**DEPARTAMENTO NACIONAL DE PLANEACIÓN**

SUBDIRECCIÓN DE GOBIERNO Y ASUNTOS INTERNACIONALES

**PRODUCTO CONTRACTUAL**

**No. DNP 1406 de 2022**

PRESTAR SERVICIOS PROFESIONALES CON PLENA AUTONOMÍA TÉCNICA Y ADMINISTRATIVA A LA SUBDIRECCIÓN DE GOBIERNO Y ASUNTOS INTERNACIONALES DEPARTAMENTO NACIONAL DE PLANEACIÓN (DNP), EN LA REALIZACIÓN DE ACTIVIDADES RELACIONADAS CON LA ACTUALIZACIÓN Y PILOTAJE DEL SISTEMA DE DECISIÓN AUTOMATIZADA ACLARO, QUE PERMITAN ANALIZAR Y PRODUCIR TEXTOS EN LENGUAJE CLARO Y ASÍ FACILITAR LA COMPRENSIÓN Y TRANSPARENCIA DE LA INFORMACIÓN PÚBLICA.

Elaborado por: Luis Felipe Arteaga Daza

**Bogotá D.C., septiembre de 2023**

Contenido

[*1.* *Introducción* 3](#_Toc145077434)

[*2.* *Antecedentes* 4](#_Toc145077435)

[*3.* *Trabajo Previo y Nueva Versión Prototipo* 4](#_Toc145077436)

[*4.* *Aplicación web* 5](#_Toc145077437)

[*5.* *Arquitectura y Funcionamiento de una Aplicación Web* 6](#_Toc145077438)

[*6.* *Desarrollo Aplicación Web Aclaro* 8](#_Toc145077439)

[*7.* *Implementación e Interfaz de Usuario* 9](#_Toc145077440)

[*7.1 Interfaz de Usuario* 9](#_Toc145077441)

[7.2. KIT IU gov.co 10](#_Toc145077442)

[*8.* *Implementación* 10](#_Toc145077443)

[*9.* *Funcionamiento Prototipo* 12](#_Toc145077444)

[*9.1.* *Corrector Ortográfico* 13](#_Toc145077445)

[*9.2.* *Simplificador de Palabras* 13](#_Toc145077446)

[*9.3.* *Resumen de texto* 14](#_Toc145077447)

[*9.4.* *Puntaje de Legibilidad* 15](#_Toc145077448)

[10. Arquitectura de la aplicación Aclaro 16](#_Toc145077449)

[11. Pruebas de carga ACLARO 16](#_Toc145077450)

[11.1. Resultados 17](#_Toc145077451)

[10.1.1. 50 usuarios 17](#_Toc145077452)

[10.1.2. 100 usuarios 18](#_Toc145077453)

[12.1.3. 200 usuarios 19](#_Toc145077454)

[10.1.4. 300 usuarios 20](#_Toc145077455)

[10.2. Resumen de Pruebas de Carga y Estrés 21](#_Toc145077456)

[Bibliografía 22](#_Toc145077457)

1. ***Introducción***

El uso de un lenguaje claro y accesible por parte de los funcionarios del Estado es fundamental para garantizar que la información llegue de manera efectiva a todas las personas, independientemente de su nivel educativo o cultural. Para lograr esto, los funcionarios del Estado deben evitar utilizar jergas técnicas y términos complicados que puedan resultar confusos o difíciles de entender para la mayoría de la población. En su lugar, deben utilizar un lenguaje sencillo, directo y fácil de comprender. Además, es importante que los funcionarios del Estado presten atención a la estructura y presentación de la información que proporcionan, utilizando párrafos cortos, frases simples y evitando el uso excesivo de tecnicismos o siglas (Garzón, Vanegas, Galeano, & Durán, 2015). En definitiva, la comunicación clara y efectiva por parte de los funcionarios del Estado es esencial para garantizar la transparencia y la participación ciudadana en la toma de decisiones, y debe ser una prioridad en todos los niveles de gobierno.

Aunque los funcionarios tienen acceso a capacitación, información y herramientas ofimáticas para apoyarlos en la redacción documentos al interior de las entidades del estado, tales como decretos, leyes, políticas, tramites, procedimientos, normas, portales web, etc.

Se presentan grandes retos y oportunidades en la mejora de la fluidez y claridad de la información que se genera y se publicada. Una de estas oportunidades en la incorporación de sistemas de Inteligencia Artificial (IA) y aprendizaje automático, los cuales han tenido un desarrollo significativo, en los últimos años (Tewari, 2022). Las IA se han convertido en una herramienta fundamental en muchos ámbitos, desde la atención medica hasta la industria manufacturera, pasando por la comunicación y el entretenimiento, mayormente impulsado por el avance de las tecnologías de procesamiento de datos y la creciente demanda de soluciones autónomas e inteligentes en diversos ámbitos.

Por lo cual incorporar la IA puede ser de gran utilidad en la tarea de simplificar el lenguaje y hacerlo más accesible para todos los ciudadanos. En el ámbito de la comunicación gubernamental, la IA se puede utilizar para analizar el lenguaje utilizado en los documentos y publicaciones oficiales y sugerir alternativas más sencillas y claras. Por ejemplo, se pueden utilizar algoritmos de procesamiento de lenguaje natural para identificar palabras y frases que puedan resultar difíciles de entender para la mayoría de los lectores. La IA puede sugerir alternativas más simples y claras para estas palabras y frases, lo que puede mejorar significativamente la comprensión de los documentos.

Además, la IA también puede utilizarse para generar texto simplificado a partir de documentos complejos. En este caso, se pueden utilizar algoritmos de resumen automático para extraer las ideas principales del texto y presentarlas de manera más sencilla y clara. En definitiva, la IA puede ser de gran utilidad para mejorar el lenguaje claro en la comunicación gubernamental. Sin embargo, es importante tener en cuenta que la tecnología aún está en desarrollo y que se necesitan esfuerzos continuos para asegurar que la IA se utilice de manera responsable y ética. Además, siempre es necesario contar con la revisión y validación humana para garantizar la calidad y la precisión de los textos simplificados.

1. ***Antecedentes***

El lenguaje claro en Colombia tuvo como enfoque orientar a los funcionarios públicos en la transparencia, la rendición de cuentas ante la ciudadanía y fomentar entre los ciudadanos su participación, esto ve plasmado dentro la guía de Guía de Lenguaje Claro para Servidores Públicos de Colombia: (Garzón, Vanegas, Galeano, & Durán, 2015).

*Uno de los fines esenciales del Estado es garantizar el goce efectivo de derechos y el cumplimiento de deberes de los ciudadanos. Esto implica que el Estado pueda transmitir de forma clara y efectiva la información sobre programas, trámites y servicios e informar sobre sus acciones a través de un ejercicio efectivo de rendición de cuentas. En este sentido, el Plan Nacional de Desarrollo 2014 – 2018 “Todos Por un Nuevo País” considera el Buen Gobierno como una de las estrategias transversales para impulsar el desarrollo de un Estado con la arquitectura institucional y las capacidades técnicas, humanas y de infraestructura física y tecnológica, que le permitan consolidar la paz, y contribuir al logro de la equidad social y el mejoramiento de la educación de los colombianos. Es así como el Gobierno Nacional se compromete en fortalecer la formulación de políticas en los ámbitos internacional, nacional y territorial a partir de una cultura de servicio al ciudadano que opere bajo parámetros de transparencia, eficacia e integridad en los servidores públicos. Al respecto, el presidente Juan Manuel Santos ha señalado que “(…) toda la información al ciudadano debe estar en lenguaje claro… Nuestro primer propósito es servir al ciudadano, y para eso tenemos que ser claros, sencillos”1. Igualmente, en el Documento CONPES 3785 de 2013 se define como una de las prioridades “ofrecer a los ciudadanos información en lenguaje claro y comprensible de manera que tengan certidumbre sobre las condiciones de tiempo, modo y lugar en las que podrán solucionar sus inquietudes y gestionar sus trámites”. En este marco, el Gobierno Nacional propone la integración de servicios como una respuesta a la necesidad de: (i) acercar los servicios de la Administración Pública a los ciudadanos; (ii) mejorar y estandarizar la atención; y (iii) fortalecer la institucionalidad del gobierno nacional y las entidades territoriales para la gestión eficiente para el desarrollo y la reducción de brechas sociales y económicas.*

Por lo tanto, es un requerimiento para las entidades del estado aplicar las recomendaciones del uso de lenguaje claro para todos los productos y en general cualquier publicación generada, con el fin tener una mayor efectividad y entendimiento del ciudadano.

1. ***Trabajo Previo y Nueva Versión Prototipo***

En el año 2020 el Departamento Nacional de Planeación – DNP, siendo el líder la Red de Lenguaje Claro en Colombia, que fue creada en el año 2018 en asociación con la: La Cámara de Representantes de Colombia, el Instituto Caro y Cuervo, la Universidad Eafit y la Universidad de los Andes, contacto a la Universidad Nacional de Colombia – Sede Medellín para la elaboración y construcción de un prototipo de software, que sirva como un instrumento a las entidades del estado en la creación y producción de documentos e información escrita que obedezca y se ajuste a las prácticas y lineamiento de lenguaje claro de cara a la comunicación de las entidades con los ciudadanos.

Por lo tanto, la Universidad Nacional de Colombia – Sede Medellín diseño una arquitectura de solución y construyo un prototipo de software, usando un enfoque en lingüística de corpus y computacional, así como modelo de IA, se esta manera permitiendo estimar la legibilidad y recomendado opciones de sustitución de palabras del lenguaje claro del español colombiano, con el fin de mejorar la legibilidad de los textos generado por colaboradores de las entidades públicas, este proyecto se denominó “SISTEMA DE APOYO PARA LA ESCRITURA DE DOCUMENTOS EN LENGUAJE CLARO” (BEDOYA, 2020).

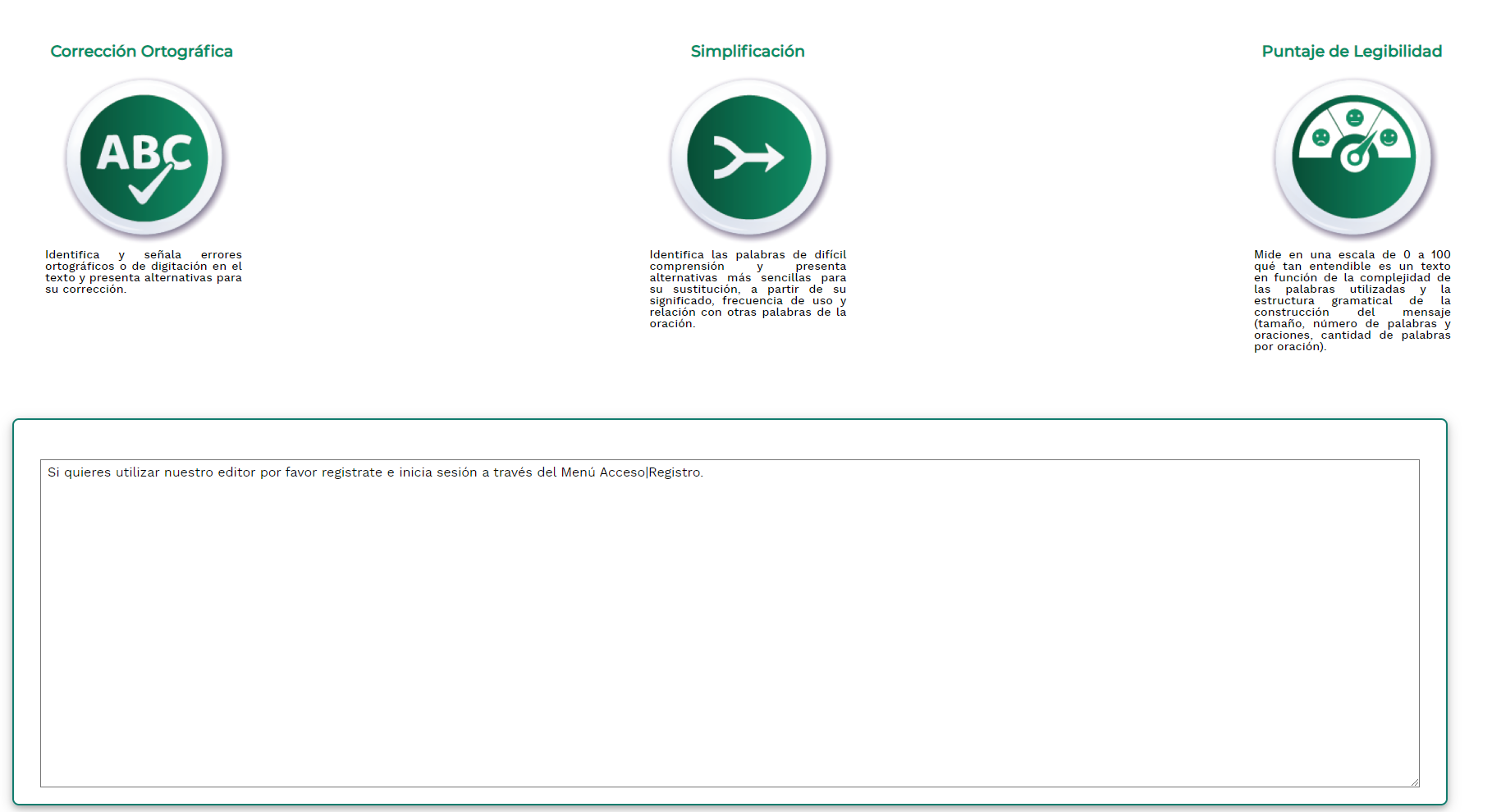
Durante el desarrollo del proyecto se construyó un Corpus (Conjunto cerrado de textos o de datos destinado a la investigación científica.) del español colombiano, a partir de un conjunto de textos obtenidos de varias fuentes digitales y su posterior transformación por técnicas de procesamiento de texto para la creación de varias fuentes de datos. Posteriormente, se construyó un modelo computacional que permitió evaluar cuantitativamente la relación semántica entre un conjunto de palabras, teniendo en cuenta su uso al interior de un Corpus.

También, se desarrolló un modelo de sustitución léxica basado en IA permitiendo obtener un conjunto de palabras similares a una palabra referencia, teniendo en cuenta el contexto de la palabra referenciada, mediante el análisis de frecuencia de uso y la relación semántica. Igualmente, se construyó un módulo de estimación de puntaje de legibilidad de un texto a partir de la implementación e integración de fórmulas de legibilidad de la literatura.

Por último, tenemos el desarrollo y elaboración del prototipo de software, que permita la entrada de un texto, permitiendo estimar el score de legibilidad, visualizar las palabras de un texto que pueden ser sustituidas con base a la frecuencia de uso, y recomendar palabras similares para realizar sustitución léxica. La versión inicial se encuentra en un servidor del DNP [*https://vcorpusprfe/lenguajeClaroWeb/*](https://vcorpusprfe/lenguajeClaroWeb/).

*Ilustración 1: Interfaz de Usuario primer prototipo*

Captura de pantalla de un celular con la imagen de la pantalla de una computadora

Descripción generada automáticamente con confianza media

Fuente: Elaboración propia

Se decidió actualizar a la primera versión del prototipo, los cambios propuestos fueron: rediseñar la interfaz de usuario, crear módulos independientes y actualizar el Corrector Ortográfico, Simplificador y el Puntaje de legibilidad y adicionar en un módulo de Resumen de Texto, dado que los cambios propuestos son demasiados y actualizar la primera versión con los propuestos conllevaría problemas y errores, y se cambió el Corrector Ortográfico y añadió nuevas funcionalidades al módulo de Puntaje de Legibilidad.

Para implementar los cambios anteriormente mencionados se decidió crear una nueva Aplicación Web, donde se integraría las nuevas funciones y módulos.

1. ***Aplicación web***

Una aplicación web es un software o programa informático que se ejecuta en un servidor web y se desarrolla para funcionar a través de un navegador web. En contraste con las aplicaciones móviles o de escritorio tradicionales que se instalan y ejecutan en un ordenador o dispositivo móvil, las aplicaciones web no requieren una instalación específica en el dispositivo del usuario. En cambio, los usuarios pueden acceder a la aplicación utilizando un navegador web estándar en cualquier dispositivo con conexión a Internet (Amazon, 2023).

Las aplicaciones web ofrecen varias ventajas en comparación con otras formas de aplicaciones. Aquí hay algunos beneficios comunes asociados a las aplicaciones web:

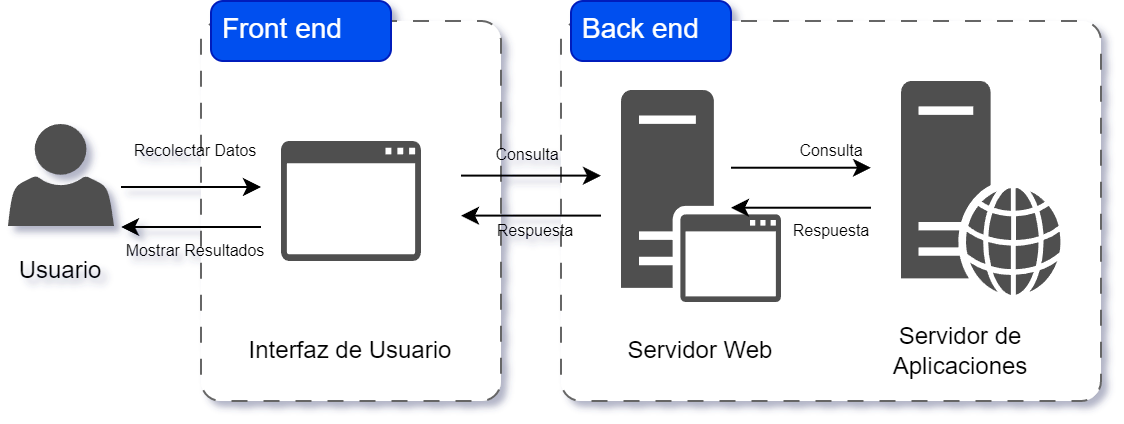
* Accesibilidad: Las aplicaciones web son accesibles desde todos los navegadores web y desde diferentes dispositivos personales, siendo estas una de sus principales ventajas.
* Plataforma independiente: Las aplicaciones web no están limitadas a un sistema operativo en particular. Funcionan en múltiples plataformas, como Windows, macOS, Linux, iOS y Android, lo que las hace más versátiles y accesibles para una audiencia más amplia.
* Desarrollo eficiente: En comparación con las aplicaciones de escritorio tradicionales, las aplicaciones web suelen requerir una inversión inicial menor. No es necesario desarrollar versiones para diferentes sistemas operativos y dispositivos, lo que puede ahorrar tiempo y recursos.
* Escalabilidad: Las empresas o entidades que utilizan aplicaciones web pueden agregar acceso a más usuarios cuando sea necesario, muchas veces sin necesidad de infraestructura adicional o hardware costoso.
* Actualizaciones y mantenimiento centralizados: Las actualizaciones de una aplicación web se realizan en el servidor, lo que significa que todos los usuarios acceden automáticamente a la versión actualizada. No es necesario actualizar la aplicación en cada dispositivo individualmente, lo que facilita el mantenimiento y garantiza una experiencia consistente para todos los usuarios.

En resumen, las aplicaciones web ofrecen ventajas en términos de accesibilidad, plataforma independiente, actualizaciones, costos reducidos, y actualizaciones instantáneas, estas ventajas las convierten en una excelente opción para los usuarios y un equipo de desarrollo.

1. ***Arquitectura y Funcionamiento de una Aplicación Web***

La arquitectura de aplicaciones web es un marco o esquema que consiste en las relaciones e interacciones entre los diferentes elementos de una aplicación, como la interfaz de usuario, las bases datos y los sistemas de middleware que comunican los elementos de la aplicación. Además, comprende un conjunto de componentes y una descripción de su interacción lógica (sharma, 2022). A continuación, se muestra la arquitectura general de una aplicación web Aclaro que se compone de las relaciones e interacciones entre los componentes de la aplicación, como los sistemas de middleware con la interfaz de usuario.

*Ilustración 2: Interfaz de Usuario primer prototipo*



Dentro de la aplicación web se cuenta con tres elementos fundamentales la interfaz de usuario que se denomina Front end es la parte del aplicativo web que permite a los usuarios ver los recursos e interactuar con las funcionalidades, y el Servidor Web que se encarga de manejar las solicitudes del usuario y Servidor de Aplicaciones que ejecuta las tareas solicitadas lo que llama el Back end de la aplicación es la parte de una aplicación web que se encarga del procesamiento y la comunicación. Una aplicación web utiliza una combinación de secuencias de comandos del Front end y secuencias de comandos Back end para funcionar, por lo tanto, el funcionamiento de una aplicación web se puede describir de la siguiente manera:

* El usuario crea una solicitud al servidor web un dispositivo conectado a internet a través la interfaz de usuario de la aplicación.
* El servidor web envía esta solicitud al servidor de aplicaciones web.
* El servidor de aplicaciones web ejecuta la tarea solicitada y luego genera los resultados de los datos requeridos.
* El servidor de aplicaciones web envía esos resultados al servidor web.
* El servidor web lleva la información solicitada al dispositivo conectado a internet
* La información solicitada aparece en la pantalla del usuario.

Este ciclo de se repite cada vez que el usuario interactúa con la aplicación web.

Uno de los cambios propuesto para la aplicación es crea módulos independientes para cada una de las nuevas herramientas propuestas. Con el fin de tener cada módulo aislado del resto y solo compartiendo recursos computacionales se puede lograr por medio del uso de API’s, esta corresponde a la interfaz de programación de aplicaciones (API por sus siglas en inglés), las cuales permiten la interacción entre programas, máquinas o cualquier sistema que requiera una conexión y que no necesariamente pueda hacerse de manera directa, sino que por medio de un lenguaje universal (usualmente archivos *JSON* o diccionarios), estas reciben una solicitud (o entrada) y devuelve un resultado (o salida), por lo que se crearon API’s para cada uno de los nuevos módulos, permita contener cada funcionalidad de manera separada permitiendo contener cada funcionalidad de manera separada.

1. ***Desarrollo Aplicación Web Aclaro***

Dentro del marco del desarrollo de la aplicación web, esta se desplego en un ambiente de prueba en un servidor del DNP, para el desarrollo se utilizaron las siguientes herramientas, Frameworks y lenguajes de programación para el desarrollo del Front endde la aplicación fue realizado en Angular, HTML, CSS, JavaScript y Bootstrap. El back end fue realizado en FastAPI y Python. Se crearon cuatro herramientas integradas dentro de la aplicación para cumplir con las necesidades del proyecto Aclaro, estas herramientas son:

* **Corrector Ortográfico:** Identifica errores ortográficos, gramaticales, puntuación y sugiere alternativas para su corrección**,** para la su implementación se utilizó LanguageTool es una herramienta de gramática de código abierto, también conocida como el corrector ortográfico de OpenOffice (LanguageTool, 2022).
* **Simplificador de palabras:** Identifica en un texto las palabras de difícil comprensión y presenta alternativas más sencillas para su sustitución, a partir de su significado, frecuencia de uso y relación con otras palabras de la oración, para su implementación se utilizó BERT (Bidirectional Encoder Representations from Transformers) es un modelo de lenguaje basado en redes neuronales que ha demostrado un rendimiento sobresaliente en tareas relacionadas con el procesamiento del lenguaje natural (NLP, por sus siglas en inglés). Fue presentado por Google en 2018 y ha sido ampliamente adoptado en la comunidad de investigación y desarrollo de NLP (Devlin, Chang, Lee, Toutanova, & Kristina, 2018). BERT utiliza una arquitectura basada en Transformers, que son modelos de aprendizaje automático que pueden capturar las relaciones a largo plazo en secuencias de palabras o tokens. Una característica clave de BERT es que es "bidireccional", lo que significa que puede tener en cuenta tanto el contexto anterior como el posterior al predecir una palabra determinada.
* **Resumen de texto:** Genera una versión más corta de un texto, conservando tanto su fluidez como la información más relevante del texto original, existen dos tipos de resumen, El resumen abstractivo es una técnica de se utiliza para generar resúmenes de texto mediante la creación de un nuevo texto que resume el contenido esencial del original en un formato diferente, para este se implementó el mT5 (multilingual T5) que es un modelo de lenguaje desarrollado por Google que se basa en la arquitectura T5 (Text-to-Text Transfer Transformer). A diferencia de otros modelos de lenguaje que están diseñados para un solo idioma, mT5 es capaz de manejar múltiples idiomas en una sola instancia (Xue, et al., 2020). La arquitectura T5 se basa en Transformers y utiliza el aprendizaje por transferencia, lo que significa que el modelo se entrena en una amplia variedad de tareas de procesamiento de lenguaje natural (NLP) utilizando una misma estructura. Estas tareas pueden incluir traducción automática, resumen de texto, generación de código, respuesta a preguntas y más.

El resumen extractivo es una técnica que implica identificar las frases o párrafos más importantes o relevantes de un texto y resumirlos en un nuevo texto más corto, para este se implementó, LexRank que es un algoritmo de resumen automático basado en grafos que utiliza el concepto de ranking de importancia de las oraciones. En lugar de utilizar el enfoque tradicional de contar palabras o frases clave, LexRank construye un grafo donde las oraciones son los nodos y las conexiones entre las oraciones representan la similitud entre ellas. Luego, calcula el puntaje de importancia de cada oración en función de la similitud con otras oraciones en el texto. Las oraciones con los puntajes más altos se seleccionan como parte del resumen final. (Erkan, Radev, & R, 2004) LexRank es especialmente útil para resumir textos largos, como artículos de noticias o documentos extensos, y ha demostrado ser efectivo en capturar la información más relevante y significativa del texto original.

* **Legibilidad del Texto:** Calcula la legibilidad de un texto mediante el índice Flesch-Szigriszt y proporciona una vista 360 de las características asociadas a la interpretación del Lenguaje claro.

Se desarrollo un módulo dentro la interfaz de usuario y una Api para cada una de las herramientas de manera separa, además se almacena características de los textos ingresados por el usuario para cada una de las interacciones en un archivo CSV(del inglés *comma-separated values*) para cada una de las herramientas, cada archivo CSV contiene la fecha y hora en que usuario utilizo la herramienta y características básicas del texto y la respuesta generada por la aplicación.

Ilustración 3 Esquema Aplicativo Web Aclaro

**Diagrama

Descripción generada automáticamente**Fuente: Elaboración propia.

1. ***Implementación e Interfaz de Usuario***

***7.1 Interfaz de Usuario***

Una interfaz de usuario, generalmente conocida como IU o UI por sus siglas en inglés (User Interface), es el medio por el cual una persona puede interactuar con un sistema informático, ya sea un dispositivo, software o aplicación. Se le puede dominar el punto de contacto y comunicación entre un usuario y la aplicación.

La interfaz de usuario presenta la información, datos y elementos visuales necesarios para que el usuario que utiliza la aplicación pueda interactuar con el sistema de manera efectiva y compresible, y esto se logra con botones, menús, formularios, icono, ventanas, entre otros elementos gráficos y visuales.

Como el objetivo principal de una interfaz es que la interacción entre el usuario y la aplicación sea intuitiva, fácil de usar y eficiente. Por lo tanto, una buena interfaz debe ser clara, coherente, concisa y estéticamente agradable, tenía en cuenta lo anteriormente mencionado la interfaz para el aplicativo ACLARO se diseñó y creo teniendo esto en mente.

Para mejorar la experiencia del usuario se decidido que cada herramienta implementada tuviera su propia sección con el fin de los usuarios pudieran no se vieran obligados a utilizar todas las herramientas simultáneamente, sino que utilizaran las herramientas que ellos consideran pertinentes. Además, de la creación de otras secciones que permitan presentar y mostrar el funcionamiento de cada herramienta y reportar errores.

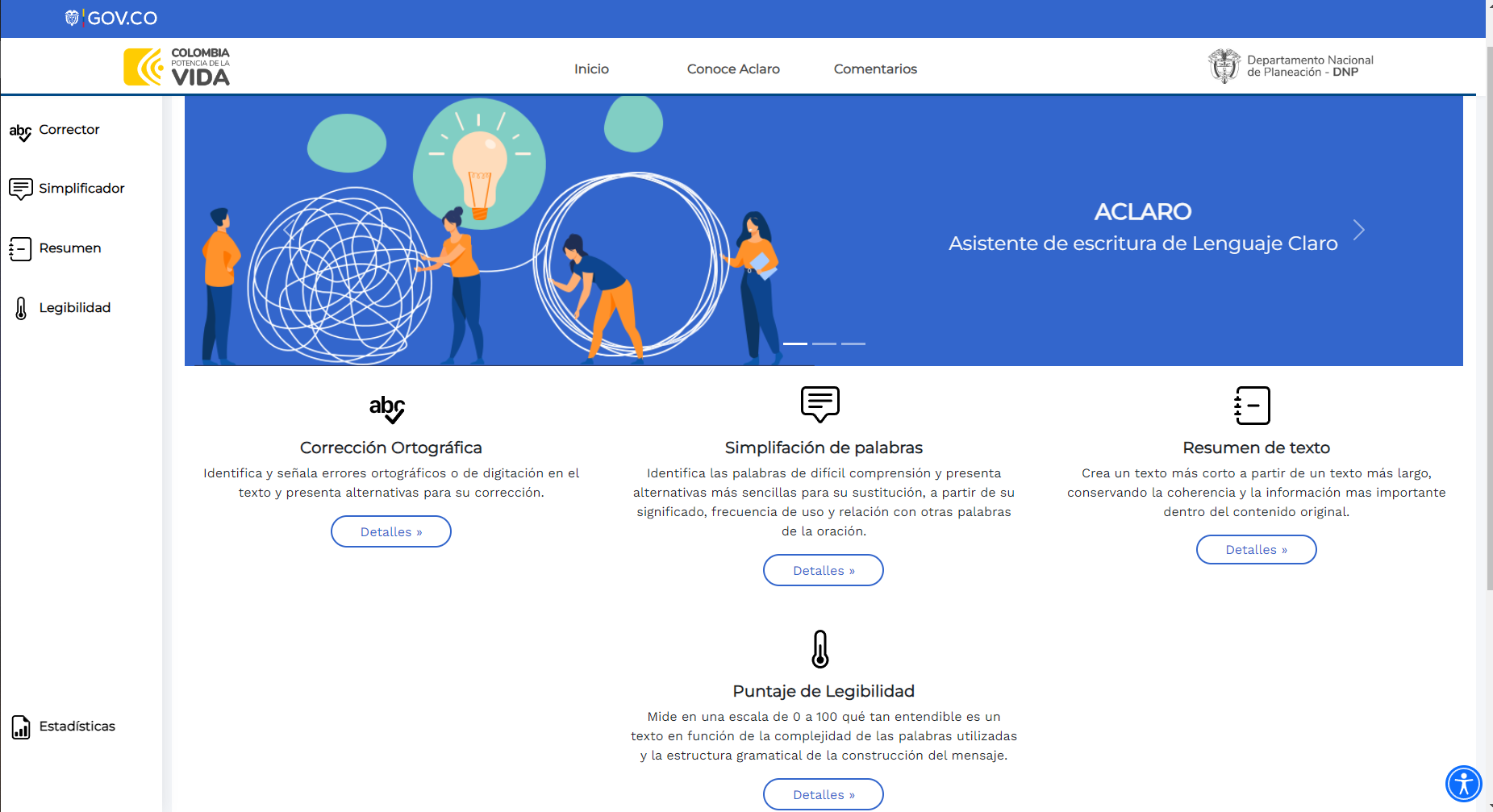
**7.2. KIT IU gov.co**

Es un conjunto de elementos gráficos y recursos visuales predefinidos que se utilizaron para el diseño de la interfaz de usuario de la aplicación ACLARO, con el fin de cumplir los lineamientos de integración del Gobierno Digital del Estado Colombiano, esto lineamientos fueron consultados en (GOV.CO, 2023) y (GOV.CO, GitLab, 2023), los elementos que fueron integrados de acuerdo a los lineamientos fueron botones, iconos, encabezado, pie de página, paleta de colores, tipográficas, formularios entre otros.

1. ***Implementación***

El desarrollo se dividió en varias etapas la primera etapa, fue crear una interfaz de usuario que cumpliera con los siguientes requerimientos: la interfaz de usuario fuera atractiva y permita una interacción sencilla para los usuarios, y que navegación sea rápida y precisa. Teniendo en cuenta los requerimientos para la interfaz se desarrolló la siguiente interfaz, la Ilustración 4 muestra el inicio de sección de la interfaz, cada que un usuario ingresa se encuentra con la Ilustración 4 Interfaz inicio de sección, en la parte superior se encuentra un carrusel de imágenes que brinda información sobre el proyecto Aclaro, en la parte lateral izquierda se encuentra una barra lateral que permite al usuario seleccionar la herramienta deseada, en la parte central e inferior se da una pequeña descripción de cada una de las herramientas y un botón que permite obtener más detalles de cada herramienta como se muestra en Ilustración 5.

Ilustración 4 Esquema Aplicativo Web Aclaro



Fuente: Elaboración propia.

Ilustración 5 Esquema Aplicativo Web Aclaro

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Fuente: Elaboración propia.

También se cuenta con una sección de comentarios como se muestra en la Ilustración 6 donde los usuarios pueden reportar errores, califican el funcionamiento y dejar comentarios de aspectos a mejorar dentro de la aplicación. Adicionalmente se tiene una sección de “Estadísticas” donde se muestran las estadísticas de las interacciones como se muestra en la Ilustración 8.

Ilustración 6 Esquema Aplicativo Web Aclaro

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente con confianza media

Fuente: Elaboración propia.

Ilustración 7 Esquema Aplicativo Web Aclaro

Interfaz de usuario gráfica, Aplicación, Correo electrónico

Descripción generada automáticamente

Fuente: Elaboración propia.

1. ***Funcionamiento Prototipo***

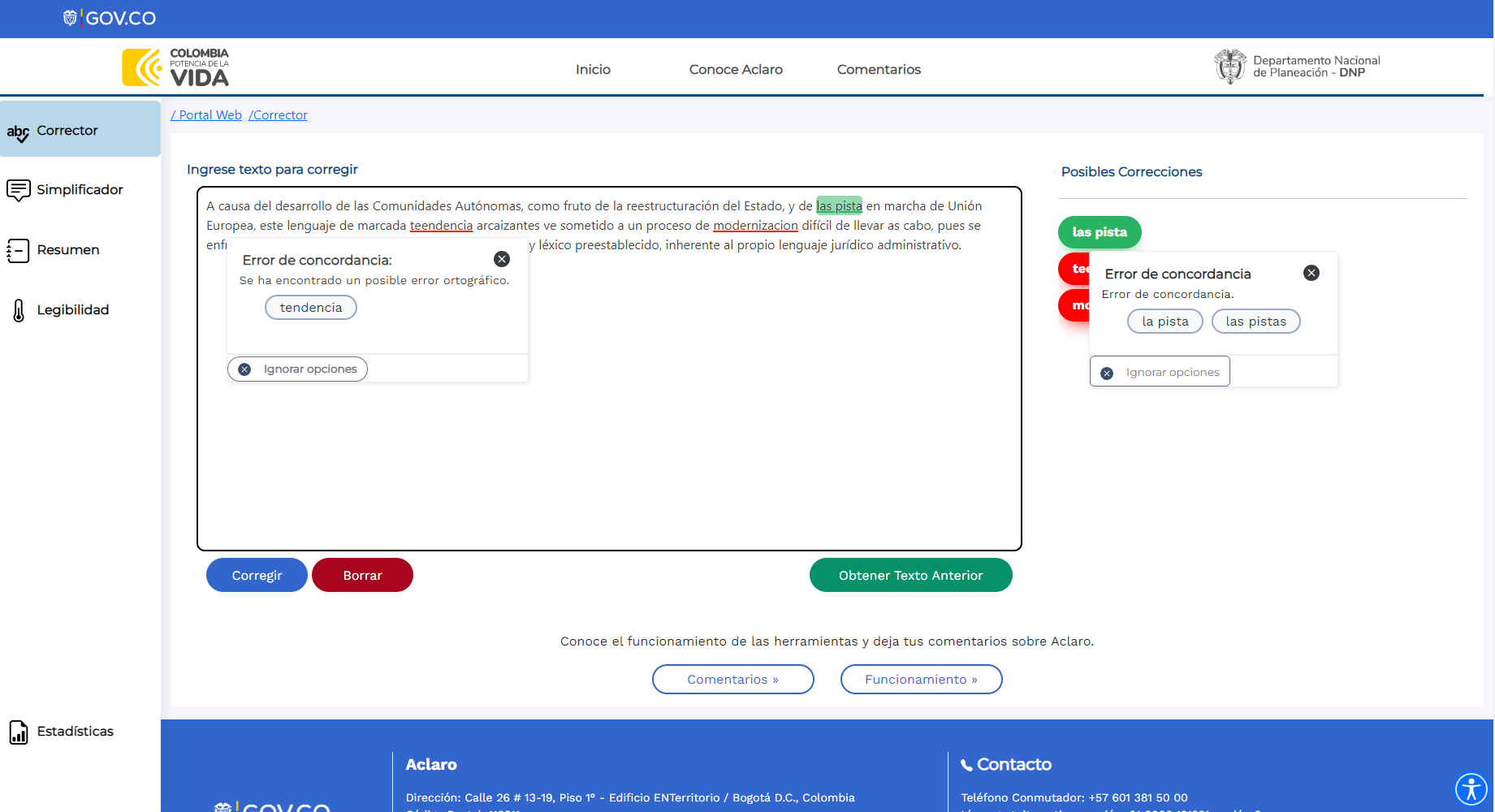
El prototipo cuenta con cuatro herramientas de las cuales se describirá su funcionamiento a continuación.

* 1. ***Corrector Ortográfico***

El usuario copia o digita el texto que desea corregir como se muestra como se muestra en la imagen 8. La interfaz tiene tres bonotes “***Ejecutar***”, “***Borrar***” y “***Obtener Texto Anterior***”, el primero envía una solicitud al servidor de aplicaciones para generar las correcciones, el segundo limpia el texto ingresado y el tercero obtiene texto de otras herramientas guardando las modificaciones hechas por la herramienta anterior.

Cuando el servidor de aplicaciones recibe la solicitud, revisa el texto en busca errores, si el texto tiene errores crea una lista con todas las posibles correcciones ortográficas, gramaticales y de estilo. Se subrayan las secciones del texto que se sugieren que sean corregidas y al dar clic en la palabra del campo de texto subrayada, se presentara una ventana emergente con el tipo de error y lista con las posibles correcciones. También se presenta el botón de “Ignorar opciones”, que simplemente elimina la sugerencia de corrección. También se mostrará la lista con los errores en la sección de Posibles correcciones dar clic sobre el elemento de la lista, se despliega una lista con las posibles correcciones, así como indicando el tipo de error. También se presenta el botón de “Ignorar opciones”, que simplemente elimina la sugerencia de corrección.

Ilustración 8 Interfaz Corrector Ortográfico



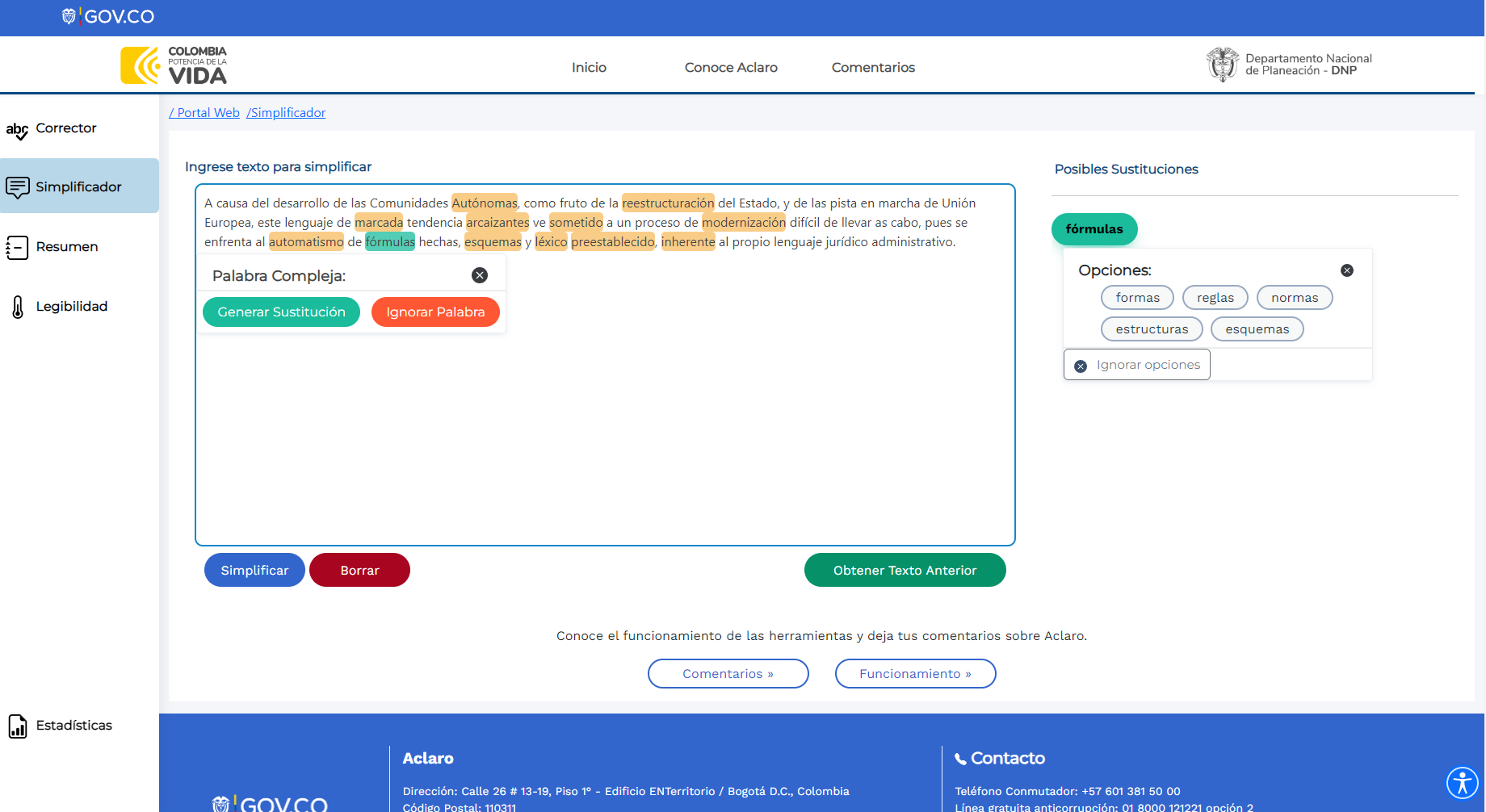
Fuente: Elaboración propia.

## ***Simplificador de Palabras***

El usuario copia o digita el texto que desea corregir como se muestra como se muestra en la Ilustración 9. La interfaz tiene tres bonotes “***Ejecutar***”, “***Borrar***” y “***Obtener Texto Anterior***”, el primero envía una solicitud al servidor de aplicaciones para generar las correcciones, el segundo limpia el texto ingresado y el tercero obtiene texto de otras herramientas guardando las modificaciones hechas por la herramienta anterior.

El servidor de aplicaciones enviara una lista con las palabras complejas dentro del texto y se resalta la palabra en el texto original, sugiriendo que esa palabra puede ser reemplazada, al dar clic sobre esta palabra de desplegaran dos opciones “Generar Sustitución” que buscara los remplazos para la palabra teniendo en cuenta el contexto de la palabra dentro del texto y se presentaran las opciones en la sección “Posibles sustituciones” para que pueda seleccionarla dando clic el usuario; o la opción “Ignorar Palabra” que simplemente elimina la sugerencia de sustitución.

Ilustración 9 Interfaz Simplificador de palabras



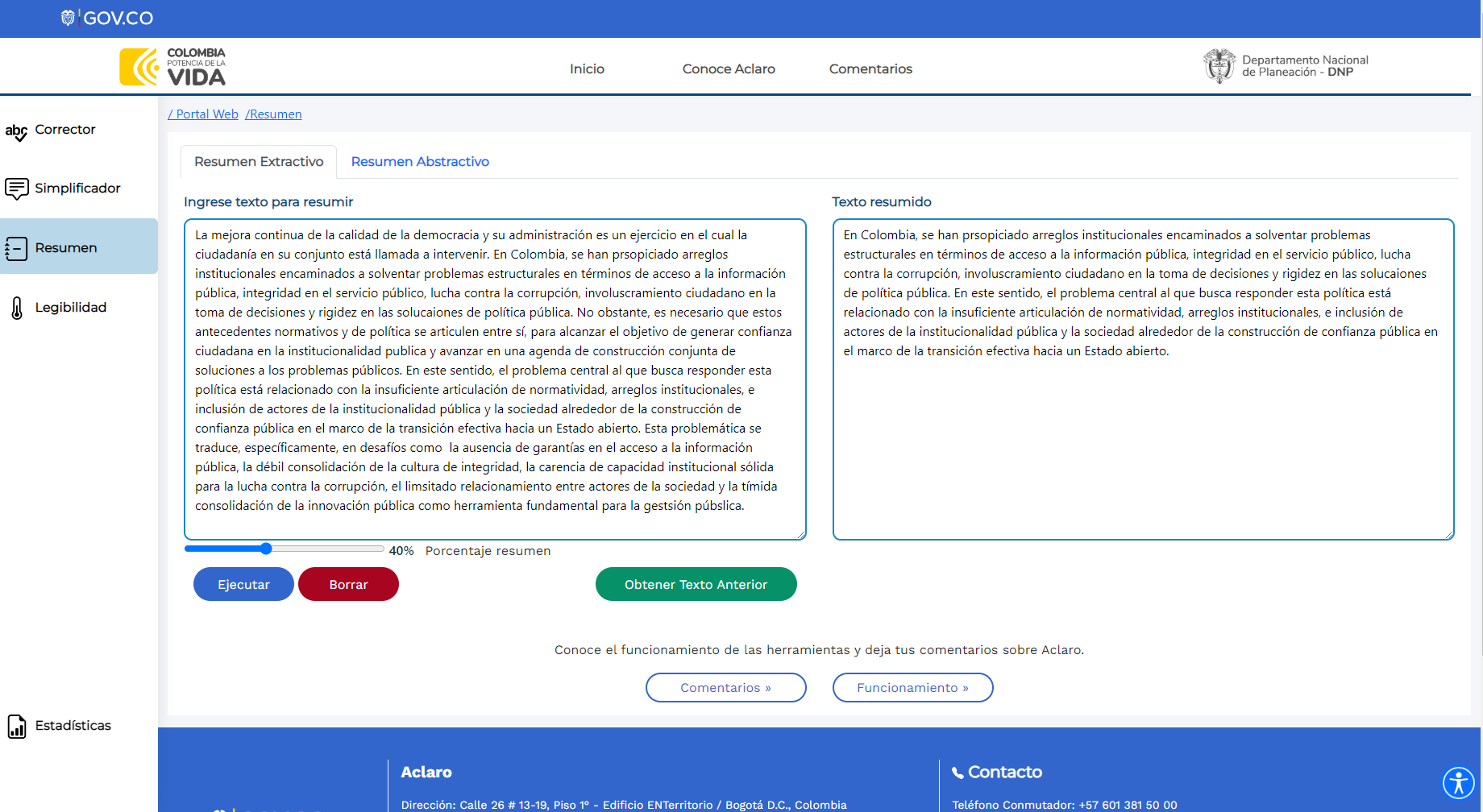
Fuente: Elaboración propia.

## ***Resumen de texto***

El usuario copia o digita el texto que desea resumir como se muestra como se muestra en la Ilustración 10. La interfaz tiene tres bonotes “***Ejecutar***”, “***Borrar***” y “***Obtener Texto Anterior***”, el primero envía una solicitud al Api para generar el resumen del texto ingresado, el segundo limpia el texto ingresado y el tercero obtiene texto de otras herramientas guardando las modificaciones hechas por la herramienta anterior, además de una barra deslizante que permite seleccionar el porcentaje del resumen para el resumen extractivo.

Cuando el servidor de aplicaciones verifica el tipo de resumen Extractivo o Abstractivo, si el usuario solicita el primero también se verifica el porcentaje.

Ilustración 10 Interfaz de Resumen



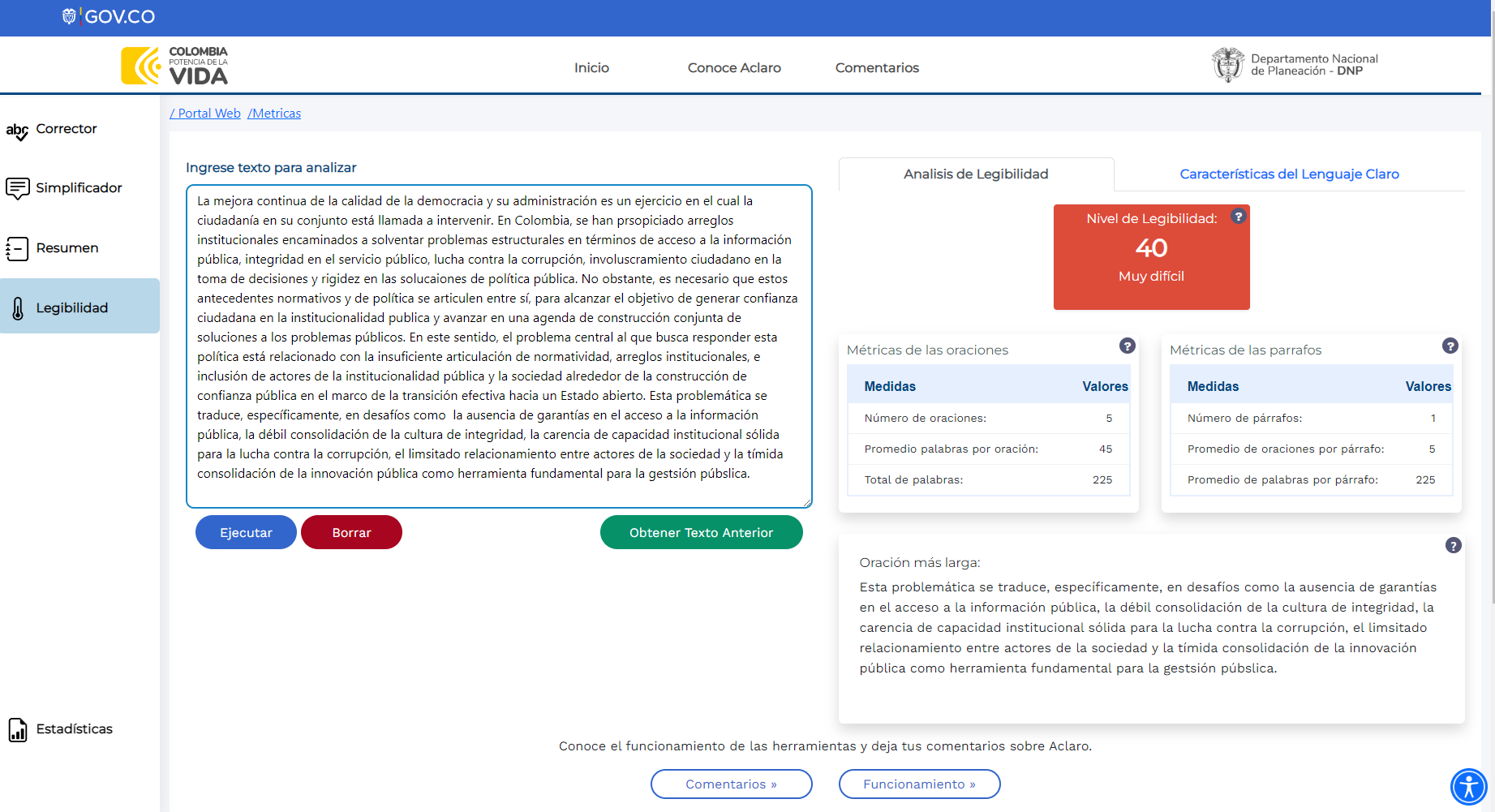
Fuente: Elaboración propia.

## **Puntaje de Legibilidad**

El usuario copia o digita el texto que desea revisar como se muestra como se muestra en la Ilustración 11. La interfaz tiene tres bonotes “***Ejecutar***”, “***Borrar***” y “***Obtener Texto Anterior***” y una sección donde se muestra el análisis, el primero envía una solicitud al Api que genera el análisis y el índice para el texto, el segundo limpia el texto ingresado y el tercero obtiene texto de otras herramientas guardando las modificaciones hechas por la herramienta anterior.

Cuando una solicitud llega al servidor de aplicaciones, se calcula índice Flesch-Szigriszt, así como un conjunto de características que permiten realizar un análisis del texto estos datos son devueltos en un array, algunas de estas caracterizas son número de oraciones, promedio de palabras por oración, numero de verbos y sustantivo entre otros.

Ilustración 11 Interfaz Puntaje de Legibilidad



Fuente: Elaboración propia.

1. **Arquitectura de la aplicación Aclaro**

La arquitectura de una aplicación web se refiere a la estructura general y al diseño de alto nivel que define cómo se organiza y se comunican sus componentes principales. (Cita) Esta estructura se crea para garantizar la eficiencia, escalabilidad, mantenibilidad y rendimiento de la aplicación web. Por lo que le diseño de la aplicación se pensó teniendo en mente los siguientes atributos.

**Usabilidad**: Durante el diseño de la página se tuvieron en cuenta las siguientes características simplicidad, responsividad, visibilidad, flexibilidad, navegación e interacción todo enfocado en las prácticas de diseño de Interfaz de Usuario, para de esta manera asegurar que la plataforma tenga una estructura visual y atractiva, coherente y amigable en la interacción del usuario.

**Seguridad**: Se implementará los elementos de seguridad que protejan el sistema y la información. Durante la interacción de un usuario no se recolectará información de este, por lo tanto, no se violará o se realizará uso indebido de la información del usuario, con el fin de cumplir con las normas de Protección de Datos Personales.

**Disponibilidad:** El software se implementará en la infraestructura del DNP para aprovechar la confiabilidad y el acceso a los recursos disponibles en ese entorno, lo que permitirá la disponibilidad constante del sitio para el usuario final. Además, el riesgo de que se produzcan fallos en la disponibilidad del sitio será bajo**.**

**Mantenibilidad**: durante el desarrollo se utilizaron buenas prácticas y patrones de diseño en el código de la solución de software, además de la implementación de las últimas versiones disponibles de las herramientas utilizadas para la construcción de la solución, permiten que sea fácil modificar, mejorar o adaptar el producto en iteraciones futuras.

**Interoperabilidad:** Se realizará mediante la integración de APIs y Servicios que permitan la comunicación del usuario con los modelos de Inteligencia Artificial y Aprendizaje de Maquina, de esta manera permitiendo que cada herramienta sea independiente y tenga su propia arquitectura**.**

**Desempeño**: Con el fin de garantizar el desempeño de la herramienta se creará el plan de pruebas y se registrará la evidencia.

**Accesibilidad:** El desarrollo será adaptado para el uso de herramientas que permitan la accesibilidad de personas con limitaciones y el uso de software que faciliten su uso, garantizado los cambios de contraste, cambios de tamaño de letra, lectura de etiquetas, mapa de sitio etc. Se realizarán las pruebas para validar que la plataforma permita su navegación e interacción de forma efectiva para todos los usuarios.

**Confiabilidad:** Se plantea una arquitectura, sencilla, escalable, bajo software libre y que permite su actualización y gestión de forma sencilla. Al interior de la aplicación, se plantea este desarrollo de forma modular. Cada uno de los modelos implementados funciona independientemente y puede ser modificado sin afectar los demás módulos.

**Calidad:** Se atienden las recomendaciones y mejores prácticas en el desarrollo de software. Se implementan de acuerdo a los manuales técnicos de MinTic(Kit\_UI) y se da Cumplimiento de los lineamientos y recomendaciones, seguimiento a la Metodología de desarrollo y mantenimiento de sistemas de informacion de la OTSI, documentación y pruebas del DNP.

**Diseño:** El diseño de la plataforma debe estar acorde a los lineamientos de diseño establecidos para el acondicionamiento gráfico de integración a GOV.CO, teniendo en cuenta el Kit UI del gobierno.

* 1. **Frameworks , modelos de lenguaje y herramientas.**

La plataforma desarrollada no hace uso de bases de datos, ya que se realiza el procesamiento a través de los modelos ya entrenados. Presentan los resultados de acuerdo con los textos analizados.

Los frameworks, modelos de lenguaje y herramientas utilizados para el desarrollo de la solución son los siguientes:

|  |  |  |
| --- | --- | --- |
| Nombre | Tipo | Ventajas |
| Angular v16 | Framework de Javascript para aplicaciones Web. | - Código abierto mantenido por Google.  - Soporte de Google y excelente soporte de la comunidad.  - Compatibilidad móvil y de escritorio  - Ofrece una codificación modulable y escalable. |
| Boostrap 5.3.0 | Framework CSS y Javascript. | - Creación de interfaces limpias.  - Diseño responsive. |
| FastAPI 0.85.1 | Framework para desarrollo WEB | - Framework moderno y rápido.  - Ofrece un alto rendimiento.  - Rápido de programar e intuitivo |
| BERT | Modelo de lenguaje | - Modelo de lenguaje natural desarrollado por Google en 2018  -Tiene la capacidad para comprender el contexto de las palabras y frases en una oración. |
| LexRank | Algoritmo de resumen | -Utilizado ampliamente en la industria y la investigación |
| LanguageTool 5.5 | Herramienta de software | - Código abierto  - Corrección de gramática y ortografía |
| mT5 | Modelo de lenguaje | -Modelo de lenguaje natural desarrollado por Google en 2020  - Procesa texto en más de 100 idiomas diferentes  - Puede realizar múltiples tareas de procesamiento de texto (Resumen, traducción) |

* 1. **Vista de la Arquitectura.**

**12.2.1. Diagrama de clases**

**(Explicación de que es un diagrama de clases)**

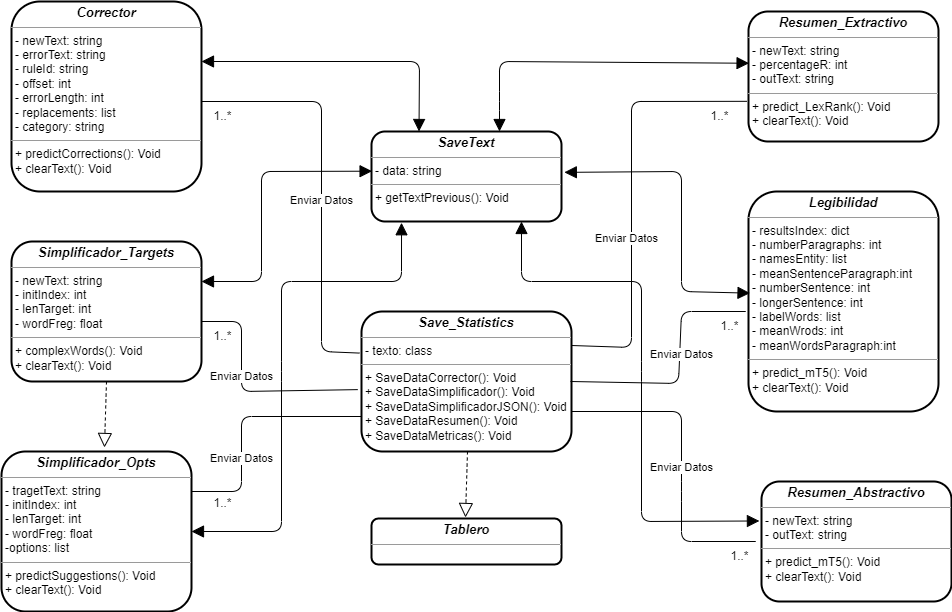


Ilustración 1. Vista lógica Diagrama de Clase

El diagrama de clases ilustra las clases que permiten la funcionalidad y la interacción del usuario con los modelos de IA y Aprendizaje de Máquina, se crearon una o dos clases para cada una de las herramientas. La clase *Corrector* se encarga de la interacción de los usuarios con la herramienta de corrección ortográficas, los atributos de la clase esta destinados a distinguir los errores dentro de texto y mostrar las posibles opciones de reemplazo, el método *predictCorrrections* se encarga de la interacción con le modelo que se encarga de las correcciones.

La herramienta de simplificación de texto cuenta con dos clases, la primera clase *Simplificador\_Targets* es la encargada de resaltar las palabras complejas dentro del texto, los atributos se encargan debrindar la información necesaria para resaltar, el método de *complexWords* interactúa con el modelo que se encarga de encontrar las palabras complejas y la segunda clase *Simplicador\_Opts* se encargar de estimar los posibles reemplazos para la palabras complejas con los datos brindados de la clase *Simplificador\_Targets,* el método *predictSuggestions* se encargar con la interacción de con el modelo de IA BERT.

La herramienta de resumen de texto cuenta con dos clases independientes que se encargadas de realizar el resumen de texto por los dos métodos propuesto. La clase *Resumen\_Extractivo* utiliza el método *predict\_LexRank* para interactuar con el modelo LexRank y la clase *Resumen\_Abstractivo* utiliza el método predict\_mT5 para generar el resumen utilizando el modelo de IA mT5.

La herramienta de legibilidad utiliza la clase *Legibilidad* y el método *predictMetrics* para calcular el puntaje de legibilidad y las características de lenguaje claro de cada texto ingresado por el usuario.

La clase *saveText* permite almacenar el texto ingresado en una herramienta por un usuario durante una sesión, se almacenan todos los cambios en el texto y permite mediante el método *getTextPrevious* utilizar el texto modificado en otras herramientas. Por último, la clase *Save\_Statistics* almacena las interacciones de los usuarios con cada una de las herramientas en archivos CSV y JSON, se crearon diferentes métodos para cada una de las herramientas, los archivos generados son utilizados en el tablero.

10.2.2. **Vista de desarrollo**

**(Explicación)**

Diagrama

Descripción generada automáticamente

Ilustración 2. Vista de desarrollo

Podemos identificar los siguientes componentes:

Servidor Aplicación Web Aclaro, es el canal por el cual los usuarios acceden además contener el código de la IU.

Servicios Webs, la solución conta de cinco servicios web, cuatro de ellos están destinados a la comunicación entres servidor wed y las APIs que se encargar de gestionar los modelos de IA, Aprendizaje de Maquina y algoritmos necesarios, estos servíos son *Servicio Web Corrector,**Servicio Web Simplificador, Servicio Web Resumen, Servicio Web Simplificado.* Para que un usuario pudiera obtener el texto que utilizar en otras herramientas se creó un servicio denominado *Servicio Web Obtener texto Anterior* que permite la comunicación entre los servicios anteriores.

APIs, se encargar de gestionar los modelos de IA, Aprendizaje de máquina y algoritmos necesarios para cada herramienta se creó una APIs para cada uno de las modelo y algoritmos para cada una de las herramientas con el fin de que le funcionamiento de cada herramienta fuera independiente las APIs creadas para las herramientas son *API Corrector Ortográfico y gramatical, API Algoritmo Palabras Complejas, API Modelo IA BERT, API Algoritmo LexRank, API Modelo IA mT5, API Algoritmo Métricas*

*Función Almacenar Interacciones* se creó con el fin de medir la trazabilidad dentro de la plataforma registrando las interacciones que un usuario realiza dentro de la plataforma, estas interacciones se registran que cada vez que un usuario realiza una petición a una API, los registros almenados se utilizan para alimentar el tablero echo Power BI, al cual tienen acceso los usuarios dentro de la página web

**10.2.3. Vista de procesos**

**(Definición)**

Se presenta los diagramas que detallan el funcionamiento e interacción de los componentes de cada una de las herramientas de la App Aclaro.

Diagrama

Descripción generada automáticamente

Ilustración 3. Diagrama de secuencia – Corrector Ortográfico

A través del diagrama anterior se modela la lógica de operación de la herramienta de corrección de texto, el usuario ingresa el texto interactuando con la interfaz, envía la petición la API y esta envía la respuesta. La interfaz muestra los errores el usuario y se encuentran errores en el texto ingresado.

Diagrama

Descripción generada automáticamente

Ilustración 4. Diagrama de secuencia – Simplificador de texto

A través del diagrama anterior se modela la lógica de operación de la herramienta de simplificación. Como primer paso el usuario ingresa el texto interactuando con la interfaz, se envía una petición a la API de palabras complejas que identifica complejas en el texto ingresado, posteriormente el usuario envía a la API simplificador las palabras complejas que desea reemplazar en el texto.

Diagrama

Descripción generada automáticamente

Ilustración 5. Diagrama de secuencia – Resumen de texto

A través del diagrama anterior se modela la lógica de operación de la herramienta de resumen. Como se observa la herramienta de resumen esta dividida en dos partes el resumen extractivo que funciona con la API del algoritmo LexRank y resumen abstractivo que funciona con la herramienta

primer paso el usuario ingresa el texto interactuando con la interfaz, se envía una petición a la API de palabras complejas que identifica complejas en el texto ingresado, posteriormente el usuario envía a la API simplificador las palabras complejas que desea reemplazar en el texto.

Diagrama

Descripción generada automáticamente

Ilustración 6. Diagrama de secuencia – Legibilidad de texto

Por último, tenemos el diagrama de la herramienta de legibilidad, el usuario interactúa con la interfaz para ingresar el texto, se envía la petición a la API y esta devuelve las métricas para el texto ingresado.

**10.2.4. Diagrama de actividades**

**(Definición)**

Diagrama

Descripción generada automáticamente

Ilustración 7. Diagrama de actividades.

A través de este diagrama se ilustrar las actividades macro a desarrollar entre el usuario y el sistema, dando claridad del momento de su ejecución y las acciones ejecutadas. Se presenta un único punto de inicio de la interacción, ilustración de los buques que se pueden presentar durante su uso y el posible fin de la interacción, ya que al ser una aplicación web, el usuario puede finalizar su interacción en cualquier momento, al cerrar la ventana del navegador.

**10.2.5. Diagrama de Casos de Uso**

**(Definición)**

Diagrama

Descripción generada automáticamente

Ilustración 8. Diagrama Casos de Uso.

A través del anterior diagrama se presentan las principales características del sistema Aclaro, donde se observa el funcionamiento las herramientas que lo componen las cuales son las herramientas de corrección, simplificación, resumen y legibilidad, así como la sección del tablero en Power BI y la sección de comentarios.

**10.2.6. Vista física.**

**(Definición)**

A continuación, se presenta la vista física del sistema que será desplegado en un servidor único del DNP

Diagrama

Descripción generada automáticamente

Ilustración 9. Diagrama de despliegue

El despliegue de la plataforma del Aclaro tendrá un Servidor con Microsoft Windows Sever donde estará alojada el aplicado que tiene los siguientes componentes.

* Servidor web: que contiene la aplicación Web basada en Angular.
* Servidor de Aplicaciones: que contiene las APIs del sistema, basado en Python y FastAPI
  1. Dimensionamiento - Crecimiento

Como esta primera es un prototipo que se pondrá a prueba, se planean futuras actualizaciones que mejoren los modelos de inteligencia artificial, así como agregar nuevas funcionalidades, para asegurar y mantener la confiabilidad en el rendimiento de la plataforma durante los próximos 3 años, las actualizaciones se centrarán en entrenar nuevos modelos y agregar nuevas funcionalidades en el sistema.

* En cuanto al Back end y Front end, se necesita un disco duro 10 GB actualmente para los dos componentes, al agregan nuevas actualizaciones este valor se duplicaría para evitar problemas de almacenamientos.
* Para los modelos de lenguaje actuales implementados ocupan el siguiente espacio y ocupan 17GB de RAM:

|  |  |
| --- | --- |
| **Modelo** | **Espacio en disco** |
| **BERT** | 1 Gb |
| **mT5** | 3.6 Gb |

Se plantea mejorar el modelo mT5, existen varias versiones de este modelo estamos trabajando con la versión más pequeña, las versiones superiores ocupan:

|  |  |
| --- | --- |
| **Modelo** | **Espacio en disco** |
| **mT5** | 3.6 Gb |
| **mT5-base** | 7 Gb |
| **mT5-large** | 14.8 Gb |
| **Total** | 25.4 |

Los modelos anteriores se pueden entrenar para varias tareas en específico como resumen de texto, clasificación, preguntas y respuesta entre otras, por lo que se necesitaría varias modelos si se quiere implementar más funcionalidades, al utilizar más modelos se plantea un total de 32GB este valor puede cambiar ya que es muy difícil predecir qué cantidad de modelos se utilizaran.

Por lo anteriormente mencionado se espera que los modelos en el futuro ocupen más espacio dentro del sistema por lo que se plantea 80GB, para los próximos 3 años

|  |  |  |
| --- | --- | --- |
|  | **Espacio en disco**  **2023** | **Espacio en disco**  **2024-2025-2026** |
| **Modelos IA** | 4.6 GB | 80 GB |
| **Front-end** | 5 GB | 10 GB |
| **Back-End** | 5 GB | 10 GB |
| **Total** | 14.6 GB | 100 GB |

Por lo tanto, se necesitaría un disco duro de 100 Gb para almacenar los modelos de IA, Back end y Front end, 32GB de RAM

1. **Pruebas de carga ACLARO**

Las pruebas de carga son un tipo de prueba de rendimiento que se realiza en aplicaciones, sistemas o infraestructuras tecnológicas para evaluar cómo se comportan bajo una carga o demanda específica de usuarios y determinar la capacidad para manejar la demanda esperada o máxima en condiciones reales. El objetivo principal de las pruebas de carga es identificar posibles problemas de rendimiento, cuellos de botella (en una aplicación es donde se produce una restricción o limitación en el rendimiento) y limitaciones en el sistema antes de que este sea lanzado oficialmente o puesto en producción. Estas pruebas nos permiten medir y evaluar el rendimiento y la estabilidad de la aplicación bajo diferentes escenarios de uso de los usuarios y asegurarnos de que pueda soportar la carga esperada sin problemas (Wikipedia, 2022).

Para realizar pruebas de carga, se simula un gran número de usuarios o transacciones que interactúan con la aplicación de manera simultánea, lo que permite recrear una carga de trabajo realista. Estos usuarios virtuales realizan acciones típicas dentro de la aplicación, como acceder a páginas, enviar formularios, hacer búsquedas, entre otras, para replicar el comportamiento real de los usuarios. Durante las pruebas de carga, se monitorea y registra el rendimiento del sistema, incluyendo el tiempo de respuesta de las solicitudes, la utilización de recursos del servidor, y la tolerancia a fallos.

Estas pruebas de carga son esenciales para identificar y corregir problemas de rendimiento antes de que estos afecten a los usuarios en producción. Lo que nos permite realizar ajustes y optimizaciones para garantizar que el sistema pueda satisfacer la demanda de los usuarios y funcione de manera eficiente y confiable en situaciones reales de uso.

* 1. **Resultados**

Para la realización de las pruebas de carga se utiliza JMeter (Apache JMeter) es una herramienta de código abierto desarrollada por Apache Software Foundati (Foundation, 2022), es una herramienta ampliamente utilizada en el mundo del desarrollo de software para evaluar el rendimiento de aplicaciones y medir cómo se comportan bajo diferentes cargas de trabajo.

A continuación, se presentadas los resultados y conclusiones de las pruebas de estrés y carga del sistema realizadas. Primero se configuro el programa JMeter una vez realizada la configuración de las pruebas, se inician las pruebas para 50, 100, 200 y 300 usuarios con los que se podrá evidenciar el uso de CPU, consultas con mayor consumo y el uso de memoria. Se evaluaron las siguientes secciones: Inicio, Corrector, Simplificador, Resumen abstractivo, Resumen Extractivo y Legibilidad.

A continuación, se presentarán los resultados obtenidos en cada una de las pruebas realizadas, considerando los errores y el punto sobre el cuál el servidor ya no era capaz de responder ninguna petición.

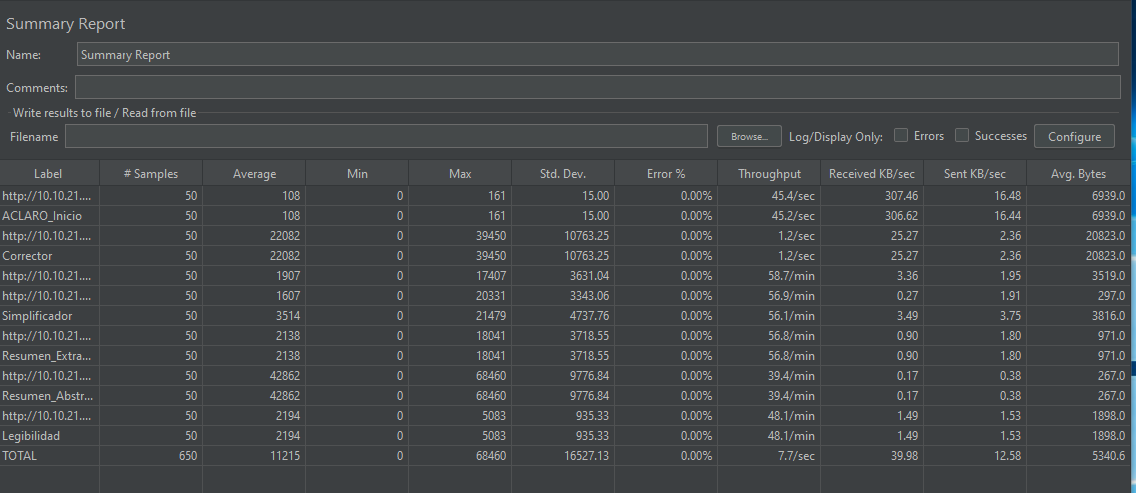
### **10.1.1. 50 usuarios**

Al ejecutar el plan de prueba de 50 usuarios se obtuvieron los resultados que se ven en la Tabla 1, en la que se puede apreciar una ausencia total de errores, el uso de la CPU alcanzó un 100%, un uso de memoria de casi el 50% y la consulta que más tiempo tardó en responder fue de 68460 milisegundos en la herramienta de “Resumen Abstractivo”, seguido por la herramienta de corrector con 39450 ms. Por el contrario, se pueden evidenciar que existen peticiones con respuesta de 0 ms. Esta prueba tuvo un tiempo de ejecución de 1 Minuto con 24 segundos.

|  |  |
| --- | --- |
| **% Error** | 0 |
| **Uso CPU** | 100% |
| **Memoria** | 15.8 GB |
| **# Muestras** | 50 |
| **Tiempo de ejecución** | 1:24 min |
| **Tiempo de respuesta mínimo** | 0 ms |
| **Tiempo de respuesta máximo** | 68460 ms |

*Tabla 1 - Resultado prueba 50 usuarios*

Ilustración 12 Resumen de resultados 50 usuarios.



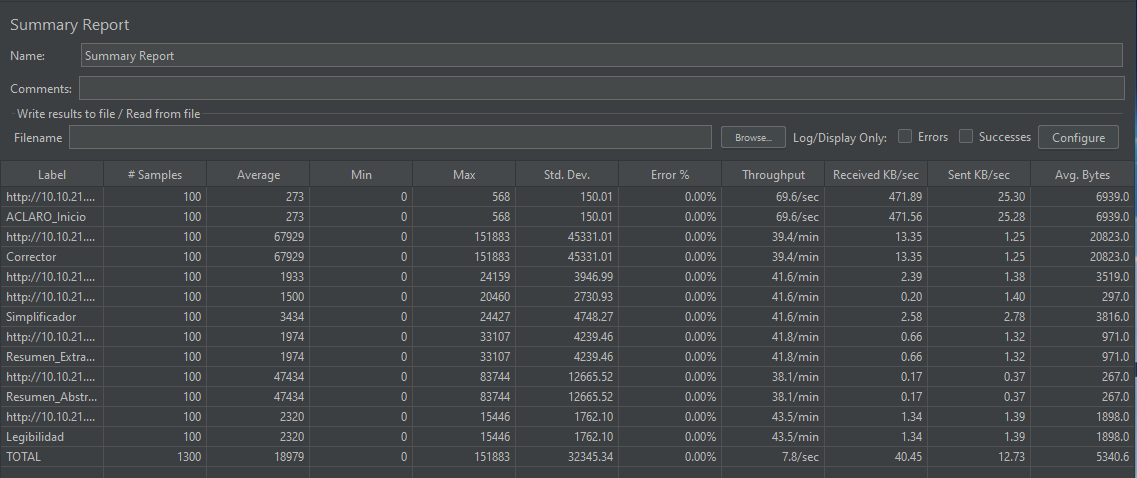
### **10.1.2. 100 usuarios**

Al ejecutar el plan de prueba de 100 usuarios se obtuvieron los resultados que se ven en la Tabla 2, en la que se puede apreciar una ausencia total de errores, el uso de la CPU alcanzó un 100%, un uso de memoria de casi el 50% y la consulta que más tiempo tardó en responder fue de 151.883 milisegundos en la herramienta de “Corrector”, seguido por la herramienta de “Resumen Abstractivo” con 83744 ms. Por el contrario, se pueden evidenciar que existen peticiones con respuesta de 0 ms. Esta prueba tuvo un tiempo de ejecución de 2 Minutos con 47 segundos.

|  |  |
| --- | --- |
| **% Error** | 0% |
| **Uso CPU** | 100% |
| **Memoria** | 15.9 GB |
| **# Muestras** | 100 |
| **Tiempo de ejecución** | 2:47 min |
| **Tiempo de respuesta mínimo** | 0 ms |
| **Tiempo de respuesta máximo** | 151883 ms |

*Tabla 2 - Resultado prueba 100 usuarios*

Ilustración 13 Resumen de resultados 100 usuarios.



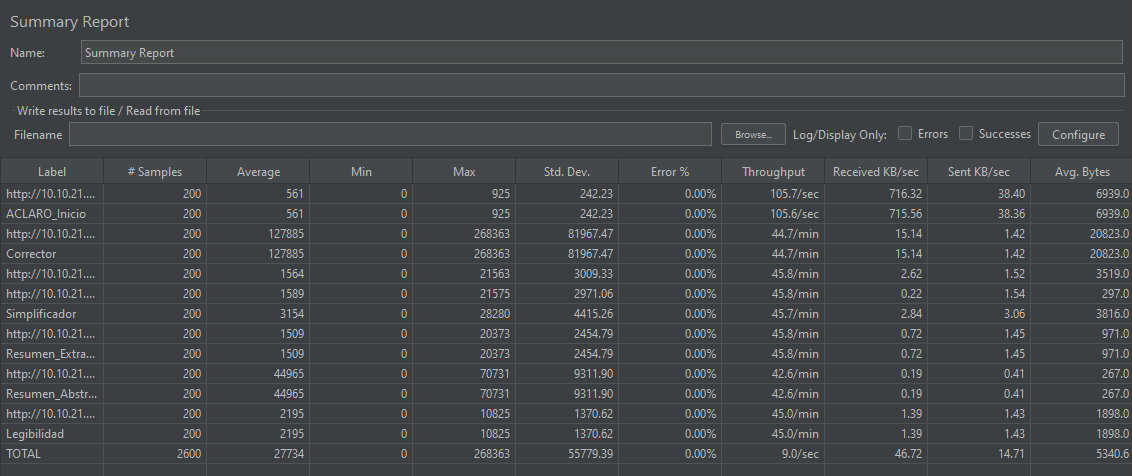
### **12.1.3. 200 usuarios**

Al ejecutar el plan de prueba de 200 usuarios se obtuvieron los resultados que se ven en la Tabla 4, en la que se puede apreciar una ausencia total de errores, el uso de la CPU alcanzó un 100%, un uso de memoria un poco superior al 50% y la consulta que más tiempo tardó en responder fue de 453340 milisegundos en la herramienta de “Corrector”, seguido por la herramienta de “Resumen Abstractivo” con 84108 ms. Por el contrario, se pueden evidenciar que existen peticiones con respuesta de 0 ms. Esta prueba tuvo un tiempo de ejecución de 7 Minutos con 57 segundos

|  |  |
| --- | --- |
| **% Error** | 0% |
| **Uso CPU** | 100% |
| **Memoria** | 16 GB |
| **# Muestras** | 200 |
| **Tiempo de ejecución** | 4:50 |
| **Tiempo de respuesta mínimo** | 0 ms |
| **Tiempo de respuesta máximo** | 268363 ms |

*Tabla 3 - Resultado prueba 200 usuarios*

Ilustración 14 Resumen de resultados 200 usuarios.



### **10.1.4. 300 usuarios**

Al ejecutar el plan de prueba de 300 usuarios se obtuvieron los resultados que se ven en la Tabla 4, en la que se puede apreciar una ausencia total de errores, el uso de la CPU alcanzó un 100%, un uso de memoria un poco superior al 50% y la consulta que más tiempo tardó en responder fue de 453340 milisegundos en la herramienta de “Corrector”, seguido por la herramienta de “Resumen Abstractivo” con 84108 ms. Por el contrario, se pueden evidenciar que existen peticiones con respuesta de 0 ms. Esta prueba tuvo un tiempo de ejecución de 7 Minutos con 57 segundos.

|  |  |
| --- | --- |
| **% Error** | 0 |
| **Uso CPU** | 100% |
| **Memoria** | 16.1 GB |
| **# Muestras** | 300 |
| **Tiempo de ejecución** | 7:57 min |
| **Tiempo de respuesta mínimo** | 0 ms |
| **Tiempo de respuesta máximo** | 456340 ms |

*Tabla 3 - Resultado prueba 300 usuarios*

Ilustración 15 Resumen de resultados 300 usuarios.

Pantalla de computadora

Descripción generada automáticamente

## **10.2. Resumen de Pruebas de Carga y Estrés**

Para tener más clara la información y resultados arrojados por JMeter y Azure Monitor se presenta la siguiente tabla, la cual contiene los datos de error, uso CPU, memoria, número de muestras, tiempos de ejecución tiempo mínimo y máximo de respuesta.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Característica\# Muestras** | **50** | **100** | **200** | **300** |
| **% Error** | 0 | 0 | 0 | 0 |
| **% Uso CPU** | 100 | 100 | 100 | 100 |
| **Memoria (GB)\*** | 15.8 | 15.9 | 16.0 | 16.1 |
| **Tiempo de respuesta mínimo(ms)** | 0 | 0 | 0 | 0 |
| **Tiempo de respuesta máximo(ms)** | 68460 | 151883 | 268363 | 456340 |

*Tabla 4 - Resultados pruebas.*

*Gráfico 1 – Comportamiento de las variables.*

# Bibliografía

*Amazon*. (2023). Obtenido de https://aws.amazon.com/es/what-is/web-application/

BEDOYA, J. W. (2020). *SISTEMA DE APOYO PARA LA ESCRITURA DE DOCUMENTOS EN LENGUAJE CLARO.*

Devlin, Chang, J. a., Lee, M.-W. a., Toutanova, K. a., & Kristina. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.

Erkan, Radev, G. a., & R, D. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization.

Garzón, J. A., Vanegas, J. E., Galeano, B. M., & Durán, J. C. (2015). *Guía de lenguaje claro para servidores públicos de Colombia.* Obtenido de https://colaboracion.dnp.gov.co/CDT/Programa%20Nacional%20del%20Servicio%20al%20Ciudadano/GUIA%20DEL%20LENGUAJE%20CLARO.pdf

*Kinsta*. (2020). Obtenido de https://kinsta.com/es/blog/arquitectura-aplicaciones-web/

*LanguageTool*. (2022). Obtenido de LanguageToolhttps://github.com/languagetool-org/languagetool

*LanguageTool*. (12 de 2022). Obtenido de GitHub: https://github.com/languagetool-org/languagetool

sharma, P. (2022). *CYCOTECK*. Obtenido de https://cynoteck.com/es/blog-post/reasons-to-treat-web-application-architecture-seriously/

Tewari, G. (7 de oct de 2022). *Forbes.* Obtenido de Recent Advancements In Artificial Intelligence: https://www.forbes.com/sites/forbesbusinesscouncil/2022/10/07/recent-advancements-in-artificial-intelligence/?sh=1e93e2ec7fa5

Xue, Constant, L. a., Roberts, N. a., Kale, A. a., Al-Rfou, M. a., Siddhant, R. a., . . . Aditya and Raffel, C. (2020). mT5: A massively multilingual pre-trained text-to-text transformer.