura Cliente/Servidor

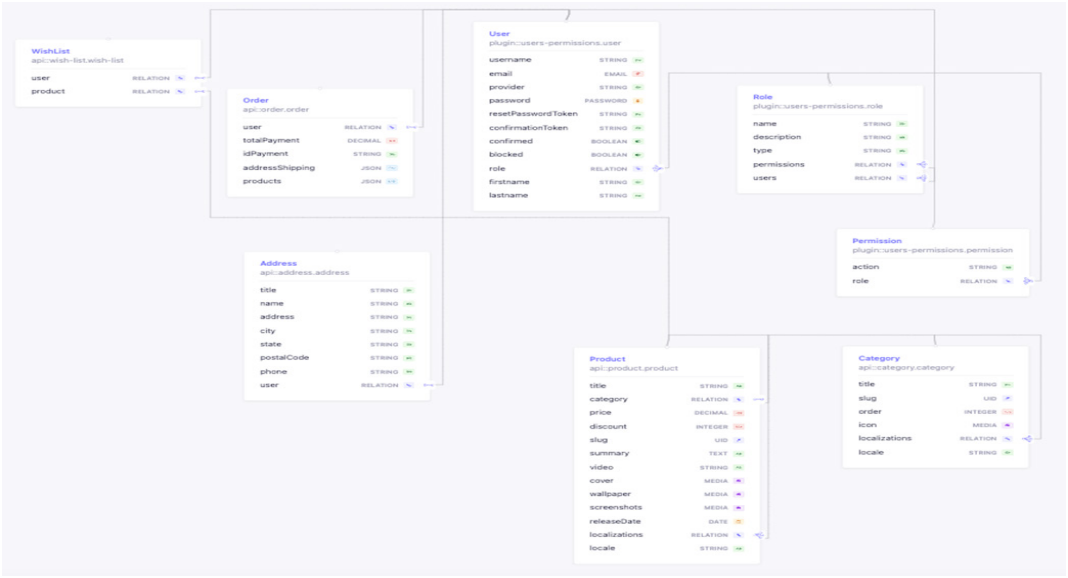


Fuente: Arias-Erick, 2024

Implementación

Durante la fase de implementación, se desarrollaron los módulos funcionales de la API, que incluyen la gestión de usuarios, productos, categorías, y métodos de pago. Se utilizó Strapi para definir los modelos de datos y establecer las relaciones entre ellos, lo que facilitó la creación y administración del contenido.

Figura 2. Diagrama de entidad relación



Fuente: Arias-Erick., 2024

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

Pruebas y validaciónt

Las pruebas unitarias y de integración se planificaron para validar las funcionalidades de la API y su correcta interacción entre los módulos. Adicionalmente, se realizaron pruebas de seguridad para identificar y corregir posibles vulnerabilidades. Herramientas como Swagger UI fueron utilizadas para generar documentación automática de la API, facilitando la validación y futura integración con otros sistemas.

Despliegue y mantenimiento

Una vez implementada y validada, la API fue desplegada en un entorno de producción. El mantenimiento se planificó para ser continuo, garantizando la adaptabilidad y rendimiento a medida que evolucionan las necesidades del negocio.

3. RESULTADOS Y DISCUSIÓN

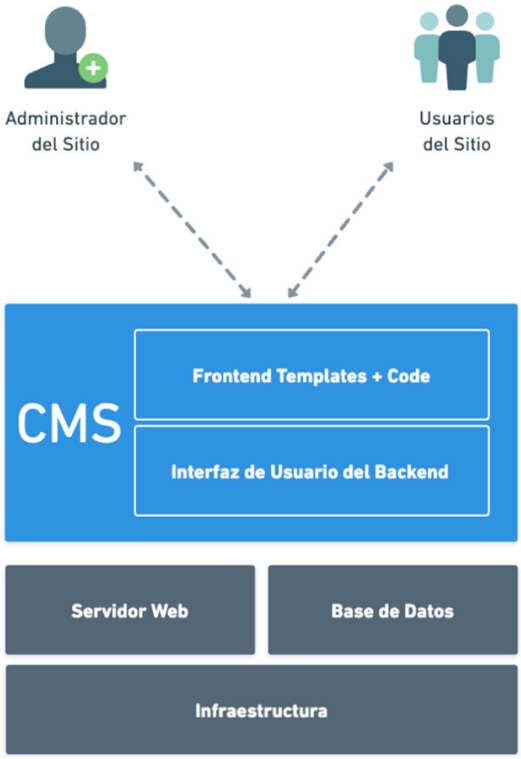
La integración de Strapi y Next.js resultó en una API adaptable y personalizable que supera las limitaciones de las arquitecturas de CMS monolíticos tradicionales. Esta sección presenta los principales resultados obtenidos durante el desarrollo y análisis de la API, y discute cómo esta solución mejora la gestión de contenido y la experiencia del usuario en aplicaciones de comercio electrónico.

Comparación entre CMS monolíticos y headless CMS

Una de las principales ventajas de utilizar un Headless CMS como Strapi es la separación entre la gestión del contenido y su presentación, lo que permite mayor flexibilidad y control. En las arquitecturas monolíticas tradicionales, la gestión del contenido y la presentación están estrechamente vinculadas, lo que limita la adaptabilidad y la personalización.

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

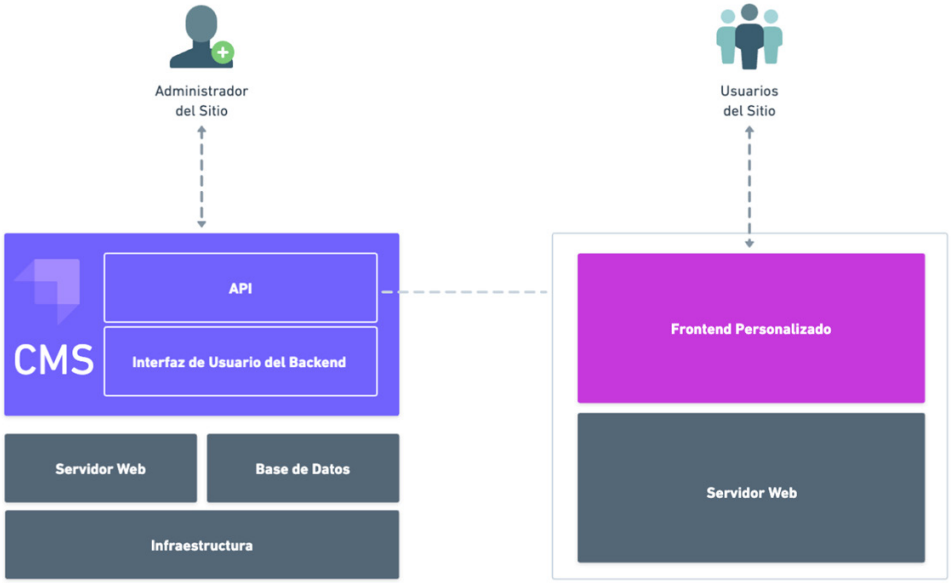
Figura 3. Estructura operativa de CMS Monolítico



Fuente: Arias-Erick, 2024

En contraste, la arquitectura headless de Strapi permite que el contenido sea gestionado de forma independiente y entregado a través de una API a cualquier frontend, como se muestra en la siguiente figura

Figura 4. Diagrama de procesos con Headless CMS (Strapi)



Fuente: Arias-Erick, 2024

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

La flexibilidad proporcionada por este enfoque permite a los desarrolladores integrar y personalizar las funciones de la API sin las restricciones impuestas por las arquitecturas monolíticas. Esta separación mejora significativamente la velocidad de desarrollo y la experiencia del usuario final (García et al., 2021).

Eficiencia en la gestión del contenido

La implementación de la API con Strapi y Next.js demostró mejoras notables en la eficiencia de la gestión del contenido. Gracias a la interfaz de administración de Strapi, los administradores pueden gestionar y actualizar contenido de manera intuitiva y rápida. La capacidad de Strapi para manejar múltiples tipos de contenido y relaciones complejas entre ellos se traduce en una gestión más eficiente y organizada de los datos (León-Monar et al., 2023).

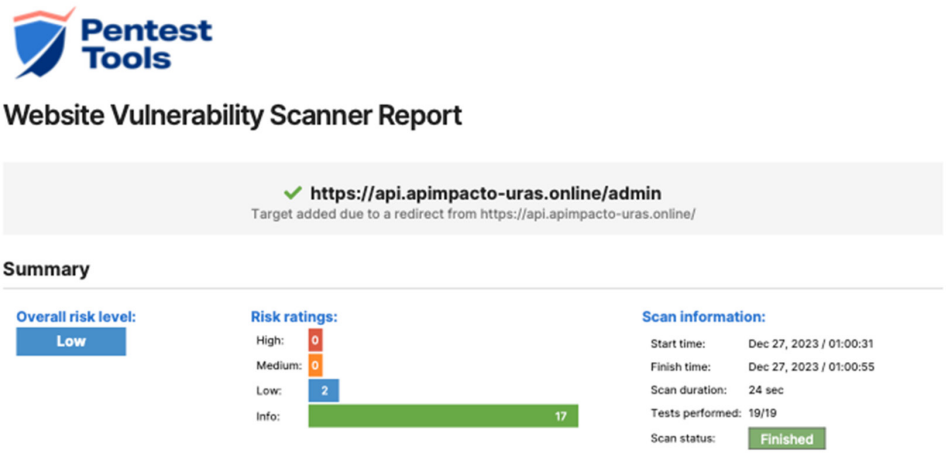
Los tiempos de carga y la mejora en el rendimiento de la aplicación fueron evaluados mediante pruebas de rendimiento. La arquitectura headless permitió la entrega de contenido de forma más eficiente, recuperando solo los datos necesarios para cada solicitud, lo que optimiza el tiempo de respuesta y reduce el consumo de recursos del servidor (Pinnis, 2022).

Evaluación de funcionalidad y seguridad

Las pruebas realizadas a la API confirmaron la correcta operación de los módulos implementados, como la gestión de usuarios y productos. La seguridad fue un aspecto crítico, y se implementaron medidas para proteger los datos y garantizar la integridad de la API. La autenticación y la autorización se configuraron para restringir el acceso a los recursos según los roles de usuario, siguiendo las mejores prácticas de seguridad en el desarrollo web (OWASP, 2023).

Para evaluar la seguridad de la API, se utilizaron herramientas como Pentest Tools y SonarQube. Las pruebas realizadas con Pentest Tools permitieron identificar y mitigar posibles vulnerabilidades, garantizando una mayor seguridad en la protección de datos y la integridad de la API.

Figura 5. Resultados de las pruebas de Pentest Tools

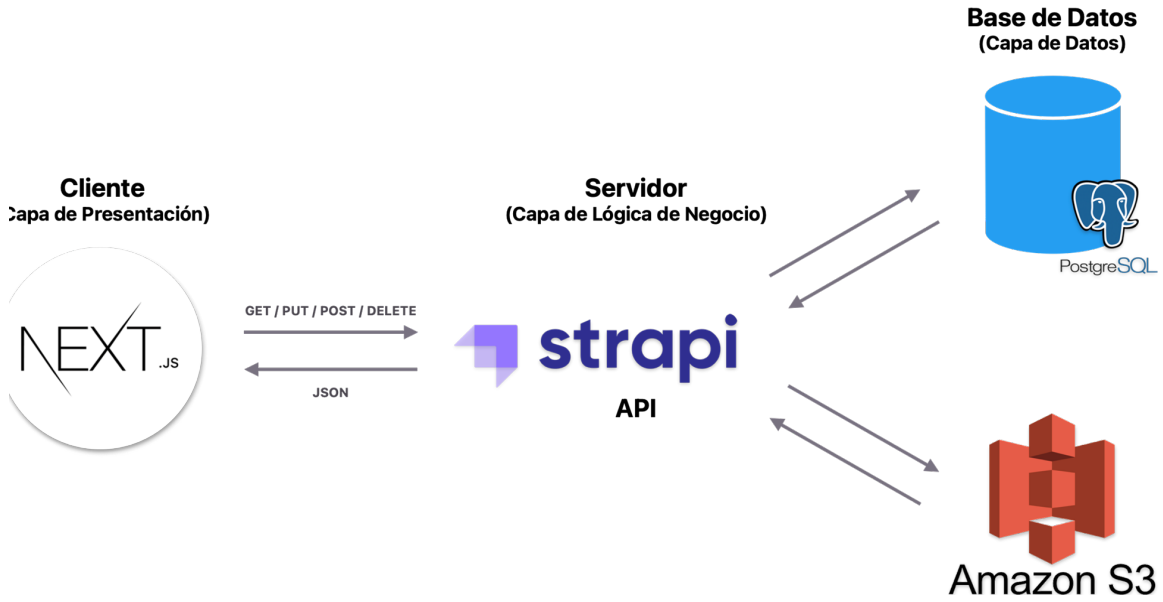


Fuente: Pentest-Tools, 2024

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

SonarQube se utilizó para analizar el código fuente en busca de posibles errores y problemas de seguridad. Esta herramienta ayudó a mejorar la calidad del código y a reforzar las medidas de seguridad implementadas.

Figura 6. Análisis de código con SonarQube



Fuente: SonarQube, 2024

Las pruebas confirmaron que la API es robusta y segura, cumpliendo con los estándares necesarios para aplicaciones de comercio electrónico. La implementación de controles de acceso y medidas de seguridad contribuyó a la protección contra ataques comunes y a la integridad de los datos.

Impacto en la experiencia de usuario y adaptabilidad

La integración de Next.js con Strapi mejoró notablemente la experiencia del usuario final. La capacidad de Next.js para generar páginas estáticas y dinámicas permitió una mayor velocidad de carga e interactividad en la aplicación, contribuyendo a una experiencia de usuario más fluida y eficiente (León-Monar et al., 2023). Además, la facilidad para gestionar contenido multilingüe y la personalización de la API posibilitaron una mejor adaptación a las necesidades de diferentes mercados y audiencias.

Los resultados demuestran que la integración de Strapi y Next.js ofrece una solución eficiente y adaptable para la gestión de contenido en aplicaciones de comercio electrónico. La flexibilidad proporcionada por la arquitectura headless y las mejoras en la experiencia del usuario hacen de esta integración una opción valiosa para emprendedores y desarrolladores. A diferencia de las arquitecturas monolíticas, la solución propuesta permite una gestión de contenido más dinámica y personalizada, alineándose con las tendencias actuales en el desarrollo web y el comercio electrónico.

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

4. CONCLUSIONES

La integración de Strapi y Next.js ha demostrado ser una solución eficiente y adaptable para la creación de APIs personalizables en aplicaciones web de comercio electrónico. Este estudio ha evidenciado cómo la arquitectura headless CMS supera las limitaciones de los CMS monolíticos tradicionales, ofreciendo una mayor flexibilidad en la gestión de contenido y una mejora significativa en la experiencia del usuario. Gracias a Strapi, se ha logrado una separación clara entre la gestión del contenido y su presentación, lo que facilita la integración con múltiples frontends y permite una personalización de la API según las necesidades específicas de cada proyecto. Esto se traduce en una mayor adaptabilidad y escalabilidad de las aplicaciones de comercio electrónico (Nandha Kumar & Anbumani, 2023).

Asimismo, Next.js ha contribuido a mejorar la experiencia del usuario al permitir la generación de páginas estáticas y dinámicas, optimizando los tiempos de carga y la interactividad de la aplicación. La interfaz de administración de Strapi simplifica la gestión y actualización del contenido, permitiendo manejar múltiples tipos de contenido y relaciones complejas de manera eficiente, lo que se traduce en una mayor eficiencia operativa (León-Monar et al., 2023). Las evaluaciones de seguridad realizadas con herramientas como Pentest Tools y SonarQube han confirmado que la API es robusta y cumple con los estándares necesarios para aplicaciones de comercio electrónico, implementando medidas de autenticación y autorización que contribuyen a la protección de datos y la integridad del sistema (OWASP, 2023).

La adopción de tecnologías como Strapi y Next.js se alinea con las tendencias actuales en desarrollo web y comercio electrónico, proporcionando una herramienta poderosa para el desarrollo de aplicaciones más flexibles, seguras y eficientes. Esta integración representa una evolución en la manera de gestionar y presentar contenido en la web, contribuyendo al éxito y crecimiento de negocios en el entorno digital (MINTEL & MPCEIP, 2021). En resumen, la solución propuesta ofrece beneficios significativos que la posicionan como una opción valiosa para emprendedores y desarrolladores en el campo del comercio electrónico.

REFERENCIAS

Axios. (s. f.). Getting Started | Axios Docs. Recuperado 24 de septiembre de 2023, de https://axios-http.com/docs/intro

Kumar, S. (2019). A Review on Client-Server Based Applications and Research Opportunity. International Journal of Scientific Research, 7(10), 3768. https://doi.org/10.24327/ijrsr.2019.1007.3768

León-Monar, P. de L., Rivadeneira-Ramos, E. P., Núñez-Aguiar, F. del R., & Albán-Trujillo, P. E. (2023). El comercio Electrónico en los Emprendimientos. 593 Digital Publisher CEIT, 8(4), 461-470. https://doi.org/10.33386/593dp.2023.4.1829

Ministerio de Telecomunicaciones y de la Sociedad de la Información, & Ministerio de Producción Comercio Exterior Inversiones y Pesca. (2021). Estrategia Nacional de Comer-

Fecha de recepción: 17 / 12 / 2024
Fecha de aceptación: 06 / 01 / 2025
Fecha de publicación: 23 / 01 / 2025

cio Electrónico. https://www.telecomunicaciones.gob.ec/wp-content/uploads/2021/05/ESTRATEGIA-NACIONAL_ENCE.pdf

Nandha Kumar, M. S. S., & Anbumani, M. P. (2023). Full Stack E-Commerce Application Using Strapi.io and Stripe Technology. International Journal of Research Publication and Reviews, 4(6), 3214-3220. www.ijrpr.com

Next.js. (s. f.). Next.js by Vercel - The React Framework. Recuperado 16 de julio de 2023, de https://nextjs.org/

OWASP. (2023). OWASP Top Ten Web Application Security Risks. Open Web Application Security Project. https://owasp.org/www-project-top-ten/

Pinnis, N. (2022). Modern Website Development with Strapi and Next.js. https://www.theseus.fi/bitstream/handle/10024/751348/Pinnis_Niko.pdf?sequence=2&isAllowed=y

PostgreSQL. (2023). PostgreSQL: The world's most advanced open source database. https://www.postgresql.org/

Singh Matharu, G., Mishra, A., Singh, H., & Upadhyay, P. (2015). Empirical Study of Agile Software Development Methodologies: A Comparative Analysis. https://doi.org/10.1145/2693208.2693233

Strapi. (s. f.). Strapi - Open source Node.js Headless CMS. Recuperado 16 de julio de 2023, de https://strapi.io/

Tailwind CSS. (s. f.). Tailwind CSS - Rapidly build modern websites without ever leaving your HTML. Recuperado 24 de septiembre de 2023, de https://tailwindcss.com/