Practica Empresarial: Desarrollo de un Sistema Web para el Control y Ejecución de Modelos de Calidad de Datos en el Mercado de Energía Mayorista para la Empresa XM

Juan Sebastián Durán Macias

Trabajo de Grado para Optar al Título de Ingeniero de Sistemas

Director

Carlos Adolfo Beltrán Castro

Magister en Gestión de la Tecnología Educativa

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Ingeniería de Sistemas e Informática

Bucaramanga

2024

Dedicatoria

Este proyecto está dedicado con profundo agradecimiento a mi familia, cuyo apoyo inquebrantable y amor constante han sido mi propulsor en este fantástico viaje. También quiero extender mi gratitud a todos aquellos que comparten la creencia en el poder transformador de la tecnología.

Creo firmemente que nuestra región tiene el potencial de ser una fuerza líder en el progreso tecnológico mundial. Con dedicación, colaboración y un enfoque apasionado hacia la innovación, podemos trazar el camino hacia un futuro donde Latinoamérica sea reconocida como una cuna de ideas audaces y soluciones vanguardistas.

Agradezco a todos los que creen en este sueño y espero ansioso contribuir al avance tecnológico de nuestra región y del mundo. Juntos, forjaremos un mañana donde la tecnología sea una herramienta poderosa para el progreso y la transformación positiva.

Agradecimientos

Quiero expresar mi profundo agradecimiento a Dios, a mi familia, a mis amigos y a mi pareja. Han sido una parte esencial en mi trayectoria personal y profesional. Este logro académico es un tributo a ustedes, quienes confiaron en mí desde el principio, incluso en los días en los que me sentía perdido y sin rumbo. Siempre extendieron su mano, sin importar las circunstancias. Les debo más de lo que puedo expresar; sin su apoyo, no sé dónde estaría. Gracias por demostrarme que el amor es desinteresado y por ser mi roca en los momentos difíciles. Espero en muchos momentos de mi vida poder devolverles, aunque sea una pequeña parte de todo lo que han hecho por mí.

Muchas gracias al profesor Carlos Beltrán por haber confiado en mi idea, darme los ánimos que necesitaba para continuar con el proyecto y por brindarme su apoyo en todo momento.

Agradezco sinceramente a mi querida alma mater, la Universidad Industrial de Santander, y a todos los integrantes que la conforman. Gracias por guiarme en la forja de mi espíritu investigativo y por enseñarme que, en la vida, con esfuerzo y dedicación, todo objetivo puede alcanzarse. Estaré siempre agradecido a la educación pública, y a nuestros compañeros combatientes, que siempre se han preocupado por los intereses del estudiantado, sin ustedes esto no sería posible.

Agradezco profundamente a SETI por su constante apoyo y atención desde el primer día de mis prácticas. Su dedicación y ayuda constante fueron fundamentales para mi aprendizaje y crecimiento durante esta etapa de mi vida.

Tabla de Contenido

Índice de Figuras

Resumen

Título: Desarrollo de un sistema web para el control y ejecución de modelos de calidad de datos en el mercado de energía mayorista para la empresa XM. [1]

Autor: Juan Sebastian Duran Macias. [2]

Palabras clave: Calidad de los Datos, Confiabilidad de Datos, Sistema Web, Control de Datos, Mercado de Energía Mayorista, Seguridad de los Datos.

Descripción: En el Mercado de Energía Mayorista, donde la toma de decisiones precisa es esencial, la calidad de los datos desempeña un papel crucial en los procesos operativos y de gestión. XM, ante la complejidad de este entorno, identifica la necesidad urgente de garantizar la integridad y confiabilidad de la información energética proveniente de diversas fuentes.

Este proyecto se enfoca en abordar los desafíos específicos del sector energético, explorando estrategias y soluciones adoptadas para cumplir con los objetivos propuestos. La aplicación web resultante actúa como una herramienta de almacenamiento eficiente, facilitando la recolección de datos para su procesamiento posterior. Responde así a la imperante necesidad de garantizar la calidad de los datos en un entorno tan dinámico y competitivo como el sector energético.

---

[1] Trabajo de grado

[2] Facultad de Ingenierías Fisicomecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Carlos Adolfo Beltrán Castro

Abstract

Title: Development of a web system for the control and execution of data quality models in the wholesale energy market for the company XM. [1]

Autor: Juan Sebastian Duran Macias. [2]

Key Words: Data Quality, Data Reliability, Web System, Data Control, Wholesale Energy Market, Data Security.

Description: In the Wholesale Energy Market, where accurate decision making is essential, data quality plays a crucial role in operational and management processes. XM, given the complexity of this environment, identifies the urgent need to guarantee the integrity and reliability of energy information from various sources.

This project focuses on addressing the specific challenges of the energy sector, exploring strategies and solutions adopted to meet the proposed objectives. The resulting web application acts as an efficient storage tool, facilitating data collection for further processing. It thus responds to the prevailing need to guarantee data quality in an environment as dynamic and competitive as the energy sector.

---

[1] Bachelor's Thesis

[2] Faculty of Physicomechanical Engineering. School of Systems and Computer Engineering. Supervisor: Carlos Adolfo Beltrán Castro

Introducción

En el ámbito empresarial y organizacional, los sistemas de información constituyen el núcleo fundamental para la gestión eficiente y accesible de datos complejos. Su papel es crucial en la toma de decisiones, convirtiendo datos en información útil que respalda las operaciones y estrategias empresariales.

La configuración de un sistema de información implica una tarea compleja que va más allá de la simple administración de datos. Requiere el diseño y ajuste preciso de los componentes del sistema para satisfacer las necesidades específicas de una organización. La correcta configuración involucra la integración de procesos, la adaptación de funcionalidades y la asignación eficiente de recursos.

Históricamente, la gestión de la calidad de datos en el sector energético se ha basado en modelos internos que abordan problemas como errores, inconsistencias y falta de uniformidad en la información. Sin embargo, la falta de una plataforma unificada para gestionar y operar estos modelos ha presentado desafíos significativos en su implementación y eficiencia.

Este trabajo de grado se propone diseñar e implementar un sistema web que mejore la precisión y confiabilidad de la información utilizada en los procesos operativos y de gestión energética. A través de objetivos específicos como el diseño de una interfaz atractiva, el desarrollo de una estructura eficiente para almacenar los resultados de los modelos y la implementación de medidas de seguridad se busca proporcionar a XM una herramienta robusta y eficaz para abordar los desafíos actuales en la gestión de la calidad de datos.

1. Planteamiento y Justificación del Problema

En el entorno altamente dinámico y competitivo del Mercado de Energía Mayorista, la calidad de los datos es un factor esencial para la toma de decisiones precisa y efectiva. Los datos energéticos provenientes de diversas fuentes pueden ser propensos a errores, inconsistencias y falta de uniformidad, lo que puede comprometer la integridad de los procesos operativos y de gestión.

La empresa XM reconoce la importancia de asegurar que los datos utilizados en sus operaciones y gestión energética sean de la más alta calidad. La implementación de modelos de calidad de datos ha surgido como una estrategia fundamental para abordar esta preocupación. Estos modelos tienen la capacidad de evaluar, transformar y mejorar automáticamente los datos en términos de precisión y confiabilidad.
En este contexto, el desarrollo de un sistema web destinado a la gestión y operación de modelos de calidad de datos se presenta como una solución estratégica. Esta plataforma web permitirá a los usuarios de XM acceder de manera conveniente a la funcionalidad de los modelos de calidad, sin necesidad de comprender los detalles técnicos de su operación interna. Al proporcionar una interfaz intuitiva y fácil de usar, el sistema permitirá a los usuarios configurar y ejecutar modelos de calidad de datos de manera eficiente, sin requerir intervención en los procesos internos de estos modelos.

2. Objetivos

2.1 Objetivo General

- Diseñar e implementar un sistema web para el control y ejecución de modelos de calidad de datos en el Mercado de Energía Mayorista, con el fin de mejorar la precisión y confiabilidad de la información utilizada en el proceso de operación y gestión energética.

2.2 Objetivos Específicos

- Diseñar una interfaz de usuario intuitiva y atractiva que facilite la interacción de los usuarios con el sistema de control y ejecución de modelos de calidad de datos.
- Diseñar y desarrollar la estructura para almacenar y gestionar los resultados de los modelos.
- Realizar validación de datos e implementar medidas de seguridad para asegurar la integridad y la calidad de la información procesada.
- Generar un manual de usuario claro y detallado para la utilización del sistema web.

3. Marco de Referencia

3.1 Antecedentes y Descripción del Problema

En el entorno altamente dinámico y competitivo del Mercado de Energía Mayorista, la calidad de los datos desempeña un papel crítico en la toma de decisiones precisas y eficaces. Los datos energéticos, provenientes de diversas fuentes y sistemas, a menudo están sujetos a errores, inconsistencias y falta de uniformidad, lo que puede socavar la integridad de los procesos operativos y de gestión.

Históricamente, las organizaciones en el sector energético han recurrido a modelos de calidad de datos internos para abordar estos problemas. Estos modelos, que evalúan y mejoran la calidad de los datos, han demostrado ser eficaces en la detección y corrección de problemas, como registros duplicados, datos incoherentes y valores fuera de rango. Sin embargo, la falta de una plataforma unificada para gestionar y operar estos modelos ha dado lugar a desafíos significativos en su implementación y eficiencia.

En este contexto, se plantea la necesidad de desarrollar un sistema web que permita el control y la ejecución centralizados de estos modelos de calidad de datos en el Mercado de Energía Mayorista. Este sistema web se centrará en la creación de una interfaz intuitiva y de fácil acceso, lo que facilitará a los usuarios la configuración y el monitoreo de los modelos de calidad de datos sin necesidad de comprender los detalles técnicos de su operación interna.

Esta investigación no solo aborda la problemática de la calidad de datos en el sector energético, sino que también se enfoca en la solución práctica que representa el desarrollo de un sistema web dedicado a esta tarea. Este sistema no solo agilizará y mejorará la calidad de los procesos de control de calidad de datos, sino que también podría servir como un modelo para otros sectores industriales que enfrentan desafíos similares en la gestión de datos.

En última instancia, este proyecto tiene como objetivo llenar un vacío existente al proporcionar una solución concreta y altamente funcional que mejora la calidad de los datos en el Mercado de Energía Mayorista, fortaleciendo la toma de decisiones y optimizando las operaciones energéticas en beneficio de la empresa XM y el sector en su conjunto.

3.2 Fundamentos Teóricos

3.2.1 Sistemas de Información y Desarrollo Web.

3.2.1.1 Sistemas de Información. Un sistema de información se define como una estructura organizada de componentes o partes que interactúan entre sí para lograr un objetivo específico. En términos generales, estos sistemas reciben entradas en forma de energía, materia del entorno o datos, y transforman estos insumos en resultados, que pueden ser materia, energía o información (Tovar, s.f.).

Asimismo, existen diversas interpretaciones para el concepto de sistema de información. Según Andreu, Ricart y Valor (1991), se trata de una estructura formal que gestiona procesos basados en datos e información estructurada según los requisitos de la organización. Su función principal es recopilar, procesar y distribuir información selectivamente, tanto para la ejecución de las actividades de la organización como para la toma de decisiones y el control.

Este sistema recopila, procesa y transforma datos de manera sistematizada, ordenada y esquematizada, con el propósito de facilitar la toma de decisiones. En este sentido, el sistema de información recibe datos como entrada, los almacena, los procesa y finalmente los convierte en resultados, que pueden ser informes, listados, índices, medidas de posición o tendencias. En el caso de un sistema de información de Recursos Humanos, se basa en datos proporcionados por diversas fuentes, como Reclutamiento y Selección de Personal, Entrenamiento y Desarrollo de Personal, Base de datos de Recursos Humanos, Evaluación del Desempeño, Administración de Salarios, Registro y Control de Personal, y Estadísticas de Personal, entre otros, para gestionar aspectos relacionados con el personal de la organización, como fallas, atrasos y disciplina, así como cuestiones relacionadas con la administración de los recursos humanos (Alvarado Rosado, Liseth Francesca, s.f.).

3.2.1.2. Desarrollo Web. Según María Coppola (Blog de HubSpot), el desarrollo web se refiere al proceso de concepción, creación y mantenimiento de un sitio web que funcione en Internet. Este proceso implica el uso de diversos lenguajes de programación, dependiendo del modelo y la funcionalidad que se quiera lograr en el sitio. Cada sitio web tiene su propia URL única que lo identifica en la vasta red de Internet.

Los sitios web se pueden clasificar de diversas maneras, pero desde la perspectiva del desarrollo web, generalmente se dividen en dos partes fundamentales:

Frontend: Esta es la parte del sitio web que interactúa directamente con los usuarios, y abarca tanto el aspecto visual como la funcionalidad. El frontend es crucial para proporcionar una experiencia de usuario (UX) satisfactoria y una interfaz de usuario (UI) amigable.

En esta parte, se diseñan y desarrollan los elementos visuales y funcionales que los usuarios ven y con los que interactúan directamente.

Backend: El backend, por otro lado, se relaciona con la parte del sitio web que está en contacto directo con el servidor. Aquí es donde se implementa el código de programación que crea la estructura del sitio. Aunque permanece en segundo plano para los usuarios finales, el backend es esencial para la funcionalidad del sitio. Se encarga de cuestiones como la accesibilidad, la gestión de bases de datos, las actualizaciones y los cambios en el sitio.

Es importante destacar que el desarrollo web es un proceso multidisciplinario que requiere una colaboración efectiva entre profesionales del frontend y el backend para crear sitios web efectivos y atractivos. La comprensión de estos dos aspectos es fundamental para cualquier proyecto de desarrollo web exitoso.

3.2.2 Arquitectura de Aplicaciones Web Escalables.

Martin Kleppmann, en su obra "Designing Data-Intensive Applications," resalta la importancia de la arquitectura de aplicaciones web escalables como un elemento esencial para asegurar que una aplicación pueda crecer y adaptarse a las cambiantes demandas del mercado y los usuarios sin sacrificar su rendimiento ni su confiabilidad. En el contexto del Mercado de Energía Mayorista, donde la gestión de grandes cantidades de datos desempeña un papel crítico, una arquitectura escalable se convierte en un pilar fundamental para garantizar que la aplicación pueda gestionar de manera eficaz el creciente volumen de datos y usuarios.

Kleppmann también destaca varios principios clave de la arquitectura escalable:

Distribución y Descentralización: En lugar de depender de un servidor central, una arquitectura escalable distribuye la carga de trabajo y los datos en varios servidores o nodos. Esto permite una mayor paralelización y distribución de la carga.

Elasticidad: La arquitectura debe ser capaz de ajustar dinámicamente su capacidad de recursos en función de la demanda. Esto se logra mediante la implementación de servidores y recursos que se pueden escalar horizontalmente.

Microservicios: La descomposición de la aplicación en microservicios más pequeños y especializados facilita un desarrollo, implementación y escalabilidad ágil. Cada microservicio puede escalarse de manera independiente según sea necesario.

Almacenamiento Escalable: La utilización de sistemas de almacenamiento distribuido y escalable, como bases de datos NoSQL, resulta fundamental para gestionar grandes volúmenes de datos de manera eficiente y escalable.

Balanceo de Carga: La implementación de técnicas de balanceo de carga garantiza la distribución equitativa de las solicitudes de los usuarios entre múltiples servidores, evitando la sobrecarga de un único servidor (Kleppmann, "Designing Data-Intensive Applications").

3.2.3 Seguridad de Datos en Aplicaciones Web.

En el contexto de aplicaciones web diseñadas para la gestión de datos altamente sensibles en el Mercado de Energía Mayorista, la seguridad de los datos se erige como una preocupación de máxima relevancia. Estas aplicaciones tienen la tarea crítica de manejar información confidencial y esencial para la operación y gestión de la infraestructura energética.

Los desafíos de seguridad a los que se enfrentan estas aplicaciones web son diversos y van desde la prevención del acceso no autorizado hasta la garantía de la integridad de los datos.

Los desafíos específicos en materia de seguridad abordados incluyen la prevención del acceso no autorizado, asegurando que únicamente usuarios autorizados puedan ingresar al sistema, y la preservación de la integridad de los datos para evitar manipulaciones no autorizadas durante su transmisión o almacenamiento. Asimismo, la confidencialidad de los datos energéticos se convierte en un componente esencial, ya que cualquier brecha en esta confidencialidad podría dar lugar a la exposición de información estratégica y crítica. Además, la protección contra amenazas externas, como ataques cibernéticos y malware, se torna de suma importancia.

En la investigación de la seguridad de datos para aplicaciones web en el Mercado de Energía Mayorista, es crucial explorar diversas prácticas recomendadas y estrategias de seguridad. Entre ellas se encuentran la implementación de una sólida autenticación y autorización de usuarios, el uso de la encriptación de datos en tránsito y en reposo, la adopción de firewalls y medidas de seguridad de red, la realización de auditorías de seguridad y el registro de eventos para detectar actividades sospechosas, la aplicación de actualizaciones y parches de seguridad regulares, la capacitación de usuarios en las mejores prácticas de seguridad y la elaboración de un plan de respuesta a incidentes ante posibles violaciones de seguridad de datos.

La investigación en esta área se torna imperativa para asegurar que la aplicación web desarrollada para el Mercado de Energía Mayorista cumpla con rigurosos estándares de seguridad. Esto garantiza que los datos energéticos sensibles estén resguardados de manera efectiva, y que la integridad y la confidencialidad de esta información sean prioridades

indiscutibles en todo momento (Sullivan y Liu, "Web Application Security: A Beginner's Guide").

3.2.4 Modelos de Calidad de Datos en el Mercado de Energía Mayorista.

En el contexto del Mercado de Energía Mayorista, la calidad de los datos se presenta como una preocupación de suma importancia. Los datos relacionados con la energía, que constituyen el cimiento de las operaciones y la gestión en este mercado, deben caracterizarse por su precisión y confiabilidad a fin de asegurar un suministro de energía eficiente y seguro. La integridad de estos datos no solo influye en la eficacia de las tareas cotidianas, sino que también ejerce un impacto significativo en la toma de decisiones estratégicas a largo plazo.

A medida que se buscan soluciones para afrontar los desafíos en cuanto a la calidad de los datos, han surgido los modelos de calidad de datos como herramientas sumamente valiosas. Estos modelos se presentan como sistemas que, a través de la implementación de reglas, algoritmos y estándares, pueden evaluar, corregir y mejorar de manera automatizada la calidad de los datos. Su capacidad para identificar errores, anomalías y discrepancias en los datos energéticos y tomar acciones concretas para rectificarlos los convierte en componentes fundamentales para el mantenimiento de la calidad de los datos en este contexto.

La utilización de estos modelos de calidad de datos se ha convertido en una práctica esencial para garantizar la fiabilidad de los datos en el Mercado de Energía Mayorista y contribuir a la toma de decisiones informadas (DAMA, "Data Quality in the Energy Industry").

En una era impulsada por los datos, garantizar su calidad se ha convertido en algo primordial para las organizaciones de diversos sectores. El sector energético no es una

excepción. Los datos confiables son cruciales para la toma de decisiones, mejorar la eficiencia operativa y permitir el crecimiento sostenible.

GDM, una compañía especializada en gestión de datos enfatiza la importancia de las expectativas en la calidad de los datos. Según GDM, "La calidad de los datos está impulsada por las expectativas. Si entendemos las expectativas, podemos identificar si el estado actual cumple con esas expectativas o, si no se cumplen, entender qué debemos hacer para lograr un resultado positivo. Para aumentar esta complejidad, la industria energética está repleta de múltiples fuentes de datos, lo que a su vez significa que diferentes usuarios tienen diferentes desafíos y expectativas.".

GDM también proporciona una definición de marco de calidad de datos como "un proceso sistemático que perfila continuamente los datos en busca de errores e implementa diversas operaciones de calidad de los datos para evitar que entren errores en el sistema." Además, GDM resalta la importancia de establecer un bucle para monitorear y resolver problemas, creando así una manera constante de entender las expectativas y garantizar que los cambios en los datos se midan y evalúen (GDM).

3.2.5 Teoría de Control de Calidad de Datos

La teoría de control de calidad de datos es un componente crucial para garantizar la precisión y confiabilidad de los datos en cualquier contexto empresarial, incluido el Mercado de Energía Mayorista. Esta teoría se centra en establecer procesos y estándares para evaluar, mejorar y mantener la calidad de los datos utilizados en las operaciones y la toma de decisiones.

Conceptos Clave de la Teoría de Control de Calidad de Datos:

Dimensiones de Calidad de Datos: La teoría de control de calidad de datos reconoce varias dimensiones de calidad, como la exactitud, integridad, coherencia y relevancia de los datos. Cada dimensión se enfoca en aspectos específicos de la calidad y tiene métricas y estándares asociados.

Evaluación de la Calidad de Datos: Esta etapa implica la evaluación de los datos para identificar problemas y discrepancias. Se utilizan técnicas como la validación, la limpieza y la normalización de datos para mejorar la calidad de los datos existentes.

Mantenimiento Continuo: La teoría de control de calidad de datos reconoce que mantener la calidad de los datos es un proceso continuo. Esto implica la implementación de políticas y procedimientos para garantizar que los datos sean precisos y confiables a lo largo del tiempo.

Estándares y Reglas: Se definen estándares y reglas claras para la entrada y gestión de datos. Estos estándares ayudan a prevenir errores y aseguran que los datos se ajusten a las expectativas de calidad.

Auditoría y Monitoreo: Se establecen procesos de auditoría y monitoreo para verificar regularmente la calidad de los datos. Esto puede incluir la detección y corrección proactiva de errores.

Aplicación en el Mercado de Energía Mayorista: En el Mercado de Energía Mayorista, la teoría de control de calidad de datos desempeña un papel fundamental debido a la importancia crítica de la precisión de los datos energéticos. Errores en la información energética pueden

resultar en decisiones operativas costosas o ineficientes, y pueden afectar la confiabilidad del suministro eléctrico.

La teoría de control de calidad de datos en este contexto podría incluir:

Validación de Datos: Verificación de que los datos energéticos cumplen con los estándares y requisitos establecidos.

Limpieza de Datos: Identificación y corrección de errores y valores atípicos en los datos energéticos.

Integración de Datos: Asegurarse de que los datos de diversas fuentes se integren de manera coherente y precisa.

Auditoría Continua: Establecimiento de procesos de auditoría para monitorear y mantener la calidad de los datos a medida que se actualizan y cambian.

La teoría de control de calidad de datos es esencial para garantizar que los datos utilizados en el Mercado de Energía Mayorista sean precisos y confiables, lo que a su vez contribuye a una gestión y operación más eficientes y confiables en la industria energética. (Olson, "Data Quality: The Accuracy Dimension")

3.2.6 Teoría de Operación y Gestión Energética

La teoría de operación y gestión energética es un conjunto de principios y prácticas que se aplican en el ámbito de la industria energética para garantizar un suministro de energía eficiente, confiable y sostenible. Esta teoría se basa en la optimización de los recursos energéticos, la toma de decisiones informadas y la gestión eficaz de los activos energéticos.

Conceptos Clave de la Teoría de Operación y Gestión Energética:

Eficiencia Energética: La teoría de operación y gestión energética busca maximizar la eficiencia en la producción, transmisión y distribución de energía. Esto implica la reducción de pérdidas y el uso eficiente de los recursos energéticos.

Monitorización y Control: Se hace hincapié en la monitorización en tiempo real de los sistemas energéticos para identificar desviaciones y tomar medidas correctivas de manera oportuna. El control avanzado permite una gestión precisa de la oferta y la demanda de energía.

Gestión de la Demanda: La teoría incluye estrategias para gestionar la demanda de energía, como la implementación de tarifas dinámicas y la promoción de la eficiencia energética en el lado del consumidor.

Integración de Energías Renovables: A medida que las energías renovables desempeñan un papel creciente en el suministro energético, la teoría de operación y gestión energética aborda cómo integrar de manera eficiente estas fuentes intermitentes en la red eléctrica.

Planificación Energética: Involucra la planificación a largo plazo del suministro energético, considerando factores como la capacidad, la expansión de la infraestructura y la diversificación de fuentes de energía.

Aplicación en el Mercado de Energía Mayorista:

En el Mercado de Energía Mayorista, la teoría de operación y gestión energética es esencial para asegurar un suministro de energía confiable y económico. Algunas aplicaciones clave en este contexto incluyen:

Programación de la Generación: Determinación de cuánta energía debe generarse y cuándo para satisfacer la demanda prevista.

Gestión de la Demanda: Implementación de programas de gestión de la demanda que incentiven a los consumidores a reducir el consumo en momentos de alta demanda.

Integración de Energías Renovables: Incorporación eficiente de la energía solar, eólica y otras fuentes renovables en la matriz energética.

Operación de Redes de Transmisión: Mantenimiento de la confiabilidad y la seguridad de las redes de transmisión de energía, incluyendo la gestión de la congestión.

La teoría de operación y gestión energética es esencial para garantizar que el suministro de energía en el Mercado de Energía Mayorista sea confiable y eficiente. Estos principios ayudan a optimizar la producción y distribución de energía, lo que a su vez tiene un impacto positivo en la economía y la sostenibilidad de la industria energética. (Abhijit Chakrabarti y Sunita Halder, "Power System Operation and Control ").

3.2.7 Modelo de Calidad de Datos en el Contexto de XM

El modelo de Calidad de Datos representa los cimientos sobre los cuales se construye una solución para la medición del estado de los datos dentro de los sistemas de información de una organización, este comprende un conjunto de tablas bajo un esquema relacional que permiten dinamizar la parametrización y ejecución de reglas de negocio alineadas con una necesidad de evaluación.

3.3 Tecnologías Involucradas en el Desarrollo del Proyecto

En la implementación de este proyecto de grado, se hará uso de diversas tecnologías que desempeñarán un papel fundamental en el diseño, desarrollo y despliegue de la aplicación web diseñada para el control y ejecución de modelos de calidad de datos en el Mercado de Energía Mayorista. A continuación, se detallan estas tecnologías clave:

Angular (Frontend): Angular, un framework de desarrollo de aplicaciones web de Google, será la base para la creación de la interfaz de usuario de la aplicación.

Su robusto sistema de componentes y herramientas de desarrollo permitirá la construcción de una interfaz dinámica y amigable para los usuarios finales.

Node.js (Backend): La lógica del servidor será implementada utilizando Node.js, un entorno de ejecución de JavaScript en el lado del servidor. Node.js proporciona una plataforma eficiente y escalable para gestionar las solicitudes de los usuarios, así como para interactuar con la base de datos y los modelos de calidad de datos.

SQL Server (Base de Datos): La base de datos SQL Server de Microsoft será utilizada para almacenar y administrar los datos energéticos críticos. Proporcionará un entorno confiable y escalable para garantizar la integridad y disponibilidad de los datos.

SonarQube (Métricas de Pruebas y Calidad del Código): SonarQube desempeñará un papel esencial en la evaluación de la calidad del código fuente, asegurando que el desarrollo se realice siguiendo las mejores prácticas y estándares de calidad.

Postman (Realización de Peticiones al Backend): La herramienta Postman se utilizará para probar y validar la funcionalidad del backend, garantizando una interacción eficiente y confiable entre el frontend y el servidor.

GitLab (Almacenamiento de Código en Repositorio): GitLab servirá como el repositorio centralizado para el almacenamiento y colaboración en el código fuente del proyecto, facilitando la colaboración y el control de versiones.

Docker (Contenedores): Docker permitirá empaquetar la aplicación y sus dependencias en contenedores independientes, lo que simplificará el proceso de implementación y garantizará la portabilidad de la aplicación en diferentes entornos.

Azure (Despliegue en Producción): Microsoft Azure se utilizará como plataforma de nube para el despliegue en producción de la aplicación, proporcionando un entorno seguro, escalable y confiable.

Swagger (Documentación y Pruebas de API): Swagger se utilizará para diseñar y documentar claramente la API de la aplicación. Facilitará la generación de documentación interactiva, permitirá a los desarrolladores probar y validar las solicitudes y respuestas de la API, y garantizará una comunicación efectiva entre los equipos de desarrollo, testing y documentación.

Estas tecnologías se seleccionaron cuidadosamente para garantizar un desarrollo efectivo y eficiente de la aplicación, así como para cumplir con los estándares de calidad y confiabilidad requeridos para el proyecto en el Mercado de Energía Mayorista. Su integración y uso apropiado desempeñarán un papel crítico en el éxito general del proyecto.

4. Metodología

El desarrollo de las actividades para este proyecto se hizo utilizando el modelo de prototipado evolutivo. Este modelo hace referencia a la ejecución de actividades concernientes al desarrollo de software en forma modular; esto quiere decir que se va a desarrollar el software por módulos, cada uno con su ciclo completo (Pressman, 2002). El uso de este modelo metodológico permite un mayor entendimiento del propósito, diseño y requerimientos del sistema, obteniendo así un sistema basado en las necesidades reales del usuario. El uso de este modelo brinda las siguientes ventajas:

- El usuario final obtiene resultados parciales de su producto.
- Los riesgos de integración del producto se mitigan en etapas tempranas porque la integración se va dando de forma progresiva conforme se va culminando cada módulo.
- El producto de software se puede ir afinando y mejorando en cada iteración.
- Se puede optar por la reutilización de componentes en cada iteración y mejorarlos.

Desventajas:

- Al no priorizarse los requisitos de manera integral desde el inicio, pueden surgir problemas posteriores en la arquitectura.
- El usuario piensa que el producto ya se encuentra terminado en las primeras entregas (Sommerville, 2002).

Los módulos propuestos para el desarrollo de este proyecto son los siguientes:

4.1 Módulos Propuestos para el Desarrollo del Proyecto

1. **Necesidades del Usuario (Frontend y Backend):** Se llevará a cabo una fase inicial de investigación para identificar las necesidades y requerimientos de los usuarios, tanto en la parte del frontend como en el backend. Esto garantizará que la aplicación cumpla con las expectativas de los usuarios y las necesidades del Mercado de Energía Mayorista.
2. **Prototipos para las Vistas:** Se crearán prototipos detallados de las vistas de la aplicación, incluyendo esquemas de colores, diseño de la interfaz de usuario, iconos y otros elementos visuales. Estos prototipos servirán como guía para el desarrollo posterior.
3. **Desarrollo de Microservicios (Backend y Frontend):** El desarrollo de la aplicación comenzará simultáneamente en el frontend y el backend. Los microservicios del backend se diseñarán y desarrollarán para gestionar la lógica de negocio y el manejo de datos, mientras que el frontend se centrará en la interfaz de usuario y la interacción con el usuario final.
4. **Documentación de Endpoints en Swagger:** Se utilizará Swagger para documentar los endpoints de la API que conecta el frontend y el backend. Esto proporcionará una referencia clara de cómo interactuar con la aplicación y permitirá una mejor colaboración entre equipos.
5. **Realización de Pruebas Funcionales:** Se diseñarán y ejecutarán pruebas funcionales para validar que la aplicación cumple con los requisitos definidos por los usuarios. Esto incluye pruebas de funcionalidad en ambas partes, frontend y backend.

6. **Creación de Tablas o Columnas (si es necesario):** En caso de que la estructura de la base de datos requiera modificaciones o adiciones de tablas o columnas, se realizarán en esta etapa para asegurar la correcta organización de los datos.
7. **Realización de Pruebas Unitarias:** Se llevarán a cabo pruebas unitarias para evaluar el funcionamiento de componentes individuales tanto en el frontend como en el backend. Esto garantiza que cada parte de la aplicación funcione correctamente.
8. **Despliegue en Contenedores Docker:** La aplicación se desplegará en contenedores Docker para facilitar su implementación y escalabilidad. Docker asegura que la aplicación se ejecute de manera consistente en diferentes entornos.
9. **Documentación del Código:** Se proporcionará una documentación completa del código fuente, incluyendo comentarios, guías de instalación y uso de la aplicación. Esto permitirá a los usuarios comprender y utilizar el software de manera efectiva.
10. **Paso a Ambiente de Pruebas:** La aplicación se moverá a un entorno de pruebas donde se realizarán pruebas exhaustivas en un ambiente controlado antes de su lanzamiento oficial.
11. **Validaciones de Seguridad y Pruebas de Vulnerabilidad:** En esta fase crítica antes del paso a producción, se llevarán a cabo una serie de validaciones de seguridad y pruebas de vulnerabilidad para garantizar que la aplicación sea robusta, resistente a ataques y cumpla con los estándares de seguridad. Este módulo incluirá**:** Pruebas de Inserción de Datos Segura, Validaciones de Datos de Entrada, Control de Acceso y Autorización, Análisis de Vulnerabilidades y Auditoría de Registros**.**

12. **Solución de Bugs (si es necesario):** En caso de identificar problemas o errores durante las pruebas en el ambiente de pruebas, se corregirán y se realizarán nuevas pruebas hasta que la aplicación sea estable y funcional.
13. **Paso a Producción:** Finalmente, la aplicación se desplegará en un entorno de producción, donde estará disponible para su uso por parte de los usuarios finales en el Mercado de Energía Mayorista.

Estos módulos proporcionan un enfoque estructurado y completo para el desarrollo de la aplicación web, asegurando que se satisfagan las necesidades de los usuarios y que la aplicación se entregue de manera eficiente y confiable.

5. Requerimientos

5.1 Requerimientos Funcionales

5.1.1 Iniciar Sesión

Descripción: Permitir a los usuarios autenticarse en la aplicación proporcionando su nombre de usuario y contraseña.

Precondiciones: El usuario debe tener credenciales válidas registradas en el sistema.

Postcondiciones: Si las credenciales son correctas, el usuario accede al sistema y se le redirige a la página principal.

5.1.2 Restablecer Contraseña

Descripción: Brindar a los usuarios la capacidad de restablecer su contraseña en caso de olvido.

Precondiciones: El usuario debe haber registrado una dirección de correo electrónico válida en el sistema.

Postcondiciones:

- Se envía un enlace de restablecimiento de contraseña al correo electrónico del usuario.
- El usuario puede establecer una nueva contraseña utilizando el enlace proporcionado.

5.1.3 Gestión de Dimensiones de Calidad

Una dimensión de calidad de datos es una característica del dato que permite analizar y medir si cumple con unos requisitos específicos del sistema.

5.1.3.1 Creación y Edición de Dimensiones de Calidad.

Descripción: Permitir a los usuarios crear y editar dimensiones de calidad.

Campos Obligatorios: Nombre de Calidad, Descripción.

5.1.3.2 Visualización de Dimensiones de Calidad.

Descripción: Mostrar las dimensiones de calidad en una tabla.

Campos que Mostrar: Nombre de Calidad, Descripción.

5.1.3.3 Eliminación de Dimensiones de Calidad.

Descripción: Permitir a los usuarios eliminar dimensiones de calidad existentes.

Precondiciones: El usuario debe haber iniciado sesión en el sistema y la dimensión de calidad a eliminar debe existir en la base de datos.

Postcondiciones: La dimensión de calidad eliminada ya no está disponible en el sistema.

5.1.4 Gestión de Estados de Regla

Almacena el tipo de estado que puede tener la regla de calidad.

5.1.4.1 Creación de Estado de Regla.

Descripción: Permite a los usuarios crear nuevos estados para las reglas.

Campos obligatorios: Estado y Nombre de Estado.

Restricción: El campo "Estado" debe tener un máximo de 1 carácter.

5.1.4.2 Edición de Estado de Regla.

Descripción: Permite a los usuarios modificar la información de los estados existentes.

Campos modificables: Estado y Nombre de Estado.

5.1.4.3 Listado de Estados de Regla.

Descripción: Muestra una tabla con los estados disponibles y sus respectivos nombres.

5.1.4.4 Eliminación de Estados de Regla.

Descripción: Permite a los usuarios eliminar estados existentes que no estén asociados a reglas activas.

Restricción: No se puede eliminar un estado que esté asociado a reglas activas.

5.1.5 Gestión de Fuentes

En este ítem se administran las bases de datos a las cuales se puede conectar el sistema para consultar la información y aplicar las reglas de calidad.

5.1.5.1 Creación y Edición de Fuentes.

Descripción: Permite a los usuarios crear y editar fuentes de datos para su posterior utilización en la consulta de información y aplicación de reglas.

Campos Obligatorios: Nombre de la Fuente, Descripción.

5.1.5.2 Eliminación de Fuentes.

Descripción: Permite a los usuarios eliminar fuentes existentes que no estén asociadas a reglas activas.

Restricción: No se puede eliminar una fuente que esté asociada a reglas activas.

5.1.5.3 Listado de Fuentes.

Descripción: Muestra una tabla con las fuentes existentes, incluyendo los campos de Nombre y Descripción.

5.1.6 Gestión de maestras

Una maestra es una entidad de negocio sobre la cual se pueden aplicar reglas de calidad para medir el estado de sus datos de manera general o a nivel de atributo, por lo general una maestra está relacionada a una tabla de base de datos en la fuente, y sus atributos corresponden con las columnas de la tabla

5.1.6.1 Creación de Maestras.

Descripción:

Paso 1: Datos Generales.

Los usuarios deben ingresar el nombre de la maestra, su prioridad, el área de negocio asociada y una descripción.

Campos Obligatorios: Nombre de la Maestra, Prioridad, Área de Negocio, Descripción.

Paso 2: Atributos.

Los usuarios deben agregar al menos un atributo a la maestra.

Campos Obligatorios del Atributo: Nombre del Atributo, Criticidad, Descripción.

Paso 3: Notificaciones (Opcional).

Los usuarios pueden opcionalmente agregar correos electrónicos para notificaciones relacionadas con reglas en la dimensión de consistencia.

5.1.6.2 Edición de Datos Generales de la Maestra.

Descripción: Los usuarios pueden modificar los cuatro campos obligatorios de la maestra (nombre, prioridad, área de negocio y descripción), siempre y cuando todos estén diligenciados.

Edición de Atributos: Los usuarios pueden desplegar la lista de atributos asociados a la maestra para agregar, modificar e inactivar atributos.

Para la modificación de atributos, se aplican los mismos requisitos que en la creación:

- Mínimo un atributo asociado.
- Todos los campos del atributo deben ser completamente diligenciados, ya que son obligatorios.

5.1.7 Gestión de Umbrales de calidad

Pantalla donde se parametrizarán los porcentajes de calidad que se tendrán en cuenta en un tablero de power bi, para mediante colores indicar el grado de cumplimiento del indicador.

5.1.7.1 Visualización y Edición de Umbrales.

Descripción: Los usuarios pueden visualizar y editar los umbrales de calidad en esta sección. Estos umbrales impactan directamente en la calidad de las maestras reflejadas en el tablero de Power BI (la responsabilidad del desarrollo del tablero no recae en el desarrollador de la aplicación).

Restricciones en la Edición: Los umbrales rara vez son modificados, pero si es necesario, deben seguir ciertas reglas lógicas para garantizar la coherencia en la representación gráfica en el tablero de Power BI.

Condiciones Lógicas de los Umbrales: Para la correcta visualización de la gráfica de la media dona, el valor mínimo debe ser menor o igual al valor máximo de cada umbral.

Para respetar la lógica de los umbrales, se deben cumplir las siguientes condiciones: umbral bajo <= umbral medio <= umbral alto.

Los valores de los umbrales deben estar en el rango de 0 a 1, y la aplicación ya impone estas restricciones.

Modificación de Umbrales Medio, Bajo y Alto: En el umbral bajo, únicamente puede modificarse el valor máximo. Este valor máximo será el valor mínimo del umbral

medio. En el umbral alto, únicamente puede modificarse el valor mínimo. Este valor mínimo será el valor máximo del umbral medio.

Si se requiere modificar tanto el valor mínimo como el valor máximo del umbral medio, el usuario deberá ajustar el umbral bajo y alto respectivamente para evitar inconsistencias en los datos.

5.1.8 Gestión de Reglas de Calidad

Validación de negocio parametrizada mediante scripts de base de datos que se ejecutan dinámicamente mediante microservicios de Python sobre las fuentes de datos con el objetivo de calcular un porcentaje de calidad del dato de una Maestra (tabla) en un Atributo (campo) para una dimensión de calidad específica.

5.1.8.1 Creación Regla de Calidad.

Descripción

Primer Paso: El usuario diligencia seis campos obligatorios, incluyendo maestra, atributo relacionado, tipo de regla, dimensión, código de identificación (único y autoincremental), y descripción.

Campos opcionales: estado (por defecto activo) y validar dominio correo.

La selección del estado inactivo se traduce en la creación de la regla como inactiva.

El campo de maestra es de tipo sugerencia, facilitando la búsqueda mediante caracteres ingresados.

Paso Especial para Dimensión CONSISTENCIA: Cuando se selecciona CONSISTENCIA, se despliega una carta adicional llamada "Detalle consistencia".

Se puede optar por validar el detalle de consistencia, y se presentan opciones para la comparación desde la fuente base o desde otras fuentes.

Si se desea comparar desde ambas partes, se deben crear dos reglas.

Segundo Paso: El usuario decide crear scripts o no, con campos obligatorios como script, fuente, estado, y observación.

El check de población base solo puede ser uno entre todos los scripts de la regla.

Para la dimensión CONSISTENCIA, no se manipula el dato de población base en la base de datos.

Importante: Si se seleccionó CONSISTENCIA, el usuario puede agregar más de dos scripts. El script de calidad solo puede contener ciertos campos como código, nombre, fechaIni, fechaFin, estado, valor e inconsistencia.

Guardado y Auditoría de Calidad: Al hacer clic en la opción de guardar, se abrirá una pop-up solicitando la información sobre quién solicitó crear la regla de calidad. Después de confirmar, la regla de calidad se crea y es visible en la tabla después de actualizarla. El icono de auditoría de calidad permite observar quién solicitó la creación y la fecha correspondiente.

5.1.8.2 Edición de Reglas de Calidad.

Descripción: Para editar la información de las reglas de calidad previamente guardadas, se siguen los siguientes pasos:

Modificación de Campos Obligatorios y Opcionales: El usuario puede modificar los seis campos obligatorios de la regla de calidad, siempre y cuando estén diligenciados.

La modificación de los dos campos opcionales también está permitida en este paso.

Detalle Consistencia y Scripts: El usuario puede desplegar la carta de "Detalle Consistencia" solo si aplica según las condiciones establecidas previamente.

También puede desplegar la carta de "Scripts", cumpliendo con los requisitos de la creación.

Si cumple con todos los requisitos, la aplicación le permite realizar modificaciones en la regla de calidad.

Confirmación de Edición: Al hacer clic en "Guardar", aparece una pop-up de confirmación de edición de la regla de calidad.

Debe ingresar quién solicitó la modificación y su observación en la pop-up de confirmación.

Después de confirmar, la aplicación procede a modificar la regla de calidad.

5.1.8.3 Inactivación de Reglas de Calidad.

Descripción: Para inactivar un registro de regla de calidad, se deben seguir los siguientes pasos:

Acceso a la Opción de Inactivar: El usuario debe ubicarse en la regla de calidad deseada.

Hacer clic en el icono de inactivar que aparece al lado izquierdo en la columna de acciones.

Confirmación de Inactivación: Después de hacer clic, aparecerá una pop-up preguntando al usuario si está seguro de inactivar la regla de calidad.

Confirmación de Usuario: Al seleccionar "Sí", se abrirá otra pop-up solicitando al usuario que ingrese quién solicitó la inactivación.

Inactivación del Estado: Al dar clic en el botón de confirmar, la aplicación procede a cambiar el estado de la regla de calidad de "Activa" a "Inactiva".

Se reflejará el cambio en la tabla, mostrando la letra "I" en la columna de estado para indicar que la regla está inactiva.

5.1.8.4 Activación de Reglas de Calidad.

Descripción: Para inactivar un registro de regla de calidad, se deben seguir los siguientes pasos:

Acceso a la Opción de Inactivar:

El usuario debe ubicarse en la regla de calidad deseada.

Hacer clic en el icono de inactivar que aparece al lado izquierdo en la columna de acciones.

Confirmación de Inactivación:

Después de hacer clic, aparecerá una pop-up preguntando al usuario si está seguro de inactivar la regla de calidad.

Confirmación de Usuario:

Al seleccionar "Sí", se abrirá otra pop-up solicitando al usuario que ingrese quién solicitó la inactivación.

También se le pedirá que exponga las razones en el apartado de observaciones.

Inactivación del Estado:

Al dar clic en el botón de confirmar, la aplicación procede a cambiar el estado de la regla de calidad de "Activa" a "Inactiva".

Se reflejará el cambio en la tabla, mostrando la letra "I" en la columna de estado para indicar que la regla está inactiva.

5.1.9 Gestión de Tipos de Reglas

Corresponde a los tipos de reglas que se desean manejar dentro del proyecto para sectorizar las mismas.

5.1.9.1 Creación y Edición de Tipos de Reglas.

Descripción: Permite a los usuarios crear y editar tipos de reglas para su posterior utilización en la consulta de información y aplicación de reglas.

Campos Obligatorios: Tipo de Regla.

5.1.9.2 Eliminación de Tipos de Reglas.

Descripción: Permite a los usuarios eliminar tipos de reglas existentes que no estén asociadas a reglas activas.

Restricción: No se puede eliminar un tipo de regla que esté asociada a reglas activas.

5.1.9.3 Listado de Tipos de Reglas.

Descripción: Muestra una tabla con los tipos de reglas existentes, incluyendo los campos de Tipo de Regla y Fecha Actualización.

5.1.10 Ejecutar Microservicios

Descripción: Permitir a los usuarios ejecutar microservicios a demanda de una API desarrollada por XM para el manejo de los modelos de calidad en la aplicación proporcionando el tipo de integración. Importante: Solo se enviará la información que permite ejecutar los microservicios, la aplicación web no interfiere en el proceso interno de estos.

Precondiciones: El usuario debe tener credenciales válidas registradas en el sistema y seleccionar un tipo de integración valido.

Postcondiciones: Llamar a los microservicios de la API proporcionada por XM y retornar un tooltip dependiendo del resultado de la ejecución de este.

5.1.11 Gestión de Autenticación y Autorización

5.1.11.1 Creación y Edición de Grupos/Roles.

Descripción: Permitir a los usuarios crear y editar grupos/roles en los cuales se definirán los permisos operacionales que tienen ellos dentro del mismo.

Campos Obligatorios: Rol, permisos operacionales tipo checkbox.

5.1.11.2 Eliminación de Grupos/Roles.

Descripción: Permite a los usuarios eliminar grupos/roles existentes que no estén asociados a usuarios creados.

Restricción: No se puede eliminar un grupo/rol que esté asociada a usuarios existentes.

5.1.11.3 Listado de Grupos/Roles.

Descripción: Muestra una tabla con los grupos/roles existentes, incluyendo los campos de nombre y estado.

5.1.11.4 Creación y Edición de Usuarios.

Descripción: Permite a los usuarios crear y editar otros usuarios que podrán acceder a la aplicación web.

Campos Obligatorios: Nombre completo, Grupo/Rol, Usuario, Contraseña, correo electrónico.

5.1.11.5 Eliminación de Usuarios.

Descripción: Permite a los usuarios eliminar otros usuarios existentes.

Restricción: No se permite eliminar el usuario admin.

5.1.11.6 Listado de Usuarios.

Descripción: Muestra una tabla con los usuarios existentes, incluyendo los campos de Nombre Completo, correo electrónico, estado y fecha de actualización.

5.1.12 Auditoria

Descripción: Permitir a los usuarios ver el rastro de acciones que se ejecutan sobre los datos y objetos del modelo de calidad para hacer seguimiento y auditoría.

Precondiciones: El usuario debe tener credenciales válidas registradas en el sistema y permisos correspondientes para la visualización de esta información.

Postcondiciones: Si las credenciales son correctas, el usuario podrá acceder al rastro de acciones de la aplicación web, en las cuales se mostrará como información el usuario asociado a la acción, la pantalla en la cual se realizó la acción, una observación, la fecha y hora en la cual se realizó dicha acción y la acción CRUD realizada.

5.2 Requerimientos No Funcionales

5.2.1 Seguridad

Descripción: Garantizar la seguridad del proceso de inicio de sesión y restablecimiento de contraseña.

Criterios de Cumplimiento: Las contraseñas deben almacenarse de forma segura y el proceso de restablecimiento debe requerir autenticación adicional.

5.2.2 Usabilidad

Descripción: Facilitar la experiencia del usuario durante el proceso de inicio de sesión y restablecimiento de contraseña.

Criterios de Cumplimiento: La interfaz debe ser intuitiva, y las instrucciones para restablecer la contraseña deben ser claras y fáciles de seguir.

5.2.3 Disponibilidad

Descripción: Garantizar la disponibilidad del sistema durante el proceso de inicio de sesión y restablecimiento de contraseña.

Criterios de Cumplimiento: El sistema debe estar disponible y responder de manera eficiente durante los procesos mencionados.

5.2.4 Validación de Campos

Descripción: Garantizar que los campos obligatorios al crear o editar dimensiones de calidad estén correctamente llenados.

Criterios de Cumplimiento: El sistema debe validar y alertar al usuario si intenta guardar una dimensión sin completar los campos obligatorios.

5.2.5 Presentación de Datos

Descripción: Asegurar una presentación clara y ordenada de las dimensiones de calidad en la tabla.

Criterios de Cumplimiento: La tabla debe ser fácilmente legible y mostrar los campos especificados de manera ordenada.

5.2.6 Eficiencia en el Procesamiento

Descripción: Garantizar un procesamiento eficiente al crear, editar y visualizar dimensiones de calidad.

Criterios de Cumplimiento: Las operaciones mencionadas deben completarse de manera rápida y eficiente, sin demoras significativas.

5.2.7 Estética del Sistema

Descripción: Asegurar una interfaz visual coherente y alineada con la identidad corporativa de la empresa.

Criterios de Cumplimiento:

- El sistema deberá adaptarse a la paleta de colores oficial de la empresa.
- La tipografía utilizada en la interfaz del sistema seguirá las pautas establecidas por la identidad visual de la empresa.
- Se buscará la armonización estética de los elementos visuales para fortalecer la identidad corporativa.
- Se mantendrá una consistencia visual en toda la interfaz del sistema, asegurando una experiencia de usuario unificada y profesional.

6. Desarrollo del Proyecto

En esta sección, se presenta una visión detallada del proceso de desarrollo del proyecto, desde su fase conceptual hasta la materialización de una aplicación web funcional.

6.1 Propuesta grafica de diseño

6.1.1 Prototipos

En la fase inicial del desarrollo, se empleó la herramienta Balsamiq para la creación de prototipos que sirvieron como los primeros bosquejos visuales de las pantallas principales de la aplicación. Este enfoque permitió esbozar de manera ágil y efectiva las interfaces de usuario, brindando una representación gráfica de la estructura y disposición de los elementos.

Balsamiq proporciona una plataforma intuitiva y fácil de usar que permitió plasmar rápidamente las ideas iniciales en forma de prototipos. Esto facilitó una comunicación más efectiva y una comprensión visual de las primeras propuestas de diseño.

Los prototipos sirvieron como una herramienta valiosa para explorar y visualizar la disposición de los elementos en las pantallas principales. Esto ayudó a identificar posibles mejoras en la usabilidad y proporcionó una base sólida para las etapas posteriores del diseño.

Los prototipos aprobados para el desarrollo de la aplicación web fueron los siguientes:

Figura 1.

Ejemplo Paper Prototype para visualización de registros.

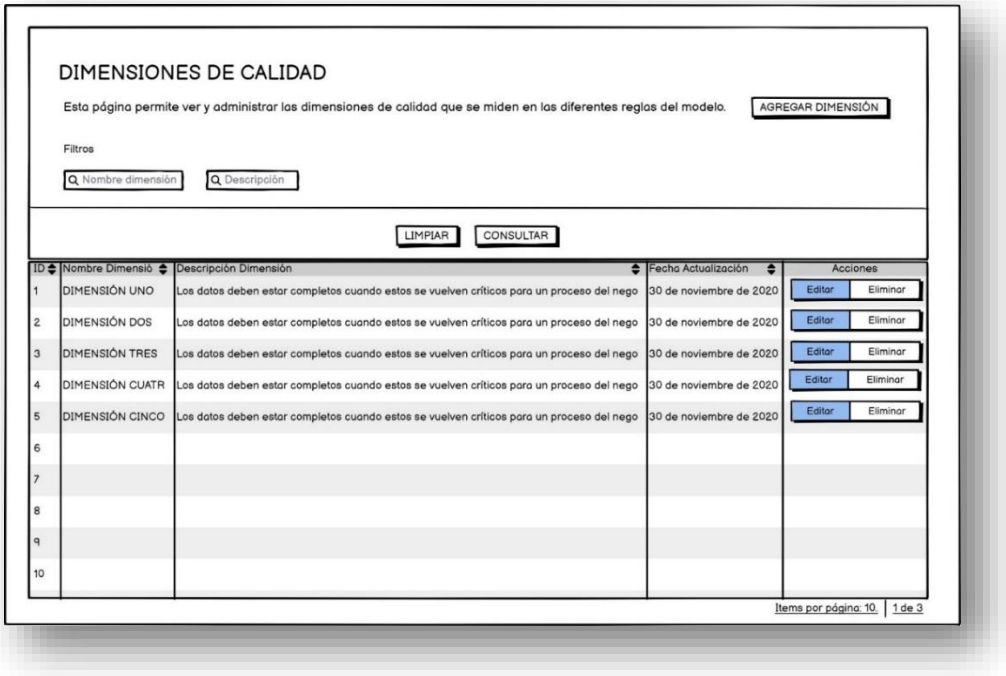


Figura 2.

Ejemplo Paper Prototype para visualización del total de registros.

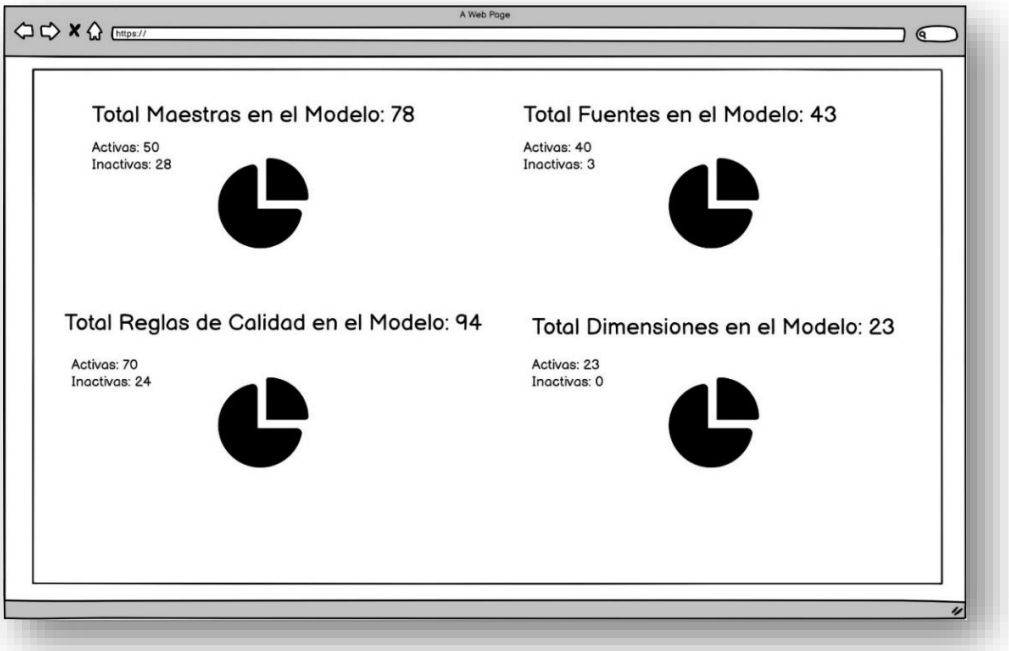


Figura 3.

Ejemplo de Paper Prototype para creación de nuevos registros.

Figura 4.

Ejemplo de Paper Prototype para creación de nuevos registros con tabla interna.

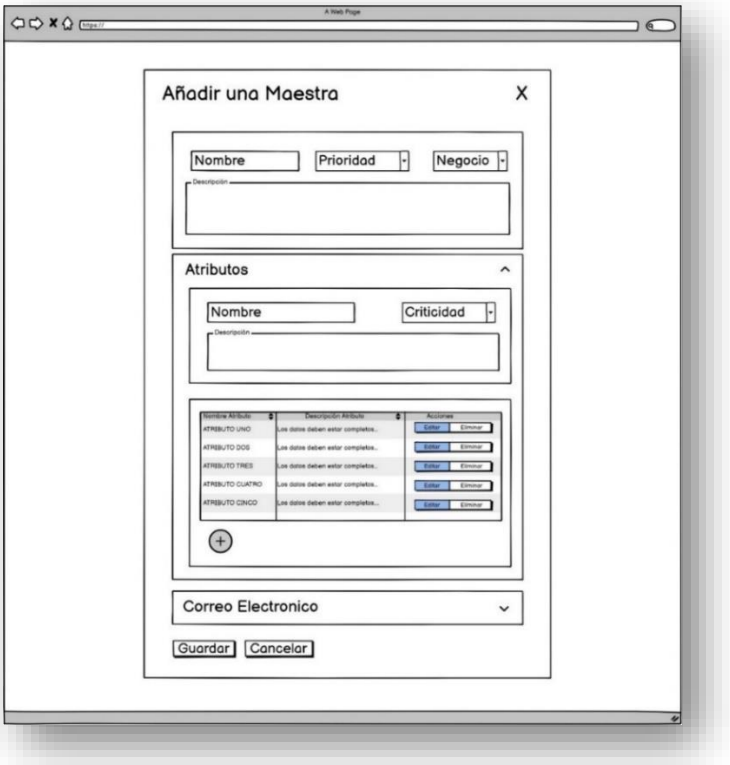


Figura 5.

Ejemplo de Paper Prototype para gestión de los umbrales de calidad.

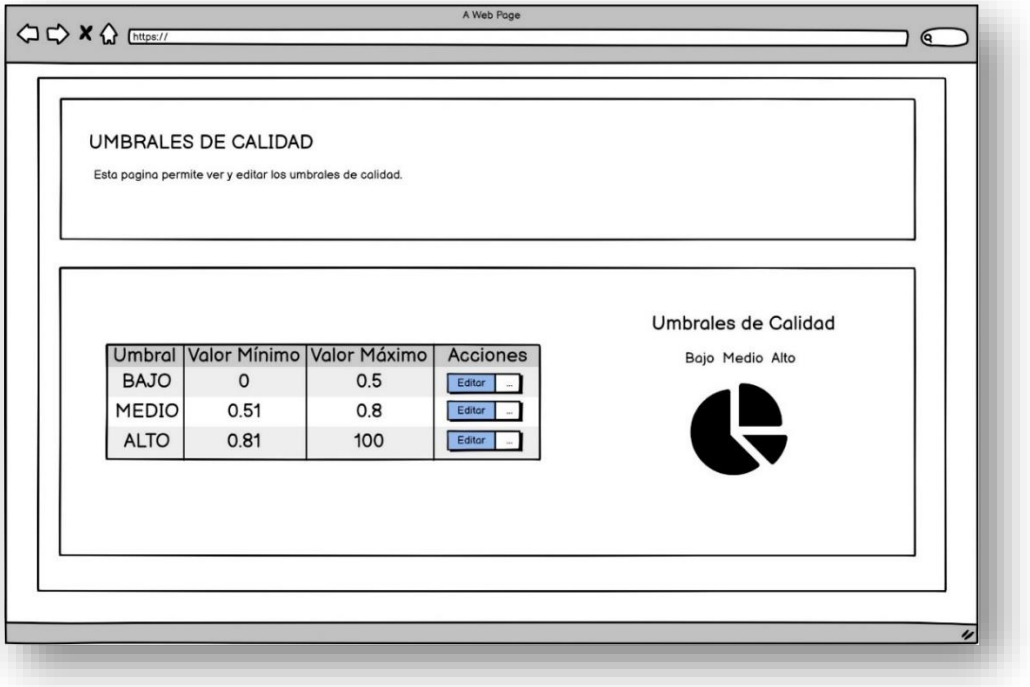


Figura 6.

Ejemplo de Paper Prototype para visualización del historial de cambios de un registro.

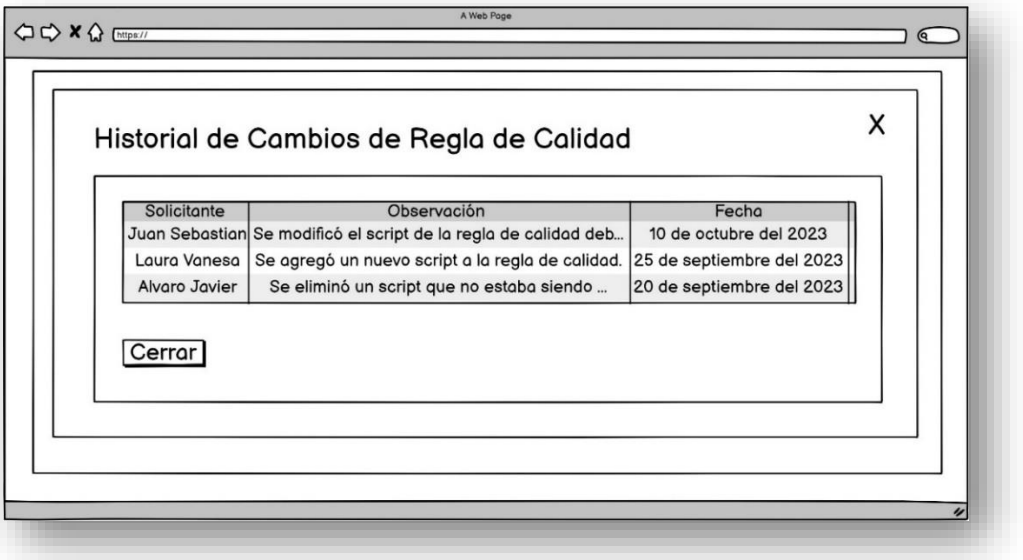


Figura 7.

Ejemplo de Paper Prototype para ventana de confirmación de cambios en un registro.

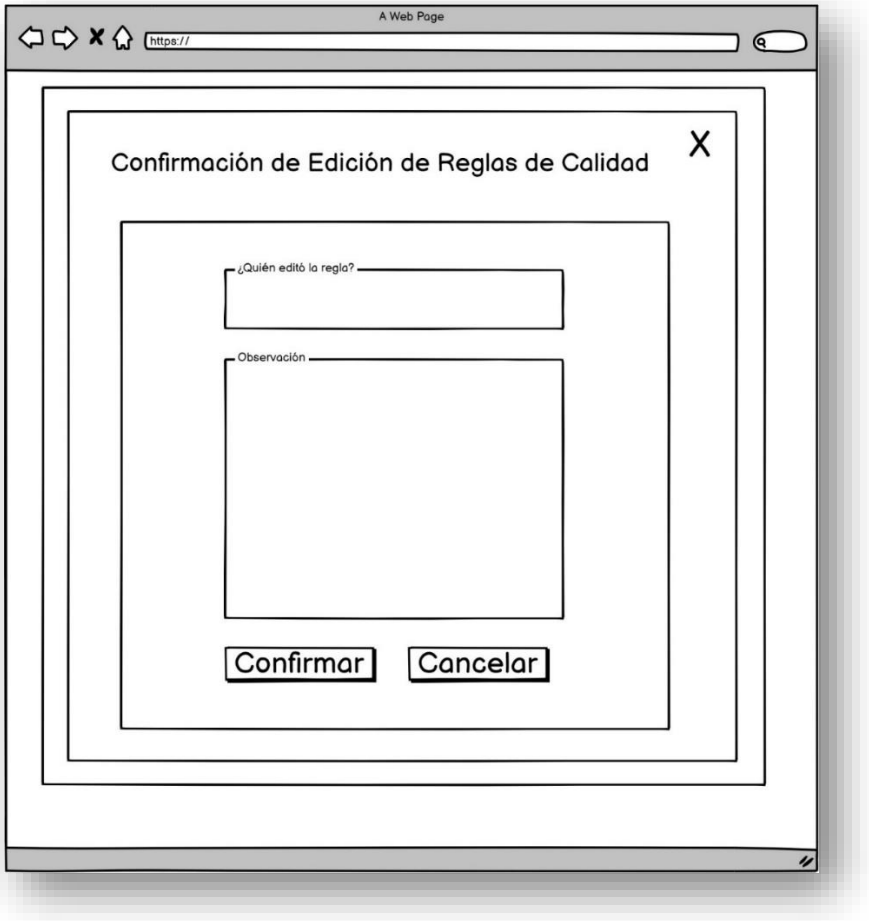


Figura 8.

Ejemplo de Paper Prototype para la pantalla de ejecución de microservicios

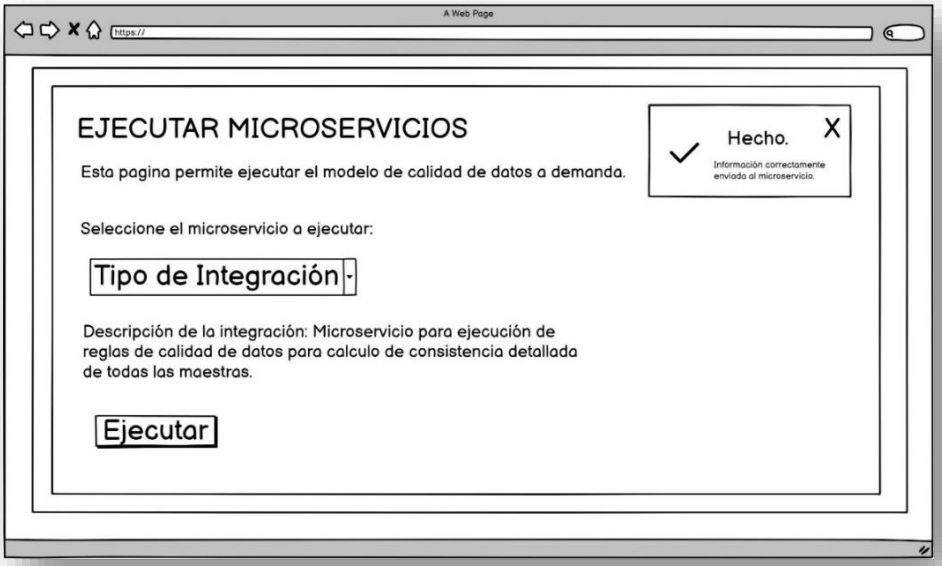


Figura 9.

Ejemplo de Paper Prototype para la pantalla de auditoría.

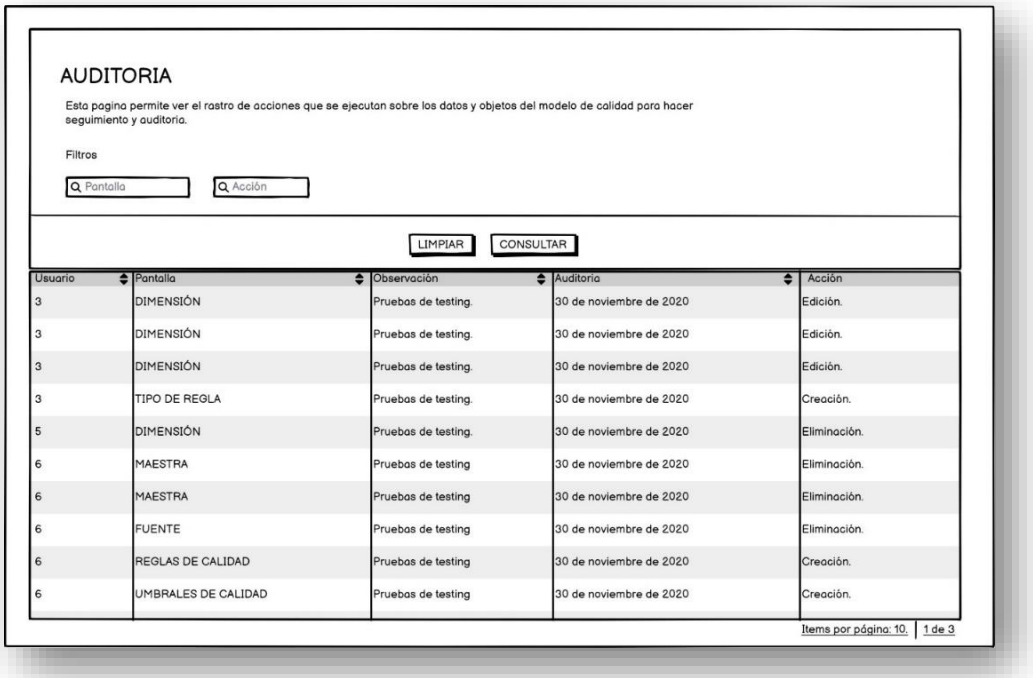


Figura 10.

Ejemplo de Paper Prototype para el inicio de sesión en la aplicación web.

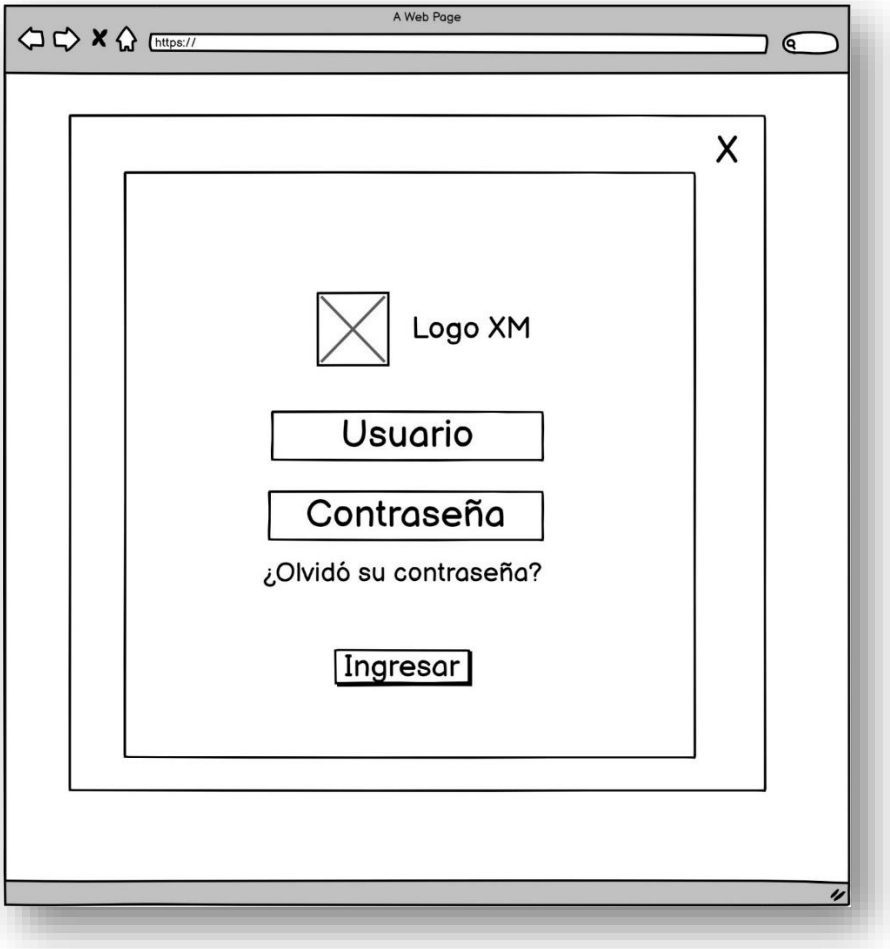


Figura 11.

Ejemplo de Paper Prototype para la barra vertical que permitirá acceder a cada una de las pantallas.

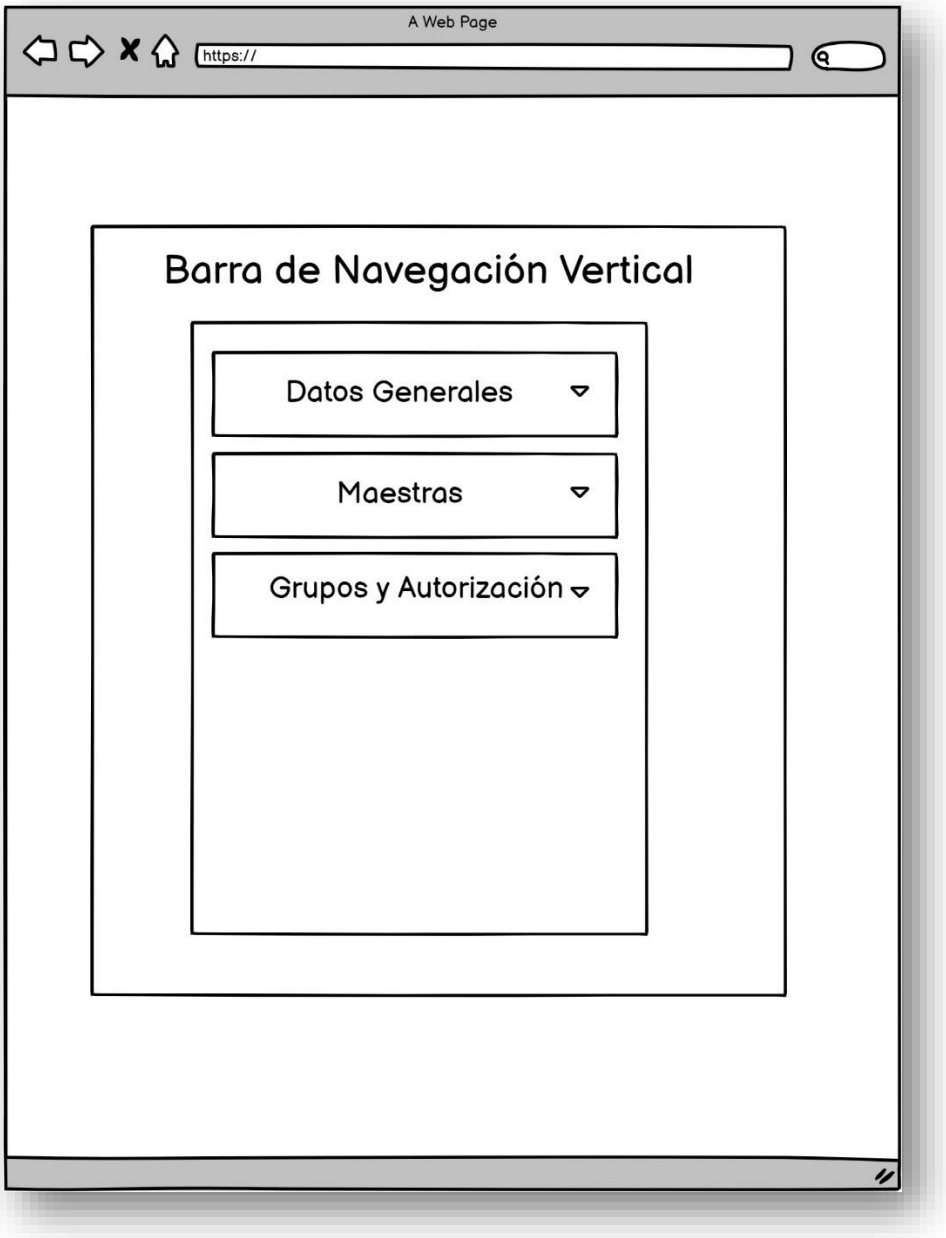


Figura 12.

Ejemplo Paper Prototype para la edición de registros que tienen tablas internas.

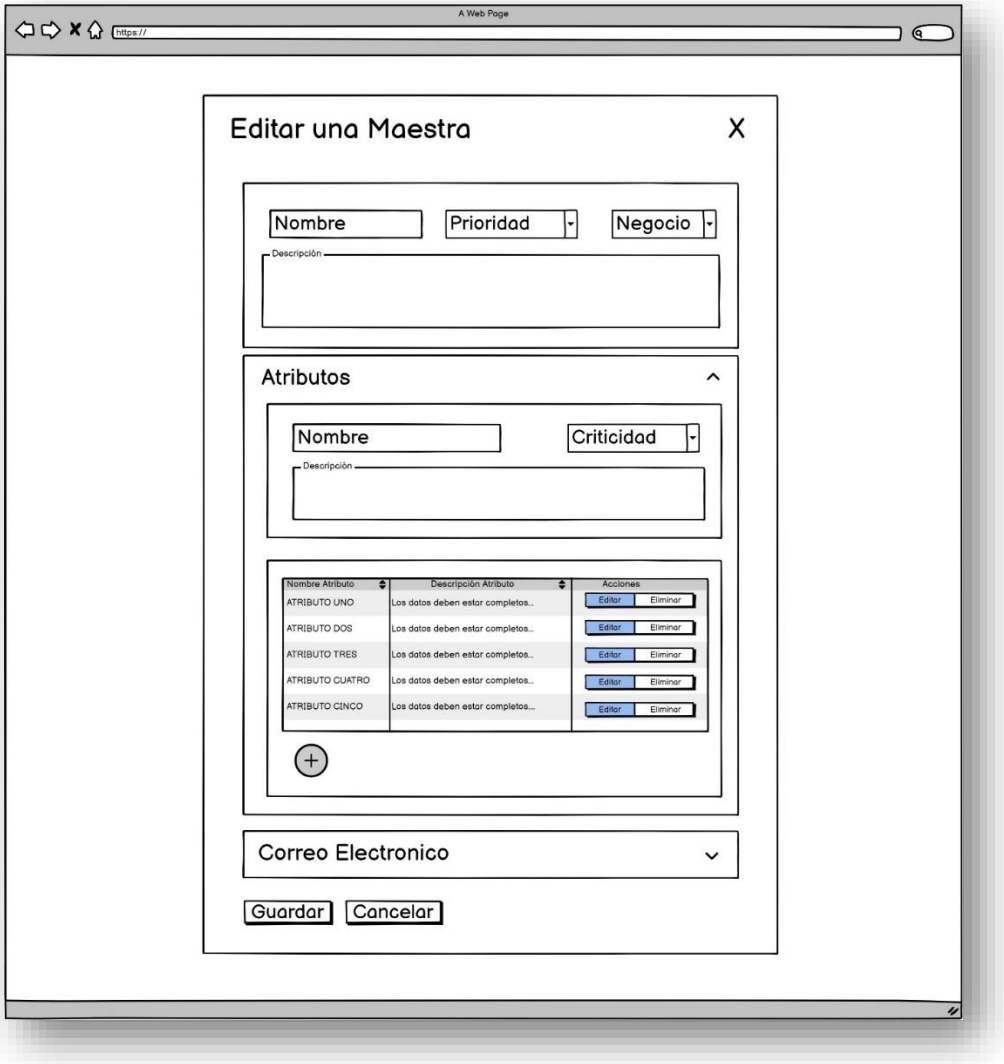
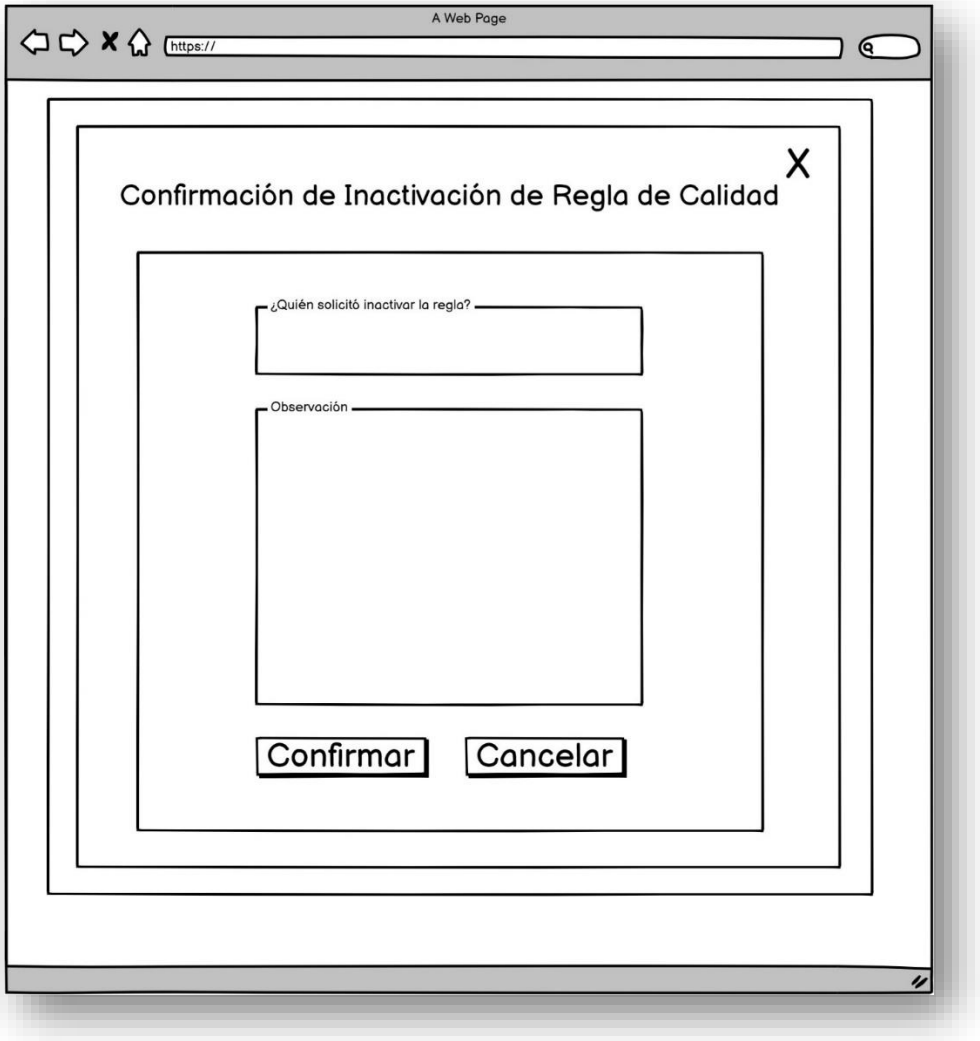


Figura 13.

Ejemplo de Paper Prototype para ventana de confirmación de inactivación de un registro.



6.1.2 Paleta de colores para el aplicativo web

Tomando como base los colores principales de XM se define una paleta para el aplicativo web.

6.1.2.1 Color principal. Se conserva como color principal el morado que desde el libro de marca indica valores a destacar como la elegancia. A su vez este color cuenta con algunos atributos a florecer como una fácil identificación para utilizarse en botones de acción, es

accesible en contraste con el color blanco y ofrece versatilidad y frescura (No satura la experiencia).

Figura 14.

Color principal de la aplicación web.

6.1.2.2 Color secundario. El color secundario es el Naranja de XM para lograr un color que acompañe las acciones principales y a su vez no lo opaque, se elige este color para poder definir algunas acciones dentro de la experiencia más no todas. A su vez este color cuenta con algunos atributos a destacar como una fácil identificación como botón de acción, es accesible en contraste con el color blanco y con texto bold, además entre sus puntos fuertes es que es un color vivo y expresa luz.

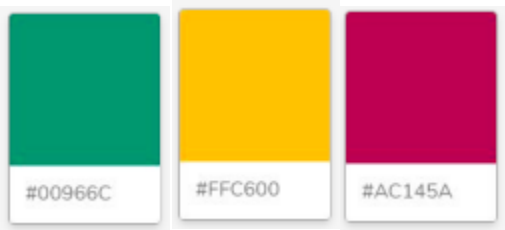
Figura 15.

Color secundario de la aplicación web.

6.1.2.3 Paleta secundaria. En este apartado se define la paleta de colores que se utilizará para apartados secundarios de la aplicación web como son los banderines de noticia, alertas del sistema y ayudas contextuales.

Figura 16.

Paleta secundaria de colores utilizados en la aplicación web.



6.1.3 Tipografía para aplicativo web

Para la tipografía del aplicativo web se escogió la tipografia Nunito, una tipografía muy versátil acorde con la marca, cuenta con bordes redondeados que la identifican como juvenil y amigable. Pero a su vez no se identifica correctamente para lectura por lo cual se realiza la unión con la tipografía Open Sans para brindar una mejor lectura y diferenciar los títulos de los párrafos con mayor facilidad.

Figura 17.

Tipografías utilizadas en la aplicación web.

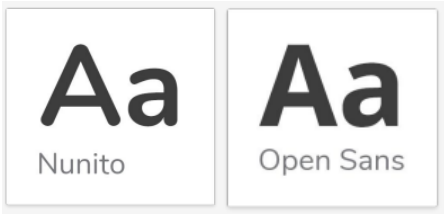
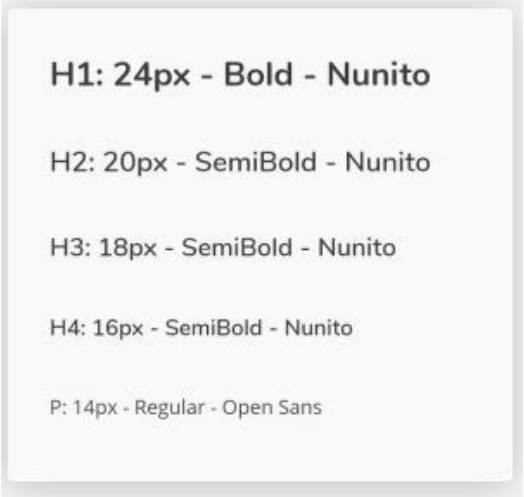


Figura 18.

Indicaciones de las implementaciones de la tipografía en la aplicación web.



6.1.4 Iconografía

Los iconos deben ser intuitivos con la acción que se desea realizar para que el usuario que utilice la aplicación no se sienta perdido, un ejemplo de ellos son los siguientes iconos:

Figura 19.

Ejemplo de iconografía utilizada en la aplicación web.

6.1.5 Prevención de errores y ayudas

Con el fin de que la aplicación sea autogestionable se crean las ayudas contextuales de modo que cuando se requiera informar al usuario sobre una recomendación previo a presentar un error en el Sistema o ya al haberlo cometido pueda recuperarse de dicho error o resolver sus dudas previamente.

6.1.6 Conclusiones aspecto gráfico y diseño

El desarrollo del diseño para la aplicación web va de la mano con la investigación realizada por parte de XM, la inducción de marca respecto a los criterios definidos previamente en su libro de identidad dieron un enfoque más holístico y no solo funcional de lo que es XM.

Dado que es una aplicación web informativa se busca el descanso en la lectura desde cuatro puntos importantes:

- El uso del blanco para generar un descanso visual e incrementar la facilidad de lectura.
- Los acentos de color para referenciar las acciones y elementos que identifican a la marca.
- La división de la lectura por bloques, esto ayuda a identificar cada una de la partes por separado. Saber cuando inician y cuando terminan.
- La jerarquía visual sobre los nombramientos de los bloques que conforman la experiencia tales como: Títulos, subtítulos, encabezados y párrafos.

6.2 Desarrollo Frontend

El punto de partida en el desarrollo fue la traslación del paper prototype inicial a un entorno tecnológico. La primera tarea se centró en la creación de una interfaz visual utilizando Angular, donde la atención primaria se dedicó a aspectos estéticos y de diseño. Este paso permitió visualizar la estructura y el flujo de la aplicación antes de incorporar la lógica funcional, brindando una representación tangible del concepto inicial.

El enfoque inicial exclusivamente estético proporcionó una sólida base visual que sirvió como referencia para el desarrollo ulterior. Con una interfaz clara y coherente en su lugar, se

procedió a inyectar funcionalidad gradualmente, asegurando que cada componente operara de manera eficiente y contribuyera al conjunto general de la aplicación.

Específicamente, la primer pantalla desarrollada y migrada fue Dimensiones de calidad. Las siguientes imágenes ilustrarán el estándar que se siguió para la pantalla de dimensiones de calidad, así como las características que esta pantalla ofrece a los usuarios. Este enfoque metodológico se replicó para el resto de las pantallas de manera consistente.

6.2.1 Pantallas de Consulta de Información

Figura 20.

Ilustración final de pantalla desarrollada para consulta de información en la aplicación web.

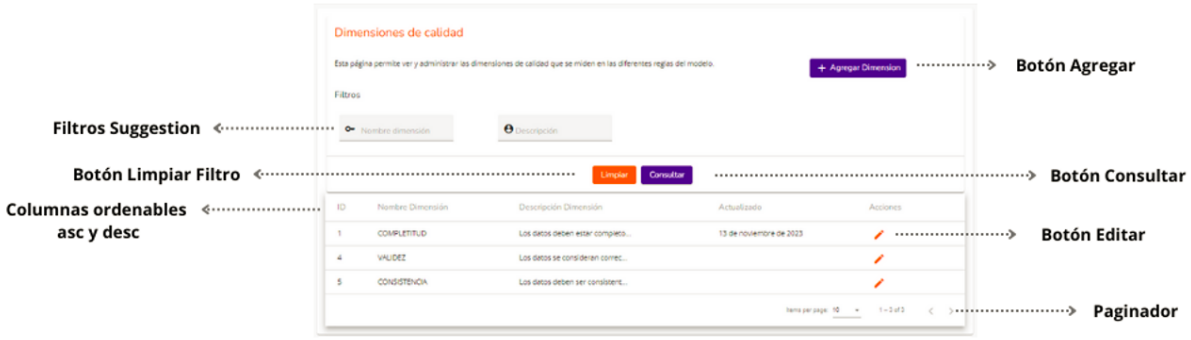


Figura 21.

Ilustración final de la distribución de acciones permitidas en la pantalla de consulta.

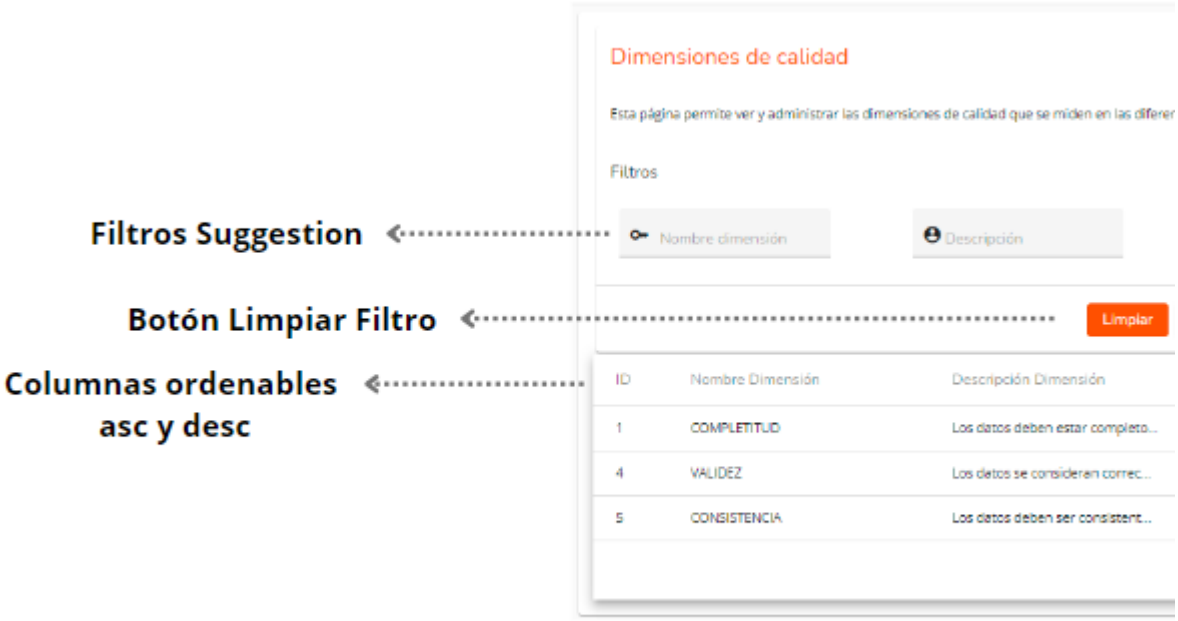
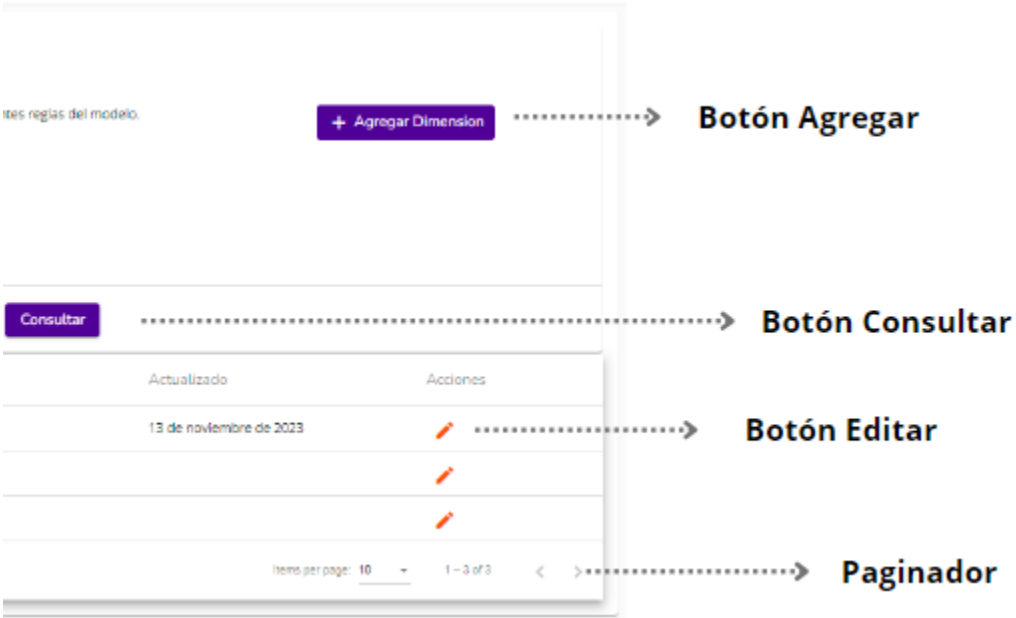


Figura 22.

Ilustración final de la distribución de acciones permitidas en la pantalla de consulta.



Las múltiples pantallas de la aplicación han sido diseñadas con un estándar coherente para ofrecer una experiencia de usuario unificada y eficiente. A continuación, se destacan algunas características clave presentes en varias pantallas:

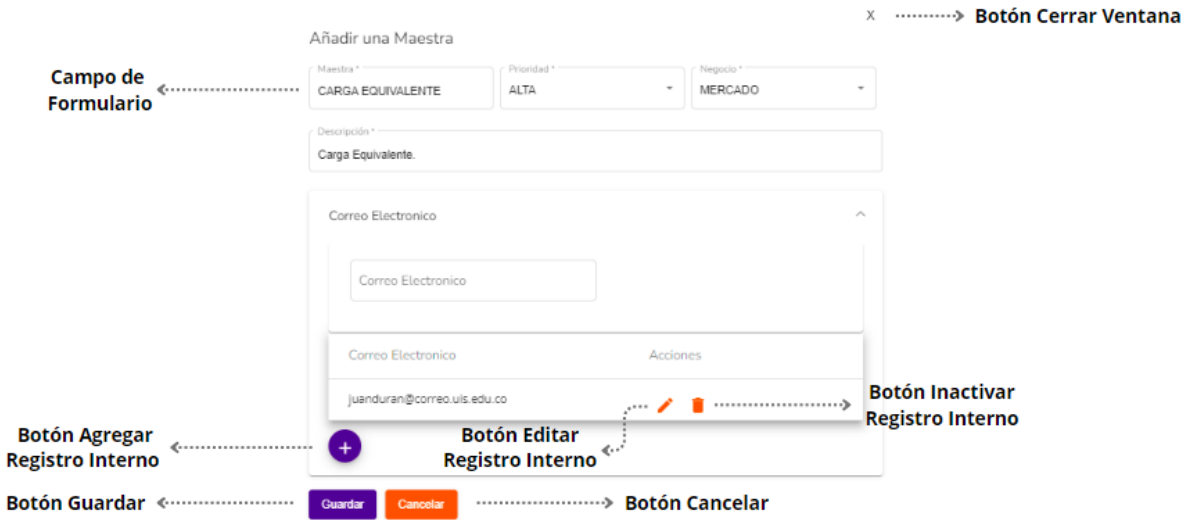
- Botones de Acción:
  - **Crear**: Permite agregar nuevos elementos.
  - **Editar**: Facilita la modificación de elementos existentes.
- Navegación y Visualización:
  - **Paginador**: Facilita la navegación entre páginas de tablas extensas.
- Interactividad de Tablas:
  - **Ordenamiento**: Las columnas de las tablas son ordenables de manera ascendente y descendente.
- Filtros y Búsquedas:
  - **Filtros Sugeridos**: Proporcionan sugerencias mientras el usuario escribe.
  - **Botón Limpiar Filtro**: Permite borrar rápidamente cualquier filtro aplicado.
  - **Botón Buscar por Filtro**: Facilita la búsqueda específica de información.

Para el tema de la creación se tomará como ejemplo el desarrollo para la pantalla de Maestras, este estándar también fue aplicado para el resto de las pantallas, un ejemplo a continuación:

6.2.2 Pantallas de Inserción

Figura 23.

Ilustración final de la distribución de acciones permitidas en la pantalla de creación de registros.



Todas las pantallas de inserción siguen un estándar coherente para proporcionar una experiencia de usuario uniforme. Estas pantallas incluyen los siguientes elementos:

- Botones de Acción:
  - **Guardar**: Permite almacenar la información introducida.
  - **Cancelar**: Permite descartar los cambios realizados.
  - **Cerrar Ventana**: Facilita el cierre de la pantalla de inserción.
  - **Campos de Formulario**: Se incluyen campos para recopilar información relevante.

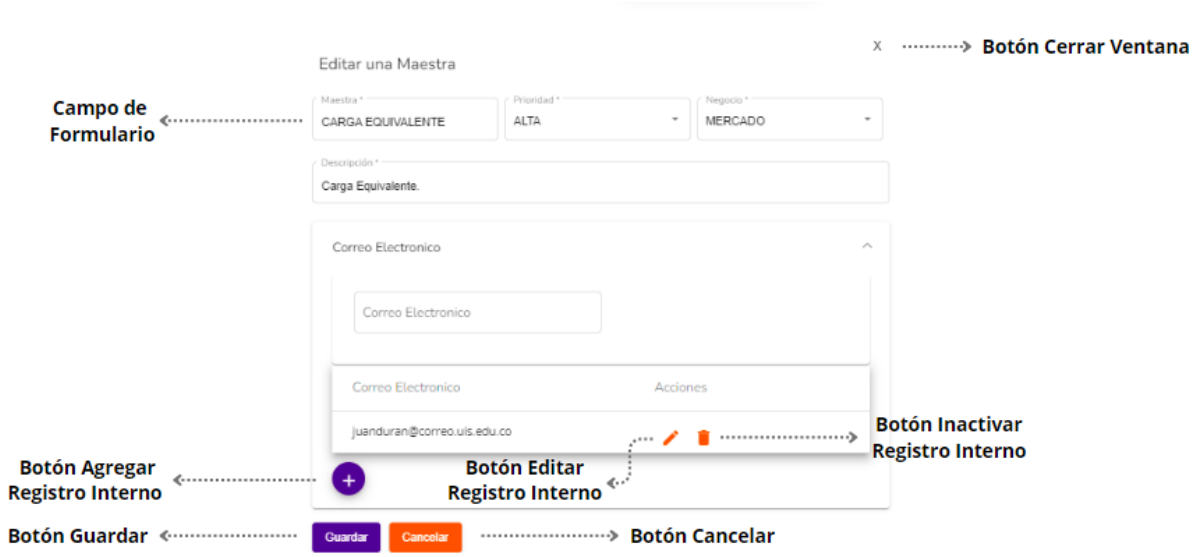
Sin embargo, algunas pantallas de inserción presentan variaciones notables en el manejo de tablas internas. Estas tablas permiten:

- Tablas Internas:
  - **Agregar Registros**: Se incluye la capacidad de insertar nuevos registros directamente en la tabla interna.
  - **Editar Registros**: Facilita la modificación de información existente dentro de la tabla.
  - **Inactivar Registros**: Permite desactivar registros dentro de la tabla, manteniendo una relación con el registro principal.

6.2.3 Pantallas de Edición

Figura 24.

Ilustración final de la distribución de acciones permitidas en la pantalla de edición de registros.



Todas las pantallas de edición siguen un estándar consistente para garantizar una experiencia de usuario uniforme. Estas pantallas incluyen los siguientes elementos:

- Botones de Acción:
  - **Guardar**: Permite guardar los cambios realizados.
  - **Cancelar**: Permite descartar modificaciones y salir sin guardar.
  - **Cerrar Ventana**: Facilita el cierre de la pantalla de edición.

Campos de Formulario: Incluyen campos para modificar la información existente.

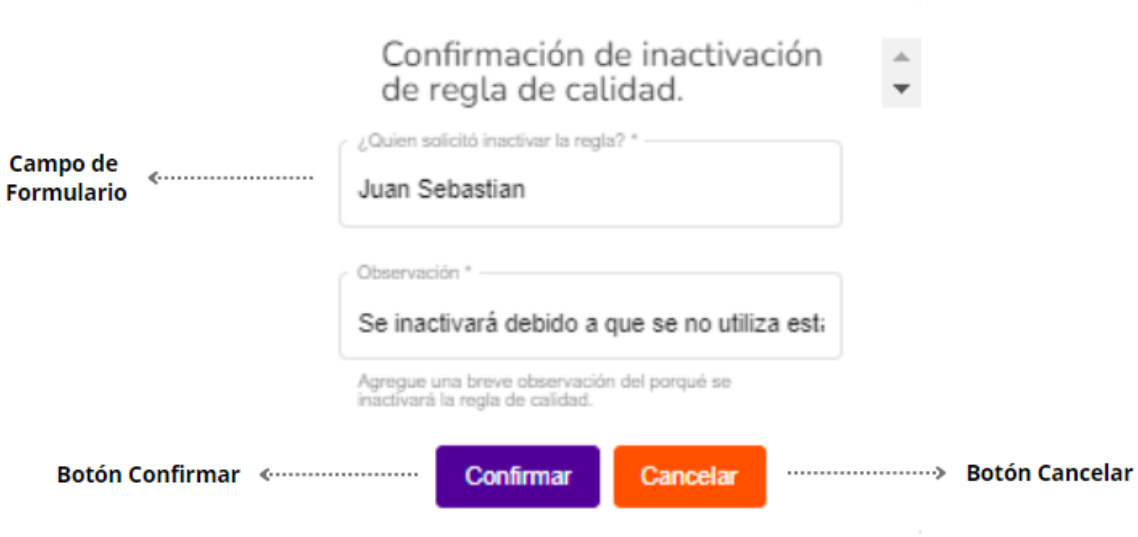
En algunas pantallas de edición, se presenta un manejo diferente con tablas internas, permitiendo:

- Tablas Internas:
  - **Agregar Registros**: Posibilidad de insertar nuevos registros directamente en la tabla interna.
  - **Editar Registros**: Facilita la modificación de información existente dentro de la tabla.
  - **Inactivar Registros**: Permite desactivar registros dentro de la tabla, manteniendo una relación con el registro principal.
  - **Edición Restringida**: Adicionalmente, algunas pantallas de edición presentan un enfoque más restringido, donde solo se pueden modificar ciertos campos. Los campos no editables aparecen deshabilitados para garantizar una gestión controlada de la información.

6.2.4 Pantallas de Inactivación

Figura 25.

Ilustración final de la distribución de acciones permitidas en la ventana de confirmación de inactivación de registro.



Todas las pantallas de inactivación siguen un estándar coherente para ofrecer una experiencia de usuario uniforme. Estas pantallas incluyen los siguientes elementos:

- Botones de Acción:
  - **Confirmar**: Permite realizar la acción de inactivar, confirmando la decisión.
  - **Cancelar**: Ofrece la opción de anular la operación y salir sin inactivar.

  Campos de Formulario: Incluyen campos relevantes para la acción de inactivación.

6.2.5 Pantallas de Historial de Cambios

Figura 26.

Ilustración final de la distribución de acciones permitidas en el apartado de visualización de historial de cambios de un registro.

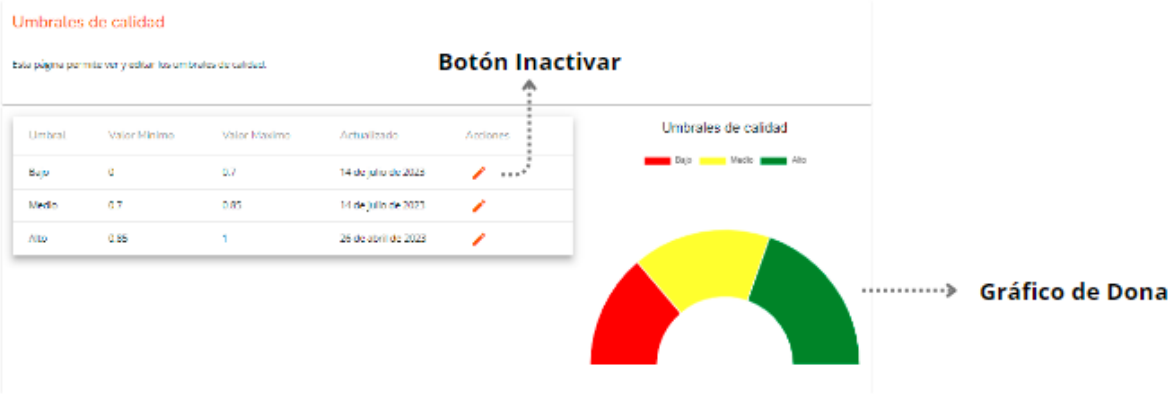
Estas pantallas están diseñadas para proporcionar a los usuarios un acceso fácil y claro al historial de cambios en las reglas de calidad. Aquí se detallan sus elementos principales:

- Botones de Acción:
  - **Cerrar**: Permite salir de la pantalla de historial de cambios.
  - **Cerrar Ventana**: Facilita el cierre completo de la ventana de visualización.

6.2.6 Pantallas de Configuración de Umbrales de Calidad

Figura 27.

Ilustración final de la distribución de acciones permitidas en la pantalla de gestión de umbrales de calidad.



Esta pantalla está diseñada para ofrecer a los usuarios la capacidad de configurar y visualizar los umbrales de calidad de manera efectiva. Sus elementos clave son:

- Botón de Edición de Umbral:
  - **Editar Umbral**: Permite a los usuarios ajustar la configuración de un umbral específico según sus necesidades.
  - **Gráfico de Dona**: Proporciona una representación visual del porcentaje de cada umbral, permitiendo una comprensión rápida y clara de la distribución.

Estos elementos combinados ofrecen a los usuarios un control preciso sobre la configuración de umbrales, junto con una representación visual que facilita la interpretación de los datos.

6.2.7 Pantallas de Ejecución de Microservicios a Demanda

Figura 28.

Ilustración final de la distribución de acciones permitidas en la pantalla de ejecución de microservicios.

Esta pantalla brinda a los usuarios la flexibilidad de ejecutar microservicios del modelo de calidad según sus necesidades. Sus características destacadas son:

- Campos de Lista de Valores:
  - **Ingresar Parámetros**: Permite a los usuarios especificar los valores necesarios para la ejecución del microservicio.

    Botón de Ejecución:

  - **Ejecutar**: Inicia el proceso de ejecución del microservicio con los parámetros proporcionados.

Estos elementos proporcionan a los usuarios una interfaz clara y eficaz para personalizar y ejecutar los microservicios de acuerdo con sus requisitos específicos.

6.2.8 Pantalla de Auditoria

Figura 29.

Ilustración final de pantalla desarrollada para el apartado de auditoría.

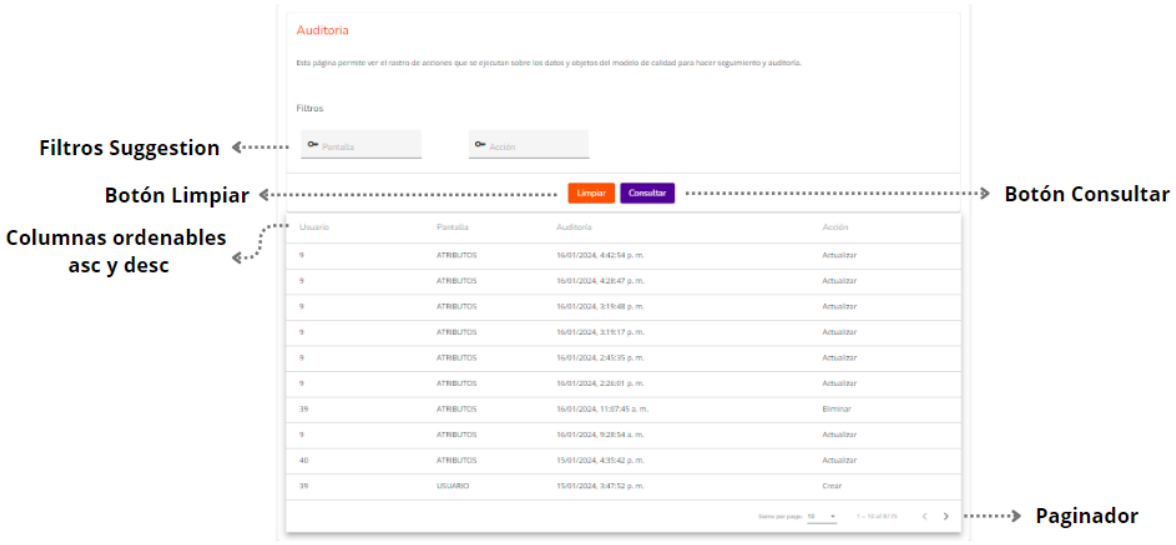


Figura 30.

Ilustración final de distribución de acciones permitidas en la pantalla de auditoría.

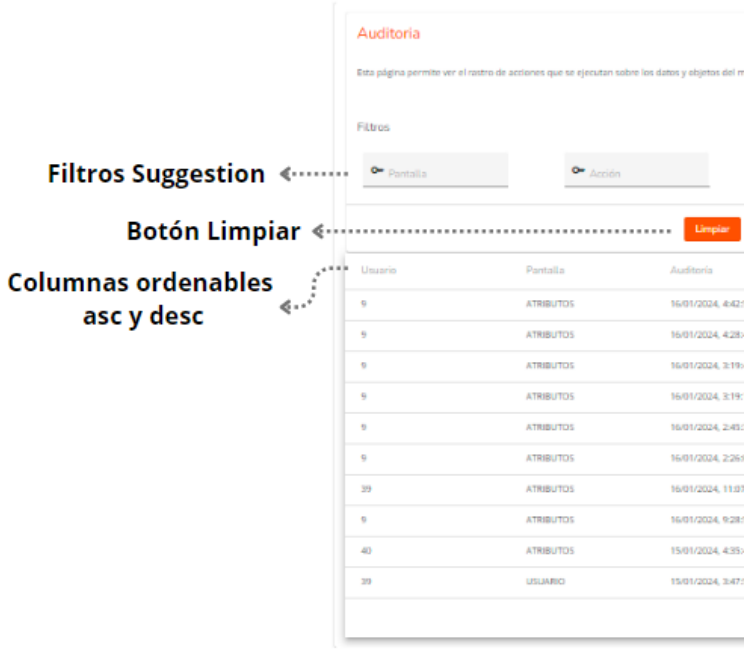
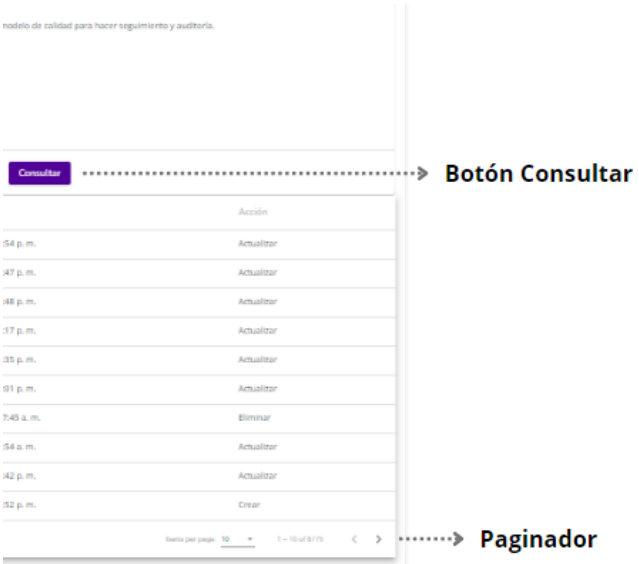


Figura 31.

Ilustración final de la distribución de acciones permitidas en la pantalla de auditoría.



La pantalla de auditoría proporciona una visión detallada de los registros históricos, permitiendo a los usuarios explorar y analizar la información de manera efectiva. Las características clave de esta pantalla incluyen:

- **Filtros de Tipo Sugerencia**: Facilitan la búsqueda proporcionando sugerencias mientras el usuario escribe, agilizando el proceso de filtrado.

Botones de Acción:

- **Limpiar Filtros:** Permite al usuario eliminar rápidamente cualquier filtro aplicado, simplificando la manipulación de datos.
- **Consultar por Filtros**: Inicia la búsqueda de registros basada en los criterios de filtro seleccionados.
- **Paginador para la Tabla**: Ofrece una navegación fácil entre las páginas de la tabla, mejorando la experiencia de usuario en tablas extensas.
- **Ordenamiento de Columnas**: Las columnas de la tabla se pueden ordenar tanto de forma ascendente como descendente, brindando flexibilidad en la exploración de datos.

6.3 Desarrollo Backend

Con el frontend establecido y enfocado en ofrecer una experiencia de usuario excepcional, se dirigió la atención hacia el backend para impulsar la funcionalidad y lógica subyacente de la aplicación. En esta sección, se detallarán las decisiones clave, metodologías y herramientas que respaldan la arquitectura y funcionalidad del lado del servidor.

Mientras que el frontend actúa como la interfaz visible para los usuarios, el backend es la columna vertebral que gestiona la lógica de negocio, la persistencia de datos y la comunicación

con servicios externos. Se examinará cómo cada componente se integra de manera armoniosa para ofrecer una aplicación cohesiva y eficiente.

6.3.1 Desarrollo de API Generica

Tras un análisis exhaustivo y la observación de que múltiples pantallas seguían un estándar similar en cuanto a la estructura de datos, se tomó la decisión estratégica de desarrollar una API genérica. Esta API, diseñada para ofrecer un CRUD genérico, tiene la capacidad de gestionar todas las transacciones de datos dentro de la aplicación.

La creación de esta API genérica surge como respuesta a la necesidad de optimizar y simplificar las operaciones recurrentes en la gestión de datos. Su enfoque modular y su capacidad para adaptarse a diversas estructuras de datos permiten un manejo eficiente y consistente de la información a lo largo de la aplicación.

Esta iniciativa no solo mejora la coherencia en el manejo de datos, sino que también proporciona una base sólida para el desarrollo futuro, facilitando la incorporación de nuevas funcionalidades de manera eficaz y escalable.

6.3.1.1 Controlador getScreenData. Este código corresponde a un controlador en un entorno de desarrollo backend, específicamente para manejar solicitudes GET relacionadas con la obtención de datos específicos de una pantalla en una aplicación. Aquí está el desglose de su funcionamiento**:**

- Parámetros de Consulta:
  - El controlador recibe los parámetros de consulta de la solicitud GET, como el desplazamiento (offset), límite (limit), estado de inactividad (inactive), estado de solo activos (onlyActive), y nombre de la pantalla (screen).

Registro de Actividades:

- Utiliza la función writeLog para registrar las acciones realizadas en el controlador en el registro de actividad. Estos registros incluyen mensajes indicando la recepción de la solicitud GET y la ejecución del método getScreenData.

- Construcción de la Consulta:
  - Se crea un objeto query que contiene las opciones de desplazamiento y límite, convertidos a números. Este objeto se utilizará posteriormente para configurar la consulta a la base de datos.

  Filtrado por Estado Activo:

  - Si la consulta incluye la opción onlyActive establecida en 'true', se agrega un filtro a la consulta para seleccionar solo los elementos activos, en este caso, aquellos con idEstado igual a 1.
- Filtrado por Estado Inactivo:
  - Si la consulta incluye la opción inactive establecida en 'true', se agrega un filtro a la consulta para seleccionar solo los elementos inactivos, en este caso, aquellos con idEstado igual a 2.
- Selección de la Pantalla:
  - Utiliza la función switchScreen para obtener el objeto correspondiente a la pantalla solicitada. Este objeto contiene métodos para interactuar con la base de datos en función de la pantalla específica.

  Búsqueda de Información:

- Si se obtiene con éxito el objeto de la pantalla (screenTaken), se realiza una búsqueda en la base de datos utilizando el método findAndCountAll con la consulta previamente construida.

➢ Respuesta al Cliente:

- Si la búsqueda es exitosa, se responde al cliente con un código de estado 200 y un objeto JSON que incluye el estado, mensaje, total de elementos y los datos recuperados de la base de datos.
- En caso de error durante la búsqueda, se responde con un código de estado 500 y un mensaje de error interno.

Este controlador está diseñado para ser reutilizable y configurable para diferentes pantallas de la aplicación, permitiendo la obtención de datos de manera eficiente y segura a través de solicitudes GET. Cabe destacar que los fragmentos específicos del código no se proporcionan debido a consideraciones de confidencialidad.

6.3.1.2 Controlador getScreenItemById. Este código corresponde a un controlador en un entorno de desarrollo backend, diseñado para manejar solicitudes que buscan obtener un elemento específico de una pantalla en la aplicación mediante su identificador único (ID). A continuación, se detalla su funcionamiento**:**

➢ Parámetros de Solicitud:

El controlador recibe dos parámetros de la solicitud:

- id: El identificador único del elemento que se desea obtener (por defecto, se establece en 1 si no se proporciona).
- screen: El nombre de la pantalla de la cual se busca el elemento.

- Selección de la Pantalla:
  - Utiliza la función switchScreen para obtener el objeto correspondiente a la pantalla solicitada (screenTaken). Este objeto contiene métodos para interactuar con la base de datos en función de la pantalla específica.
- Búsqueda del Elemento por ID:
  - Si se obtiene con éxito el objeto de la pantalla (screenTaken), se realiza una búsqueda en la base de datos utilizando el método findByPk con el identificador proporcionado (id).
- Manejo de Resultados de Búsqueda:
  - Si el elemento no se encuentra en la base de datos, se responde al cliente con un código de estado 404, indicando que el ID no fue encontrado.
  - Si se encuentra el elemento, se responde al cliente con un código de estado 200 y un objeto JSON que incluye el estado, mensaje y los datos del elemento encontrado.
- Registro de Actividades y Errores:
  - Se utilizan las funciones writeLog para registrar actividades y errores relevantes. Estos registros incluyen mensajes sobre la realización de la búsqueda del ID en la base de datos y la respuesta enviada al cliente.

  Manejo de Excepciones:

  - Cualquier error durante el proceso, ya sea en la búsqueda o en la ejecución de otras operaciones, se maneja de manera adecuada. Se responde al cliente con un código de estado 500 y un mensaje de error interno.

6.3.1.3 Controlador postScreenItem. Este código corresponde a un controlador en un entorno de desarrollo backend, específicamente diseñado para manejar solicitudes POST que implican la creación de un nuevo elemento en una pantalla de la aplicación. A continuación, se detalla su funcionamiento**:**

- Datos de la Solicitud:
  - El controlador extrae los datos relevantes de la solicitud POST, incluyendo la anotación (annotation), el nombre de la pantalla (screen), y otros datos específicos del modelo de pantalla (screenModel).
- Registro de Actividades:
  - Se utilizan las funciones writeLog para registrar actividades relevantes, como la recepción de la solicitud POST y la ejecución del método postScreenItem.

  Selección de la Pantalla:

  - Utiliza la función switchScreen para obtener el objeto correspondiente a la pantalla solicitada (screenTaken). Este objeto contiene métodos para interactuar con la base de datos en función de la pantalla específica.
- Creación del Nuevo Elemento:
  - Se crea un nuevo objeto (newItem) basado en el modelo de pantalla (screenTaken) y los datos proporcionados en la solicitud.
- Almacenamiento en la Base de Datos:
  - El nuevo elemento se guarda en la base de datos utilizando el método save().

- Respuesta al Cliente:
  - Si la creación del elemento es exitosa, se responde al cliente con un código de estado 201 indicando que el recurso ha sido creado, junto con un objeto JSON que incluye el estado, mensaje y los datos del nuevo elemento.
- Registro de Auditoría:
  - Se hace mención de la posibilidad de registrar auditoría (comentado en el código), aunque el fragmento específico de esa lógica no se proporciona por motivos de confidencialidad.

  Manejo de Errores:

  - Cualquier error durante el proceso, ya sea en la creación del elemento o en la ejecución de otras operaciones, se maneja de manera adecuada. Se responde al cliente con un código de estado 500 y un mensaje de error interno.

6.3.1.4 Controlador putScreenItem. Este código corresponde a un controlador en un entorno de desarrollo backend, específicamente diseñado para manejar solicitudes PUT que implican la actualización de un elemento existente en una pantalla de la aplicación. A continuación, se detalla su funcionamiento**:**

- Datos de la Solicitud:
  - El controlador extrae los datos relevantes de la solicitud PUT, incluyendo el nombre de la pantalla (screen), el identificador único del elemento que se desea actualizar (id), y otros datos provenientes del cuerpo de la solicitud (body).

- Registro de Actividades:
  - Se utilizan las funciones writeLog para registrar actividades relevantes, como la recepción de la solicitud PUT y la ejecución del método putScreenItem.
- Selección de la Pantalla:
  - Utiliza la función switchScreen para obtener el objeto correspondiente a la pantalla solicitada (screenTaken). Este objeto contiene métodos para interactuar con la base de datos en función de la pantalla específica.
- Búsqueda del Elemento por ID:
  - Realiza una búsqueda en la base de datos utilizando el método findByPk con el identificador proporcionado (id).
- Manejo de Resultados de Búsqueda:
  - Si el elemento no se encuentra en la base de datos, se responde al cliente con un código de estado 404, indicando que el ID no fue encontrado.
  - Si se encuentra el elemento, se procede con la actualización de sus datos.
- Generación de la Actualización:
  - Se actualiza el elemento utilizando el método update con los datos proporcionados en el cuerpo de la solicitud (body).
- Respuesta al Cliente:
  - Si la actualización del elemento es exitosa, se responde al cliente con un código de estado 201 indicando que el recurso ha sido actualizado, junto con un objeto JSON que incluye el estado, mensaje y los datos actualizados del elemento.

- Manejo de Errores:
  - Cualquier error durante el proceso, ya sea en la actualización del elemento o en la ejecución de otras operaciones, se maneja de manera adecuada. Se responde al cliente con un código de estado 500 y un mensaje de error interno.

6.3.1.5 Controlador deleteScreenItem. Este código representa un controlador en el entorno backend y está diseñado para manejar solicitudes DELETE relacionadas con la eliminación lógica de un elemento específico en una pantalla de la aplicación. A continuación, se presenta una descripción del funcionamiento del código**:**

- Datos de la Solicitud:
  - El controlador extrae los datos relevantes de la solicitud DELETE, incluyendo el nombre de la pantalla (screen), el identificador único del elemento que se desea eliminar (id), y otros datos como el comentario de la acción (annotation).
- Registro de Actividades:
  - Se utilizan las funciones writeLog para registrar actividades relevantes, como la recepción de la solicitud DELETE y la ejecución del método deleteScreenItem.
- Selección de la Pantalla:
  - Utiliza la función switchScreen para obtener el objeto correspondiente a la pantalla solicitada (screenTaken). Este objeto contiene métodos para interactuar con la base de datos en función de la pantalla específica.
- Búsqueda del Elemento por ID:
  - Realiza una búsqueda en la base de datos utilizando el método findByPk con el identificador proporcionado (id).

- Manejo de Resultados de Búsqueda:
  - Si el elemento no se encuentra en la base de datos, se responde al cliente con un código de estado 404, indicando que el ID no fue encontrado.
  - Si se encuentra el elemento, se verifica si ya se encuentra en un estado inactivo. Si es así, se responde al cliente con un código de estado 400, indicando que el elemento ya tiene un estado inactivo.
- Generación de la Eliminación Lógica:
  - Se actualiza el estado del elemento a inactivo (idEstado: itemState.inactive) utilizando el método update.
- Registro de Auditoría:
  - Se registra la acción de eliminar lógicamente en la tabla de auditoría, incluyendo detalles como el usuario que realizó la acción, el ID del elemento, la pantalla y el comentario asociado.
- Respuesta al Cliente:
  - Si la eliminación lógica del elemento es exitosa, se responde al cliente con un código de estado 200 indicando que el recurso ha sido eliminado de manera lógica, junto con un objeto JSON que incluye el estado, mensaje y datos del elemento eliminado y del usuario autenticado.
- Manejo de Errores:
  - Cualquier error durante el proceso, ya sea en la eliminación lógica del elemento o en la ejecución de otras operaciones, se maneja de manera adecuada. Se responde al cliente con un código de estado 500 y un mensaje de error interno.

6.3.1.6 Utilización de la API SendGrid. La implementación de la funcionalidad de recuperación de usuario y contraseña involucró la integración de la API SendGrid en el backend de la aplicación. SendGrid se utilizó como un servicio eficiente para el envío de correos electrónicos, garantizando un proceso seguro y confiable para los usuarios que habían olvidado sus credenciales**.**

Esta integración permitió la generación y envío automático de correos electrónicos a los usuarios afectados, proporcionándoles los pasos necesarios para recuperar su información de inicio de sesión. La utilización de SendGrid se llevó a cabo bajo estrictas medidas de seguridad para salvaguardar la confidencialidad de los datos de los usuarios en todo momento.

Este enfoque aseguró que el proceso de recuperación de cuenta fuera eficiente y sin complicaciones para los usuarios, al mismo tiempo que se mantenía la seguridad y privacidad de su información personal. La utilización de SendGrid para el envío de correos electrónicos es un testimonio de la atención a los detalles y el respeto hacia la confidencialidad de los usuarios en cada aspecto del desarrollo de la aplicación.

6.3.2 Introducción a Swagger y Ejemplos para la Pantalla de "Dimensiones de Calidad"

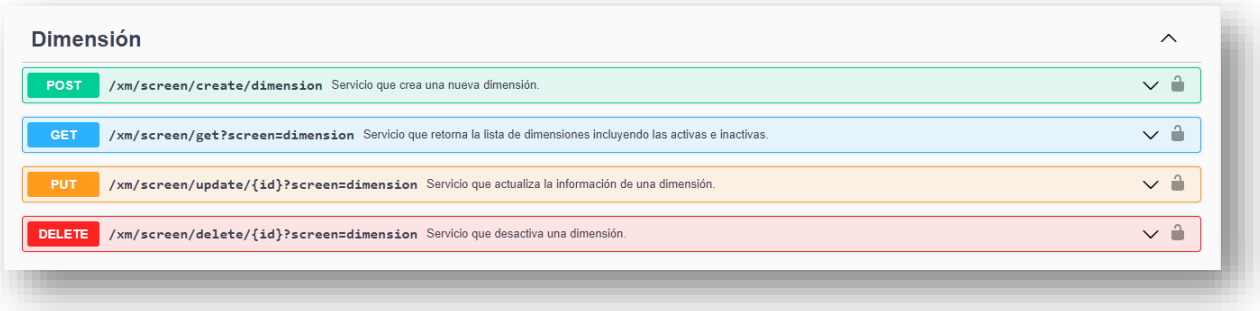
Después de haber explorado detalladamente la lógica y funcionamiento del backend a través de los controladores CRUD, se procederá a ilustrar la interfaz de Swagger, una herramienta poderosa que facilita la documentación y visualización de las APIs. Swagger proporciona una interfaz interactiva que permite a los desarrolladores comprender fácilmente cómo interactuar con los puntos finales de la API.

Figura 32.

Interfaz de swagger para la aplicación de administración de calidad de datos.

Figura 33.

Listado de endpoints para la pantalla de Dimensiones de Calidad.



A continuación, se presentarán ejemplos visuales utilizando Swagger para la pantalla de "Dimensiones de Calidad". Estos ejemplos se centrarán en las acciones CRUD básicas: Obtener datos, Crear nuevo ítem, Actualizar ítem existente y Eliminar ítem.

- **GET - Obtener Datos**: Se mostrará la interfaz de Swagger para la acción GET, ilustrando cómo realizar una solicitud para obtener datos de las "Dimensiones de Calidad".

Figura 34.

Ilustración de la parametrización del endpoint "Obtener Datos" para la pantalla Dimensiones de calidad.

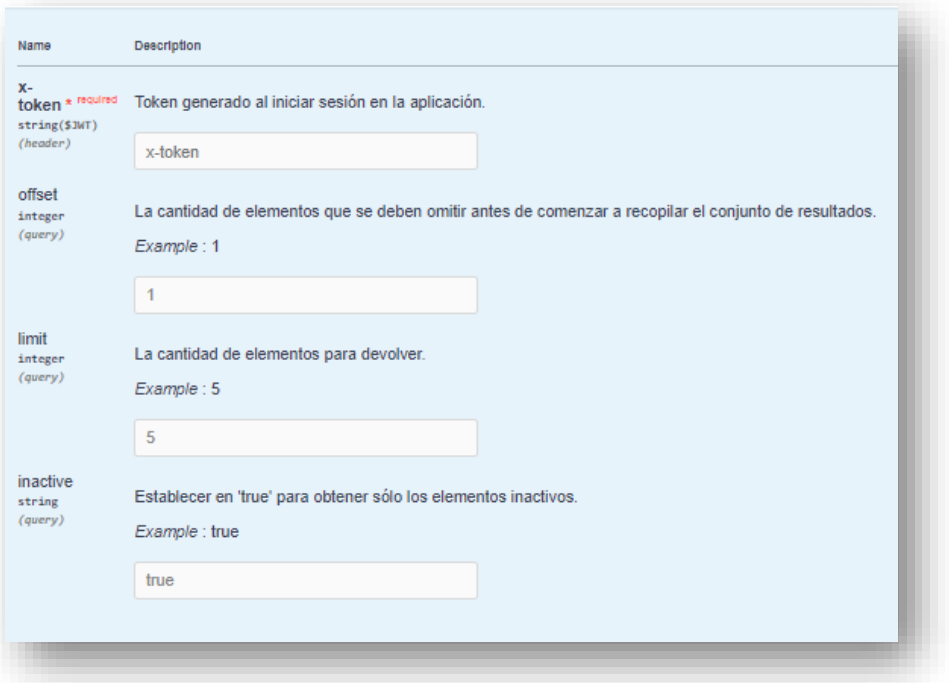
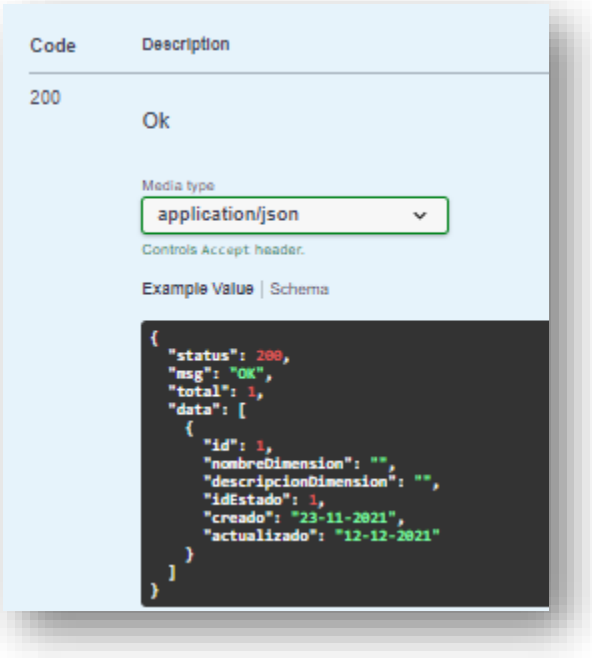


Figura 35.

Ilustración de la respuesta exitosa retornada por el endpoint "Obtener Datos" de la pantalla Dimensiones de Calidad.



- **POST - Crear Nuevo Ítem**: Se presentará la interfaz de Swagger para la acción POST, destacando cómo enviar una solicitud para crear un nuevo ítem en las "Dimensiones de Calidad".

Figura 36.

Ilustración de la respuesta exitosa del endpoint "Crear nuevo item" para la pantalla Dimensiones de Calidad.

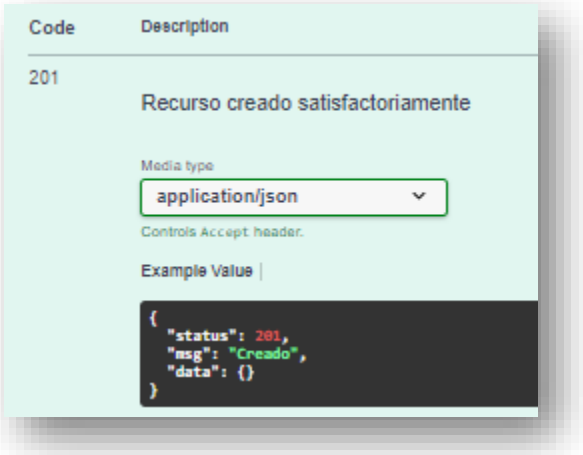


Figura 37.

Ilustración de la respuesta de una mala petición realizada al endpoint "Crear nuevo item" para la pantalla Dimensiones de Calidad.

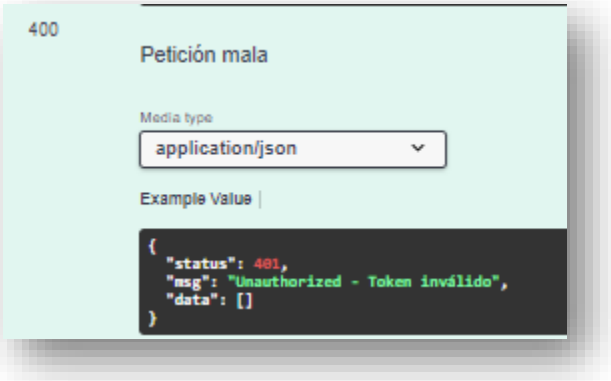
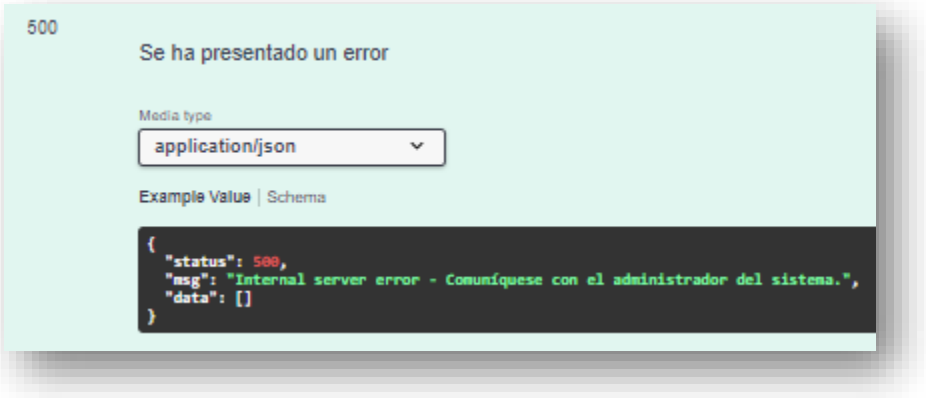


Figura 38.

Ilustración de la respuesta cuando se presenta un error en el endpoint "Crear nuevo item" para la pantalla Dimensiones de Calidad.



- **PUT - Actualizar Ítem Existente**: La interfaz de Swagger para la acción PUT será mostrada, demostrando cómo realizar una solicitud para actualizar información de un ítem existente en las "Dimensiones de Calidad".

Figura 39.

Ilustración de la parametrización del endpoint "Actualizar Item Existente" para la pantalla Dimensiones de Calidad

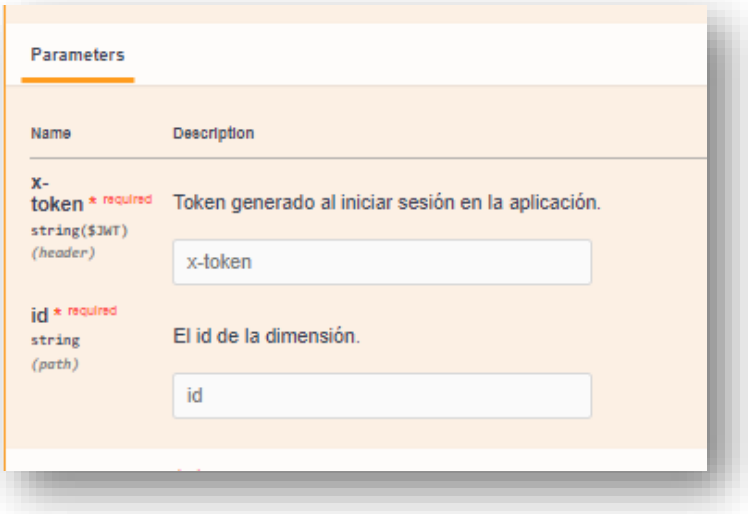
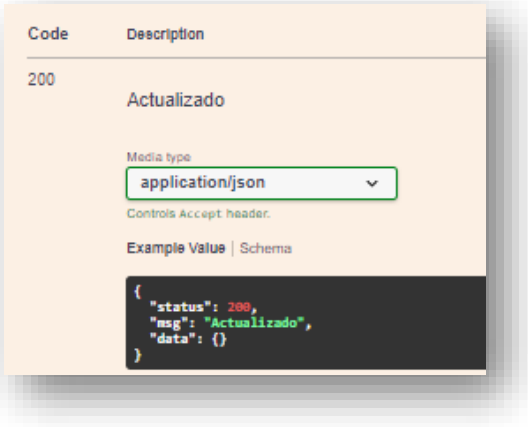


Figura 40.

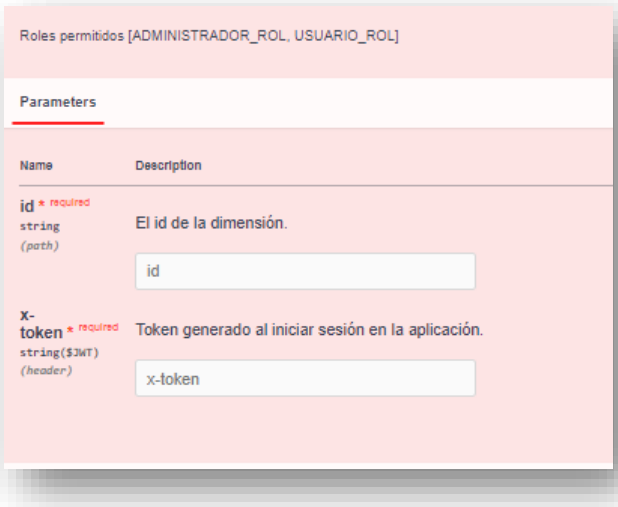
Ilustración de una respuesta exitosa del endpoint "Actualizar Existente" para la pantalla de Dimensiones de Calidad.



- **DELETE - Eliminar Ítem**: Se exhibirá la interfaz de Swagger para la acción DELETE, indicando cómo enviar una solicitud para eliminar un ítem específico de las "Dimensiones de Calidad".

Figura 41.

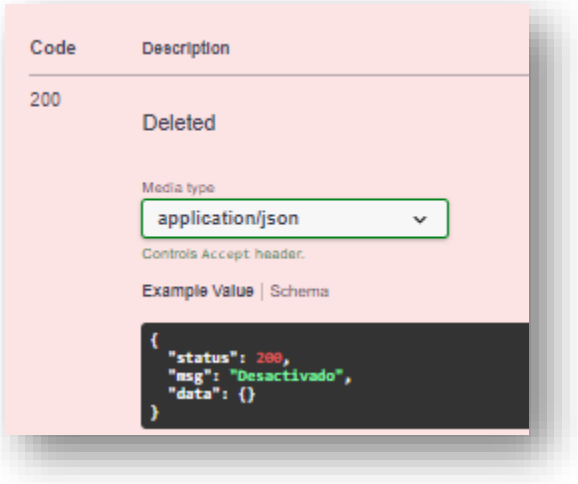
Ilustración de la parametrización del endpoint "Eliminar Item" para la pantalla Dimensiones de Calidad



.

Figura 42.

Ilustración de una respuesta exitosa del endpoint "Eliminar Item" para la pantalla Dimensiones de Calidad.



6.3.3 Introducción a la Base de Datos

La base de datos elegida para este proyecto fue Microsoft SQL Server, respaldada por la eficiente herramienta Microsoft SQL Server Management Studio 18. La implementación se estructuró en tres instancias distintas: una para desarrollo, otra para calidad (QA), y la última para producción. Este enfoque garantizó una gestión adecuada de los entornos y una separación clara entre las fases de desarrollo y las etapas de prueba y producción.

La seguridad de la base de datos fue una prioridad, por lo que el acceso a cada instancia estaba restringido. Únicamente el administrador podía permitir el acceso desde una IP específica, fortaleciendo así las medidas de seguridad y protegiendo datos sensibles de clientes. Dicha precaución, además de cumplir con estándares legales, aseguró un entorno controlado y resguardado contra posibles amenazas.

Por motivos de privacidad y confidencialidad, no es posible mostrar detalles específicos de la base de datos, ya que contiene información sensible de los clientes y datos empresariales. El acceso a estas instancias se manejó de manera cuidadosa y restringida, reflejando el compromiso del proyecto con la integridad y seguridad de los datos.

Es relevante destacar que, durante el desarrollo de este proyecto, no fue necesario realizar modificaciones en ninguna tabla preexistente o definida por la empresa XM. Esta decisión contribuyó a la estabilidad del sistema y evitó posibles conflictos con la estructura ya establecida por la empresa en su base de datos.

6.4 Pruebas Unitarias y Métricas de Código con SonarQube

Habiendo cubierto el desarrollo del frontend y backend de la aplicación de gestión de calidad de datos, es esencial abordar la calidad del código y las pruebas unitarias. En este contexto, se ha utilizado SonarQube, una plataforma líder en análisis estático de código que proporciona valiosas métricas para evaluar la calidad del código.

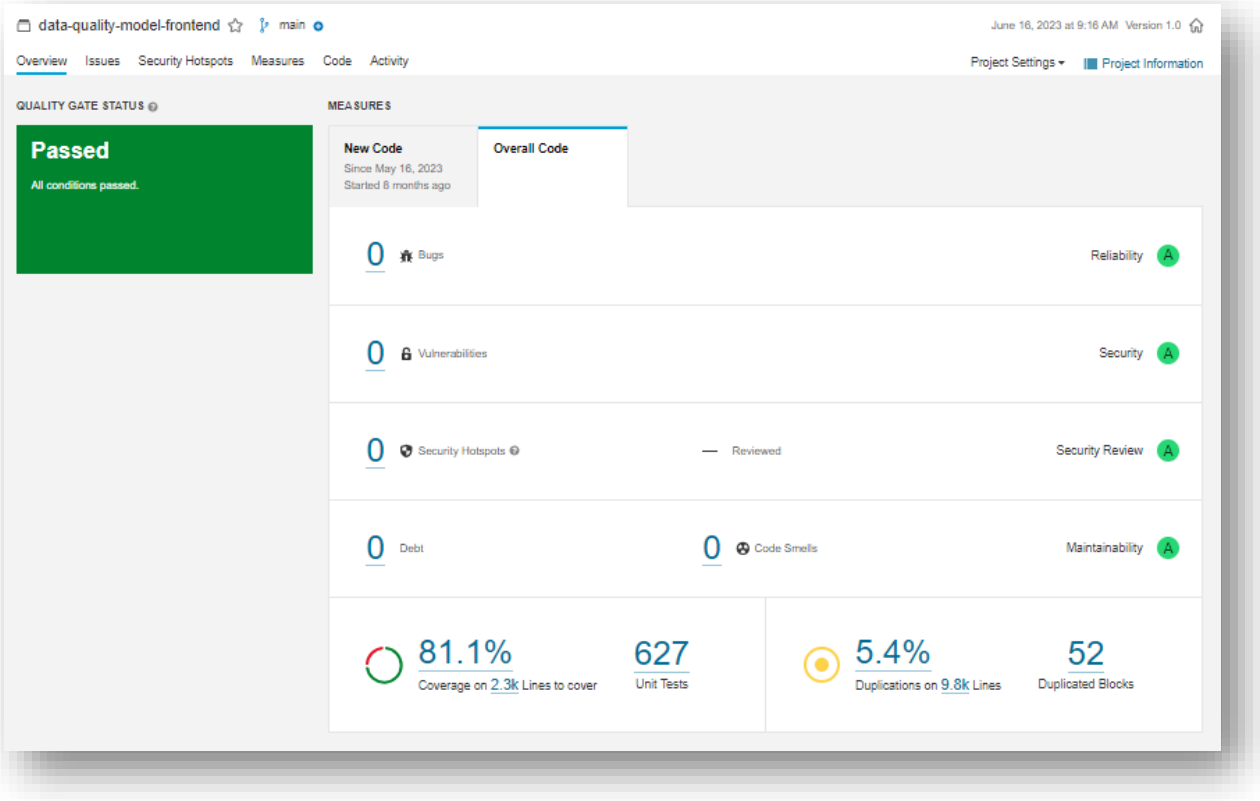
6.4.1 Pruebas Unitarias en el frontend

Para garantizar la robustez y estabilidad del frontend, se implementaron pruebas unitarias exhaustivas. Estas pruebas no solo validan la funcionalidad esperada, sino que también contribuyen a mantener un código libre de errores y fácil de mantener.

A continuación, se presentan algunas capturas de pantalla de SonarQube, que reflejan las métricas clave y la cobertura de código obtenida a través de las pruebas unitarias en el frontend:

Figura 43.

Métricas de gestión del código desarrollado para el frontend de la aplicación web.

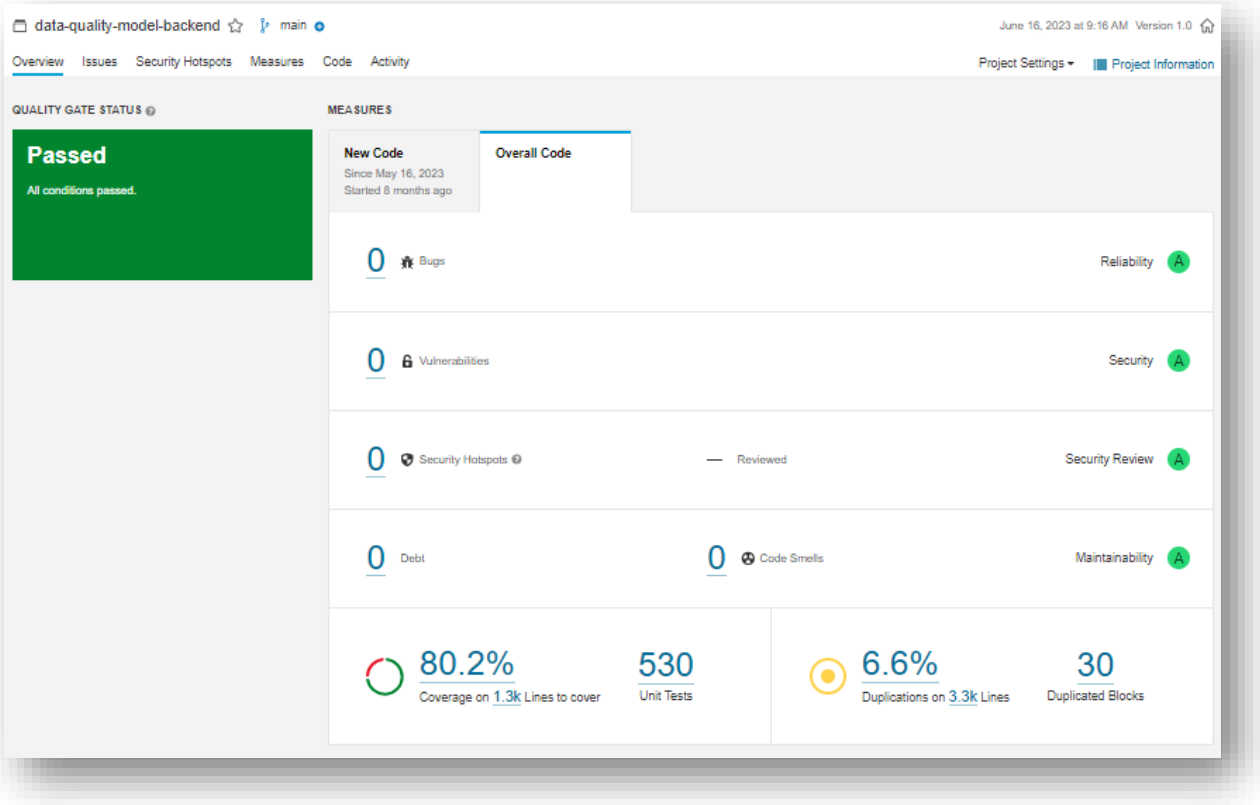


6.4.2 Pruebas Unitarias en el backend

De manera análoga, el backend ha sido sometido a pruebas unitarias exhaustivas para garantizar la confiabilidad y rendimiento de los servicios proporcionados. Las siguientes capturas de pantalla de SonarQube ilustran las métricas clave y la cobertura de código obtenida mediante las pruebas unitarias en el backend:

Figura 44.

Métricas de gestión del código desarrollado para el backend de la aplicación web.



Estas métricas ofrecen una visión detallada de la calidad del código, identificando posibles vulnerabilidades, code smells y proporcionando insights para mejorar la base de código. La implementación de pruebas unitarias y el análisis continuo con SonarQube reflejan el compromiso con los estándares de calidad y la construcción de un software sostenible y confiable.

6.4.3 Gestión de Calidad de Código: Cobertura y Reducción de Duplicados

En el proceso de desarrollo, se establecieron rigurosas políticas de calidad de código para garantizar la fiabilidad y mantenibilidad del sistema. Dos aspectos cruciales abordados fueron la cobertura mínima del código mediante pruebas unitarias y la gestión de duplicados en el código fuente.

6.4.3.1 Cobertura del código. Con el objetivo de asegurar la efectividad de las pruebas unitarias y minimizar la presencia de errores, se estableció una política de cobertura del código del 80%. Esta métrica indica el porcentaje de código que está cubierto por pruebas unitarias, permitiendo una evaluación precisa de la solidez de las pruebas implementadas.

6.4.3.2 Métricas de cobertura del código. Se superó la meta del 80% de cobertura en todas las áreas del código, garantizando una amplia validación de funcionalidades y casos de borde.

6.4.3.3 Gestión de Código Duplicado. El código duplicado puede ser fuente de errores y dificultar las tareas de mantenimiento. Por ende, se estableció un límite máximo del 8% para el código duplicado, promoviendo la creación de un código limpio y fácilmente comprensible.

6.4.3.4 Métricas de Reducción de Duplicados. Se logró mantener el código duplicado por debajo del límite establecido, evidenciando un esfuerzo continuo para eliminar redundancias y mejorar la claridad del código.

La combinación de una sólida cobertura del código mediante pruebas unitarias y la gestión efectiva del código duplicado refuerza la calidad y mantenibilidad del software. Estas métricas no solo cumplen con las políticas establecidas, sino que también contribuyen a un ciclo de desarrollo ágil y a la creación de un sistema robusto y eficiente.

6.5 Seguridad y Protección de Datos: Enfoque Integral

La seguridad de la aplicación fue un pilar fundamental durante todo el ciclo de desarrollo. Se implementaron medidas robustas para garantizar la confidencialidad, integridad y disponibilidad de la información. Algunos de los aspectos clave abordados incluyen:

6.5.1 Encriptación de Claves y Datos

Todas las claves, tanto de acceso a la aplicación como de usuarios, fueron almacenadas utilizando técnicas avanzadas de encriptación. Esto asegura que incluso en caso de acceso no autorizado a la base de datos, la información sensible permanezca protegida.

6.5.2 Gestión Segura de Usuarios

Los datos de los usuarios, incluidos nombres, contraseñas y otros detalles personales, fueron gestionados de manera segura. La autenticación se llevó a cabo mediante procesos robustos que evitan el acceso no autorizado.

6.5.3 Acceso Seguro a la Base de Datos

Se implementaron políticas y configuraciones de seguridad para restringir el acceso a la base de datos solo a usuarios autorizados. Además, se emplearon prácticas de seguridad recomendadas para evitar posibles vulnerabilidades.

6.5.4 Cifrado de Comunicaciones

La comunicación entre el frontend y el backend se realizó a través de canales seguros, utilizando protocolos de cifrado modernos. Esto asegura que la información transmitida entre los componentes de la aplicación esté protegida contra posibles interceptaciones malintencionadas.

6.5.5 Auditoría y Monitoreo

Se implementaron registros de auditoría y sistemas de monitoreo para detectar posibles actividades sospechosas o intentos de acceso no autorizado. Estos registros permiten una rápida respuesta a posibles amenazas.

6.5.6 Actualizaciones y Parches de Seguridad

Se estableció un proceso proactivo para aplicar actualizaciones y parches de seguridad de manera regular, garantizando que la aplicación esté protegida contra posibles vulnerabilidades conocidas.

El enfoque integral hacia la seguridad garantiza que la aplicación cumpla con los estándares más exigentes en la protección de datos y la privacidad del usuario. La implementación de prácticas de seguridad sólidas contribuye a la confianza de los usuarios y asegura la integridad del sistema en un entorno digital en constante evolución.

6.6 Seguridad contra Inyecciones de SQL y Otros Errores Comunes

Una de las principales preocupaciones en el desarrollo de aplicaciones es la protección contra vulnerabilidades conocidas, como las inyecciones de SQL y otros errores de seguridad comunes. En este proyecto, se adoptaron diversas medidas para salvaguardar la aplicación contra tales amenazas:

6.6.1 Prevención de Inyecciones de SQL

Todas las consultas a la base de datos se implementaron utilizando consultas parametrizadas o mediante el uso de ORM (Object-Relational Mapping). Esta práctica ayuda a

prevenir las inyecciones de SQL, donde un atacante intenta manipular las consultas para acceder o modificar datos no autorizados.

6.6.2 Validación de Entradas

Se implementaron mecanismos de validación y sanitización estrictos en el frontend y backend para asegurar que las entradas de los usuarios se ajusten a los formatos y tipos de datos esperados. Esto reduce significativamente la posibilidad de ataques de inyección o manipulación de datos.

6.6.3 Control de Acceso y Permisos

Se estableció un sistema robusto de control de acceso y permisos para garantizar que los usuarios solo puedan acceder y manipular los recursos para los cuales tienen autorización. Esto previene posibles brechas de seguridad al limitar el alcance de las acciones permitidas.

Estas medidas combinadas aseguran que la aplicación esté protegida contra vulnerabilidades comunes y ataques conocidos, proporcionando un entorno digital resistente y seguro para los usuarios finales. La prioridad constante en la seguridad contribuye a la confianza y estabilidad del sistema en el cambiante panorama de amenazas en línea.

6.7 Repositorios y Control de Versiones

El control de versiones durante este proyecto se llevó a cabo mediante GitLab, respaldado por el eficiente terminal de Git Bash. Se establecieron dos repositorios distintos: uno destinado a los desarrollos del frontend y otro para los desarrollos del backend. Esta segregación facilitó la gestión y organización de los archivos, brindando un enfoque más ordenado.

En cada repositorio, se implementaron tres ramas principales: '**develop**', '**qa**', y '**master**'. Estas ramas permitieron una administración estructurada del ciclo de vida del software, desde el desarrollo inicial hasta las fases de prueba y producción.

Por razones de privacidad y confidencialidad, no es posible mostrar detalles específicos de los repositorios, ya que contienen información sensible sobre los desarrollos que podrían ofrecer ventajas competitivas si fueran conocidos externamente. Además, para acceder y realizar acciones en cada repositorio, era obligatorio tener la VPN encendida, reforzando así las medidas de seguridad y protegiendo la integridad del código.

Se establecieron estándares en el manejo de commits para mantener la consistencia y facilitar la comprensión del historial de cambios. Por ejemplo, se empleaban claves como '**feature**' para nuevas funcionalidades, '**fix'** para correcciones de errores, y '**merge'** para fusiones de ramas. Esta convención aseguró un registro claro y organizado de los cambios realizados en el código.

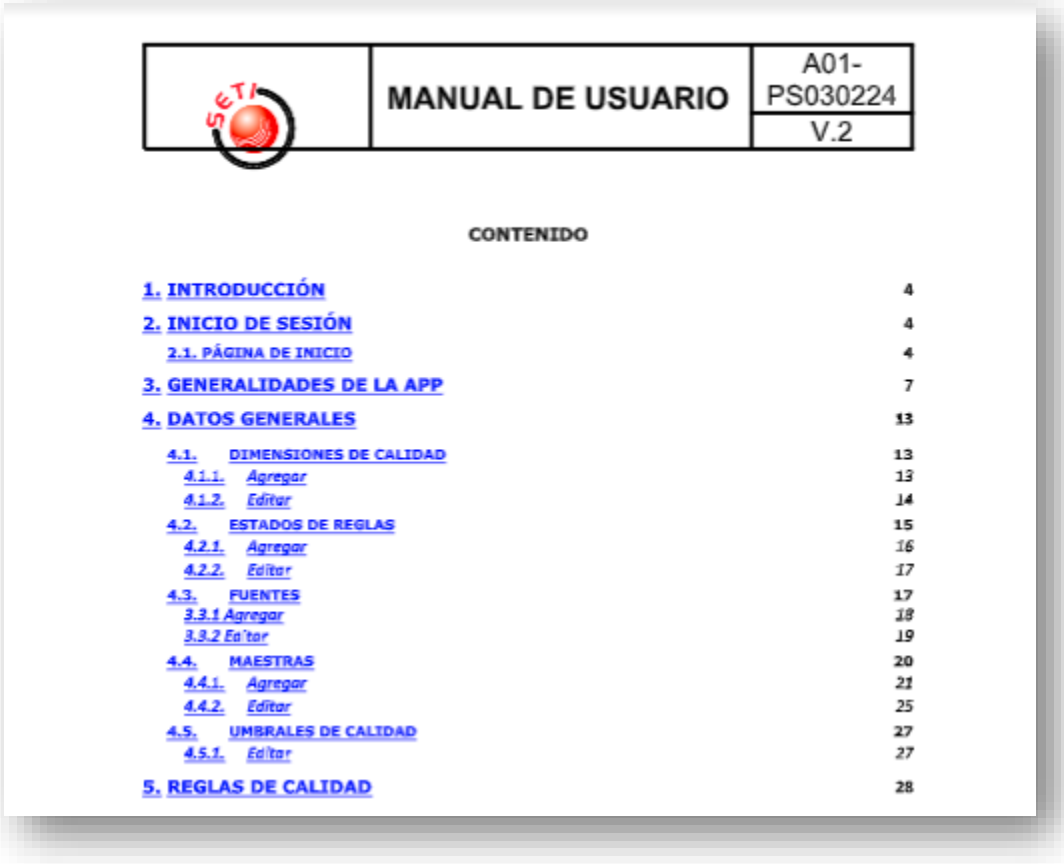
En cuanto a los merge requests, cada solicitud debía ser revisada por un supervisor antes de la fusión. Este enfoque preventivo garantizó la calidad del código, evitando la introducción de fragmentos confusos o perjudiciales para el desarrollo existente.

6.8 Manual de Usuario para la aplicación web

Con el objetivo de facilitar una experiencia de usuario óptima y maximizar la efectividad de la aplicación, se ha elaborado un completo Manual de Usuario. Este documento exhaustivo abarca todos los aspectos clave que los usuarios pueden encontrar al interactuar con la aplicación de gestión de calidad de datos para el Mercado de Energía Mayorista para la empresa XM.

Figura 45.

Ilustración de una parte del contenido del manual de usuario de la aplicación web.



MANUAL DE USUARIO | A01-PS030224 V.2

CONTENIDO

El Manual de Usuario incluye:

- Guía de Navegación:
  - Se proporciona una guía detallada sobre la navegación a través de las diferentes secciones y pantallas de la aplicación. Los usuarios pueden familiarizarse fácilmente con la estructura y el diseño de la interfaz.
- Funcionalidades Específicas:

- Cada funcionalidad y característica de la aplicación se explica en detalle. Esto incluye la creación, edición y eliminación de registros, la ejecución de microservicios a demanda, la visualización de historiales de cambios y la configuración de umbrales de calidad, entre otras.

- Validaciones y Casos Especiales:
  - Se detallan las validaciones y casos especiales que deben tenerse en cuenta al interactuar con diferentes partes de la aplicación. Esto asegura que los usuarios comprendan las restricciones y condiciones particulares asociadas con cada acción.

El Manual de Usuario se presenta como una herramienta valiosa para garantizar que los usuarios, desde nuevos ingresos hasta usuarios experimentados, puedan aprovechar al máximo todas las capacidades y funcionalidades que la aplicación tiene para ofrecer. Su formato claro y accesible brinda una referencia práctica que respalda la eficiencia y eficacia en el uso diario de la aplicación.

6.8 Fase de Pruebas y Retroalimentación de Usuarios Técnicos

Después de completar el desarrollo y alcanzar un estado funcional, la aplicación fue sometida a una fase crucial de pruebas. En este escenario, los usuarios técnicos, quienes poseen un conocimiento profundo de la plataforma y sus requisitos específicos, llevaron a cabo pruebas exhaustivas. Esta fase tuvo como objetivo identificar posibles fallos, evaluar la usabilidad y recopilar feedback esencial que contribuiría a la mejora continua de la aplicación.

Durante las pruebas:

- **Identificación de Bugs**:
  - Los usuarios técnicos exploraron la aplicación en busca de posibles bugs o errores de funcionamiento. Cualquier inconveniente identificado, desde pequeños errores de presentación hasta problemas de lógica de negocio, fue registrado y documentado.
- **Usabilidad y Experiencia del Usuario**:
  - Se evaluó la experiencia general del usuario, analizando la interfaz, la navegación y la respuesta de la aplicación. Este enfoque permitió identificar áreas donde la usabilidad podía mejorarse para garantizar una interacción fluida y sin obstáculos.
- **Feedback Detallado:**
  - Se recopiló un feedback detallado sobre cada aspecto de la aplicación. Los usuarios técnicos proporcionaron información valiosa sobre características específicas, flujo de trabajo y cualquier otro elemento relevante.
- **Corrección de Errores**:
  - Con base en la retroalimentación recibida, se llevaron a cabo correcciones y ajustes necesarios. Se abordaron los bugs identificados y se implementaron mejoras según las sugerencias proporcionadas por los usuarios técnicos.

Esta fase de pruebas y retroalimentación desempeñó un papel crucial en el perfeccionamiento de la aplicación, asegurando que estuviera lista para su implementación en el entorno de producción. La participación activa de los usuarios técnicos fue fundamental para garantizar que la aplicación cumpliera con los estándares de calidad y ofreciera una experiencia de usuario excepcional.

6.9 Implementación de Modelo DevOps y Despliegue en la Nube de Azure

En la etapa final del ciclo de desarrollo, se adoptó un enfoque de DevOps para agilizar y automatizar los procesos de desarrollo, pruebas y despliegue. Este modelo integral permitió una integración y entrega continua (CI/CD), reduciendo los tiempos de desarrollo y mejorando la eficiencia en la implementación de nuevas funcionalidades.

La implementación de DevOps incluyó:

- **Automatización del Despliegue**:
  - Se implementaron prácticas de CI/CD para automatizar la construcción y despliegue de la aplicación. Esto garantizó una entrega rápida y eficiente de nuevas versiones, así como la detección temprana de posibles problemas.
- **Contenedores para el Frontend y Backend**:
  - Se optó por el uso de contenedores para encapsular las aplicaciones, facilitando su implementación y garantizando que se ejecuten de manera coherente en cualquier entorno. Docker fue la tecnología principal utilizada para la creación de contenedores.
- **Despliegue en la Nube de Azure**:
  - La aplicación fue desplegada en la infraestructura en la nube de Azure. Esta plataforma proporciona escalabilidad, seguridad y servicios gestionados que contribuyen a un entorno de producción robusto.

Es importante señalar que, aunque la transición a DevOps y el despliegue en la nube de Azure son componentes esenciales de la implementación final, los detalles específicos de esta

fase están fuera del alcance del presente informe, ya que no tocó ser el responsable directo de dichas tareas.

7. Conclusiones

La conclusión de este proyecto refleja el cumplimiento integral de los objetivos establecidos. La aplicación web desarrollada ha demostrado ser una herramienta eficaz, simplificando significativamente las tareas de los usuarios técnicos responsables de la configuración y parametrización de los modelos de calidad de datos de la empresa XM. La implementación de esta solución no solo ha optimizado los procesos, sino que también ha sentado las bases para futuras innovaciones y mejoras en la gestión de la calidad de datos. Este logro representa un paso significativo hacia la eficiencia operativa y la excelencia en la gestión de datos para la empresa.

De este proyecto destaca no solo el cumplimiento exitoso de los objetivos propuestos, sino también el arduo aprendizaje que representó. Durante este trayecto, se adquirieron habilidades y conocimientos en diversas herramientas y tecnologías de vanguardia, fortaleciendo así la capacidad para abordar desafíos tecnológicos actuales.

Es fundamental resaltar que, en cada fase del proyecto, se mantuvo un compromiso inquebrantable con la privacidad de los datos de las personas. Se implementaron medidas sólidas para controlar las vulnerabilidades y garantizar un entorno seguro. Esta atención constante a la seguridad no solo cumplió con estándares, sino que también estableció un programa robusto y confiable.

El resultado final es una aplicación web que no solo cumple con las expectativas operativas, sino que también representa un hito en el continuo compromiso con la innovación, la privacidad y la seguridad en el desarrollo de soluciones tecnológicas.

8. Recomendaciones

Considerando el continuo desarrollo y evolución de la aplicación, se sugiere la implementación de un sistema de notificaciones en tiempo real para informar a los usuarios sobre el estado de ejecución de los microservicios. Esta característica aportaría una experiencia más interactiva y proactiva, mejorando la comunicación entre la plataforma y sus usuarios.

Siendo un poco más técnicos sería implementar un sistema de escucha de eventos que esté atento a la finalización de cada microservicio. Cuando se complete una tarea, este sistema activaría la notificación correspondiente para informar al usuario sobre el resultado de la operación.

Se sugiere también enfocar los esfuerzos en optimizar aún más la experiencia responsive de la aplicación, garantizando un acceso fluido y eficiente desde dispositivos móviles.

Referencias Bibliográficas

Batini, C., & Scannapieco, M. (2010). *Data Quality: Concepts, Methodologies and Techniques. Springer*.

Coppola, M. (2023, mayo 2). *Desarrollo web: qué es, etapas y principales lenguajes*. Hubspot.es. https://blog.hubspot.es/website/que-es-desarrollo-web

Forster, L. L. (2015). *Data quality: The accuracy dimension*. Createspace.

Guzmán, E., Máter, A., Nacional, D. M., Rosado, A., & Francesca, L. (s/f). UNIVERSIDAD NACIONAL DE EDUCACIÓN. Edu.pe. Recuperado el 25 de octubre de 2023, de https://repositorio.une.edu.pe/bitstream/handle/20.500.14039/3984/MONOGRAF%C3%8DA%20-%20ALVARADO%20ROSADO.pdf?sequence=1&isAllowed=y

Kleppmann, M. (2017). *Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems*. O'Reilly Media.

Kuma, V. (2015). *Power System Operation and Control*. Scitech Publications.

OWASP foundation, *the Open Source Foundation for Application Security*. (s/f). Owasp.org. Recuperado el 25 de octubre de 2023, de http://owasp.org

Sullivan, B., & Liu, V. (2011). *Web application security, a beginner's guide*. McGraw-Hill Education.

Unleashing data quality for the energy industry's success. (2023, junio 15). GDM. https://gdm-inc.com/unleashing-data-quality-for-the-energy-industrys-success/

Apéndices

Apéndice A. Especificaciones de los programas utilizados

- Framework para frontend: Angular versión 13
- Framework para Backend: Express JS version 4.18.2
- Sonarqube versión V 9.9.0-community
- Postman
- Sendgrid API V 7.7.0
- Docker
- Microsoft Azure
- Gitlab
- Microsoft Sql Server Management Studio 18
- Visual Studio Code V 1.69
- Balsamiq Wireframe
- Swagger V 4.1.6