Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN ENXEÑARÍA INFORMÁTICA
MENCIÓN EN COMPUTACIÓN

Perfilado automático de usuarios en corpus sociales sobre el movimiento Black Lives Matter

Estudante: David Rodríguez Bacelar

Dirección: Patricia Martín Rodilla, David Otero Freijeiro

A Coruña, julio de 2023.

Dedicatoria

Agradecimientos

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Resumen

Con todo, a lo largo de este trabajo, se profundizará más en el perfilado automático de autores, analizando diferentes algoritmos y su rendimiento, a la vez que se ofrecerá una aplicación web donde se mostrará el resultado del perfilado en un *dashboard* intuitivo y accesible. Además, utilizaremos como caso de uso y análisis la colección de referencia sobre el movimiento *Black Lives Matter (BLM)*, extraída mediante la metodología descrita en [1].

Abstract

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like ``Huardest gefburn''? Kjift -- not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Palabras clave:

- First itemtext
- Second itemtext
- Last itemtext
- First itemtext
- Second itemtext
- Last itemtext
- First itemtext

Keywords:

- First itemtext
- Second itemtext
- Last itemtext
- First itemtext
- Second itemtext
- Last itemtext
- First itemtext

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

A partir de la segunda década de los 2000, las redes sociales han experimentado un crecimiento continuado de su uso, tanto en número de usuarios como en cantidad de información generada. Como se muestra en el reporte *"Digital 2023 Global Overview Report"* (We Are Social et al., 2023) [2], a principios del año 2013 existían alrededor de 1.7 millones de usuarios en las redes sociales, mientras que a principios del año 2023, esta cifra aumentó hasta los 4.7 millones, con una variación anual media del 10.8% (se puede ver más información en la Figura 1.1). Así, plataformas como Facebook, Instagram, Twitter o YouTube, cuentan con millones de usuarios activos diariamente, donde se comparte información o se crea contenido de entretenimiento. Además, las redes sociales se han convertido en un lugar donde se debate sobre temas políticos, sociales o económicos, y donde se comparten diversas opiniones y noticias. Este hecho se puede ver, de nuevo, en el reporte mecionado anteriormente, donde se muestra que el 34.2% de los usuarios de redes sociales las utilizan para informarse sobre noticias, el 28.8% para saber cuales son los temas de actualidad y el 23.4% para compartir opiniones y debatir con otros usuarios.

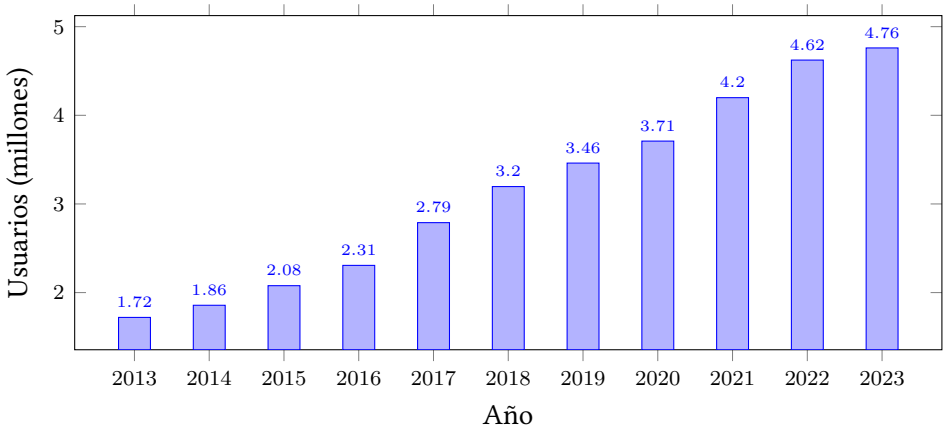

Figura 1.1: Evolución del número de usuarios en redes sociales.

1.1 Importancia de las redes sociales

Toda esta información generada tiene una gran relevancia a distintos niveles. En primer lugar, cabe destacar el impacto que tienen las redes sociales a nivel político. Y es que en la actualidad, vivimos en una campaña permanente (Blumenthal, 1980) [3], donde el acceso a la redes sociales ofrece la posibilidad a los ciudadanos de estar informados sobre política, mientras que a las instituciones de poder les permite conocer el estado de la opinión pública (Strömbäck, 2008) [4], pudiendo llegar a influenciar "mucho" o "bastante" la intención de voto (Gallardo-Paúls, 2016)[5]. En segundo lugar, la información generada en las redes sociales también tiene un gran impacto a nivel social. ¿Quiénes son los usuarios más activos? ¿Qué género predomina en las redes sociales? ¿Qué edad tienen los usuarios?. Cabe resaltar también el impacto que tienen las redes sociales a nivel económico, ya que estas plataformas se han convertido en un lugar donde las empresas publicitan sus productos y servicios y a las que destinan gran parte de sus presupuestos en publicidad (Saxena et al., 2013)[6]. Finalmente, destacar el impacto que tienen las redes sociales a nivel de seguridad, ya que estas plataformas se han convertido en un lugar donde se comparte información personal, y donde se pueden cometer delitos como el *cyberbullying* o el *grooming* (Machimbarrena et al., 2018)[7].

1.2 Perfilado de autor

Surge de esta forma la necesidad de analizar toda la información que se genera y proporiconar así herramientas útiles en todos los niveles vistos en la Sección 1.1. En este contexto, el perfilado de autor en redes sociales se ha convertido en un área de investigación de gran interés en los últimos años, posicionándose como una herramienta de creciente importancia en áreas como la seguridad, el *marketing* o la investigación forense (Rangel et al., 2013)[8].

El perfilado de autor, también conocido como *author profiling* en inglés, consiste en determinar, a partir de un texto, las características de su autor, como su género, edad, rasgos personales, etc. Para ello se hace uso de diversas técnicas de aprendizaje automático basadas en NLP (del inglés *Natural Language Processing*), que permiten extraer características lingüísticas del texto y utilizarlas para una posterior clasificación. Así, el perfilado de autor sería de gran utilidad a la hora de evaluar sospechosos teniendo en cuenta su perfil lingüístico; sería muy práctico para empresas que quieran conocer el perfil de los clientes que opinan de forma negativa y positiva de sus productos; tendría un gran valor para los partidos políticos, dado que podrían conocer cuál es el perfil de sus votantes; y sería una herramienta de gran ayuda a la hora de realizar análisis sociales sobre temas específicos.

Capítulo 2

Estado del arte del perfilado de autor

Durante los inicios del perfilado automático de autor, los algoritmos se centraban en la tarea de la clasificación por género. En esta línea, trabajos como (Koppel et al., 2002)[9] se desmarcaban de la tendencia de la época, la cual se basaba en la clasificación de textos en base a su contenido, para centrarse en la clasificación de textos **en base a su estilo**. En este caso, se centraban en la obtención del género del autor mediante el análisis de 920 documentos de carácter formal escritos en inglés con una media de alrededor de 34.300 palabras cada uno, obteniendo una precisión en la clasificación de aproximadamente el 77%.

Así, la demostración de la existencia de un estilo de escritura diferenciado entre hombres y mujeres, supuso un gran avance en el campo del perfilado de autor y dió pié a la realización de trabajos como (Argamon et al., 2003)[10], (Corney et.al, 2002)[11] o (Otterbacher et al., 2010)[12], así como también permitió el inicio de una clasificación más compleja en base a otras características como la edad, la orientación sexual o la personalidad.

Más tarde, en el año 2011 se celebraría el primer evento organizado por el *PAN* (*Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection*), un foro de investigación impulsado por la compañia Webis, que organiza eventos científicos y tareas anuales relacionadas con el análisis forense de textos digitales y rasgos estilométricos. La primera de estas tareas centrada en el perfilado de autor se celebraría en el año 2013 (Rangel et al., 2013)[8], en la que se pedía a los participantes que obtuvieran, a partir de una serie de *tweets*, la edad y el género de su autor. El ganador de este concurso obtuvo una precisión del 60% en la clasificación de género y del 67% en la clasificación de edad, haciendo uso, la mayor parte de los participantes, de técnicas de aprendizaje supervisado como los Árboles de Decisión (en inglés *Decission Trees*) o las Máquinas de Soporte Vectorial (en inglés *Support Vector Machines*) e inluyendo en sus modelos características basadas en el TF-IDF, n-gramas, etiquetas POS o características como el número de emoticonos o la frecuencia de signos de puntuación. En los siguientes años se celebrarían nuevas ediciones de esta tarea (Rangel et al., 2014[13], Rangel et

al., 2015[14], Rangel et al., 2016[15]...), añadiendo nuevas sub-tareas como el reconocimiento de rasgos personales, la ocupación o los dialectos del lenguaje, así como también alcanzando mejores resultados en la clasificación.

2.1 Algoritmos supervisados

Todos estos algoritmos se basan en lo que en recuperación de información se conoce como *TF-IDF* (del inglés *Term Frequency-Inverse Document Frequency*), que es una medida numérica que expresa cuán relevante es una palabra para un documento en una colección. La importancia aumenta proporcionalmente al número de veces que una palabra aparece en el documento, pero se compensa con la frecuencia de la palabra en la colección de documentos, lo que permite manejar el hecho de que algunas palabras son generalmente más comunes que otras.

Capítulo 3

Herramientas, técnicas y lenguajes

3.1 *Backend*

Ya que para el *backend* no se necesitaba nada excesivamente complejo, se decidió utilizar el lenguaje de programación Python [16] junto con el *framework* FastAPI [17], el cual permite crear APIs REST de forma sencilla. La decisión de utilizar Python, viene también condicionada por el hecho de que los algoritmos de perfilado de autor utilizados, así como también la mayor parte de los algoritmos de aprendizaje automático, están ya programados en Python, evitando así crear nuevos *endpoints*, *sockets* o *bindings* para la ejecución de dichos algoritmos. Destacar también que el desarrollo ágil en Python favorecía mucho al trabajo debido a su tipado dinámico, a su ejecución interpretada y a su sintaxis sencilla.

Por otro lado, para que el cógido estuviese bien organizado, se decidió utilizar el patrón de diseño DDD (del inglés *Domain-Driven Design*) [18], el cual permite separar el código en tres capas: la capa de aplicación, la capa de dominio y la capa de infraestructura. La capa de aplicación es la encargada de gestionar la entrada y salida de la aplicación que, en nuestro caso, es el controlador que maneja los *endpoints* REST; la capa de dominio es la que contiene la lógica de negocio y las entidades; y la capa de infraestructura es la responsable de administrar las interacciones internas de la aplicación que, en nuestro caso, se encarga de la comunicación con la base de datos.

Esta es una figura adaptada de https://www.baeldung.com/hexagonal-architecture-ddd-spring Łukasz Ryś 2022

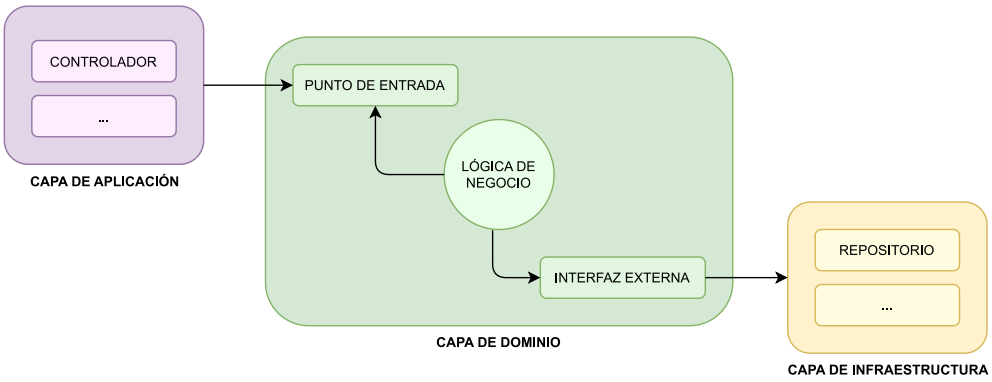

Figura 3.1: Diagrama de capas del patrón de diseño DDD

3.2 *Frontend*

En cuanto al *frontend*, se decidió utilizar NextJS [19] como herramienta para el desarrollo de la interfaz de usuario, dado que ya se contaba con bastante experiencia previa en su uso. NextJS es un *framework* basado en React [20], es decir, en la construcción de interfaces dinámicas e interactivas mediante la composición de elementos que pueden tener estado. Además, este *framework* implementa varias mejoras sobre React como por ejemplo el *server-side rendering*, algo que ayuda en gran medida al SEO (del inglés *Search Engine Optimizations*), esto es, a que los motores de búsqueda como Google puedan indexar mejor la página y, por tanto, que esta aparezca en una posición superior en los resultados de búsqueda. Además, también cuenta con optimizaciones para la carga y el renderizado de imágenes o fuentes entre otras.

Todo ello se desarrolló utilizando TypeScript [21], un lenguaje de programación que añade tipado estático a JavaScript y que está ganando mucha popularidad con respecto a la mentinibilidad, compresión y escalabilidad que proporciona a los proyectos en los que se usa (Stack Overflow, 2023) [22]. A mayores, para garantizar la consistencia de todas las entidades, y más en específico de los DTOs (del inglés *Data Transfer Object*), se utilizó Zod [23], una librería que permite definir esquemas de validación de datos, incluyendo el tipo específico de cada campo.

En cuanto el estilado de la página, se optó por emplear SASS (del inglés *Syntactically Awesome Style Sheets*) [24], un preprocesador de CSS (del inglés *Cascading Style Sheets*) que añade funcionalidades extra como son el uso de variables, bucles o anidamiento de clases. Además, dado que se estaba desarrollando una aplicación novedosa, se buscó crear un estilo propio haciendo uso de CSS "nativo", desmarcándose así de librerías que proporcionasen estilos predefinidos como Bootstrap [25] o componentes ya implementados como Material UI [26] o Chakra UI [27]. Por otra parte, para la creación de gráficos se utilizó la librería ChartJS [28],

una de las más conocidas y con más soporte en la actualidad. Finalmente, para implementar las animaciones en la interfaz, se hizo uso, en conjunto con las transiciones nativas de CSS, de Framer Motion [29], una librería que permite crear animaciones de una mayor complejidad desde JavaScript/TypeScript.

3.3 Algoritmos de perfilado

A pesar de que los algoritmos de perfilado se ejecutan en el *backend*, se ha decidido incluirlos en esta sección debido a que se trata de una parte importante y característica en el proyecto. Decir también que en este caso, las herramientas utilizadas estaban condicinadas, lógicamente, por aquellas utilizadas en las implementaciones de ambos algoritmos seleccionados.

En cuanto a la parte nuclear del aprendizaje automático, se utlizó Scikit-Learn [30], una librería de Python que proporciona una gran cantidad de algoritmos pre-implementados, así como también herramientas para la extracción de características o la validación de modelos. Además, se hizo uso de otras librerías como Tqdm [31], la cual permite mostrar el progreso de entramiento o predicción de forma visual, Pickle [32], que permite serializar objetos Python (en nuestro caso los modelos ya entrenados), o NumPy [33], una librería que proporciona estructuras de datos y herramientas para el cálculo científico.

3.4 Soporte

Para la gestión de tareas, debido a que nuestro ciclo de desarrollo era ágil, se utilizó Trello [34], una herramienta que permite crear tableros con listas de tareas, las cuales pueden ser movidas entre ellas según su estado: pendiente, en progreso o completada.

Además, debido a la importancia de mantener un control sobre los cambios realizados en el código, se optó por utilizar Git [35] como herramienta de control de versiones junto con GitHub [36], una plataforma que permite almacenar repositorios de Git de forma remota, similar a otras como GitLab [37] o BitBucket [38].

En cuanto al entorno de desarrollo o IDE (del inglés *Integrated Development Environment*), se utilizó Visual Studio Code [39], un editor gratuito y de código semi-abierto desarrollado por Microsoft, el cual brinda una gran flexibilidad para trabajar con cualquier lenguaje de programación gracias a sus extensiones. Esto hecho, posibilitó el desarrollo del *backend*, del *frontend* e inluso de esta memoria en un mismo programa, simplificando la tarea de aprender las peculiaridades de otros IDEs más específicos. Más en concreto, se utilizaron extensiones

como Prettier [40] o Black [41], las cuales permiten formatear el código de forma automática para JavaScript o Python, respectivamente; ESLint [42], una herramienta que permite detectar errores en el código JavaScript/TypeScript; o GitLens [43], una extensión que añade funcionalidades extra con respecto al control de cambios.

Con respecto, a la elaboración de los diagramas y los *wireframes* que aparecen en este trabajo, se utilizó la herramienta Draw.io [44], la cual permite crear todo tipo de diagramas *online* de forma gratuita, sin la obligación de registrarse y sin la necesidad de instalar ningún programa, pudiendo exportarlos en varios formatos, entre ellos SVG. Junto a esta herramienta, se utilizó también la librería pgfplots [45] para la creación de las gráficas en el propio LaTeX.

Mencionar Docker e implementarlo

Capítulo 4
Análisis

4.1 Requisitos funcionales

Para la definición de los requisitos funcionales que debe cumplir la aplicación, se ha hecho uso de las historias de usuario. Esta técnica permite definir los requisitos de una forma más cercana al usuario, ya que se centra en la descripción de las funcionalidades que este desea que tenga el sistema. Con todo, las historias de usuario obtenidas se muestran en la Tabla 4.1.

ID	Historia de usuario
1	Como usuario quiero poder subir mi propio dataset de textos para el perfilado
2	Como usuario quiero conocer ejemplos del formato de los datasets aceptados
3	Como usuario quiero poder seleccionar el algoritmo de perfilado que se va a utilizar
4	Como usuario quiero poder visualizar los datos obtenidos tras el perfilado
5	Como usuario quiero poder ver una lista detallada de cada persona perfilada
6	Como usuario quiero poder ordenar la lista de personas por cada uno de los campos perfilados
7	Como usuario quiero poder saber como funcionan los diferentes algoritmos de perfilado disponibles
8	Como usuario quiero ver el rendimiento de los algoritmos de perfilado disponibles

Tabla 4.1: Historias de usuario

A partir de estas historias de usuario se ha obtenido el diagrama de casos de uso que se muestra en la Figura 4.1.

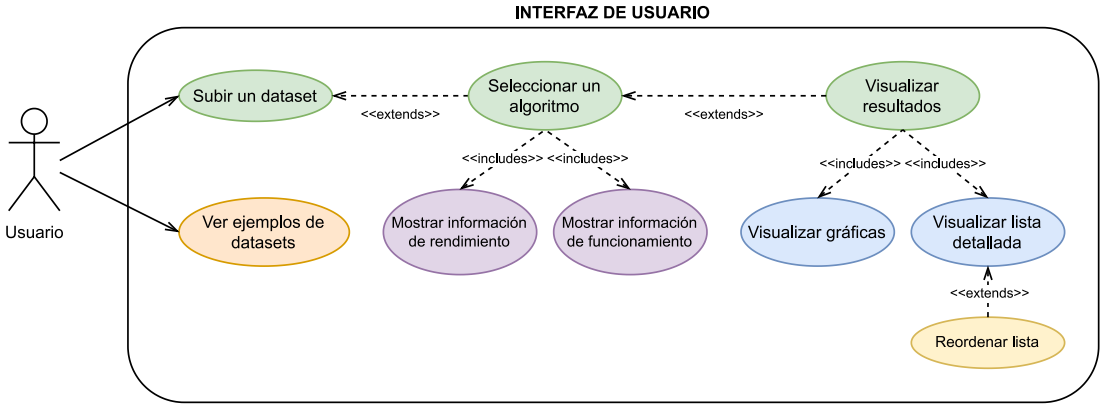

Figura 4.1: Diagrama de casos de uso

4.2 Requisitos no funcionales

Una vez definidas las funcionalidades que debe tener la aplicación, es necesario definir cuales van a ser sus requisitos no funcionales, esto es, aquellos que están relacionados en cómo debe funcionar la aplicación. Para ello, se han tenido en cuenta los aspectos recogidos en la Tabla 4.2.

Requisito	Descripción
Usabilidad	La aplicación debe ser fácil de usar, de forma que cualquier usuario pueda utilizarla sin necesidad de tener conocimientos previos sobre el perfilado de autores
Escalabilidad	La aplicación debe ser capaz de procesar datasets de cualquier tamaño, de forma que no exista un límite en el tamaño de los mismos
Portabilidad	La aplicación debe ser capaz de ejecutarse en cualquier sistema, de forma que los usuarios no tengan que preocuparse por el dispositivo que utilizan

Tabla 4.2: Requisitos no funcionales

Capítulo 5

Diseño

5.1 Arquitectura

Teniendo en cuenta los requisitos definidos en las Secciones 4.1 y 4.2, así como también las limitaciones temporales del proyecto, se ha optado por una arquitectura cliente-servidor. Esta aquitectura se ha elegido por la facilidad en su implementación y por su capacidad de intercomunicación con otros sistemas, especialmente con clientes web. Como se aprecia en la Figura 5.1, el servidor cuenta con un controlador REST capaz de redirigir las peticiones al servicio corresepondiente. En este sentido, a pesar de solo contar con un único servicio (*Servicio de perfilado*) en la versión actual, la arquitectura cliente-servidor permite añadir nuevos servicios, junto a nuevas funcionalidades, de forma sencilla y sin poner en riesgo al resto de servicios.

A mayores, el propio *frontend* web tendrá también una arquitecture basada en cliente-servidor, en la que el servidor será el encargado de ofrecer al cliente los archivos estáticos necesarios para mostrar la interfaz (HTML, CSS, JavaScript, fuentes, imágenes...).

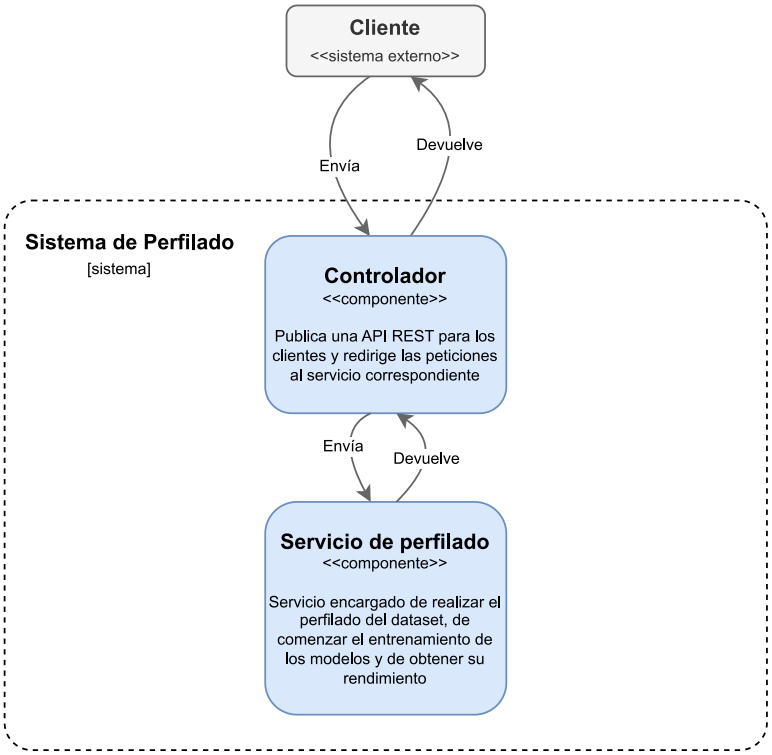

Figura 5.1: Diagrama C4 de Contenedor de la arquitectura del sistema.

5.2 Diagramas de secuencia

5.3 Prototipado

Como se mencionó anteriormente, para el diseño de los prototipos de pantallas, también conocidos como *wireframes*, se utilizó la herramienta Draw.io [44] debido a su sencillez y rapidez en la elaboración con respecto a otros programas más avanzados como pueden ser Adobe XD [46] o Figma [47].

La filosofía de diseño de la interfaz se centró en el minimalismo y la usabilidad, por lo que todo estará caracterizado por no contener excesivos elementos y por ser fácilmente comprensible para el usuario.

La pantalla de inicio, como se ve en la Figura 5.2, está compuesta simplemente por un título, un subtítulo y un campo que permitirá al usuario subir un *dataset*, tanto mediante un elemento *drag and drop* como mediante un botón que abrirá el explorador de archivos del sistema operativo.

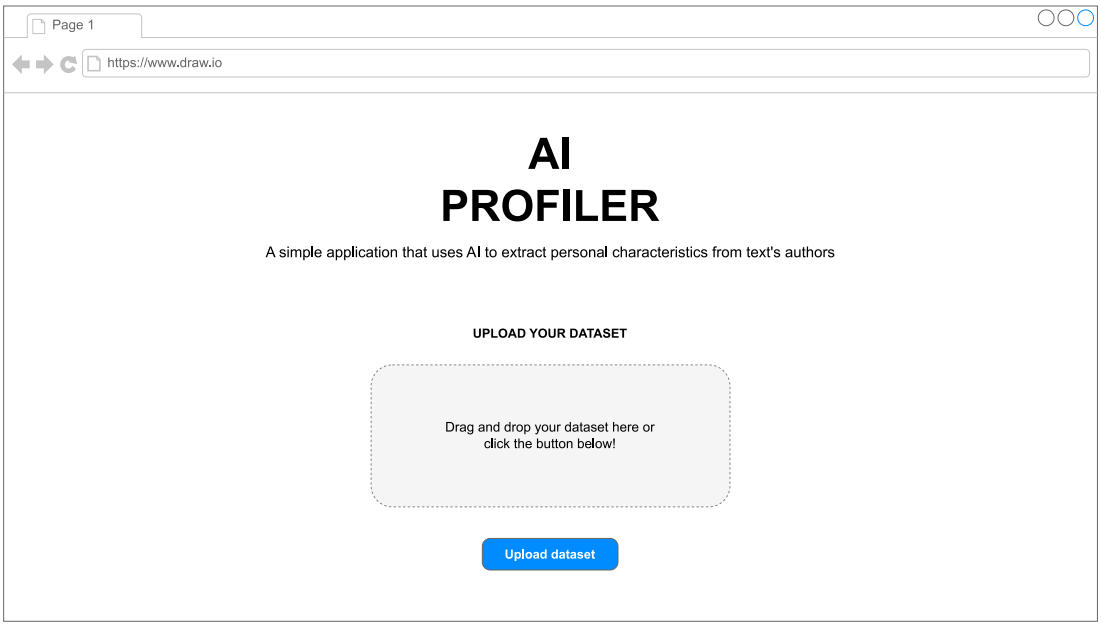

Figura 5.2: Prototipo de la pantalla de inicio

Una vez el usuario haya subido el *dataset*, el campo de subida se sustituirá por la lista de los algoritmos de perfilado disponibles, como se puede ver en la Figura 5.3. De cada algoritmo se mostrará, en una tarjeta, las características que es capaz de extraer del perfilado junto con su título. Además, ya que uno de los requisitos era el de mostrar información del funcionamiento y del rendimiento de los algoritmos, se decidió crear un *tooltip* que se mostrase cada vez que el usuario pasase el ratón por encima de cada tarjeta. Finalmente, si el usuario desease cambiar el *dataset* seleccionado, se permitirá retroceder mediante la flecha situada junto al título del paso actual.

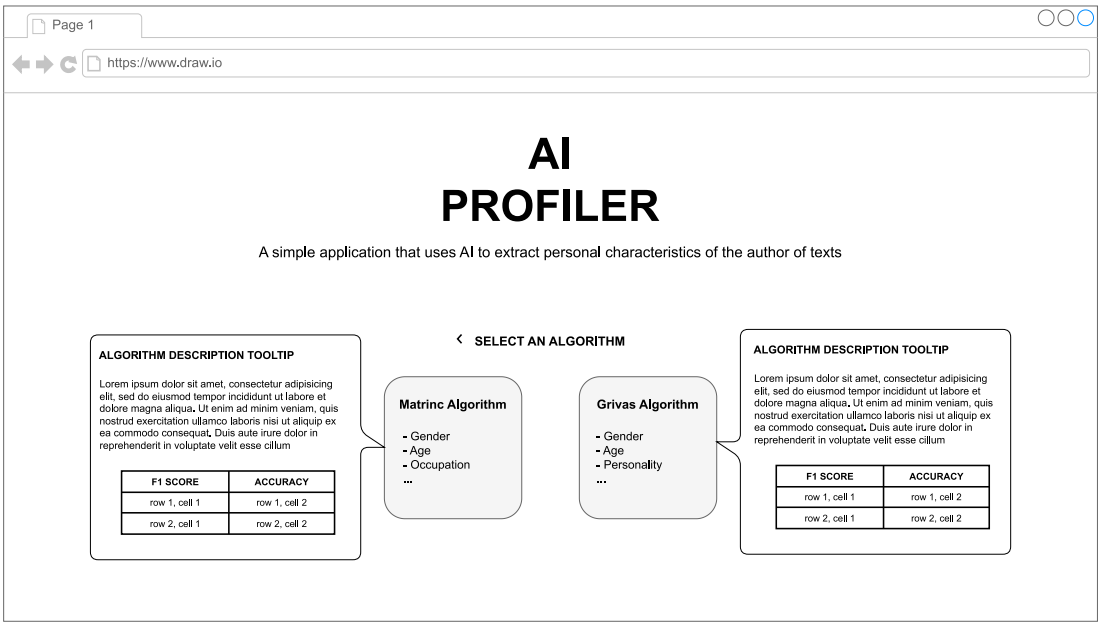

Figura 5.3: Prototipo de la selección del algoritmo de perfilado

Ya con el *dataset* y el algoritmo seleccionados, se presentará al usuario un resumen del perfilado, donde se mostrará el nombre del fichero subido, la tarjeta del algoritmo seleccionado y un botón que permitirá al usuario comenzar con el proceso de perfialdo. Además, como se aprecia en la Figura 5.4, una vez comienza el perfilado, se mostrará una barra de progreso o un *spinner* que le indicará al usuario que el proceso está en marcha. De la misma forma que en el paso anterior, si el usuario decide cambiar de algoritmo, puede retroceder haciendo uso de la flecha situada junto al título del paso actual.

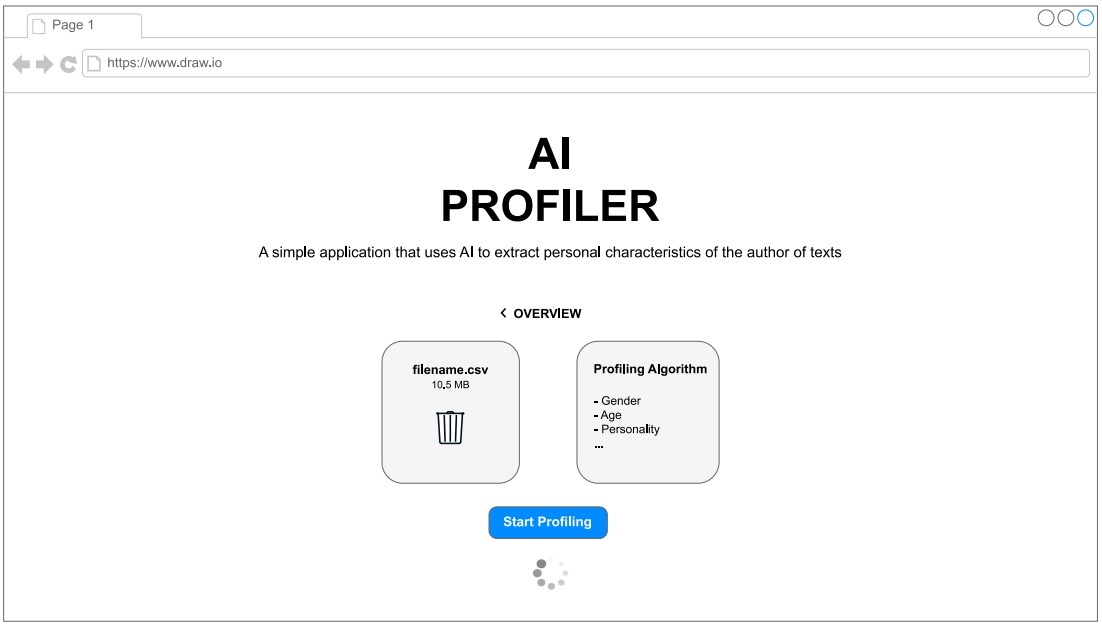

Figura 5.4: Prototipo del resumen del perfilado

Con respecto a la pantalla de ejemplos de *datasets*, como se puede ver en la Figura 5.5, se mostrará un pequeño ejemplo de 10-15 líneas aproximadamente. Asimismo, se le dará la opción al usuario de seleccionar el formato de *dataset*, ya sea NDJSON o CSV, dado que son los que soportará el *backend*.

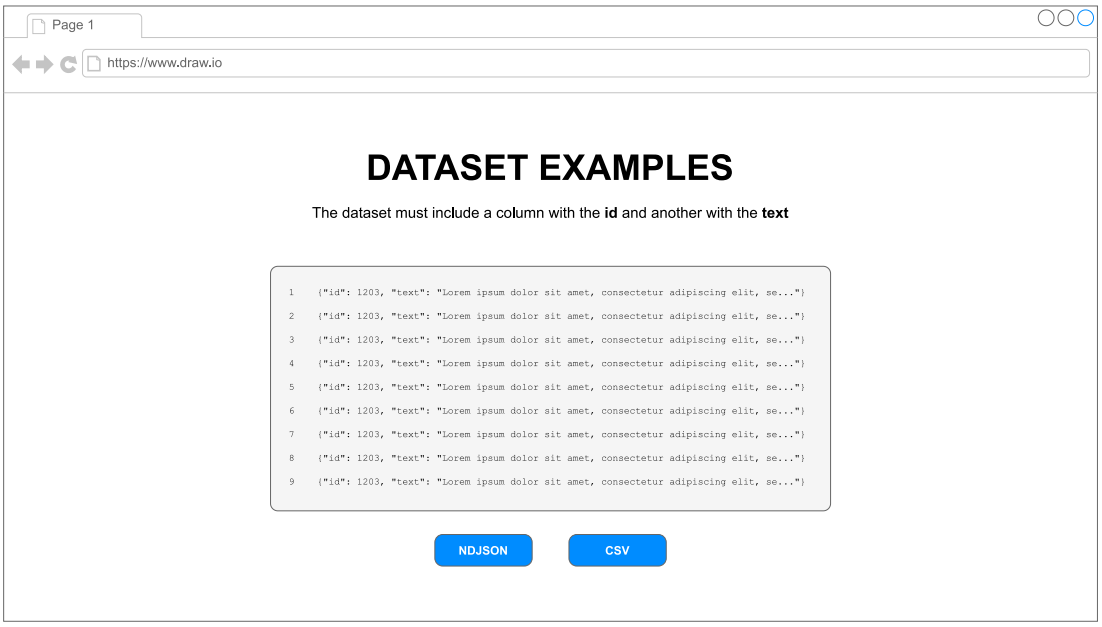

Figura 5.5: Prototipo de la pantalla de ejemplos de dataset

Cuando el perfilado haya finalizado, se mostrará al usuario un *dashboard* con los resultados obtenidos, como se puede ver en las Figuras 5.6, 5.7. Este *dashboard* contendrá los siguientes gráficos y elementos, los cuales variarán en función del algoritmo utilizado:

- **Datos generales**: El *dashboard* contiene una primera sección en la que aparece información de carácter general como son: el número total de personas perfiladas, el tiempo total del perfilado y el algoritmo utilizado.
- **Lista detallada de personas**: En esta sección, se muestra la lista de personas perfiladas de forma paginada. Además, la lista permitirá ser ordenada por cada uno de los diferentes campos, según se indica en la historia de usuario con identificador *6* de la Sección 4.1. Para ello, el usuario deberá hacer clic en el nombre de dicho campo, teniendo la opción de, volviendo a clicar, cambiar el orden de ordenación (ascendente o descendente).
- **Distribución de edad**: Para la distribución de edad, se ha optado por un gráfico de barras sobre otras opciones de representación categóricas como el gráfico circular. Esto se debe a que, teniendo cinco clases diferentes, el gráfico de barras permite una mejor visualización de los datos y una comparativa más clara entre ellos, dado que es más fácil comparar longitudes que áreas o ángulos. En la parte inferior del gráfico, se mostrará una tarjeta con la edad mediana.

- **Distribución de género**: En cuanto a la distribución de género, en cambio, se ha optado por un gráfico circular, puesto que solo existen dos clases y se desea resaltar la proporción de cada clase con respecto al total. A mayores, se mostrarán debajo del gráfico dos tarjetas con el número de personas de cada género junto con su porcentaje exacto.
- **Distribución de fama *(Martinc)***: De forma similar a la distribución de género, solo contamos con tres clases diferentes, por lo que se ha elegido un gráfico circular. Resaltar que este gráfico solo se mostrará en el caso de que se haya utilizado el algoritmo de Martinc.
- **Distribución de ocupación *(Martinc)***: En el caso de la distribución de ocupación, debido a que existen ocho clases distintas, se ha optado por un gráfico de barras. Este gráfico, al igual que el anterior, solo aparecerá si se ha empleado el algoritmo de Martinc para el perfilado.
- **Características personales *(Grivas)***: Ya que en este caso cada característica personal tendría asociado un valor decimal entre -0.5 y 0.5, se ha optado por un gráfico de barras. En este sentido, cabe resaltar que para mostrar dicho gráfico, es necesario implementar una funcionalidad que permita seleccionar a una persona de la lista detallada para mostrar sus características personales. Además, este gráfico solo estará disponible si se hace uso del algoritmo de Grivas.

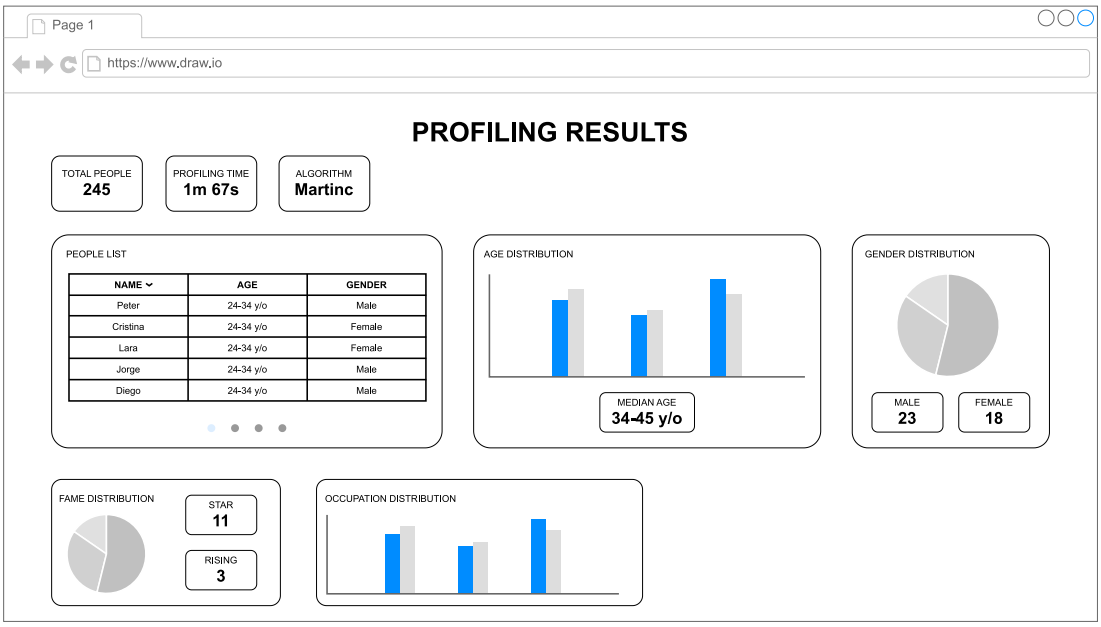

Figura 5.6: Prototipo del dashboard utilizando el algoritmo Martinc

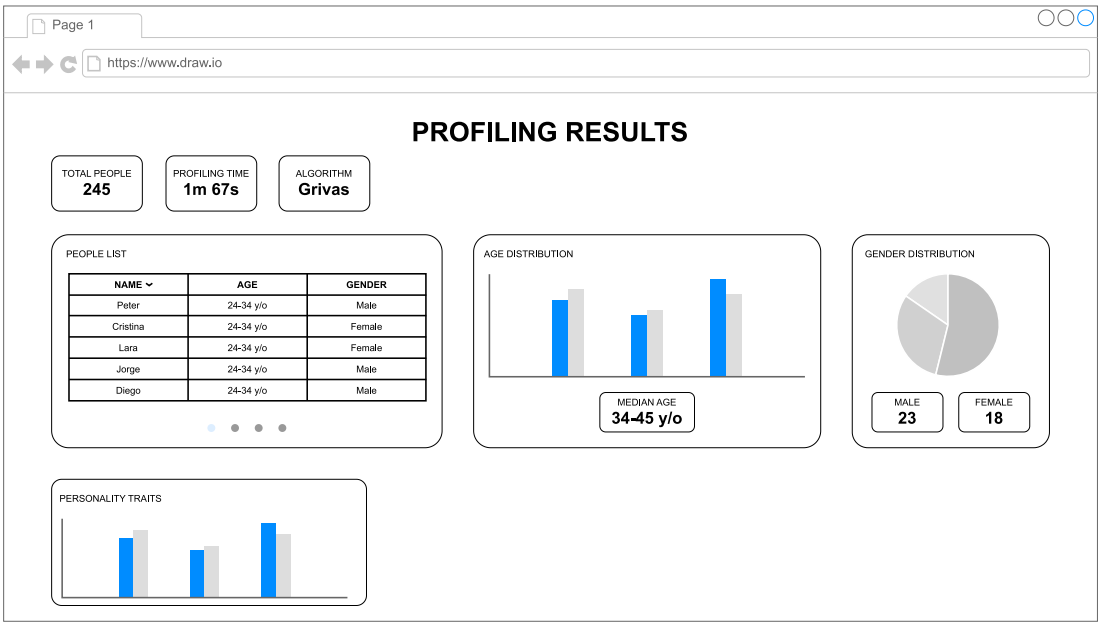

Figura 5.7: Prototipo del dashboard utilizando el algoritmo Grivas

Capítulo 6

Implementación

A lo largo de este capítulo, se expondrán las decisiones de implementación más relevantes tomadas durante el desarrollo del proyecto.

6.1 Diagrama de clases

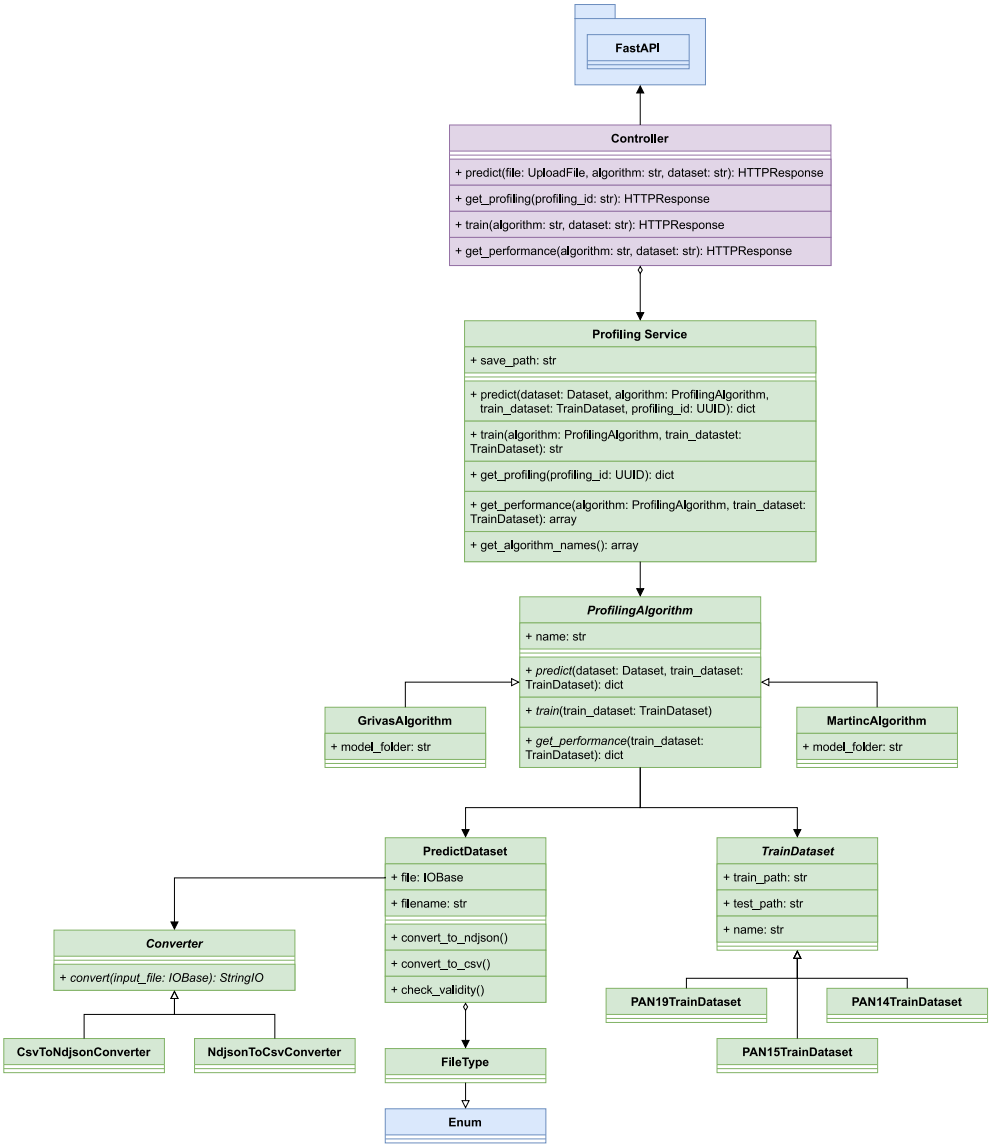

Figura 6.1: Diagrama de clases del *backend*

Capítulo 7

Caso de uso: #BLM

Capítulo 8

Planificación y costes

Capítulo 9

Conclusiones

9.1 Lecciones aprendidas

9.2 Trabajo futuro

Apéndices

Apéndice A

Material adicional

Bibliografía

[1] D. Otero, P. Martin-Rodilla, and J. Parapar, ``Building cultural heritage reference collections from social media through pooling strategies: The case of 2020's tensions over race and heritage,'' *J. Comput. Cult. Herit.*, vol. 15, no. 1, dec 2021. [Online]. Available: https://doi.org/10.1145/3477604

[2] M. We Are Social, ``Digital 2023 global overview report,'' 2023. [En línea]. Disponible en: https://datareportal.com/reports/digital-2022-global-overview-report

[3] B. Sydney, ``The permanent campaign: Inside the world of elite political operatives,'' 1980.

[4] J. Strömbäck, ``Four phases of mediatization: An analysis of the mediatization of politics,'' *The international journal of press/politics*, vol. 13, no. 3, pp. 228--246, 2008.

[5] B. Gallardo-Paúls and S. Enguix Oliver, *Pseudopolítica: el discurso político en las redes sociales.* Universitat de València, 2016.

[6] A. Saxena and U. Khanna, ``Advertising on social network sites: A structural equation modelling approach,'' *Vision*, vol. 17, no. 1, pp. 17--25, 2013.

[7] J. M. Machimbarrena, E. Calvete, L. Fernández-González, A. Álvarez-Bardón, L. Álvarez-Fernández, and J. González-Cabrera, ``Internet risks: An overview of victimization in cyberbullying, cyber dating abuse, sexting, online grooming and problematic internet use,'' *International journal of environmental research and public health*, vol. 15, no. 11, p. 2471, 2018.

[8] F. Rangel, P. Rosso, M. Koppel, E. Stamatatos, and G. Inches, ``Overview of the author profiling task at pan 2013,'' in *CLEF conference on multilingual and multimodal information access evaluation.* CELCT, 2013, pp. 352--365.

[9] M. Koppel, S. Argamon, and A. R. Shimoni, ``Automatically categorizing written texts by author gender,'' *Literary and linguistic computing*, vol. 17, no. 4, pp. 401--412, 2002.

[10] S. Argamon, M. Koppel, J. Fine, and A. R. Shimoni, ``Gender, genre, and writing style in formal written texts,'' *Text & talk*, vol. 23, no. 3, pp. 321--346, 2003.

[11] M. Corney, O. De Vel, A. Anderson, and G. Mohay, ``Gender-preferential text mining of e-mail discourse,'' in *18th Annual Computer Security Applications Conference, 2002. Proceedings.* IEEE, 2002, pp. 282--289.

[12] J. Otterbacher, ``Inferring gender of movie reviewers: exploiting writing style, content and metadata,'' in *Proceedings of the 19th ACM international conference on Information and knowledge management*, 2010, pp. 369--378.

[13] F. Rangel, P. Rosso, I. Chugur, M. Potthast, M. Trenkmann, B. Stein, B. Verhoeven, and W. Daelemans, ``Overview of the 2nd author profiling task at pan 2014,'' in *CLEF 2014 Evaluation Labs and Workshop Working Notes Papers, Sheffield, UK, 2014*, 2014, pp. 1--30.

[14] F. Rangel, F. Celli, P. Rosso, M. Potthast, B. Stein, W. Daelemans *et al.*, ``Overview of the 3rd author profiling task at pan 2015,'' in *CLEF2015 Working Notes. Working Notes of CLEF 2015-Conference and Labs of the Evaluation forum.* Notebook Papers, 2015.

[15] F. Rangel, P. Rosso, B. Verhoeven, W. Daelemans, M. Potthast, and B. Stein, ``Overview of the 4th author profiling task at pan 2016: cross-genre evaluations,'' in *Working Notes Papers of the CLEF 2016 Evaluation Labs. CEUR Workshop Proceedings/Balog, Krisztian [edit.]; et al.*, 2016, pp. 750--784.

[16] P. S. Foundation, ``Python,'' https://www.python.org/, 1991.

[17] S. Ramírez, ``Fastapi,'' https://fastapi.tiangolo.com/, 2018.

[18] E. Evans, ``Domain-driven design,'' https://www.domainlanguage.com/ddd/, 2003.

[19] Vercel, ``Next.js,'' https://nextjs.org/, 2016.

[20] Facebook, ``React,'' https://reactjs.org/, 2013.

[21] Microsoft, ``Typescript,'' https://www.typescriptlang.org/, 2012.

[22] S. O. Team, ``Stack overflow developer survey 2023,'' feb 2023. [En línea]. Disponible en: https://survey.stackoverflow.co/2023/

[23] Vercel, ``Zod,'' https://zod.dev/, 2020.

[24] S. Team, ``Sass,'' https://sass-lang.com/, 2006.

[25] B. Team, ``Bootstrap,'' https://getbootstrap.com/, 2011.

[26] M.-U. Team, ``Material-ui,'' https://material-ui.com/, 2014.

[27] C. U. Team, ``Chakra ui,'' https://chakra-ui.com/, 2019.

[28] C. Team, ``Chart.js,'' https://www.chartjs.org/, 2013.

[29] F. Team, ``Framer motion,'' https://www.framer.com/motion/, 2019.

[30] S. learn Team, ``Scikit-learn,'' https://scikit-learn.org/, 2007.

[31] T. Team, ``Tqdm,'' https://tqdm.github.io/, 2015.

[32] P. S. Foundation, ``Pickle,'' https://docs.python.org/3/library/pickle.html, 1991.

[33] N. Team, ``Numpy,'' https://numpy.org/, 2006.

[34] Atlassian, ``Trello,'' https://trello.com/, 2011.

[35] G. Team, ``Git,'' https://git-scm.com/, 2005.

[36] GitHub, ``Github,'' https://github.com/, 2008.

[37] GitLab, ``Gitlab,'' https://gitlab.com/, 2011.

[38] Bitbucket, ``Bitbucket,'' https://bitbucket.org/, 2008.

[39] Microsoft, ``Visual studio code,'' https://code.visualstudio.com/, 2015.

[40] P. Team, ``Prettier,'' https://prettier.io/, 2017.

[41] B. Team, ``Black,'' https://black.readthedocs.io/, 2018.

[42] E. Team, ``Eslint,'' https://eslint.org/, 2013.

[43] E. Amodio, ``Gitlens,'' https://gitlens.amod.io/, 2017.

[44] Jgraph, ``Draw.io,'' https://www.diagrams.net/, 2012.

[45] C. Feuersänger, ``Pgfplots,'' https://ctan.org/pkg/pgfplots, 2007.

[46] Adobe, ``Adobe xd,'' https://www.adobe.com/products/xd.html, 2016.

[47] Figma, ``Figma,'' https://www.figma.com/, 2012.