



UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO

Facultad de Ingeniería Civil, de Sistemas y de Arquitectura

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS



TRABAJO DE SUFICIENCIA PROFESIONAL

Mejoramiento de la experiencia de usuario en la plataforma de pago de servicios en el canal móvil postpago de telefónica del Perú, a través del cambio de arquitectura monolítica a arquitectura orientada a microservicios

PARA OBTENER EL TÍTULO PROFESIONAL DE

Ingeniero de Sistemas

Autor:

Bach. Saldaña Lara Axel Jesus

Asesor:

MSC. ING. Roberto Carlos Arteaga Lora

Lambayeque – Perú
28 de abril del 2026



UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO

Facultad de Ingeniería Civil, de Sistemas y de Arquitectura

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS



TRABAJO DE SUFICIENCIA PROFESIONAL

Mejoramiento de la experiencia de usuario en la plataforma de pago de servicios en el canal móvil postpago de telefónica del Perú, a través del cambio de arquitectura monolítica a arquitectura orientada a microservicios

PARA OBTENER EL TÍTULO PROFESIONAL DE

Ingeniero de Sistemas

MSC. ING. Roberto Carlos Arteaga Lora
Asesor

Axel Jesús Saldaña Lara
Bach. en Ing. de Sistemas

UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO



Facultad de Ingeniería Civil, de Sistemas y de Arquitectura



ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

TRABAJO DE SUFICIENCIA PROFESIONAL

Mejoramiento de la experiencia de usuario en la plataforma de pago de servicios en el canal móvil postpago de telefónica del Perú, a través del cambio de arquitectura monolítica a arquitectura orientada a microservicios

PARA OBTAR EL TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS

APROBADO POR EL SIGUIENTE JURADO

DR. ING. Luis Alberto Otake Oyama
Presidente

M.S.C. ING. Omar Wilton Saavedra Salazar
Secretario

MSC. ING. Oscar Efraín Capuñay Uceda
Vocal



ACTA DE SUSTENTACIÓN N° 639-2026-UI-FICSA

Siendo las 12:30 pm del día 28 de abril del 2026, se reunieron los miembros de jurado de la Trabajo de Suficiencia Profesional titulada: "MEJORAMIENTO DE LA EXPERIENCIA DE USUARIO EN LA PLATAFORMA DE PAGO DE SERVICIOS EN EL CANAL MÓVIL POSTPAGO DE TELEFÓNICA DEL PERÚ, A TRAVÉS DEL CAMBIO DE ARQUITECTURA MONOLÍTICA A ARQUITECTURA ORIENTADA A MICROSERVICIOS" con código N° IS_V_TSP_2025_002, y designado por Resolución Decanal N° 529-2025-UNPRG-FICSA con la finalidad de Evaluar y Calificar la sustentación del trabajo de suficiencia profesional antes mencionado, conformado por los siguientes docentes:

DR. ING. LUIS ALBERTO OTAKE OYAMA	PRESIDENTE
MSC. ING. OMAR WILTON SAAVEDRA SALAZAR	SECRETARIO
MSC. ING. OSCAR EFRAIN CAPUÑAY UCEDA	VOCAL

Asesorado por MSC. ING. ROBERTO CARLOS ARTEAGA LORA.

El acto de sustentación fue autorizado por OFICIO VIRTUAL N° 85-2026-UIFICSA, el trabajo de suficiencia profesional fue presentado y sustentado por el Bachiller: **AXEL JESUS SALDAÑA LARA**, tuvo una duración de 60... minutos. Después de la sustentación, y absueltas las preguntas y observaciones de los miembros del jurado; se procedió a la calificación respectiva:

	NUMERO	LETRAS	CALIFICATIVO
AXEL JESUS SALDAÑA LARA	19	DIECINUEVE	MUY BUENO

Por lo que queda APTO para obtener el Título Profesional de **INGENIERO DE SISTEMAS** de acuerdo con la Ley Universitaria 30220 y la normatividad vigente de la Facultad de Ingeniería Civil De Sistemas y de Arquitectura de la Universidad Nacional Pedro Ruiz Gallo.

Siendo las 13:30; del mismo día, se dio por concluido el presente acto académico, dándose conformidad al presente acto, con la firma de los miembros del jurado.

DR. ING. LUIS ALBERTO OTAKE OYAMA
PRESIDENTE

MSC. ING. OMAR WILTON SAAVEDRA SALAZAR
SECRETARIO

MSC. ING. OSCAR EFRAIN CAPUÑAY UCEDA
VOCAL

MSC. ING. ROBERTO CARLOS ARTEAGA LORA
ASESOR





UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
Facultad de Ingeniería Civil, de Sistemas y de Arquitectura
UNIDAD DE INVESTIGACIÓN
“Año de la unidad, la paz y el desarrollo”

CONSTANCIA DE VERIFICACIÓN DE ORIGINALIDAD DE TRABAJO DE
SUFICIENCIA PROFESIONAL

Según Res. N° 659-2020-R

Yo, ARTEAGA LORA ROBERTO CARLOS, asesor del trabajo de suficiencia profesional del bachiller:

SALDAÑA LARA AXEL JESUS TITULADO:

“MEJORAMIENTO DE LA EXPERIENCIA DE USUARIO EN LA PLATAFORMA DE PAGO DE SERVICIOS EN EL CANAL MÓVIL POSTPAGO DE TELEFÓNICA DEL PERÚ, A TRAVÉS DEL CAMBIO DE ARQUITECTURA MONOLÍTICA A ARQUITECTURA ORIENTADA A MICROSERVICIOS”

Luego de la revisión exhaustiva del documento constato que la misma tiene un índice de similitud de 19% verificable en el reporte de similitud del programa TURNITIN.

El suscrito analizó dicho reporte y concluyó que, cada una de las coincidencias detectadas NO CONSTITUYEN PLAGIO. A mi leal saber y entender, el trabajo de suficiencia profesional cumple con todas las normas para el uso de citas y referencias establecidas por la Universidad Nacional Pedro Ruiz Gallo.

Se expide la presente según lo dispuesto en la Resolución N° 659-2020-R, de fecha 8 de setiembre de 2020, que aprueba la directiva para la evaluación de originalidad de los documentos académicos, de investigación formativa y para la obtención de Grados y Títulos de la UNPRG:

Lambayeque, 30 de julio de 2025

Atentamente,

MA. ING. ROBERTO CARLOS ARTEAGA LORA
DNI 16755764

Se adjunta:

- Recibo digital de Turnitin
- Revisión de informe en Turnitin



MEJORAMIENTO DE LA EXPERIENCIA DE USUARIO EN LA
PLATAFORMA DE PAGO DE SERVICIOS EN EL CANAL MÓVIL
POSTPAGO DE TELEFÓNICA DEL PERÚ, A TRAVÉS DEL CAMBIO
DE ARQUITECTURA MONOLÍTICA A ARQUITECTURA ORIENTA

INFORME DE ORIGINALIDAD

19%	19%	1%	12%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	www.coursehero.com Fuente de Internet	7%
2	repositorio.uesiglo21.edu.ar Fuente de Internet	1%
3	Submitted to Universidad Europea de Madrid Trabajo del estudiante	1%
4	www.hostinger.co Fuente de Internet	1%
5	dspace.utb.edu.ec Fuente de Internet	1%
6	www.inbest.cloud Fuente de Internet	1%
7	repositorio.ug.edu.ec Fuente de Internet	1%
8	dspace.ups.edu.ec Fuente de Internet	1%
9	Submitted to Universidad de Guayaquil Trabajo del estudiante	1%
10	Submitted to Universidad Nacional Abierta y a Distancia, UNAD, UNAD Trabajo del estudiante	1%

Ing. Roberto Arteaga Lora
DNI 16755764



UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
Facultad de Ingeniería Civil, de Sistemas y de Arquitectura
UNIDAD DE INVESTIGACIÓN



Recibo digital

Este recibo confirma que su trabajo ha sido recibido por **Turnitin**. A continuación podrá ver la información del recibo con respecto a su entrega.

La primera página de tus entregas se muestra abajo.

Autor de la entrega: AXEL JESUS SALDAÑA LARA
Título del ejercicio: Trabajos de suficiencia profesional
Título de la entrega: Mejoramiento de la experiencia de usuario en la plataforma d...
Nombre del archivo: Trabajo_de_suficiencia_profesional-Axel_Saldan_a.pdf
Tamaño del archivo: 1.48M
Total páginas: 36
Total de palabras: 6,406
Total de caracteres: 39,331
Fecha de entrega: 06-ago-2026 12:15p. m. (UTC-0500)
Identificador de la entrega: 2722936371

UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO

Facultad de Ingeniería Civil, de Sistemas y de Arquitectura

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

TRABAJO DE SUFICIENCIA PROFESIONAL

Mejoramiento de la experiencia de usuario en la plataforma de pago de servicios en el canal móvil postpago de telefonía del Perú, a través del cambio de arquitectura monolítica a arquitectura orientada a microservicios

PARA OBTENER EL TÍTULO PROFESIONAL DE

Ingeniero de Sistemas

Autor:
Bach. Saldaña Lara Axel Jesus

Asesor:
MSC. ING. Roberto Carlos Arteaga Lora

Lambayeque - Perú
28 de abril del 2026


Ing. Roberto Arteaga Lora
DNI 16755764

Dedicatoria

A mi familia, por su amor incondicional, su paciencia y su constante apoyo en cada paso de este camino. Su confianza en mí ha sido el motor que me ha impulsado a seguir adelante.

Agradecimientos

A mis padres, quienes siempre me han brindado su amor incondicional y me han apoyado en cada paso de mi vida académica. Sin su confianza y apoyo, este logro no habría sido posible.

A el ingeniero Roberto Arteaga, por su invaluable orientación, paciencia, y apoyo constante durante la realización de esta tesis. Su experiencia y conocimientos han sido fundamentales para el desarrollo de este trabajo.

ÍNDICE GENERAL

RESUMEN.....	8
ABSTRACT	9
I. DEFINICIÓN DEL PROBLEMA	10
II. OBJETIVOS	11
OBJETIVO GENERAL	11
OBJETIVOS ESPECÍFICOS	11
III. LÍMITES DE LA INVESTIGACIÓN	12
IV. JUSTIFICACIÓN DEL INFORME	12
V. FUNDAMENTO TEÓRICO.....	12
Experiencia de Usuario (UX).....	12
Arquitectura monolítica.....	13
Arquitectura orientada a microservicios.....	13
VI. GLOSARIO DE TÉRMINOS.....	14
VII. DESARROLLO DE LA SOLUCIÓN	15
CONCLUSIONES	32
RECOMENDACIONES.....	33
REFERENCIAS BIBLIOGRÁFICAS.....	34

INDICE DE TABLAS

Tabla 1.....	15
Tabla 2.....	16

INDICE DE FIGURAS

Figura 1. Pantalla de selección de monto a pagar en la aplicación móvil.	19
Figura 2. Pantalla de confirmación de pago en la aplicación móvil.....	20
Figura 3. Pantalla de ingreso de datos de la tarjeta para efectuar el pago en la aplicación móvil.	20
Figura 4. Pantalla de confirmación de pago exitoso en la aplicación móvil.	21
Figura 5. Pantalla de selección de tarjeta registrada para efectuar el pago en la aplicación móvil.	22
Figura 6. Pantalla de confirmación de pago exitoso con detalle de la operación.....	22
Figura 7. Arquitectura monolítica (AS-IS).....	23
Figura 8. Pantalla de selección del recibo y monto a pagar en la aplicación móvil.....	25
Figura 9. Pantalla de ingreso de correo electrónico para envío de la confirmación del pago.	25
Figura 10. Pantalla de procesamiento de la transacción en curso.....	26
Figura 11. Pantalla de procesamiento del pago en curso.....	26
Figura 12. Pantalla de confirmación de pago exitoso con detalle de la operación realizada.	27
Figura 13. Pantalla de ingreso de datos de la tarjeta para realizar el pago en la aplicación móvil.	27
Figura 14. Pantalla de selección de tarjeta registrada y opción para agregar nueva tarjeta antes de efectuar el pago.	28
Figura 15. Pantalla de procesamiento de la transacción en curso.....	29
Figura 16. Pantalla de confirmación de pago exitoso con detalle de operación y opciones de compartir o descargar.	29
Figura 17. Arquitectura orientada a microservicios (TO-BE)	30
Figura 18. Gráfico de medición de NPS de usuarios entre Jul-2023 y Jul-2024.....	31

RESUMEN

La empresa Telefónica del Perú, opera en Perú desde el año 1994 y ofrece diferentes servicios de comunicación a sus clientes. En el año 2023 el servicio presentó problemas en su plataforma de pago (aplicación móvil), los clientes experimentaban retrasos y errores al realizar pagos por sus servicios, durante los meses de marzo del 2023 a junio de 2024, se observó un promedio de 17.75% en el NPS (lealtad de clientes), siendo el mes de junio el más crítico con un descenso al 12%. El sistema informático, estaba basado en una arquitectura monolítica que unía el backend y el frontend, esto hacia complicado el mantenimiento, la documentación y la incorporación de nuevas funciones. Debido al requerimiento de gerencia para aumentar el NPS, Telefónica se propuso encontrar soluciones al mantenimiento y escalabilidad del código de la pasarela de pagos de la empresa, se decidió cambiar la arquitectura del proyecto a una arquitectura orientada a microservicios. La nueva arquitectura de microservicios permitió optimizar el flujo de pagos, mejorando significativamente el NPS en la aplicación móvil de Movistar. Esto permitió que el sistema fuera más adaptable a futuros crecimientos, asegurando su escalabilidad tanto vertical como horizontal. Debido a ello, se logró aumentar el NPS en un 200% en el mes de julio del 2024 con respecto al mes de junio del mismo año.

Palabras clave: arquitectura monolítica, arquitectura orientada a servicios, experiencia de usuario

ABSTRACT

Telefónica del Perú has been operating in Peru since 1994, offering various communication services to its customers. In 2023, the payment platform (mobile application) experienced issues, with customers encountering delays and errors when making payments for services. Between March 2023 and June 2024, the average NPS (Net Promoter Score) was observed to be 17.75%, with June being the most critical month, dropping to 12%. The IT system was based on a monolithic architecture that combined the backend and frontend, making maintenance, documentation, and the addition of new features challenging. Due to management's requirement to improve the NPS, Telefónica set out to find solutions for the maintenance and scalability of the company's payment gateway code. It was decided to transition the project architecture to a microservices-oriented architecture. The new microservices architecture optimized the payment workflow, significantly improving the NPS of the Movistar mobile application. This made the system more adaptable to future growth, ensuring both vertical and horizontal scalability. As a result, the NPS increased by 200% in July 2024 compared to June of the same year.

Keywords: monolithic architecture, service-oriented architecture, user experience

I. DEFINICIÓN DEL PROBLEMA

En Telefónica del Perú, ha desempeñado un papel fundamental en la transformación de las telecomunicaciones, logró digitalizar en 4 años la red e instaló más de un millón de líneas telefónicas y desplegar 15,000 kilómetros de fibra óptica. Telefónica del Perú ofrece internet de alta velocidad, telefonía fija, planes postpago y prepago, roaming internacional, soluciones de conectividad, servicios en la nube, Movistar Play, Aplicación Mi Movistar.

El mercado de telecomunicaciones en el Perú es altamente competitivo, esta empresa enfrenta competencia con empresas como Claro, Entel Perú, Bitel, WIN y otros operadores. La competencia en el mercado peruano ha llevado a una constante innovación y mejora en la calidad de los servicios.

La forma en la que se ejecutaban los pagos era mediante una arquitectura monolítica de pagos, era un proyecto que contemplaba la aplicación de software tanto el backend y frontend en uno solo, esto hacía complicado su mantenimiento, documentación, desarrollo de nuevos features y tenía muchos problemas a raíz de que todo se ejecutaba en un solo servidor. Por estos problemas, los clientes del servicio experimentaban caídas y lentitud en sus procesos de pago.

Estas demoras prolongadas y errores en la atención del pago de recibos móviles por el canal de atención Línea Móvil, no solo afectaba la experiencia del usuario, sino que también podía impactar negativamente en la satisfacción del cliente, procesos de cobranza y la eficiencia operativa de la empresa.

El NPS (lealtad de clientes) es un indicador que es utilizado por la empresa Telefónica del Perú, su propósito es evaluar qué tan probable es que los clientes

recomienden una marca o producto a otras personas, lo que sirve como un indicador de su experiencia general y fidelidad. En el año 2023, se buscó mejorar este indicador, que no pasaba del 20% y según la muestra que se tiene tuvo picos negativos de 12%, por pedido de gerencia y arquitectura se propuso cambiar la arquitectura de proyecto monolítico a microservicios.

II. OBJETIVOS

OBJETIVO GENERAL

- Mejorar la experiencia de usuario en la plataforma de pago en el canal Móvil, de la empresa Telefónica del Perú, mediante un cambio en la arquitectura monolítica a orientada a servicios.

OBJETIVOS ESPECIFICOS

- Analizar el flujo existente de la experiencia de usuario en la plataforma de pago de servicios de la empresa Telefónica.
- Diseñar un nuevo flujo para la plataforma de pago de servicios de la APP Movistar.
- Asegurar que la plataforma de pagos sea escalable para manejar incrementos en el volumen de transacciones sin afectar el rendimiento, el mantenimiento y la capacitación continua.
- Incrementar la lealtad de clientes (NPS) del canal Móvil en el APP movistar.

III. LIMITES DE LA INVESTIGACIÓN

Debido a que los datos usados en la presente investigación son sensibles, este informe presentará de forma general algunos de los cambios propuestos. No se darán detalles específicos sobre datos, estructuras o configuraciones para no poner en riesgo la seguridad de la información de la empresa.

IV. JUSTIFICACIÓN DEL INFORME

Gracias a la formación académica en la carrera profesional de Ingeniería de Sistemas en la Universidad Nacional Pedro Ruiz Gallo, cuento con la habilidad y los conocimientos sólidos para identificar, analizar e implementar soluciones de mejora en el desarrollo de los procesos de la empresa u organización en la que me encuentro laborando.

Se espera que el presente informe, constituya un caso que presente una solución a problemas relacionados con tecnología, causados por una inadecuada planificación del proceso de ingeniería en su construcción.

V. FUNDAMENTO TEÓRICO

Experiencia de Usuario (UX)

La experiencia de usuario se refiere a la percepción y respuesta que un usuario tiene al interactuar con el sistema, abarcando cierto tipo de factores como la facilidad de uso, el diseño y la disponibilidad. Está enfocado en facilitar la usabilidad mediante una interacción eficiente usuario-sistema. Uno de los beneficios es el aumento de la satisfacción y fidelidad del usuario, sin embargo, presenta desventajas de las cuales una de ellas es que requiere una inversión continua en pruebas y mejoras (Finn 2025).

Arquitectura monolítica

La arquitectura monolítica es un modelo de desarrollo de software tradicional en el que todos los componentes de la aplicación (código, interfaz, lógica) están integrados y ejecutados en un único programa. Una de las ventajas de esta arquitectura común y centralizada facilita el desarrollo e implementación, con menos complejidad en la gestión de dependencias y buen rendimiento en aplicaciones pequeñas. Sin embargo, tiene como desventaja la dificultad de escalar y mantener una aplicación conforme va creciendo, provocando errores que afectan todo el sistema y la capacidad de crear dependencia tecnológica. (Powell y Samlley 2024)

Arquitectura orientada a microservicios

La arquitectura orientada a microservicios es un enfoque moderno que divide una aplicación en múltiples servicios pequeños, independientes y autónomos que se comunican mediante APIs (Interfaz de programación de aplicaciones). Cada microservicio se encarga de una función específica y puede ser desarrollado, desplegado y escalado de forma independiente. Esta arquitectura tiene una mayor escalabilidad, flexibilidad, resiliencia ante fallos parciales, sin embargo, también implica mayor complejidad en la orquestación de servicios, dificultad para mantener la consistencia de datos.(Amazon Web Services, Inc. 2025)

La arquitectura monolítica se caracteriza por ser una única aplicación integrada que se escala, despliega y mantiene como un conjunto, lo que puede complicar su gestión cuando el código crece y provoca que un error en alguna parte afecte a todo el sistema o aplicación. Por otro lado, la arquitectura de microservicios divide la aplicación en varios servicios autónomos que se pueden escalar y desplegar de forma

independiente, lo que facilita el mantenimiento y mejora la tolerancia a fallos al contenerlos dentro de servicios específicos. (Harris 2024)

Para desarrollar una arquitectura basada en microservicios y optimizar la experiencia de usuario, se emplean diversas tecnologías como SignalR, Angular (un framework JavaScript para crear aplicaciones de una sola página con interfaces dinámicas); y Java, un lenguaje multiplataforma y versátil ideal para construir microservicios escalables. Además, la programación asíncrona es fundamental para manejar múltiples procesos simultáneamente, mejorando la eficiencia en sistemas distribuidos. (Barbosa 2023)

VI. GLOSARIO DE TERMINOS

Sprint	Ciclo de trabajo en la metodología SCRUM, en el que planifican y completan ciertas actividades establecidas.
Reingeniería	Reestructuración de un proyecto, de una empresa o de una institución con el fin de mejorar sus resultados
Timer	Objeto que tiene como funcionalidad ejecutar una función o bloque de código cada cierto tiempo.
Monolito	Modelo tradicional de un programa de software que se complica como una unidad unificada y que es autónoma e independiente de otras aplicaciones.
NPS	Es una métrica que evalúa la satisfacción de los clientes.
JDK	Es un software para los desarrolladores de Java, le permite escribir aplicaciones que se desarrollan una sola vez y se ejecutan en cualquier lugar de cualquier máquina virtual Java.
Concurrencia	Se refiere a la habilidad de distintas partes de un programa, algoritmo, o problema de ser ejecutado en desorden o en orden parcial, sin afectar el resultado final.

CSS	Es un lenguaje de diseño que permite definir la presentación y estilo de un documento HTML, es decir, controla la apariencia de una página web.
Listener	En Apache Kafka, un listener es una interfaz de red a través de la cual los clientes (producir y consumir) pueden conectarse a los brokers de Kafka. Cada listener se identifica con un nombre único y un número de puerto.
POD	Es un conjunto de uno o varios contenedores de Linux® y constituye la unidad más pequeña de las aplicaciones de Kubernetes.

VII. DESARROLLO DE LA SOLUCIÓN

El proyecto se gestionó utilizando metodología ágil SCRUM, para el seguimiento, control y desarrollo del software del proyecto se emplearon diversas herramientas de software como:

Tabla 1.

Herramientas de software empleadas en el proyecto

N°	Herramienta	Uso
1	Jira	Gestión de tareas, planificación de sprints, seguimiento del progreso y control de incidencias dentro del marco de metodologías ágiles. (Atlassian 2023c)
2	Jenkins	Integración continua y despliegue automático de aplicaciones, permitiendo automatizar procesos de compilación, pruebas y entrega. (Jenkins 2023)
3	Bitbucket	Control de versiones basado en Git, gestión de repositorios y colaboración entre desarrolladores en el flujo de trabajo. (Atlassian 2023a)
4	SonarQube	Análisis de calidad del código, detección de vulnerabilidades, bugs y malas prácticas en el desarrollo. (SonarSource 2023)

5	Confluence	Herramienta de documentación colaborativa, utilizada para registrar decisiones, procesos, actas y manuales del proyecto. (Atlassian 2023b)
6	Microsoft Teams	Comunicación y colaboración en tiempo real, integración con otras herramientas y soporte para reuniones virtuales. (Microsoft 2023)
7	Azure DevOps	Gestión integral del ciclo de vida del desarrollo: planificación, repositorios, CI/CD, pruebas y despliegues. (InBest 2021; Microsoft 2022)
8	Angular	Framework para el desarrollo frontend basado en TypeScript, enfocado en la construcción de interfaces dinámicas y escalables. (Google 2023)
9	Spring Boot	Framework de desarrollo backend en Java que facilita la creación de aplicaciones robustas con menor configuración. (Marta 2022)
10	Java	Lenguaje de programación orientado a objetos utilizado en el backend para la lógica de negocio y servicios. (Oracle 2023)

El proyecto fue recibido con un retraso de seis meses de un equipo anterior y se logró llevar a producción en un plazo de nueve meses, equivalentes a 3 PI's (Programan Increment), 36 sprints, se avanzaron las diferentes fases del proyecto, cumpliendo con hitos importantes como la validación de pagos y la gestión de distintos tipos de deudas de los usuarios. La coordinación fue constante mediante reuniones con equipos ubicados en España y Madrid para asegurar aprobaciones y alineación.

El equipo responsable estuvo compuesto por:

Tabla 2.

Equipo del proyecto

Nº	Rol	Responsabilidad	Tiempo
1	Backend	Construir y mantener la lógica del servidor, diseñar y gestionar bases de datos, implementar medidas de	9 meses

		seguridad, desarrollar y documentar APIs, optimizar el rendimiento, asegurar la escalabilidad de la aplicación y colaborar con el equipo de frontend. (Pressman y Maxim 2020)	
2	Frontend	Implementar diseños visuales en código utilizando HTML, CSS, JavaScript y Angular; crear interfaces de usuario funcionales y atractivas; optimizar el rendimiento para tiempos de carga rápidos; garantizar accesibilidad; colaborar con diseño y backend. (MDN Web Docs 2023; W3C 2018)	9 meses
3	Líder Técnico	Definir la arquitectura del sistema, establecer estándares de calidad de código, supervisar la integración entre frontend y backend, asesorar técnicamente al equipo, resolver problemas complejos y garantizar la alineación con los objetivos del proyecto. (Bass, Clements, y Kazman 2021)	9 meses
4	Scrum Master	Facilitar la aplicación del marco Scrum, organizar reuniones diarias (dailys), planificaciones y retrospectivas; remover impedimentos que afecten al equipo; asegurar la correcta comunicación; fomentar la mejora continua y velar por el cumplimiento de tiempos. (Schwaber y Sutherland 2020)	9 meses
5	Product Owner	Definir y priorizar el backlog del producto, representar las necesidades de los usuarios y stakeholders, asegurar la alineación de funcionalidades con la visión del proyecto, validar entregables y tomar decisiones sobre prioridades y valor de negocio. (Roman 2015; Schwaber y Sutherland 2020)	9 meses

6	Technical Project Manager	Planificar y coordinar el proyecto técnico, elaborar cronogramas, asignar recursos, monitorear avances, gestionar riesgos, asegurar la comunicación con stakeholders, supervisar la documentación y garantizar el cumplimiento de tiempos, alcance y presupuesto. (PMI 2021)	9 meses
7	Fullstack	Desarrollar tanto en frontend como en backend, integrando ambas capas; resolver problemas de interoperabilidad; garantizar coherencia entre lógica de servidor y experiencia de usuario; realizar pruebas integrales y colaborar en todo el ciclo de desarrollo. (Narasimhan 2019)	9 meses

Siendo mi rol en este proyecto el de FullStack.

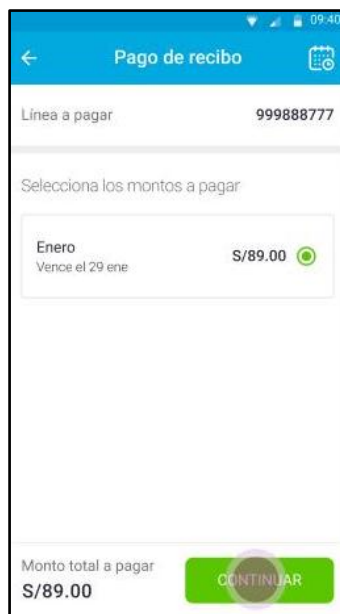
Descripción del Flujo Anterior

A continuación, se describe el flujo que existía, mediante pantallas de la aplicación móvil que utilizaba el cliente al realizar el pago de su servicio en Telefónica. En la pantalla de "Consulta de deuda" (figura 1), la información de los pagos es cifrada mediante una API de Telefónica, luego esta información se descifra en otro proyecto para que pueda visualizarse en la siguiente pantalla (figura 2). En esta segunda pantalla, el usuario ingresa su correo electrónico para recibir la confirmación del pago y, si existen, se muestran las tarjetas guardadas. Si no hay tarjetas guardadas, el usuario debe hacer clic en "Continuar", lo que activa una llamada a la API de la "pasarela de pagos". Los detalles del pago se reciben desde la pasarela de Movistar y, una vez obtenidos, se carga la pasarela de pagos, como se muestra en la figura 3. Posteriormente, el usuario ingresa los datos de su tarjeta y hace click en "Pagar". El

sistema procesa el pago (precaptura) y envía una notificación de captura al monolito. Dado que pueden recibirse varias notificaciones, estas se gestionan para evitar duplicidades. La información se transforma y se envía a una API de Telefónica, encargada de procesar la notificación en los sistemas comerciales y confirmar la captura del pago. Tras esta confirmación, se muestra la figura 4. ante cualquier fallo.

Figura 1.

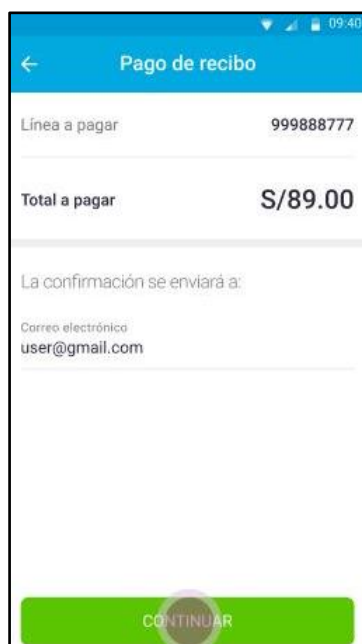
Pantalla de selección de monto a pagar en la aplicación móvil.



Nota. Captura de pantalla de aplicativo antiguo

Figura 2.

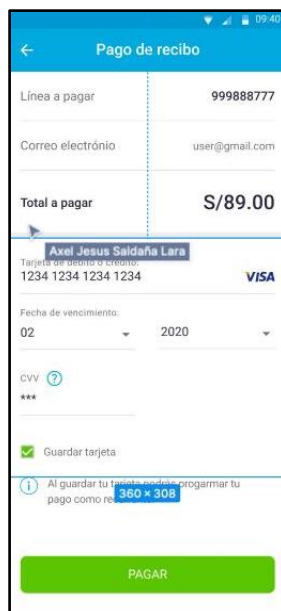
Pantalla de confirmación de pago en la aplicación móvil.



Nota. Captura de pantalla de aplicativo antiguo

Figura 3.

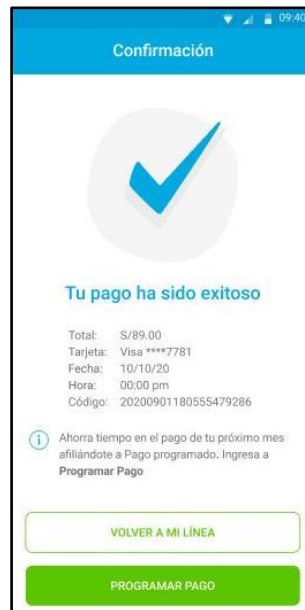
Pantalla de ingreso de datos de la tarjeta para efectuar el pago en la aplicación móvil.



Nota. Captura de pantalla de aplicativo antiguo

Figura 4.

Pantalla de confirmación de pago exitoso en la aplicación móvil.

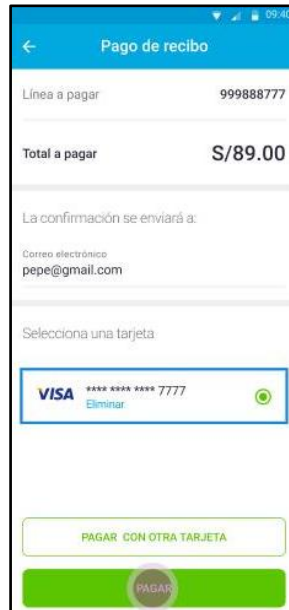


Nota. Captura de pantalla de aplicativo antiguo

Si el usuario tiene tarjetas guardadas, selecciona una de ellas y hace clic en el botón "Pagar" (Figura 5). El flujo sigue el mismo proceso descrito anteriormente, lo que finalmente lleva a mostrar un mensaje de pago exitoso (Figura 6).

Figura 5.

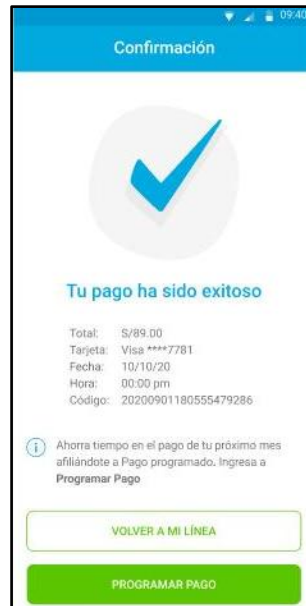
Pantalla de selección de tarjeta registrada para efectuar el pago en la aplicación móvil.



Nota. Captura de pantalla de aplicativo antiguo

Figura 6.

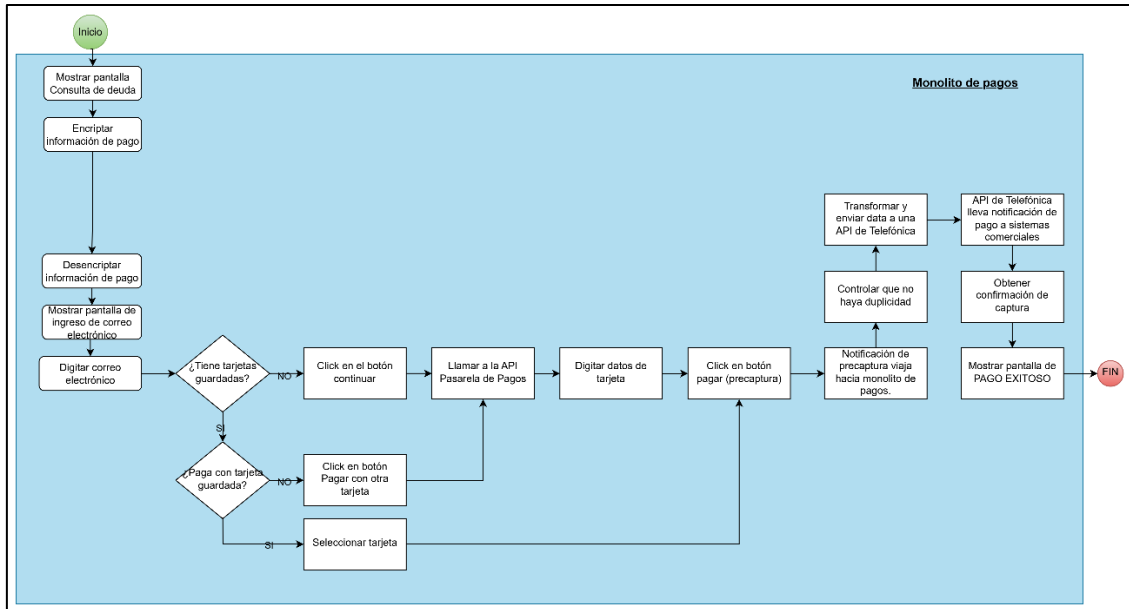
Pantalla de confirmación de pago exitoso con detalle de la operación.



Nota. Captura de pantalla de aplicativo antiguo

Figura 7.

Arquitectura monolítica (AS-IS)



El principal inconveniente en el flujo descrito radica en la cantidad de procesos que, como se muestra en la figura 7, se ejecutan dentro de un mismo proyecto. Esto genera tiempos de espera prolongados (cuellos de botella), ya que es necesario que un proceso finalice antes de que el siguiente pueda comenzar. Además, otro problema significativo es la falta de separación del código; el listener encargado de la confirmación de pago era propenso a fallar en caso de cualquier error.

A su vez, para debuggear se tenía que correr todo el proyecto e ir colocando break points en varios archivos que dificultaban su lectura y a veces no era necesario levantar todo el proyecto sino una sección específica.

Diseñar un nuevo flujo para el proceso de pago de servicios de la APP Movistar.

Se propuso separar el proyecto en backend y frontend. El frontend se optó por seguir usando Angular (Dado que la mayoría del proyecto ya estaba en Angular) y para el backend se propuso una arquitectura de microservicios para reducir la complejidad del proyecto, aumentar la capacidad de respuesta de desarrolladores ante nuevos features, facilitar la documentación, y en caso de alta demanda escalar verticalmente y/o horizontalmente los microservicios para satisfacer la demanda. Todo esto repercutiría en el aumento positivo del NPS del canal Móvil en el APP movistar.

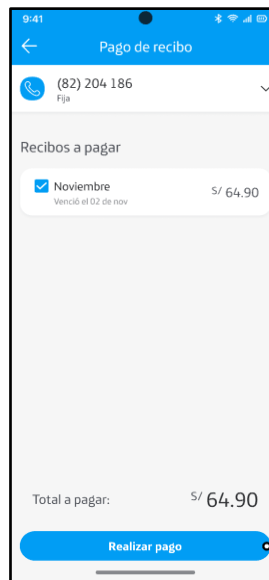
Descripción flujo nuevo

En la pantalla de "Consulta de deuda" (Figura 8), la información de pagos es cifrada mediante una API de Telefónica y luego descifrada en un proyecto que también llama a un microservicio encargado de gestionar las operaciones relacionadas con tarjetas (listado de tarjetas guardadas), lo que permite mostrar la segunda pantalla (Figura 9). En esta segunda pantalla, el usuario ingresa su correo electrónico para recibir la confirmación del pago y puede ver las tarjetas guardadas, si las tiene; en caso contrario, al hacer clic en "Continuar", inicia un microservicio llamado "pasarela". Los detalles del pago se reciben desde la pasarela de Movistar, y una vez obtenidos, se carga la pasarela de pagos. El usuario introduce los datos de su tarjeta y selecciona "Pagar" (Figura 11). El pago es procesado (pre-captura) y la notificación de captura se envía a un microservicio llamado "receptor de notificaciones", el cual gestiona posibles errores y respuestas incorrectas para evitar duplicidades. Posteriormente, se llama a otro microservicio denominado "pagos" realiza una llamada asíncrona trayendo la información inicial de pago validando y enviando al API One Premise de Pagos, esta aplica el modelo asíncrono haciendo impacto comercial y captura casi a la vez para validar y procesar el pago, enviando confirmaciones a una cola de notificaciones. Estas

son leídas por un microservicio llamado “subscriber”, que notifica al Front sobre el éxito del pago mediante SignalR y envía correos, cumpliendo la misma función en caso de error, lo que permite mostrar un mensaje de pago exitoso (Figura 13).

Figura 8.

Pantalla de selección del recibo y monto a pagar en la aplicación móvil



Nota. Captura de pantalla de aplicativo nuevo

Figura 9.

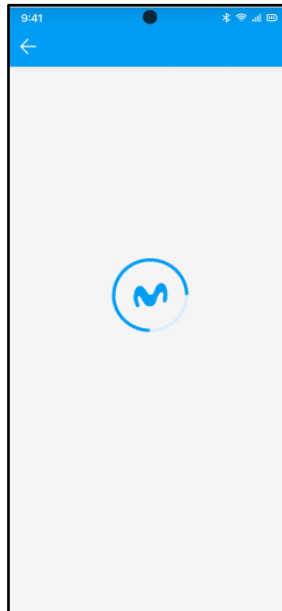
Pantalla de ingreso de correo electrónico para envío de la confirmación del pago.



Nota. Captura de pantalla de aplicativo nuevo

Figura 10.

Pantalla de procesamiento de la transacción en curso.



Nota. Captura de pantalla de aplicativo nuevo

Figura 11.

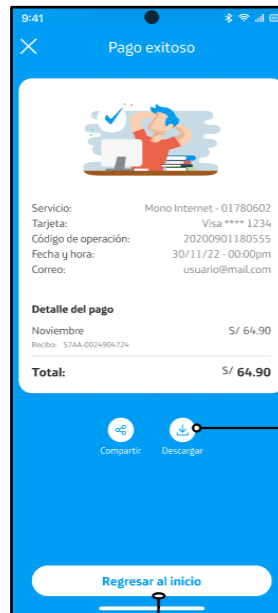
Pantalla de procesamiento del pago en curso.



Nota. Captura de pantalla de aplicativo nuevo

Figura 12.

Pantalla de confirmación de pago exitoso con detalle de la operación realizada.



Nota. Captura de pantalla de aplicativo nuevo

Figura 13.

Pantalla de ingreso de datos de la tarjeta para realizar el pago en la aplicación móvil.

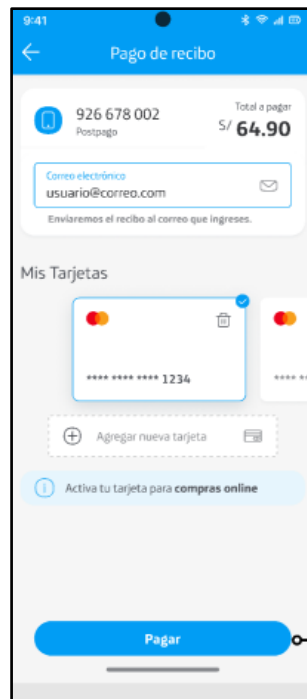


Nota. Captura de pantalla de aplicativo nuevo

Si el usuario tiene tarjetas guardadas, selecciona una de ellas y hace clic en el botón "Pagar" (Figura 14). El flujo sigue el mismo proceso descrito anteriormente, lo que finalmente lleva a mostrar un mensaje de pago exitoso (Figura 16).

Figura 14.

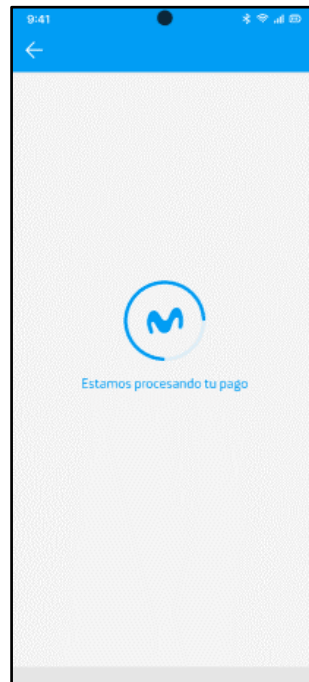
Pantalla de selección de tarjeta registrada y opción para agregar nueva tarjeta antes de efectuar el pago.



Nota. Captura de pantalla de aplicativo nuevo

Figura 15.

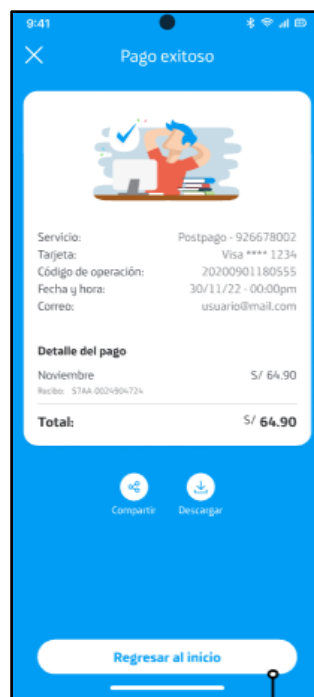
Pantalla de procesamiento de la transacción en curso.



Nota. Captura de pantalla de aplicativo nuevo

Figura 16.

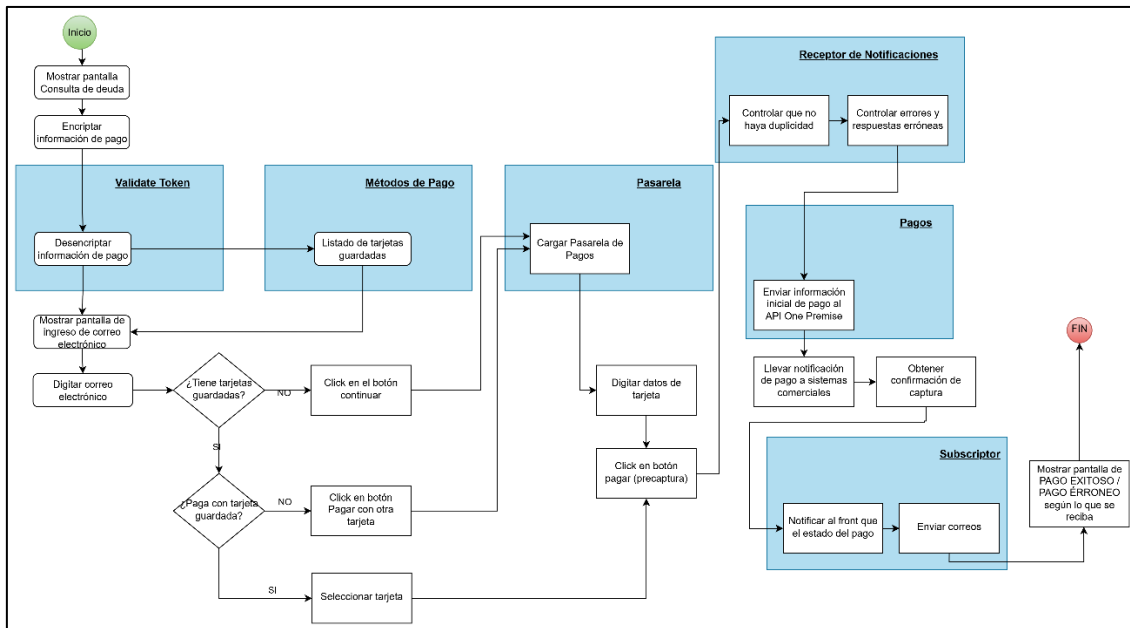
Pantalla de confirmación de pago exitoso con detalle de operación y opciones de compartir o descargar.



Nota. Captura de pantalla de aplicativo nuevo

Figura 17.

Arquitectura orientada a microservicios (TO-BE)



Asegurar que la plataforma de pagos sea escalable

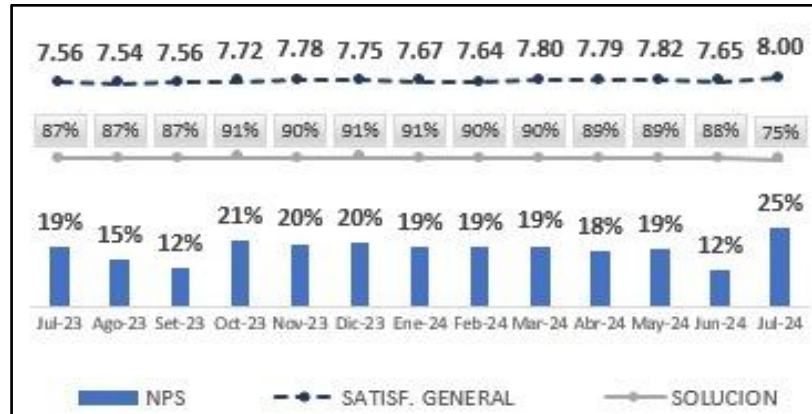
Como se puede observar en la figura, hay microservicios que se encargan de funciones específicas de la aplicación, estos a su vez en caso de alta demanda están preparados para un escalado automático, lo que quiere decir que, si uno se satura, se aumenta automáticamente un POD y hace que los usuarios que estén utilizando el aplicativo no se vean afectados en cuanto a rendimiento. Además, cada microservicio tiene logs en distintas plataformas para monitorear tanto el recurso dockerizado como el microservicio en sí, facilitando la rápida respuesta ante errores.

Otra mejora significativa, es que para debuggear casos específicos se levanta el microservicio específico a la parte o sección de la aplicación que se requiera, ya no de todo el proyecto, lo que facilita su legibilidad, rapidez y es más rápido encontrar algún fallo.

Incrementar la lealtad de clientes (NPS) del canal Móvil en el APP movistar

Figura 18.

Gráfico de medición de NPS de usuarios entre Jul-2023 y Jul-2024



Nota. Imagen enviada como reporte a equipo para evaluar resultados del producto lanzado

Al realizar el comparativo del NPS del flujo anterior (junio) y el actual (Julio) el NPS aumentó poco más del 200% y estableció récord en los últimos 12 meses.

CONCLUSIONES

Se identificó el flujo actual de la arquitectura monolítica en la que se basaba la aplicación móvil utilizada por los clientes del servicio móvil postpago de Telefónica del Perú. Este análisis permitió evidenciar problemas significativos relacionados con el mantenimiento, la documentación y la escalabilidad del sistema, especialmente en lo referente a las APIs que soportaban la plataforma de pagos.

El flujo de la plataforma de pagos fue rediseñado, destacándose mejoras sustanciales en la mantenibilidad y la documentación de los microproyectos creados como parte de esta nueva funcionalidad. Este rediseño no solo optimizó los procesos, sino que también facilitó la integración de nuevas funcionalidades.

La identificación de un bajo NPS fue un factor clave para justificar la transición hacia una arquitectura basada en microservicios. Esta transformación permitió crear un nuevo flujo que mejoró significativamente la experiencia del usuario y, en consecuencia, incrementó el NPS en el canal móvil de la aplicación Movistar.

El cambio a una arquitectura de microservicios garantizó que el sistema de pagos fuera más adaptable a futuros crecimientos, asegurando su escalabilidad tanto vertical como horizontal. Esto resultó en un sistema más eficiente para manejar fallos críticos y resolver cuellos de botella en el procesamiento de transacciones.

El modularidad de los microproyectos resultantes de la nueva arquitectura facilitó la generación de documentación técnica más detallada y comprensible, lo que a su vez mejoró la capacitación del equipo de operaciones encargado de gestionar casos en producción. Esto contribuyó a una mayor eficiencia en la resolución de incidencias y la implementación de mejoras continuas.

RECOMENDACIONES

- Establecer un proceso para realizar revisiones periódicas del código, asignar recursos para corregir problemas de arquitectura o código que puedan comprometer el rendimiento a largo plazo.
- Realizar auditorías de la experiencia del usuario periódicamente y aplicar principios de diseño centrado en el usuario (Design Thinking). Involucrar a los usuarios finales en pruebas de usabilidad para mejorar continuamente la interacción con la aplicación.
- Definir una estrategia de roadmap a mediano y largo plazo, priorizando las nuevas funcionalidades y asegurando que la infraestructura de microservicios esté preparada para escalabilidad y adaptabilidad.
- Implementar un pipeline CI/CD robusto utilizando herramientas como Jenkins, GitLab CI para integrar de manera eficiente los cambios y realizar pruebas automáticas, garantizando la estabilidad de las nuevas funcionalidades.

REFERENCIAS BIBLIOGRÁFICAS

Amazon Web Services, Inc. 2025. “Microservicios”.
<https://aws.amazon.com/es/microservices/>.

Atlassian. 2023a. “Bitbucket Cloud documentation”. <https://bitbucket.org>.

Atlassian. 2023b. “Confluence documentation”.
<https://www.atlassian.com/software/confluence>.

Atlassian. 2023c. “Jira Software documentation”.
<https://www.atlassian.com/software/jira>.

Barbosa, Gabriel Bastos. 2023. “Real-Time Application Development with SignalR and Angular”.
<https://medium.com/@gabrielbastosdeveloper/real-time-application-development-with-signalr-and-angular-74c7d869afaf>.

Bass, L., P. Clements, y R. Kazman. 2021. *Software architecture in practice*. 4a ed. Addison-Wesley.

Finn, T. 2025. “¿Qué es la experiencia de usuario (UX)?” <https://www.ibm.com/mx-es/think/topics/user-experience>.

Google. 2023. “Angular documentation”.

Harris, C. 2024. “Comparación entre la arquitectura monolítica y la arquitectura de microservicios”.
<https://www.atlassian.com/es/microservices/microservices-architecture/microservices-vs-monolith>.

InBest. 2021. “¿Qué es Azure?” <https://www.inbest.cloud/comunidad/que-es-azure>.

Jenkins. 2023. “Jenkins user documentation”.

Marta, I. 2022. “¿Qué es un Service Blueprint y cómo te puede ayudar a optimizar la experiencia de usuario?” <https://www.hiberus.com/crecemos-contigo/que-es-un-service-blueprint-y-como-te-puede-ayudar-a-optimizar-la-experiencia-de-usuario/>.

MDN Web Docs. 2023. “Frontend web developer”. <https://developer.mozilla.org/>.

Microsoft. 2022. “Azure DevOps documentation”. <https://azure.microsoft.com>.

Microsoft. 2023. “Microsoft Teams documentation”. <https://learn.microsoft.com>.

Narasimhan, P. 2019. “Full-stack development: With Java and React. Apress.”

Oracle. 2023. “Java documentation.” <https://docs.oracle.com/javase/>.

PMI. 2021. *A guide to the project management body of knowledge (PMBOK® Guide)*. 7a ed. Project Management Institute.

Powell, P., y I. Samlley. 2024. "What is Monolithic Architecture?" <https://www.ibm.com/think/topics/monolithic-architecture>.

Pressman, R., y B. Maxim. 2020. *Software engineering: A practitioner's approach*. 9a ed. McGraw-Hill.

Roman, P. 2015. "The professional product owner: Leveraging Scrum as a competitive advantage".

Schwaber, K., y J. Sutherland. 2020. "The Scrum Guide". <https://scrumguides.org>.

SonarSource. 2023. "SonarQube documentation". <https://www.sonarqube.org>.

W3C. 2018. "Web content accessibility guidelines (WCAG) 2.1. World Wide Web Consortium." <https://www.w3.org/TR/WCAG21/>.