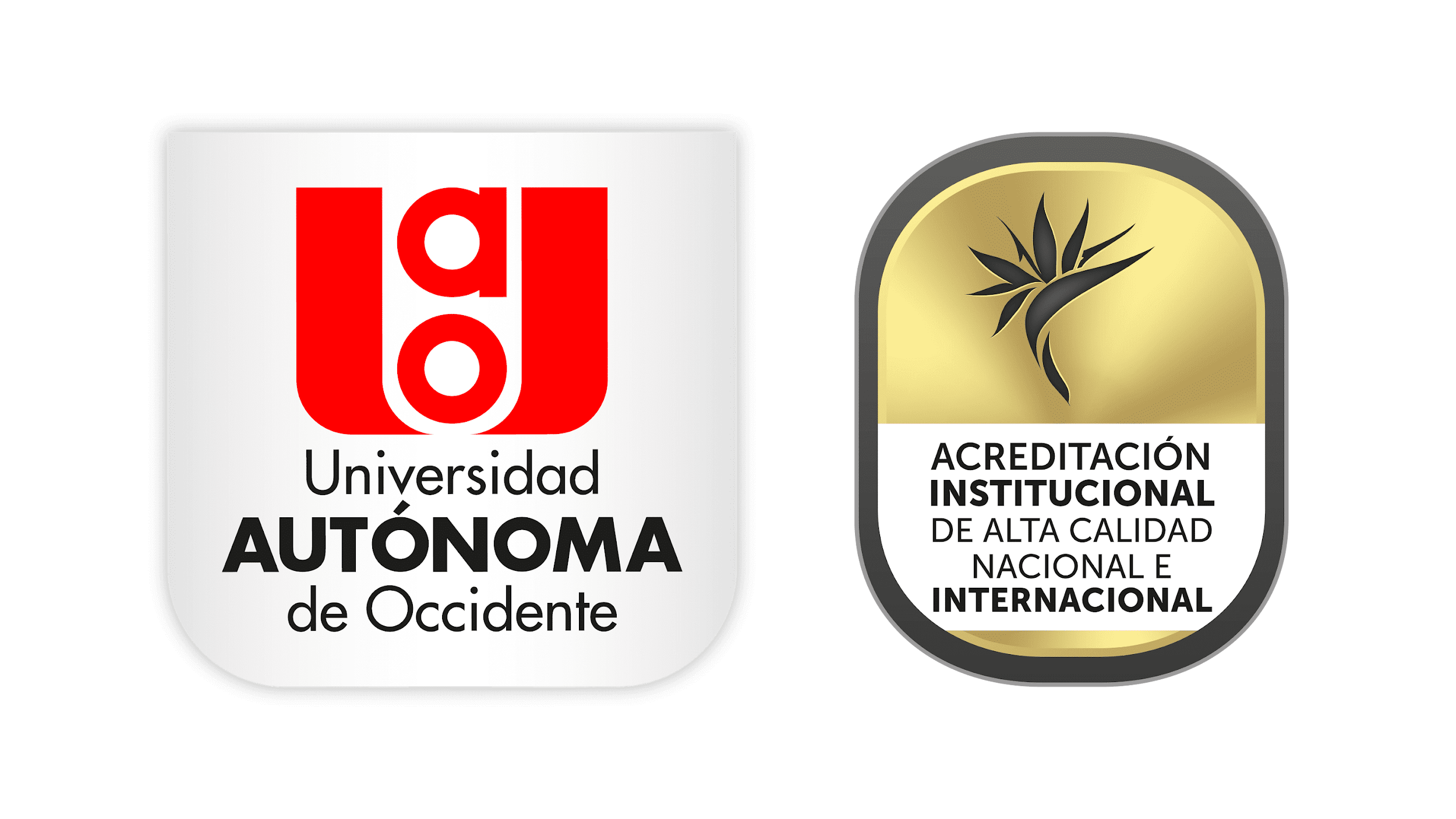
**Desarrollo del Sistema Web DocTIC: Análisis de Arquitectura, Selección de Tecnologías y Enfoque**



**INTEGRANTES DEL EQUIPO:**

**William Reyes Valencia - 2215337**

**Gerson Steven López -2205177**

**Jean Paul Delgado – 2226380**

**Sergio Andrés Herrera – 2226223**

**Etienne Samboni Piamba – 2226219**

**Julián Salamanca - 2221009**

**PROYECTO DE CURSO**

**INGENIERÍA DE SOFTWARE**

**PRESENTADO A:**

**Jhon Eder Masso Daza Ph.D**

**UNIVERSIDAD AUTÓNOMA DE OCCIDENTE FACULTAD DE INGENIERÍA**

**NÚCLEO MIDIA**

**SANTIAGO DE CALI**

**2024**

Contenido

[Introducción y objetivo general del proyecto 3](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.3znysh7)

[Metodología 3](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.2et92p0)

[Identificación de los Requisitos 3](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.tyjcwt)

[Arquitectura del Sistema y Tecnologías de desarrollo](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.3dy6vkm) 5

[Desarrollo de la metodología](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.4d34og8) 5

[Sprint 1](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.2s8eyo1) 6

[Análisis](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.17dp8vu) 6

[Diseño](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.3rdcrjn) 7

[Implementación](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.26in1rg) 9

[Sprint 2](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.35nkun2) 10

[Análisi](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.1ksv4uv)s 10

Pruebas10

[Sprint 3](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.2jxsxqh)  11

Implementación 11

[Cierre del proyecto](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.z337ya) 12

[Conclusiones y Lecciones aprendidas](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.3j2qqm3) 13

[Actualizaciones y mejoras](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.1y810tw) 13

[Referencias](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.4i7ojhp) 13

[Anexos:](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.2xcytpi) 14

[Documento de instalación y configuración del sistema](https://docs.google.com/document/d/1DgTLvze-nsIGMfLW-sLum8hpQJS2sltkuHW1t0QylQs/edit#heading=h.1ci93xb) 14

# Introducción y objetivo general del proyecto

El proyecto DocTIC, desarrollado en el marco de la asignatura Ingeniería de Software 1, surge con el propósito de aplicar y consolidar los fundamentos teóricos y prácticos adquiridos en el curso. Este sistema académico busca facilitar la publicación, valoración y retroalimentación de documentos electrónicos relacionados con múltiples áreas de las ciencias, permitiendo a los usuarios publicar, comentar y valorar los documentos de otros usuarios.

Este proyecto es relevante en un contexto académico, ya que promueve el desarrollo de habilidades en diseño, implementación y análisis de un proyecto informático bajo un enfoque iterativo e incremental. Además, fomenta la colaboración y comunicación en equipo, alineándose con los objetivos de la asignatura para formar futuros ingenieros con competencias en la construcción de software confiable, seguro y escalable.

# Metodología

Para el desarrollo del proyecto DocTIC se empleó una metodología iterativa e incremental, adaptando el marco ágil Scrum para la gestión del trabajo en equipo. Este enfoque permite entregar versiones funcionales del sistema en ciclos cortos, lo cual facilita la incorporación de mejoras y la realización de ajustes basados en las pruebas y retroalimentación obtenidas en cada iteración y retroalimentación del product owner.

La metodología Scrum asignó roles clave y una distribución clave entre los sprints para realizar entregas funcionales y una culminación eficiente de tareas: el Product Owner (docente) definió los requisitos y prioridades del sistema, supervisando el cumplimiento de los objetivos en cada entrega. Los desarrolladores llevaron a cabo las actividades de diseño, implementación y pruebas del sistema de cada épica establecida.

En cada Sprint, el equipo se centró en desarrollar funcionalidades específicas, como por ejemplo el registro de usuarios, publicación de documentos y sistema de valoraciones. Además, se realizaron reuniones de seguimiento para evaluar el progreso y ajustar el backlog según las necesidades del proyecto. Este proceso iterativo ayudó a mejorar progresivamente el sistema y a lograr entregables alineados con los requisitos definidos al principio.

# Identificación de los Requisitos

Para estructurar y organizar los requisitos de DocTIC, el equipo desarrollador utilizó Épicas e historias de Usuario (HU). Las Épicas representan funcionalidades clave que el sistema debe implementar, mientras que las Historias de Usuario detallan los requisitos específicos desde la perspectiva de los usuarios finales. Esta organización permite una priorización efectiva de tareas y facilita la planificación de los sprints de desarrollo.

### **Épica Principal**

En este proyecto, el equipo se centró en el planteamiento de la Épica número 3: **Exploración y Búsqueda de Documentos**. Esta Épica responde a la necesidad del usuario general de poder navegar y buscar documentos de interés dentro del catálogo de DocTIC. Incluye funcionalidades esenciales como listar documentos, realizar búsquedas avanzadas, aplicar filtros, visualizar detalles de documentos y gestionar descargas. Esta Épica se subdividió en múltiples Historias de Usuario (3.1 a 3.9), que abordaron distintos aspectos de la funcionalidad de exploración.

### **Historias de Usuario**

Las Historias de Usuario asignadas dentro de esta Épica cubren requerimientos específicos que permiten al usuario realizar tareas concretas en el sistema. Algunos ejemplos destacados son:

* **HU 3.1 - Listar Catálogo de Documentos**: Esta historia permite al usuario general acceder al catálogo completo de documentos, con información básica como el título y el autor, facilitando una navegación intuitiva. La tarea fue asignada a Gerson, quien se encargó de asegurar la accesibilidad y rapidez en el despliegue de esta información.
* **HU 3.2 - Buscar Documentos**: Implementada por Jean Paul, esta funcionalidad permite realizar búsquedas mediante palabras clave, autor o título, optimizando la usabilidad con características como autocompletado y mensajes para búsquedas sin resultados.
* **HU 3.3 - Filtros de Búsqueda sobre los Documentos**: Sergio desarrolló esta historia, que permite al usuario aplicar filtros avanzados (fecha, autor, categoría, entre otros) para personalizar los resultados de búsqueda.
* **HU 3.8 - Descargar los Documentos**: Julián fue responsable de esta funcionalidad, que permite a los usuarios registrados descargar documentos en formato PDF, gestionando permisos de acceso y registro de descargas en el perfil del usuario.

Documento completo:

Para cada historia se definieron criterios de aceptación detallados, asegurando que el desarrollo cumpla con los requisitos iniciales y mantenga una alta calidad de experiencia de usuario. Además de ser aceptados y validados por el product owner y el resto del grupo para mantener un estándar en el proyecto.

**Arquitectura del Sistema y Tecnologías de Desarrollo**

El sistema DocTIC ha sido diseñado siguiendo una arquitectura por capas que organiza y separa las responsabilidades del sistema para garantizar una estructura modular, escalable y fácil de mantener. esta arquitectura divide el sistema en tres capas principales: **Frontend**, **Lógica de Negocio** y **Acceso a Datos**. Esta separación permite desarrollar, probar y modificar cada capa de forma independiente, optimizando el desarrollo y mantenimiento.

### **Descripción de la Arquitectura**

1. **Frontend**
   1. **Tecnología seleccionada:** **Next.js**
   2. **Justificación:** Next.js, basado en React, proporciona funcionalidades avanzadas de renderizado en servidor (SSR), mejorando el rendimiento de la aplicación y optimizando el SEO (Next.js, s.f.). Su capacidad para manejar APIs internas simplifica la arquitectura, permitiendo una experiencia fluida y veloz en el frontend.
2. **Backend**
   1. **Tecnología seleccionada:** **Spring Boot**
   2. **Justificación:** Spring Boot fue seleccionado por su compatibilidad con MongoDB y React, además de sus robustas opciones de seguridad y autenticación. Su facilidad de integración y la experiencia previa del equipo con esta tecnología también jugaron un rol clave en su elección (Romero et al., 2012).
3. **Base de Datos**
   1. **Tecnología seleccionada:** **MongoDB**
   2. **Justificación:** MongoDB fue elegido por su rendimiento y capacidad de manejar datos no estructurados y grandes volúmenes, características esenciales para DocTIC, donde se almacenarán y consultarán numerosos documentos y comentarios (Durán, 2023).

#### **Herramientas de Desarrollo Colaborativo**

Para facilitar el trabajo en equipo, **Git y GitHub** han sido seleccionados como herramientas de control de versiones, permitiendo la creación de ramas independientes para cada funcionalidad y/o desarrollador, la implementación de Pull Requests, y la revisión de código colaborativa, asegurando la calidad y estabilidad del proyecto.

#### **Compatibilidad e Integración de Tecnologías**

La interoperabilidad entre Next.js, Spring Boot y MongoDB se facilita mediante una API REST, permitiendo una comunicación eficaz entre el frontend y el backend, y garantizando la integridad del sistema en conjunto.

# Desarrollo de la metodología

La metodología implementada para el desarrollo de *DocTIC* se caracterizó por un enfoque colaborativo y de comunicación continua, esencial para la integración de todos los miembros del equipo. La comunicación se mantuvo activa a través de WhatsApp, lo cual facilitó la coordinación de tareas diarias y la resolución rápida de dudas, garantizando que todos los integrantes estuvieran al tanto de los avances y cambios del proyecto. Adicionalmente, cada semana se realizaban reuniones virtuales mediante Google Meet, donde cada miembro presentaba sus avances, aclaraba dudas y se asignaban nuevas tareas. Estas sesiones virtuales aseguraban un seguimiento preciso del progreso del proyecto y facilitaban la solución de problemas técnicos en un espacio estructurado.

Para la organización y gestión de tareas, se utilizó la plataforma Miro como un tablero colaborativo. En él, el equipo podía visualizar las tareas pendientes, los sprints en curso, las actividades finalizadas, y los problemas identificados, además de especificar al programador responsable de cada tarea. Esta estructura organizativa permitía una administración clara de las responsabilidades y ayudaba a resolver posibles cuellos de botella en el flujo de trabajo.

Además, se realizaron reuniones presenciales periódicas para nivelar el conocimiento técnico entre los integrantes. Los miembros con mayor experiencia en lenguajes y frameworks específicos, como Next.js y Spring Boot, ofrecieron sesiones de capacitación a aquellos con menor conocimiento en estas áreas, asegurando así que cada miembro pudiera contribuir efectivamente. Esta combinación de comunicación activa, organización estructurada y aprendizaje colaborativo permitió al equipo abordar los retos del desarrollo de *DocTIC* de forma eficaz y coordinada, maximizando la cohesión y el rendimiento del grupo en cada etapa del proyecto.

## Sprint 1

### Análisis

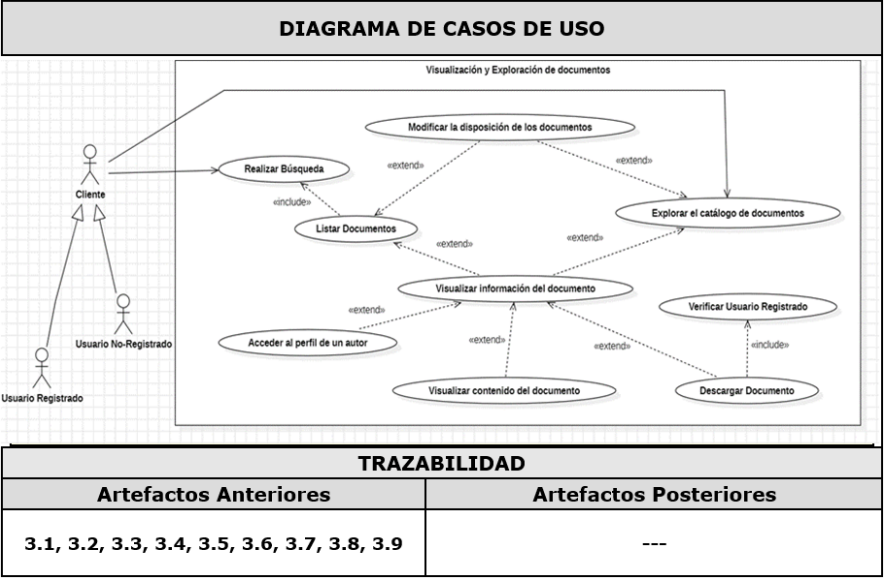
Durante el Sprint 1 se abordaron las actividades relacionadas con la épica número 3, "Exploración y Búsqueda de Documentos", la cual se centró en permitir al usuario general explorar el catálogo de publicaciones de *DocTIC*, realizar búsquedas y aplicar filtros para encontrar documentos de interés.

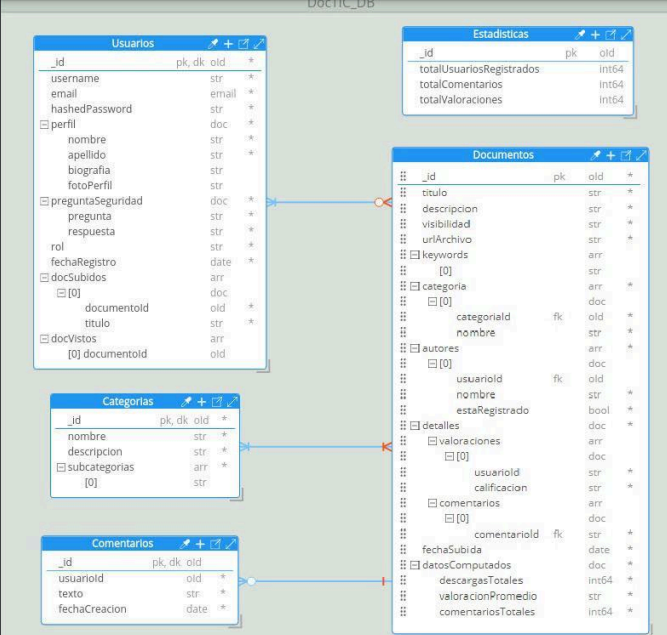
En esta etapa, se realizó el refinamiento y la especificación detallada de las historias de usuario asociadas a esta épica para asegurar la claridad en los requisitos y la alineación con los objetivos de funcionalidad. Cada historia de usuario fue analizada cuidadosamente, y los criterios de aceptación fueron revisados y actualizados según fue necesario. Esto incluyó ajustes en los detalles de interacción, como la disposición y usabilidad del campo de búsqueda, los filtros, la visualización de documentos y la información detallada del autor, que eran esenciales para garantizar una experiencia de usuario intuitiva y sin fricciones.

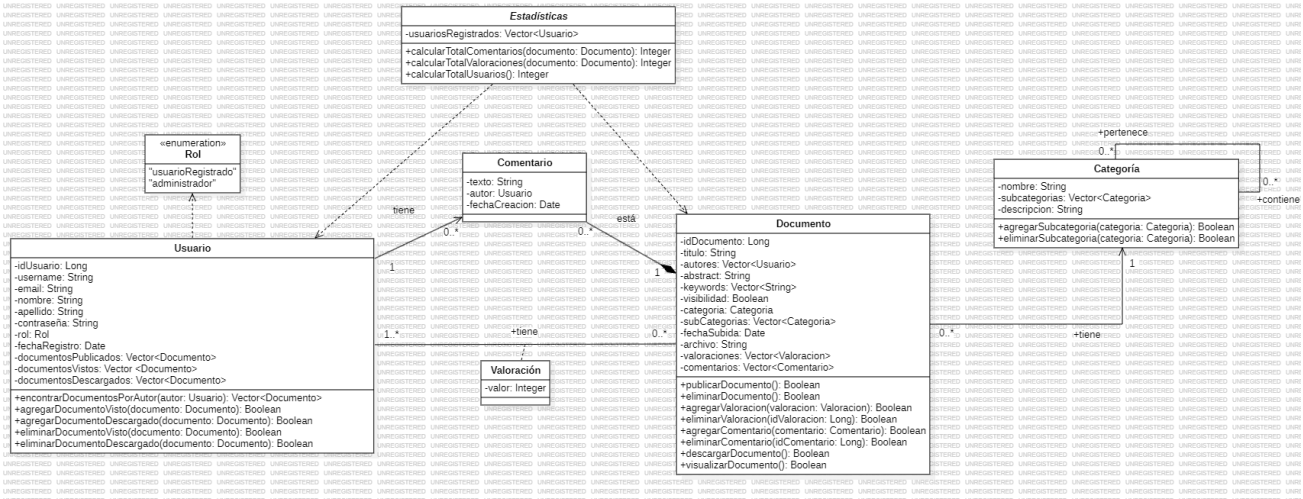
Durante el proceso de refinamiento, se discutió y acordó la implementación de un visor PDF para la visualización en línea de los documentos, junto con una interfaz responsiva que se adapte a diferentes dispositivos. Además, se establecieron los detalles sobre cómo se mostraría la información del autor y se rediseñaron algunos aspectos del sistema de búsqueda y de filtros para facilitar su uso. Finalmente, se ajustaron los requisitos para la visualización de documentos privados y los criterios de ordenamiento de documentos, logrando así una especificación robusta para guiar las actividades de desarrollo en los siguientes sprints.

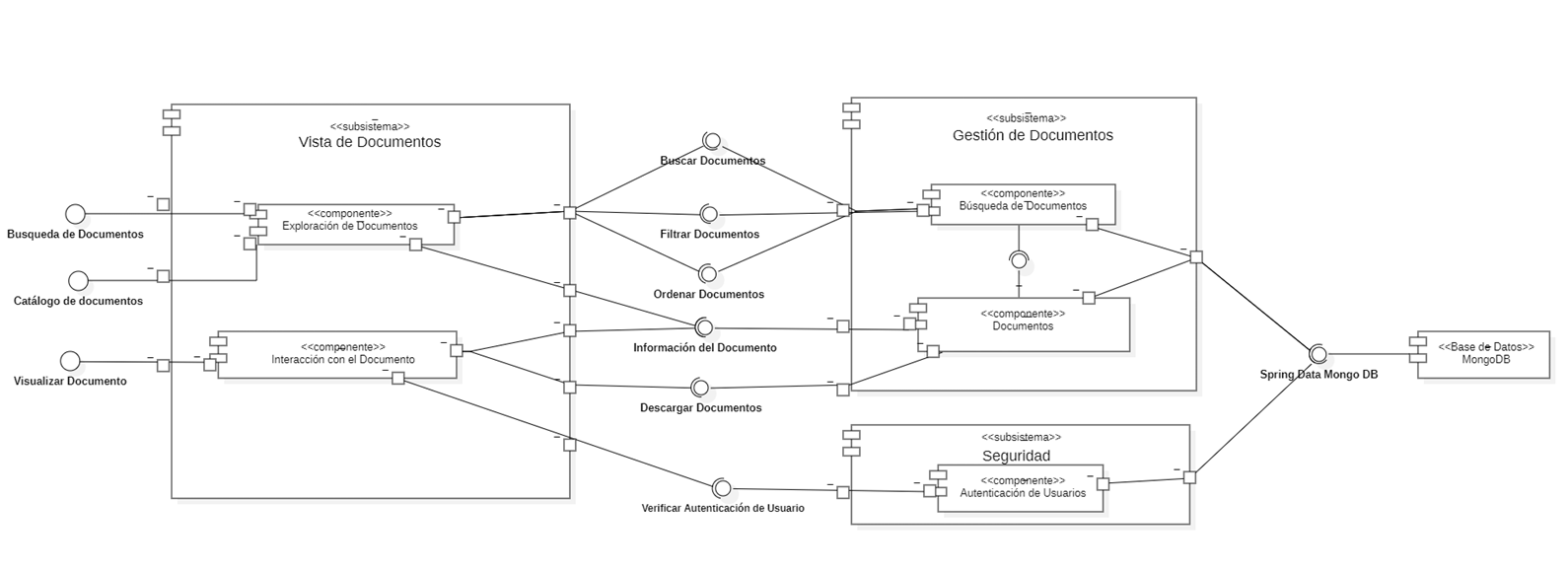
Este trabajo permitió sentar las bases para la implementación de las funcionalidades de búsqueda y exploración, asegurando que los criterios de aceptación estuvieran alineados con los objetivos del proyecto y las expectativas del usuario final.

### Diseño









### Implementación

La implementación del proyecto comenzó con el desarrollo del backend, utilizando Java y el framework Spring Boot bajo una arquitectura de software en capas. Este enfoque modular permite una separación clara de responsabilidades, facilitando el mantenimiento y escalabilidad del sistema. El backend se estructuró en distintas capas: la capa de control para gestionar las solicitudes de los usuarios, la capa de servicios para manejar la lógica de negocio y la capa de repositorio para interactuar con la base de datos.

En cuanto a la base de datos, se configuró un clúster de MongoDB para almacenar y gestionar los datos de manera eficiente. En este clúster, se ingresaron datos de ejemplo que permitirán realizar pruebas iniciales y verificar el funcionamiento de las consultas y operaciones CRUD necesarias para las funcionalidades de *DocTIC*.

Además, se desarrolló un frontend básico con el propósito de realizar pruebas iniciales. Este frontend permite verificar el flujo de datos y evaluar la integración de algunas funcionalidades clave entre el backend y la base de datos. A través de este frontend preliminar, se han realizado pruebas de las principales operaciones, asegurando así la comunicación efectiva y consistente entre los diferentes componentes del sistema.

## Sprint 2

### Análisis

Durante el Sprint 2 se completaron las épicas 2 y 3, y se elaboró un plan de pruebas para validar sus funcionalidades. En este proceso, se refinaron las historias de usuario y se especificaron los criterios de aceptación necesarios para asegurar que los desarrollos cumplieran con los requerimientos.

Para la **épica 2**, se verificó que el sistema permite la creación, eliminación y edición de publicaciones, proporcionando mensajes de confirmación y asegurando que las modificaciones se reflejen de inmediato en la interfaz de usuario. Esto incluyó pruebas de acceso a formularios de carga de documentos, validación de campos obligatorios, control de visibilidad de publicaciones y gestión de mensajes de error.

En cuanto a la **épica 3**, se realizaron pruebas de acceso y navegación al catálogo de documentos, incluyendo la funcionalidad de búsqueda, filtros y ordenamiento de resultados. Se evaluó que los filtros permitieran aplicar múltiples criterios y se mostraran en tiempo real, proporcionando una experiencia intuitiva y rápida. También se validó la visualización de documentos en un visor PDF integrado, compatible con distintos dispositivos y navegadores.

Finalmente, se culminó el desarrollo de un frontend básico para realizar pruebas iniciales de algunas funcionalidades, permitiendo una interfaz de usuario provisional para evaluar la interacción con el backend y la base de datos.

#### Pruebas

### **1. Definición del Plan de Pruebas**

Este plan de pruebas tiene como objetivo validar y verificar las funcionalidades de la aplicación web **DocTIC** mediante pruebas de caja negra, enfocándose en aspectos clave como la interacción del usuario, la visualización de documentos, el acceso a perfiles de autores y la gestión de publicaciones. Se detallan los casos de prueba con sus respectivos escenarios y se establece el tiempo estimado para cada prueba. A continuación, se documentan las fechas de ejecución y los resultados obtenidos para cada caso de prueba, incluyendo la evidencia de solución de los casos que no pasaron inicialmente.

### **3. Resultados Finales**

Al finalizar la ejecución de todos los casos de prueba, se generará un informe consolidado de los resultados obtenidos, con evidencias de las pruebas ejecutadas. En este caso todas las pruebas con un resultado satisfactorio.

## Sprint 3

Durante el Sprint 3, el equipo se centró en la implementación de la Épica 1: "Gestionar Usuarios". Esta épica aborda las funcionalidades clave relacionadas con el registro, autenticación y actualización de cuentas de usuario en el sistema DocTIC.

Al analizar las historias de usuario enmarcadas en esta épica, el equipo identificó varios desafíos relacionados con la arquitectura de la base de datos y la estrategia de autenticación. Específicamente, se determinó que las historias de usuario HU 1.1 "Crear usuario" y HU 1.2 "Actualización de datos del usuario" requerían una revisión más profunda.

### Implementación

En el Sprint 3, el equipo se enfocó en el desarrollo de las funcionalidades relacionadas con la Épica 1: "Gestionar Usuarios". Dado los ajustes realizados en la fase de análisis, el alcance de implementación se centró en las siguientes historias de usuario:

HU 1.1 - Crear usuario:

* Se implementó un formulario de registro que permite a los usuarios ingresar un nombre de usuario, contraseña y detalles personales básicos.
* Se validaron los campos requeridos y se aplicaron restricciones de seguridad, como requisitos mínimos de longitud y complejidad de la contraseña.
* Se implementó un flujo de registro que almacena los datos del usuario en la base de datos de MongoDB.
* Se envía un correo electrónico de confirmación al usuario con un enlace para activar su cuenta.

HU 1.3 - Eliminar cuenta de usuario:

* Se desarrolló una funcionalidad que permite a los usuarios eliminar permanentemente su cuenta del sistema.
* Antes de eliminar la cuenta, se solicita la contraseña actual del usuario para verificar su identidad.
* Al confirmar la eliminación, se borran todos los datos del usuario, incluyendo sus documentos publicados y cualquier información asociada.
* Se envía un correo electrónico de confirmación al usuario indicando que su cuenta ha sido eliminada.

HU 1.4 - Inicio Sesión (Autenticación):

* Se implementó un formulario de inicio de sesión que solicita el nombre de usuario y contraseña del usuario.
* Se validan las credenciales ingresadas contra los registros almacenados en la base de datos.
* Si las credenciales son válidas, se establece una sesión de usuario y se redirige al usuario a su perfil o a la página principal.
* Se implementaron controles de seguridad, como límite de intentos fallidos y bloqueo temporal de cuentas.

HU 1.5 - Recuperar credenciales:

* Se desarrolló un flujo de recuperación de contraseña que solicita el correo electrónico asociado a la cuenta.
* Si el correo electrónico es válido, se envía un mensaje al usuario con un enlace seguro para restablecer la contraseña.
* Al hacer clic en el enlace, se solicita que el usuario responda correctamente a la pregunta de seguridad previamente configurada.
* Si la respuesta es correcta, se permite al usuario establecer una nueva contraseña.

Estas historias de usuario cubren los aspectos fundamentales de gestión de cuentas de usuario, incluyendo registro, autenticación, recuperación de credenciales y eliminación de cuentas. El equipo trabajó de forma coordinada para implementar estas funcionalidades de manera robusta y segura, considerando los ajustes realizados en la fase de análisis.

# Cierre del proyecto

## Referencias

Iturralde. O, J. (2020). *Introducción a la arquitectura de software – Un enfoque. Práctico (1a ed.).*<https://fondoemprender.uts.edu.co/wp-content/uploads/2024/03/Introduccion-a-la-arquitectura-de-software-v1.0.2.pdf>

Gómez. A, F. (2020). *¿Qué sabemos de la complejidad de los proyectos?*<https://es.linkedin.com/pulse/qu%C3%A9-sabemos-de-la-complejidad-los-proyectos-andres-felipe-gomez>

Durán. M. (12 de abril de 2023). *Qué es la arquitectura en capas, ventajas y ejemplos.* HubSpot.<https://blog.hubspot.es/website/que-es-arquitectura-en-capas>

Pelayo. A. (2023). *La complejidad en el desarrollo de aplicaciones móviles: ¿amiga o enemiga del tiempo?*<https://es.linkedin.com/pulse/la-complejidad-en-el-desarrollo-de-aplicaciones-m%C3%B3viles-edgar-pelayo>

Wagner. A, A. (2024). *Arquitectura Monolítica.* Medium. [*https://medium.com/@arielwagnermovil/arquitectura-monol%C3%ADtica-274e6e1dbd56*](https://medium.com/@arielwagnermovil/arquitectura-monol%C3%ADtica-274e6e1dbd56)

*Next.JS* (s/f). Nextjs.org.<https://nextjs.org/docs/app/building-your-application/rendering>

Romero, A. C., Sanabria, J. S. G., & Cuervo, M. C. (2012). Utilidad y funcionamiento de las bases de datos NoSQL. *Facultad de Ingeniería*, *21*(33), 21-32.<https://www.redalyc.org/pdf/4139/413940772003.pdf>

Newman, S. (2015). *Building Microservices: Designing Fine-Grained Systems*. O'Reilly Media. [https://books.google.com.co/bookshl=es&lr=&id=ZvM5EAAAQBAJ&oi=fnd&pg=PT8&dq=Newman,+S.+(2015).+Building+Microservices:+Designing+Fine-Grained+Systems.+O%27Reilly+Media.&ots=ui6l8vcJTn&sig=SnOHox8t3LDO4ltoYbguq3zaLI#v=onepage&q&f=false](https://books.google.com.co/books?hl=es&lr=&id=ZvM5EAAAQBAJ&oi=fnd&pg=PT8&dq=Newman,+S.+(2015).+Building+Microservices:+Designing+Fine-Grained+Systems.+O%27Reilly+Media.&ots=ui6l8vcJTn&sig=SnOHox8t3LDO4lt-oYbguq3zaLI#v=onepage&q&f=false)

Fowler, M., & Lewis, J. (2014). *Microservices: A definition of this new architectural term*. Martinfowler.com.<https://martinfowler.com/articles/microservices.html>

# Anexos:

Una vez que se haya desplegado en un servidor para su uso en línea. El sistema está compuesto por tres tecnologías principales:

* **Frontend:** Next.js
* **Backend:** Spring Boot
* **Base de Datos:** MongoDB

### **6.1. Acceso al Sistema**

Una vez que el sistema esté desplegado en el servidor, se podrá acceder desde cualquier navegador. Para esto, solo necesitarás la **URL del sistema** que será proporcionada por el administrador del servidor.

### **6.2. Requisitos previos para la instalación (Servidor)**

Para desplegar el sistema en el servidor, asegúrate de tener lo siguiente:

* **Node.js** (versión 20 o superior) instalado en el servidor para el frontend (Next.js).
* **Java JDK 11 o superior** en el servidor para el backend junto a Spring Boot.
* **MongoDB** instalado y configurado en el servidor
* **4GB de RAM**  o más.