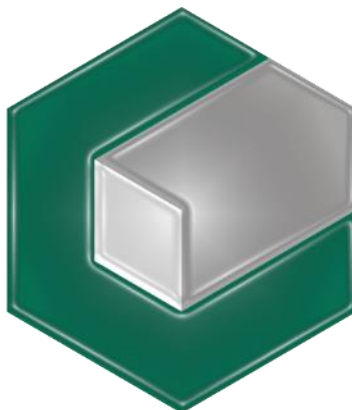


Universidad Tecnológica de La Habana “José Antonio Echeverría”

Facultad de Ingeniería Informática



**Desarrollo de capacidades habilitantes para la
adopción de industria 4.0 en los niveles de las
soluciones digitales gerenciales**

<Informe de la Práctica Profesional 1>

Autores:

Lilian Rosa Rojas Rodríguez - lrojas@ceis.cujae.edu.cu

Eduardo Alejandro González Martell - egonzalez@ceis.cujae.edu.cu

Tutores:

Dr. Carlos Ramón López Paz - carlosrl@aica.cu

Msc. Ana Lilian Infante Abreu - anai@aica.cu

Especialistas:

Ing. Pedro Velázquez Borrero - pedrydev@outlook.com

La Habana, Cuba

Marzo, 2024

Resumen

La Industria 4.0, que emplea tecnologías como el internet de las cosas (IoT) análisis de datos, IA y aprendizaje automático, digitaliza el sector industrial, optimiza procesos y aumenta la productividad para responder rápidamente a las fluctuaciones de la demanda, mejorando así la rentabilidad de una compañía. La empresa Laboratorios Farmacéuticos AICA+ del grupo BioCubaFarma está trabajando en un proyecto para desarrollar capacidades habilitantes que le permitan adoptar la Industria 4.0 en toda su infraestructura, incluyendo la digitalización de la documentación del Sistema de Gestión de la Calidad. Un sistema de documentación digital es esencial para esta gestión, ya que facilita la documentación y el intercambio de información sobre procesos, procedimientos y políticas. Esto ayuda a las partes interesadas a comprender los requisitos para mantener la excelencia empresarial. Este trabajo presenta la investigación realizada para desarrollar, implementar y probar eficientemente un sistema *backend* para resolver este problema. El desarrollo de este sistema se basa principalmente en que AICA+ es una empresa cuyas operaciones están completamente controladas por las agencias regulatorias farmacéuticas, por lo que la documentación no solo es abundante, sino que también cambia rápidamente bajo estrictos mecanismos de trazabilidad. La solución desarrollada se integra en la adopción de la Industria 4.0 y la Calidad 4.0, y no se limita al sector farmacéutico ni a AICA+, sino que puede aplicarse en otros sectores que necesiten un sistema de gestión de documentación similar.

Palabras Claves: Industria 4.0, capacidades habilitantes, backend, documentación, calidad 4.0.

Abstract

The Industry 4.0, which employs technologies such as the Internet of Things (IoT), data analysis, AI, and machine learning, digitizes the industrial sector, optimizes processes, and increases productivity to respond quickly to demand fluctuations, thus improving a company's profitability. The company Laboratorios Farmacéuticos AICA+ of the BioCubaFarma group is working on a project to develop enabling capabilities that allow it to adopt Industry 4.0 throughout its infrastructure, including the digitization of the documentation of the Quality Management System. A digital documentation system is essential for this management, as it facilitates the documentation and exchange of information about processes, procedures, and policies. This helps stakeholders understand the requirements to maintain business excellence. This work presents the research carried out to develop, implement, and efficiently test a backend system to solve this problem. The development of this system is mainly based on the fact that AICA+ is a company whose operations are completely controlled by pharmaceutical regulatory agencies, so the documentation is not only abundant, but also changes rapidly under strict traceability mechanisms. The developed solution integrates into the adoption of Industry 4.0 and Quality 4.0, and is not limited to the pharmaceutical sector or AICA+, but can be applied in other sectors that need a similar documentation management system.

Keywords: *Industry 4.0, enabling capacities, backend, documentation, quality 4.0.*

Índice de Contenido

Introducción.....	1
Capítulo 1. Fundamentación del proyecto	8
1.1 Descripción del proyecto	8
1.2 Fundamentación de las tecnologías a utilizar	10
1.3 Conclusiones Parciales	17
Capítulo 2. Solución Propuesta: AicaDocs	18
2.1 Lista de requisitos	18
2.2 Diseño de los sistemas para la persistencia de datos.....	19
2.2.1 Diseño de la base de datos.....	19
2.2.2 Diseño del sistema de almacenamiento de ficheros	24
2.3 Diseño de la solución	25
2.3.1 Conceptos fundamentales.....	25
2.3.2 Diseño de la solución	26
2.4 Desarrollo de la API	27
2.5 Despliegue de los servicios de la API	29
2.6 Conclusiones Parciales	32
Capítulo 3. Pruebas	33
3.1 Diseño de casos de pruebas.....	33
3.1.1 Conceptos fundamentales.....	33
3.1.2 Casos de pruebas diseñados.....	34
3.2 Resultado de las pruebas.....	42
3.3 Pruebas de integración automatizadas	43
3.3.1 Conceptos fundamentales.....	43
3.3.2 Diseño y resultado de las pruebas de integración.....	44
3.4 Conclusiones Parciales	45
Capítulo 4. Aplicación web: AicaDocs UI	46
4.1 Conceptos fundamentales.....	46
4.2 Desarrollo y despliegue de la aplicación web.....	48

4.3 Vistas de la aplicación web	50
4.4 Conclusiones Parciales	55
Conclusiones	56
Recomendaciones	57
Referencias Bibliográficas	58
Anexos	61

Índice de Tablas

Tabla 1: Sistema de tareas comunes y específicas (Fuente: Elaboración propia)	7
Tabla 2: Diseño de los endpoints de la solución AicaDocs (Fuente: Elaboración propia)	26
Tabla 3: Juego de datos para Documentos (Fuente: Elaboración propia)	34
Tabla 4: Juego de datos para Descargas (Fuente: Elaboración propia)	35
Tabla 5: Juego de datos para Nomencladores (Fuente: Elaboración propia) ..	35
Tabla 6: Pruebas de integración diseñadas (Fuente: Elaboración propia).	44

Índice de Figuras

Figura 1: Historia y características de las revoluciones industriales (Fuente: [3])	1
Figura 2: Sistema tradicional jerárquico de la industria en forma de pirámide (Fuente: [6])	2
Figura 3: Niveles de madurez de la Industria 4.0 (Fuente: [7])	3
Figura 4: Arquitectura de implementación de capacidades habilitantes para la transición a la Industria 4.0 (Fuente: [7])	4
Figura 5: Ecosistema AICA+ 4.0 (Fuente: AICA+)	5
Figura 6: Planificación del proyecto de AICA 4.0 (Fuente: AICA+)	8
Figura 7: Stack tecnológico en AicaDocs a utilizar (Fuente: elaboración propia)	11
Figura 8: Rendimiento de .Net (Fuente: Microsoft [22])	13
Figura 9: Pruebas de rendimiento de frameworks de desarrollo web 2023 (Fuente: TechEmpower [24])	14
Figura 10: Modelo físico de la base de datos diseñada (Fuente: Database Designer)	21
Figura 11: Consola de administración de MinIO (Fuente: MinIO)	24
Figura 12: Diagrama de clases UML de AicaDocs (Fuente: Elaboración propia)	27
Figura 13: Repositorio en Github de AicaDocs (Fuente: Github)	28
Figura 14: Implementación de Swagger UI en AicaDocs	29
Figura 15: Infraestructura de despliegue de AicaDocs (Fuente: Elaboración propia)	29
Figura 16: Dashboard de despliegue de la API de AicaDocs en Render (Fuente: Render)	30
Figura 17: Dashboard de despliegue de la base de datos de PostgreSQL de AicaDocs en ElephantSQL (Fuente: ElephantSQL)	31
Figura 18: Dashboard de despliegue del almacenamiento de objetos MinIO de AicaDocs en Railway (Fuente: Railway)	32
Figura 19: Resultado de pruebas de integración automatizadas con Github Actions (Fuente: Github)	45

Figura 20: Stack tecnológico a utilizar en AicaDocs UI (Fuente: Elaboración propia).	48
Figura 21: Repositorio en Github de AicaDocs UI (Fuente: Github).	49
Figura 22: Dashboard de despliegue de AicaDocs UI en Render (Fuente: Render)	50
Figura 23: Pantalla de Inicio de AicaDocs UI (Fuente: AicaDocs UI)	52
Figura 24: Pantalla de Listado de Documentos de AicaDocs UI (Fuente: AicaDocs UI)	52
Figura 25: Pantalla de Crear Documento de AicaDocs UI (Fuente: AicaDocs UI)	52
Figura 26: Pantalla de Descargar Documento de AicaDocs UI (Fuente: AicaDocs UI)	53
Figura 27: Pantalla de Detalles de Documento de AicaDocs UI (Fuente: AicaDocs UI)	53
Figura 28: Pantalla de Listado de Descargas de AicaDocs UI (Fuente: AicaDocs UI)	53
Figura 29: Pantalla de Detalles de Descarga de AicaDocs UI (Fuente: AicaDocs UI)	54
Figura 30: Pantalla de Listado de Nomencladores de AicaDocs UI (Fuente: AicaDocs UI)	54
Figura 31: Pantalla de Editar Nomenclador de AicaDocs UI (Fuente: AicaDocs UI)	54
Figura 32: Pantalla de Crear Nomenclador de AicaDocs UI (Fuente: AicaDocs UI)	55

Introducción

En la actualidad, la Industria 4.0 se alza como un faro de innovación y transformación en el panorama industrial [1]. Este capítulo de la historia está marcado por la evolución de las revoluciones precedentes y la convergencia de tecnologías avanzadas que han revolucionado la forma en que producimos, interactuamos y convivimos [2] (Ver **Figura 1**).

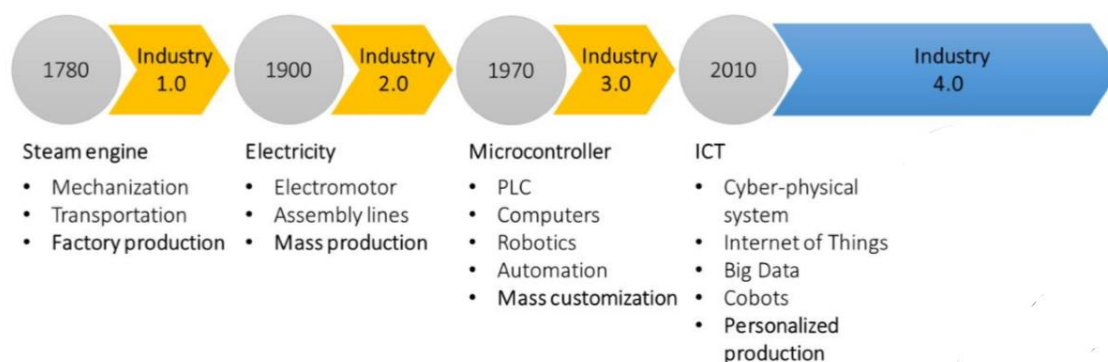


Figura 1: Historia y características de las revoluciones industriales (Fuente: [3])

La Industria 4.0 no es simplemente una evolución lineal, sino un salto cuántico hacia la automatización inteligente, la conectividad sin fronteras y la toma de decisiones basadas en datos [2]. A finales del siglo XX y principios del XXI, emergieron tecnologías disruptivas como el Internet de las Cosas (IoT), la Inteligencia Artificial (IA), la Analítica de Datos (Big Data) y la Computación en la Nube (Cloud Computing). Estas fuerzas convergieron para dar forma a la Industria 4.0 [4].

La Industria 4.0 establece una red interconectada entre las diversas entidades de una empresa, en contraste con los sistemas tradicionales que se organizaban jerárquicamente en forma de pirámide (Ver **Figura 2**). En lugar de una estructura rígida y vertical, la Industria 4.0 fomenta la colaboración horizontal y la comunicación fluida entre los componentes del proceso productivo en una estructura parecida a un grafo [5].

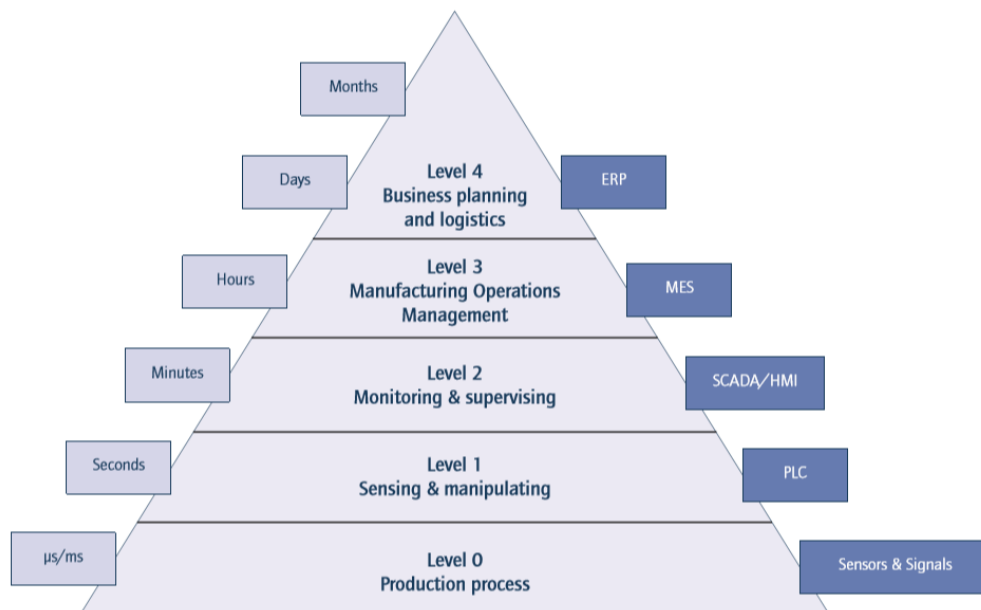


Figura 2: Sistema tradicional jerárquico de la industria en forma de pirámide (Fuente: [6])

Según [7], el potencial de creación de valor de la Industria 4.0 para la industria manufacturera alemana se estima que estará entre 70 y 140 mil millones de euros solo para el período hasta 2025. Este impacto económico refleja la transformación digital en la producción, la mejora de la productividad y la personalización masiva de productos.

El Índice de Madurez de la Industria 4.0 (Industrie 4.0 Maturity Index [7, 8]) es una herramienta crucial para guiar la transformación digital de las empresas. Fue introducido por acatech, la Academia Nacional de Ciencias e Ingeniería de Alemania [9, 10]. Proporciona a las empresas una guía para su propio proceso de transformación. Se basa en un modelo de madurez de seis etapas que analiza las capacidades necesarias en las áreas estructurales como recursos, sistemas de información, cultura y estructura organizativa [11].

La academia acatech denota 6 niveles de madurez (Ver **Figura 3**), donde cada nivel describe el desarrollo de una empresa ideal en el contexto de la Industria 4.0. Estos niveles garantizan la manejabilidad del proceso de transformación, que a menudo se extiende durante varios años [7].

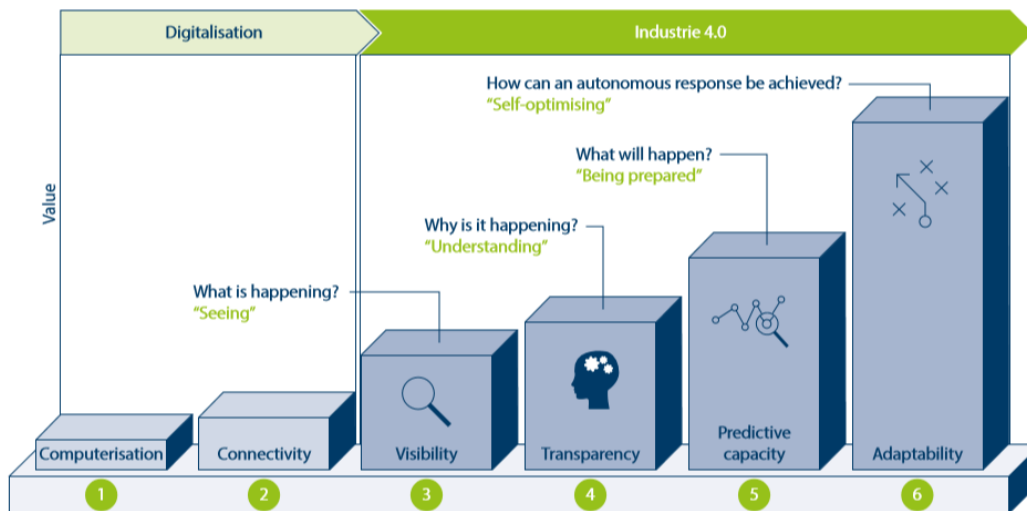


Figura 3: Niveles de madurez de la Industria 4.0 (Fuente: [7])

Los dos primeros niveles de madurez: Computarización y Conectividad, corresponden a las capacidades habilitantes para preparar a una empresa en cuestión, en su tránsito de la industria 3.0 vigente a la 4.0 [7, 8]. Los sistemas mayoritariamente implementados para otorgar dichas capacidades habilitantes son los sistemas SCADA (*Supervisory Control and Data Acquisition*), HMI (*Human-Machine Interface*), MES (*Manufacturing Execution System*), sistemas de documentación y de calidad [6, 12-15] (Ver **Figura 4**). La implementación de dichas tecnologías habilitantes permitiría como principales beneficios para cualquier empresa [11, 16-18]:

- **Automatización y control:** Supervisión y control de los procesos industriales en tiempo real, lo cual mejoraría la eficiencia y reduciría los errores humanos.
- **Recopilación de datos:** Los datos recopilados por SCADA son esenciales para el análisis y la toma de decisiones informada.
- **Visualización en tiempo real y Acceso intuitivo:** Los paneles HMI muestran datos en tiempo real, lo que facilita la monitorización y la detección temprana de errores.
- **Gestión de la producción:** Los MES optimizan la programación, el seguimiento y la ejecución de la producción.

- **Rastreo y trazabilidad:** Rastreo efectivo de productos, desde la materia prima, hasta la entrega del cliente, propiciando la fabricación de un producto inteligente.
- **Documentación digital:** La transición de la documentación en papel a sistemas digitales mejora la accesibilidad y reduce el riesgo de pérdida de información.
- **Control de calidad:** Estos sistemas garantizan que los productos cumplan con los estándares de calidad, muy complejos en sectores fuertemente regulados como el farmacéutico y el alimenticio.

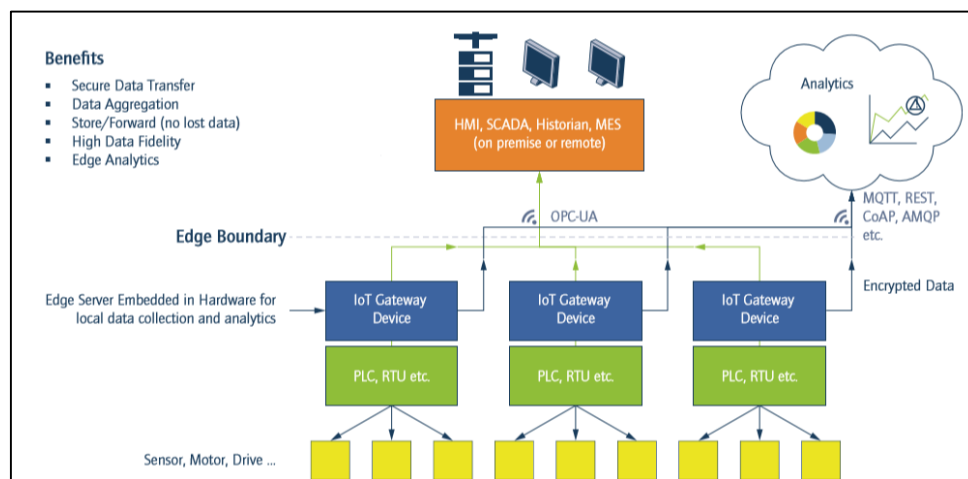


Figura 4: Arquitectura de implementación de capacidades habilitantes para la transición a la Industria 4.0 (Fuente: [7])

La empresa Laboratorios Farmacéuticos AICA+ perteneciente al grupo empresarial BioCubaFarma tiene como proyecto la integración de tecnologías digitales para la adopción de una iniciativa estratégica de Industria 4.0 en el sector biofarmacéutico cubano. Planea el desarrollo de un ecosistema de tecnologías que funcionen como capacidades habilitantes que faciliten dicha transición (Ver **Figura 5**). Este proyecto se encuentra en desarrollo activo, por lo que existen algunos arquetipos de sistemas como el documentado en [19]. Sin embargo, actualmente no se han concretado soluciones digitales que actúen como tecnologías habilitantes para cumplir el objetivo del proyecto.

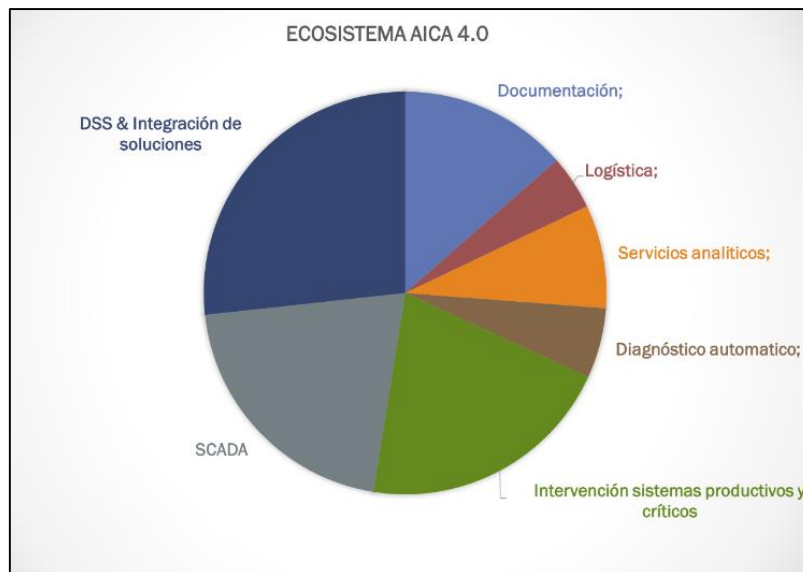


Figura 5: Ecosistema AICA+ 4.0 (Fuente: AICA+).

Por lo tanto, la **situación problemática** se presenta como la necesidad de capacidades habilitantes para la adopción de industria 4.0 en AICA+.

Por todo lo anteriormente argumentado se define como **problema de investigación** el ¿cómo desarrollar las capacidades habilitantes para la adopción de industria 4.0 en AICA+, específicamente en los niveles de las soluciones digitales gerenciales?

El **campo de estudio** en el cual se inserta la presente investigación es la Ingeniería de Sistemas y Tecnologías de la Información, aplicado al desarrollo de sistemas *backend*.

Para darle respuesta al problema planteado se define como **objetivo general** desarrollar un sistema *backend* que permita llevar el control de la documentación del Sistema de Gestión de Calidad de la empresa Laboratorios Farmacéuticos AICA+ perteneciente al grupo empresarial BioCubaFarma.

A partir de este surgen los siguientes **objetivos específicos**:

- Realizar un estudio en el estado del arte de los *stack* tecnológicos comúnmente utilizados para dar solución a los sistemas *backend* de gestión de la documentación de la calidad.

- Diseñar y desarrollar eficientemente el sistema de documentación, siguiendo los patrones de diseño frecuentemente implementados en este tipo de soluciones, y guiándose por las mejores prácticas respaldadas en la comunidad de desarrolladores.
- Desplegar el sistema desarrollado para probar su fiabilidad y usabilidad en entornos de producción.
- Diseñar y ejecutar pruebas al sistema desarrollado, tanto en entornos de producción, como de desarrollo, que permitan garantizar un grado de cobertura de fiabilidad y estabilidad suficientemente elevado como para reducir al mínimo los posibles fallos que pudiera cometer la solución digital.
- Desarrollar una aplicación web que permita probar la usabilidad y fiabilidad en el caso de uso del consumo de la API a desarrollar.
- Luego de llegar al planteamiento de los objetivos anteriores, se acordó el sistema de tareas comunes e individuales que se denota en la **Tabla 1**.

El **valor práctico** del presente trabajo radica en asimilar las tecnologías que permitan la implementación de soluciones digitales que contribuyan al desarrollo de las capacidades habilitantes para la adopción de la industria 4.0 en cualquier sector de la producción.

El presente trabajo se **estructura** en 4 capítulos:

- En el Capítulo 1 se realiza una descripción detallada del proyecto del sistema de control de la documentación del Sistema de Gestión de la Calidad a desarrollar. Además se documenta la investigación llevada a cabo para conformar y fundamentar la elección del *stack* tecnológico que se empleó en el desarrollo de la solución digital a desarrollar.
- En el Capítulo 2 se describe todo el proceso de diseño, desarrollo, implementación y despliegue de la solución *backend*.
- En el Capítulo 3 se detallan los diseños y resultados de la ejecución de las pruebas de caja negra desarrolladas. Además se resume el diseño y resultados obtenidos tras la ejecución de las pruebas de integración automatizadas planeadas para la API.

- En el Capítulo 4 se describe detalladamente el diseño, el desarrollo y el despliegue de una aplicación web para probar la usabilidad del consumo de la API.

Tabla 1: Sistema de tareas comunes y específicas (Fuente: Elaboración propia)

No	Lilian Rosa Rojas Rodríguez	Eduardo Alejandro González Martell
1	Asimilar el contexto de industria 4.0 en el dominio de calidad 4.0 (sistema de documentación)	
2	Asimilar las tecnologías habilitantes en los niveles gerenciales (componente backend)	
3	Diseñar sistema de documentación	
4	Implementar sistema de documentación - bases del proyecto (arquitectura, controlador de versiones, bases de datos)	
5	Implementar sistema de documentación: crear documento; filtro, ordenamiento y paginación de documentos; obtener documento por el id	Implementar sistema de documentación: nomencladores (alcance, proceso, tipo de documento, motivo de descarga)
6	Implementar sistema de documentación - listar descargas; filtro, ordenamiento y paginación de descargas	Implementar sistema de documentación - implementar sistema de archivos MinIO (crear documento, descargar documento)
7	Asimilar la infraestructura básica para las pruebas del backend	Despliegue del sistema de documentación para pruebas en entorno de producción (despliegue de la base de datos, sistema de archivos MinIO y de la API)
8	Ejecutar y analizar las pruebas diseñadas del backend	Desarrollar recurso web para probar el consumo de la API
9	Elaborar informe de la práctica	

Capítulo 1. Fundamentación del proyecto

El presente capítulo presenta la fundamentación teórica del proyecto y de las elecciones tecnológicas y de infraestructura llevadas a cabo. La primera sección desarrolla una descripción general y concreta del proyecto en el cual se fundamenta el presente trabajo. La segunda sección explica la infraestructura y las tecnologías empleadas en la solución.

1.1 Descripción del proyecto

La empresa Laboratorios Farmacéuticos AICA+ del grupo empresarial BioCubaFarma, se encuentra inmersa en un proyecto que permita desarrollar un ecosistema digital para la transformación de su planta de sueros en una planta inteligente desde el concepto de Industria 4.0 lo cual permitirá la integración de tecnologías digitales para la adopción de una iniciativa estratégica de Industria 4.0 en el sector biofarmacéutico cubano (Ver **Figura 6**).

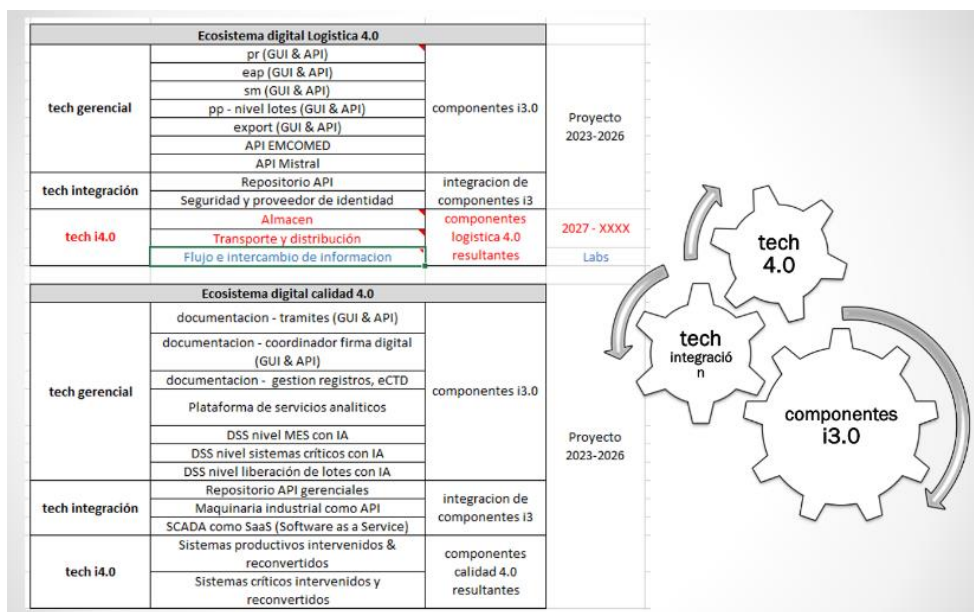


Figura 6: Planificación del proyecto de AICA 4.0 (Fuente: AICA+)

Entre los objetivos que persigue dicha investigación se encuentran:

- **Desarrollar un repositorio administrable de software especializado de tipo API (Application Programming Interface):** Permitirá ofrecer,

como servicio, los datos de las operaciones almacenadas que existen en los niveles de planta de producción, los sistemas críticos y los procesos gerenciales soportados por tecnologías digitales.

- **Desarrollar soluciones digitales:** Permitirá soluciones durante la ejecución de un estudio de caso que explore y priorice las necesidades de integración vertical y horizontal de los procesos gerenciales, los procesos tecnológicos y los sistemas críticos en un contexto estratégico de toma de decisiones de la empresa AICA+ y como parte de nuevos modelos de negocio digitales.

Vinculados a estos objetivos, el proyecto anterior pre-visualiza los siguientes resultados:

- **Plataforma para la administración de APIs** como tecnología digital especializada para la integración y administración de fuentes primarias de información de forma automática.
- **Solución digital para el proceso de Gestión de la Documentación** que acompañe la naturaleza de los cambios en el Sistema de Gestión de la Calidad bajo el marco regulatorio farmacéutico existente.

Con el objetivo de dar seguimiento y solución a los aspectos anteriores surge el presente proyecto, **AicaDocs**, el cual permitirá desarrollar una API con las funciones necesarias para poder desarrollar el *backend* de un sistema de trabajo digital para el área de proceso de gestión de la documentación, que forma parte del Sistema de Gestión de la Calidad de la empresa AICA+.

El desarrollo de este sistema se encuentra fundamentado principalmente en que AICA+ es una empresa donde todas sus operaciones están completamente controladas por las agencias regulatorias farmacéuticas, por tanto, el nivel de documentación no solo es numeroso, sino que cambia de manera muy acelerada bajo estrictos mecanismos de trazabilidad [19]. Por tanto las reglas de negocio del sector farmacéutico, y más específicamente, en el negocio de la elaboración de sueros, son muy complejos, por lo que la automatización de estos procesos permitiría la prevención de errores humanos producto del control manual del sistema y la rápida adaptabilidad propia de una modelación y solución digital [17].

El sistema a desarrollar se integra como parte de las capacidades habilitantes para la adopción de industria 4.0 en los niveles de las soluciones gerenciales, más concretamente en la rama de la Calidad 4.0, por lo que no es propio únicamente del sector farmacéutico, ni aplicable solamente en AICA+, lo cual permite desarrollar una solución que puede ser generalizada y aplicada en otros sectores empresariales que requieran de un sistema de gestión de la documentación con una estructura interna parecida al sistema desarrollado.

1.2 Fundamentación de las tecnologías a utilizar

El presente proyecto encuentra antecedentes en sistemas desarrollados con el fin de estudiar las posibles implementaciones de una solución digital para el sistema de gestión de la documentación en los laboratorios AICA+. Uno de estos estudios es resumido en la investigación [19]. En el estudio se recogen varias arquitecturas y modelos de implementación posibles para dar solución a la problemática del proyecto, así como las tecnologías a utilizar e infraestructura.

Luego de realizar una investigación de las principales tecnologías existentes para el desarrollo del sistema de documentación, se llegó, en una primera iteración y basándose solamente en el desarrollo del *backend*, al *stack* de tecnologías denotadas en la **Figura 7**.

Como plataforma de almacenamiento del repositorio del proyecto se escogió Github, plataforma de desarrollo colaborativo que aloja proyectos en la nube utilizando el sistema de control de versiones llamado Git. Github presenta como principales ventajas [20]:

- **Colaboración efectiva:** Permite una colaboración fluida y efectiva entre desarrolladores, lo cual facilita el trabajo en equipo y la revisión de código, ambos fundamentales en el desarrollo del presente proyecto.
- **Control de versiones:** Con el sistema de control de versiones de Git, los desarrolladores pueden realizar un seguimiento exhaustivo de los cambios realizados en el código.

- **Comunidad activa:** Cuenta con una potente comunidad de desarrolladores de todo el mundo, lo cual permite la resolución rápida de cualquier inconveniente durante el desarrollo del *software*.
- **Personalización de cualquier servicio host en la nube:** Esta característica permite configurar ciertos aspectos del código para poder lograr un rápido despliegue.

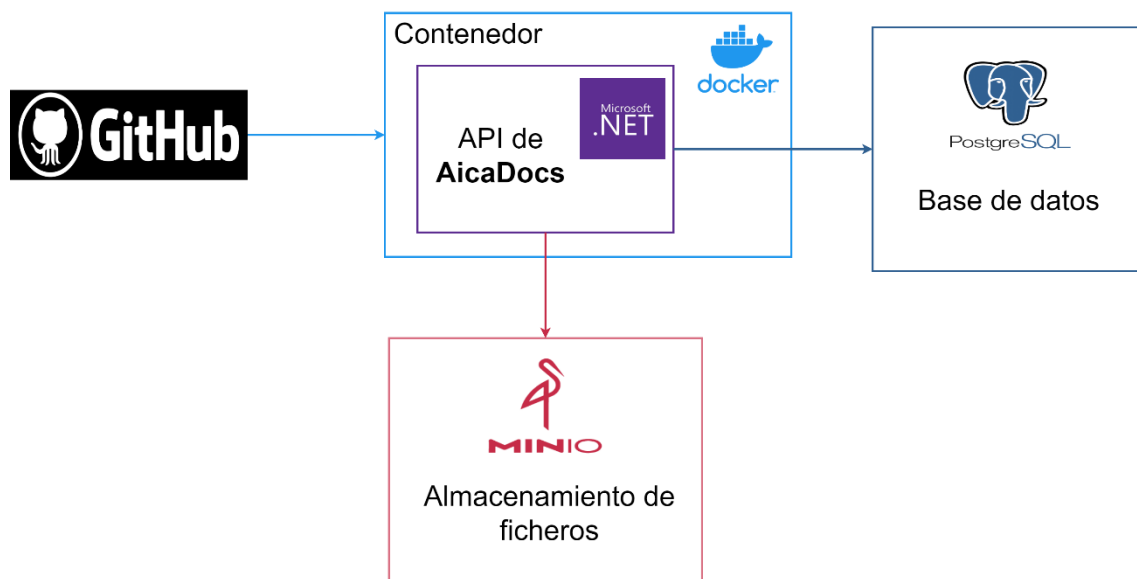


Figura 7: *Stack* tecnológico en **AicaDocs** a utilizar (Fuente: elaboración propia).

Github presenta entre sus servicios Github Actions, una plataforma de integración continua y entrega continua (CI/CD). Esto permite realizar automatizaciones sin preocuparse por características propias del servidor. El proyecto actual empleó esta característica para la ejecución y procesamiento de los resultados de las pruebas de integración automatizadas [20].

Como herramienta para facilitar el CI/CD, se escogió los contenedores *dockerizados*. Docker es una plataforma de *software* que permite crear, probar e implementar aplicaciones rápidamente. Docker utiliza la tecnología de contenedores, que posibilita la creación y uso de los contenedores en Linux. Un contenedor Docker es un paquete de *software* con todas las dependencias

necesarias para ejecutar una aplicación específica [21]. Se seleccionó esta tecnología para este propósito debido a que [21]:

- **Aislamiento eficiente:** Proporciona un aislamiento eficiente, lo que significa que cada aplicación y sus dependencias se mantienen separadas del sistema host.
- **Portabilidad:** Los contenedores *dockerizados* pueden implementarse en cualquier lugar, lo que garantiza la coherencia en todos los entornos.
- **Eficiencia de recursos de infraestructura.**
- **Facilidad de despliegue y escalabilidad.**
- **Consistencia entre entornos:** Con Docker, si una característica funciona en un entorno, funcionará en otros también, lo cual permite que la construcción y prueba de una aplicación en un contenedor *dockerizado* facilita la prevención de comportamientos inesperados.

Como plataforma de desarrollo se escogió .Net. Es una plataforma de aplicaciones de código abierto respaldada por Microsoft. Entre las razones fundamentales por la cuales se seleccionó se encuentran [22]:

- **Productividad:** Ofrece características avanzadas del lenguaje, como genéricos, LINQ y programación asincrónica, junto con amplias bibliotecas de clases y compatibilidad con varios lenguajes para desarrolladores.
- **Desarrollo multiplataforma:** Con .NET, puede tener como destino cualquier tipo de aplicación que se ejecute en cualquier plataforma. Los desarrolladores pueden reutilizar aptitudes y código en todos ellos en un entorno familiar. Esto significa que los desarrolladores pueden crear aplicaciones más rápido, con menos costo.
- **Gran ecosistema:** .NET es una plataforma de desarrollo moderno, innovador y de código abierto, con más de 5 millones de desarrolladores en todo el mundo. Se clasificó como el marco más preferido en la encuesta para desarrolladores de Stack Overflow durante tres años seguidos (2019, 2020 y 2021).
- **Desarrollo de soluciones orientadas al entorno empresarial.**

- **Rendimiento óptimo:** Las aplicaciones en este entorno ofrecen mejores tiempos de respuesta y requieren menos potencia de procesamiento (Ver **Figura 8**).
- **Integración con el ecosistema de soluciones AICA:** Las soluciones digitales en desarrollo o en planificación están dentro de esta plataforma de desarrollo por lo que permite una mejor integración entre las mismas.

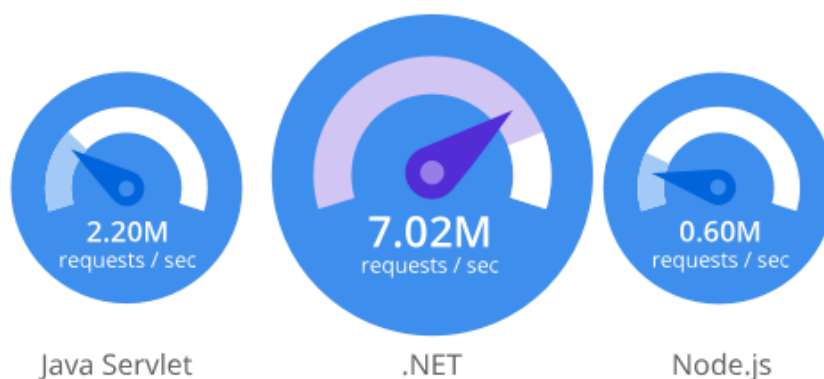


Figura 8: Rendimiento de .Net (Fuente: Microsoft [22]).

Como *framework* para desarrollar la API se escogió ASP.NET Core. Es un marco de desarrollo web moderno y de alto rendimiento para .NET. Se ejecuta en Windows, Linux, MacOS y Docker [22]. Algunas de las funcionalidades claves de ASP.NET Core son [23, 24]:

- **Rendimiento:** Según las pruebas realizadas en el banco de pruebas TechEmpower (Ver **Figura 9**), donde se comparan los marcos de aplicaciones web con tareas como serialización JSON, el acceso a bases de datos y la representación de plantillas del lado servidor, resulta que ASP.NET Core es uno de los más rápidos.
- **Moderno e innovador:** Está diseñado para permitir que los componentes en tiempo de ejecución, las API, los compiladores y los lenguajes evolucionen rápidamente, a la vez que proporciona una plataforma estable y compatible para mantener las aplicaciones en ejecución.
- **Código Abierto:** Al igual que el resto de .Net, ASP.NET Core es de código abierto en Github.

- **Implementación flexible:** El entorno de ejecución de ASP.NET Core en el que se ejecuta la aplicación se puede implementar como parte de la aplicación o instalarse de forma centralizada en el servidor web.

Rnk	Framework	Best performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	IA
1	gnet	7,013,961	100.0% (100.0)	0	Plt	Go	Non	Lin	Rea
2	faf	7,010,014	99.9% (99.9)	0	Plt	rs	Non	Lin	Rea
3	aspcore	7,006,142	99.9% (99.9)	0	Plt	C#	.NE	kes	Lin
4	pico.v	7,005,808	99.9% (99.9)	0	Mcr	v	Non	Non	Lin
5	frenio-http-lite	7,004,779	99.9% (99.9)	0	Plt	Jav	fir	Non	Lin
6	ntex [raw]	7,000,539	99.8% (99.8)	0	Plt	rs	Non	nte	Lin
7	cinatra	6,998,076	99.8% (99.8)	0	Ful	C++	Non	Non	Lin
8	wizzardo-http	6,995,129	99.7% (99.7)	0	Mcr	Jav	Non	Non	Lin
9	lithium	6,993,976	99.7% (99.7)	0	Mcr	C++	Non	Non	Lin
10	redkale-graalvm	6,981,831	99.5% (99.5)	0	Ful	Jav	red	red	Lin
11	libreactor	6,971,119	99.4% (99.4)	0	Plt	C	Non	Non	Lin
12	mormot [orm]	6,970,546	99.4% (99.4)	0	Ful	pas	Non	Non	Lin

Figura 9: Pruebas de rendimiento de *frameworks* de desarrollo web 2023 (Fuente: TechEmpower [24]).

Además de las virtudes anteriores que convierten a ASP.NET Core en una opción recomendable para el desarrollo *backend* de web APIs, debido a las características propias del *framework*, incluye soporte activo de herramientas que permiten escribir código siguiendo las mejores prácticas y patrones de diseño. Entre dichas características y herramientas se encuentran [23, 25-27]:

- **Inyección de dependencias:** La inyección de dependencias es un patrón de diseño orientado a objetos que suministra objetos a una clase en lugar de ser la propia clase la que crea dichos objetos. Se encarga de extraer la responsabilidad de la creación de instancias de un componente para delegarla en otro. Trae como principales beneficios la reducción del acoplamiento en el sistema, haciendo que los componentes que forman parte de la aplicación sean más independientes y sencillos de mantener; la testabilidad; la reutilización de código; y la flexibilidad y mantenimiento del código. En ASP.NET Core este patrón de diseño se implementa a través de un contenedor de servicios integrados, lo que permite la disponibilidad automática de estos en cualquier controlador de la aplicación.

- ***Middlewares***: Son componentes de *software* que facilitan la conexión de aplicaciones que no han sido diseñadas para conectarse entre sí y proporcionan la funcionalidad para conectarlas de manera inteligente. Entre las virtudes que trae consigo su implementación se encuentran la comunicación eficiente, la interoperabilidad, el ahorro de trabajo y el procesamiento distribuido. En ASP.NET Core, los *middlewares* se implementan a través de una canalización de servicios integrada, de manera que cuando una petición llega a un controlador, ASP.NET Core busca los servicios disponibles y automáticamente devuelve el servicio requerido cuando el controlador lo necesite.
- **Linq**: *Language Integrated Query* es una tecnología desarrollada por Microsoft que permite realizar consultas y manipulación de datos directamente en lenguajes como C#. Entre las virtudes de Linq se encuentran que provee un lenguaje familiar, puesto que los desarrolladores no tienen que aprender un nuevo lenguaje de consulta para cada tipo de fuentes de datos o formato de datos; menos codificación, pues reduce la cantidad de código que se debe escribir en comparación con un enfoque más tradicional; código legible; forma estandarizada de consultar múltiples fuentes de datos y compilación de seguridad de tiempo de las consultas.
- **ORM Entity Framework Core**: Un ORM (*Object-Relational Mapping*) es un modelo de programación que permite mapear las estructuras de una base de datos relacional (SQL Server, Oracle, MySQL, PostgreSQL) en una estructura lógica de entidades. Esto simplifica y acelera el desarrollo de aplicaciones al permitir a los desarrolladores trabajar con los datos como si fueran objetos en lugar de tablas de bases de datos. Entre los beneficios de usar un ORM incluyen la simplicidad y velocidad de uso, la reducción de código repetitivo, la facilidad de la portabilidad del código y la seguridad. Entity Framework Core es una herramienta ligera, extensible, de código abierto y multiplataforma que permite prescindir de la mayor parte del código de acceso a datos; es compatible con múltiples motores de bases de datos; permite generar modelos a partir de una base de datos existente o no; las instancias de las clases de entidad se recuperan de la base de datos por medio de Linq; y los datos se crean, se

eliminan y se modifican en la base de datos mediante instancias de las clases de entidad.

- **Métodos de extensión:** La extensión de clases es una característica de la programación orientada a objetos que permite añadir nuevos métodos a una clase existente sin modificar su código fuente. Esto se logra a través de los métodos de extensión, que son métodos estáticos que se pueden llamar como si fueran métodos de instancia en el tipo extendido, lo cual permite la reutilización de código, una mejor organización del código y flexibilidad.
- **Repository:** Es un patrón de diseño que se encarga de encapsular y centralizar la lógica relativa a la persistencia de los datos. Se basa en la idea de que todos los datos de una aplicación deben estar almacenados en un único lugar, conocido como repositorio. De esta manera, se obtiene una capa extra de abstracción en los componentes de la aplicación, evitando la repetición del código (principio DRY), facilitando el testeo y desacoplándolo de la lógica del código. En ASP.NET Core se implementa a través de un patrón de repositorio genérico con métodos asincrónicos en una API web
- **Unit of Work:** Es un patrón de diseño que se utiliza para manejar transacciones en una aplicación. Es útil cuando se tienen varias operaciones de base de datos que deben ser ejecutadas de manera transaccional, es decir, todas deben tener éxito o todas deben fallar. Se trata de un patrón en el cual internamente el objeto *Unit of Work* mantiene una lista de cambios a las entidades o demás operaciones a realizar, y cuando se desean escribir en la base de datos, se ejecuta como una única operación (transacción). En ASP.NET Core se implementa este patrón en Entity Framework Core en clases como DbContext, que funciona como una estructura homóloga a la base de datos, lo cual permite mantener en memoria la lista de cambios a las modificaciones de los objetos (dentro de las tablas representadas como DbSet<T>), para luego aplicarlos mismos tras la ejecución de los métodos `.SaveChanges()` o `.SaveChangesAsync()`.

Como método de almacenamiento de ficheros se escogió MinIO. Es un servidor de almacenamiento en la nube que es compatible con el API de Amazon S3. Es un software de almacenamiento de objetos de código abierto creado para almacenar y proteger datos no estructurados como videos, archivos de registros y ficheros de texto [28]. MinIO presenta como principales ventajas [28]:

- **Alta disponibilidad de los datos.**
- **Alto desempeño en operaciones de lectura/escritura.**
- **Almacenamiento *self-hosted*.**
- **Potente consola de administración.**

Como base de datos para la información principal de la lógica de negocios se escogió PostgreSQL. Es una base de datos de código abierto que tiene una sólida reputación por su fiabilidad, flexibilidad y soporte de estándares técnicos abiertos. A diferencia de otros sistemas de gestión de bases de datos relacionales (RDBMS) PostgreSQL soporta varios tipos de datos relacionales y no relacionales. Esto la convierte en una de las bases de datos más compatibles, estables y maduras disponibles actualmente, con más de 35 años de desarrollo y soporte continuo [29].

1.3 Conclusiones Parciales

En este capítulo se presentó la fundamentación teórica del proyecto que sustenta el presente trabajo. Se realizó un análisis de las diferentes implementaciones de otros sistemas de documentación que representaron los antecedentes del sistema actual en desarrollo. Se logró llegar a un diseño concreto de infraestructura y tecnologías a emplear resultando como las principales herramientas del *stack* tecnológico: Github, Github Actions, MinIO, Docker, PostgreSQL, .Net y ASP.NET Core.

Capítulo 2. Solución Propuesta: AicaDocs

El presente capítulo presenta el diseño, desarrollo y despliegue de la solución propuesta **AicaDocs**. La primera sección lista los principales requisitos funcionales y no funcionales que debe presentar el *software*. La segunda y tercera sección denota los diseños fundamentales de la base de datos, el sistema de almacenamiento de objetos, y la solución, explicando las principales decisiones llevadas a cabo. El resto del capítulo presenta el proceso del desarrollo de la API y el despliegue de la misma para comprobar su comportamiento y eficiencia en entornos de producción.

2.1 Lista de requisitos

El presente proyecto se centra en resolver la problemática presente en la empresa Laboratorios Farmacéuticos AICA+ del grupo empresarial BioCubaFarma, la cual requiere de un sistema digital que permita el control de la documentación del Sistema de Gestión de Calidad de la empresa. Los requisitos funcionales de la solución a desarrollar se recogen a continuación:

- **Persistencia de los datos:** El sistema debe permitir la persistencia de los datos.
- **Modelación de los documentos:** De los documentos se conocen principalmente el código, el título, la edición, las páginas, la fecha de vigencia y dos archivos asociados (pdf y word).
- **Organización específica de la documentación:** La documentación se organiza de acuerdo a un alcance, que puede ser Rector (aplica en toda la empresa) o específico (aplica en una UEB); en cuanto a proceso, que corresponde con uno de los procesos definidos en el mapa de procesos de la organización; y al tipo de documento, que es definido por los especialistas.
- **Creación de documentos:** El sistema debe permitir la creación de documentos, así como su correcto almacenamiento para garantizar la persistencia de los datos creados.
- **Búsqueda de documentos:** El sistema debe permitir la búsqueda de documentos, con filtros asociados a cada propiedad del documento.

- **Descarga de documentos:** El sistema debe permitir la descarga de documentos, donde se especifique el archivo y el formato en que se desea descargar (pdf o word).

Por otra parte, los requisitos no funcionales del sistema se recogen a continuación:

- **Usabilidad:** El sistema debe tener su funcionamiento documentado de modo que pueda ser incluido en un flujo sin necesidad de analizarlo a bajo nivel, ni requerir el código; es decir, debe ser utilizable sin requerir conocimiento sobre su funcionamiento más allá de la documentación visible.
- **Fiabilidad:** El sistema debe ser integral, estable y fiable, de modo que se reduzcan al mínimo los posibles fallos que pudiera estar presentando durante su explotación en producción.

2.2 Diseño de los sistemas para la persistencia de datos

2.2.1 Diseño de la base de datos

Uno de los requisitos funcionales de la solución a desarrollar es la persistencia de los datos. En cuanto a los datos propios de la lógica de negocios, es decir, las trazas de las descargas de los documentos, las diferentes clasificaciones de los documentos, y la información general de los documentos, se decidió para lograr el cumplimiento de este requisito, la modelación de una base de datos en PostgreSQL, cuya decisión se encuentra fundamentada en el capítulo anterior.

Durante la etapa de diseño, las entidades identificadas con sus atributos fueron:

- **Document:** Esta entidad se refiere a la modelación de los documentos y cuenta con los siguientes atributos:
 - **Id:** Identificador único del documento.
 - **Code:** Código del documento.
 - **Title:** Título del documento.
 - **Edition:** Número de edición del documento.
 - **Pages:** Número de páginas del documento.

- **DateOfValidity:** Fecha y hora con zona horaria de vigencia del documento, o sea, fecha a partir de la cual el documento entra en vigor.
- **TypeId:** Tipo de documento.
- **ProcessId:** Tipo de proceso de documento.
- **ScopeId:** Alcance de documento.
- **Download:** Esta entidad se refiere a la modelación de las trazas de las descargas y cuenta con los siguientes atributos:
 - **Id:** Identificador de la descarga.
 - **DateOfDownload:** Fecha y hora con zona horaria de la descarga.
 - **Format:** Formato del documento descargado.
 - **Username:** Usuario que realizó la descarga.
 - **DocumentId:** Documento descargado.
 - **ReasonId:** Razón de descarga.
- **Nomenclator:** Esta entidad se refiere a la modelación de los diferentes nomencladores (tipo de documento, proceso de documento, alcance de documento y razón de descarga) y cuenta con los siguientes atributos:
 - **Id:** Identificador del nomenclador.
 - **Name:** Nombre del nomenclador.
 - **Type:** Tipo de nomenclador.

Durante la etapa de diseño, también se identificaron las siguientes relaciones entre las entidades:

- **Document-Download:** Entre la entidad **Document** y la entidad **Download** existe una relación de 1 a muchos debido a que un documento se puede corresponder con múltiples descargas, pero una descarga solo puede corresponderse con un documento específico. De esta manera el extremo 1 se corresponde con la entidad **Document** y el extremo muchos con la entidad **Download**.
- **Nomenclator-Download:** Entre la entidad **Nomenclator** y la entidad **Download** existe una relación de 1 a muchos debido a que un nomenclador de tipo “razón de descarga” se puede corresponder con múltiples descargas, pero una descarga solo puede corresponderse con una “razón de descarga” específica. De esta manera el extremo 1 se

corresponde con la entidad **Nomenclator** y el extremo muchos con la entidad **Download**.

- **Nomenclator-Document:** Entre la entidad **Nomenclator** y la entidad **Do** existe una relación de 1 a muchos debido a que un nomenclador de tipo “razón de descarga” se puede corresponder con múltiples descargas, pero una descarga solo puede corresponderse con una “razón de descarga” específica. De esta manera el extremo 1 se corresponde con la entidad **Nomenclator** y el extremo muchos con la entidad **Download**.

Luego de la etapa de diseño, y un proceso de normalización, se llegó al siguiente modelo lógico global de datos:

Document(DocumentId, Code, Title, Edition, Pages, DateOfValidity, NomenclatorId [TypeId], NomenclatorId [ProcessId], NomenclatorId [ScopeId])

Download(DownloadId, DocumentId, NomenclatorId [ReasonId], DateOfDownload, Format, Username)

Nomenclator(NomenclatorId, Name, Type)

Además, se llegó a la modelación física de la base de datos de la **Figura 10**.

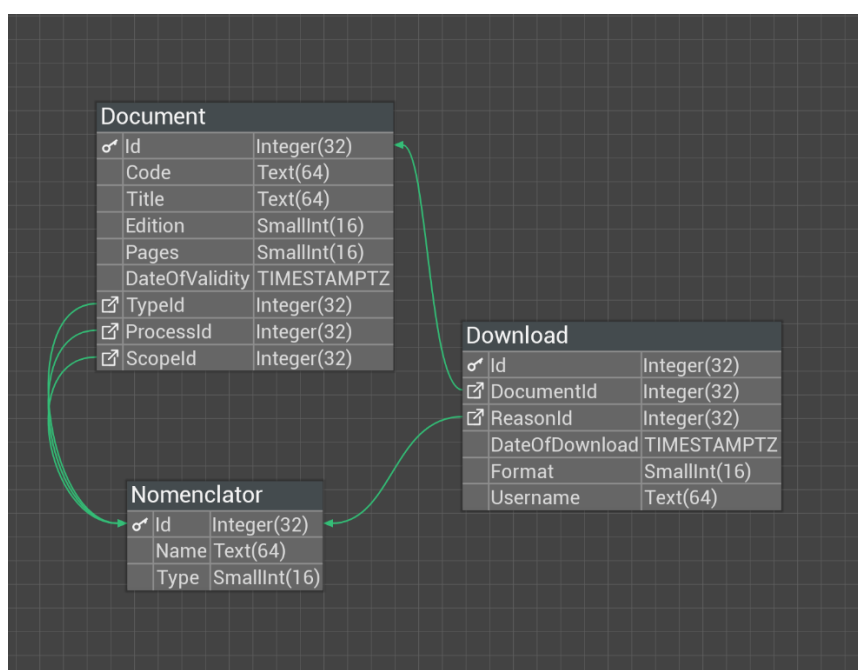


Figura 10: Modelo físico de la base de datos diseñada (Fuente: Database Designer)

Las consideraciones fundamentales tomadas en cuenta a la hora de diseñar la base de datos anterior se resumen a continuación:

- Los nomencladores tienen una implementación de herencia con discriminador (Type), lo cual permite simplificar la modelación, pues elimina de modelar una tabla por cada tipo de nomenclador. Esta implementación es posible, debido a que los nomencladores no presentan atributos específicos, que no sean comunes con los otros tipos.
- Los formatos admisibles son:
 0. Pdf
 1. Word
- Los tipos de nomenclador admisibles son:
 0. Proceso de documento
 1. Razón de descarga
 2. Alcance de documento
 3. Tipo de documento

Luego de la etapa de diseño y modelación, se obtuvieron los siguientes nomencladores para poblar de datos dicha tabla en la base de datos a generar:

- **Proceso de documento:**
 1. Gestión del Capital humano
 2. Gestión de Servicios generales
 3. Gestión de Ingeniería y mantenimiento
 4. Gestión de la Dirección
 5. Gestión de Comercial y de negocios
 6. Gestión de la Inversiones
 7. Gestión de Producción
 8. Gestión de Información y comunicación
 9. Gestión de Logística
 10. Gestión de Tecnologías Informáticas
 11. Gestión Contable-financiera
 12. Gestión de Documentación
 13. Gestión de Medición, análisis y mejora
 14. Gestión de Diseño, desarrollo e investigación

15. Gestión de Liberación de lotes

- **Razón de descarga:**

1. Distribución Impresa
2. Envío a personal externo autorizado
3. Elaboración de CTD
4. Solicitud del CECMED
5. Revisión por el área de documentación
6. Revisión del área de proceso

- **Alcance de documento:**

1. Rector
2. Aica UEB
3. Citox
4. Julio Trigo
5. Liorad
6. SH+
7. UDI

- **Tipo de documento:**

1. Procedimiento Normalizado de Operaciones
2. Registro
3. Instructiva
4. Especificación de Calidad
5. Plan Máster File
6. Planes
7. Ficha de proceso
8. Expediente maestro
9. Política
10. Programa
11. Manual
12. Manual de equipo
13. Plano
14. Documentación Técnica
15. Ficha Datos de Seguridad

2.2.2 Diseño del sistema de almacenamiento de ficheros

Uno de los requisitos funcionales de la solución a desarrollar es la persistencia de los datos. Se decidió emplear el sistema de almacenamiento de objetos Minio para poder administrar de manera eficiente los ficheros pdf y word de los documentos, lo cual se encuentra fundamentado en el capítulo anterior.

MinIO ofrece principalmente 2 herramientas fundamentales: la consola (Ver **Figura 11**), por la cual se pueden realizar tareas de administración tanto de los datos, como de los usuarios, los permisos y configuraciones generales; y la S3-API, la cual ofrece diversos *endpoints* para la administración de los datos.

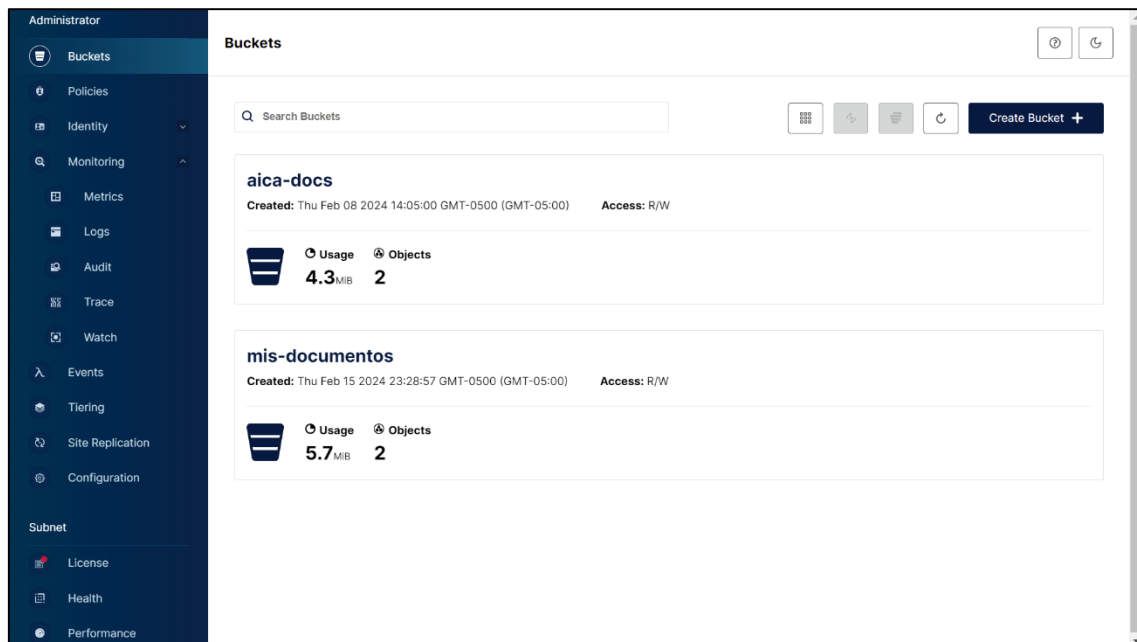


Figura 11: Consola de administración de MinIO (Fuente: MinIO)

Un *bucket* en MinIO es un almacén de objetos o ficheros. Se creó un *bucket* “aica-docs”, donde se almacenarán los datos de los ficheros en una estructura, donde los archivos pdf se guardan dentro de la carpeta “pdf” y los ficheros “docx” se guardan dentro de la carpeta “word”.

En MinIO se puede lograr la administración efectiva de los usuarios, otorgándoles permisos de uso específicos, lo cual se traduce en una mayor seguridad de la aplicación. Se creó un usuario “testuser” con derechos de escritura-lectura, que se empleará luego en la capa de desarrollo y producción

para establecer la comunicación cliente-servidor entre la API de **AicaDocs** y la API de MinIO.

2.3 Diseño de la solución

2.3.1 Conceptos fundamentales

Un *endpoint* es una URL de una API o un *backend* que se encarga de contestar a una petición. Es una ubicación digital concreta a la que se envían peticiones de información con el objetivo de obtener como respuesta la información que está en dicho punto. Los *endpoints* concretan los puntos a los que las APIs pueden acceder para conseguir recursos y ayudan a garantizar el funcionamiento correcto del *software* en el que se encuentran [30].

HTTP (*Hypertext Transfer Protocol*) es el protocolo fundamental de comunicación en la *World Wide Web*. Su propósito principal es permitir la transferencia de información, como texto, gráficos, sonidos, videos y otros archivos multimedia, entre un cliente y un servidor [31].

Los métodos de petición HTTP, también conocidos como “request methods”, son un conjunto de comandos que indican la acción que se desea realizar para un recurso determinado. Los principales métodos de petición HTTP son [31]:

- **GET**: Este método solicita una representación de un recurso específico. Las peticiones que usan el método GET sólo deben recuperar datos.
- **POST**: Este método se utiliza para enviar una entidad a un recurso en específico, causando a menudo un cambio en el estado o efectos secundarios en el servidor.
- **PUT**: Este método reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.
- **DELETE**: Este método borra un recurso en específico.
- **PATCH**: Este método es utilizado para aplicar modificaciones parciales a un recurso.

2.3.2 Diseño de la solución

Se logró el diseño efectivo del *software* a través de diferentes *endpoints* que permitieron dar solución a los requisitos funcionales del presente proyecto. Dicho diseño se encuentra modelado en la **Tabla 2** a continuación.

Tabla 2: Diseño de los *endpoints* de la solución **AicaDocs** (Fuente: Elaboración propia)

Nº	Sección	Petición	Ruta	Descripción	Parámetros	Body	MIME	Respuestas Posibles
1	Documents	GET	/document/{id}	Obtiene un documento dado el id	id: ID del documento	Ninguno	Ninguno	200 OK (body), 404 Not Found Error (body)
2	Documents	POST	/document/filter	Permite filtrar, ordenar y paginar los documentos	Ninguno	{ "code": "string", "title": "string", "edition": 32000, "pages": 32000, "dateOfValidity": "2024-02-27T08:15:39.603Z", "typeId": 1, "processId": 1, "scopeId": 1, "paginationParams": { "pageNumber": 1, "pageSize": 1 }, "sortBy": 0, "sortOrder": 0, "dateComparator": 0 }	application/json	200 OK (body), 400 Bad Request Error (body)
3	Documents	POST	/document	Crea un nuevo documento	Ninguno	{ "title": "string", "code": "code", "edition": "1", "dateOfValidity": "2024-02-27T08:24:35.226Z", "typeId": "1", "processId": "1", "scopeId": "1", "pdf": "local.pdf", "word": "word.docx" }	multipart/form-data	201 Created, 400 Bad Request Error (body)
4	Downloads	GET	/download/{id}	Obtiene una descarga dado el id	id: ID de la descarga	Ninguno	Ninguno	200 OK (body), 404 Not Found Error (body)
5	Downloads	POST	/download/filter	Permite filtrar, ordenar y paginar los documentos	Ninguno	{ "format": 0, "dateDownload": "2024-02-27T08:28:36.811Z", "username": "string", "documentId": 1, "reasonId": 1, "paginationParams": { "pageNumber": 1, "pageSize": 1 }, "sortBy": 0, "sortOrder": 0, "dateComparator": 0 }	application/json	200 OK (body), 400 Bad Request Error (body)
6	Downloads	POST	/download	Permite descargar documentos	Ninguno	{ "format": 0, "username": "string", "documentId": 1, "reasonId": 1 }	application/json	200 OK (body), 400 Bad Request Error (body), 404 Not Found (body) Error
7	Nomenclators	GET	/nomenclator/{type}	Obtiene los nomencladores de un tipo dado	type: tipo de nomenclador	Ninguno	Ninguno	200 OK (body), 400 Bad Request Error (body)
8	Nomenclators	GET	/nomenclator/{type}/{id}	Obtiene el nomenclador de un id y tipo específico	type: tipo de nomenclador; id: Id del nomenclador	Ninguno	Ninguno	200 OK (body), 400 Bad Request Error (body), 404 Not Found (body) Error
9	Nomenclators	POST	/nomenclator	Crea un nuevo nomenclador	Ninguno	{ "name": "string", "type": 0 }	application/json	201 Created, 400 Bad Request Error (body)
10	Nomenclators	PATCH	/nomenclator/{id}	Edita un nomenclador	id: Id del nomenclador	{ "name": "string" }	application/json	200 OK, 400 Bad Request Error (body), 404 Not Found (body) Error

2.4 Desarrollo de la API

Luego de la etapa de diseño, se comenzó a implementar la solución *backend*, siguiendo las pautas trazadas. La implementación de los modelos de datos principales, se encuentra documentada en el diagrama de clases UML de la **Figura 12**. La implementación de este diagrama de clases tiene variaciones en la solución debido a las características inherentes del ecosistema empleado que permiten una mayor eficiencia en el producto terminado.

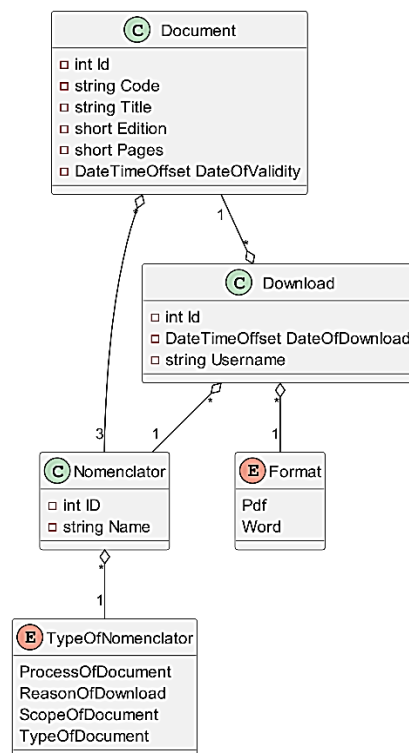


Figura 12: Diagrama de clases UML de **AicaDocs** (Fuente: Elaboración propia)

Para el control de versiones y almacenamiento en la nube del código se creó un repositorio en Github (Ver **Figura 13**) al cual se puede acceder a través del siguiente enlace: <https://github.com/EduardoProfe666/AicaDocsApi>.

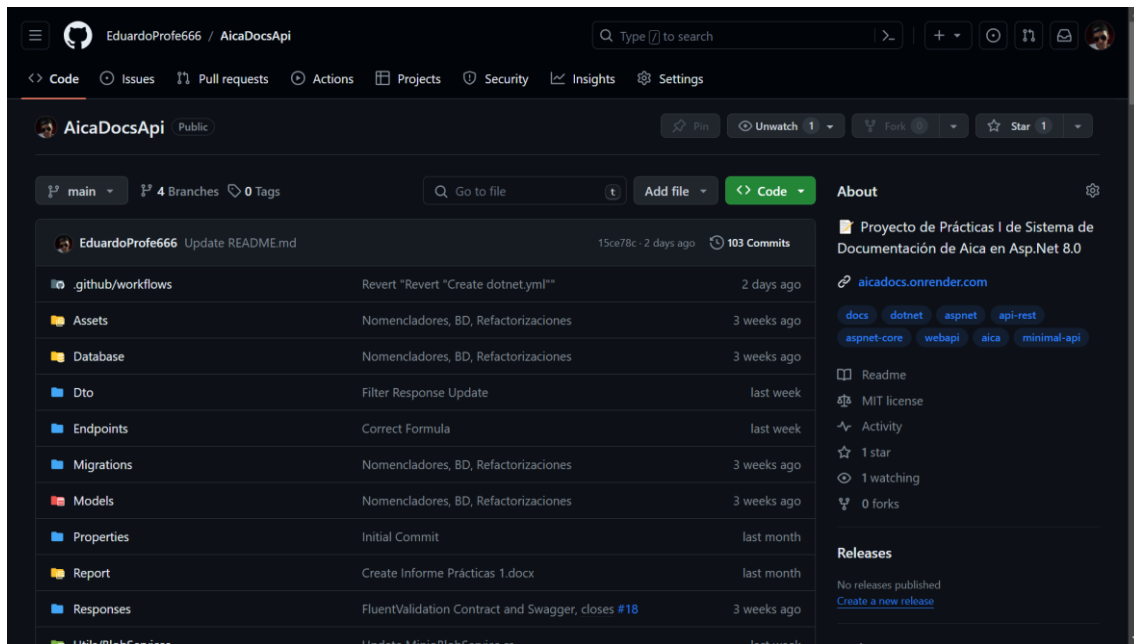


Figura 13: Repositorio en Github de **AicaDocs** (Fuente: Github)

Algunos de los aspectos y características novedosas a destacar en la implementación resultante son:

- Se emplearon satisfactoriamente las herramientas clásicas del *framework* y el ecosistema empleado, además de que se implementó una considerable cantidad de patrones de diseño, lo cual permitió lograr una mejor y veloz experiencia de desarrollo.
- Gracias al paquete `MicroElements.Swashbuckle.FluentValidation`, se logró agregar una segura capa de validaciones a la aplicación, aumentando su confiabilidad y su usabilidad.
- Se utilizó el paquete `SautinSoft.PdfFocus` para poder conocer de forma automática la cantidad de páginas de los documentos.
- Se generó documentación automática y una plataforma de prueba de la API gracias a la herramienta Swagger UI (Ver **Figura 14**), la cual se implementó satisfactoriamente.
- Se documentaron exhaustivamente los *endpoints*, de manera tal que permitiera el consumo de los mismos, sin necesidad de conocer el funcionamiento interno.

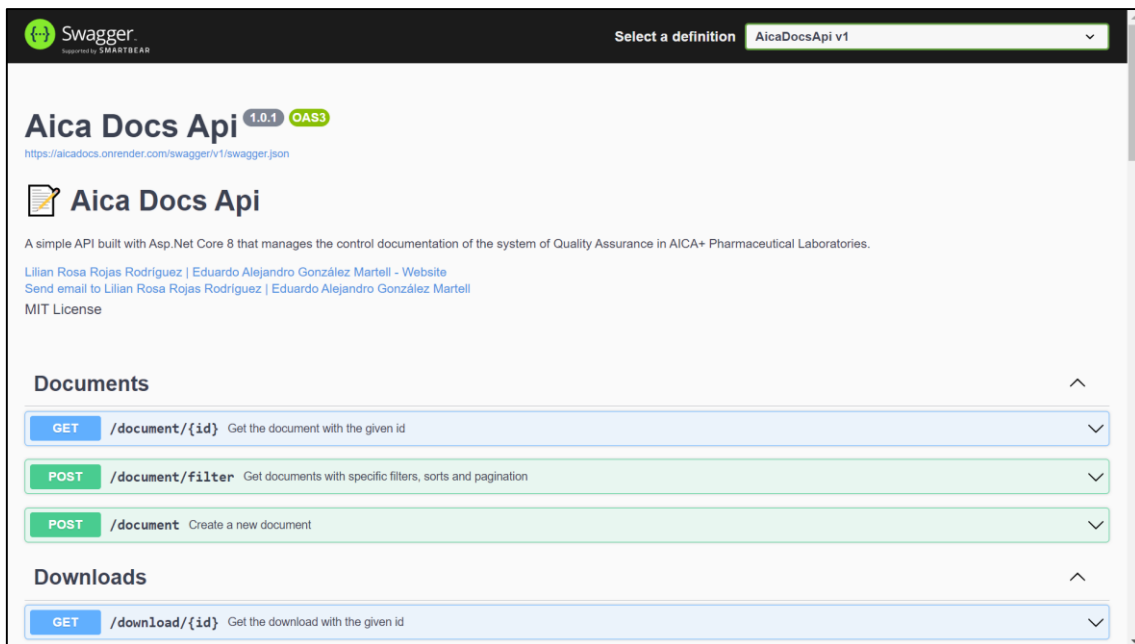


Figura 14: Implementación de Swagger UI en **AicaDocs**.

2.5 Despliegue de los servicios de la API

Para poder probar el correcto funcionamiento de la API desarrollada en entorno de producción se decidió desplegar los servicios que conforman **AicaDocs**: la API, la base de datos y el sistema de almacenamiento de objetos (Ver **Figura 15**).

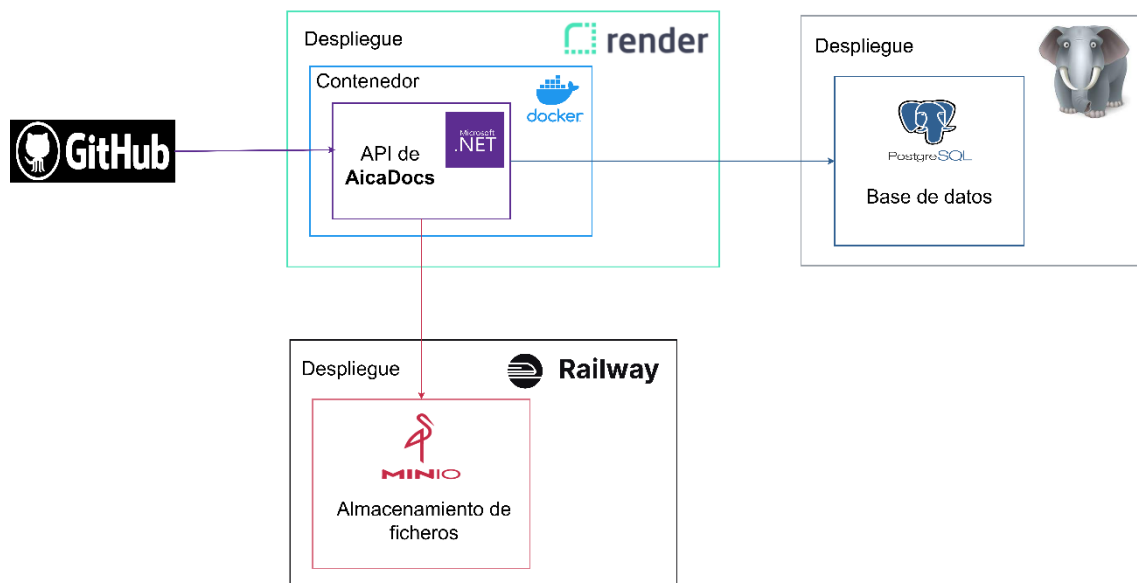


Figura 15: Infraestructura de despliegue de **AicaDocs** (Fuente: Elaboración propia)

Para desplegar la API, se decidió utilizar los servicios de despliegue y alojamiento web que ofrece <https://render.com>, debido a que ofrece un sistema sencillo para desplegar contenedores *dockerizados* y exponer los servicios en una url pública; además de que otorga un plan de uso totalmente gratuito, útil en el escenario actual de los desarrolladores del presente proyecto. Como se aprecia en la **Figura 16** se desplegó exitosamente la API de **AicaDocs**. La url pública para poder acceder al despliegue es <https://aicadocs.onrender.com>.

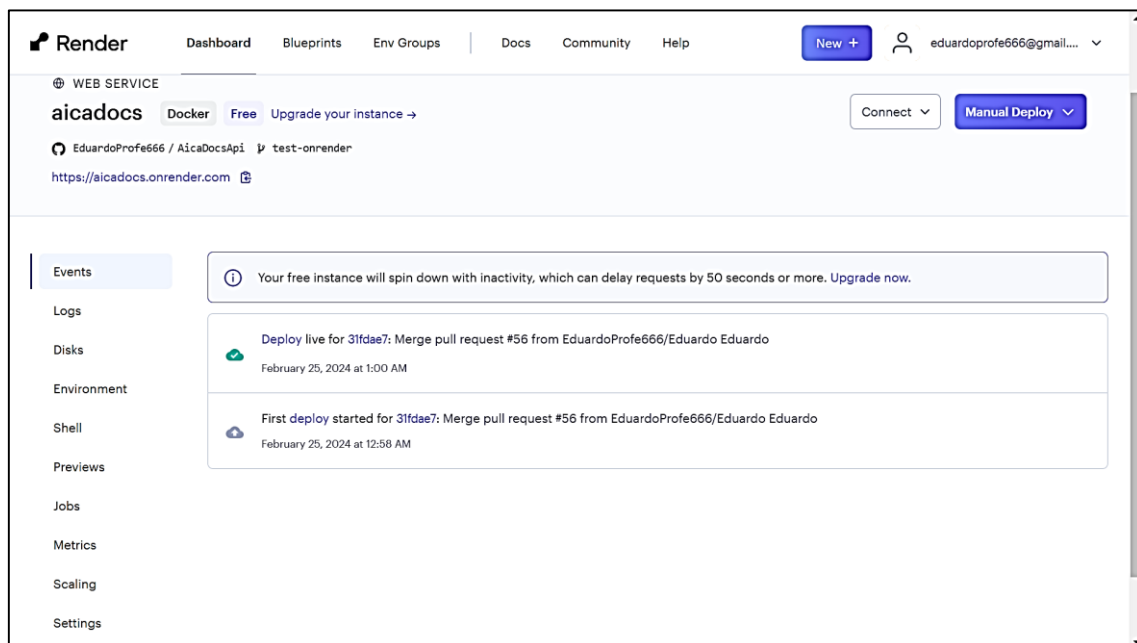


Figura 16: *Dashboard* de despliegue de la API de **AicaDocs** en Render
(Fuente: Render)

Para desplegar la base de datos de PostgreSQL, se decidió utilizar los servicios de despliegue y alojamiento web que ofrece <https://www.elephantsql.com/>, debido a que ofrece un sistema sencillo para desplegar la base de datos y exponer los servicios para su consumo por la API; además de que otorga un plan de uso totalmente gratuito, útil en el escenario actual de los desarrolladores del presente proyecto. Como se aprecia en la **Figura 17** se desplegó exitosamente la base de datos de **AicaDocs**.

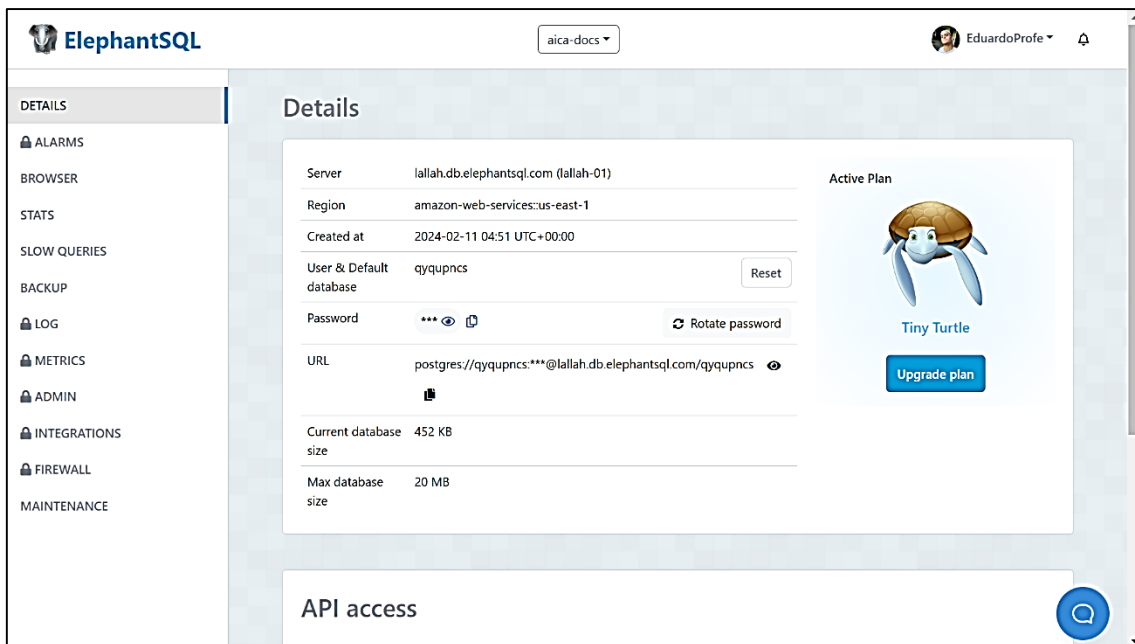


Figura 17: *Dashboard* de despliegue de la base de datos de PostgreSQL de **AicaDocs** en ElephantSQL (Fuente: ElephantSQL)

Para desplegar el sistema de almacenamiento de objetos de MinIO, se decidió utilizar los servicios de despliegue y alojamiento web que ofrece <https://railway.app/>, debido a que ofrece un sistema sencillo para desplegar sistemas complejos como MinIO y exponer los servicios para su consumo por la API; además de que otorga un plan de uso totalmente gratuito, útil en el escenario actual de los desarrolladores del presente proyecto. Como se aprecia en la **Figura 18** se desplegó exitosamente la base de datos de **AicaDocs**. La url pública para poder acceder al despliegue es <https://aicadocs-s3.up.railway.app/>. Es necesario destacar que para consumir el despliegue de los ficheros desde Cuba, se hace necesario un **VPN** para poder acceder completamente a estos servicios.

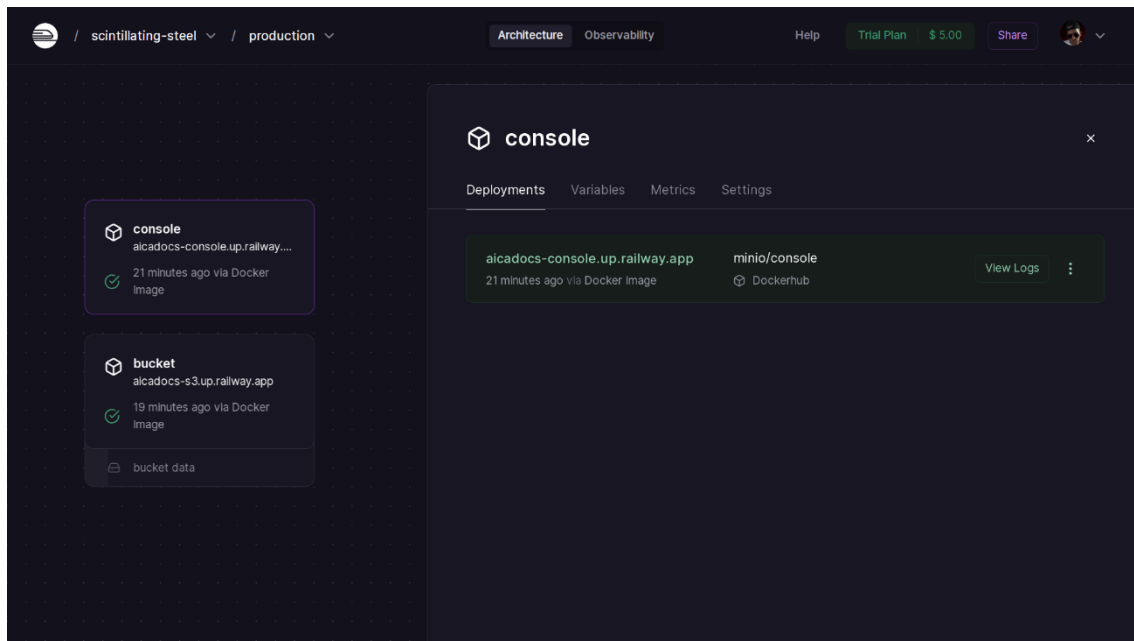


Figura 18: *Dashboard* de despliegue del almacenamiento de objetos MinIO de **AicaDocs** en Railway (Fuente: Railway)

2.6 Conclusiones Parciales

En este capítulo se presentó el diseño, desarrollo y el despliegue de la solución propuesta. Se realizó un análisis de los requisitos funcionales y no funcionales para poder luego diseñar todas las partes fundamentales del sistema a desarrollar. Luego se desarrolló el sistema **AicaDocs** a través de las pautas de diseño anteriormente ideadas. Para concluir se desplegaron los servicios fundamentales de la API para poder comprobar el correcto funcionamiento de la misma en entornos de producción y no solo de desarrollo, obteniendo resultados positivos.

Capítulo 3. Pruebas

El presente capítulo presenta el diseño y resultado de la ejecución de las pruebas al sistema desarrollado en diferentes escenarios. Las secciones primera y segunda denotan el diseño de los casos de prueba de caja negra y el resultado de la ejecución de los mismos tanto en entornos de producción como de desarrollo. La tercera sección presenta las pruebas de integración automatizadas ejecutadas tanto local como en Github.

3.1 Diseño de casos de pruebas

3.1.1 Conceptos fundamentales

Las pruebas de caja negra son una técnica de análisis de la funcionalidad de un sistema sin tener en cuenta la estructura interna del código, las rutas internas ni la información referente a la implementación. Este enfoque se centra en la entrada y salida del software, basándose en sus especificaciones y requisitos, para validar que el software o aplicación cumple con los objetivos funcionales establecidos [32].

El uso de pruebas de caja negra es esencial por varias razones [32]:

- **Independencia del lenguaje y la plataforma:** A diferencia de las pruebas de caja blanca, que requieren conocimiento específico del lenguaje de programación y la plataforma en la que se ejecuta el *software*, las pruebas de caja negra pueden realizarse de manera independiente. Esto facilita su aplicación en una variedad de sistemas y tecnologías sin necesidad de conocimientos técnicos específicos del *software*.
- **Enfoque en los requisitos del usuario:** Estas pruebas se centran en la experiencia del usuario final, validando que el sistema funcione según lo esperado desde la perspectiva del usuario. Esto es crucial para asegurar que el *software* cumpla con los requisitos definidos y proporcione una experiencia de usuario satisfactoria.
- **Eficiencia en la detección de errores:** Al centrarse en las entradas y salidas del sistema, las pruebas de caja negra pueden ser eficientes en la identificación de errores que no afecten directamente a la estructura

interna del código, como errores en la lógica de negocio o en la interacción con el usuario.

- **Flexibilidad y adaptabilidad:** Dado que no dependen de la estructura interna del *software*, las pruebas de caja negra pueden ser adaptadas a diferentes contextos y requisitos sin necesidad de cambios significativos en el proceso de prueba. Esto las hace una opción versátil para equipos de desarrollo que buscan una metodología de prueba flexible y adaptable.
- **Seguridad y cumplimiento:** Las pruebas de caja negra son fundamentales para asegurar que el *software* cumple con las normativas de seguridad y cumplimiento, como las regulaciones de protección de datos personales. Al enfocarse en la funcionalidad externa del sistema, estas pruebas pueden ayudar a identificar vulnerabilidades que podrían ser explotadas por actores malintencionados.

Las pruebas de caja negra son una técnica valiosa en el proceso de desarrollo de *software* debido a su enfoque en la funcionalidad del sistema desde la perspectiva del usuario, su capacidad para realizarse de manera independiente del lenguaje y la plataforma, y su eficacia en la identificación de errores que no afectan directamente la estructura interna del código [32].

3.1.2 Casos de pruebas diseñados

A continuación se presentan los diseños de los 47 casos de pruebas para los 10 *endpoints* de la solución **AicaDocs** desarrollada. Los juegos de datos empleados se encuentran en las tablas **Tabla 3**, **Tabla 4** y **Tabla 5**.

Tabla 3: Juego de datos para Documentos (Fuente: Elaboración propia)

Id	Code	Title	Edition	Pages	DateOfValidity	TypeId	ProcessId	Scopeld
1	Prueba	Prueba 1	1	1	18/2/24 3.40 AM	14	30	1
2	Prueba	Prueba 2	2	1	18/2/24 3.40 AM	14	30	1
3	Prueba	Prueba 3	3	1	18/2/24 3.40 AM	14	30	1
6	Test	Test 3	3	1	18/2/24 3.40 AM	16	31	2
5	Test	Test 2	2	1	18/2/24 3.40 AM	16	31	2
4	Test	Test 1	1	1	18/2/24 3.40 AM	16	31	2
7	Prueba P	Prueba P	1	1	18/2/24 3.40 AM	18	33	4
8	Prueba P	Prueba P	2	1	18/2/24 3.40 AM	18	33	4
9	Prueba P	Prueba P	3	1	18/2/24 3.40 AM	18	33	4
10	Test P	Test P	1	1	18/2/24 3.40 AM	19	34	5
11	prueba11	Prueba11	1	3	18/2/24 5.46 AM	28	41	1

Tabla 4: Juego de datos para Descargas (Fuente: Elaboración propia)

Id	DateOfDownload	Format	Username	DocumentId	ReasonId
1	2024-02-18 04:12:32.105265+00	0	Usuario de Prueba	4	9
2	2024-02-18 04:15:43.372264+00	1	Usuario de Prueba	1	10
3	2024-02-18 04:15:49.329132+00	0	Usuario de Prueba	1	10
4	2024-02-18 04:15:57.564983+00	1	Usuario de Prueba	4	11
5	2024-02-18 04:16:11.553938+00	0	Usuario de Prueba	8	10
6	2024-02-18 04:16:19.994539+00	1	Usuario de Prueba	6	10
7	2024-02-18 04:16:28.782679+00	0	Usuario de Prueba	5	10
8	2024-02-18 04:16:41.476681+00	1	Usuario de Prueba	6	10
9	2024-02-18 04:17:34.240971+00	0	Usuario de Prueba	5	11
10	2024-02-18 04:17:40.115523+00	1	Usuario de Prueba	5	11
11	2024-02-18 08:28:26.28195+00	0	lilyrosa	1	13

Tabla 5: Juego de datos para Nomencladores (Fuente: Elaboración propia)

Id	Name	Type
1	Rector	2
2	Aica UEB	2
3	Citox	2
4	Julio Trigo	2
5	Liorad	2
6	SH+	2
7	UDI	2
8	Distribución Impresa 1	1
9	Envío a personal externo autorizado	1
10	Elaboración de CTD	1
11	Solicitud del CECMED	1
12	Revisión por el área de documentación	1
13	Revisión del área de proceso	1
14	Procedimiento Normalizado de Operaciones	3
16	Registro	3
17	Instructiva	3
18	Especificación de Calidad	3
19	Plan Máster File	3
20	Planes	3

21	Ficha de proceso	3
22	Expediente maestro	3
23	Política	3
24	Programa	3
25	Manual	3
26	Manual de equipo	3
27	Plano	3
28	Documentación Técnica	3
29	Ficha Datos de Seguridad	3
30	Gestión del Capital humano	0
31	Gestión de Servicios generales	0
32	Gestión de Ingeniería y mantenimiento	0
33	Gestión de la Dirección	0
34	Gestión de Comercial y de negocios	0
35	Gestión de la Inversiones	0
36	Gestión de Producción	0
37	Gestión de Información y comunicación	0
38	Gestión de Logística	0
39	Gestión de Tecnologías Informáticas	0
40	Gestión Contable- financiera	0
41	Gestión de Documentación	0
42	Gestión de Medición, análisis y mejora	0
43	Gestión de Diseño, desarrollo e investigación	0
44	Gestión de Liberación de lotes	0

El *endpoint* GET con dirección /document/{id} tiene la función de devolver un documento dado su identificador. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 3**, se tuvieron en cuenta los escenarios:

- Obtención del documento dado un identificador válido, para el que se esperaba un json con la información del documento y la respuesta OK (200).
- Obtención de un documento dado un identificador no existente, para el que se esperaba la notificación del error y la respuesta Not Found (404).

El *endpoint* POST con dirección /document/filter tiene la función de filtrar los documentos de acuerdo a la información que contengan sus parámetros. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 4**, se tuvieron en cuenta los escenarios:

- Obtener los documentos dados todos los parámetros correctos, para el que se esperaba un json con la información de todos los documentos que cumplieran con las condiciones impuestas y la respuesta OK (200).
- Obtener los documentos dados todos los parámetros en null, para el que se esperaba un json con la información de todos los documentos que cumplieran con las condiciones impuestas y la respuesta OK (200).
- Obtener los documentos dados un código y un título válidos y el resto de los parámetros en null, para el que se esperaba un json con la información de todos los documentos que cumplieran con las condiciones impuestas y la respuesta OK (200).
- Obtener los documentos dados un código y un título no válidos y el resto de los parámetros en null, para el que se esperaba la notificación de los errores cometidos y la respuesta Bad Request (400).
- Obtener los documentos dada la edición correcta, para el que se esperaba un json con la información de todos los documentos que cumplieran con las condiciones impuestas y la respuesta OK (200).

- Obtener los documentos dada la edición y las páginas con valores no válidos, para el que se esperaba la notificación de los errores cometidos y la respuesta Bad Request (400).
- Obtener los documentos dados el identificador del tipo, proceso y alcance no válidos, para el que se esperaba la notificación de los errores cometidos y la respuesta Bad Request (400).
- Obtener los documentos dada la paginación no válida, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).
- Obtener los documentos dados la fecha y el comparador de fecha correctos, para el que se esperaba un json con la información de todos los documentos que cumplieran con las condiciones impuestas y la respuesta OK (200).
- Obtener los documentos dado que la edición, páginas, identificador de alcance, número de páginas, tamaño de páginas y el orden presentan valores incorrectos, para el que se esperaba la notificación de los errores cometidos y la respuesta Bad Request (400).

El *endpoint* POST con dirección /document tiene la función de crear un nuevo documento con todos los parámetros necesarios. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 5**, se tuvieron en cuenta los escenarios:

- Crear un documento con éxito, puesto que todos los datos de entrada contenían valores correctos, para el que se esperaba la respuesta Created (201).
- Crear un documento con un título no válido, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).
- Crear un documento con una edición no válida, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).
- Crear un documento con un código null, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).

- Crear un documento con el identificador del tipo y del proceso no válidos, para para el que se esperaba la notificación de los errores cometidos y la respuesta Bad Request (400).
- Crear un documento dado que el documento que debía tener extensión .pdf tuviera extensión .docx

El *endpoint* GET con dirección /download/{id} tiene la función de devolver una descarga dado su identificador. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 6**, se tuvieron en cuenta los escenarios:

- Obtención de la descarga dado un identificador válido, para el que se esperaba un json con la información del documento y la respuesta OK (200).
- obtención de la descarga dado un identificador no existente, para el que se esperaba la notificación del error y la respuesta Not Found (404).

El *endpoint* POST con dirección /download/filter tiene la función de filtrar las descargas de acuerdo a la información que contengan sus parámetros. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 7**, se tuvieron en cuenta los escenarios:

- Obtener las descargas dados todos los parámetros correctos, para el que se esperaba un json con la información de todas las descargas que cumplieran con las condiciones impuestas y la respuesta OK (200).
- Obtener las descargas dados la fecha y el comparador de fecha correctos, para el que se esperaba un json con la información de todas las descargas que cumplieran con las condiciones impuestas y la respuesta OK (200).
- Obtener las descargas dados todos los parámetros en null, para el que se esperaba un json con la información de todas las descargas que cumplieran con las condiciones impuestas y la respuesta OK (200).
- Obtener las descargas dados el identificador del documento y la razón correctos, para el que se esperaba un json con la información de todas las descargas que cumplieran con las condiciones impuestas y la respuesta OK (200).

- Obtener las descargas dados el identificador del documento y la razón no válidos, para el que se esperaba la notificación de los errores cometidos y la respuesta Bad Request (400).
- Obtener las descargas dado el formato correcto, para el que se esperaba un json con la información de todas las descargas que cumplieran con las condiciones impuestas y la respuesta OK (200).
- Obtener las descargas dado el formato no válido, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).
- Obtener las descargas dado que el formato, el identificador de la razón y el comparador de fechas no son válidos, para el que se esperaba la notificación de los errores cometidos y la respuesta Bad Request (400).
- Obtener las descargas dada la paginación no válida, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).

El *endpoint* POST con dirección /download tiene la función de realizar la descarga de un documento dado un formato .pdf o .docx. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 8**, se tuvieron en cuenta los escenarios:

- Descargar el documento con éxito puesto que todos los parámetros son correctos, para el que se esperaba un json con la url para descargar el documento y la respuesta OK (200).
- Descargar un documento dado el formato no válido, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).
- Descargar un documento dado un usuario null, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).
- Descargar un documento dado el identificador del documento no existente, para el que se esperaba la notificación del error y la respuesta NotFound (404).
- Descargar un documento dada una razón no válida, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).

El *endpoint* GET con dirección /nomenclator/{type} tiene la función de devolver los nomencladores que pertenecen al tipo dado. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 9**, se tuvieron en cuenta los escenarios:

- Obtener los nomencladores dado un tipo correcto, para el que se esperaba un json con la información de todos los nomencladores que cumplieran con la condición impuesta y la respuesta OK (200).
- Obtener los nomencladores dado un tipo no válido, para el que se esperaba la notificación del error y la respuesta Not Found (404).

El *endpoint* GET con dirección /nomenclator/{type}/{id} tiene la función de devolver el nomenclador que pertenece al tipo e identificador dados. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 10**, se tuvieron en cuenta los escenarios:

- Obtener el nomenclador dados el identificador y el tipo correctos, para el que se esperaba un json con la información del nomenclador que cumpliera con la condición impuesta y la respuesta OK (200).
- Obtener el nomenclador dados el identificador incorrecto y el tipo correcto, para el que se esperaba la notificación del error y la respuesta Not Found (404).
- Obtener el nomenclador dados el identificador correcto y el tipo incorrecto, para el que se esperaba la notificación del error y la respuesta Not Found (404).

El *endpoint* POST con dirección /nomenclator tiene la función de crear un nomenclador con los parámetros necesarios. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 11**, se tuvieron en cuenta los escenarios:

- Crear un nomenclador con éxito puesto que todos los parámetros contenían valores correctos, para el que se esperaba la respuesta Created (201).
- Crear un nomenclador con un nombre no válido, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).

- Crear un nomenclador con un tipo no válido, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).
- Crear un nomenclador con un nombre y un tipo no válidos, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).

El *endpoint* PATCH con dirección `/nomenclador/{id}` tiene la función de actualizar el nombre un nomenclador. Para el diseño de los casos de prueba de este, que se pueden observar con más detalles en el **Anexo 12**, se tuvieron en cuenta los escenarios:

- Actualizar un nomenclador con éxito puesto que los datos son válidos, para el que se esperaba la respuesta OK (200).
- Actualizar un nomenclador dado un identificador no válido, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).
- Actualizar un nomenclador dado un identificador no existente, para el que se esperaba la notificación del error y la respuesta Not Found (404).
- Actualizar un nomenclador dado un identificador correcto y un nombre no válido, para el que se esperaba la notificación del error cometido y la respuesta Bad Request (400).

3.2 Resultado de las pruebas

Las pruebas diseñadas fueron ejecutadas con resultados satisfactorios tanto en entornos de desarrollo (pruebas locales) como en entornos de producción (pruebas al despliegue). En ambos casos se empleó principalmente la plataforma de pruebas que ofrece la herramienta Swagger UI, disponible gracias a la documentación generada de forma automática en el *software* desarrollado. En aquellos casos donde no era posible llevar a cabo la prueba diseñada por cuestiones de validaciones requeridas en esta herramienta se empleó la extensión JetClient para los IDEs de JetBrains, la cual no presenta estas limitaciones.

3.3 Pruebas de integración automatizadas

3.3.1 Conceptos fundamentales

Las pruebas de integración son un tipo de pruebas de *software* que evalúan la forma en que interactúan y operan varios módulos de aplicaciones de *software* de forma cohesiva. El sistema se divide en componentes conocidos como módulos o unidades, cada uno responsable de una tarea específica. Cuando estos componentes se combinan para desarrollar todo el sistema de *software*, se examinan cuidadosamente las conexiones entre cada módulo para descubrir cualquier problema potencial.

La automatización de las pruebas de integración se realiza a través de la aplicación de herramientas de *software* para automatizar el proceso manual de revisión y validación de un producto de *software*. Las pruebas automatizadas se pueden realizar bajo demanda o como parte de una canalización de integración continua/entrega continua (CI/CD), lo que permite a los desarrolladores obtener comentarios inmediatos sobre los cambios en el código. Para automatizar las pruebas de *software*, los desarrolladores crean una serie de pruebas utilizando un lenguaje de *scripting* o una herramienta de automatización de pruebas.

Las pruebas automatizadas introducen un nivel de eficacia que las pruebas manuales simplemente no pueden igualar. Permite la ejecución rápida de casos de prueba, reduciendo significativamente el tiempo y el esfuerzo necesario. Además, las pruebas automatizadas favorecen la reducción del tamaño del equipo de control de calidad y permiten que este se centre en funciones más delicadas.

El *mocking* es una técnica utilizada en las pruebas de *software*, donde se crea un objeto simulado o “*mock*” que imita el comportamiento de un objeto real. Los *mocks* se definen como herramientas de implementación de una interfaz que devuelve el valor solicitado. En las pruebas unitarias, un *mock* puede utilizarse como un punto de observación en las situaciones en las que se necesite una comprobación del comportamiento enfocado en prevenir la inclusión de requisitos no testeados.

El objetivo principal de los *mocks* es aislar y centrarse en el código que se está probando y eliminar las dependencias en otros objetos. Esto permite a los desarrolladores probar el código en un entorno controlado donde pueden simular diferentes escenarios y comportamientos.

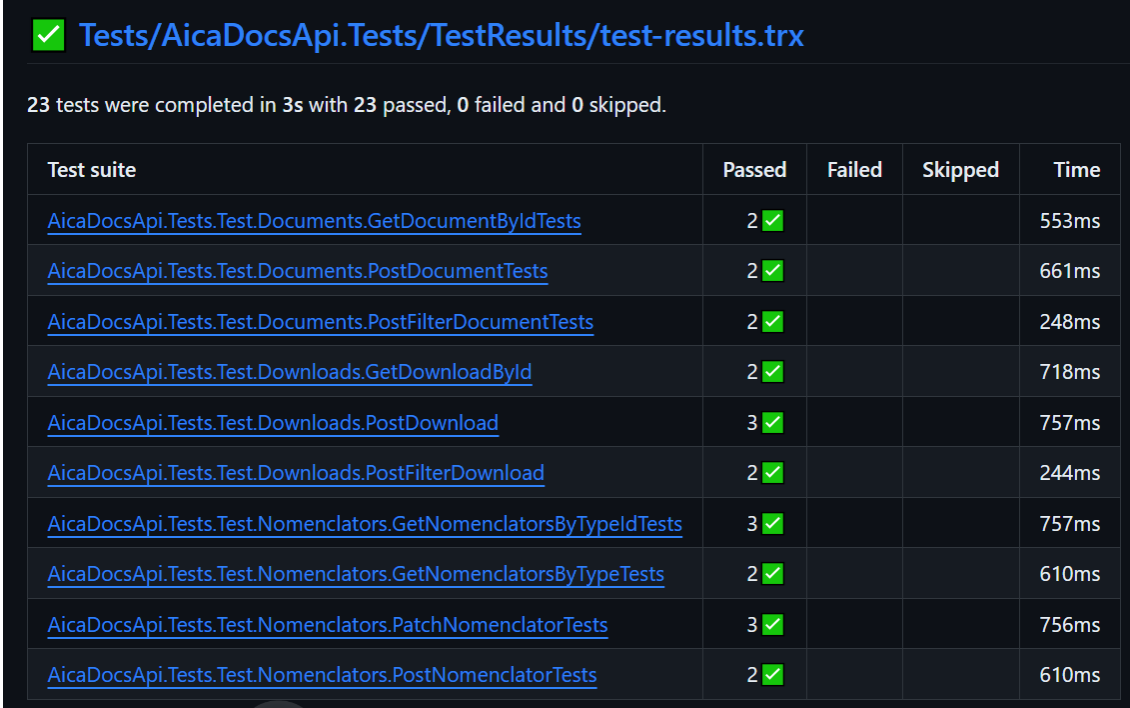
3.3.2 Diseño y resultado de las pruebas de integración

En el repositorio de la API **AicaDocs** en Github, en la rama “Lilian”, se encuentran las pruebas de integración diseñadas. Se emplearon las tecnologías y paquetes XUnit y Moq principalmente, lo que permitió realizar pruebas de integración automatizadas para múltiples casos de prueba, con datos “mockeados”. Las pruebas diseñadas se resumen en la **Tabla 6**

Tabla 6: Pruebas de integración diseñadas (Fuente: Elaboración propia).

No	Fichero de la prueba	Pruebas
1	AicaDocsApi.Tests.Test.Documents.GetDocumentByIdTests	TestOk, TestNotFound
2	AicaDocsApi.Tests.Test.Documents.PostDocumentTests	TestCreated, TestBadRequest
3	AicaDocsApi.Tests.Test.Documents.PostFilterDocumentTests	TestBadRequest, TestOk
4	AicaDocsApi.Tests.Test.Downloads.GetDownloadById	TestNotFound, TestOk
5	AicaDocsApi.Tests.Test.Downloads.PostDownload	TestBadRequest, TestNotFound, TestOk
6	AicaDocsApi.Tests.Test.Downloads.PostFilterDownload	TestBadRequest, TestOk
7	AicaDocsApi.Tests.Test.Nomenclators.GetNomenclatorsByTypeTests	TestBadRequest, TestNotFound, TestOk
8	AicaDocsApi.Tests.Test.Nomenclators.GetNomenclatorsByTypeTests	TestBadRequest, TestOk
9	AicaDocsApi.Tests.Test.Nomenclators.PatchNomenclatorTests	TestBadRequest, TestNotFound, TestOk
10	AicaDocsApi.Tests.Test.Nomenclators.PostNomenclatorTests	TestBadRequest, TestCreated

Se empleó el servicio de Github Actions para poder automatizar las pruebas diseñadas, de manera tal que se ejecutaran y diagnosticaran cada vez que hubiera algún cambio en la rama de las pruebas. En la **Figura 19** se ilustra los resultados de las últimas pruebas de integración ejecutadas.



✓ Tests/AicaDocsApi.Tests/TestResults/test-results.trx

23 tests were completed in 3s with 23 passed, 0 failed and 0 skipped.

Test suite	Passed	Failed	Skipped	Time
AicaDocsApi.Tests.Test.Documents.GetDocumentByIdTests	2 ✓			553ms
AicaDocsApi.Tests.Test.Documents.PostDocumentTests	2 ✓			661ms
AicaDocsApi.Tests.Test.Documents.PostFilterDocumentTests	2 ✓			248ms
AicaDocsApi.Tests.Test.Downloads.GetDownloadById	2 ✓			718ms
AicaDocsApi.Tests.Test.Downloads.PostDownload	3 ✓			757ms
AicaDocsApi.Tests.Test.Downloads.PostFilterDownload	2 ✓			244ms
AicaDocsApi.Tests.Test.Nomenclators.GetNomenclatorsByIdTests	3 ✓			757ms
AicaDocsApi.Tests.Test.Nomenclators.GetNomenclatorsByTypeTests	2 ✓			610ms
AicaDocsApi.Tests.Test.Nomenclators.PatchNomenclatorTests	3 ✓			756ms
AicaDocsApi.Tests.Test.Nomenclators.PostNomenclatorTests	2 ✓			610ms

Figura 19: Resultado de pruebas de integración automatizadas con Github Actions (Fuente: Github)

3.4 Conclusiones Parciales

En este capítulo se presentó el diseño y resultado de la ejecución de las pruebas al sistema desarrollado en diferentes escenarios. Tanto las pruebas de caja negra, como las pruebas de integración automatizadas diseñadas tuvieron resultados satisfactorios.

Capítulo 4. Aplicación web: AicaDocs UI

El presente capítulo presenta la modelación, desarrollo y despliegue de la aplicación web **AicaDocs UI** que permite probar el consumo de la API **AicaDocs**. La primera sección presenta un conjunto de conceptos fundamentales que sirven como fundamentación teórica para comprender los principios básicos usados en la solución propuesta. La segunda sección documenta todo el proceso de desarrollo y despliegue del recurso visual web. La última sección denota las diferentes vistas de la aplicación web creada.

4.1 Conceptos fundamentales

Una aplicación web es un *software* que se ejecuta en el navegador web [27]. Estas aplicaciones permiten a las empresas comunicarse con sus clientes y ofrecer servicios de forma remota. Algunas de las características y beneficios de las aplicaciones web incluyen [33]:

- **Accesibilidad y compatibilidad:** Las aplicaciones web son accesibles desde todos los navegadores web, a través de diferentes dispositivos y desde diferentes ubicaciones.
- **Escalabilidad:** Las aplicaciones web pueden agregar usuarios cuando sea necesario, sin requerir infraestructura adicional o *hardware* costoso.
- **Desarrollo eficiente:** El proceso de desarrollo para aplicaciones web es relativamente sencillo y rentable para las empresas.
- **Simplicidad para el usuario:** Los usuarios no tienen que descargar las aplicaciones web, lo que las hace fáciles de acceder.
- **Seguridad:** Las aplicaciones web reciben actualizaciones de *software* y seguridad de manera automática, lo que significa que siempre están actualizadas y presentan menor riesgo de sufrir brechas de seguridad.

Las aplicaciones web pueden ser de diferentes tipos, los cuales presentan sus propias características y usos. Entre los principales tipos se encuentran [34]:

- **Aplicaciones web estáticas:** Son sitios diseñados para presentar información a los visitantes sin permitir que interactúen con el contenido más allá de su lectura.

- **Aplicaciones web dinámicas:** Estos sitios están en un proceso constante de actualización que hace que haya diferentes contenidos cada vez que se visitan.
- **Aplicaciones web progresivas (PWA):** Son aplicaciones web que utilizan las últimas tecnologías web para hacer que se comporten como aplicaciones nativas.
- **Single Page Application (SPA):** Las Aplicaciones de Página Única cargan todo el contenido necesario en una sola página, utilizando el navegador como una plataforma para ejecutar el código y renderizar la interfaz de usuario. El uso de las SPA permite actualizar la página web sin necesidad de recargarla, traduciéndose en mejoras considerables de tiempo de actualización de las páginas y mejorando la experiencia de usuario (UX).
- **Server Side Render (SSR):** Las aplicaciones que utilizan el Renderizado del Lado del Servidor, también llamadas páginas isomorfas, vienen a solucionar los problemas que tenían las páginas SPA. Mejoran la indexación por parte de los robots, ya que el código HTML se sirve en una sola vez y los robots no tienen que ejecutar nada en JavaScript para poder obtenerlas. Parte del renderizado se realiza por parte del servidor, lo que mejora mucho la performance de la aplicación, y hace que también que la indexación y el SEO (*Search Engine Optimization*) se mejore, lo que se traduce en una mayor oportunidad de volumen de usuarios y/o visitantes en la página.

Razor Pages es una tecnología desarrollada por Microsoft e introducida en ASP.NET Core MVC que facilita la creación de aplicaciones web. Se utiliza para crear aplicaciones web dinámicas con Renderizado del Lado del Servidor (SSR). Permite hacer que la codificación de escenarios enfocados en páginas sea más fácil y productiva en el ecosistema de .Net. Permite programar la lógica en C# y el aspecto de vistas con HTML, CSS, JS y notaciones propias. Basado en la última versión de ASP.NET Core, Razor Pages soporta el desarrollo multiplataforma y puede ser implementado en los sistemas operativos Windows, Unix y Mac [34].

4.2 Desarrollo y despliegue de la aplicación web

Para poder probar en un escenario realista el consumo de la API de **AicaDocs**, se decidió desarrollar una aplicación web en Razor Pages, **AicaDocs UI**, utilizando el *stack* tecnológico que se denota en la **Figura 20**. Dicho *stack* se encuentra fundamentado por tutoriales oficiales de los desarrolladores del *framework* en Microsoft [23], que recomiendan seguir dicha configuración por la simplicidad de desarrollo que brinda. Además de las tecnologías denotadas en la figura se usó la biblioteca de clases CSS Bootstrap para estilar la aplicación web y el *framework* de JS JQuery para facilitar el trabajo con el lenguaje.

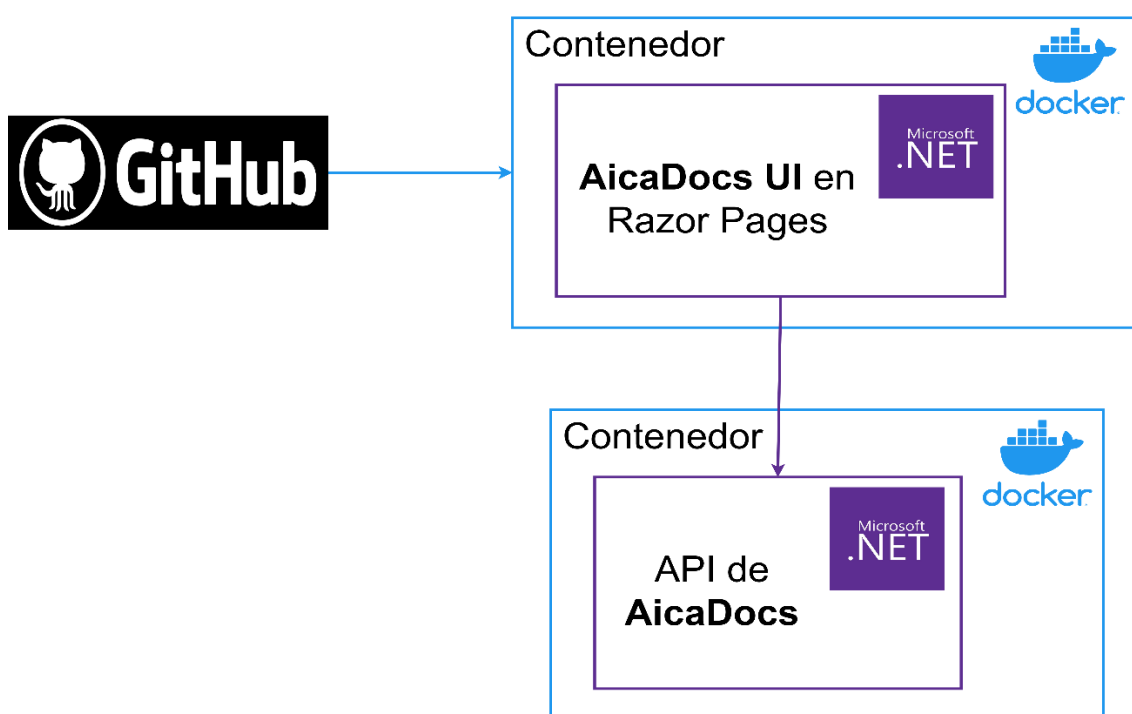


Figura 20: *Stack* tecnológico a utilizar en **AicaDocs UI** (Fuente: Elaboración propia).

Para el control de versiones y almacenamiento en la nube del código se creó un repositorio en Github (Ver **Figura 21**) al cual se puede acceder en el siguiente enlace: <https://github.com/EduardoProfe666/AicaDocsUI>.

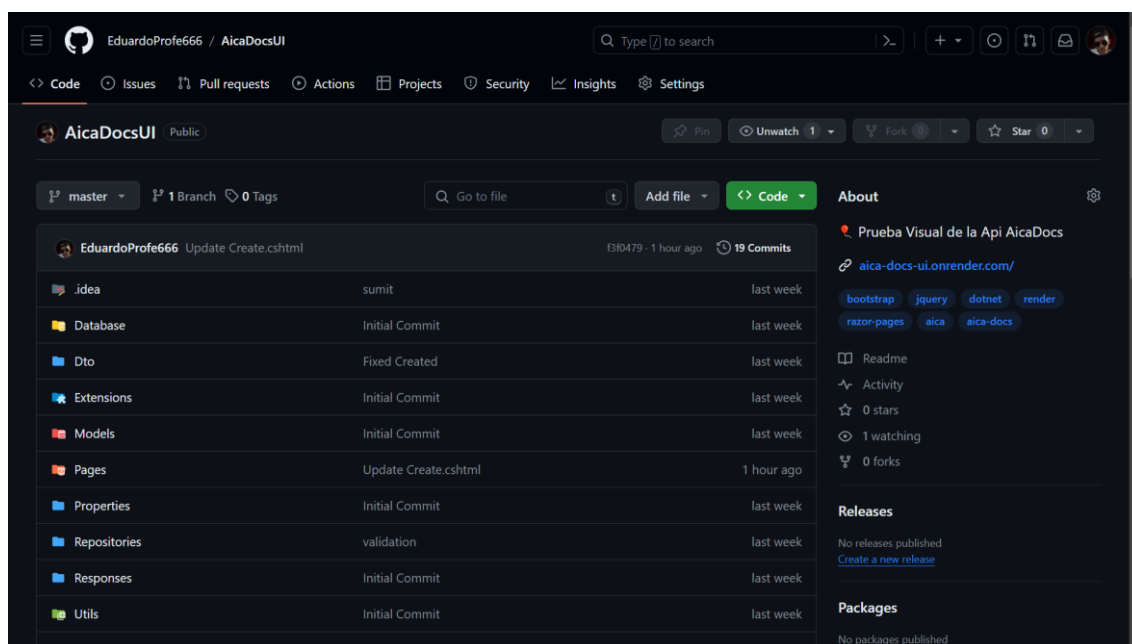


Figura 21: Repositorio en Github de AicaDocs UI (Fuente: Github).

Algunos de los aspectos y características novedosas a destacar en la implementación resultante son:

- **Patrón Repository:** Para desarrollar la aplicación web empleando las mejores prácticas y patrones de diseño, se implementó en la solución el patrón Repository. En el directorio Repositories se encuentran las principales implementaciones de este método. Se crearon capas de servicios para cada una de las secciones de la API, para otorgar mayor facilidad a la hora de consumir la API **AicaDocs**.
- **Validaciones a nivel de interfaz:** Se implementaron las mismas validaciones a nivel de interfaz que se encontraban en la API para poder mejorar la usabilidad de la aplicación web.
- **Bootstrap y JQuery:** Se empleó la biblioteca de clases CSS Bootstrap para aplicar estilos a la aplicación web, y el *framework* JQuery para facilitar el trabajo con JS.

Luego de probar satisfactoriamente la aplicación web de forma local, consumiendo el despliegue en Render de la API **AicaDocs**, se decidió desplegar también **AicaDocs UI**, para comprobar el correcto funcionamiento de la solución visual web en entorno de producción. Para ello, se decidió utilizar nuevamente

los servicios de despliegue y alojamiento web que ofrece <https://render.com>, debido a que ofrece un sistema sencillo para desplegar contenedores *dockerizados* y exponer los servicios en una url pública; además de que otorga un plan de uso totalmente gratuito, útil en el escenario actual de los desarrolladores del presente proyecto. Como se aprecia en la **Figura 22** se desplegó exitosamente **AicaDocs UI**. La url pública para poder acceder al despliegue es <https://aicadocs-ui.onrender.com>.

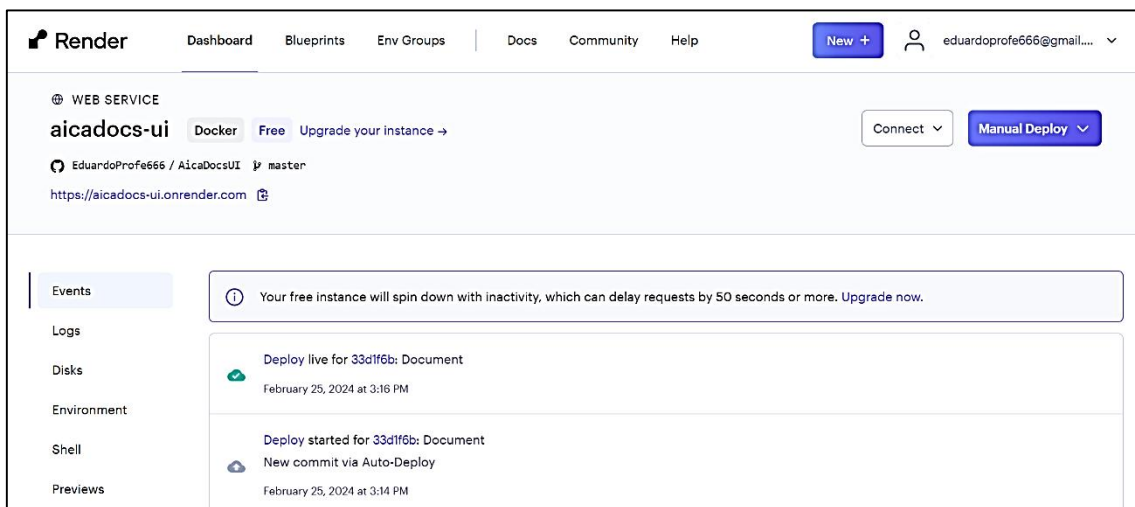


Figura 22: *Dashboard* de despliegue de **AicaDocs UI** en Render (Fuente: Render)

4.3 Vistas de la aplicación web

A continuación se documentan en las diferentes vistas de la aplicación web **AicaDocs UI**, en ambas versiones de escritorio y móvil, demostrando la correcta implementación del *responsive design*, y el enrutado resultante:

- **Inicio:** Pantalla que sirve como *landing page* para la solución web. Se encuentra disponible en la ruta “/” (Ver **Figura 23**).
- **Listado de Documentos:** Pantalla que permite listar, filtrar y paginar los documentos, así como realizar las diferentes acciones relacionadas. Se encuentra disponible en la ruta “/Documents” (Ver **Figura 24**).

- **Crear Documento:** Pantalla que permite crear documentos. Se encuentra disponible en la ruta “/Documents/Create” (Ver **Figura 25**).
- **Descargar Documento:** Pantalla que permite descargar documentos. Se encuentra disponible en la ruta “/Documents/Download” (Ver **Figura 26**).
- **Detalles de Documento:** Pantalla que permite acceder a los detalles de los documentos. Se encuentra disponible en la ruta “/Documents/Details” (Ver **Figura 27**).
- **Listado de Descargas:** Pantalla que permite listar, filtrar y paginar las descargas, así como realizar las diferentes acciones relacionadas. Se encuentra disponible en la ruta “/Downloads” (Ver **Figura 28**).
 - **Detalles de Descarga:** Pantalla que permite acceder a los detalles de las descargas. Se encuentra disponible en la ruta “/Downloads/Details” (Ver **Figura 29**).
- **Listado de Nomencladores:** Pantalla que permite listar y filtrar los nomencladores, así como realizar las diferentes acciones relacionadas. Se encuentra disponible en la ruta “/Nomenclators/” (Ver **Figura 30**).
 - **Crear Nomenclador:** Pantalla que permite crear nuevos nomencladores. Se encuentra disponible en la ruta “/Nomenclators/Create” (Ver **Figura 31**).
 - **Editar Nomenclador:** Pantalla que permite editar nomencladores. Se encuentra disponible en la ruta “/Nomenclators/Edit” (Ver **Figura 32**).

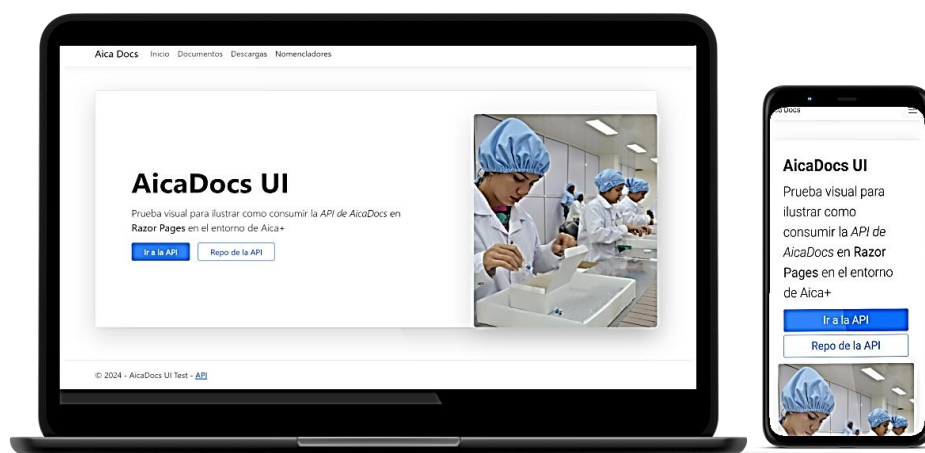


Figura 23: Pantalla de Inicio de AicaDocs UI (Fuente: AicaDocs UI)

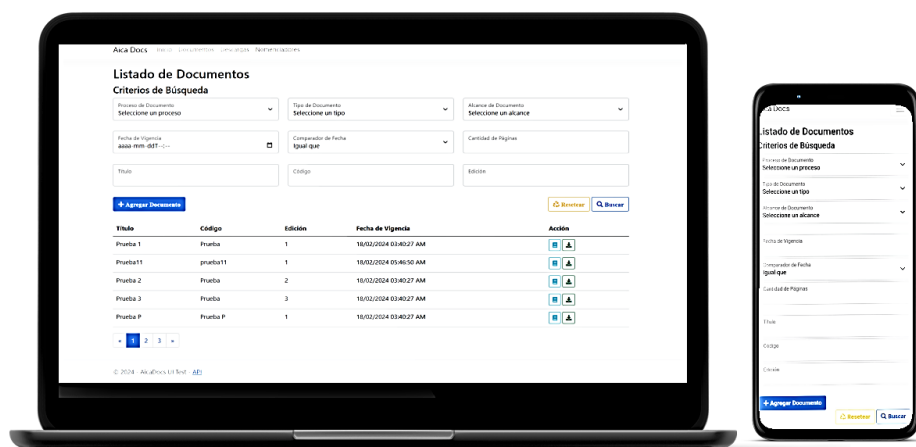


Figura 24: Pantalla de Listado de Documentos de AicaDocs UI (Fuente: AicaDocs UI)

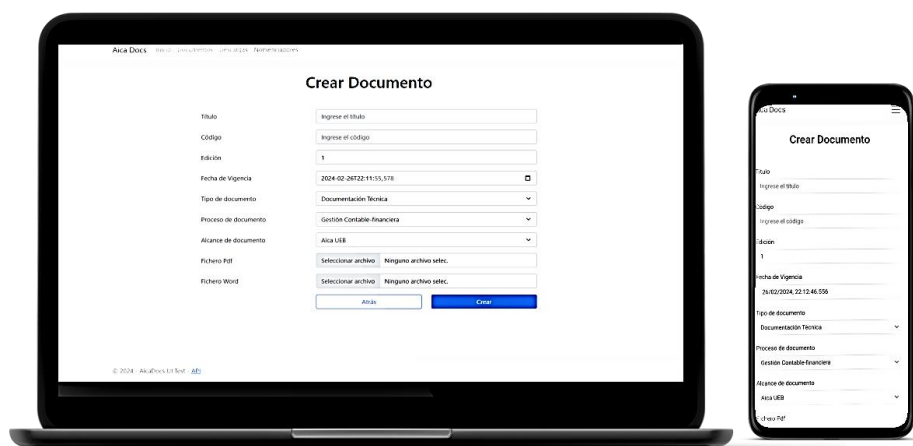


Figura 25: Pantalla de Crear Documento de AicaDocs UI (Fuente: AicaDocs UI)

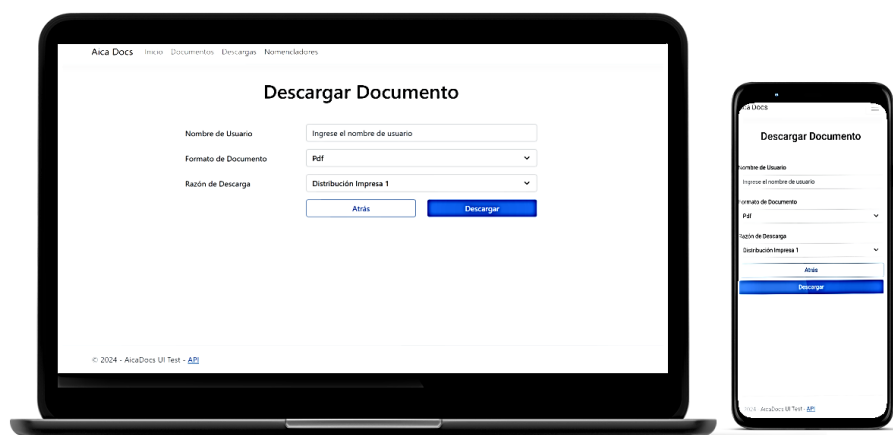


Figura 26: Pantalla de Descargar Documento de AicaDocs UI (Fuente: AicaDocs UI)

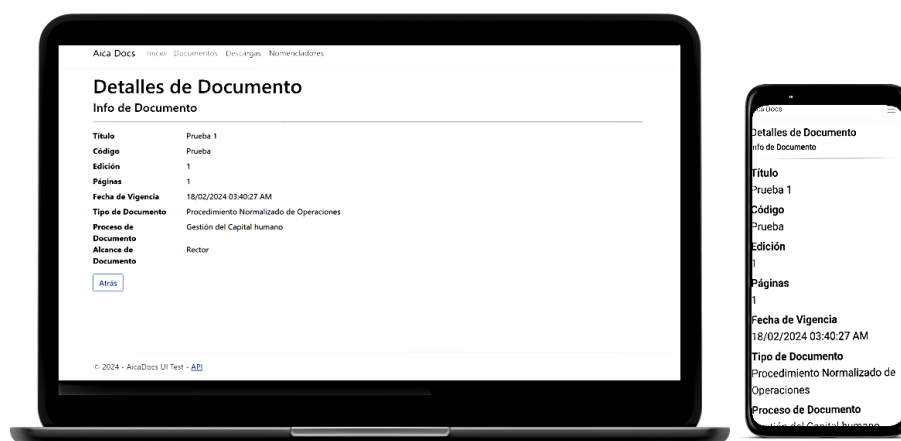


Figura 27: Pantalla de Detalles de Documento de AicaDocs UI (Fuente: AicaDocs UI)

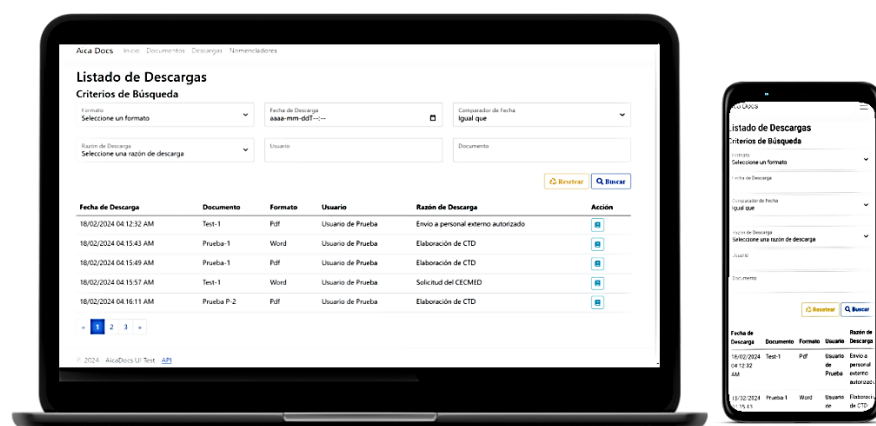


Figura 28: Pantalla de Listado de Descargas de AicaDocs UI (Fuente: AicaDocs UI)

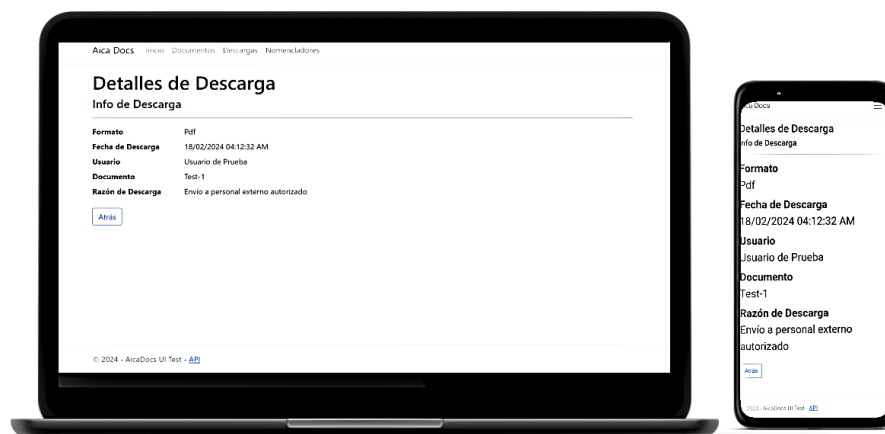


Figura 29: Pantalla de Detalles de Descarga de AicaDocs UI (Fuente: AicaDocs UI)

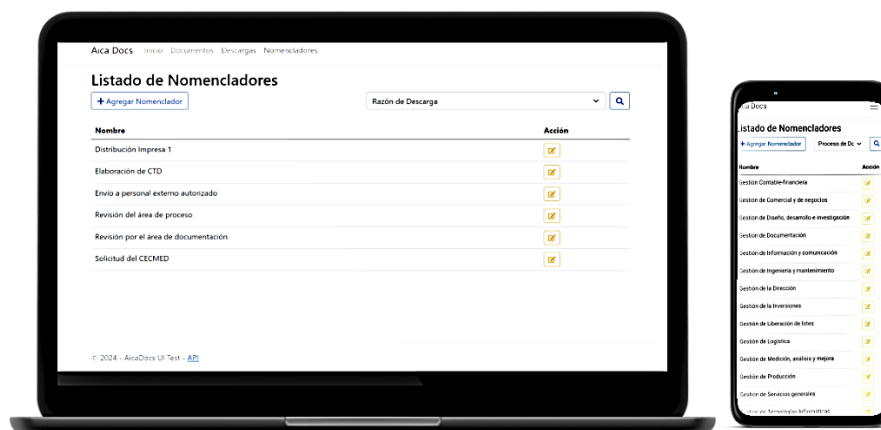


Figura 30: Pantalla de Listado de Nomencladores de AicaDocs UI (Fuente: AicaDocs UI)

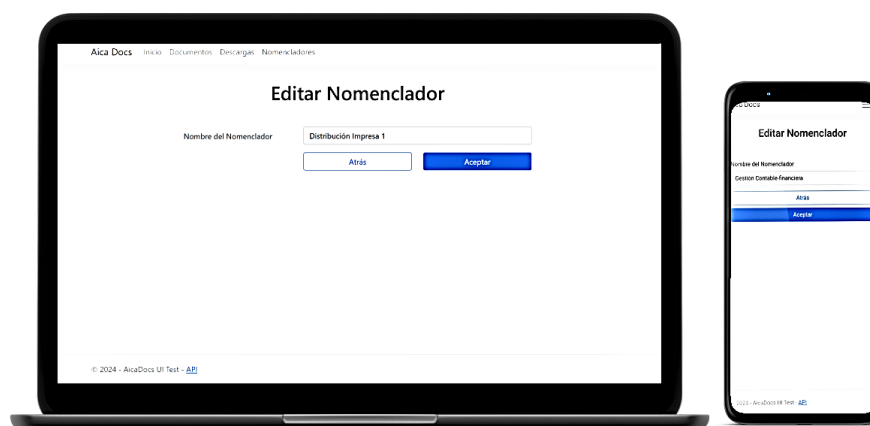


Figura 31: Pantalla de Editar Nomenclador de AicaDocs UI (Fuente: AicaDocs UI)

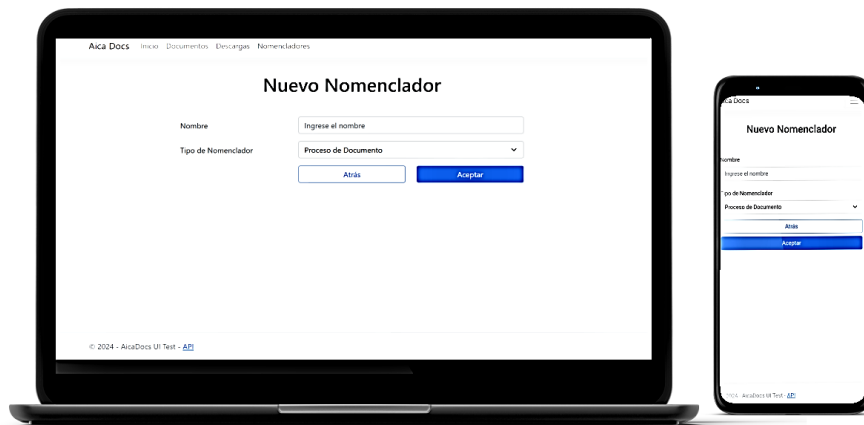


Figura 32: Pantalla de Crear Nomenclador de **AicaDocs UI** (Fuente: **AicaDocs UI**)

4.4 Conclusiones Parciales

En este capítulo se documentó todo el proceso de la creación y despliegue de la aplicación **AicaDocs UI** que permitió comprobar con éxito el consumo de la API **AicaDocs**.

Conclusiones

En el presente trabajo se presentan la investigación, diseño e implementación llevados a cabo para satisfacer el objetivo general del mismo. A partir de lo expuesto en sus capítulos, pueden ser enunciadas las siguientes conclusiones:

- Se logró la asimilación de los conceptos fundamentales para poder comprender el impacto de la industria 4.0 en las empresas.
- Se logró la asimilación del ecosistema .Net, plataforma que se empleará para desarrollar otras implementaciones de capacidades habilitantes para la transición a la industria 4.0, como son los sistemas SCADA, MES y EIS/DSS.
- El *stack* de tecnologías utilizadas para el desarrollo del sistema *backend* resultó óptimo para lograr mayor rapidez en el desarrollo y despliegue de la solución digital.
- Se probó con éxito un escenario de despliegue de los servicios de la solución digital **AicaDocs** desarrollada: la API, la base de datos y el sistema de almacenamiento de objetos.
- Se logró diseñar y ejecutar con éxito pruebas de caja negra y de integración automatizadas en diferentes entornos para poder comprobar la efectividad del sistema desarrollado.
- Se logró con eficiencia el desarrollo y despliegue de un recurso GUI web **AicaDocs UI** para probar un escenario de consumo de la API, resultando en exitosa la prueba.

Recomendaciones

Una vez finalizado el desarrollo de la investigación, se plantean los siguientes apartados a modo de recomendación:

- Mejorar el despliegue de los servicios de la API en los siguientes aspectos:
 - Desplegar los servicios de la base de datos, la gestión de objetos y el *backend* en sí, en servidores propios o en externos con un servicio profesional real, sustituyendo los despliegues de prueba actuales.
- Mejorar la implementación del recurso visual para probar el consumo de la API en los siguientes aspectos:
 - Implementar la característica de ordenamiento en las tablas que permiten filtrado.
 - Mejorar la implementación del *responsive design* para dispositivos móviles.
 - Implementar gráficos y resúmenes de datos para incrementar la funcionalidad y productividad.
- Mejorar el diseño y la implementación de las pruebas de caja negra en los siguientes aspectos:
 - Incrementar el número de pruebas diseñadas y la cantidad de casos de uso comprendidos para poder incrementar el nivel de cobertura de las pruebas.
 - Ejecutar las pruebas en otros entornos aparte de los ya usados para consolidar la validez de los resultados de las mismas.
- Mejorar las pruebas de integración automatizadas en los siguientes aspectos:
 - Uso de la sintaxis propia de la librería de pruebas XUnit y no la sintaxis general implementada.
 - Incrementar el número de pruebas para aumentar el grado de cobertura de las pruebas.
- Adoptar una arquitectura *Multi-Tenant* para poder escalar a SaaS (*Software as a Service*) como modelo de negocio.

Referencias Bibliográficas

- [1] C. Koldewey, D. Hobscheidt, C. Pierenkemper, A. Kühn, and R. Dumitrescu, "Increasing Firm Performance through Industry 4.0—A Method to Define and Reach Meaningful Goals," *Sci*, vol. 4, p. 39, 10/18 2022.
- [2] M. Piccarozzi, B. Aquilani, and C. Gatti, "Industry 4.0 in Management Studies: A Systematic Literature Review," *Sustainability*, vol. 10, p. 3821, 10/22 2018.
- [3] M. Mladineo, L. Celent, V. Milković, and I. Veza, "Current State Analysis of Croatian Manufacturing Industry with Regard to Industry 4.0/5.0," *Machines*, vol. 12, p. 87, 01/23 2024.
- [4] A. Frank, L. Dalenogare, and N. Ayala, "Industry 4.0 technologies: Implementation patterns in manufacturing companies," *International Journal of Production Economics*, vol. 210, 01/09 2019.
- [5] F. Albayrak and O. Poyrazoğlu, "A Systematic Literature Review on Lean, Industry 4.0, and Digital Factory," *Journal of the Knowledge Economy*, 12/02 2023.
- [6] R. Dumitrescu *et al.*, "acatech Maturity Index Smart Services," *acatech National Academy of Science and Engineering*, 2023.
- [7] G. Schuh, R. Anderl, R. Dimitrescu, A. Krüger, and M. ten Hompel, "acatech Industrie 4.0 Maturity Index. Managing the Digital Transformation of Companies. Update 2020," *acatech National Academy of Science and Engineering*, 2020.
- [8] G. Schuh, R. Anderl, R. Dimitrescu, A. Krüger, and M. ten Hompel, "acatech Using the Industrie 4.0 Maturity Index in Industry. Current challenges, case studies and trends," *acatech National Academy of Science and Engineering*, 2020.
- [9] J. Ahamad, "Applications of Industry 4.0 in the Pharmaceutical Sector," 2023.
- [10] J. Dawale, R. Singh, K. Gawai, R. Gimonkar, J. Pawar, and C. Ghodke, "Industry 4.0," *International Journal of Advanced Research in Science, Communication and Technology*, pp. 9-11, 10/08 2023.

- [11] A. Digalwar, R. Pandey, S. Singh, and A. Sharma, *Industry 4.0 Implementation: Evidence from Indian Industries*. 2024, pp. 23-34.
- [12] H. Kagermann and W. Wahlster, "Ten Years of Industrie 4.0," *Sci*, vol. 4, p. 26, 06/28 2022.
- [13] J. Nenadál, D. Vykydal, P. Halfarová, and E. Tylečková, "Quality 4.0 Maturity Assessment in Light of the Current Situation in the Czech Republic," *Sustainability*, vol. 14, p. 7519, 06/20 2022.
- [14] M. Flamini and M. Naldi, "Maturity of Industry 4.0: A Systematic Literature Review of Assessment Campaigns," *Journal of Open Innovation: Technology, Market, and Complexity*, vol. 8, p. 51, 03/11 2022.
- [15] H. Tariq and S. Sultan, "A Secure Interoperable API Wrapper Tunnel for Integration of GIS-Based ICS-IIoT and Digital Twins in Industry 4.0 Clouds," *WSEAS TRANSACTIONS ON COMPUTERS*, vol. 21, pp. 131-138, 04/21 2022.
- [16] J. Brandenburger *et al.*, *Quality4.0 -- Transparent product quality supervision in the age of Industry 4.0*. 2020.
- [17] P. Amrih and R. Damayanti, "Pharma 4.0 Quality Management Challenge: A Literature Review," *Sci*, 05/23 2023.
- [18] A. Giri, "Designing a SAAS Platform for Industry 4.0 Smart Manufacturing," *International Journal for Research in Applied Science and Engineering Technology*, vol. 11, pp. 2445-2451, 05/31 2023.
- [19] C. R. López Paz, P. Velázquez Borrero, A. L. Infante Abreu, and K. Escalera Fariñas, "Documentation System in the AICA+ Laboratories. Case Study of Microservices, Devops and Incidents Management," *RCTD*, vol. 4, no. 1, 2023.
- [20] Github, "Github Documentation," 2024.
- [21] AWS, "Docker," 2024.
- [22] Microsoft, "¿Por qué elegir .NET?," 2024.
- [23] Microsoft. (2024). *Official Documentation of ASP.NET 8.0*. Available: <https://learn.microsoft.com/es-es/aspnet/core/?view=aspnetcore-8.0>
- [24] TechEmpower, "Best plaintext responses per second," 2023.
- [25] A. Giretti, *Coding Clean, Reliable, and Safe REST APIs with ASP.NET Core 8: Develop Robust Minimal APIs with .NET 8*. 2023.

- [26] S. M. Khan, "Simplified ASP.NET Core Web API with Clean Architecture and Chain of Responsibility," 11/03 2023.
- [27] S. M. Khan, *API Development Using Asp.Net Core Web API A practical approach for developing the APIs in Asp.net Core*. 2023.
- [28] MinIO, "MinIO," 2024.
- [29] Postgres, "PostgreSQL," 2024.
- [30] L. Liu, M. Bahrami, J. Park, and W.-P. Chen, "Web API Search: Discover Web API and Its Endpoint with Natural Language Queries," 2020, pp. 96-113.
- [31] M. Milošević, V. Mladenovic, and U. Pešović, "Evaluation of HTTP/3 Protocol for Internet of Things and Fog Computing Scenarios," *Studies in Informatics and Control*, vol. 30, pp. 75-84, 09/30 2021.
- [32] A. Martin-Lopez, A. Arcuri, S. Segura, and A. Ruiz-Cortés, *Black-Box and White-Box Test Case Generation for RESTful APIs: Enemies or Allies?* 2021.
- [33] A. Giretti, "Basics of Clean REST APIs," 2023, pp. 91-212.
- [34] A. Freeman, "Using Razor Pages," 2022, pp. 629-661.

Anexos

Anexo 1: Anexo A de Lilian Rosa Rojas Rodríguez

Tareas	Fecha de entrega	Rol(es) que desarrolla(n) con la tarea
Reunión de inicio de la práctica	15/1/2024 (Semana 1)	-
Asimilar el contexto de industria 4.0 en el dominio de calidad 4.0 (sistema de documentación)	Semana 1	AN, AS
Asimilar las tecnologías habilitantes en los niveles gerenciales (componente backend)	Semana 2	DS, PG, GC
Implementar sistema de documentación – nomencladores (alcance, proceso, tipo de documento, motivo de descarga)	Semana 2-7	DS, PG
Implementar sistema de documentación – crear documento, listar documentos, filtrar documentos por nombre, código, alcance, proceso y tipo de documento, descargar documento.	Semana 2-7	DS, PG
Asimilar la infraestructura básica para las pruebas del backend	Semana 5-6	PB
Ejecutar y analizar las pruebas diseñadas del backend	Semana 5-6	PB
Elaborar informe de la práctica	Semana 4-7	EE
Entregar informe de la práctica al tutor	Semana 7	EE
Rectificar señalamientos del informe	Semana 7	EE
Entrega del informe final de la práctica	4/3/24 (semana 8)	EE
Defensa de la práctica	6-8/3/24 (semana 8)	Todos

Anexo 2: Anexo A de Eduardo Alejandro González Martell

Tareas	Fecha de entrega	Rol(es) que desarrolla(n) con la tarea
Reunión de inicio de la práctica	15/1/2024 (Semana 1)	-
Asimilar el contexto de industria 4.0 en el dominio de calidad 4.0 (sistema de documentación)	Semana 1	AN, AS
Asimilar las tecnologías habilitantes en los niveles gerenciales (componente backend)	Semana 2	DS, PG, GC
Implementar sistema de documentación – nomencladores (alcance, proceso, tipo de documento, motivo de descarga)	Semana 2-7	DS, PG
Implementar sistema de documentación – crear documento, listar documentos, filtrar documentos por nombre, código, alcance, proceso y tipo de documento, descargar documento.	Semana 2-7	DS, PG
Asimilar la infraestructura básica para las pruebas del backend	Semana 5-6	PB
Ejecutar y analizar las pruebas diseñadas del backend	Semana 5-6	PB
Elaborar informe de la práctica	Semana 4-7	EE
Entregar informe de la práctica al tutor	Semana 7	EE
Rectificar señalamientos del informe	Semana 7	EE
Entrega del informe final de la práctica	4/3/24 (semana 8)	EE
Defensa de la práctica	6-8/3/24 (semana 8)	Todos

Anexo 3: Diseño de prueba de caja negra para el *endpoint* “GET /document/{id}”

Historia de Usuario	HU1: Obtener un documento dado su id
Desarrollador	Lilian Rosa Rojas Rodríguez
Probador	Lilian Rosa Rojas Rodríguez
Fecha	17 de febrero de 2024

CP	Escenario	Nombre de variable de entrada	Valor	Resultados Esperados
1	Obtener documento dado un id correcto	id	1	Se muestra el código de respuesta 200 (OK) y el json <pre>{ "data": { "id": 0, "code": "string", "title": "string", "edition": 0, "pages": 0, "dateOfValidity": "2024-02-18T04:35:36.462Z", "typeId": 0, "processId": 0, "scopeId": 0 } }</pre>
2	No obtener documento por id incorrecto	id	300	Se muestra el código de respuesta 404 (Not Found) y el json <pre>{ "problemDetails": { "status": 404, "errors": { "Document Id": ["Doesn't exist a document with the given id"] } } }</pre>

				}
--	--	--	--	---

Anexo 4: Diseño de prueba de caja negra para el *endpoint* “POST /document/filter”

Historia de Usuario	HU2: Obtener los documentos con una característica determinada
Desarrollador	Lilian Rosa Rojas Rodríguez
Probador	Lilian Rosa Rojas Rodríguez
Fecha	17 de febrero de 2024

CP	Escenario	Nombre de variable de entrada	Valor	Resultados Esperados
1	Obtener los documentos dados todos los parámetros correctos	code	Prueba	Se muestra el código de respuesta 200 (OK) y el json { "data": { "data": [{ "id": 1, "code": "Prueba", "title": "Prueba 1", "edition": 1, "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeId": 14, "processId": 30, "scopeId": 1 }], } }
		title	Prueba	
		edition	1	
		pages	1	
		dateOfValidity	2024-02-19T02:30:07.642Z	
		typeId	14	
		processId	30	
		scopeId	1	
		pageNumber	1	
		pageSize	5	
		sortBy	0	
		sortOrder	0	
		dateComparator	4	

				<pre>"totalPages": 1 } }</pre>																										
2	Obtener los documentos dados todos los parámetros en null	<table><tr><td>code</td><td>null</td></tr><tr><td>title</td><td>null</td></tr><tr><td>edition</td><td>null</td></tr><tr><td>pages</td><td>null</td></tr><tr><td>dateOfValidity</td><td>null</td></tr><tr><td>typeId</td><td>null</td></tr><tr><td>processId</td><td>null</td></tr><tr><td>scopeId</td><td>null</td></tr><tr><td>pageNumber</td><td>1</td></tr><tr><td>pageSize</td><td>5</td></tr><tr><td>sortBy</td><td>0</td></tr><tr><td>sortOrder</td><td>0</td></tr><tr><td>dateComparator</td><td>4</td></tr></table>	code	null	title	null	edition	null	pages	null	dateOfValidity	null	typeId	null	processId	null	scopeId	null	pageNumber	1	pageSize	5	sortBy	0	sortOrder	0	dateComparator	4		<pre>Se muestra el código de respuesta 200 (OK) y el json { "data": { "data": [{ "id": 1, "code": "Prueba", "title": "Prueba 1", "edition": 1, "pages": 1, "dateOfValidity": "2024- 02- 18T03:40:27.429+00:00", "typeId": 14, "processId": 30, "scopeId": 1 }, { "id": 2, "code": "Prueba", "title": "Prueba 2", "edition": 2, "pages": 1, "dateOfValidity": "2024- 02- 18T03:40:27.429+00:00", "typeId": 14, "processId": 30,</pre>
code	null																													
title	null																													
edition	null																													
pages	null																													
dateOfValidity	null																													
typeId	null																													
processId	null																													
scopeId	null																													
pageNumber	1																													
pageSize	5																													
sortBy	0																													
sortOrder	0																													
dateComparator	4																													

				<pre> "scopeld": 1 }, { "id": 3, "code": "Prueba", "title": "Prueba 3", "edition": 3, "pages": 1, "dateOfValidity": "2024- 02- 18T03:40:27.429+00:00", "typeld": 14, "processId": 30, "scopeld": 1 }, { "id": 4, "code": "Test", "title": "Test 1", "edition": 1, "pages": 1, "dateOfValidity": "2024- 02- 18T03:40:27.429+00:00", "typeld": 16, "processId": 31, "scopeld": 2 }, { "id": 5, "code": "Test", </pre>
--	--	--	--	---

				<pre>"title": "Test 2", "edition": 2, "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeld": 16, "processId": 31, "scopeld": 2 }], "totalPages": 3 } }</pre>
3	Obtener los documentos dado el código y el título válidos	code	Prueba	Se muestra el código de respuesta 200 (OK) y el json <pre>{ "data": { "data": [{ "id": 1, "code": "Prueba", "title": "Prueba 1", "edition": 1, "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeld": 14, "processId": 30, "scopeld": 1 }, </pre>
title		Prueba		
edition		null		
pages		null		
dateOfValidity		null		
typeld		null		
processId		null		
scopeld		null		
pageNumber		1		
pageSize		5		
sortBy		0		
sortOrder		0		
dateComparator		4		

				<pre> { "id": 2, "code": "Prueba", "title": "Prueba 2", "edition": 2, "pages": 1, "dateOfValidity": "2024- 02- 18T03:40:27.429+00:00", "typeId": 14, "processId": 30, "scopeId": 1 }, { "id": 3, "code": "Prueba", "title": "Prueba 3", "edition": 3, "pages": 1, "dateOfValidity": "2024- 02- 18T03:40:27.429+00:00", "typeId": 14, "processId": 30, "scopeId": 1 }, { "id": 7, "code": "Prueba P", "title": "Prueba P", "edition": 1, </pre>
--	--	--	--	--

				<pre> "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeId": 18, "processId": 33, "scopeId": 4 }, { "id": 8, "code": "Prueba P", "title": "Prueba P", "edition": 2, "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeId": 18, "processId": 33, "scopeId": 4 }], "totalPages": 2 } } </pre>
4	Obtener los documentos dado el código y el título no válidos	code	Prueba Pruebapru eba prueba prueba prueba prueba prueba prueba prueba	<p>Se muestra el código de respuesta 400 (Bad Request) y el json</p> <pre> { "problemDetails": { "status": 400, "errors": { </pre>

			prueba prueba prueba prueba	"Title": ["Title must have a maximum length of 64 characters"], "Code": ["Code must have a maximum length of 64 characters"] } } } }
		title	Pruebapru eba prueba prueba prueba prueba prueba prueba prueba prueba prueba prueba prueba prueba	
		edition	null	
		pages	null	
		dateOfValidi ty	null	
		typeld	null	
		processId	30	
		scopeld	1	
		pageNumbe r	1	
		pageSize	5	
		sortBy	0	
		sortOrder	0	
		dateCompar ator	4	
5	Obtener los documentos dada la edición correcta	code	null	Se muestra el código de respuesta 200 (OK) y el json { "data": {
		title	null	
		edition	1	
		pages	null	

		dateOfValidity	null	<pre> "data": [{ "id": 1, "code": "Prueba", "title": "Prueba 1", "edition": 1, "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeId": 14, "processId": 30, "scopeId": 1 }, { "id": 4, "code": "Test", "title": "Test 1", "edition": 1, "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeId": 16, "processId": 31, "scopeId": 2 }, { "id": 7, "code": "Prueba P", "title": "Prueba P", </pre>
		typeId	null	
		processId	null	
		scopeId	null	
		pageNumber	1	
		pageSize	5	
		sortBy	0	
		sortOrder	0	
		dateComparator	4	

				<pre> "edition": 1, "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeld": 18, "processId": 33, "scopeld": 4 }, { "id": 10, "code": "Test P", "title": "Test P", "edition": 1, "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeld": 19, "processId": 34, "scopeld": 5 }, { "id": 11, "code": "prueba11", "title": "Prueba11", "edition": 1, "pages": 3, "dateOfValidity": "2024-02-18T05:46:50.312+00:00", "typeld": 28, </pre>
--	--	--	--	---

				<pre>"processId": 41, "scopeId": 1 }], "totalPages": 1 } }</pre>																										
6	Obtener los documentos dada la edición y las páginas no válidas	<table><tr><td>code</td><td>null</td></tr><tr><td>title</td><td>null</td></tr><tr><td>edition</td><td>0</td></tr><tr><td>pages</td><td>0</td></tr><tr><td>dateOfValidity</td><td>null</td></tr><tr><td>typeId</td><td>null</td></tr><tr><td>processId</td><td>null</td></tr><tr><td>scopeId</td><td>null</td></tr><tr><td>pageNumber</td><td>1</td></tr><tr><td>pageSize</td><td>5</td></tr><tr><td>sortBy</td><td>0</td></tr><tr><td>sortOrder</td><td>0</td></tr><tr><td>dateComparator</td><td>4</td></tr></table>	code	null	title	null	edition	0	pages	0	dateOfValidity	null	typeId	null	processId	null	scopeId	null	pageNumber	1	pageSize	5	sortBy	0	sortOrder	0	dateComparator	4		<pre>Se muestra el código de respuesta 400 (Bad Request) y el json { "problemDetails": { "status": 400, "errors": { "Edition": ["Edition must be greater than 0 and less than 32001"], "Pages": ["Pages must be greater than 0 and less than 32001"] } } }</pre>
code	null																													
title	null																													
edition	0																													
pages	0																													
dateOfValidity	null																													
typeId	null																													
processId	null																													
scopeId	null																													
pageNumber	1																													
pageSize	5																													
sortBy	0																													
sortOrder	0																													
dateComparator	4																													
7	Obtener los documentos dado el id del tipo, el id del proceso y el id del	<table><tr><td>code</td><td>Prueba</td></tr><tr><td>title</td><td>Prueba</td></tr><tr><td>edition</td><td>null</td></tr><tr><td>pages</td><td>null</td></tr></table>	code	Prueba	title	Prueba	edition	null	pages	null		<pre>Se muestra el código de respuesta 400 (Bad Request) y el json {</pre>																		
code	Prueba																													
title	Prueba																													
edition	null																													
pages	null																													

	alcance no válidos	dateOfValidity	null	"problemDetails": { "status": 400, "errors": { "Type id must be greater than 0" }, "ProcessId": ["Process id must be greater than 0"], "ScopeId": ["Scope id must be greater than 0"] }
		typeId	-5	
		processId	-3	
		scopeId	-8	
		pageNumber	1	
		pageSize	5	
		sortBy	0	
		sortOrder	0	
		dateComparator	4	
8	Obtener los documentos dada la paginación no válida	code	Prueba	Se muestra el código de respuesta 400 (Bad Request) y el json { "problemDetails": { "status": 400, "errors": { "PaginationParams.PageNumber": ["Page Number of Pagination Params must be greater than 0"], } } }
		title	Prueba	
		edition	null	
		pages	null	
		dateOfValidity	null	
		typeId	14	
		processId	30	
		scopeId	1	
		pageNumber	0	
		pageSize	0	
		sortBy	0	
		sortOrder	0	

		dateComparator	4	"PaginationParams.PageSize": ["Page Size of Pagination Params must be greater than 0"] } }
9	Obtener los documentos dada la fecha y el date comparator válidos	code	null	Se muestra el código de respuesta 200 (OK) y el json { "data": { "data": [{ "id": 1, "code": "Prueba", "title": "Prueba 1", "edition": 1, "pages": 1, "dateOfValidity": "2024-02-18T03:40:27.429+00:00", "typeId": 14, "processId": 30, "scopeId": 1 }], "totalPages": 1 } }
		title	null	
		edition	null	
		pages	null	
		dateOfValidity	2024-02-19T02:30:07.642Z	
		typeId	14	
		processId	30	
		scopeId	1	
		pageNumber	1	
		pageSize	5	
		sortBy	0	
		sortOrder	1	
		dateComparator	4	

10	Obtener los documentos dado que la edición, páginas, id de alcance, número de páginas, tamaño de páginas y el orden presentan valores incorrectos	code	Lily	Se muestra el código de respuesta 400 (Bad Request) y el json { "problemDetails": { "status": 400, "errors": { "Edition": ["Edition must be greater than 0 and less than 32001"], "Pages": ["Pages must be greater than 0 and less than 32001"], "Scopeld": ["Scope id must be greater than 0"], "SortBy": ["SortBy of download must be valid"], "PaginationParams.PageNumber": ["Page Number of Pagination Params must be greater than 0"], "PaginationParams.PageSize": [
		title	Lilita	
		edition	0	
		pages	0	
		dateOfValidity	2024-02-19T02:30:07.642Z	
		typeld	14	
		processId	30	
		scopeld	0	
		pageNumber	0	
		pageSize	0	
		sortBy	-1	
		sortOrder	0	
		dateComparator	4	

			eba1prueba1prueba1p rueba1prueba1prueba 1prueba1	Request) y el json {
		code	Prueba11	
		edition	1	"problemDetails": {
		dateOfValidity	2024-02-18T05:46:50.312Z	"status": 400,
		typeId	28	"errors": {
		processId	41	"Title": [
		scopeId	1	"Title must have a maximum length of 64 characters"
		pdf	Prueba11.pdf]
		word	Prueba11.docx	}
				}
3	Crear documento con una edición no válida	title	Prueba11	Se muestra el código de respuesta 400 (Bad Request) y el json
		code	Prueba11	{
		edition	-5	"problemDetails": {
		dateOfValidity	2024-02-18T05:46:50.312Z	"status": 400,
		typeId	28	"errors": {
		processId	41	"Edition":
		scopeId	1	[
		pdf	Prueba11.pdf	"Edition must be greater than 0
		word	Prueba11.docx	

				and less than 32001"] } } }
4	Crear documento con un código null	title	Prueba11	Se muestra el código de respuesta 400 (Bad Request) y el json { "problemDetails": { "status": 400, "errors": { "Code": ["Code must not be empty"] } } }
		code	null	
		edition	1	
		dateOfValidity	2024-02-18T05:46:50.312Z	
		typeId	28	
		processId	41	
		scopeId	1	
		pdf	Prueba11.pdf	
		word	Prueba11.docx	
5	Crear documento con el id de tipo y el id de proceso no válidos	title	Prueba11	Se muestra el código de respuesta 400 (Bad Request) y el json {
		code	Prueba11	
		edition	1	
		dateOfValidity	2024-02-18T05:46:50.312Z	
		typeId	0	
		processId	0	

		scopeld	1	<pre> "problemDetails": { "status": 400, "errors": { "TypeId": ["Type id must be greater than 0"], "ProcessId": ["Process id must be greater than 0"] } } </pre>
		pdf	Prueba11.pdf	
		word	Prueba11.docx	
6	Crear documento subiendo documento .docx en lugar de pdf	title	Prueba11	<pre> Se muestra el código de respuesta 400 (Bad Request) y el json { "problemDetails": { "status": 400, "errors": { </pre>
		code	Prueba11	
		edition	1	
		dateOfValidity	2024-02-18T05:46:50.312Z	
		typeid	28	
		processId	41	
		scopeld	1	
		pdf	Prueba11.docx	
		word	Prueba11.docx	

				<pre> "Pdf": ["Only pdf files allowed (*.pdf)"] } } } </pre>
--	--	--	--	---

Anexo 6: Diseño de prueba de caja negra para el *endpoint* "GET /download/{id}"

Historia de Usuario	HU4: Obtener una descarga dado el id
Desarrollador	Lilian Rosa Rojas Rodríguez
Probador	Lilian Rosa Rojas Rodríguez
Fecha	17 de febrero de 2024

CP	Escenario	Nombre de variable de entrada	Valor	Resultados Esperados
1	Obtener descarga dado un id correcto	id	2	Se muestra el código de respuesta 200 (OK) y el json <pre> { "data": { "id": 2, "dateOfDownload": "2024-02-18T04:15:43.372264+00:00", "format": 1, "username": "Usuario de Prueba", "documentId": 1, "reasonId": 10 } } </pre>
2	No obtener descarga por id incorrecto	id	40	Se muestra el código de respuesta 404 (Not Found) y el json <pre> { "problemDetails": { "status": 404, </pre>

				<pre> "errors": { "Download Id": ["Doesn't exist a download with the given id"] } } </pre>
--	--	--	--	--

Anexo 7: Diseño de prueba de caja negra para el *endpoint* “POST /download/filter”

Historia de Usuario	HU5: Obtener las descargas con una característica determinada
Desarrollador	Lilian Rosa Rojas Rodríguez
Probador	Lilian Rosa Rojas Rodríguez
Fecha	17 de febrero de 2024

CP	Escenario	Nombre de variable de entrada	Valor	Resultados Esperados
1	Obtener las descargas con éxito dados todos los parámetros	format	0	Se muestra el código de respuesta 200 (OK) y el json <pre> { "data": { "data": [{ "id": 9, "dateOfDownload": "2024-02-18T04:17:34.240971+00:00", "format": 0, "username": "Usuario de Prueba", "documentId": 5, "reasonId": 11 }], "totalPages": 1 } } </pre>
		dateOfDownload	2024-02-19T01:06:53.088Z	
		username	usuario	
		documentId	5	
		reasonId	11	
		pageNumber	1	
		pageSize	2	
		sortBy	0	
		sortOrder	1	
		dateComparator	4	
2		format	null	

	Obtener las descargas dada la fecha y la comparación para antes de la fecha	dateOfDownload	2024-02-19T01:06:53.088Z	Se muestra el código de respuesta 200 (OK) y el json { "data": { "data": [{ "id": 11, }] } "dateOfDownload": "2024-02-18T08:28:26.28195+00:00", "format": 0, "username": "lilyrosa", "documentId": 1, "reasonId": 13 }, { "id": 10, } "dateOfDownload": "2024-02-18T04:17:40.115523+00:00", "format": 1, "username": "Usuario de Prueba", "documentId": 5, "reasonId": 11 }], "totalPages": 6 }
		username	null	
		documentId	null	
		reasonId	null	
		pageNumber	1	
		pageSize	2	
		sortBy	0	
		sortOrder	1	
		dateComparator	4	
3	Obtener las descargas con todos los parámetros en null	format	null	Se muestra el código de respuesta 200 (OK) y el json { "data": { "data": [{ "id": 11, }] } "dateOfDownload": "2024-02-18T08:28:26.28195+00:00", "format": 0,
dateOfDownload		null		
username		null		
documentId		null		
reasonId		null		
pageNumber		1		
pageSize		2		
sortBy		0		
sortOrder		1		
dateComparator	4			

				<pre>"username": "lilyrosa", "documentId": 1, "reasonId": 13 }, { "id": 10, "dateOfDownload": "2024-02- 18T04:17:40.115523+ 00:00", "format": 1, "username": "Usuario de Prueba", "documentId": 5, "reasonId": 11 }], "totalPages": 6 } }</pre>
4	Obtener las descargas dado el id del documento y el id de la razón correctos	format	null	Se muestra el código de respuesta 200 (OK) y el json <pre>{ "data": { "data": [{ "id": 10, "dateOfDownload": "2024-02- 18T04:17:40.115523+ 00:00", "format": 1, "username": "Usuario de Prueba", "documentId": 5, "reasonId": 11 }, { "id": 9, "dateOfDownload": "2024-02- 18T04:17:34.240971+ 00:00", "format": 0,</pre>
dateOfDownlo ad		null		
username		null		
documentId		5		
reasonId		11		
pageNumber		1		
pageSize		2		
sortBy		0		
sortOrder		1		
dateComparat or		4		

				<pre>"username": "Usuario de Prueba", "documentId": 5, "reasonId": 11 }], "totalPages": 1 } }</pre>																				
5	Obtener las descargas dado el id del documento y el id de la razón no válidos	<table><tr><td>format</td><td>null</td></tr><tr><td>dateOfDownload</td><td>null</td></tr><tr><td>username</td><td>null</td></tr><tr><td>documentId</td><td>-9</td></tr><tr><td>reasonId</td><td>-5</td></tr><tr><td>pageNumber</td><td>1</td></tr><tr><td>pageSize</td><td>2</td></tr><tr><td>sortBy</td><td>0</td></tr><tr><td>sortOrder</td><td>1</td></tr><tr><td>dateComparator</td><td>4</td></tr></table>	format	null	dateOfDownload	null	username	null	documentId	-9	reasonId	-5	pageNumber	1	pageSize	2	sortBy	0	sortOrder	1	dateComparator	4		Se muestra el código de respuesta 400 (Bad Request) y el json <pre>{ "problemDetails": { "status": 400, "errors": { "DocumentId": ["Document id must be greater than 0"], "ReasonId": ["Reason id must be greater than 0"] } } }</pre>
format	null																							
dateOfDownload	null																							
username	null																							
documentId	-9																							
reasonId	-5																							
pageNumber	1																							
pageSize	2																							
sortBy	0																							
sortOrder	1																							
dateComparator	4																							
6	Obtener las descargas dado el formato	<table><tr><td>format</td><td>0</td></tr><tr><td>dateOfDownload</td><td>null</td></tr><tr><td>username</td><td>null</td></tr><tr><td>documentId</td><td>null</td></tr><tr><td>reasonId</td><td>null</td></tr><tr><td>pageNumber</td><td>1</td></tr><tr><td>pageSize</td><td>2</td></tr><tr><td>sortBy</td><td>0</td></tr><tr><td>sortOrder</td><td>1</td></tr><tr><td>dateComparator</td><td>4</td></tr></table>	format	0	dateOfDownload	null	username	null	documentId	null	reasonId	null	pageNumber	1	pageSize	2	sortBy	0	sortOrder	1	dateComparator	4		Se muestra el código de respuesta 200 (OK) y el json <pre>{ "data": { "data": [{ "id": 11, "dateOfDownload": "2024-02-18T08:28:26.28195+00:00", "format": 0, "username": "lilyrosa", "documentId": 1, "reasonId": 13 }, { "id": 9,</pre>
format	0																							
dateOfDownload	null																							
username	null																							
documentId	null																							
reasonId	null																							
pageNumber	1																							
pageSize	2																							
sortBy	0																							
sortOrder	1																							
dateComparator	4																							

				<pre>"dateOfDownload": "2024-02- 18T04:17:34.240971+ 00:00", "format": 0, "username": "Usuario de Prueba", "documentId": 5, "reasonId": 11 }], "totalPages": 3 } }</pre>																				
7	Obtener las descargas dado el formato no válido	<table><tr><td>format</td><td>2</td></tr><tr><td>dateOfDownload</td><td>null</td></tr><tr><td>username</td><td>null</td></tr><tr><td>documentId</td><td>null</td></tr><tr><td>reasonId</td><td>null</td></tr><tr><td>pageNumber</td><td>1</td></tr><tr><td>pageSize</td><td>2</td></tr><tr><td>sortBy</td><td>0</td></tr><tr><td>sortOrder</td><td>1</td></tr><tr><td>dateComparator</td><td>4</td></tr></table>	format	2	dateOfDownload	null	username	null	documentId	null	reasonId	null	pageNumber	1	pageSize	2	sortBy	0	sortOrder	1	dateComparator	4		<pre>Se muestra el código de respuesta 400 (Bad Request) y el json { "problemDetails": { "status": 400, "errors": { "Format": ["Format must be valid"] } } }</pre>
format	2																							
dateOfDownload	null																							
username	null																							
documentId	null																							
reasonId	null																							
pageNumber	1																							
pageSize	2																							
sortBy	0																							
sortOrder	1																							
dateComparator	4																							
8	Obtener las descargas dado que el formato, el id de la razón y comparador de fechas no son válidos	<table><tr><td>format</td><td>-1</td></tr><tr><td>dateOfDownload</td><td>2024-02-19T01:06:53.088Z</td></tr><tr><td>username</td><td>Lily</td></tr><tr><td>documentId</td><td>5</td></tr><tr><td>reasonId</td><td>0</td></tr><tr><td>pageNumber</td><td>1</td></tr><tr><td>pageSize</td><td>2</td></tr><tr><td>sortBy</td><td>1</td></tr><tr><td>sortOrder</td><td>-1</td></tr><tr><td>dateComparator</td><td>-1</td></tr></table>	format	-1	dateOfDownload	2024-02-19T01:06:53.088Z	username	Lily	documentId	5	reasonId	0	pageNumber	1	pageSize	2	sortBy	1	sortOrder	-1	dateComparator	-1		<pre>Se muestra el código de respuesta 400 (Bad Request) y el json { "problemDetails": { "status": 400, "errors": { "Format": ["Format must be valid"], "ReasonId": ["Reason id must be greater than 0"], "SortOrder": ["SortOrder of download must be valid"] } } }</pre>
format	-1																							
dateOfDownload	2024-02-19T01:06:53.088Z																							
username	Lily																							
documentId	5																							
reasonId	0																							
pageNumber	1																							
pageSize	2																							
sortBy	1																							
sortOrder	-1																							
dateComparator	-1																							

				} }
9	Obtener las descargas dada una paginación no válida	format	0	Se muestra el código de respuesta 400 (Bad Request) y el json { "problemDetails": { "status": 400, "errors": { "PaginationParams.PageNumber": ["Page Number of Pagination Params must be greater than 0"], "PaginationParams.PageSize": ["Page Size of Pagination Params must be greater than 0"] } } }
		dateOfDownload	null	
		username	null	
		documentId	null	
		reasonId	null	
		pageNumber	0	
		pageSize	0	
		sortBy	0	
		sortOrder	1	
dateComparator	4			

Anexo 8: Diseño de prueba de caja negra para *endpoint* "POST /download"

Historia de Usuario	HU6: Descargar un documento en un formato específico
Desarrollador	Eduardo Alejandro González Martell
Probador	Lilian Rosa Rojas Rodríguez
Fecha	17 de febrero de 2024

CP	Escenario	Nombre de variable de entrada	Valor	Resultados Esperados
1	Descargar documento con éxito	format	0	Se muestra el código de respuesta 200 (OK) y el json { "data": "https://bucket-production-b52d.up.railway.app/aica-docs/pdf/Prueba1.pdf?X-
		username	Lilyrosa	
		documentId	1	
		reasonId	13	

				Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=tes tuser%2F20240218%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20240218T082826Z&X-Amz-Expires=300&X-Amz-SignedHeaders=host&X-Amz-Signature=1f8bf179d51b93a4d1542ce90d0bfc57b290147a6f4543ce75285af4e91d4d31"}
2	Descargar documento dado un formato no válido	format	2	Se muestra el código de respuesta 400 (Bad Request) y el json <pre>{ "problemDetails": { "status": 400, "errors": { "Format": ["Format must be valid"] } } }</pre>
		username	Lilyrosa	
		documentId	1	
		reasonId	13	
3	Descargar documento dado un usuario null	format	0	Se muestra el código de respuesta 400 (Bad Request) y el json <pre>{ "problemDetails": { "status": 400, "errors": { "Username": ["Username must not be empty"] } } }</pre>
		username	null	
		documentId	1	
		reasonId	13	
4	Descargar documento dado un id de documento	format	0	Se muestra el código de respuesta 404 (Not Found) y el json <pre>{ "problemDetails": { "status": 404,</pre>
		username	Lilyrosa	
		documentId	90	
		reasonId	13	

	no existente			<pre> "errors": { "Document Id": ["Doesn't exist a document with the given id"] } </pre>
5	Descargar documento dada una razón no válida	format	0	Se muestra el código de respuesta 400 (Bad Request) y el json <pre> { "problemDetails": { "status": 400, "errors": { "ReasonId": ["Reason Id must be valid"] } } } </pre>
		username	Lilyrosa	
		documentId	1	
		reasonId	1	

Anexo 9: Diseño de caso de prueba de caja negra para el *endpoint* “GET /nomenclator/{type}”

Historia de Usuario	HU7: Obtener los nomencladores de un tipo dado
Desarrollador	Eduardo Alejandro González Martell
Probador	Lilian Rosa Rojas Rodríguez
Fecha	17 de febrero de 2024

CP	Escenario	Nombre de variable de entrada	Valor	Resultados Esperados
1	Obtener nomencladores dado un tipo correcto	type	1	Se muestra el código de respuesta 200 (OK) y el json <pre> { "data": [{ "id": 9, "name": "Envío a personal externo autorizado", "type": 1 }] } </pre>

				<pre> }, { "id": 10, "name": "Elaboración de CTD", "type": 1 }, { "id": 11, "name": "Solicitud del CECMED", "type": 1 }, { "id": 12, "name": "Revisión por el área de documentación", "type": 1 }, { "id": 13, "name": "Revisión del área de proceso", "type": 1 }, { "id": 8, "name": "Distribución Impresa", "type": 1 }] } </pre>
2	No obtener nomencladores por type incorrecto	type	55	<p>Se muestra el código de respuesta 404 (Not Found) y el json</p> <pre> { "problemDetails": { "status": 400, "errors": { "Nomenclator Type": ["Type of Nomenclator must be valid"] } } } </pre>

Anexo 10: Diseño de prueba de caja negra para el *endpoint* “GET /nomenclator/{type}/{id}”

Historia de Usuario	HU8: Obtener nomenclador dado su tipo e id
Desarrollador	Eduardo Alejandro González Martell
Probador	Lilian Rosa Rojas Rodríguez
Fecha	17 de febrero de 2024

CP	Escenario	Nombre de variable de entrada	Valor	Resultados Esperados
1	Obtener nomenclador dado un id y un tipo correctos	id	41	Se muestra el código de respuesta 200 (OK) y el json <pre>{ "data": { "id": 41, "name": "Gestión de Documentación", "type": 0 } }</pre>
		type	0	
2	Obtener nomenclador dado un id incorrecto y un tipo correcto	id	1	Se muestra el código de respuesta 404 (Not Found) y el json <pre>{ "problemDetails": { "status": 404, "errors": { "Nomenclator Id Type": ["Doesn't exist a nomenclator with the given Id and Type"] } } }</pre>
		type	0	
3	Obtener nomenclador dado un id correcto y un tipo incorrecto	id	1	Se muestra el código de respuesta 404 (Not Found) y el json <pre>{ "problemDetails": { "status": 400, "errors": { "Nomenclator Type": ["Type of Nomenclator must be valid"] } } }</pre>
		type	-1	

Anexo 11: Diseño de prueba de caja negra para el endpoint “POST /nomenclator”

Historia de Usuario	HU9: Crear un nomenclador
Desarrollador	Eduardo Alejandro González Martell
Probador	Lilian Rosa Rojas Rodríguez
Fecha	17 de febrero de 2024

[illegible]

	tipo no válidos		ba1prueba1 prueba1pru eba1prueba 1prueba1pr ueba1prueb a1prueba1p rueba	<pre>"problemDetails": { "status": 400, "errors": { "Name": ["Name must have a maximum length of 64 characters"], "Type": ["Type must be valid"] } }</pre>
		type	-1	

Anexo 12: Diseño de prueba de caja negra para el *endpoint* “PATCH /nomenclator/{id}”

Historia de Usuario	HU10: Actualizar el nombre de un nomenclador dado el id
Desarrollador	Eduardo Alejandro González Martell
Probador	Lilian Rosa Rojas Rodríguez
Fecha	17 de febrero de 2024

CP	Escenario	Nombre de variable de entrada	Valor	Resultados Esperados
1	Actualizar un nomenclador con éxito	id	1	Se muestra el código de respuesta 200 (OK)
		name	rector	
2	Actualizar un nomenclador dado un id no válido	id	0	Se muestra el código de respuesta 400 (Bad Request) y el json <pre>{ "problemDetails": { "status": 404, "errors": { "Nomenclator Id": ["Doesn't exist a nomenclator with the given Id"] } } }</pre>
		name	rector	

