UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

Tema:

MOTOR DE SUGERENCIAS APLICANDO WEB SCRAPING PARA LA TOMA DE DECISIÓN EN LA COMPRA DE CALZADO EN LA LÍNEA DEPORTIVA

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Tecnologías de la Información

ÁREA: Gestión de tecnologías de la información

LÍNEA DE INVESTIGACIÓN: Tecnologías de información y Sistemas de control

AUTOR: Jorge Luis Ibarra Lucas

TUTOR: Ing. Rubén Eduardo Nogales Portero, Mg

Ambato – Ecuador

febrero - 2024

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: MOTOR DE SUGERENCIAS APLICANDO WEB SCRAPING PARA LA TOMA DE DECISIÓN EN LA COMPRA DE CALZADO EN LA LÍNEA DEPORTIVA desarrollado bajo la modalidad Proyecto de Investigación por el señor Jorge Luis Ibarra Lucas, estudiante de la Carrera de Ingeniería en Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, febrero 2024

Firmado electrónicamente por:
RUBEN EDUARDO
NOGALES PORTERO

Ing. Rubén Eduardo Nogales Portero, Mg

TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: MOTOR DE SUGERENCIAS APLICANDO WEB SCRAPING PARA LA TOMA DE DECISIÓN EN LA COMPRA DE CALZADO EN LA LÍNEA DEPORTIVA es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

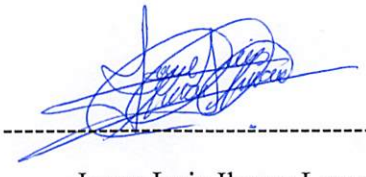
Ambato, febrero 2024

Jorge Luis Ibarra Lucas
C.C. 1313401117
AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, febrero 2024



Jorge Luis Ibarra Lucas

C.C. 1313401117

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por el/la señor/señorita Jorge Luis Ibarra Lucas, estudiante de la Carrera de Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado MOTOR DE SUGERENCIAS APLICANDO WEB SCRAPING PARA LA TOMA DE DECISIÓN EN LA COMPRA DE CALZADO EN LA LÍNEA DEPORTIVA, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con la señora Presidente del Tribunal.

Ambato, febrero 2024

Firmado electrónicamente por:
ELSA PILAR URRUTIA URRUTIA

Ing. Elsa Pilar Urrutia Urrutia, Mg.
PRESIDENTE DEL TRIBUNAL

Firmado electrónicamente por:
FELIX OSCAR FERNANDEZ PENA

PhD. Ing. Félix Oscar Fernández Peña
PROFESOR CALIFICADOR

Firmado electrónicamente por:
JULIO ENRIQUE BALAREZO LOPEZ

PhD. Julio Enrique Balarezo López
PROFESOR CALIFICADOR

DEDICATORIA

Dedico este proyecto a Dios, este es el fruto de su fidelidad y su gran amor a mi vida, el cual me ha otorgado la sabiduría necesaria para culminar uno de varios proyectos importantes en mi etapa universitaria y ha sido mi luz en medio de la oscuridad.

A mis queridos padres, por haberme brindado total apoyo en medio del proceso, en el que a pesar de los problemas han estado conmigo y me alentaron cuando he decaído. Esta es la cosecha de su paciencia y amor que han cultivado en mi vida, se lo dedico a ustedes con todo mi corazón.

A mis hermanos quienes han sido mi mayor motivación para desarrollar este proyecto, por su compañía y amor incondicional que han sido el combustible de felicidad en mi corazón.

AGRADECIMIENTO

Expreso mi profundo agradecimiento a Dios por haberme brindado la capacidad de amar lo que hago durante cada etapa de aprendizaje en clases junto a mis compañeros y amigos.

Quiero extender mi más sincero agradecimiento a mis padres por su generosa inversión, tanto económica como emocional, para permitirme completar este proyecto. Su constante apoyo y dedicación han sido fundamentales en este camino. A mi hermana, un pilar inquebrantable en mi vida, le agradezco de corazón por ser mi inspiración y motivación para superarme cada día.

Asimismo, no puedo pasar por alto el invaluable apoyo brindado por mis amigos cercanos. Su presencia y habilidades han sido un pilar fundamental en cada fase de este proyecto, permitiéndome crecer y desenvolverme de la mejor manera posible.

ÍNDICE GENERAL DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN EJECUTIVO

La compra en línea de calzado deportivo se ve desafiada por la amplia variedad de marcas y modelos, complicando las decisiones para los consumidores. La imposibilidad de probar los zapatos antes de comprarlos hace que dependan de información proporcionada por fabricantes y reseñas, las cuales pueden ser inexactas o incompletas. Esto dificulta la toma de decisiones informadas, llevando a los consumidores a invertir tiempo y esfuerzo en comparar modelos y marcas, generando frustración y desánimo.

Para mejorar la experiencia de compra, este proyecto presenta una solución integral con técnicas de Web Scraping mediante Scrapy y Selenium, junto con un sistema de recomendación basado en contenidos, con el objetivo de mejorar el proceso de toma de decisiones en la compra de calzado deportivo y ahorrar el tiempo de búsqueda.

El motor de sugerencias aprovecha la información recopilada mediante Web Scraping en las páginas de diversas marcas, eliminando la necesidad de búsquedas exhaustivas de productos. Estos datos alimentan un sistema de recomendación basado en contenidos implementado en Flask, actuando como servidor Web Server Gateway Interface (WSGI). Además, se implementa una interfaz de usuario interactiva mediante el framework ReactJs, brindando a los usuarios la capacidad de visualizar recomendaciones de productos de manera intuitiva. Estas recomendaciones se generan basándose en sus preferencias individuales y en productos previamente seleccionados. Los resultados obtenidos de esta implementación revelan una mejora significativa en el proceso de toma de decisiones de los usuarios, simplificando la búsqueda y proporcionando recomendaciones personalizadas que se alinean con sus preferencias individuales.

Palabras clave: Basado en contenido, sistema de recomendación, compras online, calzado deportivo, toma de decisión, Flask, ReactJs, Web Scraping, Selenium, Scrapy.

ABSTRACT

Online footwear purchases are often challenged by the vast array of brands and designs, posing a difficulty for consumers in making decisions. The inability to physically try on shoes prior to purchasing leaves buyers reliant on information provided by manufacturers and reviews, which may be inaccurate or incomplete. This reliance on potentially unreliable sources makes informed decision-making even more challenging, leading consumers to invest time and effort in comparing models and brands, generating frustration and discouragement. To enhance the shopping experience, this project presents a comprehensive solution with Web Scraping techniques using Scrapy and Selenium, along with a content-based recommendation system, aiming to improve the decision-making process in the purchase of athletic footwear and save search time.

The recommendation engine leverages information gathered through Web Scraping on pages of various brands, eliminating the need for exhaustive product searches. These data feed into a content-based recommendation system implemented in Flask, acting as a Web Server Gateway Interface (WSGI) server. Additionally, an interactive user interface is implemented using the ReactJs framework, providing users with the ability to intuitively view product recommendations. These recommendations are generated based on their individual preferences and previously selected products.

The results obtained from this implementation reveal a significant improvement in the decision-making process for users, simplifying the search and providing personalized recommendations that align with their individual preferences.

Keywords: Content based, recommendation system, online purchase, footwear, decision making, Flask, ReactJs, Web Scraping, Selenium, Scrapy.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

MOTOR DE SUGERENCIAS APLICANDO WEB SCRAPING PARA LA TOMA DE DECISIÓN EN LA COMPRA DE CALZADO EN LA LÍNEA DEPORTIVA.

1.1.1 Planteamiento del problema

A inicios de la humanidad el calzado adquiere un papel crucial, especialmente cuando los desplazamientos implicaban terrenos incómodos durante caminatas o cacerías en aquella época. Por esta razón, se inició el proceso de curtido de la piel de animales con el propósito de utilizarla en diversas aplicaciones, siendo la protección y cuidado de los pies uno de los objetivos primordiales de este desarrollo. Con el transcurso del tiempo, el calzado experimentó una notable evolución en cuanto a su implementación y diseño, adaptándose a diversos usos específicos como la moda casual, formal, deportiva y laboral. En el contexto actual del 2023, donde la actividad de caza ha disminuido, la necesidad de calzado persiste, impulsada significativamente por las condiciones geográficas y la cultura de la sociedad. En consecuencia, la venta de calzado se ha convertido en un producto altamente demandado, por ello, las empresas dedicadas a este sector implementan estrategias como el marketing digital y el comercio electrónico en un entorno tecnológico para satisfacer esta creciente demanda [1].

En contexto a la venta de calzado, las industrias se benefician significativamente del comercio electrónico, debido a la creciente tendencia de compras por medio de Internet. Esta estrategia respalda la utilización de una red global de computadoras interconectadas, para permitir una comunicación eficiente y el intercambio de información clasificada según las preferencias individuales de cada usuario [2]. Por lo tanto, cada vez más personas prefieren realizar sus compras desde la comodidad de sus hogares, aprovechando las ventajas el e-commerce que incluyen la rapidez y facilidad de compra ante la amplia variedad de productos disponibles y los precios competitivos que disponen las empresas del calzado [3].

En este escenario, las empresas emplean el desarrollo de páginas web y el marketing digital como herramientas fundamentales para instaurar procesos de comercio en línea. De esta manera, pueden ofrecer información pertinente sobre sus productos de manera atractiva y visible para los usuarios interesados, facilitando así la interacción y el acceso a los productos y servicios que ofrecen [4]. En consecuencia, el comercio de calzado en la línea deportiva trasciende con el uso de Internet de esta forma facilita la obtención de información sobre los diferentes diseños y características que tienen disponibles las empresas hacia los usuarios, el cual, genera amplias opciones de productos de calzado, donde, por su lado la exponencial búsqueda de estos productos para el usuario conlleva a ser lento para su decisión de compra [5].

De esta forma, la existencia de una amplia variedad de páginas web que ofrecen calzado deportivo conlleva a una complicación significativa en la toma de decisiones durante el proceso de compra. Este exceso de opciones puede resultar abrumador para los usuarios, extendiendo considerablemente el tiempo que destinan a decidir qué producto adquirir. Además, la abundancia de alternativas y la diversidad de características presentes en el calzado deportivo generan confusión, lo que se traduce en una dificultad para tomar decisiones informadas y rápidas por parte de los consumidores. Este problema se agrava por la falta de herramientas y recursos que permitan a los usuarios comparar de forma sencilla y eficaz las características de los diferentes productos. Como resultado, los usuarios se hallan abrumados por el tiempo invertido en la extensa búsqueda del producto, lo que también suele llevar una compra que no se adapta a sus necesidades o preferencias.

1.2 Antecedentes investigativos

En esta sección, se describe la implementación de una investigación bibliográfica con el objetivo de identificar artículos relevantes para el estudio de investigación. La búsqueda se realizó en las siguientes bases de datos científicos: Scopus, IEEE Xplore, ACM Digital Library, Wiley, MDPI, Springer y Computational Communication Research.

Para la obtención de los artículos, se utilizaron cadenas de búsqueda que incluyen las palabras clave Web Scraping, Recommender System, Websites, Sports Shoes, Decision Making y Crawling que sustentan el trabajo mediante fundamentos tecnológicos en base al tema del proyecto de investigación y objetivos, también la búsqueda se limitó a publicaciones publicadas entre los años 2019 y 2023.

A continuación, en la Tabla 1 se describen las cadenas de búsqueda implementadas para cada base de dato científico y el número de artículos o revistas científicas resultantes.

Tabla 1. Cadenas de Búsqueda

	Revista	**Cadena**	**N° Artículos**
1ra. Cadena de búsqueda	Scopus	TITLE-ABS-KEY ("web scraping" AND "recommender system" AND ("websites" OR "sports shoes") AND "Decision making" AND "Crawling") AND (LIMIT-TO (PUBYEAR, 2019) OR LIMIT-TO (PUBYEAR , 2023))	**1**
	IEEE Xplore	("All Metadata":"web scraping") AND ("All Metadata":"recommender system") AND ("All Metadata":"websites") OR ("All Metadata":"sports shoes") AND (("All Metadata":"Decision making") OR ("All Metadata":"Crawling")) Filters Applied: 2019 – 2023	**0**
	ACM Digital Library	[All: "web scraping"] AND [All: "recommender system"] AND [[All: "websites"] OR [All: "sports shoes"]] AND [All: "Decision making"] AND [All: "Web Crawling"]AND [E-Publication Date:(01/01/2019 TO 12/31/2023)]	**1**
	Willey	"web scraping" AND "recommender system" AND ("websites" OR "sports shoes") AND "Decision making" AND "Crawling" Applied Filters: 2019-2023	**0**
	Springer	"web scraping" and "recommender system" and ("websites" or "sports shoes") and "Decision making" and "Crawling" within 2019 - 2023	**0**
	MDPI	"web scraping" and "recommender system" and ("websites" or "sports shoes") and "Decision making" and "Crawling" within 2019 - 2023	**0**
	Computational Comnunication Research	"web scraping" and "recommender system" and ("websites" or "sports shoes") and "Decision making" and "Crawling" Applied Filters: 2019 - 2023	**0**
2da. Cadena de búsqueda	Scopus	TITLE-ABS-KEY ("web scraping" AND "recommender system") AND (LIMIT-TO (PUBYEAR, 2019) OR LIMIT-TO (PUBYEAR , 2023))	**9**
	IEEE Xplore	("All Metadata":"web scraping") AND ("All Metadata":"recommender system") Filters Applied: 2019 - 2023	**3**

	ACM Digital Library	[All: "web scraping"] AND [All: "recommender system"] AND [E-Publication Date:(01/01/2019 TO 12/31/2023)]	**2**
	Willey	"web scraping" AND "recommender system" Applied Filters: 2019-2023	**6**
	Springer	"web scraping" and "recommender system" within 2019 - 2023	**4**
	MDPI	“web scraping” and “recommender system” Applied Filters: 2019 - 2023	**2**
	Computational Comnunication Research	“web scraping” and “recommender system” Applied Filters: 2019 - 2023	**1**
		TOTAL	**29**

El contexto de establecer una revisión literaria radica en la necesidad de explorar y comprender el desarrollo de sistemas de recomendación junto con el uso de la técnica de Web Scraping. Este análisis tiene como objetivo principal adquirir conocimientos sobre los diversos tipos de sistemas de recomendación y las herramientas específicas que son esenciales para la implementación efectiva de Web Scraping junto la implementación de un modelo de recomendación.

En secuencia, se llevó a cabo su búsqueda en 7 bases de datos científicos, generando un conjunto de 29 artículos científicos que están directamente vinculados a las palabras clave seleccionadas. Por lo tanto, es necesario excluir artículos para obtener información que acote y sea fidedigna a este proyecto, por ello, en la Tabla 2 se presentan los criterios detallados de inclusión y exclusión que fueron aplicados para la selección de los artículos que se consideran relevantes. Estos criterios sirven como guía fundamental para garantizar la calidad y pertinencia de la información recopilada en este estudio.

Tabla 2. Criterios de inclusión y exclusión

Inclusión	Artículos que implementen modelos de sistemas de recomendación y el motivo de uso del modelo.
	Investigaciones que detallen el proceso específico de Web Scraping para la recopilación de datos.
	Artículos que vinculen características de páginas web a un Sistema de Recomendación.
	Artículos que sean de Ciencia Computacional e Ingeniería.
	Artículos redactados en idiomas distintos al inglés o español.

Exclusión	Artículos que no conjuguen la implementación de Web Scraping para un Sistema de Recomendación.
	Artículos que no se enfoquen a la recomendación de procesos para la toma de decisión en un hardware o servidores.
	Artículos que no estén relacionados con la recomendación de contenidos.

Al haber obtenido un total de 29 artículos se descartaron 4 artículos porque no estaban establecidos entre los idiomas en inglés y español, se descartaron 10 articulo dado que no cumplían de forma conjunta la implementación de Web Scraping para un motor de sugerencias, se descartaron 2 artículos ya que, se enfocan en establecer recomendaciones de procesos para un Sistema Operativo (OS) o un servidor y se descartaron 8 artículos dado que están implementando recomendaciones pero no mediante contenidos, sino mediante historial de usuarios, quedando un total de 5 artículos científicos.

En su artículo científico, Herrera Rivera Jerson Erick [6] diseña e implementa una arquitectura de Sistema de Recomendación basado en contenido con el objetivo de ofrecer sugerencias eficientes y objetivas a los estudiantes, sobre qué Cursos en Línea Masivos y Abiertos (MOOC's) podrían matricularse en las plataformas educativas Udemy junto a edX los usuarios, según sus preferencias. En este sentido, el autor desarrolla Web Scraping para la extracción de información relevante como: el contenido del curso, la duración, la metodología junto a otras características importantes para los estudiantes e implementar un sistema de recomendación, el cual, ha creado gran satisfacción en los estudiantes con los resultados obtenidos. Además, se destaca la utilidad de herramientas de consulta automática como el web crawler y web scraping, que han servido para visitar y recolectar información automáticamente, reduciendo significativamente el tiempo de consulta y procesamiento de datos de cada plataforma

El problema que aborda el autor en esta investigación es la dificultad que tienen los estudiantes universitarios, especialmente aquellos que estudian temas de ciencia y tecnología, para encontrar cursos en línea que se ajusten a sus intereses y necesidades. Los estudiantes tienen que navegar entre cientos de cursos de diferentes plataformas y encontrar algún curso que se ajuste a sus intereses, lo que el autor asegura ser tedioso

y que consume mucho tiempo. Por lo tanto, se decidió desarrollar un sistema de recomendación de MOOC's basado en contenido que utilizara técnicas de web crawling y web scraping para analizar el contenido de los cursos y sugerir los más relevantes para cada estudiante.

En consecuencia, el autor establece enlaces de inicio con el uso de Web Crawler para cada plataforma. A partir de los cuales, mediante una búsqueda recursiva, todos los cursos que son encontrados se almacenan en una lista para su posterior procesamiento con el uso de la herramienta denominado Selenium. Después, realiza un recorrido iterativo por cada enlace almacenado en la lista, aplicando Web Scraping en cada página Web correspondiente al curso. Esto hizo el autor con el objetivo de extraer información relevante al contenido, identificando y guardando las etiquetas deseadas de la página Web. Posteriormente, la información capturada se almacena mediante un parser, en el que según su implementación se desarrolla mediante la herramienta Scrapy.

A su vez, el autor utiliza el modelo de factorización de matrices en la implementación de su sistema de recomendación para descomponer una matriz de interacciones entre usuarios y elementos en dos matrices más pequeñas. Donde, una de estas matrices corresponde a los usuarios y la otra a los elementos. El cual, con el uso de algoritmos de filtrado colaborativo, el autor aprovecha la información de las interacciones entre usuarios y elementos para encontrar similitudes entre ellos.

De esta forma, el autor pudo predecir las tendencias de cursos y las posibles valoraciones por parte de cada estudiante.

De acuerdo con D. Ralph, Y. Li, G. Wills y N. Gree [7], en su artículo solventan un problema sobre lo nuevos usuarios en grandes conjuntos de datos, donde hay pocos datos etiquetados disponibles para la mayoría de los ítems y no se conocen etiquetas para el nuevo usuario. El cual, presentan una técnica de recomendación llamada Transitive Semantic Relationships (TSR) basándose en la similitud semántica y relaciones entre elementos tecnológicos para ofrecer recomendaciones personalizadas y relevantes al usuario.

Por lo tanto, los autores utilizaron la técnica de Web Scraping para extraer información de texto de las películas de la base de datos de IMDb, para así, obtener información de texto necesario, donde aplicaron el uso de herramientas denominados Beautiful Soup y Scrapy para extraer información de sitios web.

De acuerdo con los datos obtenidos, los autores al utilizaron la técnica TSR comparando entre modelos de recomendación como: filtrado colaborativo y basado en contenido. El cual, en conjunción a dicha técnica seleccionada, implementa también sistema de recomendación agregando interacciones de usuario para encontrar usuarios similares y recomendar los ítems que les gustaron.

Por lo tanto, los autores determinan una comparación de algoritmos para implementar el modelo TSR, mediante métricas referente a la precisión que establezca una mayor exactitud. Por lo tanto, los autores determinaron el uso de las siguientes métricas *HR @10* (Hit Rate @10), *HR @5* (Hit Rate @5), *HR @1* (Hit Rate @1) se refieren a la precisión en la recomendación de elementos en una lista ordenada de acuerdo con su puntaje de relevancia. Estas métricas evalúan cuántos de los elementos recomendados se encuentran entre los 10, 5 o 1 elementos más relevantes.

Donde, **HR @10** mide cuántos de los elementos recomendados se encuentran entre los 10 elementos más relevantes para el usuario. Es decir, evalúa la precisión de las recomendaciones dentro de los 10 principales resultados. También, **HR @5** es similar a HR @10, pero se enfoca en los 5 elementos más relevantes. Mide cuántos de los elementos recomendados están dentro de estos 5 primeros. A continuación, **HR @1** evalúa si el elemento más relevante recomendado se encuentra en la posición número 1. Es una métrica más estricta que se centra en la recomendación más importante.

De esta forma, los autores concluyeron que, la técnica TSR-a es efectiva en conjuntos de datos con muy pocas etiquetas y en situaciones de nuevos usuarios, dado a que se logró una precisión de 0.754 para HR@10, 0.509 para HR@5 y 0.145 para HR@1. Además, se encontró que la inclusión de evidencia de apoyo en forma de rutas adicionales a un objetivo puede tener un impacto positivo en el rendimiento del modelo, pero que el peso utilizado puede tener un gran impacto en el rendimiento. También se encontró que la inclusión de rutas adicionales en la puntuación puede tener

un efecto negativo si las etiquetas son extremadamente dispersas y no están concentradas.

En base al artículo de Nilesh, Kumari, Madhu, Hazarika, Pritom, Raman y Vishal [8] describen la elaboración de un sistema de recomendación de recetas de cocina india implementando Web Scraping para el etiquetado de las recomendaciones. El cual, proporcionan a los usuarios recomendación personalizadas en sus preferencias y los ingredientes utilizados en las recetas.

El propósito fundamental de los autores radica en suministrar a los usuarios recomendaciones de recetas de cocina india inexploradas, donde, toman en consideración sus preferencias personales y los ingredientes que han sido de su agrado en recetas previamente seleccionadas. En base a las preferencias consideradas por los usuarios, los autores diseñaron un sistema utilizando técnicas de aprendizaje automático, como el modelo de Procesamiento del Lenguaje Natural denominado por los autores como “bolsa de palabras” y también el modelo por similitud coseno, con el fin de facilitar estas recomendaciones.

También, los autores determinaron la selección de fuentes de datos en línea que contenían recetas de cocina india. Estas fuentes proporcionaban información detallada sobre ingredientes, pasos de preparación y otros detalles relevantes sobre las recetas. Después, utilizaron herramientas y tecnologías de Web Scraping para extraer datos de estas fuentes. En particular, se empleó el framework Scrapy 1.5.1, una herramienta especializada en la extracción de datos de sitios web.

En consecuencia, los autores diseñaron un código personalizado en Scrapy para llevar a cabo el rastreo de las páginas web de las fuentes de datos. Este código permitió la extracción de información específica de las recetas, asegurando su almacenamiento de forma estructurada. Esta fase de extracción involucró la captura de datos como los nombres de las recetas, los ingredientes, los pasos de preparación, las imágenes y otros atributos relevantes.

Tras la extracción de datos los autores procedieron con una etapa de limpieza y procesamiento de la información, con el propósito de garantizar la coherencia y relevancia de los datos. De esta forma, se llevaron a cabo acciones como la eliminación

de números, signos de puntuación y otros elementos no esenciales, así mismo, los autores lograron el almacenamiento de los datos estructurados en una base de datos para su utilización posterior en el sistema de recomendación.

Después de obtener los datos estructurados, los autores implementaron un sistema de recomendación basado en contenido, ya que recomienda recetas de cocina india en función de los ingredientes utilizados en las recetas que al usuario le han gustado previamente. Esto significa que el sistema sugiere recetas que comparten similitudes en términos de ingredientes con las recetas que el usuario ha marcado como favoritas.

En síntesis, los autores concluyeron que el modelo de recomendación basado en contenido junto la implementación de técnicas de procesamiento de lenguaje natural y aprendizaje automático para analizar ingredientes y características de recetas, generan recomendaciones precisas. También, el proyecto resalta el potencial de los modelos de recomendación basados en contenido en el ámbito culinario.

Según M. Nathalie, D. Wilhelm, R. Yesim y H. Thomas [9], investigaron la viabilidad de combinar Web crawling, Web Scraping y resumen automático de Procesamiento de Lenguaje Natural (NLP). Sobre todo, los autores desarrollaron un prototipo que pueda proporcionar recomendaciones sobre tecnología de manera eficiente y efectiva.

A continuación, los autores inspeccionaron la implementación de extracción de características con Web Scraping, utilizando herramientas sistemáticas, tal como el Framework (Marco de Trabajo) Scrapy desarrollado en Python, el cual, permitió a los autores extraer grandes cantidades de datos de manera eficiente y escalable, automatizando así la extracción de información de los sitios Web seleccionados. Por otro lado, la herramienta Mechanical Soup, siendo una biblioteca de Python posibilitó a los autores la interacción programática con los sitios Web y la extracción de información mediante técnicas de Web Scraping.

Del mismo modo, los autores utilizaron PySpider, siendo otro Framework de Web Scraping desarrollado en Python, donde lo utilizaron para extraer información de manera eficiente y sistemática, automatizando la extracción y procesamiento de los datos de interés. El cual, también los autores usaron la herramienta Beautiful Soup, una biblioteca desarrollada en Python, para analizar y obtener datos específicos de los

sitios Web seleccionados, el cual, permite aplicar técnicas programáticas de Web Scraping.

Posteriormente, los autores llevaron a cabo la transformación de los datos extraídos realizaron una limpieza de los datos con el fin de eliminar cualquier ruido o información innecesaria. Después, los autores estructuraron adecuadamente los datos para su posterior procesamiento, además, utilizaron la técnica de Procesamiento de Lenguaje Natural (NLP) para analizar y extraer información relevante de los textos, el cual, como resultado de estos procesos, lograron construir un sistema eficiente que ayuda a los investigadores a tomar decisiones informadas sobre a qué revistas enviar sus manuscritos. Esto se basa en la experiencia y publicaciones anteriores de dichas revistas, lo que garantiza una selección más precisa y acertada.

No obstante, los autores concluyeron en que, estos sistemas de recomendación pueden ser útiles en situaciones en las que los datos estructurados no están disponibles o no son suficientes para generar recomendaciones precisas.

Además, los autores Loecherbach, Felicia y Trilling [10] desarrollaron un framework denominado 3bij3 en el que implementa sistemas de recomendación basado en contenidos y de filtrado colaborativo, aplicando la técnica de Web Scraping en la selección y consumo de noticias, para comparar diferentes tipos de personalización en la selección y consumo de noticias. Esta técnica permitió a los autores tener un flujo constante de información, lo cual fue fundamental para ofrecer recomendaciones precisas y relevantes a los usuarios.

El problema que motivó a los autores implementar diferentes sistemas de recomendación en la aplicación 3bij3, fue la necesidad de investigar los efectos de los sistemas de recomendación en la selección y consumo de noticias. Dado que los sistemas de recomendación son cada vez más prevalentes en las plataformas de noticias en línea, era importante comprender cómo estos sistemas afectan el comportamiento de los usuarios en términos de selección y consumo de noticias.

Además, los autores utilizaron la herramienta Inca para realizar Web Scraping que fue desarrollada por los mismos. Esta herramienta les permitió extraer contenido de diferentes fuentes Web datos de cualquier fuente de noticias, siempre y cuando los

datos raspados se almacenen en una base de datos *Elasticsearch***. Elasticsearch** es un motor de búsqueda y análisis de datos distribuido y de código abierto que se utiliza para buscar, analizar y visualizar grandes cantidades de datos en tiempo real elaborado por ellos mismo. Por tanto, los autores accedieron a los feeds de Really Simple Syndication (RSS), los cuales se utilizan para distribuir contenido Web actualizado de manera regular. Luego, extrajeron información de los sitios Web de noticias, incluyendo el título, teaser, texto e imágenes. Posteriormente, estos datos fueron procesados y estructurados en los algoritmos de recomendación. Donde, los autores utilizaron diferentes sistemas de recomendación para seleccionar y presentar noticias a los usuarios como basado en contenidos y de filtrado colaborativo, por lo tanto, se limitaron a dar uso de sistemas de recomendación basados en contenido debido al tamaño de la muestra y el período de prueba.

En consecuencia, desarrollaron un sistema de recomendación basado en contenidos de artículos sobre noticias. El cual, obtiene las características y contenidos de los artículos, para que el sistema compare dichos atributos con el perfil de interés del usuario. Permitiendo a los autores proporcionar recomendaciones personalizadas y adaptadas a los intereses individuales de cada usuario. De esta manera, garantizaron que las recomendaciones sean relevantes y se ajusten a las preferencias de cada usuario.

Llegando a la información impartida en los diferentes artículos, se establece la comparación de herramientas sobre Web Scraping en la **Tabla 3**, y de modelos de Sistemas de Recomendación en la **Tabla 4.**

Tabla 3. Comparación de herramientas Web Scraping de diferentes artículos

Referencia del artículo	Descripción de la cita	Herramientas Web Scraping	Razón de herramienta seleccionado
[6]	Sistema de Recomendación basado en contenido para cursos MOOCs en Udemy y edX.	Selenium, Scrapy	Se eligió para extraer información relevante de los cursos y permitir el sistema de recomendación.
[7]	Recomendación de películas basada en Transitive Semantic Relationships (TSR) con Web Scraping.	Beautiful Soup, Scrapy	Se utilizó para extraer información de películas de IMDb, esencial para el modelo TSR basado en contenido.
[8]	Sistema de recomendación de recetas de cocina india utilizando Web Scraping.	Scrapy	Se empleó para extraer datos de recetas de cocina india de fuentes en línea, crucial para el sistema de recomendación.
[9]	Sistema de recomendación de revistas científicas con Web Crawling, Web Scraping y NLP.	Scrapy, Beautiful Soup	Fue utilizado para extraer información relevante sobre revistas científicas de fuentes web.
[10]	Sistema de recomendación de noticias con Web Scraping y Elasticsearch.	Inca	Se utilizó para obtener datos de noticias de fuentes web en tiempo real, esencial para las recomendaciones. Fue elaborado por los autores del artículo.

En base a las herramientas implementadas en los diferentes artículos para el desarrollo de Web Scraping en la **Tabla 3**, muestra que las herramientas de Web Scraping más utilizadas son Scrapy, Selenium y Beautiful Soup. Dado que, Scrapy es un framework de Web Scraping que ofrece una serie de ventajas, como la facilidad de uso, la flexibilidad y la escalabilidad, Selenium es una herramienta de automatización de navegadores que permite controlar un navegador web y realizar acciones en él, como navegar por páginas, hacer clic en enlaces y rellenar formularios y Beautiful Soup es una biblioteca de Python que permite analizar documentos HTML y XML.

Tabla 4. Comparación de modelos de recomendación de diferentes artículos

Referencia del artículo	Descripción de la cita	Modelo de recomendación implementado	Razón de elección del modelo de recomendación
[6]	Sugerir cursos MOOCs en Udemy y edX basados en preferencias de los estudiantes.	Basado en Contenido y filtrado colaborativo mediante factorización de matrices	Se eligió el enfoque basado en contenido, ya que se recomiendan cursos similares en función de las características de los cursos y las preferencias del usuario.
[7]	Recomendación de películas basada en similitud semántica y relaciones tecnológicas.	Transitive Semantic Relationships (TSR)	Se eligió TSR basado en contenido para ofrecer recomendaciones personalizadas y relevantes basadas en la similitud semántica y las relaciones entre elementos tecnológicos.
[8]	Recomendación de recetas de cocina india basada en ingredientes y preferencias del usuario.	Basado en Contenido	El enfoque basado en contenido se utilizó para recomendar recetas basadas en los ingredientes que el usuario ha disfrutado previamente.
[9]	Recomendación de revistas científicas basada en características de las publicaciones y preferencias de los autores.	Basado en Contenido	El sistema de recomendación basado en contenido se eligió para comparar revistas científicas en función de las características de sus publicaciones y las preferencias de los autores.
[10]	Recomendación de noticias basada en la relevancia de los artículos y las preferencias del usuario.	Basado en Contenido	Se optó por el enfoque basado en contenido para ofrecer recomendaciones de noticias relevantes basadas en la relevancia de los artículos y las preferencias del usuario.

En base a los artículos revisados en la **Tabla 4** sobre modelos de recomendación implementados, se observa que el modelo de filtrado basado en contenido es extensamente adoptado. Dado que, este modelo analiza el contenido en diversos contextos y sugiere elementos relevantes para cada usuario según sus intereses y necesidades específicas.

1.3 Fundamentación teórica

En esta sección, se presenta el marco conceptual del estudio. Donde, explican los conceptos clave, se resumen las teorías y enfoques relevantes, el cual, justifica su uso en el estudio.

1.3.1 Motor de Sugerencias

Un sistema de recomendación es un software que utiliza técnicas de inteligencia artificial y aprendizaje automático para analizar los datos de un usuario y ofrecer recomendaciones personalizadas de productos, servicios o contenido de interés. Estos sistemas se utilizan en diversas plataformas, como comercio electrónico, streaming de video y música, y redes sociales, con el objetivo de mejorar la experiencia del usuario y aumentar su satisfacción y fidelidad [11].

Los sistemas de recomendación tienen como objetivo simplificar la búsqueda y selección de información relevante, evitando la saturación de opciones y facilitando la toma de decisión. Por tanto, son herramientas útiles para abordar diversos problemas, como la recomendación de nuevos elementos en sistemas colaborativos o la personalización de la experiencia del usuario según sus preferencias y comportamientos. Por lo tanto, existen diferentes métodos en los sistemas de recomendación que varían en la información necesaria y los procesos involucrados. Estos métodos abordan el problema de la sobrecarga de información al proporcionar a los usuarios información sintetizada para facilitar la toma de decisión. Pueden basarse en el análisis del comportamiento previo del usuario, en el filtrado colaborativo con usuarios similares o en el análisis del contenido de los elementos recomendables con relación a las preferencias del usuario [10].

Por otra parte, García Escudero y Luis Ángel Berrío Galindo [12] explican que los sistemas de recomendación desempeñan un papel crucial en la personalización de experiencias para los usuarios, facilitando la sugerencia de elementos relevantes. Entre los enfoques más comúnmente utilizados para desarrollar motores de recomendación, se encuentran *el filtrado colaborativo, el filtrado basado en contenido, los sistemas híbridos, el aprendizaje automático y los sistemas basados en conocimiento*.

El **filtrado colaborativo**, como primer método, se fundamenta en el análisis de las relaciones entre usuarios y los elementos que han sido consumidos positivamente en el pasado. Esta estrategia busca identificar patrones de preferencia compartidos entre usuarios con perfiles similares, generando así recomendaciones más precisas y personalizadas.

Por otro lado, el **filtrado basado en contenido** compara características específicas de los elementos preferidos por el usuario con otros de naturaleza similar, centrándose en aspectos inherentes a los elementos en lugar de en las relaciones entre usuarios. Este enfoque se caracteriza por proporcionar recomendaciones basadas en las propiedades y atributos intrínsecos de los elementos, ofreciendo una perspectiva más detallada y específica. En el que se sustentan en las similitudes entre los perfiles creados para los elementos y los usuarios. La clave radica en identificar productos similares a los ya consumidos por el usuario en análisis.

Para el análisis de elementos con contenido textual en este modelo, se emplea una lista de palabras clave derivada de la descripción del elemento o mediante la identificación de las más frecuentes en el propio elemento. La métrica comúnmente utilizada para evaluar la similitud entre dos documentos es el coeficiente de Dice, expresado como:

$$PalabrasClave(Bi, Bj) = 2 \times \frac{|\text{palabras clave}(Bi) \cap \text{palabras clave}(Bj)|}{|\text{palabras clave}(Bi)| + |\text{palabras clave}(Bj)|} \quad (1)$$

Sin embargo, este enfoque otorga igual importancia a todas las palabras, sin considerar su relevancia. Por ello, se recurre a la métrica TF-IDF (term frequency-inverse document frequency). El espacio euclidiano se establece con dimensiones igual al número de términos, y la coordenada de cada documento es el producto de TF y IDF.

$$\text{TF}(i, j) = \frac{\text{frecuencia}(i, j)}{\max\big(\text{frec}(j)\big)} \quad (2)$$

$$\text{IDF}(i) = \log\left(\frac{N}{n(i)}\right) \tag{3}$$

El peso combinado TF-IDF se determina como:

$$\text{TF-IDF}(i,j) = \text{TF}(i,j) \times \text{IDF}(i) \tag{4}$$

La similaridad entre documentos se mide comúnmente con la similitud del coseno, donde la matriz de TF-IDF representa el espacio de palabras y documentos. Para dos documentos:

$$\text{similaridad}(A,B) = \frac{A \cdot B}{||A|| \times ||B||} \tag{5}$$

Con estas similitudes y las valoraciones de los usuarios, se pueden realizar predicciones. Utilizando técnicas como k-nearest neighbours, se identifican objetos más cercanos a uno dado. Luego, se analiza cuántos de estos objetos han gustado al usuario y se realiza una recomendación en función de esa proporción. Si las valoraciones son numéricas, se puede aplicar una media ponderada según la similitud de los objetos al elemento en cuestión para obtener una predicción.

También, una aproximación más integral se logra mediante los **sistemas híbridos**, que combinan tanto el filtrado colaborativo como el basado en contenido. Esta fusión de enfoques busca superar las limitaciones individuales de cada método, aprovechando sus fortalezas respectivas para proporcionar recomendaciones más robustas y precisas. La sinergia resultante permite mejorar significativamente la calidad y relevancia de las sugerencias ofrecidas a los usuarios.

Adicionalmente, el **aprendizaje automático** se destaca por su capacidad para analizar extensos conjuntos de datos y descubrir patrones de comportamiento, mejorando así la capacidad predictiva de los motores de recomendación.

Este enfoque implica el uso de algoritmos avanzados que pueden adaptarse y evolucionar a medida que se recopilan más datos, permitiendo una mejora continua en la precisión y relevancia de las recomendaciones.

Por su parte, los sistemas **basados en conocimiento** emplean reglas específicas relacionadas con los elementos a recomendar, ofreciendo una perspectiva más estructurada y fundamentada en información previa. Estos sistemas utilizan conocimientos predefinidos sobre los elementos y las preferencias del usuario para generar recomendaciones, lo que puede ser especialmente útil en situaciones donde la disponibilidad de datos es limitada o la interpretación de patrones es compleja.

En base a estos modelos de recomendación, se considera que, para implementarlo eficazmente, es esencial recopilar una variedad de datos sobre usuarios y elementos. Estos datos incluyen información de interacción, que revela comportamientos y preferencias; datos demográficos, como edad y ubicación; datos contextuales, que consideran factores temporales como la hora del día o el clima; datos sociales, que exploran conexiones en redes sociales para técnicas de filtrado colaborativo; y datos de contenido, que detallan atributos específicos de los elementos. La combinación y análisis de estos datos constituyen la base para la generación de recomendaciones personalizadas, mejorando así la experiencia del usuario.

1.3.2 Web Scraping

Web Scraping es un método automatizado para recopilar y extraer información disponible en la World Wide Web (WWW). Dado a que, su popularidad ha crecido debido al aumento significativo de datos en línea, esta técnica ofrece amplias posibilidades para obtener grandes volúmenes de información de diversas fuentes. Sin embargo, puede ser complejo debido a las medidas implementadas por algunos sitios Web para dificultar el acceso, como protección de datos, Prueba de Turing Completamente Automatizada y Pública para Diferenciar entre Computadoras y Humanos (CAPTCHA) o bloqueo de bots automatizados. Para superar estas limitaciones y obtener los datos deseados, a menudo es necesario utilizar técnicas avanzadas o herramientas como proxys o sistemas antibloqueo [13].

Con esa finalidad, Web Scraping es una técnica automatizada que se emplea para recopilar datos de páginas Web, a diferencia de la recolección manual, que resulta tediosa y poco práctica cuando se trata de grandes volúmenes de información. Web Scraping permite automatizar el proceso, ahorrando tiempo, energía y recursos, el cual, funciona conectándose al servidor de un sitio Web y así extraer la información necesaria para satisfacer una determinada necesidad o contexto. Para llevar a cabo el Web Scraping, es pertinente desarrollar un programa en algún lenguaje de programación que identifica y sigue patrones en los datos de la página Web. Este programa se encarga de la extracción y procesamiento adecuado de los datos para almacenarlos en una estructura coherente y significativa [14].

De cierto modo, la autora Nicole Luise Vicente Stenhouse [15] asegura que existe una amplia variedad de herramientas denominados "scrapers", frameworks e Interfaz de Programación de Aplicaciones (API), tanto comerciales como de código abierto, que permiten su implementación. Una de las técnicas populares es utilizar la estructura del modelo Modelo de Objetos del Documento (DOM) en las páginas HyperText Markup Language (HTML), por lo tanto, es común desarrollar Web Scrapers utilizando lenguajes de programación populares como Java, Python o Perl, junto con librerías externas que facilitan el soporte para funciones relacionadas con el protocolo HTTP, la autenticación, el control de historial, las cookies y los certificados Secure Sockets Layer (SSL). Estas librerías también proporcionan funciones de parseo que permiten extraer datos relevantes utilizando técnicas como XPath o la estructura semántica del HTML. Sin embargo, estas librerías tienen algunas limitaciones, como la necesidad de importar y configurar correctamente, así como la adición de un componente adicional para realizar el parseo y la extracción de datos. Estas limitaciones pueden superarse utilizando frameworks específicos para el Web Scraping, como Scrapy, que es ampliamente utilizado y escrito en Python. HabScraper, por ejemplo, se basa en Scrapy como su framework principal, lo que simplifica tareas comunes como la gestión de solicitudes HTTP, el parseo de HTML y la extracción de datos estructurados. Al aprovechar frameworks como Scrapy, los desarrolladores pueden mejorar la eficiencia y la flexibilidad en la extracción automatizada de datos, agilizando el proceso de desarrollo de Web Scrapers.

Por ello, la autora destaca que la extracción de datos de páginas web, es un proceso esencial en diversas aplicaciones, ya que, se lleva a cabo mediante un conjunto de herramientas y técnicas especializadas, el cual, hay dos componentes fundamentales que en este contexto son el *Web Crawler* y el *Extractor de Datos*. El **Web Crawler**, también conocido como araña web, se encarga de seguir enlaces, recopilar URLs y descargar páginas concurrentemente mediante algoritmos recursivos. Este componente sigue reglas específicas y añade nuevos enlaces a una cola para su procesamiento posterior. Por otro lado, el **Extractor de Datos** analiza el contenido de las páginas web, utilizando selectores como XPath o Selectores CSS para identificar patrones comunes en la estructura HTML y extraer información deseada.

La autora adhiere, que es esencial abordar con precisión las situaciones donde las páginas web cargan contenido de manera dinámica mediante interacciones del usuario, dado que los Web Scrapers deben no solo analizar las solicitudes estáticas, sino también manejar de manera hábil las interacciones dinámicas. En este contexto, la capacidad del scraper para interpretar peticiones en tiempo real, emular la actividad del usuario, y procesar actualizaciones en la interfaz se vuelve crucial. Esto implica la manipulación efectiva del DOM, la gestión de solicitudes asíncronas, la identificación de patrones de comportamiento específicos, así como el manejo de sesiones y cookies para replicar autenticación y adaptarse a cambios en la estructura de la página en tiempo real. La eficacia de un Web Scraper en entornos dinámicos radica en su capacidad para emular con precisión el comportamiento humano y adaptarse a cambios en la dinámica de carga de contenido y estructura de la página.

Por lo tanto, la autora resume el proceso de Web Scraping en tres pasos: establecer la conexión con el sitio Web a través del protocolo HTTP, obtener y extraer el contenido relevante utilizando diferentes técnicas y librerías, y finalmente, convertir los datos extraídos a un formato adecuado para su posterior uso. Es importante tener en cuenta que al utilizar Web Scraping se deben respetar las restricciones establecidas por los sitios Web y seguir las políticas de uso ético y legal.

Como aporte adicional, también un enfoque innovador en el ámbito del Web Scraping es presentado por UzunExt, diseñado para mejorar los tiempos de respuesta mediante el uso de métodos en cadena en el proceso de extracción de datos.

Además, herramientas como Urllib2, Selenium, libcurl, Apache HttpClient, Jsoup, BeautifulSoup, scrapy, Web-Harvest son utilizadas para establecer conexiones estables y brindar soporte en el proceso de extracción, seleccionadas según la compatibilidad y funcionalidades ofrecidas [16].

1.3.3 NLP

Según los autores Torres, Mónica María Echeverri and Manjarrés-Betancur, Roberto [17] indican que NLP, según su acrónimo de "Procesamiento de Lenguaje Natural" en español, se refiere a un área interdisciplinaria dentro de la inteligencia artificial y la lingüística. Su enfoque principal radica en la interacción entre las computadoras y el lenguaje humano. El objetivo central de NLP consiste en capacitar a las computadoras para comprender, interpretar y generar lenguaje humano de manera coherente y natural.

Además, los autores aciertan que el Procesamiento de Lenguaje Natural (NLP) encuentra aplicación en diversos contextos, ampliando su utilidad en los Asistentes *Virtuales, Análisis de Sentimientos, Traducción Automática, Extracción de Información*.

Los **Asistentes Virtuales**, facilitan la interacción natural entre usuarios y sistemas informáticos, ya sea mediante lenguaje hablado o escrito, mejorando la experiencia de usuario en plataformas como chatbots y asistentes de voz.

También el **Análisis de Sentimientos**, el cual, va más allá de la mera interpretación del lenguaje, permitiendo comprender actitudes y emociones expresadas en texto. Esto resulta esencial para investigaciones de mercado, análisis de redes sociales y evaluación de comentarios de clientes.

En secuencia la **Traducción Automática**, facilita la traducción instantánea entre idiomas, siendo esencial para la comunicación global en entornos digitales, comerciales y académicos.

Para la **Extracción de Información**, según los autores por el proceso de identificar y extraer datos específicos de conjuntos no estructurados, como noticias o documentos

legales, contribuyendo a la automatización de tareas de análisis de información. El cual, conviene positivamente para el proceso del modelo de recomendación basado en contenidos.

También **Resumen Automático**, permitiendo la síntesis rápida y precisa de grandes volúmenes de texto, beneficiando la gestión eficiente de información en áreas como investigación, revisión de literatura y resumen de noticias.

De acuerdo con estas aplicaciones representan solo una fracción de las múltiples utilidades del NLP, destacando su versatilidad y su capacidad para innovar en diversos sectores, desde la atención al cliente hasta la investigación científica.

Los autores especifican el uso del proceso NLP desplegándolo mediante diversas herramientas y plataformas que posibilitan a desarrolladores y usuarios finales interactuar eficazmente con el lenguaje natural en *Plataformas de Procesamiento de Lenguaje Natural, APIs de NLP, Librerías de NLP, Desarrollo de Chatbots y Asistentes Virtuales.*

Plataformas de Procesamiento de Lenguaje Natural

Empresas líderes como Google (Dialogflow), Amazon (Lex), IBM (Watson), entre otras, ofrecen plataformas que capacitan a los desarrolladores para concebir aplicaciones y sistemas capaces de comprender y generar lenguaje natural de manera avanzada.

APIs de NLP

Numerosas empresas proporcionan interfaces de programación de aplicaciones (APIs) que posibilitan a los desarrolladores integrar funciones de procesamiento de lenguaje natural en sus propias aplicaciones y sistemas, ofreciendo flexibilidad y personalización.

Librerías de NLP

Se destacan librerías de código abierto como NLTK (Natural Language Toolkit) para Python, que proveen herramientas y recursos esenciales para trabajar con texto y lenguaje natural, fomentando la accesibilidad y el desarrollo colaborativo.

Desarrollo de Chatbots y Asistentes Virtuales

NLP desempeña un papel central en la creación de chatbots y asistentes virtuales que interactúan de manera natural con los usuarios, comprendiendo consultas y comandos de manera inteligente.

De esta manera el NLP se puede implementar de manera efectiva en la recomendación de contenidos. Mediante el análisis del contenido de artículos o publicaciones en redes sociales, NLP puede identificar el tema principal, el tono y la intención del autor. Esta información se utiliza para recomendar contenido relacionado o similar a los usuarios, mejorando la experiencia de usuario y la relevancia de las sugerencias.

Adicionalmente, NLP se aplica para analizar el comportamiento de los usuarios en línea, incluyendo búsquedas y clics. Este análisis permite determinar sus intereses y preferencias, facilitando la recomendación de contenido relevante y personalizado. La capacidad de NLP para interpretar tanto el contenido como las interacciones del usuario optimiza la precisión de las recomendaciones, contribuyendo a la satisfacción del usuario y la eficacia del sistema de recomendación.

1.3.4 Web 2.0

El autor Marino Latorre [4] indica que la Web 2.0 representa una evolución en la tecnología Web, centrándose en la interacción y colaboración entre usuarios en comunidades en línea. Esta segunda generación de tecnología Web ofrece una amplia gama de servicios como redes sociales, blogs, wikis, chat, foros, álbumes de fotografías, presentaciones en línea, entre otros. Estos servicios fomentan la colaboración y el intercambio de información de manera ágil entre los usuarios de una comunidad o red social.

Por lo tanto, el autor conlleva que la Web 2.0, como paradigma evolutivo de la World Wide Web, se distingue por su arquitectura centrada en el usuario, enfocada en la colaboración y la interactividad. Este cambio no solo implica una transición hacia plataformas que facilitan la consumición de información, sino que también enfatiza la creación y compartición activa de contenido. Aspectos técnicos clave incluyen la implementación de tecnologías como AJAX para mejorar la interactividad web,

permitiendo actualizaciones de contenido sin recargar la página, así como el uso extensivo de API para integrar servicios y datos, promoviendo la creación de aplicaciones más robustas y conectadas.

En este contexto, las redes sociales desempeñan un papel fundamental al facilitar la creación de comunidades en línea y potenciar la participación de los usuarios. La colaboración se expande a través de plataformas de colaboración en tiempo real, sistemas de gestión de contenidos colaborativos y herramientas de edición compartida. En resumen, la Web 2.0 no solo representa una evolución en la interacción con la web, sino que también impulsa avances tecnológicos subyacentes, creando experiencias de usuario más enriquecedoras, colaborativas y socialmente conectadas.

1.3.5 Marketing Digital

El Marketing Digital se enfoca en la promoción de empresas mediante el uso de medios digitales, incluyendo sitios Web, redes sociales y técnicas de Optimización para Motores de Búsqueda (SEO). Su objetivo principal es dar a conocer los productos, ofertas y descuentos de la empresa con el fin de aumentar las ventas. Para evaluar los beneficios potenciales del marketing digital en el contexto del calzado deportivo, es posible establecer un presupuesto que contemple escenarios futuros tanto con el proyecto de marketing como sin él. De esta manera, se pueden analizar y comparar los resultados esperados y medir el alcance y la visibilidad que el marketing digital podría proporcionar. El objetivo final es transformar ese impacto en resultados comerciales positivos y un mayor éxito en el mercado [18].

1.3.6 Ventas Online

El término "ventas online" se refiere al proceso de compra y venta de productos o servicios a través de Internet. Según un estudio realizado por ComScore, se observa que América Latina se encuentra en las primeras etapas de desarrollo en lo que respecta a las ventas online, pero se evidencia una clara tendencia hacia el crecimiento de las compras en línea en la región, lo cual, implica el desarrollo de una plataforma de comercio electrónico que permita a los clientes realizar compras a través de Internet. Implementando la creación de una tienda virtual en un sitio Web, la integración de un

sistema de pago en línea y la implementación de medidas de seguridad para salvaguardar la información de los clientes. Además, es crucial contar con personal capacitado para brindar atención al cliente y brindar asesoramiento tanto antes como después de la compra [21].

Las tendencias de las páginas Web, se destaca la importancia de los Modelos de Personalización en Buscadores Web (MPBW), los cuales tienen como objetivo ofrecer resultados personalizados a los usuarios, reduciendo la cantidad de documentos irrelevantes que se les muestra. Estos modelos han experimentado un crecimiento significativo en los últimos tiempos y suelen ser objeto de estudio y trabajo por parte de diversos investigadores con el fin de abordar problemas específicos. Lo que borda la necesidad de soluciones innovadoras que abarquen la diversidad de usuarios multilingües y la protección de la privacidad en las aplicaciones para la búsqueda Web personalizada. Esto implica considerar la adaptación de los resultados de búsqueda según las preferencias y características individuales de los usuarios, así como garantizar la seguridad y la confidencialidad de sus datos personales [22].

Esto implica la veracidad sobre la búsqueda productos para su compra online, y la abundancia de modelos sobre el calzado deportivo acorde al contexto del calzado deportivo. De esta forma, se puede establecer una búsqueda abundante de páginas web que abarquen la venta en línea y a su vez, implica la detección de muchos productos que para algunos consumidores es abrumador.

1.3.7 Compra de calzado en la línea deportiva

Es importante considerar diversos factores que influyen en la experiencia del usuario y su predisposición a recomendar el sitio Web. Entre estos factores destacan la demostrabilidad de los productos, la atención al cliente, la intención de volver a visitar el sitio y la confianza en el mismo. Se ha observado que la demostrabilidad de los productos es crucial en las páginas Web de calzado deportivo, ya que los usuarios desean tener una idea clara de cómo lucirán los productos antes de realizar una compra. Esto se puede lograr a través de imágenes de alta calidad, descripciones detalladas y reseñas de clientes satisfechos. La atención al cliente también juega un papel fundamental. Los usuarios valoran la disponibilidad de asistencia y soporte en caso de

tener preguntas o problemas relacionados con su compra. La rapidez y eficiencia en la respuesta a las consultas del usuario contribuyen a una experiencia positiva y aumentan la confianza en el sitio Web. La intención de volver a visitar el sitio Web es otro factor importante. Si los usuarios han tenido una experiencia satisfactoria en su compra y se han sentido cómodos navegando por el sitio, es más probable que vuelvan en el futuro para realizar nuevas compras. La facilidad de uso, la navegación intuitiva y la personalización de la experiencia pueden fomentar esta intención de volver [5].

1.3.8 Interfaz de Usuario

Para la elaboración de este proyecto es pertinente reconocer, la forma en que el usuario implementa la interacción con el motor de sugerencias, así que, es importante reconocer las herramientas necesarias para implementarlo junto a su teoría. Esto permitiría visualizar la lógica del modelo de recomendación al usuario.

En este contexto, desde el punto de Diego Palacios [21], indica que una Interfaz de Usuario (UI) es la manera en que los usuarios interactúan con sistemas informáticos. De esta forma, los elementos que componen esta interfaz van desde pantallas, páginas web, botones, menús e iconos, a su vez, sirven para permitir a los usuarios interactuar con el software o la aplicación. La interfaz de usuario tiene un objetivo primordial que es facilitar la comunicación efectiva entre el sistema y el usuario y permitirles operar el software de manera intuitiva y eficiente.

En este contexto, el autor indica que la implementación de una interfaz de usuario implica el diseño y desarrollo de los elementos visuales y funcionales que componen la interacción entre el usuario y el sistema. Esto puede incluir la creación de diseños visuales, la disposición de elementos en la pantalla, la navegación entre diferentes secciones, la respuesta a las acciones del usuario, entre otros aspectos. De este modo, la implementación de una interfaz de usuario puede realizarse utilizando diferentes tecnologías y herramientas, dependiendo del tipo de sistema o aplicación en cuestión.

Por lo tanto, en base a la exclamación del autor la implementación de una UI para el motor de sugerencias a implementar implica el desarrollo del diseño y desarrollo de

componentes visuales y funcionales que permitieran a los usuarios interactuar de manera efectiva.

En este caso, el autor aborda la implementación de una interfaz de usuario para una aplicación web, indicando principios fundamentales sobre UI, como el d*iseño centrado en el usuario*, *diseño responsivo*, *uso de componentes*, *diseño visual coherente*, *pruebas de usabilidad.*

Asimismo, el **diseño centrado en el usuario** toma en cuenta las necesidades y expectativas de los usuarios para diseñar una interfaz de usuario que fuera fácil de usar e intuitiva. También, el **diseño responsivo** implementa una interfaz de usuario que se adapta a diferentes resoluciones de pantalla, permitiendo que la aplicación fuera fácil de usar en diferentes dispositivos. De esta forma, el **uso de componentes** implica el uso de tecnologías o Frameworks para implementar una interfaz de usuario basada en componentes, lo que permite una mayor modularidad y reutilización de código. Es así, que en este contexto el **diseño visual coherente** es requerido mediante la utilización de paletas de colores, para mantener una consistencia visual en todos los elementos de la interfaz de usuario. Dicho esto, las es importante implementar **pruebas de usabilidad**, para evaluar la efectividad y eficiencia de la interfaz de usuario, y se realizaron ajustes en función de los resultados obtenidos.

A su vez, en base al artículo de los autores Verma, Ankit, Kapoor, Chavi, Sharma, Abhishek, Mishra, Biswajit [22], indican el desarrollo de una aplicación web es favorable para sistemas, por ello, los autores desarrolaron una aplicación web mediante los frameworks React para construir una interfaz de usuario receptiva y fácil de usar y mediante la lógica Back-end utilizaron Flask, debido a que es un marco de trabajo de Python que se utiliza para construir aplicaciones web. Es especialmente útil para aplicaciones web pequeñas y medianas, ya que es fácil de aprender y usar, en el contexto de los autores para implementar el modelo de procesamiento del lenguaje natural (NLP) y para desplegar la aplicación web en la red deseada.

El proceso que implementaron los autores oara crear una aplicación web con React y Flask implicaron varios pasos, desde el desarrollo del Front-end hasta la implementación del Back-end mediante el *Desarrollo del Front-end con React*,

Desarrollo del Back-end con Flask, Integración *Front-end y Back-end* y el *Despliegue en la Nube*.

Desarrollo del Front-end con React

- Utilizaron React para crear componentes de interfaz de usuario interactivos y receptivos.
- Diseñaron y desarrollaron las páginas de la aplicación web, como la página de inicio de sesión, la página de registro, el panel de control del usuario, etc.
- Implementaron la navegación entre las diferentes secciones de la aplicación utilizando React Router.
- Utilizaron tecnologías complementarias como React Strap y Bootstrap para garantizar que la aplicación sea receptiva y se vea bien en dispositivos móviles y de escritorio.

Desarrollo del Back-end con Flask

- Utilizaron Flask para implementar la lógica del Back-end de la aplicación web.
- Diseñaron y desarrollaron las API y rutas necesarias para manejar las solicitudes del cliente y realizar operaciones en la base de datos.
- Implementaron la lógica de autenticación y autorización para gestionar el acceso de los usuarios a la aplicación.
- Integraron el modelo de procesamiento del lenguaje natural (NLP) utilizando la biblioteca Scikit-learn de Python para análisis de texto.

Integración Front-end y Back-end

- Conectaron el Front-end desarrollado en React con el Back-end desarrollado en Flask para permitir la comunicación entre el cliente y el servidor.
- Definieron cómo manejar las solicitudes del cliente en el Back-end y cómo se enviarán las respuestas al Front-end.

Despliegue en la Nube

- Utiliza Flask para desplegar la aplicación web en la nube, por ejemplo, en un servicio como Heroku.

- Configura la infraestructura en la nube para alojar la aplicación web y garantizar su disponibilidad para los usuarios.

De esta forma, los autores implementaron el desarrollo del Front-end receptivo con React, la implementación del Back-end con Flask, la integración de ambos y el despliegue de la aplicación web en la nube. Implicando la influencia de implementarlo en el desarrollo del proyecto de investigación.

1.4 Objetivos

1.4.1 Objetivo general

Desarrollar un motor de sugerencias basado en Web Scraping que mejore el tiempo de toma de decisión en la compra de calzado en la línea deportiva.

1.4.2 Objetivos específicos

- Implementar un algoritmo de Web Scraping para recopilar información detallada sobre precios, modelos, tallas y valoraciones disponibles de calzado deportivo de tiendas en línea.

- Proponer un motor de sugerencias basado en Web Scraping para proporcionar recomendaciones personalizadas en la toma de decisión sobre la línea de calzado deportivo.

- Determinar el tiempo requerido por los usuarios para tomar una decisión en la compra de calzado en la línea deportiva.

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

En el presente proyecto se realizó una encuesta a personas que son conocedores sobre el calzado deportivo y la revisión bibliográfica establecida en los antecedentes investigativos con respecto al tema de investigación.

Materiales utilizados en el desarrollo del proyecto

- Laptop.
- Internet.
- Hojas.
- Herramientas de Web Scraping (Framework Scrapy, Selenium y BeautifulSoup).
- Modelo de recomendación basado en contenidos.
- Sistema de gestión de base de datos no estructurado MongoDB.

Encuesta realizada a personas conocedoras sobre la venta y compra de calzado en la línea deportiva

La finalidad de la encuesta radica en recopilar información sustancial sobre las especificaciones más relevantes del calzado deportivo. Asimismo, se busca entender la dinámica relacionada con la recomendación de este producto a clientes, particularmente por parte de individuos con experiencia en la comercialización de calzado deportivo. Por lo tanto, para llevar a cabo este proceso de recolección de datos, se implementó la encuesta en establecimientos especializados en la venta de calzado deportivo, específicamente en locales ubicados en Cevallos y en comercios de calzado en Ambato, como Juan Cajas, abarcando también locales situados en el núcleo central de la ciudad.

De acuerdo, con la elaboración de la encuesta los ítems de cada pregunta contienen opciones como escala de Likert para medir el acuerdo y desacuerdo de los vendedores, también contienen preguntas abiertas para obtener mayor detalle de la opinión de los vendedores, mediante el cuestionario reflejado en el **Anexo A**.

A continuación, antes de implementar la encuesta de la muestra calculada, es imprescindible validar cada ítem ante una muestra independiente y aleatoria para no denotar ningún sesgo hacia 11 vendedores de calzado deportivo implementado en el **Anexo B**, para que la confiabilidad de las preguntas del cuestionario sea óptima mediante implementando Alpha de Cronbach, el cual, se elabora con ítems que contienen una escala de medición, en este caso de Likert.

En consecuencia, el resultado de validación según la **Tabla 31**, es del 0.795, el cual demuestra una confiabilidad excelente según los rangos de confiabilidad de Alpha de Cronbach en la **Tabla 32**.

2.2 Métodos

2.2.1 Modalidad de la investigación

A continuación, las modalidades en la presente investigación son las siguientes:

Modalidad de campo

Esta investigación se despliega bajo la modalidad de investigación de campo, para llevar a cabo la recopilación de datos a través de encuestas directas dirigidas a individuos con experiencia en la comercialización de calzado deportivo. Estas encuestas se ejecutan en establecimientos ubicados en las ciudades de Ambato y Cevallos. De esta manera, el investigador busca obtener información integral acerca de las características más destacadas y enfatizadas por los vendedores en el ámbito del calzado deportivo. En consecuencia, mediante estos atributos identificados, se da inicia el proceso de recolección de datos, el cual, aborda de manera eficaz la problemática planteada con relación al Web Scraping y avanzando hacia el desarrollo de los objetivos específicos establecidos.

Modalidad bibliográfica-documental

La investigación será bibliográfica-documental puesto a que se basará con la realimentación de datos a través de revistas, proyectos de grado que se basan al tema investigativo, libros o artículos, el cual, será vital para la cimentación de la fundamentación teórica que estudiará temas como: Motor de Sugerencias, Web Scraping, Procesamiento del Lenguaje Natural (NLP), Web 2.0, Marketing Digital, Ventas online, Compra de calzado en la línea deportiva.

Modalidad experimental

La investigación se implementa mediante la modalidad experimental, debido a la implementación de un diseño de estudio, donde se realizan pruebas controladas de variables específicas para evaluar el rendimiento y la eficacia del Motor de Sugerencias mediante datos extraídos con Web Scraping en la toma de decisión para la compra de calzado en la línea deportiva. Dado que, de esta forma permite obtener conclusiones más precisas y fiables sobre el funcionamiento del sistema y su impacto en la toma de decisión de los usuarios en la mejora de tiempo en la decisión de compra de calzado deportivo.

2.2.2 Población y muestra

Dado que la población se estima como infinita, se procede establecer una muestra representativa mediante la fórmula de población infinita, tomando en cuenta los siguientes parámetros:

e = Error estándar.
Z = Nivel de confianza.
p = Probabilidad de éxito.
q = Probabilidad de fracaso.
n = Tamaño de la muestra.

Se establecen parámetros clave, como el error estándar (e) del 10%, nivel de confianza del (Z) 95%, y probabilidades de éxito (p) y fracaso (q) de 0.5, para garantizar resultados confiables y representativos en la investigación de campo, para así implementar el algoritmo de Web Scraping sobre características del calzado en la línea deportiva. Estos parámetros permiten determinar el tamaño óptimo de muestra (n)

necesario para obtener datos significativos y generalizables de una población estimada como infinita, asegurando la precisión y confiabilidad de los resultados.

$$n = \frac{1{,}96^2 * 0{,}5 * 0{,}5}{0{,}10^2} = 96{,}040 \approx 96 \quad (6)$$

Tabla 5. Muestra de población infinita

Población	Muestra	Porcentaje
Conocedores en calzado de línea deportiva.	96	100%
Total	96	100%

De esta forma según la **Tabla 5** se puede determinar características relevantes de páginas Web relacionadas con la venta de calzado deportivo, por lo que, se requiere una muestra proporcional de 96 personas conocedoras en el ámbito. La selección de estas personas se basará en su conocimiento y experiencia en el sector, garantizando la obtención de datos significativos y representativos para el estudio. La recopilación de información se realizará mediante encuestas diseñadas para obtener opiniones y percepciones de expertos en el campo, con el propósito de obtener una comprensión profunda de las preferencias y necesidades del mercado de calzado deportivo. La muestra obtenida permitirá recopilar datos relevantes para el análisis y desarrollo de Web Scraping enfocado en las características del calzado deportivo de la línea deportiva.

2.2.3 Recolección de información

En base a la confiabilidad demostrada, en esta sección se implementa el análisis requerido acorde a los resultados de la encuesta aplicada.

Resultado de la encuesta realizada a personas conocedoras sobre la venta y compra de calzado en la línea deportiva

Ítem 1: ¿Qué tan importante es la marca para los clientes al momento de comprar calzado deportivo?

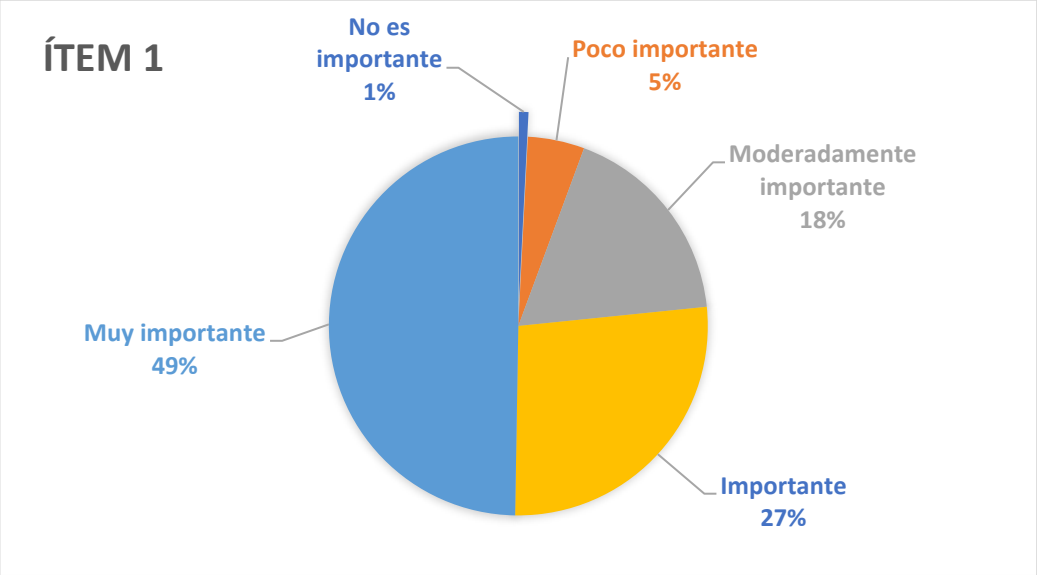


Figura 1. Importancia de la marca del calzado deportivo para el cliente según vendedores

Análisis e interpretación de resultados

De acuerdo con los resultados ilustrados en la **Figura 1**. se demuestra que:

- La importancia de la marca para los clientes, según los vendedores es altamente relevante, dado que, los clientes prefieren enfocarse por la calidad de la marca que reconocen.
- Por el otro lado la una pequeña parte, consideran que no suelen enfocarse totalmente por la marca, debido a que pocos clientes desconocen marcas.
- Esto es debido a que la mayoría de los clientes que requieren calzado se basan en las marcas que más les agrada, lo cual, escogen en la importancia que les va a brindar.

Ítem 2: ¿Qué tan importante es el precio para los clientes al momento de comprar calzado deportivo?

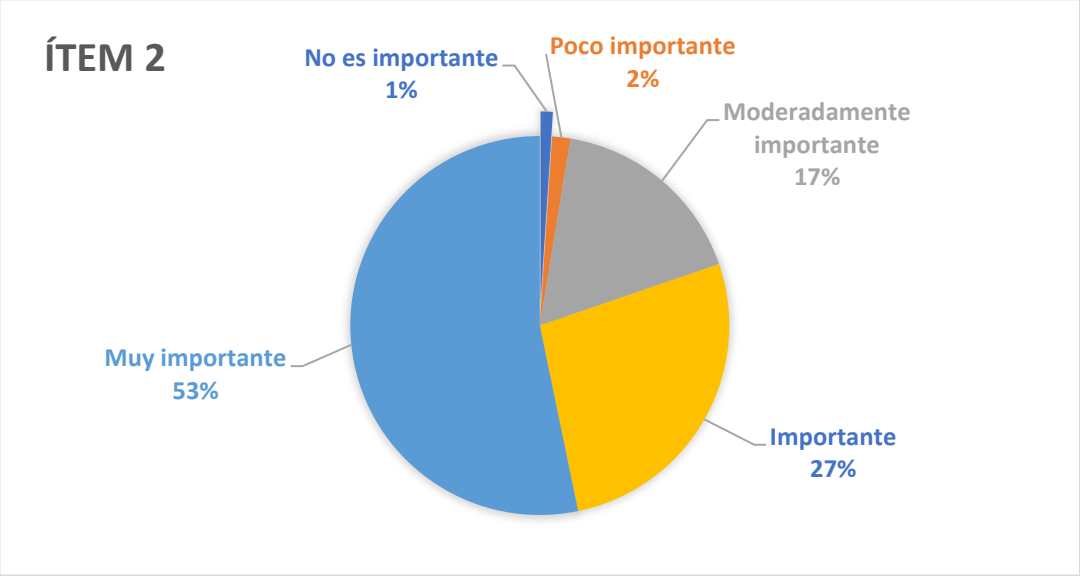


Figura 2. Importancia del precio en el calzado deportivo para el cliente según vendedores

Análisis e interpretación de resultados

De acuerdo con los resultados de la **Figura 2**, se comprende lo siguiente:

- De acuerdo con los vendedores, el precio se posiciona como una de las características más cruciales para los clientes en nuestro país. Dado que, en su mayoría, los clientes optan por calzado que se ajuste a su presupuesto, incluso llegando al punto de recurrir al regateo como una estrategia de negocio para satisfacer sus necesidades y exigencias.

- A pesar de que la mayoría de los vendedores consideran que la mayoría de sus clientes es muy importante el precio, la minoría establecen que usualmente sus clientes solicitan el producto sin mayor relevancia del precio dado a que ya están predispuestos a comprar un producto en específico.

- Se denota esta tendencia, debido a que los clientes se fijan en el precio que se ajusta a su bolsillo, dependiendo en la oferta y demanda del calzado.

Ítem 3: ¿Qué tan importante es el modelo para los clientes al momento de comprar calzado deportivo?

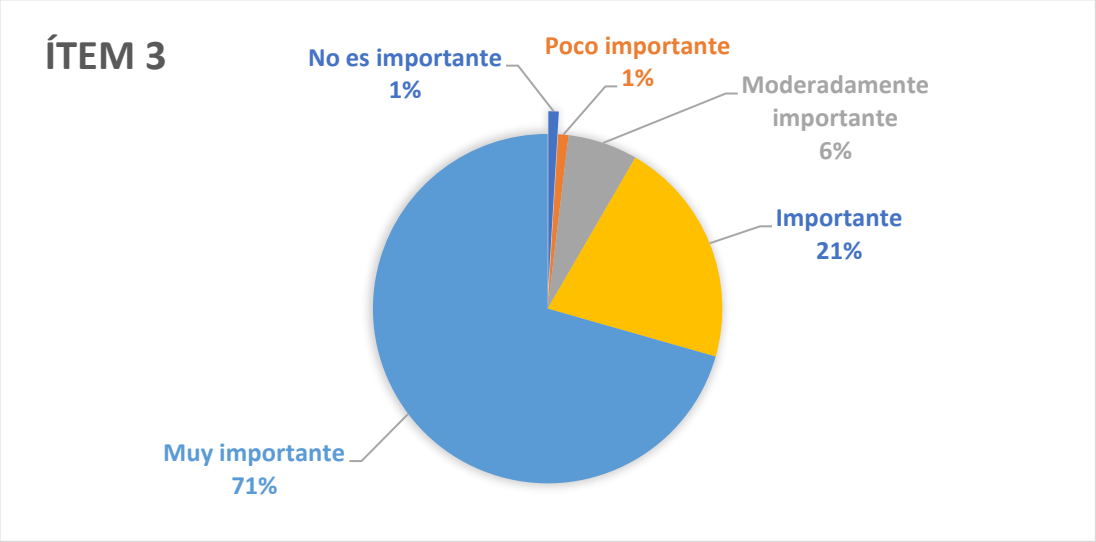


Figura 3. Importancia del modelo en el calzado deportivo para el cliente según vendedores

Análisis e interpretación de resultados

De acuerdo con los resultados de la **Figura 3**. Importancia del modelo en el calzado deportivo para el cliente según vendedores., se demuestra lo siguiente:

- Se denota mayor interés en el modelo por el cliente según los vendedores. Dado a que muchos clientes prefieren guiarse por marcas que están a la moda y son pocos clientes que no se guían a la moda.

- Los pocos clientes que resaltan la poca importancia del modelo según los vendedores suelen especificar el calzado que más se adecue al material que está elaborado el calzado deportivo en diferencia al modelo.

- Esto es notorio, ya que, los consumidores siempre buscan algo que sea actualizado, de acuerdo, a la mayoría de las personas. De esta manera eligen lo que está en tendencia abarcando los modelos que son socialmente aceptados.

Ítem 4: ¿Qué tan importante es el diseño para los clientes al momento de comprar calzado deportivo?

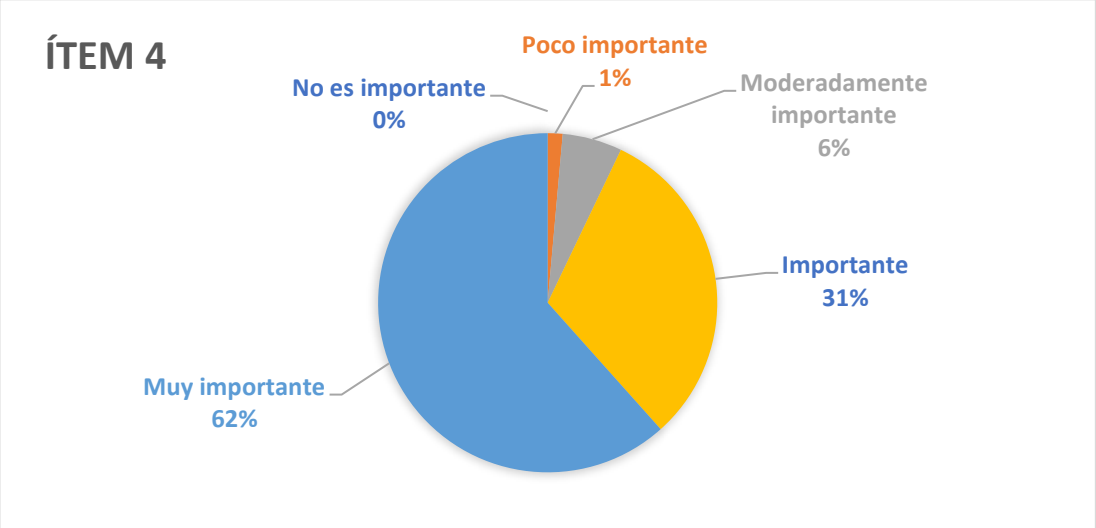


Figura 4. Importancia del diseño en el calzado deportivo para el cliente

Análisis e interpretación de resultados

De acuerdo con los resultados de la **Figura 4**, se demuestra lo siguiente:

- Los vendedores consideran que el diseño es una característica medianamente importante en comparación al resto de características, dado a que muchas veces el cliente se fija mucho más por la calidad del producto y su precio antes que el diseño del producto.

- Por otro lado, la importancia del diseño para el cliente según los vendedores no deja de ser mínima porque suelen fijarse directamente al diseño del calzado, según su color o forma.

- También, una ínfima parte de los vendedores destacan que sus clientes suelen ser más exigentes por la calidad del calzado que por su diseño.

- Por lo tanto, los consumidores siempre buscan algo diseños actualizados, que use la mayoría de las personas, lo cual se escoge lo que está en tendencia.

Ítem 5: ¿Cuál es la característica más importante que los clientes solicitan al momento de comprar calzado deportivo?

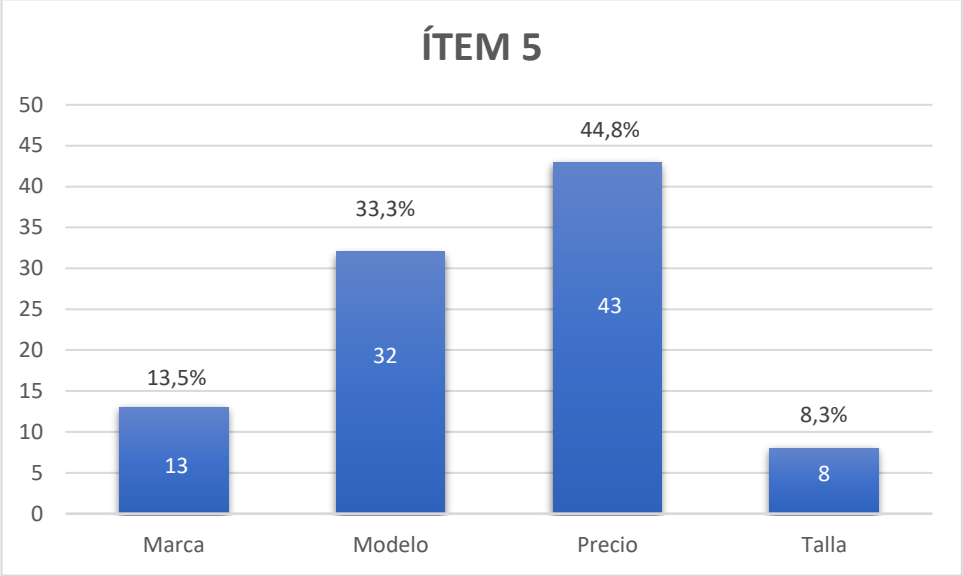


Figura 5. Características importantes para el cliente al momento de comprar el calzado deportivo

Análisis e interpretación de resultados

De acuerdo con los resultados de la **Figura 5**, se demuestra lo siguiente:

- Según los vendedores, confirman que los clientes tienden a tener mayor decisión del calzado por el precio, dado a que muchos clientes suelen ser exigentes en la parte económica basándose a la calidad del producto.

- Del mismo modo, la marca y la talla resaltan necesarios para el cliente según los clientes, pero en base a su necesidad, no es lo más interesante dado a que la economía del cliente resalta y lo llamativo del modelo del calzado es más enriquecedor.

- A pesar de estar el precio con mayor importancia, hay una alineación en cuanto al modelo con el precio, ya que, el consumidor le llama la atención el precio con más accesibilidad para poder lucir el modelo ante la moda social.

Ítem 6: ¿Cuál es la característica más importante que los proveedores destacan al momento de venderle calzado deportivo?

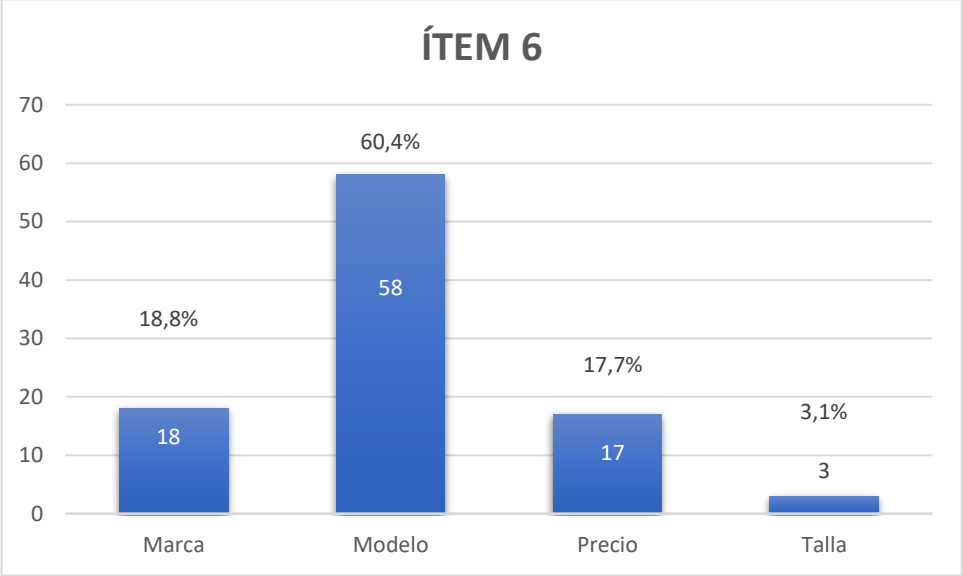


Figura 6. Características importantes para el vendedor al momento de comprar al proveedor de calzado deportivo

Análisis e interpretación de resultados

De acuerdo con los resultados de la **Figura 6**, se demuestra lo siguiente:

- En cambio, para el vendedor es más importante el modelo al momento de comprar a sus proveedores, dado a que el cliente solicita en mayor parte con el modelo en comparación con las otras características.

- Del mismo modo, la marca, el precio y la talla son elegidos por un bajo índice de vendedores, demostrando que el mejor medio de inversión en las características de calzado deportivo es mediante el modelo.

- En conclusión, esto indica que también al vender el calzado deportivo el precio, la marca y el modelo va a predominar, pero a veces el modelo puede ser predecible para que el consumidor se fije más en el tipo de calzado deseado.

Ítem 7: ¿Cuál es el color más popular de calzado deportivo, según su experiencia con sus clientes?

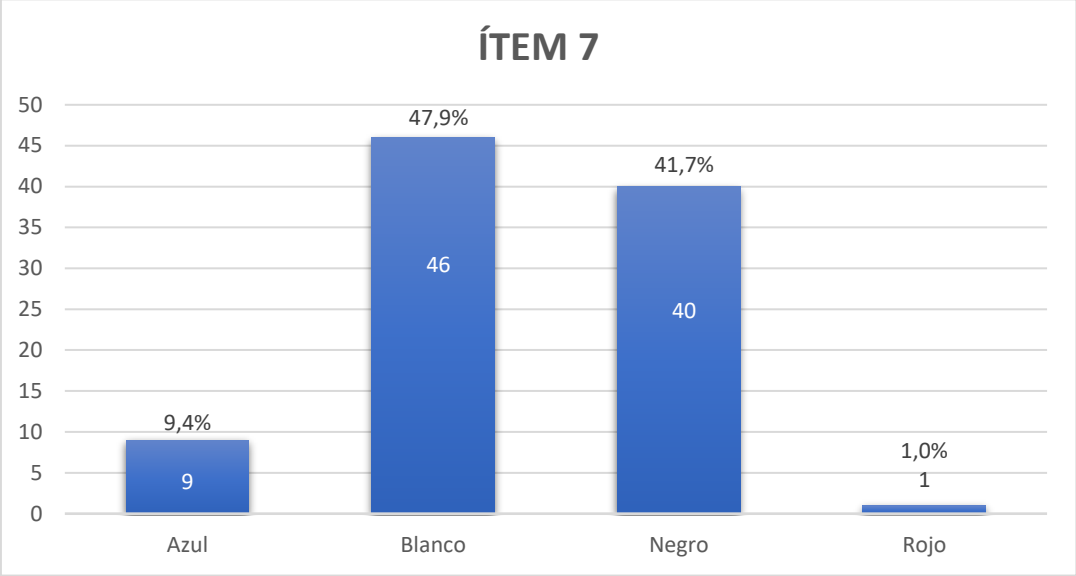


Figura 7. Color más popular de calzado deportivo

Análisis e interpretación de resultados

De acuerdo con los resultados de la **Figura 7**, se demuestra lo siguiente:

- Según los vendedores los colores más populares y solicitados en el calzado deportivo, son los colores blanco y negro, dado a que, en comparación con los colores Azul y blanco son más recomendados entre la clientela.

- De esta manera también resalta la importancia del color para el cliente según los vendedores, dado a que, el color blanco es más usado por las Instituciones como colegios o escuelas para su tiempo de recreación y el negro como un color neutro en el que combina con casi todos los colores.

- El color negro y blanco van del par al momento de adquirir calzado deportivo debido a que se puede combinar de acuerdo con la mayoría de estilos, para adaptarlo a cualquier estilo de vida.

Ítem 8: ¿Cómo ayuda a los clientes a tomar una decisión de compra?

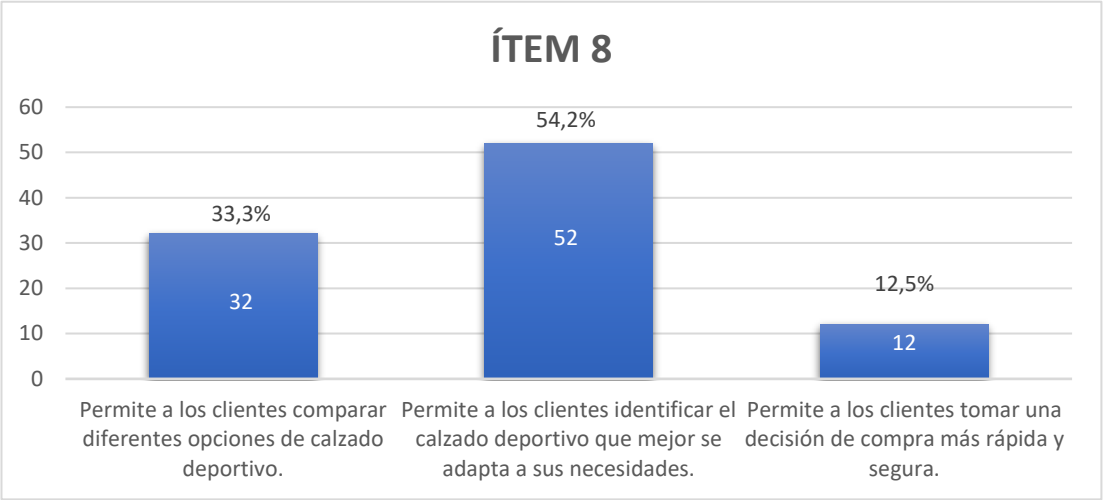


Figura 8. Conocimiento de recomendación para el cliente para su decisión de compra

Análisis e interpretación de resultados

Según los resultados de la **Figura 8**, se demuestra lo siguiente:

- Los vendedores permiten a los clientes identificar el calzado deportivo que mejor se adapte a sus necesidades, el cual, implica a las necesidades específicas del cliente como la comodidad o el tipo de uso que implementaría el cliente con el producto, denotando la importancia de recomendar al cliente según sus necesidades.

- En cambio, el vendedor, con menor medida permite a los clientes comprar diferentes opciones de calzado deportivo, e indica la opción de que el vendedor le presente sus productos e indicando la variedad de diseños, marcas, modelos y precios del calzado deportivo. Donde, desarrolla la necesidad de recomendar la calidad del producto acorde a su precio, modelo o marca.

- El vendedor usualmente permite a los clientes tomar una decisión de compra más rápida y segura para ayudar en la decisión de compra del cliente. Implica que el cliente tenga en mente lo que desee y el vendedor permita que el cliente desarrolle una decisión sobre el producto respondiendo dudas del cliente.

- Por último, estos resultados recomiendan que la toma de decisión en la compra de calzado deportivo, hay que permitirle al consumidor obtener varias opciones de acuerdo con su necesidad cotidiana, esto establece una opinión guiada para finalmente comprar el calzado.

Ítem 9: ¿Cuál es la marca y modelo de calzado deportivo más solicitado a los proveedores, según su experiencia?

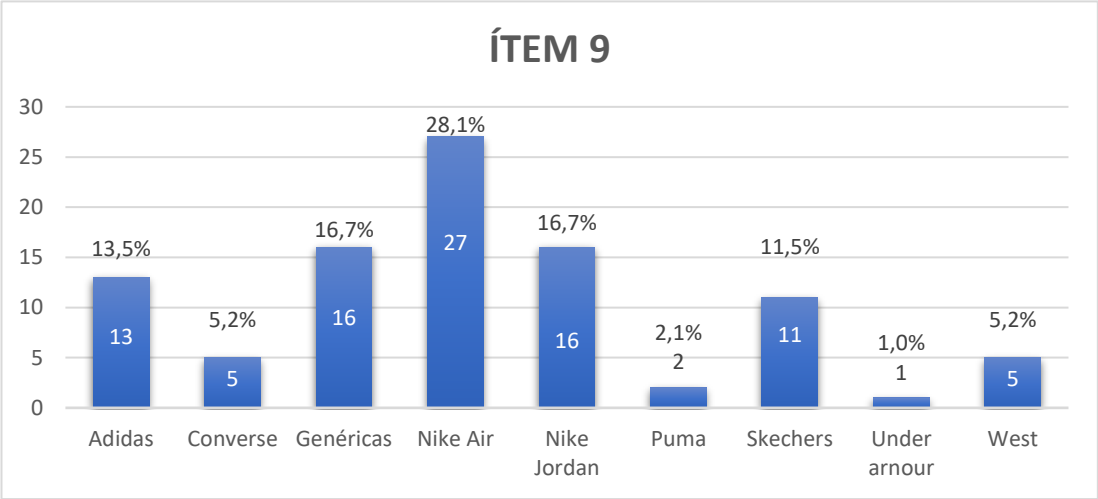


Figura 9. Marca y modelo de calzado deportivo más solicitado

Análisis e interpretación de resultados

Según los resultados de la **Figura 9**, se demuestra lo siguiente:

- Las respuestas de los vendedores indican que la marca más solicitada por los clientes es Nike dado a que sus modelos y diseños están a la moda según la búsqueda de sus clientes.

- Asimismo, algunos vendedores consideran que las marcas mas relevantes son Adidas y Skechers dado a que convencen en gran parte al cliente por sus diseños y modelos llamativos.

- Por otra parte, una gran porción de los vendedores considera que los clientes suelen confiar más en marcas nacionales que suelen relacionarse con marcas genéricas y diseños que suelen ser comparables con las grandes marcas internacionales, dado a que, el cliente conoce la calidad del producto nacional.

Esto señala que, ya sea una marca original o una imitación, los clientes muestran preferencia por marcas y modelos que reciben recomendaciones basadas en su popularidad y alta demanda en el mercado.

Ítem 10: ¿Qué tan útil es la recomendación del calzado deportivo a la hora de elegir para los clientes al momento de comprar?

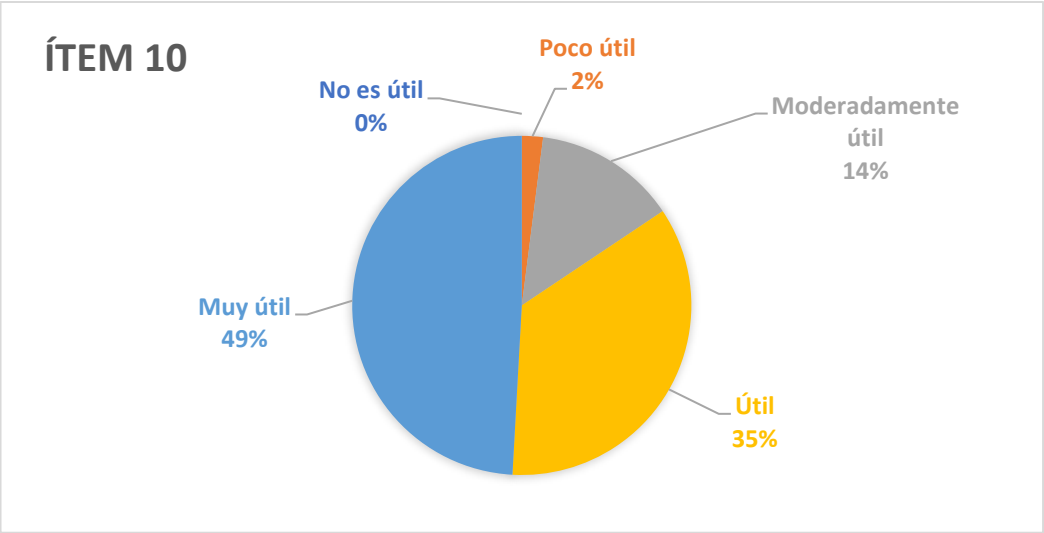


Figura 10. Utilidad de la recomendación de calzado deportivo para el cliente

Análisis e interpretación de resultados

Según los resultados de la **Figura 10**, se demuestra lo siguiente:

- Según los vendedores, es moderadamente útil la recomendación al cliente, dado que el vendedor demuestra las características más importantes al cliente y en base a ello el cliente pueda establecer una decisión o negocio.

- De cierto modo, hay un grupo pequeño de vendedores que resaltan que los clientes no suelen necesitar recomendaciones, dado que, son clientes que ya saben lo que quieren y usualmente no necesitan una recomendación.

- En conclusión, estos resultados establecen la necesidad de una guía importante de recomendación para elección de compra, debido a que algunos clientes no están informados sobre los diferentes modelos existentes por diferentes marcas.

Ítem 11: ¿Qué tan probable es que los clientes sigan una recomendación al momento de comprar calzado deportivo?

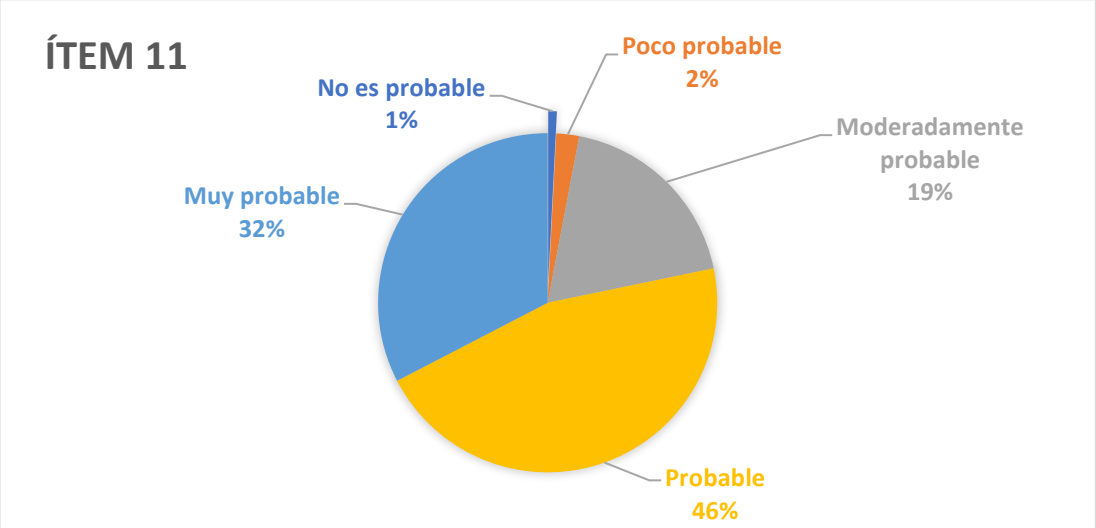


Figura 11. Probabilidad de la decisión de compra ante una recomendación del cliente

Análisis e interpretación de resultados

Según los resultados de la **Figura 11**, se demuestra lo siguiente:

- Más de la mitad de los vendedores aciertan que sus clientes tienden a decidir su producto en base a su recomendación y presentación, el cual aplican métodos como la de establecer preguntas al cliente, para así, determinar su necesidad y acertar a la decisión de compra.

- Por otro lado, pocos vendedores establecen que sus sugerencias al cliente, no suele ser positivas, dado a que el cliente muchas veces se fija en los modelos y diseños que se hallan en el local. Por lo tanto, esto sugiere que el nivel de insatisfacción del cliente con respecto a una recomendación se traduce en cierta duda o desconfianza.

- En consecuencia, según los resultados, esto respalda la idea de que el nivel de confianza en una recomendación es un factor determinante en la toma de decisión de compra por parte de los clientes. De esta forma, el consumidor tiene información sobre el calzado que va a adquirir a lo largo de varias opiniones y detalles que el vendedor puede llegar a ser útil.

2.2.4 Fichas de contenido

En esta sección se establece una investigación sobre puntos clave para la implementación de este proyecto, por lo cual, es necesario obtener fichas de contenidos para mantener una retroalimentación de la teoría prescindible a este a proyecto.

Tabla 6. Tabla de contenido No. 1

Ficha de contenido No. 1	
Tema	Diseño de un sistema de recomendación basado en ganancias que usa Machine Learning para balancear los beneficios para el usuario y la empresa.
Subtema	Motor de Sugerencias.
Referencia	Juan Fernando Riofrío Valarezo, "Diseño de un sistema de recomendación basado en ganancias que usa machine learning para balancear los beneficios para el usuario y la empresa.," Quito: EPN, 2022., Master's thesis 2022.
Contenido	El trabajo de titulación de Juan Fernando Riofrío Valarezo se enfoca en el diseño de un sistema de recomendación basado en ganancias que utiliza machine learning para balancear los beneficios para el usuario y la empresa. El sistema propuesto, denominado Sistema de Recomendación Multi-consciente (MARS), utiliza un método de agregación de ranking ponderada para re-ranquear los artículos y optimiza los pesos iterativamente utilizando una variante del algoritmo de Gradiente Descendente. El objetivo es controlar el impacto en el usuario para no comprometer la lealtad del cliente, al mismo tiempo que aumenta la rentabilidad para la empresa.
Fecha de consulta	22/9/2023

Tabla 7. Tabla de contenido No. 2

Ficha de contenido No. 2	
Tema	Developing a framework for researching recommender systems and their effects.
Subtema	Motor de Sugerencias.
Referencia	Felicia Loecherbach and Damian Trilling, "3bij3–Developing a framework for researching recommender systems and their effects," Computational Communication Research, vol. 2, pp. 53–79, 2020.
Contenido	El documento presenta un marco de trabajo para investigar los sistemas de recomendación de noticias y sus efectos en la percepción y consumo de noticias. Los autores han desarrollado una aplicación web llamada 3bij3, que permite realizar experimentos de campo a gran escala para rastrear el uso de los participantes de artículos de noticias seleccionados por diferentes algoritmos de recomendación. Este enfoque innovador proporciona a los investigadores control sobre el sistema de recomendación en estudio y crea un entorno realista para que los participantes interactúen con él. El documento también describe cómo se integran el web scraping, diferentes métodos para comparar y clasificar artículos de noticias y diferentes sistemas de recomendación en la aplicación 3bij3.
Fecha de consulta	22/9/2023

Tabla 8. Tabla de contenido No. 3

Ficha de contenido No. 3	
Tema	Sistemas de recomendación.
Subtema	Motor de Sugerencias.
Referencia	García Escudero y Luis Ángel Berrío Galindo, "Sistemas de recomendación," Universidad de Valladolid. Facultad de Ciencias, 2020.
Contenido	El documento "Sistemas de Recomendación" aborda la importancia y el funcionamiento de los sistemas de recomendación en diversos contextos. Se discuten diferentes algoritmos, como el filtro basado en contenido y el filtro colaborativo, así como la factorización de matrices. Además, se menciona un ejemplo práctico relacionado con un concurso creado por Netflix, que supuso un avance significativo en el campo de los sistemas de recomendación. El documento también explora la importancia de la información implícita y explícita de los usuarios, así como la introducción de componentes temporales en los modelos de recomendación. Se presentan enfoques para entrenar modelos, como el descenso de gradiente estocástico, y se discute la captura de interacciones entre usuarios e ítems, así como los sesgos asociados. Además, se detallan las formas de implementar un filtro basado en contenido y se menciona su relevancia en la recomendación de productos en Internet.
Fecha de consulta	22/9/2023

Tabla 9. Tabla de contenido No. 4

Ficha de contenido No. 4	
Tema	Aplicación de técnicas de web scraping y procesamiento del lenguaje natural para la extracción y evaluación de información de una página web de empleo.
Subtema	Web Scraping.
Referencia	Guillermo Valle Gutiérrez-Luis Pita-Romero Rodríguez, "Aplicación de técnicas de web scraping y procesamiento del lenguaje natural para la extracción y evaluación de información de una página web de empleo," ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI), vol. 7, pp. 224–240, 2021.
Contenido	El proyecto "Aplicación de técnicas de web scraping y procesamiento del lenguaje natural para la extracción y evaluación de información de una página web de empleo" realizado por Guillermo Valle Gutiérrez, bajo la dirección de Luis Pita-Romero Rodríguez, se centra en el estudio de la viabilidad de aplicar técnicas de procesamiento del lenguaje natural para el análisis de sentimientos en valoraciones extraídas de una página web de empleo. El trabajo incluye la exploración e implementación de un módulo de web scraping, así como el análisis de viabilidad de la aplicación de técnicas de procesamiento del lenguaje natural para el análisis de sentimiento en valoraciones. Se abordan conceptos fundamentales de web scraping y procesamiento del lenguaje natural, y se describen las tecnologías utilizadas en el proyecto. El estudio concluye con resultados adecuados en términos de precisión en las valoraciones extraídas de la página web Indeed, lo que sugiere la viabilidad de estas técnicas para el análisis de sentimientos en valoraciones de empleo. El proyecto se presenta en la Escuela Técnica Superior de Ingeniería (ICAI) de la Universidad Pontificia Comillas en Madrid, y ha sido autorizado para su digitalización, depósito y divulgación en red, lo que permite su acceso y consulta por parte de la comunidad académica y el público en general.
Fecha de consulta	22/9/2023

Tabla 10. Tabla de contenido No. 5

Ficha de contenido No. 5	
Tema	Búsqueda automatizada en bolsas de trabajo online con técnicas de web scraping.
Subtema	Web Scraping.
Referencia	Leonel Cruz Norberto, "Búsqueda automatizada en bolsas de trabajo online con técnicas de web scraping," 2022.
Contenido	El documento aborda la transformación laboral durante la pandemia, destacando la aceptación del trabajo virtual. Se enfoca en el crecimiento del ecosistema digital en América Latina, subrayando el auge del e-commerce. Se resalta la importancia de los sistemas computacionales para el funcionamiento empresarial eficiente y se menciona la práctica esencial del web scraping en la era del Big Data. La propuesta final implica la creación de una aplicación de escritorio que emplea técnicas de web scraping para automatizar la búsqueda de perfiles laborales, ahorrando tiempo y recursos en la industria de Atracción de Talento.
Fecha de consulta	22/9/2023

Tabla 11. Tabla de contenido No. 6

Ficha de contenido No. 6	
Tema	HabScraper: herramienta automatizada para la extracción de datos con web scraping.
Subtema	Web Scraping.
Referencia	Nicole Luise Vicente Stenhouse, "HabScraper: herramienta automatizada para la extracción de datos con web scraping.," Universitat de les Illes Balears, 2018.
Contenido	El documento destaca el papel fundamental de Internet como fuente de datos y la necesidad de extraer información de manera eficiente en entornos empresariales. Se presenta el caso de HabScraper, una aplicación web diseñada para conectar usuarios con servicios del hogar. Ante la creciente demanda, se propone el uso de un web scraper automatizado para extraer datos de contactos de directorios web de empresas. HabScraper no solo extrae datos, sino que también los procesa para crear registros de empresas en la base de datos. El documento se estructura con una introducción al web scraping, la presentación de HabScraper, detalles de implementación y conclusiones.
Fecha de consulta	22/9/2023

Tabla 12. Tabla de contenido No. 7

Ficha de contenido No. 7	
Tema	Uso de técnicas de Web Scraping para obtención automática de bases de datos en la Web.
Subtema	Web Scraping.
Referencia	Rogelio Mijangos-Espinosa, Alicia Martínez Rebollar, Hugo Estrada-Esquivel, and Yasmín Hernández Pérez, "Uso de técnicas de Web Scraping para obtención automática de bases de datos en la Web.," Res.
Contenido	El artículo destaca la prevalencia de la web como principal canal de comunicación global y cómo millones de usuarios la utilizan para transmitir datos a diversas entidades. Sin embargo, el volumen masivo de datos en la web dificulta su captura sistemática y oportuna. En respuesta, se presenta una herramienta de software que implementa técnicas de web scraping para buscar, seleccionar y descargar bases de datos alojadas en la web. Este sistema ha sido evaluado en la obtención y almacenamiento de información sobre el COVID-19 en México, demostrando su utilidad para la recopilación y análisis de datos.
Fecha de consulta	22/9/2023

Tabla 13. Tabla de contenido No. 8

Ficha de contenido No. 8	
Tema	Historia del web, 1.0, 2.0, 3.0 y 4.0.
Subtema	Web 2.0.
Referencia	Marino Latorre, "Historia del web, 1.0, 2.0, 3.0 y 4.0," Universidad Marcelino Champagnat, vol. 1, 2018.
Contenido	El documento aborda el tema de la World Wide Web (www) como una red de documentos interconectados mediante hipervínculos en Internet, utilizando tecnología digital que incluye textos, gráficos y archivos diversos. A diferencia de Internet, la web es un subconjunto accesible mediante navegadores y alberga información que se puede consumir con interactividad. No obstante, Internet abarca diversas plataformas, como correo electrónico, redes sociales, juegos, etc. La web ha evolucionado desde la 1.0 (consumo unidireccional) hasta la 4.0, destacando la web semántica en la 3.0 y la inteligencia predictiva en la 4.0, introducida en 2016.
Fecha de consulta	22/9/2023

Tabla 14. Tabla de contenido No. 9

Ficha de contenido No. 9	
Tema	Plan de marketing digital para almacenes deportivos El Kimono en la provincia de Imbabura.
Subtema	Marketing Digital.
Referencia	Marilin Carolina Barrera Echeverría, "Plan de marketing digital para almacenes deportivos El Kimono en la provincia de Imbabura," B.S. thesis 2018.
Contenido	Este trabajo de grado, realizado en Ibarra, aborda la creación de un "Plan de Marketing Digital" para los almacenes deportivos "El kimono" en la provincia de Imbabura. Inicia con un diagnóstico situacional, destacando bajos ingresos, lo que motiva la necesidad de incursionar en medios digitales. El análisis FODA revela oportunidades para aplicar estrategias de marketing. El estudio de mercado emplea diversas herramientas para identificar oferta, demanda y demanda insatisfecha. La propuesta de Marketing Digital se centra en la presencia online a través de página web, redes sociales y SEO para aumentar ventas. Se presenta un presupuesto de marketing considerando escenarios futuros para evaluar el beneficio potencial.
Fecha de consulta	22/9/2023

Tabla 15. Tabla de contenido No. 10

Ficha de contenido No. 10	
Tema	Ventas online: Factores claves de la experiencia de compra.
Subtema	Ventas Online
Referencia	Rodrigo Gerhardt, "Ventas online: Factores claves de la experiencia de compra," Universidad Siglo 21, B.S. thesis June 2021.
Contenido	El documento se introduce en el contexto de la creciente influencia tecnológica en la vida cotidiana, las compras y ventas en línea experimentan un cambio significativo. Este fenómeno requiere una adaptación tanto de los consumidores como de los vendedores, instando a las tiendas físicas a ingresar al mundo virtual y a los comerciantes en línea a mejorar su posición competitiva. El estudio se centra en el comportamiento del consumidor en el entorno digital, destacando la importancia de mejorar la experiencia de compra en las tiendas virtuales. La investigación, basada en herramientas cualitativas y cuantitativas, revela que la confianza, influenciada por opiniones de terceros, es crucial para los consumidores en línea, destacando la calidad del producto, el precio y el tiempo de envío como factores clave para la recomendación.
Fecha de consulta	22/9/2023

Tabla 16. Tabla de contenido No. 11

Ficha de contenido No. 11	
Tema	Análisis de tendencias en la personalización de los resultados en buscadores web.
Subtema	Ventas Online
Referencia	Eric Bárbaro Utrera Sust, Alfredo Simón Cuevas, and José A. Olivas, "Análisis de tendencias en la personalización de los resultados en buscadores web.," Revista Cubana de Ciencias Informáticas, vol. 12, pp. 126–146, 2018.
Contenido	El documento examina el crecimiento exponencial de la web y la necesidad de personalizar los resultados de búsqueda, ya que los buscadores tradicionales devuelven los mismos resultados para consultas similares. Se enfoca en Modelos de Personalización para Buscadores Web (MPBW) con énfasis en la búsqueda personalizada. La investigación revisa las etapas clave en la construcción y evaluación de MPBW, incluyendo la modelación de características de usuarios, la recopilación de información, la representación de datos, el trabajo multi-idioma, la confiabilidad de datos, la creación de perfiles y el uso de agentes. El documento concluye destacando desafíos y direcciones futuras en el campo del MPBW.
Fecha de consulta	22/9/2023

Tabla 17. Tabla de contenido No. 12

Ficha de contenido No. 12	
Tema	Análisis de la experiencia del usuario en la plataforma web para la compra de calzado deportivo en Runa Store.
Subtema	Compra de calzado en la línea deportiva
Referencia	Cynthia Beatriz Alvarez García, Adriana Nicole Rosales Olivas, and Luis Fernando Lucas Valera Lalangui, "Análisis de la experiencia del usuario en la plataforma web para la compra de calzado deportivo en Runa Store," 2020.
Contenido	En el documento se aborda la evaluación de la experiencia del usuario en la plataforma de comercio electrónico Runa Store, en el contexto del crecimiento del mundo online. El estudio contextualiza el problema, presenta objetivos, viabilidad y limitaciones. Explora literatura sobre marketing experiencial y modelos de experiencia web, como el de Constantinides y Geurts. Argumenta la relevancia de generar una experiencia web en el comercio electrónico, examina el contexto de la plataforma y su competencia. Describe el alcance, enfoque y diseño de la investigación, incluyendo la técnica de encuesta. El quinto capítulo presenta resultados, análisis y conclusiones para contribuir a la literatura sobre la experiencia web en plataformas de comercio electrónico en Perú.
Fecha de consulta	22/9/2023

En base a la **Tabla 6**, **Tabla 7**, **Tabla 8**, **Tabla 9**, **Tabla 10**, **Tabla 11**, **Tabla 12**, **Tabla 13**, **Tabla 14**, **Tabla 15**, **Tabla 16**, y la **Tabla 17**, se pudo recopilar y resumir información relevante en contexto a la necesidad fundamental del desarrollo del proyecto, contribuyendo a la calidad, integridad y éxito del proyecto de investigación.

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

3.1 Análisis y discusión de los resultados

En esta sección, se procederá a realizar un análisis detallado de los datos recopilados a partir de las encuestas realizadas. Los resultados obtenidos revelan información significativa que respalda la relevancia del calzado deportivo en la sociedad, según la percepción de los vendedores de calzado deportivo en las localidades de Ambato y Cevallos.

De este modo, se destacan las características más vistosas y las marcas que tienen mayor relevancia para el consumidor. En este contexto, se logró obtener características relevantes como el modelo, la marca, el precio, las tallas, y el color, el cual, son útiles para la implementación de Web Scraping.

Por lo tanto, después de haber obtenido estos resultados, se desarrolla el proceso para implementar la extracción de datos mediante Web Scraping, para que en base a los datos extraídos se establezca un modelo de recomendación, para así, implementar el modelo de recomendación en función a los datos obtenidos, en consecuencia, desarrollar una interfaz para la demostración del motor de sugerencias al usuario. Estos procesos serán implementados en secuencia como se demuestra en la **Figura 12**.

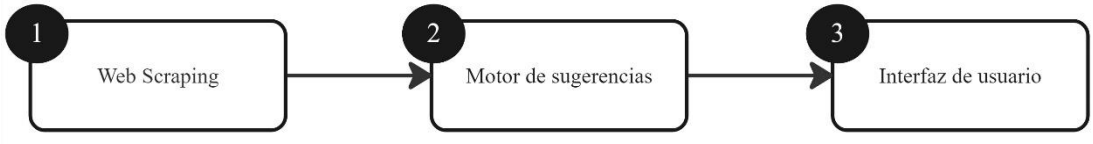


Figura 12. Pasos de la implementación del proyecto

De esta manera se puede establecer las fases requeridas para implementar el desarrollo de propuesta, según la información obtenida.

3.2 Análisis comparativo de metodologías Ágiles

En esta sección se establece la selección de una metodología ágil para el desarrollo de la propuesta de solución del proyecto de investigación, por ello, es importante dar uso de una metodología ágil, por la posibilidad de adaptarse a entornos en constante cambio según el desarrollo del proyecto, la gestión eficaz de proyectos, el fomento de la colaboración y la responsabilidad colectiva, para aportar una entrega continua y temprana de valor de tareas asignadas. Además de buscar mejorar los tiempos de entrega, las metodologías ágiles promueven modelos adaptativos y permiten cambios en diversas fases del proyecto, facilitando la retroalimentación del equipo de desarrollo. Esta flexibilidad se vuelve crucial en entornos empresariales dinámicos, donde la rapidez y la capacidad de adaptación son elementos esenciales para el éxito de los proyectos. Como parte del enfoque metodológico, se implementará una tabla comparativa entre cuatro metodologías ágiles.

Según Greace Kelly Diaz Moreno [28] indica que las metodologías ágiles, como *Lean Software Development*, *Scrum*, *Kanban* y *Scrumban*, son enfoques de gestión de proyectos que se centran en la flexibilidad, la adaptabilidad y la entrega de valor de manera rápida y eficiente. Estas metodologías se desarrollan a través de la implementación de principios específicos que buscan optimizar el proceso de desarrollo de proyectos de sistemas.

Scrum

La autora acierta que Scrum es un framework ágil conocido por su capacidad de adaptación, su enfoque iterativo y su eficacia en la gestión de proyectos. Se destaca por fomentar la transparencia en la comunicación y la creación de un ambiente de responsabilidad colectiva. La metodología se organiza en ciclos de trabajo cortos llamados Sprints, donde se priorizan las funcionalidades más importantes para su entrega temprana, lo que permite una rápida adaptación a los cambios.

Un componente clave de Scrum es la Backlog Priorizado del Producto, que consiste en una lista de requerimientos del negocio y del proyecto ordenados por importancia en forma de historias de usuario. Durante cada sprint, el equipo se enfoca en la creación

de entregables en incrementos, contribuyendo al desarrollo continuo del producto. Scrum también se caracteriza por limitar el trabajo en progreso y por fomentar la autoorganización de los equipos de trabajo, lo que impulsa la colaboración y la toma de decisiones compartida. Este enfoque modular y centrado en la entrega temprana de valor hace que Scrum sea una elección popular en entornos que requieren flexibilidad y adaptabilidad.

Lean Software Development

La autora indica que Lean se fundamenta en siete principios que tienen como objetivo gestionar los cambios y optimizar el proceso de desarrollo de sistemas.

Estos principios incluyen:

- Eliminación de actividades no valiosas: Se busca identificar y eliminar todas las actividades que no aportan valor al negocio, reduciendo así el desperdicio y centrándose en lo esencial.

- Toma de decisiones tardías: Se aboga por postergar las decisiones hasta el último momento posible, permitiendo contar con información más actualizada y relevante.

- Liberación temprana de funcionalidades: El objetivo es liberar funcionalidades de manera temprana en el proceso de desarrollo, lo que facilita obtener retroalimentación temprana y ajustar el enfoque según las necesidades del cliente.

- Empoderamiento del equipo: Se fomenta la autorresponsabilidad y empoderamiento de los equipos de desarrollo, permitiéndoles tomar decisiones y ser dueños de su propio proceso.

- Construcción del producto con integridad: Se busca garantizar la coherencia y la integridad del producto final desde el inicio del proceso de desarrollo.

- Perspectiva global del proyecto: Se promueve una visión holística del proyecto, considerando el impacto de las decisiones en todo el sistema.

- Creación de equipos cohesionados: Se enfatiza la importancia de construir equipos de trabajo cohesionados y eficientes, donde la colaboración y la comunicación son fundamentales.

Además, se destaca la relevancia de la motivación individual, la comunicación cara a cara, el software funcionando como medida principal de progreso, el desarrollo sostenible, la atención a la excelencia técnica y al buen diseño, así como la maximización de la cantidad de trabajo no realizado.

Scrumban

La autora indica que Scrumban es una metodología híbrida que fusiona elementos de Scrum y Kanban, ofreciendo una combinación de reglas que ambas metodologías por separado no posibilitan. Esta aproximación se considera avanzada y está orientada a mejorar de manera continua el proceso de desarrollo. La flexibilidad y adaptabilidad son características distintivas de Scrumban, ya que permite ajustes en diferentes momentos según la realidad del proyecto y facilita la retroalimentación del equipo de desarrollo.

Existen dos enfoques principales en relación con Scrumban como un modelo híbrido: uno donde los elementos de Scrum se aplican directamente al enfoque de Kanban, y otro donde los elementos de Kanban se incorporan al enfoque de Scrum. Esta versatilidad brinda un mayor grado de flexibilización en la implementación, lo que lo hace adecuado para adaptarse a una variedad de contextos y necesidades específicas.

Tabla 18. Comparación de Metodologías Agiles

Metodología	Descripción	Adaptabilidad y Flexibilidad	Visibilidad del Proceso	Entrega Temprana de Valor	Gestión del Trabajo en Curso	Ciclos de Retroalimentación	Facilidad de Implementación
Scrum	Framework ágil adaptable, iterativo y eficaz que prioriza la transparencia y responsabilidad colectiva. Se organiza en sprints con entrega incremental y limita el trabajo en curso.	Altamente adaptable a cambios en los requisitos durante los sprints planificados.	Ofrece visibilidad a través de reuniones diarias y artefactos como el burndown chart.	Planificación en sprints para entrega incremental.	Limita el trabajo en curso durante los sprints.	Retroalimentación regular al final de cada sprint.	Estructura bien definida, pero puede requerir cambios culturales.
Kanban	Sistema visual basado en tarjetas que optimiza el flujo de trabajo y maximiza la eficiencia. Proporciona una visión clara del estado de cada tarea y permite ajustes en tiempo real.	Ofrece flexibilidad continua y permite cambios en cualquier momento.	Proporciona una visualización clara del flujo de trabajo mediante tableros visuales.	Enfocado en maximizar la eficiencia y mejorar continuamente.	Controla activamente el trabajo en curso para mejorar la eficiencia.	Ofrece retroalimentación continua y rápida.	Más fácil de implementar, especialmente en equipos ya existentes.
Lean Software Development	Enfoque en la eliminación de actividades sin valor y liberación temprana de funcionalidades. Busca reducir el desperdicio y optimizar el proceso de desarrollo.	Se enfoca en eliminar actividades sin valor y permite ajustes tempranos.	Busca optimizar el proceso y ofrece transparencia en las actividades.	Busca liberar funcionalidades temprano y reducir el desperdicio.	Reduce el trabajo innecesario y optimiza el flujo de trabajo.	Enfocado en la mejora continua y la retroalimentación constante.	Requiere cambios culturales y prácticas específicas.

Scrumban	Metodología híbrida que combina componentes de Scrum y Kanban. Permite adoptar una combinación de reglas que ambas metodologías por separado no permiten.	Combina la flexibilidad de Kanban con la estructura de Scrum, permitiendo ajustes graduales.	Combina la visualización de Kanban con la estructura de Scrum.	Permite entrega temprana de funcionalidades mientras se mantiene la estructura de Scrum.	Combina el control del trabajo en curso de Kanban con la estructura de Scrum.	Combina la retroalimentación estructurada de Scrum con la flexibilidad de Kanban.	Combina la estructura de Scrum y la simplicidad de Kanban.

Según la detallada comparación presentada en la **Tabla 18**, Scrumban se posiciona como la metodología adecuada y versátil para la implementación en el desarrollo del proyecto. Su marcada flexibilidad en la planificación supera a Scrum puro, permitiendo ajustar las reglas según las necesidades específicas del proyecto y realizar cambios en diferentes momentos. Scrumban destaca al ofrecer ciclos de retroalimentación tanto al concluir los sprints de forma continua a lo largo del flujo de trabajo, facilitando ajustes ágiles en proyectos de corta duración. Su capacidad de adaptación a diversos entornos y necesidades específicas del proyecto se traduce en la flexibilidad para ajustar la cantidad de épicas y reglas de acuerdo con la dinámica del equipo y las exigencias del proyecto. En consecuencia, Scrumban se presenta como una elección que permite realizar ajustes en tiempo real, basándose en las lecciones aprendidas durante el desarrollo del proyecto. Esta característica es especialmente relevante para enfrentar los desafíos y cambios esenciales a proyectos de corta duración.

A diferencia de la metodología Lean, Scrumban se posiciona como la opción más adecuada debido a su flexibilidad y capacidad para adaptarse a diferentes soluciones entre los miembros del equipo de desarrollo. Su estructura híbrida, combinando componentes de Scrum y Kanban, lo convierte en un enfoque ágil y adaptable que puede optimizarse para satisfacer las necesidades específicas del proyecto con restricciones temporales y equipos reducidos.

Por lo tanto, en base a los pasos de implementación del proyecto en la Figura 12, se implementó el siguiente tablero Scrumban para gestionar el desarrollo de un sistema que incluye funcionalidades de Web Scraping, un Motor de Sugerencias, en secuencia la implementación de la Interfaz de Usuario se desglosaron tareas para un proyecto Back-end y Front-end para la selección y recomendación de calzado deportivo. Donde se establecen Épicas, Historias de Usuario y Tareas.

Tabla 19. Tablero Scrumban

Épica	Historia de Usuario	Tareas
Desarrollo de la Funcionalidad de Web Scraping	El usuario desea que el sistema recopile información actualizada sobre calzados deportivos.	Definir herramientas para la funcionalidad de Web Scraping.
		Establecer una lista de páginas web de marcas de calzado deportivo e implementen ventas en línea.
		Desarrollar el proceso de Web Crawler y Scraping.
		Almacenar datos extraídos en MongoDB.
		Realizar pruebas y asegurar la calidad del proceso de Web Scraping.
		Documentar el proceso y requisitos técnicos.
Desarrollo del Motor de Sugerencias	El usuario desea un motor de sugerencias basado en Web Scraping para mejorar la toma de decisiones en la compra de calzado deportivo.	Investigar el modelo de recomendación a implementar en el motor de sugerencias.
		Implementar el algoritmo de recomendación.
Desarrollo del Back-end	El usuario desea ser informado sobre el estado del proceso de extracción de datos mediante SocketIO y establecer consultas a los datos almacenados de MongoDB extraídos.	Configurar comunicación SocketIO entre Scrapy y el Back-end.
		Notificar inicio y finalización de extracción de datos al Front-end.
		Desarrollar lógica para consultas de calzados deportivos.
		Implementar servicio API que retorne los resultados del modelo de recomendación implementado.
		Diseñar e implementar API para consultas de calzados.
		Documentar la API del Back-end.
Desarrollo del Front-end	El usuario desea seleccionar calzados deportivos y recibir recomendaciones basadas en sus elecciones.	Implementar interfaz de usuario para la selección de calzado.
		Integrar comunicación asincrónica con el Back-end para recibir actualizaciones en tiempo real sobre el estado del proceso de Scrapy y consultas de calzados.
		Mostrar recomendaciones basadas en las elecciones del usuario y detalles de consultas de calzados.
		Realizar pruebas de interfaz de usuario.

En la **Tabla 19**, se desplegaron épicas para representar unidades de trabajo significativas para segmentar su solución en tareas más pequeñas, en contexto a los procesos de implementación del proyecto predefinidos en la **Figura 12**.

De acuerdo con las épicas, se desglosaron tareas que permite resolver la implementación de cada uno y obtener como resultado el motor de sugerencias. También se establecieron historias de usuario para expresar los requisitos en un lenguaje comprensible y centrado en el usuario según las épicas definidas.

3.3 Desarrollo de la propuesta

En esta sección se establece el desarrollo de todas y cada una de las tareas específicas de cada épica impuesta en la **Tabla 19**.

Antes de establecer el desarrollo de las épicas, se estableció el orden las épicas a mediante el diagrama de la **Figura 13**, para identificar el proceso requerido en la implementación del motor de sugerencias:

Donde cada asignación se lo representa, de la siguiente manera:

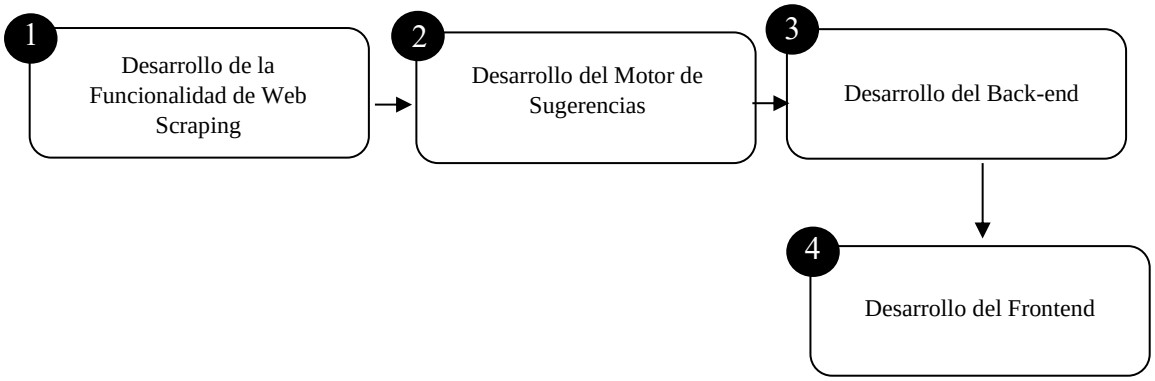


Figura 13. Proceso de elaboración del Motor de Sugerencias

3.3.1 Desarrollo de la funcionalidad de Web Scraping

Se empieza recopilando información sobre las herramientas para desarrollar Web Scraping de diferentes fuentes de páginas web sobre la venta de calzado deportivo para. Donde, según la **Tabla 3** se implementa una comparación sobre las herramientas usadas en la extracción de características de páginas web, según artículos en contexto de la implementación de un sistema de recomendación implementado Web Scraping.

Por lo tanto, en base a las herramientas implementadas en los diferentes artículos para el desarrollo de Web Scraping en la **Tabla 20** y también la **Tabla 3**, se implementa el uso del Framework Scrapy, Selenium, por la facilidad de exportar los datos extraídos mediante archivo csv o json con métodos de búsqueda de etiquetas y la herramienta Selenium para obtener obtener páginas web de forma recursiva, como herramientas necesarias a establecer el proceso de extracción de datos y solucionando la primera tarea de la épica, el cual, trata sobre definir herramientas para la funcionalidad de Web Scraping.

Tabla 20. Comparación de herramientas Web Scraping

Herramienta	Lenguaje de Programación	Facilidad de Uso	Flexibilidad	Escalabilidad	Soporte de JavaScript	Comunidad y Documentación
BeautifulSoup	Python	Fácil	Baja	Baja	No	Buena
Selenium	Varios (Python, Java, C#)	Moderada	Alta	Alta	Sí	Buena
Scrapy	Python	Moderada	Alta	Alta	No	Buena
Puppeteer (Headless Chrome)	JavaScript (Node.js)	Moderada	Alta	Alta	Sí	Buena

De esta forma, para implementar el proceso de Web Scraping se desglosan tareas en un diagrama en la **Figura 14**Figura 14 para implementar desarrollar el proceso de Web Crawler y Scraping:

Figura 14. Proceso de recopilación de datos

a. Proceso de Web Crawler

Para iniciar con el proceso de Web Crawler, se implementa el uso de la fórmula de población infinita para obtener páginas de marcas de calzado deportivo, por el desconocimiento de la cantidad especifica de páginas web sobre marcas de calzado deportivo en Internet.

$$n = \frac{Z_{\alpha}^{2} * p * q}{e^{2}} \tag{8}$$

Donde, cada variable será usada de la siguiente forma:

e = Error estándar
Z = Nivel de confianza 90%.
p = Probabilidad de éxito.
q = Probabilidad de fracaso.
n = Tamaño de la muestra.

Los valores del error estándar (e) del 15%, el nivel de confianza (Z) del 90%, y las probabilidades de éxito (p) y fracaso (q) del 0.5 son seleccionados estratégicamente para garantizar la precisión y generalización de los resultados. Estos parámetros son fundamentales para obtener una muestra representativa y confiable, lo que respalda la validez y utilidad de los resultados en el estudio de la selección de páginas web en este ámbito.

$$n = \frac{1{,}645^2 * 0{,}5 * 0{,}5}{0{,}15^2} = 30{,}067 \approx 30 \quad (9)$$

Después se desarrolla una lista de páginas web de marcas de calzado deportivo e implementen ventas en línea en la **Tabla 21** para implementar el proceso de Web Crawler:

Tabla 21. Lista de links estáticos para Web Crawling

No.	Marca	Página web
1	Nike	https://www.nike.com/w/shoes-y7ok
2	Adidas	https://www.adidas.com/us/search?q=shoes
3	Puma	https://us.puma.com/us/en/puma/shop-all-shoes
4	Reebok	https://www.reebok.com/c/600000057/collection-shoes?page=1
5	Under Armour	https://www.underarmour.com/en-us/c/shoes/
6	New Balance	• https://www.newbalance.com/men/shoes/all-shoes/ • https://www.newbalance.com/women/shoes/all-shoes/
7	ASICS	https://www.asics.com/us/en-us/new- arrivals/c/aa80101000
8	La Sportiva	https://www.lasportivausa.com/footwear.html
9	Converse	ttps://www.converse.com/shop/shoes
10	The North Face	https://www.thenorthface.com/en-us/search/product?q=shoes%20footwear
11	Vans	https://www.vans.com/en-us/shoes-c00081
12	Umbro	https://umbropremier.com/collections/umbro-men-footwear
13	Joma	• https://www.joma-sport.com/en_IE/man/footwear-man • https://www.joma-sport.com/en_IE/woman/footwear-woman
14	Speedo	https://us.speedo.com/
15	Mizuno	https://www.mizunousa.com/running/shoes?pageSize=54&
16	Diodora	https://www.diadora.com/en/us/men-shoes
17	StepSport	https://www.stepsport.gr/en/brands/erke-59.html
18	Salomon	• https://www.salomon.com/en-us/shop/men/shoes.html • https://www.salomon.com/en-us/shop/women/shoes.html
19	Teva	https://www.teva.com/men-view-all/
20	Tommy Hilfiger	• https://usa.tommy.com/en/women/shoes-accessories/shoes • https://usa.tommy.com/en/men/shoes-accessories/shoes
21	Venus	https://www.venus.com.ec/hombre/zapato-casual/masculino-femenino.html
22	Fila	https://www.fila.com/shoes?prefn1=gender&prefv1=mens|womens|unisex
23	Babolat	https://www.babolat.com/us/tennis/shoes.html
24	Bullpadel	https://www.bullpadel.com/gb/29-footwear
25	Cole Han	• https://www.colehaan.com/mens-zerogrand • https://www.colehaan.com/womens-shoes
26	Columbia	• https://www.columbia.com/c/shoes-boots-mens • https://www.columbia.com/c/womens-shoes
27	Brooks	• https://www.brooksrunning.com/en_us/mens/shoes/?sz=72

		• https://www.brooksrunning.com/en_us/womens/shoes/?sz=72
28	Le coq sportif	• https://www.lecoqsportif.com/eu_EN/men/shoes.html • https://www.lecoqsportif.com/eu_EN/women/shoes.html
29	Air Jordan	https://www.nike.com/w/jordan-shoes-37eefzy7ok
30	Nordstrom	https://www.nordstrom.com/

De acuerdo con la lista establecida con la selección de marcas y modelos de calzado deportivo, se establece criterios de inclusión y exclusión, ya que algunas páginas tienen restricciones que limitan este proceso.

Tabla 22. Criterios de exclusión e inclusión para la selección de páginas a implementar Web Crawler.

Inclusión	• Páginas web escritas en inglés. • Páginas web con carga de datos dinámicos o estáticos.
Exclusión	• Páginas web que establecen restricciones anti scrapers. • Páginas web que nieguen peticiones http a la página mediante sensores de solicitudes.

En base a la **Tabla 22**, se procede a la exclusión de las páginas detalladas en la **Tabla 21**. Este procedimiento se lleva a cabo debido a que algunas páginas establecen la activación de un protocolo anti-scrapers durante la implementación del proceso de extracción. Dicho protocolo, gestionado por la página a través de interacciones dinámicas y control de descarga de datos, impide la conexión y obstaculiza la extracción de datos. A pesar de intentar establecer conexiones mediante diversas cabeceras que incorporan la propiedad "user-agent", no ha sido posible superar estas restricciones y efectuar la extracción de datos de dichas páginas, teniendo como resultado 18 páginas excluidas en la **Tabla 23**.

Tabla 23. Páginas excluidas para implementar Web Scraping

12	Umbro	https://umbropremier.com/collections/umbro-men-footwear
13	Joma	• https://www.joma-sport.com/en_IE/man/footwear-man • https://www.joma-sport.com/en_IE/woman/footwear-woman
14	Speedo	https://us.speedo.com/
15	Mizuno	https://www.mizunousa.com/running/shoes?pageSize=54&
16	Diodora	https://www.diadora.com/en/us/men-shoes
17	StepSport	https://www.stepsport.gr/en/brands/erke-59.html
18	Salomon	• https://www.salomon.com/en-us/shop/men/shoes.html • https://www.salomon.com/en-us/shop/women/shoes.html
19	Teva	https://www.teva.com/men-view-all/
20	Tommy Hilfiger	• https://usa.tommy.com/en/women/shoes-accessories/shoes • https://usa.tommy.com/en/men/shoes-accessories/shoes

21	Venus	https://www.venus.com.ec/hombre/zapato-casual/masculino-femenino.html
22	Fila	https://www.fila.com/shoes?prefn1=gender&prefv1=mens|womens|unisex
23	Babolat	https://www.babolat.com/us/tennis/shoes.html
24	Bullpadel	https://www.bullpadel.com/gb/29-footwear
25	Cole Han	• https://www.colehaan.com/mens-zerogrand • https://www.colehaan.com/womens-shoes
26	Columbia	• https://www.columbia.com/c/shoes-boots-mens • https://www.columbia.com/c/womens-shoes
27	Brooks	• https://www.brooksrunning.com/en_us/mens/shoes/?sz=72 • https://www.brooksrunning.com/en_us/womens/shoes/?sz=72
28	Le coq sportif	• https://www.lecoqsportif.com/eu_EN/men/shoes.html • https://www.lecoqsportif.com/eu_EN/women/shoes.html
30	Nordstrom	https://www.nordstrom.com/

En este contexto, es pertinente especificar páginas que serán sometidas a extracción mediante Web Scraping. Este criterio de selección se fundamenta en el uso de un lenguaje general como el inglés, permitiendo la extracción de datos a través de Scrapy mediante la implementación aleatoria de cabeceras con la propiedad "user-agent" en la siguiente tabla.

Tabla 24. Paginas seleccionadas para implementar Web Scraping

No.	Marca	Página web
1	Nike	https://www.nike.com/w/shoes-y7ok
2	Adidas	https://www.adidas.com/us/search?q=shoes
3	Puma	https://us.puma.com/us/en/puma/shop-all-shoes
4	Reebok	https://www.reebok.com/c/600000057/collection-shoes?page=1
5	Under Armour	https://www.underarmour.com/en-us/c/shoes/
6	New Balance	• https://www.newbalance.com/men/shoes/all-shoes/ • https://www.newbalance.com/women/shoes/all-shoes/
7	ASICS	https://www.asics.com/us/en-us/new- arrivals/c/aa80101000
8	La Sportiva	https://www.lasportivausa.com/footwear.html
9	Converse	ttps://www.converse.com/shop/shoes
10	The North Face	https://www.thenorthface.com/en-us/search/product?q=shoes%20footwear
11	Vans	https://www.vans.com/en-us/shoes-c00081
29	Air Jordan	https://www.nike.com/w/jordan-shoes-37eefzy7ok

Por lo tanto, en base a la lista de enlaces obtenidas sobre páginas de calzado deportivo de la **Tabla 24**, se desarrolla una secuencia de actividades, el cual, se realiza siguiendo la muestra de páginas web de calzado deportivo, permitiendo la creación de una búsqueda recursiva de todos los enlaces asociados a productos de calzado deportivo, para recopilar y almacenarlo, preparándose así para la ejecución de la técnica de Web Scraping [6].

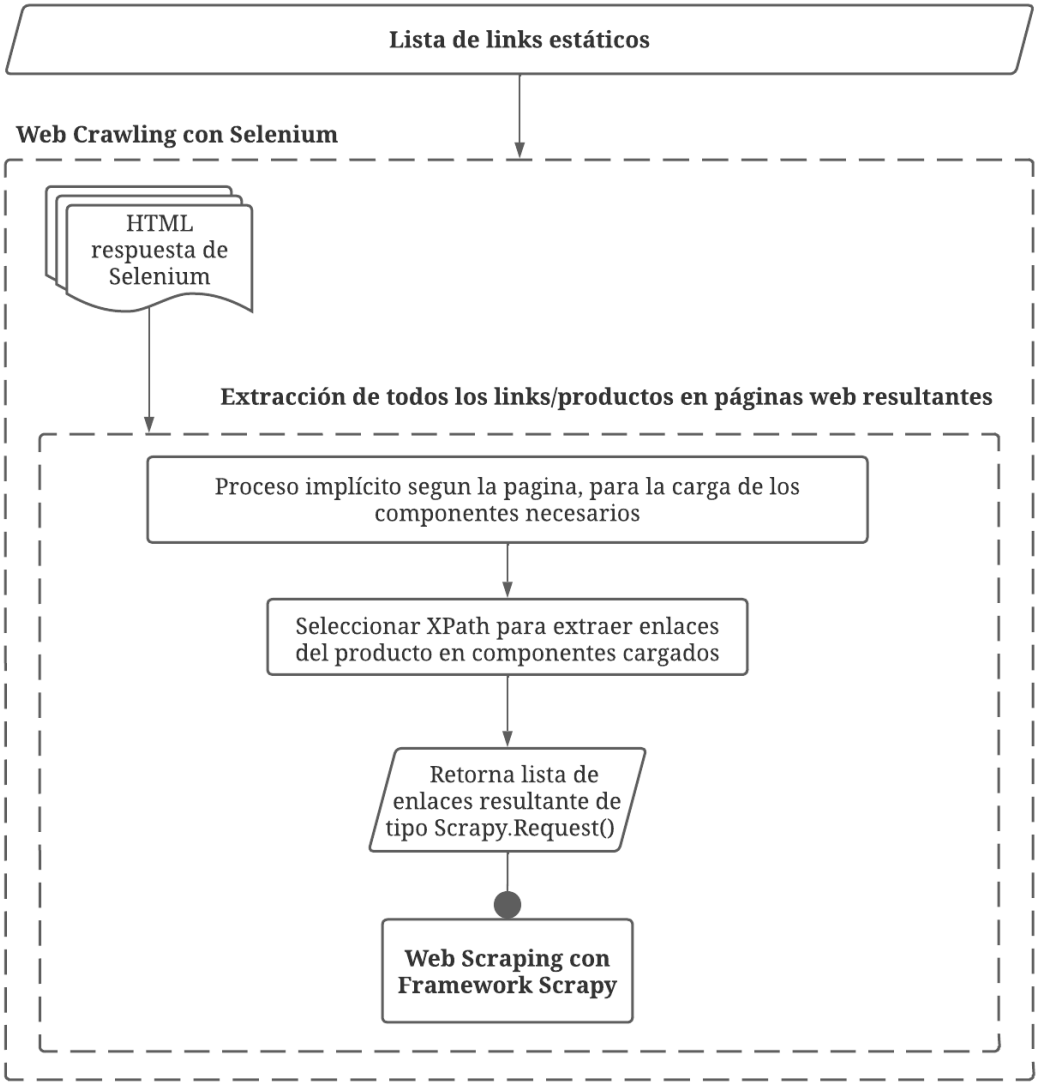


Figura 15. Diagrama de Flujo para implementar Web Crawling

El diagrama de flujo de la **Figura 15**Figura 15 muestra el proceso de extracción de datos web mediante el uso del marco Selenium por su capacidad para la automatización de navegadores. Se emplea para interactuar con sitios web, especialmente aquellos que presentan dinamismo en su contenido. El procedimiento inicia con una solicitud a la página web que se obtiene la lista de enlaces de calzado deportivo, y la respuesta a esta solicitud se almacena en un objeto HTML.

En primer lugar, se emplea el método get() de Selenium para cargar la página web, simulando la acción de un usuario que visita la página en un navegador convencional. Una vez cargada la página, se utiliza el método find_elements() de Selenium para localizar los elementos que se desean extraer. La identificación de estos elementos se realiza mediante la expresión XPath.

Una vez localizados los elementos, se recurre a los métodos get_attribute() o text() de Selenium para extraer los datos. El método get_attribute() se utiliza para extraer atributos de los elementos, mientras que text() se emplea para recuperar el texto contenido en dichos elementos, como se lo demuestra en el **Anexo C**.

Después, los datos extraídos se almacenan en memoria para establecer el proceso de Web Scraping en cada uno de ellos.

b. Web Scraping

En esta sección, se lleva a cabo la fase de obtención de características del calzado deportivo de los enlaces obtenidos. Con base en los resultados de la encuesta implementada, se identifican y determinan características como: color, marca, modelo, precio y tallas. Donde, también se requiere la extracción de características como: imágenes para reconocer el producto, tipo para reconocer la descripción del calzado, url_calzado para poder dirigirse al producto y url_raiz que representa el enlace inicial que se visitó en la sección del Web Crawler.

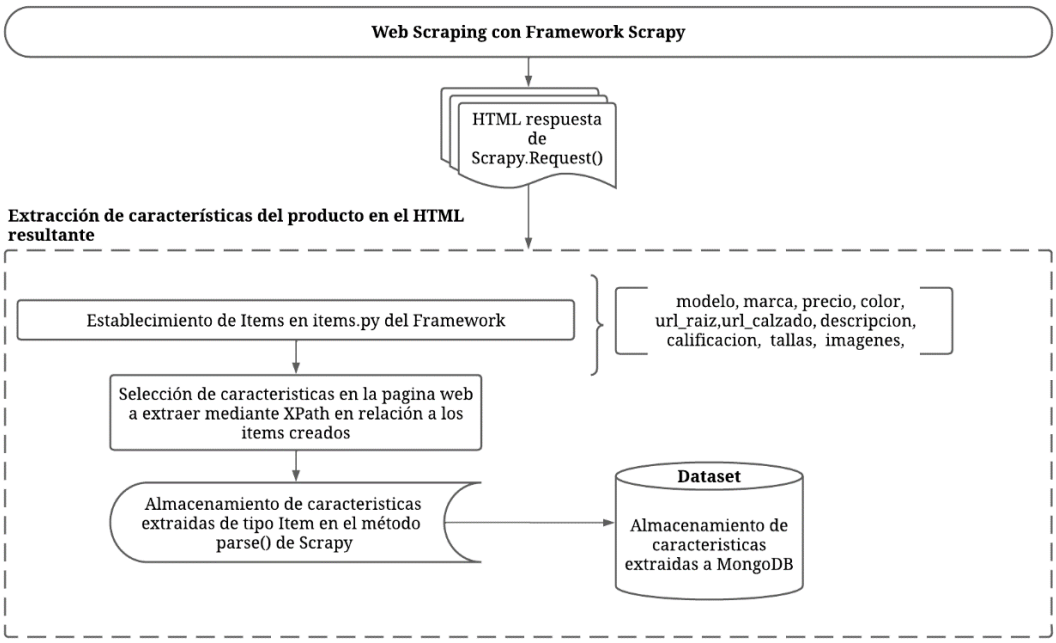


Figura 16. Diagrama de Flujo para implementar Web Scraping

El diagrama de flujo de la **Figura 16** ilustra el proceso de Web Scraping utilizando el marco Scrapy. Scrapy es una herramienta especializada en extracción de datos web, empleada para recopilar información de sitios web. El proceso se inicia con una solicitud a la página web objetivo, cuya respuesta se guarda en un objeto HTML.

A continuación, se crean los elementos en el archivo items.py según el marco requerido del Framework Scrapy. Los elementos son estructuras de datos que se utilizan para almacenar los datos extraídos de la página web, donde se crean en función de las características que se extrae en la página web.

Por lo tanto, el paso a realizar el Web Scraping se establece a cada producto de la lista de enlaces que se obtuvo en el proceso de Web Crawling. Esto se puede hacer utilizando el método parse( ) del framework Scrapy.

Las características extraídas se almacenan en los elementos. Los elementos se almacenan en un objeto Item en el método parse() del marco, demostrando este proceso en el **Anexo D**. En este contexto, se implementa una estrategia para protegerse de cambios en los sitios web objetivo al extraer información utilizando expresiones XPath. Esto se logra mediante el manejo de excepciones, donde se capturan posibles

errores como la falta de elementos específicos o problemas de tiempo de espera al buscar elementos en la página web como “NoSuchElementException” y “TimeoutException”. Además, se utiliza la herramienta Selenium para interactuar dinámicamente con el navegador web, lo que permite realizar acciones como cerrar ventanas emergentes o esperar a que ciertos elementos se carguen correctamente, adaptándose así a posibles cambios en la estructura de la página. Además, se capturan y procesan atributos dinámicos de elementos como las URL de las imágenes de productos mediante expresiones regulares, lo que garantiza que las URL sean consistentes y no estén sujetas a cambios en la estructura de la página.

Una vez que se han obtenido los elementos durante el proceso de extracción de datos, se emplean expresiones XPath para seleccionar las características específicas de las páginas web que se desean extraer. Estas expresiones XPath proporcionan una forma eficiente de identificar y recuperar elementos particulares de un documento XML o HTML, lo que facilita la extracción precisa de la información deseada. Sin embargo, durante el análisis de las páginas web de algunas marcas, se detectaron inconvenientes relacionados con etiquetas que no se renderizan debido a su naturaleza dinámica, que depende de la interacción del usuario. Para superar este obstáculo, se optó por analizar la estructura HTML de estas páginas y obtener documentos JSON que contienen la información necesaria sobre los modelos de calzado deportivo. Se implementó una función que utiliza expresiones regulares y recursividad para extraer las características requeridas de estos documentos JSON ilustrado en el **Anexo E**, lo que garantiza la obtención completa y precisa de los datos a pesar de la complejidad de la estructura de la página web. Este enfoque permite una extracción efectiva de información incluso en casos donde las etiquetas HTML no se renderizan de manera estática implementando el método de búsqueda de etiquetas en el **Anexo F**.

En este contexto, no se optó en gestionar dichas expresiones en una base de datos, ya que, el proceso de extraer etiquetas no solo infiere en obtener características mediante Xpath, sino también de forma dinámica etiquetas al archivo JSON obtenido y también si las expresiones XPath se almacenan en una base de datos externa, se requeriría una sincronización constante entre la base de datos y el código fuente para asegurarse de que las expresiones XPath estén actualizadas y coincidan con la estructura actual de la

página web. Esto puede ser complicado de gestionar y puede introducir una capa adicional de complejidad.

Finalmente, los elementos extraídos con Scrapy se almacenan en una base de datos no estructurado, en este caso, MongoDB en lugar de una base de datos estructurada, como SQL, debido a la naturaleza de los datos que se extrayeron. En este sentido, no es necesario definir la estructura de la base de datos de antemano permitiendo datos semi-estructurados o no estructurados, como texto libre, listas de elementos variables y otros formatos diversos. MongoDB es adecuado para manejar este tipo de datos, ya que permite almacenarlos sin necesidad de una estructura fija y se lo puede escalar horizontalmente siendo de gran utilidad para proyectos de scraping que pueden involucrar grandes volúmenes de datos, solucionando la tarea de almacenar datos extraídos en MongoDB.

De acuerdo con la inserción de datos se la creación de la colección de datos, donde se ingresan todas las características extraídas mediante Web Scraping con el fin de obtener un Dataset para entrenar directamente al modelo de recomendación, con una estructura de datos ilustrado en el **Anexo H**, de esta forma se ingresan los datos. En secuencia, se estableció la conexión a MongoDB en el archivo

A continuación, antes de insertar los datos en medio del proceso Web Scraping, se valida la inexistencia de etiquetas HTML mediante Xpath, el cual, implica a la obtención de una de las características vacías, por lo tanto, se elimina el item negando a la inserción del modelo a la base de datos, ya que, las etiquetas establecidas con Xpath están generalizadas según el calzado deportivo que dispone la marca mediante características importantes que se analizaron en las páginas de cada marca extraídas, por ello en el **Anexo I**, se valida mediante expresiones regulares resultantes características como “precio”, “calificacion”, “fecha” y “tallas”. También se implementó una validación de modelos duplicados extraídos, donde se establezca una actualización del modelo anterior con el modelo extraído, y en caso contrario se ingrese como nuevo modelo.

En este sentido, para implementar la inserción de datos extraídos a MongoDB, se implementó el uso de la biblioteca pymongo y parámetros de configuración del Framework Scrapy en el archivo settings.py ilustrado en el **Anexo J** en la clase en los

parámetros “MONGODB_SERVER”, “MONGODB_PORT”, “MONGODB_DB” y “MONGODB_COLLECTION”, también, de acuerdo con las clases de configuración implementada en el archivo pipelines.py “MongoPipeline”, el cual, está ingresado en el parámetro “ITEM_PIPELINES”.

Se implementó un algoritmo de Web Scraping utilizando Scrapy para recopilar información detallada sobre precios, modelos, tallas y valoraciones de clientes ante modelos disponibles de calzado deportivo en tiendas en línea. Este algoritmo se diseñó para extraer datos específicos relacionados con la calificación de los productos, aprovechando las reseñas de los usuarios para contextualizar la calidad de cada modelo. Este enfoque permitió obtener una visión completa de las características de los productos, incluidas las valoraciones que otorgan los clientes ante las reseñas, lo que facilitó la evaluación de la calidad y la toma de decisiones informadas por parte de los usuarios.

3.3.2 Desarrollo del Motor de Sugerencias

En esta sección se establece la decisión de establecer un modelo de recomendación en base a las características obtenidas, por lo tanto, en la **Tabla 25** se establece una comparación entre los modelos de recomendación de diferentes algoritmos de recomendación, para generar recomendaciones personalizadas para cada usuario.

Tabla 25. Comparación de modelos de recomendación

Características	Colaborativo	Basado en Contenido	Híbrido	Ponderación individual por característica			Ponderación máxima por característica
				Colaborativo	Basado en Contenido	Híbrido	
Datos Requeridos	Necesita historial de usuario y preferencias similares de múltiples usuarios.	Puede trabajar con datos del usuario y características del elemento.	Combina datos de usuarios y elementos.	0	2	1	3
Personalización	Bueno para usuarios con historiales de actividad ricos.	Puede ser más efectivo para usuarios nuevos o con historiales limitados.	Combina la fuerza de la personalización de ambos enfoques.	0	2	1	3
Naturaleza del Modelo	Basado en relaciones de usuario a usuario.	Basado en características de elementos y preferencias del usuario.	Puede ser basado en usuario, elemento o factor.	0	2	1	3
Manejo de Datos Faltantes	Dificultad con usuarios o elementos nuevos sin historial.	Puede manejar datos faltantes mejor, especialmente en sistemas de contenido amplio.	Depende de cómo se implemente, puede tener ventajas en manejar datos faltantes.	0	3	0	3
Serendipia	Bueno para descubrimiento serendípico al basarse en comportamientos de otros usuarios.	Menos propenso a descubrimiento serendípico debido a la dependencia de características conocidas.	Puede incorporar elementos serendípicos, dependiendo de la naturaleza de la combinación.	2	0	1	3

Algoritmos del modelo	Filtrado colaborativo basado en usuario o artículo, SVD, modelos de factorización.	Modelos basados en contenido: TF-IDF, Embeddings, Árboles de decisión.	Enfoques híbridos pueden incluir técnicas de colaborativo y basado en contenido.	1	1	1	3
Modelos Matemáticos	Filtrado colaborativo basado en memoria:	Modelos de contenido basados en regresión lineal:	Modelo híbrido ponderado:	1	1	1	3
Evaluación	Métricas como RMSE, MAE para evaluar la precisión de las predicciones.	Métricas como precisión, exhaustividad y F1-score para evaluar la calidad de las recomendaciones.	Dependerá de la naturaleza específica del modelo híbrido.	0	2	1	3
Implementación	Necesita manejar grandes conjuntos de datos y problemas de escalabilidad.	Puede ser más sencillo de implementar en algunos casos.	La implementación puede ser compleja según la combinación de enfoques.	0	2	1	3
			Ponderación individual total	4	17	9	30

3.3.3 Construcción del modelo de recomendación

Tras examinar los artículos pertinentes y realizar una comparación detallada, resaltada en la **Tabla 4** para la revisión general y en la **Figura 13** para la comparativa específica de modelos de recomendación, se observa que el modelo de filtrado basado en contenido emerge como la opción más propicia para la implementación de este proyecto. Este modelo se revela idóneo, dado que, analiza datos que recomiendan productos directamente en función de las necesidades del usuario, prescindiendo de la exigencia de un historial del usuario, considerando que el dataset contiene información de productos y no de historial de usuarios. Además, el modelo de filtrado basado en contenido opera en sintonía con los requisitos expresados por el usuario

En consecuencia, en este proyecto se lleva a cabo la implementación del modelo de filtrado basado en contenido. Utilizando el dataset basado en los datos obtenidos con Web Scraping, se tiene la capacidad de recomendar el producto detallado al usuario, considerando las características que el usuario seleccione. Este enfoque garantiza que las recomendaciones generadas estén directamente alineadas a las necesidades individuales del usuario.

En base a la implementación del modelo de recomendación basado en contenidos, se destacó el uso de bibliotecas de Scikit-learn, debido a que incluyen métodos pertinentes para tareas de clasificación y regresión asociadas con sistemas de recomendación. La biblioteca ofrece una documentación clara, facilitando la experimentación y selección de los algoritmos más adecuados para el motor de sugerencias. Además, la capacidad de Scikit-learn para integrarse con herramientas de preprocesamiento de datos es esencial, permitiendo realizar manipulaciones efectivas en los datos de contenido antes de aplicar los algoritmos. Su implementación eficiente en Python garantiza un rendimiento óptimo, crucial para el procesamiento eficiente de grandes conjuntos de datos inherentes a sistemas de recomendación. La comunidad activa y el mantenimiento continuo respaldan la confiabilidad y actualización constante de la biblioteca, lo que consolida la elección de Scikit-learn para el desarrollo del modelo de sugerencias basado en contenidos.

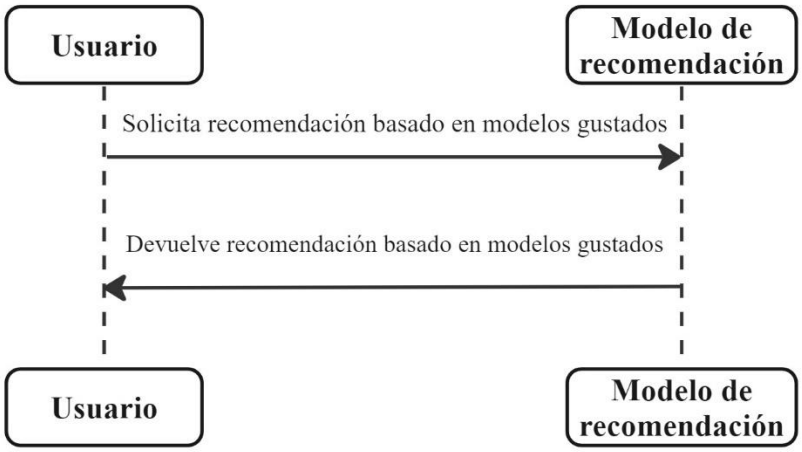


Figura 17. Diagrama de secuencia del motor de sugerencias

En base a la lógica implementada en la **Figura 17**, inicialmente el usuario selecciona modelos de calzado deportivo de su interés, el cual, son enviados al modelo de recomendación basado en contenidos. Además, antes de poder generar la recomendación, se lleva a cabo el Procesamiento del Lenguaje Natural (NLP) utilizando el método TF-IDF. Este proceso permite analizar y entender el contenido textual asociado con cada modelo de calzado.

Para ello, se define la conexión a MongoDB utilizando la biblioteca pymongo, donde, se establece una consulta de todos los calzados deportivos almacenados según el procedimiento de extracción con Scrapy establecido anteriormente. En este contexto, los datos obtenidos se almacenan en un objeto DataFrame para su posterior procesamiento, donde, los atributos relevantes como "color", "modelo", "descripción" y "precio" se combinan en un DataFrame único para facilitar el NLP.

De esta forma, para realizar el análisis de NLP, se utiliza el método TF-IDF mediante la biblioteca scikit-learn, específicamente el componente TfidfVectorizer. Este método procesa automáticamente el texto del DataFrame combinado anteriormente y calcula las puntuaciones TF-IDF para cada término. Estas puntuaciones indican la importancia relativa de cada palabra dentro del conjunto total de documentos, para ello, se utilizan parámetros específicos, como "stop_words", que elimina palabras comunes pero irrelevantes como artículos, pronombres y preposiciones durante la tokenización del texto. Esta eliminación simplifica el procesamiento y evita sesgos en las frecuencias de las palabras.

Además, se emplearon los parámetros *"min_df"* y *"max_df"* para establecer límites en la frecuencia de los términos. **"min_df"** determina la frecuencia mínima que debe tener un término en los documentos para ser considerado relevante, filtrando términos poco comunes que no se consideran representativos del contenido general. Por otro lado, **"max_df"** establece la frecuencia máxima de un término en los documentos como una fracción del tamaño total de documentos. Esto evita que términos muy frecuentes, que podrían no ser informativos debido a su alta presencia en la colección de textos, afecten el cálculo del TF-IDF.

Posteriormente, se utiliza el método de Similitud por Cosenos para establecer la similitud más cercana a los calzados seleccionados por el usuario, que facilita de la misma forma la biblioteca Scikit-learn específicamente denominado “linear_kernel” para calcular la similitud entre el modelo de calzado seleccionado por el usuario y otros modelos de calzado disponibles en el conjunto de datos. Este cálculo se realiza utilizando la matriz de similitud por coseno, la cual se genera mediante el método TF-IDF utilizando el método “fit_transform” del objeto TfidfVectorizer, lo que convierte el texto de los modelos de calzado en una representación numérica que refleja su importancia relativa y la función “linear_kernel” aplicada a la matriz TF-IDF. La matriz de similitud por coseno se basa en características que se tomaron en cuenta en el DataFrame, donde se combinaron las características "color", "modelo", "descripción" y "precio" de los modelos de calzado. De esta manera, el método de Similitud por Cosenos proporciona una medida cuantitativa de la similitud entre el calzado seleccionado por el usuario y otros modelos disponibles en función de estas características, lo que permite establecer recomendaciones de modelos de calzado deportivo similares retornando puntuaciones, evaluadas en una escala de 0 a 1, el cual, indica cuán similares son los calzados deportivos entre sí en función de las características consideradas, como el color, el modelo, la descripción y el precio, donde la mayor puntuación indica una mayor similitud, tal y como se demuestra el desarrollo en el **Anexo G**.

En secuencia, con el resultado del método TF-IDF y la matriz de similitud por cosenos, se desarrolla un método para implementar el modelo de recomendación basado en contenidos denominado “get_recommendations”. Donde, este método toma como argumentos una lista de modelos de calzado deportivo identificados por su propiedad

"id" seleccionados por el usuario, y utiliza la relación de similitud entre los términos extraídos previamente para generar recomendaciones de modelos similares.

Asimismo, en la implementación del método “get_recommendations” se calcula la similitud promedio entre los productos seleccionados y todos los demás productos en el conjunto de datos, luego se ordenan los calzados deportivos según esta similitud promedio y se selecciona los 60 modelos de calzados deportivos más similares por modelos, excluyendo el propio producto seleccionado para evitar redundancias.

El criterio utilizado para determinar la cantidad mínima de productos seleccionados para que la recomendación sea válida es dos, ya que permite capturar una diversidad mínima de preferencias permitiendo generar sugerencias basadas en al menos dos puntos de referencia al usuario, también permite mantener la estabilidad del algoritmo, ya que, al obtener dos modelos, se puede calcular una similitud promedio suficientemente representativa para generar recomendaciones significativas, e incluso permite preservar la eficiencia computacional del mismo, ya que, el cálculo de similitudes se realiza entre pares de productos y el número de combinaciones posibles aumenta significativamente con el número de productos seleccionados. En este sentido, se genera un DataFrame con las recomendaciones, incluyendo la información de los productos y sus puntajes de similitud, y los ordena por marca en orden ascendente y puntaje de similitud en orden descendente devolviendo un objeto JSON que contiene la información de los productos recomendados, incluyendo sus identificadores, características y puntajes de similitud para validar los datos en el desarrollo mostrados en el **Anexo K**.

De esta forma, se logró establecer el algoritmo del modelo de recomendación basado en contenidos sobre modelos de calzados deportivos que el usuario haya seleccionado, brindando recomendaciones personalizadas sobre sus gustos.

3.3.4 Desarrollo de interfaz del usuario

Para implementar el motor de sugerencias, en esta sección se establece el desarrollo de una interfaz intuitiva para el usuario, en el que se pueda establecer recomendación basados en los contenidos interactuados con el usuario.

Tabla 26. Comparación de Frameworks Frontend

Características	React	Angular	Vue.js	Justificación
Facilidad de aprendizaje	Alta	Moderada	Moderada	React es conocido por su curva de aprendizaje amigable.
Reactividad	Alta	Alta	Alta	Todos los frameworks son reactivos y actualizan la interfaz automáticamente.
Comunidad y Soporte	Amplia	Amplia	Amplia	React, Angular y Vue.js tienen comunidades activas y un buen soporte.
Rendimiento	Eficiente	Eficiente	Eficiente	Cada framework tiene un rendimiento eficiente en diferentes contextos.
Herramientas de Desarrollo	Abundantes	Abundantes	Abundantes	React, Angular y Vue.js tienen una amplia variedad de herramientas de desarrollo.

En base a los resultados de la **Tabla 26**, se optó por el uso del Framework React, ya que, su curva de aprendizaje no es muy elevada y la documentación al respecto, es actualizada, según la comparativa establecida en la Tabla 26.

Por lo tanto, para interconectar el proceso de Scrapy con el usuario mediante React, se establece la selección de un proyecto Back-end para establecer servicios de recomendación, SocketIO y consultas a MongoDB, para ello se establece la siguiente comparación.

Tabla 27. Comparación entre Frameworks Back-end

Característica	Flask	Django	Express.js	Justificación
Facilidad de configuración	Sencilla	Moderada	Sencilla	Flask es conocido por su simplicidad y facilidad de configuración.
Integración con bases de datos	Buena	Excelente	Buena	Flask y otros frameworks tienen buenas integraciones con diferentes bases de datos.
Rendimiento	Eficiente	Eficiente	Eficiente	Cada framework tiene un rendimiento eficiente en diferentes contextos.
Escalabilidad	Moderada	Buena	Buena	Flask es adecuado para aplicaciones pequeñas a medianas, mientras que otros frameworks pueden escalar mejor para aplicaciones más grandes.
Lenguaje de programación	Python	Python	Javascript	El lenguaje detonante es Python, debido a que es enfocado para tareas intensivas de CPU, mientras que JavaScript es más adecuado para interacciones dinámicas y en tiempo real.

Según las especificaciones de la **Tabla 27**, se optó por usar como Back-end Flask, dado que su lenguaje principal es Python, y para el sistema de recomendación basado en contenidos implementado, es beneficiosos por las bibliotecas implementadas y rapidez, en comparación con los demás, es adecuado para aplicaciones pequeños.

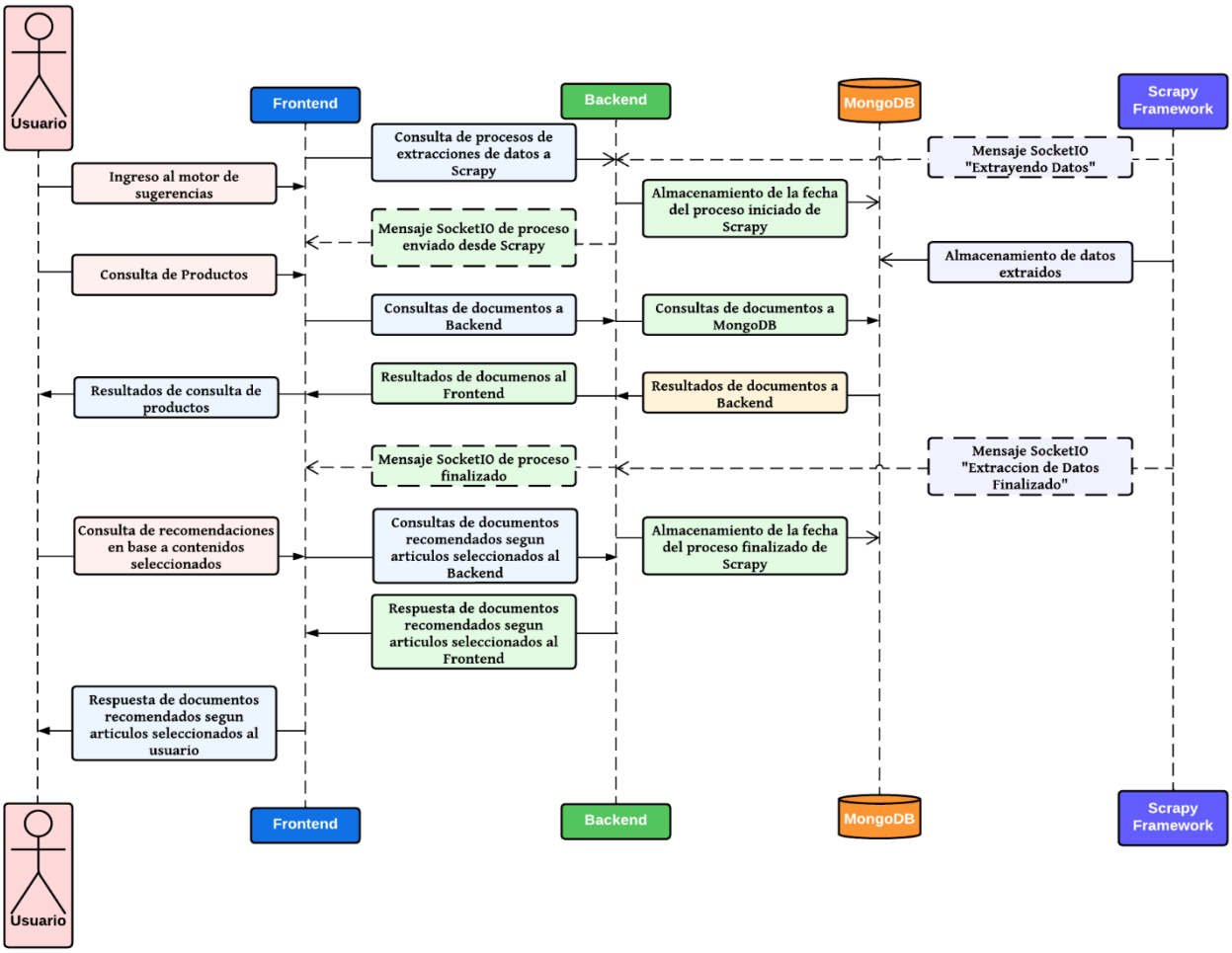


Figura 18. Diagrama de secuencia del motor de sugerencias

La **Figura 18** muestra el proceso en background, donde, el usuario establece una interacción con la interfaz Front-end con React, presentándole todos los productos almacenados en MongoDB después de ejecutar el proyecto Scrapy. Este enfoque facilita la visualización de consultas y recomendaciones de modelos mediante el Back-end con Flask. De esta forma, el usuario establece una interacción de forma intuitiva y comprensible en el motor de sugerencias.

- ***Front-end***

En esta sección, se llevó a cabo la implementación del proyecto en React con el propósito de brindar al usuario una representación visual clara de todos los modelos

obtenidos a través del Web Scraping. Además, se ha incorporado la funcionalidad que permite al usuario seleccionar varios modelos de calzado deportivo con un doble clic desde una computadora o con un solo clic desde un dispositivo móvil, tal y como, se encuentra implementado en el

Anexo L. Adicionalmente, se ha integrado la visualización de los calzados deportivos utilizando carruseles mediante la biblioteca react-slick. Este componente posibilita la visualización de la lista de imágenes asociadas a cada modelo de calzado deportivo obtenido desde la base de datos. Esta implementación ofrece al usuario una experiencia visual más detallada, mostrando las imágenes correspondientes a cada modelo de calzado deportivo para cada marca extraída de la página oficial.

En cuanto al diseño de la Interfaz de Usuario (UI), se ha optado por utilizar el framework basado en CSS conocido como Tailwind CSS. La elección de esta herramienta se debe a su facilidad para implementar diseños, cuadrículas y paletas de colores de forma rápida y sencilla.

Este proceso no solo facilita la visualización de los modelos al usuario más enriquecida aprovechando los datos extraídos, sino que también establece consultas a la API del Back-end, específicamente el modelo seleccionado. En secuencia, se han proporcionado opciones para que el usuario pueda ver recomendaciones basadas en la propiedad 'id' del modelo seleccionado. Asimismo, se ha integrado la funcionalidad de brindar recomendaciones, como se detalla en el **Anexo M** adjunto.

En consecuencia, se ha incorporado el uso de la recepción de un mensaje mediante SocketIO para notificar al usuario cuando el proyecto Scrapy esté en ejecución. Debido a la funcionalidad de permitir actualizar dinámicamente el estado de todos los modelos, reflejando así los modelos de calzado deportivo más recientes en tiempo real. Este aspecto se desarrolla en detalle en el **Anexo N** correspondiente, el cual, este componente proporciona una manera de notificar al usuario sobre el estado del proceso de extracción de datos y le permite cerrar la alerta si lo desea. Esta funcionalidad permite al usuario estar informado sobre las actualizaciones en tiempo real del proyecto Scrapy, pero cerrar la alerta no afecta el proceso de extracción de datos en segundo plano, ya que este continuará ejecutándose independientemente de la interacción del usuario con la alerta. Una vez finalizado el proceso de extracción de

datos, se enviará una nueva notificación al usuario informando que la actualización de datos se ha completado.

En este contexto, cabe destacar que esta sección reviste una importancia fundamental para el usuario, ya que facilita la interacción con el motor de sugerencias y garantiza la actualización en tiempo real de los modelos de calzado deportivo, contribuyendo así a una experiencia más enriquecedora y relevante, permitiendo ahorrar tiempo en la búsqueda del calzado deportivo ideal para el usuario.

- ***Back-end***

Por lo tanto, en esta sección, se establece el proceso de servicios mediante una Interfaz de Programación de Aplicaciones (API), para establecer la lógica de negocio de la aplicación, en este contexto, para la establecer la solución de peticiones que establezca el proyecto Front-end.

De esta manera, se inicializó el proyecto mediante Flask Framework con Python, el cual, en la configuración del app.py, se detalla la comunicación mediante SocketIO para la configuración de mensajería entre el proyecto Scrapy, tal y como se lo detalla en el **Anexo O**.

Después, según las rutas creadas en el proyecto se establecieron consultas para obtener modelos y especificaciones de calzados deportivos, para que así, brindar consultas de documentos necesarios a MongoDB, de esta forma, se establece una interpretación de los modelos extraídos y almacenado en la base de datos no estructurada más operable para cualquier lenguaje de programación, ya que, los resultados se visualización en formato JSON, dicho proceso se encuentra detallado en el **Anexo P**, donde establece la obtención de mejores productos por marcas, de acuerdo con la calificación de reseñas extraídas que se obtuvo del modelo del calzado deportivo.

De esta manera, creada la ruta para las consultas especificadas para cada una según los modelos extraídos, se insertó la ruta para establecer el servicio del modelo de recomendación, el cual, recibe como argumento una lista de id´s sobre los productos encantados por el usuario, y así brindar recomendaciones, anteriormente establecido en la construcción del modelo de recomendación en el **Anexo K**, pero acoplado en

Flask con las mismas bibliotecas de análisis y procesamiento de datos, tal y como, se lo demuestra en el **Anexo Q**.

De esta manera se estableció una forma más estructurada y segmentada, para separar procesos en el que se pueda aprovechar los datos extraídos y de esta forma visualizar al usuario, según sus especificaciones.

- ***MongoDB***

En esta sección, se almacenan todos los documentos de cada modelo de calzado deportivo, los cuales, son extraídos directamente desde el proyecto Scrapy. Este proyecto, a su vez, proporciona respuestas a las consultas realizadas desde el Back-end.

De esta manera, el uso de esta base de datos no estructurada juega un papel crítico debido a la naturaleza de los datos generados por el proceso Scrapy. Debido a los datos que se agrupan en conjuntos significativos, por lo tanto, la flexibilidad dinámica de almacenamiento de MongoDB resulta esencial. En este contexto, esta característica ahorra considerablemente el tiempo, ya que no se requiere la creación de tablas y referencias típicas de bases de datos estructuradas. En el contexto de este proyecto, esta elección se presenta conveniente. De esta forma, al evitar la rigidez de las bases de datos estructuradas, MongoDB permite adaptarse a la variedad de datos provenientes del Scrapy de manera eficiente. Este enfoque no solo simplifica el proceso de almacenamiento, sino que también mejora la eficacia al eliminar la necesidad de ajustarse a un esquema predeterminado.

En consecuencia, la elección de una base de datos no estructurada como MongoDB para este proyecto se justifica por su capacidad para manejar datos dinámicos de manera eficiente, contribuyendo así a una implementación más ágil y efectiva.

- ***Scrapy Framework***

En esta sección, se establece la automatización de extracción de datos, el cual se encarga en informar al usuario del proceso inmediato, debido a la importancia de brindar información actualizada y relevante.

En este contexto, se estableció un documento que permita la comunicación al usuario, el cual, se conecta directamente al servidor Back-end, y este les informe a todos los usuarios que se conectaron al servicio, tal y como se lo demuestra en el **Anexo R**.

- ***Usuario***

En consecuencia, a todo el proceso en background que el usuario no visualiza, pero está presente según la utilización de la interfaz del usuario, se implementan de acuerdo con la solicitud de varios modelos en base a la marca establecida por el usuario y de este modo poder reconocer los modelos que le agraden, para que, en este contexto se brinde una recomendación personalizada.

3.4 Funcionalidad del motor de sugerencias

En esta sección se establece la visualización de la funcionalidad del motor de sugerencias que se basa con datos extraídos diariamente en páginas web de venta online sobre calzado en la línea deportiva.

Al iniciar el motor de sugerencias, se le permite visualizar al usuario una introducción sobre la funcionalidad del motor de sugerencias en la **Figura 19**. Se puede observar un menú en el lado izquierdo, donde, se puede seleccionar cualquier marca en el botón de **Marcas Registradas**, asimismo, en la parte inferior del recuadro blanco de información, se puede acceder a las diferentes marcas reconocidas por el usuario dando clic a cualquiera de los iconos representativos. ¡Error! No se encuentra el origen de la r eferencia.

Figura 19. Página principal del motor de sugerencias

En consecuencia, después de la selección de la marca deseada, se visualizan todos los modelos extraídos posteriormente. Esto permite al usuario tener una variedad de opciones categorizados como “Nuevos productos” como se muestra en la **Figura 20** y “Mejores calificados” según la **Figura 20**. También, está implementado un filtrado de modelos según un rango de precios requeridos por el usuario y nombres de modelos, para simplificar la búsqueda y visualizar todos los modelos que ofrece la marca en la en base a la **Figura 22**:

Figura 20. Página visual de marcas y modelos de calzado deportivo categorizado por Nuevos productos

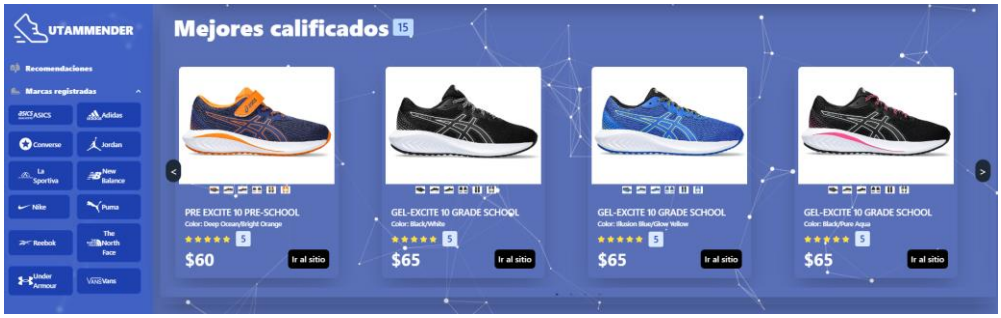


Figura 21. Página visual de marcas y modelos de calzado deportivo categorizado por Mejores calificados

Figura 22. Página visual de marcas y modelos de calzado deportivo por filtros de búsqueda

En base a los modelos reflejados sobre la marca seleccionada por el usuario, en la **Figura 23** se observa la forma en que se establece una animación y diferenciación con un borde rojo y un corazón animado del modelo seleccionado por el usuario, aplicando doble clic desde la computadora o si es en celular con un toque. En consecuencia, los modelos seleccionados son enviados al modelo de recomendación basados en características de modelos seleccionados, para obtener la recomendación solicitada de calzados deportivo. En cambio, si se desea cancelar la selección, hay que establecer la misma acción con doble clic en computadora o un clic en el celular.

Figura 23. Selección de modelos gustados por el usuario

Después de haber seleccionado los productos interesados por el usuario, al dar clic en el botón “Recomendaciones” ubicado en el lado superior izquierdo, se visualiza las diferentes marcas que son recomendados en base a los productos seleccionados, según como se demuestra en la **Figura 24** se observan diferentes marcas recomendadas en base a las características de los modelos seleccionados, y en consecuencia, al dar clic sobre las marcas, se desglosan todos los modelos recomendados como en la **Figura 25**. A todo este proceso, si el usuario no ha seleccionado algún modelo, se le avisará el requerimiento de seleccionar algún modelo, como lo demuestra la **Figura 26**:

Figura 24. Marcas recomendadas, según datos seleccionados por el usuario

Figura 25. Modelos recomendados según la marca

Figura 26. Aviso de selección de modelos requeridos para su recomendación

Como parte de avisos, al implementar la ejecución del proyecto Scrapy, el cual, en consecuencia, a la extracción de datos se visualiza un mensaje de aviso al usuario como se observa en la **Figura 27**, debido a que la información que el usuario visualizará futuramente será actual.

Figura 27. Aviso de actualización de información diaria al usuario.

3.5 Experimentación del motor de sugerencias

En esta sección, se llevó a cabo la experimentación del producto final utilizando una muestra poblacional representativa a 30 personas con experiencia en la venta de calzado deportivo. En este contexto, la muestra se dividió en un grupo de control conformado por 15 personas que no utilizaron la aplicación y otro grupo de investigación de 15 personas que, si utilizaron el motor de sugerencias con el objetivo de determinar si el proyecto investigado y desarrollado, es viable y beneficia a la comunidad de acuerdo con la disminución del tiempo en la toma de decisión de compra de calzado deportivo.

Por lo tanto, para lograr obtener resultados significativos, se optó por emplear fichas de observación de tiempo, para evaluar el tiempo de decisión en la compra de calzado deportivo tanto para el grupo de control como para el grupo de investigación. Las especificaciones detalladas de estas fichas se encuentran disponibles en el

Anexo S. El propósito principal de este enfoque es extraer conclusiones valiosas que resulten claros sobre la eficacia y el impacto del motor de sugerencias en la toma de decisiones de los usuarios.

Tabla 28. Observaciones de tiempo de decisión sin el uso del motor de sugerencias

Búsqueda de calzado deportivo sin motor de sugerencias (Grupo de control)				
Vendedor	**Lugar geográfico**	**Tiempo en minutos**		
		Selección del modelo	**Selección de la marca**	**Decisión de compra**
1	Avenida Cevallos	5	3	10
2	Avenida Cevallos	30	10	40
3	Avenida Cevallos	5	2	8
4	Avenida Cevallos	5	2	7
5	Avenida Cevallos	25	1	30
6	Avenida Cevallos	10	3	20
7	Avenida Cevallos	75	60	80
8	Calle Montalvo	15	5	20
9	Calle Montalvo	30	20	40
10	Calle Montalvo	33	17	45
11	Sector del mercado modelo de Ambato	15	10	22
12	Sector del mercado modelo de Ambato	10	2	20
13	Calle Montalvo	30	5	40
14	Sector del mercado modelo de Ambato	20	5	30
15	Sector del mercado modelo de Ambato	25	15	35

Tabla 29. Observaciones de tiempo de decisión con el uso del motor de sugerencias

Búsqueda de calzado deportivo con motor de sugerencias (Grupo de investigación)				
Vendedor	**Lugar geográfico**	**Tiempo en minutos**		
		Selección del modelo	**Selección de la marca**	**Decisión de compra**
16	Sector de Juan Cajas	2.00	1.60	2.28
17	Sector de Juan Cajas	5.05	4.45	5.40
18	Sector de Juan Cajas	3.25	2.72	3.50
19	Sector de Juan Cajas	1.70	1.58	2.10
20	Sector de Juan Cajas	2.30	1.90	2.43
21	Avenida Cevallos	2.07	1.80	2.62
22	Sector de Juan Cajas	4.32	3.93	4.60
23	Sector de Juan Cajas	4.12	3.65	4.58
24	Sector de Juan Cajas	2.55	2.33	2.78
25	Calle Montalvo	2.55	2.30	3.27
26	Avenida Cevallos	3.10	2.18	3.20
27	Sector de Juan Cajas	5.23	4.55	5.53
28	Sector de Juan Cajas	2.20	1.95	2.47
29	Avenida Cevallos	3.03	2.73	3.27
30	Avenida Cevallos	2.70	2.22	3.03

Los resultados de las fichas de observación de tiempo se pueden observar de forma resumida en la **Tabla 28** con el grupo de control y **Tabla 29** con el grupo de investigación, la diferencia de tiempo en la decisión de compra de calzado deportivo entre ambos grupos, mostrando una reducción significativa en el tiempo total cuando se utiliza el motor de sugerencias.

En evidencia a la obtención de resultados, para profundizar la comprensión de la funcionalidad del proyecto, es importante llevar a cabo un análisis detallado e interpretación de los resultados obtenidos de las fichas de observación de tiempo. De esta forma, al entender la forma en que el uso de esta tecnología impacta positivamente en la eficiencia del tiempo de decisión, se podrá respaldar de manera más sólida la relevancia y eficacia del proyecto en el contexto de la experiencia del usuario y la mejora del proceso de decisión de compra de calzado deportivo.

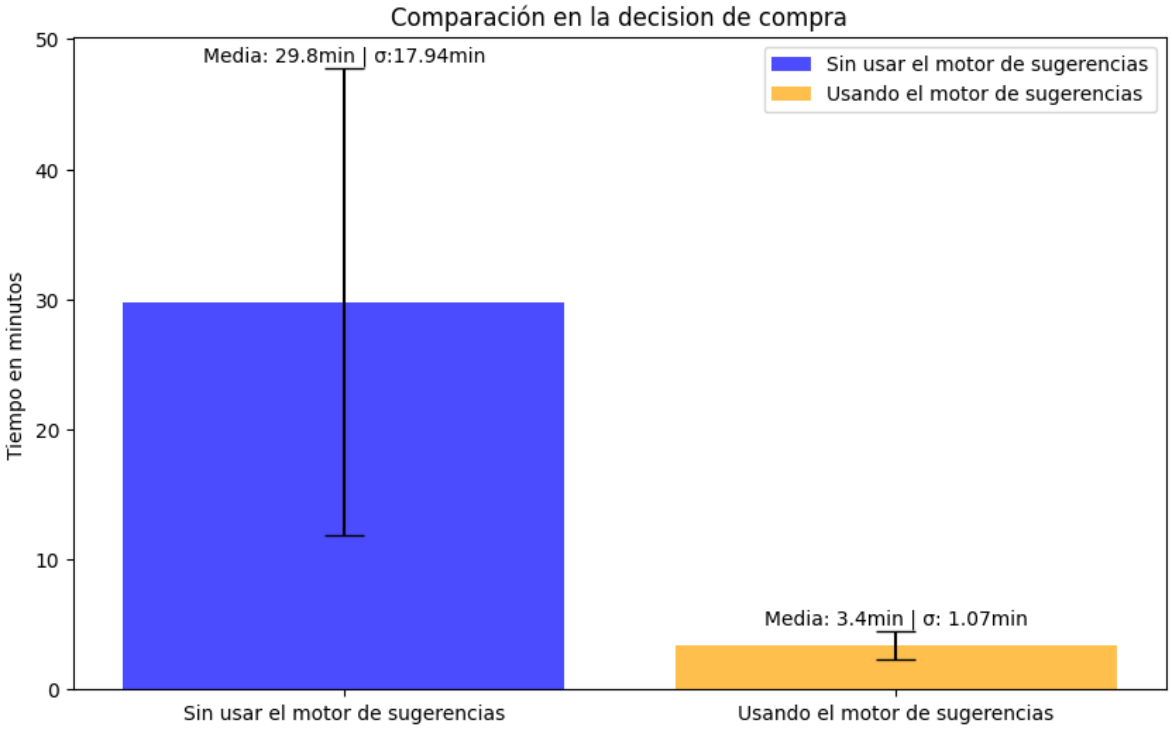


Figura 28. Gráfico de barras de cada tiempo de decisión de compra estimado por el vendedor

Por lo tanto, según los resultados de la gráfica en la **Figura 28** se muestra visualmente la comparación en la decisión de compra entre los dos escenarios: "Sin usar el motor de sugerencias" siendo el grupo de control y "Usando el motor de sugerencias" como el grupo de investigación. En consecuencia, la comparación de la media de ambos

grupos demuestra que los tiempos de decisión tienden a ser más altos sin el uso del motor de sugerencias, en cambio, el tiempo de decisión con el uso del motor de sugerencias es significativamente menor. Además, se puede observar en base a la desviación estándar obtenida en el grupo de investigación, el tiempo de búsqueda y decisión de compra ronda entre los 2.3 minutos y 4.47 minutos, por otra parte, en el grupo de control donde no usan el motor de sugerencias, rondan entre los 47.94 minutos y 11,66 minutos en buscar y tomar una decisión sobre la compra de calzado deportivo.

En términos prácticos, la implementación del motor de sugerencias ha demostrado ser efectiva en la reducción del tiempo necesario para que los usuarios tomen decisiones en la compra de calzado deportivo. Esto se evidencia en la comparación entre el grupo que utilizó el motor de sugerencias y el grupo de control, donde se observó una disminución significativa en el tiempo de decisión. La evaluación experimental se centró en la calidad de la decisión de compra de calzado deportivo, evaluando tanto la satisfacción de los modelos recomendados al usuario como su preferencia al momento de informar su decisión de compra. Además, se analizó el tiempo de decisión de compra del usuario y la disminución de tiempo durante la búsqueda de modelos de calzado deportivo. Este enfoque cumplió con el tercer objetivo específico al analizar detalladamente el tiempo requerido por los usuarios para tomar decisiones, tanto con el motor de sugerencias como sin él, y al evaluar la eficacia del motor de sugerencias en la mejora del proceso de selección de calzado deportivo. La experimentación proporcionó información valiosa sobre la utilidad y el impacto del motor de sugerencias en la experiencia de compra de los usuarios, demostrando que el proyecto es funcional y ayuda a la comunidad a tomar decisiones en menos tiempo.

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- La extracción de características se completó con éxito, pero presentó desafíos notables, especialmente en la interpretación detallada de páginas web dinámicas. Donde, algunos elementos que se renderizaban dinámicamente requerían enfoques alternativos, como la obtención de archivos JSON cuando la extracción por XPath resultaba inviable. Por lo tanto, se optó por inspeccionar los consumos Fetch/XHR de la página web mediante la gestión de red del navegador, para obtener el consumo API REST y usar como fuente de características el objeto json resultante. En cambio, al no hallar dicho consumo de la página, por última opción fue renderizar los elementos dinámicos nuevamente con Selenium. A pesar de que este enfoque permitió resolver el problema, fue evidente que debido a la naturaleza más intensiva del proceso el tiempo empleado fue mayor.

- Después de obtener datos extraídos mediante Web Scraping, se analizaron características importantes para la implementación del modelo de recomendación. Inicialmente, se consideró la implementación de un modelo de recomendación basado en filtrado colaborativo, sin embargo, esta opción se descartó debido a la necesidad de contar con un historial de interacciones de los usuarios con modelos específicos de calzado deportivo. En su lugar, se implementó un modelo de recomendación basado en contenidos, debido a la posibilidad de proporcionar recomendaciones personalizadas según los requisitos del usuario. Esta elección resultó en una implementación satisfactoria del modelo de recomendación para el motor de sugerencias.

- La implementación de React para la interfaz de usuario requería la integración con Flask para facilitar la conexión con Scrapy y garantizar la actualización constante de datos diariamente en el motor de sugerencias. Esta integración permitió la extracción diaria de datos, asegurando la visualización e interacción de la información más reciente en la interfaz del usuario.

- En conclusión, el desarrollo del motor de sugerencias otorga una ayuda sustancial para los vendedores de calzado deportivo, según los resultados de las fichas de observación establecido en la fase de experimentación. En consecuencia, los vendedores reflejaron una considerable reducción en el tiempo de decisión al utilizar el motor de sugerencias para la decisión de compra de calzado deportivo. Este impacto positivo subraya la funcionalidad del proyecto, demostrando su utilidad y contribución beneficiosa a la comunidad involucrada.

4.2 Recomendaciones

- Para enriquecer la oferta de calzado deportivo recomendado, se sugiere ampliar la extracción de datos en modelos con más marcas. Esto proporcionará a los usuarios una mayor variedad de opciones, mejorando así la relevancia de las recomendaciones y brindando una experiencia de compra más completa para el usuario.

- Es recomendable implementar un motor de sugerencias con usuarios. Esto permitirá la recopilación y análisis del historial de interacciones de múltiples usuarios, facilitando la adopción de un modelo de recomendación predictivo. Dicha implementación asegurará recomendaciones de calzado deportivo más precisas, personalizadas y sorpresivamente gustosas para el usuario.

- Se recomienda expandir la aplicación del motor de sugerencias a productos similares, como ropa u otros artículos con características afines. Esta ampliación no solo diversificará las recomendaciones, sino que también incrementará la utilidad del motor para el usuario, proporcionando sugerencias más variadas y adaptadas a sus preferencias.

- Para potenciar la interacción con los usuarios, se sugiere el desarrollo de una aplicación móvil que aproveche el servicio API previamente desarrollado con Flask. Esto no solo aumentará la accesibilidad del sistema, sino que también proporcionará a los usuarios una experiencia más dinámica e integrada.

REFERENCIAS BIBLIOGRÁFICAS

[1] María José and Murillo Ley Rommel Iván, Rommel Iván Gutierrez Quiroz, "PROYECTO DE PREFACTIBILIDAD PARA LA CREACIÓN DE EMPRESA DISTRIBUIDORA DE ZAPATOS DEPORTIVOS, CON ENTREGA A DOMICILIO, EN LA URB. METRÓPOLIS PARROQUIA PASCUALES, CANTÓN GUAYAQUIL-ECUADOR, ENERO-JULIO DE 2020.," no. Undécimo, p. 143, 2021.

[2] Cristian Ibáñez, "Internet y Aprendizaje," *Comunicaciones En Humanidades*, vol. 2, August 2019.

[3] Katherine Arroyave Jiménez, Vinicio Cevallos Ponce, Shirley Rosario Ponce Merino, and María Leonor Parrales Poveda, "E-commerce: experiencias de usuarios en sus compras por internet," *Revista Publicando*, vol. 8, pp. 225–239, 2021.

[4] Marino Latorre, "Historia de las web, 1.0, 2.0, 3.0 y 4.0," *Universidad Marcelino Champagnat*, vol. 1, 2018.

[5] Cynthia Beatriz Alvarez García, Adriana Nicole Rosales Olivas, and Luis Fernando Lucas Valera Lalangui, "Análisis de la experiencia del usuario en la plataforma web para la compra de calzado deportivo en Runa Store," 2020.

[6] Jerson Erick Herrera Rivera, "Sistema de Recomendación Usando Web Scraping Para Matrícula en MOOCs de Estudiantes en Carrera de Ingeniería en Universidad Pública de Arequipa," *Industry, Innovation, and Infrastructure for Sustainable Cities and Communities: Proceedings of the 17th LACCEI International Multi-Conference for Engineering, Education and Technology*, 2019.

[7] David Ralph, Yunjia Li, Gary Wills, and Nicolas G. Green, "Recommendations from cold starts in big data," *Computing*, vol. 102, pp. 1323–1344, January 2020.

[8] Nilesh, Madhu Kumari, Pritom Hazarika, and Vishal Raman, "Recommendation of Indian cuisine recipes based on ingredients," , 2019, pp. 96 – 99, Cited by: 15. [Online]. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85069211925&doi=10.1109%2fICDEW.2019.00-28&partnerID=40&md5=9f64405a7b622805cd5dde511d958aaf

[9] Nathalie Campos Macias, Wilhelm Düggelin, Yesim Ruf, and Thomas Hanne, "Building a Technology Recommender System Using Web Crawling and Natural Language Processing Technology," *Algorithms*, vol. 15, p. 272, August 2022. [Online]. http://dx.doi.org/10.3390/a15080272

[10] Felicia Loecherbach and Damian Trilling, "3bij3–Developing a framework for researching recommender systems and their effects," *Computational Communication Research*, vol. 2, pp. 53–79, 2020.

[11] Juan Fernando Riofrío Valarezo, "Diseño de un sistema de recomendación basado en ganancias que usa machine learning para balancear los beneficios para el usuario y la empresa.," Quito: EPN, 2022., Master's thesis 2022.

[12] García Escudero y Luis Ángel Berrío Galindo, "Sistemas de recomendación," *Universidad de Valladolid. Facultad de Ciencias*, 2020.

[13] Guillermo Valle Gutiérrez-Luis Pita-Romero Rodríguez, "Aplicación de técnicas de web scraping y procesamiento del lenguaje natural para la extracción y evaluación de información de una página web de empleo," *ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)*, vol. 7, pp. 224–240, 2021.

[14] Leonel Cruz Norberto, "Búsqueda automatizada en bolsas de trabajo online con técnicas de web scraping," 2022.

[15] Nicole Luise Vicente Stenhouse, "HabScraper: herramienta automatizada para la extracción de datos con web scraping.," *Universitat de les Illes Balears*, 2018.

[16] Rogelio Mijangos-Espinosa, Alicia Martínez Rebollar, Hugo Estrada-Esquivel, and Yasmín Hernández Pérez, "Uso de técnicas de Web Scraping para obtención automática de bases de datos en la Web.," *Res. Comput. Sci.*, vol. 151, pp. 143–157, 2022.

[17] Mónica María Echeverri Torres and Roberto Manjarrés-Betancur, "Asistente virtual académico utilizando tecnologías cognitivas de procesamiento de lenguaje natural," *Revista Politécnica*, vol. 16, pp. 85–96, 2020.

[18] Marilin Carolina Barrera Echeverría, "Plan de marketing digital para almacenes deportivos El Kimono en la provincia de Imbabura," B.S. thesis 2018.

[19] Rodrigo Gerhardt, "Ventas online: Factores claves de la experiencia de compra," Universidad Siglo 21, B.S. thesis June 2021.

[20] Eric Bárbaro Utrera Sust, Alfredo Simón Cuevas, and José A. Olivas, "Análisis de tendencias en la personalización de los resultados en buscadores web.," *Revista Cubana de Ciencias Informáticas*, vol. 12, pp. 126–146, 2018.

[21] Diego Fernando Palacios Hoyos, "Reconstrucción de interfaz gráfica de usuario basada en tecnología React para aplicación web de mantenimiento de enlaces rotos," Universidad de los Llanos, Ph.D. dissertation 2020.

[22] Ankit Verma, Chavi Kapoor, Abhishek Sharma, and Biswajit Mishra, "Web Application Implementation with Machine Learning," in *2021 2nd International Conference on Intelligent Engineering and Management (ICIEM)*, 2021, pp. 423-428.

[23] Reza Jafari Ziarani and Reza Ravanmehr, "Serendipity in Recommender Systems: A Systematic Literature Review," *Journal of Computer Science and Technology*, vol. 36, pp. 375–396, March 2021. [Online]. https://link.springer.com/article/10.1007/s11390-020-0135-9#citeas

[24] Selenium. (2023) Selenium Documentation. [Online]. https://www.selenium.dev/documentation/

[25] Scrapy. (2023) Scrapy. [Online]. https://docs.scrapy.org/en/latest/intro/overview.html

[26] Cristian Werner García Ramírez and Silvia Karol Sánchez Gárate, "La inteligencia de negocios y la analítica de datos en los procesos empresariales," *Revista científica de sistemas e informática*, vol. 1, pp. 38–53, 2021.

[27] Batta Mahesh, "Machine learning algorithms-a review," *International Journal of Science and Research (IJSR).[Internet]*, vol. 9, pp. 381–386, 2020.

[28] Denniye Hinestroza Ramírez, "Búsqueda automatizada en bolsas de trabajo online con técnicas de web scraping," 2018.

[29] Greace Kelly Diaz Moreno, "Metodologías de gestión de proyectos de sistemas. Una revisión de la literatura científica," 2019. [Online]. https://hdl.handle.net/11537/21929

[30] Crummy. (2023) Crummy: The site. [Online]. https://www.crummy.com/software/BeautifulSoup/bs4/doc/

[31] Luis Cerda-Leiva, Luis Araya-Castillo, and Nicolás Barrientos Oradini, "¿Cuánto se ha avanzado en proporcionar analítica e inteligencia de negocios a las pymes?," *Investigación & Desarrollo*, vol. 19, pp. 167–175, 2019.

[32] Bárbara Bron Fonseca and Omar Mar Cornelio, "SISTEMAS DE RECOMENDACIÓN PARA LA TOMA DE DECISIONES. ESTADO DEL ARTE: SISTEMAS DE RECOMENDACIÓN PARA LA TOMA DE DECISIONES," *UNESUM-Ciencias. Revista Científica Multidisciplinaria. ISSN 2602-8166*, vol. 6, pp. 149–164, January 2022.

[33] Luis Miguel Ruiz Garzón Daniela Apraez Solarte, "Plataforma scrumban para la aplicación y desarrollo de soluciones informáticas," *Fundación Universitaria de Popayán*, 2019.

ANEXOS

Anexo A. Encuesta

Encuesta establecida a personas con experiencia en la venta de calzado en la línea deportiva.

1. ¿Qué tan importante es la marca para los clientes al momento de comprar calzado deportivo?

   1: No es importante

   2: Poco importante

   3: Moderadamente importante

   4: Importante

   5: Muy importante

   1 2 3 4 5

2. ¿Qué tan importante es el precio para los clientes al momento de comprar calzado deportivo?

   1: No es importante

   2: Poco importante

   3: Moderadamente importante

   4: Importante

   5: Muy importante

   1 2 3 4 5

3. ¿Qué tan importante es el modelo para los clientes al momento de comprar calzado deportivo?

   1: No es importante

   2: Poco importante

   3: Moderadamente importante

4: Importante

5: Muy importante

1 2 3 4 5

4. ¿Qué tan importante es el diseño para los clientes al momento de comprar calzado deportivo?

1: No es importante

2: Poco importante

3: Moderadamente importante

4: Importante

5: Muy importante

1 2 3 4 5

5. ¿Cuál es la característica más importante que los clientes solicitan al momento de comprar calzado deportivo?

- ◯ Precio
- ◯ Modelo
- ◯ Marca
- ◯ Talla

6. ¿Cuál es la característica más importante que los proveedores destacan al momento de vender calzado deportivo?

- ◯ Precio
- ◯ Modelo
- ◯ Marca
- ◯ Talla

7. ¿Cuál es el color más popular de calzado deportivo, según su experiencia con sus clientes?

   - ◯ Negro
   - ◯ Blanco
   - ◯ Azul
   - ◯ Rojo
   - ◯ Otro (Especifique el color)

8. ¿Cómo ayuda a los clientes a tomar una decisión de compra?

   - ◯ Permite a los clientes comparar diferentes opciones de calzado deportivo.
   - ◯ Permite a los clientes identificar el calzado deportivo que mejor se adapta a sus necesidades.
   - ◯ Permite a los clientes tomar una decisión de compra más rápida y segura.

9. ¿Cuál es la marca y modelo de calzado deportivo más solicitado a los proveedores, según su experiencia?

   - ◯ Nike Air
   - ◯ Nike Jordan
   - ◯ Adidas
   - ◯ Converse
   - ◯ Puma
   - ◯ Genéricas

10. ¿Qué tan útil es la recomendación del calzado deportivo a la hora de elegir para los clientes al momento de comprar?

    1: No es útil

    2: Poco útil

    3: Moderadamente útil

    4: Útil

    5: Muy útil

    1 2 3 4 5

11. ¿Qué tan probable es que los clientes sigan una recomendación al momento de comprar calzado deportivo?

    1: Muy poco probable

    2: Poco probable

    3: Moderadamente probable

    4: Probable

    5: Muy probable

Anexo B. Validación del instrumento de recolección de datos

Medición de confiabilidad mediante Alpha de Cronbach:

$$\propto = \frac{K}{K-1}\left[1 - \frac{\sum S_i^2}{S_T^2}\right] \quad (7)$$

Donde cada variable significa:

α	es el coeficiente de confiabilidad del cuestionario.
K	es el número de ítems del instrumento.
$\sum S_i^2$	es la sumatoria de las varianzas de los ítems.
S_T^2	es la varianza total del instrumento.

En base a la ecuación del Alpha de Cronbach se establece el resultado de los ítems de la siguiente forma:

Tabla 30. Validación del instrumento de recolección de datos

ENCUESTADOS	ITEM 1	ITEM 2	ITEM 3	ITEM 4	ITEM 11	ITEM 12	SUMA
E1	5	5	4	4	5	4	27
E2	4	5	5	4	5	4	27
E3	5	5	5	5	5	5	30
E4	4	5	5	5	3	4	26
E5	5	3	5	5	4	5	27
E6	4	3	3	2	3	3	18
E7	5	5	5	5	5	4	29
E8	5	5	3	3	4	3	23
E9	5	5	5	4	4	4	27
E10	4	5	5	5	4	4	27
E11	4	5	5	5	5	5	29
VARIANZA	0,248	0,595	0,612	0,926	0,562	0,446	
SUMATORIA DE VARIANZAS	3,388429752						
VARIANZA DE LA SUMA DE LOS ÍTEMS	10,04958678						

Por lo tanto, según los resultados de la **Tabla 30**, se establece el siguiente resultado:

Tabla 31. Resultado de validación del instrumento de recolección de datos

α	Coeficiente de confiabilidad del cuestionario	**0,795**
K	Número de ítems del instrumento	6
$\sum S_i^2$	Sumatoria de las varianzas de los ítems.	3,388
S_T^2	Varianza total del instrumento.	10,050

Según el coeficiente de confiabilidad del cuestionario resultante por el Alpha de Cronbach es de excelente confiabilidad, en base la **Tabla 32** que califica mediante rangos el coeficiente resultante.

Tabla 32. Rangos de confiabilidad de Alpha de Cronbach

RANGO	CONFIABILIDAD
0.53 a menos	Confiabilidad nula
0.54 a 0.59	Confiabilidad baja
0.60 a 0.65	Confiable
0.66 a 0.71	Muy confiable
0.72 a 0.99	Excelente confiabilidad
1	Confiabilidad perfecta

```
    def start_requests(self):
        chrome_options = webdriver.ChromeOptions()
        prefs
={    "profile.default_content_setting_values.notifications":2,
                    "translate_whitelists": {"en":"es"},
                    "translate":{"enabled":"true"}}
        chrome_options.add_experimental_option("prefs",prefs )
        # Establecer a español
        chrome_options.add_argument("--lang=es")
        # chrome_options.add_argument("--headless")
        # Opciones de navegación
        driver = webdriver.Chrome(chrome_options)
        driver.maximize_window()
        driver.get(self.link_raiz)
        ##### ELEMENTOS #####
       # Elemento para obtener links de cada calzado
        # xpath_links = "//div[@class='glass-product-card-container
with-variation-carousel']/div/a"
        # Obtener Objeto para el link y para saber si es nuevo
        xpath_elemntoCard = "//div[@class='glass-product-card color-
variations__fixed-size plp-product-card___2Tf-5 product-card-
content___3R3aL lift-image___3FzoX']/a"
        link_elements =[]
        # cont = 0
        while True:
            # Cerrar modal imprevisto
            banner_selector  = '//*[@id="account-portal-
modal"]/div/div/button'
            # Alternativamente, intenta cerrar el banner presionando la
tecla Escape
            try:
                close_button = driver.find_element(By.XPATH,
banner_selector)
                close_button.click()

            except:
                pass  # Si no se puede cerrar con la tecla Escape,
continúa

            # Si aparece una alerta se deberia cerrar la alerta
            truco.cerrar_alerta(driver)

            # Validar que los productos sean calzados deportivos
            for link in (driver.find_elements(By.XPATH,
xpath_elemntoCard)):
```

```
                print("Vamos a ver la data:")
                texto_html = link.get_attribute("outerHTML")
                soup = BeautifulSoup(texto_html, 'html.parser')
                # Obtener el href del elemento <a>
                href = soup.find('a')['href']
                print(f"href: {href}")
                # Obtener el texto del segundo <span> dentro de <p>
                try:
                    esNuevo = soup.find('p', class_='glass-product-
card__label').find_all('span')[2].get_text()
                    print(f"Texto de 'best seller': {esNuevo}")
                    # time.sleep(20)
                except Exception as e:
                    print('No se hallo etiqueta de nuevo')
                    esNuevo = None
                yield scrapy.Request(href,meta={'esNuevo': esNuevo})

            # Validar si el elemento se pudo dar click
            try:
                # Elemento boton para avanzar
                boton_avanzar = driver.find_element(By.LINK_TEXT,
'NEXT')

                # Clickear
                boton_avanzar.click()

            except StaleElementReferenceException:
                print("Elemento obsoleto. Volviendo a buscar el botón
Next.")
                # continue  # Volver al inicio del bucle para intentar
de nuevo

            except NoSuchElementException:
                print("No hay más botones Next. Saliendo del bucle.")
                break
            except ElementClickInterceptedException:
                print("No se pudo dar click en el elemento")

        driver.quit()
        print(f'# de links: {self.cont_links}')
        print(f'Links: {link_elements}')
```

```
    def parse(self, response):
        # Obtener el HTML de la respuesta
        # html_content = response.body.decode(response.encoding)
        #
truco.save_html_to_file(html_content=html_content,titulo='HTML_Adidas')
        esNuevo = response.meta.get('esNuevo')
        item = ZapatillaItem()
        item['modelo'] =
response.xpath("//div[@class='content___1BnBS']//h1[@class='name___120F
N']//span/text()").get()
        item['marca']  = 'Adidas'
        item['precio'] = response.xpath("//div[@class='product-
price___2Mip5 gl-vspace']/div/div/div/div[1]/text()").get()
        # item['color']  = response.xpath("//div[@class='sidebar-color-
chooser___i7JXW']/div[@class='color-label___2hXaD']/text()").get()
        item['url_raiz'] = self.link_raiz
        item['url_calzado'] = response.url
        # Elemento que contiene el json para obtener rating del
producto
        json_element_rating=
response.xpath("/html/body/script[6]/text()").get().replace("window.DAT
A_STORE = JSON.parse",'')[2:-3]
        # Decodificacion del json para el rating
        json_element_rating=
json_element_rating.encode().decode('unicode-escape')
        json_rating=json.loads(json_element_rating)
         # Elemento que contiene el json para obtener tallas y
descripcion del producto
        json_element =
response.xpath("//body/script[7]/text()").get().replace('window.REACT_Q
UERY_DATA = ','')
        data_json=json.loads(json_element)
        tallas = truco.find_jsonLabel(obj=data_json,etiqueta='size')
        imagenes =
truco.find_jsonLabel(obj=data_json,etiqueta='image_url')
        color = truco.find_jsonLabel(obj=data_json,etiqueta='color')[0]
        # color =
truco.find_jsonLabel(obj=data_json,etiqueta='reviewCount')[0]
        calificacion
=  truco.find_jsonLabel(obj=json_rating,etiqueta="overallRating")

        print('DATA JSON:')
        print(calificacion)
        # item['descripcion'] = response.xpath("//div[@class='text-
content___13aRm']/h3/text()").getall()
```

```
        item['descripcion'] =
str(truco.find_jsonLabel(obj=data_json,etiqueta='subtitle')[0]).strip()

        # print(f'Color calzado: {color}')
        item['tallas'] =  (truco.limpiar_lista(tallas))
        item['imagenes'] = imagenes
        item['color']  = color
        if esNuevo:
            item['calificacion'] = -1 if str(esNuevo).lower() == 'new'
else calificacion[0]
        else:
            if len(calificacion)>0:
                item['calificacion']  = calificacion[0]
            else:
                item['calificacion']  = 0
        print(item['calificacion'] )
        yield item
```

Anexo E. Obtención de características mediante documento JSON

```
    def find_jsonLabel(obj, etiqueta):
        """
        Función para encontrar todas las ocurrencias de una etiqueta en
un objeto JSON de forma recursiva.

        Args:
            obj (dict o list): Objeto JSON o lista en la que se buscará
la etiqueta.
            etiqueta (str): Etiqueta que se desea encontrar en el
objeto JSON.

        Returns:
            list: Lista de todos los valores asociados a la etiqueta
encontrada en el objeto JSON.
        """

        resultado = []  # Lista para almacenar los resultados
encontrados

        def buscar_recursivamente(obj):
            """
            Función interna para buscar la etiqueta de forma recursiva
en el objeto JSON.

            Args:
```

```
                obj (dict o list): Objeto JSON o lista en el que se
buscará la etiqueta.
            """
            if isinstance(obj, dict):
                for key, value in obj.items():
                    if key == etiqueta:
                        resultado.append(value)  # Añadir el valor
asociado a la etiqueta encontrada
                    elif isinstance(value, (dict, list)):
                        buscar_recursivamente(value)  # Continuar
buscando en sub-objetos JSON
            elif isinstance(obj, list):
                for item in obj:
                    buscar_recursivamente(item)  # Continuar buscando
en elementos de la lista

        buscar_recursivamente(obj)  # Llamar a la función interna para
iniciar la búsqueda
        return resultado  # Devolver la lista de resultados encontrados
```

Anexo F. Utilización de la función de búsqueda de etiquetas en archivo JSON extraído con Scrapy

```
    def parse(self, response):
        # Obtener el HTML de la respuesta
        # html_content = response.body.decode(response.encoding)
        #
truco.save_html_to_file(html_content=html_content,titulo='HTML_Adidas')
        esNuevo = response.meta.get('esNuevo')
        item = ZapatillaItem()
        item['modelo'] =
response.xpath("//div[@class='content___1BnBS']//h1[@class='name___120F
N']//span/text()").get()
        item['marca']  = 'Adidas'
        item['precio'] = response.xpath("//div[@class='product-
price___2Mip5 gl-vspace']/div/div/div/div[1]/text()").get()
        # item['color']  = response.xpath("//div[@class='sidebar-color-
chooser___i7JXW']/div[@class='color-label___2hXaD']/text()").get()
        item['url_raiz'] = self.link_raiz
        item['url_calzado'] = response.url
        # Elemento que contiene el json para obtener rating del
producto
        json_element_rating=
response.xpath("/html/body/script[6]/text()").get().replace("window.DAT
A_STORE = JSON.parse",'')[2:-3]
        # Decodificacion del json para el rating
```

```
        json_element_rating=
json_element_rating.encode().decode('unicode-escape')
        json_rating=json.loads(json_element_rating)
         # Elemento que contiene el json para obtener tallas y
descripcion del producto
        json_element =
response.xpath("//body/script[7]/text()").get().replace('window.REACT_Q
UERY_DATA = ','')
        data_json=json.loads(json_element)
        tallas = truco.find_jsonLabel(obj=data_json,etiqueta='size')
        imagenes =
truco.find_jsonLabel(obj=data_json,etiqueta='image_url')
        color = truco.find_jsonLabel(obj=data_json,etiqueta='color')[0]
        # color =
truco.find_jsonLabel(obj=data_json,etiqueta='reviewCount')[0]
        calificacion
=  truco.find_jsonLabel(obj=json_rating,etiqueta="overallRating")

        print('DATA JSON:')
        print(calificacion)
        # item['descripcion'] = response.xpath("//div[@class='text-
content___13aRm']/h3/text()").getall()
        item['descripcion'] =
str(truco.find_jsonLabel(obj=data_json,etiqueta='subtitle')[0]).strip()

        # print(f'Color calzado: {color}')
        item['tallas'] =  (truco.limpiar_lista(tallas))
        item['imagenes'] = imagenes
        item['color']  = color
        if esNuevo:
            item['calificacion'] = -1 if str(esNuevo).lower() == 'new'
else calificacion[0]
        else:
            if len(calificacion)>0:
                item['calificacion']  = calificacion[0]
            else:
                item['calificacion']  = 0
        print(item['calificacion'] )
        yield item
```

Anexo G. Implementación del proceso TF-IDF

```
from bson import ObjectId
from flask import Blueprint, jsonify, request
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
import pandas as pd
```

```
import re
from pymongo import MongoClient

# Conexión a MongoDB y creación del DataFrame
client = MongoClient(
    "localhost"
)
database = client["calzado_deportivo"]
collection = database["calzados"]


# Función para quitar números de un texto
def quitar_numeros(string_argumento):
    return re.sub(r"\d+", "", string_argumento.lower())


# Crear DataFrame y características de texto
cursor = collection.find({})
df = pd.DataFrame(list(cursor))
df["text_features"] = (
    df[["color", "modelo", "descripcion", "precio"]]
    .astype(str)
    .apply(lambda x: " ".join(x), axis=1)
    .apply(quitar_numeros)
)

# Aplicar TF-IDF a las características de texto
tfidf_vectorizer = TfidfVectorizer(
    stop_words="english",  # Establece el idioma de las palabras vacías
a eliminar durante la tokenización (por defecto: "english").
    min_df=10,             # Establece la frecuencia mínima de
documento para incluir un término en el vocabulario (por defecto: 1).
    max_df=0.85            # Establece la frecuencia máxima de
documento para incluir un término en el vocabulario como una fracción
del tamaño total de documentos (por defecto: 1.0).
)
tfidf_matrix = tfidf_vectorizer.fit_transform(df["text_features"])
cosine_similarities = linear_kernel(tfidf_matrix)
```

Anexo H. Diseño de la colección denominado calzado_deportivo

```
{
    "modelo": "Grade School Curry 11 Basketball Shoes",
    "marca": "Under Armour",
    "precio": 100,
    "color": "White / Royal / Versa Blue",
    "descripcion": "Curry Brand Shoes and Gear",
```

```
    "url_raiz": "https://www.underarmour.com/en-us/c/shoes/",
    "url_calzado": "https://www.underarmour.com/en-
us/p/curry_brand_shoes_and_gear/grade_school_curry_11_basketball_shoes/
3026619.html?dwvar_3026619_color=100",
    "imagenes": [
      "https://underarmour.scene7.com/is/image/Underarmour/3026619-
100_A?rp=standard-
30pad%7CpdpMainDesktop&scl=1&fmt=jpg&qlt=75&resMode=sharp2&cache=on%2Co
n&bgc=f0f0f0&wid=566&hei=708&size=536%2C688",
      "https://underarmour.scene7.com/is/image/Underarmour/3026619-
100_TOE?rp=standard-
30pad%7CpdpMainDesktop&scl=1&fmt=jpg&qlt=75&resMode=sharp2&cache=on%2Co
n&bgc=f0f0f0&wid=566&hei=708&size=536%2C688",
      "https://underarmour.scene7.com/is/image/Underarmour/3026619-
100_PAIR?rp=standard-
30pad%7CpdpMainDesktop&scl=1&fmt=jpg&qlt=75&resMode=sharp2&cache=on%2Co
n&bgc=f0f0f0&wid=566&hei=708&size=536%2C688",
      "https://underarmour.scene7.com/is/image/Underarmour/3026619-
100_SOLE?rp=standard-
30pad%7CpdpMainDesktop&scl=1&fmt=jpg&qlt=75&resMode=sharp2&cache=on%2Co
n&bgc=f0f0f0&wid=566&hei=708&size=536%2C688"
    ],
    "tallas": [
      "3.5",
      "4",
      "4.5",
      "5",
      "5.5",
      "6",
      "6.5",
      "7"
    ],
    "calificacion": -1,
    "fecha": {
      "$date": "2023-12-19T07:55:53.407Z"
    }
  }
```


Anexo I. Validación de la inserción de datos extraídos a MongoDB

```
from itemadapter import ItemAdapter
# from scrapy.utils import log
import pymongo
from scrapy.exceptions import DropItem
import logging
from datetime import datetime
```

```python
class ScrapingCalzadoDeportivoPipeline:
    def process_item(self, item, spider):
        return item

class MongoPipeline:
    collection_name = "scrapy_items"

    def __init__(self, mongo_uri, mongo_db, mongo_port,
mongo_collection):
        self.mongo_uri = mongo_uri
        self.mongo_db = mongo_db
        self.mongo_port = mongo_port
        self.mongo_collection = mongo_collection

    @classmethod
    def from_crawler(cls, crawler):
        return cls(
            mongo_uri=crawler.settings.get("MONGODB_SERVER"),
            mongo_db=crawler.settings.get("MONGODB_DB"),
            mongo_port=crawler.settings.get("MONGODB_PORT"),
            mongo_collection=crawler.settings.get("MONGODB_COLLECTION")
,
        )

    def open_spider(self, spider):
        self.client = pymongo.MongoClient(self.mongo_uri)
        self.db = self.client[self.mongo_db]

    def close_spider(self, spider):
        self.client.close()

    def process_item(self, item, spider):
        #  vacío o nulo en el ítem
        if any(value is None or not value for value in
ItemAdapter(item).values()):
            logging.warning("Ítem eliminado debido a campos nulos o
vacíos.")
            raise DropItem("Ítem eliminado debido a campos nulos o
vacíos.")

        # Procesar los datos restantes y realizar la inserción en la
base de datos
        item['precio'] = float(str(item['precio'].strip()).replace("$",
""))
        item['calificacion'] = round(float(item['calificacion']), 2)
        item['fecha'] = datetime.now()
        item['tallas'] = [str(s).strip().replace(' ', '') for s in
item['tallas']]
```

```
        # Verificar duplicados antes de la inserción mediante el
url_calzado
        existing_item =
self.db[self.mongo_collection].find_one({'url_calzado':
item['url_calzado']})
        if existing_item:
            logging.warning("Duplicado encontrado. Actualizando
registro en MongoDB.")
            self.db[self.mongo_collection].replace_one({'_id':
existing_item['_id']}, ItemAdapter(item).asdict())
        else:
            self.db[self.mongo_collection].insert_one(ItemAdapter(item)
.asdict())
            logging.warning("Nuevo producto agregado a MongoDB.")

        return item
```

Anexo J. Archivo de configuración de Scrapy denominado settings.py

```
# Scrapy settings for scraping_calzado_deportivo project
#
# For simplicity, this file contains only settings considered important
or
# commonly used. You can find more settings consulting the
documentation:
#
#     https://docs.scrapy.org/en/latest/topics/settings.html
#     https://docs.scrapy.org/en/latest/topics/downloader-
middleware.html
#     https://docs.scrapy.org/en/latest/topics/spider-middleware.html

BOT_NAME = "scraping_calzado_deportivo"

SPIDER_MODULES = ["scraping_calzado_deportivo.spiders"]
NEWSPIDER_MODULE = "scraping_calzado_deportivo.spiders"


# Crawl responsibly by identifying yourself (and your website) on the
user-agent
# USER_AGENT = "scraping_calzado_deportivo
(+http://www.yourdomain.com)"
USER_AGENT_LIST = [
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3",
```

```
    "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3",
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36",
    # Agrega más agentes de usuario según sea necesario
]
# Obey robots.txt rules
ROBOTSTXT_OBEY = False

# Scrapy-Splash settings
# SPLASH_URL = 'http://localhost:8050'

# DOWNLOADER_MIDDLEWARES = {
#     'scrapy_splash.SplashCookiesMiddleware': 723,
#     'scrapy_splash.SplashMiddleware': 725,
#     'scrapy.downloadermiddlewares.httpcompression.HttpCompressionMidd
leware': 810,
# }
DOWNLOADER_MIDDLEWARES = {
    "scrapy.downloadermiddlewares.useragent.UserAgentMiddleware": None,
    "scrapy_user_agents.middlewares.RandomUserAgentMiddleware": 400,
}

# SPIDER_MIDDLEWARES = {
#     'scrapy_splash.SplashDeduplicateArgsMiddleware': 100,
# }
# DUPEFILTER_CLASS = 'scrapy_splash.SplashAwareDupeFilter'
# HTTPCACHE_STORAGE = 'scrapy_splash.SplashAwareFSCacheStorage'

# Configure maximum concurrent requests performed by Scrapy (default:
16)
# CONCURRENT_REQUESTS = 32

# Configure a delay for requests for the same website (default: 0)
# See https://docs.scrapy.org/en/latest/topics/settings.html#download-
delay
# See also autothrottle settings and docs
DOWNLOAD_DELAY = 3
# The download delay setting will honor only one of:
# CONCURRENT_REQUESTS_PER_DOMAIN = 16
# CONCURRENT_REQUESTS_PER_IP = 16

# Disable cookies (enabled by default)
# COOKIES_ENABLED = False

# Disable Telnet Console (enabled by default)
# TELNETCONSOLE_ENABLED = False
```

```
# Override the default request headers:
# DEFAULT_REQUEST_HEADERS = {
#    "Accept":
"text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8",
#    "Accept-Language": "en",
# }

# Enable or disable spider middlewares
# See https://docs.scrapy.org/en/latest/topics/spider-middleware.html
# SPIDER_MIDDLEWARES = {
#    "scraping_calzado_deportivo.middlewares.ScrapingCalzadoDeportivoSp
iderMiddleware": 543,
# }

# Enable or disable downloader middlewares
# See https://docs.scrapy.org/en/latest/topics/downloader-
middleware.html
# DOWNLOADER_MIDDLEWARES = {
#    "scraping_calzado_deportivo.middlewares.ScrapingCalzadoDeportivoDo
wnloaderMiddleware": 543,
# }

# Enable or disable extensions
# See https://docs.scrapy.org/en/latest/topics/extensions.html
# EXTENSIONS = {
#    "scrapy.extensions.telnet.TelnetConsole": None,
# }

# Configure item pipelines
# See https://docs.scrapy.org/en/latest/topics/item-pipeline.html
# MongoSB Settings
ITEM_PIPELINES = {
    "scraping_calzado_deportivo.pipelines.ScrapingCalzadoDeportivoPipel
ine": 300,
    "scraping_calzado_deportivo.pipelines.MongoPipeline": 800,
    # 'scraping_calzado_deportivo.extension.NotifyExtension': 500,
}

MONGODB_SERVER =
"mongodb+srv://colis90:uiMQv55mZZ31lX21@cluster0.xuwpdxk.mongodb.net/?r
etryWrites=true&w=majority"
MONGODB_PORT = 27017
MONGODB_DB = "calzado_deportivo"
MONGODB_COLLECTION = "calzados"

# Enable and configure the AutoThrottle extension (disabled by default)
# See https://docs.scrapy.org/en/latest/topics/autothrottle.html
# AUTOTHROTTLE_ENABLED = True
```

```
# The initial download delay
# AUTOTHROTTLE_START_DELAY = 5
# The maximum download delay to be set in case of high latencies
# AUTOTHROTTLE_MAX_DELAY = 60
# The average number of requests Scrapy should be sending in parallel
to
# each remote server
# AUTOTHROTTLE_TARGET_CONCURRENCY = 1.0
# Enable showing throttling stats for every response received:
# AUTOTHROTTLE_DEBUG = False

# Enable and configure HTTP caching (disabled by default)
# See https://docs.scrapy.org/en/latest/topics/downloader-
middleware.html#httpcache-middleware-settings
# HTTPCACHE_ENABLED = True
# HTTPCACHE_EXPIRATION_SECS = 0
# HTTPCACHE_DIR = "httpcache"
# HTTPCACHE_IGNORE_HTTP_CODES = []
# HTTPCACHE_STORAGE =
"scrapy.extensions.httpcache.FilesystemCacheStorage"

# Set settings whose default value is deprecated to a future-proof
value
REQUEST_FINGERPRINTER_IMPLEMENTATION = "2.7"
TWISTED_REACTOR =
"twisted.internet.asyncioreactor.AsyncioSelectorReactor"
FEED_EXPORT_ENCODING = "utf-8"
```

Anexo K. Implementación del modelo de recomendación basado en contenidos

```
from bson import ObjectId
from flask import jsonify

# Función para obtener recomendaciones
def get_recommendations(selected_product_ids, cosine_similarities, df):
    try:
        # Obtener los índices de los productos seleccionados en el
DataFrame
        selected_indices = [
            df[df["_id"] == ObjectId(product_id)].index[0]
            for product_id in selected_product_ids
        ]

        # Calcular la similitud promedio entre los productos
seleccionados y todos los demás productos
```

```
        avg_cosine_similarities =
cosine_similarities[selected_indices].mean(axis=0)

        # Ordenar los productos según su similitud promedio y
seleccionar los 60 productos más similares
        sim_scores = sorted(
            enumerate(avg_cosine_similarities), key=lambda x: x[1],
reverse=True
        )[1:61]

        # Obtener los índices de los productos y los puntajes de
similitud
        product_indices, similarity_scores = zip(*sim_scores)

        # Crear un DataFrame con las recomendaciones y añadir el
puntaje de similitud
        recommendations_df = df.iloc[list(product_indices)].copy()
        recommendations_df["similarity_score"] = similarity_scores

        # Ordenar las recomendaciones primero por marca y luego por
puntaje de similitud
        recommendations_df = recommendations_df.sort_values(
            by=["marca", "similarity_score"], ascending=[True, False]
        )

        # Convertir el identificador del producto a una cadena de texto
        recommendations_df["_id"] =
recommendations_df["_id"].astype(str)

        # Devolver las recomendaciones en formato JSON
        return recommendations_df.to_json(orient="records")

    except Exception as e:
        # Manejar cualquier excepción que ocurra durante el proceso
        print(f"Error en obtener recomendacion: {e}")
        return jsonify({"error": str(e)}), 500
```

Anexo L. Desarrollo del componente para la interacción de selección del modelo gustado por el usuario

```
import React, { useEffect, useState } from "react"; // Importar React y
los hooks necesarios
import { truncarTexto } from "../utils/formatUtil"; // Importar una
función de utilidad para truncar texto
import "slick-carousel/slick/slick.css"; // Importar estilos CSS para
el slider
```

```
import "slick-carousel/slick/slick-theme.css"; // Importar estilos de
tema para el slider
import StarRating from "./estrellas_calificacion"; // Importar el
componente de calificación de estrellas
import { FaHeartBroken } from "react-icons/fa"; // Importar el ícono de
corazón roto desde React Icons
import { deviceDetection } from "../utils/localDataUtil"; // Importar
una función de utilidad para detectar el dispositivo
import Slider from "react-slick"; // Importar el componente Slider de
react-slick

// Definir el componente ProductCard
const ProductCard = ({ producto, estado }) => {
  // Definir los estados locales necesarios
  const [productosSeleccionados, setProductosSeleccionados] =
useState([]); // Estado para almacenar los productos seleccionados
  const [tooltipVisible, setTooltipVisible] = useState(false); //
Estado para mostrar u ocultar el tooltip
  const [corazonVisible, setCorazonVisible] = useState(false); //
Estado para mostrar u ocultar el corazón de "favorito"
  const [corazonRotoVisible, setCorazonRotoVisible] = useState(false);
// Estado para mostrar u ocultar el corazón roto
  const [bordeMarcado, setBordeMarcado] = useState(false); // Estado
para marcar el borde del producto

  // Configuración del slider
  const settings = {
    customPaging: function(i) { // Personalizar los puntos de
navegación del slider
      return (
        <a>
          <img src={producto.imagenes[i + 1]} /> {/* Mostrar miniatura
de la siguiente imagen */}
        </a>
      );
    },
    dots: true, // Mostrar puntos de navegación
    dotsClass: "slick-dots slick-thumb", // Clases personalizadas para
los puntos de navegación
    prevArrow: <SamplePrevArrow />, // Ocultar el botón de anterior
    nextArrow: <SamplePrevArrow />, // Ocultar el botón de siguiente
    infinite: true, // Habilitar el bucle infinito
    speed: 500, // Velocidad de la transición
    slidesToShow: 1, // Número de slides a mostrar
    slidesToScroll: 1, // Número de slides a desplazar
    waitForAnimate: false, // No esperar a que termine la animación
para interactuar
    fade: true, // Animación de desvanecimiento entre slides
```

```
  };

  // Función para renderizar la flecha personalizada del slider
  function SamplePrevArrow(props) {
    const { className, style, onClick } = props;
    return (
      <div
        className={className}
        style={{ ...style, visibility:"hidden" }} // Ocultar la flecha
        onClick={onClick}
      />
    );
  }

  // Efecto para establecer el borde marcado
  useEffect(() => {
    setBordeMarcado(estado); // Establecer el estado del borde marcado
basado en el estado pasado como prop
  }, []);

  // Función para manejar la lógica al detectar un dispositivo y hacer
clic
  const handleDispositivoDetectadoParaClickear = (idProducto) => {
    const device = deviceDetection(navigator); // Detectar el
dispositivo del usuario
    switch (device) {
      case "Celular":
        insertarProducto(idProducto); // Insertar el producto en
dispositivos móviles
        break;

      case "Tablet":
        insertarProducto(idProducto); // Insertar el producto en
tabletas
        break;

      default:
        break;
    }
  };

  // Función para truncar la descripción del producto
  if (producto.descripcion !== undefined) {
    producto.descripcion = producto.descripcion
      ? truncarTexto(producto.descripcion) // Truncar el texto si
existe descripción del producto
      : "";
  }
```

```
  // Función para insertar o eliminar un producto
  const insertarProducto = (id) => {
    // Obtener productos almacenados en localStorage
    const productosGuardados =
      JSON.parse(localStorage.getItem("productosSeleccionados")) || [];

    // Verificar si el producto ya está en la lista
    if (!productosGuardados.includes(id)) {
      setCorazonRotoVisible(false); // Ocultar el corazón roto
      console.log("Data ingresada"); // Registro de consola
      // Añadir el nuevo producto al array
      const nuevosProductos = [...productosGuardados, id];

      // Actualizar el estado y localStorage
      setProductosSeleccionados(nuevosProductos);
      localStorage.setItem(
        "productosSeleccionados",
        JSON.stringify(nuevosProductos)
      );
      console.log(nuevosProductos); // Registro de consola
      setCorazonVisible(true); // Mostrar el corazón
      setTimeout(() => setCorazonVisible(false), 1000); // Ocultar el
corazón después de 1 segundo
    } else {
      setCorazonVisible(false); // Ocultar el corazón
      console.log("Data eliminada"); // Registro de consola
      // Filtrar el producto de la lista
      const nuevosProductos = productosGuardados.filter(
        (idArray) => idArray !== id
      );
      console.log(nuevosProductos); // Registro de consola

      // Actualizar el estado y localStorage
      setProductosSeleccionados(nuevosProductos);
      localStorage.setItem(
        "productosSeleccionados",
        JSON.stringify(nuevosProductos)
      );
      setCorazonRotoVisible(true); // Mostrar el corazón roto
      setTimeout(() => setCorazonRotoVisible(false), 1000); // Ocultar
el corazón roto después de 1 segundo
      setBordeMarcado(false); // Establecer el borde marcado como falso
    }
  };

  // Retornar el JSX del componente ProductCard
  return (
```

```
    <div className="rounded-lg w-auto h-full" key={producto._id}>
      <div
        className={`  bg-opacity-0 rounded-lg  ${
          productosSeleccionados.includes(producto._id) || bordeMarcado
            ? "border-red-500 border-4 shadow-lg"
            : "shadow-2xl "
        } shadow bg-white-800  transition transform hover:scale-105
duration-300 cursor-pointer relative`}
        onDoubleClick={() => insertarProducto(producto._id)}
        onMouseEnter={() => setTooltipVisible(true)}
        onMouseLeave={() => setTooltipVisible(false)}
        onClick={() => {
          handleDispositivoDetectadoParaClickear(producto._id);
        }} // Lógica para detectar el dispositivo donde se conecta e
implementar el clic
      >
        <div className="mb-2 p-2 px-2" key={producto._id}>
          <Slider {...settings} className="animate-fade animate-ease-
in-out rounded-lg">
            {producto.imagenes.map((imagen, index) => (
              <div >
                <img
                  className="w-full h-auto object-cover rounded-t-lg"
                  src={imagen}
                />
              </div>
            ))}
          </Slider>
        </div>
        <div className="px-2 pb-2 pt-2 lg:px-5 lg:pb-5 lg:pt-6">
          <h2 className="text-sm lg:text-xl font-semibold tracking-
tight text-white">
            {producto.modelo}
          </h2>
          <h2 className="text-xs lg:text-sm font-semibold tracking-
tight text-white">
            Color: {producto.color}
          </h2>
          <div className="flex items-center mt-1.5 mb-1.5">
            {producto.calificacion === -1 ? (
              <span className="text-xs lg:text-lg px-2 rounded bg-
green-800 text-white">
                <p>Nuevo</p>
              </span>
            ) : (
              <>
                <div className="flex items-center space-x-1 rtl:space-
x-reverse">
```

```
                  <StarRating
calificacion={producto.calificacion}></StarRating>
                </div>
                <span className="bg-blue-100 text-xs lg:text-xl font-
semibold px-1.5 lg:px-2.5 lg:py-0.5 rounded bg-blue- text-blue-800 ms-
3">
                  <p>{producto.calificacion}</p>
                </span>
              </>
            )}
          </div>
          <div className="flex items-center justify-between">
            <span className="text-sm lg:text-4xl font-bold text-white">
              ${producto.precio}
            </span>
            <a
              href={producto.url_calzado}
              className="text-xs lg:text-lg text-white bg-black-
700  focus:ring-4 focus:outline-none font-medium rounded-lg px-2 py-1.5
text-center bg-black hover:bg-gray-700 focus:ring-blue-800"
            >
              Ir al sitio
            </a>
          </div>
          {corazonVisible && (
            <div className="absolute top-2 right-2 text-red-500 z-50">
              <svg className="w-6 h-6 fill-current" viewBox="0 0 24
24">
                <path d="M12 21.35l-1.45-1.32C5.4 14.25 2 11.28 2 8.5 2
5.42 4.42 3 7.5 3c1.74 0 3.41.81 4.5 2.09C16.09 3.81 17.76 3 19.5 3
22.58 3 25 5.42 25 8.5c0 2.78-3.4 5.75-8.55 11.54L12 21.35z" />
              </svg>
            </div>
          )}
          {corazonRotoVisible && (
            <div className="absolute top-2 right-2 text-red-500 z-50">
              <FaHeartBroken className="w-6 h-6 fill-current" />
            </div>
          )}
          {tooltipVisible &&
            (producto.descripcion === "" ? (
              ""
            ) : (
              <div className="absolute top-0 left-0 p-2 bg-gray-800
text-white text-xs lg:text-sm rounded-lg opacity-90">
                {producto.descripcion}
              </div>
            ))}
```

```
        </div>
      </div>
    </div>
  );
};

export default ProductCard; // Exportar el componente ProductCard
```

Anexo M. Desarrollo del componente para la visualización de modelos recomendados

```
import axios from "axios";

// const URI = "http://193.203.164.171:5000/recomendacion";
const URI = "http://192.168.0.105:5000/recomendacion";

const obtenerRecomendacionPorId = async (ids) => {
  try {
    const idsParametros = ids.map((id) => `ids=${id}`).join("&");
    const response = await axios.get(
      `${URI}/recomendacionesByIds?${idsParametros}`
    );

    const productosAgrupados = response.data.reduce((agrupados,
producto) => {
      const marca = producto.marca;

      if (!agrupados[marca]) {
        agrupados[marca] = { marca: marca, productos: [] };
      }

      agrupados[marca].productos.push(producto);

      return agrupados;
    }, {});

    return productosAgrupados;
  } catch (error) {
    if (error.response) {
      // El servidor respondió con un código de estado diferente de 2xx
      console.error(
        "Error en la respuesta del servidor:",
        error.response.status,
        error.response.data
      );
    } else if (error.request) {
      // La solicitud fue realizada pero no se recibió una respuesta
      console.error("No se recibió respuesta del servidor:",
error.request);
    } else {
      // Ocurrió un error durante la configuración de la solicitud
      console.error(
        "Error durante la configuración de la solicitud:",
        error.message
      );
```

```
    }
    return null;
  }
};

export { obtenerRecomendacionPorId };
```

Anexo N. Desarrollo del componente para la visualización del mensaje enviado desde el proyecto Scrapy al Front-end del usuario

```
import React, { useState, useEffect } from "react";
import io from "socket.io-client";

const MensajeScrapy = () => {
  const [mensajeScrapy, setMensajeScrapy] = useState(null);
  const [mostrarMensaje, setMostrarMensaje] = useState(false);

  useEffect(() => {
    const socket = io("http://193.203.164.171:5000?Key-Cliente=React",
{
      transports: ["websocket"],
      query: {
        debug: true,
      },
    });

    socket.on("scrapy_message", (data) => {
      console.log("Mensaje recibido desde Flask:", data.message);
      setMensajeScrapy(data.message);
      setMostrarMensaje(true);
    });

    // Limpia la conexión al desmontar el componente
    return () => {
      socket.disconnect();
    };
  }, []);

  const handleCerrarAviso = () => {
    setMostrarMensaje(false);
  };

  const bannerClasses = `fixed top-0 z-50 flex justify-between h-15 p-4
border-b rounded-xl backdrop-blur-lg animate-flip-down animate-ease-in-
out ${
```

```
    mostrarMensaje ? "animate-normal" : "animate-reverse"
  } ${mensajeScrapy ? `bg-gray-600` : ""}`;

  return (
    <div>
      {mostrarMensaje ? (
        <div className="h-12">
          {mostrarMensaje ? (
            <div>
              <div
                id="sticky-banner"
                tabIndex="-1"
                className={`fixed top-0 z-50 flex justify-between h-15
p-4 border-b rounded-xl backdrop-blur-lg ${
                  mostrarMensaje
                    ? " animate-flip-down animate-ease-in-out animate-
normal"
                    : " animate-flip-down animate-ease-in-out animate-
reverse"
                } ${
                  mensajeScrapy
                    ? mensajeScrapy.estado === "iniciado"
                      ? " bg-gray-600 "
                      : mensajeScrapy.estado === "completado"
                      ? " bg-gray-600 "
                      : mensajeScrapy.estado === "extrayendo"
                      ? " bg-gray-600 "
                      : " "
                    : " "
                }`}
              >
                <div className="flex items-center mx-auto">
                  <span>HEY!!</span>
                  <span classNam="w-2"> </span>
                  <p className="flex items-center text-sm font-normal
text-gray-50">
                    <span className="inline-flex p-1 me-3 bg-gray-200
rounded-full   w-6 h-6 items-center justify-center flex-shrink-0">
                      <svg
                        className="w-3 h-3 text-black "
                        aria-hidden="true"
                        xmlns="http://www.w3.org/2000/svg"
                        fill="currentColor"
                        viewBox="0 0 18 19"
                      >
                        <path d="M15 1.943v12.114a1 1 0 0 1-1.581.814L8
11V5l5.419-3.871A1 1 0 0 1 15 1.943ZM7 4H2a2 2 0 0 0-2 2v4a2 2 0 0 0 2
```

```
2v5a2 2 0 0 0 2 2h1a2 2 0 0 0 2-2V4ZM4 17v-5h1v5H4ZM16
5.183v5.634a2.984 2.984 0 0 0 0-5.634Z" />
                    </svg>
                    <span className="sr-only">Light bulb</span>
                  </span>
                  <span>
                    {mensajeScrapy
                      ? mensajeScrapy.estado === "iniciado"
                        ? "En este momento está iniciando la
actualización de datos."
                        : mensajeScrapy.estado === "completado"
                        ? "En este momento ha completado la
actualización de datos."
                        : mensajeScrapy.estado === "extrayendo"
                        ? "En este momento se están obteniendo nuevos
calzados deportivos."
                        : "No hay estado de scrapy"
                      : "No hay mensaje"}
                    {mensajeScrapy ? mensajeScrapy.porcentaje : ""}
                  </span>
                </p>
              </div>
              <div className="flex items-center">
                <button
                  onClick={handleCerrarAviso}
                  data-dismiss-target="#sticky-banner"
                  type="button"
                  className="flex-shrink-0 inline-flex justify-center
w-7 h-7 items-center text-gray-400 hover:bg-gray-200 hover:text-gray-
900 rounded-lg text-sm p-1"
                >
                  <svg
                    className="w-3 h-3"
                    aria-hidden="true"
                    xmlns="http://www.w3.org/2000/svg"
                    fill="none"
                    viewBox="0 0 14 14"
                  >
                    <path
                      stroke="currentColor"
                      strokeLinecap="round"
                      strokeLinejoin="round"
                      strokeWidth="2"
                      d="m1 1 6 6m0 0 6 6M7 7l6-6M7 7l-6 6"
                    />
                  </svg>
                  <span className="sr-only">Close banner</span>
                </button>
```

```
              </div>
            </div>
          </div>
        ) : (
          <div
            id="sticky-banner"
            tabIndex="-1"
            className={`fixed z-50 flex justify-between w-full h-15
p-4 border-b rounded-xl  bg-gray-50 ${
              mostrarMensaje
                ? " animate-flip-down animate-ease-in-out animate-
normal"
                : " animate-flip-down animate-ease-in-out animate-
reverse"
            }`}
          >
            <div className="flex items-center mx-auto">
              <span>HEY!!</span>
              <span className="w-2"> </span>
              <p className="flex items-center text-sm font-normal
text-gray-50">
                <span className="inline-flex p-1 me-3 bg-gray-200
rounded-full w-6 h-6 items-center justify-center flex-shrink-0">
                  <svg
                    aria-hidden="true"
                    xmlns="http://www.w3.org/2000/svg"
                    fill="currentColor"
                    viewBox="0 0 18 19"
                  >
                    <path d="M15 1.943v12.114a1 1 0 0 1-1.581.814L8
11V5l5.419-3.871A1 1 0 0 1 15 1.943ZM7 4H2a2 2 0 0 0-2 2v4a2 2 0 0 0 2
2v5a2 2 0 0 0 2 2h1a2 2 0 0 0 2-2V4ZM4 17v-5h1v5H4ZM16
5.183v5.634a2.984 2.984 0 0 0-5.634Z" />
                  </svg>
                  <span className="sr-only">Light bulb</span>
                </span>
                <span>Prueba de mensaje scrapy</span>
              </p>
            </div>
            <div className="flex items-center">
              <button
                data-dismiss-target="#sticky-banner"
                type="button"
                className={`flex-shrink-0 inline-flex justify-center
w-7 h-7 items-center text-gray-400 hover:bg-gray-200 hover:text-gray-
900 rounded-lg text-sm p-1.5`}
                >
```

```
                <svg
                  className="w-3 h-3"
                  aria-hidden="true"
                  xmlns="http://www.w3.org/2000/svg"
                  fill="none"
                  viewBox="0 0 14 14"
                >
                  <path
                    stroke="currentColor"
                    strokeLinecap="round"
                    strokeLinejoin="round"
                    strokeWidth="2"
                    d="m1 1 6 6m0 0 6 6M7 7l6-6M7 7l-6 6"
                  />
                </svg>
                <span className="sr-only">Close banner</span>
              </button>
            </div>
          </div>
        )}
      </div>
    ) : (
      ""
    )}
  </div>
);
};
export default MensajeScrapy;
```

Anexo O. Inicio desarrollo del proyecto Back-end mediante Flask Framework

```python
import json
from flask import Flask, request
from flask_socketio import SocketIO

from routes.calzadoRoutes import calzado_routes
from routes.dataProcesoRoutes import data_proceso_routes
from routes.recomendacionRoutes import recomendaciones_routes
from services.mongoService import configure_mongo
from utils.websocketUtil import handle_websocket_message
from utils.mensajesFormatos import printInfoSuccesful, printWarning
from config.expressConfig import configure_app
import time

app = Flask(__name__)
configure_app(app)
socketio = SocketIO(app, cors_allowed_origins="*")
configure_mongo(app)


# Middleware para medir el tiempo de ejecución de las solicitudes
@app.before_request
def before_request():
    request.start_time = time.time()


@app.after_request
def after_request(response):
    if hasattr(request, "start_time"):
        elapsed_time = time.time() - request.start_time
        elapsed_time_format = (
            f"{elapsed_time * 1000:.2f} ms"
            if elapsed_time < 1
            else f"{elapsed_time:.2f} seg"
        )
        app.logger.info(f"Tiempo de ejecución: {elapsed_time_format}")
    return response


# Registrar los blueprints
app.register_blueprint(calzado_routes, url_prefix="/calzado_deportivo")
app.register_blueprint(data_proceso_routes,
url_prefix="/procesoScrapy")
app.register_blueprint(recomendaciones_routes,
url_prefix="/recomendacion")
```

```
# Logica de Websocket
@socketio.on("connect")
def handle_connect():
    keys = ["Scrapy", "React"]
    key_cliente = request.args.get("Key-Cliente")
    mensaje = f"Cliente conectado desde {'Scrapy' if keys[0] ==
key_cliente else 'Aplicación React'}"
    printInfoSuccesful(mensaje) if key_cliente in keys else
printWarning(
        "Intento de conexión no autorizado. Clave compartida
incorrecta."
    )
    return key_cliente in keys


@socketio.on("message")
def handle_message(message):
    message_data = json.loads(message)
    handle_websocket_message(message_data)
    socketio.emit("scrapy_message", {"message": message_data})
    # if str(message_data['estado']) == 'completado':
    #     guardarTablaSimilitudPorCoseno()


if __name__ == "__main__":
    socketio.run(app)
```

Anexo P. Desarrollo de la implementación de ruta para consultas sobre calzados deportivos a MongoDB

```
from flask import Blueprint, jsonify, request
from models.calzadoModel import calzados as CalzadoDeportivo
from datetime import datetime, timedelta
from bson import (
    json_util,
)  # Importa json_util desde bson para manejar la serialización de
ObjectId

calzado_routes = Blueprint("calzado_routes", __name__)


@calzado_routes.route("/", methods=["GET"])
def obtener_calzados():
    try:
        calzados = CalzadoDeportivo.objects().to_json()
        return jsonify(calzados)
```

```
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/total_por_marca", methods=["GET"])
def total_por_marca():
    try:
        marca = request.args.get("marca")
        if not marca:
            return (
                jsonify({"error": 'Se requiere el parámetro "marca" en
la consulta.'}),
                400,
            )

        total_por_marca = CalzadoDeportivo.objects(marca=marca).count()
        return jsonify({"total": total_por_marca})
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/marcas", methods=["GET"])
def obtener_marcas():
    try:
        marcas = CalzadoDeportivo.objects.distinct("marca")
        marcas_ordenadas = sorted(marcas, key=lambda x: x.lower())
        return jsonify({"marcas": marcas_ordenadas})
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/precios", methods=["GET"])
def obtener_precios():
    try:
        precios = CalzadoDeportivo.objects.distinct("precio")
        precios_en_decimales = [float(precio.replace("$", "")) for
precio in precios]
        return jsonify({"preciosEnDecimales": precios_en_decimales})
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/precios_por_marca", methods=["GET"])
def precios_por_marca():
    try:
        marca = request.args.get("marca")

        if not marca:
```

```
            return (
                jsonify({"error": 'Se requiere el parámetro "marca" en
la consulta.'}),
                400,
            )

        precios_y_cantidad =
CalzadoDeportivo.objects(marca=marca).aggregate(
            [
                {"$group": {"_id": "$precio", "cantidad": {"$sum":
1}}},
                {"$project": {"_id": 0, "precio": "$_id", "cantidad":
1}},
                {"$sort": {"precio": 1}},
            ]
        )

        resultados = [
            {"precio": float(item["precio"]), "cantidad":
item["cantidad"]}
            for item in precios_y_cantidad
        ]

        return jsonify({"resultados": resultados})
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/productos_por_marca", methods=["GET"])
def productos_por_marca():
    try:
        marca = request.args.get("marca")

        if not marca:
            return (
                jsonify({"error": 'Se requiere el parámetro "marca" en
la consulta.'}),
                400,
            )

        productos_por_marca = CalzadoDeportivo.objects(marca=marca)

        if not productos_por_marca:
            return (
                jsonify(
                    {"error": f"No se encontraron productos para la
marca: {marca}."}
                ),
```

```
                404,
            )
        # Convertir el resultado a una lista de diccionarios
        resultados = serializarProducto(productos_por_marca)
        return jsonify(resultados)
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/producto_por_modelo/<modelo>", methods=["GET"])
def producto_por_modelo(modelo):
    try:
        producto = CalzadoDeportivo.objects(modelo=modelo).first()

        if not producto:
            return jsonify({"error": "Producto no encontrado."}), 404

        # Convertir el objeto a un diccionario antes de devolverlo
        producto_dict = {
            "modelo": producto.modelo,
            "marca": producto.marca,
            "precio": producto.precio,
            "color": producto.color,
            "url_raiz": producto.url_raiz,
            "url_calzado": producto.url_calzado,
            "descripcion": producto.descripcion,
            "calificacion": producto.calificacion,
            "tallas": producto.tallas,
            "imagenes": producto.imagenes,
            "fecha": producto.fecha,
        }

        return jsonify(producto_dict)
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/numero_calzados_por_marca", methods=["GET"])
def numero_calzados_por_marca():
    try:
        result = CalzadoDeportivo.objects.aggregate(
            [
                {"$group": {"_id": "$marca", "numeroCalzados": {"$sum":
1}}},
                {"$project": {"_id": 0, "marca": "$_id",
"numeroCalzados": 1}},
            ]
        )
```

```
        # Convertir el resultado a una lista antes de ordenarla
alfabéticamente
        result_list = list(result)

        # Ordenar la lista alfabéticamente por la clave "marca"
        result_list.sort(key=lambda x: x["marca"])

        return jsonify(result_list)
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/top25", methods=["GET"])
def top25():
    try:
        result = CalzadoDeportivo.objects.aggregate(
            [
                {"$sort": {"calificacion": -1}},
                {"$group": {"_id": "$marca", "productos": {"$push":
"$$ROOT"}}},
                {
                    "$project": {
                        "marca": "$_id",
                        "_id": 0,
                        "productos": {"$slice": ["$productos", 25]},
                    }
                },
            ]
        )

        # Convertir ObjectId a cadenas en el resultado
        result_list = [
            {
                "marca": item["marca"],
                "productos": [
                    {**product, "_id": str(product["_id"])}
                    for product in item["productos"]
                ],
            }
            for item in result
        ]

        return jsonify(result_list)
    except Exception as e:
        return jsonify({"error": str(e)}), 500
```

```python
@calzado_routes.route("/mejores_productos_por_marca", methods=["GET"])
def mejores_productos_por_marca():
    try:
        marca_parametro = request.args.get("marca")

        if not marca_parametro:
            return jsonify({"error": "La marca es un parámetro
requerido."}), 400

        # Consulta los mejores 5 productos directamente sin necesidad
de agrupación
        result = (
            CalzadoDeportivo.objects(marca=marca_parametro)
            .order_by("-calificacion")
            .limit(15)
        )

        # Convertir el cursor a una lista de diccionarios
        result_list = serializarProducto(result)

        if not result_list:
            return (
                jsonify(
                    {
                        "error": f"No se encontraron productos para la
marca: {marca_parametro}."
                    }
                ),
                404,
            )
        return jsonify(result_list)
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/productos_nuevos", methods=["GET"])
def productos_nuevos():
    try:
        productos_nuevos = CalzadoDeportivo.objects(calificacion=-1)

        if not productos_nuevos:
            return jsonify({"error": "No se encontraron productos
nuevos."}), 404

        productos_nuevos_por_marca = {}
        for producto in productos_nuevos:
            marca = producto.marca
            if marca not in productos_nuevos_por_marca:
```

```
                productos_nuevos_por_marca[marca] = []

            if len(productos_nuevos_por_marca[marca]) < 25:
                productos_nuevos_por_marca[marca].append(producto)

        resultado_formateado = [
            {"marca": marca, "productos": productos}
            for marca, productos in productos_nuevos_por_marca.items()
        ]

        return jsonify(resultado_formateado)
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/productos_nuevos_por_marca", methods=["GET"])
def productos_nuevos_por_marca():
    try:
        marca = request.args.get("marca")

        if not marca:
            return (
                jsonify(
                    {
                        "error": "Se debe proporcionar la marca como
parámetro de consulta."
                    }
                ),
                400,
            )

        productos_nuevos_por_marca = CalzadoDeportivo.objects(
            marca=marca, calificacion=-1
        )

        if not productos_nuevos_por_marca:
            return jsonify({"marca": marca, "productos": []})

        results = serializarProducto(productos_nuevos_por_marca)

        # Convierte ObjectId a cadenas en los resultados
        results = json_util.loads(json_util.dumps(results))

        return jsonify({"marca": marca, "productos": results})
    except Exception as e:
        return jsonify({"error": str(e)}), 500
```

```python
@calzado_routes.route("/productos_en_fecha", methods=["GET"])
def productos_en_fecha():
    try:
        fecha = request.args.get("fecha")

        if not fecha:
            return (
                jsonify(
                    {
                        "error": "Se debe proporcionar la fecha como
parámetro de consulta."
                    }
                ),
                400,
            )

        fecha_objeto = datetime.strptime(fecha, "%Y-%m-%d")
        fecha_sin_hora = datetime(
            fecha_objeto.year, fecha_objeto.month, fecha_objeto.day
        )

        resultados = CalzadoDeportivo.objects(
            fecha__gte=fecha_sin_hora, fecha__lt=fecha_sin_hora +
timedelta(days=1)
        )

        hay_productos_en_fecha = len(resultados) > 0

        return jsonify({"hayProductosEnFecha": hay_productos_en_fecha})
    except Exception as e:
        return jsonify({"error": str(e)}), 500


@calzado_routes.route("/total_calzados", methods=["GET"])
def total_calzados():
    try:
        total_calzados = CalzadoDeportivo.objects.count()
        return jsonify({"total_calzados": total_calzados})
    except Exception as e:
        return jsonify({"error": str(e)}), 500


def (productos):
    resultados = [
        {
            "modelo": producto.modelo,
            "precio": producto.precio,
            "marca": producto.marca,
```

```
            "precio": producto.precio,
            "color": producto.color,
            "url_raiz": producto.url_raiz,
            "url_calzado": producto.url_calzado,
            "descripcion": producto.descripcion,
            "calificacion": producto.calificacion,
            "tallas": producto.tallas,
            "imagenes": producto.imagenes,
            "fecha": producto.fecha,
            "_id": str(producto.id),
        }
        for producto in productos
    ]
    return resultados
```

Anexo Q. Desarrollo de la implementación de ruta para brindar resultados del modelo de recomendación

```
from bson import ObjectId
from flask import Blueprint, jsonify, request
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
import pandas as pd
import re
from pymongo import MongoClient

recomendaciones_routes = Blueprint("recomendaciones", __name__)

# Conexión a MongoDB y creación del DataFrame
client = MongoClient(
    "mongodb+srv://colis90:uiMQv55mZZ31lX21@cluster0.xuwpdxk.mongodb.ne
t/?retryWrites=true&w=majority"
)
database = client["calzado_deportivo"]
collection = database["calzados"]


# Función para quitar números de un texto
def quitar_numeros(string_argumento):
    return re.sub(r"\d+", "", string_argumento.lower())


# Crear DataFrame y características de texto
cursor = collection.find({})
df = pd.DataFrame(list(cursor))
df["text_features"] = (
```

```
    df[["color", "modelo", "descripcion", "precio"]]
    .astype(str)
    .apply(lambda x: " ".join(x), axis=1)
    .apply(quitar_numeros)
)

# Aplicar TF-IDF a las características de texto
tfidf_vectorizer = TfidfVectorizer(stop_words="english", min_df=10,
max_df=0.85) # Este metodo tiene predefinido la implementacion de NLP
tfidf_matrix = tfidf_vectorizer.fit_transform(df["text_features"])
cosine_similarities = linear_kernel(tfidf_matrix)


# Función para obtener recomendaciones
def get_recommendations(selected_product_ids, cosine_similarities, df):
    try:
        selected_indices = [
            df[df["_id"] == ObjectId(product_id)].index[0]
            for product_id in selected_product_ids
        ]
        avg_cosine_similarities =
cosine_similarities[selected_indices].mean(axis=0)
        sim_scores = sorted(
            enumerate(avg_cosine_similarities), key=lambda x: x[1],
reverse=True
        )[1:61]

        product_indices, similarity_scores = zip(*sim_scores)
        recommendations_df = df.iloc[list(product_indices)].copy()
        recommendations_df["similarity_score"] = similarity_scores

        recommendations_df = recommendations_df.sort_values(
            by=["marca", "similarity_score"], ascending=[True, False]
        )

        recommendations_df["_id"] =
recommendations_df["_id"].astype(str)
        return recommendations_df.to_json(orient="records")
    except Exception as e:
        print(f"Error en obtener recomendacion: {e}")
        return jsonify({"error": str(e)}), 500


# Ruta en Flask para obtener recomendaciones por IDs de productos
@recomendaciones_routes.route("/recomendacionesByIds", methods=["GET"])
def obtener_recomendaciones():
    try:
        selected_ids = request.args.getlist("ids")
```

```
        recommendations = get_recommendations(selected_ids,
cosine_similarities, df)
        return recommendations
    except Exception as e:
        return jsonify({"error": str(e)}), 500
```

Anexo R. Desarrollo de la comunicación SocketIO desde Scrapy al servidor Flask

```
# codigo con socketio
import json
import socketio
from scrapy.crawler import CrawlerProcess
from scrapy.utils.project import get_project_settings

# Crear una instancia de SocketIO
sio = socketio.Client()


def enviar_mensaje():
    # Conectar al servidor Flask-SocketIO
    sio.connect("http://193.203.164.171:5000?Key-Cliente=Scrapy")
    # sio.connect("http://localhost:5000?Key-Cliente=Scrapy")
    # Enviar un mensaje al servidor
    mensaje = {"estado": "iniciado", "tipo": "scrapy","porcentaje": 0}
    sio.send(json.dumps(mensaje))
    print(f"Mensaje enviado desde Scrapy: {mensaje}")


# Función para enviar un mensaje indicando que la extracción ha sido
completada
def enviar_mensaje_completado():
    # Enviar un mensaje al servidor
    mensaje = {"estado": "completado", "tipo": "scrapy","porcentaje":
100}
    sio.emit("message", json.dumps(mensaje))
    print(f"Mensaje enviado desde Scrapy: {mensaje}")


# Función para enviar un mensaje indicando procentaje de avance de la
extracción de datos
def enviar_mensaje_procentaje(percentage):
    # Enviar un mensaje al servidor
    mensaje = {"estado": "extrayendo", "tipo": "scrapy", "porcentaje":
round(float(percentage),2)}
    sio.emit("message", json.dumps(mensaje))
    print(f"Mensaje enviado desde Scrapy: {mensaje}")
```

```python
# Función para correr todas las arañas
def run_all_spiders():
    process = CrawlerProcess(get_project_settings())

    # Ejecutar la función para enviar el mensaje antes de iniciar las
arañas
    (enviar_mensaje())

    # Obtener la cantidad total de arañas
    total_spiders = len(process.spiders.list())
    for i, spider_name in enumerate(process.spiders.list()):
        if spider_name == "Adidas":
            # Calcular y enviar el porcentaje de avance
            percentage = (i + 1) / total_spiders * 100
            enviar_mensaje_procentaje(percentage)
            process.crawl(spider_name)
    process.start()

    # Enviar un mensaje indicando que la extracción ha sido completada
    (enviar_mensaje_completado())


# Manejar el evento de conexión
@sio.event
async def connect():
    print("Conectado al servidor de SocketIO")


# Manejar el evento de desconexión
@sio.event
async def disconnect():
    print("Desconectado del servidor de SocketIO")


if __name__ == "__main__":
    # Ejecutar la aplicación
    (run_all_spiders())
```

Anexo S. Fichas de observaciones de tiempo

Ficha de observación de tiempo		
Proceso: Uso del motor de sugerencias **Observador:** Jorge **Fecha:**10-1-2024		
Objetivo	Determinar el tiempo requerido por los usuarios para tomar una decisión en la compra de calzado en la línea deportiva.	
Lugar geográfico		
Proceso de toma de decisión del calzado deportivo		
Paso	**Elemento de trabajo**	**Tiempo transcurrido**
1	Selección de calzado deportivo	
2	Selección de la marca del modelo	
3	Decisión de compra	
	Tiempo total	

Ficha de observación de tiempo		
Proceso:Búsqueda de calzado deportivo sin motor de sugerencias **Observador:** Jorge Ibarra **Fecha:**10-1-2024		
Objetivo	Determinar el tiempo requerido por los usuarios para tomar una decisión en la compra de calzado en la línea deportiva.	
Lugar geográfico		
Proceso de toma de decisión del calzado deportivo		
Paso	**Elemento de trabajo**	**Tiempo transcurrido**
1	Selección de calzado deportivo	
2	Selección de la marca del modelo	
3	Decisión de compra	
	Tiempo total	