Universidad Politécnica

de Madrid

Escuela Técnica Superior de

Ingenieros Informáticos

Grado en ingeniería informática

Trabajo Fin de Grado

Lectura Fácil: Índices de Lecturabilidad 2.0

Autor: Blanca Rodríguez González

Tutor(a): María del Carmen Suárez de Figueroa Baonza

Cotutor: Isam Diab Lozano

Madrid, junio 2024

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Ingeniería Informática
Título: Lectura Fácil: Índices de Lecturabilidad 2.0
Febrero 2024

Autor: Blanca Rodríguez González

Tutores:

María del Carmen Suárez de Figueroa Baonza

Departamento de Inteligencia Artificial. Ontology Engineering Group (OEG)

ETSI Informáticos

Universidad Politécnica de Madrid

Isam Diab Lozano

Departamento de Inteligencia Artificial Ontology Engineering Group (OEG) ETSI Informáticos

Universidad Politécnica de Madrid

Agradecimientos

Me gustaría agradecer a todos mis familiares que me han apoyado durante estos cuatro años de mucho esfuerzo, ellos siempre han confiado en mí.

Por otro lado, me gustaría agradecer a todas las grandes amistades que he comenzado gracias a esta carrera, mis amigos y mi novio han estado también siempre ahí, entendiendo todas las complicaciones que hemos superado juntos.

Y por último me gustaría agradecer la labor de todos aquellos profesores que hacen del aprendizaje una motivación, aquellos que incitan a querer saber más sobre su materia y aquellos cuya verdadera pasión es enseñar y que sus alumnos aprendan, especialmente me gustaría agradecer el trabajo de Guillermo Román, Guillermo Vigueras, Francisco Rosales, María Luisa Córdoba y Alfonso Zamora. También agradecer a mis tutores del Trabajo de Fin de Grado, María del Carmen Suarez de Figueroa e Isam Diab, por estar siempre a mi disposición para sacar este trabajo a delante.

Es un orgullo poder decir que he obtenido el grado de Ingeniería Informática en la Universidad Politécnica de Madrid.

Resumen

Hoy en día, la creciente conciencia de las dificultades en la comprensión lectora impulsa el desarrollo de tecnologías que faciliten el acceso a la información para todos los sectores de la población.

Este problema es tan importante ya que el acceso igualitario a la información es un derecho reconocido. Cada vez se analizan más los pasos para llegar a formar textos legibles ya que se considera que es esencial que todo el mundo tenga acceso a la información. Hay distintas maneras de analizar la lecturabilidad de un texto y muchas variables a tener en cuenta.

Hace dos años un antiguo alumno de la Universidad Politécnica de Madrid realizó un trabajo de fin de grado sobre este mismo tema, desarrollando Indicum, una aplicación web la cual evalúa distintas pautas, normativas e índices que indican cuanto de legible era un texto introducido. En este TFG se analizaban índices como la cantidad de signos de puntuación, las pautas UNE o el impacto de las abreviaturas [1].

Como se ha mencionado anteriormente, cada vez se analizan más estas situaciones. Obteniendo nuevos índices o métricas a tener en cuenta para poder dar una valoración más ajustada al concepto de un texto comprensible.

Sin embargo, no todo es el análisis si no que, se ha considerado que se debe cuidar la estética de nuestras soluciones haciéndolas lo más intuitivas y usables posible. Partiendo de una base de que el usuario no tiene por qué tener un alto conocimiento tecnológico.

El presente trabajo busca la combinación de estos dos puntos mencionados, mejorando la interfaz de Indicium y añadiéndole nuevas métricas que hemos considerado importantes como por ejemplo la longitud de las frases, el uso de extranjerismos, la frecuencia escrita o la frecuencia oral. A lo largo de este trabajo se aborda todo el proceso de desarrollo, desde la exploración de

nuevas tecnologías hasta la implementación y las pruebas necesarias.

Esta propuesta surge como una contribución significativa en el avance tecnológico, considerando la relevancia del tema y su impacto en una amplia parte de la población. Se busca proporcionar a los usuarios herramientas para evaluar la legibilidad de sus textos, promoviendo una comunicación más efectiva y accesible para todos.

Abstract

Today, the growing awareness of reading comprehension difficulties is driving the development of technologies that facilitate access to information for all sectors of the population.

This issue is so important because equal access to information is a recognised right. Increasingly, the steps to achieve readable texts are being analysed as it is considered essential for everyone to have access to the information. There are different ways of analysing the readability of a text and many variables to consider.

Two years ago, a former student of the Universidad Politécnica de Madrid carried out a final thesis on the same subject named Indicium, this web evaluates different guidelines, regulations and indexes that indicated how readable a text was. This dissertation analysed indexes such as the number of punctuation marks, UNE guidelines or the impact of abbreviations [1].

As it was mentioned earlier, more and more of these situations are being analysed. New indexes or metrics are being considered to give a more accurate assessment of the concept of a comprehensible text.

However, it is not all about analysis, we have also considered that we must take care of the aesthetics of our solutions, making them as intuitive and usable as possible. Starting from the premise that the user does not need to have a high level of technological knowledge.

The present work seeks to combine these two points, improving the interface of Indicium and adding new metrics that we have considered important, such as the length of sentences, the use of foreign words, the written frequency or the

spoken frequency. The entire development process will be addressed, from the exploration of new technologies to implementation and testing.

This project is proposed as a significant contribution to technological progress, considering the relevance of the topic and its impact on a large part of the population. It seeks to provide users with tools to assess the readability of their texts, promoting more effective and accessible communication for all.

Tabla de contenidos

Índice de Ilustraciones.

1.Introducción

El acceso a la información a día de hoy está en continuo crecimiento gracias al internet. Una de las maneras principales para acceder a esta información es por medio de la lectura, lo que nos hace considerar que la capacidad de lectura y la comprensión de un texto es un derecho fundamental.

La comprensión lectora se define como el proceso que permite generar un significado con valor al interactuar con un texto, actualmente la lectura se considera un acto de razonamiento en sí mismo [2].
Sin embargo, hay diversos motivos por los cuales este derecho, no es alcanzable para todo el mundo.
Algunos de los motivos por los que esto sucede son:

- Trastornos de la lectura: hay diversos trastornos hoy en día que afectan a la lectura tal y como pueden ser la dislexia (el trastorno más conocido, afecta a la dificultad de comprensión, o rapidez de lectura), alexia (pérdida de capacidad de comprensión lectora tras un derrame cerebral o un accidente), hiperlexia (estas personas presentan una alta habilidad de lectura sin embargo presentan dificultades de comprensión.) [3]
- Contexto familiar o social: lo cual puede haber provocado una falta cultura de vocabulario [4].
- Falta de interés: la falta de interés en la lectura puede llevar a la falta de atención a la hora de leer, lo cual conlleva a una falta de compresión lectora. [5]
- Problemas de memoria: la falta de memoria a corto plazo afecta en gran medida a la compresión lectora ya que permite retener la información ya procesada por un corto periodo de tiempo, mientras se procesa la nueva información que se va leyendo. [5]

Por estos motivos, se ha establecido una necesidad de desarrollar una serie de pautas a seguir para hacer los textos más legibles. Entre ellas, hay varias pautas ya desarrolladas como pueden ser las de la UNE (acrónimo de **U**na **N**orma **E**spañola), con un conjunto de normas, normas experimentales e informes (estándares) creados en los Comités Técnicos de Normalización (CTN) de la

Asociación Española de Normalización (UNE), antes llamada AENOR. La UNE ha desarrollado unos índices sobre la proporción de signos de puntuación que facilitan la compresión, o la ausencia de frases en pasiva. También hay otros estudios que indican que las frases cortas son más fáciles de entender o aquellas que poseen palabras comunes.

Según estas pautas mencionadas y alguna más, se realizó una aplicación para poder analizar los textos y evaluarlos, indicando así posibles mejoras para facilitar la lectura lo máximo posible a los distintos grupos de personas, Esta aplicación, Indicium [1], fue desarrollada por un alumno de la universidad en 2022. Esta aplicación web evaluaba distintos puntos que mencionaremos con más profundidad en capítulos posteriores y mostraba los resultados al usuario.

Durante este período de dos años, se han investigado nuevos índices que podrían ser incorporados, y se ha llegado a la conclusión de que se requiere un desarrollo más avanzado en la interfaz de usuario de la aplicación, con el objetivo de hacerla más usable e intuitiva para el usuario final.

De esta manera, la finalidad principal de este trabajo de fin de grado es la mejora de un servicio web útil para solucionar un problema que afecta a gran parte de la población y que defiende un derecho innegociable, dentro de las mejoras a desarrollar nos encontramos con tres puntos principales.

En primer lugar, se destaca la mejora de la interfaz gráfica de la aplicación. En el contexto del trabajo anterior, se dedicó una considerable cantidad de tiempo al desarrollo de la aplicación en sí, lo que limitó la oportunidad de diseñar una interfaz visualmente atractiva y amigable para el usuario final.

En segundo lugar, está la ampliación de los índices que observa la aplicación al evaluar la calidad del texto, ya que, como se ha mencionado, ha habido distintos estudios que han descubierto más factores que afectan a que un texto sea más fácil de leer.

Por último, se plantea el despliegue de la aplicación en un entorno Docker, con el fin de garantizar su accesibilidad desde cualquier dispositivo mediante una

dirección IP pública. Este paso resulta crucial para asegurar la disponibilidad y utilidad de la herramienta en un amplio espectro de contextos y usuarios.

Los contenidos a tratar en los siguientes capítulos son:

- Capitulo 2: Estado de la Cuestión: Este capítulo se enfoca en revisar y analizar la literatura existente sobre el tema de la comprensión lectora y las tecnologías desarrolladas para mejorarla. Se incluye un estudio detallado de investigaciones previas, trabajos académicos relevantes, así como las herramientas y enfoques utilizados en el campo.
- Capitulo 3: Diseño de la solución: Se muestra el prototipo de interfaz gráfica a desarrollar, así como las principales mejoras de nuevos índices que se van a añadir en la aplicación web ya existente.
- Capitulo 4: Implementación: Se detalla la implementación técnica de la solución, abordando las herramientas y tecnologías utilizadas para desarrollar la aplicación web y las decisiones de programación tomadas.
- Capitulo 5: Desarrollo del despliegue en Docker. Se explica el procedimiento de despliegue de la aplicación.
- Capitulo 6: Pruebas: Se describen las pruebas realizadas para la comprobación del correcto funcionamiento de las nuevas incorporaciones a la aplicación web.
- Capitulo 7: Conclusiones y líneas futuras: Se mencionan las conclusiones obtenidas tras la finalización del TFG y los aprendizajes obtenidos, así como las posibles mejoras del mismo. Se analiza el impacto que puede tener este TFG en la sociedad y en que puede contribuir.
- Capitulo 8: Bibliografía.

2.Estado de la Cuestión

En esta sección se puede observar un resumen de Indicium, el trabajo de fin de grado de Antonio Ventosa, indagando en los índices y las pautas UNE utilizadas. También se muestran algunos trabajos similares previos a este.

2.1. Indicium

En este trabajo de fin de grado del año 2022, Antonio Ventosa llevó a cabo el desarrollo de una aplicación web que analiza la lecturabilidad de un texto. El motivo de este trabajo fue similar al de este trabajo de fin de grado, ya que lo propuso la misma tutora.

En este trabajo se realizó un estudio exhaustivo sobre qué factores afectaban al hecho de que un texto fuese más legible, también se estudió quiénes eran los principales usuarios a los que podría ir orientada la aplicación y aunque se llegó a la conclusión de que esta aplicación podría ser usada por cualquier persona, se decidió hacer un enfoque más profesional.

Una vez estudiados y analizados distintos trabajos anteriores como pudo ser Legible [6], una aplicación que realiza y presenta análisis de índices a partir de una entrada de texto, se observó que dicha herramienta estaba diseñada para usuarios expertos en la materia. Si bien este trabajo se enfoca en el ámbito profesional, se mantiene el objetivo de que sea accesible y comprensible para cualquier usuario.
Por eso se decidieron utilizar dos elementos para analizar los textos, índices de lecturabilidad y pautas UNE, los cuales se detallan con más precisión en las siguientes secciones.

2.1.1. Índices de lecturabilidad utilizados para el análisis de los textos.

Se utilizaron distintos índices de lecturabilidad los cuales analizan distintas cualidades del texto que hacen que un texto sea más fácil de leer. A

continuación, se puede ver los distintos tipos de índices que se analizan en la web de Indicium desarrollada por Antonio Ventosa.

- Índices ortográficos: se deben incluir puntos, limita las comas y evitar el punto y coma en los textos
- Índices silábicos: se deben usar palabras cortas y repetir palabras frente al empleo de sinónimos u otras formas.
- Índices léxicos: el vocabulario debe de ser cotidiano y próximo a la lengua, evitando palabras complejas o abstractas. Esto se comprobaba mediante una lista publicada por la RAE de palabras frecuentes en el castellano, esta lista contenía 5000 palabras comunes.
- Índices sintácticos: se deben usar oraciones breves y simples, evitando la subordinación
- Índices de lecturabilidad en español: propuestas particulares de investigadores como Fernández-Huerta, Inflesz, mu, Szigriszt, Gutiérrez Polini y Crawford.

2.1.2. Pautas UNE.

Los índices descritos en la Sección 2.1.1 se relacionaron con un subconjunto de las pautas UNE de lectura fácil. Dicha relación se estableció con la ayuda de D. Oscar García Muñoz [7], responsable del área de Accesibilidad de Plena Inclusión de Madrid[1]. La relación entre índices y pautas de lectura fácil permite establecer ciertos indicios, relacionados con la lecturabilidad, sobre el texto original. Si todos los índices se encontraban dentro de un rango establecido se cumplía enteramente la pauta relacionada, si está en el rango de incumplimiento, la pauta no se cumple. Y si no pertenece a ninguno de esos rangos, se cumple de manera parcial.

[1] https://plenainclusionmadrid.org/

2.1.3. Infraestructura de la aplicación web

Para la parte de diseño web se utilizó el framework Django, cuyas interfaces estas desarrollada en HTML, pero permite que sus páginas puedan ser redirigidas en base a una url introducida por el usuario.
Para la parte de procesamiento de lenguaje natural se utilizó spaCy[2], una librería para el procesamiento del lenguaje natural (PLN) implementada en Python, Sin embargo, esta herramienta no implementaba la utilidad de silabeador y para ello se utilizó una nueva herramienta llamada Pyverse, la cual se utilizaba para el recuento de silabas en las oraciones de texto, siguiendo las reglas de la poesía castellana.

2.2. Otros trabajos previos.

Otros trabajos previos que se han desarrollado similares a este han sido aplicaciones como Clara [8], la cual es una aplicación que analiza e indica la situación sobre un texto incluido, pero sin mostrar información de los datos analizados. Esta aplicación, va a ser utilizada para la mejora de la interfaz de Indicium, ya que es una referencia de cómo podríamos hacerlo

Por un lado, nos podemos encontrar también distintos analizadores de texto, pero enfocado más al análisis morfológico o sintáctico no tanto relacionado con la facilidad de comprensión. Algunos de estos ejemplos que podemos encontrar serian Linguakit [9] o Lexicool [10].

Linguakit realiza un análisis morfológico detallado de cada palabra y proporciona información sobre el análisis sintáctico de las frases introducidas. En contraste, Lexicool se enfoca en contar el número de frases, palabras y caracteres, además de ofrecer información estadística sobre la repetición de frases y palabras clave.

Otras aplicaciones relevantes en este campo incluyen Hemingway App [11], es una aplicación que permite mejorar la claridad de un texto escrito en inglés,

[2] https://spacy.io/

este texto puede ser académico como profesional, otorga una puntuación a la lecturabilidad o comprensibilidad del texto en función a unos parámetros determinados y propone alternativas que mejoren la lecturabilidad. Grammarly [12] es un asistente de escritura estadounidense basado en la nube. Revisa errores de ortografía, gramática, puntuación, claridad, y otros aspectos de textos en inglés. A su vez, detecta plagio y sugiere reemplazos para los errores identificados. Sin embargo, muchas de las funcionalidades de esta aplicación son de pago.

Además, el Test de Lecturabilidad de Fleisch-Kincaid [13] proporciona una medida tradicional de legibilidad, donde una puntuación más alta indica una lectura más simple, siendo una herramienta útil para evaluar la accesibilidad del texto según diferentes niveles educativos, esta puntuación se calcula en función de la longitud de las frases y la complejidad de las palabras que haya en ellas. La puntuación obtenida oscila entre el 0 y el 100, dicha puntuación nos puede enfocar hacia qué edad está orientada el texto introducido.

Finalmente, se destacan dos asistentes de Lectura Fácil: E2R-Helper [14], que permite obtener índices de lecturabilidad de textos en español y realizar verificaciones detalladas según la norma UNE 153101 EX de mayo de 2018; y Asistente-Lectura-Fácil-INICO [15], una herramienta de ayuda para la adaptación a lectura fácil. Estos trabajos previos proporcionan un marco de referencia valioso para el desarrollo y mejora continua de nuestra aplicación.

3.Diseño de la aplicación

En este capítulo se abordan los aspectos fundamentales relacionados con las modificaciones realizadas en la interfaz gráfica de la aplicación, destacando el diseño del prototipo final para la aplicación web de Indicium. Además, se discute en detalle el proceso de diseño de los nuevos índices introducidos, los cuales han sido desarrollados para mejorar la calidad y profundidad del análisis de legibilidad ofrecido por la herramienta.

3.1. Interfaz gráfica.

La elaboración de esta sección ha requerido una dedicación significativa debido a la importancia de mejorar la experiencia visual del usuario. En este sentido, se consideró esencial diferenciar claramente la pantalla de introducción de datos de la de visualización de resultados. Esta diferenciación implicó un desafío adicional en cuanto al manejo de datos de solución y la vinculación de los gráficos con los valores generados por nuestra calculadora de índices.

En todas las pantallas hay una parte común: la parte lateral izquierda, donde se muestra el logo de la aplicación y una breve descripción del uso de esta. En la parte superior, se indica el objetivo principal de esta página.

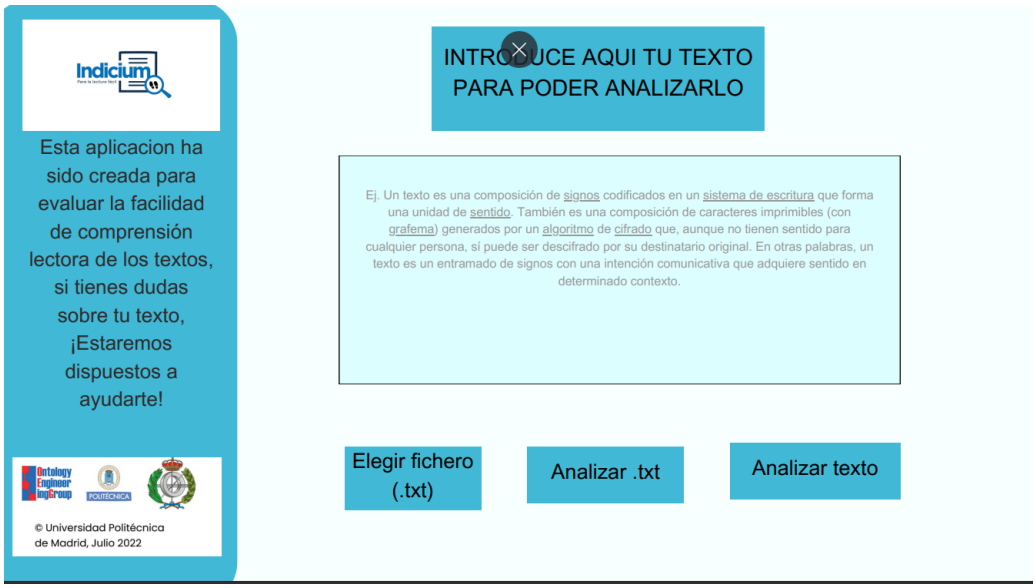

Figura 1. Prototipo de la página de inicio.

Como se puede observar en la Figura 1, se puede ver el prototipo de la página principal, en esta página se encuentra la caja donde introducir el texto deseado, y debajo del mismo, el botón para enviar el texto a analizar, el cual envía a la pantalla de resultados (Figura 2.) También hay disponible una opción de añadir un .TXT para analizar.

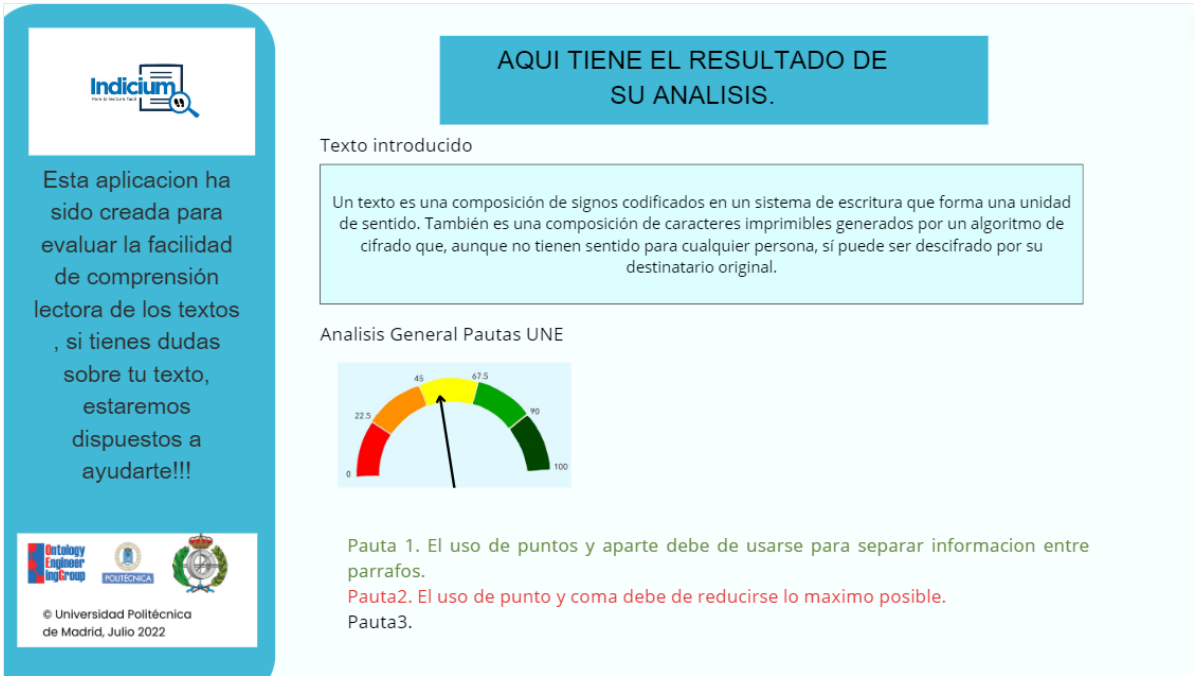

Figura 2. Indicios en base a las pautas de lectura fácil

La página que podemos ver en la Figura 2, es una de las posibles maneras de ver el análisis. Tiene cuatro elementos principales.

Por un lado, tenemos una caja de texto donde se puede observar el texto que se ha analizado, a continuación, se encuentra un velocímetro que marca el porcentaje de satisfacción general de cumplimiento de las pautas UNE, después se observa una leyenda para las frases que están a continuación ya que estas vienen coloreadas en función de su grado individual de satisfacción.

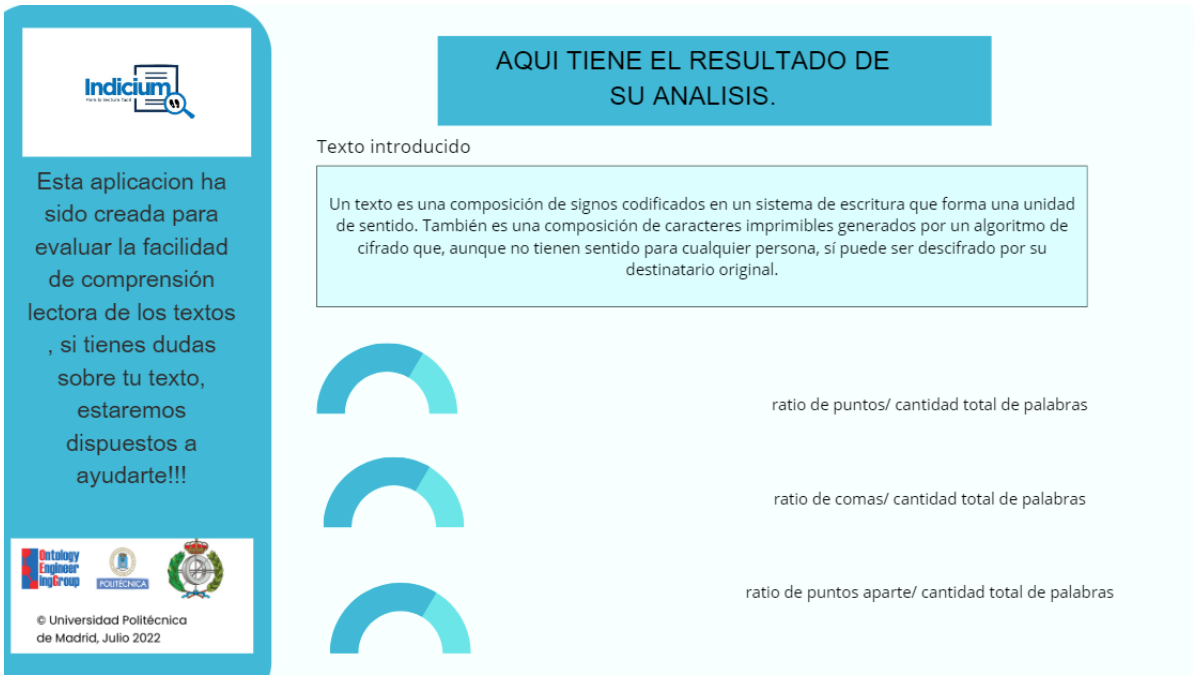

Figura 3. Índices de lecturabilidad.

La página que muestra la Figura 3 nos muestra los índices de lecturabilidad que se han obtenido de una manera más numérica, está dividida por secciones.

- Análisis léxico: muestra la frecuencia léxica o la diversidad de palabras.
- Análisis Ortográfico: se muestran las ratios de puntos, de comas, puntos y coma, dos puntos y punto y aparte.
- Análisis sintáctico: se muestra información como la media de palabras por frase o la complejidad oracional.
- Análisis silábico: índices relacionados con la extensión silábica o las palabras polisémicas
- Análisis de lecturabilidad: análisis relacionados con fórmulas matemáticas relacionadas con la lecturabilidad.

3.2. Nuevos índices de lecturabilidad.

Los nuevos índices que se han incorporado en la aplicación son los siguientes:

- Índice de variación de vocabulario: en este índice se va a mostrar una proporción de la cantidad de palabras distintas en el texto pertenecientes a las palabras poco frecuentes del mismo. De esta manera cuantas más

palabras poco frecuentes diferentes haya en el texto más complejo será la lectura de este.

- Índice de extranjerismos. Se ha establecido un índice para calcular el porcentaje de palabras en otros idiomas distinto al castellano nos encontremos en el texto. Cuantas más palabras tenga el texto en extranjero más complicada puede ser su lectura.
- Índice de complejidad estructural oracional. Aquí podemos ver una proporción de la cantidad de oraciones coordinadas y yuxtapuestas entre la cantidad de oraciones subordinadas. Este análisis se ha realizado ya que las oraciones subordinadas pueden complicar la lectura, de esta manera, cuanto más bajo sea este índice más fácil de leer será el texto introducido.
- Índice de complejidad silábica: este índice calcula la relación entre el número de silabas y el número de silabas de baja frecuencia.

4.Implementación

En este capítulo, se detalla el proceso de mejora de la aplicación Indicium, centrándonos en dos aspectos fundamentales que han sido objeto de desarrollo: la interfaz de usuario y los nuevos índices de análisis. A lo largo de esta sección, se explica en profundidad cada una de estas áreas, destacando los objetivos, métodos y resultados alcanzados en el marco de esta mejora significativa de la herramienta.

4.1. Implementación de Interfaz.

Como se ha mencionado anteriormente, la interfaz anterior de la aplicación web presentaba una simplicidad visual que dificultaba la comprensión de los resultados del análisis. Al estar desarrollada en HTML, la creación de una interfaz básica es relativamente sencilla, dependiendo del nivel de complejidad que se desee alcanzar mediante el uso de CSS para el estilo. Tras varios cursos de HTML y CSS se comprendió que la practica utilizada en la anterior interfaz podría ser mejorada ya que integraba el estilo CSS dentro de los propios comandos de HTML sin embargo, se recomienda separar la parte de modelaje en un documento aparte, el problema ante esta solución era que Django muestra complicaciones a la hora de linkear los documentos HTML con CSS y para ahorrar estas complicaciones se recurrió a una práctica que aunque no es la más recomendable era mejor que la usada hasta el momento.

Dicha práctica consiste en introducir al comienzo del fichero HTML en la parte del <head> un <style> que contenga todo el código de estilos referenciados a este documento. Esto se contemplará más adelante con detalle.

De esta manera se decidió dividir la aplicación en tres pantallas en lugar de tener todo en una única, pudiendo diferenciar la pantalla donde introduces los datos, una pantalla para indicios obtenidos según la norma UNE y otra pantalla para obtener resultados de los índices de lecturabilidad.

4.1.1. Implementación de la pantalla principal.

Dentro de esta sección, se presenta una visualización detallada del código HTML utilizado, junto con el código de estilos en CSS asociado. Esta división permite una comprensión más clara de la estructura y el diseño de la interfaz de usuario de la aplicación Indicium, proporcionando así una visión integral de los elementos que componen la experiencia visual y funcional para el usuario final.

4.1.1.1. Código en HTML

En esta primera página hay una barra lateral con una breve descripción del objetivo de esta aplicación. A su derecha se encuentra una caja de texto con un texto de explicación algo más extensa donde se explica que tipo de análisis se realiza en nuestra aplicación y la finalidad de cada uno de los botones que se encuentran debajo de esta caja. En esta caja es donde se introduce el texto que deseas analizar. Debajo de la caja se encuentra la posibilidad de añadir un fichero .TXT para analizar y los dos botones mencionados anteriormente, uno para ir al análisis de indicios de las pautas UNE y el otro para ir al análisis de índices de lecturabilidad.

Para esto se ha dividido el código en contenedores ya que esto facilita el trabajo a la hora de darle forma a la aplicación. Se debe poner un contenedor principal el cual albergara dos contenedores secundarios: el lateral y el central.

Dentro del contenedor lateral se encuentra el logo y el texto mencionado, el codigo de este contenedor se puede apreciar en la Figura 4.

```
<body>
    <div class="container">
        <div class="lateral">
            {% load static %}
            <img src="{% static " mainpage/media/INDICIUM4.png" %}" alt="Logo" width="300" height="200" align="center">
            <div class="lateralTextos">
                <p>Esta aplicación ha sido creada para evaluar la facilidad de comprensión lectora de los textos. Si
                    tienes dudas sobre tu texto, ¡estaremos dispuestos a ayudarte!</p>
            </div>
            {% load static %}
            <img src="{% static " mainpage/media/logo2.jpg" %}" alt="Logo2" width="300" height="200" align="center">
        </div>
```

Figura 4. Código del contenedor lateral común a todas las páginas.

Por otro lado, dentro del contenedor central, el cual podemos observar en la Figura 5, se encuentran los elementos mencionados, un título de la página, una

caja de texto (form) para introducir el texto deseado y dos botones de tipo action (post) que serán los que accionen el análisis y el paso a la siguiente página. La única diferencia destacable con el prototipo de diseño es que el logo de la universidad finalmente se decidió colocarlo en el footer de la página y que se diseñaron únicamente dos botones que enviaran el texto o el fichero en lugar de poner dos botones por cada manera de introducir el texto a analizar.

```
<div class="central">
    <p id="textoSuperior"><b>Lecturabilidad de textos en Castellano.</b></p>
    <!-- TXT Form -->
    <form action="" method="post" enctype="multipart/form-data" id="combinedForm" class="myFormClass">
        {% csrf_token %}
        <textarea name="inputTxt" id="inputTxt" rows="5" cols="50"  placeholder="Escribe aquí tu texto.

    <br>
    <br>
    <p id="textoFichero">También puedes añadir tu texto mediante un fichero (.txt)</p>
 <label for="fileInput"> Seleccione un fichero .TXT: </label>
<input type="file" name="file" id="fileInput">

        <div class="buttonHolder" >
            <input type="submit" class="submit" value="Calcular Indicios" id="BotonIndicios" name="BotonIndicios">
            <input type="submit" class="submit" value="Calcular Índices de Lecturabilidad" id="BotonIndices"
            style="margin-left: 225px;" name="BotonIndices">
        </div>
```

Figura 5. Código del contenedor central de la página principal.

El footer mencionado es común a todas las páginas y viene descrito en la clase base.html que es la clase padre de las otras tres, podemos observar su desarrollo en la Figura 6.

```
<footer id="footer">
    <!-- Copyright -->
    <div class="text-center p-3" style="background-color: rgba(0, 0, 0, 0.1);">
    {% load static %}
    <img src="{% static "/mainpage/media/logo3.png" %}" alt="Logo2" width="500" height="150"
    align="center">
            <b> © Universidad Politécnica de Madrid, Junio 2024.</b>
        <a class="text-dark" href="https://www.fi.upm.es/"><b>fi.upm.es</b></a>
    </div>
```

Figura 6. Código del footer de la aplicación web.

4.1.1.2. Código de css

A continuación, se presenta el código HTML con la estructura y clases utilizadas para la visualización y diseño de la interfaz de usuario. Cada clase se ha definido con el propósito específico de aplicar estilos y funcionalidades particulares a los elementos correspondientes, siguiendo las directrices del proyecto. Las clases incluyen comandos para añadir fondo, establecer la fuente de texto (Arial), aplicar colores institucionales del departamento de Ontología, dar formato a la caja de texto con texto de ejemplo que se desvanece al escribir, y diseñar los

botones para redirigir a las siguientes páginas. El código css a destacar se puede observar en las Figuras 7 y 8

```css
.container {
    display: flex;
}

.lateral {
    width: 300px;
    font-size: 20px;
    font-family: Arial;
    background-color: #0068CC
}

.lateralTextos {
    color: aliceblue;
    margin: 50px;
}
```

Figura 7. Código del estilo CSS del contenedor lateral de la página inicial de la aplicación web.

```css
.central {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 20px;
    margin: 50px;
    padding: 20px;
}
#inputTxt{
    border-radius:10px ; height: 400px; width: 100%;
    background-color: lightblue; margin-top: 20px;
}
#textoSuperior {
    color: #0068CC;
    font-size: 40px;

}
.buttonHolder {
    display:flex;
    border-radius: 15px;
    margin-right: 580px;
    margin-top: 20px;
}

.submit {
    color: aliceblue;
    background-color: #0068CC;

    padding: 10px 20px;
    border-radius: 15px;
}
```

Figura 8. Código del estilo CSS del contenedor central de la página inicial de la aplicación web.

Finalmente podemos observar en las siguientes figuras (Figura 9 y Figura 10) el resultado final obtenido con los cambios mencionados en esta sección.

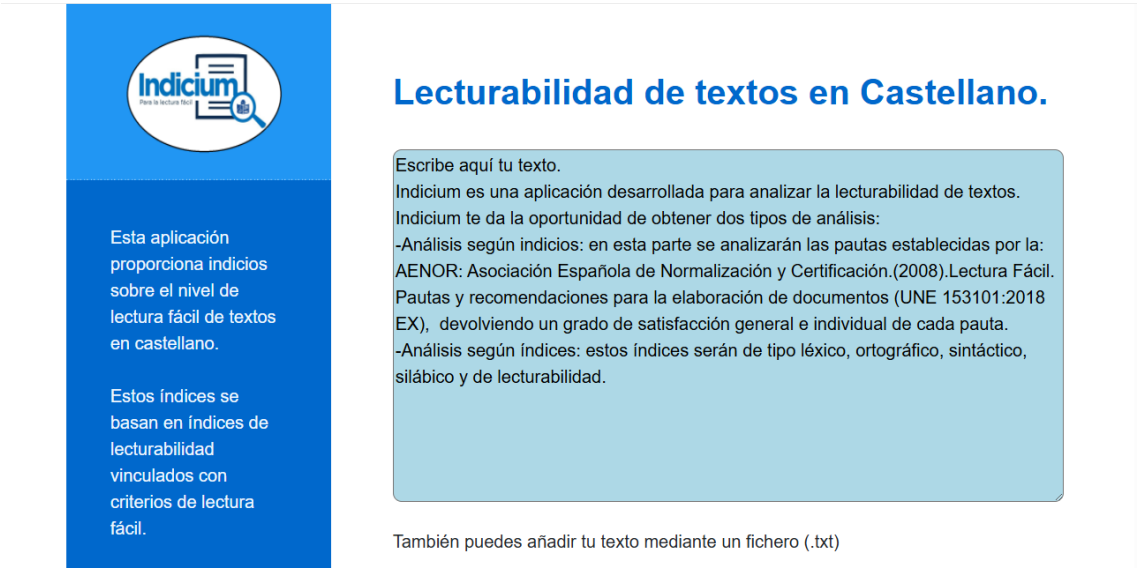

Figura 9: Pantalla de inicio final.

Figura 10: Footer final

4.1.2. Implementación de la pantalla de indicios

La pantalla de indicios está diseñada para mostrar de manera clara y estructurada los resultados del análisis de legibilidad. Está compuesta por dos contenedores principales, siguiendo la misma estructura utilizada en el resto de las pantallas de la aplicación.

El primer contenedor, el contenedor lateral, es igual que las clases anteriores.

El segundo contenedor, el contenedor central, es donde se ubican todos los elementos específicos de la pantalla de indicios. En este contenedor se encuentran:

- Caja de lectura: Aquí se muestra el texto que ha sido introducido para su análisis, proporcionando al usuario una visualización clara de la entrada de datos.
- Velocímetro: Este elemento, como se muestra en el diseño, ofrece una representación visual del nivel de legibilidad del texto analizado.
- Pautas de la UNE: Se presentan de forma listada y coloreada, facilitando la comprensión de las pautas establecidas por la norma UNE para la evaluación de la legibilidad de textos. El color de las frases viene detallada a su vez en una leyenda dentro de un desplegable.

Esta disposición de elementos en el contenedor central garantiza una visualización organizada y efectiva de los resultados del análisis de legibilidad en la pantalla de resultados de la aplicación 'Indicium'.

4.1.2.1. Desarrollo en HTML

La implementación del velocímetro en la pantalla de resultados se llevó a cabo mediante el desarrollo de una aguja dinámica. Esta aguja se mueve en función del porcentaje obtenido durante el análisis, recorriendo más o menos parte del velocímetro según el nivel de legibilidad alcanzado. Inicialmente, se intentó crear el velocímetro utilizando HTML y CSS, pero debido a la complejidad para obtener el resultado deseado, se optó por integrar una figura del velocímetro y colocar una aguja dinámica sobre ella, lo cual resultó más efectivo y sencillo de implementar. Podemos apreciar el desarrollo del velocímetro en la Figura 11.

```
<div class="centro">
    <label for="textoArea"> Este es tu texto</label>
    <textarea id="textoArea" class="textoArea" rows="4" cols="50" readonly>{{ texto }}</textarea>
    <div class="UNE">
        <p id="uneTitulo"> Cumplimiento de la Norma UNE </p>
        <div class="velocimetro">
            {% load static %}
            <img src="{% static "/mainpage/media/velocimetro.jpg" %}" alt="Velocímetro" width="300" height="150">
            <div class="aguja" id="aguja"></div>
        </div>
        <p style="margin-left: 18%;">{{ progreso }}</p>
        <script>
            function actualizarAguja(valor) {
            const angulo = (valor - 50 ) * 180 / 100; // Calcula el ángulo de rotación (ajustado)
            document.querySelector('.velocimetro .aguja').style.transform = `rotate(${angulo}deg)`;
            }

            // Ejemplo de uso
            actualizarAguja({{progreso}}); // Rota la aguja al 50%
        </script>
```

Figura 11. Código de la caja de texto de lectura y del velocímetro.

Justo debajo del velocímetro, se incluye una leyenda que describe los colores utilizados para señalar las frases según las pautas UNE y sus respectivos significados. Para aplicar estos colores, se utilizó el código proporcionado por mi compañero, el cual recogía los análisis realizados y los representaba mediante diferentes colores, facilitando así la interpretación de los resultados.

Además, se incorporó un botón de 'Analizar otro texto' al final de la pantalla de resultados. Esta función permite al usuario regresar a la pantalla de inicio y realizar un nuevo análisis con un texto diferente, brindando una experiencia de uso más dinámica y práctica para el usuario. A continuación, podemos observar en la Figura 12 todo lo mencionado en este párrafo.

```
<p class="explicacion">
 <span>A continuacion, las pautas estaran coloreadas con colores en funcion del grado de
 satisfaccion de cumplimiento de las normas UNE.<br></span>
 <span class="rojo">No se cumple<br></span>
 <span class="rojo2">Se cumple un 1-22%<br></span>
 <span class="naranja">Se cumple parcialmente un 22%-44% <br></span>
 <span class="amarillo">Se cumple un 45%-67%<br></span>
 <span class="verdeC">Se cumple a un 67%-90%<br></span>
 <span class="verdeO">Se cumple al 90-100%</span>
 </p>

     <p>Pautas de Ortotipografía</p>
     <ul>
        <li style="color:{{pauta3_ortotipografia}}">Pauta 3. Se debe utilizar el punto y aparte para separar
                                                    frases u oraciones que presenten ideas diferentes. </li>
        <li style="color:{{pauta4_ortotipografia}}">Pauta 4. Las ideas enlazadas se deberían separar mediante
                                                    un punto en lugar de una coma</li>
        <li style="color:{{pauta5_ortotipografia}}">Pauta 5. Debería sustituirse el punto y seguido por un punto y
                                                    aparte o por una conjunción. </li>
        <li style="color:{{pauta7_ortotipografia}}"> Pauta 7. No se debe utilizar el punto y coma (;). </li>
     </ul>
```

Figura 12. Código de la leyenda de colores y un ejemplo de un grupo de pautas.

4.1.2.2. Desarrollo en css

En cuanto al desarrollo de CSS para esta pantalla, el código respectivo se muestra en la Figura 13, el mayor esfuerzo se centró en implementar el movimiento de la aguja del velocímetro. Esto implicó ajustar el porcentaje obtenido al semicírculo del velocímetro y lograr que la aguja se desplazara de manera coherente según el valor del porcentaje. Para lograr este efecto, fue necesario trabajar con la propiedad de transformación rotate en CSS, calculando los grados necesarios para que la aguja se moviera de manera proporcional al porcentaje alcanzado. Este proceso demandó un trabajo

minucioso de ajuste y pruebas para asegurar un movimiento fluido y preciso de la aguja en respuesta al porcentaje obtenido durante el análisis de legibilidad.

```css
.velocimetro{
    position: relative;
    width: 300px;
}

.aguja {
  position: absolute;
  bottom: 5%;
  left: 50%;
  transform: translate(-50%, -50%) rotate(-90deg);
  transform-origin: bottom center;
  width: 4px; /* Ancho de la aguja */
  height: 110px; /* Longitud de la aguja */
  background-color: red; /* Color de la aguja */
}
.explicacion{
  border: 1px solid black; /* Borde para la caja */
  padding: 10px; /* Espaciado interno */
  width: 700px;
    font-family: Arial;
  font-size: MEDIUM;

}
```

Figura 13. Código del estilo del velocímetro y de la caja de la leyenda.

A continuación, podemos observar en las Figuras 14, 15 y 16 el resultado final de la pantalla de indicios.

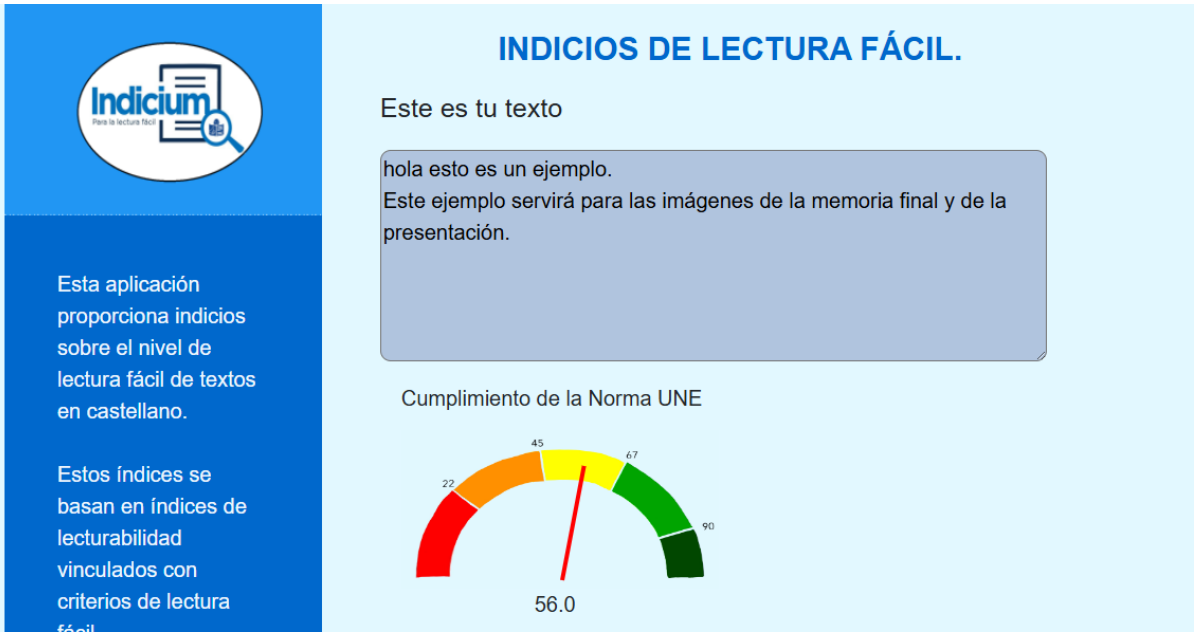

Figura 14: Resultado final pantalla indicios 1

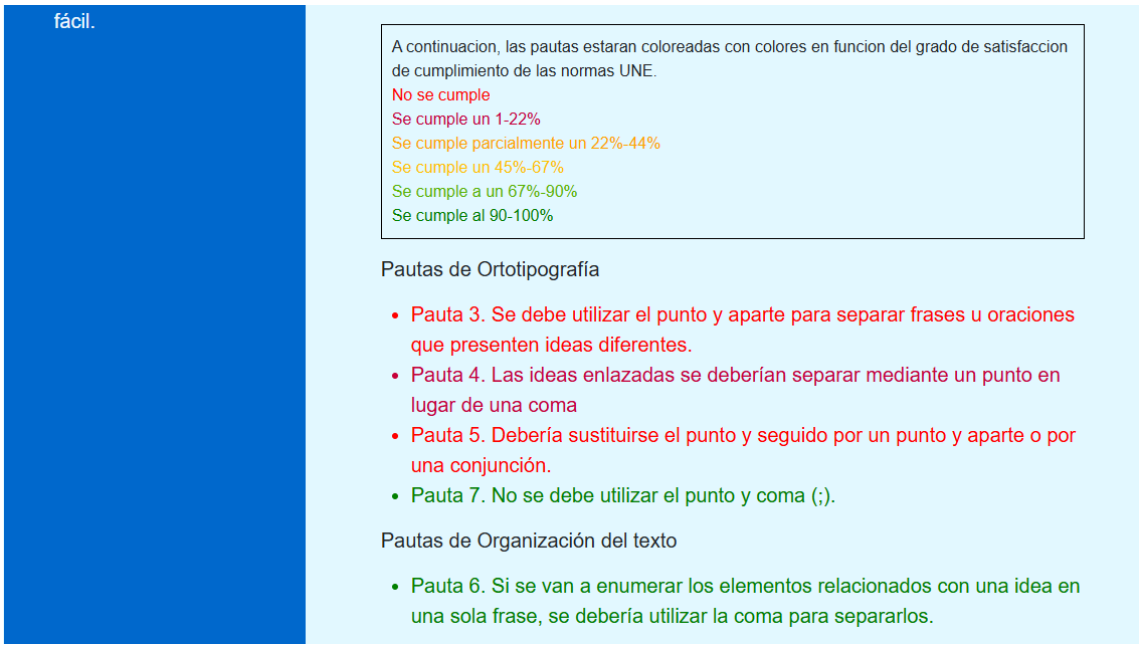

Figura 15: Resultado final pantalla indicios 2

Figura 16: Resultado final pantalla indicios 3

4.1.3. Desarrollo de la pantalla de índices

La segunda pantalla de resultados, denominada 'Índices de Lecturabilidad', ha sido cuidadosamente diseñada y organizada en cinco apartados distintos, cada uno representando un tipo específico de análisis. Antes de presentar los

resultados de cada análisis, se muestra nuevamente el texto introducido, siguiendo la misma estructura de la pantalla anterior para mantener la coherencia visual y facilitar la comparación de resultados.

En la parte inferior de este texto introductorio se han añadido una serie de botones, uno para cada tipo de análisis. Estos botones permiten al usuario navegar directamente al comienzo de cada sección correspondiente al tipo de análisis seleccionado. Además, cada tipo de análisis incluye un botón titulado 'Explicación Índices', el cual al ser clicado despliega un breve resumen sobre el análisis presentado y su interpretación.

Para mejorar la experiencia de usuario y la navegación dentro de la página, se ha diseñado un botón adicional en la esquina inferior izquierda de la pantalla denominado 'Subir al comienzo de la página'. Este botón facilita al usuario volver rápidamente al inicio de la página en cualquier momento, especialmente útil dada la cantidad de índices presentados en la pantalla.

En la Figura 17 podemos ver el desarrollo de la lista de navegación. Y en la Figura 18 el botón de subir al comienzo de la página.

```
<div class="central">

    <p id="textoSuperior"><b>ÍNDICES DE LECTURABILIDAD.</b></p>
    <div class="centro">
        <label for="textoArea" style="font-size: 25px; font-family: Arial, Helvetica, sans-serif;">
        Este es tu texto:</label>
    <textarea class="textoArea" id="textoArea" rows="4" cols="50" readonly>{{ texto }}</textarea>

    <p style="font-size: 20px; font-family: Arial, Helvetica, sans-serif;"> <br>Seleccione el Analisis
    <ul>
      <li><a href="#seccion1" class="lista">Análisis Ortografico</a></li>
      <li><a href="#seccion2" class="lista">Análisis Léxico </a></li>
      <li><a href="#seccion3" class="lista" >Análisis Silábico</a></li>
      <li><a href="#seccion4" class="lista">Análisis Sintáctico</a></li>
      <li><a href="#seccion5" class="lista">Análisis de Lecturabilidad</a></li>
    </ul>
```

Figura 17. Código de la caja de texto de lectura y el índice de tipos de análisis.

```
<a href="#" class="boton-arriba">Subir al comienzo de la página</a>
```

Figura 18. Código del botón “subir al comienzo de la página”

A parte de la distribución de contenedores mencionada en la pantalla de indicios, esta pantalla tiene una característica. Cada tipo de análisis tiene un contenedor asociado, ya que así es más fácil dar formato a cada sección por separado.

4.1.3.1. Desarrollo en HTML

Analizaremos cada análisis según su tipo.

4.1.3.1.1. Análisis Ortográfico

En esta sección se presentan los datos calculados relacionados con las ratios de puntos, comas, puntos aparte y puntos y coma en relación con la cantidad total de palabras. Este análisis se conecta con los índices mencionados anteriormente, proporcionando una representación más numérica y directa de los resultados que informan las pautas previas.

Cada ratio se visualiza en un contenedor individual que contiene un conjunto de semicírculos solapados, formando un diagrama de medialuna. A la derecha de estos diagramas, se incluye una breve descripción de la ratio que se está mostrando. La razón para separar cada análisis en contenedores individuales radica en la necesidad de organizar los elementos relacionados de manera coherente. De este modo, el texto explicativo se encuentra alineado a la derecha del gráfico correspondiente, y ambos elementos se disponen verticalmente uno debajo del otro, facilitando así una comprensión clara y estructurada de la información presentada.

En la siguiente Figura 19 se muestra un ejemplo de una de las cajas de explicación de índices y un ejemplo de un diagrama de medialuna.

```
<div id="seccion2">
<p id="tituloAnalisis"> <b>Indices Lexicos</b> </p>

<div id="mostrarTexto" onclick="mostrarTexto2()"> Explicación Índices </div>
<div id="textoOculto2" style="display: none;">
  <p class="explicacion"> Los siguientes índices nos ayudan a obtener resultados
    sobre el nivel léxico del texto. En este caso, cuantas más palabras haya similares
    (por consecuencia, más bajo sea el índice de diversidad de palabras), más alto sea
    el porcentaje de palabras comunes y menor sea el pocentaje de extranjerismos, más fácil
    será de leer el texto. Sobre el índice de variación cuanto más alto sea menos palabras
    distintas poco comunes habrá en el texto, lo cual también facilitará su lectura. </p>
</div>

<div class="porcentajesOrtograficos" style="--porcentaje: {{ind_diversidad_palabras}}; --color: fores
  <p><b>Índice de diversidad de palabras</b></p>
    <div class="aux">
    <svg width="150" height="150">
      <circle  r="65" cx="50%" cy="50%" pathlength="100" />
      <circle  r="65" cx="50%" cy="50%" pathlength="100" />
    </svg>
    <span >{{ind_diversidad_palabras}} </span>
    <p style="margin-left: 200px; margin-top: 20px;"> Ratio de palabras distintas en el texto</p>
    </div>
  </div>
```

Figura 19. Código del botón "Explicación índices" y un ejemplo de uno de los diagramas de medialuna de muestreo de resultados.

4.1.3.1.2. Análisis Léxico

Esta sección muestra los resultados relacionados con la diversidad de palabras y la frecuencia de repetición de estas. La representación gráfica y estructural de esta sección sigue el mismo formato que el análisis ortográfico.

4.1.3.1.3. Análisis silábico

En esta sección, la distribución sigue el mismo formato que en los apartados anteriores. Los índices presentados están relacionados con la extensión silábica, así como con la frecuencia de palabras trisílabas y polisémicas.

La presencia de palabras trisílabas y polisémicas puede complicar la comprensión lectora. Por esta razón, estos índices son importantes para evaluar la legibilidad del texto.

En la siguiente Figura 20, podemos observar la sección de código del análisis silábico.

```
<div id="seccion3">
<p id="tituloAnalisis"> <b>Indices Silábicos</b> </p>

<div id="mostrarTexto" onclick="mostrarTexto3()"> Explicación Índices </div>
<div id="textoOculto3" style="display: none;">
  <p class="explicacion"> Los siguientes índices nos ayudan a analizar la sección silábica.
    Cuanto más alta sea la extensión silábica y menor sea la cantidad de palabras trisílabas
    o polisémicas, tendremos un texto más legible. </p>
</div>
<p><b>Índice de extension silabica: {{ind_extension_silabica}} </b></p>
<p><b>Índice de baja frecuencia silabica: {{ind_baja_frecuencia}} </b></p>

iv class="porcentajesOrtograficos" style="--porcentaje: {{ind_palabras_tris}}; --color: forestgr
  <p><b>Índice de palabras trisilabas</b></p>
  <div class="aux">
  <svg width="150" height="150">
    <circle  r="65" cx="50%" cy="50%" pathlength="100" />
    <circle  r="65" cx="50%" cy="50%" pathlength="100" />
  </svg>
  <span >{{ind_palabras_tris}} </span>
  <p style="margin-left: 200px; margin-top: 20px;"> Ratio de palabras trisílabas en el texto</p>
  </div>
</div>
```

Figura 20. Código de la sección de Índices Silábicos

4.1.3.1.4. Análisis Sintáctico

La presentación de datos en esta sección es menos visual, ya que no se utilizan gráficos. Sin embargo, la comprensión de los índices se facilita mediante la explicación detallada de cada uno.

Primero, se presenta la media de palabras por frase. Este índice es crucial, ya que cuanto más largas sean las frases, más difícil será su comprensión. Adicionalmente, se realizan tres análisis de complejidad oracional. Estos análisis evalúan la importancia del uso de frases coordinadas y yuxtapuestas, y subrayan la necesidad de reducir al máximo las frases subordinadas, ya que estas últimas dificultan la comprensión. La estructura de las oraciones influye significativamente en la legibilidad, y estos índices proporcionan una visión clara sobre cómo las diferentes construcciones oracionales afectan la facilidad de lectura.

4.1.3.2. Desarrollo en css

El código CSS más destacable de esta página se centra en varios elementos clave que contribuyen a su funcionalidad y estética:

- Formateo de los Gráficos de Semicírculos: El CSS se utiliza para dar forma y estilo a los gráficos de semicírculos que representan diversos índices de análisis. Podemos observar cómo se llevó a cabo en las Figuras 21 y 22.

```css
@keyframes rellenar2{
to{
    stroke-dasharray: calc((var(--porcentaje) /2 )) 100; /* Calcula la longitud de la línea punteada */
}
}
```

Figura 21. Código de la función que rellena los diagramas de medialuna.

```css
.porcentajesOrtograficos span{
position: absolute;
top: 0%;
left: 0%;
bottom: 0%;
right:79%;
display: flex;
align-items: center;
justify-content: center;
font: 25px/1em Verdana;
font-size: 20px;
}

.porcentajesOrtograficos circle{
fill: none;
stroke-width: 20;
transform: rotate(-180deg);
transform-origin: 50%;
stroke-dasharray: 50 50;
stroke: #AAA;
}
.porcentajesOrtograficos circle:nth-child(2){ /* el segundo c
stroke: var(--color);
stroke-dasharray: 0 100;
animation: rellenar2 2s linear forwards;
}
```

Figura 22. Código de los círculos que forman los diagramas de medialuna.

- Botones Generados por una Lista Enlazada: Los botones en la página se crean a partir de una lista enlazada, a cuyos elementos se les ha dado forma de botón mediante CSS.

  Se puede observar cómo se desarrolló en la Figura 23.

```css
.lista{
  display: inline-block;
    padding: 8px 16px;
    background-color: #0068CC;
    color: white;
    text-decoration: none;
    border-radius: 5px;
    transition: background-color 0.3s ease;
}
.lista:hover {
    background-color: #0056b3;
}
```

Figura 23. Código del estilo del índice de tipos de análisis.

- Formateo de los Botones de Explicación: Cada tipo de análisis posee un botón 'Explicación Índices' que, al ser clicado, despliega un resumen sobre el análisis. El CSS asegura que estos botones sean visualmente distintivos y funcionales, utilizando propiedades de estilo para tamaño, color y posición.

  La figura 24 muestra el código css relacionado con el botón mostrar texto.

```css
    .explicacion{
      border: 1px solid black; /* Borde para la caja */
      padding: 10px; /* Espaciado interno */
      width: 700px;
        font-family: Arial;
      font-size: MEDIUM;

    }
    #mostrarTexto {
  cursor: pointer; /* Cambia el cursor al pasar sobre el área clickeable */
  background-color: lightblue;
  padding: 10px;
  width: fit-content;
  font-family: Arial;
    font-size: 16px;
}
```

Figura 24. Código del estilo del botón de mostrar explicación.

- Botón 'Subir al Comienzo de la Página': Este botón, ubicado en la esquina inferior izquierda de la pantalla, permite al usuario regresar rápidamente al inicio de la página. Su diseño incluye estilos CSS para tamaño, color,

posición fija y efectos interactivos para mejorar la navegación y usabilidad. La figura 25 muestra el desarrollo de este botón.

```css
.boton-arriba {
  display: block;
  position: fixed;
  bottom: 20px; /* Ajusta la posición vertical del botón */
  right: 100px; /* Ajusta la posición horizontal del botón */
  padding: 10px 20px;
  background-color: #0068CC; /* Color de fondo del botón */
  color: #fff; /* Color del texto del botón */
  border: none;
  border-radius: 5px;
  cursor: pointer;
  font-size: 16px;
}
```

Figura 25. Código del estilo del botón de subir al comienzo de la pantalla.

El uso de CSS en estos elementos es crucial para crear una interfaz de usuario intuitiva y estéticamente agradable, mejorando significativamente la experiencia de navegación y comprensión de los resultados de análisis.

Podemos observar en las siguientes Figuras 26, 27, 28, 29 el resultado final de la pantalla de índices de lecturabilidad.

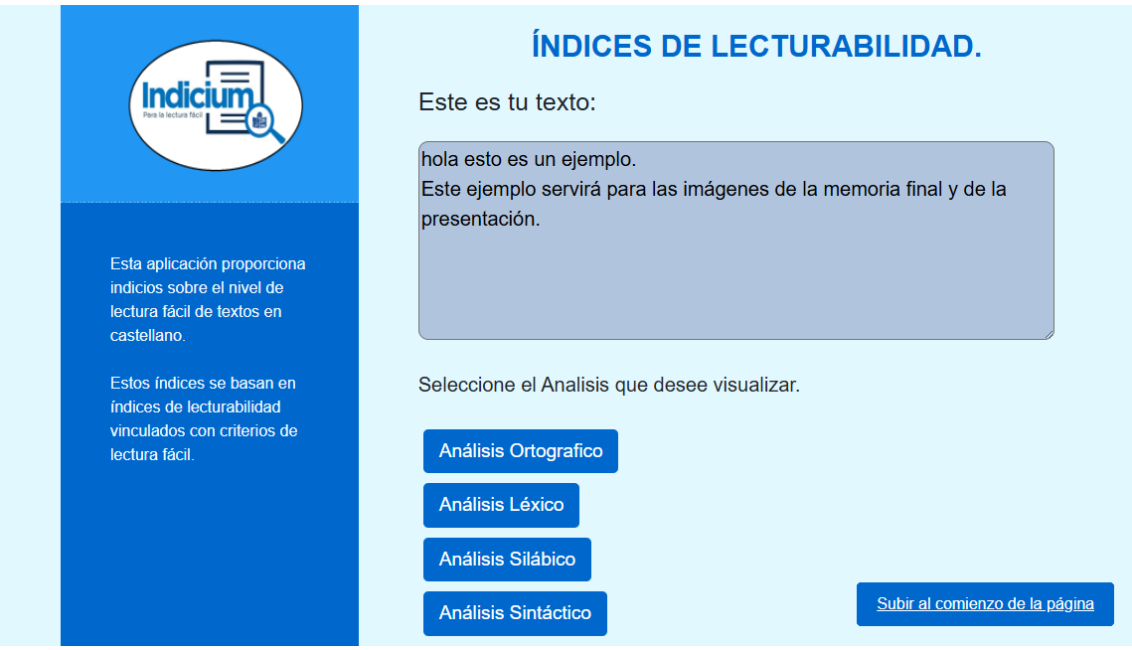

Figura 26: resultado final pantalla índices de lecturabilidad 1.

Figura 27: resultado final pantalla índices de lecturabilidad 2.

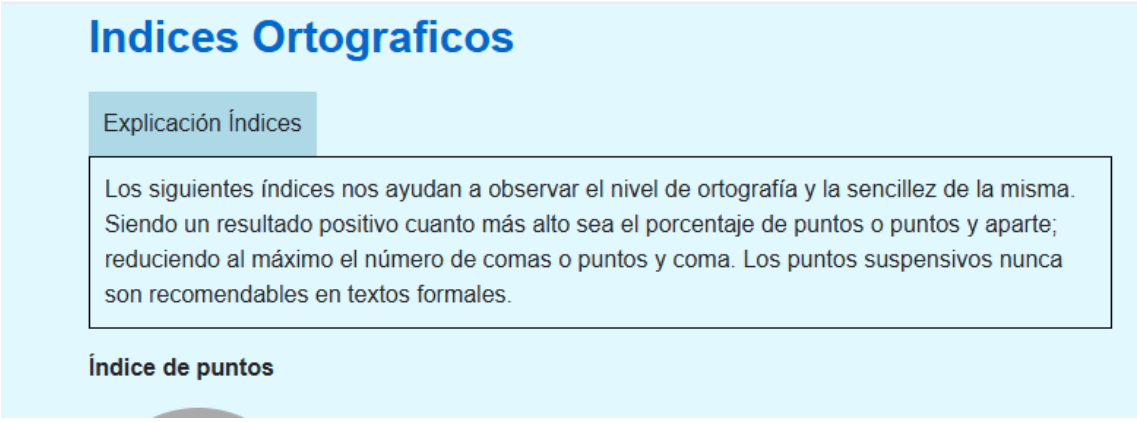

Figura 28: Caja de Explicación de índices desplegada.

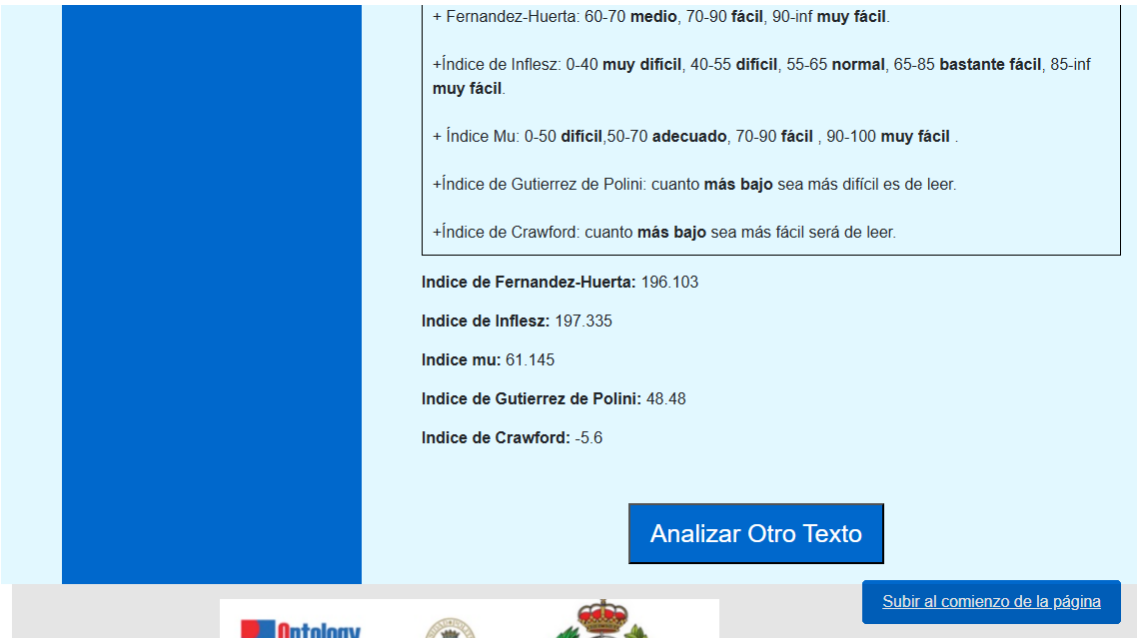

Figura 29: resultado final pantalla índices de lecturabilidad 3.

4.2. Comunicación entre pantallas

Tras decidir elaborar la aplicación web utilizando Django, fue necesario investigar cómo se enlazan las pantallas en esta herramienta. Django gestiona la navegación entre pantallas a través de dos componentes fundamentales: urls.py y views.py.

En la Figura 30 podemos ver el desarrollo de urls.py: Este archivo define las URL o rutas de la aplicación y las asocia con las funciones o vistas correspondientes en views.py. Funciona como un mapa de direcciones, indicando a Django qué vista debe mostrar cuando un usuario accede a una URL específica. Por ejemplo, una URL puede estar vinculada a la pantalla de introducción de datos o a la pantalla de resultados de análisis.

```python
from django.urls import path
from django.contrib.staticfiles.urls import staticfiles_ur

from . import views

urlpatterns = [
    path('', views.mainpage, name='mainpage'),
    path('', views.resultados, name='resultados'),
    path('', views.indices, name='indices'),
]

urlpatterns += staticfiles_urlpatterns()
```

Figura 30. Código de la clase urls.py.

En la Figura 31,32 y 33 podemos ver el desarrollo de views.py: Este archivo contiene las funciones de vista que procesan las solicitudes del usuario. En nuestra aplicación, views.py se encarga de gestionar las peticiones HTTP y devolver un resultado que implica un cambio de pantalla. Las funciones de vista pueden manejar tanto peticiones POST, como al introducir un texto para analizar, como peticiones GET, cuando se navega a la pantalla de resultados o se desea realizar un nuevo análisis.

```python
if request.method == 'POST':
    if not request.FILES: #No se ha introducido un fichero (caja de texto)
        form = TextInputForm(request.POST)
        fileinput = UploadFileForm()
        if form.is_valid():
            texto=form.data['inputTxt']
            index_dic = calculator.index_calculator(form.data['inputTxt'])

    else: #Se ha introducido un fichero
        form = TextInputForm()
        fileinput = UploadFileForm(request.POST, request.FILES)
        if fileinput.is_valid():
            aux = ""
            for chunk in request.FILES['file'].chunks():
                aux += str(chunk.decode('UTF-8'))
            texto=aux
            aux += ' \0'
            index_dic = calculator.index_calculator(aux)
```

Figura 31. Código de la gestión de peticiones HTTP.

```python
if (form.is_valid() or fileinput.is_valid()) and not index_dic is None:

    #Analisis ortografico
    ind_puntos = int( index_dic.get('indice_ortografico')['ind_puntos'] * 100)
    ind_puntosaparte = int(index_dic.get('indice_ortografico')['ind_puntosaparte'] *100)
    ind_comas = int(index_dic.get('indice_ortografico')['ind_comas'] *100)
    ind_puntoycoma = int(index_dic.get('indice_ortografico')['ind_puntoycoma'] *100)
    ind_puntossuspensivos = int(index_dic.get('indice_ortografico')['ind_puntossuspensivos'] *100)
```

Figura 32. Código de la obtención de los resultados.

```python
if 'BotonIndicios' in request.POST:
    return render(request,'resultados.html', { 'ind_puntos': ind_puntos, 'ind_puntosaparte':
                                'ind_extension_silabica' : ind_extension_silabica, 'ind_pala
                                'ind_diversidad_palabras' : ind_diversidad_palabras, 'ind_fr
                                'ind_palabras_frase' : ind_palabras_frase, 'ind_global_compl
                                'ind_complej_estruc_proposicional' : ind_complej_estruc_prop
                                'progreso' : progreso, 'pauta3_ortotipografia' : pauta3_orto
                                'pauta6_organizacion' : pauta6_organizacion, 'pauta1_frases'
                                'pauta1_vocabulario' : pauta1_vocabulario, 'pauta6_vocabular
if 'BotonIndices' in request.POST:
    return render(request,'indices.html', { 'ind_puntos': ind_puntos, 'ind_puntosaparte': ind
                                'ind_extension_silabica' : ind_extension_silabica, 'ind_pala
                                'ind_diversidad_palabras' : ind_diversidad_palabras, 'ind_fr
                                'ind_palabras_frase' : ind_palabras_frase, 'ind_global_compl
                                'fernandez_huerta_index' : fernandez_huerta_index, 'inflesz_
                                'progreso' : progreso, 'pauta3_ortotipografia' : pauta3_orto
                                'pauta6_organizacion' : pauta6_organizacion, 'pauta1_frases'
                                'pauta1_vocabulario' : pauta1_vocabulario, 'pauta6_vocabular
                                'ind_variacion_vocab': ind_variacion_vocab,})
```

Figura 33. Código del envío de los resultados a las pantallas.

Además de procesar estas solicitudes,se integra con el back-end de la aplicación web para enviar las respuestas adecuadas al usuario.

El proceso de comunicación entre pantallas en nuestra aplicación web se puede resumir de la siguiente manera: cuando un usuario interactúa con la aplicación, se envía una solicitud HTTP a una URL específica definida en urls.py. Esta solicitud es manejada por una función en views.py, que procesa la solicitud y devuelve una respuesta apropiada, ya sea presentando una nueva pantalla o actualizando la actual. Este flujo asegura una navegación fluida y coherente entre las diferentes secciones de la aplicación.

4.3. Desarrollo de los nuevos índices de lecturabilidad

El proyecto no consiste únicamente en realizar mejoras de interfaz si no también en implementar nuevos índices, la incorporación de estos índices es debido a que se ha considerado necesario introducir nuevas pautas para hacer el análisis más exhaustivo.

A continuación, se observa cada índice que se ha desarrollado, con su explicación y con su desarrollo explicado.

Debido a la extensión de este apartado de código, su desarrollo de Python se podrá encontrar en el Anexo.

4.3.1. Índice de extranjerismos

Este índice calcula el porcentaje de palabras en extranjero distintas presentes en el texto, ya que esto dificultaría la lectura. El desarrollo de este índice ha sido complicado, ya que, aunque varias librerías analizan el lenguaje en el que se escriben las palabras, no son muy exactas y muchas veces daban un resultado erróneo. Tras probar distintas librerías y algunas de las más confiables no permitían instalar la última versión, recurrimos a un recurso ofrecido por la RAE con una lista de unas 650 mil palabras aproximadamente en castellano.

El desarrollo de este índice implicó los siguientes pasos:

1. Recopilación del Fichero de Palabras: Se descargó y almacenó el fichero de la RAE (Real Academia Española) que contiene todas las palabras en español, este fichero se obtiene ejecutando un repositorio público de la RAE[3].
2. Desarrollo de la Función de Análisis: Se desarrolló una función que lee el fichero de palabras y las almacena en una estructura de datos eficiente para su posterior consulta.

[3] https://github.com/JorgeDuenasLerin/diccionario-espanol-txt

3. Listas de Palabras en Español y Extranjeras: Se creó una lista para almacenar las palabras en español obtenidas del fichero y otra lista para registrar las palabras extranjeras identificadas en el texto.
4. Comprobación de Palabras en el Texto: La función recorre el texto introducido por el usuario y verifica cada palabra contra la lista de palabras en español. Si una palabra no se encuentra en esta lista, se verifica si ya está en la lista de palabras extranjeras. Si no está, se añade a la lista de palabras extranjeras.
5. Cálculo del Porcentaje: Finalmente, se calcula el porcentaje de palabras extranjeras distintas, dividiendo el tamaño de la lista de palabras extranjeras entre el número total de palabras en el texto.

En la Figura 34. Podemos observar el código de la implementación de este índice.

```
def extranjerismos_index(document):
    wordcount = 0 #Contador de palabras
    foreingCount=0
    lex_count = 0
    encontrado=1
    array_palabras =[]

    ruta_archivo2 = os.path.join(os.path.dirname(__file__),"palabras_todas.txt")


    with open(ruta_archivo2, 'r', encoding='utf-8') as file2:
        for linea in file2:
        # Divide la línea en palabras utilizando el espacio como separador
            palabras_linea = linea.split()
        # Agrega cada palabra al array de palabras
            array_palabras.extend(palabras_linea)



    for token in  document:
        if not token.is_punct and not token.is_space: #Si es una palabra en general
            wordcount += 1
            resultado = any(str(token).lower() == str(palabra).lower() for palabra in array_palabras)
            if resultado == False:
                foreingCount+=1
```

```
 result = {
     'ind_cantidad_extranjerismos' : round(foreingCount / wordcount*100), #Indice de rat
 }

 return result

 return result
```

Figura 34. Código de la función que calcula los extranjerismos.

4.3.2. Índice de variación de vocabulario

Este índice consiste en calcular la cantidad de palabras distintas dentro de las palabras poco frecuentes presentes en el texto.

Para el desarrollo de este índice, se ha utilizado una lista de frecuencias de la Real Academia Española (RAE), ya usada en el TFG de Antonio, en esta lista salen las palabras más frecuentes usadas en el castellano ordenadas en función de su frecuencia. En este contexto, el índice calcula la cantidad de palabras de baja frecuencia, es decir, aquellas que no están presentes en la lista de palabras frecuentes, y determina cuántas de estas palabras distintas aparecen en el texto introducido.

Cuanto mayor sea el índice, peor será la lecturabilidad del texto ya que habrá más palabras poco frecuentes y distintas en el texto, observar el código en la Figura 35.

```
for token in document:
    if not token.is_punct and not token.is_space: #Si es una palabra en general
        wordcount += 1
        if not wordlist.__contains__(token.text.lower()): #Solo incluye palabras unicas
            wordlist.append(token.text.lower())

    frec = df[df['Orden'] == token.text.lower()]  # Busca en el dataframe la aparicion de la

    if (not token.is_punct and not token.is_space) and not token.is_stop:  # Si es una palabr
        lex_count += 1

    if (not token.is_punct and not token.is_space) and len(frec) == 0:  # Si es una palabra q
        lowfreccount += 1
        if not wordlistFrec.__contains__(token.text.lower()): #Solo incluye palabras unicas
            wordlistFrec.append(token.text.lower())
```

Figura 35: Código del cálculo de palabras poco comunes.

4.3.3. Complejidad silábica de palabras

Para el desarrollo de este índice, se han contado la cantidad de sílabas que siguen un patrón de sílaba poco frecuente. Los patrones de sílabas poco frecuentes contemplados en el castellano son las siguientes (considerándose V vocal y C consonante):

- CCV
- CCVC

- VCC
- CVCC
- CCVCC

El índice, por tanto, es la división de la cantidad de silabas con patrón poco frecuente entre el total de silabas de todo el texto.

Para saber si una sílaba sigue el patrón indicado se ha desarrollado un array con cada tipo de patrón de silaba, por cada sílaba del texto se comprueba si cumple o no el patrón establecido, el cual podemos ver en la Figura 36 y el array en la Figura 37.

```
SYLLABLE_PATTERNS = [r'(?![AEIOUaeiou])[A-Za-z](?![AEIOUaeiou])[A-Za-
z][aeiouAEIOUáéíóúÁÉÍÓÚüÜ](?![AEIOUaeiou])[A-Za-z](?![AEIOUaeiou])[A-Za-
z]',r'(?![AEIOUaeiou])[A-Za-z](?![AEIOUaeiou])[A-Za-
z][aeiouAEIOUáéíóúÁÉÍÓÚüÜ]',r'[aeiouAEIOUáéíóúÁÉÍÓÚüÜ](?![AEIOUaeiou])[A-
Za-z](?![AEIOUaeiou])[A-Za-z]',r'(?![AEIOUaeiou])[A-Za-
z](?![AEIOUaeiou])[A-Za-z][aeiouAEIOUáéíóúÁÉÍÓÚüÜ](?![AEIOUaeiou])[A-Za-
z]',r'[aeiouAEIOUáéíóúÁÉÍÓÚüÜ](?![AEIOUaeiou])[A-Za-z](?![AEIOUaeiou])[A-
Za-z]',r'(?![AEIOUaeiou])[A-Za-
z][aeiouAEIOUáéíóúÁÉÍÓÚüÜ](?![AEIOUaeiou])[A-Za-z](?![AEIOUaeiou])[A-Za-
z]']
```

Figura 36. Array con los patrones de las silabas poco comunes.

```
for syl in aux:
    for pattern in SYLLABLE_PATTERNS:
        if (re.search(pattern,syl)):
            sil_frec +=1
            break
```

Figura 37. Bucle que identifica las silabas que pertenecen a ese array.

4.3.4. Índice de complejidad estructural oracional

En este índice se analiza la complejidad oracional teniendo en cuenta las oraciones subordinadas, coordinadas y yuxtapuestas.

Las oraciones subordinadas son aquellas que tienen una oración principal y unida a esta por un nexo tiene una oración la cual no tiene significado por si

sola. Por otro lado, la oración coordinada es aquella cuyo nexo de conexión es una conjunción y finalmente una oración yuxtapuesta son dos proposiciones unidas mediante una coma, un punto y coma o dos puntos.

Teniendo estas definiciones de oraciones se implementó un código con la librería spaCy que facilita la identificación de oraciones coordinadas en caso de encontrar una conjunción en la misma, una oración yuxtapuesta en caso de encontrar uno de los signos de puntuación mencionados anteriormente y por último esta librería tiene la capacidad de identificar verbos principales o auxiliares, pudiendo identificar las oraciones subordinadas.

Una vez obtenida esta información, se realiza la división de la suma de las oraciones coordinadas y yuxtapuestas entre las oraciones subordinadas, este resultado se interpreta de manera que cuanto más alto sea el número más legible será el texto ya que las oraciones subordinadas pueden dificultar la lectura. Toda la función queda desarrollada en la Figura 38.

```
for token in sentence:
    if token.tag_ == 'VERB' and token.dep_ != 'ROOT' and token.dep_ != 'AUX':
        propositioncount += 1
        prop += 1
    if token.tag_ == 'CCONJ':
        cconj += 1
        coordcount2+=1
        propconj+=1
    if token.tag_ == 'SCONJ':
        sconj += 1
        propsub+=1
    if token.tag_ == 'VERB' and (token.dep_ == 'csubj' or token.dep_ == 'ccomp' or
        issub = True
        subordinada+=1
    if (str(token) == ',' or str(token) == ':' or str(token) == ';') :
        yuxtCount += 1
        yuxt= True
        propyuxt+=1


if (yuxt==True and issub==True):

    yuxtCount-=1
    propyuxt+=1

    propyuxt-=1
```

Figura 38. Código que identifica los distintos tipos de oraciones.

5.Desarrollo de despliegue del servicio web con Dockers.

El despliegue de la aplicación en un servicio Docker[4] da la facilidad de tener la aplicación desplegada de manera permanente y si se le asigna una IP publica cualquier persona puede acceder a ella.

Por tanto, tras la investigación sobre cómo se realiza dicho despliegue se llegó a la conclusión de que lo necesario era:

- Tener instalada la aplicación de Docker Desktop[5].
- Generar el fichero Docker-compose.yml, un archivo de configuración utilizado por Docker Compose para definir y ejecutar aplicaciones multicontenedor. Con Docker Compose, puedes definir todos los servicios, redes y volúmenes necesarios para tu aplicación en un solo archivo YAML
- Generar el fichero Dockerfile. es un archivo de texto que contiene una serie de instrucciones para construir una imagen de Docker. Estas instrucciones definen los pasos necesarios para configurar el entorno dentro de la imagen, instalar dependencias, copiar archivos y ejecutar comandos.
- Ejecutar en la terminal los comandos correspondientes para la formación de la imagen.

A continuación, se puede observar los ficheros mencionados en las FIguras 39 y 40.

[4] https://www.docker.com/
[5] https://www.docker.com/products/docker-desktop/

```dockerfile
FROM python:3.10-buster

ENV PYTHONUNBUFFERED 1
ENV PYTHONONDONTWRITEBYTECODE =1


RUN apt-get update && apt-get install -y ca-certificates


RUN python -m pip install --upgrade pip

RUN mkdir /lectura_facil

WORKDIR /lectura_facil


COPY requirements.txt /lectura_facil/

COPY es_core_news_sm-3.7.0-py3-none-any.whl /lectura_facil/

RUN pip install /lectura_facil/es_core_news_sm-3.7.0-py3-none-any.whl


RUN python -m pip install --trusted-host objects.githubusercontent.com -r /lectura_facil/requirements.txt

COPY . /lectura_facil/

EXPOSE 8000

CMD ["pyhton", "manage.py", "runserver", "0.0.0.0:8000"]
```

Figura 39. Código de la clase Dockerfile.

```yaml
services:
  app:
    image: blancarg/tfg_lectura_facil-app:latest
    build: .
    container_name: django
    command: python manage.py runserver 0.0.0.0:8000
    volumes:
      - .:/lectura_facil
    ports:
      - '8000:8000'
```

Figura 40. Código de la clase Docker-compose.yml

6.Pruebas

Para evaluar la interfaz de la aplicación web, se han planificado pruebas desde dos perspectivas distintas, utilizando herramientas específicas para cada una. En primer lugar, se ha empleado Selenium[6] para evaluar la funcionalidad de la interfaz, mientras que Lighthouse[7] se utilizará para analizar la accesibilidad web de cada una de las páginas de la aplicación.

Además de estas pruebas de accesibilidad y funcionalidad, se han realizado pruebas de compatibilidad en diversos navegadores, incluyendo Google Chrome y Microsoft Edge.

Estos tres tipos de pruebas se consideran fundamentales para garantizar la calidad y la usabilidad de la interfaz de la aplicación web desarrollada.

Adicionalmente, se han desarrollado test unitarios en Python relacionados con los nuevos índices implementados, asegurando así su correcto funcionamiento y su integración en la aplicación.

6.1. Pruebas con Selenium

Las pruebas realizadas por Selenium se basan en comprobar el funcionamiento de la navegabilidad de la aplicación web, estas pruebas se desarrollan en dos tramos si el desarrollo de las mismas es mediante la API de Selenium IDE.

Selenium IDE es una herramienta que permite grabar acciones mientras se interactúa con una página web. Posteriormente, es posible agregar comprobaciones sobre estas interacciones para verificar que los clics enlaces a las páginas correctas o realizan las actividades previstas. Este enfoque de pruebas es útil para validar la funcionalidad básica de la interfaz y asegurar una experiencia de usuario fluida y consistente [16].

---

[6] https://www.selenium.dev/documentation/ide/

[7] https://chrome.google.com/webstore/detail/lighthouse/blipmdconlkpinefehnmjammfjpmpbjk/related?hl=es

El proceso de pruebas comienza con la creación de un video que registra las acciones que se desean revisar. Por ejemplo, para verificar si los botones conducen a las páginas correctas, se comprueba el título de la página resultante para determinar si es "Indicios" o "Índices de Lecturabilidad". Del mismo modo, para verificar si los botones de "Extender Explicaciones" funcionan correctamente, se comprueba el texto mostrado después de hacer clic en el botón de explicación.

Una vez completadas estas acciones, se verifica el retorno a la página principal comprobando si la caja de texto es editable, ya que en las páginas de resultados esta caja es solo de lectura. Después de redactar todas las pruebas, se ejecutan simulando las acciones descritas. El resultado de cada prueba se muestra como "OK" para aquellas que se satisfacen correctamente.

Luego, se pueden exportar las pruebas para que cualquier persona pueda ejecutarlas y verificar el correcto funcionamiento de la aplicación. Esto permite una fácil replicación y validación del comportamiento esperado de la aplicación. Las pruebas se pueden redactar utilizando distintos comandos de Selenium IDE, como "assert text", que verifica si el texto identificado coincide con un valor establecido (Figura 41).

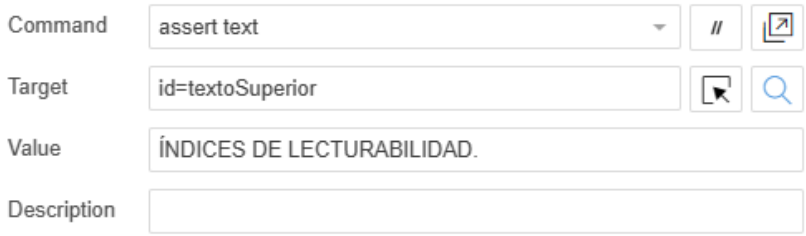

Figura 41. Ejemplo de un test de Selenium.

Los resultados obtenidos tras las pruebas realizadas sobre la aplicación de Indicium se pueden observar en la Figura 42.

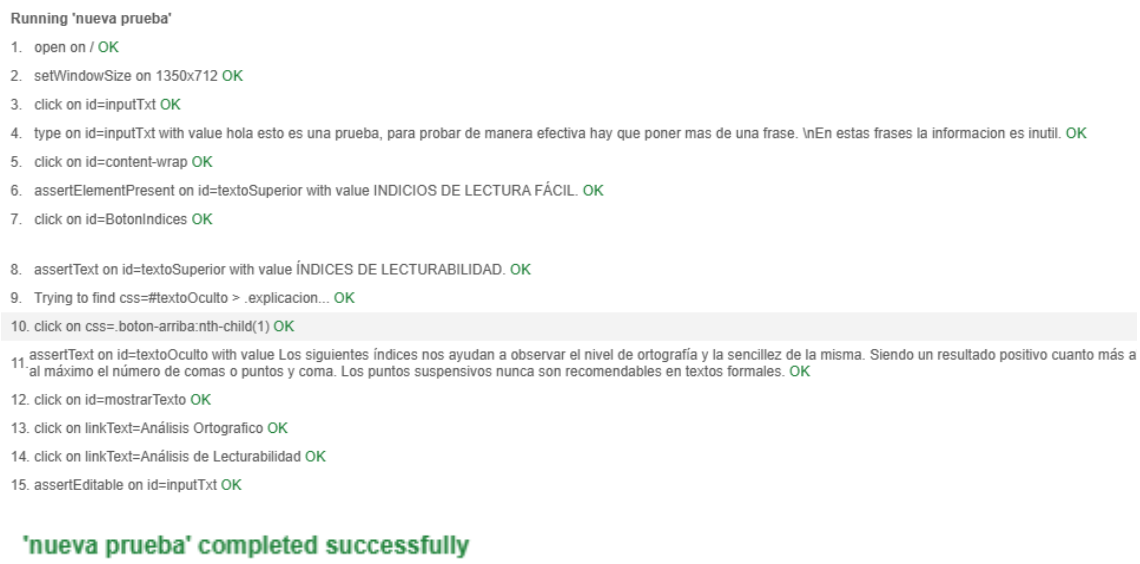

Figura 42. Resultado de los test de Selenium.

6.2. Pruebas con Lighthouse

Lighthouse es una herramienta de código abierto desarrollada por Google que se utiliza para evaluar la accesibilidad y el rendimiento de las páginas web [17].

La herramienta Lighthouse realiza pruebas y análisis en diferentes áreas clave, como la velocidad de carga de la página, la accesibilidad para personas con discapacidades, la optimización para dispositivos móviles, la seguridad de la página y más. Luego, genera informes detallados que muestran los resultados de estas pruebas junto con recomendaciones sobre cómo mejorar el rendimiento y la calidad de la página.[8]

De esta manera una vez terminado el desarrollo de la interfaz se ejecutó la extensión de Google para cada una de las páginas de la aplicación web. Los resultados obtenidos tras un par de cambios fueron muy positivos, obteniendo un 100% de accesibilidad en dos de las pantallas y un 94% en la pantalla de resultados. El resultado de esta pantalla es debido a la coloración de las pautas en función del porcentaje de satisfacción que según las reglas de accesibilidad establecidas por W3C.

A continuación, los resultados de Lighthouse en las Figuras 43,44 y 45:

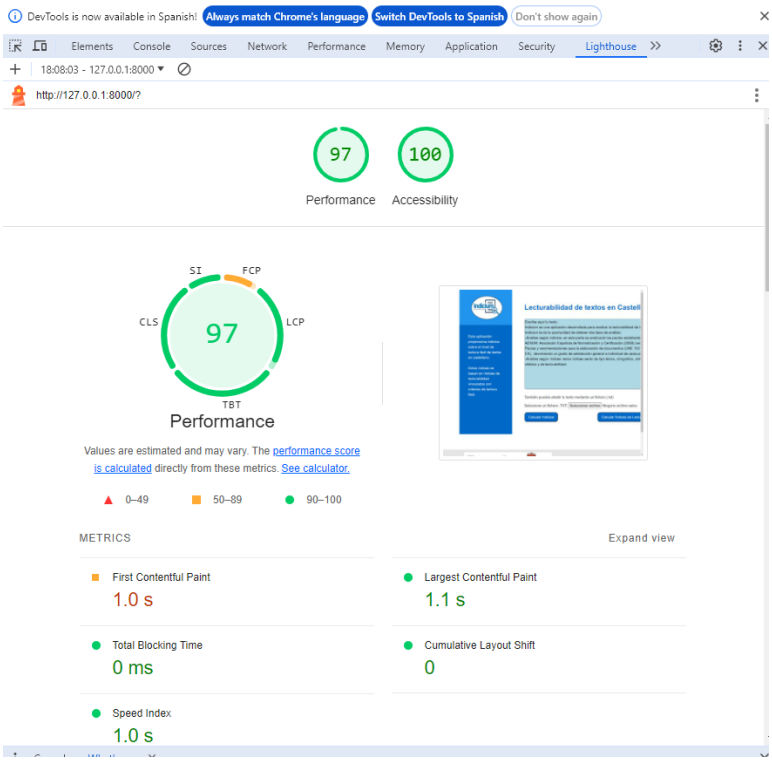

Figura 43. Resultados obtenidos de la accesibilidad de la página principal.

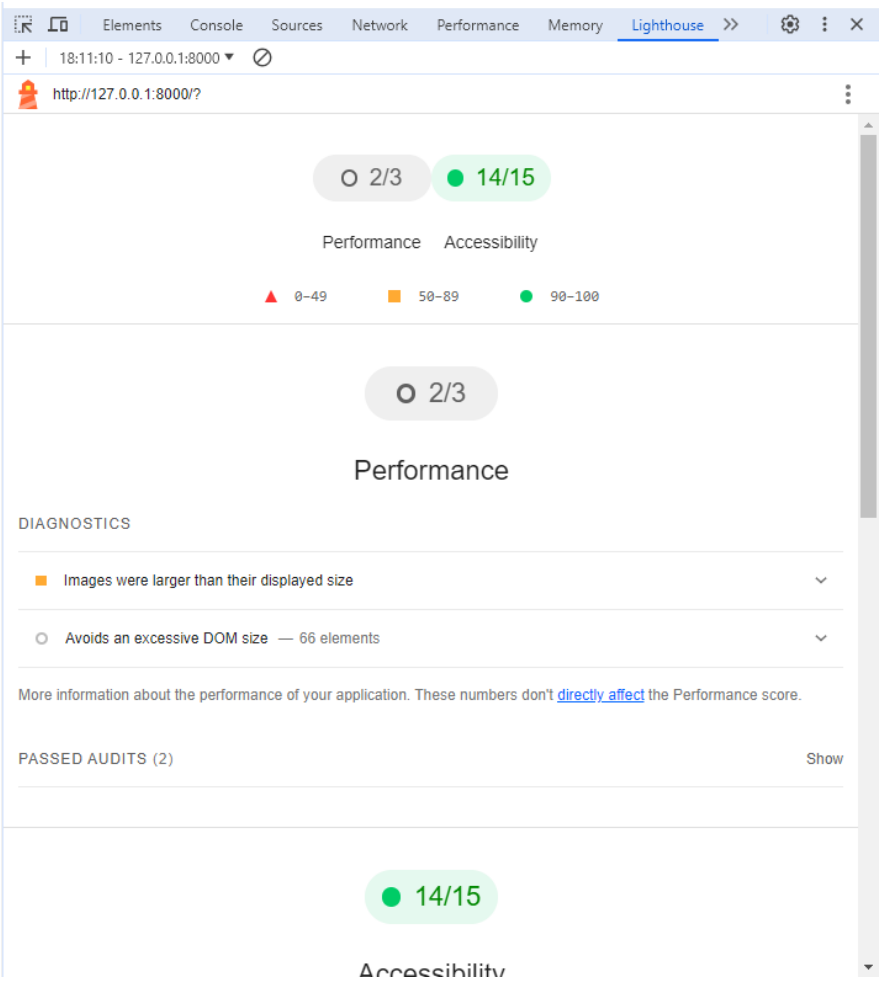

Figura 44. Resultados de accesibilidad de la página de Indicios.

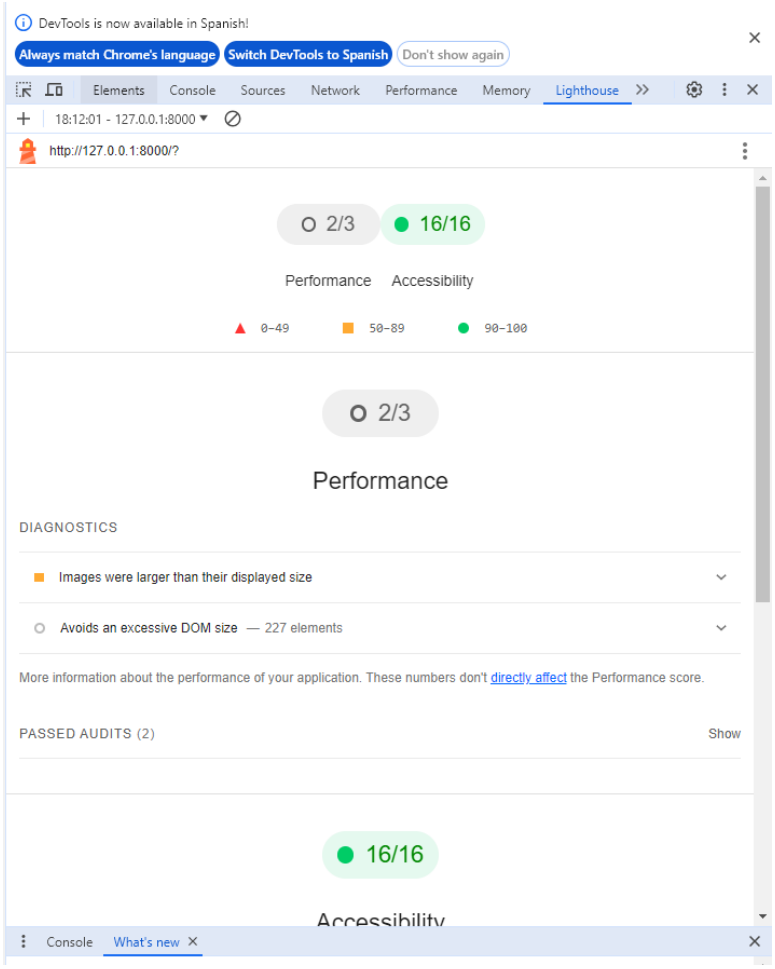

Figura 45. Resultados de accesibilidad de la página de Índices de lecturabilidad.

6.3. Pruebas Unitarias sobre los Índices desarrollados.

También se han desarrollado un conjunto de pruebas unitarias en Python para comprobar el correcto funcionamiento de los índices desarrollados. Para esto, se ha calculado manualmente cual es el resultado esperado para las funciones y estas han sido programadas según la librería de pruebas de Python unitest.

Se han desarrollado quince pruebas con ejemplos básicos para forzar situaciones complejas como pueden ser el control de la división entre 0. Y

posteriormente se han implementado pruebas con ejemplos de textos reales obtenidos de una serie de páginas web que se han listado en la bibliografía.

A continuación, se puede ver en la Tabla 1, cada uno de los test implementados con una breve explicación del resultado que se debería obtener.

Nombre del test	Descripción del test	Valor deseado	Valor obtenido
Extranjerismos 1	Detección de alguna palabra en inglés.	15	15
Extranjerismos 2	Ninguna palabra en extranjero	0	0
Extranjerismos 3	Todo en extranjero	100	100
Extranjerismos 4	Palabras en mallorquín	24	24
Variación de vocabulario 1	Una palabra muy poco frecuente	7.0	7.0
Variación de vocabulario 2	Todas las palabras comunes	0.0	0.0
Variación de vocabulario 3 y 4	Diversidad de palabras	5.6 y 5.2	5.6 y 5.2
Complejidad oracional 1	Una frase de cada tipo	2.0	2.0
Complejidad oracional 2	No hay subordinadas	0.0	0.0
Complejidad oracional 3	Una frase con sentido	2.0	2.0
Complejidad Silábica 1,2,3 y 4	Frases con silabas que cumplen los	0.152,0.111,0.0, 0.101	0.152,0.111,0.0, 0.101

	patrones establecidos		

Tabla 1: Resumen de las pruebas Unitarias

6.3.1. Test de Índice de Extranjerismos.

```
def test_indice_extranjerismos(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "La Wikipedia es una de las páginas web con más visitas del mundo."
    resultado = extranjerismos_index(nlp(texto))
    self.assertEqual(resultado['ind_cantidad_extranjerismos'], 15)  # Suponiendo
```

Figura 46. Test nº1 de extranjerismos.

En este índice hay dos palabras en extranjero-> Wikipedia y web.

El total de palabras son 13. 2/13 =0.15*100 =15

Por este motivo el assertEqual contiene un 15 al final de la línea.

```
def test_indice_extranjerismos2(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "El rey, por su parte, ha querido destacar que el autor practica todos los género
    resultado = extranjerismos_index(nlp(texto))
    self.assertEqual(resultado['ind_cantidad_extranjerismos'], 0)  # Suponiendo que devuelve
```

Figura 47. Test nº2 de extranjerismos.

En este segundo caso la frase introducida es la siguiente:

```
El rey, por su parte, ha querido destacar que el autor practica todos los
géneros y ha valorado la influencia de las literaturas populares, la
tradición de la oralidad, la costumbre de reunirse para contar historias y
la biblioteca familiar.
```

Esta frase no contiene extranjerismos por ese motivo el resultado es 0.

Este texto es completo en ingles por lo que se obtiene un 100%

```python
def test_indice_extranjerismos3(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "here we have some English words"
    resultado = extranjerismos_index(nlp(texto))
    self.assertEqual(resultado['ind_cantidad_extranjerismos'], 100)  # Texto en ingles
```

Figura 48. Test nº3 de extranjerismos.

El siguiente texto introducido es una frase con palabras en mallorquín

```python
def test_indice_extranjerismos4(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "Al evento de presentación asistieron la presidenta del Govern de Baleares, Marga Prohens, y el
    resultado = extranjerismos_index(nlp(texto))
    self.assertEqual(resultado['ind_cantidad_extranjerismos'], 24)  # Texto en ingles
```

Figura 49. Test nº4 de extranjerismos.

6.3.2. Test Índice de frecuencia de palabras.

En estos índices el cálculo se hace en función de la cantidad de palabras poco frecuentes dentro de una lista establecida como palabras frecuentes. La cantidad total de palabras se divide entre el total de palabras poco frecuentes distintas presentes en el texto. Por tanto.

Hay una palabra poco frecuente en un total de 7 palabras por eso el índice obtenido es de 7.

```python
def test_indice_frecuencia_palabras(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "la palabra salamandresa es muy poco frecuente." # s
    texto =nlp(texto)
    resultado = lexic_index(texto)
    self.assertEqual(resultado['ind_variacion_vocab'], 7.0)
```

Figura 50. Test nº1 de variación de vocabulario.

Si todas las palabras son frecuentes se contempla el caso de división entre 0 para que no haya una excepción.

```
def test_indice_frecuencia_palabras2(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "que me haces." # aun que sea una frase sin senti
    texto =nlp(texto)
    resultado = lexic_index(texto)
    self.assertEqual(resultado['ind_variacion_vocab'], 0.0)
```

Figura 51. Test nº2 de variación de vocabulario.

En los dos casos siguientes se han usado textos con algunas palabras poco frecuentes y otras palabras frecuentes, y siguiendo la lista obtuvimos los datos establecidos en el assertEquals, el cual coincide con el resultado obtenido con la aplicación.

```
def test_indice_frecuencia_palabras5(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "La palabra desambiguación te sale mucho cuando usas la Wikipedia.Es una palabra muy larga y difícil."
    texto =nlp(texto)
    resultado = lexic_index(texto)
    self.assertEqual(resultado['ind_variacion_vocab'], 5.667)

def test_indice_frecuencia_palabras4(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "La Junta Directiva de Plena Inclusión Madrid atestiguó su “firme compromiso con la accesibilidad cogn
    texto =nlp(texto)
    resultado = lexic_index(texto)
    self.assertEqual(resultado['ind_variacion_vocab'], 5.25)
```

Figura 52. Test nº3 y nº4 de variación de vocabulario.

6.3.3. Test de complejidad oracional.

En este caso se han establecido dos pruebas forzando las situaciones para que haya una oración subordinada, una oración coordinada y una oración yuxtapuesta. Para obtener de este modo el resultado: 2.0

```
def test_indice_complejidad_oracional(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "El perro ladra y el gato maúlla. Llueve, el suelo se moja. Cuando llegue el tren saldremos de viaje."
    texto =nlp(texto)
    orth_index = orthographic_index(texto)
    resultado = syntactic_index(nlp(texto),orth_index['word_count'])
    self.assertEqual(resultado['ind_complej_estruc_oracional'], 2.0)  # Suponiendo que devuelve el índice de complejidad
```

Figura 53. Test nº1 de complejidad oracional.

Por otro lado, se ha implementado el caso en el que no haya oraciones subordinadas para comprobar que este controlada la división por 0.

```python
def test_indice_complejidad_oracional2(self):
    nlp = spacy.load('es_core_news_sm') # si no hay subordinada
    texto = "El perro ladra y el gato maúlla. Llueve, el suelo se moja. "
    texto =nlp(texto)
    orth_index = orthographic_index(texto)
    resultado = syntactic_index(nlp(texto),orth_index['word_count'])
    self.assertEqual(resultado['ind_complej_estruc_oracional'], 0.0)  # Suponiendo que devuelve el í
```

Figura 54. Test nº2 de complejidad oracional

Y se ha implementado un test extra con una frase coherente de una página web.

```python
def test_indice_complejidad_oracional4(self):
    nlp = spacy.load('es_core_news_sm') # si no hay subordinada
    texto = "En la Wikipedia hay artículos con palabras muy difíciles y frases muy largas. Por e
    texto =nlp(texto)
    orth_index = orthographic_index(texto)
    resultado = syntactic_index(nlp(texto),orth_index['word_count'])
    self.assertEqual(resultado['ind_complej_estruc_oracional'], 2.0)  # no hay yuxtapuesta
```

Figura 55. Test nº3 de complejidad oracional

Donde la frase es la siguiente

```
En la Wikipedia hay artículos con palabras muy difíciles y frases muy
largas. Por ejemplo: en el artículo de acera usa la palabra paramento en
lugar de muro. La palabra desambiguación te sale mucho cuando usas la
Wikipedia.
```

6.3.4. Test de índice vocálico.

Finalmente se han elaborado diversos test sobre este índice en el cual consiste en contar las silabas que cumplen los patrones establecidos y dividirlos entre el total de silabas.

```python
def test_indice_vocalico(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "la matrona fue muy buena en su trabajo."
    resultado = syllabic_index(nlp(texto))
    self.assertEqual(resultado['ind_baja_frecuencia'], 0.154)  # Suponiendo que devuelve el i

def test_indice_vocalico5(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "El Comité Paralímpico Español presentó este lunes los objetivos del equipo nacio
    resultado = syllabic_index(nlp(texto))
    self.assertEqual(resultado['ind_baja_frecuencia'], 0.067)  # Suponiendo que devuelve el i

def test_indice_vocalico2(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "el atril era de madera." # otro patron
    resultado = syllabic_index(nlp(texto))
    self.assertEqual(resultado['ind_baja_frecuencia'], 0.111)  # Suponiendo que devuelve el i

def test_indice_vocalico3(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "no hay silabas contenidas en los casos analizados." # otro patron
    resultado = syllabic_index(nlp(texto))
    self.assertEqual(resultado['ind_baja_frecuencia'], 0.0)  # Suponiendo que devuelve el inc

def test_indice_vocalico4(self):
    nlp = spacy.load('es_core_news_sm') #Spanish
    texto = "La Junta Directiva de Plena Inclusión Madrid atestiguó su “firme compromiso con
    resultado = syllabic_index(nlp(texto))
    self.assertEqual(resultado['ind_baja_frecuencia'], 0.101)  # Suponiendo que devuelve el i
```

Figura 56. Tests de índice vocálico.

En el caso 1: hay 2 de 13 silabas que cumplen el patrón.

En el caso 5 la oración es:

```
El Comité Paralímpico Español presentó este lunes los objetivos del equipo
nacional de cara a los próximos Juegos Paralímpicos de París 2024, así como
a los nueve deportistas de Baleares preseleccionados para competir en la
gran cita, que tendrá lugar en la capital francesa del 28 de agosto al 8
de septiembre.
```

Hay 6 silabas que cumplen el patrón de 54 silabas totales.

En el caso 2: hay una de 8 silabas que cumple el patrón.

Y así sucesivamente con los 5 casos.

Estos casos han ayudado a la comprobación del correcto funcionamiento de las funciones.

7. Conclusiones y líneas futuras

Durante el desarrollo de este Trabajo de Fin de Grado, se han alcanzado los objetivos principales de mejorar la interfaz de la página web y expandir su conjunto de índices de análisis. Sin embargo, como es común en proyectos de esta envergadura, han surgido oportunidades de mejora y desafíos que podrían abordarse en futuras iteraciones.

En cuanto al aspecto visual y funcional de la interfaz, se logró una mejora significativa que se alinea con la visión inicial del proyecto. La interfaz resultante es más intuitiva y atractiva para el usuario, lo que mejora la experiencia general de navegación. Sin embargo, aunque el diseño general es sólido, aún existen áreas específicas que podrían pulirse para alcanzar un nivel óptimo de usabilidad. Por ejemplo, ciertos elementos de diseño podrían refinarse para mejorar la coherencia visual y la accesibilidad, y se podrían explorar nuevas técnicas de diseño para optimizar la disposición de los elementos en la pantalla.

En cuanto a los índices de análisis desarrollados, se lograron avances significativos, aunque se tuvo que reducir el alcance debido a limitaciones de recursos y tiempo. La imposibilidad de elaborar un índice que analizara los grados de subordinación de las frases es un ejemplo de cómo las limitaciones técnicas pueden afectar el desarrollo de un proyecto. Sin embargo, esta limitación también presenta una oportunidad para futuras investigaciones y mejoras en el ámbito de los análisis de texto.

En términos de eficiencia y rendimiento de la página web, se identificaron áreas en el código que podrían optimizarse para mejorar la velocidad de carga y la capacidad de respuesta. Un análisis exhaustivo del código podría revelar oportunidades para reducir la complejidad y mejorar la eficiencia, lo que tendría un impacto positivo en la experiencia del usuario.

Como línea futura de trabajo, se sugiere continuar con la implementación de nuevos índices o la mejora de los existentes. Dado el tiempo limitado del Trabajo de Fin de Grado, no se pudieron cubrir todos los aspectos con la profundidad deseada, lo que deja espacio para futuras investigaciones y desarrollos. Además, se podrían explorar nuevas tecnologías y metodologías para abordar los desafíos

técnicos identificados y mejorar aún más la calidad y la funcionalidad de la aplicación.

En resumen, a pesar de los desafíos encontrados y las áreas de mejora identificadas, el Trabajo de Fin de Grado ha logrado cumplir con sus objetivos principales. La interfaz de la página web ha sido mejorada significativamente, y se han incorporado nuevos índices de análisis para ofrecer una experiencia más completa y útil para los usuarios y se ha conseguido realizar el despliegue del servicio web mediante la herramienta Docker.

7.1. Análisis de impacto

El enfoque primordial de este proyecto ha sido facilitar a las personas la verificación de la validez de sus textos desde la perspectiva de la metodología de la lectura fácil, abordando tanto la interpretación cualitativa como cuantitativa de esta calidad. Inicialmente dirigido a expertos, la mejora de la interfaz tiene como objetivo principal hacer accesible esta herramienta a un público más amplio, incluyendo a aquellos menos familiarizados con el tema. La intención es democratizar el acceso a nuestros índices, permitiendo que incluso las personas menos entendidas puedan comprender la información proporcionada. El impacto principal de este Trabajo de Fin de Grado es de naturaleza social, ya que no solo aspiramos a ayudar a individuos a nivel individual, sino también a subrayar la importancia de la comprensión lectora en nuestra sociedad.

Este TFG, a nivel personal me ha ayudado tanto en la adquisición de nuevos conocimientos académicos relacionados con la informática como puede ser el despliegue de un servicio con la herramienta Docker, o el desarrollo de una aplicación web mediante HTML y CSS ya que nunca había trabajado con ninguna de ellas. También he podido aprender las necesidades de un texto para que este sea más fácil de leer, lo cual me será de ayuda en mi vida laboral.

Me alegra saber que este Trabajo de Fin de Grado tenga un fin de ayudar a que las personas con dificultades puedan llegar a tener el derecho innegociable de la lectura.

8.Bibliografía

[1] Ventosa Pérez, A. (s.f.). Servicio Web de Lectura Facil. Calculador de Indices de Lecturabilidad. Recuperado de https://oa.upm.es/71168/1/TFG_ANTONIO_VENTOSA_PEREZ.pdf

[2] Liefeldt, P. (s.f.). Comprensión lectora: importancia y estrategias para su desarrollo. Recuperado de https://es.linkedin.com/pulse/comprensi%C3%B3n-lectora-importancia-y-estrategias-para-su-dar%C3%ADo-liefeldt

[3] Instituto Nacional de Salud Infantil y Desarrollo Humano (s.f.). Trastornos de la lectura. Recuperado de https://espanol.nichd.nih.gov/salud/temas/reading/informacion/trastornos

[4] Revista Científica de la Universidad Nacional de Asunción, Volumen(Número), páginas. Recuperado de https://revistascientificas.una.py/index.php/rcff/article/view/2717

[5] La Dislexia. Comprensión Lectora y Estrategias Metacognitivas. Retrieved from https://ladislexia.net/comprension-lectora-y-estrategias-metacognitivas/

[6] Legible. (s.f.). Recuperado de https://legible.es/

[7] UNE 153101:2018 EX. Lectura Fácil. Pautas y recomendaciones para la elaboración de documentos.

[8] Comunicación Clara. (s.f.). Recuperado de https://clara.comunicacionclara.com/

[9] LinguaKit. (s.f.). Análisis Completo. Recuperado de https://linguakit.com/es/analisis-completo#google_vignette

[10] Lexicool. (s.f.). Text Analyzer. Recuperado de https://www.lexicool.com/text_analyzer.asp?IL=3

[11] UNED. (s.f.). Hemingway. Blogs UNED. Recuperado de https://blogs.uned.es/herramientasautocorreccionescrituraingles/hemingway/

[12] Wikipedia. (2024, mayo 19). Grammarly. En Wikipedia, la enciclopedia libre. Recuperado de https://es.wikipedia.org/wiki/Grammarly

[13] Readable. Flesch Reading Ease and Flesch-Kincaid Grade Level. Recuperado de https://readable.com/readability/flesch-reading-ease-flesch-kincaid-grade-level/

[14] Universidad de Salamanca. E2R Helper. Recuperado de http://recursos-inico.usal.es:3838/e2rh/

[15] Universidad de Salamanca. Asistente de Lectura Fácil. Recuperado de https://eapoyo-inico.usal.es/asistente-lectura-facil/

[16] TRBL Services. Introducción a Selenium IDE. Recuperado de https://trbl-services.eu/blog-introduccion-selenium-ide/

[17] SEMrush. Cómo Utilizar Google Lighthouse. Recuperado de https://es.semrush.com/blog/como-utilizar-google-lighthouse/?kw=&cmp=ES_SRCH_DSA_Blog_ES&label=dsa_pagefeed&Network=g&Device=c&utm_content=678247170630&kwid=dsa-2229731903223&cmpid=19249322774&agpid=157746394729&BU=Core&extid=109498533778&adpos=&gad_source=1&gclid=Cj0KCQjwxqayBhDFARIsAANWRnSiTmoNPn2qgfRRirFns2HJcBP91LLo9W48TO_raApbK0t0S8TnWtgaAkYDEALw_wcB

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
	Fecha/Hora	Sun Jun 02 11:29:44 CEST 2024
	Emisor del Certificado	EMAILADDRESS=camanager@etsiinf.upm.es, CN=CA ETS Ingenieros Informaticos, O=ETS Ingenieros Informaticos - UPM, C=ES
	Numero de Serie	561
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)