UNIVERSIDAD PARTICULAR INTERNACIONAL SEK

UISEK BUSINESS Y DIGITAL SCHOOL

Trabajo de Fin de Carrera Titulado:

“Desarrollo e Implementación de un Sistema Service Desk con Flujos Personalizables, Notificaciones, Base de conocimientos, Métricas de Rendimiento y Gestión de Bolsa de Horas para una Empresa de Servicios Tecnológicos”

Realizado por:

EDUARDO NICOLÁS LOZA CORREA

Director del trabajo de titulación:

PHD. VIVIANA ELIZABETH CAJAS CAJAS

Requisito para la obtención del título de:

INGENIERO DE SOFTWARE

Quito, febrero 2026

DECLARACIÓN JURAMENTADA

Yo, EDUARDO NICOLAS LOZA CORREA, con cédula de identidad No. 1719088161, declaro bajo juramento que el trabajo aquí desarrollado es de mi autoría, que no ha sido previamente presentado para ningún grado o calificación profesional; y que ha consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración, cedo mis derechos de propiedad intelectual correspondiente a este trabajo, a la UNIVERSIDAD INTERNACIONAL SEK, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

Eduardo Nicolas Loza Correa

C.C: 1719088161

DECLARATORIA

El presente Trabajo de investigación titulado:

“Desarrollo e Implementación de un Sistema Service Desk con Flujos Personalizables, Notificaciones, Base de conocimientos, Métricas de Rendimiento y Gestión de Bolsa de Horas para una Empresa de Servicios Tecnológicos”

Realizado por:

EDUARDO NICOLAS LOZA CORREA

Como requisito para la obtención del título de:

INGENIERO DE SOFTWARE

Ha sido dirigido por el profesor:

PHD. VIVIANA ELIZABETH CAJAS CAJAS

Quien considera que constituye un trabajo original de su autor

PHD. VIVIANA ELIZABETH CAJAS CAJAS

DIRECTOR

DEDICATORIA

El presente trabajo se lo dedico a Dios y a todos los miembros de mi familia, quienes fueron fundamentales al brindarme su apoyo, motivación y acompañarme siempre para culminar esta importante etapa de mi vida.

A mi hermano Andrés, que desde el cielo me cuida y protege, ha sido ese impulso constante para seguir alcanzando mis metas y cumpliendo mis sueños.

AGRADECIMIENTO

A mis padres, por sus consejos, su sacrificio, su apoyo incondicional y por inculcarme valores que me han guiado a lo largo de mi vida. Gracias por motivarme siempre a seguir adelante y por ser mis guías. Mi mayor deseo es hacerles sentir orgullosos de este logro.

A mi hermana Andrea, quien siempre ha sido mi ejemplo a seguir y ha estado a mi lado en cada momento que lo necesité. Su apoyo y comprensión fue un pilar fundamental en este camino.

A la Universidad Internacional SEK por formar parte de mi formación académica y el fortalecimiento de mi carrera profesional.

RESUMEN

El documento actual presenta el desarrollo e implementación de un sistema Service Desk orientado a mejorar la gestión de tickets en una empresa de servicios tecnológicos. Esta propuesta se enmarca en el contexto actual, donde las plataformas de soporte técnico cumplen una función clave en la eficiencia operativa y en la calidad de la atención al cliente. En este sentido, el uso del módulo Service Desk de Redmine evidenció diversas limitaciones funcionales que dificultan la automatización de procesos, la correcta trazabilidad de la información y el seguimiento adecuado de las incidencias. La investigación se desarrolló bajo un enfoque descriptivo y propositivo, apoyándose en una metodología ágil basada en el marco Scrum, lo que permitió organizar el trabajo en ciclos iterativos y validar de manera progresiva las funcionalidades del sistema. Se desarrolló una plataforma que integra la gestión de tickets y proyectos, incorporando flujos flexibles con acuerdos de nivel de servicio, notificaciones automáticas y un módulo de reportes que incluye métricas y KPIs orientados al control del desempeño operativo y de la bolsa de horas. Adicionalmente una base de conocimientos que sugiere soluciones previas contribuyendo la reducción de incidencias repetitivas y a la optimización del proceso de atención. El sistema cuenta con almacenamiento seguro y un esquema de gestión documental que asegura la integridad y disponibilidad de la información asociada a cada incidencia. Los resultados evidencian que la solución desarrollada mejora la eficiencia operativa del área de soporte, optimiza el flujo general de proyectos y tickets, reduce la duplicidad de incidencias, fortalece el cumplimiento de los acuerdos de nivel de servicio y aporta información relevante para la gestión y planificación organizacional.

Palabras clave: gestión de tickets, métricas y KPIS, Service Desk, soporte técnico.

ABSTRACT

This document presents the development and implementation of a Service Desk system aimed at improving ticket management within a technology services company. This proposal is framed within the current context, where technical support platforms play a key role in operational efficiency and service quality delivered to clients. In this regard, the use of the Redmine Service Desk module revealed several functional limitations that hinder process automation, proper information traceability, and effective incident tracking. The research was conducted under a descriptive and propositional approach, supported by an agile methodology based on the Scrum framework, which enabled the organization of work into iterative cycles and the progressive validation of system functionalities. As a result, a platform was developed that integrates ticket and project management, incorporating flexible workflows with service level agreements, automated notifications, and a reporting module that includes metrics and KPIs for monitoring operational performance and hour-balance consumption. Additionally, the system features a knowledge base that suggests previously applied solutions, contributing to the reduction of repetitive incidents and the optimization of the support process. The platform also includes secure storage and a document management scheme that ensures the integrity and availability of information associated with each incident. The results demonstrate that the developed solution enhances the operational efficiency of the support area, optimizes the overall flow of projects and tickets, reduces incident duplication, strengthens compliance with service level agreements, and provides relevant information for organizational management and planning.

Keywords: metrics and KPIs, Service Desk, technical support, ticket management.

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

CAPITULO 1

INTRODUCCIÓN

1.1 Planteamiento del problema

1.1.1 Diagnóstico del problema

En el contexto actual de la transformación digital, según (Lanet, 2024) el soporte técnico ha dejado de ser un simple servicio de resolución de incidencias para convertirse en un pilar con un impacto directo en la eficiencia operativa, la productividad del equipo y la competitividad general de las organizaciones. Sin embargo, muchas empresas operan con herramientas que no responden a las demandas del entorno digital moderno. Tal es el caso del módulo Service Desk de Redmine (García de Zúñiga, 2024) que fue sacado a producción en el año 2006. Esta herramienta presenta diversas limitaciones significativas en sus flujos de trabajo poco flexibles, ausencia de notificaciones, alertas automáticas y la deficiencia en la gestión de almacenamiento de ficheros y gestión documental de los tickets.

Estas carencias tienen un impacto directo en la capacidad operativa del equipo de soporte técnico duplicando esfuerzos y no garantizando la calidad en atención a sus clientes. Estas limitaciones dificultan el cumplimiento de acuerdos de nivel de servicio (Aranda, 2025) que puede traer sanciones contractuales y pérdida de confianza en la empresa. Por otro lado, la ausencia de trazabilidad documental tras el cierre de un ticket puede representar problemas mayores en auditorías.

1.1.2 Pronóstico del problema

La persistencia del uso del módulo antiguo de Redmine puede traer consecuencias negativas significativas a mediano y largo plazo. Según lo planteado por (Almeida & Rodríguez, 2017) trabajar con un sistema desorganizado con los documentos dispersos sin una trazabilidad dificultan la búsqueda eficiente de información de manera oportuna y entorpece tanto el trabajo interno como externo de la empresa.

Por otra parte (Wuchner, 2024) advierte que el software antiguo se denota que se vuelve cada vez menos confiable con el tiempo y más propensos a fallas y errores. Estos problemas no solo afectan el rendimiento técnico del sistema, sino que también tiene impactos importantes en usabilidad. Desde la perspectiva operativa de la empresa la rigidez de los

flujos de trabajo y la ausencia de automatización de acciones específicas afecta en gran medida la satisfacción del cliente. Esto se traduce en una menor productividad ya que el equipo técnico dedica más tiempo en tareas manuales y repetitivas. Posteriormente (Zaldivar Gamboa, 1995) habla sobre la carencia de métricas claras y reportes dificulta la toma de decisiones basada en datos. Al no tener estos datos no se tiene un total entendimiento y correcto control de tiempos y esfuerzo invertido en el equipo.

1.1.3 Control del pronóstico

Frente a las limitaciones operativas y limitaciones técnicas por el paso del tiempo la solución al problema planteado se centra en la implementación de un sistema web moderno de gestión de tickets que permita superar las limitaciones del módulo Service Desk de Redmine. El sistema debe estar alineado con las tendencias y exigencias tecnológicas actuales incorporando flujos de trabajo flexibles (IBM, 2025) para el manejo de proyectos sea dinámico y no existan tiempos de retraso en acciones asignaciones innecesarias. Las notificaciones automáticas permitirán conocer en tiempo real la apertura de tickets urgentes y así garantizar el cumplimiento de acuerdo de nivel de servicio. Adicionalmente la base de conocimiento almacena soluciones previamente aplicadas a problemas similares. Esto brindará al usuario final la posibilidad de consultar y resolver incidencias de forma autónoma, disminuyendo así la carga de trabajo del equipo técnico y reduciendo la creación de tickets repetitivos por las mismas causas. El módulo estadístico va a permitir generar métricas y reportes útiles, así como tener un mejor control de bolsa de horas.

Como señala (Venegas, 2022) la importancia de una automatización de los sistemas de gestión de tickets es clave para mejorar la calidad del servicio. Su estudio señala que ayuda en reducción de errores, evita reasignaciones innecesarias y disminuye en gran medida los tiempos de resolución de problemas. Estas nuevas tendencias tecnológicas contribuyen a que las empresas manejen grandes volúmenes de tickets de manera eficiente, minimizando costos y mejorando la calidad del soporte al usuario.

1.1.4 Formulación del problema

¿El desarrollo de un sistema web moderno de gestión de tickets puede mejorar la eficiencia operativa, la trazabilidad documental y la calidad del soporte técnico, superando las limitaciones del módulo Service Desk de Redmine en una empresa de servicios tecnológicos?

1.1.5 Sistematización del problema

- ¿Cuáles son las deficiencias técnicas y operativas más relevantes del módulo Service Desk de Redmine que limitan la gestión eficiente de tickets?
- ¿Qué elementos deberían integrarse en una plataforma web actualizada para optimizar el rendimiento del área de soporte técnico?
- ¿Cómo las notificaciones automáticas ayudan a garantizar el cumplimiento de los acuerdos de nivel de servicio?
- ¿Qué tipos de automatizaciones contribuirían a acortar los tiempos de atención y prevenir esfuerzos redundantes?
- ¿Cómo un sistema de gestión documental integrado facilita el acceso y asegura la conservación y trazabilidad de los tickets?
- ¿Qué indicadores clave deberían incorporarse al sistema para medir la calidad del soporte brindado y apoyar la toma de decisiones estratégicas?

1.2 Justificación

La modernización del sistema de gestión de tickets para la empresa de servicios tecnológicos se presenta como una necesidad estratégica y operativa que responde a los desafíos actuales. El soporte técnico ha evolucionado hasta convertirse en un elemento esencial que impacta directamente en la eficiencia operativa, la productividad del equipo y la competitividad organizacional (Franco, 2024). No obstante, la herramienta actualmente utilizada, el módulo Service Desk de Redmine, muestra limitaciones que afectan la calidad del servicio y pueden comprometer la percepción de los clientes más exigentes, especialmente cuando se manejan altos volúmenes de solicitudes o incidentes críticos que requieren seguimiento oportuno.

Varios estudios coinciden en que contar con un sistema de tickets moderno y automatizado es clave para manejar eficientemente una gran cantidad de solicitudes, dar prioridad a los casos más urgentes y mantener una comunicación clara y fluida entre los usuarios y el equipo técnico (Zendesk, 2024). Todas las funcionalidades propuestas que el software tendrá integradas con las tendencias tecnológicas modernas permitirán automatizar funciones, sugerir soluciones basadas en bases de conocimiento y optimizar flujos de trabajo, lo que reduce significativamente los tiempos de resolución y la carga operativa. (López Vargas & Vázquez Chávez, 2016) señalan que la correcta gestión de incidencias en base a un software moderno asegura un orden y control durante todo el proceso, lo cual es fundamental para mantener la calidad y trazabilidad en el soporte técnico.

La implementación de un nuevo Service Desk resulta justificada porque permite elevar la calidad del soporte técnico, mejorar la experiencia del usuario y fortalecer la capacidad de respuesta de la empresa ante auditorías, reclamos o solicitudes complejas. Al mismo tiempo, potencia la competitividad organizacional al contar con una plataforma escalable, flexible y sostenible en el tiempo. Un sistema renovado también facilita la toma de decisiones basada en métricas, permitiendo identificar tendencias, optimizar recursos y anticipar necesidades, lo que contribuye directamente a la estabilidad operativa y al crecimiento de la empresa.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar e implementar un Service Desk web moderno y escalable para la gestión eficiente de tickets para una empresa de servicios tecnológicos con el fin de mejorar la productividad del equipo de soporte técnico, garantizar la trazabilidad documental y generar métricas clave que apoyen la toma de decisiones.

1.3.2 Objetivos específicos

- Levantar y analizar los requerimientos funcionales y técnicos del área de soporte técnico para definir las especificaciones del sistema Service Desk web.
- Diseñar e implementar un módulo de gestión de tickets que incorpore flujos de trabajo flexibles y reglas de acuerdos de nivel de servicio de acuerdo con las necesidades operativas de la empresa.
- Desarrollar una base de conocimientos que permita registrar, consultar y sugerir soluciones a incidencias recurrentes, reduciendo la duplicidad de respuestas y los tiempos de atención.
- Incorporar un sistema de notificaciones por correo electrónico y alertas automáticas que facilite el cumplimiento de los acuerdos de nivel de servicio, así como la actualización continua del estado de los tickets.
- Integrar un módulo estadístico para la generación de métricas clave, seguimiento de acuerdos de nivel de servicio y control del consumo de la bolsa de horas por proyecto.
- Implementar un mecanismo de gestión documental que garantice el almacenamiento, organización y trazabilidad de la información asociada a los tickets.
- Ejecutar pruebas funcionales para validar el correcto funcionamiento del Service Desk y su cumplimiento con los requerimientos definidos.

CAPITULO 2

MARCO TEÓRICO

2.1 Antecedentes sobre Service Desk

A medida que las organizaciones comenzaron a incorporar tecnología en sus operaciones, el soporte técnico surgió inicialmente como un conjunto de prácticas informales atendidas por personal especializado que resolvía problemas conforme aparecían, sin procedimientos estandarizados ni roles claramente definidos. Sin embargo, el aumento progresivo de la dependencia de los sistemas y el crecimiento del número de usuarios hicieron evidente la necesidad de adoptar estructuras más organizadas para gestionar incidentes y solicitudes. Este proceso dio lugar primero al Help Desk y, posteriormente, a la consolidación del Service Desk como función esencial dentro de la gestión de TI.

Bajo las buenas prácticas de ITIL (Mohanakrishnan, 2022), el Service Desk se concibe como el punto único de contacto entre los usuarios y el área de servicios tecnológicos, actuando como un canal centralizado para la atención de incidentes y requerimientos (Ramirez Bravo & Donoso Jaurés, 2006). Su evolución responde tanto a la necesidad de profesionalizar el soporte técnico como a la adopción de metodologías orientadas a optimizar la gestión de servicios en entornos corporativos cada vez más complejos y distribuidos globalmente. En este contexto, el Service Desk según (Schwenke, 2023) se convierte en un componente clave para mejorar la comunicación operativa y la experiencia de los usuarios finales, reforzando su papel dentro de la estrategia de servicios de TI.

En la actualidad, el Service Desk supera ampliamente el enfoque reactivo de sus primeras versiones. Además de resolver incidentes y atender solicitudes, contribuye al cumplimiento de los acuerdos de niveles de servicio, a la administración del entorno tecnológico y a la coordinación de diversos recursos y áreas involucradas. (Franzen, 2023) menciona que su operación promueve la trazabilidad de los procesos, la estandarización de actividades, la reducción de tiempos de respuesta y la centralización del conocimiento institucional. Con el apoyo de métricas, automatización y una visión orientada al servicio, el Service Desk se ha consolidado como un habilitador fundamental para la productividad, la continuidad operativa y la calidad del servicio en las organizaciones contemporáneas.

2.2. Definición de ITSM

La Gestión de Servicios de Tecnologías de la Información (ITSM)(Galup et al., 2009) puede entenderse como un enfoque estructurado que organiza y controla la prestación de servicios tecnológicos para asegurar que operen con estabilidad, seguridad y calidad. Su propósito central es alinear la operación técnica con las necesidades del negocio, generando valor tanto para los usuarios internos como para los clientes externos. Bajo esta lógica, se promueve la estandarización de actividades, la asignación clara de responsabilidades, la optimización de recursos y la disminución de la variabilidad en la resolución de incidentes y solicitudes.

En términos estratégicos como menciona (Salle, 2004) el ITSM funciona como un marco disciplinado que permite conectar de forma efectiva la operación tecnológica con los objetivos institucionales. Gracias a ello, los servicios de TI dejan de percibirse únicamente como un soporte reactivo y pasan a desempeñar un papel fundamental en la productividad y en la ejecución de procesos clave del negocio. Este cambio de perspectiva se refleja en la distinción comúnmente presentada entre TI como proveedor de servicios y TI como socio estratégico, como se observa en la Tabla 1 donde se evidencian las diferencias en madurez, enfoque y aportación de valor

Tabla 1

TI como proveedor de servicios o como proveedor estratégico

Proveedor de Servicios	Socio Estratégico
• El área de TI se enfoca principalmente en mejorar la eficiencia operativa. • Los presupuestos se establecen con base en comparaciones o métricas externas del mercado. • TI se considera una función que puede operar de manera independiente del negocio.	• TI orienta sus esfuerzos a apoyar el crecimiento y la evolución del negocio. • La asignación presupuestaria se define en relación con las prioridades estratégicas de la organización. • La función tecnológica se integra de forma inseparable con las operaciones del negocio.

• La tecnología se percibe como un costo que debe mantenerse bajo control. • Los responsables de TI se desempeñan sobre todo como especialistas técnicos.	• TI se interpreta como una inversión que requiere administración y seguimiento. • Los líderes de TI actúan como solucionadores de problemas orientados a los objetivos empresariales.

2.3 Relación ITIL e ITSM

La relación entre ITSM e ITIL es estrecha y funciona de manera complementaria. Mientras ITSM define el enfoque general para administrar servicios tecnológicos dentro de una organización, ITIL actúa como un marco de mejores prácticas que orienta la forma en que dichos servicios deben diseñarse, operarse y mejorarse. Este marco proporciona lineamientos operativos, políticas, procesos y metodologías que permiten estandarizar la prestación de servicios y elevar su calidad.

Dentro de ITIL, el Service Desk se establece como la función responsable de atender incidentes, solicitudes, comunicaciones y escalamiento. Su papel resulta esencial para mantener los acuerdos de nivel de servicio, servir como punto único de contacto, documentar cada caso y facilitar el seguimiento del ciclo de vida de los problemas. La implementación de estas directrices favorece una mayor eficiencia operativa, disminuye los tiempos de respuesta y permite tener mayor control sobre el entorno tecnológico.

La conexión entre ITSM e ITIL también se evidencia en el enfoque de mejora continua, el cual impulsa a las organizaciones a detectar fallas, ajustar procesos, capacitar al personal y adaptar flujos de trabajo según las necesidades del negocio y los cambios tecnológicos. Gracias a esto, ITIL se convierte en un elemento metodológico clave para lograr una gestión de servicios TI madura y consistente.

Según (Mora et al., 2013) ITIL v3 incorpora una fase completa denominada Service Design, que destaca la importancia de diseñar servicios alineados con los niveles de calidad esperados. Aunque ITIL no define un proceso único para el diseño del servicio, propone cinco dimensiones clave: los servicios, las herramientas de gestión, las arquitecturas tecnológicas, los procesos y los métodos de medición. Su objetivo es crear servicios

innovadores cuyos componentes, políticas y arquitecturas satisfagan tanto las necesidades actuales como futuras del negocio.

2.4 Modelos existentes

Los modelos de atención técnica han evolucionado conforme aumentan las necesidades operativas de las organizaciones. Entre los esquemas más representativos se encuentran el Help Desk y el Service Desk, los cuales responden a distintos niveles de madurez y complejidad en la gestión del soporte.

El Help Desk según (IBM, 2022) constituye la forma más básica y temprana de asistencia tecnológica. Su función principal es recibir, registrar y resolver incidencias de manera reactiva, con el objetivo de restaurar rápidamente la operación habitual de los sistemas. Este enfoque se orienta a inconvenientes de primer nivel, generalmente relacionados con fallas técnicas o interrupciones operativas. Sin embargo, su alcance es limitado debido a que no incorpora actividades más amplias relacionadas con la gestión de servicios ni maneja solicitudes especializadas o de carácter estratégico.

En contraste, de acuerdo a (Betz & Hewitt, 2020) el Service Desk surge como una evolución del modelo tradicional de Help Desk, ampliando sus capacidades más allá de la simple resolución de incidentes. Este enfoque asume el rol de punto central de contacto entre los usuarios y el área de TI, integrando actividades como la gestión de solicitudes, administración del conocimiento, comunicación con usuarios, seguimiento de acuerdos de servicio, control de accesos y coordinación con proveedores externos. Además, su estructura permite mantener trazabilidad, mejorar la integración entre áreas y aportar un mayor valor al negocio.

La diferencia fundamental entre ambos modelos radica en su alcance y su contribución al ciclo de vida del servicio. Mientras el Help Desk suele ser suficiente en contextos operativos simples y con baja demanda tecnológica, el Service Desk resulta esencial en organizaciones con múltiples servicios, procesos críticos o entornos donde se requiere mantener una gestión integral y proactiva. Como se analiza en la Tabla 2, ambos modelos pueden coexistir dependiendo del nivel de madurez de cada organización, funcionando como etapas progresivas dentro de la evolución de la gestión del soporte técnico.

Tabla 2

Help Desk vs Service Desk

Help Desk	Service Desk
• Centrado en resolver fallas y restablecer la operación lo antes posible. • Predomina la atención reactiva ante problemas técnicos. • Adecuado para organizaciones con baja complejidad tecnológica o necesidades operativas simples. • Se percibe como un soporte técnico operativo. • Documentación limitada y enfoque en el registro de incidencias. • Atiende problemas de primer nivel con procesos básicos y poco estructurados.	• Orientado a gestionar servicios de forma integral, incluyendo incidentes, solicitudes y comunicación con usuarios. • Combina acciones reactivas y proactivas para asegurar continuidad y calidad del servicio. • Realiza actividades más amplias como administración del conocimiento, seguimiento de niveles de servicio, coordinación con áreas y proveedores. • Recomendado para entornos con múltiples servicios, procesos críticos y alto volumen de usuarios. • Se integra como un facilitador de servicios que aporta valor al negocio. • Registro completo, trazabilidad, métricas y uso de bases de conocimiento.

2.5 Tipos de Service Desk

Los modelos de Service Desk (Zendesk, 2020) pueden clasificarse según distintos criterios, ya sea por su ubicación física, el tipo de usuario al que dan soporte, el flujo de comunicación o la estructura del soporte que ofrecen. Estas configuraciones permiten a cada organización adoptar el modelo que mejor se ajuste a su operación, su madurez tecnológica y su estrategia

de servicio. Esta clasificación se la puede dividir según distintos enfoques, gestiones y modelos.

2.5.1 Según su ubicación

Service Desk Local

El Service Desk local se caracteriza por operar dentro de la misma sede donde se encuentran los usuarios. Su principal ventaja es la cercanía, lo que facilita la comunicación directa y permite resolver incidentes críticos con rapidez. Aunque mejora notablemente la experiencia del usuario, suele implicar mayores costos operativos debido a la necesidad de personal técnico presencial.

Service Desk Centralizado

En este modelo, el soporte se ofrece desde una unidad central que atiende los requerimientos de todas las sucursales de una organización. Permite un mejor control de la demanda, simplifica la estandarización de procesos y optimiza el uso de recursos técnicos. No obstante, requiere equipos capacitados para comprender las particularidades de cada área o dependencia que atiende.

Service Desk Virtual

El Service Desk virtual opera mediante herramientas tecnológicas de acceso remoto, lo que permite asistir a usuarios en cualquier ubicación geográfica. Este modelo aprovecha canales digitales como correo electrónico, chat o videollamadas, siendo común en empresas que gestionan operaciones distribuidas. Su principal fortaleza es la flexibilidad y la reducción de costos asociados a infraestructura física.

2.5.2 Según enfoque hacia el cliente

Service Desk Interno

Este modelo se orienta exclusivamente a brindar soporte a los miembros de la organización. Su propósito es asegurar la continuidad operativa del personal mediante la atención de incidencias, fallas o solicitudes relacionadas con herramientas internas.

Service Desk Externo

El Service Desk externo atiende a clientes que utilizan productos o servicios de la organización. Puede intervenir en diversas etapas del ciclo de relación con el cliente, desde

dudas previas a una compra hasta asistencia posterior. Su efectividad depende de comprender las expectativas y el nivel de servicio requerido

2.5.3 Según flujo de comunicación

Service Desk Inbound

En el modelo Inbound, es el usuario quien inicia el contacto para solicitar soporte. La organización responde en función de la demanda, generalmente a través de canales formales como llamadas, tickets o mensajes.

Service Desk Outbound

En este caso, el punto de contacto lo inicia la organización. Puede utilizarse para seguimiento de casos, confirmación de entregas, encuestas de satisfacción o comunicación preventiva respecto a cambios o incidentes.

2.5.4 Según estructura de soporte

Service Desk de Nivel Único

En este esquema, un solo grupo de agentes gestiona todo tipo de solicitudes. Se utiliza en entornos donde las consultas son simples o recurrentes, aunque no es común en organizaciones con infraestructura compleja.

Service Desk Multinivel

Este es el modelo más extendido. Divide la atención en niveles: un primer filtro que gestiona incidentes frecuentes, niveles intermedios que atienden solicitudes más técnicas y un nivel especializado para problemas complejos. Esta estructura asegura una atención más eficiente y adecuada a cada tipo de requerimiento.

2.6 Tendencias modernas

En los últimos años, el Service Desk ha experimentado un cambio profundo impulsado por la transformación digital, la inteligencia artificial y la necesidad de ofrecer experiencias más fluidas a los usuarios. De acuerdo con (Danby, 2025), la evolución de la gestión de servicios está marcada por una visión más centrada en el empleado, el uso intensivo de datos para la toma de decisiones, la automatización inteligente y la incorporación de modelos de autoservicio más prácticos y accesibles. Estas tendencias buscan no solo mejorar la

eficiencia operativa, sino también generar mayor valor empresarial mediante experiencias más ágiles, personalizadas y sostenibles.

Una de las características más relevantes es la creciente influencia de la inteligencia artificial en la gestión de servicios. La IA contribuye a mejorar la productividad, reducir costos, acelerar la resolución de incidentes y elevar la satisfacción de los empleados al permitir análisis más profundos y decisiones más informadas. Además, destaca que las organizaciones están adoptando un enfoque más holístico, donde la gestión de servicios ya no se limita exclusivamente a TI, sino que se expande hacia áreas como recursos humanos, finanzas y operaciones, fortaleciendo así la estrategia de Enterprise Service Management.

De igual manera, (Gesto, 2024) señala que la madurez alcanzada por la inteligencia artificial durante 2024 ha permitido que el Service Desk transite hacia modelos más proactivos y autónomos. Según el autor, 2025 consolidó tendencias como la automatización inteligente, métricas basadas en el desempeño de la IA y la expansión del ITSM hacia toda la organización. También enfatiza que los Service Desk modernos deben evolucionar hacia plataformas más resilientes y orientadas al usuario, capaces de anticiparse a los problemas incluso antes de que se generen tickets, reforzando así la continuidad operativa y la seguridad.

Ambos autores coinciden en que el Service Desk actual se centra en la experiencia del usuario, la anticipación de necesidades, la reducción de fricciones y la mejora continua sustentada en datos y automatización. En conjunto, estas tendencias posicionan al Service Desk como un componente estratégico para la transformación digital y el éxito operativo de las organizaciones.

2.7 Acuerdos de nivel de servicios

Los Acuerdos de Nivel de Servicio (SLA)(Aranda, 2025) constituyen documentos formales que especifican los compromisos que un proveedor asume respecto a la calidad y disponibilidad de un servicio. Estos acuerdos detallan parámetros medibles como tiempos de respuesta, tiempos de resolución, niveles de disponibilidad y responsabilidades de cada parte que permiten evaluar de manera objetiva el desempeño del servicio. Este tipo de acuerdo busca establecer expectativas claras, minimizar interpretaciones ambiguas y facilitar procesos de mejora continua.

Desde la perspectiva del cliente, un SLA garantiza que los servicios recibidos cumplen con los niveles acordados, mientras que, para el proveedor, el acuerdo ayuda a delimitar sus obligaciones y reducir posibles disputas. Normalmente, los SLA también especifican los insumos que el cliente debe proporcionar para que el proveedor pueda operar adecuadamente.

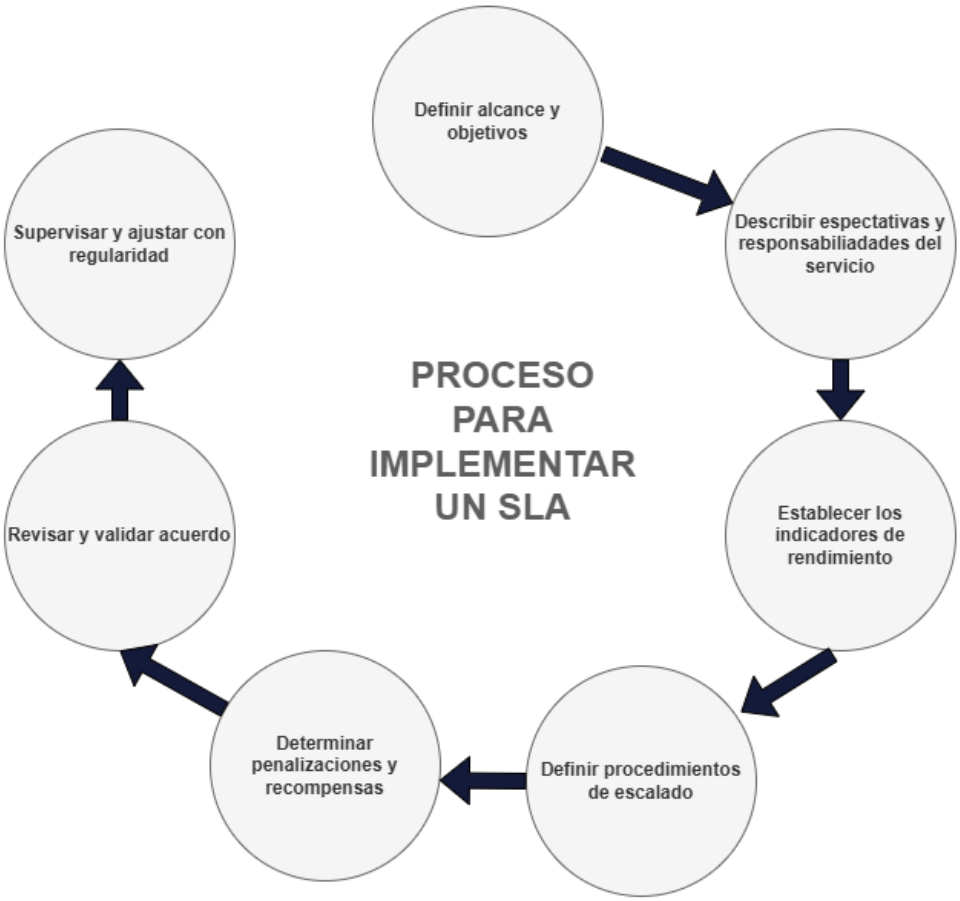
Los acuerdos de nivel de servicio aportan varios beneficios a las organizaciones, entre ellos:

- Claridad de expectativas: Ambas partes conocen con precisión los estándares de rendimiento.
- Mayor responsabilidad: El uso de métricas y posibles sanciones fomenta el cumplimiento.
- Calidad sostenida del servicio: El seguimiento continuo impulsa mejoras.
- Mecanismos para resolver conflictos: Permiten evaluar objetivamente discrepancias.
- Relaciones más sólidas: Favorecen la confianza entre clientes y proveedores.

En la gestión de servicios se manejan diversos tipos de acuerdos de nivel de servicio, cada uno enfocado en regular los compromisos específicos entre las áreas involucradas y garantizar que las expectativas de calidad se cumplan de forma medible y consistente. Esta variedad de acuerdos existe debido a que las organizaciones requieren mecanismos diferenciados para controlar los tiempos de respuesta, la disponibilidad, el alcance operativo y los niveles de soporte ofrecidos; por ello, en la tabla correspondiente se presentan los tipos más utilizados con el fin de mostrar de manera clara cómo se estructuran estas relaciones y qué elementos los distinguen dentro del ciclo de prestación del servicio. Posteriormente, también se contempla el ciclo de vida del servicio de acuerdo a (Nicolazzo et al., 2024), el cual describe de forma secuencial cómo evoluciona un servicio desde su diseño inicial hasta su operación continua, pasando por etapas que buscan asegurar valor, estabilidad y mejora permanente.

Figura 1

Pasos para implementar un SLA



Nota. Adaptado de *Service Level Agreements and Security SLA: A Comprehensive Survey* (p. 7), por S. Nicolazzo, A. Nocera y W. Pedrycz, 2024

Como se observa en la Figura 1, el gráfico representa este proceso de manera visual, destacando cómo cada fase influye en la siguiente y evidenciando que la gestión de servicios no es un evento aislado, sino un sistema continuo donde el diseño adecuado, la transición ordenada y la operación eficiente garantizan la calidad final que percibe el cliente.

Se incluyen también las métricas más comunes asociadas a la gestión del servicio, las cuales permiten evaluar el desempeño efectivo del servicio y compararlo de manera objetiva con los compromisos pactados en los acuerdos de nivel de servicio. Según (Valle, 2025) las prácticas recomendadas de ITIL para la gestión de niveles de servicio, los indicadores de

desempeño más utilizados incluyen medidas relacionadas con tiempos, cumplimiento y satisfacción del usuario.

Tabla 3

Métricas SLA

Métrica	Descripción
Tiempo de respuesta	Cuantifica el intervalo de tiempo transcurrido desde que un usuario registra una solicitud o incidente hasta que el equipo de soporte emite la primera atención formal.
Tiempo de resolución	Mide el lapso total que toma resolver completamente una solicitud o incidente desde su registro hasta su cierre definitivo.
Número de SLA monitorizados	Indica cuántos acuerdos de nivel de servicio están siendo efectivamente supervisados y reportados en un periodo determinado.
Cumplimientos de SLA	Representa el porcentaje de solicitudes o incidentes que fueron gestionados dentro de los parámetros y tiempos acordados en los SLA pactados.

La Tabla 3 resume elementos esenciales que ayudan a identificar tendencias, detectar desviaciones y tomar decisiones informadas, ya que cada métrica aporta una perspectiva específica sobre la eficiencia, el tiempo de atención, la disponibilidad o la satisfacción del usuario.

Posteriormente (Malnik, 2023) menciona que es importante considerar las consecuencias vinculadas al incumplimiento de los acuerdos o al manejo inadecuado del ciclo de servicio, ya que estos fallos pueden derivar en una reducción significativa de la confianza de los usuarios, impactos operativos, interrupciones en procesos críticos y, en casos más severos, pérdidas económicas por la necesidad de correcciones urgentes o penalizaciones. La falta de control sobre estas áreas también puede afectar la percepción general del servicio y comprometer la estabilidad de la operación de TI en su conjunto, lo cual refuerza la

necesidad de monitorear continuamente los niveles de servicio y aplicar acciones de mejora cuando sean necesarias.

2.8 KPIs

Para la variable correspondiente a los Indicadores Clave de Desempeño se tomaron como referencia lineamientos establecidos en marcos de gestión de servicios de TI, ya que, de acuerdo con (Berrospi, 2022), estos indicadores permiten evaluar de forma estructurada la eficacia operativa y la capacidad del servicio para cumplir con los estándares definidos. Bajo esta perspectiva, los KPIs seleccionados funcionan como herramientas que facilitan el control del desempeño del área de soporte en sitio, permitiendo analizar su contribución al cumplimiento de los objetivos del negocio.

Asimismo, tal como señala la documentación técnica y las buenas prácticas asociadas a la gestión de servicios (Gratas, 2023), los KPIs son esenciales para monitorear la calidad del soporte brindado, identificar tendencias, reconocer áreas críticas y asegurar que la experiencia del usuario final se mantenga alineada con las expectativas institucionales. Su medición continua proporciona información confiable para la toma de decisiones y respalda las acciones de mejora que se implementan dentro del proceso. Las dimensiones consideradas para esta variable fueron las siguientes: la medición de la gestión de nivel de servicio, orientada a verificar el grado de cumplimiento de niveles de servicio y la percepción del usuario; la medición de la gestión de la capacidad, que permite identificar situaciones en las que el recurso técnico resulta limitado; la medición de la gestión de incidencias y cambios en sitio, enfocada en analizar tiempos de atención y resolución; y la medición de la evaluación del servicio, que integra la valoración de los usuarios sobre el soporte recibido y las oportunidades de optimización detectadas.

La incorporación de estas dimensiones garantiza un análisis completo y estructurado del desempeño del área de soporte en sitio, favoreciendo la visibilidad del estado real del servicio y fortaleciendo la toma de decisiones orientadas a la mejora continua.

2.9 Desarrollo ágil scrum

El enfoque ágil Scrum (Craddock, 2024) se reconoce como un marco de trabajo que surge para dar respuesta a las dinámicas cambiantes y a la incertidumbre que caracteriza a los entornos de desarrollo de software. Este método aparece como una alternativa flexible frente a modelos tradicionales que suelen presentar limitaciones cuando los requerimientos

evolucionan con rapidez. Scrum propone una forma de organización basada en ciclos cortos y continuos, donde el equipo entrega avances incrementales que permiten validar el progreso y ajustar la dirección del proyecto cada vez que sea necesario. La esencia de esta metodología radica en fomentar la colaboración, aprovechar la experiencia acumulada durante el desarrollo y mantener una mejora continua mediante momentos sistemáticos de inspección y adaptación. Además, uno de sus fundamentos centrales es conservar una visibilidad constante del trabajo, de modo que tanto los integrantes del equipo como los interesados externos puedan comprender el estado del proyecto y participar en decisiones oportunas que contribuyan al logro de los objetivos planteados. En conjunto, estos elementos convierten a Scrum en un mecanismo eficaz para trabajar en entornos complejos, ya que permite reaccionar rápidamente ante cambios sin comprometer la calidad del producto final.

2.9.1 Ventajas scrum en desarrollos incrementales

Scrum se destaca en proyectos que requieren entregas incrementales. Según indica (Palacios, 2023) uno de sus atributos más relevantes es la capacidad de incorporar modificaciones en los requerimientos en cualquier fase del desarrollo sin generar un impacto negativo considerable, debido a la estructura iterativa y los ciclos breves que caracterizan a esta metodología. Cada Sprint, como señala (Craddock, 2024) representa una oportunidad para entregar una parte funcional del producto, lo que favorece una visión orientada al valor y permite a los involucrados evaluar el desempeño del equipo y validar si el camino seguido continúa siendo el adecuado.

Otro aspecto significativo es el rol que cumplen las reuniones retrospectivas, las cuales proporcionan un espacio formal para analizar lo ocurrido durante el Sprint, identificar oportunidades de mejora y realizar ajustes inmediatos que optimicen el trabajo futuro. Esta retroalimentación frecuente fomenta un aprendizaje constante y genera un entorno en el que los equipos suelen experimentar un mayor compromiso con los objetivos del proyecto, incrementando así la motivación, la creatividad y el rendimiento colectivo.

A ello se suma la transparencia inherente a Scrum: el estado del producto, el avance real de las tareas y la carga de trabajo son información disponible para todos los participantes, lo que facilita la toma de decisiones fundamentadas y oportunas. Esta visibilidad continua no solo promueve la confianza entre los miembros del equipo, sino que también permite a los interesados externos intervenir de manera informada, anticiparse a riesgos y asegurar que el proyecto avance conforme a las expectativas. Por estas razones, Scrum se consolida como

un enfoque especialmente útil en desarrollos incrementales donde la flexibilidad, la comunicación y la entrega progresiva de valor resultan esenciales.

A partir de los conceptos y fundamentos teóricos abordados en el marco teórico, se presenta el marco metodológico, en el cual se detalla el tipo de estudio, la metodología de desarrollo, así como el levantamiento de la información y los métodos, técnicas e instrumentos utilizados el desarrollo del estudio.

CAPITULO 3

MARCO METODOLÓGICO

3.1 Tipo de estudio

La investigación que se presenta se inscribe dentro de un estudio con un enfoque descriptivo y propositivo según los lineamientos metodológicos planteados por (Hernández Sampieri & Fernandez-Collado, 2014). No se limita únicamente al análisis de una problemática existente, sino que además plantea una solución tecnológica orientada a optimizar los procesos operativos. El enfoque descriptivo se fundamenta en el análisis detallado de las limitaciones observadas en el módulo Service Desk de la plataforma Redmine, destacando cómo estas fallas afectan negativamente la trazabilidad de los documentos, el rendimiento del personal técnico y el cumplimiento de los acuerdos de nivel de servicio establecidos.

Por otro lado, el carácter propositivo del estudio se refleja en la formulación y desarrollo de una solución tecnológica concreta: un sistema web de gestión de tickets, diseñado con criterios de escalabilidad, automatización y modernización. Este sistema contempla funcionalidades clave como flujos de trabajo flexibles, alertas automáticas, una base de conocimiento reutilizable y herramientas de medición del desempeño.

3.2 Metodología de desarrollo

Se emplea una metodología ágil sustentada en el marco Scrum, ya que este modelo permite abordar proyectos dedicados como el Service Desk mediante ciclos breves y entregas incrementales de funcionalidad. Bajo este enfoque, el trabajo se organiza en Sprints que abarcan etapas de planificación, desarrollo, revisión y análisis retrospectivo, lo que posibilita una evolución continua del producto. de acuerdo con el artículo especializado de (Estrada Velasco et al., 2021). Gracias a esta estructura se pueden realizar modificaciones en los requerimientos y asegurar los componentes esenciales del desarrollo plateado. Asimismo, Scrum promueve un entorno de revisión continua favoreciendo una retroalimentación continua y correcciones a tiempo en caso que se necesite.

3.2.1 Justificación de la metodología

La selección de la metodología ágil Scrum para el desarrollo se justifica por su capacidad para gestionar proyectos caracterizados por cambios frecuentes, incertidumbre y la necesidad de entregar valor de manera continua. Este enfoque resulta particularmente adecuado en entornos donde los requerimientos evolucionan conforme avanza el proyecto, lo común en Service Desk y soporte técnico. Scrum estructura el trabajo en ciclos breves denominados Sprints, lo que permite revisar, ajustar y priorizar funcionalidades en intervalos, manteniendo la alineación entre el desarrollo y las necesidades reales del usuario.

Uno de los argumentos centrales para la adopción de Scrum es su impacto positivo en la calidad del producto. De acuerdo al estudio publicado por (Alami & Krancher, 2022) los equipos que trabajan de manera iterativa mediante Sprints combinados con prácticas de evaluación temprana logran reducir significativamente la cantidad de defectos a lo largo del proyecto. Esta reducción está asociada a la retroalimentación constante, la inspección continua y la capacidad de detectar fallas con anterioridad evitando costos a futuro y garantizando entregas más estables y confiables.

Otro aspecto fundamental es la transparencia y comunicación entre los involucrados. Según el artículo especializado publicado por (Ghimire et al., 2022) las metodologías ágiles fomentan una dinámica colaborativa que mejora la comprensión compartida del proyecto, favorece la toma de decisiones y fortalece la relación entre equipos técnicos y Stakeholders. Los eventos propios de Scrum como las reuniones de revisión, planificación y retrospectivas permiten que los avances y prioridades se comuniquen de forma clara y oportuna, generando un entorno donde la retroalimentación es continua y las expectativas se mantienen alineadas.

Además, Scrum ofrece ventajas significativas en términos de gestión del alcance y uso eficiente de los recursos. Como señala el análisis presentado por (Hassani-Alaoui et al., 2020) la flexibilidad inherente a este marco de trabajo permite ajustar el backlog y reorganizar prioridades sin comprometer la estabilidad del proceso. Esto contribuye a mantener los proyectos dentro de los límites previstos, al mismo tiempo que maximiza la entrega de valor real en cada iteración. El estudio resalta que los proyectos gestionados con enfoques ágiles tienden a obtener mejores resultados en términos de tiempo, satisfacción del usuario y adaptación al cambio.

Scrum favorece la motivación del equipo, ya que otorga autonomía al grupo de desarrollo y promueve la autoorganización. En conjunto, la evidencia académica respalda que Scrum es una metodología adecuada para proyectos como la implementación de un Service Desk, donde se requiere adaptación constante, calidad progresiva, visibilidad total del trabajo y validación frecuente por parte de los usuarios.

3.3 Definición de scrum en el proyecto

3.3.1 Roles

En el marco de este proyecto se adoptó Scrum como enfoque ágil de desarrollo; sin embargo, debido a que se trata de un trabajo académico ejecutado por un único desarrollador, se realizó una adaptación de los roles tradicionales (Hernández et al., 2022) donde se redistribuyen de manera acorde a las necesidades del proyecto y a la estructura propia de la tesis.

Product Owner

Fue asumido por el tutor académico, quien actuó como la figura responsable de validar los avances, priorizar los entregables de acuerdo con los objetivos de la investigación y asegurar que el producto final respondiera adecuadamente al problema planteado.

Scrum Master

Fue desempeñado por el autor del proyecto, quien se encargó de organizar el flujo de trabajo, mantener el orden del backlog, supervisar el cumplimiento de la metodología y resolver cualquier impedimento relacionado con la planificación.

Development Team

Fue ejercido por el autor del proyecto, asumiendo todas las tareas técnicas asociadas al análisis, diseño, desarrollo, pruebas e integración del sistema. Esta adaptación mantiene la esencia de Scrum, pero la ajusta a un escenario en el que un solo miembro es responsable del desarrollo, sin perder la estructura metodológica necesaria para asegurar un proceso organizado y trazable.

3.3.2 Artefactos

Para garantizar una gestión ordenada del proyecto y mantener la coherencia con el marco metodológico, se emplearon los artefactos fundamentales definidos por Scrum. Estos elementos permitieron estructurar las funcionalidades, organizar el trabajo en ciclos

iterativos y asegurar que cada incremento del sistema cumpliera con los criterios establecidos.

Product Backlog

Es el registro donde se acumulan todos los requerimientos funcionales (Díaz & Sánchez, 2022) y no funcionales identificados durante el levantamiento de información. Este listado incluye prioridades preliminares y descripciones necesarias para comprender cada funcionalidad del sistema.

Sprint Backlog

Corresponde a la selección de tareas que se abordaron durante cada iteración. Para este proyecto, el Sprint Backlog se organizó considerando los objetivos del sprint, la complejidad técnica de las tareas y el tiempo disponible para su ejecución.

Se presenta el Sprint Backlog organizado por Sprints en el Anexo A, el cual sirvió como guía estructural para distribuir las actividades:

3.3.3 Eventos

El proyecto implementó los eventos fundamentales de Scrum (Palacios, 2023) adaptados a la naturaleza de un desarrollo individual. El primero de estos eventos fue el Sprint Planning, en el cual se definió el alcance de cada iteración, se seleccionaron los elementos más relevantes del backlog y se establecieron las actividades a desarrollar durante el sprint. Esta planificación incluyó la estimación del esfuerzo y la definición de criterios de éxito asociados al incremento esperado.

Posteriormente se aplicó el Daily Scrum que se utilizó como una rutina de autoevaluación diaria. Esta consistía en revisar el avance del día, identificar tareas pendientes y documentar de forma breve cualquier bloqueo o ajuste necesario. En esta etapa se realizó el Sprint Review, espacio utilizado para finalizar un sprint y realizar una revisión general para concretar la finalización del Sprint, adicional se presentaron los avances al tutor académico, recibir observaciones y validar el cumplimiento de los objetivos planteados.

3.4 Levantamiento de información

El levantamiento de información como indica (Suárez, 2019) fue una fase clave para comprender el funcionamiento actual del proceso de soporte, identificar las limitaciones del módulo usado en Redmine y establecer los requerimientos que orientarán el desarrollo del nuevo sistema Service Desk. Para ello se aplicaron técnicas cualitativas basadas en

observación directa y experiencia práctica dentro del entorno real, analizando flujos de trabajo, tiempos de respuesta, métodos de registro, procedimientos de cierre y la gestión documental asociada a cada incidencia. Esta evaluación permitió detectar problemas recurrentes como la inexistencia de alertas automáticas, nula trazabilidad en archivos adjuntos y ausencia de métricas que faciliten evaluar el rendimiento técnico y la administración de la bolsa de horas.

Durante esta etapa se observaron patrones repetitivos en las solicitudes y actividades manuales que podrían automatizarse con una plataforma más moderna. El conocimiento adquirido desde el uso real del sistema proporcionó una perspectiva interna sobre las necesidades del equipo técnico y clientes, destacando funcionalidades consideradas esenciales para mejorar la atención. Con la síntesis de esta información fue posible construir un conjunto de requerimientos coherentes y alineados con la realidad operativa, asegurando que la propuesta tecnológica responda a necesidades concretas y fortalezca el proceso de gestión de tickets en su totalidad.

Adicionalmente, el levantamiento de información se complementó mediante encuestas aplicadas a profesionales del área de soporte y usuarios recurrentes de Redmine, cuyos resultados se presentan en el Anexo B. Estas encuestas incluyeron preguntas de opción múltiple con escalas de conformidad, las cuales permitieron obtener datos cuantitativos sobre el nivel de satisfacción en el uso diario. Asimismo, se incorporaron preguntas abiertas que aportaron información relevante sobre dificultades operativas, expectativas frente a un nuevo sistema y métricas o KPIs. La integración de estos resultados permitió validar y reforzar los hallazgos obtenidos desde la experiencia práctica, asegurando que la propuesta del nuevo Service Desk se sustente tanto en el uso real como en la percepción directa de los usuarios finales.

3.5 Criterios de validación

Con el objetivo de garantizar que el software desarrollado cumpla con los objetivos planteados, se establecieron criterios de validación orientados a evaluar su correcto funcionamiento, desempeño, seguridad y facilidad de uso. La verificación inicial se centrará en comprobar el cumplimiento íntegro de los requerimientos funcionales levantados previamente, revisando el comportamiento de cada módulo y las interacciones entre ellos. También se evaluará la integridad y consistencia de los datos almacenados, comprobando operaciones de creación, edición y consulta, junto con la trazabilidad documental. En el

módulo de autenticación se valorarán aspectos de seguridad, manejo de sesiones y restricciones de acceso.

El proceso de pruebas (Mera Paz, 2016) contempla la ejecución de envíos de notificaciones por correo mediante un servidor de pruebas, con el fin de demostrar su funcionamiento sin necesidad de infraestructura externa. Paralelamente, se analizará la usabilidad del sistema evaluando navegabilidad, claridad visual y facilidad de aprendizaje de la interfaz, tomando como referencia los principios de Nielsen. (Barros et al., 2024). Cada avance será sometido a revisión durante los Sprints establecidos con el tutor académico, permitiendo retroalimentación continua hasta alcanzar un producto estable y funcional.

3.6 Resultados esperados

Como resultado de la implementación del nuevo Service Desk se prevé disponer de una plataforma web capaz de gestionar tickets de forma organizada desde su apertura hasta el cierre, con procesos más ágiles, mejor control documental y visualización de indicadores clave para la toma de decisiones. Se espera que la herramienta disminuya los tiempos de respuesta del equipo técnico mediante flujos de trabajo más claros, asignación oportuna de solicitudes y notificaciones automáticas que alerten sobre prioridades o acuerdos de nivel de servicio activos, lo que favorecerá el cumplimiento de acuerdos de nivel de servicio. El sistema permitirá también gestionar la bolsa de horas de forma precisa y transparente para los clientes.

Otro resultado importante será la mejora en el manejo de archivos y registros mediante el almacenamiento centralizado que conserve el historial completo de cada ticket, facilitando auditorías y consultas posteriores. La incorporación de una base de conocimiento ofrecerá acceso rápido a soluciones previas, disminuyendo la recurrencia de incidencias repetitivas y fomentando la autonomía del usuario final. A su vez, el apartado de reportes entregará métricas relacionadas con tiempos de atención, volumen de solicitudes y utilización de horas, contribuyendo a una planificación operativa más eficiente.

CAPITULO 4

DESARROLLO E IMPLEMENTACIÓN

A continuación se aborda la fase de desarrollo e implementación, donde se desarrolla la ejecución de los Sprints conforme a la metodología Scrum, permitiendo organizar las actividades, gestionar los avances y consolidar los resultados obtenidos durante el proceso.

4.1 Sprint 1

4.1.1 Introducción al sprint

Nombre

Sprint 1: Planificación, configuración y preparación del entorno de desarrollo.

Duración

4 semanas

Enfoque

Este primer sprint tuvo como propósito establecer los cimientos del proyecto. En este periodo se definieron los requerimientos funcionales y no funcionales, se eligieron las tecnologías que se emplearán en el desarrollo. Asimismo, se redactaron las historias de usuario y se planteó la arquitectura general del sistema junto con sus módulos principales. Adicionalmente se preparó el entorno de desarrollo para asegurar que las fases posteriores evolucionen sobre una base bien estructurada.

4.1.2 Objetivos del sprint

- Levantar y clasificar los requerimientos funcionales y no funcionales
- Identificar los roles para delimitar permisos y responsabilidades.
- Elaborar historias de usuario que alimenten el backlog general del proyecto.
- Construir diagramas base del sistema: casos de uso, componentes, arquitectura y esquema de base de datos.
- Configurar los repositorios, dependencias y contenedores iniciales necesarios para el desarrollo.

4.1.3 Ejecución del sprint

Planificación del proyecto

La planificación general del proyecto correspondiente al Sprint 1 en el cual se desarrolló bajo el marco ágil Scrum descrito en el capítulo 3. Durante esta fase se realizó el levantamiento de información clave para delimitar el alcance de la solución y consolidar una base sólida del proyecto. Se documentaron los requerimientos funcionales y no funcionales, se identificaron los roles del sistema y se redactaron las historias de usuario que dieron forma al backlog inicial. Paralelamente, se seleccionaron las tecnologías, herramientas y extensiones de apoyo a utilizar durante el desarrollo.

Se graficaron los diagramas esenciales para comprender la estructura del sistema: casos de uso, componentes, arquitectura general y el esquema de la base de datos. Estos insumos permitieron visualizar la solución de forma integral y sirvieron como soporte para el diseño técnico que se abordará en los siguientes Sprints.

Toda la planificación y organización de tareas quedó registrada en el Anexo A: Sprint Backlog, donde se describen las actividades programadas, los entregables esperados y su relación con los objetivos trazados, garantizando el cumplimiento de los objetivos y avances del proyecto.

Requerimientos Funcionales

Tras concluir el proceso de recopilación y análisis de la información relacionada con las necesidades operativas del sistema, se definieron los requerimientos funcionales que orientan el diseño y desarrollo propuesto. Estos requerimientos, presentados en la Tabla 4, representan las funcionalidades esenciales que debe ofrecer la plataforma para garantizar una gestión eficiente de usuarios, proyectos, tickets y los respectivos módulos de notificaciones, métricas y reportería.

Los requerimientos detallados en esta tabla abarcan aspectos como la administración de cuentas, el control de accesos, la creación y seguimiento de tickets, el uso de bandejas y filtros avanzados, la gestión documental, la trazabilidad de actividades y el registro de consumo de horas por proyecto. Asimismo, consideran elementos clave como la generación de reportes, el envío de notificaciones automáticas y de correo junto con la integración de una base de conocimientos capaz de sugerir soluciones basadas en incidencias previas.

Tabla 4

Requerimientos Funcionales

Código	Requerimiento Funcional
RF1	Se debe permitir el registro y administración de usuarios.
RF2	Se debe permitir asignar distintos roles.
RF3	Se debe permitir el bloqueo de usuarios
RF4	Se debe controlar el acceso mediante autenticación de usuarios.
RF5	Se debe permitir recuperar contraseña en caso de olvido
RF6	Se debe permitir la gestión de perfil y datos personales.
RF7	Se debe permitir la creación de proyectos con asignación de usuarios.
RF8	Se debe permitir la edición de proyectos permitiendo cambiar usuarios.
RF9	Se debe permitir eliminación de proyectos.
RF10	Se debe permitir la creación de tickets por parte de usuarios internos y externos.
RF11	Se debe permitir clasificar los tickets por proyecto, prioridad y estado.
RF12	Se debe permitir asignar tickets a bandeja general de proyectos o usuario específico.
RF13	Se debe permitir registrar comentarios e hilos a los tickets
RF14	Se debe permitir almacenar documentos e imágenes por tickets e hilos.
RF15	Se debe permitir la visualización de los documentos e imágenes.
RF16	Se debe registrar el consumo de horas por ticket y por proyecto.
RF17	Se debe registrar todo el historial del ticket junto con documentos adjuntos.

RF18	Se debe sugerir soluciones de tickets anteriores al crear un ticket.
RF19	Se debe mostrar una bandeja con filtros avanzados para buscar tickets.
RF20	Se debe enviar notificación por correo cuando se apertura un ticket con SLA.
RF21	Se debe enviar notificación por correo con código de seguridad al recuperar contraseña.
RF22	Se debe enviar notificación cuando se asigne usuario a proyecto, se asigne ticket, se cambie de estado prioridad o actualización.
RF23	Se deben generar reportes con distintas métricas y KPIs
RF24	Se debe permitir visualizar gráficos y Dashboards con reportes.
RF25	Se debe permitir tener filtros dentro del Dashboard de reportería.

Requerimientos No Funcionales

Además de las funcionalidades principales que debe cumplir el sistema, se establecieron también los requerimientos no funcionales, los cuales se encuentran organizados en la Tabla 5. Estos requerimientos complementan el diseño del Service Desk al definir las condiciones técnicas, de seguridad, desempeño y calidad que la plataforma debe garantizar durante su operación. Su incorporación resulta fundamental para asegurar que el sistema no solo brinde las funciones esperadas, sino que también opere con altos niveles de estabilidad, confiabilidad y protección de la información.

Dentro de estos requerimientos se incluyen parámetros como los tiempos máximos de respuesta en operaciones críticas, la disponibilidad del servicio, el rendimiento en la generación de reportes y la adecuada protección de los documentos almacenados mediante Bucket de Amazon S3. Asimismo, se especifican medidas relacionadas con la autentificación y control de accesos mediante AWS Cognito (Tolosa-Cuadrado & González-Sanabria, 2014), el uso de tokens para resguardar los Endpoints de la API y la restricción de

rutas internas del Frontend para evitar accesos no autorizados. También se consideran aspectos vinculados a la escalabilidad del sistema, la mantenibilidad del código y la compatibilidad con diversos navegadores modernos. Estos requerimientos garantizan que el sistema final sea robusto, seguro y adaptable a las necesidades crecientes de la organización.

Tabla 5

Requerimientos No Funcionales

Código	Requerimiento No Funcional
RNF1	Los Endpoints deben dar respuesta máximo 5 segundos
RNF2	Las notificaciones por correo deben ser como máximo de 10 segundos de espera.
RNF3	Se debe garantizar seguridad utilizar AWS Cognito para la autenticación, administración y recuperación de credenciales.
RNF4	Las contraseñas deben gestionarse mediante políticas seguras de Cognito.
RNF5	Se debe asegurar toda la documentación, imágenes y adjuntos almacenados en Amazon S3 y deben estar protegidos con políticas de acceso privadas
RNF6	Los tokens de acceso generados por Cognito deben tener expiración configurable y renovarse mediante mecanismos seguros.
RNF7	Se debe garantizar una disponibilidad mínima del 99 %
RNF8	La interfaz debe ser intuitiva, organizada por módulos y accesible para usuarios con poca experiencia técnica.
RNF9	El sistema debe ser compatible con navegadores modernos.
RNF10	El diseño debe ser responsive, permitiendo acceso desde cualquier dispositivo.

RNF11	El almacenamiento de archivos en Amazon S3 debe permitir escalar indefinidamente conforme aumente el volumen de datos.
RNF12	El sistema debe soportar aumento de usuarios concurrentes sin degradación significativa del sistema.
RNF13	Todos los Endpoints de la API deben estar protegidos mediante tokens garantizando que solo usuarios autenticados puedan realizar operaciones.
RNF14	El sistema debe validar tokens recibidos verificando firma, expiración, permisos y roles.
RNF15	Las rutas del Frontend deben ser accesibles únicamente para usuarios con sesión activa.
RNF16	El sistema debe cerrar sesión y revocar tokens automáticamente después del tiempo configurado de inactividad.

Métricas y KPIs

A partir del levantamiento de requerimientos detallado en el Capítulo 3, se definió un conjunto de métricas y KPIs que serán incorporados en el nuevo sistema Service Desk. La selección de estos indicadores se fundamentó tanto en la experiencia operativa diaria adquirida durante el uso del módulo de soporte de Redmine como en la información obtenida a través de encuestas aplicadas a profesionales del área de soporte como usuarios recurrentes del servicio. Dichas encuestas aportaron datos cuantitativos, mediante escalas de conformidad y selección múltiple, así como información cualitativa derivada de preguntas abiertas, lo que permitió identificar necesidades reales de control, seguimiento y evaluación del servicio.

De manera complementaria, el establecimiento de las métricas se sustentó en estudios previos vinculados a la gestión de servicios de tecnologías de la información. En particular, se consideraron los planteamientos desarrollados por (Bayona-Oré & Hostos, 2022) y

(Trinkenreich et al., 2015) quienes destacan la necesidad de utilizar indicadores de desempeño coherentes con las buenas prácticas definidas por el marco ITIL, con el fin de fortalecer el control de las operaciones, analizar los tiempos de atención y resolución. Como resultado se establecieron métricas orientadas al control operativo de tickets, medición de tiempos, evaluación del desempeño y verificación del cumplimiento de los acuerdos de nivel de servicio, asegurando que el Service Desk propuesto responda de forma efectiva a la realidad del entorno de trabajo.

Las métricas operativas permiten visualizar el volumen, estado y características de las solicitudes gestionadas por el Service Desk. Estos indicadores facilitan el control diario de la carga de trabajo y la identificación de cuellos de botella en el proceso de atención. La Tabla 6 presenta las métricas operativas definidas para el sistema.

Tabla 6

Métricas operativas

Métrica	Descripción
Total de tickets	Representa la cantidad total de solicitudes registradas en el sistema durante un periodo determinado.
Tickets abiertos	Indica el número de tickets que se encuentran activos y pendientes de resolución.
Tickets cerrados	Refleja la cantidad de solicitudes que han sido completamente resueltas y finalizadas.
Estados	Muestra la distribución de los tickets según su estado actual, lo que permite identificar en qué etapa del proceso se concentran las solicitudes.
Prioridades	Clasifica los tickets de acuerdo con su nivel de urgencia.
Antigüedad de tickets	Mide el tiempo transcurrido desde la creación de cada ticket, permitiendo

	detectar solicitudes con tiempos de atención prolongados.

Las métricas de tiempo están orientadas a medir la eficiencia del equipo de soporte y el esfuerzo invertido en la atención de solicitudes. Estas métricas permiten evaluar la productividad y detectar oportunidades de mejora en la gestión del tiempo. La Tabla 7 resume los principales indicadores de tiempo considerados.

Tabla 7

Métricas de tiempo

Métrica	Descripción
Tiempo promedio de resolución	Indica el tiempo medio necesario para cerrar un ticket desde su registro hasta su resolución.
Tiempo registrado promedio	Representa el promedio de tiempo registrado por ticket, reflejando el esfuerzo operativo invertido.
Tiempo total registrado	Corresponde a la suma total de horas trabajadas en todos los tickets gestionados.
Actualizaciones totales	Contabiliza el número total de interacciones realizadas en los tickets.
Actualizaciones públicas	Muestra las interacciones visibles para los usuarios finales.
Actualizaciones privadas	Representa las comunicaciones internas del equipo de soporte.

Los KPIs de desempeño permiten evaluar la productividad y el consumo de recursos por proyecto, proporcionando información clave para la toma de decisiones administrativas. La Tabla 8 presenta los principales indicadores definidos en este ámbito.

Tabla 8

KPIs de desempeño

Métrica	Descripción
Top usuario por tiempo	Identifica al usuario que registra la mayor cantidad de horas trabajadas, permitiendo analizar la distribución de la carga laboral.
Top proyecto por tiempo	Muestra el proyecto que concentra el mayor esfuerzo operativo.
Top tickets por tiempo	Identifica las solicitudes que han requerido mayor inversión de tiempo, facilitando el análisis de complejidad o recurrencia.

Las métricas de SLA permiten verificar el grado de cumplimiento de los compromisos contractuales establecidos con los clientes. Estos indicadores son fundamentales para evaluar la calidad del servicio prestado y el uso adecuado de la bolsa de horas. La Tabla 9 presenta las métricas SLA incorporadas en el sistema.

Tabla 9

Métricas SLA

Métrica	Descripción
Primera respuesta	Define el tiempo máximo permitido para brindar la atención inicial a un ticket.
Resolución máxima	Establece el plazo límite para resolver completamente una solicitud.
Bolsa de horas	Representa el total de horas disponibles según el contrato.
Horas usadas	Muestra la cantidad de horas ya consumidas.
Horas restantes	Indica las horas aún disponibles dentro de la bolsa contratada.

Utilización de bolsa	Expresa el porcentaje de uso del total de horas contratadas.
Tickets con SLA	Identifica las solicitudes sujetas a acuerdos de nivel de servicio.
Tickets sin SLA	corresponde a los tickets que no aplican a dichos acuerdos.
Cumplimiento SLA	mide el porcentaje de tickets gestionados dentro de los parámetros establecidos.

En conjunto, estas métricas y KPIs permiten un control integral del Service Desk, facilitando el monitoreo continuo del desempeño, la identificación de desviaciones y la aplicación de acciones correctivas oportunas. Su implementación contribuye a mejorar el servicio, fortalecer la toma de decisiones y asegurar una gestión alineada con las expectativas operativas y contractuales.

Historias de Usuario

Con el fin de organizar de manera clara los requerimientos del sistema y asegurar que cada funcionalidad cumpliera con las necesidades del usuario final, se optó por emplear Historias de Usuario (Izaurralde, 2013) como mecanismo principal de definición. Este enfoque, recomendado por el marco de trabajo Scrum, permite describir las funciones desde la perspectiva del usuario, establecer criterios de aceptación verificables y garantizar que el desarrollo se mantenga alineado con los objetivos del proyecto.

Bajo esta metodología, se elaboró un conjunto estructurado de historias que representan las acciones clave dentro del sistema. Para facilitar su interpretación y planificación, dichas historias fueron agrupadas y resumidas en la Tabla 6, donde se presenta una visión general de cada una. No obstante, para una revisión más detallada del contenido íntegro de cada historia, estas se encuentran documentadas de forma completa en el Anexo B: Historias de Usuario.

Tabla 10

Historias de Usuario

Código	Título de Historia
HU-1	Inicio de Sesión (Login)
HU-2	Registro de Usuario
HU-3	Formulario de Recuperación de Contraseña
HU-4	Header
HU-5	Barra de navegación
HU-6	Pantalla de Inicio (Home)
HU-7	Gestión de Perfil de Usuario
HU-8	Actualización de Contraseña
HU-8	Listado de Usuarios
HU-10	Edición de Usuario
HU-11	Listado de Proyectos
HU-12	Actualización de Proyecto
HU-13	Creación de Nuevo Proyecto
HU-14	Listado de Tickets
HU-15	Creación nuevo Ticket
HU-16	Visualización de Ticket
HU-17	Creación de Nuevo Hilo en Ticket
HU-18	Pantalla “Mi Bandeja”
HU-19	Pantalla de Notificaciones
HU-20	Indicador de Notificaciones no Leídas (Badge)
HU-21	Pantalla Reportes general
HU-22	Pantalla Reportes SLA
HU-23	Pantalla Home

Identificación de Roles

Durante el proceso de planificación se determinó que para un funcionamiento ordenado del Service Desk, era necesario establecer cuatro perfiles con distintos niveles de acceso y funciones. Según (Atkinson, 2020) la creación de estos roles permite distribuir

responsabilidades, controlar permisos y garantizar que cada actor interactúe únicamente con las actividades que le competen. De esta forma, se definieron los roles de Administrador, Project Manager, Usuario y Cliente detallados en más profundidad en la Tabla 7.

Tabla 11

Roles y Responsabilidades

Rol	Responsabilidades	Justificación
Administrador	Administra el sistema a nivel global, gestiona usuarios, permisos y configuraciones.	Se requiere una figura con control total para asegurar la correcta operación del Service Desk.
Project Manager	Supervisa proyectos asignados, crea nuevos proyectos y coordina la ejecución de tareas y tickets asociados. Actúa como responsable del proyecto.	Permite tener un líder operativo sin otorgar privilegios completos como el administrador.
Usuario	Perfil estándar encargado de generar, gestionar y responder tickets.	Mantiene el flujo operativo del sistema mediante la atención directa de solicitudes.
Cliente	Representa al usuario externo, registra incidentes relacionados con su proyecto y consultar el estado de sus tickets no privados.	Proporciona acceso controlado al Service Desk para reportar problemas sin intervenir en gestiones internas.

Diagrama de Casos de Uso

Con los roles y sus responsabilidades ya establecidos, se desarrolló el diagrama de casos de uso con la finalidad de representar de manera visual qué operaciones puede ejecutar cada actor dentro del sistema. Esta representación en la Figura 2 permite comprender los accesos y restricciones asociados a cada perfil, así como las interacciones que se generan con las funcionalidades disponibles.

Figura 2

Diagrama Casos de Uso

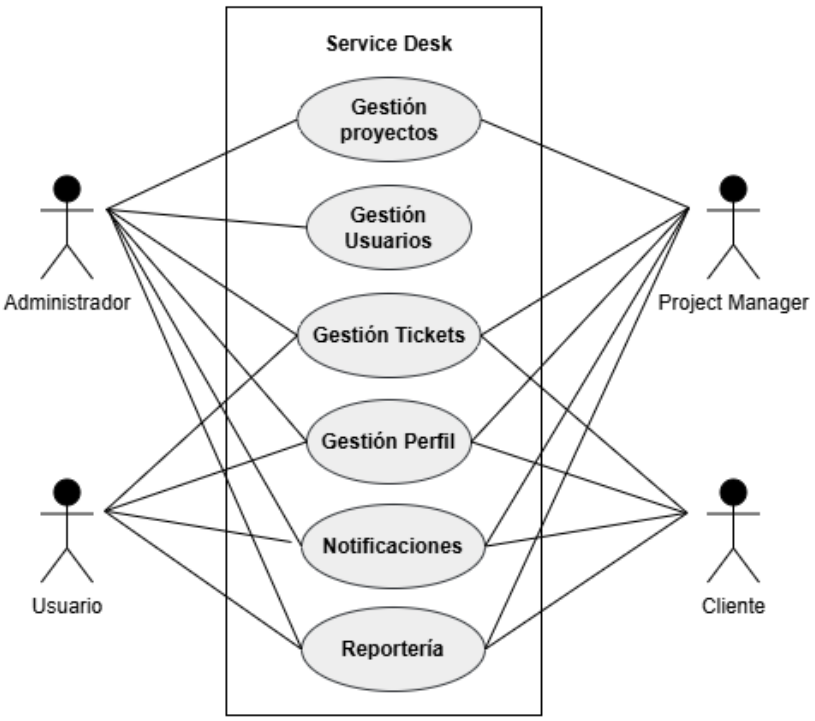
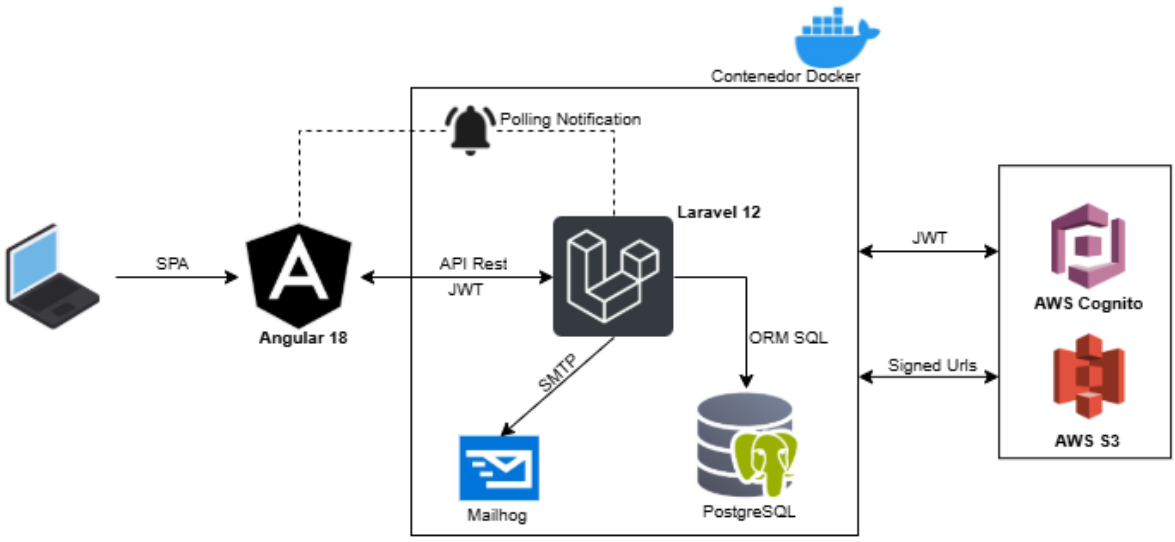


Diagrama de Arquitectura

El diagrama de arquitectura representado en la Figura 3 proporciona una perspectiva global del sistema y de cómo se distribuyen sus elementos. Para la construcción de la solución se decidió utilizar una arquitectura monolítica (Jovanović, 2024), apoyada en servicios ofrecidos por AWS con el fin de asegurar disponibilidad y facilitar la gestión de recursos. Dentro del esquema se pueden identificar las capas que lo componen: la interfaz de usuario correspondiente al Frontend, la capa lógica del Backend, el motor de base de datos, los servicios complementarios en la nube. Asimismo, el Backend se ejecuta mediante contenedores Docker, lo que permite estandarizar el entorno y simplificar tareas de despliegue y mantenimiento. En conjunto, este diagrama permite comprender la organización interna del software y la forma en que cada componente interactúa dentro de la arquitectura definida.

Figura 3

Diagrama de Arquitectura



Esquema Base de Datos

El esquema de base de datos representado en la Figura 4 fue elaborado para el Service Desk representa de manera organizada y lógica los elementos que integran la base de datos. Esta estructura permite visualizar cómo se relacionan las diversas entidades y cómo estas conexiones contribuyen al cumplimiento de los objetivos funcionales establecidos. El diseño se desarrolló tomando en cuenta los requerimientos definidos para el proyecto, garantizando así que la base de datos respalde adecuadamente el flujo de información y las operaciones necesarias para el funcionamiento del sistema.

Figura 4

Esquema Base de Datos

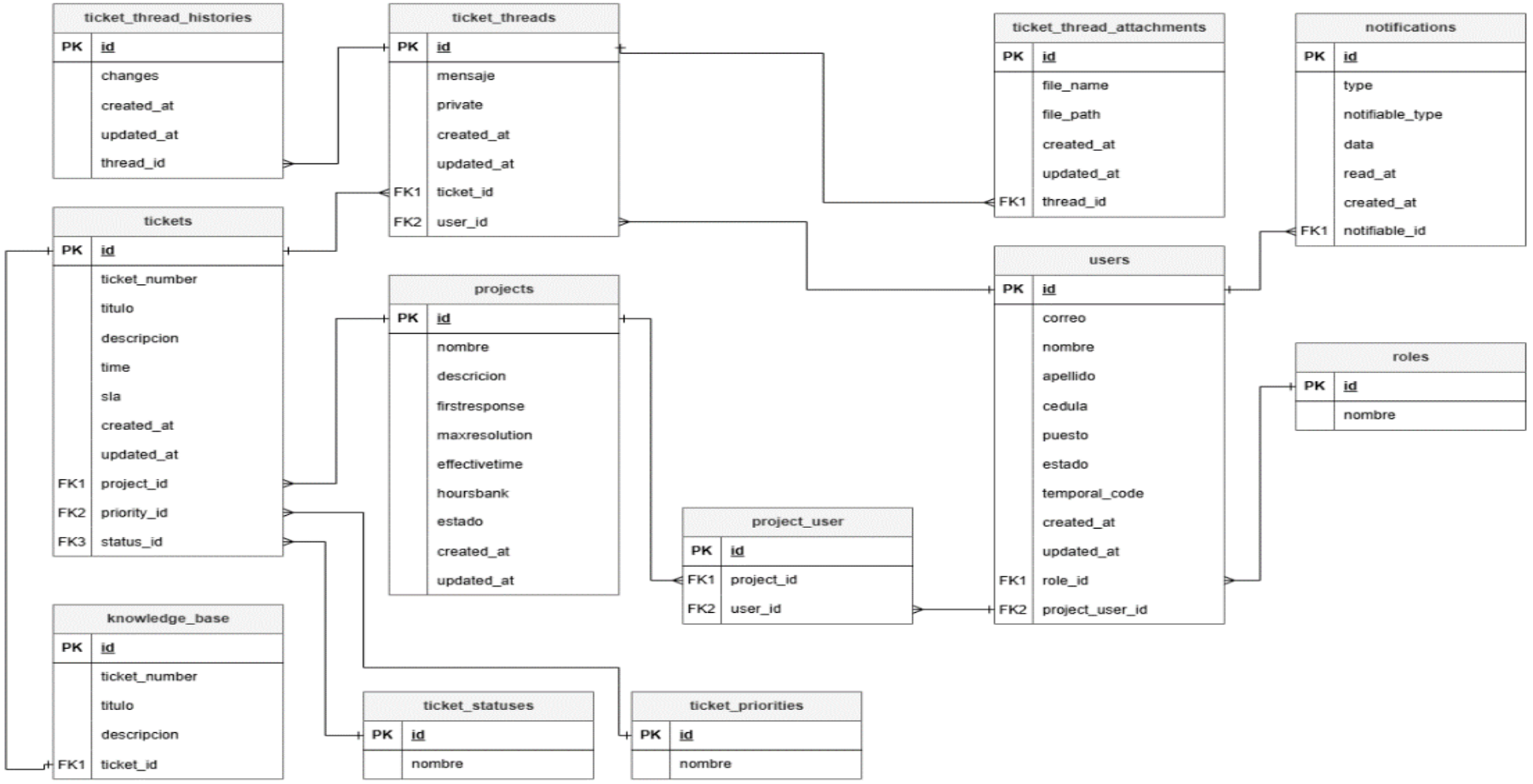
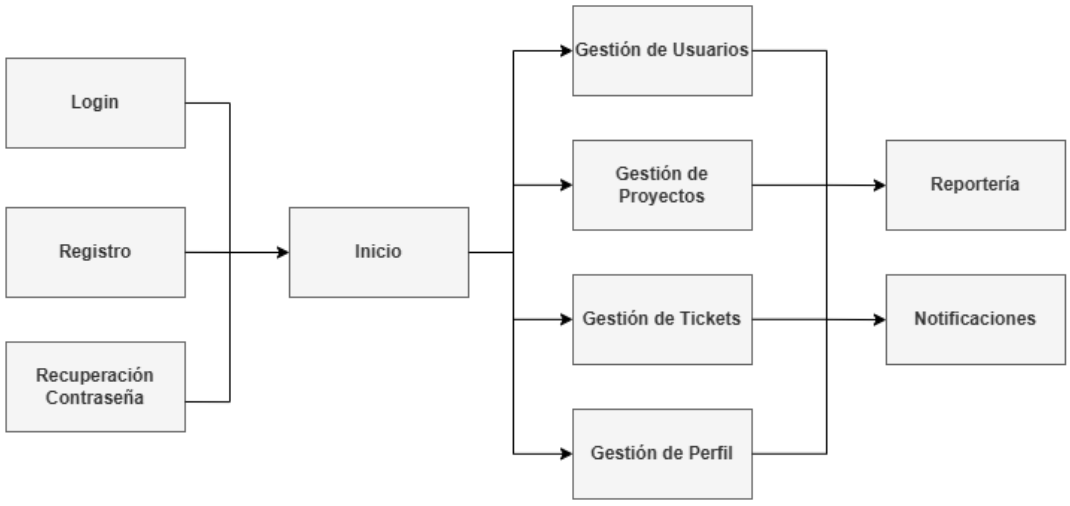


Diagrama de Componentes

El diagrama de componentes ilustrado en la Figura 5 representa la estructura funcional del Service Desk, abarcando desde los puntos de interacción del usuario hasta los módulos encargados de la gestión de proyectos y tickets. Mediante esta representación es posible comprender cómo se organizan los elementos del sistema, así como las rutas de navegación que conectan cada módulo. Se busca contar con una interfaz clara y fácil de recorrer, evitando duplicidad de funciones y delimitando de manera precisa las responsabilidades de cada componente. Se pretende favorecer la comunicación entre las partes del sistema y optimizar los procesos internos para lograr un flujo de trabajo más eficiente y automatizado.

Figura 5

Diagrama de Componentes



Lenguajes de Programación

Tabla 12

Lenguajes de programación

TypeScript	Se utilizó debido a que permite un desarrollo más seguro y estructurado del Frontend. Su tipado fuerte reduce errores en tiempo de ejecución y facilita el mantenimiento del código.

SCSS	Se incorporó para gestionar estilos de forma modular y organizada. Sus ayudan a mantener coherencia visual y simplifica la creación de diseños escalables y componentes reutilizables mejorando la mantenibilidad del proyecto.
PHP	Se empleó para el Backend debido a su compatibilidad con Laravel, su facilidad para gestionar lógica del lado del servidor y su rendimiento optimizado.

Frameworks

Tabla 13

Frameworks

Angular	Fue elegido por su arquitectura robusta basada en componentes, ideal para construir interfaces dinámicas y organizadas. Su control del ciclo de vida de la aplicación es ideal para el tipo de proyecto.
Laravel	Se seleccionó por su estructura modular y clara, su ORM integrado facilita la gestión y comunicación con la base de datos.

Base de Datos

Tabla 14

Base de Datos

PostgreSQL	Se usó por su estabilidad y su rendimiento. Su compatibilidad con Laravel y su capacidad para manejar grandes volúmenes de datos lo convierten en una opción ideal para el Service Desk.

Infraestructura y Contenerización

Tabla 15

Infraestructura y Contenerización

Docker	Se utilizó para contenerizar el Backend. La idea principal es que el proyecto cuente con una configuración consistente asegurando estabilidad y duración a largo plazo evitando conflictos.
Mailhog	Se integró para simular el envío de correos y contar con un flujo de correo de forma segura y controlada.
Opcache	Se habilitó para mejorar el rendimiento del Backend. Se reduce la carga del servidor y acelera la respuesta en operaciones repetitivas.

Servicios y Herramientas Externas

Tabla 16

Servicios y Herramientas

AWS Cognito	Se implementó para gestionar la autenticación de usuarios de forma segura, evitando almacenar credenciales y datos sensibles directamente en la base de datos. Ofrece escalabilidad, soporte para múltiples flujos de login, registro, recuperación de contraseña y cumplimiento de buenas prácticas de seguridad.
AWS S3	Se utilizó como repositorio de archivos ya que proporciona un entorno altamente seguro y diseñado para garantizar la integridad y consistencia de los datos fundamental para el Service Desk. Su capacidad de escalar sin límites ni problemas de saturación asegura que los archivos se mantengan íntegros.
GitHub	Se empleó como plataforma de control de versiones y el seguimiento de cambios

Extensiones y Dependencias

Tabla 17

Extensiones y Dependencias

aws-sdk	Esta librería permitió integrar el Backend con los servicios de AWS integrando sus servicios.
php-jwt	Se utilizó para manejar la creación y validación de tokens JWT en procesos de autenticación
aws-amplify	Se incorporó para facilitar la conexión del Frontend con AWS Cognito y AWS S3.
jwt-decode	Permite interpretar tokens JWT en el Frontend, extrayendo datos del usuario sin necesidad de consultas adicionales al Backend.
Angular Material	Se empleó para crear una interfaz moderna y consistente. Permitió acelerar el diseño visual y garantizar uniformidad en la interfaz.
ngx-file-drop	Esta extensión se utilizó para permitir carga de archivos mediante arrastrar y soltar. Mejora la experiencia del usuario y simplifica la interacción con los módulos de almacenamiento de archivos.
ngx-charts	Se integró para generar reportes visuales dinámicos. Facilita la interpretación de datos mediante gráficos modernos e interactivos.
Tailwind-css	Permite el manejo de clases y estilos para los componentes, así como también el manejo responsivo.

Configuración Técnica del Proyecto

Como punto inicial del desarrollo se generaron los repositorios Github como se muestra en la Figura 6 y 7 destinados al control de versiones tanto del Frontend como del Backend. Esta estructura permitió organizar el código, mantener un historial claro de cambios. En la Figura 6 y la Figura 7 se muestran las vistas correspondientes de los repositorios.

Figura 6

Repositorio Github Frontend

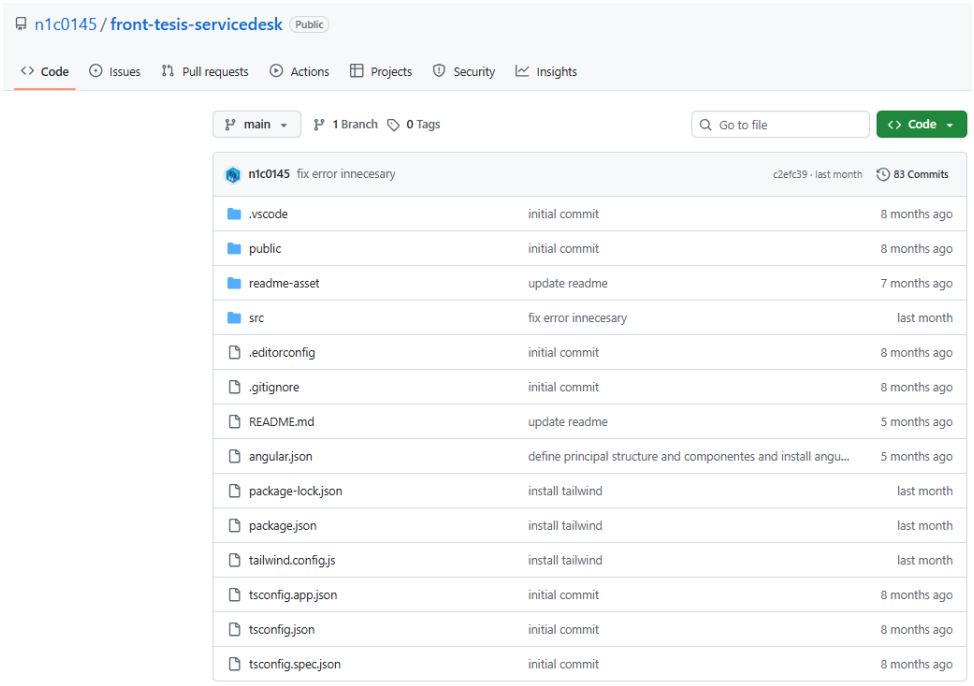
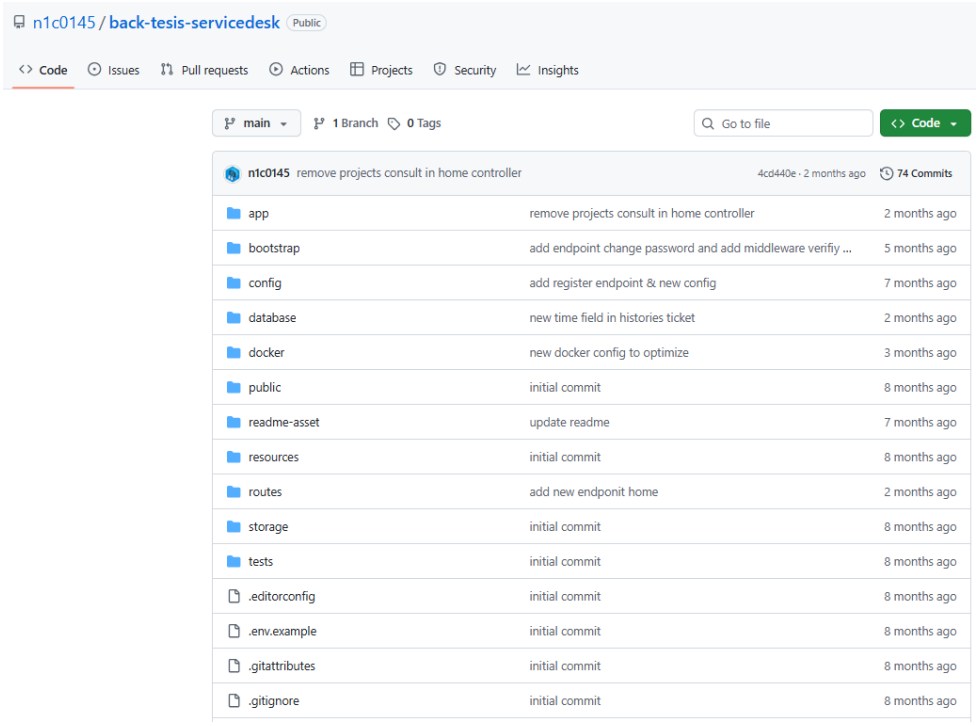


Figura 7

Repositorio Github Backend



Posteriormente se definieron los archivos necesarios para la contenerización del Backend, específicamente el Dockerfile y el archivo "docker-compose.yml". Su objetivo fue garantizar un entorno de ejecución estable y reproducible, independiente del sistema operativo del equipo de desarrollo.

El Dockerfile como se observa en la Figura 8 se diseñó para construir la imagen base del Backend bajo un entorno PHP 8.2 con FPM. Se instalaron dependencias del sistema necesarias para ejecutar Laravel, para gestionar las dependencias del proyecto y conectarse a PostgreSQL asegurando rendimiento y compatibilidad con la base de datos.

Figura 8

Dockerfile

```
FROM php:8.2-fpm

RUN apt-get update && apt-get install -y \
    git zip unzip libpq-dev libonig-dev \
    && docker-php-ext-install pdo_pgsql mbstring opcache pcntl

RUN docker-php-ext-enable opcache

COPY ./docker/opcache.ini /usr/local/etc/php/conf.d/opcache.ini

COPY --from=composer:2 /usr/bin/composer /usr/bin/composer

WORKDIR /var/www/html

COPY composer.json composer.lock ./
RUN composer install --no-dev --no-scripts --no-autoloader
COPY . .

RUN composer dump-autoload --optimize \
    && chown -R www-data:www-data storage bootstrap/cache \
    && chmod -R 775 storage bootstrap/cache
```

El archivo "docker-compose.yml" como se observa en la Figura 9 se empleó para orquestar los servicios que componen el Backend, facilitando su puesta en marcha mediante un solo comando donde se incluye la configuración para construir la imagen usando el Dockerfile previamente descrito. Se asignaron variables de entorno necesarias para enlazar la aplicación con la base de datos, Mailhog (Yung, 2019) y otros componentes internos.

Figura 9

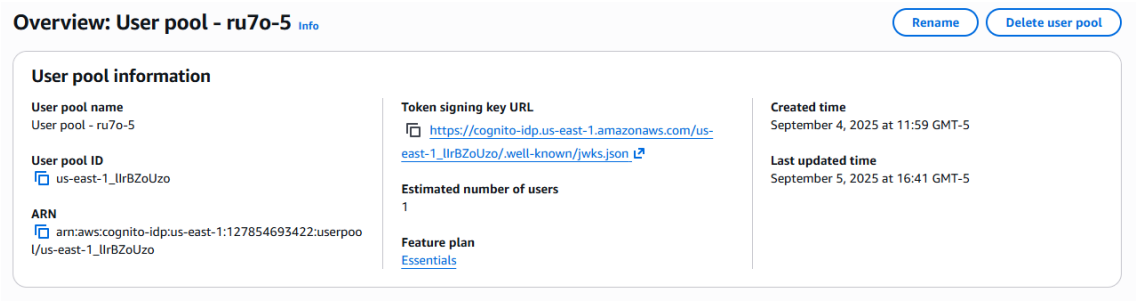
Docker Compose

```yaml
services:
  app:
    build:
      context: .
      dockerfile: docker/Dockerfile
    container_name: back_servicedesk
    restart: always
    ports:
      - "8000:8000"
    environment:
      APP_ENV: local
      APP_DEBUG: false
      DB_CONNECTION: pgsql
      DB_HOST: db
      DB_PORT: 5432
      DB_DATABASE: tesis_servicedesk
      DB_USERNAME: postgres
      DB_PASSWORD: root
      CACHE_DRIVER: array
      SESSION_DRIVER: file
      VIEW_COMPILED_PATH: /var/www/html/storage/framework/views
      MAIL_MAILER: smtp
      MAIL_HOST: mailhog
      MAIL_PORT: 1025
      MAIL_USERNAME: null
      MAIL_PASSWORD: null
      MAIL_ENCRYPTION: null
      MAIL_FROM_ADDRESS: no-reply@educativo.test
      MAIL_FROM_NAME: "Servicio Desk Educativo"
    volumes:
      - ./:/var/www/html
    depends_on:
      - db
      - mailhog
    command: php artisan serve --host=0.0.0.0 --port=8000

  db:
    image: postgres:15
    container_name: bdd_servicedesk
    restart: always
    environment:
      POSTGRES_DB: tesis_servicedesk
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: root
    ports:
      - "5432:5432"
    volumes:
      - postgres_data:/var/lib/postgresql/data

  mailhog:
    image: mailhog/mailhog
    container_name: mailhog_servicedesk
    restart: always
    ports:
      - "1025:1025"
      - "8025:8025"

volumes:
  postgres_data:
```

Una vez finalizada la estructura local, se procedió a configurar los servicios en AWS. En primer lugar, se creó la cuenta y posteriormente se definió el User Pool en Amazon Cognito, componente encargado de manejar la autenticación y administración de usuarios del sistema. La Figura 10 ilustra la configuración del pool de usuarios.

Figura 10

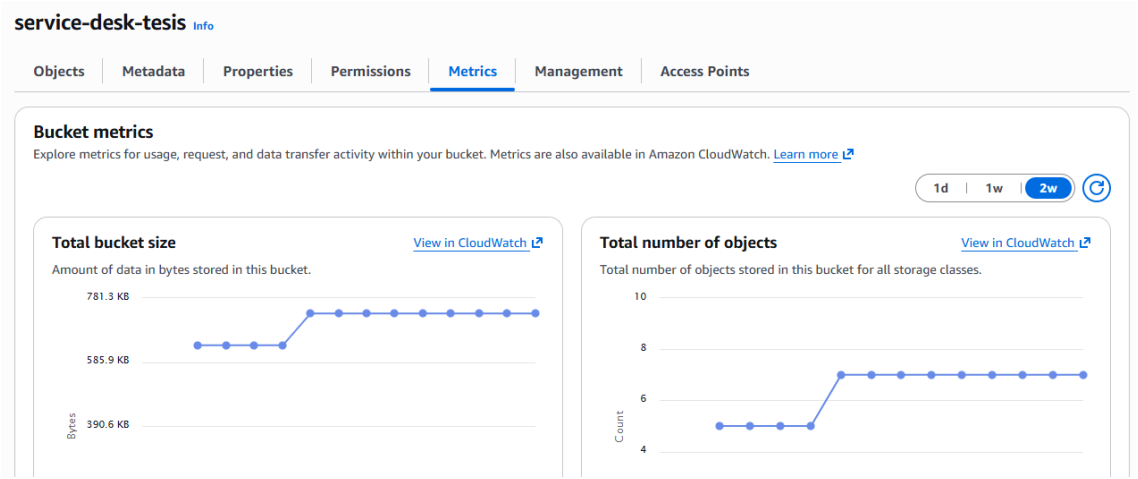
AWS Cognito User pool



Adicionalmente se generó el Bucket en Amazon S3 como se aprecia en la Figura 11, que cumple el rol de almacenamiento de archivos del sistema, como adjuntos, evidencias o documentos. Este repositorio se habilitó con las políticas necesarias para garantizar acceso controlado y comunicación.

Figura 11

AWS S3 Bucket



Con la infraestructura lista, se actualizaron los archivos de ambiente del Backend y del Frontend, incorporando las credenciales, identificadores de AWS Cognito, rutas del Bucket S3, parámetros de conexión a la base de datos y configuraciones internas de la aplicación. Estas

variables permiten mantener la seguridad del proyecto evitando que información sensible quede expuesta en el repositorio.

Como parte del proceso de ordenamiento del proyecto como se observa en la Figura 12 y Figura 13, se redactó un README técnico para cada repositorio. Estos documentos describen la forma de instalación, comandos necesarios para levantar el entorno, dependencias requeridas y consideraciones generales del sistema, proporcionando una guía clara para futuros desarrolladores.

Figura 12

Readme Github Frontend

README

Frontend ServiceDesk

Aplicación web ServiceDesk desarrollada con Angular CLI versión 18.

Requisitos Previos

Tener instalado lo siguiente:

- Node.js (versión >=20.0.0)
- npm (versión ≥9.0.0)

1.Instalación de Angular CLI versión 18

Para instalar Angular CLI versión 18 globalmente en tu equipo ejecutar el siguiente comando `npm install -g @angular/cli@18`

2. Clonar el repositorio

```
git clone https://github.com/n1c0145/front-tesis-servicedesk.git
```

3. Instalación de Dependencias

Dentro del directorio del repositorio ejecutar el comando `npm install`

Librerías de UI utilizadas:

`Angular Material` `Tailwind css`

4. Levantar el Servidor de Desarrollo

Dentro del directorio del repositorio ejecutar el comando `ng serve -o`

Diagrama de Arquitectura FrontEnd

El siguiente diagrama muestra cómo se estructura la arquitectura del frontend Angular:

Figura 13

Readme Github Backend

README

Backend ServiceDesk

Servidor para ServiceDesk, desarrollado con Laravel 11 y conectada a base de datos PostgreSQL.

Requisitos Previos

Tener instalado lo siguiente:

- Docker Desktop
- DBeaver o PgAdmin para gestionar la base de datos.

1. Clonar el repositorio

2. Configurar archivo .env: Se configura la conexión a PostgreSQL y AWS

3. Levantar los contenedores con Docker con el comando `docker compose up -d --build`

4. Instalar dependencias y configurar Laravel:

`docker compose exec app bash` Abre una terminal dentro del contenedor para ejecutar comandos directamente en el entorno.

`composer install` Instala todas las dependencias PHP definidas en composer.json dentro del contenedor.

`php artisan key:generate` Genera la clave de la aplicación Laravel (APP_KEY)

`php artisan migrate` Ejecuta las migraciones para crear las tablas de la base de datos definidas en tu proyecto Laravel.

5. Acceder a la aplicación

- Levantar los contenedores (Laravel + PostgreSQL): `docker compose up -d` Esto inicia el backend y la base de datos en segundo plano. Se ejecuta `php artisan serve` automaticamnte desde el contenedor.
- Verificar que los contenedores estén corriendo: `docker ps`
- Acceder a Laravel en el navegador: Acceder a Laravel en el navegador: `http://localhost:8000`.
- Opcional: ver logs del servidor para monitorear requests y errores: `docker compose logs -f app`

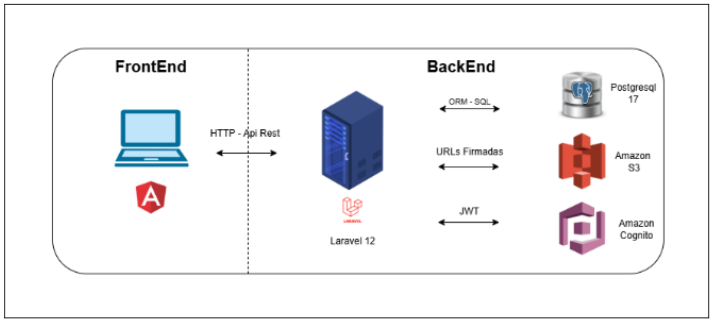
Dependencias Utilizadas:

`aws-sdk SDK` oficial para PHP que permite trabajar con servicios como S3 y Cognito.

`firebase/php-jwt` Librería para decodificar y validar tokens JWT generados por AWS Cognito.

Diagrama de Arquitectura

El siguiente diagrama muestra cómo se estructura la arquitectura del backend de Laravel.



4.1.4 Retrospectiva del sprint

El sprint logró cumplir los objetivos establecidos y permitió construir una base sólida que funcionará como punto de partida para las siguientes fases del proyecto. Como resultado, se dejó preparada una estructura técnica, documental y organizativa que facilita iniciar el desarrollo con una visión clara y bien sustentada. La planificación detallada, el levantamiento del backlog inicial y la configuración del entorno conformaron un fundamento estable para los próximos Sprints, garantizando un proceso ordenado y controlado. Además, la adecuada definición del alcance permitió registrar correctamente tanto los requerimientos funcionales como los no funcionales.

De igual manera, la selección apropiada de tecnologías y servicios externos fortaleció una arquitectura consistente y alineada con las necesidades del sistema. La elaboración de diagramas, historias de usuario, roles y configuraciones técnicas contribuyó a establecer un marco de trabajo sólido. Como aspecto a mejorar, se identificó que algunas podrían requerir más tiempo del previsto. Al tratarse de un sprint fuertemente enfocado en la planificación, es posible que surjan ajustes o modificaciones no contempladas inicialmente debido a cambios en los requerimientos.

4.2 Sprint 2

4.2.1 Introducción al sprint

Nombre

Sprint 2: Autenticación y gestión de usuarios

Duración

3 semanas

Enfoque

El Sprint 2 estuvo orientado al desarrollo del sistema de autenticación y a la gestión de usuarios tanto como el manejo personal de perfil y el manejo de usuario desde el lado del administrador, componentes fundamentales para garantizar la seguridad y el control de acceso dentro de la aplicación. Este sprint tomó como punto de partida la arquitectura y planificación definidas previamente, enfocándose en la implementación de mecanismos que permitan registrar usuarios, autenticar credenciales, recuperar contraseñas y administrar perfiles de forma segura.

Durante este período se trabajó tanto en el Backend como en el Frontend, asegurando una correcta integración entre ambos mediante Endpoints definidos. En el Backend se desarrollaron las estructuras necesarias para el manejo de usuarios y roles, junto con la implementación de autenticación basada en tokens con Cognito y validación de estados. Por su parte en el Frontend se diseñaron las interfaces correspondientes a los procesos de acceso, gestión de cuenta y gestión de usuarios.

Este sprint sentó las bases de seguridad del sistema, habilitando funcionalidades críticas para el uso controlado de la plataforma según los distintos roles.

4.2.2 Objetivos del sprint

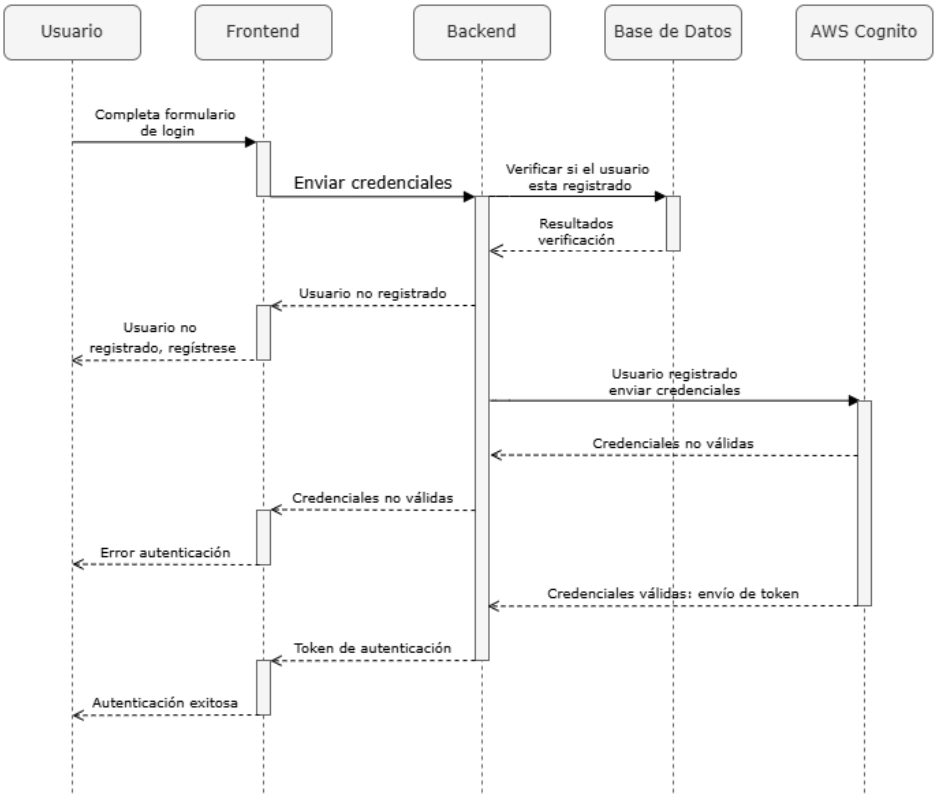
- Implementar el sistema de autenticación de usuarios mediante registro e inicio de sesión seguro con Cognito e integrar autenticación basada en JWT para reforzar la seguridad
- Desarrollar el proceso de recuperación y cambio de contraseña utilizando mecanismos de validación por código por envió de correo con Mailhog.
- Gestionar roles y estados de usuario para controlar el acceso a las funcionalidades del sistema.
- Integrar autenticación basada en JWT y servicios externos de identidad para reforzar la seguridad.
- Desarrollar las interfaces y servicios del Frontend necesarios para la gestión de usuarios, perfil de usuario y la protección de rutas.

4.2.3 Ejecución del sprint

Como parte de la ejecución del Sprint 2, se definieron y documentaron los principales flujos de interacción relacionados con la autenticación. Para ello, se elaboraron diagramas de secuencia que permiten visualizar de forma clara la comunicación entre los distintos componentes.

Figura 14

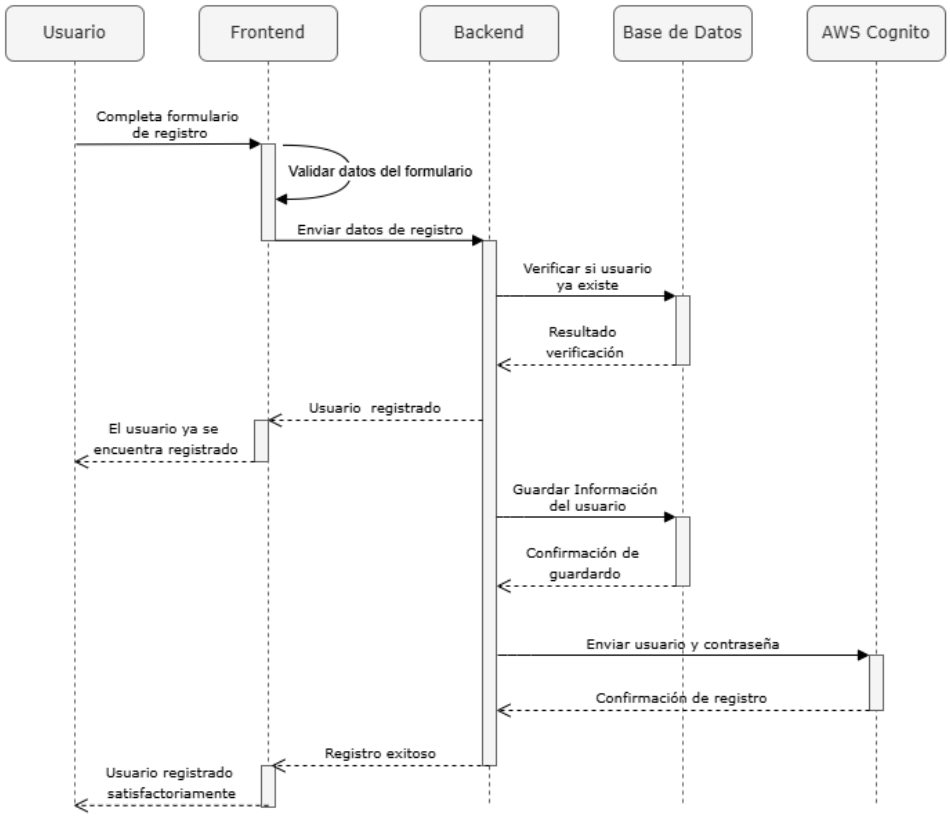
Diagrama de secuencia Autenticación



El diagrama de inicio de sesión detallado en la Figura 14 describe el proceso mediante el cual un usuario intenta acceder al sistema utilizando sus credenciales. El flujo inicia desde la interfaz del Frontend, donde el usuario ingresa sus datos, los cuales son enviados al Backend para una validación preliminar. En esta etapa, el sistema verifica la existencia del usuario en la base de datos local antes de delegar la autenticación al servicio de AWS Cognito. Una vez validadas las credenciales, Cognito retorna un token de acceso que permite establecer una sesión segura. Este enfoque híbrido contribuye a mantener el control interno de los usuarios y facilita la trazabilidad del proceso de autenticación.

Figura 15

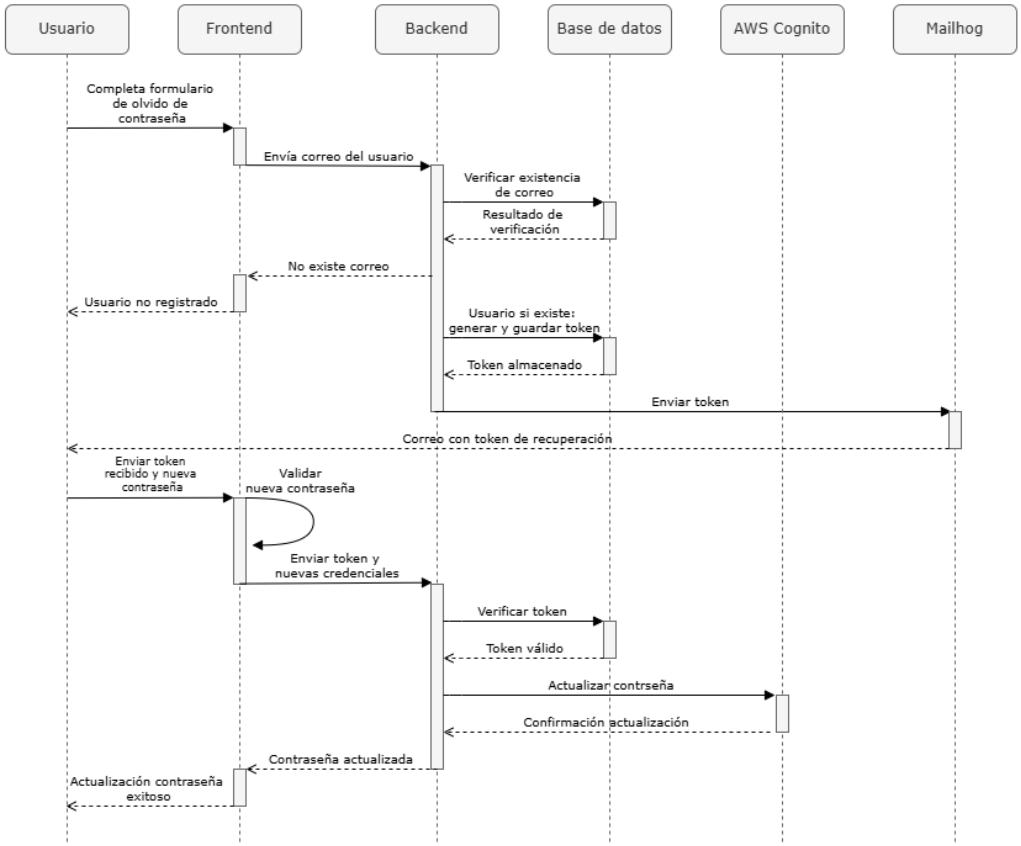
Diagrama de secuencia Registro



El diagrama de registro representado en la Figura 15 describe el procedimiento seguido para la creación de nuevas cuentas dentro del sistema. En este flujo, los datos ingresados por el usuario son validados inicialmente en el Frontend y posteriormente enviados al Backend para comprobar que no exista un registro previo. Una vez confirmada la inexistencia del usuario, la información se almacena en la base de datos y se completa el registro en AWS Cognito. Este proceso garantiza la sincronización entre la gestión local de usuarios y el servicio externo de identidad, asegurando consistencia y confiabilidad en la administración de cuentas.

Figura 16

Diagrama de secuencia Olvido de contraseña



El diagrama de olvido de contraseña que se muestra en la Figura 16, ilustra el mecanismo implementado para permitir a los usuarios restablecer sus credenciales en caso de olvido. El flujo inicia con el envío del correo electrónico al Backend, donde se valida su existencia en la base de datos. Posteriormente, se genera un token de recuperación que es transmitido a través del servicio de autenticación y enviado al usuario. Una vez validado el token, el sistema permite la actualización segura de la contraseña. Este procedimiento combina validaciones locales con servicios externos, fortaleciendo la seguridad y la trazabilidad del proceso.

Para reforzar la seguridad del sistema y proteger los Endpoints expuestos, se implementó un Middleware encargado de validar los tokens de autenticación emitidos por AWS Cognito como se aprecia en la Figura 17. Este componente actúa como una capa intermedia que intercepta las solicitudes entrantes antes de permitir el acceso a los recursos protegidos.

Figura 17

Middleware Verificación Token Cognito

```php
class VerifyCognitoToken
{
    public function handle(Request $request, Closure $next)
    {
        try {
            $token = $this->getBearerToken($request);

            $region = env('AWS_DEFAULT_REGION', 'us-east-1');
            $userPoolId = env('COGNITO_USER_POOL_ID');
            $clientId = env('COGNITO_CLIENT_ID');
            $issuer = "https://cognito-idp.{$region}.amazonaws.com/{$userPoolId}";

            $jwks = json_decode(file_get_contents("{$issuer}/.well-known/jwks.json"), true);
            $keys = JWK::parseKeySet($jwks);
            $decoded = JWT::decode($token, $keys);

            if ($decoded->iss !== $issuer) throw new Exception('Invalid issuer');
            if (isset($decoded->exp) && $decoded->exp < time()) throw new Exception('Token expired');
            if ($clientId && isset($decoded->aud) && $decoded->aud !== $clientId) throw new Exception('Invalid audience');
            if (isset($decoded->token_use) && $decoded->token_use !== 'access') throw new Exception('Invalid token use');

            $request->attributes->add([
                'jwt_payload' => (array) $decoded,
                'cognito_user' => $decoded
            ]);

            return $next($request);
        } catch (Exception $e) {
            return response()->json(['error' => 'Unauthorized', 'message' => $e->getMessage()], 401);
        }
    }
    private function getBearerToken(Request $request): string
    {
        $auth = $request->header('Authorization');
        if (!$auth || !str_starts_with($auth, 'Bearer ')) {
            abort(401, 'Authorization header missing or invalid');
        }
        return trim(substr($auth, 7));
    }
}
```

El middleware desarrollado se encarga de extraer el token desde la cabecera de autorización, verificar su autenticidad utilizando las claves públicas proporcionadas por AWS Cognito y validar aspectos como el emisor, la vigencia, el tipo de token y la audiencia. En caso de que el token no cumpla con las condiciones establecidas, el sistema bloquea el acceso y devuelve una respuesta de error. Este mecanismo asegura que únicamente los usuarios autenticados y con sesiones válidas puedan interactuar con las funcionalidades protegidas del Backend.

Con el fin de centralizar la comunicación con el servicio de autenticación externo, se desarrolló un servicio específico encargado de manejar las operaciones relacionadas con AWS Cognito.

Figura 18

Cognito Service

```php
class CognitoService
{
    protected $client;
    protected $clientId;

    public function __construct()
    {
        $this->client = new CognitoIdentityProviderClient([
            'region' => env('AWS_DEFAULT_REGION'),
            'version' => '2016-04-18',
            'credentials' => [
                'key' => env('AWS_ACCESS_KEY_ID'),
                'secret' => env('AWS_SECRET_ACCESS_KEY'),
            ],
        ]);

        $this->clientId = env('COGNITO_APP_CLIENT_ID');
    }

    public function registerUser($email, $password)
    {
        try {
            return $this->client->signUp([
                'ClientId' => $this->clientId,
                'Username' => $email,
                'Password' => $password,
                'UserAttributes' => [
                    [
                        'Name' => 'email',
                        'Value' => $email,
                    ],
                ],
            ]);
        } catch (AwsException $e) {
            return ['error' => $e->getAwsErrorMessage()];
        }
    }

    public function getClient()
    {
        return $this->client;
    }

    public function loginUser($email, $password)
    {
        try {
            $result = $this->client->initiateAuth([
                'AuthFlow' => 'USER_PASSWORD_AUTH',
                'ClientId' => $this->clientId,
                'AuthParameters' => [
                    'USERNAME' => $email,
                    'PASSWORD' => $password,
                ],
            ]);

            return $result->toArray();
        } catch (\Exception $e) {
            return ['error' => $e->getMessage()];
        }
    }
}
```

Este servicio como se observa en la Figura 18 encapsula las funcionalidades de registro e inicio de sesión de usuarios, permitiendo abstraer la lógica de autenticación y facilitar su reutilización dentro del sistema. Al centralizar estas operaciones, se mejora la mantenibilidad del código y se reduce el acoplamiento entre los controladores y los servicios externos, contribuyendo a una arquitectura más limpia y escalable.

En el Frontend se implementaron mecanismos destinados a controlar el manejo de la sesión del usuario y el acceso a las rutas protegidas de la aplicación.

Figura 19

Interceptor

```
export const authInterceptor: HttpInterceptorFn = (req, next) => {
  const router = inject(Router);

  const token = localStorage.getItem('accessToken');
  const expires = localStorage.getItem('tokenExpiresAt');
  const now = new Date().getTime();

  if (token && expires && now >= Number(expires)) {
    localStorage.clear();
    router.navigate(['/login']);
    return next(req);
  }

  if (token) {
    req = req.clone({
      setHeaders: {
        Authorization: `Bearer ${token}`
      }
    });
  }

  return next(req);
};
```

El interceptor desarrollado tiene como función principal adjuntar automáticamente el token de autenticación a cada solicitud HTTP enviada al Backend, tal como se muestra en la Figura 19. Adicionalmente se encarga de verificar la vigencia del token y, en caso de expiración, limpiar la información de la sesión y redirigir al usuario a la pantalla de inicio de sesión. Este enfoque permite gestionar de forma centralizada la autenticación en las comunicaciones del Frontend

Figura 20

Guard

```
export const authGuard: CanActivateFn = (route, state) => {
  const router = inject(Router);

  const token = localStorage.getItem('accessToken');
  const expires = localStorage.getItem('tokenExpiresAt');
  const roleId = Number(localStorage.getItem('roleId'));

  const now = new Date().getTime();

  if (!token || !expires || now >= Number(expires)) {
    localStorage.clear();
    router.navigate(['/login']);
    return false;
  }

  const allowedRoles = route.data?.['roles'] as number[] | undefined;

  if (allowedRoles && !allowedRoles.includes(roleId)) {
    router.navigate(['/']);
    return false;
  }

  return true;
};
```

Por su parte, el Guard mostrado en la Figura 20 controla el acceso a las vistas protegidas del sistema, validando que el usuario cuente con una sesión activa antes de permitir la navegación. En caso contrario, el sistema restringe el acceso y redirige al usuario, garantizando así que solo usuarios autenticados puedan acceder a funcionalidades sensibles.

Como complemento a los mecanismos de control de acceso implementados en el Frontend, se desarrolló un servicio conforme a la Figura 21 dedicado a la comunicación con el API del Backend. Este componente tiene como finalidad centralizar las solicitudes HTTP y facilitar el consumo de los Endpoints definidos durante el sprint.

Figura 21

API Service

```
export class ApiService {


  private baseUrl = 'http://127.0.0.1:8000/api';

  constructor(private http: HttpClient) { }

  get<T>(endpoint: string): Observable<T> {
    return this.http.get<T>(`${this.baseUrl}/${endpoint}`);
  }

  post<T>(endpoint: string, body: any): Observable<T> {
    return this.http.post<T>(`${this.baseUrl}/${endpoint}`, body);
  }

  patch<T>(endpoint: string, body: any): Observable<T> {
    return this.http.patch<T>(`${this.baseUrl}/${endpoint}`, body);
  }
}
```

Como parte del desarrollo del Frontend, se crearon las interfaces correspondientes a los procesos de inicio de sesión, registro y recuperación de contraseña, priorizando la usabilidad y la validación de datos ingresados por el usuario.

Una vez autenticado, el usuario puede acceder a su perfil personal, desde donde es posible visualizar y actualizar su información, así como realizar el cambio de contraseña. Adicionalmente, se implementó una interfaz específica para el rol de administrador, desde la cual es posible visualizar la lista de usuarios registrados, editar información, cambiar de rol y bloquear cuentas. Este módulo permite una gestión eficiente de los usuarios y refuerza el control de acceso dentro de la plataforma, cumpliendo con los objetivos de seguridad y administración definidos para el Sprint.

4.2.4 Retrospectiva del sprint

El Sprint 2 permitió cumplir de manera satisfactoria los objetivos planteados, logrando implementar un sistema de autenticación funcional y una gestión de usuarios alineada con los requerimientos del proyecto. Además de la implementación de un método de recuperación de contraseña de forma segura. Como resultado, se incorporaron mecanismos de seguridad que garantizan el control de acceso, la protección de rutas y la validación de roles.

La integración entre el Frontend y el Backend se realizó de forma progresiva, permitiendo validar flujos. El uso de autenticación basada en token y middleware contribuyó a fortalecer la seguridad del sistema y a mantener una arquitectura ordenada. Asimismo, el desarrollo de componentes reutilizables y servicios centralizados mejoró la mantenibilidad del código.

Como aspectos a mejorar, se identificó que la integración de servicios externos de autenticación requirió mayor tiempo del inicialmente estimado y tuvo mayor complejidad de lo que se estaba estimado, lo que evidenció la necesidad de ajustar la planificación de tareas técnicas complejas. Además, surgieron pequeños cambios en los flujos de usuario que fueron ajustados durante el Sprint. Estas observaciones servirán como referencia para optimizar la estimación y priorización de actividades en los próximos Sprints.

4.3 Sprint 3

4.3.1 Introducción al sprint

Nombre

Sprint 3: Módulo de proyectos, tickets y notificaciones

Duración

3 semanas

Enfoque

El Sprint 3 estuvo enfocado en la implementación del núcleo funcional del sistema, centrado en la gestión de proyectos, la administración de tickets y la comunicación mediante notificaciones. Este sprint representó un avance significativo en la plataforma, ya que permitió materializar el flujo principal de interacción entre usuarios, proyectos y tickets consolidando la lógica operativa del sistema.

Se desarrollaron múltiples componentes tanto en el Backend como en el Frontend, garantizando una integración fluida entre ambos. En el Backend se diseñaron y crearon las estructuras de base de datos. Asimismo, se implementaron Endpoints que permitieron la creación, consulta, actualización y gestión del ciclo de vida de los tickets, incluyendo su asignación, cambio de estado y cierre.

Se incorporaron mecanismos de notificación en tiempo real y asincrónica, con soporte para envío de correos electrónicos y control de notificaciones leídas y no leídas. También se integró la funcionalidad de carga de archivos en los hilos de los tickets mediante almacenamiento en servicios externos, lo que permitió enriquecer la comunicación entre usuarios.

Desde el Frontend, se desarrollaron interfaces para la creación, visualización y administración de tickets, así como bandejas de gestión y filtros avanzados. Este sprint sentó las bases operativas del sistema.

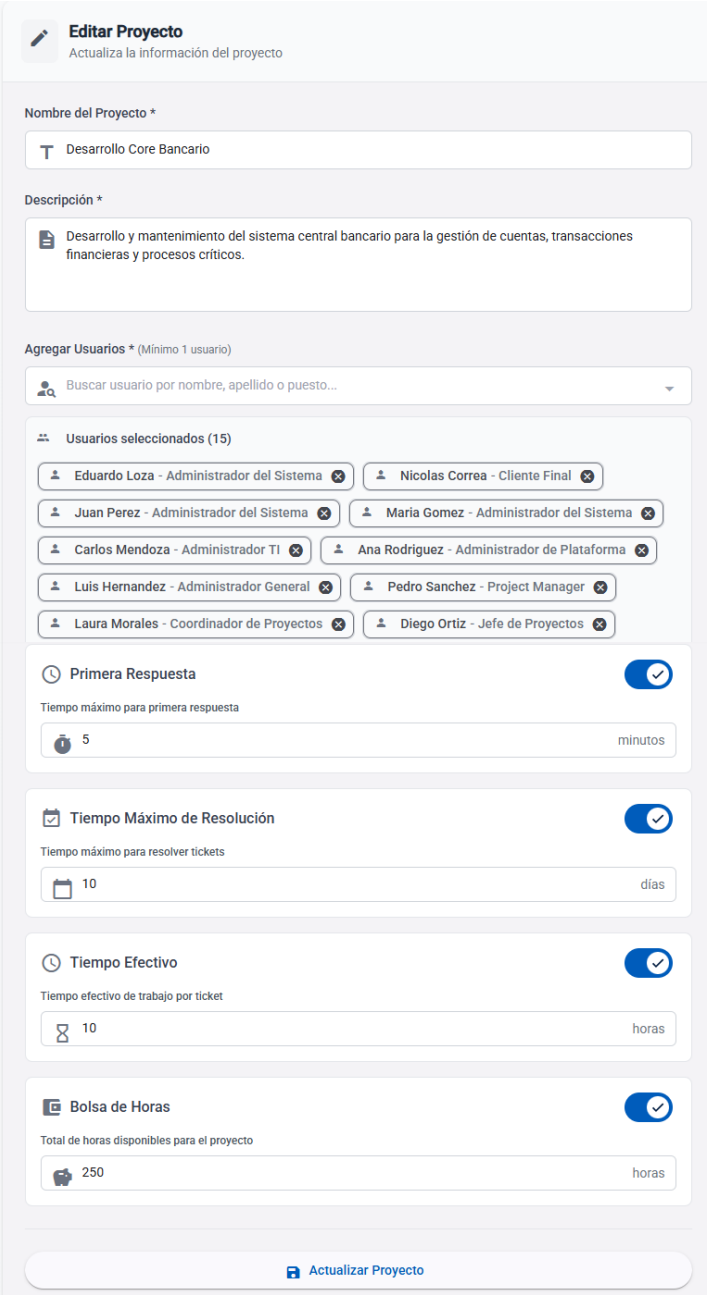
4.3.2 Objetivos del sprint

- Implementar el módulo de proyectos y su relación con los usuarios, permitiendo la asignación y administración de accesos.
- Desarrollar el sistema de tickets con soporte para hilos, prioridades y estados.
- Crear un sistema de notificaciones que informe a los usuarios sobre eventos relevantes dentro de la plataforma.
- Integrar la carga y gestión de archivos adjuntos en los hilos de los tickets utilizando almacenamiento externo.
- Desarrollar las interfaces del Frontend necesarias para la gestión completa de tickets, proyectos y notificaciones.
- Configurar el envío de correos electrónicos automáticos asociados a eventos críticos del sistema.

4.3.3 Ejecución del sprint

Durante la ejecución del Sprint 3 se llevó a cabo el desarrollo de los módulos de proyectos y tickets, los cuales constituyen los pilares fundamentales del sistema. El módulo de proyectos fue diseñado con el objetivo de organizar y agrupar los tickets de forma estructurada, permitiendo una mejor administración del flujo de trabajo y una clara delimitación del alcance de cada proyecto. Cada uno funciona como un contenedor lógico que agrupa tickets, facilitando la asignación de usuarios, la definición de permisos y el control de accesos según el rol. Asimismo, los proyectos permiten establecer reglas específicas que posteriormente son utilizadas para la generación de métricas, reportes y análisis de desempeño del sistema. Dentro de cada proyecto se definen los SLA, lo que permite adaptar los tiempos de respuesta y resolución de acuerdo con las necesidades particulares de cada contexto operativo. La correcta relación entre proyectos, usuarios y tickets también facilita la trazabilidad de la información y la auditoría de las acciones realizadas. Como se observa en la Figura 22 se puede observar la interfaz que permite la visualización de proyectos, así como la asignación de usuarios y la configuración de sus parámetros principales mencionados.

Figura 22
Vista Proyectos



La definición de prioridades para los tickets fue otro aspecto clave abordado durante el Sprint. La prioridad fue incorporada como un atributo fundamental para clasificar los tickets según su impacto y urgencia, facilitando la toma de decisiones y la correcta asignación de recursos. Como se observa en la Tabla 18 se optó por definir cuatro niveles de prioridad debido a su

simplicidad y efectividad para cubrir la mayoría de escenarios operativos, evitando una clasificación excesivamente compleja que pudiera dificultar la gestión

Tabla 18

Prioridades de tickets

Prioridades	Descripción
Alta	Tickets que representan un impacto crítico sobre la operación del sistema o de los servicios principales. Este tipo de solicitudes requiere atención inmediata.
Media	Tickets que generan un impacto moderado y deben ser atendidos en un tiempo razonable.
Baja	Tickets de bajo impacto operativo y pueden ser atendidos de manera programada sin generar riesgos significativos.
Sin Asignar	Tickets que han sido registrados recientemente y aún se encuentran en etapa de análisis inicial

Otro elemento esencial fue la gestión de los estados de los tickets. La distribución de los tickets por estados permite representar de manera clara y ordenada el ciclo de vida completo de cada solicitud, desde su creación hasta su cierre definitivo. Como se muestra en la Tabla 19 se definió un conjunto de estados específicos con el objetivo de mantener un flujo lógico, comprensible y alineado con las mejores prácticas de gestión de servicios.

Tabla 19

Estados de tickets

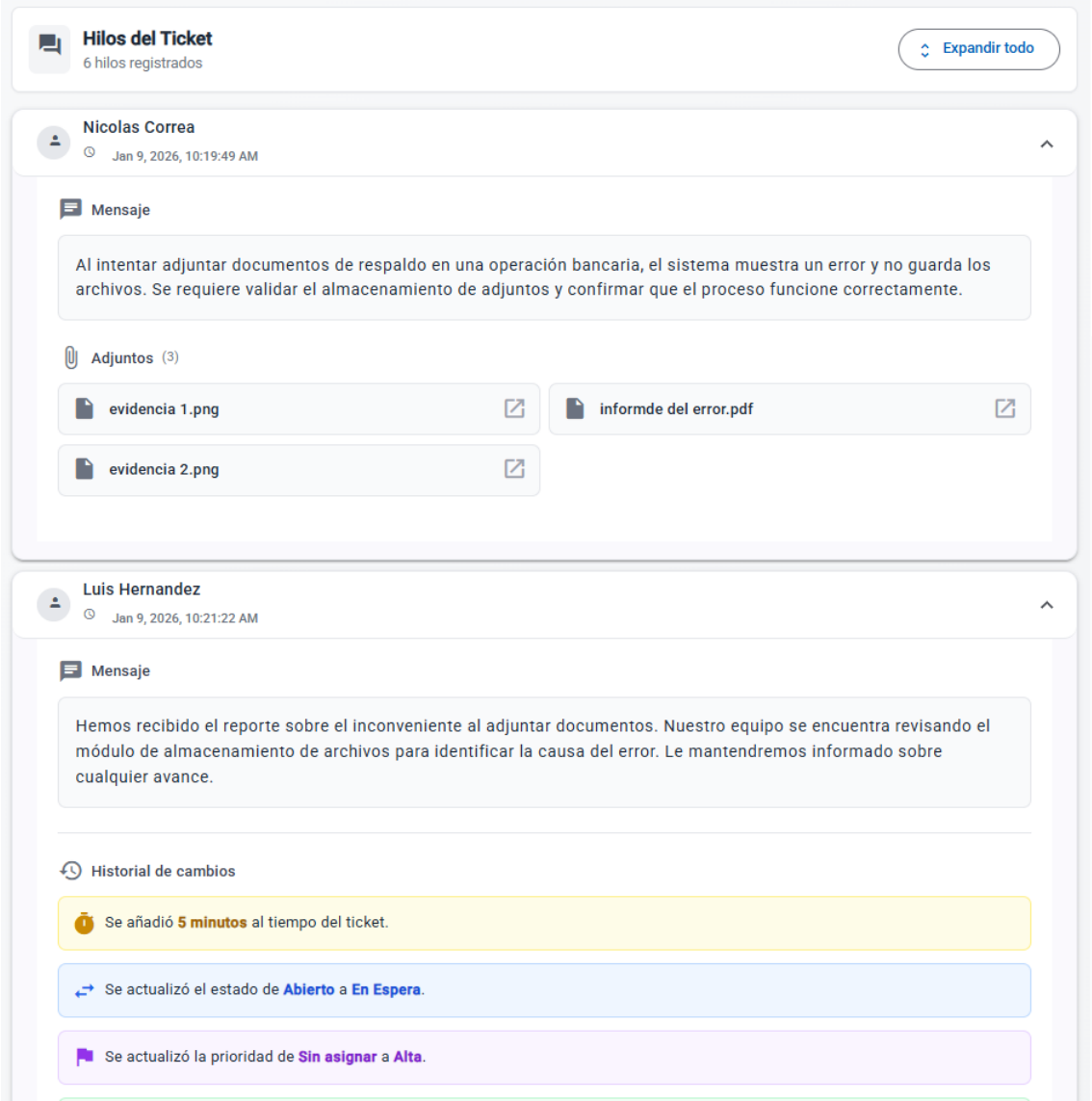
Estados	Descripción
Abierto	El ticket ha sido registrado en el sistema y se encuentra disponible para su revisión inicial. En esta etapa aún no ha sido atendido por el equipo de soporte y está pendiente de evaluación.
Primera respuesta	Representa el momento en el que el equipo de soporte emite la primera comunicación hacia el usuario.
Se necesita más información	Utilizado cuando los datos entregados por el usuario no son suficientes para continuar con la atención del ticket
En progreso	El ticket está siendo trabajado activamente por el personal de soporte
En espera	El ticket se encuentra temporalmente detenido debido a dependencias externas, como la respuesta del usuario o la intervención de terceros.
Resuelto	Estado que confirma que se ha implementado una solución a la incidencia.
Cerrado	Estado final del ticket, en el cual se confirma que la solución fue aceptada y el proceso de atención ha concluido de manera satisfactoria.

La implementación de hilos de conversación dentro de los tickets representó una mejora significativa en la gestión y trazabilidad de la comunicación. Cada ticket incorpora hilos que permiten registrar de manera cronológica todas las interacciones entre usuarios y agentes de soporte, incluyendo respuestas, solicitudes de aclaración y actualizaciones relevantes. Tal como

se aprecia en la Figura 23, este enfoque asegura que toda la información asociada a un ticket se mantenga centralizada en un único espacio, evitando la dispersión de datos y la pérdida de contexto. Los hilos facilitan un control más preciso del historial del ticket, permitiendo la revisión de acciones previas y la correcta identificación de responsabilidades a lo largo del proceso. Adicionalmente, esta funcionalidad incrementa la transparencia del flujo de atención, ya que tanto el usuario como el equipo de soporte pueden visualizar el progreso del ticket y las decisiones adoptadas en cada una de sus etapas, favoreciendo una gestión más clara y ordenada.

Figura 23

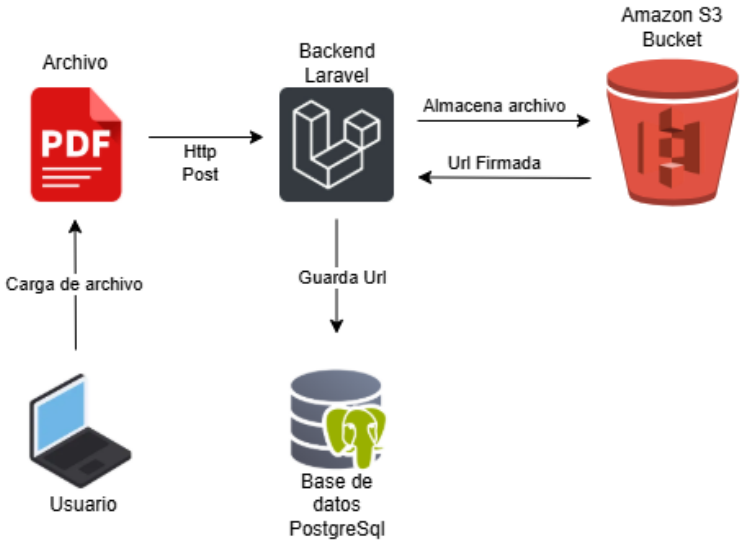
Hilos de Tickets



Adicional se integró la funcionalidad de carga y gestión de archivos adjuntos en los hilos de los tickets, utilizando almacenamiento externo en servicios S3. Esta decisión se tomó con el objetivo de garantizar niveles de seguridad, disponibilidad y trazabilidad de la información. El flujo de carga de archivos inicia cuando el usuario adjunta un archivo desde la interfaz, lo que genera una solicitud al Backend para su procesamiento. Posteriormente, el sistema almacena el archivo en un Bucket de S3, asegurando que los datos no se guarden directamente en el servidor de la aplicación. Como se observa en la Figura 24 en el diagrama de arquitectura se puede observar claramente este flujo, desde la carga del archivo hasta su almacenamiento y posterior recuperación de las Urls firmadas.

Figura 24

Diagrama de S3 Bucket



El módulo de notificaciones, cuyo objetivo principal es mantener informados a los usuarios sobre eventos relevantes dentro del sistema de forma oportuna y eficiente. Este módulo fue diseñado para operar mediante dos canales principales: notificaciones internas visibles en la plataforma web del Service Desk y notificaciones enviadas por correo electrónico. En el caso de la aplicación web, las notificaciones son consultadas periódicamente mediante un mecanismo de Polling, lo que permite que el usuario reciba alertas actualizadas sin necesidad de recargar manualmente la interfaz.

Por otro lado, se integró el envío de correos electrónicos automáticos para eventos críticos como se observa en la Figura 25, garantizando que la información llegue al usuario incluso cuando no se encuentra conectado a la plataforma. Entre los eventos que generan notificaciones se incluyen aquellos casos en los que un ticket está asociado a un acuerdo de nivel de servicio. En el archivo de configuración de notificaciones como se muestra en la Figura 26, se puede observar la implementación de los métodos encargados de almacenar la notificación en la base de datos y de generar el mensaje de correo electrónico correspondiente. Cabe recalcar que se está utilizando el servidor de pruebas Mailhog para el servicio en desarrollo.

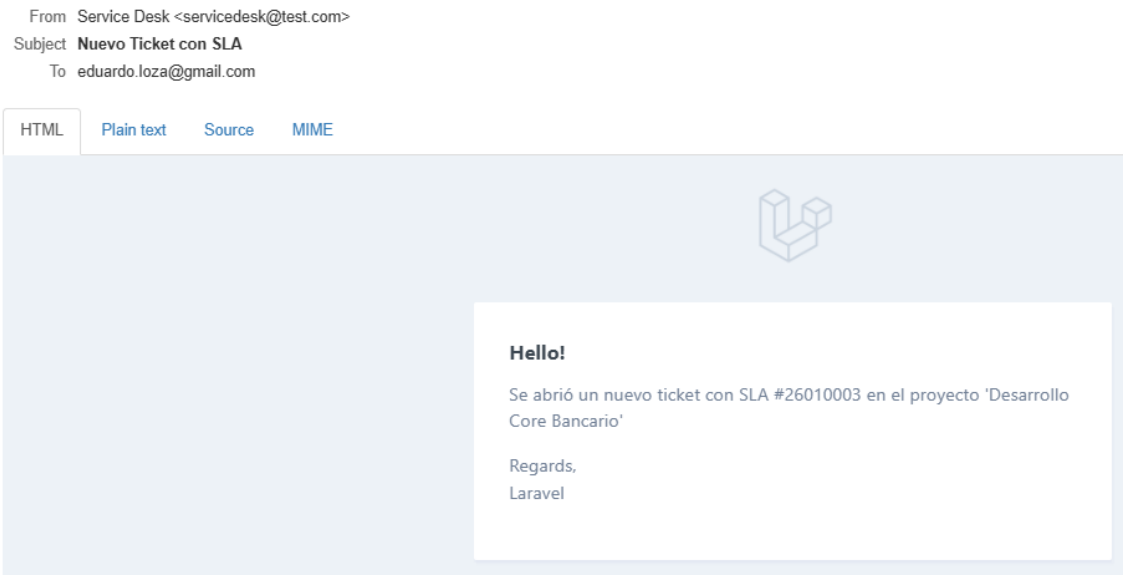
Figura 25

Configuración de notificaciones

```php
public function toDatabase($notifiable): array
{
    return [
        'title' => 'Nuevo ticket SLA',
        'message' => "Se abrió un nuevo ticket con SLA #{$this->ticketNumber} en el proyecto '{$this->projectName}'.",
    ];
}

public function toMail($notifiable)
{
    return (new MailMessage)
        ->subject('Nuevo Ticket con SLA')
        ->line("Se abrió un nuevo ticket con SLA #{$this->ticketNumber} en el proyecto '{$this->projectName}'");
}
```

Figura 26

Correos electrónicos enviados



Este enfoque mixto permite mejorar la comunicación entre el sistema y los usuarios, reducir el riesgo de omitir eventos importantes y reforzar el cumplimiento de los acuerdos de nivel de servicio definidos.

4.3.4 Retrospectiva del sprint

El Sprint 3 permitió consolidar de manera efectiva el flujo principal del sistema, logrando la implementación de un módulo robusto de gestión de proyectos y tickets que cumple con los requerimientos funcionales definidos. Se consiguió establecer una estructura sólida tanto a nivel de base de datos como de lógica de negocio, lo que facilitó la creación, seguimiento y cambio de estado de tickets, así como su correcta asociación con proyectos y usuarios.

Uno de los principales logros de este sprint fue la correcta integración entre Backend y Frontend, lo que permitió validar de forma temprana los flujos de creación y gestión de tickets. La implementación de hilos de conversación y la posibilidad de adjuntar archivos que son almacenados en los Bucket de s3 aportaron un valor significativo al sistema, mejorando la comunicación entre los distintos componentes involucrados.

Durante el desarrollo se identificaron ciertos desafíos técnicos, especialmente en la gestión de estados de los tickets y la sincronización de notificaciones. La definición de reglas claras para los cambios de estado y prioridades requirió ajustes durante el sprint, lo que implicó cambios en la lógica inicialmente implementada. De igual forma, la integración del servicio de almacenamiento de archivos demandó mayor tiempo de configuración y pruebas para garantizar la seguridad y estabilidad del sistema.

Otro aspecto relevante fue la necesidad de optimizar algunos Endpoints para mejorar el rendimiento en consultas complejas, como la obtención de listas de tickets por los distintos filtros.

4.4 Sprint 4

4.4.1 Introducción al sprint

Nombre

Sprint 4: Estilos, maquetación y base de conocimientos

Duración

3 semanas

Enfoque

El Sprint 4 estuvo orientado a mejorar significativamente la experiencia de usuario mediante la aplicación de estilos visuales consistentes, el diseño responsivo de las vistas y el desarrollo

inicial del módulo de base de conocimientos. Este sprint tuvo como objetivo principal elevar la calidad visual y de usabilidad de la plataforma, asegurando una interfaz más intuitiva y coherente.

En el Frontend se trabajó en la implementación de estilos globales utilizando librerías de diseño modernas lo que permitió unificar la apariencia de la aplicación. Se incorporaron principios de diseño responsivo para garantizar una correcta visualización en distintos dispositivos. Se desarrollaron componentes reutilizables como alertas, diálogos y Spinners con el fin de mejorar la consistencia visual y reducir la duplicación de código.

Se inició el desarrollo del módulo de base de conocimientos, el cual tiene como finalidad sugerir tickets pasados para evitar la apertura de tickets redundantes con soluciones previas y reducir la generación de incidencias repetitivas.

4.4.2 Objetivos del sprint

- Mejorar la experiencia visual de la aplicación mediante estilos globales consistentes.
- Implementar diseño responsivo para garantizar accesibilidad en distintos dispositivos.
- Desarrollar componentes reutilizables que optimicen la mantenibilidad del Frontend.
- Diseñar y estructurar el módulo de base de conocimientos.
- Integrar la lógica para la detección de tickets similares desde el Backend.

4.4.3 Ejecución del sprint

Durante el Sprint se llevó a cabo un trabajo enfocado principalmente en la maquetación de la interfaz, la consolidación de una identidad visual uniforme y la mejora integral de la experiencia de usuario. Uno de los primeros avances fue la aplicación de estilos globales que permitieron establecer una consistencia visual en todas las vistas del sistema. Esto implicó la definición de colores y tamaños de elementos, lo cual contribuyó a que la plataforma presentara una apariencia más profesional y coherente.

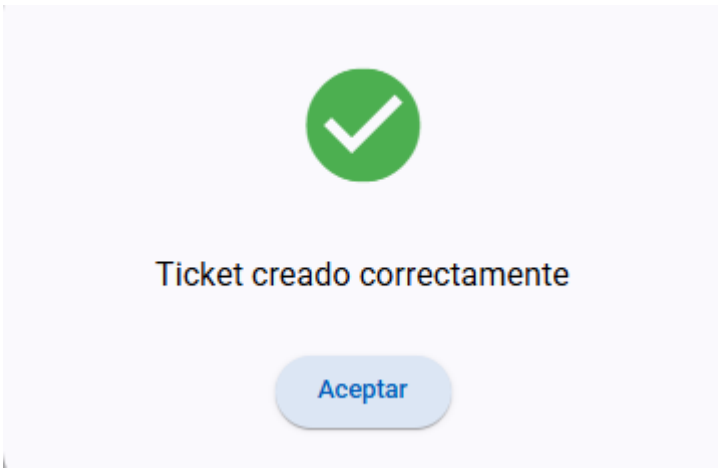
La maquetación realizada contempló principios de diseño responsivo, asegurando que las pantallas se adapten correctamente a distintos tamaños de dispositivos sin perder funcionalidad ni legibilidad. Gracias a este enfoque, fue posible identificar y corregir desalineaciones, problemas de espaciado y variaciones visuales anteriormente no evidentes. Adicional, se trabajó en la reutilización de componentes visuales, lo que redujo la duplicación de código y facilitó la mantenibilidad del Frontend, permitiendo que futuras mejoras puedan implementarse de forma más ágil.

Desde el punto de vista de la experiencia de usuario, estos cambios permitieron una navegación más intuitiva, con interfaces claras y predecibles. El usuario puede identificar fácilmente acciones, estados del sistema lo que mejora la interacción general con la plataforma. Todas las pantallas desarrolladas del proyecto pueden ser revisadas en el Anexo D, donde se presentan las capturas correspondientes a cada vista del sistema.

Como parte del fortalecimiento de la interfaz, se desarrolló un componente de alertas reutilizable que puede ser utilizado en todo el proyecto. Este componente permite mostrar distintos tipos de mensajes según el contexto de la acción realizada por el usuario como alertas de éxito, error e información. Cada tipo de alerta cuenta con una configuración específica que facilita su identificación. El componente admite la visualización de uno o dos botones, dependiendo del escenario, permitiendo acciones como aceptar o cancelar una operación. Gracias a esta estandarización, se logró unificar la forma en la que el sistema comunica eventos importantes mejorando la claridad e interpretación de acciones. Un ejemplo del funcionamiento de este componente puede observarse en la Figura 27, donde se muestra un ejemplo de una alerta desplegada en pantalla.

Figura 27

Ejemplo de Alerta



Otro elemento desarrollado durante el sprint fue el Spinner de carga, diseñado como un componente reutilizable para todo el proyecto. Este componente se utiliza como indicador visual durante procesos que requieren tiempo de espera, como consultas al Backend o carga de información. Su implementación permite informar al usuario que el sistema se encuentra

procesando una acción, evitando la percepción de bloqueos o fallos. El uso consistente de este componente en las distintas vistas contribuye a una experiencia de usuario más fluida y comprensible. La representación visual de este Spinner puede apreciarse en la Figura 28, donde se muestra su uso durante un proceso de carga.

Figura 28

Spinner

Durante el Sprint también se implementaron los componentes de Header y menú de navegación como elementos incrustados y reutilizables en todas las vistas del sistema cuando el usuario se encuentra autenticado. El Header actúa como un espacio permanente que integra el nombre del Service Desk, un área destinada a notificaciones y accesos rápidos relevantes para el usuario como acceso a perfil y el cierre de sesión.

El menú de navegación presenta las distintas opciones disponibles según el rol del usuario, garantizando que cada perfil visualice únicamente las funcionalidades que le corresponden. Esta lógica mejora la seguridad y la usabilidad, evitando accesos innecesarios. Las capturas correspondientes a estos componentes pueden observarse en las Figuras 29 y 30, donde se muestran ejemplos del Header y del menú respectivamente.

Figura 29

Header

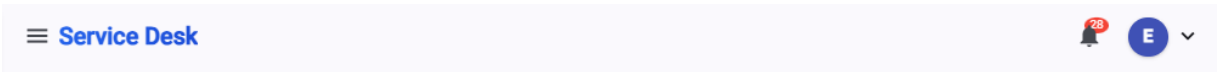
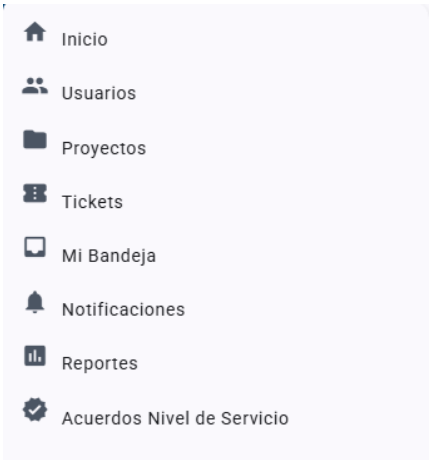


Figura 30

Menú de opciones



Posteriormente, se desarrolló el módulo de base de conocimientos, cuyo objetivo es permitir la consulta de tickets cerrados que han sido previamente resueltos. Estos tickets pasan a almacenarse en una nueva tabla destinada a la búsqueda de incidencias pasadas, lo que posibilita que los usuarios revisen soluciones existentes antes de generar nuevos tickets.

Desde el Backend se implementó un método de búsqueda que analiza el título y la descripción ingresados por el usuario, extrayendo palabras clave relevantes y comparándolas con los registros almacenados en la base de conocimientos. A partir de estas coincidencias se calcula un puntaje de similitud que considera tanto coincidencias exactas como aproximadas, utilizando un sistema de ranking que prioriza los resultados más relevantes. Solo los tickets que superan un mínimo de similitud son retornados, limitando además la cantidad de resultados para facilitar la revisión. Un ejemplo de este módulo y sus resultados puede observarse en la Figura 31, donde se muestra el método del controlador de búsqueda de la base de conocimientos.

Figura 31

Método tickets similares

```php
public function similarTickets(Request $request)
{
    $data = $request->validate([
        'titulo' => 'required|string',
        'descripcion' => 'nullable|string',
    ]);
    $texto = strtolower($data['titulo'] . ' ' . ($data['descripcion'] ?? ''));

    $palabras = collect(preg_split('/\s+/', $texto))
        ->filter(fn($p) => strlen($p) > 3)
        ->unique()
        ->values();

    if ($palabras->count() == 0) {
        return response()->json([
            'success' => true,
            'matches' => [],
        ]);
    }

    $query = KnowledgeBase::query();
    $query->select('*');

    $query->where(function ($q) use ($palabras) {
        foreach ($palabras as $p) {
            $q->orWhere('titulo', 'LIKE', "%$p%")
                ->orWhere('descripcion', 'LIKE', "%$p%");
        }
    });

    $candidatos = $query->get();
    $resultados = $candidatos->map(function ($item) use ($palabras) {

        $textoKB = strtolower($item->titulo . ' ' . $item->descripcion);
        $score = 0;
        foreach ($palabras as $p) {
            if (str_contains($textoKB, $p)) {
                $score += 2;
                continue;
            }
            $distancia = levenshtein($p, $textoKB);
            if ($distancia <= 3) {
                $score += 1;
            }
        }
        $item->similarity_score = $score;
        return $item;
    })
        ->filter(fn($x) => $x->similarity_score >= 3)
        ->sortByDesc('similarity_score')
        ->take(3)
        ->values();

    return response()->json([
        'success' => true,
        'matches' => $resultados,
    ]);
}
```

4.4.4 Retrospectiva del sprint

El Sprint 4 permitió avanzar de manera significativa en la mejora de la experiencia de usuario, logrando una interfaz más clara, ordenada y visualmente coherente. Uno de los principales beneficios de este sprint fue la creación de componentes reutilizables, lo que permitió estandarizar comportamientos visuales y reducir la complejidad del Frontend. Esto facilitó el desarrollo de nuevas vistas y mejoró la mantenibilidad del código a largo plazo. La maquetación de vistas responsivas permitió detectar y corregir inconsistencias visuales que no habían sido visibles en etapas anteriores.

La integración entre el módulo de tickets y la base de conocimientos presentó ciertos retos, especialmente en la identificación de tickets similares. Fue necesario ajustar la lógica del Backend para mejorar la relevancia de los resultados, lo que implicó pruebas adicionales y refinamiento de criterios.

Se detectó que la carga visual inicial de la aplicación aumentó ligeramente debido a la incorporación de nuevas librerías de estilos, lo que evidenció la necesidad de optimizar recursos y cargas. En general se cumplió con los objetivos y se dejó el software preparado para la incorporación de métricas y análisis en el siguiente sprint.

4.5 Sprint 5

4.5.1 Introducción al sprint

Nombre

Sprint 5: Módulo de estadísticas

Duración

3 semanas

Enfoque

El Sprint 5 estuvo enfocado en la implementación del módulo de estadísticas y visualización de métricas con el objetivo de proporcionar información clave para la toma de decisiones y el conteo de bolsa de horas Este Sprint permitió transformar los datos generados por los tickets y proyectos en indicadores comprensibles y útiles para los usuarios y administradores.

En el Backend se desarrollaron Endpoints específicos para la generación de reportes generales, métricas de SLA y datos resumidos para el panel principal. Se incorporaron filtros por fecha y proyecto, lo que permitió una mayor flexibilidad en el análisis de la información.

En el Frontend se diseñaron Dashboards interactivos que permitieron visualizar las métricas de forma clara y dinámica junto con gráficos que aporten valor.

4.5.2 Objetivos del sprint

- Implementar métricas clave para el análisis del desempeño del sistema.
- Desarrollar Endpoints de reportes generales, SLA y panel principal.
- Incorporar filtros dinámicos por fecha y proyecto en los reportes.
- Diseñar Dashboards visuales e interactivos en el Frontend.
- Facilitar la toma de decisiones mediante información clara y estructurada.
- Realizar conteo de bolsa de horas de manera automática.

4.5.3 Ejecución del sprint

Durante el Sprint 5 se desarrolló e integró el módulo de estadísticas, consolidando un componente clave del sistema orientado al análisis y visualización de información estratégica. Este sprint permitió transformar los datos operativos generados por la gestión de tickets y proyectos en Dashboards claros y estructurados facilitando la comprensión del desempeño general del Service Desk y el control de la bolsa de horas.

La centralización de la información permite a los usuarios acceder de forma rápida y ordenada a métricas relevantes sin necesidad de revisar múltiples vistas o reportes independientes. Los Dashboards fueron diseñados para mostrar información resumida, combinando indicadores numéricos y gráficos que aportan una visión general del estado del sistema. Esta centralización reduce el tiempo de análisis y mejora la eficiencia en la supervisión de la operación diaria.

Se incorporaron filtros dinámicos por proyecto y por rango de fechas brindando mayor flexibilidad en el análisis de la información. Estos filtros permiten segmentar los datos de acuerdo con necesidades específicas, como evaluar el desempeño de un proyecto en particular o analizar el comportamiento del sistema en un periodo determinado.

En el módulo de reportes generales, presentado en la Figura 32, se integraron métricas operativas y de tiempo que permiten monitorear el volumen de tickets, su estado, prioridades y antigüedad, así como indicadores relacionados con la eficiencia del equipo de soporte. Asimismo, se implementaron KPIs de desempeño orientados al análisis del esfuerzo operativo. Estos indicadores permiten identificar proyectos o tickets que demandan mayor cantidad de horas.

Un componente fundamental mostrado en la Figura 33 del fue la incorporación de métricas asociadas a los acuerdos de nivel de servicio. Estas métricas permiten evaluar el cumplimiento de los compromisos establecidos con los clientes, considerando tiempos de primera respuesta,

resolución máxima y consumo de la bolsa de horas. El seguimiento de horas utilizadas, horas restantes y porcentaje de utilización proporciona una visión clara del estado contractual de cada proyecto, facilitando el control financiero y operativo.

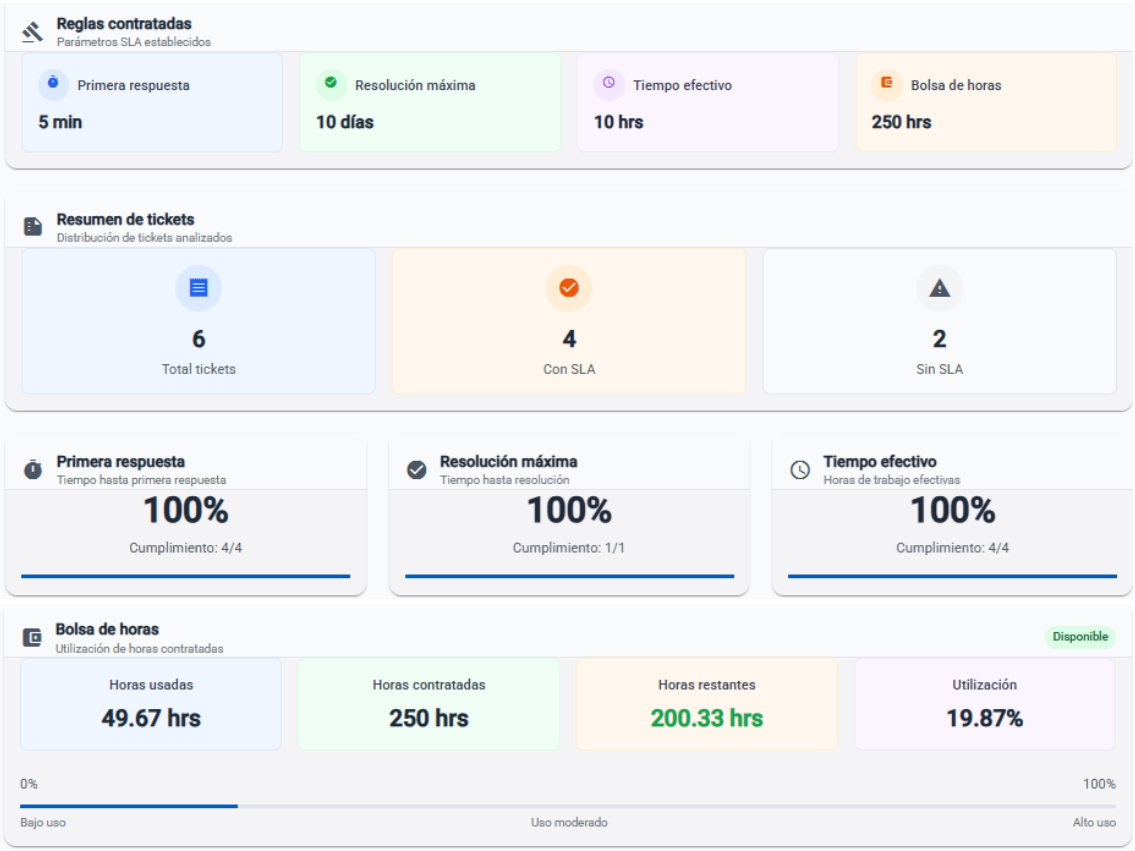
En conjunto, el módulo de Dashboards desarrollado fortalece el seguimiento del rendimiento del sistema, proporcionando información confiable y actualizada para la toma de decisiones.

Figura 32

Reportes generales

Figura 33

Reportes acuerdo de nivel de servicio



4.5.4 Retrospectiva del sprint

El Sprint 5 permitió completar el sistema con un módulo de análisis que aporta un alto valor estratégico a la plataforma. La implementación de métricas y Dashboards facilitó la visualización permitiendo identificar tendencias, áreas de mejora y utilización de recursos como la bolsa de horas de cada proyecto.

Uno de los principales logros fue la correcta definición de indicadores clave, lo que aseguró que los reportes generados fueran relevantes y alineados con los objetivos del proyecto. La incorporación de filtros dinámicos mejoró la flexibilidad del análisis, permitiendo a los usuarios adaptar la información según sus necesidades específicas.

En términos técnicos el manejo de grandes volúmenes de datos evidenció la necesidad de optimizar consultas y mejorar el rendimiento de los gráficos para evitar sobrecarga del Frontend. En conclusión, el Sprint cumplió con los objetivos propuestos y cerró de manera sólida el desarrollo del sistema, integrando funcionalidad operativa, experiencia de usuario y análisis de datos

CAPITULO 5

PRUEBAS Y RESULTADOS

5.1 Resultados de la implementación

En concordancia con el objetivo general del proyecto, orientado al diseño e implementación de un Service Desk web moderno que busca la mejora de la productividad del equipo de soporte, se muestran los resultados obtenidos tras el desarrollo del sistema. Esto permitió materializar los objetivos específicos definidos, integrando los distintos componentes garantizando una gestión eficiente de tickets, trazabilidad de la información y generación de métricas relevantes para la toma de decisiones.

El sistema desarrollado consolida en una única plataforma los procesos clave del área de soporte técnico, permitiendo una administración estructurada de usuarios mediante el servicio de autenticación AWS Cognito, lo que fortalece la seguridad y el control de accesos. Asimismo, se implementó un módulo de gestión de proyectos que facilita la asignación de usuarios, la definición de acuerdos de nivel de servicio y el control del consumo de la bolsa de horas, optimizando la planificación operativa.

La gestión de tickets constituye el eje central del sistema, incorporando estados configurables, niveles de prioridad e hilos de conversación que permiten un seguimiento detallado de cada incidencia. A esto se suma la posibilidad de adjuntar archivos almacenados en Amazon S3, garantizando la organización y trazabilidad documental asociada a cada caso. De igual manera, se integraron mecanismos de alertas y notificaciones tanto a nivel web como por correo electrónico, contribuyendo al cumplimiento oportuno de los SLA establecidos. Complementariamente, el sistema incorpora una base de conocimiento alimentada a partir de soluciones previamente cerradas, la cual permite sugerir respuestas ante incidencias recurrentes, reduciendo la generación de tickets duplicados y evitando la repetición de soluciones ya documentadas.

La integración entre Frontend y Backend permitió una experiencia de usuario fluida y coherente, apoyada por un almacenamiento estructurado en la base de datos que garantiza la persistencia, consistencia y disponibilidad de la información del sistema. Como se observa en la Tabla 20, las funcionalidades planificadas fueron implementadas en su totalidad, cumpliendo con los requerimientos funcionales y técnicos definidos al inicio del proyecto.

Tabla 20

Resultados de la implementación

Funcionalidad	Estado	Observación
Gestión de usuarios	Implementado	Autenticación segura y control de accesos
Módulo de proyectos	Implementado	Permite asignación de usuarios y definición de acuerdos de niveles de servicio,
Gestión de tickets	Implementado	Manejo de estados, prioridades e hilos.
Reportes de acuerdos de nivel de servicio.	Implementado	Visualización de cumplimiento.
Gestión de bolsa de horas	Implementado	Control del consumo por proyecto
Alertas del sistema	Implementado	Avisos automáticos por actualizaciones
Correos web	Implementado	Envío por correo electrónico
Archivos adjuntos	Implementado	Almacenamiento seguro y trazable
Integración Frontend–Backend	Implementado	Comunicación eficiente con Endpoints.
Almacenamiento en base de datos	Implementado	Persistencia de información operativa e histórica
Métricas de rendimiento	Implementado	Generación de indicadores clave

5.2 Pruebas funcionales

Con el fin de verificar el correcto funcionamiento del sistema desarrollado y validar el cumplimiento de los requerimientos funcionales definidos, se ejecutaron pruebas funcionales

orientadas a evaluar los principales procesos. Estas pruebas permitieron comprobar que las funcionalidades implementadas responden de manera adecuada a los distintos escenarios de uso.

Las pruebas se realizaron en un entorno controlado simulando el comportamiento real de los usuarios del sistema y considerando distintos perfiles de acceso. Los resultados obtenidos permitieron confirmar que las operaciones se ejecutan conforme a lo esperado. Como se muestra en la Tabla 21, todos los casos de prueba funcionales definidos fueron ejecutados satisfactoriamente evidenciando que el sistema cumple con los requerimientos funcionales establecidos.

Tabla 21

Casos de prueba de prueba funcionales del sistema

Código	Caso de Prueba	Resultado esperado	Resultado obtenido
PF-1	Inicio de sesión de usuario	El sistema permite el acceso mediante credenciales válidas	El acceso se realiza correctamente
PF-2	Cierre de sesión	La sesión del usuario finaliza de forma segura	La sesión se cierra sin inconvenientes
PF-3	Registro de nuevos usuarios	El sistema registra usuarios con los datos ingresados	El usuario es creado correctamente
PF-4	Recuperación de contraseña olvidada	El sistema permite iniciar el proceso de recuperación	El proceso se ejecuta correctamente
PF-5	Generación de código para restablecimiento de contraseña	Se envía un código de verificación al usuario	El código es generado y enviado correctamente al correo.
PF-6	Actualización de datos del perfil	El usuario puede modificar su información personal	Los cambios se guardan correctamente

PF-7	Cambio de contraseña	El sistema actualiza la contraseña del usuario	La contraseña se modifica correctamente
PF-8	Listado de usuarios	El sistema muestra el listado completo de usuarios	La información se visualiza correctamente
PF-9	Edición de usuario	Se permite modificar la información del usuario	Los datos se actualizan sin errores
PF-10	Cambio de rol de usuario	El administrador actualiza el rol asignado	El rol se modifica correctamente
PF-11	Bloqueo de usuario	El acceso del usuario es restringido	El usuario queda bloqueado correctamente
PF-12	Creación de proyectos	El sistema permite registrar nuevos proyectos	El proyecto se crea correctamente
PF-13	Edición de proyectos	Se pueden modificar los datos del proyecto	Los cambios se reflejan correctamente
PF-14	Definición de reglas personalizadas por proyecto	El sistema permite configurar reglas específicas	Las reglas se aplican correctamente
PF-15	Eliminación de proyectos	El sistema permite eliminar proyectos existentes	El proyecto es eliminado correctamente con un borrado lógico
PF-16	Listado de tickets	Se visualiza el conjunto de tickets registrados	La información se presenta correctamente
PF-17	Aplicación de filtros avanzados en tickets	El sistema filtra tickets según criterios definidos	Los filtros funcionan conforme a lo esperado

PF-18	Apertura de un ticket	El sistema registra una nueva incidencia	El ticket se crea correctamente
PF-19	Visualización de tickets	Se permite consultar el detalle del ticket	El contenido se muestra correctamente con todo su historial
PF-20	Adjuntar archivos a un ticket	El sistema permite asociar archivos al ticket	Los archivos se adjuntan correctamente
PF-21	Actualización de ticket mediante hilos	Se registran respuestas y seguimientos	Los hilos se almacenan correctamente
PF-22	Visualización de alertas del sistema	El sistema muestra alertas activas	Las alertas se visualizan correctamente
PF-23	Recepción de notificaciones por correo	El usuario recibe correos del sistema	Los correos llegan correctamente
PF-24	Consulta del historial de notificaciones	Se muestra el registro de notificaciones enviadas	El historial se presenta correctamente
PF-25	Visualización de reportes generales	El sistema genera reportes consolidados	Los reportes se muestran correctamente
PF-26	Consulta de reportes de SLA	Se visualiza el cumplimiento de acuerdos	La información es correcta y actualizada
PF-27	Revisión de la bolsa de horas	El sistema muestra el consumo de horas	La información se presenta correctamente

5.3 Pruebas de tiempo de respuesta

Con el propósito de evaluar el rendimiento del sistema y verificar su comportamiento ante las solicitudes realizadas por los usuarios, se llevaron a cabo pruebas de tiempo de respuesta sobre los Endpoints del Backend. Estas pruebas permitieron medir la eficiencia del sistema en términos de procesamiento y entrega de información.

Para la ejecución de estas pruebas, cada Endpoints fue llamado en varias ocasiones consecutivas bajo condiciones controladas, registrando el tiempo de respuesta obtenido en cada ejecución. Posteriormente, se calculó un valor promedio por Endpoint, obteniendo una medición representativa y permita un análisis más preciso del comportamiento general del sistema.

Los tiempos promedio obtenidos fueron evaluados utilizando una escala de clasificación definida previamente, la cual establece rangos de desempeño según el tiempo de respuesta registrado. Dicha clasificación, presentada en la Tabla 22 permitió valorar de manera uniforme el rendimiento de cada Endpoint analizado.

Tabla 22

Rangos de evaluación tiempos de respuesta

Tiempo	Calificación
<3 segundos	Muy bueno
>3 segundos y <5 segundos	Bueno
>5 segundos	Regular

Los resultados consolidados se presentan en la Tabla 23. A partir de estos datos fue posible identificar que la mayoría de los servicios se mantienen dentro de rangos aceptables de respuesta, lo que evidencia un desempeño adecuado del Backend y que operaciones principales del sistema no afectan la experiencia del usuario final.

Tabla 23

Evaluación Endpoints tiempos de respuesta

Endpoint	Tipo	Promedio	Resultado
/login	post	4.37	Bueno
/logout	post	1.16	Muy bueno
/register	post	3.31	Bueno
/forgot-password-code	post	4.01	Bueno
/reset-password	post	3.36	Bueno
/notifications-poll/{userId}	get	0.81	Muy bueno
/change-password	post	2.92	Muy bueno
/users/disable/{userId}	post	2.75	Muy bueno
/create-projects	post	3.99	Bueno
/projects	get	4.11	Bueno
/projects/{id}	get	2.67	Muy bueno
/delete-project/{id}	patch	1.87	Muy bueno
/update-project/{id}	patch	3.01	Bueno
/create-ticket	post	5.18	Regular
/new-thread	post	4.72	Bueno
/ticket/{id}	get	2.88	Muy bueno
/notifications/{userId}	get	3.45	Bueno
/notifications-unread/{userId}	get	1.11	Muy bueno
/mark-read/{notificationId}	post	1.34	Muy bueno
/get-profile/{id}	get	2.71	Muy bueno
/update-profile/{id}	patch	3.01	Bueno
/profiles	get	4.11	Bueno
/projects-byuser'	post	1.20	Muy bueno
/users-byproject	post	1.32	Muy bueno
/tickets	post	5.60	Regular
/tickets-listprojects	get	2.89	Muy bueno
/tickets-listusers	get	2.57	Muy bueno
/similar-tickets	post	3.98	Bueno
/general-report	post	4.01	Bueno

/sla-report	post	3.94	Bueno
/home	post	3.24	Bueno

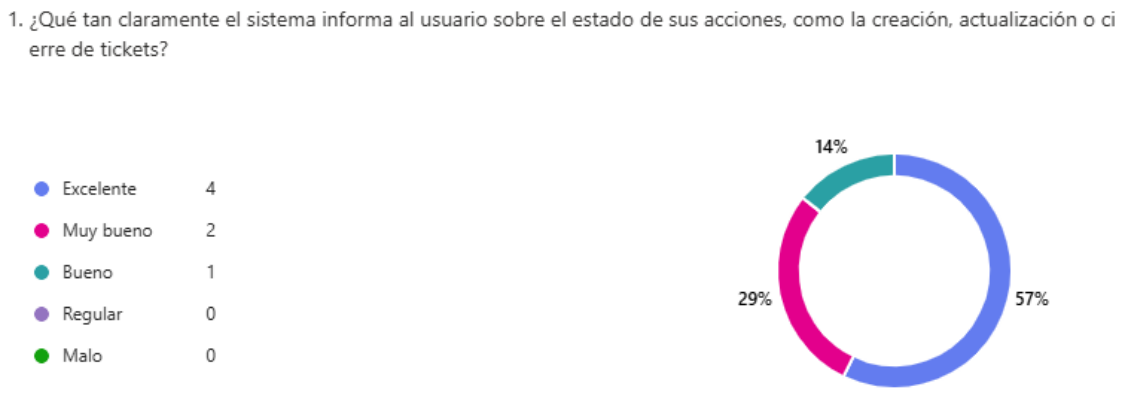
5.4 Pruebas de usabilidad

Las pruebas de usabilidad tienen como finalidad evaluar la calidad de uso del sistema desde la perspectiva de la interacción humano con el computador. Estas pruebas se llevarán a cabo tomando como referencia las heurísticas de usabilidad propuestas por Nielsen (Mahfudz et al., 2022) las cuales constituyen un conjunto de principios para identificar problemas de diseño en interfaces digitales. La evaluación permitirá analizar aspectos como la facilidad de uso, la consistencia del sistema, la prevención de errores, la claridad de la información presentada y la eficiencia en la ejecución de tareas.

El proceso de evaluación se realizó en un conjunto de preguntas estructuradas, dirigidas a expertos en el área de Tecnologías de la Información, quienes cuentan con experiencia en un Service Desk. La información recopilada permitirá evaluar la experiencia del usuario final. De esta manera, las pruebas de usabilidad permiten garantizar que el producto desarrollado sea intuitivo, eficiente y acorde a las necesidades de los usuarios.

Figura 34

Encuesta usabilidad pregunta 1

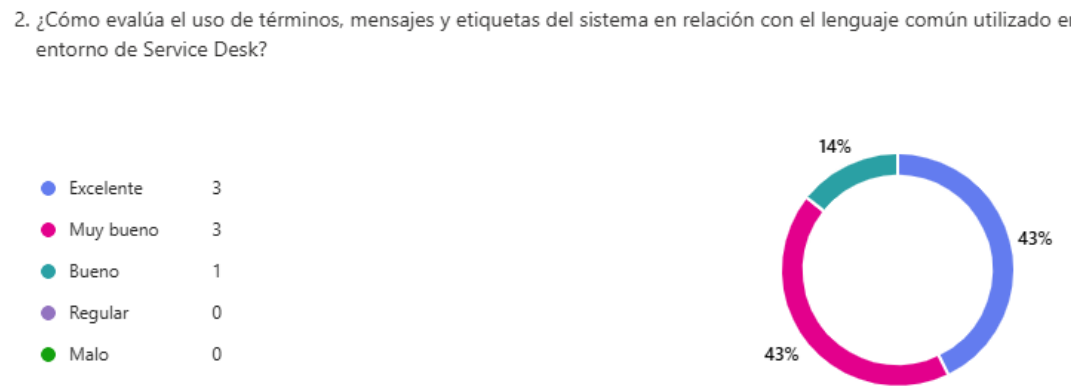


La mayoría de los encuestados percibe que el sistema informa de manera adecuada sobre las acciones que se ejecutan, como la apertura o cierre de tickets. Lo que demuestra que la retroalimentación brindada es clara y contribuye a disminuir la incertidumbre del usuario. Tal

como se observa en la Figura 34, esta valoración positiva indica que el sistema mantiene al usuario constantemente informado.

Figura 35

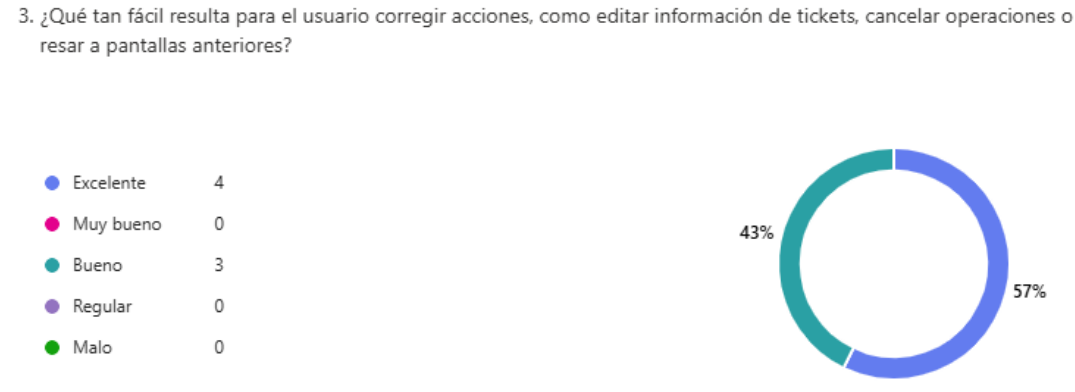
Encuesta usabilidad pregunta 2



El 86% de los participantes evalúa de forma favorable el uso de términos, mensajes y etiquetas dentro del sistema, evidenciando que el lenguaje empleado es coherente con el utilizado habitualmente en contextos de soporte técnico. De acuerdo con la visualización presentada en la Figura 35, el sistema logra transmitir los mensajes de manera clara y precisa.

Figura 36

Encuesta usabilidad pregunta 3

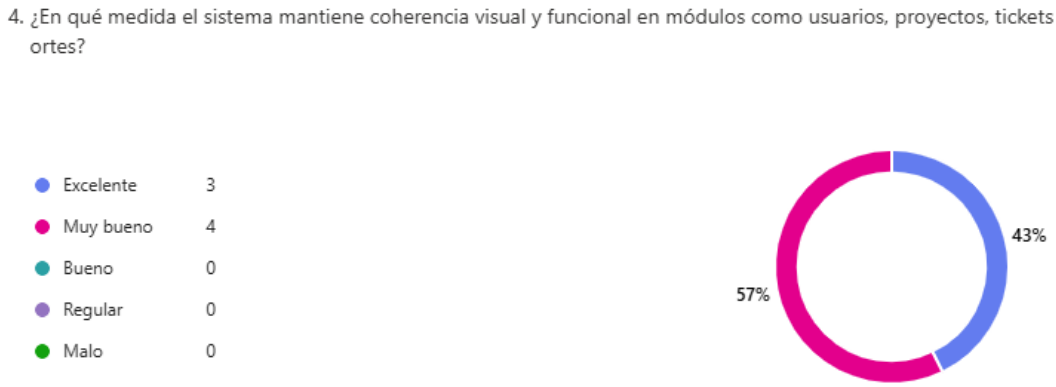


Si bien el 57% de los usuarios califica como “excelente” la posibilidad de corregir errores o modificar acciones, un 43% la considera “buena”. Esto sugiere que, aunque el sistema permite

rectificar operaciones, existen aspectos que podrían mejorarse. Esta distribución, reflejada en la Figura 36 señala oportunidades para optimizar la experiencia de corrección de errores.

Figura 37

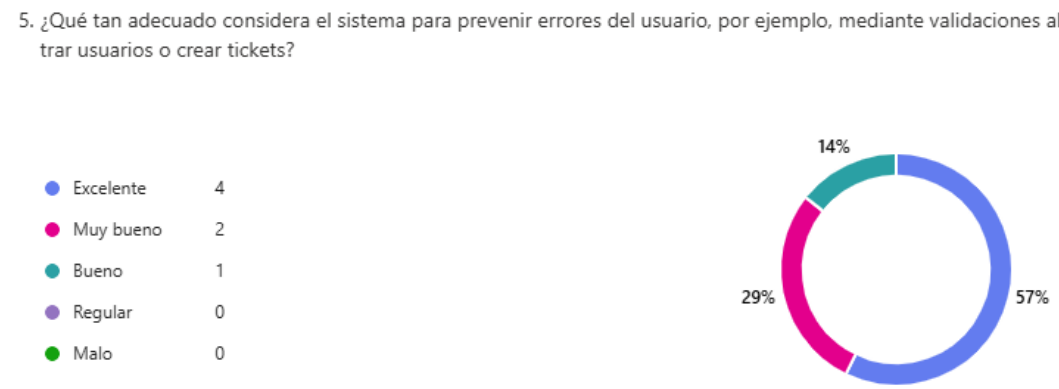
Encuesta usabilidad pregunta 4



La uniformidad en el diseño y funcionamiento de los distintos módulos, como usuarios, proyectos y reportes, obtiene una aprobación total, con el 100% de las respuestas entre "muy bueno" y "excelente". Esto demuestra que el sistema mantiene una estructura coherente que facilita la navegación y el reconocimiento de sus secciones. Tal como se aprecia en la Figura 37, esta consistencia favorece una experiencia de uso continua y ordenada.

Figura 38

Encuesta usabilidad pregunta 5

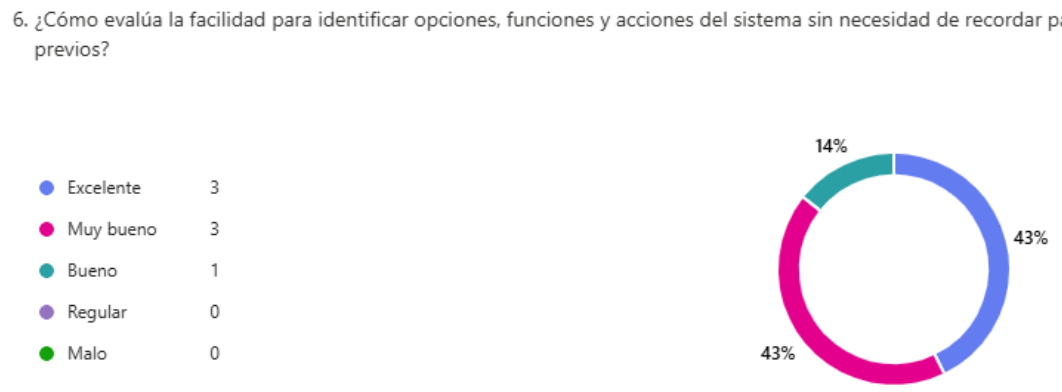


Un 86% de los usuarios considera que el sistema incorpora mecanismos eficaces para prevenir errores, especialmente mediante validaciones en procesos clave como la creación de usuarios o

tickets. Según lo mostrado en la Figura 38, esta percepción indica que el sistema anticipa fallos frecuentes y disminuye la probabilidad de errores por parte del usuario.

Figura 39

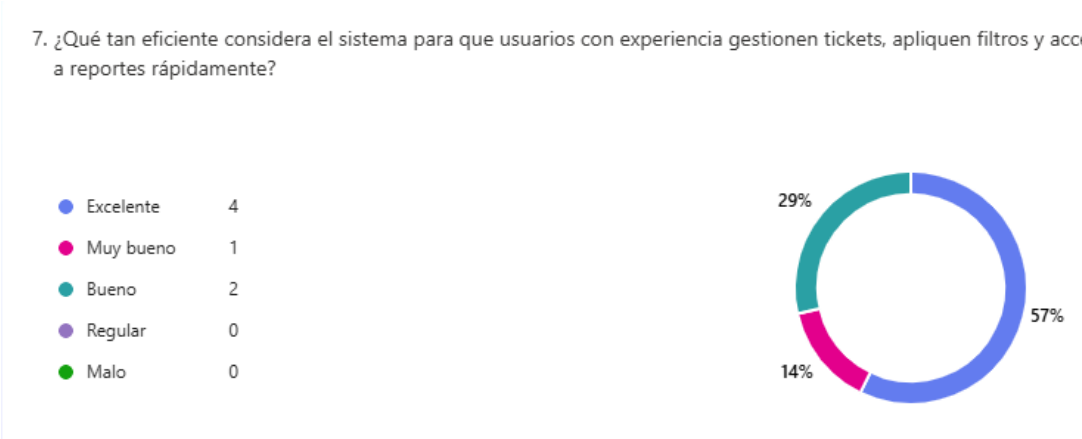
Encuesta usabilidad pregunta 6



La mayoría de los encuestados valora positivamente la facilidad para identificar funciones sin necesidad de memorizar pasos previos. Esto sugiere que la interfaz es intuitiva y presenta las opciones de manera visible y accesible. Tal como se observa en la Figura 39, esta característica reduce la carga cognitiva y favorece el uso autónomo del sistema.

Figura 40

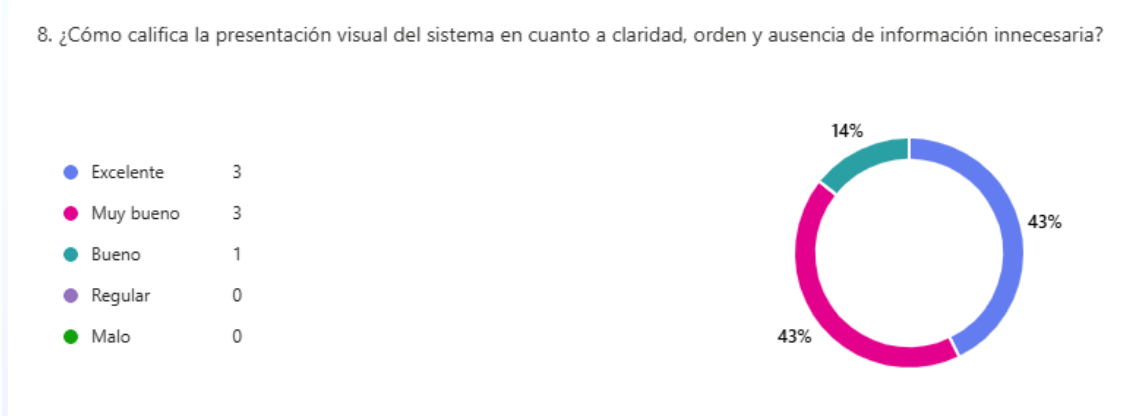
Encuesta usabilidad pregunta 7



El 71% de los usuarios considera que el sistema es eficiente para tareas como la gestión de tickets, el uso de filtros y la consulta de reportes. Esta evaluación, reflejada en la Figura 40, evidencia que las funcionalidades avanzadas responden adecuadamente a las necesidades de usuarios expertos.

Figura 41

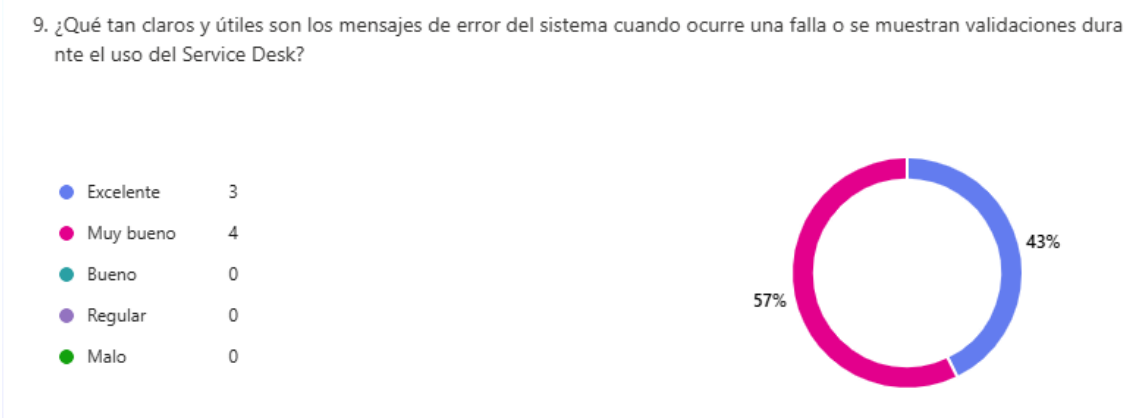
Encuesta usabilidad pregunta 8



La claridad y el orden visual de la interfaz son valorados positivamente por el 86% de los encuestados. De acuerdo con el gráfico presentado en la Figura 41, esta percepción resalta la adecuada organización de los elementos y la ausencia de información innecesaria.

Figura 42

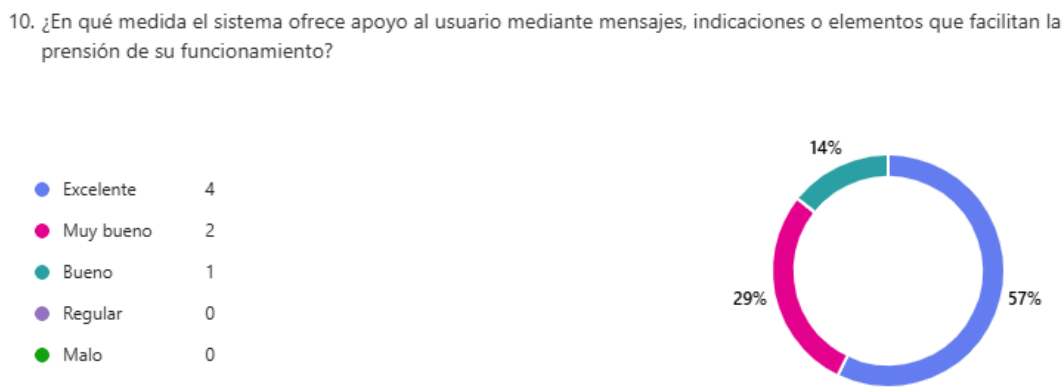
Encuesta usabilidad pregunta 9



La totalidad de los encuestados considera que los mensajes de error son claros y útiles, lo que demuestra una comunicación efectiva ante fallos o validaciones incorrectas. El gráfico reflejado en la Figura 42 indica que el sistema no solo informa sobre el problema, sino que también orienta al usuario para su correcta resolución.

Figura 43

Encuesta usabilidad pregunta 10



El 86% de los participantes valora de manera positiva el apoyo brindado por el sistema a través de mensajes informativos e indicaciones. Tal como se muestra la Figura 43, esta apreciación sugiere que el sistema proporciona ayudas contextuales que facilitan el aprendizaje y promueven un uso más autónomo de sus funcionalidades.

A partir de los resultados obtenidos se puede evidenciar que el sistema cumple de manera satisfactoria con los principios establecidos para una adecuada interacción entre el usuario y la plataforma. Las valoraciones emitidas por los especialistas reflejan una percepción positiva en aspectos clave como retroalimentación del sistema, claridad del lenguaje, consistencia en el diseño, prevención de errores y eficiencia en la ejecución de tareas.

En conjunto, los datos recopilados permiten concluir que el sistema desarrollado satisface los criterios de usabilidad planteados, garantizando una experiencia intuitiva, eficiente y alineada con las necesidades del entorno de Service Desk. Por lo tanto, se puede afirmar que las pruebas realizadas validan el cumplimiento de los objetivos establecidos en esta fase de evaluación.

CAPITULO 6

DISCUSIÓN

6.1 Conclusiones

- El levantamiento de los requerimientos funcionales y técnicos permitió identificar con claridad las necesidades del Service Desk, estableciendo una base sólida para el desarrollo. Este proceso facilitó la definición de especificaciones coherentes con los requerimientos operativos.
- El diseño e implementación del módulo de gestión de tickets permitió estructurar flujos de trabajo adaptables a distintos escenarios, incorporando reglas personalizadas de acuerdos de nivel de servicio. Esto posibilitó una gestión más eficiente de las solicitudes, mejorando el control y seguimiento.
- La creación de una base de conocimiento facilitó la reutilización de soluciones documentadas, permitiendo identificar incidencias recurrentes y reducir la duplicidad de respuestas. Esta funcionalidad contribuyó a optimizar los tiempos de resolución y trabajo.
- La incorporación de notificaciones automáticas por correo electrónico y alertas internas permitió mejorar la comunicación entre el sistema y los usuarios, favoreciendo el cumplimiento de los acuerdos de nivel de servicio y mantenerlos informados.
- La integración del módulo estadístico posibilitó la generación de métricas y KPIs clave, el cumplimiento de los acuerdos de nivel de servicio y el consumo de la bolsa de horas por proyecto. Esta información fue fundamental para la toma de decisiones y la evaluación continua.
- La implementación de un mecanismo de gestión documental garantizó el almacenamiento, organización y trazabilidad de la información asociada a los tickets. Esto permitió mantener un historial confiable para su seguimiento incluso tras el cierre.
- La ejecución de pruebas funcionales y de tiempos de respuesta permitió validar el funcionamiento del sistema web, asegurando que los módulos implementados cumplieran los requerimientos definidos. Los resultados confirmaron estabilidad, confiabilidad y adecuado desempeño del Service Desk.

6.2 Recomendaciones

- Se sugiere realizar un monitoreo periódico y detallado de los costos asociados a los servicios de AWS, con el fin de prevenir incrementos inesperados en la facturación. Un control constante permitirá optimizar el uso de los recursos en la nube y evitar gastos innecesarios.
- Es recomendable realizar de manera regular respaldo de la base de datos, garantizando así la protección de la misma y así proteger la información de imprevistos que puedan comprometer la integridad de los datos.
- Ante eventuales procesos de auditoría, se aconseja mantener actualizada y supervisada de forma continua la gestión del almacenamiento en Amazon S3. Esto permitirá asegurar la correcta generación, conservación y disponibilidad de los archivos, reduciendo el riesgo de pérdidas de información.
- En caso de incorporar nuevas funcionalidades al sistema, se recomienda que estas sean diseñadas bajo un esquema basado en roles de usuario, lo cual facilitará una asignación adecuada de permisos y una gestión más clara y organizada de las responsabilidades dentro de la plataforma.
- Si el sistema es desplegado en un entorno de nube, y considerando que el Backend se encuentra Dockerizado, se sugiere configurar el envío de correos electrónicos utilizando servicios confiables de prueba o producción, como Mailtrap u otros similares, para garantizar la correcta gestión y validación de las notificaciones por correo.

BIBLIOGRAFIA

Alami, A., & Krancher, O. (2022). How Scrum adds value to achieving software quality? *Empirical Software Engineering*, *27*(7), 165. https://doi.org/10.1007/s10664-022-10208-4

Almeida, G. C., & Rodríguez, E. M. P. (2017). Diagnóstico de los sistemas de información en las empresas priorizadas según los requerimientos actuales. *Palabra Clave (La Plata)*, *6*(2), 1-11.

Aranda, H. (2025, abril 21). *¿Qué es un Acuerdo de Nivel de Servicio (SLA)?* ¿Qué es un Acuerdo de Nivel de Servicio (SLA)? https://invgate.com/es/itsm/service-level-management/service-level-agreement

Atkinson, R. (2020, febrero 4). Which IT Support and IT Service Management Roles? *ITSM.Tools*. https://itsm.tools/which-it-support-and-itsm-roles-does-your-organization-have-and-need/

Barros, I., Santos, F., & Candini, S. (2024). *Heurísticas en la evaluación de la usabilidad de aplicaciones móviles: Conceptos y aplicación*. 59-67. https://doi.org/10.26439/ciis2023.7080

Bayona-Oré, S., & Hostos, M. (2022). Metrics for Performance Improvement in Organisations Using Scrum, ITIL and CMMI. *WSEAS TRANSACTIONS ON ELECTRONICS*, *13*, 89-99. https://doi.org/10.37394/232017.2022.13.12

Berrospi, E. E. (2022). *Implementación de kpis en el área de soporte en sitio para evaluar la productividad en una empresa de telecomunicaciones*. https://repositorio.usil.edu.pe/server/api/core/bitstreams/96640eb0-1718-49fa-9d66-59ffef7416c8/content

Betz, C., & Hewitt, A. (2020, octubre 6). *¿Qué es un service desk? | IBM*. ¿Qué es un service desk? https://www.ibm.com/mx-es/think/topics/service-desk

Craddock, A. (2024, abril 14). *Top 10 Benefits of Scrum and Frequent Challenges*. APMG International. https://apmg-international.com/article/top-10-benefits-scrum-and-frequent-challenges

Danby, S. (2025, enero 29). *Las 10 principales tendencias de ITSM a tener en cuenta en 2025*. https://blog.invgate.com/es/tendencias-de-itsm

Díaz, I., & Sánchez, J. (2022). *Metamorfosis: Un Marco para el Análisis de Requisitos Funcionales*. 1-12.

Estrada Velasco, M. V., Núñez Villacis, J. A., Saltos Chávez, P. R., & Cunuhay Cuchipe, W. C. (2021). Revisión Sistemática de la Metodología Scrum para el Desarrollo de Software. *Dominio de las Ciencias*, *7*(Extra 4), 54.

Franco. (2024, marzo 22). *Innovación en Soporte Técnico en Línea para la Era Digital*. https://chatia.app/innovacion-en-soporte-tecnico-en-linea-la-era-digital/

Franzen, R. (2023, mayo 8). *A History of Help Desks (So Far)*. https://www.thinkhdi.com/library/supportworld/2023/history-of-help-desks

Galup, S. D., Dattero, R., Quan, J. J., & Conger, S. (2009). An overview of IT service management. *Commun. ACM*, *52*(5), 124-127. https://doi.org/10.1145/1506409.1506439

García de Zúñiga, F. (2024, octubre 16). *Redmine: Qué es y principales ventajas | Blog de Arsys*. Arsys. https://www.arsys.es/blog/redmine-gestion-proyectos-cloud

Gesto, A. (2024, noviembre 19). *8 Service Desk Trends to Look Out For in 2025*. https://blog.invgate.com/service-desk-trends

Ghimire, D., Charters, S., Ghimire, D., & Charters, S. (2022). The Impact of Agile Development Practices on Project Outcomes. *Software*, *1*(3), 265-275. https://doi.org/10.3390/software1030012

Gratas, B. (2023, mayo 3). *Las 15 métricas para help desk y service desk más importantes para medir el rendimiento*. https://blog.invgate.com/es/metricas-para-help-desk-y-service-desk

Hassani-Alaoui, S., Cameron, A.-F., & Giannelia, T. (2020). *"We Use Scrum, but …": Agile Modifications and Project Success*. 10.

Hernández, L. maría, Lira, J., & Pech, Y. (2022, noviembre 15). *Los roles del marco de trabajo Scrum: Un análisis de competencias y habilidades - ProQuest*. 450-459.

Hernández Sampieri, R., & Fernandez-Collado, C. F. (2014). *Metodología de la investigación* (P. Baptista Lucio, Ed.; Sexta edición). McGraw-Hill Education. https://apiperiodico.jalisco.gob.mx/api/sites/periodicooficial.jalisco.gob.mx/files/metodologia_de_la_investigacion_-_roberto_hernandez_sampieri.pdf

IBM. (2022, marzo 22). *What Is a Help Desk?* https://www.ibm.com/think/topics/help-desk

IBM. (2025, noviembre 28). *IBM Control Desk*. https://www.ibm.com/docs/es/control-desk/7.6.1?topic=changes-managing-workflows

Izaurralde, M. P. (2013). *Caracterización de Especificación de Requerimientos en entornos Ágiles: Historias de Usuario*.

https://www.institucional.frc.utn.edu.ar/sistemas/lidicalso/pub/file/Tesis/Anteproyecto_Requerimientos_en_Metodolog%C3%ADas_Agiles.pdf

Jovanović, M. (2024, marzo 9). *What Is a Modular Monolith?* https://www.milanjovanovic.tech/blog/what-is-a-modular-monolith

Lanet. (2024, diciembre 29). *Importancia Soporte Técnico TI En Era Digital: Tendencias 2024*. https://www.lanet.mx/soporte-tecnico-ti/

López Vargas, Y., & Vázquez Chávez, A. (2016). La Gestión de Servicios de soporte técnico en el ciclo de vida del desarrollo de software. *Revista Cubana de Ciencias Informáticas, 10*, 46-60.

Mahfudz, M. S., Agusti, F., Zahra, S. A., & Dhini, B. R. (2022). HEURISTIC EVALUATION ANALYSIS USING THE 10 NIELSEN RULE USABILITY METHOD ON THE KAI ACCESS APPLICATION. *Proceeding of International Conference on Science, Health, And Technology*, 325-337. https://doi.org/10.47701/icohetech.v3i1.2154

Malnik, J. (2023, octubre 31). 13 Most Important Help Desk KPIs to Track and Measure Help Desk Performance. *Databox*. https://databox.com/most-important-help-desk-kpis

Mera Paz, J. (2016). Análisis del proceso de pruebas de calidad de software. *Ingeniería Solidaria*, *12*(20), 163-176. https://doi.org/10.16925/in.v12i20.1482

Mohanakrishnan, M. (2022, noviembre 13). *ITIL Service Desk Guide: Process, Best Practices*. ITIL Service Desk Guide: Process, Best Practices. https://www.knowledgehut.com/blog/it-service-management/itil-service-desk

Mora, M., Raisinghani, M., & Gelman, O. (2013). *A Comparison of Service Design Processes in Relevant International ITSM Models and Standards*. https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1515&context=sprouts_all

Nicolazzo, S., Nocera, A., & Pedrycz, W. (2024). *Service Level Agreements and Security SLA: A Comprehensive Survey*. https://doi.org/10.48550/arXiv.2405.00009

Palacios, J. (2023). *Scrum: Guía fundamental • Jeronimo Palacios*. https://jeronimopalacios.com/scrum/

Ramirez Bravo, P., & Donoso Jaurés, F. (2006). *METODOLOGÍA ITIL Descripción, Funcionamiento y Aplicaciones* [Universidad de Chile]. https://repositorio.uchile.cl/bitstream/handle/2250/108405/donoso_f.pdf

Salle, M. (2004). *IT Service Management and IT Governance: Review, Comparative Analysis and their Impact on Utility Computing*. 2.

Schwenke, E. (2023, febrero 22). *What is an IT Service Desk?* https://otrs.com/blog/itsm/it-service-desk/

Suárez, L. M. R. (2019). *CALIDAD EN EL LEVANTAMIENTO DE REQUERIMIENTOS EN PROYECTOS DE SOFTWARE*.

Tolosa-Cuadrado, C. L., & González-Sanabria, J. S. (2014). Revista Científica. *Revista Científica*, *19*(2), 134-147. https://doi.org/10.14483/23448350.6500

Trinkenreich, B., Santos, G., & Perini Barcellos, M. (2015). Metrics to Support IT Service Maturity Models—A Systematic Mapping Study: *Proceedings of the 17th International Conference on Enterprise Information Systems*, 330-337. https://doi.org/10.5220/0005376003300337

Valle, N. (2025, noviembre 20). *Top 15 Help Desk Metrics to Measure IT Support Performance*. https://blog.invgate.com/service-desk-kpi

Venegas, A., Villar, E., & Mendoza, A. (2022). *Machine Learning para Automatizar los Sistemas de Tickets de Soporte: Una Revisión de Literatura | Campus*. https://portalrevistas.aulavirtualusmp.pe/index.php/rc/article/view/2264?articlesBySimilarityPage=6

Wuchner, A. (2024, diciembre 12). Software al Final de su Vida Útil: Riesgos, Peligros y Qué Hacer a Continuación. *Kiteworks | Your Private Data Network*. https://www.kiteworks.com/es/cumplimiento-regulatorio/end-life-software-risks/

Yung. (2019, septiembre 5). *Mailhog: How to Use It and Why*. https://mailtrap.io/blog/mailhog-explained/

Zaldivar Gamboa, Jh. (1995). *Métricas de Software* [Unam]. https://ru.dgb.unam.mx/server/api/core/bitstreams/a929b317-4c2a-4c2f-aa83-bac15ceb3122/content

Zendesk. (2020, octubre 7). *9 tipos de service desk para equipos de soporte exitosos*. https://www.zendesk.com.mx/blog/tipos-de-service-desk/

Zendesk. (2024, febrero 23). *Ticket de soporte técnico: Definición y funciones*. https://www.zendesk.com.mx/blog/ticket-de-soporte-tecnico/

ANEXOS

ANEXO A: SPRINT BACKLOG

Sprint 1: Planificación, configuración y preparación del entorno de desarrollo

Objetivo: Establecer la base técnica del proyecto, recopilar requerimientos, definir arquitectura y preparar el entorno de desarrollo.

Actividades:

Planificación y documentación

- Planificación del proyecto
- Recolección de requerimientos funcionales y no funcionales
- Identificar métricas y KPIs
- Identificación de actores y roles
- Redacción de historias de usuario
- Redacción de los objetivos del proyecto para tesis

Diagramas y arquitectura

- Diagrama de casos de uso
- Diagrama de componentes
- Diagrama de arquitectura técnica (frontend, backend, base de datos, aws)
- Esquema base de datos

Configuración técnica

- Crear repositorios Git
- Crear estructura de carpetas frontend y backend
- Crear Dockerfile y docker-compose.yml
- Configurar Opcache
- Configurar Mailhog para pruebas de correo
- Configurar cuenta de Aws
- Configurar pool de usuarios
- Configurar bucket de s3 para almacenamiento
- Configurar variables de entorno .env
- Instalar dependencias base:

Instalar dependencias:

Backend: aws-sdk, php-jwt

Frontend: aws-amplify, jwt-decode, Angular Material, ngx-file-drop, ngx-charts, tailwindcss

- Redactar README técnico inicial en ambos repositorios

Sprint 2: Autenticación y gestión de usuarios

Objetivo: Implementar el sistema de autenticación, recuperación de contraseña y gestión básica de usuarios.

Actividades:

Backend

- Crear tablas:

users

roles

- Crear endpoints:

POST /register

POST /login

POST /forgot-password-code

POST /reset-password

POST /change-password

PATCH /users/disable/{userId}

GET /get-profile/{id}

PATCH /update-profile/{id}

GET /profiles

- Integrar Cognito y JWT
- Crear middleware auth:api y cognito
- Validar roles y estados de usuario

Frontend

- Crear layouts: login, register, forgot-password
- Validaciones de formularios
- Crear servicio de autenticación
- Consumir endpoints de login, registro y recuperación
- Crear guard para rutas protegidas
- Crear componente de perfil y edición de perfil
- Crear componente de listado de usuarios

- Crear botón para bloquear un usuario
- Crear interceptor para manejo de sesión

Sprint 3: Módulo de proyectos, tickets y notificaciones

Objetivo: Desarrollar el sistema de tickets con hilos, prioridades y notificaciones.

Actividades:

Backend

- Crear tablas:

projects

projet_user

tickets

ticket_threads

ticket_priorities

ticket_statuses

notifications

- Crear endpoints:

POST /create-ticket

POST /new-thread

GET/ /projects

GET/ /projects/{id}

POST/ /create-projects

PATCH/ /delete-project/{id}

PATCH/ /update-project/{id}

GET /ticket/{id}

POST /tickets

GET /tickets-listprojects

GET /tickets-listusers

GET /notifications/{userId}

GET /notifications-unread/{userId}

POST /mark-read/{notificationId}

GET /notifications-poll/{userId}

- Implementar lógica de asignación, cierre, cambio de estado
- Integrar subida de archivos a S3 en hilos

- Configurar envío de correos
- Crear notificaciones:

Ticket creado

Ticket asignado

Ticket cerrado

Cambio de estado

Cambio de prioridad

Asignación a proyecto

Ticket SLA

Olvido contraseña

Frontend

- Crear componentes:

Ticket list

Ticket view

Ticket update

New ticket

Bandeja de tickets

Filtros avanzados

- Crear componente de notificaciones:

Badge de cantidad

Lista de notificaciones

Marcar como leído

- Mostrar historial de hilos y archivos adjuntos

Sprint 4: Estilos, maquetación y base de conocimientos

Objetivo: Mejorar la experiencia visual y comenzar el módulo de base de conocimientos.

Actividades:

Frontend

- Aplicar estilos globales con Angular Material
- Aplicra estilos reponsivos con Tailwindcss
- Crear componentes reutilizables:

Alertas

Diálogos

Spinner

- Maquetar vistas responsivas
- Crear módulo base de conocimientos:

Estructura de categorías y artículos

Vista de artículos

Navegación por temas

Filtros y búsqueda

Backend

- Crear tabla tickets_knowledge
- Crear endpoints:

POST /similar-tickets

Sprint 5: Módulo de estadísticas

Objetivo: Implementar métricas clave, dashboard y visualización de datos.

Actividades:

Backend

- Crear endpoints para métricas:

POST /general-report

POST /sla-report

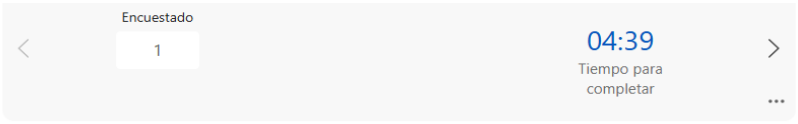
POST /home

- Agregar filtros por fecha, proyecto
- Documentar indicadores clave

Frontend

- Crear dashboard de estadísticas general
- Crear dashboard de home
- Crear dashboard de sla
- Visualizar métricas con filtros dinámicos
- Documentar utilidad de métricas en la tesis

ANEXO B: ENCUESTAS



1. Por favor, indique su cargo o rol dentro de la organización *

Database Administrator DBA

2. ¿Qué tan conforme se encuentra con el módulo Service Desk de Redmine? *

- ( ) Muy satisfecho
- ( ) Satisfecho
- ( ) Neutral
- (•) Insatisfecho
- ( ) Muy insatisfecho

3. ¿Cuáles son las principales limitaciones o problemas técnicos que encuentra al usar el módulo actual? *

no hay trazabilidad clara de los tickets relacionados a cambios de datos. Los adjuntos de los tickets se pierden al cerrar los tickets.

4. ¿Qué procesos de su trabajo se ven más afectados por las limitaciones del sistema actual? *

Auditorías y seguimiento de cambios en los datos. La falta de trazabilidad retrasa la verificación de incidentes

5. ¿Consideraría útil si el sistema contara con notificaciones y alertas automáticas del sistema? ¿Por qué? *

Sería muy útil. Actualmente no recibimos avisos sobre tickets urgentes o relacionados

6. ¿Ha tenido dificultades para acceder a información de tickets cerrados o documentos relacionados? *

Todo el tiempo debo buscar manualmente en varios archivos y carpetas para encontrar la información

7. ¿Qué tan importante considera la sugerencia de tickets para evitar apertura de tickets repetitivos y así mejorar eficiencia? *

- (•) Muy importante
- ( ) Poco importante
- ( ) Nada importante

8. ¿Qué tipo de reportes o métricas considera imprescindibles para el service desk? *

Historial de cambios, Sumatoria de tiempos de resolución , reportes por impacto en la base de datos

9. Si pudiera cambiar o agregar funcionalidades al sistema, ¿Cuáles serían y por qué? *

Trazabilidad completa de tickets, alertas y notificaciones

10. ¿Qué esperaría de un service desk que actualmente no ofrece Redmine? *

Un sistema que permita seguimiento completo de tickets, alertas críticas y acceso a documentación histórica

Encuestado

2 Anónimo

07:08
Tiempo para completar

1. Por favor, indique su cargo o rol dentro de la organización *

Coordinador de área en Producción

2. ¿Qué tan conforme se encuentra con el módulo Service Desk de Redmine? *

- ( ) Muy satisfecho
- ( ) Satisfecho
- (•) Neutral
- ( ) Insatisfecho
- ( ) Muy insatisfecho

3. ¿Cuáles son las principales limitaciones o problemas técnicos que encuentra al usar el módulo actual? *

Existen problemas en el flujo de asignación de tickets es rígido y muchas veces los tickets quedan sin responsable asignado o se duplican. Adicional la interfaz no permite revisar el estado de varios tickets simultáneamente.

4. ¿Qué procesos de su trabajo se ven más afectados por las limitaciones del sistema actual? *

Supervisión de incidencias en producción y coordinación con el equipo de soporte. Problemas al asignar tickets Retrasos en la asignación afectan directamente la continuidad de los servicios especialmente cuando hay gran carga de tickets abiertos.

5. ¿Consideraría útil si el sistema contara con notificaciones y alertas automáticas del sistema? ¿Por qué? *

Sería útil para priorizar problemas urgentes y reducir tiempos de reacción. Especialmente para los clientes mas complejos.

6. ¿Ha tenido dificultades para acceder a información de tickets cerrados o documentos relacionados? *

No es recurrente pero cuando necesito revisar tickets cerrados de semanas atrás.

7. ¿Qué tan importante considera la sugerencia de tickets para evitar apertura de tickets repetitivos y así mejorar eficiencia? *

- (•) Muy importante
- ( ) Poco importante
- ( ) Nada importante

8. ¿Qué tipo de reportes o métricas considera imprescindibles para el service desk? *

-Cantidad de tickets abiertos -Tiempos de resolución -Tickets criticos sin atender -Bolsa de horas distribuida por cliente y por proyecto mes etc.

9. Si pudiera cambiar o agregar funcionalidades al sistema, ¿Cuáles serían y por qué? *

Dashboard con visualización de data importante a los proyectos designados o coordinados por mi.

10. ¿Qué esperaría de un service desk que actualmente no ofrece Redmine? *

Priorizar urgencias y obtener reportes claros para tomar decisiones.

Encuestado

3

11:53

Tiempo para completar

1. Por favor, indique su cargo o rol dentro de la organización *

Ingeniero de Soporte

2. ¿Qué tan conforme se encuentra con el módulo Service Desk de Redmine? *

- ( ) Muy satisfecho
- ( ) Satisfecho
- ( ) Neutral
- ( ) Insatisfecho
- (•) Muy insatisfecho

3. ¿Cuáles son las principales limitaciones o problemas técnicos que encuentra al usar el módulo actual? *

El sistema no notifica cambios de estado ni asignaciones. Se generan tickets duplicados porque los usuarios no saben si su solicitud ya fue registrada

4. ¿Qué procesos de su trabajo se ven más afectados por las limitaciones del sistema actual? *

Atención de los tickets diarios y resolución de incidencias

5. ¿Consideraría útil si el sistema contara con notificaciones y alertas automáticas del sistema? ¿Por qué? *

Es fundamental para no dejar tickets urgentes sin atender y reducir tiempos de respuesta

6. ¿Ha tenido dificultades para acceder a información de tickets cerrados o documentos relacionados? *

Si, especialmente cuando necesito referencias de soluciones previas para los problemas recurrentes

7. ¿Qué tan importante considera la sugerencia de tickets para evitar apertura de tickets repetitivos y así mejorar eficiencia? *

- (•) Muy importante
- ( ) Poco importante
- ( ) Nada importante

8. ¿Qué tipo de reportes o métricas considera imprescindibles para el service desk? *

Conteo de bolsa de horas y tiempo de trabajo

9. Si pudiera cambiar o agregar funcionalidades al sistema, ¿Cuáles serían y por qué? *

Alertas automáticas y posibilidad de reasignar tickets fácilmente

10. ¿Qué esperaría de un service desk que actualmente no ofrece Redmine? *

Reportes en bolsa de horas para reportar a clientes de manera mas pronta. Así como también una interfaz mas moderna.

Encuestado

4

07:37

Tiempo para completar

1. Por favor, indique su cargo o rol dentro de la organización *

Soporte TI

2. ¿Qué tan conforme se encuentra con el módulo Service Desk de Redmine? *

- ( ) Muy satisfecho
- ( ) Satisfecho
- ( ) Neutral
- (•) Insatisfecho
- ( ) Muy insatisfecho

3. ¿Cuáles son las principales limitaciones o problemas técnicos que encuentra al usar el módulo actual? *

A veces debo hacer seguimiento manual de tickets. Los archivos adjuntos se pierden y toca hacer respaldo manual de archivos en maquinas locales.

4. ¿Qué procesos de su trabajo se ven más afectados por las limitaciones del sistema actual? *

Coordinación de los integrantes al momento de recibir múltiples tickets como los de primera respuesta o los de prioridad alta que lleguen.

5. ¿Consideraría útil si el sistema contara con notificaciones y alertas automáticas del sistema? ¿Por qué? *

Si especialmente para los de primera respuesta.

6. ¿Ha tenido dificultades para acceder a información de tickets cerrados o documentos relacionados? *

Si, la búsqueda de información histórica es complicada y consume mucho tiempo porque toca buscar en las maquinas virtualizadas que tenemos. Además ello conlleva a tener que hacer depuraciones mensuales que consumen más tiempo aún.

7. ¿Qué tan importante considera la sugerencia de tickets para evitar apertura de tickets repetitivos y así mejorar eficiencia? *

- (•) Muy importante
- ( ) Poco importante
- ( ) Nada importante

8. ¿Qué tipo de reportes o métricas considera imprescindibles para el service desk? *

Mostrar tickets por prioridades, por incidencia, por consumo de horas, cargas de trabajo por proyectos

9. Si pudiera cambiar o agregar funcionalidades al sistema, ¿Cuáles serían y por qué? *

Mejora en almacenamiento de los ficheros, notificaciones, primera respuesta automática ,sugerencias de soluciones. para mayor eficiencia en el trabajo y respuesta de tickets.

10. ¿Qué esperaría de un service desk que actualmente no ofrece Redmine? *

Que centralice toda la información, añadir reportería, que reduzca tareas repetitivas y facilite el cumplimiento de SLA

ANEXO C: HISTORIAS DE USUARIO

HU-1
Inicio de Sesión (Login) Como usuario registrado del sistema, necesito autenticarme mediante mi correo electrónico y contraseña para acceder de manera segura a la plataforma y utilizar sus funcionalidades. • El sistema debe presentar un formulario que contenga obligatoriamente los campos de correo electrónico y contraseña. • Ambos campos deben ser de ingreso obligatorio; mientras alguno se encuentre vacío, el botón de iniciar sesión debe permanecer deshabilitado. • El campo de correo electrónico debe validarse automáticamente para verificar que cumple con un formato válido. • El formulario debe incluir un enlace visible para la opción "¿Olvidaste tu contraseña?", el cual debe redirigir al proceso de recuperación. • Debe incluirse un segundo enlace con el texto "¿Aún no tienes cuenta? Regístrate", que redirija a la pantalla de registro. • Al enviar el formulario con credenciales válidas, el sistema debe autenticar al usuario y generar un token de seguridad. • Una vez autenticado correctamente, el usuario debe ser redirigido a la pantalla principal del sistema. • En caso de credenciales incorrectas, el sistema debe mostrar un mensaje de error informativo sin especificar cuál dato es incorrecto. • La sesión iniciada debe mantenerse activa durante un periodo de una hora, salvo que el usuario cierre sesión manualmente.

HU-2
Registro de Usuario Como visitante del sistema, deseo crear una cuenta ingresando mis datos personales para poder acceder a la plataforma.

- El sistema debe mostrar un formulario de registro con los campos: correo electrónico, cédula, nombres, apellidos, puesto, contraseña y confirmación de contraseña.
- El correo electrónico ingresado debe validarse para verificar que no se encuentre previamente registrado en el sistema.
- La cédula debe cumplir con el formato establecido según las reglas definidas.
- La contraseña debe cumplir obligatoriamente con las siguientes condiciones:
    - Mínimo ocho caracteres.
    - Al menos una letra mayúscula.
    - Al menos un número.
    - Al menos un carácter especial.
- El campo de confirmación de contraseña debe coincidir exactamente con la contraseña ingresada.
- El botón de registro debe permanecer deshabilitado hasta que todos los campos estén completos y correctamente validados.
- El formulario debe incluir un enlace para redirigir al inicio de sesión en caso de que el usuario ya tenga una cuenta.
- Al completar el registro exitosamente, el sistema debe mostrar un mensaje de confirmación.

Tras el registro exitoso, el usuario debe ser redirigido automáticamente a la pantalla de inicio de sesión.

HU-3

Formulario de Recuperación de Contraseña

Como usuario que olvidó su contraseña, necesito un mecanismo seguro para restablecerla y recuperar el acceso al sistema.

- El proceso de recuperación debe dividirse en dos etapas claramente diferenciadas.
- En la primera etapa, el sistema debe solicitar el ingreso del correo electrónico del usuario.
- El botón para continuar debe activarse únicamente cuando el correo tenga un formato válido.

- Al confirmar el correo, el sistema debe enviar un código temporal al correo electrónico registrado.
- En la segunda etapa, el sistema debe solicitar:
    - El código temporal recibido por correo.
    - La nueva contraseña.
    - La confirmación de la nueva contraseña.
- La nueva contraseña debe cumplir las reglas de seguridad definidas.
- El sistema debe validar que el código temporal ingresado sea correcto y vigente.
- Si el proceso es exitoso, se debe mostrar un mensaje de confirmación y redirigir al usuario al inicio de sesión.
- En caso de error, el sistema debe mostrar mensajes claros indicando problemas con el código o las contraseñas.

HU-4

Header
Como usuario autenticado, necesito un encabezado fijo que me permita navegar, acceder a mi perfil y gestionar mi sesión.
• El encabezado debe mostrarse de forma fija en la parte superior del sistema.
• Debe incluir un botón alineado a la izquierda que permita mostrar u ocultar la barra de navegación lateral.
• El nombre del sistema “Service Desk” debe visualizarse de forma clara en el encabezado.
• En la parte derecha debe mostrarse un ícono de usuario representado por la inicial del nombre.
• Al seleccionar el ícono de usuario, debe desplegarse un menú con dos opciones:
  ○ Acceso al perfil del usuario.
  ○ Cierre de sesión.
• Al cerrar sesión, el sistema debe eliminar el token de autenticación y limpiar el almacenamiento local.
• El cierre de sesión debe redirigir automáticamente al inicio de sesión.

- Junto al ícono del usuario debe mostrarse un ícono de notificaciones con un indicador numérico de notificaciones no leídas.
- Al seleccionar el ícono de notificaciones, se debe mostrar un listado de las notificaciones pendientes.

HU-5

Barra de navegación

Como usuario del sistema, necesito una barra de navegación clara para acceder rápidamente a las diferentes secciones.

- La barra de navegación debe mostrar un listado de opciones de acceso.
- Las opciones disponibles deben incluir: inicio, proyectos, tickets, mi bandeja, notificaciones y reportes.
- Las opciones visibles deben adaptarse según el rol del usuario autenticado.
- Para el rol administrador, debe mostrarse adicionalmente la opción de gestión de usuarios.
- Cada opción debe redirigir correctamente a la pantalla correspondiente.
- La barra debe poder mostrarse u ocultarse mediante el botón ubicado en el encabezado.

HU-6

Pantalla de Inicio (Home)

Como usuario autenticado, deseo visualizar un resumen de mis actividades principales al ingresar al sistema.

- La pantalla de inicio debe mostrarse automáticamente al ingresar al sistema luego de una autenticación exitosa.
- La vista debe estar dividida en dos secciones principales claramente diferenciadas.
- La primera sección debe mostrar un listado de los tickets asignados al usuario autenticado.
- Cada ticket debe mostrar información básica como número, título y estado.

- Al seleccionar un ticket desde esta sección, el sistema debe redirigir a la pantalla de visualización del ticket correspondiente.
- La segunda sección debe mostrar los proyectos en los cuales el usuario se encuentra asignado.
- Cada proyecto debe identificarse visualmente por su nombre.
- Al seleccionar un proyecto, el sistema debe redirigir a la bandeja de tickets asociada a dicho proyecto.
- En caso de no existir tickets o proyectos asignados, el sistema debe mostrar un mensaje informativo.

HU-7
Gestión de Perfil de Usuario Como usuario del sistema, necesito consultar y actualizar mi información personal para mantener mis datos actualizados. • El sistema debe mostrar un formulario con la información personal del usuario autenticado. • Los campos de correo electrónico, cédula y rol deben mostrarse bloqueados y no permitir edición. • Los campos de nombre, apellido y puesto deben ser editables. • El formulario debe cargarse con los datos actuales del usuario. • El botón de actualización del perfil debe permanecer deshabilitado mientras existan campos obligatorios vacíos o inválidos. • Al presionar el botón de actualización, el sistema debe validar la información ingresada. • Si la actualización es exitosa, se debe mostrar un mensaje de confirmación. • En caso de error, el sistema debe notificar al usuario mediante un mensaje informativo. • En la parte inferior de la pantalla debe existir un botón para actualizar la contraseña. • Al seleccionar esta opción, el sistema debe redirigir a la pantalla de cambio de contraseña.

HU-8
Actualización de Contraseña Como usuario autenticado, deseo cambiar mi contraseña para reforzar la seguridad de mi cuenta. • El sistema debe mostrar un formulario con tres campos obligatorios: contraseña actual, nueva contraseña y confirmación de nueva contraseña. • La nueva contraseña debe cumplir con las reglas de seguridad definidas por el sistema. • El botón para actualizar la contraseña debe permanecer deshabilitado hasta que todos los campos estén completos y sean válidos. • El sistema debe validar que la contraseña actual ingresada sea correcta. • La confirmación de contraseña debe coincidir con la nueva contraseña ingresada. • Al completarse el proceso correctamente, el sistema debe mostrar un mensaje de confirmación. • En caso de que ocurra un error, se debe mostrar un mensaje indicando la causa del problema.

HU-9
Listado de Usuarios Como administrador, necesito visualizar el listado de usuarios registrados para gestionar sus cuentas. • El sistema debe mostrar una tabla con el listado de todos los usuarios registrados. • La tabla debe incluir las columnas: nombre, correo electrónico, puesto, rol y estado. • Debe existir un campo de búsqueda que permita filtrar usuarios de forma rápida. • El listado debe contar con paginación para facilitar la navegación. • Las columnas deben permitir ordenamiento ascendente y descendente. • Cada fila del listado debe incluir un botón de edición.

- Al seleccionar el botón de edición, el sistema debe redirigir a la pantalla de edición del usuario correspondiente.
- El acceso a esta funcionalidad debe estar restringido únicamente al rol administrador.

HU-10
Edición de Usuario Como administrador, deseo modificar la información y el rol de un usuario para mantener un control adecuado del sistema. • El sistema debe mostrar un formulario con la información del usuario seleccionado. • El formulario debe contener los mismos campos definidos en la gestión de perfil del usuario. • Los campos de correo electrónico y cédula deben mostrarse bloqueados y no permitir edición. • Los campos de nombre, apellido y puesto deben ser editables. • Se debe incluir un campo adicional que permita modificar el rol del usuario. • El botón de actualización debe permanecer deshabilitado hasta que el formulario sea válido. • En la parte inferior debe existir un botón para bloquear al usuario. • Al seleccionar la opción de bloqueo, el sistema debe mostrar una alerta de confirmación. • Si la acción se completa correctamente, se debe mostrar un mensaje de confirmación. • En caso de error, el sistema debe notificar al administrador mediante un mensaje informativo.

HU-11
Listado de Proyectos Como usuario autorizado, necesito consultar los proyectos disponibles para gestionar mi trabajo.

- El sistema debe mostrar una pantalla con un listado de proyectos registrados.
- El listado debe presentarse en formato de tabla.
- La tabla debe incluir las columnas: nombre del proyecto, descripción, usuario creador y número de usuarios asignados.
- Debe existir un campo de búsqueda que permita filtrar proyectos de manera rápida por nombre.
- El listado debe contar con paginación para facilitar la navegación cuando existan múltiples registros.
- Las columnas deben permitir ordenamiento ascendente y descendente.
- Al seleccionar el nombre de un proyecto, el sistema debe redirigir a la bandeja de tickets del proyecto correspondiente.
- Cada fila debe incluir una opción para editar el proyecto, visible únicamente para los roles administrador y Project Manager.
- En la parte superior debe existir un botón para crear un nuevo proyecto, accesible solo para los roles administrador y Project Manager.

HU-12
Actualización de Proyecto Como administrador o Project Manager, necesito editar la información de un proyecto para mantenerla actualizada. • El sistema debe mostrar un formulario con la información actual del proyecto seleccionado. • El formulario debe permitir editar el nombre y la descripción del proyecto. • Debe incluir un componente tipo chips que permita agregar o quitar usuarios asignados al proyecto. • El listado de usuarios disponibles debe corresponder únicamente a usuarios activos. • El botón de actualización debe permanecer deshabilitado mientras el formulario esté incompleto o contenga errores de validación. • El sistema debe validar los cambios antes de permitir su almacenamiento. • Al actualizar correctamente, se debe mostrar un mensaje de confirmación.

- En caso de error, el sistema debe mostrar un mensaje informativo.
- La pantalla debe incluir un botón para eliminar el proyecto.
- Antes de eliminar, el sistema debe mostrar una alerta de confirmación.

HU-13

Creación de Nuevo Proyecto

Como administrador o Project Manager, deseo registrar nuevos proyectos en el sistema.

- El sistema debe mostrar un formulario para la creación de proyectos.
- El formulario debe incluir los campos obligatorios de nombre y descripción.
- Debe permitir la asignación de usuarios mediante un componente tipo chips.
- Solo deben mostrarse usuarios activos para su asignación.
- El botón de creación debe permanecer deshabilitado hasta que todos los campos obligatorios estén completos y validados.
- El acceso a esta funcionalidad debe estar restringido a los roles administrador y Project Manager.
- Al crear el proyecto exitosamente, el sistema debe mostrar un mensaje de confirmación.
- En caso de error durante la creación, se debe notificar al usuario mediante un mensaje informativo.

HU-14

Listado de Tickets

Como usuario del sistema, necesito consultar los tickets registrados para dar seguimiento a su estado.

El sistema debe mostrar una pantalla con el listado de todos los tickets disponibles según el rol del usuario.

El listado debe presentarse en formato de tabla.

- La tabla debe incluir las columnas: número de ticket, título, prioridad, estado, usuario creador y fecha de creación.
- Debe existir un campo de búsqueda que permita filtrar tickets rápidamente.
- El listado debe contar con paginación.
- Las columnas deben permitir ordenamiento ascendente y descendente.
- Debe existir un panel de filtros avanzados desplegable.
- Los filtros avanzados deben permitir filtrar por:
    - Proyecto asignado.
    - Usuario creador.
    - Estado del ticket.
    - Prioridad.
    - Rango de fechas.
- Debe existir un botón para aplicar los filtros seleccionados.
- Debe existir un botón para limpiar los filtros aplicados.
- En la parte superior de la pantalla debe mostrarse un botón para crear un nuevo ticket.
- Al seleccionar el número de un ticket, el sistema debe redirigir a la pantalla de visualización del ticket.

HU-15
Creación de Ticket Como usuario, necesito registrar tickets para reportar incidencias o solicitudes. El sistema debe mostrar un formulario para la creación de tickets. • El formulario debe permitir seleccionar el proyecto al cual se asociará el ticket. • Debe incluir los campos de título y descripción como obligatorios. • Para el rol cliente, debe existir una opción que permita marcar el ticket como SLA. • El formulario debe permitir adjuntar archivos mediante arrastre o selección manual.

- Debe existir una sección de opciones avanzadas.
- Dentro de las opciones avanzadas, el usuario debe poder:
    - Asignar el ticket a la bandeja general.
    - Asignar el ticket a un usuario específico perteneciente al proyecto.
- Para todos los roles, excepto el cliente, debe ser obligatorio seleccionar la prioridad del ticket.
- El botón de guardar debe permanecer deshabilitado hasta que el formulario esté completo y validado.
- Al crear el ticket exitosamente, el sistema debe mostrar un mensaje de confirmación.
- En caso de error, el sistema debe mostrar un mensaje informativo.

HU-16
Visualización de Ticket Como usuario, deseo consultar la información completa de un ticket para conocer su evolución. • El sistema debe mostrar una pantalla dedicada a la visualización del ticket seleccionado. • En la parte superior debe mostrarse una cabecera con la información general del ticket. • La cabecera debe incluir: número de ticket, título, estado actual, prioridad, proyecto al que pertenece, usuario creador y usuario asignado. • Debe mostrarse la fecha de creación y la fecha de la última actualización. • El sistema debe calcular y mostrar el tiempo transcurrido desde la creación del ticket. • Debajo de la cabecera debe mostrarse el mensaje inicial del ticket. • El número y título del ticket deben visualizarse de forma destacada. • En la parte superior debe existir un botón para crear un nuevo hilo. • Debe mostrarse un listado de los hilos asociados al ticket. • El sistema debe permitir expandir o contraer cada hilo de forma individual.

- Debe existir una opción para expandir todos los hilos o contraerlos en conjunto.
- Cada hilo debe mostrar su mensaje, archivos adjuntos y el registro de cambios realizados.
- Los registros de cambios deben incluir modificaciones de estado, prioridad y usuario responsable.
- Los hilos marcados como privados no deben ser visibles para el rol cliente.
- Todos los demás roles deben poder visualizar tanto hilos públicos como privados.

HU-17
Creación de Nuevo Hilo en Ticket Como usuario autorizado, necesito agregar nuevos mensajes a un ticket para registrar avances o cambios. • Al seleccionar la opción de nuevo hilo, el sistema debe mostrar un formulario en una ventana modal. • El formulario debe incluir un campo obligatorio para el mensaje del hilo. • Debe existir una opción que permita marcar el hilo como privado o público. • El sistema debe permitir registrar el tiempo invertido en minutos. • Debe incluir un campo desplegable para modificar la prioridad del ticket. • Debe incluir un campo desplegable para modificar el estado del ticket. • El formulario debe permitir adjuntar archivos mediante arrastre o selección manual. • Para el rol cliente, la opción de mensaje privado no debe mostrarse. • El formulario debe incluir botones para cancelar o guardar el hilo. • Al guardar correctamente, el sistema debe mostrar un mensaje de confirmación. • En caso de error, se debe mostrar un mensaje informativo.

HU-18
Pantalla “Mi Bandeja” Como usuario autenticado, necesito visualizar únicamente los tickets asignados a mí para gestionar mi trabajo. • La pantalla “Mi Bandeja” debe presentar un listado de tickets. • El listado debe reutilizar la estructura y funcionalidades del listado general de tickets. • Los tickets mostrados deben estar filtrados automáticamente por el usuario asignado. • Deben mantenerse las opciones de búsqueda, paginación, ordenamiento y filtros avanzados. • Al seleccionar un ticket, el sistema debe redirigir a la pantalla de visualización del ticket correspondiente.

HU-19
Pantalla de Notificaciones Como usuario, deseo consultar todas mis notificaciones para mantenerme informado de eventos importantes. • El sistema debe mostrar una pantalla con el listado completo de notificaciones del usuario. • Cada notificación debe mostrar un título descriptivo. • Debe incluir el mensaje asociado a la notificación. • Se debe mostrar la fecha y hora de generación. • Las notificaciones que ya han sido leídas deben visualizarse sin resaltado. • Las notificaciones pendientes de lectura deben mostrarse con un resalto visual distintivo. • El sistema debe permitir identificar fácilmente el estado de lectura de cada notificación.

HU-20

Indicador de Notificaciones no Leídas (Badge)

Como usuario, necesito identificar rápidamente las notificaciones pendientes de lectura.

- El sistema debe mostrar un indicador tipo badge asociado al ícono de notificaciones.
- El badge debe reflejar la cantidad de notificaciones no leídas.
- Al interactuar con el badge, debe desplegarse un listado de notificaciones pendientes.
- Cada notificación debe incluir una opción para marcarla como leída.
- Al marcar una notificación como leída, esta debe eliminarse del listado del badge.
- Si el usuario se desplaza hasta el final del listado, debe mostrarse una opción para marcar todas como leídas.
- La opción "marcar todas como leídas" debe redirigir a la pantalla general de notificaciones.

HU-21

Pantalla Reportes general

Como usuario, necesito acceder a una pantalla de reportes generales que me permita filtrar la información por proyecto y por rango de fechas, para analizar el desempeño y evolución de los tickets.

- El sistema debe mostrar una pantalla dedicada a reportes generales.
- Debe incluir un filtro para seleccionar uno o varios proyectos.
- Debe incluir un filtro para seleccionar un rango de fechas (desde/hasta).
- Al aplicar los filtros, deben actualizarse todos los indicadores y gráficos de la pantalla.
- La pantalla debe mostrar métricas agregadas como:
    - Total de tickets
    - Tickets abiertos y cerrados
    - Tickets con SLA
    - Tiempo promedio de resolución

- Tiempo registrado promedio
- Número de actualizaciones (públicas y privadas)
- Tiempo total invertido

- Debe incluir visualizaciones como:
  - Gráfico de estado de tickets
  - Gráfico de prioridades
  - Antigüedad de tickets
  - Top tickets por tiempo
  - Top proyectos por tiempo
  - Top usuarios por tiempo
- Los datos deben actualizarse dinámicamente al cambiar los filtros.
- Debe existir una opción para limpiar los filtros y mostrar todos los datos.

HU-22

Pantalla Reportes SLA

Como usuario, necesito acceder a una pantalla de reportes SLA por proyecto y rango de fechas, para evaluar el cumplimiento de los acuerdos de servicio contratados.

- El sistema debe mostrar una pantalla dedicada al reporte SLA.
- Debe incluir un filtro obligatorio para seleccionar el proyecto.
- Debe incluir un filtro opcional para seleccionar un rango de fechas (desde/hasta).
- Al aplicar los filtros, debe generarse el reporte con los datos correspondientes.
- El encabezado del reporte debe mostrar:
  - Nombre del proyecto
  - Rango de fechas seleccionado
  - Fecha y hora de generación del reporte
- Debe mostrarse una sección de Reglas Contratadas, que incluya:
  - Tiempo máximo para primera respuesta
  - Tiempo máximo de resolución
  - Tiempo efectivo esperado
  - Bolsa de horas contratadas
- Debe mostrarse un resumen de cumplimiento por cada regla:
  - Porcentaje de cumplimiento

- Número de tickets que cumplen / no cumplen

- Debe incluir métricas de uso de la bolsa de horas:
  - Horas contratadas
  - Horas utilizadas
  - Horas restantes
  - Porcentaje de utilización
  - Estado de disponibilidad
- Si no hay tickets en el periodo seleccionado, debe mostrarse un mensaje informativo.

HU-23

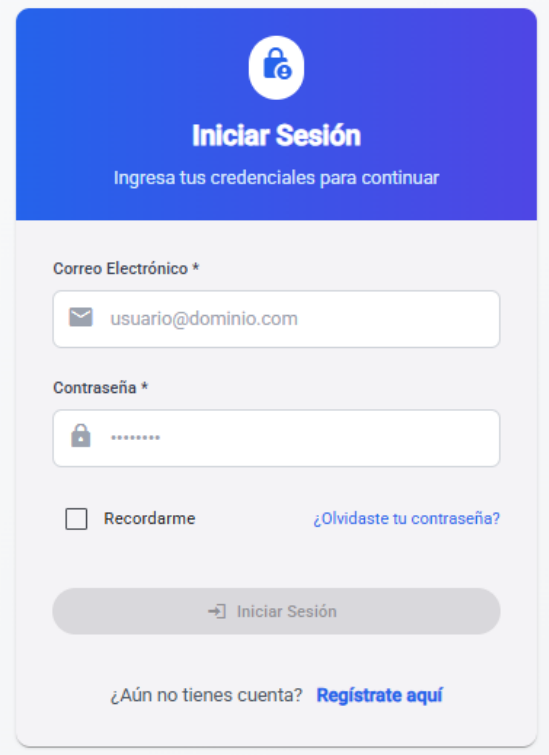
Pantalla Home

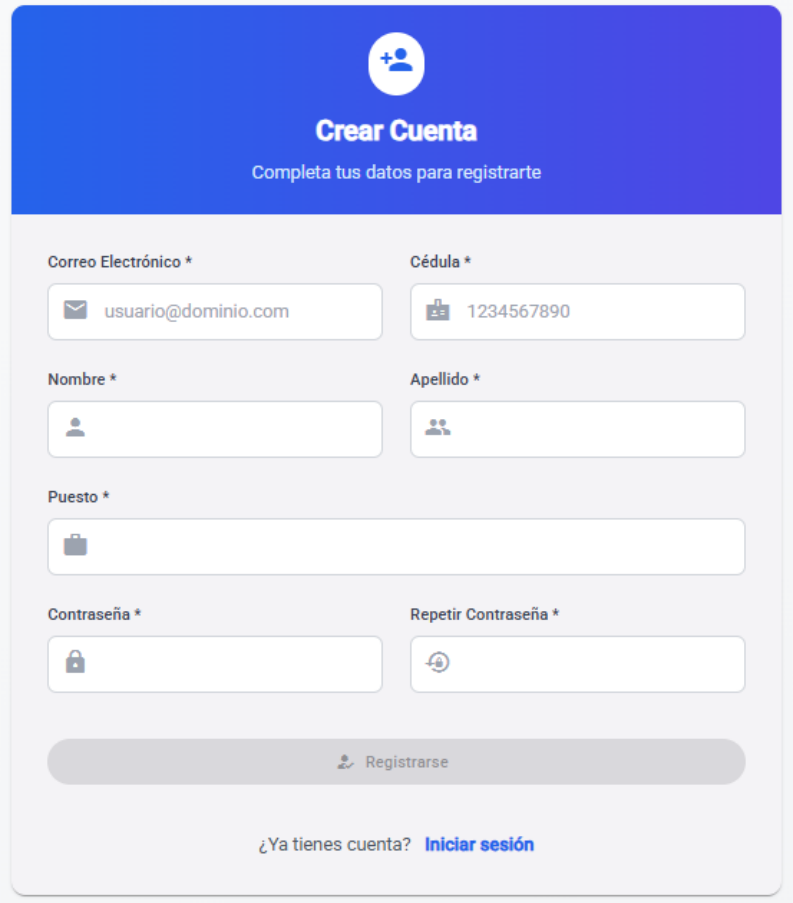
Como usuario, necesito una pantalla principal que me muestre un resumen de mi actividad y acceso rápido a mis tickets y proyectos, para gestionar eficientemente mis tareas diarias.

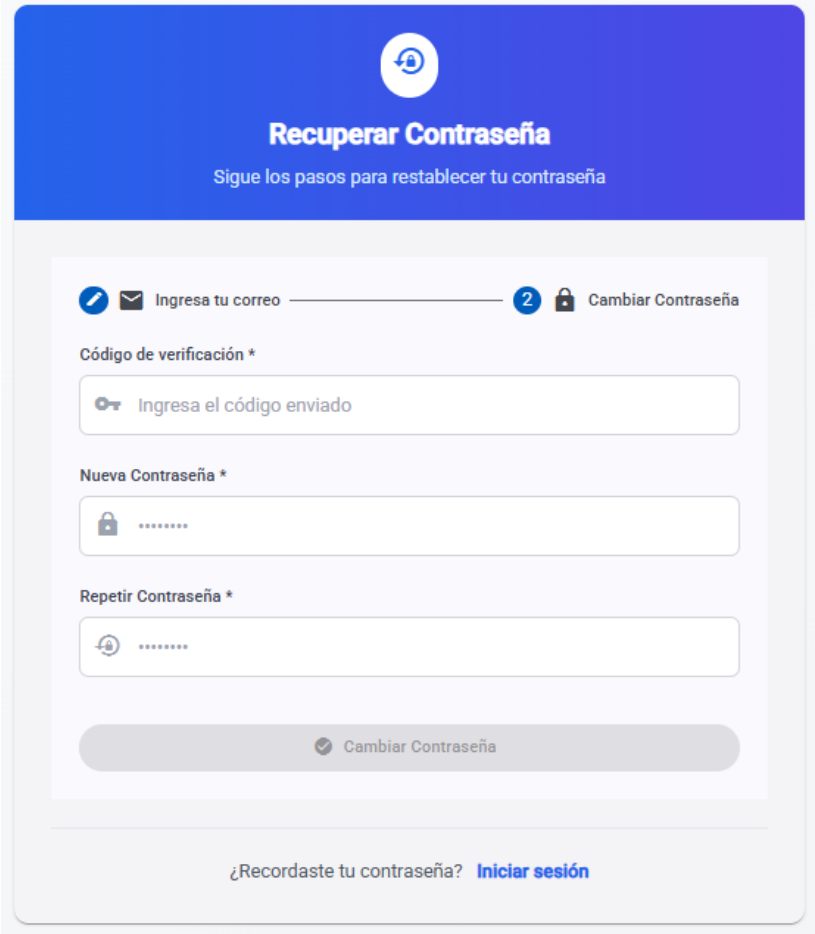
- El sistema debe mostrar una pantalla de inicio personalizada al iniciar sesión.
- Debe incluir un encabezado con el nombre del usuario, correo electrónico y roles asignados.
- Debe mostrar métricas generales del usuario:
  - Total de tickets creados
  - Total de tickets asignados
  - Tickets de alta prioridad
  - Total de actualizaciones realizadas
  - Tiempo total registrado (en minutos y horas)
- Debe incluir una sección “Mi Bandeja” con los tickets asignados al usuario:
  - Número de ticket
  - Título
  - Estado actual
  - Prioridad
- Debe incluir una sección “Mis Proyectos” con los proyectos en los que el usuario participa.

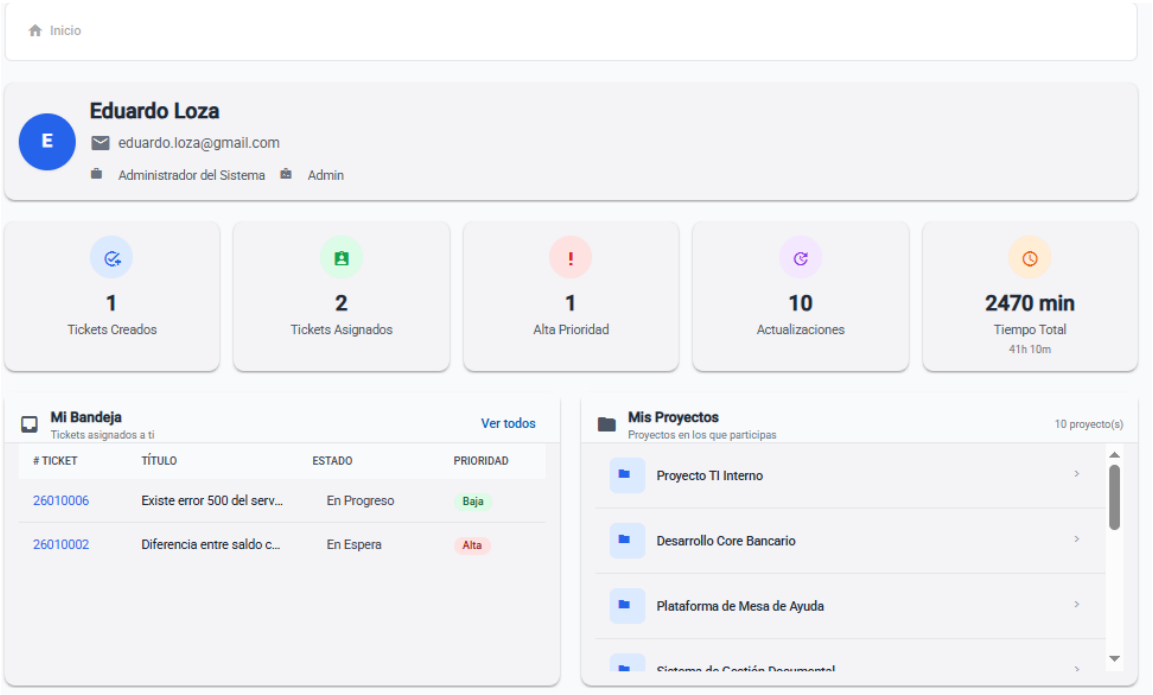
- Las métricas y listados deben actualizarse dinámicamente según la actividad del usuario.
- Los tickets deben estar ordenados por fecha de asignación o prioridad.
- Debe permitir navegación directa desde los tickets o proyectos hacia sus respectivas pantallas detalladas.

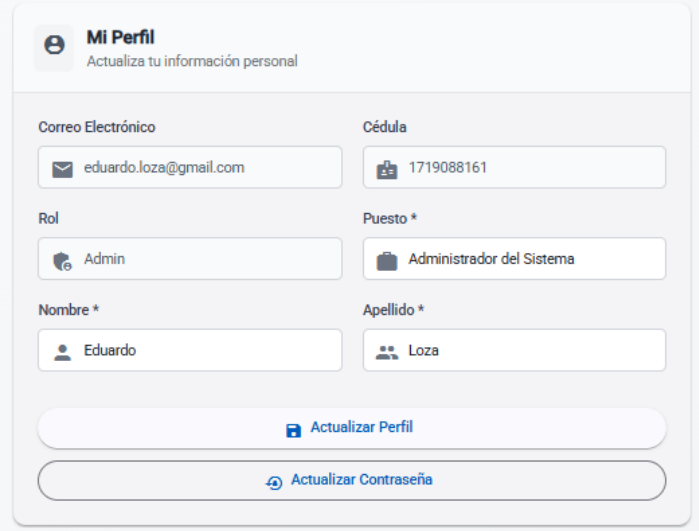
ANEXO D: PANTALLAS DEL SISTEMA

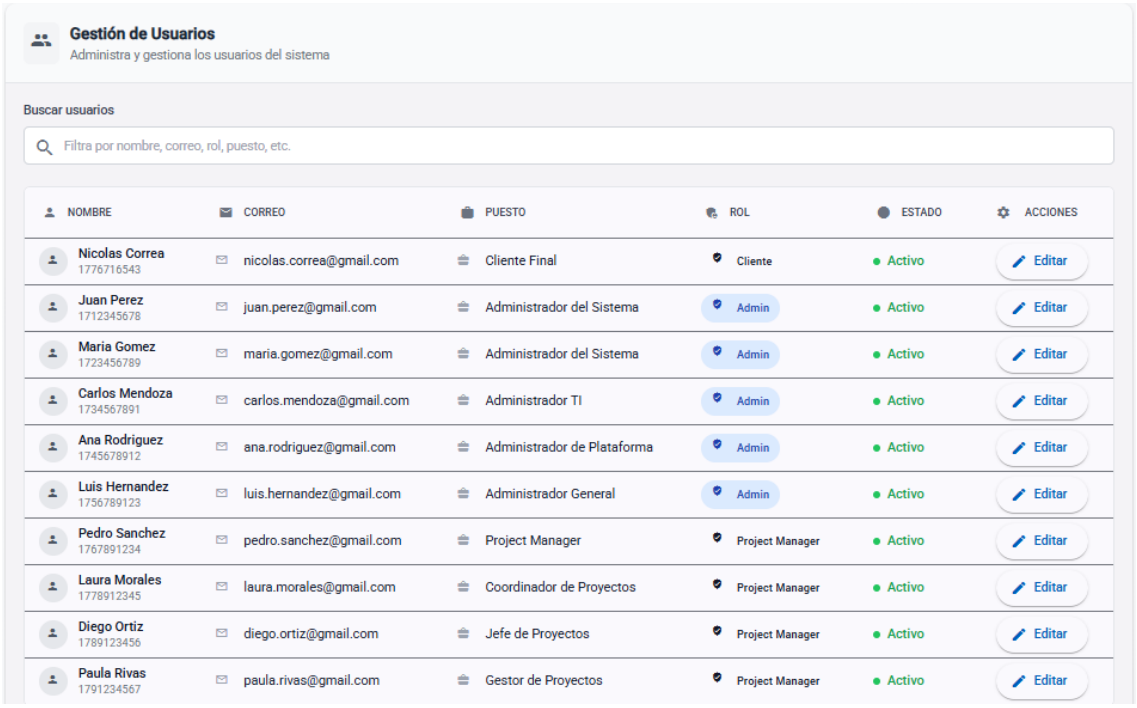


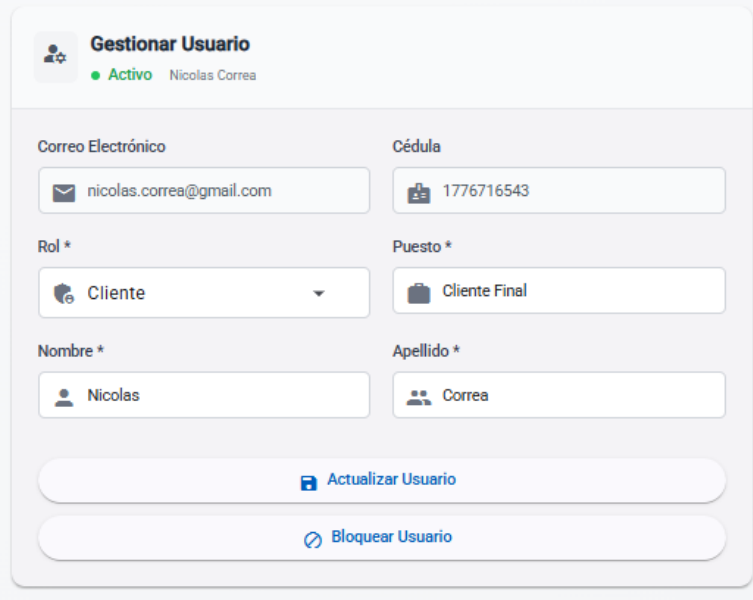


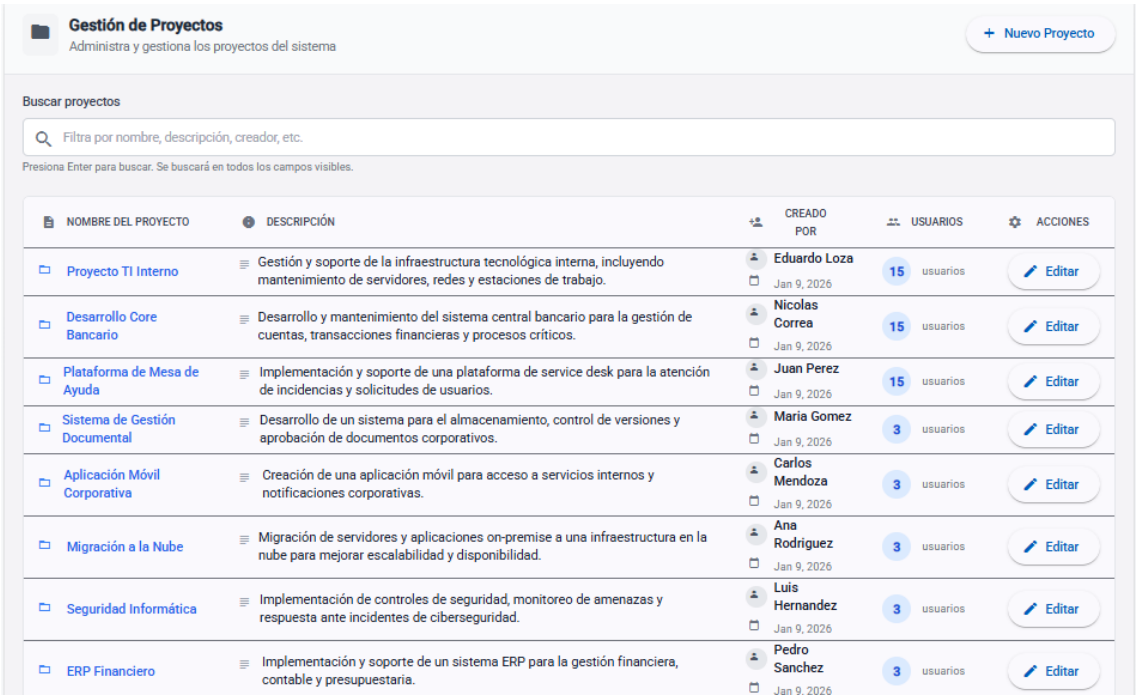

Recuperar Contraseña
Sigue los pasos para restablecer tu contraseña
Ingresa tu correo
2
Cambiar Contraseña
Código de verificación *
Ingresa el código enviado
Nueva Contraseña *
Repetir Contraseña *
Cambiar Contraseña
¿Recordaste tu contraseña? Iniciar sesión

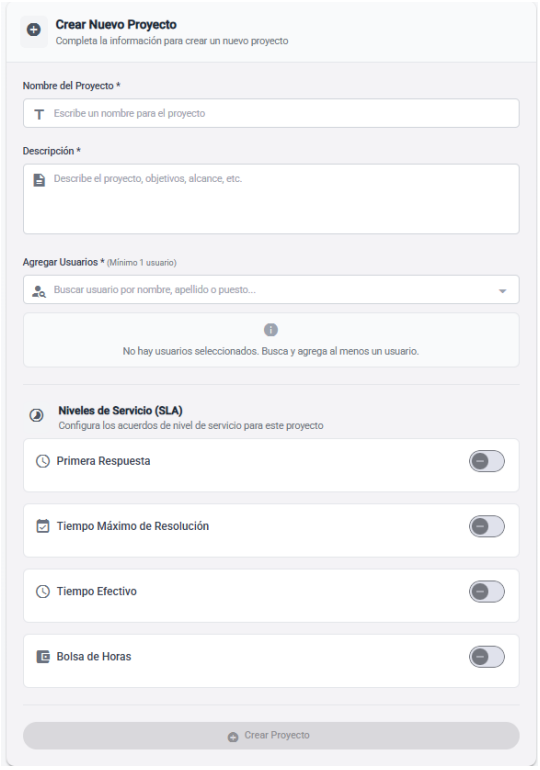

Inicio
E
Eduardo Loza
eduardo.loza@gmail.com
Administrador del Sistema
Admin
1
Tickets Creados
2
Tickets Asignados
1
Alta Prioridad
10
Actualizaciones
2470 min
Tiempo Total
41h 10m
Mi Bandeja
Tickets asignados a ti
Ver todos
TICKET
TÍTULO
ESTADO
PRIORIDAD
26010006
Existe error 500 del serv...
En Progreso
Baja
26010002
Diferencia entre saldo c...
En Espera
Alta
Mis Proyectos
Proyectos en los que participas
10 proyecto(s)
Proyecto TI Interno
Desarrollo Core Bancario
Plataforma de Mesa de Ayuda

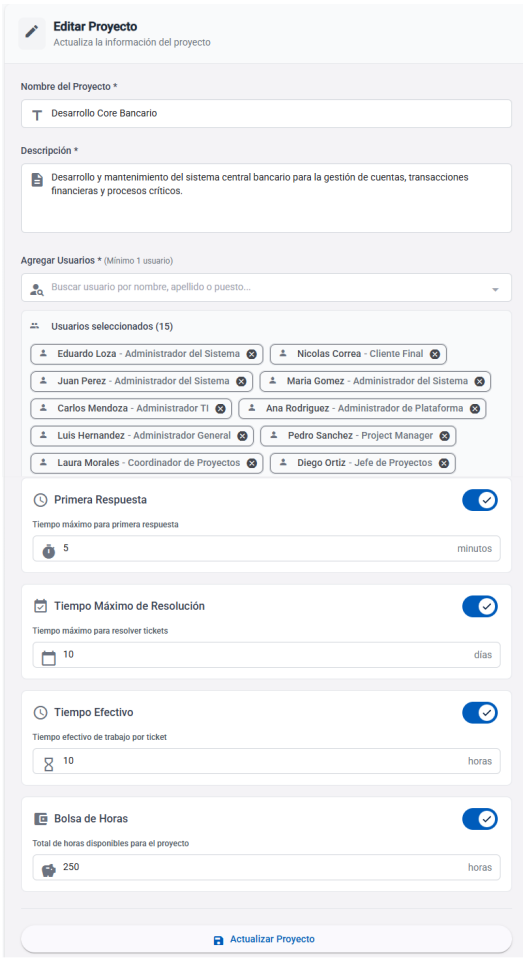

Mi Perfil
Actualiza tu información personal
Correo Electrónico
eduardo.loza@gmail.com
Cédula
1719088161
Rol
Admin
Puesto *
Administrador del Sistema
Nombre *
Eduardo
Apellido *
Loza
Actualizar Perfil
Actualizar Contraseña

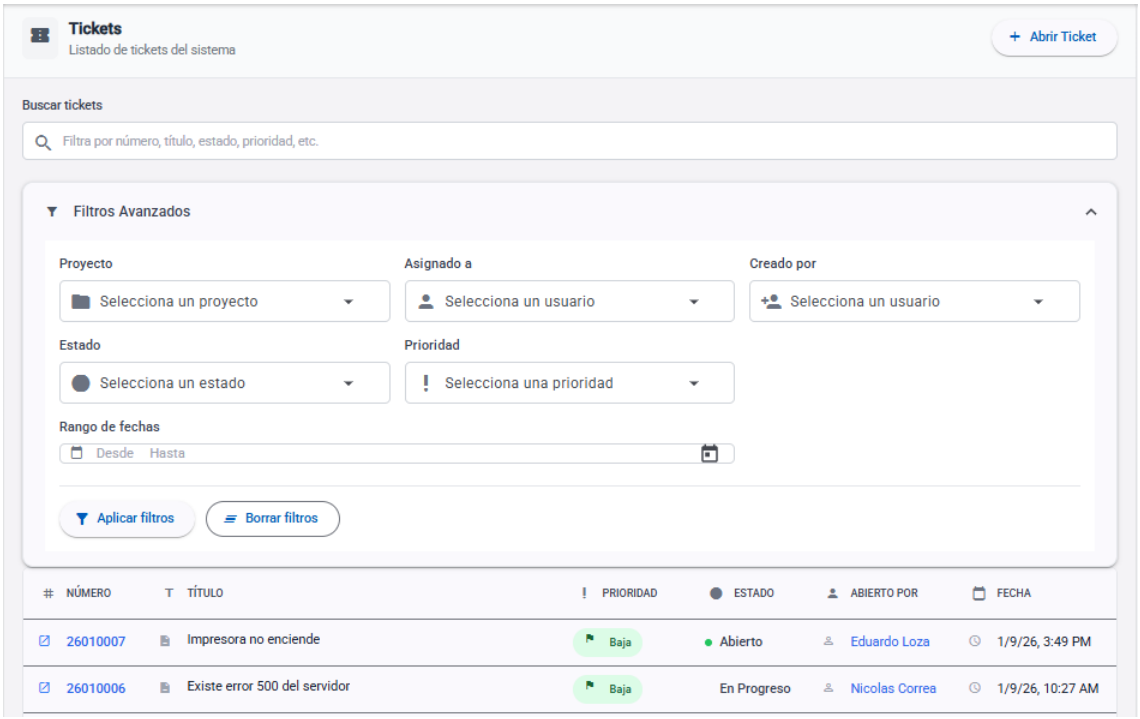

Gestión de Usuarios
Administra y gestiona los usuarios del sistema
Buscar usuarios
Filtra por nombre, correo, rol, puesto, etc.
NOMBRE
CORREO
PUESTO
ROL
ESTADO
ACCIONES
Nicolas Correa 1776716543 nicolas.correa@gmail.com Cliente Final Cliente Activo Editar
Juan Perez 1712345678 juan.perez@gmail.com Administrador del Sistema Admin Activo Editar
Maria Gomez 1723456789 maria.gomez@gmail.com Administrador del Sistema Admin Activo Editar
Carlos Mendoza 1734567891 carlos.mendoza@gmail.com Administrador TI Admin Activo Editar
Ana Rodriguez 1745678912 ana.rodriguez@gmail.com Administrador de Plataforma Admin Activo Editar
Luis Hernandez 1756789123 luis.hernandez@gmail.com Administrador General Admin Activo Editar
Pedro Sanchez 1767891234 pedro.sanchez@gmail.com Project Manager Project Manager Activo Editar
Laura Morales 1778912345 laura.morales@gmail.com Coordinador de Proyectos Project Manager Activo Editar
Diego Ortiz 1789123456 diego.ortiz@gmail.com Jefe de Proyectos Project Manager Activo Editar
Paula Rivas 1791234567 paula.rivas@gmail.com Gestor de Proyectos Project Manager Activo Editar

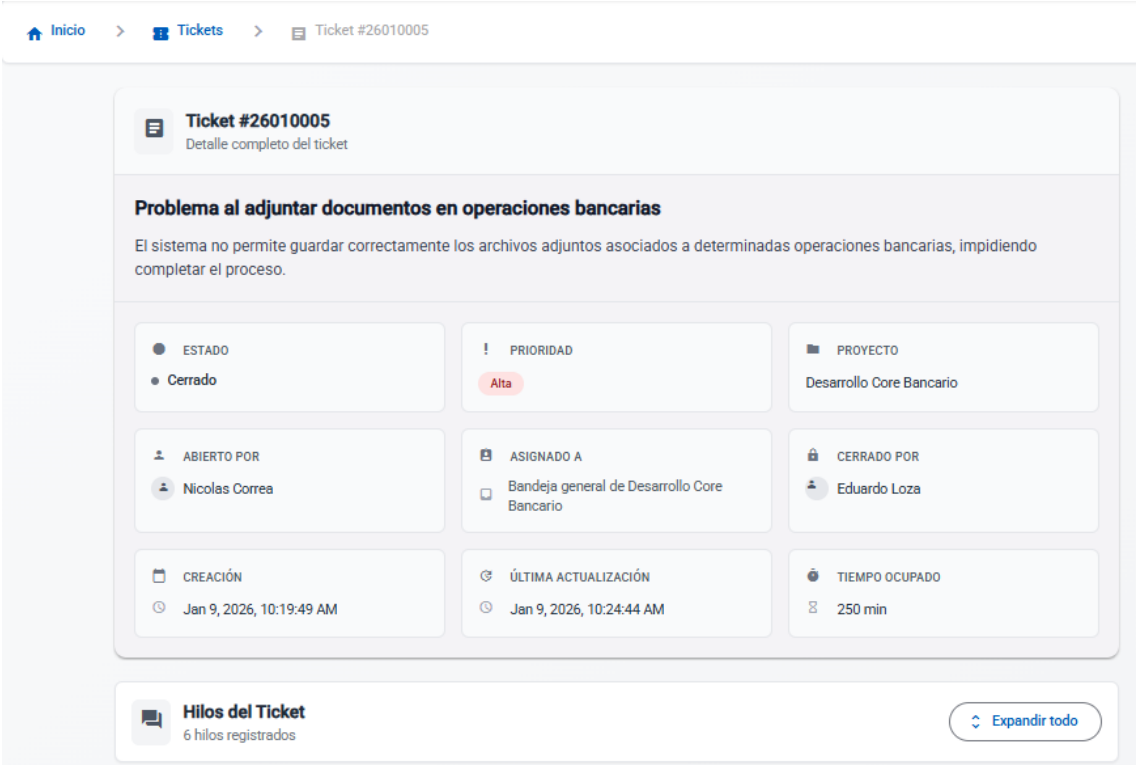

Gestionar Usuario
Activo
Nicolas Correa
Correo Electrónico
nicolas.correa@gmail.com
Cédula
1776716543
Rol *
Cliente
Puesto *
Cliente Final
Nombre *
Nicolas
Apellido *
Correa
Actualizar Usuario
Bloquear Usuario

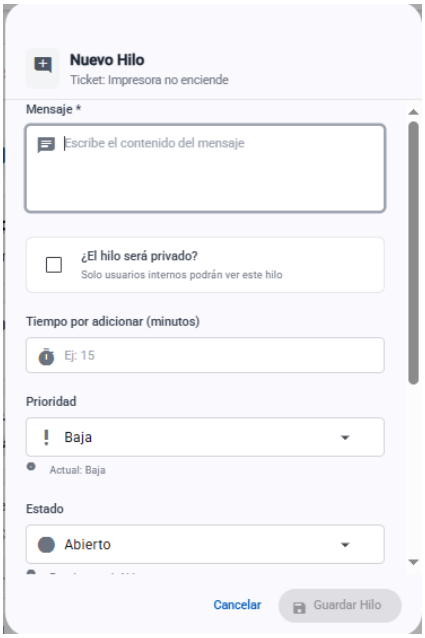

Gestión de Proyectos
Administra y gestiona los proyectos del sistema
Nuevo Proyecto
Buscar proyectos
Filtra por nombre, descripción, creador, etc.
Presiona Enter para buscar. Se buscará en todos los campos visibles.
NOMBRE DEL PROYECTO
DESCRIPCIÓN
CREADO POR
USUARIOS
ACCIONES
Proyecto TI Interno
Gestión y soporte de la infraestructura tecnológica interna, incluyendo mantenimiento de servidores, redes y estaciones de trabajo.
Eduardo Loza
Jan 9, 2026
15 usuarios
Editar
Desarrollo Core Bancario
Desarrollo y mantenimiento del sistema central bancario para la gestión de cuentas, transacciones financieras y procesos críticos.
Nicolas Correa
Jan 9, 2026
15 usuarios
Editar
Plataforma de Mesa de Ayuda
Implementación y soporte de una plataforma de service desk para la atención de incidencias y solicitudes de usuarios.
Juan Perez
Jan 9, 2026
15 usuarios
Editar
Sistema de Gestión Documental
Desarrollo de un sistema para el almacenamiento, control de versiones y aprobación de documentos corporativos.
Maria Gomez
Jan 9, 2026
3 usuarios
Editar
Aplicación Móvil Corporativa
Creación de una aplicación móvil para acceso a servicios internos y notificaciones corporativas.
Carlos Mendoza
Jan 9, 2026
3 usuarios
Editar
Migración a la Nube
Migración de servidores y aplicaciones on-premise a una infraestructura en la nube para mejorar escalabilidad y disponibilidad.
Ana Rodriguez
Jan 9, 2026
3 usuarios
Editar
Seguridad Informática
Implementación de controles de seguridad, monitoreo de amenazas y respuesta ante incidentes de ciberseguridad.
Luis Hernandez
Jan 9, 2026
3 usuarios
Editar
ERP Financiero
Implementación y soporte de un sistema ERP para la gestión financiera, contable y presupuestaria.
Pedro Sanchez
Jan 9, 2026
3 usuarios
Editar

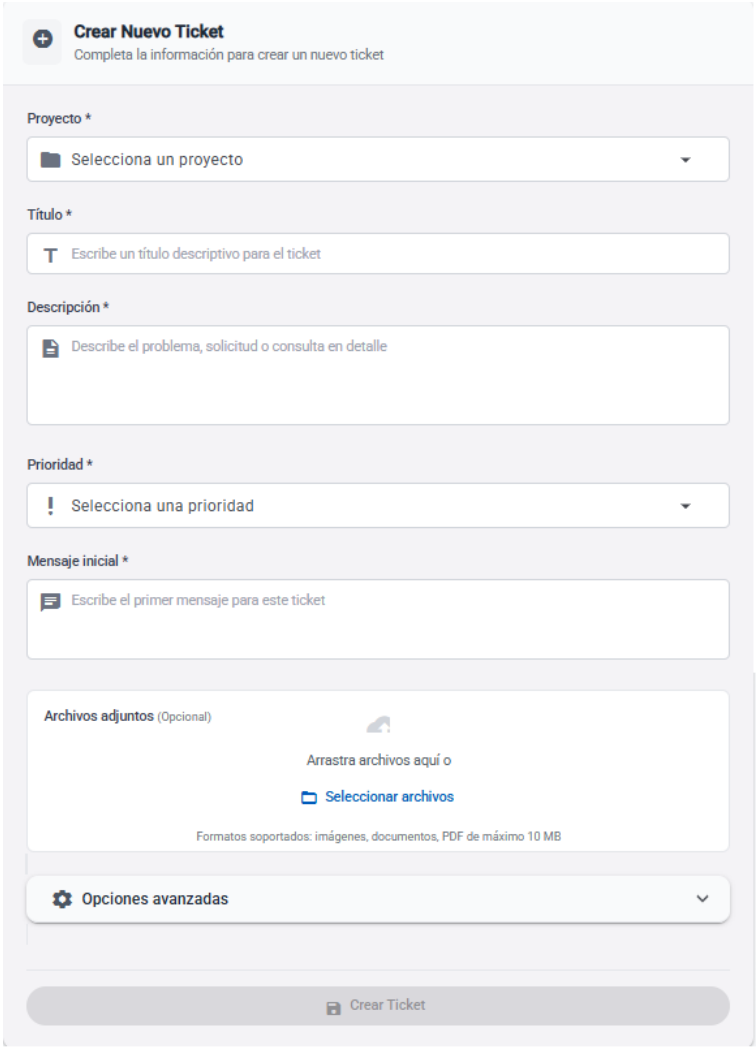

Crear Nuevo Proyecto
Completa la información para crear un nuevo proyecto
Nombre del Proyecto *
Escribe un nombre para el proyecto
Descripción *
Describe el proyecto, objetivos, alcance, etc.
Agregar Usuarios * (Mínimo 1 usuario)
Buscar usuario por nombre, apellido o puesto...
No hay usuarios seleccionados. Busca y agrega al menos un usuario.
Niveles de Servicio (SLA)
Configura los acuerdos de nivel de servicio para este proyecto
Primera Respuesta
Tiempo Máximo de Resolución
Tiempo Efectivo
Bolsa de Horas
Crear Proyecto

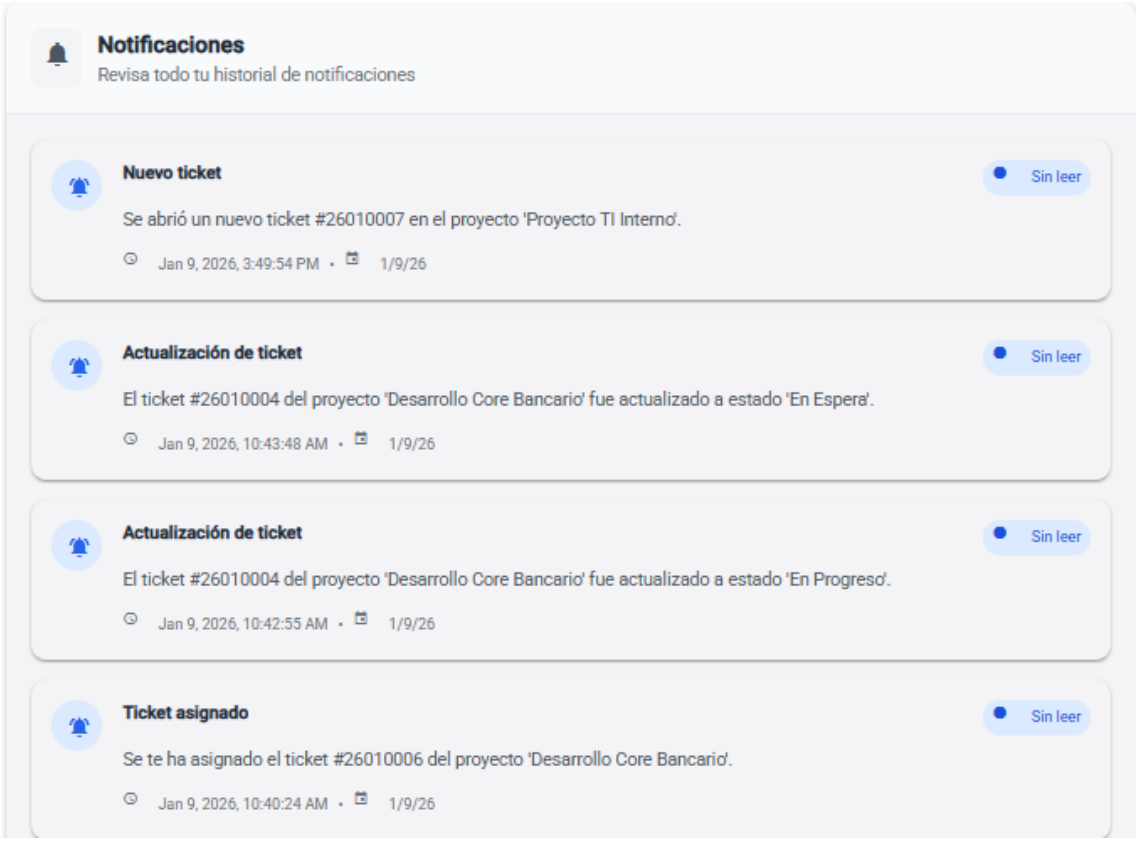

Editar Proyecto
Actualiza la información del proyecto
Nombre del Proyecto *
Desarrollo Core Bancario
Descripción *
Desarrollo y mantenimiento del sistema central bancario para la gestión de cuentas, transacciones financieras y procesos críticos.
Agregar Usuarios * (Mínimo 1 usuario)
Buscar usuario por nombre, apellido o puesto...
Usuarios seleccionados (15)
Eduardo Loza - Administrador del Sistema
Nicolas Correa - Cliente Final
Juan Perez - Administrador del Sistema
Maria Gomez - Administrador del Sistema
Carlos Mendoza - Administrador TI
Ana Rodriguez - Administrador de Plataforma
Luis Hernandez - Administrador General
Pedro Sanchez - Project Manager
Laura Morales - Coordinador de Proyectos
Diego Ortiz - Jefe de Proyectos
Primera Respuesta
Tiempo máximo para primera respuesta
5
minutos
Tiempo Máximo de Resolución
Tiempo máximo para resolver tickets
10
días
Tiempo Efectivo
Tiempo efectivo de trabajo por ticket
10
horas
Bolsa de Horas
Total de horas disponibles para el proyecto
250
horas
Actualizar Proyecto

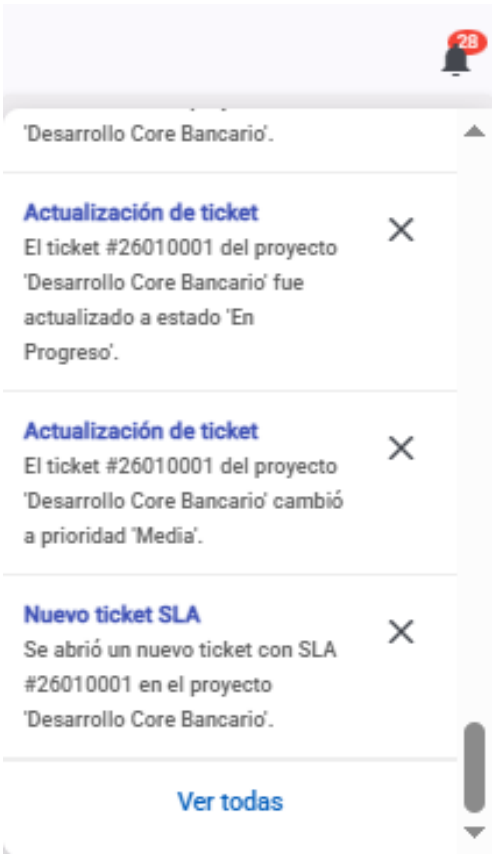

Tickets
Listado de tickets del sistema
+ Abrir Ticket
Buscar tickets
Filtra por número, título, estado, prioridad, etc.
Filtros Avanzados
Proyecto
Selecciona un proyecto
Asignado a
Selecciona un usuario
Creado por
Selecciona un usuario
Estado
Selecciona un estado
Prioridad
Selecciona una prioridad
Rango de fechas
Desde Hasta
Aplicar filtros
Borrar filtros
NÚMERO
TÍTULO
PRIORIDAD
ESTADO
ABIERTO POR
FECHA
26010007
Impresora no enciende
Baja
Abierto
Eduardo Loza
1/9/26, 3:49 PM
26010006
Existe error 500 del servidor
Baja
En Progreso
Nicolas Correa
1/9/26, 10:27 AM

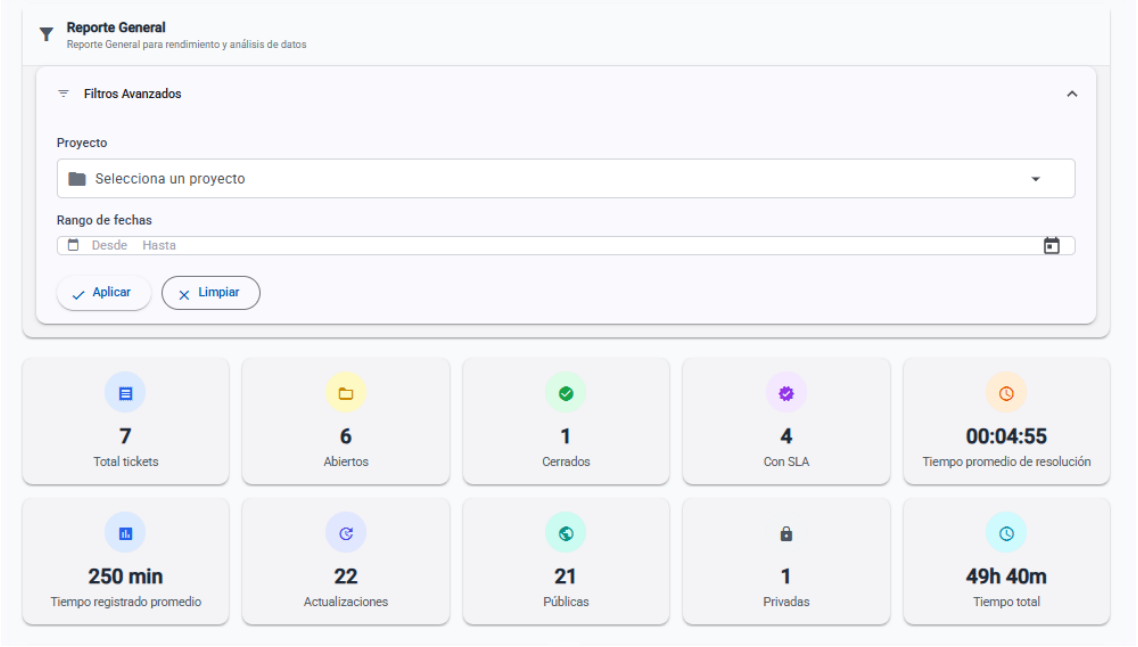

Inicio
Tickets
Ticket #26010005
Ticket #26010005
Detalle completo del ticket
Problema al adjuntar documentos en operaciones bancarias
El sistema no permite guardar correctamente los archivos adjuntos asociados a determinadas operaciones bancarias, impidiendo completar el proceso.
ESTADO
Cerrado
PRIORIDAD
Alta
PROYECTO
Desarrollo Core Bancario
ABIERTO POR
Nicolas Correa
ASIGNADO A
Bandeja general de Desarrollo Core Bancario
CERRADO POR
Eduardo Loza
CREACIÓN
Jan 9, 2026, 10:19:49 AM
ÚLTIMA ACTUALIZACIÓN
Jan 9, 2026, 10:24:44 AM
TIEMPO OCUPADO
250 min
Hilos del Ticket
6 hilos registrados
Expandir todo

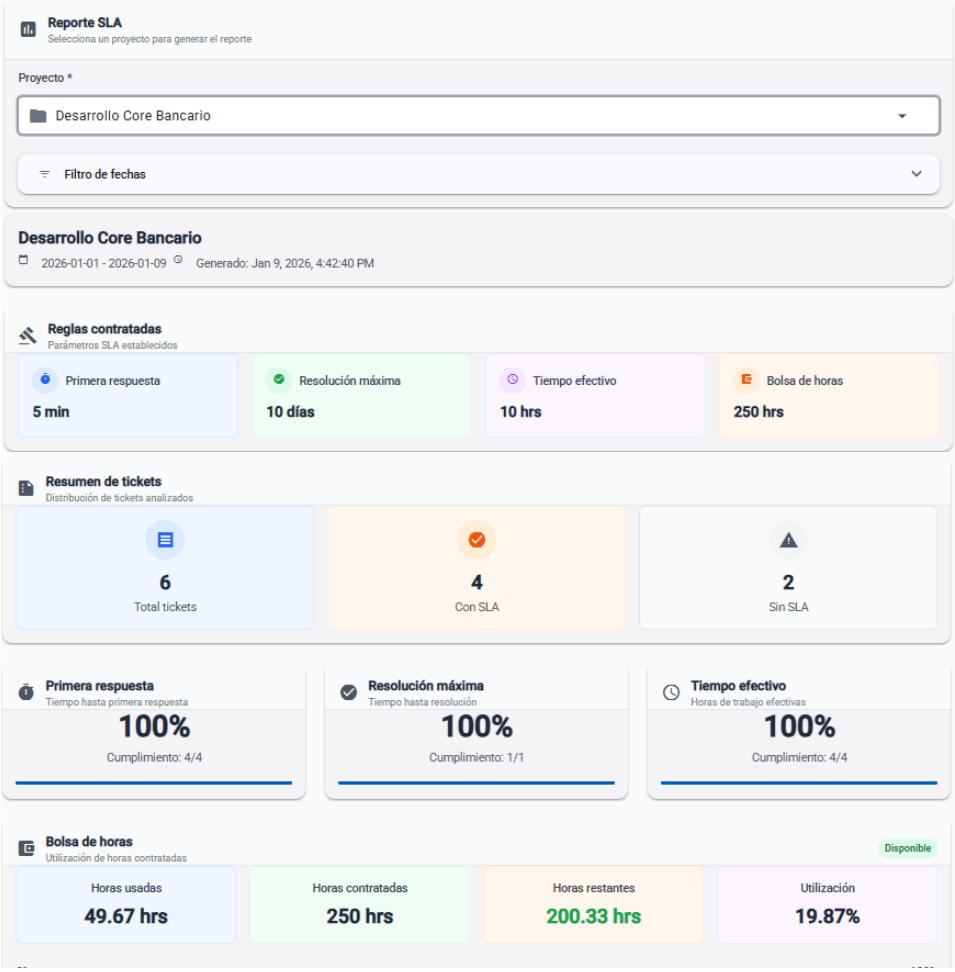

Nuevo Hilo
Ticket: Impresora no enciende
Mensaje *
Escribe el contenido del mensaje
¿El hilo será privado?
Solo usuarios internos podrán ver este hilo
Tiempo por adicionar (minutos)
Ej: 15
Prioridad
Baja
Actual: Baja
Estado
Abierto
Cancelar
Guardar Hilo


Crear Nuevo Ticket
Completa la información para crear un nuevo ticket
Proyecto *
Selecciona un proyecto
Título *
Escribe un título descriptivo para el ticket
Descripción *
Describe el problema, solicitud o consulta en detalle
Prioridad *
Selecciona una prioridad
Mensaje inicial *
Escribe el primer mensaje para este ticket
Archivos adjuntos (Opcional)
Arrastra archivos aquí o
Seleccionar archivos
Formatos soportados: imágenes, documentos, PDF de máximo 10 MB
Opciones avanzadas
Crear Ticket


Notificaciones
Revisa todo tu historial de notificaciones
Nuevo ticket
Sin leer
Se abrió un nuevo ticket #26010007 en el proyecto 'Proyecto TI Interno'.
Jan 9, 2026, 3:49:54 PM • 1/9/26
Actualización de ticket
Sin leer
El ticket #26010004 del proyecto 'Desarrollo Core Bancario' fue actualizado a estado 'En Espera'.
Jan 9, 2026, 10:43:48 AM • 1/9/26
Actualización de ticket
Sin leer
El ticket #26010004 del proyecto 'Desarrollo Core Bancario' fue actualizado a estado 'En Progreso'.
Jan 9, 2026, 10:42:55 AM • 1/9/26
Ticket asignado
Sin leer
Se te ha asignado el ticket #26010006 del proyecto 'Desarrollo Core Bancario'.
Jan 9, 2026, 10:40:24 AM • 1/9/26


28
'Desarrollo Core Bancario'.
Actualización de ticket
El ticket #26010001 del proyecto 'Desarrollo Core Bancario' fue actualizado a estado 'En Progreso'.
Actualización de ticket
El ticket #26010001 del proyecto 'Desarrollo Core Bancario' cambió a prioridad 'Media'.
Nuevo ticket SLA
Se abrió un nuevo ticket con SLA #26010001 en el proyecto 'Desarrollo Core Bancario'.
Ver todas


Reporte General
Reporte General para rendimiento y análisis de datos
Filtros Avanzados
Proyecto
Selecciona un proyecto
Rango de fechas
Desde
Hasta
Aplicar
Limpiar
7
Total tickets
6
Abiertos
1
Cerrados
4
Con SLA
00:04:55
Tiempo promedio de resolución
250 min
Tiempo registrado promedio
22
Actualizaciones
21
Públicas
1
Privadas
49h 40m
Tiempo total


Reporte SLA
Selecciona un proyecto para generar el reporte
Proyecto *
Desarrollo Core Bancario
Filtro de fechas
Desarrollo Core Bancario
2026-01-01 - 2026-01-09
Generado: Jan 9, 2026, 4:42:40 PM
Reglas contratadas
Parámetros SLA establecidos
Primera respuesta
5 min
Resolución máxima
10 días
Tiempo efectivo
10 hrs
Bolsa de horas
250 hrs
Resumen de tickets
Distribución de tickets analizados
6
Total tickets
4
Con SLA
2
Sin SLA
Primera respuesta
Tiempo hasta primera respuesta
100%
Cumplimiento: 4/4
Resolución máxima
Tiempo hasta resolución
100%
Cumplimiento: 1/1
Tiempo efectivo
Horas de trabajo efectivas
100%
Cumplimiento: 4/4
Bolsa de horas
Utilización de horas contratadas
Disponible
Horas usadas
49.67 hrs
Horas contratadas
250 hrs
Horas restantes
200.33 hrs
Utilización
19.87%