Documentación técnica para migración web a tecnologías modernas

I. Introducción

Este documento presenta una justificación técnica para la migración de la plataforma web de su infraestructura actual, basada en WordPress, hacia una arquitectura moderna, escalable y de alto rendimiento. La decisión de considerar una migración surge de la necesidad de abordar ciertas limitaciones inherentes a la plataforma actual que impactan negativamente en la experiencia del usuario, el rendimiento del sitio, la seguridad y la agilidad del equipo de desarrollo.

La migración a una arquitectura moderna ofrece una solución integral, centrada en:

- **Rendimiento:** Optimizando drásticamente los tiempos de carga.
- **SEO:** Mejorando el posicionamiento en buscadores.
- **Agilidad de desarrollo:** Facilitando la creación e implementación de nuevas funcionalidades.
- **Mantenibilidad:** Simplificando el mantenimiento y las actualizaciones.
- **Experiencia del usuario:** Ofreciendo una navegación más fluida.
- **Seguridad:** Fortaleciendo la seguridad de la plataforma.

II. Estado actual de las webs

Las plataformas web de dorsia.es y clinicasorigen.es, ambas construidas con WordPress, presentan limitaciones técnicas comunes que afectan negativamente la experiencia del usuario, el rendimiento, la seguridad, el mantenimiento, la escalabilidad y la experiencia de desarrollo. A continuación, se detallan estas limitaciones, respaldadas por datos de rendimiento obtenidos con auditorías, y que se muestran en las capturas adjuntas en el documento:

A. Rendimiento Deficiente:

El rendimiento web es fundamental. Tiempos de carga lentos impactan negativamente la experiencia del usuario, el SEO y las conversiones. Los datos de rendimiento de ambas webs evidencian problemas significativos:

- **dorsia.es**
    - **First Contentful Paint (FCP):** Si bien este valor está dentro del rango aceptable, una optimización mejoraría la experiencia inicial del usuario.
    - **Largest Contentful Paint (LCP):** Este valor está al borde de lo aceptable y necesita optimización. El tiempo de carga del contenido principal es considerable.

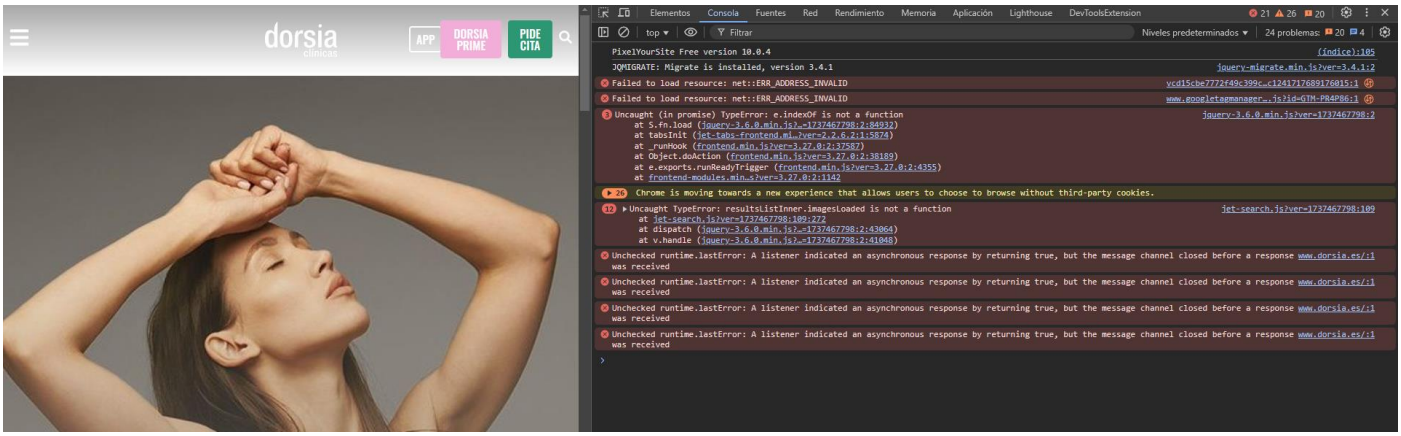
- **Time to First Byte (TTFB): :** Este valor es relativamente alto e indica una respuesta lenta del servidor.

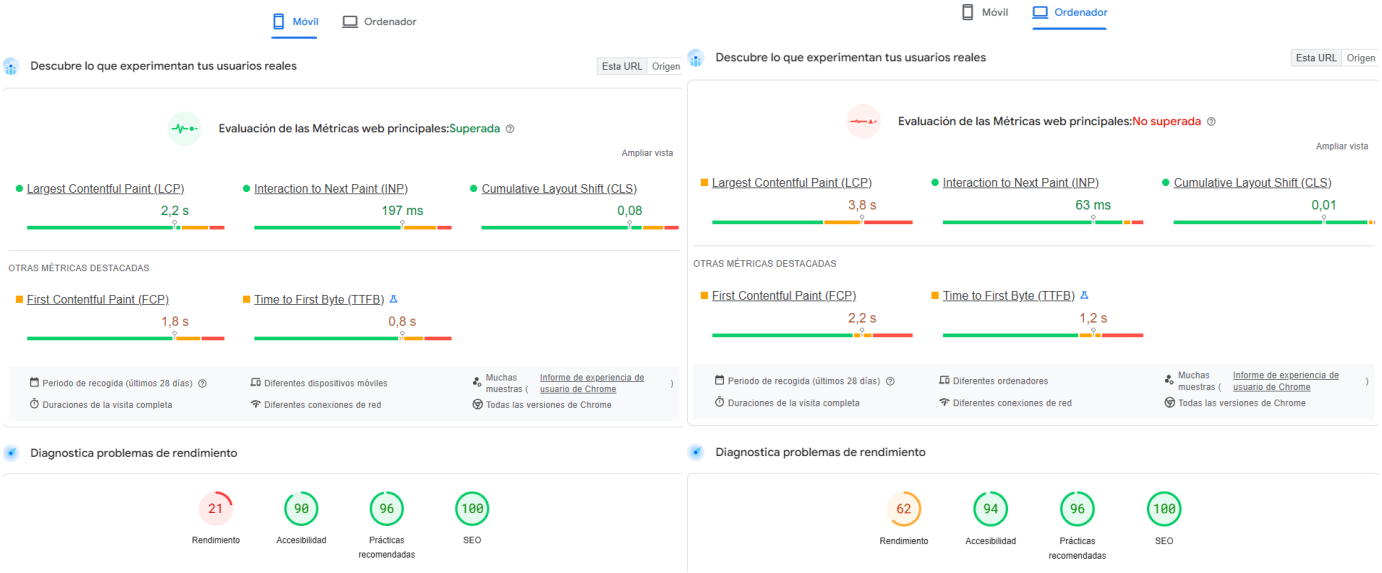
- **clinicasorigen.es**
  - **First Contentful Paint (FCP):** Este valor es superior al de dorsia.es y también requiere optimización.
  - **Largest Contentful Paint (LCP):** Este es el problema más crítico. El LCP supera significativamente el umbral recomendado, indicando una carga lenta del contenido principal.
  - **Time to First Byte (TTFB):** Este valor es aún mayor que el de dorsia.es, sugiriendo posibles problemas de rendimiento del servidor o configuración del hosting.

- Impacto en el Negocio:

- **Aumento de la Tasa de rebote:** Los usuarios abandonan sitios lentos, resultando en pérdida de potenciales clientes.
- **Disminución de las conversiones:** La lentitud dificulta la navegación y la realización de acciones importantes, como solicitar una cita, lo que reduce las conversiones.
- **Impacto Negativo en el SEO:** Google penaliza los sitios lentos, afectando la visibilidad orgánica.

- Dorsia





https://pagespeed.web.dev/analysis/https-www-dorsia-es/ovh8nba5vi?form_factor=desktop

- Origen

https://pagespeed.web.dev/analysis/https-clinicasorigen-es/h8qam5rtj3?form_factor=**desktop**

B. Seguridad vulnerable:

La arquitectura de WordPress, si bien ofrece flexibilidad y un amplio ecosistema, presenta vulnerabilidades inherentes que deben ser consideradas cuidadosamente:

- ## Vulnerabilidades en Plugins y Temas:

  - **Dependencia de terceros:** WordPress depende en gran medida de plugins y temas desarrollados por terceros. Si bien esto ofrece una gran variedad de funcionalidades y diseños, también introduce un riesgo significativo. La calidad del código y las prácticas de seguridad de estos desarrolladores varían considerablemente.

  - **Código malicioso o con errores:** Plugins y temas, especialmente aquellos desactualizados o provenientes de fuentes no confiables, pueden contener código malicioso (malware) o errores de programación que crean vulnerabilidades explotables. Estas vulnerabilidades pueden permitir a atacantes:

    - **Inyectar código malicioso (XSS - Cross-Site Scripting):** Este tipo de ataque permite a los atacantes inyectar scripts maliciosos en el sitio web, que se ejecutan en el navegador de los usuarios. Esto puede permitir el robo de cookies, el redireccionamiento a sitios maliciosos o la suplantación de identidad.
    - **Realizar inyección SQL:** Esta vulnerabilidad permite a los atacantes manipular las consultas a la base de datos, lo que puede resultar en el acceso no autorizado a información sensible, la modificación o eliminación de datos, o incluso la toma de control del servidor.
    - **Ejecutar código remoto (RCE - Remote Code Execution):** Esta es una de las vulnerabilidades más graves, ya que permite a los atacantes ejecutar código arbitrario en el servidor, lo que les da un control total sobre el sitio web.

  - **Falta de actualizaciones:** Los plugins y temas desactualizados son una de las principales fuentes de vulnerabilidades. Los desarrolladores publican actualizaciones para corregir errores de seguridad, pero si estas actualizaciones no se aplican, el sitio web permanece vulnerable a ataques conocidos.

- ## Dificultad en la Gestión de la Seguridad:

  - **Complejidad de la gestión de plugins:** Un sitio WordPress típico puede utilizar una gran cantidad de plugins, lo que dificulta el seguimiento de las actualizaciones de seguridad de cada uno.

  - **Incompatibilidades entre plugins:** Las actualizaciones de un plugin pueden generar incompatibilidades con otros plugins o con el propio WordPress, lo que puede causar errores en el sitio web o incluso dejarlo inaccesible.

  - **Falta de un sistema de seguridad centralizado:** WordPress no ofrece un sistema de seguridad centralizado que gestione automáticamente las actualizaciones de seguridad

de todos los componentes. Esto requiere una gestión manual y constante, lo que aumenta el riesgo de errores humanos.

- **Ataques de fuerza bruta:** Los atacantes pueden intentar acceder al panel de administración de WordPress mediante ataques de fuerza bruta, probando miles de combinaciones de nombres de usuario y contraseñas.

- **Vulnerabilidades en el core de WordPress:** Aunque menos frecuentes, también se han descubierto vulnerabilidades en el núcleo (core) de WordPress. Estas vulnerabilidades pueden afectar a un gran número de sitios web si no se aplican las actualizaciones de seguridad correspondientes.

Consecuencias de una brecha de seguridad:

Una brecha de seguridad en dorsia.es o clinicasorigen.es podría tener graves consecuencias:

- **Pérdida de datos sensibles de los usuarios:** Información personal, datos de contacto, historial médico, etc., podrían verse comprometidos.
- **Daño a la reputación de la marca:** Una brecha de seguridad puede erosionar la confianza de los clientes y dañar la imagen de la empresa.
- **Pérdidas económicas:** La reparación de los daños causados por un ataque, las posibles multas por incumplimiento de normativas de protección de datos y la pérdida de clientes pueden generar importantes pérdidas económicas.
- **Interrupción del servicio:** Un ataque puede dejar el sitio web inaccesible durante un tiempo considerable, lo que interrumpe la actividad del negocio.

C. Mantenimiento complejo y costoso:

El mantenimiento de WordPress puede ser laborioso y costoso a largo plazo:

- **Gestión de actualizaciones:** Actualizar WordPress, plugins y temas puede generar incompatibilidades y requerir tiempo de desarrollo para solucionar conflictos.
- **Dependencia de plugins:** La necesidad de plugins para funcionalidades básicas aumenta la complejidad y puede afectar el rendimiento.
- **Deuda técnica:** La acumulación de código personalizado y la dependencia de plugins pueden generar código difícil de mantener y actualizar.

D. Escalabilidad limitada:

WordPress no está optimizado para alto tráfico:

- **Rendimiento Degradado con Picos de Tráfico:** Ante un aumento repentino de visitas, el rendimiento puede degradarse significativamente, incluso causando caídas.
- **Configuraciones Complejas para Escalar:** Escalar WordPress requiere configuraciones avanzadas, como caché avanzado, balanceadores de carga y servidores dedicados, lo que implica mayor coste y complejidad.

E. Experiencia de desarrollo obsoleta:

El desarrollo con WordPress presenta desventajas frente a tecnologías modernas como JavaScript (React, Next.js):

- **Menor productividad:** El flujo de desarrollo es menos eficiente y ágil que con JavaScript y sus frameworks modernos.
- **Menor disponibilidad de talento:** El ecosistema de desarrolladores de JavaScript es más amplio y dinámico.

III. Solución propuesta

Para abordar las limitaciones de la plataforma actual basada en WordPress, se propone una migración a una arquitectura moderna y robusta basada en JavaScript, utilizando un conjunto de tecnologías de vanguardia que ofrecen un rendimiento superior, mayor seguridad, escalabilidad, mantenibilidad y una mejor experiencia de desarrollo. Esta nueva arquitectura se compone de los siguientes elementos clave:

- **Next.js 15 y React:** Next.js es un framework de React que proporciona una serie de características y optimizaciones que mejoran significativamente el rendimiento y la experiencia de desarrollo. React, por su parte, es una biblioteca de JavaScript para construir interfaces de usuario interactivas y dinámicas. La combinación de ambos ofrece:
    - **Rendimiento Excepcional:** Next.js ofrece varias estrategias de renderizado que optimizan la entrega de contenido:
        - **Server-Side Rendering (SSR):** Las páginas se renderizan en el servidor en cada solicitud, lo que mejora significativamente el First Contentful Paint (FCP) y el SEO, ya que los motores de búsqueda reciben un HTML completo y listo para indexar.
        - **Static Site Generation (SSG):** Las páginas se generan en tiempo de compilación y se almacenan como archivos estáticos, lo que permite servirlas desde una CDN (Content Delivery Network) con un rendimiento excepcional.

Ideal para páginas con contenido que no cambia con frecuencia, como "Quiénes Somos" o "Contacto".
        - **Incremental Static Regeneration (ISR):** Permite regenerar las páginas estáticas en segundo plano a intervalos regulares, combinando las ventajas de SSG (alto rendimiento) con la capacidad de mostrar contenido actualizado dinámicamente.

    - **Optimización de Imágenes Nativa:** Next.js ofrece capacidades integradas para optimizar imágenes, incluyendo la compresión automática, el cambio de tamaño y la conversión a formatos modernos como WebP, lo que reduce el peso de las imágenes y mejora los tiempos de carga.
    - **Enrutamiento Optimizado y Mejor SEO:** Next.js proporciona un sistema de enrutamiento basado en archivos que simplifica la gestión de las rutas del sitio y mejora el SEO al generar URLs limpias y optimizadas. Además, facilita la gestión de metadatos SEO específicos para cada página.
    - **Fast Refresh:** Esta característica acelera el ciclo de desarrollo al permitir ver los cambios en el código de forma casi instantánea en el navegador, sin necesidad de recargar la página completa.

- **Payload 3.0 (CMS Headless):** Payload es un CMS headless de código abierto que proporciona una API robusta y flexible para la gestión de contenido. Al ser headless, separa el backend (donde se gestiona el contenido) del frontend (donde se muestra el contenido), lo que ofrece una mayor flexibilidad y escalabilidad.
    - **API-First y Flexibilidad:** Payload expone una API GraphQL/REST que permite acceder al contenido desde cualquier plataforma o dispositivo. Esto facilita la integración con otras aplicaciones y la creación de experiencias personalizadas.
    - **Experiencia de Edición Intuitiva:** Payload ofrece una interfaz de usuario intuitiva para la gestión de contenido, lo que facilita la creación, edición y publicación de contenido por parte del equipo de marketing o editores.
    - **Tipado con TypeScript:** Payload está construido con TypeScript, lo que mejora la calidad del código, la detección temprana de errores y la colaboración en equipo.

- **Tailwind CSS (Estilizado Eficiente):** Tailwind CSS es un framework CSS basado en utilidades que permite estilizar rápidamente las interfaces de usuario utilizando clases predefinidas.
    - **Desarrollo Rápido y Consistente:** Tailwind CSS acelera el desarrollo al proporcionar un conjunto completo de utilidades que se pueden combinar para crear estilos personalizados sin necesidad de escribir CSS personalizado en la mayoría de los casos. Esto también garantiza la consistencia visual en todo el sitio.
    - **Fácil Mantenimiento:** Al utilizar clases predefinidas, el código CSS es más legible y fácil de mantener.

- **Shadcn/ui (Componentes Reutilizables):** Shadcn/ui es una colección de componentes de interfaz de usuario reutilizables construidos con React y Tailwind CSS.
    - **Aceleración del Desarrollo y Consistencia en la UI:** Shadcn/ui proporciona componentes listos para usar que aceleran el desarrollo del frontend y garantizan la coherencia en la interfaz de usuario.
    - **Personalización:** Los componentes de Shadcn/ui son altamente personalizables y se pueden adaptar fácilmente a las necesidades específicas del proyecto.
    - **Accesibilidad:** Los componentes de Shadcn/ui se construyen siguiendo las mejores prácticas de accesibilidad, lo que garantiza que el sitio web sea utilizable por personas con discapacidades.

- **TypeScript (Tipado Estático):** TypeScript añade tipado estático a JavaScript, lo que mejora la calidad del código, la detección temprana de errores y la colaboración en equipo.
    - **Detección temprana de errores:** TypeScript permite detectar errores en tiempo de desarrollo, antes de que lleguen a producción, lo que ahorra tiempo y reduce el riesgo de fallos.
    - **Mejor mantenibilidad:** El tipado estático facilita la comprensión y el mantenimiento del código, especialmente en proyectos grandes y complejos.
    - **Mejor colaboración:** TypeScript mejora la comunicación y la colaboración entre los desarrolladores al proporcionar una documentación implícita del código.

En conjunto, estas tecnologías ofrecen una solución integral, abordando los problemas de rendimiento, seguridad, mantenimiento, escalabilidad y experiencia de desarrollo. La nueva arquitectura permitirá crear un sitio web más rápido, seguro, escalable, fácil de mantener y con una mejor experiencia de usuario.

IV. Beneficios Detallados de la Migración

La migración a una arquitectura moderna basada en Next.js, Payload, Tailwind CSS, Shadcn/ui y TypeScript ofrece una serie de beneficios tangibles que impactarán positivamente en dorsia.es y clinicasorigen.es, abordando las limitaciones de la plataforma actual basada en WordPress. Estos beneficios se detallan a continuación:

1. **Rendimiento Superior:**

    a. **Tiempos de Carga Drásticamente Reducidos:** Gracias al renderizado híbrido de Next.js (SSR, SSG, ISR), se espera una reducción significativa en los tiempos de carga. El SSR mejora el FCP y el LCP al renderizar las páginas en el servidor, mientras que el SSG genera páginas estáticas que se sirven desde una CDN para una velocidad óptima. El ISR permite la actualización de contenido estático sin necesidad de recompilar todo el sitio. Esto se traduce en una experiencia de usuario mucho más fluida y rápida.

b. **Optimización Automática de Imágenes:** La optimización de imágenes nativa de Next.js reduce el peso de las imágenes sin sacrificar la calidad visual, lo que contribuye a una carga más rápida de las páginas.
c. **Mejora en las Métricas Core Web Vitals:** La combinación de estas optimizaciones impactará positivamente en las métricas Core Web Vitals (LCP, FID, CLS), mejorando el SEO y la experiencia del usuario. Por ejemplo, se espera que el LCP, que actualmente es un problema en ambas webs, se reduzca significativamente, cumpliendo con los estándares de Google.

2. **Escalabilidad y Modernidad:**

a. **Escalabilidad Horizontal:** La arquitectura propuesta permite escalar horizontalmente añadiendo más servidores para manejar picos de tráfico sin afectar el rendimiento. Esto es crucial para asegurar la disponibilidad del sitio durante campañas de marketing o momentos de alta demanda.
b. **Infraestructura Moderna:** El uso de tecnologías modernas como Node.js y la posibilidad de desplegar en plataformas como Vercel o Netlify facilita la gestión de la infraestructura y permite aprovechar las últimas innovaciones en el campo del desarrollo web.

3. **Flexibilidad de Desarrollo:**

a. **Desarrollo Ágil y Eficiente:** El uso de React y Next.js, junto con Tailwind CSS y Shadcn/ui, agiliza el desarrollo de nuevas funcionalidades y la creación de interfaces personalizadas.
b. **Personalización Avanzada:** React permite crear componentes reutilizables y altamente personalizables, lo que facilita la implementación de diseños complejos y la adaptación a las necesidades específicas de cada web (dorsia.es y clinicasorigen.es). Por ejemplo, se podrían crear componentes interactivos para mostrar información de tratamientos, ubicaciones de clínicas o testimonios de pacientes de una forma mucho más atractiva y funcional que con WordPress.
c. **Integraciones Simplificadas:** La API-First de Payload facilita la integración con otros sistemas y servicios, como CRMs, sistemas de gestión de citas o plataformas de marketing automation.

4. **Seguridad Mejorada:**

La migración a una arquitectura moderna con Next.js y un CMS headless como Payload ofrece mejoras sustanciales en la seguridad en comparación con la plataforma WordPress actual. Estas mejoras se centran en la reducción de la superficie de ataque, un mayor control sobre las prácticas de seguridad y la mitigación de vulnerabilidades comunes:

- **Reducción Drástica de la Superficie de Ataque:**

    - **Minimización de la Dependencia de Plugins:** WordPress, como se ha mencionado, depende en gran medida de plugins de terceros para extender su funcionalidad. Cada plugin representa un posible punto de entrada para atacantes si contiene vulnerabilidades. La nueva arquitectura, al utilizar un CMS headless y un conjunto de herramientas modernas y bien mantenidas, reduce drásticamente la necesidad de plugins de terceros, minimizando así la superficie de ataque.
    - **Desacoplamiento Frontend-Backend (CMS Headless):** Al separar el frontend (Next.js) del backend (Payload), se aísla la lógica de presentación del contenido de la lógica de gestión de datos. Esto dificulta que los atacantes exploten vulnerabilidades en el frontend para acceder a la base de datos o al servidor.
    -
- **Mayor Control sobre las Prácticas de Seguridad:**

    - **Código Base Más Controlado:** Al tener un control total sobre el código del frontend con Next.js y un CMS headless con un código base más reducido y auditado, se facilita la implementación de prácticas de seguridad robustas y la revisión del código en busca de vulnerabilidades.
    - **Actualizaciones Más Sencillas y Seguras:** Las actualizaciones de Next.js, React, Payload y otras dependencias se gestionan de forma centralizada y son generalmente más sencillas y rápidas de aplicar que las actualizaciones de WordPress, plugins y temas. Esto reduce el tiempo que el sitio web permanece vulnerable a exploits conocidos.
    -
    - **Implementación de Medidas de Seguridad Modernas:** La nueva arquitectura facilita la implementación de medidas de seguridad modernas, como:
        - **Content Security Policy (CSP):** Permite definir una política de seguridad que controla los recursos que se pueden cargar en la página, mitigando ataques XSS.
        - **HTTP Strict Transport Security (HSTS):** Fuerza el uso de conexiones HTTPS, protegiendo contra ataques de "man-in-the-middle".
        - **Protección contra ataques de fuerza bruta:** Se pueden implementar medidas más robustas para prevenir ataques de fuerza bruta al panel de administración, como el bloqueo de IPs tras un número determinado de intentos fallidos o la autenticación de dos factores.
        -
- **Mitigación de Vulnerabilidades Comunes en WordPress:**

    - **Menor Riesgo de Inyección SQL:** Al utilizar una API bien definida y ORMs (Object-Relational Mappers) con Payload, se reduce significativamente el riesgo de inyección SQL, una de las vulnerabilidades más comunes en WordPress.
    - **Mitigación de XSS:** El uso de React y las buenas prácticas de desarrollo frontend ayudan a prevenir ataques XSS, ya que React escapa automáticamente los datos renderizados, evitando la ejecución de scripts maliciosos.
- **Seguridad por diseño:** Al construir la nueva arquitectura desde cero, se pueden incorporar consideraciones de seguridad desde las primeras etapas del desarrollo, siguiendo principios de "seguridad por diseño" (security by design).

En resumen, la migración a una arquitectura moderna no solo mejora el rendimiento y la experiencia del usuario, sino que también representa una mejora significativa en la seguridad. La reducción de la superficie de ataque, el mayor control sobre las prácticas de seguridad y la mitigación de vulnerabilidades comunes en WordPress hacen que la nueva plataforma sea mucho más robusta y segura.

5. **SEO y Accesibilidad:**

    a. **Optimización SEO Nativa:** Next.js ofrece características integradas para mejorar el SEO, como el renderizado del lado del servidor (SSR), la generación de sitemaps y la gestión de metadatos.
    b. **Mejor Accesibilidad:** El uso de React y Tailwind CSS, junto con los componentes de Shadcn/ui, facilita el cumplimiento de las pautas de accesibilidad web (WCAG), lo que hace que el sitio sea más accesible para personas con discapacidades.

6. **Mantenimiento Simplificado:**

    a. **Menor Dependencia de Plugins:** Al utilizar un CMS headless y un conjunto de herramientas modernas, se reduce significativamente la dependencia de plugins de terceros, lo que simplifica el mantenimiento y reduce el riesgo de conflictos e incompatibilidades.
    b. **Actualizaciones Más Sencillas:** Las actualizaciones de las tecnologías utilizadas en la nueva arquitectura son más sencillas y menos propensas a generar problemas en comparación con las actualizaciones de WordPress, plugins y temas.

7. **Experiencia del Usuario (UX) Mejorada:**

    a. **Navegación Fluida e Intuitiva:** Los tiempos de carga más rápidos y las interfaces más interactivas proporcionadas por React y Next.js mejoran significativamente la experiencia del usuario, haciendo que la navegación sea más fluida e intuitiva.

b. **Interfaces Modernas y Atractivas:** El uso de Tailwind CSS y Shadcn/ui permite crear interfaces modernas y atractivas que mejoran la imagen de marca de dorsia.es y clinicasorigen.es.

8. **Ecosistema y Futuro:**
    a. **Ecosistema JavaScript Maduro y en Constante Evolución:** El ecosistema de JavaScript es uno de los más grandes y activos del mundo, lo que garantiza el acceso a una gran cantidad de recursos, herramientas y talento.
    b. **Tecnologías de Vanguardia:** La nueva arquitectura se basa en tecnologías de vanguardia que se actualizan constantemente, lo que asegura la longevidad y la adaptabilidad de la plataforma a las futuras necesidades del negocio.

V. Beneficios Adicionales y Mejores Prácticas

Además de los beneficios principales de la migración, la implementación de las siguientes mejores prácticas contribuirá a una mayor calidad, eficiencia y escalabilidad del proyecto:

- **Testing (Pruebas Automatizadas):**

La implementación de una estrategia de testing integral es fundamental para asegurar la calidad del código, prevenir errores y garantizar la estabilidad de la aplicación. Se implementarán los siguientes tipos de pruebas:

- **Pruebas Unitarias:** Prueban unidades individuales de código (funciones, componentes) de forma aislada, verificando su correcto funcionamiento. Esto ayuda a detectar errores en las etapas tempranas del desarrollo.
- **Pruebas de Integración:** Prueban la interacción entre diferentes módulos o componentes, asegurando que funcionen correctamente en conjunto.
- **Pruebas de Interfaz de Usuario (UI):** Verifican que la interfaz de usuario se renderiza correctamente y funciona según lo esperado en diferentes navegadores y dispositivos.
- **Pruebas de Accesibilidad:** Aseguran que el sitio web cumple con las pautas de accesibilidad WCAG, lo que lo hace utilizable para personas con discapacidades.
- **Pruebas End-to-End (E2E):** Simulan el flujo completo del usuario, desde que accede al sitio web hasta que completa una acción específica (ej., solicitar una cita). Estas pruebas verifican que todas las partes del sistema funcionan correctamente en conjunto y proporcionan una mayor confianza en la estabilidad de la aplicación.
- **Pruebas de Regresión:** Se ejecutan después de cada cambio en el código para asegurar que las funcionalidades existentes no se hayan visto afectadas.

La automatización de estas pruebas, mediante el uso de frameworks como Jest, React Testing Library, Cypress o Playwright, agilizará el proceso de desarrollo y garantizará la consistencia de las pruebas.

- **Design System (Sistema de Diseño):**

Un sistema de diseño es un conjunto de componentes reutilizables, guías de estilo y patrones de diseño que definen la identidad visual y la experiencia de usuario de un producto. La implementación de un design system ofrece los siguientes beneficios:

    - **Consistencia Visual:** Asegura una apariencia coherente en todo el sitio web, mejorando la imagen de marca y la experiencia del usuario.
    - **Eficiencia en el Desarrollo:** Facilita la creación de nuevas interfaces al proporcionar componentes predefinidos y reutilizables.
    - **Mantenimiento Simplificado:** Facilita la actualización y el mantenimiento del diseño, ya que los cambios se realizan en un solo lugar y se propagan automáticamente a todos los componentes.
    - **Mejora la Colaboración:** Facilita la comunicación entre diseñadores y desarrolladores al proporcionar un lenguaje común y una documentación clara.
- **Optimización para Móviles (Mobile-First):**

Dado el creciente uso de dispositivos móviles para acceder a internet, la estrategia "Mobile-First" es crucial. Esto implica diseñar y desarrollar el sitio web priorizando la experiencia en dispositivos móviles y luego adaptándolo a pantallas más grandes. Los beneficios de esta estrategia son:

    - **Mejor Experiencia de Usuario en Móviles:** Se asegura una experiencia óptima para la mayoría de los usuarios.
    - **Mejora el SEO:** Google indexa principalmente la versión móvil de los sitios web, por lo que una buena experiencia móvil es fundamental para el SEO.
    - **Mejora el Rendimiento:** Al optimizar primero para móviles, se reduce el peso de la página y se mejoran los tiempos de carga en todos los dispositivos.
- **PostHog (o alternativas como Mixpanel, Amplitude, Google Analytics con seguimiento de eventos):**

Implementar una herramienta de análisis de productos como PostHog (o una alternativa) proporciona información valiosa sobre el comportamiento del usuario:

    - **Análisis del Comportamiento del Usuario:** Permite entender cómo interactúan los usuarios con el sitio web, identificando patrones de uso, puntos de fricción y áreas de mejora.
    - **Medición del Impacto de la Migración:** Permite comparar las métricas de rendimiento y comportamiento antes y después de la migración, validando el éxito del proyecto.
    - **Optimización Continua:** Facilita la toma de decisiones basadas en datos para optimizar la experiencia del usuario y mejorar las conversiones.
    - **Seguimiento de Eventos:** Permite rastrear acciones específicas del usuario (clics, envíos de formularios, etc.), lo que proporciona una visión más granular del comportamiento.

- **Integración Continua/Entrega Continua (CI/CD):**

La implementación de un flujo de CI/CD automatiza el proceso de construcción, prueba y despliegue de la aplicación:

    - **Automatización del Proceso de Desarrollo:** Automatiza tareas repetitivas, como la compilación del código, la ejecución de pruebas y el despliegue en diferentes entornos.
    - **Entrega Continua de Valor:** Permite realizar despliegues frecuentes y rápidos, lo que agiliza la entrega de nuevas funcionalidades y correcciones de errores.
    - **Reducción de Errores en Producción:** La automatización de las pruebas y el despliegue reduce el riesgo de errores humanos y asegura la calidad del código en producción.
    - **Mayor Eficiencia del Equipo de Desarrollo:** Libera al equipo de tareas manuales, permitiéndoles enfocarse en el desarrollo de nuevas funcionalidades.

- **Mejor integración con Hal**
    - Facilidad de integrar formularios
    - Posibilidad de integrar nuevas funcionalidades para mejorar la experiencia del usuario

La implementación de estas mejores prácticas, junto con la migración a la nueva arquitectura, sentará las bases para una plataforma web robusta, escalable, segura y con una excelente experiencia de usuario.

VII. Conclusión

La presente justificación técnica ha analizado en detalle las limitaciones de la actual plataforma de dorsia.es y clinicasorigen.es, ambas construidas con WordPress. Los datos de rendimiento recopilados, junto con las vulnerabilidades inherentes a la arquitectura de WordPress y su ecosistema de plugins, la complejidad del mantenimiento, las limitaciones de escalabilidad y la experiencia de desarrollo obsoleta, demuestran la necesidad crítica de una modernización.

Como se ha demostrado, ambas webs sufren problemas de rendimiento que afectan negativamente la experiencia del usuario, el SEO y, en última instancia, las conversiones. Los altos valores de LCP, FCP y TTFB evidencian la necesidad de una optimización profunda que la arquitectura actual dificulta. Además, la dependencia de plugins de terceros incrementa la superficie de ataque y complica la gestión de la seguridad, exponiendo a ambas webs a riesgos innecesarios.

La solución propuesta, una arquitectura moderna basada en Next.js, React, Payload, Tailwind CSS, Shadcn/ui y TypeScript, ofrece una respuesta integral a estos desafíos. Esta nueva arquitectura no solo aborda los problemas de rendimiento mediante la optimización de la entrega de contenido y la reducción del peso de las páginas, sino que también mejora significativamente la seguridad al minimizar la dependencia de plugins y proporcionar un mayor control sobre las prácticas de seguridad.

Los beneficios de la migración son múltiples y tangibles:

- **Rendimiento Superior:** Tiempos de carga drásticamente reducidos, mejorando la experiencia del usuario y el SEO.
- **Escalabilidad y Modernidad:** Mayor capacidad para manejar picos de tráfico y una infraestructura adaptada a las últimas tecnologías.
- **Flexibilidad de Desarrollo:** Mayor agilidad para implementar nuevas funcionalidades y personalizaciones.
- **Seguridad Mejorada:** Menor superficie de ataque y mayor control sobre las prácticas de seguridad.
- **SEO y Accesibilidad:** Optimización para motores de búsqueda y cumplimiento de estándares de accesibilidad.
- **Mantenimiento Simplificado:** Menor dependencia de plugins y actualizaciones más sencillas.
- **Experiencia del Usuario Mejorada:** Navegación más fluida e interfaces más modernas.
- **Ecosistema y Futuro:** Uso de tecnologías de vanguardia con un amplio ecosistema y soporte a largo plazo.

Además, la implementación de mejores prácticas como testing automatizado, un design system, la optimización para móviles, el uso de herramientas de análisis como PostHog y la implementación de CI/CD fortalecerán aún más la plataforma y el proceso de desarrollo.

En conclusión, la migración a una arquitectura moderna es una inversión estratégica que permitirá a dorsia.es y clinicasorigen.es superar las limitaciones de la plataforma actual, ofrecer una experiencia de usuario excepcional, mejorar el posicionamiento en buscadores, fortalecer la seguridad y agilizar el desarrollo de nuevas funcionalidades. Esta migración no solo resolverá los problemas existentes, sino que también sentará las bases para el crecimiento y la innovación a largo plazo.

Referencias:

https://payload-website.damenor.dev

https://payloadcms.com/compare/wordpress

https://nextjs.org

https://ui.shadcn.com

https://posthog.com

Información Adicional sobre el Ecosistema WordPress:

Es importante mencionar que el ecosistema de WordPress se encuentra actualmente en un período de incertidumbre debido a disputas legales entre Automattic (la empresa detrás de WordPress.com y contribuida principal a WordPress.org) y WP Engine. Estas disputas, que incluyen demandas y acusaciones mutuas, han generado preocupación en la comunidad y podrían tener implicaciones a largo plazo para el futuro del ecosistema WordPress. Si bien no hay indicios de un cierre inminente de WordPress.org, estas tensiones subrayan la importancia de considerar la estabilidad y la dirección a largo plazo de la plataforma al tomar decisiones tecnológicas. Esta situación refuerza aún más la justificación de migrar a una arquitectura más moderna y con un mayor control sobre el stack tecnológico.

https://www.searchenginejournal.com/mullenweg-says-lawsuits-could-end-wordpress/537336

https://www.youtube.com/watch?v=mttNWSRU30A&ab_channel=midulive