Desarrollo de la aplicación Moises Fase III

Tatiana Elizabeth Sánchez Sanin

Práctica empresarial para optar al título de Ingeniera de Sistemas otorgado por UdeA

Orientadores
Sandra Patricia Zabala Orrego, Ingeniera Informática especialista en Gerencia
Jhoan Esteban Patiño Ciro, Administrador de Empresas

Universidad de Antioquia
Facultad de Ingeniería
Pregrado en Ingeniería de Sistemas Virtual
Medellín
2024

Dedicatoria

Con profunda gratitud dedico este trabajo a mis padres, quienes con su infinita paciencia me han enseñado la importancia de la perseverancia y el esfuerzo constante.

Agradecimientos

Quiero expresar mi más sincero agradecimiento a mis asesores, Jhoan Patiño Ciro en representación de SETI S.A.S y a la profesora Sandra Zabala, cuya paciencia, dedicación y conocimientos fueron fundamentales para la culminación exitosa de mis prácticas profesionales. Agradezco a mis padres, cuyo apoyo incondicional y aliento me motivaron a alcanzar esta meta. Asimismo, expreso mi gratitud a la Universidad de Antioquia por proporcionarme una formación de calidad y a la empresa SETI S.A.S por abrirme las puertas y permitirme realizar mis prácticas profesionales, experiencia que enriqueció enormemente este estudio.

TABLA DE CONTENIDO

LISTA DE FIGURAS

SIGLAS, ACRÓNIMOS Y ABREVIATURAS

UdeA	Universidad de Antioquia
S.A.S	Sociedad por Acciones Simplificadas
SETI	Servicios Especializados de Tecnología e Informática
Fig.	Figura
HTTPS	Protocolo Seguro de Transferencia de Hipertexto
SSL/TSL	Capa de Puertos Seguros/Seguridad de la capa de transporte
REST	Transferencia de Estado Representacional
Apps	Aplicaciones
Host	Anfitrión
UI	Interfaz de Usuario
PC	Computadora Personal
DB	Base de Datos
ASP	Servidor de Aplicaciones
SO	Sistema Operativo
HTTP	Protocolo de Transferencia de Hipertexto
JSON	Notación de Objetos de JavaScript
XML	Lenguaje de Marcado Extensible
SPAs	Aplicación de página única
V.	Versión
NodeJS	Entorno en tiempo de ejecución multiplataforma
MVC	Modelo-Vista-Controlador
SGBD	Sistema de Gestión de Bases de Datos
ORG	Organización
DevOps	Acrónimo de Development y Operations

RESUMEN

En este documento se realiza la presentación del desarrollo de una aplicación web de uso interno en la empresa SETI S.A.S, la cual se dedica a ofrecer servicios tecnológicos a diferentes clientes a lo largo del país. Entre los servicios ofrecidos se encuentra el desarrollo de aplicaciones a la medida para soluciones internas y para las solicitudes de los clientes.

El proyecto tiene como nombre Moises, el cual en la fase III de desarrollo tiene como objetivo la mejora de la experiencia de usuario y el proceso de ejecución y reporte de las métricas necesarias para el buen funcionamiento de los servicios prestados. Se realizaron mejoras a la aplicación de usabilidad, accesibilidad y modernización, siguiendo buenas prácticas de programación y del diseño de la arquitectura propuesto por el equipo encargado.

En el desarrollo del proyecto se utilizó el marco de trabajo Scrum, lo que permitió un proceso interactivo e incremental, facilitando la adaptación a los cambios y la mejora continua del producto. Al finalizar el proyecto se logró cumplir con los requisitos solicitados por los stackeholders, proporcionando una herramienta eficiente y alineada con las necesidades de la empresa y sus usuarios. La aplicación resultante no solo optimiza la gestión interna de SETI S.A.S., sino también potencia la calidad de los servicios ofrecidos a sus clientes.

Palabras clave — **Aplicación web, desarrollo web, usabilidad, experiencia de usuario, desarrollo de software.**

ABSTRACT

This document presents the development of an internal web application for SETI S.A.S., a company dedicated to providing technological services to various clients throughout the country. Among the services offered is the development of custom applications for internal solutions and for customer requests.

The project is named Moises, and in its phase III of development, the objective is to improve the use experience and the process of executing and reporting the metrics necessary for the proper functioning of the services provided. Improvements were made to the application's usability, accessibility, and modernization, following the good programming practices and the architectural design proposed by the responsible team.

Scrum framework was used in the project development, which allowed for an interactive and incremental process, facilitating adaptation to changes and continuous product improvement. Upon completion of the project, the requirements requested by the stakeholders were met, providing an efficient tool aligned with the needs of the company and its users. The result application not only optimizes SETI S.A.S.'s internal management but also enhances the quality of the technological services offered to its clients.

Keywords — **Web application, web development, usability, user experience, software development.**

I. INTRODUCCIÓN

Este documento presenta el desarrollo de una aplicación web de uso interno para la empresa SETI S.A.S. (Fig. 1), la cual se dedica a ofrecer servicios tecnológicos a diversos clientes a nivel nacional. La creación de esta aplicación, denominada "Moises", surge de la necesidad de mejorar la gestión interna y optimizar la calidad de los servicios tecnológicos prestados. En particular, se detectó la falta de una herramienta eficiente para la ejecución y reporte de métricas críticas para el funcionamiento de los servicios, lo que afectaba tanto la experiencia de los usuarios internos como la satisfacción de los clientes.

Fig. 1. Logo de SETI S.A.S.

Nota: https://seti.com.co

El objetivo principal del proyecto "Moises" en su fase III es mejorar la experiencia de usuario, así como modernizar la interfaz y la accesibilidad de la aplicación. Esto incluye la implementación de mejores prácticas en programación y diseño arquitectónico, lo cual es esencial para garantizar una solución moderna y efectiva.

La justificación de este proyecto radica en la necesidad de adaptar las herramientas internas de SETI S.A.S. a las exigencias actuales del mercado, permitiendo una mejor gestión y monitoreo de los servicios ofrecidos. Para ello, se empleó el marco de trabajo Scrum, que facilitó un proceso iterativo e incremental, promoviendo una continua adaptación y mejora del producto. Con esta metodología, se logró cumplir con los requisitos establecidos por los stakeholders, resultando en una herramienta alineada con las necesidades de la empresa y sus usuarios.

II. PLANTEAMIENTO DEL PROBLEMA

La aplicación Moises en su estado actual presenta algunas limitaciones que afectan la experiencia del usuario y la eficiencia del negocio. Estas limitaciones incluyen:

- Funcionalidad limitada para la creación y edición de clientes: El proceso actual es manual y repetitivo, lo que consume tiempo y aumenta el riesgo de errores.
- Diseño poco intuitivo: La interfaz actual no es fácil de usar, lo que dificulta la navegación y la comprensión de la información.
- Falta de automatización en la gestión de métricas: La generación de informes y el análisis de datos se realizan manualmente, lo que es un proceso lento e ineficiente.
- Seguridad de la información: La aplicación actual no cuenta con medidas de seguridad robustas para proteger la información confidencial de los usuarios.

Estas limitaciones impactan negativamente en la experiencia del usuario, la eficiencia del negocio y la toma de decisiones.

El proyecto Moises fase III busca abordar estas limitaciones a través de la modernización de la aplicación, la mejora del diseño, la automatización de la gestión de métricas, la implementación de medidas de seguridad y la optimización del proceso de toma de decisiones.

A. Antecedentes

El diagrama de componentes de una aplicación muestra cómo se organizan y conectan los diferentes componentes de la aplicación [1]. Cada componente es un módulo individual que realiza una función específica. Los componentes están conectados entre sí mediante interfaces, que definen cómo pueden interactuar entre sí.

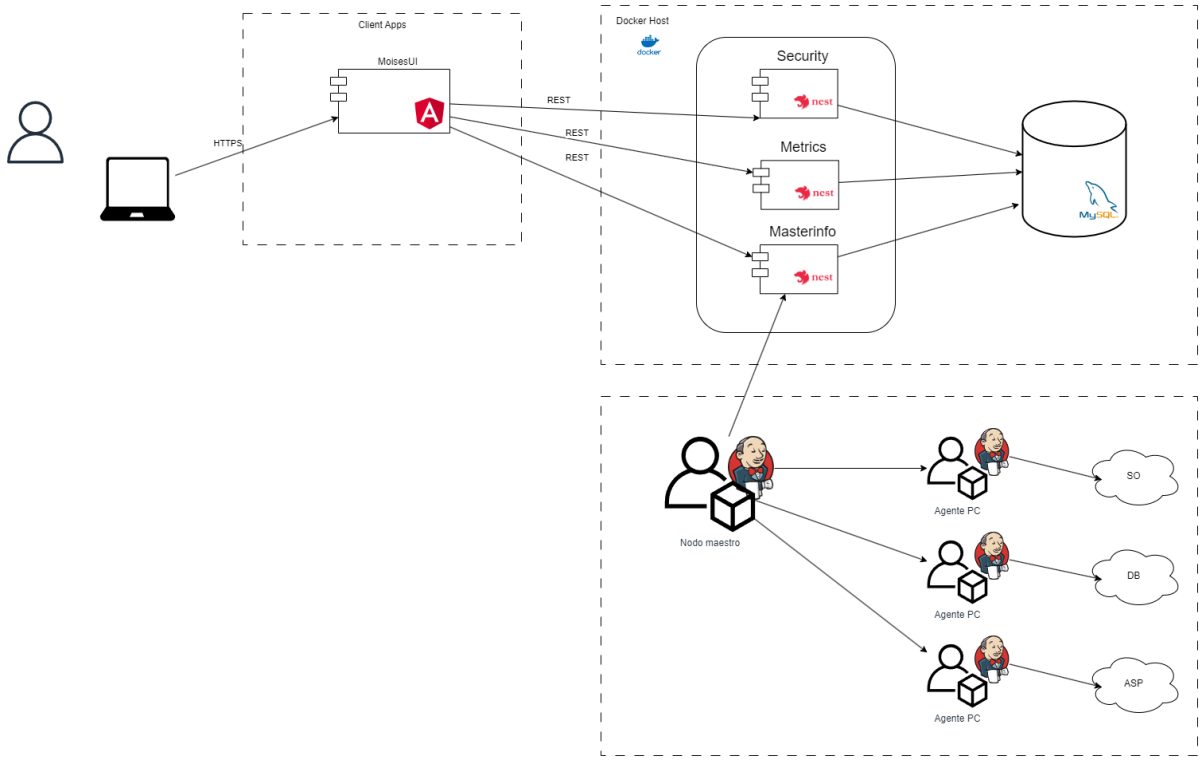

Fig. 2. Diagrama de arquitectura de Moises

Componentes del diagrama

El diagrama de componentes (Fig. 2.) muestra una aplicación compuesta por los siguientes componentes:

- Client Apps: Este componente representa las aplicaciones cliente que utilizan la aplicación. Las aplicaciones cliente pueden ser aplicaciones web, aplicaciones móviles o aplicaciones de escritorio.
- Docker Host: Este componente es el servidor que alberga los contenedores Docker. Los contenedores Docker son unidades autónomas de software que contienen el código de la aplicación y sus dependencias.
- MoisesUI: Este componente es el módulo principal de la aplicación. Es responsable de la interfaz de usuario y permitir que la aplicación esté disponible.
- Security: Este componente es un microservicio de backend responsable de la seguridad de la aplicación. Proporciona funciones como autenticación, autorización y cifrado.

- Masterinfo: Este componente es un microservicio de backend responsable de las funcionalidades principales de la aplicación.
- Metrics: Este componente es el encargado de la lógica del negocio de la aplicación y de las funcionalidades relacionadas a las métricas.
- Nest: Este componente es un marco de desarrollo web para JavaScript. Se utiliza para desarrollar el backend de la aplicación.
- Angular: Es un framework de desarrollo para JavaScript. Se utiliza para desarrollar el frontend de la aplicación
- Nodo maestro: Este componente es el nodo Jenkins central de la aplicación. Es responsable de gestionar los demás nodos y de proporcionar servicios a las aplicaciones cliente.
- Agente PC: Este componente es un agente que se ejecuta en los ordenadores de los usuarios. Es responsable de recopilar datos y enviarlos al servidor.
- DB: Este componente es una base de datos contratada por un cliente, en la qué se almacenan datos.
- ASP: Este componente es una aplicación web perteneciente a un cliente.
- SO: Este componente es un sistema operativo contratado cómo un servicio por un cliente.

Conexiones entre componentes

Los componentes del diagrama están conectados entre sí mediante las siguientes interfaces:

- HTTPS: HTTPS (Hypertext Transfer Protocol Secure) es un protocolo de red seguro que se utiliza para la comunicación entre un navegador web y un servidor web. HTTPS utiliza cifrado SSL/TLS para proteger la privacidad y la integridad de los datos transmitidos. Esto significa que los datos no pueden ser interceptados y leídos por terceros. HTTPS es esencial para proteger la información confidencial que se transmite en línea.
- REST: REST (Representational State Transfer) es un estilo arquitectónico para el diseño de interfaces de red, principalmente en el contexto de aplicaciones web [2]. REST no define un conjunto específico de protocolos o tecnologías, sino que

proporciona un conjunto de principios y directrices que los desarrolladores pueden seguir al diseñar interfaces. Las interfaces RESTful suelen ser fáciles de usar y escalables, y pueden ser implementadas utilizando una variedad de tecnologías, incluyendo HTTP, JSON y XML.

III. JUSTIFICACIÓN

El desarrollo del proyecto Moises responde a la necesidad de mejorar la eficiencia operativa y la experiencia del usuario dentro de la empresa SETI S.A.S., así como de optimizar la calidad de los servicios tecnológicos ofrecidos a los clientes. La selección de este tema se fundamenta en la creciente demanda de soluciones tecnológicas personalizadas y en la importancia de contar con herramientas internas que estén alineadas con las exigencias del mercado actual.

El interés en este tema radica en su relevancia para el entorno empresarial contemporáneo, donde la tecnología juega un papel crucial en la competitividad y en la capacidad de respuesta a las necesidades del cliente. Con el desarrollo de Moises, se busca no solo satisfacer los requisitos internos de SETI S.A.S., sino también establecer un estándar de calidad y eficiencia que pueda ser replicado en otras áreas de la empresa e incluso en proyectos futuros.

El aporte principal del proyecto radica en la implementación de una solución tecnológica que mejora la gestión interna, optimiza los procesos y proporciona una herramienta escalable y adaptable a los cambios. Además, este proyecto ofrece una oportunidad para aplicar y validar metodologías ágiles como Scrum en un entorno real, demostrando su eficacia en la entrega de valor incremental y en la mejora continua del producto.

IV. OBJETIVOS

A. Objetivo general

Modernizar y mejorar la accesibilidad y usabilidad de la aplicación Moises para optimizar la experiencia de usuario y el proceso de gestión de métricas e información en SETI S.A.S.

B. Objetivos específicos

1. Modernizar la funcionalidad de creación y edición de clientes:
    a. Implementar una interfaz gráfica intuitiva y moderna.
    b. Incluir opciones de búsqueda y filtrado para facilitar la gestión de clientes.
    c. Automatizar tareas repetitivas para agilizar el proceso de creación y edición.
    d. Integrar la funcionalidad con otros sistemas de la empresa.
2. Mejorar el diseño de la aplicación:
    a. Utilizar la paleta de colores definida por la organización.
    b. Mejorar la organización y la legibilidad del contenido.
    c. Incluir elementos visuales que faciliten la comprensión de la información.
3. Optimizar el proceso de gestión de métricas:
    a. Permitir la visualización de las métricas en tiempo real según los parámetros de búsqueda.
4. Implementar un plan de pruebas para garantizar la calidad de la aplicación:
    a. Realizar pruebas unitarias, de integración y aceptación.
    b. Identificar y corregir errores y bugs.
    c. Garantizar la compatibilidad de la aplicación con diferentes navegadores y dispositivos.
5. Documentar la aplicación para facilitar su uso y mantenimiento:
    a. Elaborar manuales de usuario y guías de instalación.
    b. Incluir comentarios en el código fuente de la aplicación.
    c. Brindar soporte técnico a los usuarios de la aplicación.
6. Implementar mejoras y actualizaciones a la aplicación de forma continua:

   a. Incorporar nuevas funcionalidades en base a las necesidades de los usuarios.
   b. Corregir errores y bugs que se puedan presentar.
7. Asegurar la seguridad de la información de la aplicación:
   a. Implementar medidas de seguridad para proteger la información de los usuarios.
   b. Realizar pruebas de seguridad y vulnerabilidades.
8. Gestionar el proyecto de forma eficiente:
   a. Seguir el cronograma de actividades propuesto.
   b. Comunicar el progreso del proyecto a las partes interesadas.
   c. Gestionar los riesgos y las contingencias del proyecto.

V. MARCO TEÓRICO

El presente marco teórico tiene como objetivo contextualizar el desarrollo de Moises como aplicación web empresarial, utilizando las tecnologías Angular, NestJS, Base de Datos relacional MySQL y su correspondiente despliegue en Kubernetes. Se abordarán las características principales de estas tecnologías y sus ventajas en el desarrollo de aplicaciones empresariales para construir una solución de software completa.

Angular (Fig. 3) es un framework de JavaScript open-source desarrollado por Google [3]. Se basa en TypeScript, un superset de JavaScript que añade características cómo tipado estático y el uso de clases. Se utiliza principalmente para crear aplicaciones web de una sola página (SPAs) que son altamente interactivas y dinámicas. Su popularidad reside en su robusta arquitectura, así como en su completa biblioteca de herramientas y su enfoque en la eficiencia del desarrollo.

Fig. 3. Logo de Angular V.17.

Nota: fuente https://angular.dev

Angular es el framework predilecto para el desarrollo de aplicaciones empresariales gracias a su escalabilidad que permite crear aplicaciones robustas y escalables que pueden soportar grandes cantidades de usuarios y datos, y gracias a su arquitectura modular facilita el mantenimiento y la actualización de las aplicaciones a largo plazo. Este framework ofrece características de seguridad integradas que protegen las aplicaciones de vulnerabilidades y ataques. La gran cantidad de herramientas y bibliotecas que posee ayuda a los desarrolladores a ser más productivos y eficientes [4].

Entre los beneficios más importantes se destaca la facilidad de aprendizaje, la creación de componentes reutilizables que se pueden usar en diferentes partes de la aplicación, y ya que está cuenta con un módulo de enrutamiento es posible crear aplicaciones web con diferentes rutas y vistas de forma sencilla. La inyección de dependencias también es posible, lo que facilita el desarrollo y las pruebas.

Angular, combinado con RxJS, permite crear interfaces de usuario altamente reactivas y dinámicas [5]. RxJS proporciona un enfoque declarativo para manejar flujos de datos asíncronos, lo que facilita la gestión de eventos como clics de botones, cambios en formularios y respuestas de servicios [6]. Por ejemplo, se puede utilizar RxJS para implementar una búsqueda en tiempo real que actualice los resultados a medida que el usuario escribe, creando una experiencia de usuario más fluida y responsiva.

Para garantizar una experiencia de usuario intuitiva y visualmente atractiva, se utiliza Angular Material. Esta biblioteca proporciona una amplia gama de componentes prediseñados que siguen las pautas de Material Design, lo que asegura una apariencia consistente y profesional [7]. Por ejemplo, se utilizaron las tablas de Angular Material para mostrar los datos de forma clara y organizada, y los formularios para facilitar la interacción del usuario con la aplicación. Los iconos de Angular Material son una parte fundamental de esta biblioteca, permitiendo personalizar la interfaz de usuario con una amplia variedad de símbolos y gráficos [8].

Por otro lado, NestJS (Fig. 4) es un framework de JavaScript open-source desarrollado con TypeScript. Se basa en NodeJS y está inspirado en Angular, Spring y Express [9]. Este es principalmente utilizado para crear aplicaciones web del lado del servidor que son altamente escalables y modulares. Algunas de las ventajas de este framework son la creación de aplicaciones web robustas y escalables. La arquitectura modular de NestJS facilita el desarrollo, el mantenimiento y la actualización de las aplicaciones a lo largo del tiempo, también implementa patrones de diseño conocidos como MVC e inyección de dependencias, lo que permite la creación de código limpio y reutilizable. Asimismo, NestJS facilita la escritura de pruebas unitarias y de integración, lo que ayuda a garantizar la calidad del código [2].

Fig. 4. Logo de NestJS.

Nota: fuente https://cdn.icon-icons.com Esta es una página dedicada a la creación de íconos y logos de diversas tecnologías y entidades.

NestJS, al estar inspirado en Angular y Spring, fomenta la adopción de los principios SOLID. Se aplican estos principios para crear un código más limpio y mantenible, por ejemplo, cada controlador en NestJS se encargará de una única responsabilidad, como manejar solicitudes HTTP específicas. Esto facilita la comprensión y el mantenimiento del código a largo plazo [10].

Continuando con el sistema de gestión de bases de datos (SGBD) relacional MySQL (Fig. 5). Este es open-source y desarrollado por Oracle Corporation [11]. Es uno de los SGBD más populares del mundo, utilizado por millones de empresas y organizaciones. Se caracteriza por su facilidad de uso, escalabilidad y rendimiento. MySQL se encuentra disponible en una variedad de plataformas y ofrece diversas herramientas para administrar bases de datos MySQL, como MySQL Workbench y MySQL Administrator [12]. También ofrece una serie de características de seguridad para proteger los datos, y al ser open-source es un software gratuito.

Fig. 5. Logo del motor relacional de base de datos MySQL.

Nota: fuente https://res.cloudinary.com

Docker (Fig. 6) es una plataforma de contenedores que permite a los desarrolladores empaquetar aplicaciones y sus dependencias en contenedores, asegurando que el software se ejecute de manera uniforme y consistente en cualquier entorno [13]. Docker se utiliza ampliamente en el desarrollo de aplicaciones modernas debido a su capacidad para facilitar la creación, implementación y ejecución de aplicaciones en contenedores, lo que mejora la eficiencia y la portabilidad del software [14]. Al utilizar Docker, es posible crear entornos de desarrollo y producción idénticos, lo que reduce problemas relacionados con la configuración del entorno. Además, la integración con Kubernetes permite una orquestación avanzada de contenedores, gestionando su despliegue, escalabilidad y mantenimiento de manera automática y eficiente [15].

Fig. 6. Logo de Docker.

Nota: fuente https://d1.awsstatic.com

Dadas estas tecnologías, es fundamental implementar pruebas unitarias y de integración para garantizar una alta calidad en el código desarrollado. Para este propósito, se utiliza Jest (Fig. 7), una herramienta de testing robusta y eficiente. Jest es una biblioteca de pruebas de JavaScript desarrollada por Facebook, ampliamente utilizada en la comunidad de desarrollo debido a su facilidad de configuración y su capacidad para trabajar con una variedad de frameworks y bibliotecas [16].

Fig. 7. Logo de Jest.

Nota: fuente https://jestjs.io

Jest es compatible con Angular y NestJS, lo que permite realizar pruebas exhaustivas tanto en el frontend como en el backend de la aplicación. Para Angular, Jest ofrece una configuración optimizada para pruebas de componentes, servicios y otros elementos de la interfaz de usuario. Además, su compatibilidad con TypeScript facilita la detección de errores tipográficos y de lógica, proporcionando una cobertura de pruebas más completa.

En el backend, Jest se integra perfectamente con NestJS, permitiendo la realización de pruebas unitarias e integración sobre controladores, servicios y módulos. La capacidad de Jest para realizar mocking de dependencias es crucial para aislar componentes individuales durante las pruebas, lo que ayuda a identificar problemas específicos sin interferencias de otros módulos. Jest también ofrece características avanzadas como la generación de informes de cobertura de código, lo cual es esencial para evaluar la efectividad de las pruebas y asegurar que todas las rutas críticas de código sean evaluadas. Por último, Jest es conocido por su rapidez y eficiencia, gracias a su capacidad de ejecución de pruebas en paralelo y al uso de un motor de pruebas altamente optimizado. Esto resulta especialmente beneficioso en proyectos grandes, donde la rapidez en la retroalimentación sobre el estado del código es crucial para mantener un flujo de desarrollo ágil [17].

En conclusión, el uso de Angular, NestJS, MySQL, Docker y Kubernetes conforma un stack tecnológico robusto para el desarrollo de aplicaciones web empresariales, ofreciendo una combinación de características que los hacen ideales para crear aplicaciones robustas, escalables y seguras, y la integración de Jest proporciona una infraestructura de pruebas integral y eficiente que, en conjunto con las tecnologías mencionadas anteriormente, asegura la calidad y fiabilidad de la aplicación.

VI. METODOLOGÍA

En el desarrollo del proyecto se ha implementado la metodología ágil Scrum (Fig. 8), caracterizada por su enfoque flexible e iterativo, lo que la hace ideal para proyectos con requisitos cambiantes e inciertos. Esta metodología se centra en la entrega de valor de manera incremental y continua al cliente, promoviendo la mejora continua y la adaptación a los cambios. Además, fomenta la transparencia y la colaboración entre los miembros del equipo, lo que resulta en una mayor visibilidad del progreso del proyecto y un entorno de trabajo cohesionado.

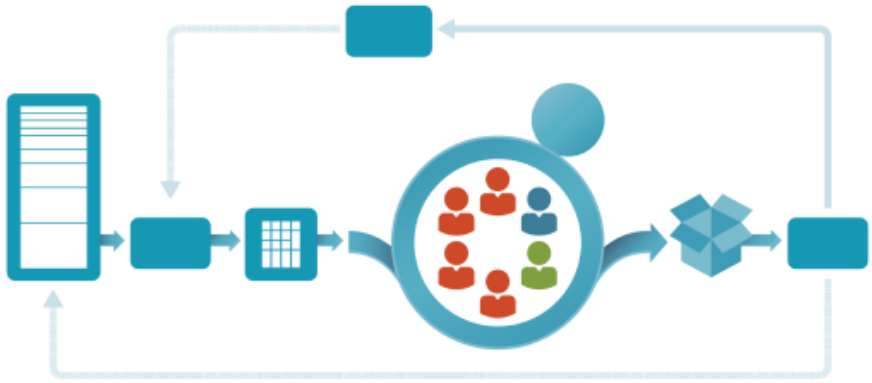

Fig. 8. Representación del marco Scrum.

Nota: fuente https://static.scrum.org

Algunos de los principios del marco Scrum son la entrega de valor incremental y continuo al cliente, la autoorganización de los equipos que toman decisiones de manera autónoma, la capacidad de adaptación a los cambios, la concentración en una meta específica durante cada iteración o sprint y el fomento de la colaboración y comunicación efectiva entre todos los miembros de equipo.

En el proyecto desarrollado, el equipo de trabajo está compuesto por:

- Un gerente de proyectos (Scrum Master), quien facilita el proceso Scrum y ayuda al equipo a trabajar eficazmente.
- Un arquitecto empresarial quien proporciona una base técnica para el desarrollo efectivo del software.

- Un analista funcional para los requisitos del negocio (Product Owner).
- Un analista funcional para los requisitos técnicos.
- Un líder técnico.
- Tres desarrolladores, responsables de la construcción del producto.

Artefactos de Scrum:

- Product Backlog: Lista priorizada de funcionalidades y características del producto.
- Sprint Backlog: Lista de tareas y funcionalidades a desarrollar en cada sprint.
- Sprint Increment; Producto funcional al final de cada sprint.

Eventos de Scrum:

- Sprint Planning: Planificación del sprint y definición del Sprint Backlog.
- Daily Scrum: Reuniones diarias para discutir el progreso y coordinar el trabajo.
- Sprint Review: Presentación del Sprint Increment al cliente y recepción de feedback.
- Sprint Retrospective: Reflexión sobre el sprint y búsqueda de mejoras para el próximo ciclo.

Para gestionar el proyecto se utilizan herramientas como Azure DevOps para la gestión de proyectos y Microsoft Teams para facilitar la comunicación del equipo. Los sprints tienen una duración de dos semanas, permitiendo iteraciones rápidas y adaptaciones continuas.

La implementación de Scrum en el proyecto Moises ha traído numerosos beneficios, como una mayor flexibilidad para adaptarse a los cambios en los requisitos, una entrega continua de valor al cliente, una mayor transparencia en el progreso del proyecto y una colaboración mejorada entre todos los miembros del equipo.

VII. RESULTADOS

Como parte del desarrollo de la aplicación Moises en su fase III se realizó la modernización de la funcionalidad de clientes y diseño de la aplicación. Se ha implementado una nueva interfaz gráfica intuitiva y moderna para el inicio de sesión de los usuarios (Fig. 9 y 10), tanto internos como externos, permitiendo que las personas puedan acceder a la información correspondiente a su cargo y cumpliendo con las medidas de seguridad.

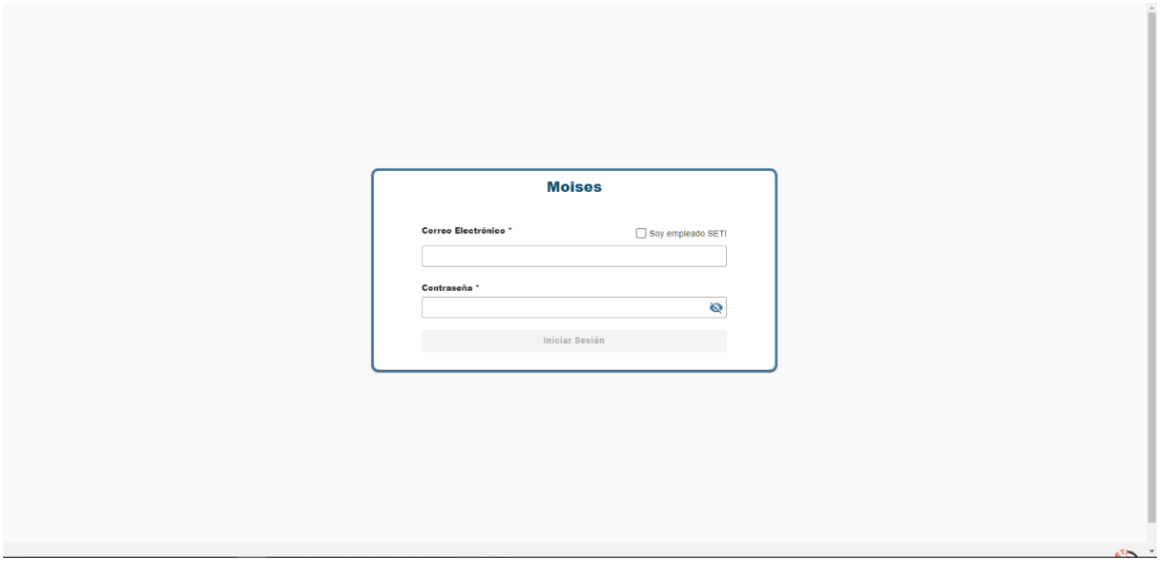

Fig. 9. Inicio de sesión para usuarios externos.

Fig. 10. Inicio de sesión para usuarios SETI.

Se ha mejorado la organización y legibilidad del contenido añadiendo una pantalla de inicio (Fig. 11 y 12) que facilita la comprensión de la información por parte de los usuarios. Se han incluido elementos visuales como gráficos e iconos que complementan la información textual y la hacen más atractiva y comprensible, aplicando la paleta de colores definida por la organización y mejorando la coherencia visual de la aplicación.

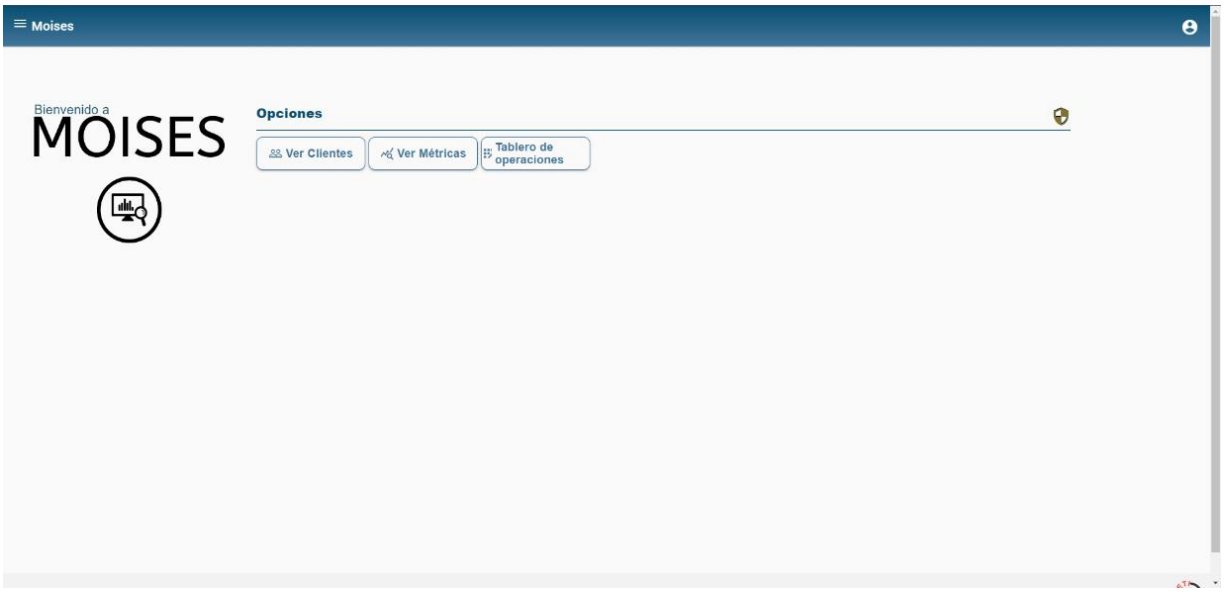

Fig. 11. Aplicación en colores azules.

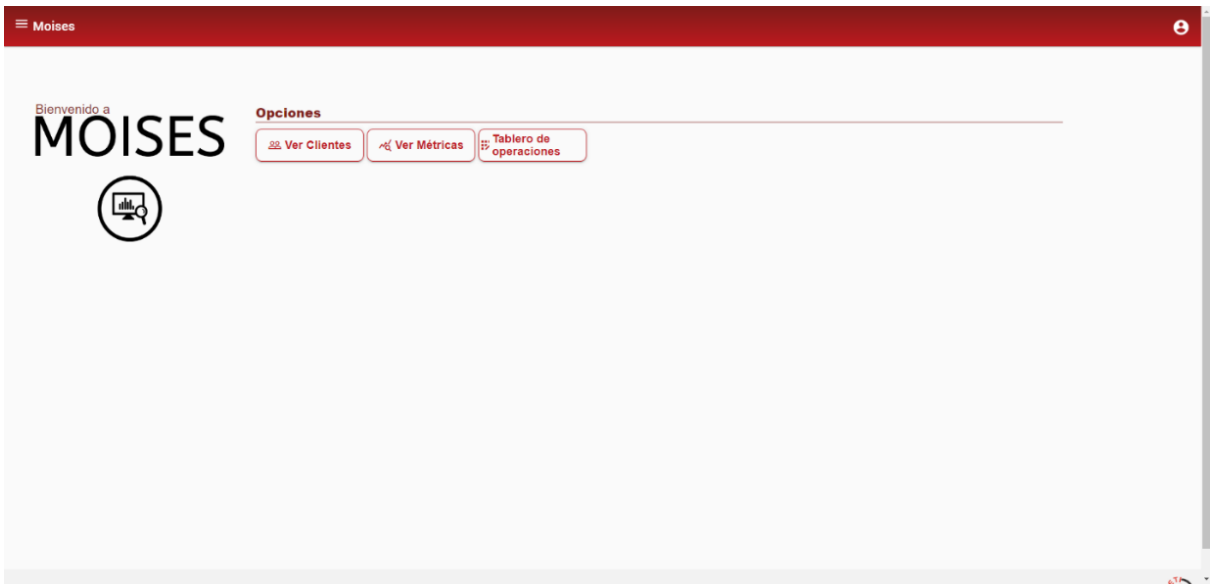

Fig. 12. Pantalla de inicio.

Además, se incorporó una barra lateral de búsqueda al lado izquierdo y una barra lateral a la derecha con la información del usuario (Fig. 13 y 14) para mejorar la experiencia del usuario, lo que permite a los usuarios navegar de manera fluida entre las diferentes pantallas de la aplicación y realizar configuraciones básicas en la aplicación como es el cambio de color y el cierre de sesión. Esta barra lateral, cuenta con un diseño acorde a los estándares y colores, incluye íconos personalizados para identificar cada pantalla de forma más certera.

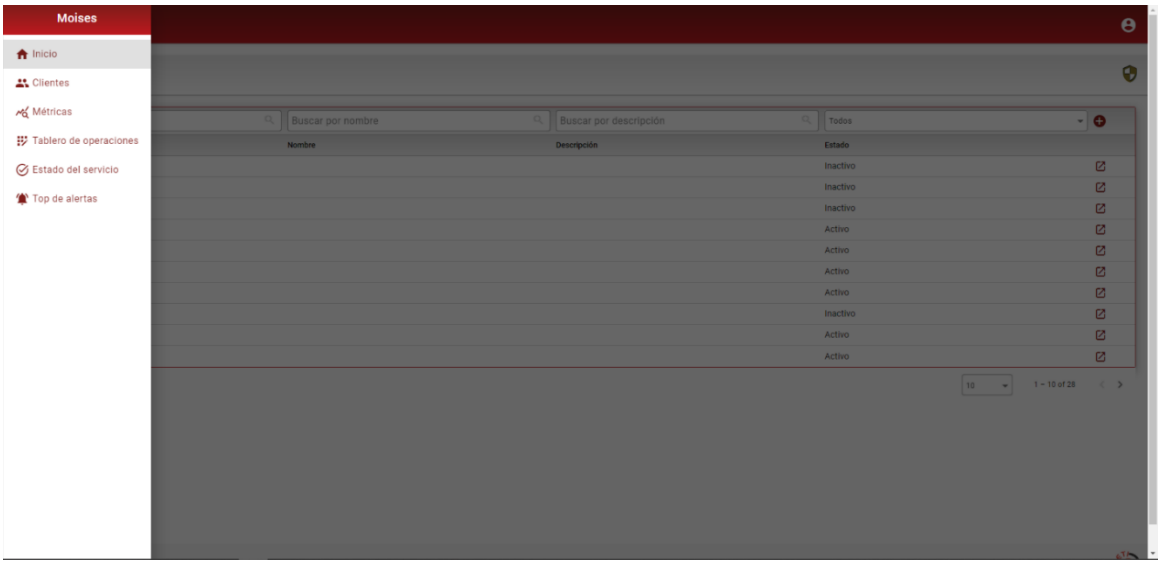

Fig. 13. Barra lateral de la aplicación.

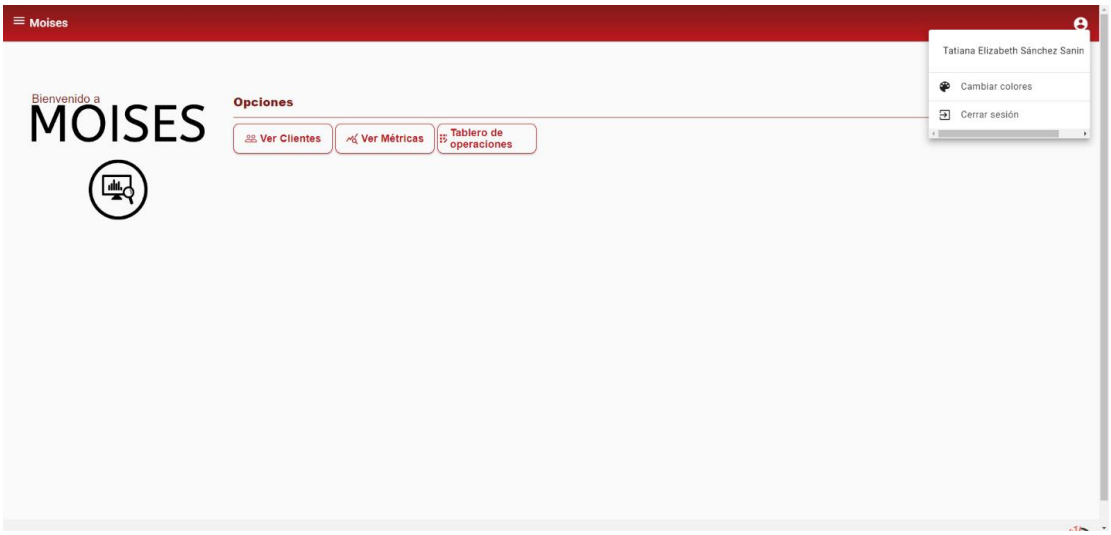

Fig. 14. Información de la sesión del usuario.

Se han definido los diferentes roles con distintos niveles de acceso a la información y funcionalidades. A través de una ventana emergente se detallan los permisos específicos para cada rol, incluyendo la capacidad de crear, editar, eliminar y visualizar datos, lo cuál solo puede ser asignado por un usuario administrador.

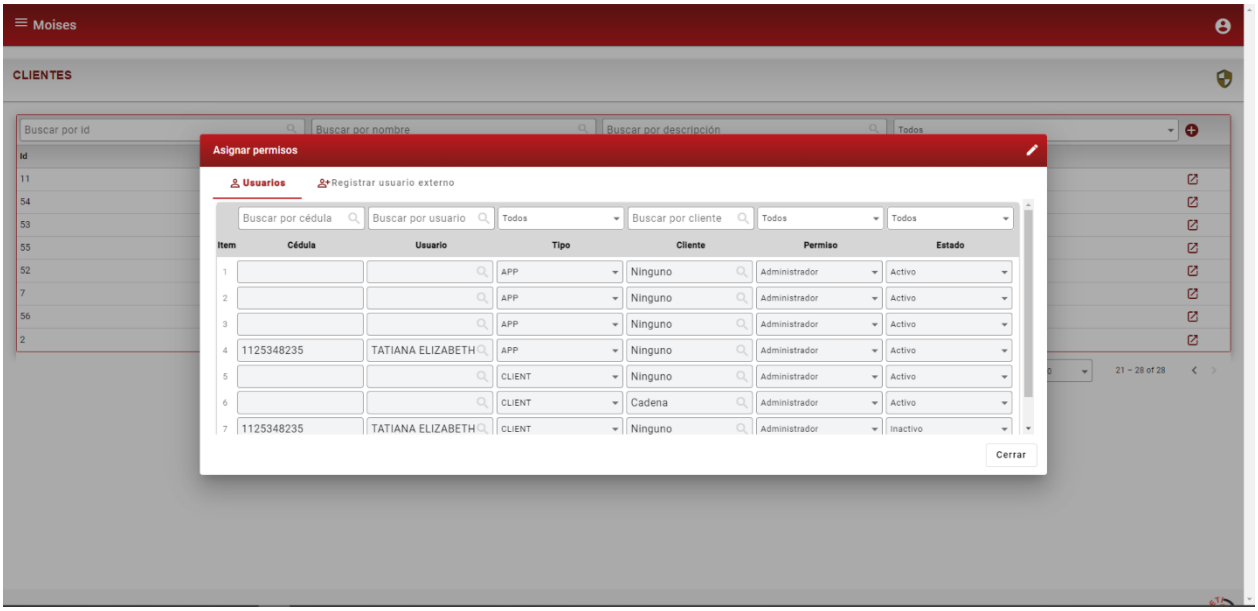

Fig. 15. Componente de asignación de permisos a usuarios.

Además, la plataforma permite al administrador invitar a usuarios externos (Fig. 16), otorgándoles permisos personalizados para acceder a la información específica que requieran. Con el fin de garantizar la seguridad, el control de acceso y limitando su visibilidad a la información relevante.

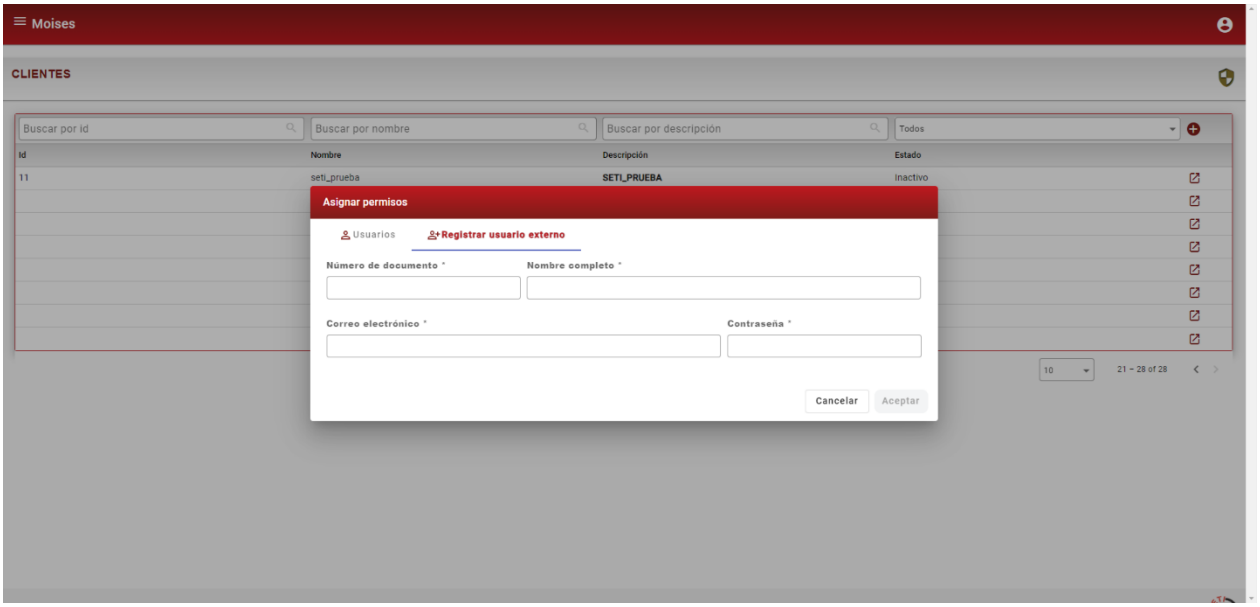

Fig. 16. Componente de registro de usuarios.

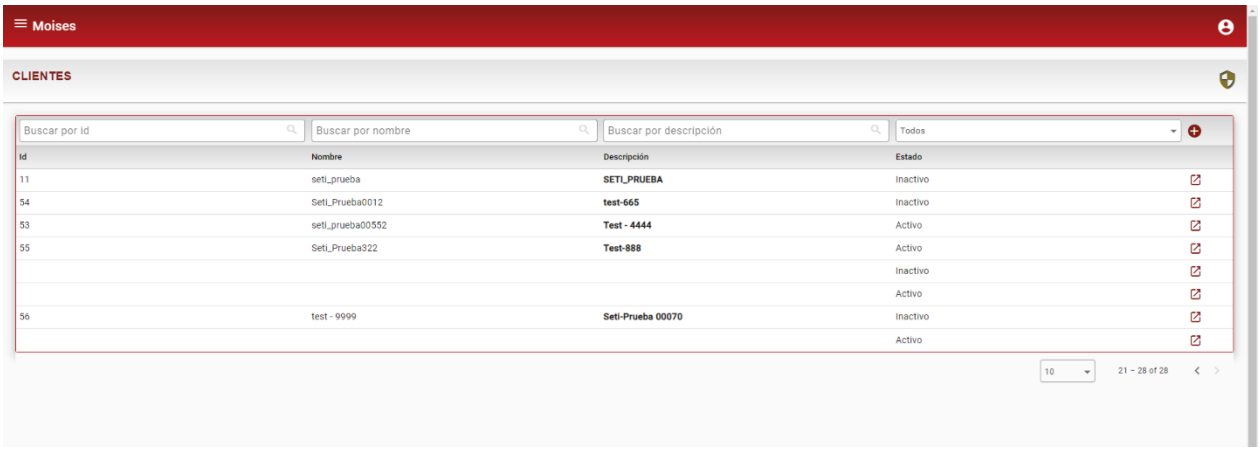

Fig. 17. Tabla de listado de clientes.

Los usuarios autorizados pueden acceder a un panel de control personalizado que les permite visualizar y gestionar la información de los clientes asignados. Esto incluye configurar datos del cliente (Fig. 18), administrar integraciones (Fig. 19), y analizar métricas detalladas del cliente (Fig. 20), los servicios, servidores y recursos asociados a cada cliente.

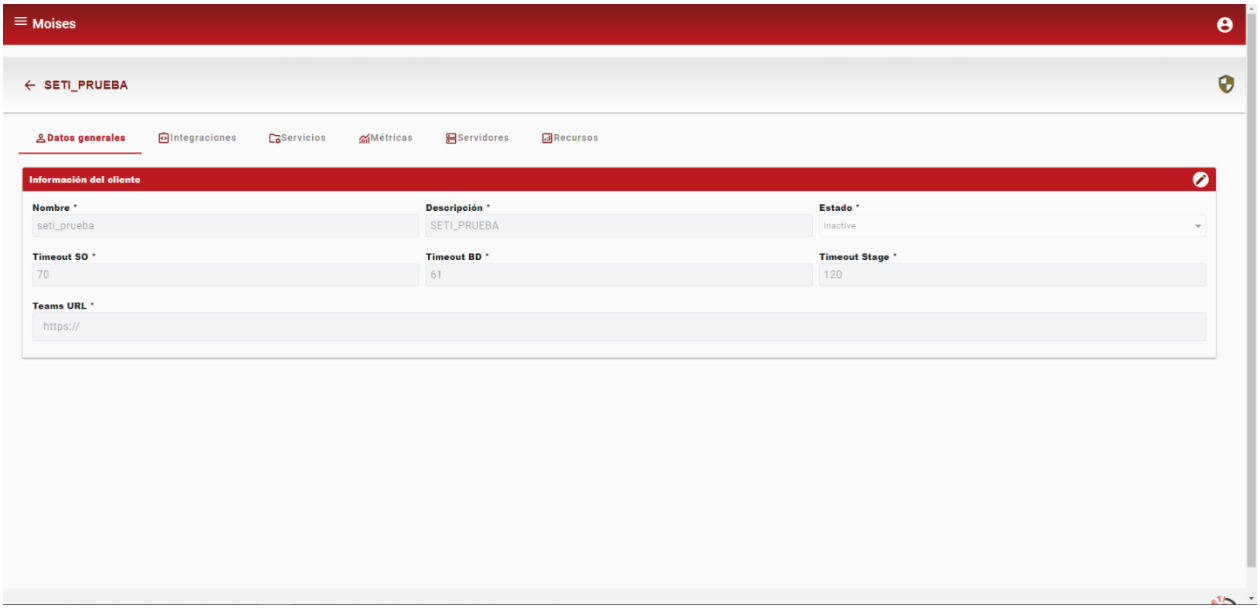

Fig. 18. Información general del cliente.

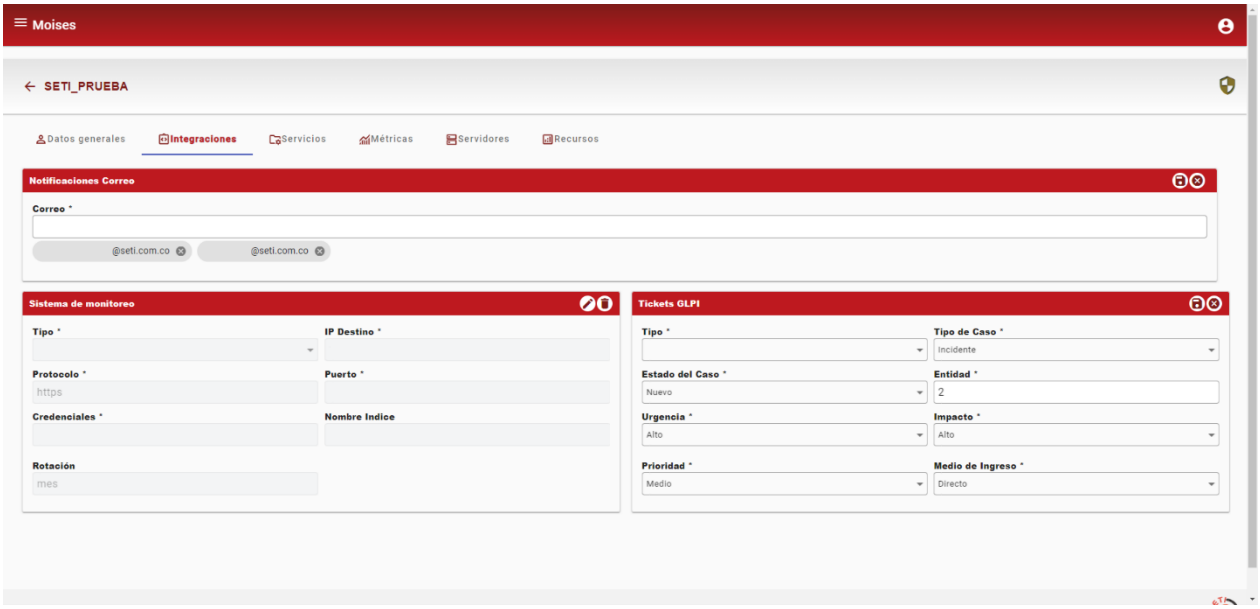

Fig. 19 Datos de integraciones del cliente.

La tabla de métricas (Fig. 20) ofrece una vista personalizada y filtrable de los datos de cada cliente. Los usuarios pueden aplicar filtros a campos como nombre, tipo y estado, y realizar búsquedas avanzadas para encontrar rápidamente las métricas que necesitan. También es posible importar conjuntos de métricas predefinidos según tipo y fabricante. Esta funcionalidad, junto con

las opciones de personalización, permite a los usuarios adaptar la tabla a sus necesidades específicas y obtener rápidamente la información necesaria para tomar decisiones informadas.

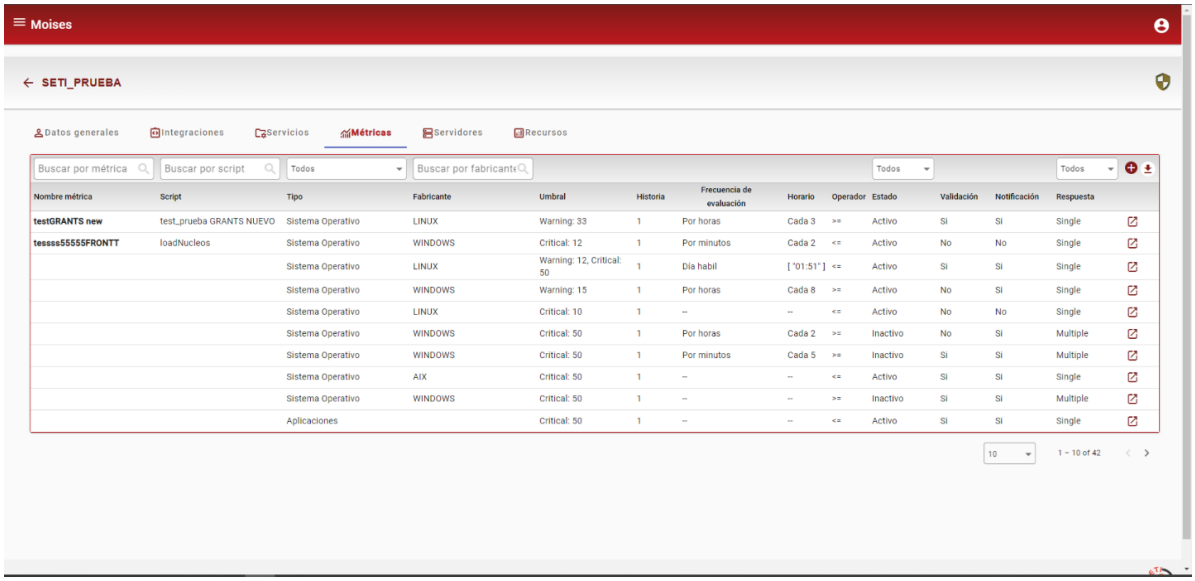

Fig. 20. Lista de métricas asociadas al cliente.

La tabla de métricas proporciona una vista general de los datos, pero los usuarios también pueden acceder a una vista detallada de cada métrica individual (Fig. 21). Esta vista permite editar los datos directamente, ofreciendo un control total sobre la información y facilitando la actualización y corrección de estos.

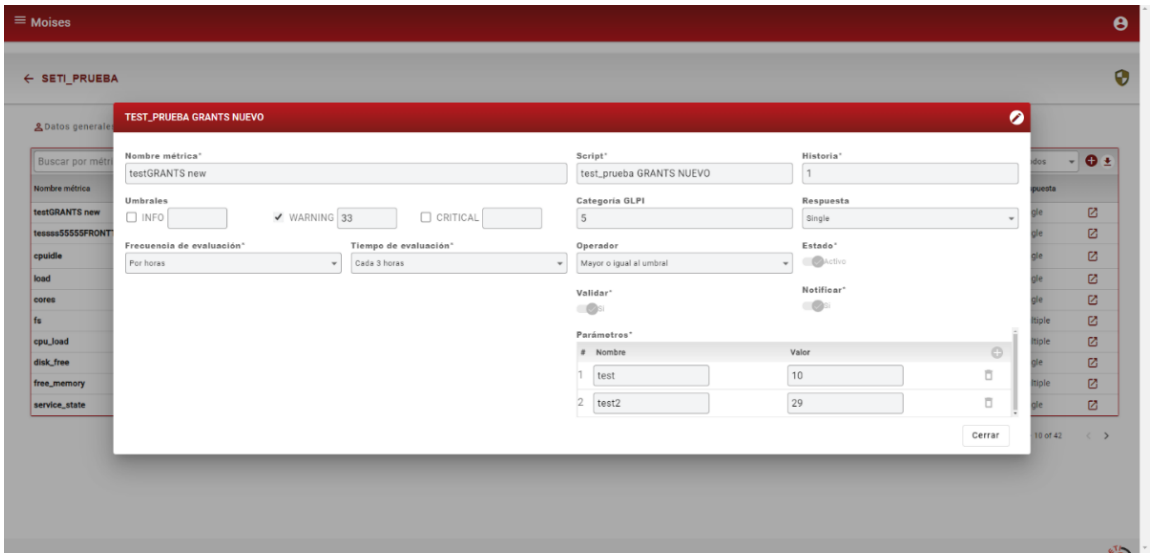

Fig. 21. Vista de edición de una métrica.

La tabla de recursos (Fig. 22) presenta una vista detallada de todas las bases de datos y servidores de aplicaciones, incluyendo información como nombre, tipo, fabricante, servicio asociado, servidor asociado y estado. Los usuarios pueden filtrar los resultados por cualquier combinación de estos campos y realizar acciones como editar la información del recurso y crear nuevos recursos (Fig. 23).

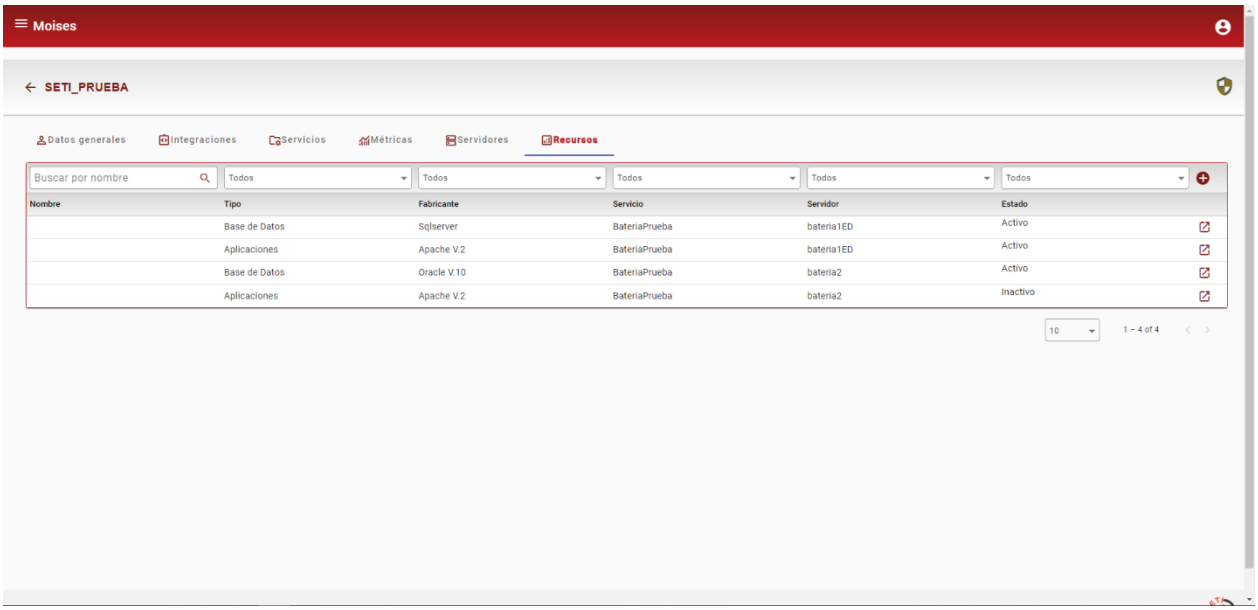

Fig. 22. Lista de recursos asociados a un cliente.

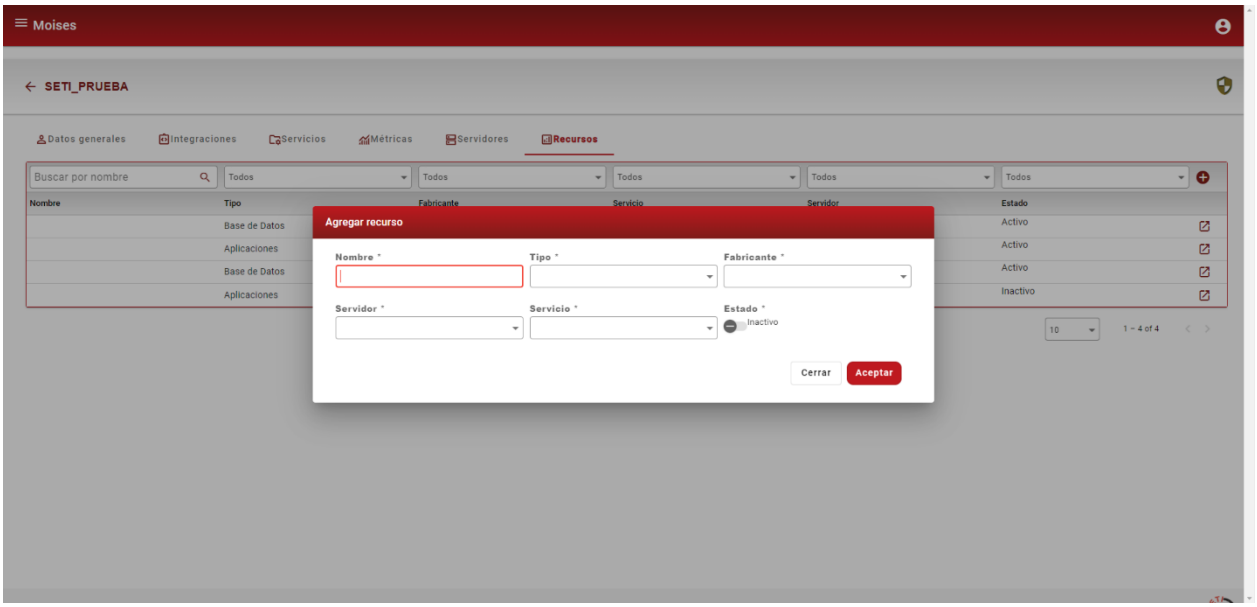

Fig. 23. Agregar un recurso.

La pantalla de edición de un recurso (Fig. 24) ofrece una interfaz intuitiva para gestionar y personalizar la información de cada elemento. Los usuarios pueden modificar campos como nombre, fabricante y estado, así como visualizar y filtrar las métricas asociadas. Esta integración de datos facilita la comprensión del rendimiento del recurso y permite una gestión más eficiente de la infraestructura.

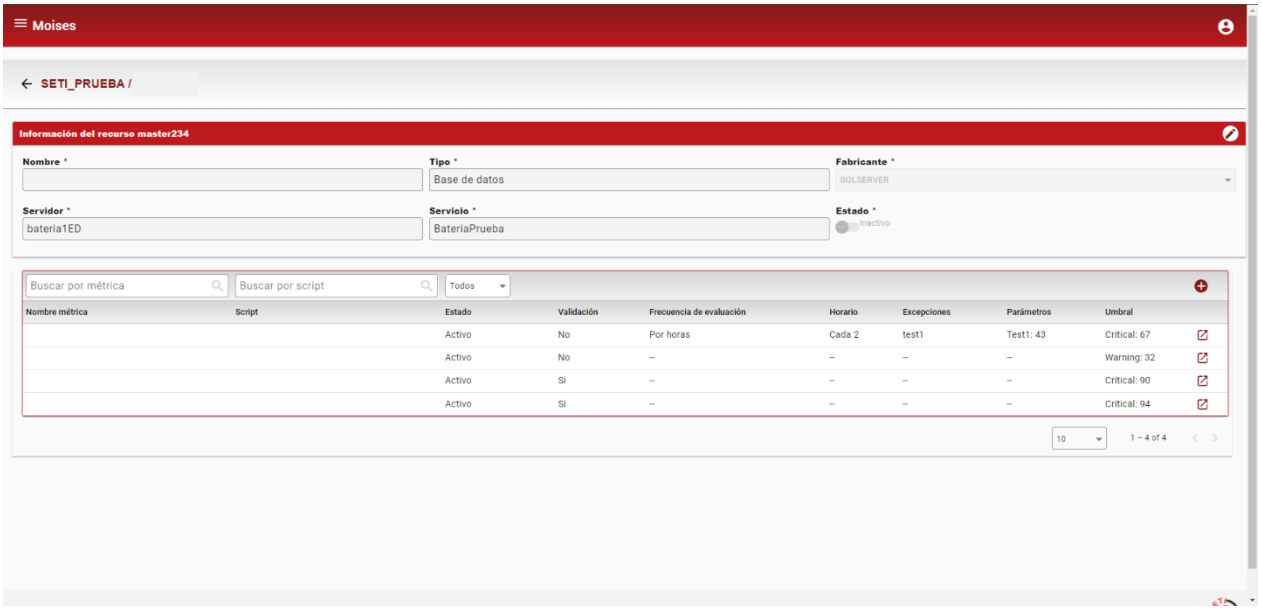

Fig. 24. Información de un recurso.

Cada tipo de recurso cuenta con características específicas que se reflejan en la información mostrada al agregar y editar un recurso (Fig. 27). Por ejemplo, los servidores de aplicaciones presentan una sección de inclusiones (Fig. 25), mientras que las bases de datos muestran una lista de excepciones (Fig. 26). Esta personalización garantiza que los usuarios tengan acceso a la información más relevante para gestionar cada tipo de recurso de manera eficiente.

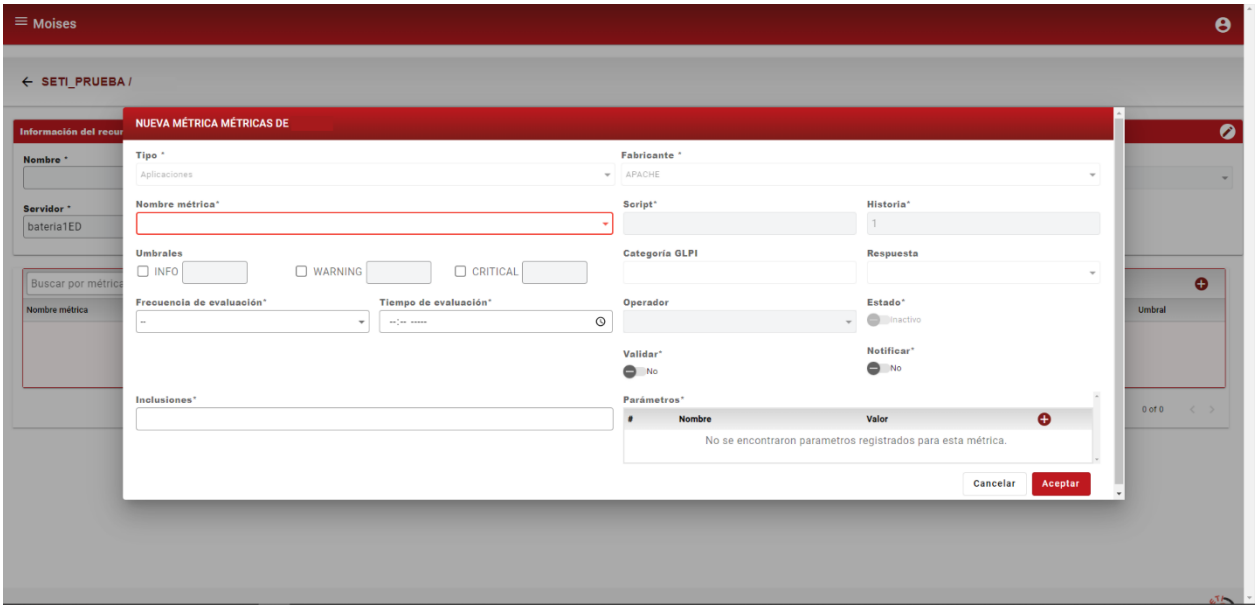

Fig. 25. Información de un recurso.

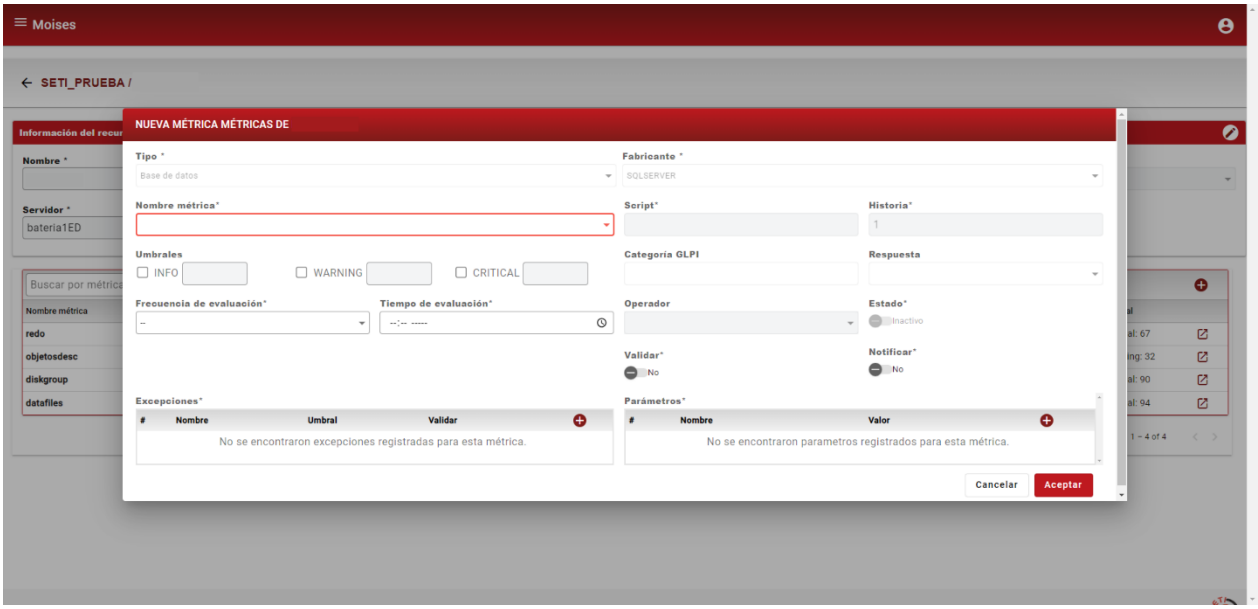

Fig. 26. Agregar métrica de un recurso de base de datos.

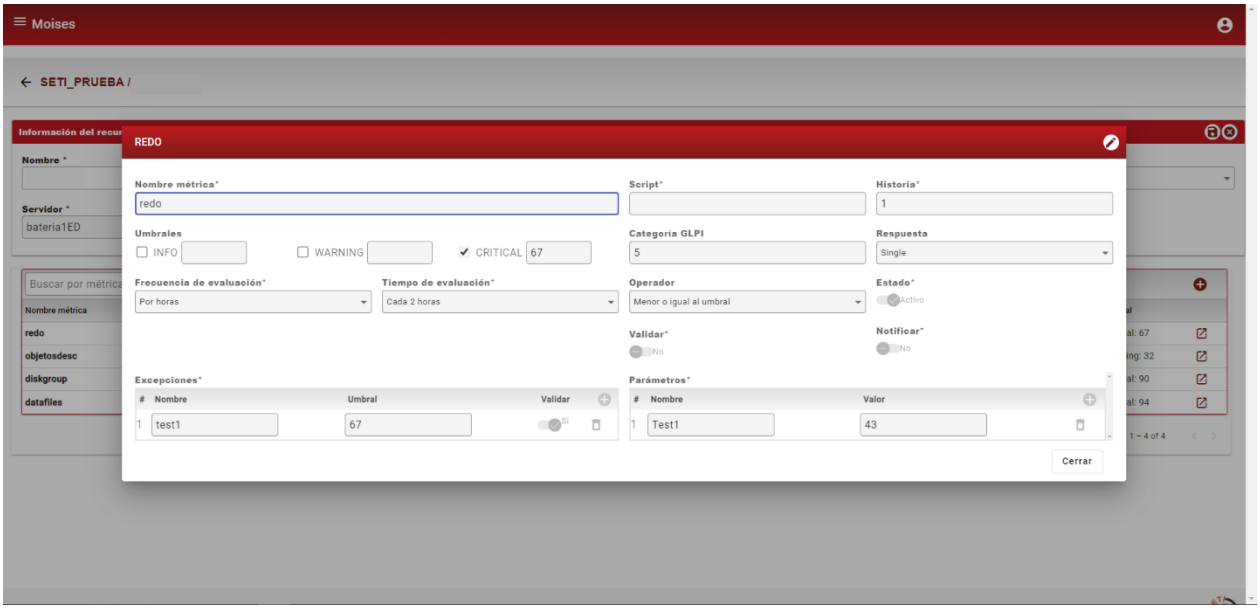

Fig. 27. Editar una métrica de un recurso.

El tablero de operaciones (Fig. 28) es la interfaz principal para que los usuarios externos puedan monitorear el estado de los clientes a los que tienen acceso. Esta pantalla, diseñada con una interfaz intuitiva, proporciona una visión general y actualizada de la información más relevante, facilitando la toma de decisiones y el seguimiento de los clientes.

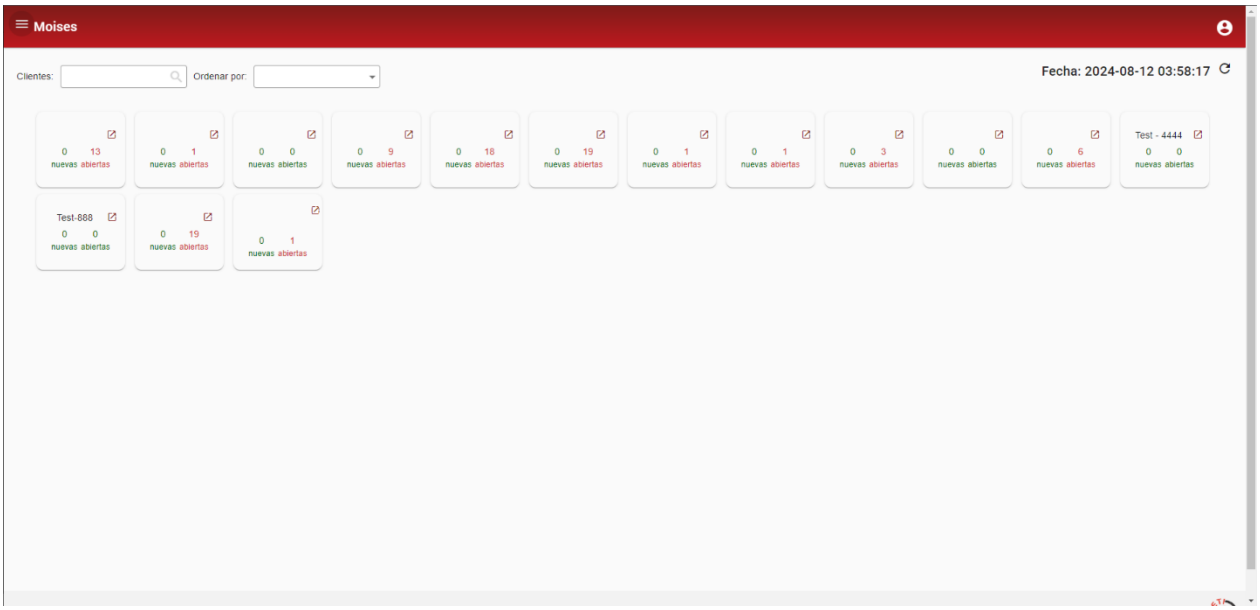

Fig. 28. Tablero de operaciones.

El tablero de operaciones proporciona una visión holística del estado de los clientes, incluyendo una sección de alarmas (Fig. 29) que permite acceder a una variedad de diagramas (Fig. 30, 31, 32). Estos diagramas visualizan el comportamiento de los servicios y métricas específicas, facilitando la identificación de patrones, tendencias y anomalías, lo que permite tomar decisiones informadas basadas en evidencia.

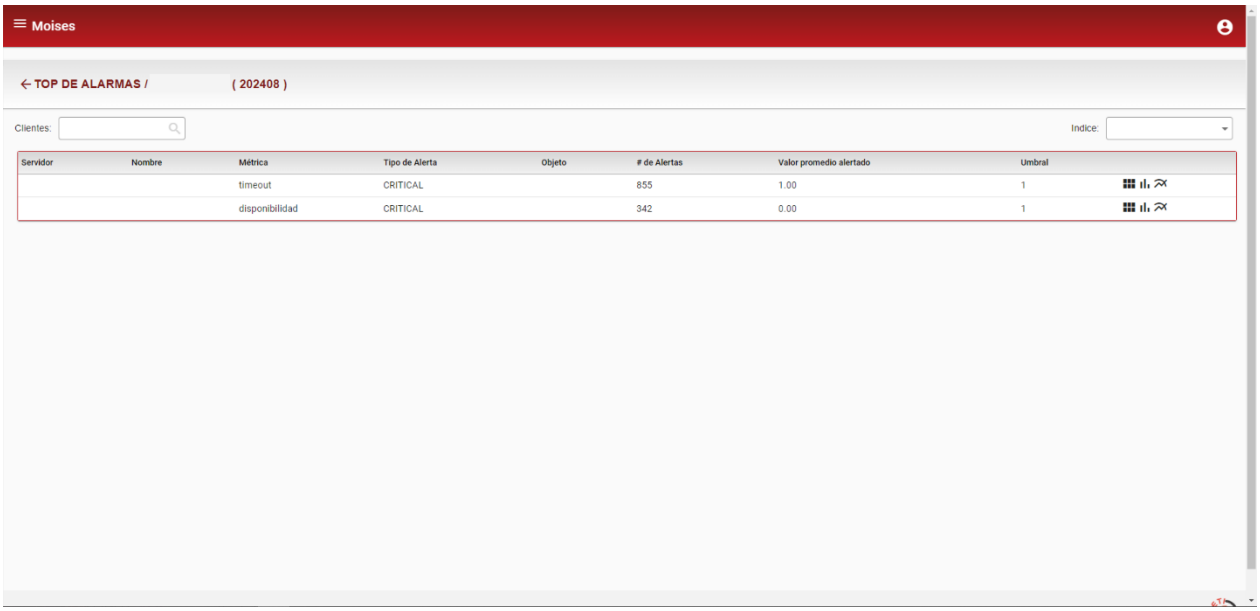

Fig. 29. Alarmas de un cliente.

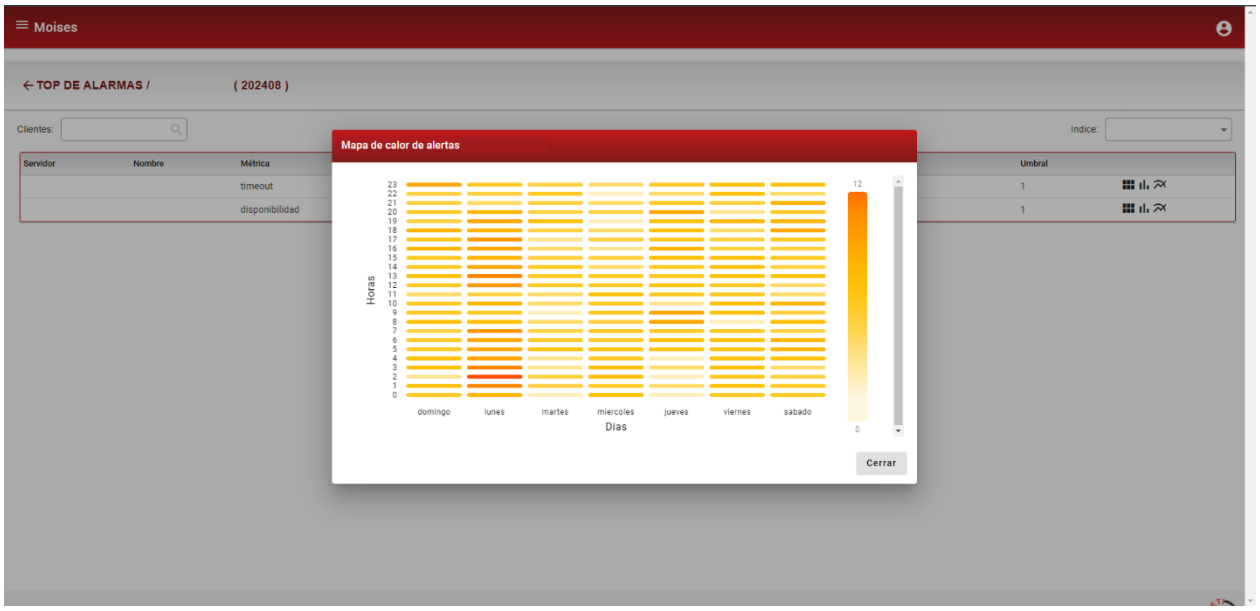

Fig. 30. Mapa de calor de una alarma.

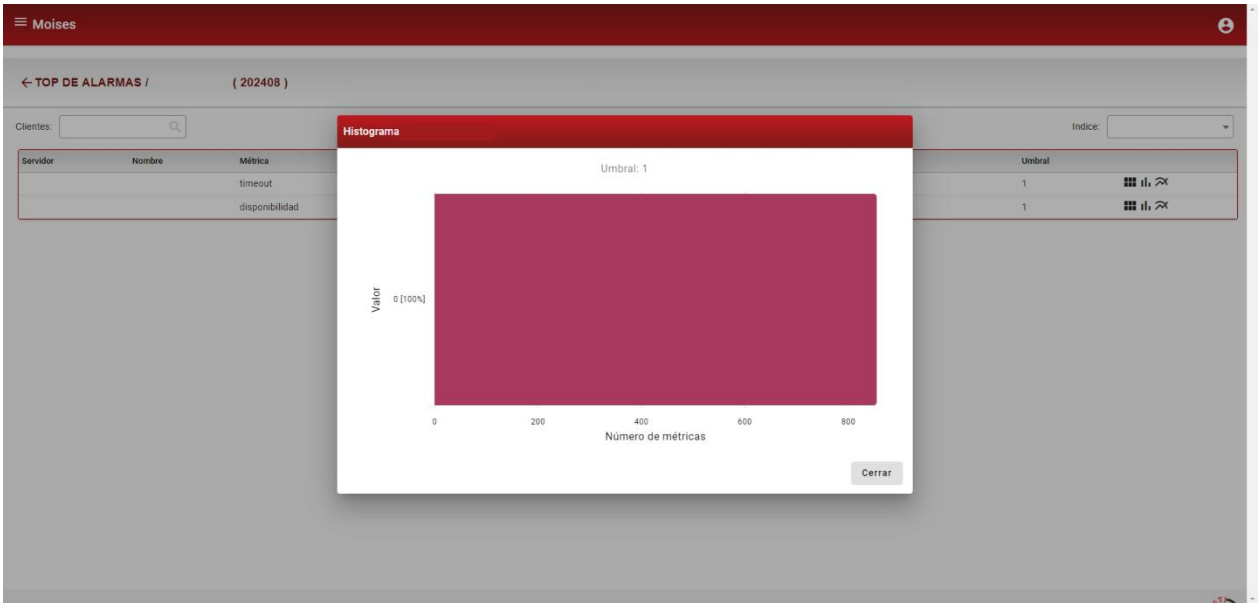

Fig. 31. Histograma de una alarma.

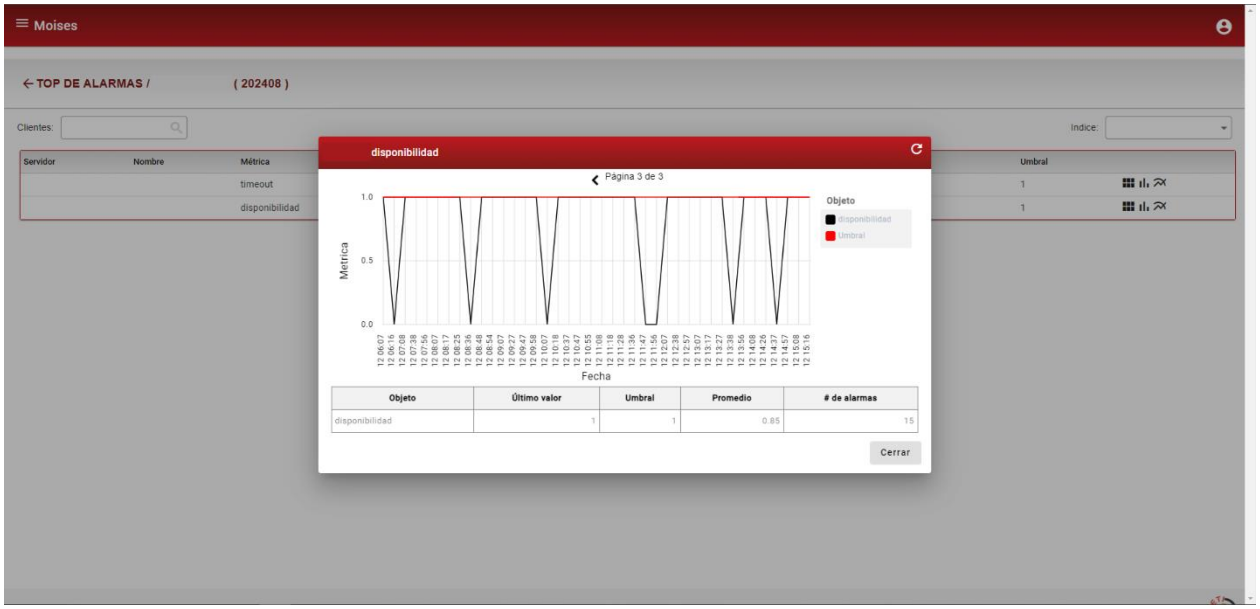

Fig. 32. Diagrama de una alarma.

Los clientes disponen de listas de alertas (Fig. 33) que resumen los incidentes y proporcionan la información más relevante para su resolución. Esta combinación de listas de alertas y diagramas detallados proporciona una herramienta poderosa para la gestión de incidentes y la toma de decisiones informadas.

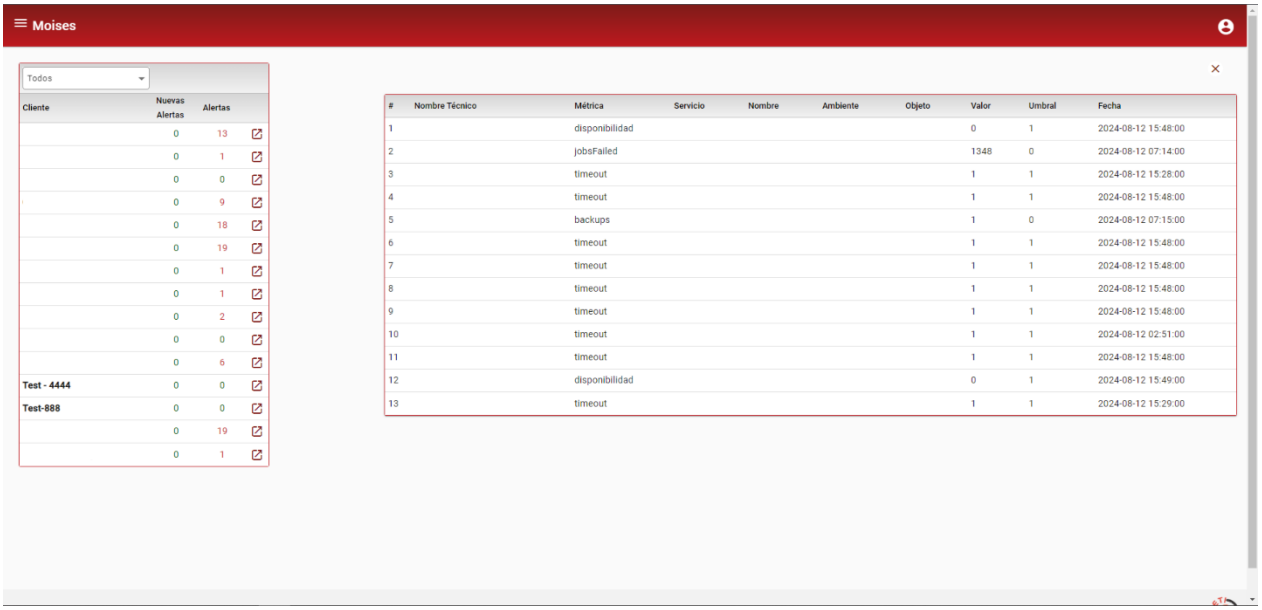

Fig. 33. Alertas de un cliente.

La plataforma permite a los clientes monitorear de cerca el estado de sus servicios a través de una tabla gráfica (Fig. 34) que muestra el estado de cada componente (sistema operativo, base de datos, servidor de aplicaciones) dentro de un ambiente específico. Esta visualización, combinada con las métricas asociadas, facilita la identificación rápida de posibles problemas y permite a los clientes tomar decisiones informadas para garantizar la continuidad de sus operaciones.

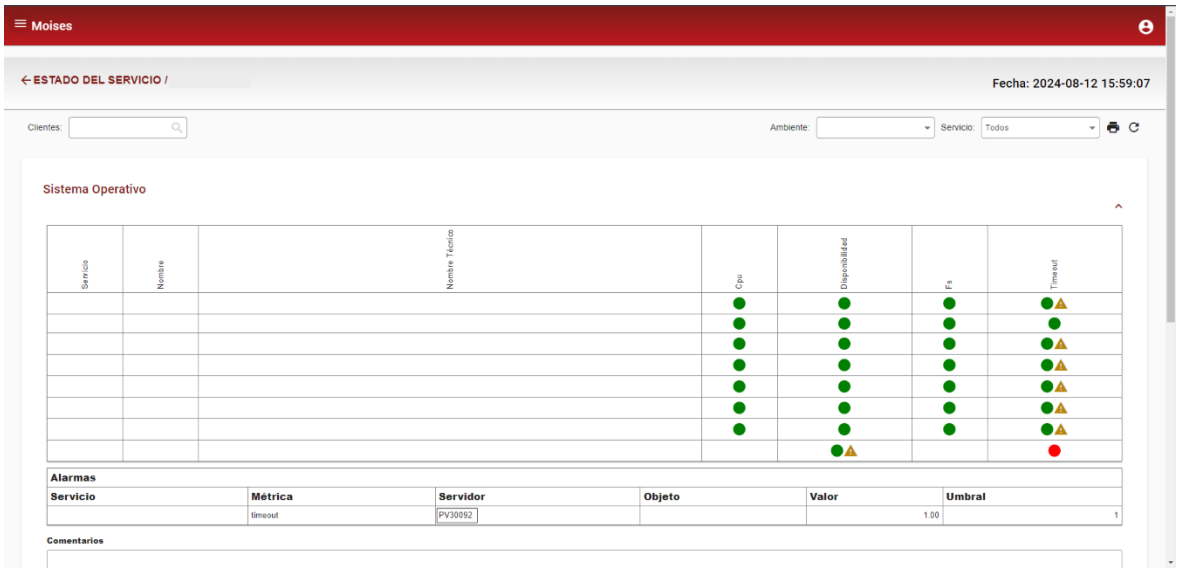

Fig. 34. Estado del servicio de un cliente.

Los diagramas detallados que visualizan el estado de los servicios pueden exportarse en formato PDF (Fig. 35), permitiendo a los usuarios generar informes personalizados y compartirlos con otros miembros del equipo. Esta funcionalidad es ideal para documentar incidentes, crear presentaciones y facilitar la colaboración en la resolución de problemas.

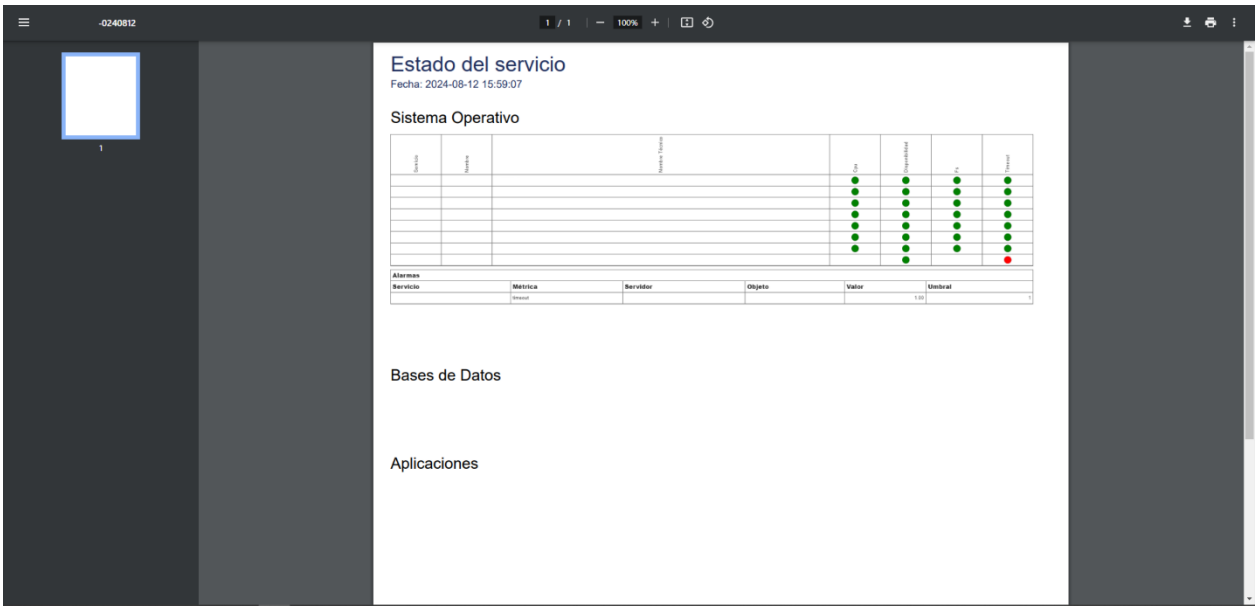

Fig. 35. Archivo exportado del estado del servicio.

VIII. CONCLUSIONES

- La importancia de la planificación y la definición clara de objetivos SMART (específicos, medibles, alcanzables, relevantes y con un plazo definido), permite alinear al equipo y avanzar de manera organizada hacia el logro de metas comunes.
- La comunicación fluida y transparente entre los miembros del equipo es esencial para evitar malentendidos, fomentar la colaboración y resolver problemas de manera oportuna.
- El valor de las metodologías ágiles a través del marco Scrum permite una mejor adaptación a los cambios, responder a las necesidades del usuario de manera iterativa y entregar valor de forma incremental.
- La importancia de la gestión del tiempo y el cumplimiento de plazos. La capacidad de organizar el trabajo, priorizar tareas y cumplir con los plazos establecidos es fundamental para el éxito del proyecto.
- El valor de la diversidad de perspectivas y la colaboración entre las habilidades individuales. Un equipo formado por personas con diferentes habilidades y experiencias enriquece el proceso de desarrollo y aporta soluciones más creativas e innovadoras.
- La capacidad de gestionar conflictos de manera constructiva. Es inevitable que surjan diferencias de opinión en un equipo de trabajo. La clave está en abordar los conflictos de manera abierta, respetuosa y enfocada en encontrar soluciones que beneficien al proyecto.
- La importancia de dar y recibir feedback es una herramienta valiosa para el crecimiento individual y del equipo, enfocado en la mejora continua.
- Reconocer y celebrar los logros alcanzados en equipo fortalece la motivación, el compromiso y el sentido de pertenencia.

REFERENCIAS

[1] G. F. Rozas, *Introducción a UML, Lenguaje Unificado de Modelado,* Udemy, 2020.

[2] F. Herrera, *Nest: Desarrollo backend escalable con Node,* Udemy, 2023.

[3] Google, «Introduction to Angular docs,» Google, 2010-2024.

[4] F. Herrera, *Angular: De Cero a Experto,* Udemy, 2024.

[5] E. Meijer, «Reactive Extensions Library for JavaScript,» Microsoft, 2009.

[6] F. Herrera, *ReactiveX - RxJs: De cero hasta los detalles,* Udemy, 2024.

[7] Google, «Angular Material,» Google, 2010-2024.

[8] MIT, «Nest Documentation,» MIT by Kamil Mysliwiec, 2017-2024.

[9] F. Herrera, *Principios SOLID y Clean Code,* Udemy, 2023.

[10] Oracle, «MySQL Documentation,» Oracle, 2024.

[11] P. Tilotta, *SQL - Curso completo de Bases de Datos - de 0 a Avanzado,* Udemy, 2024.

[12] Docker Inc., «Docker Docs,» Docker Inc., 2013-2024.

[13] F. Herrera, *Docker - Guía práctica de uso para desarrolladores,* Udemy, 2023.

[14] A. Training, *Kubernetes al completo,* Udemy, 2024.

[15] OpenJS Foundation, «Getting started with Jest,» Jest, 2024.

[16] E. P. Morillas, *Angular con Jest: Pruebas unitarias profesionales,* Udemy, 2023.

[17] Google, «Guía de iconos de material,» Google, 2024.