# ESCUELA POLITÉCNICA NACIONAL

## FACULTAD DE INGENIERÍA DE SISTEMAS

## DESARROLLO DE UN SISTEMA DE TOMA DE DECISIONES PARA APOYO A LOS INVESTIGADORES ECUATORIANOS BASADO EN MOTORES DE BÚSQUEDA Y DE RECOMENDACIÓN

## IMPLEMENTACIÓN DEL MÓDULO DE ANALÍTICA: DASHBOARDS Y ESTADÍSTICAS PARA LA TOMA DE DECISIONES

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SOFTWARE**

**JOFFRE ALEXANDER CÓNDOR TIPÁN**
joffre.condor@epn.edu.ec

**DIRECTOR: PhD. LORENA KATHERINE RECALDE CERDA**
lorena.recalde@epn.edu.ec

**DQM, JULIO 2024**

# CERTIFICACIONES

Yo, Joffre Alexander Cóndor Tipán, declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

Joffre Alexander Cóndor Tipán

Certifico que el presente trabajo de integración curricular fue desarrollado por Joffre Alexander Cóndor Tipán, bajo mi supervisión.

PhD. Lorena Katherine Recalde Cerda
DIRECTOR

# DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Joffre Alexander Cóndor Tipán

Lorena Katherine Recalde Cerda

Danny Ruben Cabrera Aguilar
Rommel Paúl Masabanda Castro
Jhon Fernando Sangopanta Naula

# DEDICATORIA

En primer lugar, quisiera dedicar este trabajo a mis padres, Lorena Tipán y Luis Cóndor, cuyo amor y apoyo incondicional han sido el pilar de mi vida. Gracias por enseñarme la importancia de la dedicación y la perseverancia, y por estar siempre a mi lado, brindándome su sabiduría y cariño.

A mi perrito Jaco, que aunque ya no está conmigo, siempre fue una compañía incondicional y ahora es una fuente de recuerdos llenos de felicidad.

A mis perritos Ares, Loki, y Max, que me han acompañado durante las duras madrugadas y han sido una constante fuente de alegría y lealtad.

Finalmente, a mi abuelito, que me ve desde el cielo, por su apoyo y enseñanzas que aún guían mi camino.

# AGRADECIMIENTO

Quisiera expresar mi más sincero agradecimiento a mi tutora, PhD. Lorena Recalde, por su invaluable orientación, paciencia y apoyo durante todo el proceso. Su experiencia y consejos fueron fundamentales para el desarrollo de este trabajo.

A mis padres, por su amor incondicional y por siempre creer en mí. Su apoyo ha sido mi mayor fortaleza.

A mis amigos de la universidad, por su constante ánimo y por hacer este camino más llevadero con su compañía y motivación.

A mis profesores, por compartir su conocimiento y por su dedicación a lo largo de mi formación académica. Sus enseñanzas han sido una fuente de inspiración.

A mis compañeros de tesis, por su colaboración y apoyo incondicional. Juntos superamos desafíos y celebramos los logros de este proyecto.

# ÍNDICE DE CONTENIDO

# ÍNDICE DE FIGURAS

# ÍNDICE DE TABLAS

# RESUMEN

CENTINELA es un proyecto que se presenta como una solución para los desafíos de comunicación y colaboración que enfrentan los investigadores ecuatorianos. El proyecto se centra en la integración de las funcionalidades de motor de búsqueda y visualización de redes de coautoría del sistema ResNet, con las funcionalidades de consenso y recomendación para la formación de grupos de investigación del sistema ResearchDecide.

Al contar con las funcionalidades de ambos sistemas, CENTINELA se postula como una herramienta robusta en la que los investigadores tendrán acceso a información relevante y podrán colaborar activamente en sus propios grupos de investigación. Para el desarrollo del sistema se emplearon las tecnologías Angular para el frontend, que unifica las funcionalidades de ResNet y ResearchDecide. Por otro lado, para el backend se utilizó Django Rest Framework, asegurando la escalabilidad del sistema.

En el proyecto, además, se incluye el desarrollo del Componente D, denominado “Módulo de Analítica: Dashboards y estadísticas para la toma de decisiones”, que proporcionará dashboards a través del seguimiento de la metodología Noetix integrada en SCRUM. Este componente permitirá que tanto investigadores como instituciones puedan evaluar el rendimiento de sus propias investigaciones, obteniendo métricas que les sirvan como guía para tomar decisiones informadas respecto a sus futuras investigaciones.

***PALABRAS CLAVE -*** investigación, visualización de datos, redes de coautoría, dashboard, colaboración científica, Ecuador

# ABSTRACT

CENTINELA is a project that presents itself as a solution to the communication and collaboration challenges faced by Ecuadorian researchers. The project focuses on integrating the search engine and co-authorship network visualization functionalities of the ResNet system with the consensus and recommendation functionalities for forming research groups of the ResearchDecide system.

With the functionalities of both systems, CENTINELA positions itself as a robust tool where researchers will have access to relevant information and can actively collaborate within their research groups. For the system's development, Angular was used for the frontend, which unifies the functionalities of ResNet and ResearchDecide. On the other hand, Django Rest Framework was used for the backend, ensuring the system's scalability.

Additionally, the project includes the development of Component D, called "Analytics Module: Dashboards and statistics for decision making", which will provide dashboards through the Noetix methodology integrated with SCRUM. This component will allow both researchers and institutions to evaluate the performance of their own research, obtaining metrics to guide informed decision-making regarding future research.

***KEYWORDS -*** research, data visualization, co-authorship networks, dashboard, scientific collaboration, Ecuador

# 1. INTRODUCCIÓN

## 1.1. Planteamiento del problema

La investigación científica en Ecuador ha crecido significativamente en los últimos años, pues según [1], el país ha tenido una producción de más de 29000 publicaciones, lo que ha hecho que desde 2015, Ecuador se haya convertido en uno de los países líderes de crecimiento en producción científica en América Latina.

Para afrontar el aumento de la investigación científica y ofrecer soluciones a los desafíos que conlleva la investigación en el país, surge el interés de desarrollar un proyecto en el que se fomente la colaboración entre investigadores.

A pesar de la existencia de la plataforma REDI, los investigadores aún experimentan dificultades para acceder a información trascendental para su investigación, esto debido en gran medida a que no existe interoperabilidad entre plataformas, lo que ocasiona problemas de comunicación y colaboración entre investigadores.

El proyecto actual propone integrar las plataformas ResNet y ReSearchDecide. ResNet se enfoca en mostrar las redes de coautoría de investigadores usando la información presente en Scopus [2], mientras que ReSearchDecide se enfoca en la creación de grupos de investigadores, apoyar la toma de decisiones y recomendar temas de investigación basados en las preferencias de los autores [3] [4].

La integración de ambos sistemas busca optimizar recursos y fomentar una colaboración efectiva entre investigadores en una sola plataforma llamada CENTINELA. Además, pretende mejorar la calidad de la investigación mediante recomendaciones precisas y mantener a los investigadores actualizados con las últimas tendencias.

La Figura 1.1 presenta el diagrama general de la plataforma CENTINELA, donde se pueden distinguir los componentes A, B, C y D.

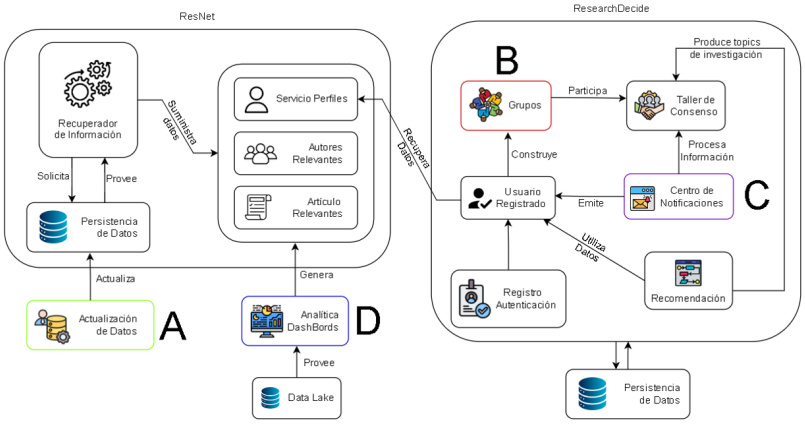



Figura 1.1: Diagrama general del Sistema Centinela unificando los sistemas ResNet, ReSearchDecide y sus respectivos componentes A, B, C y D.

En cuanto al presente proyecto, además de la integración de las plataformas ResNet y ReSearchDecide, se plantea desarrollar el componente D, denominado “Módulo de Analítica: Dashboards y estadísticas para la toma de decisiones”, el cual busca la implementación de Dashboards para la representación visual de los datos, lo que proporcionará a los investigadores una percepción visual y descriptiva de los datos relevantes para sus campos de investigación.

El módulo de Analítica permitirá a los investigadores evaluar el impacto de sus propias investigaciones y proyectos, proporcionando métricas clave y estadísticas relevantes a nivel nacional que sirvan al investigador como una guía para realizar investigaciones futuras.

Estas herramientas visuales y analíticas no solo facilitarán la toma de decisiones estratégicas a nivel individual, sino que también serán valiosas para las instituciones al evaluar el rendimiento general de la investigación y asignar recursos de manera más informada.

## 1.2. Justificación teórica

Con el crecimiento de la investigación científica en Ecuador es fundamental contar con una plataforma que resuelva los desafíos de comunicación y colaboración a los que se enfrentan los investigadores.

CENTINELA se presenta como una solución, ya que al contar con las funcionalidades de redes de coautoría y de redes sociales, será una plataforma unificada en la que el investigador puede encontrar información, comunicarse y colaborar activamente con su grupo de investigación.

Por otro lado, manejar grandes volúmenes de datos requiere un conocimiento especia-

lizado en herramientas de procesamiento, lo cual representa un desafío significativo [5]. En este contexto, las representaciones visuales de datos son fundamentales, ya que permiten identificar agrupaciones, relaciones y tendencias que facilitan la interpretación [6]. La visualización de datos no solo apoya el análisis, sino que también permite explorar y descubrir nuevo conocimiento [7]. Por lo tanto, la inclusión de representaciones gráficas, como los dashboards, en CENTINELA será de gran utilidad para los usuarios, ya que les proporcionará información visualmente accesible, facilitando así su análisis e interpretación [8].

## 1.3. Justificación metodológica

Las metodologías utilizadas para el desarrollo de CENTINELA y el componente D serán SCRUM y la metodología Noetix para desarrollo de dashboards. SCRUM fue elegida porque su enfoque brinda flexibilidad para adaptarse a los cambios en los requisitos, además de que permite que el proyecto se avance en incrementos que se entregan al final de cada sprint. Por otro lado, la metodología Noetix fue elegida debido a que sus fases se pueden ajustar perfectamente al marco SCRUM a través de historias de usuario.

## 1.4. Justificación práctica

CENTINELA será una plataforma integral diseñada para investigadores ecuatorianos, proporcionando acceso centralizado a todos los recursos concernientes a datos académicos. Esto contribuirá en que se fomente la investigación en el país y se mejore la eficiencia y calidad de investigación, impulsando el desarrollo científico y tecnológico en Ecuador.

## 1.5. Objetivos

### 1.5.1. Objetivo general

Integrar los sistemas ResNet y ReSerchDecide en una sola plataforma llamada CENTINELA, en la que además se implemente el módulo de Analítica para que la plataforma cuente con representaciones visuales de los datos.

### 1.5.2. Objetivos específicos

1. Analizar y comprender a detalle los sistemas ResNet y ReSerchDecide para identificar sus funcionalidades, requerimientos y posibles áreas de mejora.

2. Diseñar e implementar una arquitectura de integración que permita la interoperabilidad

entre los diferentes sistemas, garantizando una coherencia de datos y compatibilidad de las plataformas.

3. Comprender el estado del arte en cuanto a formas de representación visual de datos.
4. Desarrollar el módulo de analítica: Dashboards y estadísticas para la toma de decisiones.
5. Evaluar las funcionalidades implementadas del módulo de analítica.

## 1.6. Alcance

El alcance principal del proyecto se enfoca en desarrollar CENTINELA, basado en la integración de los sistemas ResNet y ResearchDecide, para luego servir como base del desarrollo de los otros componentes de dicho proyecto. El objetivo principal es asegurar que ambas plataformas operen como una sola en una renovada y única interfaz de usuario. Esto implica la unificación de la experiencia del usuario y la optimización de la eficiencia operativa al consolidar funcionalidades clave en un solo punto de acceso. Además se incluirán dashboards y otras representaciones gráficas que muestren información académica del país en general, de todas las afiliaciones, tópicos y autores presentes en la información extraída de Scopus.

## 1.7. Marco Teórico

### 1.7.1. Visualización de datos

La visualización de datos es el proceso de representación de datos en formato gráfico, de tal manera que se facilite el análisis e interpretación de datos grandes y complejos, lo que lo convierte en un medio eficiente para la transmisión de conceptos [8]. El objetivo de la visualización de datos es el de comunicar de manera efectiva la información en un modo visual revelando patrones y tendencias que no son fáciles de ver en datos tabulares o texto y que podrían facilitar la comprensión y la toma de decisiones. La visualización de datos debe cumplir con [9]:

- Indicaciones claras de cómo se relacionan o comparan los valores mostrados entre sí.
- Representar cantidades de forma precisa.

- Facilitar la comparación entre cantidades.
- Facilitar la detección de máximos y mínimos.
- Ser claro en los objetivos y usos de la visualización.

### 1.7.2. Métodos de visualización de datos

Los métodos de visualización de datos son técnicas o herramientas que permiten representar datos de manera visual y facilitan la interpretación de patrones, tendencias y relaciones en un conjunto de datos.

#### Gráfico de líneas

En este tipo de gráfico como se observa en la Figura 1.2, se busca mostrar la evolución de una variable a lo largo del tiempo o la relación que existe con otra variable, por esta razón sirve como una manera eficaz de comparación entre varias variables pues las líneas de apilamiento se pueden utilizar para mostrar tendencias de varias variables al mismo tiempo [10].

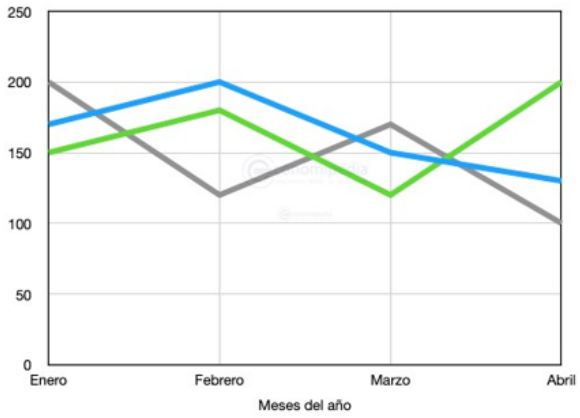



Figura 1.2: Gráfico de líneas.

#### Gráfico de barras

Como se observa en la Figura 1.3, este tipo de gráfico representa la magnitud de una categoría usando barras que pueden ser horizontales o verticales [10]. Este tipo de gráfico resulta útil para comparar cantidades en diferentes grupos.

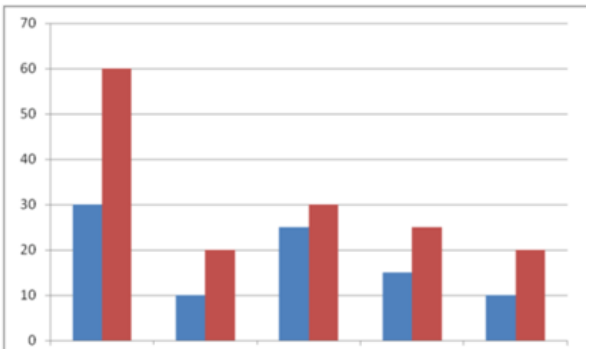



Figura 1.3: Gráfico de barras.

### Gráfico de mapa de árbol

Este tipo de gráfico se caracteriza por presentar los datos de forma jerárquica por medio de la división del área de visualización en rectángulos [11], como se observa en la Figura 1.4.

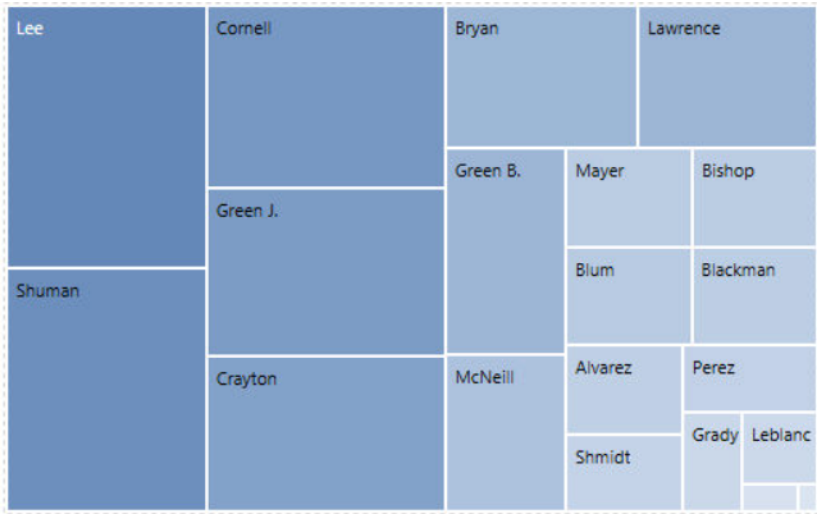



Figura 1.4: Gráfico de mapa de árbol.

### Mapa de coropletas

Este tipo de gráfico destaca porque se usa una escala de un mismo color sobre un mapa para representar datos cuantitativos relativos, como ratios o densidades [12], como se muestra en la figura 1.5.

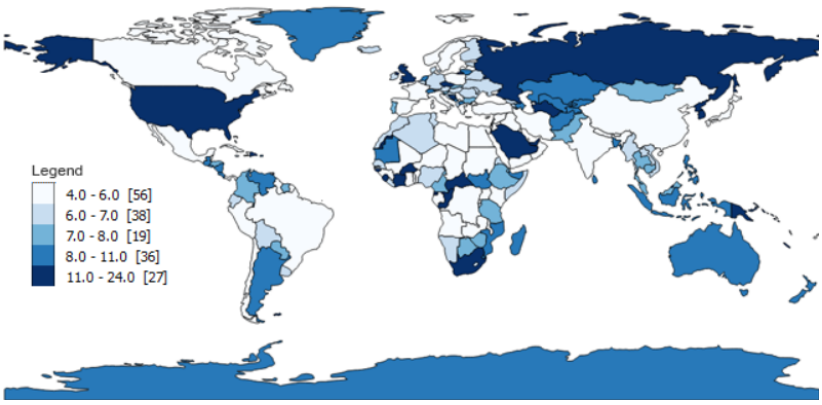



Figura 1.5: Mapa de coropletas.

### 1.7.3. Dashboard

Los dashboard son herramientas que permiten agrupar, centralizar, compartir y presentar de manera visual los datos relacionados con una organización, lo que facilitará la toma de decisiones [13]. Un dashboard o panel de control es una interfaz gráfica que proporciona una forma rápida y resumida de visualización de datos clave o relevantes para una actividad específica. Los dashboard son diseñados para ofrecer una vista de los datos que sea de fácil interpretación y así permitir que los usuarios puedan tomar decisiones informadas. Los dashboard cuentan con distintos elementos visuales, entre ellos se suelen encontrar gráficos, tablas, indicadores de estado, mapas y otros componentes gráficos que presenten información de forma clara y concisa como se muestra en la Figura 1.6.


Figura 1.6: Dashboard.

## HERRAMIENTAS PARA REALIZAR DASHBOARDS

Entre las herramientas que pueden emplearse para la realización de dashboards se encuentran:

- **Power BI:** es una herramienta de análisis de datos desarrollado por Microsoft, su uso se basa en que el usuario debe ingresar los datos ya sea en forma de hojas de cálculo, bases de datos, documentos de texto, etc., y estos datos se transformarán para eliminar errores y datos redundantes para posteriormente realizar visualizaciones como dashboards [14].
- **Tableau:** es un servicio de visualización de datos desarrollado por Salesforce, su funcionamiento se basa en arrastrar y soltar datos a través de una interfaz intuitiva [15].
- **D3.js:** es una librería de JavaScript diseñada para mostrar información de forma gráfica e interactiva haciendo uso de HTML, SVG y CSS convirtiéndola en una de las herramientas mas potentes para la visualización de datos [16].
- **Dash-Plotly:** es una librería de código abierto de Python, ideal para crear e implementar aplicaciones web analíticas para el análisis de datos, exploración de datos, visualización, modelado, y reportes [17].
- **Chart.js:** es una librería de código abierto de JavaScript que puede emplearse para la visualización de datos en ocho tipos de gráficos estadísticos básicos entre los que se encuentran los gráficos de barras, de líneas y de pastel [18].
- **HighCharts:** es una librería de JavaScript para la creación de gráficos interactivos en aplicaciones web, tiene soporte para mas de 20 tipos de gráficos con distintos tipos de configuraciones [19].
- **Bokeh:** es una librería de código abierto de Python que permite la creación de visualizaciones estadísticas interactivas y de aspecto profesional como diagramas de líneas, barras, histogramas, entre otras [20].

### Arquitectura hexagonal

También conocido como patrón de puertos y adaptadores, es una forma de diseñar software con el fin de crear aplicaciones flexibles y mantenibles. Su enfoque es el de separar

el núcleo de la aplicación de las dependencias externas como almacenes de datos [21]. La arquitectura hexagonal consta de las siguientes capas [22]:

- **Dominio:** en esta capa es donde se encuentra el núcleo de la como lo son las entidades y reglas del negocio.
- **Puertos o Infraestructura:** es donde se alojaran las acciones que pueden realizarse con sistemas externos.
- **Adaptadores o aplicación:** son la implementación de los puertos y actúan como traductores entre el dominio y los medios externos.

# 2. METODOLOGÍA

Las metodologías elegidas para el desarrollo del proyecto son SCRUM y la metodología Noetix “Dashboard Development and Deployment: A Methodology for Success”.

## 2.1. SCRUM

SCRUM es un marco de trabajo ágil y liviano que proporciona una guía para administrar el desarrollo de software, además en este marco se promueve la colaboración, flexibilidad y la entrega continua de valor [23]. En SCRUM se contemplan los roles Product Owner, Scrum Master y Development Team [23]. El Product Owner es quien priorizará el Product Backlog y quien acepta o rechaza el producto. El Scrum Master es el encargado de promover la correcta implementación de SCRUM y eliminar impedimentos. EL Development Team es el encargado de llevar a cabo el trabajo, el cual es distribuido a través de sprints de 1 a 3 semanas, en los que se realizan las tareas que han sido elegidas para formar parte del Sprint Backlog. SCRUM cuenta con los artefactos [23]:

- Product backlog: lista de tareas que han sido aprobadas por el Product Owner.
- Sprint Backlog: conjunto de tareas seleccionadas para desarrollarse en un sprint.
- Incremento: es el conjunto de todo lo desarrollado en un sprint.

## 2.2. Establecimiento de Roles

Para el desarrollo del proyecto CENTINELA se han asignado los siguientes roles:

| **Rol** | **Nombre** |
|---|---|
| Product Owner | Lorena Recalde |
| Scrum Master | Lorena Recalde |
| Development Team | Danny Cabrera, Joffre Cóndor, Rommel Masabanda, Fernando Sangopanta |

Tabla 2.1: Roles y Nombres del Equipo Scrum.

## 2.3. Metodología Noetix

La metodología “Dashboard Development and Deployment: A Methodology for Succes” de la empresa Noetix sirve como una guía para la implementación exitosa de un dashboard, a través de las siguientes fases [24]:

1. **Planificación:** en esta fase se identificarán roles y responsabilidades para los miembros del equipo, se definirán los objetivos generales y el alcance del dashboard, además se identificarán los indicadores a analizar.
2. **Recopilación de requisitos:** en esta fase se realizarán actividades para determinar cuáles son los requisitos del dashboard en base a las necesidades y expectativas de los stakeholders, así también se definirá la presentación y la funcionalidad del dashboard y esto se verá plasmado en prototipos.
3. **Diseño:** en esta fase se refinará la interfaz de usuario y se establecerán las fuentes de datos, estructura de los datos, las consultas para acceder a los datos y drill paths.
4. **Construcción y validación:** en esta fase se realizarán tres tareas:
   - **Implementación del front-end:** en esta tarea se tomarán las decisiones finales de diseño de la interfaz de usuario, como los gráficos, alertas visuales y funciones interactivas que se usarán, una vez definido el diseño final de la interfaz de usuario se la desarrollará.
   - **Implementación de consultas:** en esta tarea se crearán las consultas para recuperar la información necesaria de la base de datos.
   - **Configurar programación, actualización y seguridad:** en esta tarea se configurarán las consultas para que se ejecuten regularmente y actualicen el dashboard, además se implementarán reglas de seguridad para mostrar información adecuada según el usuario.
   - **Validación del dashboard:** una vez se completó el desarrollo del dashboard se lo deberá probar para asegurar que cumple con los objetivos especificados.
5. **Despliegue:** con el dashboard construido y probado ya puede llevarse a producción donde se implementarán requisitos de seguridad en el entorno de producción y se realizará la integración en una red corporativa.

6. **Mantenimiento:** en esta fase es donde se tomarán las medidas necesarias para poder mantener el dashboard, y permitir mejoras futuras.

## 2.4. Integración de metodologías

SCRUM es una metodología flexible que se puede usar como contenedor de otras metodologías, en este caso en SCRUM se incluirá la metodología "Dashboard Development and Deployment: A Methodology for Success"de Noetix como se muestra en la Tabla 2.2.

| Sprint | Historias de usuario | Descripción |
|---|---|---|
| 1 | ▪ Como miembro del equipo, quiero definir claramente el objetivo del dashboard, para tener una visión clara y objetiva de que se busca mostrar en el dashboard.<br>▪ Como miembro del equipo del proyecto, quiero determinar KPIs importantes, para garantizar que el dashboard cumpla con su objetivo planteado.. | Estas historias de usuario pertenecen a la fase de Planificación de la metodología Noetix. |

| Sprint | Historias de usuario | Descripción |
|---|---|---|
| 2 | ▪ Como miembro del equipo de proyecto, quiero explorar diferentes opciones para la presentación y funcionalidad del dashboard, para proporcionar una experiencia personalizada a los usuarios.<br>▪ Como miembro del equipo del proyecto, quiero identificar la información a mostrarse en el dashboard, para garantizar que los usuarios puedan explorar y analizar los datos de manera eficiente.<br>▪ Como miembro del equipo de proyecto, quiero utilizar herramientas y tecnologías que faciliten el prototipado del dashboard, para validar o descartar elementos o funcionalidades. | Estas historias de usuario pertenecen a la fase de Recopilación de requisitos de la metodología Noetix. |
| 3 | ▪ Como miembro del equipo del proyecto, quiero evaluar la fuente de los datos para planificar el acceso a los datos requeridos por los KPIs.<br>▪ Como miembro del equipo del proyecto, quiero diseñar las consultas que me permitirán extraer información, para poblar una base de datos específica para el consumo del dashboard. | Estas historias de usuario pertenecen a la fase de Diseño de la metodología Noetix. |

| Sprint | Historias de usuario | Descripción |
|---|---|---|
| 4 | ▪ Como miembro del equipo de desarrollo, quiero implementar una base de datos No SQL en la que se almacenen los datos necesarios para la construcción del dashboard, permitiendo así realizar consultas eficientes y proporcionar información actualizada para el dashboard.<br>▪ Como miembro del equipo de desarrollo, quiero implementar dashboards del país, afiliaciones y tópicos, para que todos los usuarios puedan tener a su alcance una forma interactiva de ver la información de investigación | Estas historias de usuario pertenecen a la fase de Construcción y validación de la metodología Noetix. |
| 5 | ▪ Como miembro del equipo de desarrollo, quiero integrar el dashboard en un ambiente de producción, para poder realizar pruebas con usuarios.<br>▪ Como usuario administrador, quiero poder actualizar los datos almacenados en la base de datos No SQL, para que los dashboards siempre presenten información actualizada.. | Estas historias de usuario pertenecen a las fases de Despliegue y Mantenimiento de la metodología Noetix. |

Tabla 2.2: Integración de la metodología Noetix con SCRUM

## 2.5. Gestión del proyecto

Para la gestión del proyecto, utilizamos Azure Boards, un servicio web que optimiza el seguimiento y la organización de proyectos bajo metodologías ágiles, como SCRUM. Azure Boards ofrece herramientas robustas para planificar y visualizar el progreso de las tareas, facilitando la asignación de responsabilidades y el monitoreo del avance del equipo en cada

Sprint.

La Figura 2.1 presenta una primera aproximación del Product Backlog en Azure Boards. Inicialmente, el backlog se desarrolló en forma de historias de usuario épicas y características (features), debido a la incertidumbre existente en el proyecto. A medida que el proyecto avance, se refinará el backlog para adaptarse a los cambios en los requisitos y asegurar una gestión más precisa de las tareas.


Figura 2.1: Product Backlog sin refinar.

## 2.6. Sprint 0

### 2.6.1. Planificación del Sprint 0

#### Objetivos del Sprint

- Definir tecnologías a usar en la implementación del proyecto.
- Definir la arquitectura final del sistema.
- Definir estructura de carpetas para el proyecto.

- Levantar ambiente para el desarrollo del proyecto.

### 2.6.2. Ejecución del Sprint 0

#### Elección de tecnologías

La elección de tecnologías para el desarrollo del proyecto se detallan a continuación en la Tabla 2.3:

| Tecnología | Descripción | Versión |
|---|---|---|
| Docker | Es una plataforma diseñada para desarrollar, implementar y ejecutar aplicaciones en contenedores [25]. | 25.0.3 |
| Angular | Es un framework de aplicaciones web de una sola página desarrollado y mantenido por Google [26]. | 16.2.14 |
| Django | Es un framework para Python que permite la creación de aplicaciones web de forma rápida [27]. | 3.15.1 |
| D3.js | Es una librería gratuita de JavaScript para crear visualizaciones de datos [28]. | 7.9.0 |
| Bootstrap | Es un framework para front-end que facilita la creación de aplicaciones web responsivas [29]. | 5.2.3 |
| Tailwind | Es un framework de CSS que ofrece una gran variedad de clases para construir y personalizar diseños [30]. | v3 |
| Ngx-Charts | Es una biblioteca de gráficos para Angular que cuenta con una amplia variedad de gráficos interactivos y personalizables [31]. | 20.5.0 |

Tabla 2.3: Tabla de tecnologías, descripciones y versiones

---

#### Arquitectura final del sistema

Como se muestra en la Figura 2.2, la arquitectura final de CENTINELA se compone de tres bases de datos, un front-end unificado y dos backends que estarán comunicados.

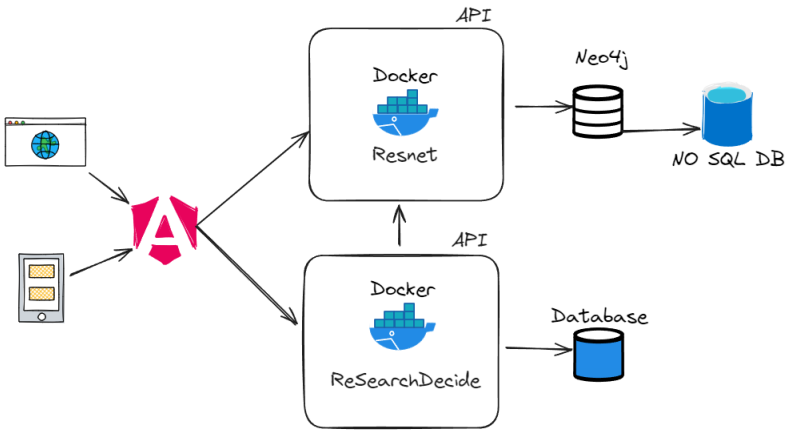



Figura 2.2: Arquitectura Centinela.

El front-end unificado desarrollado en Angular permitirá a los usuarios acceder a las funciones del sistema a través de una interfaz intuitiva y responsiva, asegurando una experiencia de uso de alta calidad independientemente del dispositivo que se utilice.

En cuanto a los dos backends, ambos estarán desarrollados en Django y dockerizados, con funcionalidades distintas basadas en los sistemas ResNet y ResearchDecide respectivamente. El backend basado en ResNet contará con una API REST que se comunicará con dos bases de datos: una base de datos orientada a grafos en Neo4j, que servirá para realizar consultas para el motor de búsqueda, y una base de datos No-SQL en MongoDB, que se utilizará para consultas en el módulo de analítica del sistema. Por su parte, el backend basado en ResearchDecide se comunicará con el otro backend para consultar la información de la base de datos de Neo4j. Además, también contará con una API REST para comunicarse con su propia base de datos, donde se almacenará la información de perfiles de usuario y grupos de investigadores.

### Estructura de carpetas

Para definir la estructura de carpetas se reunió todo el equipo de desarrollo y se eligió un estructura de carpetas hexagonal de dos capas, domain y presentation, las cuales contienen lo siguiente[32]:

- **Domain:** En esta carpeta se incluyen todos los modelos de negocio, casos de uso, interactores y abstracciones de repositorio para la aplicación.

- **Presentation:** En esta carpeta se encontrará todo lo que pertenece a la UI, como los archivos de estilo y componentes.

### Ambiente de desarrollo

Para el levantamiento del ambiente de desarrollo se creó una organización en Github, donde se alojarán cada uno de los repositorios del sistema.

Una vez que se creó la organización, como equipo se llegó a la convención de commits y esta quedó documentada en un repositorio llamado tech-governance, que también incluye documentación de cómo contribuir al proyecto.

En la Figura 2.3, se puede observar que en el repositorio de front-end, además se configuró dependencias iniciales y se empleó la estructura de carpetas ya definida en 2.6.2.

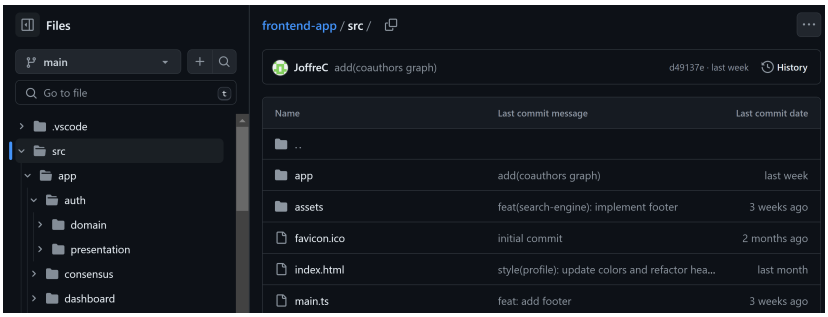



Figura 2.3: Repositorio Front-end.

Con el repositorio de front-end preparado, cada integrante ya podía clonar el repositorio y con eso estar listo para comenzar el desarrollo en su respectiva rama.

## 2.6.3. Revisión y retrospectiva del Sprint 0

El Sprint 0 fue evaluado en base a los criterios de aceptación de las historias de usuario, por lo que se concluyó que el Sprint 0 culminó satisfactoriamente, al haber cumplido con los objetivos.

- Se definió las tecnologías ha usarse en el desarrollo del proyecto.
- Se definió una arquitectura escalable para el sistema.
- Se definió una estructura de carpetas limpia y escalable para el proyecto.
- Se levantó el ambiente de desarrollo.

## 2.7. Sprint 1

### 2.7.1. Objetivos del Sprint

- Diseñar mockups para las pantallas de la aplicación.
- Implementar el diseño de la búsqueda de autores.
- Implementar el enrutamiento para mostrar la página de un autor seleccionado.
- Implementar el enrutamiento para mostrar la página de un artículo seleccionado.
- Completar la fase de Planificación de la metodología Noetix.

### 2.7.2. Sprint Planning

En el Sprint 1 se realizarán las historias de usuario detalladas en la Tabla 2.4.

| **Sprint Backlog 1** |
|---|
| **HU-SE-01: Búsqueda de artículos relevantes** |
| *Como usuario no registrado deseo poder encontrar artículos relevantes dado un tema de investigación para poder acceder rápidamente a información útil y actualizada que apoye mi estudio o trabajo* |
| **Criterios de Aceptación:** |
| - **Resultados de Búsqueda:** Al ingresar un tema de investigación, debo recibir una lista de artículos relevantes. Los resultados deben ser presentados de manera clara, mostrando al menos el título del artículo, autores principales y fecha de publicación. |
| - **Filtros y Ordenación:** Debo poder aplicar filtros para refinar los resultados de búsqueda por criterios como fecha de publicación, tipo de artículo (revista, conferencia, etc.), y relevancia. Debo poder ordenar los resultados por relevancia, fecha de publicación o número de citas. |
| - **Acceso a Detalles del Artículo:** Al hacer clic en un resultado de búsqueda, debo poder ver detalles completos del artículo, incluyendo el resumen completo, la lista completa de autores, afiliaciones y enlaces a la publicación original o al texto completo si está disponible. |
| **Tareas:** |
| T1. Componente para listar artículos. |

| |
|---|
| T2. Paginación para extraer resultados limitados. |
| T3. Componente para filtrar la información por años. |
| T4. Enrutamiento para redirigir hacia la página del artículo. |
| T5. Página para visualizar la información general de un artículo. |
| **HU-SE-02: Búsqueda de autores** |
| *Como usuario no registrado deseo poder ver los investigadores que tengan colaboraciones en artículos con afiliaciones ecuatorianas para mantenerme informado sobre sus investigaciones y campos de estudio.* |
| **Criterios de Aceptación:** |
| - **Resultados de Búsqueda:** Al ingresar un tema de investigación, debo recibir una lista de artículos relevantes. Los resultados deben ser presentados de manera clara, mostrando al menos el título del artículo, autores principales y fecha de publicación. |
| - **Filtros y Ordenación:** Debo poder aplicar filtros para refinar los resultados de búsqueda por criterios como fecha de publicación, tipo de artículo (revista, conferencia, etc.) y relevancia. Debo poder ordenar los resultados por relevancia, fecha de publicación o número de citas. |
| - **Acceso a Detalles del Artículo:** Al hacer clic en un resultado de búsqueda, debo poder ver detalles completos del artículo, incluyendo el resumen completo, la lista completa de autores, afiliaciones y enlaces a la publicación original o al texto completo si está disponible. |
| **Tareas:** |
| T1. Desarrollar la interfaz para poder seleccionar distintos tipos de búsqueda. |
| T2. Componente para poder visualizar la lista de autores que serán resultado de la búsqueda. |
| T3. Página para el perfil de investigador. |
| T4. Restricción para usuarios no registrados. |
| T5. Componente de paginación para extraer resultados limitados. |
| T6. Enrutamiento para dirigirse a la página de un autor seleccionado. |
| **HU-MA-01: Definir el objetivo del dashboard** |
| *Como miembro del equipo, quiero definir claramente el objetivo del dashboard, para tener una visión clara y objetiva de qué se busca mostrar en el dashboard.* |
| **Criterios de Aceptación:** |
| - **Objetivo del dashboard:** Se ha definido el objetivo del dashboard. |

| **Tareas:** |
| --- |
| T1. Redactar el objetivo del dashboard. |
| **HU-MA-02: Definir KPIs** |
| *Como miembro del equipo del proyecto, quiero determinar KPIs importantes, para garantizar que el dashboard cumpla con su objetivo planteado.* |
| **Criterios de Aceptación:** |
| - **Identificación de KPIs:** Se han identificado los KPIs para la realización del dashboard. |
| **Tareas:** |
| T1. Identificar los KPIs para el dashboard. |

Tabla 2.4: Historias de Usuario del Sprint 1

### 2.7.3. Ejecución del Sprint

#### Diseño de mockups

Para asegurar que el sistema cumpla con los requerimientos y arquitectura de un único front-end, se llevará a cabo el diseño de mockups, los cuales fueron realizados en la herramienta Figma y sirvieron como base para el desarrollo del aplicativo en su versión móvil y web. El resultado de los mockups puede verse en Anexos Mockups 6.1.

#### Diseño de la búsqueda de autores

Para la búsqueda de autores, en la pantalla home se incluyó una barra de búsqueda con un seleccionador de tipo de búsqueda, en el que se podrá elegir la búsqueda de autores (Author) como se muestra en la Figura 2.4.

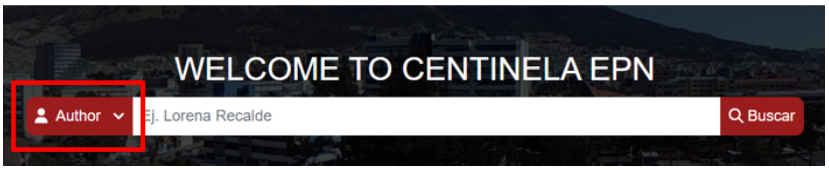



Figura 2.4: Barra de búsqueda con seleccionador en Author.

Con el seleccionador implementado, se diseñó la tabla de los autores resultantes de una búsqueda en la que se muestra *Name, Affiliation, Articles, Topics, Affiliations, Citations y Status*, como se muestra en la Figura 2.5

| Name | Affiliation | Articles | Topics | Affiliations | Citations | Status |
|---|---|---|---|---|---|---|
| Lorena R. Pérez-Floriano | Universidad Diego Portales | 4 | 21 | 6 | 1995 | updated |
| Lorena Isabel Barona López | | 16 | 39 | 3 | 396 | updated |
| Lorena Maribel Meneses-Olmedo | | 4 | 26 | 2 | 288 | updated |
| Lorena Maniguaje Rincón | Instituto Nacional de Pesquisas Da Amazonia | 5 | 8 | 1 | 145 | updated |
| Lorena Recalde | Escuela Politécnica Nacional | 5 | 31 | 2 | 54 | updated |
| Lorena Jaramillo-Mediavilla | Universidad Técnica del Norte | 1 | 8 | 1 | 1 | updated |
| Brenda Lorena Pillajo Sánchez | Pontificia Universidad Católica del Ecuador | 1 | 6 | 2 | 0 | updated |

Figura 2.5: Tabla de resultados para la búsqueda de autor.

## Enrutamiento para ingreso a perfil

Cuando un autor es seleccionado en la tabla de resultados de la Figura 2.6, se debe redireccionar al perfil del autor, para esto se usó una función de TypeScript que lleva al perfil del autor en base al *scopus_id*, esta función irá ligada a cada fila de un autor, en la Figura 2.6 se puede ver el código de esta implementación.

```
<tr
  mat-row
  *matRowDef="let row; columns: displayedColumns"
  class="hover:cursor-pointer hover:bg-gray-200"
  (click)="goToAuthor(row.scopus_id)"
></tr>
```

```
goToAuthor(scopus_id: string) :void {
  const url :string  = this.router.serializeUrl(this.router.createUrlTree( commands: ['/profile', scopus_id]));
  window.open(url, target: '_blank');
}
```

Figura 2.6: Routing para perfil.

## Enrutamiento para mostrar la página de artículos

En los resultados de una búsqueda de artículos relevantes, al seleccionar uno de los resultados, se deberá redirigir a la página en la que se muestra la información del artículo. Para lograr esto se desarrolló una función de TypeScript que lleva a la página de información de un artículo a través de su *scopus_id*, como se muestra en la Figura 2.7.

```
<tr
  mat-row
  *matRowDef="let row; columns: displayedColumns"
  (click)="seeMoreInformation(row.scopus_id)"
  class="hover:cursor-pointer hover:bg-gray-200"
></tr>
```

```
seeMoreInformation(scopusId: string) :void  {
  const url :string  = this.router.serializeUrl(
    this.router.createUrlTree( commands: ['home/article', scopusId])
  );
  window.open(url, target: '_blank');
}
```

Figura 2.7: Routing para artículo.

## Fase de Planificación de la Metodología Noetix

### Objetivo del dashboard

El objetivo del dashboard será el de proporcionar una visualización clara e interactiva de la evolución histórica de artículos, autores, afiliaciones y tópicos científicos en Ecuador. El dashboard permitirá que investigadores y otros usuarios interesados puedan explorar tendencias y patrones en la contribución científica del país, facilitando la toma de decisiones y el impulso de futuras investigaciones.

**Definición de KPIs**

El KPI (Indicador clave de rendimiento) identificado es el número de artículos publicados, ya que este indicador refleja directamente la producción científica y permite realizar un análisis detallado a niveles país, provincia, afiliación, tópico y autor. Esta medida permite tener una visión integral de cómo se distribuye y evoluciona la actividad científica en Ecuador, facilitando la identificación de áreas de fortaleza y oportunidades de mejora en el ámbito científico.

### 2.7.4. Sprint Retrospective

Con la evaluación de los criterios de aceptación de las historias de usuario asignadas al sprint, se concluye que el Sprint 1 culminó satisfactoriamente al haber cumplido con los objetivos.

- Se diseñaron los mockups para el aplicativo.
- Se implementó el diseño para la búsqueda de autores.
- Se implementó el enrutamiento para mostrar la página de un autor seleccionado.
- Se implementó el enrutamiento para mostrar la página de un artículo seleccionado.
- Se completó la fase de Planificación de la metodología Noetix.

## 2.8. Sprint 2

### 2.8.1. Objetivos del Sprint

- Implementar los grafos de coautoría para un autor.
- Implementar la pestaña en la que se mostrarán los artículos de un autor.
- Completar la fase de Recopilación de requisitos de la Metodología Noetix.

### 2.8.2. Sprint Planning

En el Sprint 2 se realizaran las historias de usuario detalladas en la Tabla 2.5.

| **Sprint Backlog 2** |
|---|
| **HU-SE-03: Red de coautoría** |
| *Como usuario no registrado, deseo poder ver la red de coautoría de un autor para visualizar un grafo con los autores con los que ha colaborado, así como la fuerza de esas colaboraciones.* |
| **Criterios de Aceptación:** |
| - **Visualización del Grafo de coautoría:** El usuario no registrado debe poder acceder a una página o sección donde se muestre un grafo interactivo de la red de coautoría de un autor específico. El grafo debe mostrar nodos que representen a los autores y enlaces que representen las colaboraciones entre ellos. |
| - **Información del Nodo:** Al pasar el cursor sobre un nodo (autor) en el grafo, debe aparecer información básica sobre el autor, como su nombre completo y afiliación. Al hacer clic en un nodo, se debe mostrar un panel con detalles adicionales del autor, como el número de publicaciones y las áreas de investigación principales. |
| - **Fuerza de la Colaboración:** Los enlaces entre nodos deben variar en grosor o color para indicar la fuerza de la colaboración, basada en el número de publicaciones conjuntas o la duración de la colaboración. Debe haber una leyenda que explique el significado de los diferentes grosores o colores de los enlaces. |
| - **Interactividad:** El grafo debe ser interactivo, permitiendo al usuario hacer zoom y desplazar la vista para explorar diferentes partes de la red de coautoría. El usuario debe poder filtrar las colaboraciones mostradas por año o relevancia. |
| **Tareas:** |
| T1. Diseñar e implementar la interfaz y el grafo interactivo. |
| T2. Definir y aplicar un esquema visual para la fuerza de la colaboración. |
| T3. Implementar funcionalidades interactivas y filtros. |
| T4. Agregar un panel de detalles con información adicional sobre el autor. |
| **HU-SE-04: Artículos de un investigador** |
| *Como usuario no registrado, quiero poder ver los artículos de un investigador para conocer su trabajo y las publicaciones en las que ha contribuido.* |
| **Criterios de Aceptación:** |

| |
|---|
| - **Visualización de Artículos:** El usuario debe poder acceder a una página o sección donde se muestren los artículos de un investigador específico. La lista de artículos debe incluir el título del artículo y la fecha de publicación. |
| - **Detalle del Artículo:** Al hacer clic en un artículo de la lista, el usuario debe ser redirigido a una página de detalles que muestre información adicional del artículo, como el resumen, los coautores, y enlaces al texto completo si están disponibles. |
| **Tareas:** |
| T1. Diseñar e implementar la interfaz y el grafo interactivo. |
| T2. Definir y aplicar un esquema visual para la fuerza de la colaboración. |
| T3. Implementar funcionalidades interactivas y filtros. |
| T4. Agregar un panel de detalles con información adicional sobre el autor. |
| **HU-MA-03: Explorar opciones** |
| *Como miembro del equipo del proyecto, quiero explorar diferentes opciones para la presentación y funcionalidad del dashboard, para proporcionar una experiencia personalizada a los usuarios.* |
| **Criterios de Aceptación:** |
| - **Tecnologías:** Se han definido las tecnologías a utilizarse para desarrollar el dashboard. |
| - **Gráficos:** Se ha definido qué clase de gráficos se emplearán en el dashboard. |
| **Tareas:** |
| T1. Definir tecnologías para implementar el dashboard. |
| T2. Definir gráficos que se encontrarán en el dashboard. |
| **HU-MA-04: Identificar elementos** |
| *Como miembro del equipo del proyecto, quiero identificar la información a mostrarse en el dashboard, para garantizar que los usuarios puedan explorar y analizar los datos de manera eficiente.* |
| **Criterios de Aceptación:** |
| - **Definir elementos a mostrarse:** Se han definido los elementos que se mostrarán en el dashboard. |
| **Tareas:** |
| T1. Definir elementos del dashboard. |
| **HU-MA-05: Prototipar dashboard** |

| |
|---|
| *Como miembro del equipo del proyecto, quiero utilizar herramientas y tecnologías que faciliten el prototipado del dashboard, para validar o descartar elementos o funcionalidades.* |
| **Criterios de Aceptación:** |
| - **Construcción de prototipo:** Se ha construido un prototipo de dashboard. |
| **Tareas:** |
| T1. Construir prototipo de dashboard. |

Tabla 2.5: Historias de Usuario del Sprint 2

### 2.8.3. Ejecución del Sprint

#### Implementación de grafos de coautoría.

Para representar la red de coautoría de un autor se emplearán grafos en los que cada nodo representará a un autor y las aristas la fuerza de colaboración entre ellos. Para el desarrollo de los grafos fue necesario usar la librería D3.js y una serie de directivas para las funciones de arrastrar y de zoom. La Figura 2.8 muestra la implementación del grafo para representar una red de coautoría.

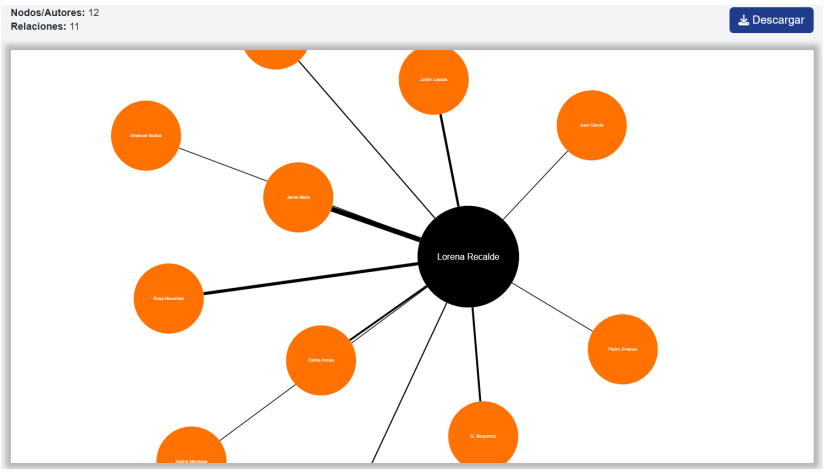



Figura 2.8: Red de coautoría.

Como se muestra en la Figura 2.9, para cada nodo se implementó un popover que muestra el nombre y el grado del autor. Además, en el nombre se implementó redireccionamiento para que se pueda ir al perfil del autor que se seleccione.

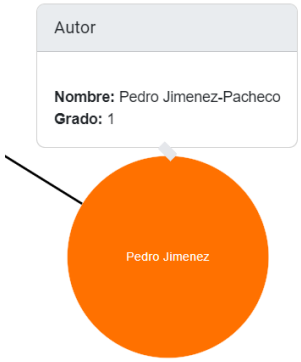



Figura 2.9: Popover en red de coautoría.

### Implementación de la pestaña de artículos de autor.

Para mostrar los artículos de un autor en el perfil, se implementó una tabla en la que se mostrará la información *title* y *publication date*, como se muestra en la Figura 2.10.

| Title | Publicaction date |
|---|---|
| Women in politics and their presence in twitter: Argentina as a case study | 2019-04-01 |
| Systematic Mapping on Embedded Semantic Markup Validated with Data Mining Techniques | 2021-01-01 |
| What kind of content are you prone to tweet? Multi-topic preference model for tweeters | 2020-01-01 |
| A First Spotlight: Introducing Educational Robotics in the Ecuadorian Public School | 2021-03-01 |
| Making open educational resources discoverable: A JSON-LD generator for OER semantic annotation | 2021-07-28 |
| Cognitive systems for urban planning: A literature review | 2020-01-01 |
| Collaboration-based Urban Planning Platform: Modeling Cognition to Co-create Cities | 2020-04-01 |
| Graph-based analysis of housing needs: Towards a citizen-oriented urban planning approach | 2019-11-01 |
| Analyzing embedded semantic with JSON-LD and microdata for educational resources in large scale web datasets | 2019-12-01 |
| Framing the Cacerolazo: An Analysis of a Social Protest in Ecuador | 2022-04-01 |
| Finding the appropriate housing: A fuzzy-model-based recommender system | 2021-07-28 |

Figura 2.10: Tabla de artículos de un autor.

### Fase de Recopilación de requisitos de la Metodología Noetix.

#### Explorar opciones

Se exploraron diversas tecnologías y tipos de gráficos para el proyecto. Entre las opciones tecnológicas para crear gráficos, se consideraron ngx-charts, D3.js, chart.js, highcharts y bokeh. Se seleccionaron ngx-charts y D3.js por su fácil integración con Angular y su implementación sencilla. En cuanto a los tipos de gráficos, se evaluaron nubes de palabras, mapas de coropletas, diagramas de barras, gráficos de líneas de tendencia, mapas de árbol y diagramas de pastel. Finalmente, se descartaron los diagramas de pastel.

### Identificar elementos

En base al KPI de número de artículos, el dashboard incluirá los siguientes elementos:

- Resumen anual de artículos: Se utilizará un gráfico de línea de tendencia para visualizar claramente el número de artículos producidos en el país cada año.
- Afiliaciones con más artículos: Un diagrama de barras mostrará una comparación directa entre las distintas afiliaciones.
- Temas con más artículos: Se empleará un mapa de árbol para visualizar de manera clara la contribución científica de cada tema.
- Número de artículos por provincia: Un mapa de coropletas reflejará la intensidad del color según el número de artículos en cada provincia del Ecuador.

### Prototipar dashboard

Para este prototipo se usó ngx-charts para las gráficas de líneas de tendencia, diagrama de barras y mapa de árbol. Para el mapa de coropletas de Ecuador se uso D3.js, para lo cual fue necesario buscar un archivo geojson que contenga las provincias y sus correspondientes coordenadas para que D3.js pueda dibujar el mapa y asignar un color, la función para crear el mapa se encuentra en la Figura 2.11.

```
private drawMap(data: any, articlesData: any): void {
  const projection :d3.GeoProjection  = d3.geoMercator()
    .center( point: [this.x, this.y])
    .scale(this.scale);
  const geoGenerator :d3.GeoPath<any, d3.GeoPermissi...  = d3.geoPath().projection(projection);
  const articleMap :Map<string, number>  = new Map<string, number>(articlesData.map((d: any) => [d.province_name, d.total_articles]));
  const colorScale :d3.ScaleSequential<string, nev...  = d3.scaleSequential(d3.interpolateBlues)
    .domain( domain: [1000, 30000]);
  this.g.selectAll('path')
    .data(data.features)
    .join('path')
    .attr('d', geoGenerator)
    .attr('fill', (d: any) => {
      const totalArticles :number  = articleMap.get(d.properties.dpa_despro) || 0;
      return colorScale( value: totalArticles + 10000);
    })
    .append('title')
}
```

Figura 2.11: Función para crear mapa de coropletas.

Como se puede observar en la Figura 2.12, la disposición de los gráficos en el dashboard se ha diseñado para optimizar la claridad y la comprensión de los datos presentados.

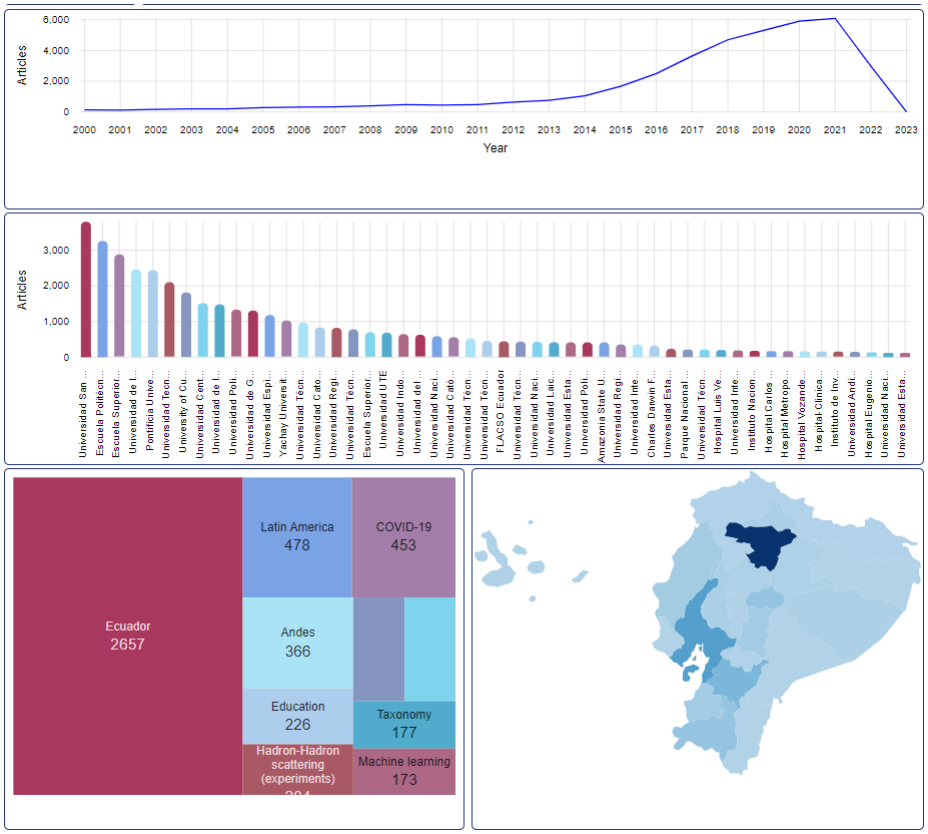



Figura 2.12: Prototipo dashboard.

En la parte superior, se ubica el gráfico de línea de tendencia, que ofrece una visión anual del número de artículos, permitiendo una rápida identificación de patrones temporales y tendencias. Esta posición prominente resalta la importancia de la evolución temporal de la producción científica.

En la sección media, se encuentra el diagrama de barras, que facilita una comparación directa y clara entre las afiliaciones con más artículos. Su ubicación central destaca las diferencias entre instituciones, haciendo evidente cuál es la más productiva.

Finalmente, en la parte inferior, el mapa de árbol y el mapa de coropletas comparten una fila. El mapa de árbol permite visualizar de manera clara y jerárquica la contribución de cada tema, mientras que el mapa de coropletas proporciona un contexto geográfico de la distribución de artículos por provincia. Al compartir espacio, ambos gráficos complementan la comprensión de cómo se distribuye la producción científica tanto temática como geográficamente.

Esta organización busca guiar al usuario de lo general a lo específico, ofreciendo primero una visión global y temporal, luego institucional, y finalmente temática y geográfica.

### 2.8.4. Revisión y retrospectiva del Sprint 2

Con la evaluación de los criterios de aceptación de las historias de usuario asignadas al sprint, se concluye que el Sprint 2 culminó satisfactoriamente al haber cumplido con los objetivos.

- Se implementó los grafos para las redes de coautoría para un autor
- Se implementó la pestaña en la que se mostraran los artículos de un autor.
- Se completó la fase de Recopilación de requisitos de la Metodología Noetix.

## 2.9. Sprint 3

### 2.9.1. Objetivos del Sprint

- Implementar el diseño de la búsqueda de autores relevantes.
- Completar la Fase Diseño de la metodología Noetix.

### 2.9.2. Sprint Planning

En el Sprint 3 se realizaran las historias de usuario detalladas en la Tabla 2.6.

| **Sprint Backlog 3** |
|---|
| **HU-SE-09: Búsqueda de autores relevantes** |
| *Como usuario no registrado, quiero poder ver los autores relevantes a un tópico para poder informarme sobre cómo se han organizado los investigadores para realizar contribuciones en un tópico.* |
| **Criterios de Aceptación:** |
| - **Resultados de Búsqueda:** Al ingresar un tema de investigación (tópico), el usuario debe observar un conjunto de grafos que muestren las comunidades que han contribuido al tópico. |
| - **Filtros y número de resultados:** Debo poder aplicar filtros de afiliación y ajustar el número de resultados que obtengo en la búsqueda. |
| **Tareas:** |

| |
|---|
| T1. Diseñar e implementar la interfaz para la presentación de los resultados de búsqueda. |
| T2. Diseñar e implementar los filtros de afiliaciones y de número de resultados. |
| **HU-MA-06: Evaluación fuente de datos** |
| *Como miembro del equipo del proyecto, quiero evaluar la fuente de los datos para planificar el acceso a los datos requeridos por los KPIs.* |
| **Criterios de Aceptación:** |
| - **Evaluar las entidades de la base de datos de Neo4j:** Se han evaluado las propiedades y relaciones de las entidades que tiene la base de datos de Neo4j. |
| **Tareas:** |
| T1. Realizar consultas a la base de datos de Neo4j para ver el contenido de sus entidades. |
| **HU-MA-07: Consultas** |
| *Como miembro del equipo del proyecto, quiero diseñar las consultas que me permitirán extraer información para poblar una base de datos específica para el consumo del dashboard.* |
| **Criterios de Aceptación:** |
| - **Consultas diseñadas:** Se han diseñado las consultas que permitirán poblar la base de datos dedicada para el dashboard. |
| **Tareas:** |
| T1. Diseñar las consultas para poblar la base de datos dedicada al dashboard. |

Tabla 2.6: Historias de Usuario del Sprint 3

### 2.9.3. Ejecución del Sprint

#### Implementación de la búsqueda de autores relevantes.

Para la búsqueda de autores relevantes se incluyó la opción *Relevant Authors* en la barra de búsqueda de la pantalla home como se observa en la Figura 2.13.

Con el seleccionador implementado, se diseñó la respuesta a la búsqueda, en este caso sería un conjunto de grafos simbolizando comunidades de autores relacionados al tópico buscado, como se observa en la Figura 2.14.

Como se observa en la Figura 2.15 para esta búsqueda se implementó el apartado de

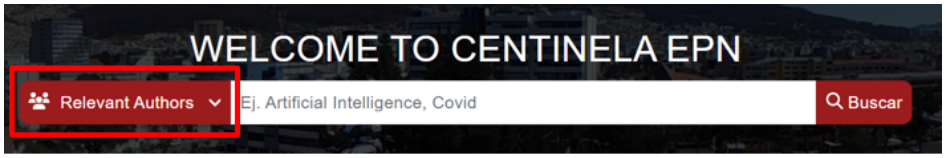



Figura 2.13: Barra de búsqueda con seleccionador en Relevant Authors.

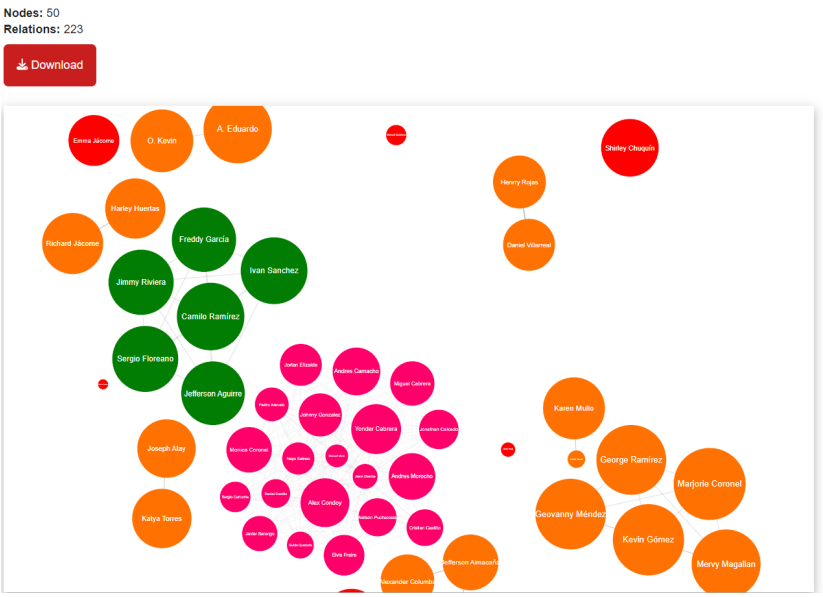



Figura 2.14: Comunidades resultantes de búsqueda de Relevant Authors.

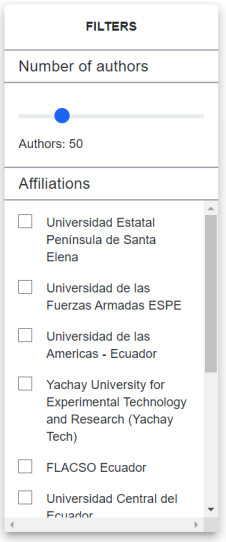



Figura 2.15: Filtros para el resultado de la búsqueda de Relevant Authors.

filtros en el que el usuario podrá ajustar el número de autores que se muestran, y mostrar los autores pertenecientes a la o las afiliaciones que se seleccionen.

## Fase de Diseño de la Metodología Noetix.

### Evaluación fuente de datos

Para obtener los datos necesarios para la realización de las gráficas estadísticas para el dashboard, se realizó un análisis de la base de datos Neo4j, haciendo especial énfasis en la obtención del KPI número de artículos por autor, afiliación, topic, provincia y de forma global del país.

La Figura 2.16, muestra el modelo de la base de datos de Neo4j, la cual cuenta con las entidades Author, Affiliation, Article y Topic, y las relaciones WROTE, BELONGS_TO, USES, AFFILIATED_WITH y CO_AUTHORED.

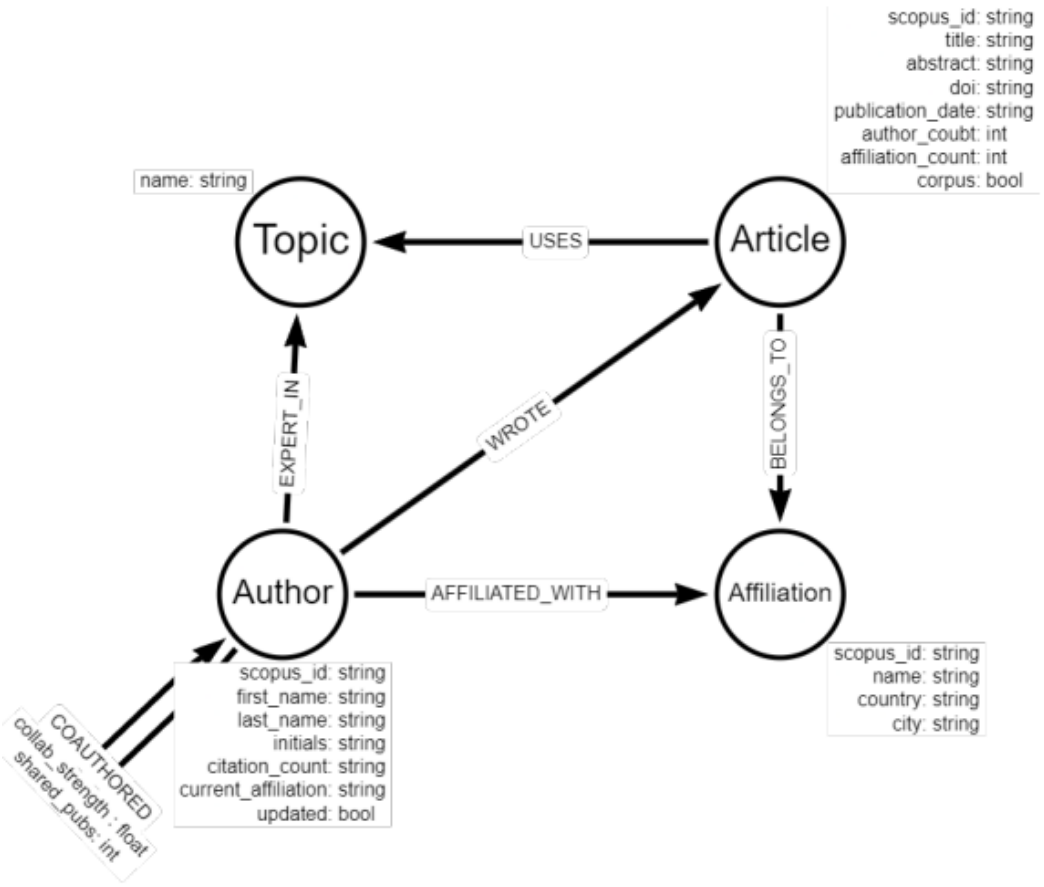



Figura 2.16: Modelo de la base de datos de Neo4j.

Para evaluar el contenido de la base de datos, se utilizó Neo4j Browser en donde se realizaron las consultas Cypher[1] presentes en la Tabla 2.7.

1

[1] Cypher: Es un lenguaje de consulta usado para interactuar con la base de datos Neo4j.

| **Consulta** | **Descripción** |
|---|---|
| MATCH (affiliation:Affiliation) RETURN affiliation | Esta consulta se realizó para ver las propiedades de la entidad Affiliation. |
| MATCH (author:Author) RETURN author | Esta consulta se realizó para ver las propiedades de la entidad Author. |
| MATCH (article:Article) RETURN article | Esta consulta se realizó para ver las propiedades de la entidad Article. |
| MATCH (topic:Topic) RETURN topic | Esta consulta se realizó para ver las propiedades de la entidad Topic. |
| MATCH p=()-[r:AFFILIATED_WITH]->() RETURN p | Esta consulta se realizó para ver las entidades involucradas en la relación AFFILIATED_WITH. |
| MATCH p=()-[r:BELONGS_TO]->() RETURN p | Esta consulta se realizó para ver las entidades involucradas en la relación BELONGS_TO. |
| MATCH p=()-[r:CO_AUTHORED]->() RETURN p | Esta consulta se realizó para ver las entidades involucradas en la relación CO_AUTHORED. |

| Consulta | Descripción |
|---|---|
| MATCH p=()-[r:USES]->() RETURN p | Esta consulta se realizó para ver las entidades involucradas en la relación USES. |
| MATCH p=()-[r:WROTE]->() RETURN p | Esta consulta se realizó para ver las entidades involucradas en la relación WROTE. |

Tabla 2.7: Consultas Cypher para evaluación de contenido de Neo4j

Con los resultados de las consultas se pudo analizar detalladamente cada una de las entidades y relaciones presentes en la base de datos de Neo4j, los resultados de este análisis se pueden ver en la Tabla 2.8.

| Entidad | Propiedades | Relaciones |
|---|---|---|
| **Affiliation** | **scopus_id:** ID único de Scopus<br>**country:** País de la afiliación<br>**city:** Ciudad en la que se encuentra la afiliación<br>**name:** Nombre de la afiliación | |
| **Article** | **scopus_id:** ID único de Scopus<br>**affiliation_count:** Número de afiliaciones<br>**publication_date:** Fecha de publicación<br>**corpus:** Parte del corpus (True/False)<br>**abstract:** Resumen del artículo<br>**title:** Título del artículo<br>**author_count:** Número de autores<br>**doi:** DOI del artículo | **BELONGS_TO->Affiliation:** Asocia un artículo a una afiliación.<br>**USES->Topic:** Relaciona un artículo con un tópico específico. |

| Entidad | Propiedades | Relaciones |
|---|---|---|
| **Author** | **scopus_id:** ID único de Scopus<br>**auth_name:** Nombre del autor<br>**initials:** Iniciales del autor<br>**last_name:** Apellido del autor<br>**first_name:** Nombre del autor | **AFFILIATED_WITH->AFFILIATION:** Indica la afiliación del autor<br>**CO_AUTHORED->AUTHOR:** Muestra la coautoría entre autores.<br>**WROTE->ARTICLE:** Indica qué autor escribió qué artículo.<br>**EXPERT_IN->TOPIC:** Indica los tópicos en los que el autor tiene mayor experiencia. |
| **Topic** | **name:** Nombre del tópico | |

Tabla 2.8: Entidades y Relaciones en la BD Neo4j

De este análisis se resalta lo siguiente:

- La propiedad scopus_id de las entidades servirá para evitar duplicados al momento de crear una base de datos dedicada a contener la información de consumo para el dashboard.
- La propiedad publication_date de Article permitirá asociar el KPI número de artículos, con los años.
- La propiedad city de Affiliation facilitará la ubicación provincial del número de artículos, sin embargo existen registros de afiliaciones sin este campo, por lo que su ubicación en una provincia no será posible.
- La relación BELONGS_TO permitirá el conteo de artículos por año de una afiliación y a su vez de una provincia.
- La relación WROTE permitirá el conteo de artículos por año de un autor.
- La relación USES permitirá el conteo de artículos por año de un TOPIC.

**Consultas para poblar la base de datos No SQL**

Después de haber analizado las fuentes de datos, se pudo establecer las consultas que servirán para construir una base de datos No SQL dedicada solamente para el consumo de información para el dashboard. Estas consultas se pueden ver en la Tabla 2.9.

| **Consulta** | **Descripción** |
|---|---|
| MATCH (ar:Article)-[b:BELONGS_TO]->(af:Affiliation) OPTIONAL MATCH (ar)-[u:USES]->(t:Topic) RETURN af.scopus_id, af.name, af.city, ar.scopus_id, ar.publication_date, t.name | Esta consulta se usará para obtener información de artículos y topics de Provincias. |
| MATCH (au:Author)-[w:WROTE]->(ar:Article) OPTIONAL MATCH (ar)-[u:USES]->(t:Topic) RETURN au.scopus_id, ar.scopus_id, ar.publication_date, t.name | Esta consulta se usará para obtener el número de artículos y topics por año de Autores. Además, también se empleará para contar los autores a nivel País. |
| MATCH (ar:Article) OPTIONAL MATCH (ar)-[u:USES]->(t:Topic) RETURN ar.scopus_id, ar.publication_date, t.name | Esta consulta ayudará a obtener un conteo total de artículos y topics por año a nivel País. |
| MATCH (au:Author)-[w:WROTE]->(ar:Article) OPTIONAL MATCH (ar)-[u:USES]->(t:Topic) RETURN au.scopus_id, ar.scopus_id, ar.publication_date, t.name | Esta consulta se usará para obtener el número de autores por año a nivel País. |
| MATCH (ar:Article)-[b:BELONGS_TO]->(af:Affiliation) OPTIONAL MATCH (ar)-[u:USES]->(t:Topic) RETURN af.scopus_id, af.name, ar.scopus_id, ar.publication_date, t.name | Esta consulta se usará para obtener el número de afiliaciones por año a nivel País, y obtener el número de artículos y topics por año en Afiliaciones. |

Tabla 2.9: Consultas Cypher para creación de BD NoSQL

### 2.9.4. Revisión y retrospectiva del Sprint 3

Con la evaluación de los criterios de aceptación de las historias de usuario asignadas al sprint, se concluye que el Sprint 3 culminó satisfactoriamente al haber cumplido con los

objetivos.

- Se implementó el diseño de la búsqueda de autores relevantes.
- Se completó la Fase Diseño de la metodología NOETIX.

## 2.10. Sprint 4

### 2.10.1. Objetivos del Sprint

- Implementar la búsqueda de autores relevantes en el backend.
- Completar la Fase Construcción y Validación de la metodología Noetix.

### 2.10.2. Sprint Planning

En el Sprint 4 se realizaran las historias de usuario detalladas en la Tabla 2.10.

| **Sprint Backlog 4** |
|---|
| **HU-SE-09: Búsqueda de autores relevantes** |
| *Como usuario no registrado, quiero poder ver los autores relevantes a un tópico para poder informarme sobre cómo se han organizado los investigadores para realizar contribuciones en un tópico.* |
| **Criterios de Aceptación:** |
| - **Implementación de la funcionalidad en el backend:** Se han implementado las funciones para buscar autores relevantes y obtener sus afiliaciones. |
| **Tareas:** |
| T1. Implementar función para encontrar autores relevantes en TFIDF. |
| T2. Implementar función para encontrar los autores relevantes y sus afiliaciones. |
| **HU-MA-08: Diseño e implementación de la base de datos No SQL** |
| *Como miembro del equipo de desarrollo, quiero implementar una base de datos No SQL en la que se almacenen los datos necesarios para la construcción del dashboard, permitiendo así realizar consultas eficientes y proporcionar información actualizada para el dashboard.* |
| **Criterios de Aceptación:** |
| - **Arquitectura de la base de datos:** Se ha definido una arquitectura para la base de datos No SQL. |

| |
|---|
| - **Modelar la base de datos No SQL con Django Rest Framework:** Se ha modelado la base de datos No SQL en Django Rest Framework. |
| - **Poblar la base de datos No SQL:** Se ha poblado la base de datos No SQL. |
| - **Serializadores:** Se han implementado los serializadores necesarios para emitir las respuestas de la base de datos en formato JSON. |
| - **Vistas:** Se han implementado las vistas necesarias para que el dashboard pueda solicitar la información. |
| **Tareas:** |
| T1. Diseñar la arquitectura de la base de datos No SQL. |
| T2. Crear los modelos de la base de datos No SQL en Django Rest Framework. |
| T3. Implementar las funciones de limpieza y análisis de la información obtenida de Neo4j para poblar la base de datos No SQL. |
| T4. Implementar los serializadores de las entidades para emitir las respuestas en formato JSON. |
| T5. Implementar las vistas necesarias para receptar las solicitudes de información del dashboard. |
| **HU-MA-09: Construcción del dashboard** |
| *Como miembro del equipo de desarrollo, quiero implementar dashboards del país, afiliaciones y tópicos para que todos los usuarios puedan tener a su alcance una forma interactiva de ver la información de investigación.* |
| **Criterios de Aceptación:** |
| - **Dashboard general:** Se ha implementado el dashboard general que muestra información del país. |
| - **Dashboard de afiliación:** Se ha implementado el dashboard de afiliaciones que muestra información personalizada para cada afiliación. |
| - **Dashboard de tópicos:** Se ha implementado el dashboard de tópicos que muestra información personalizada para cada tópico. |
| - **Selector de año:** Se ha implementado la función para ajustar la información presentada en los dashboards respecto a un año o hasta un año. |
| **Tareas:** |
| T1. Implementar el dashboard general. |
| T2. Implementar el dashboard de afiliaciones. |

| |
|---|
| T3. Implementar el dashboard de tópicos. |
| T4. Implementar el selector de año para cada dashboard. |

Tabla 2.10: Historias de Usuario del Sprint 4

### 2.10.3. Ejecución del Sprint

#### Implementar la búsqueda de autores relevantes en el backend

Para que la búsqueda de autores relevantes de un tópico, obtenga una respuesta, se creó una función en la que se encontrarán los autores relevantes a un tópico en un Modelo TFIDF[2] como se muestra en la Figura 2.17.

```
def get_most_relevant_docs_by_topic_v2(self, topic, authorSize):
    preprocessed_topic = self.preprocess_topic(topic)

    if all(token in self.model['vocabulary'] for token in preprocessed_topic):
        token_ids = [self.model['vocabulary'][token]
                     for token in preprocessed_topic]
        data = {}
        for tokenId in token_ids:
            data[tokenId] = [item[0] for item in self.model['matrix'].getcol(
                tokenId).sorted_indices().toarray()]
        df_result = pd.DataFrame(data=data, index=self.model['indexes'])
        if authorSize:
            return df_result[(df_result != 0).all(1)].sum(axis=1).sort_values(ascending=False).head(authorSize)
        else:
            return df_result[(df_result != 0).all(1)].sum(axis=1).sort_values(ascending=False)
    else:
        return pd.Series()
```

Figura 2.17: Función para encontrar autores relevantes en TFIDF.

En un servicio de autores, tendremos una función que usará la búsqueda de autores relevantes del Modelo TFIDF, como se muestra en la Figura 2.18.

```
def find_most_relevant_authors_by_topic(self, topic: str, authors_number: int):
    try:
        m = Model("author")
        authors = m.get_most_relevant_docs_by_topic_v2(topic, authors_number)
        return authors
    except Exception as e:
        raise Exception(f"Error finding most relevant authors by topic: {e}")
```

Figura 2.18: Función para encontrar autores relevantes en TFIDF.

En el mismo servicio, como se muestra en la Figura 2.19 se creará otra función que se encargue de encontrar las afiliaciones de los autores relevantes encontrados que servirán para mostrarse en los filtros.

2

```python
def find_authors_by_affiliation_filter(self, filter_type: str, affiliations_ids: List[str],
                                       authors_ids: List[str]) -> List[object]:
    try:
        print('Filter type: ', filter_type)
        print('Affiliations IDs: ', affiliations_ids)
        print('Authors IDs: ', authors_ids)
        authors_str = [f'"{w}"' for w in authors_ids]
        affiliations_str = [f'"{w}"' for w in affiliations_ids]
        authors_ids_str = ', '.join(map(str, authors_str))
        affiliations_ids_str = ', '.join(map(str, affiliations_str))
        if filter_type == 'include':
            query = f"""
               MATCH (a:Author)-[:AFFILIATED_WITH]->(aff:Affiliation)
               WHERE a.scopus_id IN [{authors_ids_str}] AND aff.scopus_id IN [{affiliations_ids_str}]
               RETURN a
               """
        else:
            query = f"""
               MATCH (a:Author)-[:AFFILIATED_WITH]->(aff:Affiliation)
               WHERE a.scopus_id IN [{authors_ids_str}] AND NOT aff.scopus_id IN [{affiliations_ids_str}]
               RETURN a
               """
        results, _ = db.cypher_query(query)
        authors = [Author.inflate(row[0]) for row in results]
        print("Len of authors extracted: ", len(authors))
        return authors
    except Exception as e:
        raise Exception(f"Error finding authors by affiliation filter: {e}")
```

Figura 2.19: Función para encontrar afiliaciones de autores relevantes.

## Fase de Construcción y Validación de la metodología Noetix

### Estructura de la base de datos No SQL

Para la realización de consultas dedicadas al dashboard, se implementó una base de datos No SQL en MongoDB, para ello se definió una primera estructura de colecciones, como se muestra en la Tabla 2.11.

| Estructura de la colección | Descripción |
|---|---|
| Author = {<br>“idScopus”: number,<br>“years”: YearContribution[],<br>“totalArticles”: number,<br>“topics”: TopicContribution[]<br>} | Los documentos de esta colección contendrían el id de scopus del autor, el total de artículos y documentos embebidos para el resumen de la contribución anual de artículos del autor y para el resumen de la contribución anual de tópicos. En base a los documentos de esta colección además se hacían cálculos en tiempo real para obtener el número de artículos y tópicos acumulados hasta cierto año para un autor. |

[2]TFIDF: técnica utilizada para la recuperación de información y minería de textos.

| Estructura de la colección | Descripción |
|---|---|
| Affiliation = {<br>“idAffiliation”: number,<br>“name”: string,<br>“years”: YearContribution[],<br>“totalArticles”: number,<br>“topics”: TopicContribution[]<br>} | Los documentos de esta colección contendrían el id de la afiliación, su nombre, el total de artículos y documentos embebidos para el resumen de la contribución anual de artículos de la afiliación y para el resumen de la contribución anual de tópicos. En base a los documentos de esta colección se hacían cálculos en tiempo real para obtener el número de artículos y tópicos acumulados hasta cierto año para una afiliación. |
| Province = {<br>“name”: string,<br>“years”: YearContribution[],<br>“totalArticles”: number,<br>“topics”: TopicContribution[]<br>} | Los documentos de esta colección contendrían el nombre de la provincia, el total de artículos y documentos embebidos para el resumen de la contribución anual de artículos de la provincia y para el resumen de la contribución anual de tópicos. En base a los documentos de esta colección se hacían cálculos en tiempo real para obtener el número de artículos y tópicos acumulados hasta cierto año de una provincia. |
| Country = {<br>“years”: YearContribution[],<br>“topics”: TopicContribution[]<br>} | En esta colección se incluiría un único documento que contendría los documentos embebidos para el resumen de la contribución anual de artículos del país y para el resumen de la contribución anual de tópicos. En base a este documento se incluían cálculos en tiempo real para obtener el número de artículos y tópicos acumulados hasta cierto año. |
| TopicContribution = {<br>“name”: string,<br>“years”: YearContribution[],<br>“totalArticles”: number<br>} | Este documento contendría el nombre del tópico, el total de artículos y documentos embebidos para el resumen de la contribución anual de artículos del tópico. |
| YearContribution = {<br>“numArticles”: number,<br>“year”: number<br>} | Este documento contendría el número de artículos y el año correspondiente. Se utilizaría para resumir la contribución anual de artículos para autores, afiliaciones, provincias y tópicos. |

Tabla 2.11: Estructuras y Descripciones del Documento

Sin embargo, las consultas que se realizaban a la base de datos con esta estructura tomaban un tiempo que se encontraba en intervalos desde 30 a 120 segundos, por lo que se consideró un rediseño de la estructura, tomando en cuenta lo siguiente:

- Los documentos embebidos agregan un gran tiempo de respuesta a las consultas, esto es debido a que se contaba solo con un cluster. La carga de trabajo se alcanzaba rápidamente debido a la gran extensión de este tipo de documentos, por lo que los documentos embebidos quedaron descartados para una nueva estructura de base de datos.
- Los cálculos en tiempo real también agregaban tiempo de respuesta a las consultas, por lo que se eligió crear colecciones con los valores calculados, esta solución se ha elegido debido a que la base de datos de Neo4j no cambia en periodos grandes de tiempo, por lo que los valores calculados estarían vigentes por el mismo periodo de tiempo.

Teniendo en cuenta los puntos anteriormente mencionados, se construyó una nueva estructura de base de datos como se puede ver en la Tabla 2.12.

| **Estructura de la colección** | **Descripción** |
|---|---|
| Author = {<br>“scopus_id”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus del autor y el total de artículos. |
| AuthorTopics = {<br>“scopus_id”: number,<br>“topic_name”: string,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus del autor, el nombre del tópico y el total de artículos con ese tópico del autor. Un autor puede tener varios tópicos. Por lo tanto, esta colección tendrá un documento distinto por cada tópico del autor. |

| Estructura de la colección | Descripción |
|---|---|
| AuthorTopicsAcumulated = {<br>“scopus_id”: number,<br>“topic_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus del autor, el nombre del tópico, el año y el total de artículos acumulados hasta el año. Un autor puede tener varios tópicos y las publicaciones sobre cada tópico pueden darse en distintos años. Por lo tanto, esta colección tendrá un documento distinto para cada autor, por cada tópico y año de publicación de artículos con ese tópico. |
| AuthorTopicsYear = {<br>“scopus_id”: number,<br>“topic_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus del autor, el nombre del tópico, el año y el total de artículos con el tópico en el año. Un autor puede tener varios tópicos y las publicaciones sobre cada tópico pueden darse en distintos años. Por lo tanto, esta colección tendrá un documento distinto para cada autor, por cada tópico y año de publicación de artículos con ese tópico. |
| AuthorYear = {<br>“scopus_id”: number,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus del autor, el año y el total de artículos en el año. Un autor puede tener varios años en los que ha publicado. Por lo tanto, esta colección tendrá un documento distinto por cada año de publicación del autor. |
| AuthorAcumulated = {<br>“scopus_id”: number,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus del autor, el año y el total de artículos acumulados hasta el año. Un autor puede tener varios años en los que ha publicado. Por lo tanto, esta colección tendrá un documento distinto por cada año de publicación del autor. |
| Affiliation = {<br>“scopus_id”: number,<br>“name”: string,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus de la afiliación, su nombre y el total de artículos. |

| Estructura de la colección | Descripción |
|---|---|
| AffiliationTopics = {<br>“scopus_id”: number,<br>“name”: string,<br>“topic_name”: string,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus de la afiliación, su nombre, el nombre del tópico y el total de artículos del tópico. Una afiliación puede tener varios tópicos. Por lo tanto, esta colección tendrá un documento distinto por cada tópico de la afiliación. |
| AffiliationTopicsAcumulated = {<br>“scopus_id”: number,<br>“name”: string,<br>“topic_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus de la afiliación, su nombre, el nombre del tópico, el año y el total de artículos acumulados hasta el año. Una afiliación puede tener varios tópicos y las publicaciones sobre cada tópico pueden darse en distintos años. Por lo tanto, esta colección tendrá un documento distinto para cada afiliación, por cada tópico y año de publicación de artículos con ese tópico. |
| AffiliationTopicsYear = {<br>“scopus_id”: number,<br>“name”: string,<br>“topic_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el id de Scopus de la afiliación, su nombre, el nombre del tópico, el año y el total de artículos en el año. Una afiliación puede tener varios tópicos y las publicaciones sobre cada tópico pueden darse en distintos años. Por lo tanto, esta colección tendrá un documento distinto para cada afiliación, por cada tópico y año de publicación de artículos con ese tópico. |
| AffiliationYear = {<br>“scopus_id”: number,<br>“name”: string,<br>“year”: number,<br>“total_articles”: number,<br>“total_topics”: number<br>} | Los documentos de esta colección contienen el id de Scopus de la afiliación, su nombre, el año, el total de artículos y el total de tópicos de los que se realizaron publicaciones en el año. Una afiliación puede tener varios años en los que ha publicado. Por lo tanto, esta colección tendrá un documento distinto por cada año de publicación de la afiliación. |

| Estructura de la colección | Descripción |
|---|---|
| AffiliationAcumulated = {<br>“scopus_id”: number,<br>“name”: string,<br>“year”: number,<br>“total_articles”: number,<br>“total_topics”: number<br>} | Los documentos de esta colección contienen el id de Scopus de la afiliación, su nombre, el año, el total de artículos y el total de tópicos de los que ha realizado publicaciones hasta el año. Una afiliación puede tener varios años en los que ha publicado. Por lo tanto, esta colección tendrá un documento distinto por cada año de publicación de la afiliación. |
| CountryAcumulated = {<br>“year”: number,<br>“total_authors”: number,<br>“total_articles”: number,<br>“total_affiliations”: number,<br>“total_topics”: number<br>} | Los documentos de esta colección contienen el año, el total de autores, el total de artículos, el total de afiliaciones y el total de tópicos. Todos los valores totales son los acumulados hasta el año en el país. En el país se realizan publicaciones anuales. Por lo tanto, esta colección tendrá un documento distinto por cada año de publicación en el país. |
| CountryTopics = {<br>“topic_name”: string,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el nombre del tópico y el total de artículos en el país. Esta colección tendrá un documento distinto por cada tópico del que se haya publicado alguna vez en el país. |
| CountryTopicsAcumulated = {<br>“topic_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el nombre del tópico, el año y el total de artículos publicados con el tópico, acumulados hasta el año en el país. Esta colección tendrá un documento distinto por cada tópico y año de publicación del tópico que haya en el país. |
| CountryTopicsYear = {<br>“topic_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el nombre del tópico, el año y el total de artículos publicados con el tópico en el país. Esta colección tendrá un documento distinto por cada tópico y año de publicación del tópico que haya en el país. |

| Estructura de la colección | Descripción |
|---|---|
| CountryYear = {<br>“year”: number,<br>“total_authors”: number,<br>“total_articles”: number,<br>“total_affiliations”: number,<br>“total_topics”: number<br>} | Los documentos de esta colección contienen el año, el total de autores, el total de artículos, el total de afiliaciones y el total de tópicos en el país. Todos los valores totales son para el año. Esta colección tendrá un documento distinto por cada año de publicación que haya en el país |
| Province = {<br>“province_name”: string,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el nombre de la provincia y el total de artículos. |
| ProvinceAcumulated = {<br>“province_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el nombre de la provincia, el año y el total de artículos acumulados hasta el año. En una provincia se realizan publicaciones anuales. Por lo tanto, esta colección tendrá un documento distinto por cada año de publicación en una provincia. |
| ProvinceTopicsAcumulated = {<br>“province_name”: string,<br>“topic_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el nombre de la provincia, el nombre del tópico, el año y el total de artículos acumulados que han sido publicados con el tópico hasta el año. Una provincia puede tener varios tópicos y las publicaciones sobre cada tópico pueden darse en distintos años. Por lo tanto, esta colección tendrá un documento distinto para cada provincia, por cada tópico y año de publicación de artículos con ese tópico. |
| ProvinceTopicsYear = {<br>“province_name”: string,<br>“topic_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el nombre de la provincia, el nombre del tópico, el año y el total de artículos publicados con el tópico en el año. Una provincia puede tener varios tópicos y las publicaciones sobre cada tópico pueden darse en distintos años. Por lo tanto, esta colección tendrá un documento distinto para cada provincia, por cada tópico y año de publicación de artículos con ese tópico. |

| Estructura de la colección | Descripción |
|---|---|
| ProvinceYear = {<br>“province_name”: string,<br>“year”: number,<br>“total_articles”: number<br>} | Los documentos de esta colección contienen el nombre de la provincia, el año y el total de artículos en el año. En una provincia se realizan publicaciones anuales. Por lo tanto, esta colección tendrá un documento distinto por cada año de publicación en una provincia. |

Tabla 2.12: Estructuras y Descripciones del Documento

**Conexión de MongoDB**

La nueva estructura de base de datos, se la implementó con Django Rest Framework. Para que MongoDB pueda usarse con Django Rest Framework se utilizó la biblioteca MongoEngine, y con esta se configuró la conexión a la base de datos, como se muestra en la Figura 2.20.

```
mongo_db_name = os.environ.get('MONGO_DB_NAME')
mongo_db_username = os.environ.get('MONGO_DB_USERNAME')
mongo_db_password = os.environ.get('MONGO_DB_PASSWORD')
mongo_host = os.environ.get("MONGO_DB_HOST")
mongo_port = os.environ.get("MONGO_DB_PORT")
mongo_uri = f'mongodb://{mongo_db_username}:{mongo_db_password}@{mongo_host}:{mongo_port}/{mongo_db_name}?authSource=admin'
mongoengine.connect(host=mongo_uri)
```

Figura 2.20: Conexión a la base de datos MongoDB.

**Modelos de MongoDB**

Los modelos son la forma de representar los documentos a crearse en la base de datos. Para la creación de modelos, MongoEngine provee la clase Document que mapeará el modelo en la base de datos. La Figura 2.21 muestra la implementación del documento *Affiliation* con MongoEngine.

```
class Affiliation(Document):
    scopus_id = fields.IntField(required=True, unique=True)
    name = fields.StringField()
    total_articles = fields.IntField()
```

Figura 2.21: Implementación del documento *Affiliation*.

En la Figura 2.22, se puede observar los documentos que fueron implementados de la misma forma que el documento *Affiliation*.

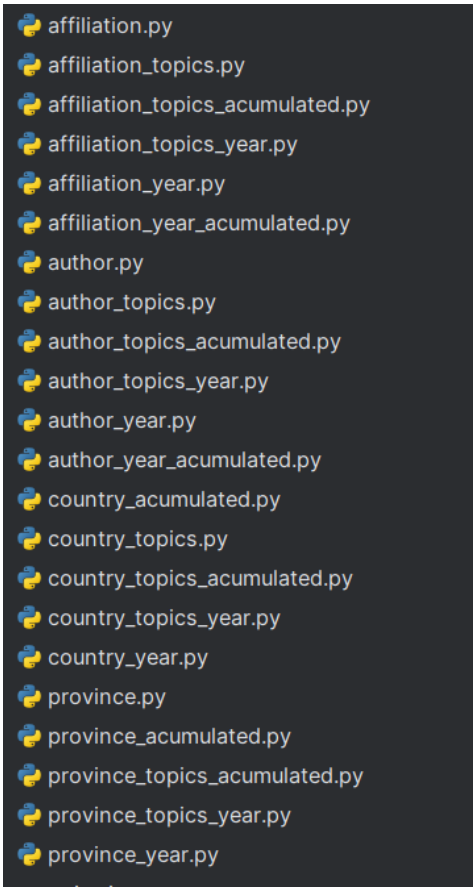



Figura 2.22: Documentos implementados.

**Población de la base de datos MongoDB**

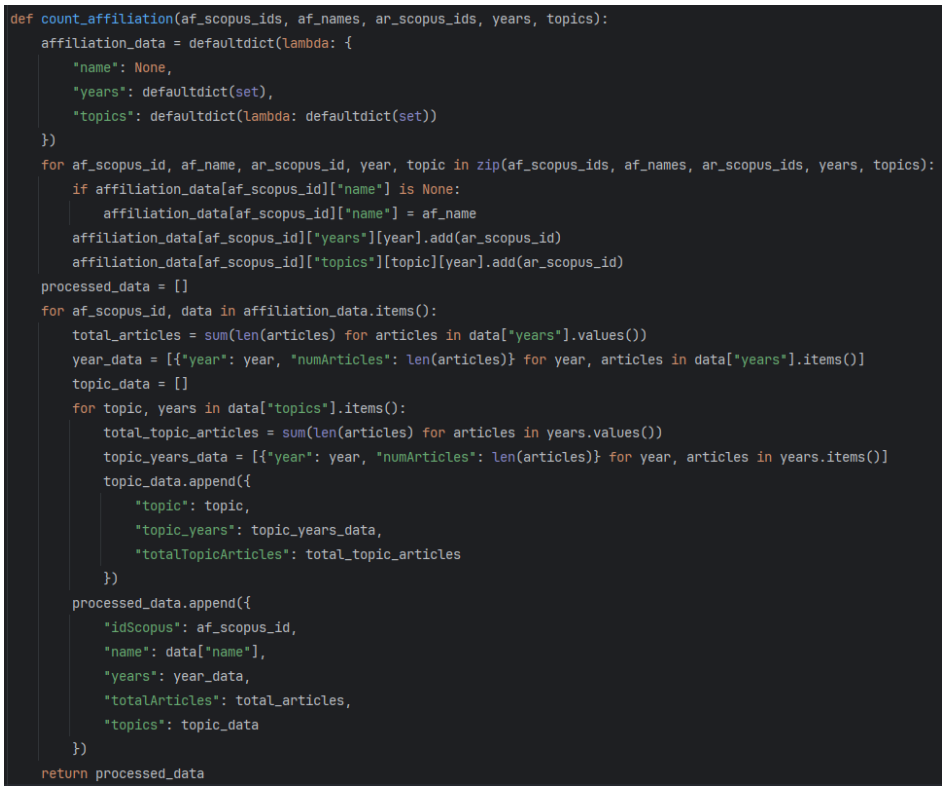
En base a las consultas definidas en 2.9.3, los documentos se poblarán con la información resultante de las siguientes funciones de limpieza y análisis del resultado de las consultas:

- **extract_year:** Como se muestra en la Figura 2.23, esta función sirve para extraer solamente el año de la fecha de publicación de un artículo.

```
def extract_year(years_column):
    years = []
    for date in years_column:
        year = date[:4]
        years.append(year)
    return years
```

Figura 2.23: Función de extracción de año.

- **count_affiliation:** Como se muestra en la Figura 2.24, esta función sirve para obtener la información de publicaciones por año de todas las afiliaciones.



```python
def count_affiliation(af_scopus_ids, af_names, ar_scopus_ids, years, topics):
    affiliation_data = defaultdict(lambda: {
        "name": None,
        "years": defaultdict(set),
        "topics": defaultdict(lambda: defaultdict(set))
    })
    for af_scopus_id, af_name, ar_scopus_id, year, topic in zip(af_scopus_ids, af_names, ar_scopus_ids, years, topics):
        if affiliation_data[af_scopus_id]["name"] is None:
            affiliation_data[af_scopus_id]["name"] = af_name
        affiliation_data[af_scopus_id]["years"][year].add(ar_scopus_id)
        affiliation_data[af_scopus_id]["topics"][topic][year].add(ar_scopus_id)
    processed_data = []
    for af_scopus_id, data in affiliation_data.items():
        total_articles = sum(len(articles) for articles in data["years"].values())
        year_data = [{"year": year, "numArticles": len(articles)} for year, articles in data["years"].items()]
        topic_data = []
        for topic, years in data["topics"].items():
            total_topic_articles = sum(len(articles) for articles in years.values())
            topic_years_data = [{"year": year, "numArticles": len(articles)} for year, articles in years.items()]
            topic_data.append({
                "topic": topic,
                "topic_years": topic_years_data,
                "totalTopicArticles": total_topic_articles
            })
        processed_data.append({
            "idScopus": af_scopus_id,
            "name": data["name"],
            "years": year_data,
            "totalArticles": total_articles,
            "topics": topic_data
        })
    return processed_data
```

Figura 2.24: Función de análisis de información de afiliaciones.

- **count_authors:** Como se muestra en la Figura 2.25, esta función sirve para obtener la información de publicaciones por año de todos los autores.

```python
def count_authors(entity_id_column, articles_id_column, years_column, topics):
    author_data = defaultdict(lambda: {
        "years": defaultdict(set),
        "topics": defaultdict(lambda: defaultdict(int))
    })

    for entity_id, article_id, year, topic in zip(entity_id_column, articles_id_column, years_column, topics):
        author_data[entity_id]["years"][year].add(article_id)
        author_data[entity_id]["topics"][topic][year] += 1

    processed_data = []
    for entity_id, data in author_data.items():
        total_articles = sum(len(articles) for articles in data["years"].values())
        year_data = [{"year": year, "numArticles": len(articles)} for year, articles in data["years"].items()]

        topic_data = []
        for topic, years in data["topics"].items():
            total_topic_articles = sum(years.values())
            topic_years_data = [{"year": year, "numArticles": count} for year, count in years.items()]
            topic_data.append({
                "topic": topic,
                "topic_years": topic_years_data,
                "totalTopicArticles": total_topic_articles
            })

        processed_data.append({
            "idScopus": entity_id,
            "years": year_data,
            "totalArticles": total_articles,
            "topics": topic_data
        })
    return processed_data
```

Figura 2.25: Función de análisis de información de autores.

- **count_articles_topics_per_year_country:** Como se muestra en la Figura 2.26, esta función sirve para obtener la información de años y de tópicos por año del país.

```python
def count_authors(entity_id_column, articles_id_column, years_column, topics):
    author_data = defaultdict(lambda: {
        "years": defaultdict(set),
        "topics": defaultdict(lambda: defaultdict(int))
    })

    for entity_id, article_id, year, topic in zip(entity_id_column, articles_id_column, years_column, topics):
        author_data[entity_id]["years"][year].add(article_id)
        author_data[entity_id]["topics"][topic][year] += 1

    processed_data = []
    for entity_id, data in author_data.items():
        total_articles = sum(len(articles) for articles in data["years"].values())
        year_data = [{"year": year, "numArticles": len(articles)} for year, articles in data["years"].items()]

        topic_data = []
        for topic, years in data["topics"].items():
            total_topic_articles = sum(years.values())
            topic_years_data = [{"year": year, "numArticles": count} for year, count in years.items()]
            topic_data.append({
                "topic": topic,
                "topic_years": topic_years_data,
                "totalTopicArticles": total_topic_articles
            })

        processed_data.append({
            "idScopus": entity_id,
            "years": year_data,
            "totalArticles": total_articles,
            "topics": topic_data
        })
    return processed_data
```

Figura 2.26: Función de análisis de información del país.

- **get_articles_topics_info:** Como se muestra en la Figura 2.27, esta función sirve para obtener la información por año y acumulado de artículos y tópicos del país.

```
def get_articles_topics_info(countries):
        new_topics = topics_per_year[year] - acumulated_topics
        num_new_topics = len(new_topics)
        topics_per_year_list.append({"name": str(year), "value": num_new_topics})
        acumulated_topics.update(new_topics)
        topics_acumulative_list.append({"name": str(year), "value": len(acumulated_topics)})

    # Invertir las listas para el orden descendente de los años
    articles_per_year_list = [{"name": str(year), "value": articles_per_year[year]} for year in sorted_article_years]
    articles_acumulative_list = [{"name": str(year), "value": articles_acumulative[year]} for year in
                                 sorted_article_years]
    articles_per_year_list.reverse()
    articles_acumulative_list.reverse()
    topics_per_year_list.reverse()
    topics_acumulative_list.reverse()

    response_data = {
        "Articles": {
            "Per_year": articles_per_year_list,
            "Acumulative": articles_acumulative_list
        },
        "Topics": {
            "Per_year": topics_per_year_list,
            "Acumulative": topics_acumulative_list
        }
    }
    return response_data
```

Figura 2.27: Función para obtención de información acumulada en el país.

- **find_province:** Como se muestra en la Figura 2.28, esta función sirve encontrar la provincia de las afiliaciones en base al campo City.

```
def find_province(city_name):
    if city_name is None:
        return -1, 'Pendiente'

    city_name = city_name.upper()
    for province_id, province_info in location_data.items():
        if "provincia" in province_info:
            provincia = province_info["provincia"].upper()
            if city_name == provincia:
                return province_id, provincia
            if "cantones" in province_info:
                for canton_id, canton_info in province_info["cantones"].items():
                    if "canton" in canton_info:
                        if city_name == canton_info["canton"].upper():
                            return province_id, provincia
                    if "parroquias" in canton_info:
                        for parroquia_id, parroquia_name in canton_info["parroquias"].items():
                            if city_name == parroquia_name.upper():
                                return province_id, provincia
    return -1, 'Pendiente'
```

Figura 2.28: Función para encontrar provincia.

- **count_province:** Como se muestra en la Figura 2.29, esta función sirve para obtener

la información de años y de tópicos por año por provincia.

```
def count_province(processed_data):
        province_data[province_id]["topics"][topic][year] += 1

    final_list = []
    for province_id, data in province_data.items():
        total_articles = sum(articles for articles in data["years"].values())
        year_data = [{"year": year, "numArticles": articles} for year, articles in data["years"].items()]

        topic_data = []
        for topic, years in data["topics"].items():
            total_topic_articles = sum(years.values())
            topic_years_data = [{"year": year, "numArticles": articles} for year, articles in years.items()]
            topic_data.append({
                "topic": topic,
                "topic_years": topic_years_data,
                "totalTopicArticles": total_topic_articles
            })

        final_list.append({
            "id_provincia": province_id,
            "provincia": province_data[province_id]["province_name"],
            "num_articles": total_articles,
            "years": year_data,
            "topics": topic_data
        })

    return final_list
```

Figura 2.29: Función de análisis de información de provincia.

**Serializadores de MongoDB**

Los serializadores son la forma de reformatear los objetos en formatos almacenables y transmitibles como JSON. MongoEngine también trae por defecto la clase DocumentSerializer que dará un formato de diccionario de Python al documento serializado, y este podrá transmitirse como JSON. La Figura 2.30 muestra el serializador del documento *Affiliation*, de la cual se serializarán todos sus campos.

```
class AffiliationSerializer(DocumentSerializer):
    JoffreC
    class Meta:
        model = Affiliation
        fields = '__all__'
```

Figura 2.30: Serializador del documento Affiliation.

En la Figura 2.31, se puede observar los serializadores que fueron implementados de la misma forma que el serializador del documento *Affiliation*.

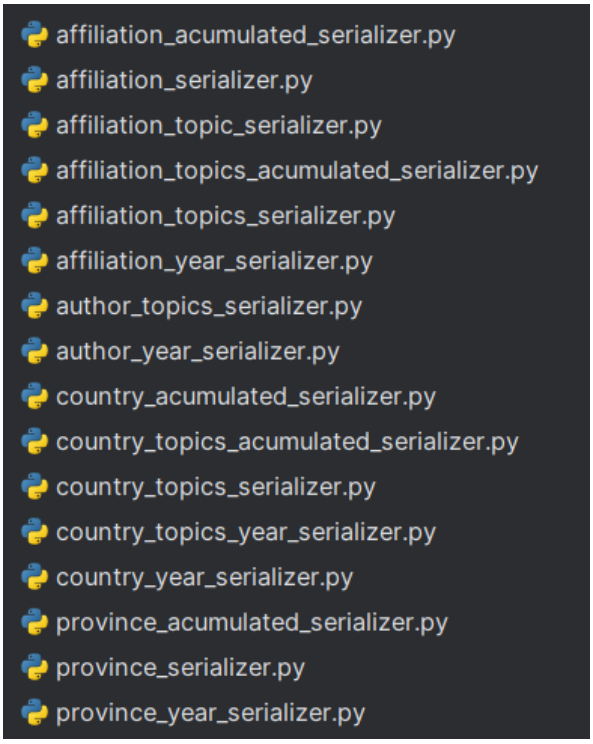



Figura 2.31: Serializadores implementados.

### Vistas

Las Vistas son donde se manejará las solicitudes y respuestas de un cliente. La Figura 2.32 muestra la vista implementada para el documento *Author*.

```python
class AuthorViews(viewsets.ModelViewSet):

    @extend_schema(
        description="Get author publication years",
        responses=AuthorYearSerializer(many=True),
        tags=['Authors'],
        parameters=[
            OpenApiParameter(name='scopus_id', type=str, location=OpenApiParameter.QUERY, description='Scopus ID')
        ]
    )
    @action(detail=False, methods=['get'])
    def get_author_years(self, request): ...
            return Response({"error": str(e)}, status=status.HTTP_500_INTERNAL_SERVER_ERROR)

    @extend_schema(
        description="Get topics associated with an author",
        responses=AuthorTopicsSerializer(many=True),
        tags=['Authors'],
        parameters=[
            OpenApiParameter(name='scopus_id', type=str, location=OpenApiParameter.QUERY, description='Scopus ID')
        ]
    )
    @action(detail=False, methods=['get'])
    def get_topics(self, request): ...
```

Figura 2.32: Vista de Author.

En la Figura 2.33, se puede observar las vistas implementadas de forma similar a la vista del documento *Author*.

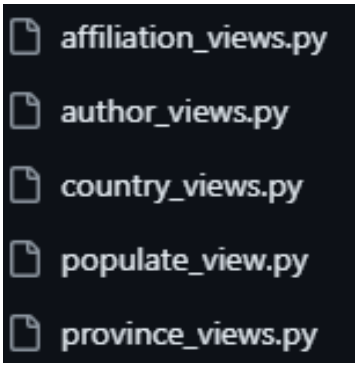



Figura 2.33: Vistas implementadas.

**Construcción del dashboard**

Con el prototipo definido en la Fase de Recopilación de requisitos 2.8.3, se construyó el dashboard general presentado en la Figura 2.34, que cuenta con la función de cambiar la información presentada en base a un año que puede ser seleccionado desde la parte superior del mismo. Además, se incluyó en la parte superior un resumen en el que se podrá ver información de cuantos autores, afiliaciones, artículos y áreas de conocimiento (tópicos) existen en el Ecuador.

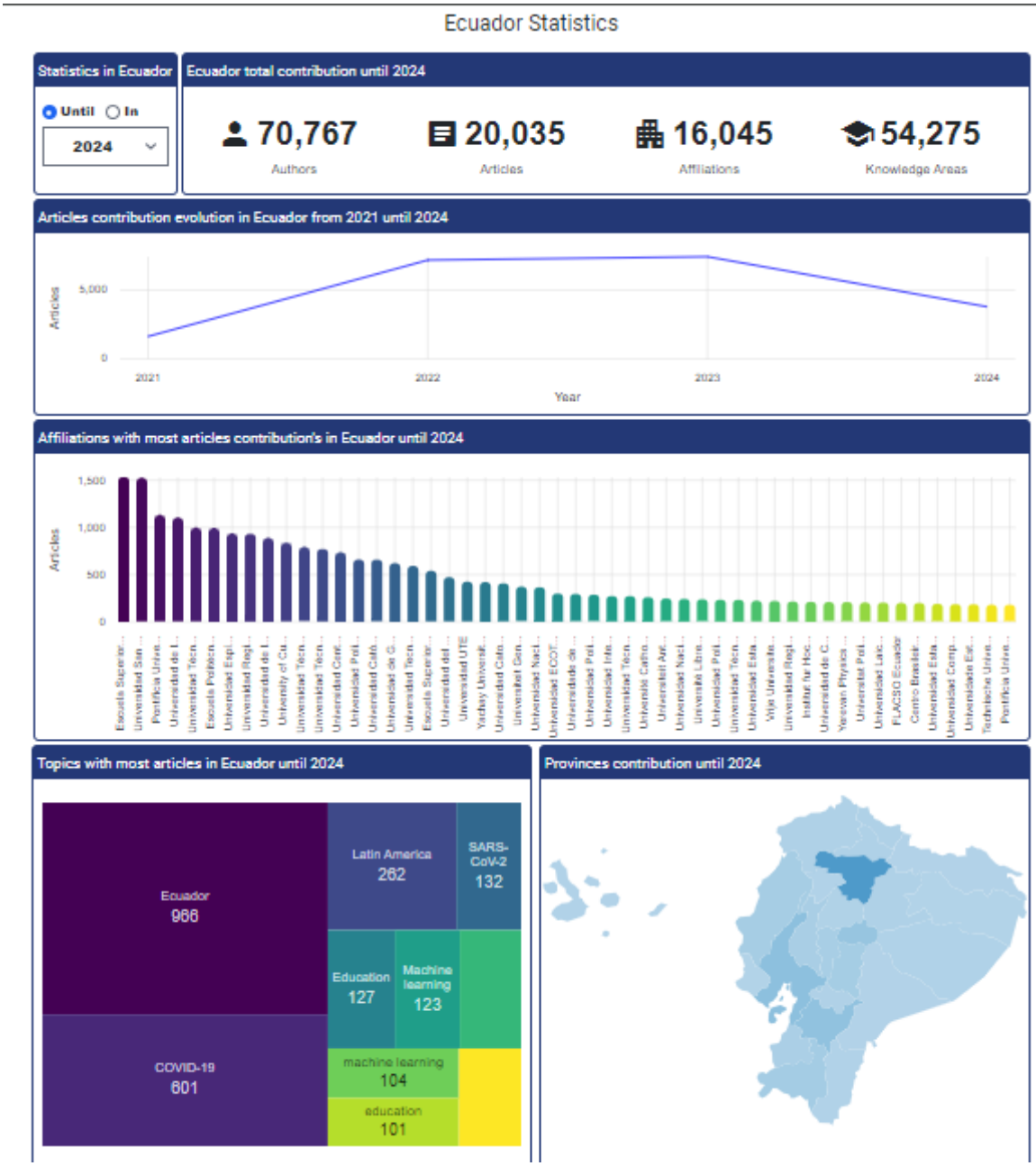



Figura 2.34: Dashboard General.

Como se observa en la Figura 2.35 , se desarrolló un dashboard que muestra la información respecto a afiliaciones, donde cada afiliación tendrá sus gráficas personalizadas. Además, en este dashboard también se cuenta con la función de seleccionar el año para que la información se ajuste.


Figura 2.35: Dashboard para una afiliación.

La Figura 2.36 muestra el dashboard desarrollado para presentar la información respecto a tópicos, donde cada tópico también tendrá sus gráficas personalizadas. Además, este dashboard también incluye la función de seleccionar el año para que la información se ajuste.

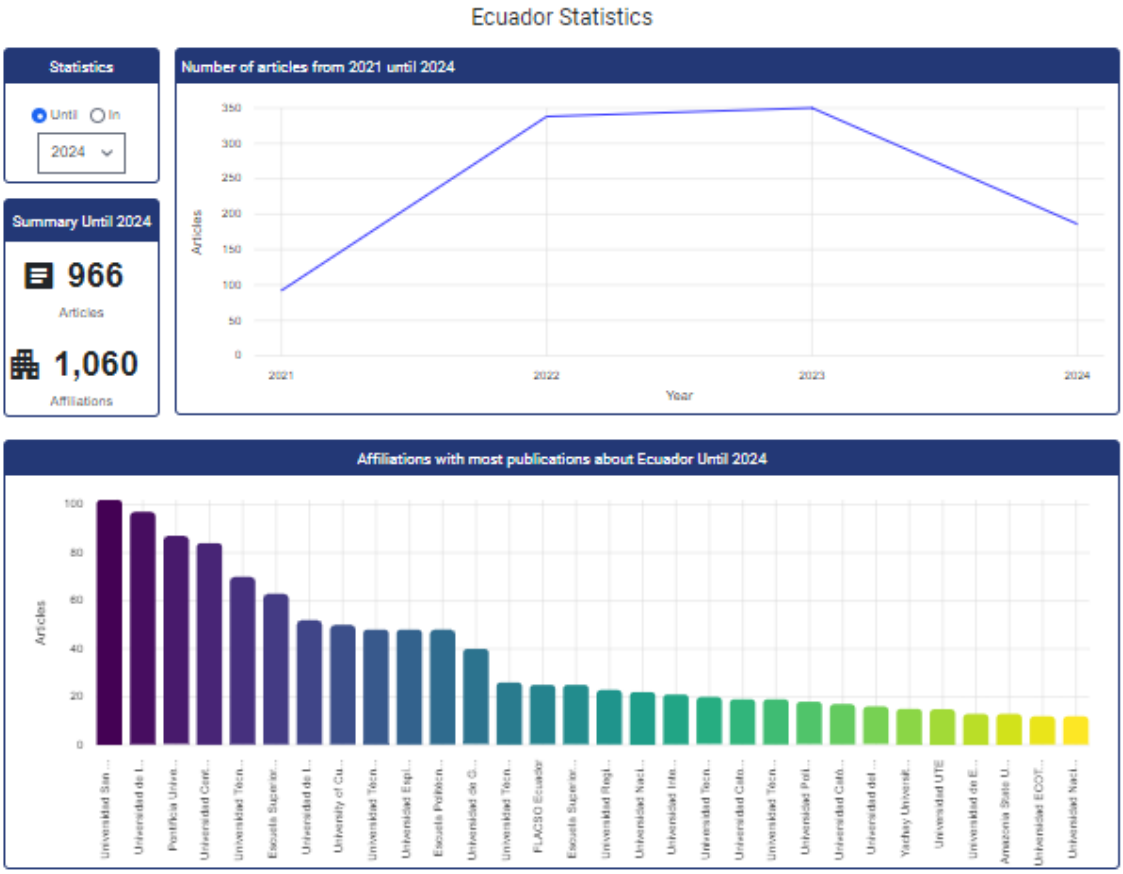



Figura 2.36: Dashboard para un topic.

### 2.10.4. Revisión y retrospectiva del Sprint 4

Con la evaluación de los criterios de aceptación de las historias de usuario asignadas al sprint, se concluye que el Sprint 4 culminó satisfactoriamente al haber cumplido con los objetivos.

- Se implementó la búsqueda de autores relevantes en el backend.
- Se completó la Fase de Construcción y Validación de la metodología Noetix.

## 2.11. Sprint 5

### 2.11.1. Objetivos del Sprint

- Completar las Fases de Despliegue y Mantenimiento de la metodología Noetix.

- Preparar el release final de la aplicación para el despliegue en un entorno controlado.

### 2.11.2. Sprint Planning

En el Sprint 5 se realizaran las historias de usuario detalladas en la Tabla 2.13.

| **Sprint Backlog 5** |
|---|
| **HU-MA-10: Integrar dashboard a entorno de producción** |
| *Como miembro del equipo de desarrollo, quiero integrar el dashboard en un ambiente de producción para poder realizar pruebas con usuarios.* |
| **Criterios de Aceptación:** |
| - **Base de datos dockerizada:** Se ha dockerizado la base de datos MongoDB para que su despliegue se pueda realizar junto con los demás componentes de CENTINELA. |
| **Tareas:** |
| T1. Dockerizar la base de datos MongoDB. |
| **HU-MA-11: Actualización de la base de datos No SQL** |
| *Como usuario administrador, quiero poder actualizar los datos almacenados en la base de datos No SQL para que los dashboards siempre presenten información actualizada.* |
| **Criterios de Aceptación:** |
| - **Método de actualización:** Se han implementado las funciones para actualizar la base de datos de MongoDB en el backend. |
| - **Notificador de estado:** Se ha implementado un elemento en el dashboard del administrador que muestra el estado de la base de datos de MongoDB. |
| **Tareas:** |
| T1. Implementar función para borrar la base de datos de MongoDB. |
| T2. Implementar función para crear y poblar la base de datos de MongoDB inmediatamente después de su eliminación. |
| T3. Implementar notificador de estado de la base de datos de MongoDB en el dashboard del administrador. |

Tabla 2.13: Historias de Usuario del Sprint 5

### 2.11.3. Ejecución del Sprint

#### Fases de Despliegue y Mantenimiento de la metodología Noetix.

**Dockerización de la base de datos MongoDB**

La base de datos de MongoDB se desplegará a través de Docker por lo que se incluirá un contenedor de Mongo en el archivo *compose* de la aplicación, como se muestra en la Figura 2.37.

```
mongo:
  container_name: mongo
  image: mongo:latest
  restart: always
  ports:
    - "27017:27017"
  environment:
    - MONGO_INITDB_ROOT_USERNAME=${MONGO_DB_USERNAME}
    - MONGO_INITDB_ROOT_PASSWORD=${MONGO_DB_PASSWORD}
  volumes:
    - ./mongo_db/data:/data/db
```

Figura 2.37: Contenedor de Mongo.

**Mantenimiento de la base de datos de MongoDB**

Para que la información de la base de datos se mantenga actualizada con la base de datos de Neo4j, se ha optado por eliminar la información de MongoDB y realizar una actualización inmediata, esto es debido a que no hay forma en que no se afecte el rendimiento de Neo4j, con la que se pueda conocer qué información se encuentra en Neo4j y no en MongoDB. Para esto se desarrolló una función que elimina la base de datos de MongoDB, como se muestra en la Figura 2.38.

```python
def drop_database(self):
        try:
            dl = get_db()
            db_name = dl.name
            mongoengine.connection.get_connection().drop_database(db_name)
        except Exception as e:
            print(f"Error dropping database: {e}")
```

Figura 2.38: Función para eliminar la base de datos de MongoDB.

Para que la base de datos se cree y poble inmediatamente después de su eliminación se creó la función de la Figura 2.39, que se encarga de borrar, crear y poblar la base de datos en ese orden.

```python
def populate(self):
    self.drop_database()
    self.populate_country()
    self.populate_affiliation()
    self.populate_province()
    self.populate_author()
```

Figura 2.39: Función para eliminar, crear y poblar la base de datos de MongoDB.

Para que el usuario administrador sepa que la base de datos MongoDB se encuentra desactualizada, en el frontend se creó un elemento que notifique de su estado al administrador, como se muestra en la Figura 2.40.

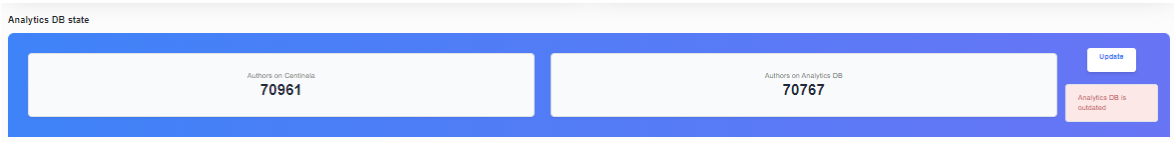



Figura 2.40: Sección de notificación de estado de la base de datos de MongoDB en el frontend.

### Preparación del release final

Para el release final de la aplicación se la ha encapsulado en contenedores de Docker. La Figura 2.41 muestra el archivo dockerfile para el frontend.

```
FROM node:18-alpine3.18

RUN npm install -g @angular/cli

WORKDIR /app

COPY . .

RUN npm install

EXPOSE 4200

CMD ["ng", "serve", "--host", "0.0.0.0", "--port", "4200", "--disable-host-check"]
```

Figura 2.41: Archivo dockerfile para el frontend.

Como se muestra en la Figura 2.42, para el backend también se creará un archivo dockerfile, necesario para su ejecución en un contenedor de Docker. Además, las bases de datos de Neo4j y MondgoDB también serán encapsuladas en contenedores de Docker, por lo que se creará un archivo compose en el que se encapsule el backend con ambas bases de datos, como se muestra en la Figura 2.43.

```
FROM python:3.11-bookworm

WORKDIR /app

COPY requirements.txt .

RUN pip install --no-cache-dir -r requirements.txt

COPY . .
EXPOSE 8001

CMD ["python", "manage.py", "runserver", "0.0.0.0:8001", "--noreload"]
```

Figura 2.42: Archivo dockerfile para el backend.




Figura 2.43: Archivo compose.yaml para los contenedores de bases de datos y backend.

### 2.11.4. Revisión y retrospectiva del Sprint

Con la evaluación de los criterios de aceptación de las historias de usuario asignadas al sprint, se concluye que el Sprint 5 culminó satisfactoriamente al haber cumplido con los objetivos.

- Se completó la Fases de Despliegue y Mantenimiento de la metodología Noetix.
- Se preparó el release final de la aplicación para su despliegue.

# 3. EVALUACIÓN Y RESULTADOS

En base a una Encuesta de usabilidad SUS, se evaluó las funcionalidades de motor de búsqueda de autores, autores relevantes y artículos relevantes, así como los dashboards de CENTINELA. Participaron 14 estudiantes de la Facultad de Ingeniería en Sistemas, quienes nunca habían probado sistemas similares y para evaluar estas funcionalidades, cada participante hizo 5 tareas que pueden revisarse en Anexo Tareas 6.4.

## 3.1. Resultados de la Encuesta SUS

Los resultados de la encuesta para cada participante se presentan a continuación en la Tabla 3.1.

| Encuestado | Resultado sus |
|---|---|
| 1 | 72.5 |
| 2 | 70.0 |
| 3 | 80.0 |
| 4 | 60.0 |
| 5 | 87.5 |
| 6 | 85.0 |
| 7 | 85.0 |
| 8 | 52.5 |
| 9 | 62.5 |
| 10 | 67.5 |
| 11 | 55.0 |
| 12 | 77.5 |
| 13 | 52.5 |
| 14 | 85.0 |

Tabla 3.1: Encuestados y Resultados.

## 3.2. Análisis de resultados

Los resultados de la Encuesta SUS muestran que las funcionalidades de búsqueda y dashboards de CENTINELA tienen una puntuación media de 70.89 sobre 100. Usando como base la Escala SUS de la Figura 3.1, la puntuación media obtenida de indica que el sistema tiene buena usabilidad con un margen de mejora.

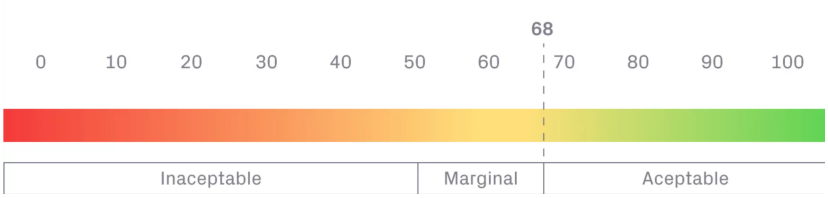



Figura 3.1: Escala SUS.

En cuanto a los resultados individuales de los encuestados presentes en el Anexo Resultados 6.5, se resalta lo siguiente:

- Los resultados de los encuestados 1, 2, 3, 5, 6, 7, 12 y 14, sugieren que CENTINELA tiene una muy buena usabilidad, con áreas que podrían mejorarse, pero que no afectan el uso.
- Los resultados de los encuestados 4, 9 y 10, sugieren que CENTINELA tiene una usabilidad aceptable, con áreas que deben mejorarse, ya que pueden representar dificultades al momento de usar el sistema.
- Los resultados de los encuestados 8, 11 y 13, sugieren que CENTINELA tiene problemas importantes de usabilidad, es decir encontraron una complejidad innecesaria al momento de usar el sistema.

## 3.3. Análisis por pregunta

La Figura 3.2, muestra la puntuación promedio obtenida por cada pregunta.

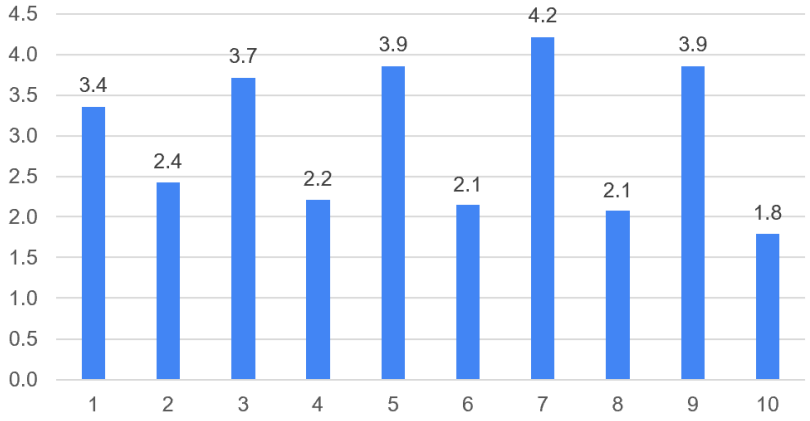



Figura 3.2: Resultados Preguntas.

A continuación se presentan los resultados específicos para cada pregunta.

### 3.3.1. Pregunta 1: “Creo que me gustaría utilizar este sistema con frecuencia”

La puntuación promedio de esta pregunta es de 3.4, lo que refleja una aceptación moderada del sistema por parte de los encuestados.

### 3.3.2. Pregunta 2: “Encontré el sistema innecesariamente complejo”

La puntuación promedio de esta pregunta es de 2.4, lo que indica una percepción mínima de complejidad en las funcionalidades evaluadas.

### 3.3.3. Pregunta 3: “Pensé que el sistema era fácil de usar”

La puntuación promedio de esta pregunta es de 3.7, lo que sugiere que la mayoría de los encuestados consideran la interfaz intuitiva y poco compleja.

### 3.3.4. Pregunta 4: “Creo que necesitaría el apoyo de un técnico para poder utilizar este sistema”

La puntuación promedio de esta pregunta es de 2.2, lo que indica que, aunque la mayoría de los encuestados no tuvieron problemas para realizar las tareas, existe una minoría que pudo necesitar asistencia técnica, lo cual sugiere un área de mejora en algunas funcionalidades.

### 3.3.5. Pregunta 5: “Encontré que las diversas funciones de este sistema estaban bien integradas”

La puntuación promedio de esta pregunta es de 3.9, lo que indica que los encuestados perciben que todas las partes del sistema funcionan como un conjunto cohesivo y coherente.

### 3.3.6. Pregunta 6: “Pensé que había demasiada inconsistencia en este sistema”

La puntuación promedio de esta pregunta es de 2.1, lo que indica una mínima percepción de inconsistencia en el sistema, sugiriendo un área de mejora en la consistencia de la interfaz.

### 3.3.7. Pregunta 7: “Me imagino que la mayoría de la gente aprendería a utilizar este sistema muy rápidamente”

La puntuación promedio de esta pregunta es de 4.2, lo que indica que el sistema es fácil de aprender a usar, es decir, la curva de aprendizaje del sistema es considerada baja.

### 3.3.8. Pregunta 8: “Encontré el sistema muy complicado de usar”

La puntuación promedio de esta pregunta es de 2.1, lo que sugiere que los usuarios no encuentran el sistema complicado de usar, aunque existen detalles que podrían mejorarse para reducir aún más la complejidad.

### 3.3.9. Pregunta 9: “Me sentí muy seguro usando el sistema”

La puntuación promedio de esta pregunta es de 3.9, lo que indica que los usuarios tienden a sentirse seguros utilizando el sistema.

### 3.3.10. Pregunta 10: “Necesitaba aprender muchas cosas antes de empezar con este sistema”

La puntuación promedio de esta pregunta es de 1.8, lo que indica que los usuarios sienten que se necesita algo de conocimiento previo para poder usar el sistema.

# 4. CONCLUSIONES Y RECOMENDACIONES

## 4.1. Conclusiones

Se desarrolló la primera versión de CENTINELA, una plataforma unificada en la que un investigador puede encontrar información, comunicarse y colaborar con su grupo de investigación, haciendo de CENTINELA una herramienta prometedora para potenciar la investigación en Ecuador.

La adopción de la metodología SCRUM facilitó la adaptación a los cambios y la comunicación entre los miembros del equipo, permitiendo que el proyecto cumpliera sus objetivos sin contratiempos.

La metodología Noetix para el desarrollo de dashboards permitió que la implementación de los dashboards en CENTINELA fuera un éxito, gracias a que proporciona una serie de pasos bien definidos y fáciles de integrar con SCRUM, a través de historias de usuario.

La arquitectura hexagonal empleada para desarrollar CENTINELA ha aportado múltiples beneficios en términos de desarrollo como de mantenimiento de software, puesto que ha permitido lograr una clara separación de responsabilidades, promoviendo un código más modular y fácil de entender, en el que se pueden implementar nuevos componentes y servicios sin necesidad de implementar cambios significativos en el núcleo de la aplicación.

El uso de una base de datos MongoDB dedicada exclusivamente al módulo de analítica reduce la carga de trabajo sobre la base de datos Neo4j, haciendo que la plataforma responda de forma rápida y eficiente en cada una de sus funcionalidades.

Las representaciones gráficas con las que cuenta CENTINELA ayudarán a que tanto investigadores como instituciones puedan tomar decisiones informadas sobre futuras investigaciones.

Los resultados de la encuesta SUS indican que CENTINELA es medianamente fácil de usar, por lo que la plataforma necesita mejoras en distintas áreas para optimizar la expe-

riencia del usuario.

## 4.2. Recomendaciones

Se recomienda implementar una forma en la que la base de datos de MongoDB no sea borrada, sino más bien solo se actualice la información que ya se encuentra almacenada, puesto que el método actual para mantenerse al día con la base de datos de Neo4j, deja a CENTINELA sin gráficas estadísticas por cierto período de tiempo.

Para mejorar la usabilidad de CENTINELA, y basándonos en el análisis del uso y la retroalimentación de los usuarios, se recomienda hacer cambios en la forma de presentar el grafo de coautoría de un autor. Los encuestados encontraron confusa la interpretación de la información del grafo. Además, se sugiere utilizar abreviaturas en lugar de nombres completos de afiliaciones en las gráficas de los dashboards.

Se recomienda implementar una sección de dashboards de información de Provincias puesto que la información necesaria para esto ya se encuentra almacenada en la base de datos MongoDB.

Se recomienda mantener el enfoque de la arquitectura hexagonal para el desarrollo de futuras versiones de CENTINELA, puesto que permite que el código sea más fácil de mantener y evolucionar.

# 5. REFERENCIAS BIBLIOGRÁFICAS


[1] G. Herrera-Franco, N. Montalván-Burbano, C. Mora-Frank y L. Bravo-Montero, «Scientific research in Ecuador: A bibliometric analysis,» *Publications*, vol. 9, n.º 4, pág. 55, 2021.

[2] J. N. A. Túqueres, *Desarrollo de sistema basado en minería de datos para la búsqueda y visualización de redes de investigadores con filiación en instituciones ecuatorianas y sus áreas académicas*, Quito, jun. de 2023. dirección: `http://bibdigital.epn.edu.ec/handle/15000/24749`.

[3] D. E. A. Checa, *Desarrollo de un sistema de toma de decisiones y consenso grupal aplicado a los motores de recomendación: desarrollo del módulo de manejo de usuarios, grupos y notificaciones para el sistema de toma de decisiones y consenso grupal aplicado a los motores de recomendación*, Quito, ago de 2023. dirección: `http://bibdigital.epn.edu.ec/handle/15000/25087`.

[4] R. F. P. Molina, *Desarrollo de un sistema de toma de decisiones y consenso grupal aplicado a los motores de recomendación: desarrollo de los módulos de presentación de la recomendación y de selección de tópicos del sistema de recomendación para grupos de investigadores que soporte la toma de decisiones y el consenso*, Quito, oct. de 2023. dirección: `http://bibdigital.epn.edu.ec/handle/15000/25084`.

[5] G. S. Aguilar, M. V. G. Sánchez y H. Carrillo, «ViBlioSOM: Visualización de información bibliométrica mediante el mapeo autoorganizado,» *Revista española de documentación científica*, vol. 25, n.º 4, págs. 477-484, 2002.

[6] J. Alhuay-Quispe, A. Estrada-Cuzcano y L. Bautista-Ynofuente, «Analysis and data visualization in bibliometric studies,» *JLIS. it*, vol. 13, n.º 2, págs. 58-73, 2022.

[7] I. Sajovic y B. Boh Podgornik, «Bibliometric analysis of visualizations in computer graphics: a study,» *Sage Open*, vol. 12, n.º 1, pág. 21 582 440 211 071 105, 2022.

[8] E. Schab, R. Rivera, L. Bracco et al., «Minería de datos y visualización de información,» en *XX Workshop de Investigadores en Ciencias de la Computación (WICC 2018, Universidad Nacional del Nordeste)*., 2018.

[9] J. Minguillón, «Introducción a la visualización de datos,» *International journal of computer science and information technology research*, 2016.

[10] S. S. Ajibade y A. Adediran, «An overview of big data visualization techniques in data mining,» *International Journal of Computer Science and Information Technology Research*, vol. 4, n.º 3, págs. 105-113, 2016.

[11] Y. Tu y H.-W. Shen, «Visualizing changes of hierarchical data using treemaps,» *IEEE transactions on visualization and computer graphics*, vol. 13, n.º 6, págs. 1286-1293, 2007.

[12] F. Ormeling y P. Bajos, «Mapas temáticos,» *Recuperado de https://icaci. org/files/documents/wom/06_IMY_WoM_es. pdf*, 2015.

[13] Y. Córdova Viera, J. Martínez Borrego y E. Córdova Viera, «Propuesta de metodología para el diseño de dashboard,» *Revista Cubana de Transformación Digital*, 2021.

[14] V. Krishnan, «Research data analysis with power bi,» *INFLIBNET Centre*, 2017.

[15] M. Angulo Pinedo, «Diseño y desarrollo de cuadro de mandos para el seguimiento y visualización de datos de asignaturas,» *ETSI_Informatica*, 2021.

[16] N. Q. Zhu, *Data visualization with D3. js cookbook*. Packt Publishing Ltd, 2013.

[17] H. Kudale, M. Phadnis, P. Chittar, K. Zarkar y B. Bodhke, «A Review Of Data Analysis And Visualization Of Olympics Using Pyspark And Dash-Plotly,» *International Research Journal of Modernization in Engineering Technology and Science www. irjmets. com@ International Research Journal of Modernization in Engineering*, vol. 4, n.º 6, 2022.

[18] H. Da Rocha, *Learn Chart. js: Create interactive visualizations for the web with chart. js 2*. Packt Publishing Ltd, 2019.

[19] B. Shahid, *Highcharts essentials*. Packt Publishing Ltd, 2014.

[20] K. Jolly, *Hands-on data visualization with Bokeh: Interactive web plotting for Python using Bokeh*. Packt Publishing Ltd, 2018.

[21] AWS, *Patrón e arquitectura hexagonal*. dirección: `https://docs.aws.amazon.com/es_es/prescriptive-guidance/latest/cloud-design-patterns/hexagonal-architecture.html#:~:text=Las%20arquitecturas%20hexagonales%20permiten%20aislar,y%20los%20intermediarios%20de%20mensajes.` (visitado 2024).

[22] Paradigma, *Arquitectura hexagonal en Angular: cómo mejorar la estructura de tus aplicaciones*. dirección: `https://www.paradigmadigital.com/dev/arquitectura-hexagonal-angular-como-mejorar-estructura-aplicaciones/` (visitado 2023).

[23] A. Srivastava, S. Bhardwaj y S. Saraswat, «SCRUM model for agile methodology,» en *2017 International Conference on Computing, Communication and Automation (ICCCA)*, IEEE, 2017, págs. 864-869.

[24] Noetix, «Dashboad Development and Deployment: A Methodology for success,» Noetix Corporation, inf. téc., 2004.

[25] I. Docker, «Docker,» *línea].[Junio de 2017]. Disponible en: https://www. docker. com/what-docker*, 2020.

[26] Angular, *What is Angular?* Dirección: `https://angular.dev/overview` (visitado 2024).

[27] Django, *Django: The web framework for perfectionists with deadlines*. dirección: `https://www.djangoproject.com/` (visitado 2024).

[28] D3, *What is D3?* Dirección: `https://d3js.org/what-is-d3` (visitado 2024).

[29] Bootstrap, *Get started with Bootstrap*. dirección: `https://getbootstrap.com/docs/5.3/getting-started/introduction/` (visitado 2024).

[30] Tailwind, *Taailwind Css*. dirección: `https://tailwindcss.com/` (visitado 2024).

[31] Ngx-charts, *Ngx-xharts Introduction*. dirección: `https://swimlane.gitbook.io/ngx-charts` (visitado 2024).

[32] Medium, *Clean Architecture for Angular Applications*. dirección: `https://medium.com/taager-tech-blog/clean-architecture-for-angular-applications-b7ab140f0d5a` (visitado 2022).

# 6. ANEXOS

## 6.1. Anexo 1:

Prototipos del proyecto en Figma

## 6.2. Anexo 2:

Repositorio del frontend del proyecto en Github

## 6.3. Anexo 3:

Repositorio del backend del proyecto en Github

## 6.4. Anexo 4:

Tareas evaluadas en OneDrive

## 6.5. Anexo 5:

Resultados Encuesta SUS en OneDrive