Forma

Descripción generada automáticamente con confianza baja

FACULTAD DE INGENIERÍA

Carrera de Ingeniería de Sistemas Computacionales

INNOVACIÓN TECNOLÓGICA EN EL DESARROLLO WEB CLINICO: MIGRACION DE WORDPRESS A UNA SOLUCION PERSONALIZADA CON REACT Y NEXT.JS

Trabajo de suficiencia profesional para optar al título profesional de:

**Haga clic o pulse aquí para escribir texto.**

**Autor:**

Marcos Daniel Romero Garcia

Asesor:

Mg. Fernando Alex Sierra Liñan

ORCID: 0000-0002-0687-3377

Lima - Perú

2025

# Informe de Similitud

(Copie y pegue como imagen la hoja del reporte global)

# Dedicatoria

El presente trabajo va dedicado con un gran cariño a mi padres y hermana, quienes fueron mi apoyo a lo largo de todo este camino, siempre motivándome a escalar profesionalmente año tras año. Quiero hacer una mención especial a mi madre, quien ha sido la persona que más ha valorado y fomentado mi desarrollo como profesional. Y un especial agradecimiento a mi asesor por guiarme en cada paso del desarrollo del trabajo de suficiencia profesional.

# Agradecimiento

En primer lugar, quiero agradecer a mi madre Yesy Garcia, que fue la persona que más ha fomentado en mis las ganas de seguir adelante.

A mi padre, Clever Romero, por enseñarme los valores que me inculcó desde pequeño, los cuales han sido fundamentales en los momentos más difíciles de mi vida.

Y, por último, pero no menos importante a mi hermana Daniela Romero, quien ha sido mi mayor apoyo en las buenas y en las malas a lo largo de mi vida.

**Tabla de contenidos**

[Informe de Similitud 2](#_Toc202719763)

[Dedicatoria 3](#_Toc202719764)

[Agradecimiento 4](#_Toc202719765)

[Índice de tablas 7](#_Toc202719766)

[Índice de Figuras 8](#_Toc202719767)

[Índice de ecuaciones 9](#_Toc202719768)

[RESUMEN EJECUTIVO 10](#_Toc202719769)

[CAPÍTULO I. INTRODUCCIÓN 11](#_Toc202719770)

[1.1. Contexto 11](#_Toc202719771)

[1.2. Datos principales de la empresa 12](#_Toc202719772)

[1.3. Localización de la empresa 13](#_Toc202719773)

[1.4. Cultura organizacional 14](#_Toc202719774)

[1.4.1. Misión 14](#_Toc202719775)

[1.4.2. Visión 14](#_Toc202719776)

[1.5. Diagrama de estructura organizacional 15](#_Toc202719777)

[CAPÍTULO II. MARCO TEÓRICO 17](#_Toc202719778)

[CAPÍTULO III. DESCRIPCIÓN DE LA EXPERIENCIA 18](#_Toc202719779)

[CAPÍTULO IV. RESULTADOS 19](#_Toc202719780)

[CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES 21](#_Toc202719781)

[REFERENCIAS 22](#_Toc202719782)

[ANEXOS 23](#_Toc202719783)

# Índice de tablas

# Índice de Figuras

[Figura 1. Reporte de la ficha RUC 11](#_Toc202719742)

[Figura 2. Mapa relieve de localización de la empresa 12](#_Toc202719743)

[Figura 3. Mapa satelital de localización de la empresa. 13](#_Toc202719744)

[Figura 4. Organigrama principal de gerencia 14](#_Toc202719745)

[Figura 5. Organigrama del área de investigación 14](#_Toc202719746)

[Figura 6. Organigrama del área de gestión y operaciones 1 14](#_Toc202719747)

[Figura 7. Organigrama del área de gestión y operaciones 2 15](#_Toc202719748)

[Figura 8. Organigrama del área de tecnología y transformación digital 15](#_Toc202719749)

[Figura 9. Organigrama del área de marketing y comercial 16](#_Toc202719750)

# Índice de ecuaciones

(El presente índice se fijará en función a la naturaleza del trabajo. Las ecuaciones se emplean habitualmente en investigaciones en ingeniería)

# RESUMEN EJECUTIVO

Explica en 200 palabras el entorno en el cual se desarrolló la experiencia profesional precisando el proyecto o problema laboral afrontado, herramientas o modelos utilizados para el desarrollo de la solución, los resultados y las conclusiones, así como las competencias profesionales aplicadas.

# CAPÍTULO I. INTRODUCCIÓN

## Contexto

Este trabajo presenta la experiencia adquirida en la empresa Insalud Corp E.I.R.L., en la práctica de la innovación tecnológica, en específico, migración de la aplicación web clínica de Wordpress a una solución a medida construida con React.js y Next.js.

La necesidad de esta propuesta surgió a raíz de las numerosas limitaciones del anterior sistema, las cuales incluían, bajo rendimiento, mala configuración, escasa escalabilidad, dependencias de plugins y una notable falta de personalización para poder integrar funcionalidades avanzadas que vayan alineadas con los nuevos objetivos de la organización. Como desarrollador web del área, fui testigo de estas deficiencias y debido a ello es que propuse un rediseño integral basado en tecnologías mas modernas y escalables ajustadas a las necesidades futuras de la empresa. Esto permitió mejorar el rendimiento, seguridad, la comunicación con el Backend y lo más importante, la experiencia del usuario.

La nueva plataforma fue diseñada e implementada utilizando herramientas como React, Next.js, GitHub, Figma, y servicios como Amazon S3 y Vercel. Actualmente, ya se encuentra en etapa de producción y sigue siendo escalada con nuevas funcionalidades y secciones. Esta iniciativa ha contribuido a fortalecer la infraestructura digital de la organización, permitiendo una administración clínica más eficiente, segura, personalizable y adaptable.

## Historia de la empresa

INSALUD es una institución de salud privada. Se fundó en el año 2020 en Lima, Perú. Ofrece servicios médicos en urología, ginecología, salud sexual y medicina regenerativa.

Desde su comienzo, la empresa ha puesto su atención en la tecnología y la prevención. Estos son pilares para dar una atención médica de calidad que se adapta a lo que los pacientes piden. Su modo de atención no solo trata enfermedades, sino que también mejora la vida de las personas. Utiliza procedimientos modernos, seguros y poco invasivos.

INSALUD ha crecido de forma constante. La gente confía en la institución. Por esto, ha podido abrir centros en otros países como Ecuador y Panamá. Su plan de negocio busca llegar a ser un modelo en salud sexual y regenerativa en América Latina. Esto se consigue al mejorar sus servicios sin parar. También, desarrolla proyectos de investigación y usa nuevas formas de trabajo.

## Datos principales de la empresa

Los datos de Insalud Corp E.I.R.L según el registro de la Superintendencia Nacional de Administración Tributaria (SUNAT) son los siguientes:

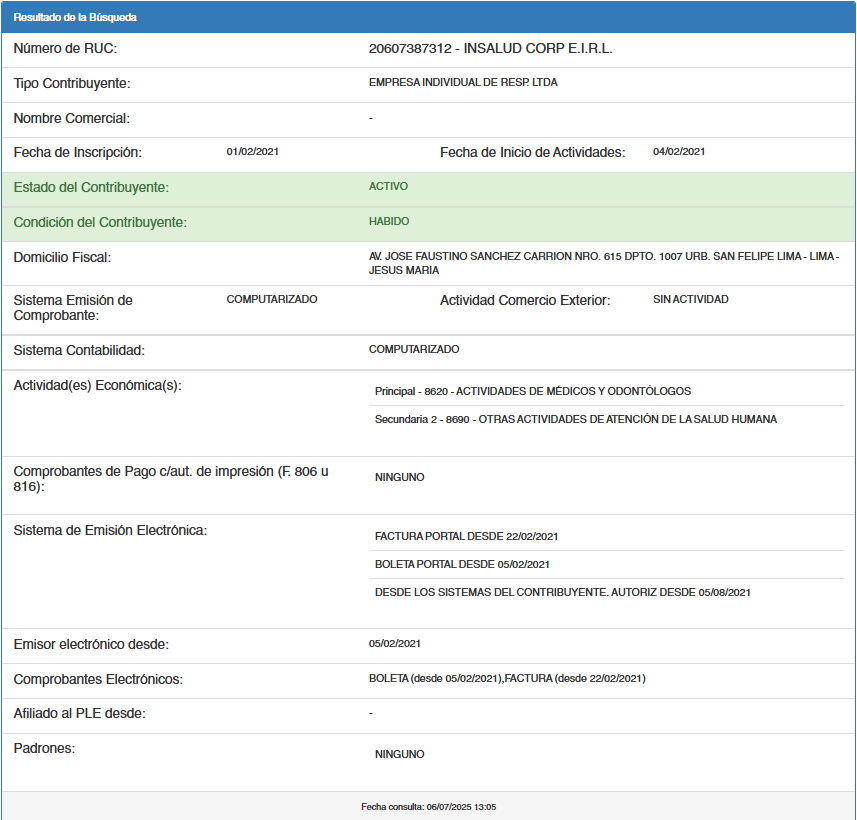


Figura 1. Reporte de la ficha RUC

*Fuente: Sunat*

## Localización de la empresa

La empresa se localiza en Av. Manuel Olguín 335, Santiago de Surco 15023, en el distrito de Santiago de surco, en la provincia metropolitana de Lima, Lima, Perú. De referencia a una cuadra de la universidad de Lima.

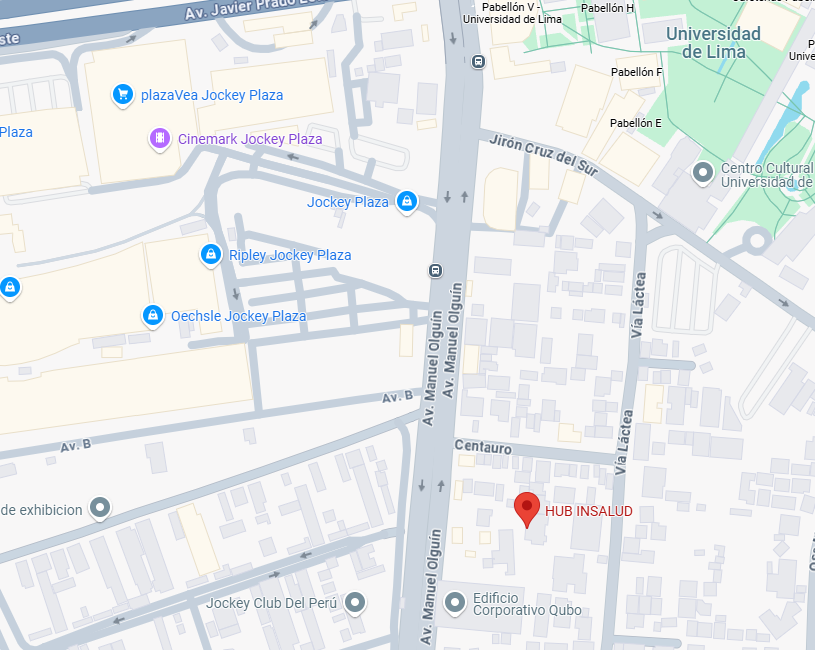


Figura 2. Mapa relieve de localización de la empresa

Fuente: Google Maps 2025

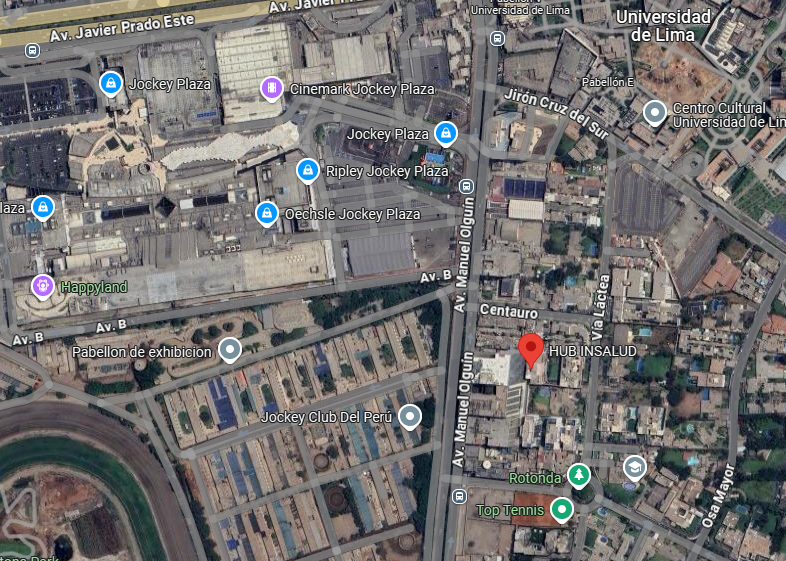


Figura 3. Mapa satelital de localización de la empresa.

Fuente: Google Maps2025

## Cultura organizacional

Insalud es una empresa dedicada a brindar atención integral en salud sexual y reproductiva, INSALUD construye su cultura organizacional sobre pilares como la empatía, la prevención, la calidad médica y el acceso informado. Se fomenta un entorno de respeto, profesionalismo e innovación constante, donde cada integrante del equipo comparte el compromiso de educar, prevenir y acompañar. Esta cultura impulsa su propósito de transformar la salud íntima en América Latina, con una visión ética, humana y cercana.

### Misión

Reducir el impacto del envejecimiento en la calidad de vida de nuestros pacientes con terapias no invasivas.

### Visión

Ser el líder regional en medicina regenerativa y antiaging, con operaciones en la mayoría de los países de LATAM para el 2027.

## Diagrama de estructura organizacional

La empresa está organizada en diferentes áreas, con el Directorio y la Gerencia General ubicados en la sede principal, junto a los departamentos de Marketing y Comercial, Administración, así como Tecnología y Cambio Digital.

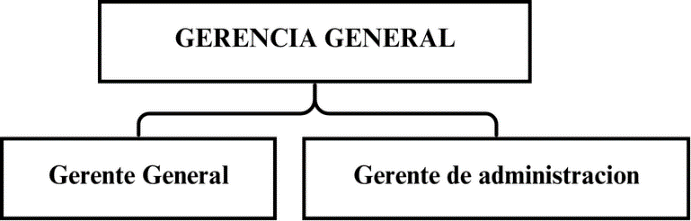


Figura 4. Organigrama principal de gerencia

*Fuente: Insalud*

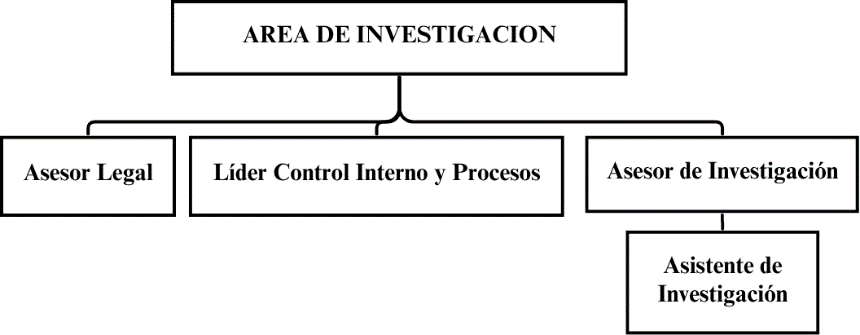


Figura 5. Organigrama del área de investigación

*Fuente: Insalud*

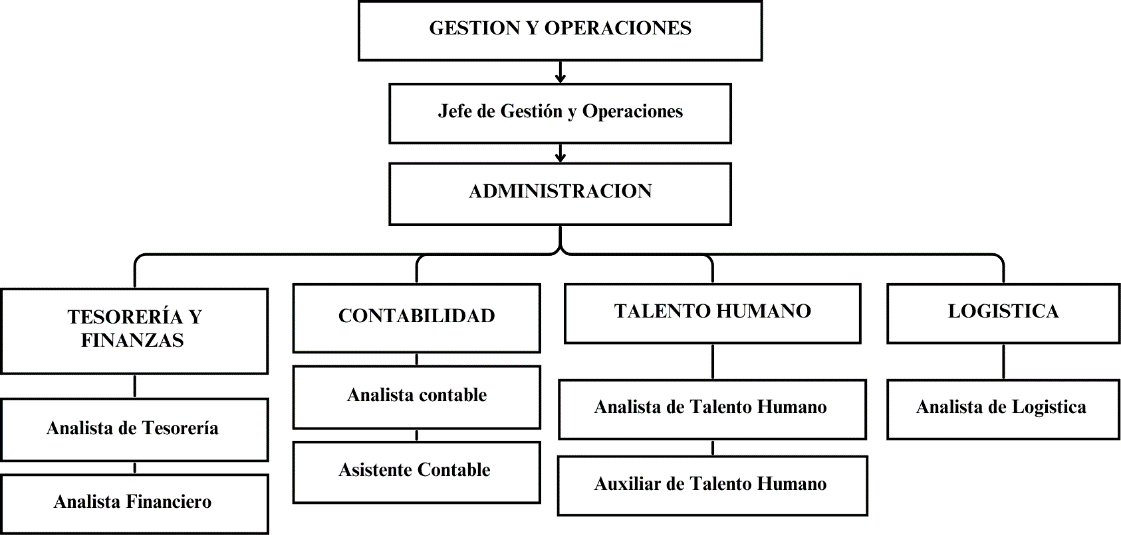


Figura 6. Organigrama del área de gestión y operaciones 1

*Fuente: Insalud*

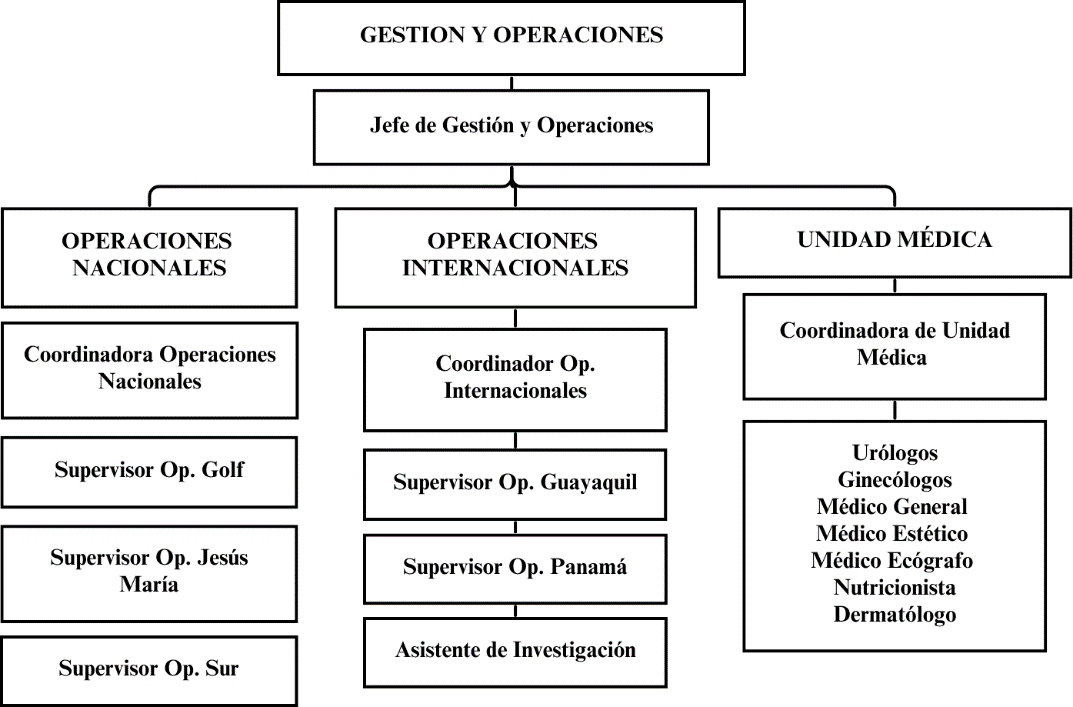


Figura 7. Organigrama del área de gestión y operaciones 2

*Fuente: Insalud*

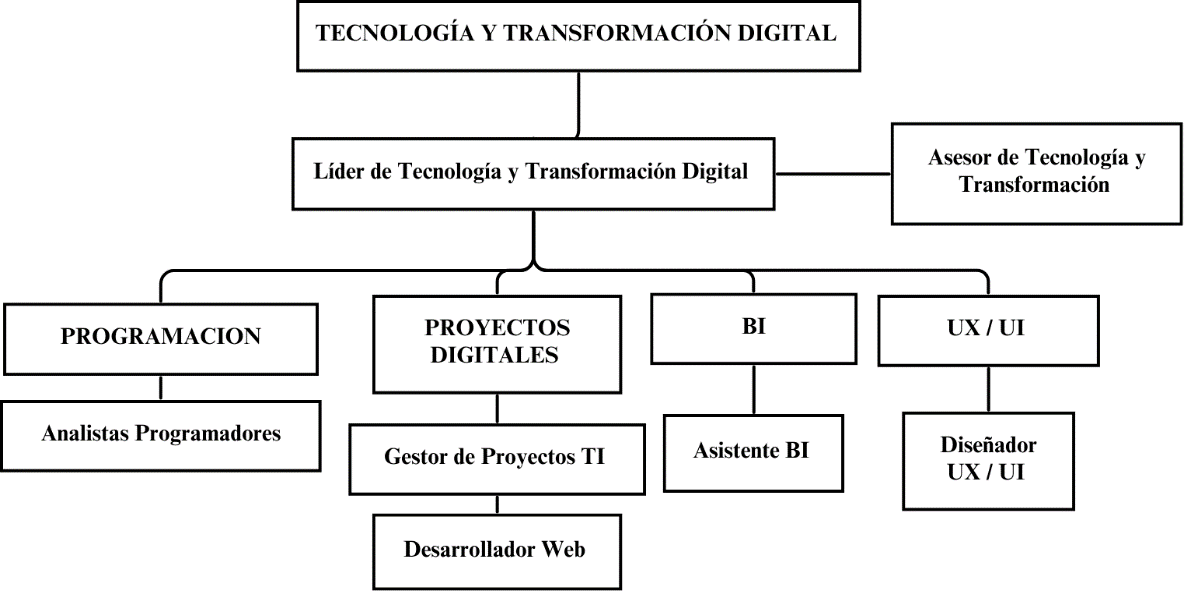


Figura 8. Organigrama del área de tecnología y transformación digital

*Fuente Insalud*

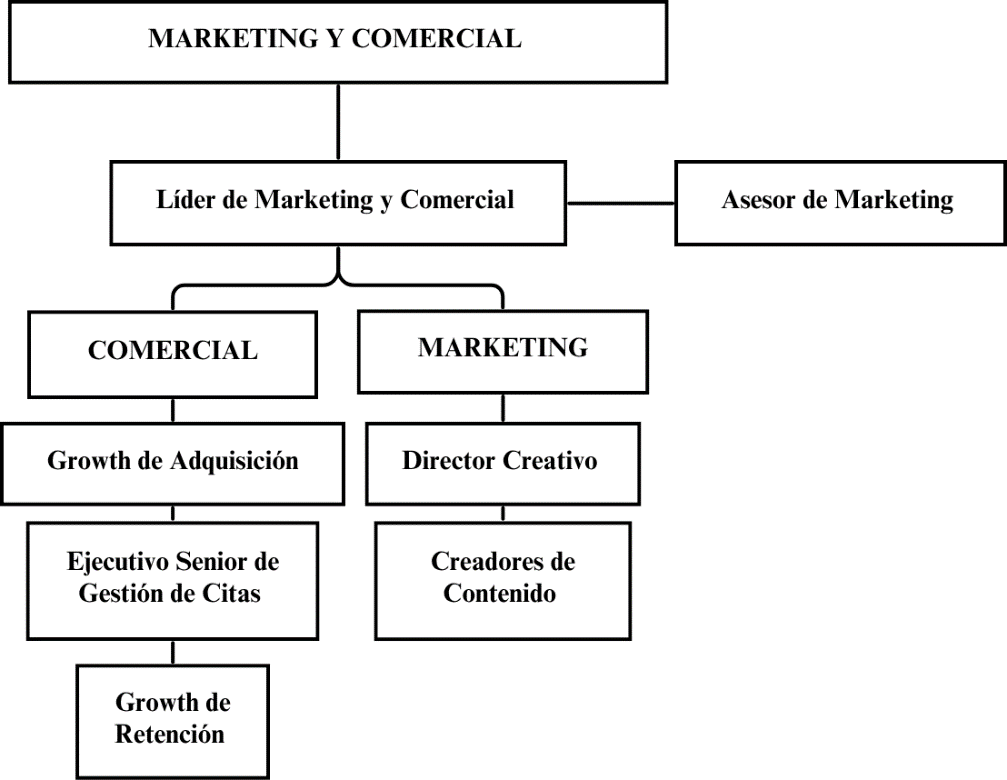


Figura 9. Organigrama del área de marketing y comercial

*Fuente: Insalud*

# CAPÍTULO II. MARCO TEÓRICO

## 2.1. Modernización de Sistemas Web

### 2.1.1. Evolución de las Tecnologías Web

El progreso en las tecnologías de la web ha experimentado varios cambios desde el establecimiento de la World Wide Web en 1990. Según Berners-Lee et al. (2001), se fundó la web moderna, y se proporcionaron los principios clave, que han definido todas las direcciones futuras en el desarrollo de aplicaciones web desde hace treinta años. El desarrollo desde los sitios web estáticos a las aplicaciones web dinámicas es un ejemplo de uno de los cambios más cruciales en la historia de la tecnología.

Según Garrett (2005), la evolución web se puede subdividir en tres generaciones principales, a saber: web 1.0, que se define principalmente por tener contenido estático; Web 2.0 que adoptó la interactividad y la colaboración del usuario; y web 3.0 que adoptó tecnologías semánticas y experiencias más personalizadas. Esta lenta pero audaz evolución de la web ha ocasionado la necesidad de migrar desde las soluciones existentes, configuraciones y arquitecturas heredadas hasta estructuras más modernas y escalables.

La estructura web convencional, que se basa en el modelo cliente-servidor básico, ha demostrado limitaciones considerables cuando se aplica a aplicaciones empresariales complejas. Fielding (2000) introdujo el estilo arquitectónico REST . REST definía los principios clave para crear aplicaciones web escalables y mantenibles. Estos principios influyen en estructuras modernas como React y Next.js, que dividen claramente la lógica del frontend y el backend y permiten relacionarse con bases de datos relacionales como MySQL utilizando API.

### 2.1.2 Limitaciones de los Sistemas de Gestión de Contenidos Tradicionales

Los sistemas de gestión de contenidos (CMS) como WordPress, desarrollados tienen varias limitaciones considerables cuando se implementan en aplicaciones complejas, por lo tanto la escalabilidad no es su fuerte. Morville y Rosenfeld (2006) Se reconoció que estas restricciones provienen de la estructura monolítica propia de estos sistemas, la cual agrupa la lógica de presentación, la administración de contenidos y las funciones de la aplicación en una única estructura unificada.

La dependencia de plugins y extensiones de terceros es una vulnerabilidad crítica en los sistemas CMS. Según OWASP (2021), aproximadamente el 52% de las fallas de seguridad en WordPress se originan en complementos que están mal programados o que no han sido actualizados.

Esta dependencia no solo pone en peligro la seguridad de las aplicaciones web, sino que también genera problemas en cuanto a compatibilidad, rendimiento y escalabilidad, inconvenientes que se incrementan y se agravan a medida que la aplicación se expande. Por esta razón, las estructuras modernas basadas en React incorporan una validación de datos segura a través de bibliotecas específicas como Zod, la cual ofrece verificación de tipos durante la ejecución y validación de esquemas y tipos, eliminando la necesidad de complementos externos para funciones esenciales.

El funcionamiento de los sistemas de gestión de contenido tradicionales sufre un impacto significativo debido a su estructura de renderizado del lado del servidor. Cada vez que un usuario realiza una solicitud, es necesario procesar la página completa, lo cual incluye consultas a la base de datos, renderización de plantillas y creación de HTML. Este enfoque genera tiempos de carga largos y experiencias de usuario insatisfactorias, particularmente en aplicaciones que requieren interactividad en tiempo real.

El diseño de aplicaciones web actuales necesita enfoques de gestión de proyectos que permitan una flexible adaptación y una entrega progresiva de características. Según Schwaber y Sutherland (2020), la metodología Scrum ofrece un esquema ágil que resulta particularmente útil para proyectos de desarrollo tecnológico en los que los requerimientos pueden cambiar a lo largo del proceso. Implementar Scrum en proyectos de migración tecnológica posibilita la entrega recurrente de elementos funcionales, lo que ayuda a realizar validaciones tempranas de las decisiones arquitectónicas y a ajustarse a las nuevas exigencias del negocio.

**2.1.3 Principios de la Arquitectura Web Moderna**

La arquitectura web moderna se fundamenta en principios de separación de responsabilidades, modularidad y escalabilidad. Hunt y Thomas (2019) establecieron que las aplicaciones web contemporáneas deben adoptar patrones arquitectónicos que permitan el desarrollo independiente de componentes frontend y backend, facilitando la mantenibilidad y la evolución tecnológica.

El paradigma de desarrollo basado en componentes constituye un principio fundamental de la arquitectura web moderna. Gamma et al. (1994) introdujeron patrones de diseño que han influenciado significativamente el desarrollo de frameworks modernos como React. Este enfoque permite la creación de interfaces de usuario modulares, reutilizables y mantenibles, donde cada componente encapsula su lógica específica y puede ser desarrollado, probado y desplegado de manera independiente.

La arquitectura de aplicaciones de página única (Single Page Applications - SPA) representa una evolución significativa en la experiencia del usuario web. Fink y Flatow (2013) demostraron que las SPA proporcionan experiencias más fluidas y responsivas al eliminar las recargas completas de página, manteniendo el estado de la aplicación en el cliente y comunicándose con el servidor únicamente para intercambiar datos específicos.

El renderizado híbrido, que combina el renderizado del lado del servidor (SSR) y del lado del cliente (CSR), ofrece beneficios optimizados tanto para el rendimiento inicial como para la experiencia del usuario posterior. Vercel (2023) ha demostrado que esta aproximación mejora significativamente los tiempos de carga inicial y la optimización para motores de búsqueda, mientras mantiene la interactividad dinámica característica de las aplicaciones web modernas.

La implementación de APIs REST y GraphQL como capa de comunicación entre frontend y backend permite la creación de sistemas desacoplados y escalables. Facebook (2015) introdujo GraphQL como una alternativa más eficiente para el intercambio de datos, permitiendo a las aplicaciones cliente solicitar únicamente la información específica requerida, reduciendo el tráfico de red y mejorando el rendimiento general del sistema.

**2.1.4 Beneficios de la Migración Tecnológica**

La migración desde sistemas CMS tradicionales hacia arquitecturas web modernas produce beneficios cuantificables en múltiples dimensiones operativas. Google (2020) documentó que las aplicaciones web desarrolladas con frameworks modernos experimentan mejoras promedio del 40% en los tiempos de carga inicial y del 60% en la responsividad de la interfaz de usuario.

El rendimiento mejorado resulta de la optimización del renderizado y la gestión eficiente del estado de la aplicación. Las técnicas de code splitting y lazy loading, implementadas nativamente en frameworks modernos, permiten cargar únicamente los recursos necesarios para cada vista específica, reduciendo significativamente el tiempo de carga inicial y el consumo de ancho de banda.

La mantenibilidad del código se ve significativamente mejorada a través de la modularización y la separación de responsabilidades. La arquitectura basada en componentes facilita la identificación y resolución de errores, reduce la duplicación de código y permite la implementación de pruebas unitarias más efectivas. Estos factores contribuyen a una reducción estimada del 30% en los costos de mantenimiento a largo plazo.

La escalabilidad horizontal se facilita mediante la separación del frontend y backend, permitiendo el escalado independiente de cada capa según las demandas específicas del sistema. Esta flexibilidad arquitectónica resulta especialmente ventajosa en entornos empresariales donde los patrones de uso pueden variar significativamente entre diferentes módulos de la aplicación.

La experiencia del usuario se ve enriquecida por la implementación de interfaces más responsivas e intuitivas. La capacidad de actualizar selectivamente componentes específicos de la interfaz sin afectar el estado global de la aplicación proporciona una experiencia más fluida y comparable a las aplicaciones nativas de escritorio o móviles.

**✅ 2.1.5 Seguridad en Arquitecturas Web Modernas**

La seguridad es una preocupación central en el diseño de arquitecturas web modernas. A medida que las aplicaciones se descentralizan y se comunican mediante APIs, se incrementa el riesgo de ataques como inyecciones de código, ataques CSRF (Cross-Site Request Forgery), XSS (Cross-Site Scripting) y exposición de datos sensibles. Según OWASP (2021), el uso de validaciones estrictas, autenticación robusta y control de acceso granular son elementos clave para minimizar vulnerabilidades en aplicaciones modernas.

Frameworks como Next.js promueven buenas prácticas de seguridad integradas, como el uso de tokens de autenticación JWT, encabezados HTTP seguros y mecanismos para evitar renderizado malicioso (XSS). Además, bibliotecas como Zod, utilizadas para la validación de datos en el frontend, contribuyen a mitigar riesgos al asegurar que solo datos bien estructurados sean procesados. A nivel de red, el uso de HTTPS obligatorio, protección contra CORS mal configurado y el despliegue en plataformas seguras como Vercel fortalecen aún más la postura de seguridad de las aplicaciones web modernas (OWASP, 2021; Vercel, 2024).

2.2 Tecnologías de Desarrollo Web Moderno

La creación de aplicaciones web ha evolucionado hacia un enfoque más estructurado, modular y centrado en el rendimiento. En este contexto, tecnologías como Next.js, React, el uso de TypeScript, el despliegue en Vercel, y la adopción de metodologías ágiles como Scrum, representan los pilares fundamentales de los proyectos web modernos.

2.2.1 React y el desarrollo basado en componentes

React es una biblioteca de JavaScript creada por Meta (anteriormente Facebook), que permite construir interfaces de usuario a través de componentes reutilizables. Su modelo declarativo y su uso del DOM virtual permiten actualizar vistas de forma eficiente, facilitando el desarrollo de interfaces complejas de forma mantenible y escalable (Meta, 2023).

2.2.2 Next.js como framework integral

Next.js es un framework basado en React, desarrollado por Vercel, que ofrece un enfoque integral para construir aplicaciones web modernas. A diferencia de usar múltiples herramientas por separado, Next.js proporciona:

Renderizado híbrido (SSR y SSG),

Enrutamiento basado en archivos,

Optimización automática de imágenes y fuentes,

API Routes para lógica del servidor sin necesidad de infraestructura backend adicional,

Y despliegue directo y sin fricción en Vercel, su plataforma nativa (Vercel, 2024).

Esto simplifica la arquitectura y permite mantener todo el proyecto en un solo entorno, eliminando la necesidad de herramientas como Webpack, Vite o bases de datos externas si no se requieren.

2.2.3 TypeScript y la robustez del código

TypeScript extiende JavaScript añadiendo tipado estático opcional. Esto ayuda a prevenir errores comunes durante el desarrollo, mejora el autocompletado en editores de código y refuerza la robustez general del proyecto (Microsoft, 2024). En combinación con React y Next.js, TypeScript permite un desarrollo más predecible y mantenible.

2.2.4 Despliegue profesional con Vercel

Vercel es la plataforma oficial para desplegar aplicaciones Next.js. Ofrece integración directa con repositorios Git, CI/CD automatizado, actualizaciones instantáneas mediante "push", y manejo de dominios personalizados. Su modelo serverless elimina la necesidad de configurar servidores, reduciendo el tiempo de puesta en producción (Vercel, 2024).

2.2.5 Metodología Scrum

Scrum es una de las metodologías ágiles más utilizadas en el desarrollo profesional de software. Organiza el trabajo en ciclos llamados sprints, e involucra roles definidos (Scrum Master, Product Owner y equipo de desarrollo), así como reuniones clave: daily stand-ups, sprint planning, reviews y retrospectives. Esta estructura promueve la mejora continua, transparencia, y capacidad de adaptación a cambios en los requisitos (Schwaber & Sutherland, 2020).

**✅ 2.2.6 Control de Versiones con Git y GitHub**

El uso de sistemas de control de versiones es esencial en el desarrollo profesional de software. Git es una herramienta de control de versiones distribuido que permite llevar un historial detallado de los cambios realizados en el código fuente, facilitando la colaboración, el trabajo en ramas paralelas, la resolución de conflictos y la trazabilidad del desarrollo (Chacon & Straub, 2014). GitHub, por su parte, proporciona una plataforma basada en la nube donde los repositorios Git pueden ser alojados, gestionados y compartidos, integrando además herramientas de revisión de código, seguimiento de issues y automatización de flujos de trabajo.

En proyectos modernos, GitHub se combina con flujos de trabajo de integración continua y despliegue continuo (CI/CD), como los que proporciona Vercel, lo cual permite que cada cambio aprobado en el repositorio se despliegue automáticamente en el entorno productivo. Esta integración acelera los ciclos de entrega y minimiza errores humanos en el proceso de publicación.

2.2.6 Justificación de las tecnologías utilizadas

La combinación de Next.js, React, TypeScript, despliegue en Vercel y gestión del proyecto con Scrum, representa una selección alineada con las buenas prácticas actuales del desarrollo web profesional. Permite entregar productos rápidos, optimizados, escalables y adaptados a las necesidades del usuario final, sin incurrir en configuraciones complejas ni depender de tecnologías innecesarias.

# CAPÍTULO III. DESCRIPCIÓN DE LA EXPERIENCIA

Describir detalladamente cuándo y cómo fue el proceso de ingreso a la empresa, mencionar las personas involucradas en el proyecto laboral, explicar las funciones que desempeñó y cómo se desarrolló el proyecto en cada etapa, desde los objetivos, la estrategia, metodología, modelos o herramientas utilizadas para desarrollar la solución desde la identificación del problema, diagnóstico, planificación hasta su implementación del proyecto laboral.

Precisar las consideraciones éticas que siguió y/o se cumplieron en la experiencia.

# CAPÍTULO IV. RESULTADOS

Presenta y analiza los resultados obtenidos con el proyecto laboral. Se puede hacer uso de tablas y figuras (en algunos casos ecuaciones), sin redundar en la información presentada.

Usar el formato de tabla según estilo APA 7.

Ejemplo:

Tabla

Descripción generada automáticamente

Usar el formato de figura según estilo APA 7.

Ejemplo:

Imagen que contiene Gráfico

Descripción generada automáticamente

***Nota****: para esta sección se recomienda considerar un máximo de 5 páginas.*

# CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

**Conclusiones**

Presentar las conclusiones del proyecto desarrollado en la experiencia profesional y las lecciones aprendidas. Las conclusiones deben sustentar cuáles y cómo aplicó sus competencias profesionales en su experiencia laboral.

**Recomendaciones**

Presenta recomendaciones en base a las lecciones aprendidas

# REFERENCIAS

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American*, 284(5), 34–43. <https://www.scientificamerican.com/article/the-semantic-web/>

Chacon, S., & Straub, B. (2014). *Pro Git* (2nd ed.). Apress. <https://git-scm.com/book/en/v2>

Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures* (Doctoral dissertation, University of California, Irvine). <https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf>

Fink, C., & Flatow, M. (2013). *Single Page Web Applications: JavaScript end-to-end*. Manning Publications. https://www.manning.com/books/single-page-web-applications

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Garrett, J. J. (2005). *Ajax: A new approach to web applications*. Adaptive Path. <https://adaptivepath.org/ideas/ajax-new-approach-web-applications/>

Google. (2020). *Web Vitals*. https://web.dev/vitals/

Hunt, A., & Thomas, D. (2019). *The Pragmatic Programmer: Your Journey to Mastery* (20th Anniversary ed.). Addison-Wesley.

Meta. (2023). *React – A JavaScript library for building user interfaces*. <https://react.dev/>

Microsoft. (2024). *TypeScript: JavaScript with syntax for types*. <https://www.typescriptlang.org/>

Morville, P., & Rosenfeld, L. (2006). *Information Architecture for the World Wide Web* (3rd ed.). O'Reilly Media.

OWASP. (2021). *OWASP Top Ten Web Application Security Risks 2021*. https://owasp.org/Top10

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game*. <https://scrumguides.org/>

Vercel. (2023). *Next.js Documentation*. https://nextjs.org/docs

Vercel. (2024). *Deploy web applications with Vercel*. https://vercel.com/docs

Facebook. (2015). *Introduction to GraphQL*. https://graphql.org/learn/

Sobre los estilos de redacción:

El formato de la tesis, las citas y las referencias se harán de acuerdo con el Manual de Publicaciones de la American Psychological Association séptima edición, los cuales se encuentran disponibles en todos los Centros de Información de UPN, bajo la siguiente referencia:

**Código**: 808.06615 APA/D

También pueden consultar la siguiente página web:

<http://www.apastyle.org/learn/tutorials/index.aspx>

Nota importante: La Facultad de Salud (excepto Psicología) utilizará el estilo Vancouver.

# ANEXOS

En los anexos colocar sólo aquellos complementos que signifique evidencias o procedimientos realizados en la investigación de manera pertinente y suficiente. (Evitar excesos innecesarios)

Cada evidencia en los anexos va en hoja independiente.

Cada hoja que contenga un anexo debe ser numerada: ANEXO N° 1. Título del anexo.