

Base De Datos

I. RESUMEN

Para esta base de datos, se realizó una consulta inicial de 500 artículos relacionados con el tema de ('Frameworks' AND 'Web Development'). Posteriormente, se lleva a cabo un análisis de estos artículos para identificar los más relevantes en relación con el tema elegido. Este proceso de análisis permite seleccionar aquellos artículos que mejor se ajustan a los criterios establecidos, proporcionando así una base sólida para la investigación o la búsqueda de información específica en el ámbito del desarrollo web y frameworks.

II. INTRODUCCIÓN

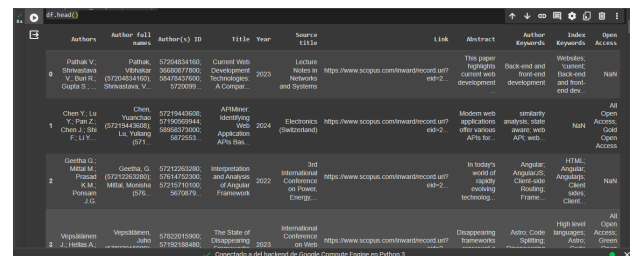
La presente investigación se llevó a cabo con el objetivo de explorar y analizar diversos artículos según el tema de frameworks y el desarrollo web, utilizando una metodología que combina el análisis de datos y la revisión bibliográfica. Para la recopilación de artículos, se utilizó la plataforma Scopus, reconocida por su extensa base de datos de investigación científica. Para ello, se realizó una búsqueda utilizando términos clave relacionados con framework y desarrollo web, lo que resultó en una selección inicial de 500 artículos pertinentes.

Una vez recopilados, los artículos se exportaron en formato CSV y se trasladaron al entorno de Google Colab, para el análisis de datos. Utilizando el código preestablecido proporcionado en Python por el instructor, se llevó a cabo un análisis detallado de los artículos descargados, con el objetivo de identificar aquellos más relevantes y significativos para el tema de interés.

III. RESULTADOS

1. Se empieza con un conjunto de importaciones en Python que trae herramientas para análisis y procesamiento de texto. La biblioteca **Pandas** facilita la manipulación de datos tabulares, mientras que **NLTK** ofrece una funciones para procesamiento de lenguaje natural, como la tokenización de oraciones y palabras, y la eliminación de stopwords. La función **matplotlib.pyplot** permite la creación de visualizaciones como gráficos, y **WordCloud** es especialmente útil para generar nubes de palabras que resalten las más frecuentes en un texto.

2. Posteriormente descargando el conjunto de palabras vacías (stopwords) de la biblioteca **NLTK**. Estas palabras comunes se eliminan del análisis de texto ya que en su mayoría no aportan significado contextual, como "a", "el", "en", etc. Así se filtran mejor y mejora la calidad del análisis de texto.
3. Con la siguiente función de **nltk.download('punkt')** se descarga el modelo de datos necesario para utilizar el tokenizador de oraciones y palabras proporcionado por **NLTK**.
4. Se utiliza la biblioteca **Pandas** para leer el archivo CSV llamado '**scopus.csv**' y cargar su contenido en el DataFrame. Luego, el método **head()** se utiliza para mostrar las primeras filas del DataFrame, lo que proporciona una vista previa de los datos y cómo estos están estructurados.



	Authors	Author full name	Author(s) ID	Title	Year	Source title	Link	Abstract	Index keywords	Open Access
0	Patlak V, Stepanova V, Ben R, Gupta S, Stevanova V	Patlak V, Stepanova V, Ben R, Gupta S, Stevanova V	57204534100, 5650577800, 58479437600, 57204534100, 57204534100	Current Web Development Technologies: A Comparison	2023	Lecture Notes in Networks and Systems	https://www.scopus.com/wardirect/article/pii/S2467986923000000	This paper highlights current web development technologies and their impact on the industry.	Web development, Back-end and front-end development, Python, JavaScript, React, Node.js	Yes
1	Chen Y, Li Y, Pan Z, Chen J, Shi F, Li Y	Chen Y, Li Y, Pan Z, Chen J, Shi F, Li Y	57215443600, 57190505844, 56955275000, 56955275000, 56955275000	API-driven Development: A Comprehensive Review	2024	Electronics	https://www.scopus.com/wardirect/article/pii/S2079910X24000000	Modern web applications often require various APIs to interact with external services and data sources.	APIs, Web development, Cloud computing, Data integration, Software development	Yes
2	Gentile G, Milla M, Pineda R, Milla M, Pineda R	Gentile G, Milla M, Pineda R, Milla M, Pineda R	57212963300, 57194723300, 57194723300, 57194723300, 57194723300	3rd International Conference on Frontiers in Energy	2023	Information and Analytics in Energy	https://www.scopus.com/wardirect/article/pii/S2467986923000000	In today's world of rapidly evolving technology, energy systems are becoming increasingly interconnected and data-driven.	Energy, Analytics, Cloud computing, Data integration, Software development	Yes
3	Vasilevski J, Hristov A	Vasilevski J, Hristov A	57192110000, 57192110000	The State of Disrupting in 2023	2023	International Conference on Web	https://www.scopus.com/wardirect/article/pii/S2467986923000000	Disrupting technologies are reshaping the business landscape, creating new opportunities and challenges.	Disrupting technologies, Business, Innovation, Digital transformation	Yes

Figura 1. Estructuración de los datos.

5. Luego de ello, se crea un conjunto de palabras vacías (stopwords) en inglés utilizando la biblioteca **NLTK**. La función **stopwords.words('english')** carga un conjunto predefinido de stopwords en inglés, y luego este conjunto se convierte en un conjunto de Python utilizando la función **set()**. Una vez creado, este conjunto puede ser utilizado para filtrar stopwords de un texto en inglés durante el procesamiento de lenguaje natural.
6. Después se seleccionan 150 títulos más frecuentes agrupados por la columna 'Source title' del DataFrame, que contiene los datos de los artículos obtenidos de la base de datos Scopus. Ya teniendo esto se crea una visualización utilizando **Matplotlib**. Se especifica un tamaño de

figura de 20 por 5 pulgadas con la función `(plt.figure(figsize=(20, 5)))`. Por otra parte, Se utiliza la función `kind='bar'` en el método `plot()` para generar un gráfico de barras que muestra la cantidad de títulos para cada fuente. Finalmente, `plt.show()` se utiliza para mostrar la visualización.

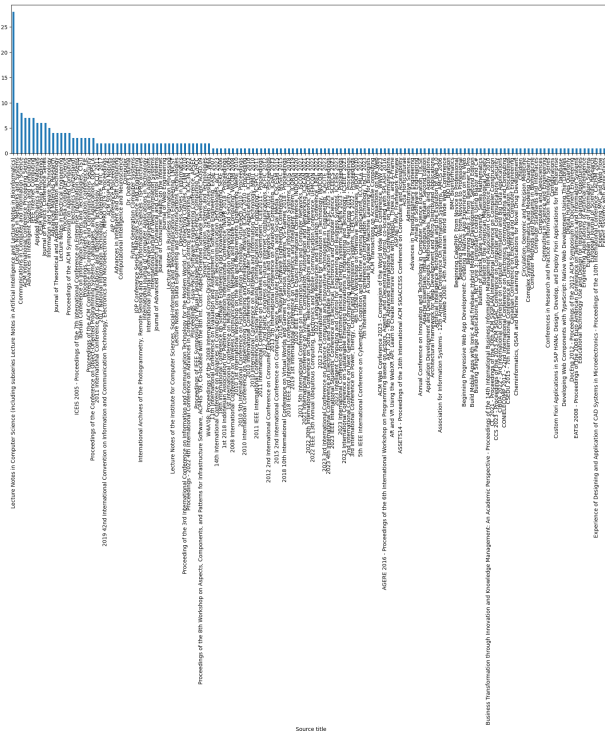


Figura 2. Agrupación de artículos por título.

- Se agrupan los datos por el título de la fuente ('Source title') y cuenta el número de artículos de cada fuente. Luego, selecciona las 10 fuentes con más artículos y crea un gráfico de barras que muestra el número de artículos publicados por cada una de estas fuentes.

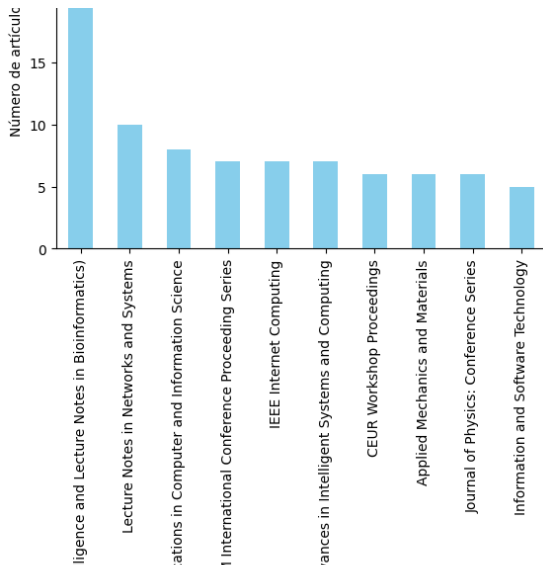


Figura 3. Fuentes con más artículos.

- Se usa La función `clean_text` la cual prepara y limpia el texto antes de realizar un análisis de texto o procesamiento de lenguaje natural (NLP). Para ello se elimina la puntuación del texto, como caracteres irrelevantes que no aportan significado al análisis. Además, filtra las palabras vacías, conocidas como stopwords.

- Posteriormente se realiza un análisis de los títulos de los artículos en el DataFrame, con el objetivo de identificar las palabras más frecuentes y relevantes en el de texto. Para ello se cuenta la frecuencia de cada palabra en todos los títulos y crea un DataFrame que contiene estas palabras junto con su frecuencia de aparición. Este DataFrame se ordena según la frecuencia de las palabras de mayor a menor, y se le asigna a cada palabra un rango numérico en función de su frecuencia. De esta manera se ofrece una visión rápida de las palabras más comunes en los títulos de los artículos.

	Word	Frequency	Rank
0	web	305	1.0
1	development	162	2.0
2	framework	116	3.0
3	application	62	4.0
4	based	58	5.0
5	applications	56	6.0
6	using	52	7.0
7	system	46	8.0
8	design	36	9.0
9	information	34	10.0
10	study	32	11.0
11	frameworks	29	12.0
12	management	28	13.0

Figura 4. Ranking De Palabras.

13	software	28	13.0
14	learning	25	14.0
15	international	25	14.0
16	approach	25	14.0
17	engineering	25	14.0
18	data	23	15.0
19	model	22	16.0

Next steps: [Generate code with word_df](#)

Figura 5. Ranking De Palabras.

10. Ya teniendo el ranking de palabras se genera una gráfica con una nube de palabras a partir de los títulos de los artículos, lo que proporciona una vista rápida de las palabras más frecuentes en el texto. Para lograr esto, se aplica una función de limpieza de texto a cada título. Luego, concatena todos los títulos limpios en un solo texto. Finalmente, utilizando la biblioteca **WordCloud**, se crea una nube de palabras donde el tamaño de cada palabra está determinado por su frecuencia en el texto. Es decir, las palabras más frecuentes son más prominentes, lo que permite identificar rápidamente los temas dominantes en los títulos de los artículos.

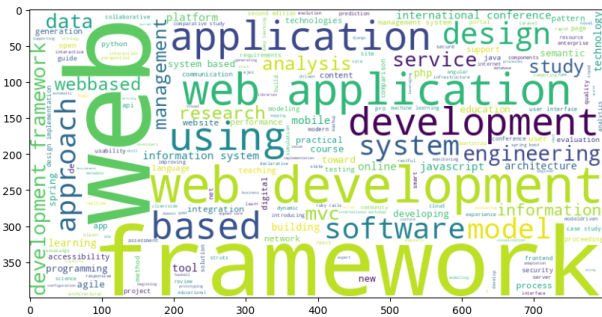


Figura 6. Nube De Palabras.

11. Se utiliza la biblioteca Matplotlib para crear un gráfico de barras que muestra las 25 palabras más frecuentes en el DataFrame. Para ello, se ordena el DataFrame por la frecuencia de las palabras en orden descendente y se seleccionan las 25 palabras más frecuentes. Luego, se utiliza el método `plot()` del DataFrame para generar el gráfico de barras, especificando las columnas 'Word' y 'Frequency', y se indica que se desea un gráfico de barras con la función `kind='bar'`. Esto proporciona una representación concisa de las palabras más frecuentes en el DataFrame, lo que permite identificar las principales palabras clave.

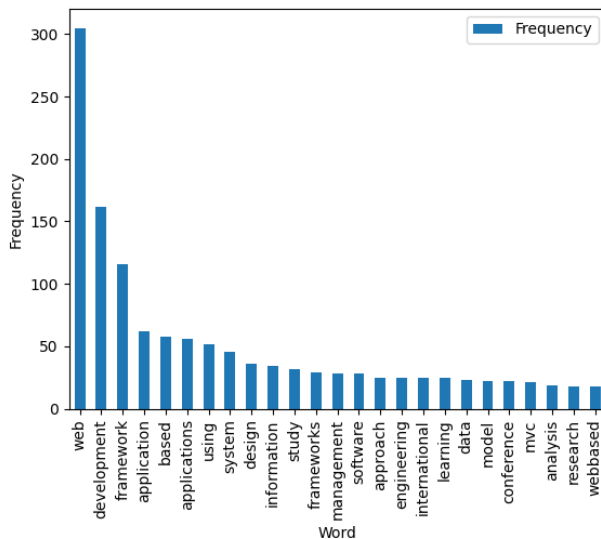


Figura 7. Palabras más frecuentes.

12. Teniendo en cuenta lo anterior se realiza un análisis detallado de los títulos de los artículos, enfocándose en identificar aquellos que contienen un mayor número de palabras clave. Es importante hacer uso de la función *clean_text* ya que limpia los títulos y utilizando un objeto Counter, se cuentan las frecuencias de las palabras clave en los títulos, así se seleccionan las 15 palabras clave más comunes para el análisis. Luego, se define una función *count_keywords* para determinar cuántas de estas palabras clave están presentes en cada título de artículo. Esto permite identificar los más relevantes en función de la presencia de estas palabras clave. Finalmente, se visualizan los 10 principales artículos con la mayor cantidad de palabras clave a través de un gráfico de barras.

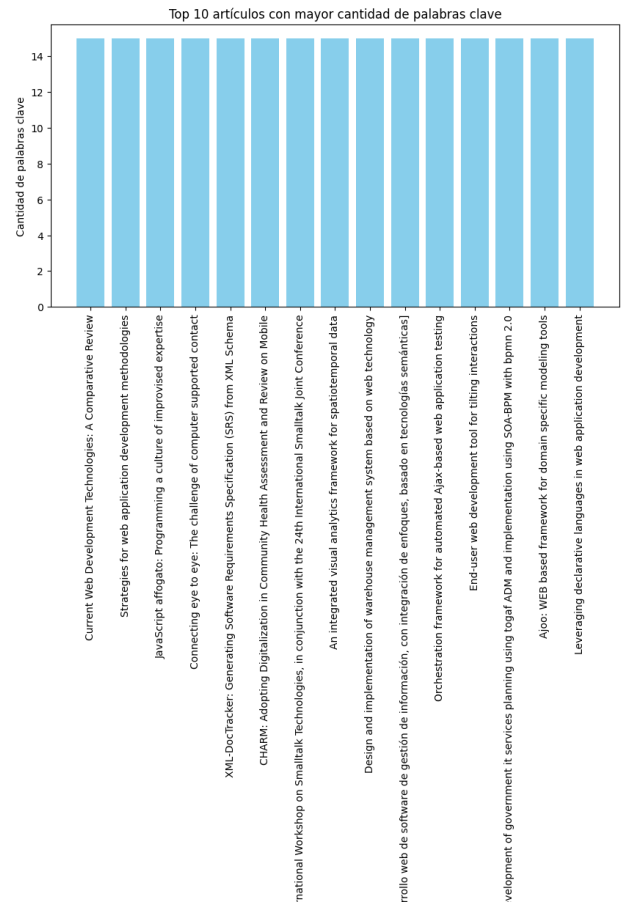


Figura 8. Artículos con más palabras clave.

13. Ya teniendo los artículos más relevantes se resumen los abstracts de los artículos que contienen un mayor número de palabras clave. Para ello, utiliza el DataFrame previamente ordenado por el recuento de palabras clave para seleccionar los abstracts de los 10 artículos principales. Luego, define una función llamada *summarize_abstract* que toma un abstract como entrada y lo resume en tres oraciones clave. Esto se logra mediante el cálculo de una puntuación para cada oración. Posteriormente, aplica esta función de resumen a cada abstract seleccionado utilizando el método *apply*, generando así una serie de

resúmenes para los abstracts de los artículos seleccionados. Finalmente, muestra cada resumen del 1 al 10 para indicar su posición en la lista de abstracts seleccionados.

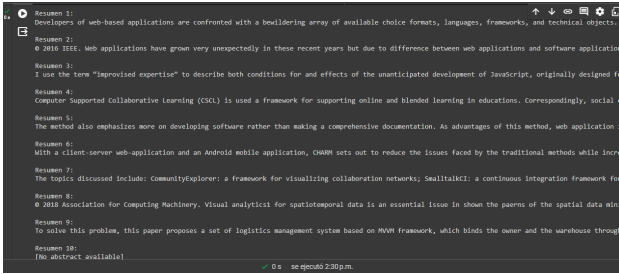


Figura 9. Resumen de los 10 artículos.

14. Al identificar los artículos más relevantes mediante el conteo de palabras clave tanto en sus títulos como en sus abstracts. Se cuenta la aparición de palabras clave en cada título de artículo, utilizando una función `lambda` que aplica la limpieza de texto y luego suma el número de palabras clave presentes en el título. Luego, se seleccionan los abstracts de los 10 artículos con mayor conteo de palabras clave. Se hace un resumen con la función `summarize_abstract` se aplica a cada abstract seleccionado y se almacena junto con el título correspondiente en el DataFrame `top_articles`. Finalmente, se muestra cada título de artículo junto con su resumen correspondiente.

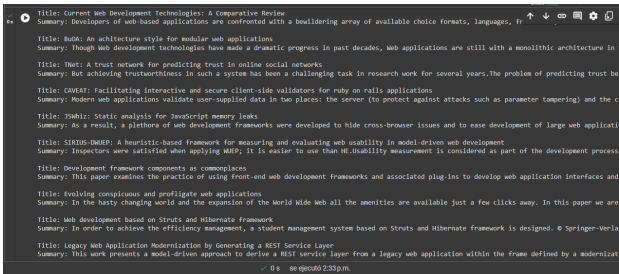


Figura 10. Artículos más relevantes.

IV. CONCLUSIONES

1. La visualización de los resultados mediante gráficos y resúmenes permite una comprensión rápida y clara de la información. Esto es crucial para comunicar los hallazgos a colegas, superiores o cualquier otra parte interesada de manera efectiva, facilitando así la toma de decisiones fundamentadas.
2. El código proporcionado es una herramienta poderosa para analizar múltiples artículos. Utiliza técnicas como el conteo de palabras clave y la generación de resúmenes para obtener una comprensión detallada del contenido. Esto es especialmente útil para identificar patrones y temas comunes entre los artículos, facilitando la investigación multidisciplinaria y el análisis comparativo.
3. La capacidad de análisis profundo es especialmente útil para investigadores y analistas que buscan comprender la

amplitud y la diversidad de la literatura científica en un área específica. Además, la automatización de tareas y la visualización clara de los resultados contribuyen a una mayor eficiencia en el proceso de análisis.

V. REFERENCIAS

1. Google colab. (s/f). Google.com. Recuperado el 18 de mayo de 2024, de <https://colab.research.google.com/drive/1Ottgq4wSFXIWbppmWMCU07EXqh1OtK3Q>
2. (S/f). Edu.co:2443. Recuperado el 18 de mayo de 2024, de <https://recursosvirtuales.konradlorenz.edu.co:2443/login?url=https://www.scopus.com/home.url>