UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

GRADO EN INGENIERÍA DEL SOFTWARE

EXTRACCIÓN Y ANÁLISIS DE SENTIMIENTOS Y TENDENCIAS SOBRE COVID-19 EN REDES SOCIALES

EXTRACTION AND ANALYSIS OF SENTIMENTS AND TRENDS ABOUT COVID-19 IN SOCIAL NETWORKS

Realizado por
EUGENIO LÓPEZ PORRAS

Tutorizado por
EDUARDO GUZMÁN DE LOS RISCOS

Departamento
DEPARTAMENTO DE LENGUAJES Y CIENCIAS DE LA COMPUTACIÓN

UNIVERSIDAD DE MÁLAGA

MÁLAGA, JUNIO 2021

Resumen

Durante la pandemia de COVID-19 ha aumentado el acceso y el uso de las redes sociales por parte de todos los grupos de edad y estratos sociales. El uso de las redes sociales tiene múltiples objetivos, entre los que se encuentran el entretenimiento, la comunicación familiar, profesional o grupal social, además de la búsqueda de información. Una de los principales características de esta pandemia es la diversidad de opiniones generadas con respecto al virus, las vacunas o la gestión de los gobiernos, afectadas en gran medidas por las llamadas "fake news", difusión de tratamientos falsos, información incorrecta sobre vacunas y bulos conspiranoicos.

Con este TFG se pretende construir una herramienta que mediante el uso de técnicas de extracción de datos y de Ciencia de Datos se pueda analizar tanto información general sobre los tuits y los usuarios, así como los sentimientos de los usuarios respecto a temas introducidos manualmente pero especialmente enfocado al COVID-19 de forma que se puedan establecer tendencias e incluso posteriores análisis más complejos.

Palabras clave:

twitter, análisis de datos, análisis de sentimientos, covid

Abstract

During the COVID-19 pandemic, access and use of social networks has increased by all age groups and social strata. The use of social networks has multiple objectives, among which are entertainment, family, professional or social group communication, in addition to the search for information. One of the main characteristics of this pandemic is the diversity of opinions generated regarding the virus, vaccines or government management, largely affected by so-called "fake news", spreading of false treatments, incorrect information about vaccines and conspiracy hoaxes.

The aim of this TFG is to build a tool that, through the use of data extraction and Data Science techniques, can analyze both general information about tweets and users, as well as the feelings of users regarding topics entered manually but especially focused on COVID-19 so that trends and even later more complex analyzes can be established.

Keywords:

twitter, data analysis, sentiment analysis, covid.

Índice

1

Introducción

1.1 Motivación

Este proyecto surge a partir de una propuesta de una línea de trabajo del tutor de Trabajo Fin de Grado (TFG) del autor. La principal motivación para realizar este trabajo consiste en realizar una herramienta que aporte alguna innovación en la tarea de análisis de datos y sentimientos sobre todo a partir de la pandemia mundial originada por el COVID-19, al mismo tiempo que se aplican conceptos adquiridos en la carrera del Grado en Ingeniería de Software y usando técnicas y tecnologías recientes que puedan tener interés académico en el autor.

1.2 Análisis de mercado y justificación.

Para justificar la creación de la aplicación se ha realizado un análisis de mercado, haciendo una investigación de los principales competidores de la aplicación en este contexto, centrándose en sus características principales y los problemas que presentan.

Aplicación	Características	Problemas
Social Searcher	- Es una aplicación web y móvil que proporciona muchos tipos de análisis en tiempo real, en varias redes sociales simultáneamente y una interfaz de usuario muy atractiva.	- No permite seleccionar una única red social. - No se puede elegir el número de mensajes a analizar. - No permite descargar los conjuntos de datos. - Las funciones gratuitas son muy limitadas.
Sentiment Viz	- Es una aplicación web que presenta muchos tipos de análisis en tiempo real y diferentes visualizaciones sobre mensajes extraídos de Twitter.	- No se puede cambiar el número de mensajes a analizar. - No permite descargar los conjuntos de datos. - La interfaz de usuario está anticuada. - Sin soporte desde 2019.
HappyGrumpy	- Es una aplicación web centrada en el análisis de sentimientos y relaciones entre personas y marcas en Twitter, clasifica los resultados como “Happy” o “Grumpy”.	- Solo hace un tipo de análisis (positivo o negativo). - No permite búsqueda por términos, solo marcas o usuarios.

		- Necesitas entrar con tu cuenta de Twitter para utilizarla. - Sin soporte desde 2019.
text2data	- Es un complemento para Excel y Google Sheets que analiza un conjunto de datos y realiza una análisis de sentimientos sobre él.	- No recoge datos por sí mismo. - Poca variedad de clasificación. - No es una aplicación independiente.
Sentiment Analysis for Twitter	- Es una aplicación móvil que proporciona un análisis de sentimientos en tiempo real de mensajes extraídos de Twitter.	- Solo hace un tipo de análisis (positivo o negativo). - La interfaz de usuario no es atractiva. - No permite descargar los conjuntos de datos. - Sin soporte desde 2018.

Tabla 1.1. Análisis de los competidores.

A través de este análisis, la aplicación aportará las siguientes características para superar a los competidores:

- El usuario puede elegir los términos que deben aparecer en los tuits a recolectar y la cantidad de tuits.

- Tanto la recolección como el análisis son en tiempo real.
- Posibilidad de descargar de forma sencilla los conjuntos de datos generados en formato .csv para facilitar posteriores análisis.
- Interfaz de usuario atractiva y fácil de usar.

Los principales competidores de la aplicación son Social Searcher y Sentiment Viz, pero ambos presentan graves problemas que la aplicación subsana (sobre todo no poder elegir el número de tuits a analizar ni la posibilidad de descargar los conjuntos de datos).

1.3 Objetivos

Para un desarrollo satisfactorio de la aplicación, se perseguirán los siguientes objetivos.

- Recolección y almacenamiento de tuits en tiempo real que contengan unas palabras introducidas manualmente por el usuario.
- Tratamiento del texto de los tuits recolectados usando técnicas de procesamiento de lenguaje natural (NLP).
- Visualización de los datos analizados usando diferentes tipos de gráficos.
- Posibilidad de descarga de los conjuntos de datos generados.
- Interfaz de usuario accesible, agradable y usable.

1.4 Estructura de la memoria

Los capítulos en los que los que se encuentra dividida la memoria son los siguientes:

Capítulo 2: Técnicas y tecnologías utilizadas.

Este capítulo está dedicado a describir las técnicas necesarias para realizar la recolección y tratamiento de los tuits, así como el posterior análisis de sentimientos. A su vez, también se describen las tecnologías utilizadas para el desarrollo de la aplicación.

Capítulo 3: Metodología.

En este capítulo se describirá la metodología de trabajo que se va a usar y se justificará su uso.

Capítulo 4: Especificación de requisitos.

Este capítulo tratará sobre los requisitos funcionales y no funcionales que debe satisfacer la aplicación desarrollada.

Capítulo 5: Diseño y estructura del sistema.

Esta sección describirá el diseño y la estructuración del sistema mediante diagramas estructurales y de interacción.

Capítulo 6: Implementación.

En este capítulo se describirá cómo se ha desarrollado la implementación de la aplicación y la justificación de las decisiones tomadas.

Capítulo 7: Pruebas realizadas.

Aquí se describirá la información sobre las pruebas realizadas sobre la aplicación, los requisitos afectados por esas pruebas y los resultados obtenidos mediante la elaboración de una plan maestro de pruebas (PMP).

Capítulo 8: Conclusiones.

Para finalizar la memoria, en este capítulo se analizará si se han cumplido los objetivos deseados para la aplicación y las posibles dificultades que hayan surgido en el desarrollo. Además, se detalla posibles mejoras y nuevas funcionalidades que pueden añadirse a la aplicación.

Apéndices:

Manual de Instalación.

Apéndice en el que se describen los pasos necesarios para instalar y hacer funcionar la aplicación de forma satisfactoria.

Manual de Usuario.

Documento destinado a explicar el funcionamiento de la aplicación y las posibles interacciones que se pueden darse entre el usuario y el sistema.

Plan de Pruebas.

Plan que establece la cronología y condiciones para la aplicación de las pruebas de manera que se obtenga un sistema que pueda entrar en operación con la totalidad de las funcionalidades requeridas para su funcionamiento.

2

Metodología utilizada

Para el desarrollo de este trabajo se ha decidido usar una metodología ágil, en concreto Scrum [1]. El uso de esta metodología se debe principalmente a que las características de este proyecto se corresponden con los proyectos a los que va orientado esta metodología (desarrollo evolutivo y flexible a cambios, autonomía y comunicación, planificación,...). Además, esta metodología ya ha sido usada por el autor en varias asignaturas del grado de Ingeniería de Software por lo que cuenta con experiencia en el uso de esta.

Esta metodología tiene una serie de elementos que la caracterizan. A continuación se describe en qué consisten y su correspondencia con elementos del proyecto:

- **The Scrum Team:** es el conjunto de personas que conforman el equipo de desarrollo para el proyecto. En este caso está compuesto exclusivamente por el autor.

- **Product backlog:** es una lista en la que se recogen todos los requerimientos iniciales del producto a desarrollar. Para este proyecto, los requerimientos se corresponden con los requisitos recogidos en el capítulo 3. Especificación de requisitos.

- **Product Owner:** es quien se encarga de organizar las características recogidas en el Product backlog y de maximizar el valor de la solución generada. Este rol lo desempeñan tanto el autor como su tutor.

- **Sprint Backlog:** es una lista en la que se incluyen las tareas específicas a realizar en un Sprint concreto. En este proyecto se construirán entre el autor y su tutor en cada Sprint Planning.

- **Sprint Planning:** es la reunión o reuniones en las que se planifica las tareas a realizar en el siguiente Sprint mediante un Sprint Backlog. Para este proyecto se realizarán entre el autor y su tutor y en ellas también se analizará el trabajo realizado, además, se intentará que se produzcan, como máximo, cada dos semanas.

- **Scrum Master:** es el líder del proyecto y quien dirige cada Sprint Planning. Este rol está desempeñado por el tutor.

3

Especificación de requisitos

3.1 Requisitos Funcionales

RF-1. Inserción de los datos en la aplicación.

RF-1.1. El usuario puede introducir en un campo de texto las palabras que quiere que aparezcan en los tuits a buscar.

RF-1.2. El usuario puede introducir en un campo numérico cuántos tuits quiere que se recolecten.

RF-2. Recolección de los tuits.

RF-2.1. El sistema recolecta tuits en tiempo real usando la API de Twitter.

RF-2.2. Los tuits recolectados deben incluir una o varias palabras de las que el usuario introduce en el campo habilitado para ello.

RF-2.3. El número de tuits recolectados debe ser, como máximo, el número que introduce el usuario en el campo habilitado para ello.

RF-2.4. El sistema debe procesar el texto de los tuits para facilitar el posterior análisis.

RF-2.5. El sistema almacenará los tuits en una base de datos y se borrarán con cada nuevo uso del sistema.

RF-3. Análisis de los tuits.

RF-3.1. El sistema ordena los tuits de mayor a menor número de retuits.

RF-3.2. El sistema analiza el usuario de cada tuit y ordena los usuarios de mayor a menor número de seguidores.

RF-3.3. El sistema agrupa los tuits por la hora en la que se escriben y los ordena por horas.

RF-3.4. El sistema clasifica los tuits según si su texto representa un sentimiento positivo, neutral o negativo.

RF-3.5. El sistema calcula cuántos tuits hay de cada sentimiento.

RF-3.6. El sistema analiza las palabras de cada tuit de sentimiento positivo y las ordena de mayor a menor número de apariciones.

RF-3.7. El sistema analiza las palabras de cada tuit de sentimiento neutral y las ordena de mayor a menor número de apariciones.

RF-3.8. El sistema analiza las palabras de cada tuit de sentimiento negativo y las ordena de mayor a menor número de apariciones.

RF-4. Representación.

RF-4.1. El sistema permite elegir entre dos tipos de representación: General o Sentimientos.

RF-4.2. Cuando el tipo de representación es “General”, el sistema muestra mediante un gráfico de barras los tuits de mayor a menor número de retuits.

RF-4.3. Cuando el tipo de representación es “General”, el sistema muestra mediante un gráfico de barras los usuarios de los tuits de mayor a menor número de seguidores.

RF-4.4. Cuando el tipo de representación es “General”, el sistema muestra mediante un histograma el número de tuits por hora.

RF-4.5. Cuando el tipo de representación es “Sentimientos”, el sistema muestra mediante un gráfico de sectores el número de tuits por cada sentimiento.

RF-4.6. Cuando el tipo de representación es “Sentimientos”, el sistema muestra mediante un gráfico de barras las palabras con más apariciones en tuits positivos.

RF-4.7. Cuando el tipo de representación es “Sentimientos”, el sistema muestra mediante un gráfico de barras las palabras con más apariciones en tuits neutrales.

RF-4.8. Cuando el tipo de representación es “Sentimientos”, el sistema muestra mediante un gráfico de barras las palabras con más apariciones en tuits negativos.

RF-5. Descarga de conjuntos de datos.

RF-5.1. El sistema permite descargar el conjunto de datos principal en formato csv.

RF-5.2. El sistema permite descargar los conjuntos de datos secundarios, usados para generar cada uno de los distintos gráficos.

3.2 Requisitos No Funcionales

RNF-1. Tiempo de respuesta. El sistema tiene que responder rápidamente a las peticiones que se realicen. El sistema tarda menos de 2 segundos en las peticiones de navegación entre páginas y 1 minuto en la petición de recolección de tuits por cada 500 tuits recolectados.

RNF-2. Funcionamiento esperado. Las operaciones realizadas deben producir un resultado acorde a lo que debería esperar cualquier usuario del sistema.

RNF-3. Interfaz de usuario. La interfaz debe ser usable, accesible y resultar atractiva para el usuario.

4

Tecnologías y técnicas utilizadas

4.1 Tecnologías utilizadas

Para crear la aplicación se han utilizado diversas tecnologías, frameworks y librerías que han facilitado su desarrollo. En este apartado se describen las principales características y la justificación de su uso.

4.1.1 Dash

Dash es el framework en Python más descargado para crear aplicaciones web de Ciencia de Datos y Aprendizaje Automático, por lo que resulta especialmente útil para análisis de datos, exploración de datos, visualización, modelado y control de instrumentos e informes. Proporciona tanto back-end como front-end y vincula elementos de la interfaz de usuario como menús desplegables, controles deslizantes y gráficos directamente al código analítico en Python. [2]

Es de código abierto y su primera versión es de Junio de 2017. [3]

Está desarrollado sobre *Plotly.js, React* y *Flask*. *Plotly.js* [4] es una biblioteca de código abierto de JavaScript de alto nivel que incluye más de 40 tipos de gráficos, incluidos gráficos 3D, gráficos estadísticos y mapas SVG.

React [5] es una librería de JavaScript mantenida tanto por Facebook como por la comunidad de software libre. Es de código abierto, declarativa y flexible para crear interfaces de usuario. Permite crear interfaces de cualquier tipo de dificultad ya que se basa en componer pequeñas e independientes piezas de código llamadas "componentes". Entre los sitios web que usan React se encuentran Netflix, PayPal, Reddit o Twitter [6].

Flask [7] es un framework ligero de aplicaciones web en Python pensado para el back-end y basado en WSGI (dispatching) y Jinja2 (plantillas web). No impone un arquitectura concreta y no necesita de capa de abstracción para la base de datos. Está especialmente pensado para ser minimalista y así construir sitios web de forma sencilla y rápida.

Además, incorpora *Bootstrap* mediante unos tipos de componentes llamados *dash-bootstrap-components,* los cuáles son muy fáciles de usar y permiten incorporar elementos de bootstrap de forma sencilla al sitio web, ayudando mucho a la hora de construir la interfaz de usuario.

Como alternativa a esta tecnología se consideró el uso de *Django* (un framework para el back-end del sitio web) junto a *React* pero dado que el proyecto es de pequeño tamaño, se estimó que el uso de *Dash* era más apropiado, ya que proporciona tanto back-end como front-end de una forma mucho más rápida.

4.1.2 MongoDB

MongoDB [8] es una base de datos noSQL de código abierto orientada a documentos en formato JSON (codificados en BSON), por lo que no hay esquema de la base de datos. Además, cuenta con particionamiento horizontal (*auto-sharding*) por división de rangos y consultas paralelas, junto a una buena escalabilidad.

La decisión de usar *MongoDB* se basa en que para este tipo de datos es más rápido y cómodo usar una base de datos no relacional, y la rapidez es uno de los requisitos principales de la aplicación. Dentro de las bases de datos no relaciones, se decidió usar *MongoDB* porque ya había trabajado con ella anteriormente y es muy fácil de configurar en Python gracias a la librería *Pymongo* [9].

4.1.3 Tweepy

Tweepy [10] es una librería de código abierto para Python que permite acceder a la API de Twitter, por lo que da acceso a operaciones como extracción de tuits, publicación de tuits, hacer retuit y seguir a otros usuarios.

Hay algunas alternativas a *Tweepy* como *Python Twitter Tools* o *twython* pero *Tweepy* es la mayoritariamente usada y la que más soporte presenta.

4.1.4 TextBlob

TextBlob [11] es una biblioteca de Python orientada a procesar datos textuales. Proporciona una API simple para realizar tareas de procesamiento del lenguaje natural (NLP). Entre las tareas que puede realizar se encuentran: análisis de sentimientos, etiquetado de textos, corrección de palabras, tokenización (separar texto en palabras y frases)…

La característica que se ha usado es la de análisis de sentimientos, la cual permite, dado un texto, clasificarlo en tres tipos de sentimientos: positivo, negativo o neutral, además de indicar con qué grado de intensidad (polaridad).

Como alternativa, existe *Vader* y realmente son muy parecidas, se eligió *TextBlob* porque tiene más soporte y la sintaxis es más intuitiva. Otra alternativa sería entrenar un modelo creado desde cero, pero ya existen varios estudios que demuestran que para la mayoría de casos el resultado obtenido por *TextBlob* o *Vader* es igual o superior a modelos que presenten un rendimiento aproximado, y la velocidad de procesamiento es muy importante para este proyecto [12].

4.2 Técnicas utilizadas

Para poder realizar una recogida y análisis de sentimientos de forma más eficiente y efectiva es necesario hacer un preprocesamiento del texto que incluyen los tuits. Este apartado está dedicado a describir qué técnicas se han utilizado para ello.

4.2.1 Eliminar y sustituir palabras o caracteres no necesarios

En el contenido de un tuit hay mucha información que no es interesante para realizar un análisis de sentimientos: menciones a otros usuarios, enlaces a imágenes o sitios web, caracteres no alfabéticos, palabras vacías (stopwords),... por lo que se han realizado las siguientes sustituciones usando patrones regex y librerías auxiliares:

- Menciones a otros usuarios: se sustituyen por “USER”.
- Enlaces: se sustituyen por “URL”.
- Emoticonos alfabéticos: se sustituyen por su sentimiento (por ejemplo, :(= sad).
- Palabras vacías: se eliminan mediante la lista que incluye *nltk.corpus*.
- Caracteres no alfabéticos: se eliminan.

- Más de dos letras iguales consecutivas: se reemplazan por solo dos letras (por ejemplo, “yeaaaaaah” quedaría como “yeaah”).

4.2.2 Stemming y lemmatization

Stemming es una técnica que usa unos algoritmos de derivación para reducir una palabra a su raíz o base. El funcionamiento de estos algoritmos es relativamente sencillo ya que se basan en eliminar una serie de prefijos y sufijos establecidos. Un ejemplo del funcionamiento de estos algoritmos puede ser el siguiente:

- Palabra original (entrada): Likely
- Raíz de la palabra (salida): Like

Lemmatization es una técnica más refinada que el *stemming* ya que para eliminar los prefijos y sufijos tiene además en cuenta unos diccionarios morfológicos que permiten hacer un análisis morfológico más preciso y dando lugar a un mejor resultado.

Estas técnicas son muy útiles en el análisis de sentimientos ya que permiten reducir el ruido en los datos, por el hecho de reducir todas las formas derivadas de una palabra a la base común. Además, una de las ventajas extra que proporciona es que elimina emoticonos y caracteres no reconocidos.

En esta aplicación se usa tanto *stemming* como *lemmatization* en el tratamiento de cada palabra mediante *WordNetLemmatizer* de la librería *nltk.corpus*.

5

Diseño y estructura del sistema

5.1 Modelado estructural.

5.1.1. Diagrama estructural de la base de datos.

En este apartado se describe la estructura de la colección en la que se encuentra los datos de los tuits almacenados en MongoDB. El objetivo de este diagrama es especificar de forma rápida y gráfica la relación entre los datos almacenados para entender mejor su tratamiento.

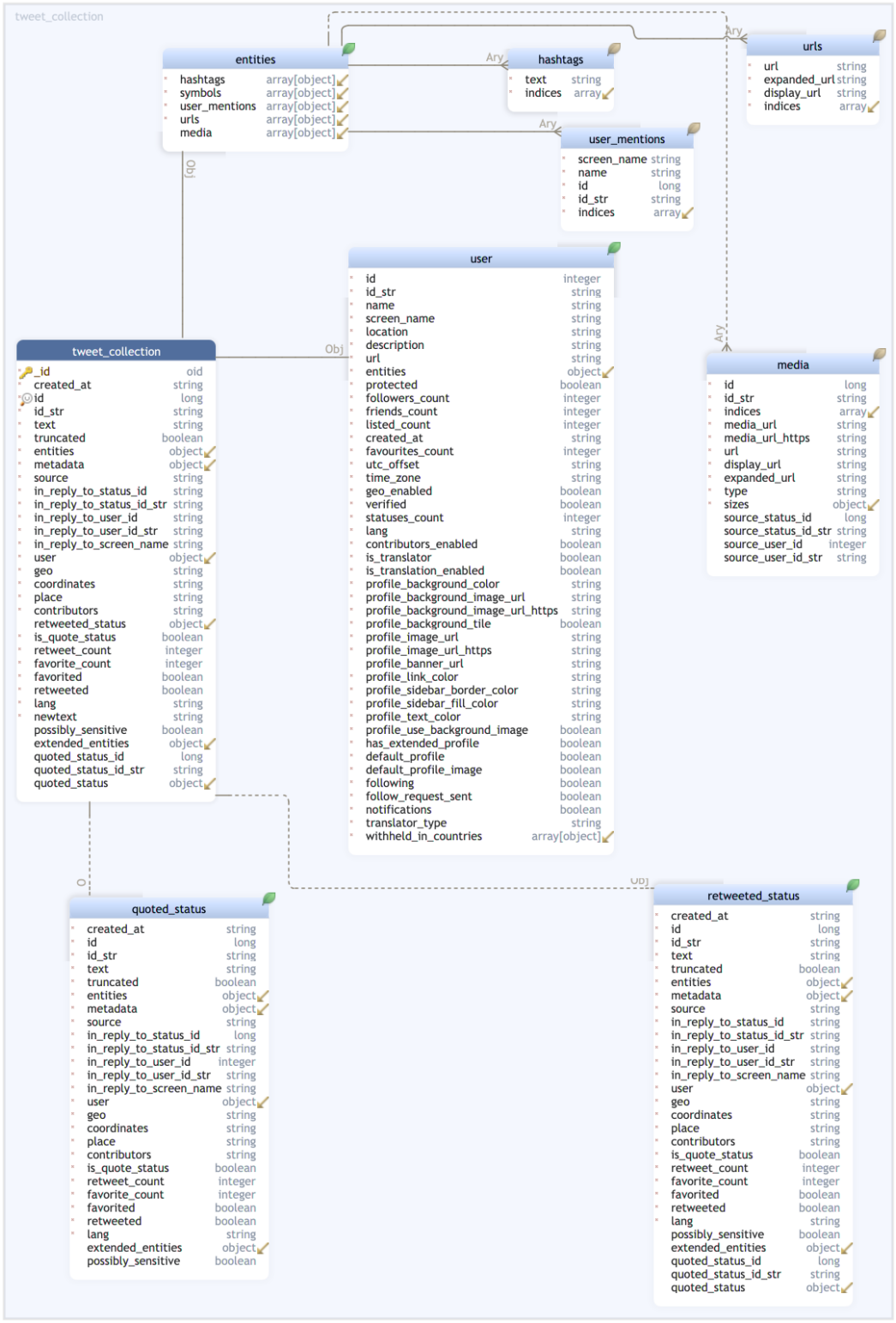


Figura 5.1 Diagrama estructural de la base de datos.

El diagrama total de la colección es mayor debido a la recursividad propia de los tuits pero las entidades principales son las siguientes:

- *tweet_collection*: es la entidad principal de la colección, ya que almacena los datos más importantes de cada tuit. Algunos de estos datos son:
    - *text*: el texto original del tuit.
    - *new_text*: texto del tuit preprocesado para poder hacer análisis de sentimientos.
    - *lang*: lenguaje del tuit, muy importante para obtener los tuits que estén escritos en inglés y así poder hacer un análisis de sentimientos válido.
    - *retweet_count*: número de retuits que ha recibido ese tuit.
    - *favorite_count*: número de "me gustas" que ha recibido el tuit.
    - *user*: objeto en el que se encuentra información del usuario que ha escrito el tuit.

- *quoted_status*: es una entidad que solo tiene datos cuando el tuit está citando a otro. Los datos son prácticamente iguales a *tweet_collection* ya que también almacena los datos del tuit.

- *quoted_status*: es una entidad que solo tiene datos cuando el tuit es un retuit a otro. Los datos son prácticamente iguales a *tweet_collection* ya que también almacena los datos del tuit.

- user: en esta entidad se almacenan los datos de los usuarios de twitter que han escrito alguno de los tuits recogidos. Algunos de estos datos son:
    - *name*: nombre único de cada usuario.
    - *screen_name*: nombre a mostrar de cada usuario, no tiene que ser único.

- *description*: descripción del perfil del usuario.

- *followers_count*: número de seguidores del usuario.

- *friends_count*: número de personas que sigue el usuario.

- *entities*: es una entidad que forma parte de los metadatos e incluye otras entidades que forman parte de los datos de un tuit:

  - *hashtags*: entidad que permite almacenar los hashtags que aparecen en un tuit.

  - *users_mentions*: entidad que permite almacenar las menciones a otros usuarios que aparecen en un tuit.

  - *urls*: entidad que permite almacenar los enlaces externos que aparecen en un tuit.

  - *media:* entidad que permite almacenar los archivos de imagen o vídeo que hay en un tuit.

5.1.2 Diagrama de componentes.

En esta sección se proporciona una descripción y representación a alto nivel de la estructura del sistema. Su principal objetivo es describir los diferentes componentes que forman el sistema y mostrar las interacciones entre estos.

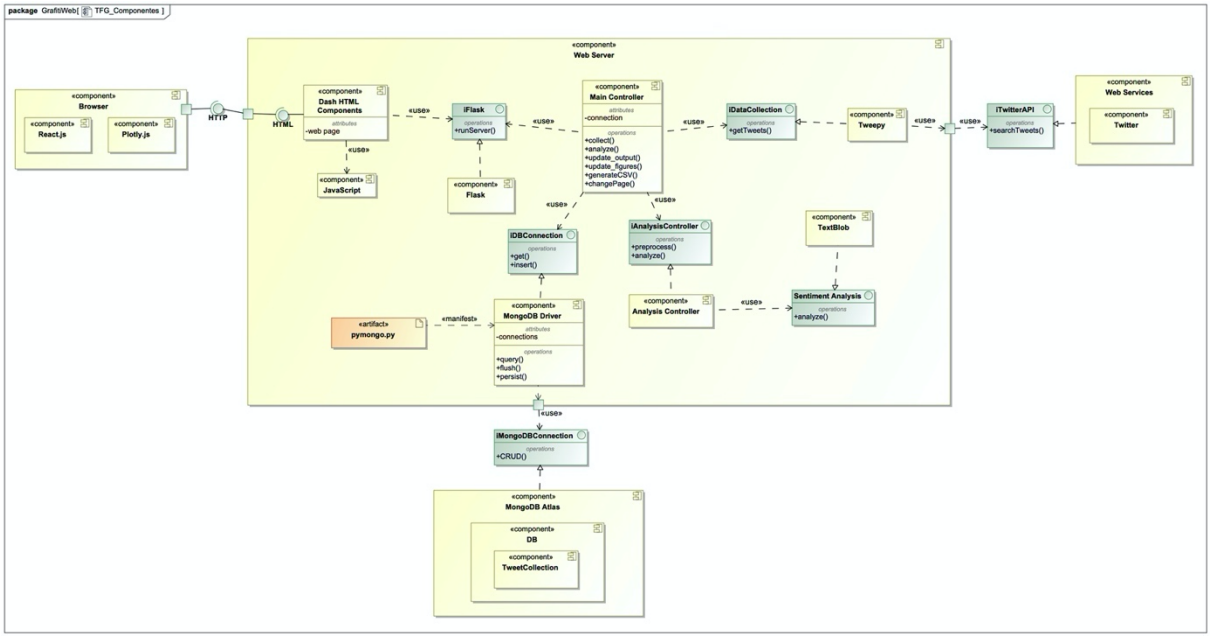


Figura 5.2 Diagrama de componentes.

En el diagrama de componentes se describen todos los componentes que aparecen en el sistema, las dependencias entre estos y las operaciones que proporciona cada uno.

A continuación, se define cada uno de los componentes individualmente:

- *MongoDBAtlas*: es un servicio que proporciona una base de datos MongoDB en nube global pensada para aplicaciones modernas que es distribuida y segura por defecto, además de estar disponible como servicio totalmente gestionado en AWS, Azure y Google Cloud. Esta base de datos está formada por colecciones en las que se almacenan los datos. En este caso la colección que almacena los datos se llama TweetCollection, para ver más detalles sobre los datos que contiene esta colección se puede acudir al diagrama estructural de la base de datos.

- *Pymongo*: es la librería de Python que proporciona el driver de conexión con la base de datos MongoDB, facilitando la configuración de la conexión y los métodos CRUD necesarios para trabajar con la base de datos.

- *Twitter API*: es la herramienta que proporciona los tuits que se van a analizar. Este servicio permite extraer una gran cantidad de textos a partir de los tuits, que habitualmente se corresponde con las opiniones y los sentimientos de los usuarios de *Twitter* acerca de un tema. Es sin duda, la fuente de información más usado en ciencia de datos sobre redes sociales y, sobre todo, en el análisis de sentimientos.

- *Tweepy*: es una librería de Python que facilita el acceso a la API de *Twitter* y proporciona diferentes mecanismos para realizar peticiones HTTP. Además, incluye herramientas para mantener sesión con el servidor OAuth de Twitter y funciones para hacer llamadas a las distintas herramientas de la API.

- *Textblob*: es una librería de Python orientada a procesar datos textuales. Proporciona una API simple para realizar tareas de procesamiento del lenguaje natural (NLP). Gracias a ella se puede realizar el análisis de sentimientos de forma rápida y con resultados satisfactorios.

- Analysis Controller: es el controlador que se encarga de facilitar los métodos de análisis de sentimientos mediante las llamadas a *Textblob*.

- Main Controller: es el controlador principal de la aplicación. Agrupa las diferentes funcionalidades en un único lugar para facilitar las conexiones. Proporciona las llamadas principales de la aplicación delegando parte de su funcionando al resto de controladores secundarios.

- *Flask:* es un framework ligero de aplicaciones web en Python pensado para el back-end y basado en WSGI (dispatching) y Jinja2 (plantillas web). Se encarga de

gestionar el arranque del servidor y la interacción entre la interfaz de usuario con los controladores.

- *Dash Html Components:* es una de las librerías que incorpora *Dash* para poder crear la interfaz de usuario usando elementos *HTML*. Junto al uso de otras librerías como *Dash Bootstrap* facilitan el desarrollo de la interfaz y permiten definir el funcionamiento de los diferentes elementos.

- *Browser*: este elemento es el que permite al usuario interactuar con el sistema. Recibe la representación de la interfaz procesada por Dash, Plotly.js y React.js y se la presenta al usuario para que pueda realizar las acciones sobre la aplicación y ver los resultados generados por esta.

5.1.3 Diagrama de despliegue.

En esta sección se proporciona una descripción y representación a bajo nivel de la estructura física del sistema. Su principal objetivo es mostrar las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos.

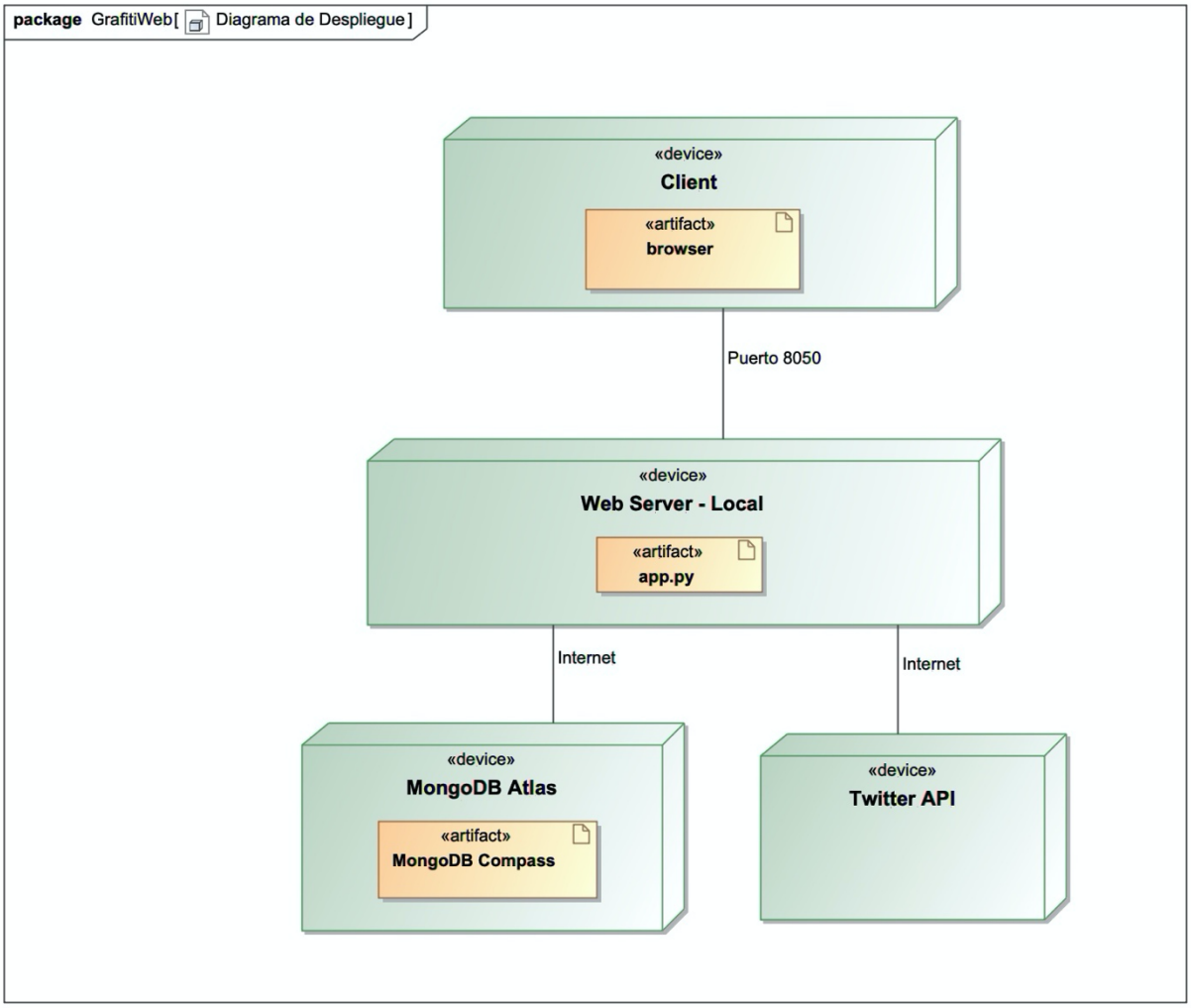


Figura 5.3 Diagrama de despliegue.

Los nodos que aparecen son:

- Client: Es el nodo que representa el dispositivo mediante el que el usuario accede a la aplicación. Al ser una aplicación local el puerto usado es el 8050 (Dash), si fuera una aplicación en la web el puerto sería 80 (*HTTP*), pero en cualquiera de los casos el artefacto que lo representa es el nevegador web.

- Web Server – Local: Este nodo hace referencia al dispositivo en el que se encuentra el servidor de la aplicación. Está gestionado por el artefacto app.py en el que Flask realiza toda la configuración necesaria para el funcionamiento del servidor.

- Twitter API: Nodo que incluye todas las herramientas para la interacción con datos de Twitter.

- *MongoDBAtlas:* Como ya se explicó en el diagrama de paquetes, MongoDBAtlas es el servicio en la nube que incluye la base de datos del sistema. El artefacto que gestiona este nodo es *MongoDBCompass*, la herramienta que permite la gestión del clúster a nivel local.

5.2 Modelos de interacción.

5.2.1. Diagramas de secuencia.

El propósito de los diagramas de secuencia definidos en esta sección es describir claramente las interacciones entre los diferentes componentes del sistema al realizar ciertos tipos de operaciones, de forma que se entienda el funcionamiento del sistema en conjunto ante ciertas acciones.

Se han realizado dos diagramas de secuencia, los cuáles describen el comportamiento de los diferentes componentes del sistema antes las dos tareas principales que un usuario puede realizar en el sistema.

Visualización de los gráficos resultantes del análisis de sentimientos

El objetivo de este diagrama es describir en profundidad el orden de operaciones que se realizan en una tarea que engloba la mayoría de componentes del sistema.

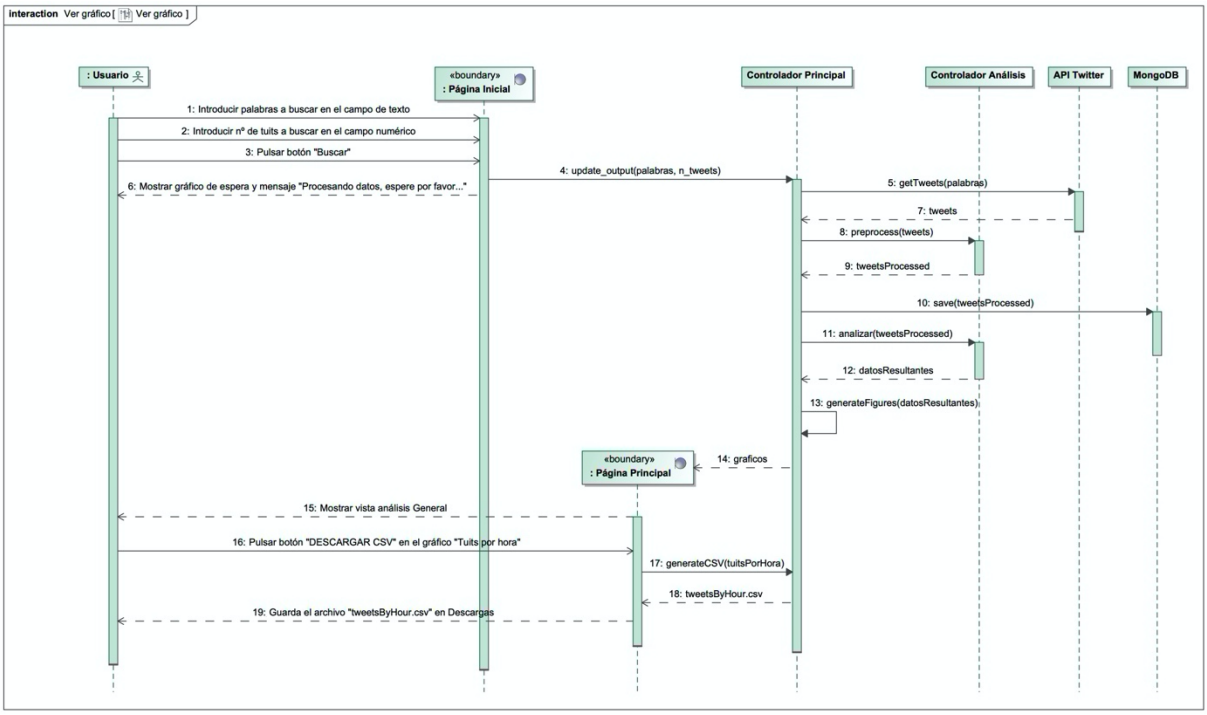


Figura 5.4 Diagrama de secuencia "Visualización gráficos resultantes".

Los aspectos más interesantes del diagrama son:

- *Página Inicial*: es la página inicial del programa, en ella se muestra la interfaz que permite al usuario introducir los campos necesarios para elaborar el análisis y luego transmitirlos al controlador principal.
- *Controlador Principal*: es el controlador principal que gestiona el comportamiento del sistema. Su funcionamiento ya se ha explicado anteriormente, pero en este diagrama se aprecia la interacción con la API de Twitter, el controlador de análisis y la Base de Datos y que el orden de las operaciones es el esperado.
- *Controlador Análisis:* es el controlador encargado del análisis de los textos, tanto del preprocesamiento de los tuits antes de guardarlos en la Base de Datos como de realizar el análisis de sentimientos.
- *MongoDB:* es la base de datos del sistema. Para operar con ella se utiliza Tweepy que permite llevar a cabo la operación *save(tweets)* con el fin de almacenar los tuits con el preprocesamiento ya realizado.

- *Página Principal*: es la página principal del programa, en ella se muestra el desplegable que permite seleccionar el tipo de análisis y los gráficos resultantes de haber realizado el análisis y con los que el usuario puede observar los resultados.

Descarga del conjunto de datos de uno de los gráficos del análisis general

El objetivo de este diagrama es identificar las diferencias en el comportamiento del sistema cuando se realiza otra actividad específica como es la descarga de un documento csv que incluye uno de los conjuntos de datos resultantes del análisis.

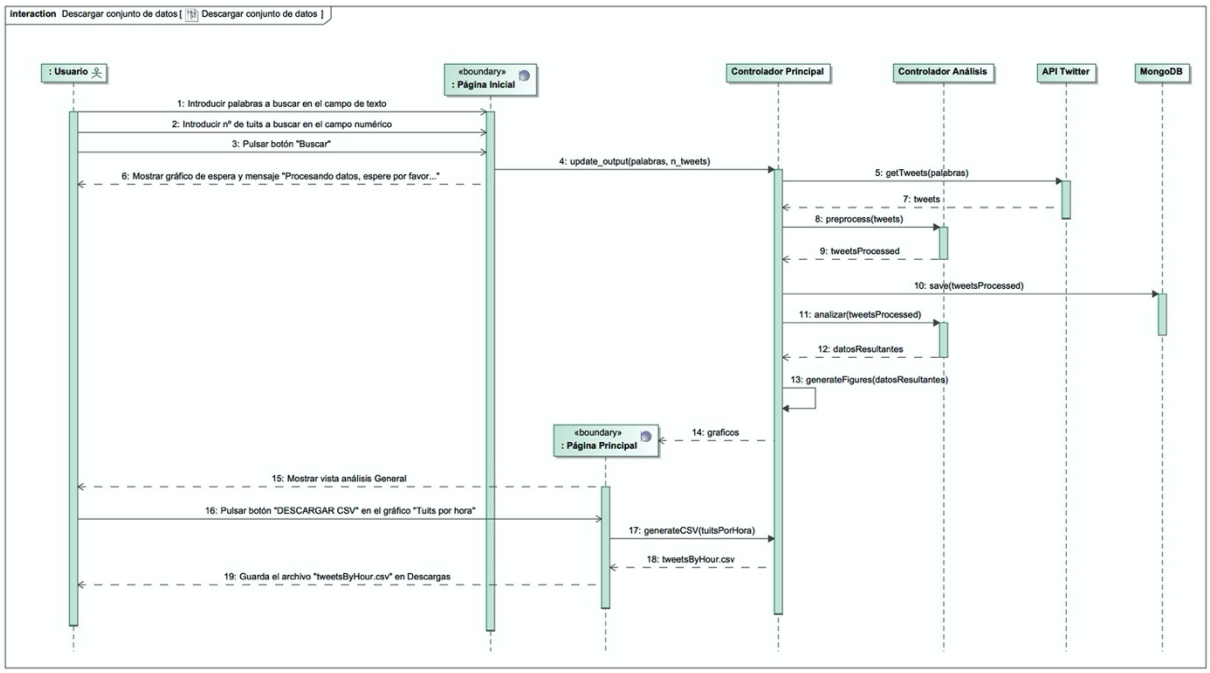


Figura 5.5 Diagrama de secuencia "Descarga de un conjunto de datos".

El comportamiento inicial del programa siempre es el mismo ya que hay que realizar el análisis de los tuits, la diferencia se encuentra tras realizar el análisis que en lugar de cambiar el tipo de análisis, se pulsa el botón "DESCARGAR CSV" de uno de los gráficos generados a partir del análisis generado y se observa que se llama a la función *generateCSV()* del controlador principal, que genera el fichero csv con el conjunto de datos esperado y se produce la descarga.

6

Implementación

Este apartado está dedicado a describir las archivos principales de la aplicación, de modo que quede claro su funcionamiento y a qué parte de la aplicación corresponde cada una.

6.1 Index.py

Este archivo se encarga de la página base de la aplicación. Incluye la parte gráfica de la barra de navegación que aparece en la parte superior de la aplicación y del mensaje de error cuando se va a una página que no existe en la aplicación.

Además, incluye parte del comportamiento del controlador principal de la aplicación ya que se encarga del funcionamiento del redireccionamiento entre páginas, mediante de *callback* "display_page" que recibe la URL del navegador y detecta si se ha producido un cambio, si la URL se refiere a una página de la aplicación redirecciona a la página correspondiente.

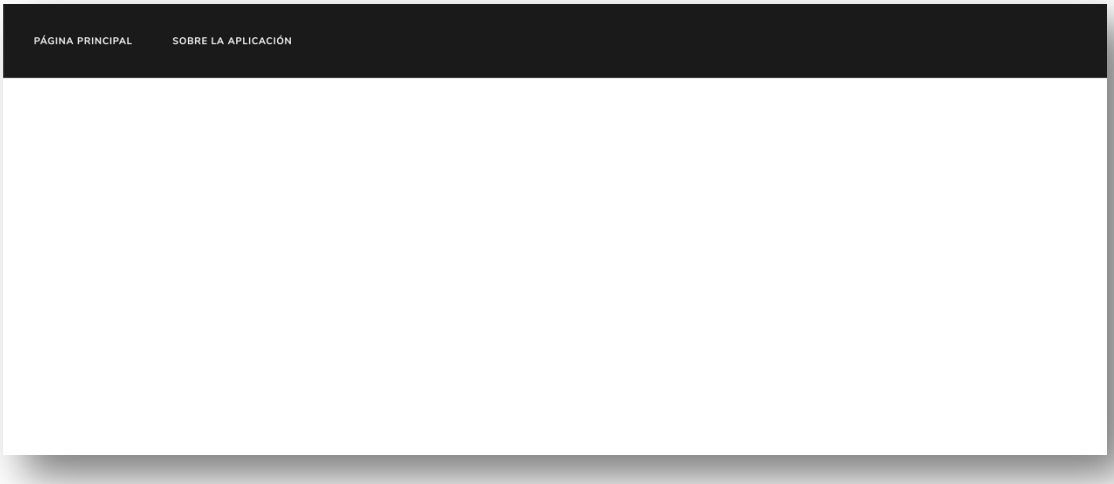


Figura 6.1 Página base.

Figura 6.2 Mensaje de error.

6.2 Home.py y Collecting.py

Estos archivos se encargan de la página de inicio de la aplicación y el funcionamiento general. Se ocupa de mostrar los campos de texto para el buscador y el número de tuits, y la información correspondiente.

También incluye el controlador de análisis de la aplicación y parte del controlador principal, ya que cuenta con las siguientes funciones:

- collect(): función que usa Tweepy para poder acceder a la API de Twitter y obtener los tuits necesarios para el análisis. Una vez recogidos, preprocesa cada uno de los tuits antes de llamar al método save(). El funcionamiento del preprocesamiento en detalle se describe en el apartado ***4.2 Técnicas utilizadas.***

- save(): función que usa el driver de MongoDB para guardar los tuits que han sido previamente preprocesados.

- analyze(): función que usa TextBlob para asignar un sentimiento a cada tuit. Una vez se ha analizado cada tuit, se generan los conjuntos de datos que son necesarios para cada gráfico.

- update_output(): función de *callback* que recibe la pulsación del botón "BUSCAR", llama a las funciones collect() y analyze() y muestra un gráfico y mensaje indicando que se están procesando los datos.

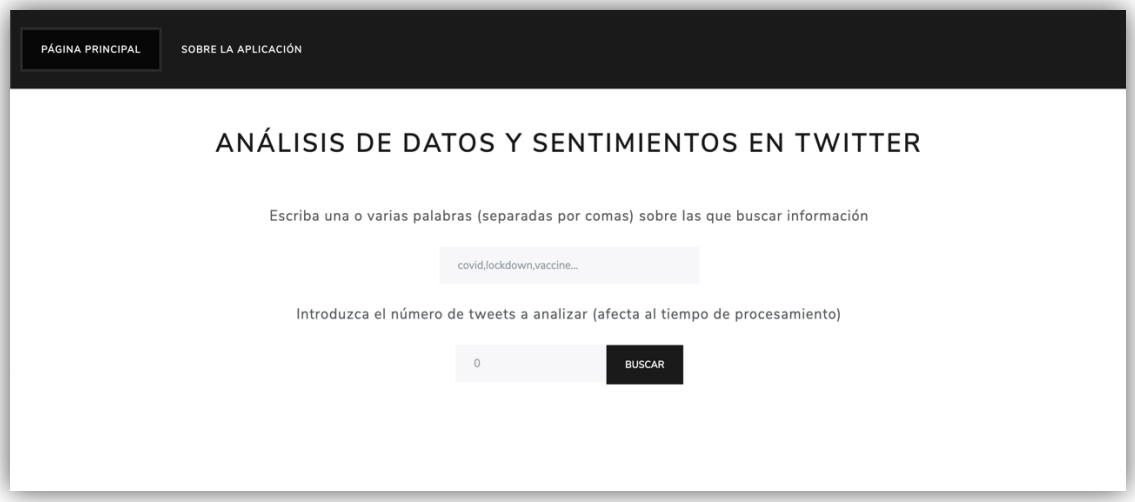


Figura 6.3 Página principal.

6.3 About.py

Este archivo solo está encargado de mostrar la interfaz de la página de información de la aplicación. Esta página incluye información sobre el propósito de la aplicación y un resumen de su funcionamiento.

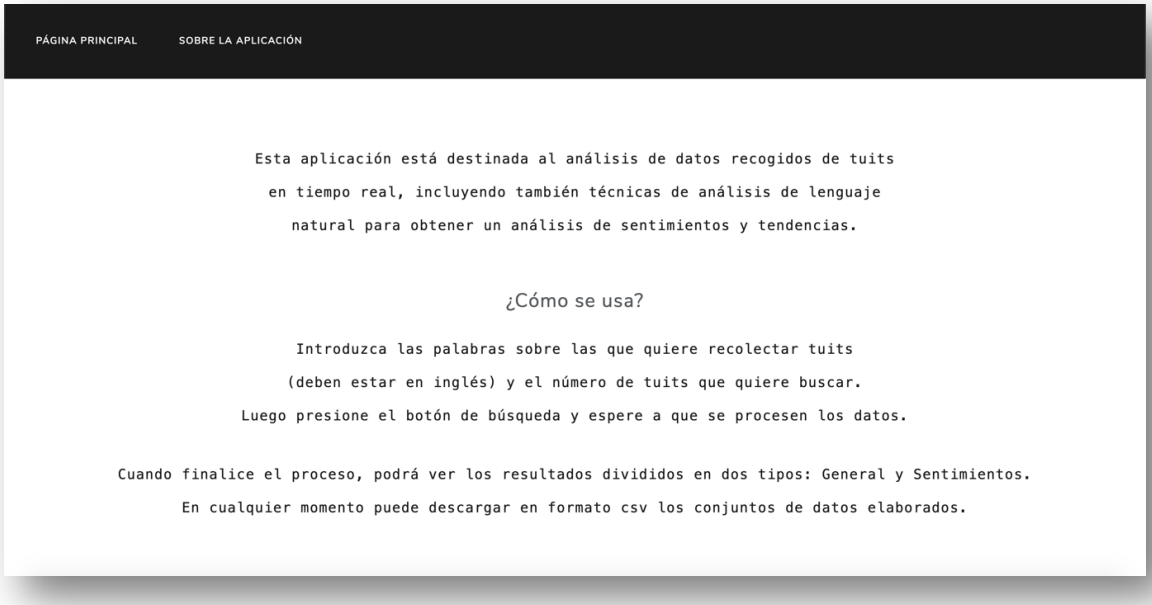


Figura 6.4 Página de información.

6.4 Main_page.py

Es el archivo encargado de la página base para la representación de los resultados. Incluye un desplegable que permite elegir cual es la representación que se quiere ver, como del botón que permite descargar todo el conjunto de datos en formato .csv.

Además, tiene parte del funcionamiento del controlador principal, ya que cuenta con las siguientes funciones:

- update_figures(): función de *callback* que recibe la selección del desplegable y genera los gráficos relacionados con el análisis general o de sentimientos según el tipo elegido.
- generateCSV(): función que permite generar el csv del conjunto de datos elegidos, ya sea el conjunto de datos general o el específico de algún gráfico.

Figura 6.5 Página principal.

6.5 General.py

Es el archivo encargado de la página que representa los gráficos a partir de datos generales de los tuits. Se encarga de cargar los diferentes gráficos a partir de los conjuntos de datos obtenidos previamente y de permitir la descarga de cada conjunto de datos mediante un botón debajo de cada gráfico.

Los gráficos que se muestran son:

- Tuits ordenados por número de retuits.
- Usuarios ordenados por número de seguidores.
- Número de tuits recogidos en cada hora.

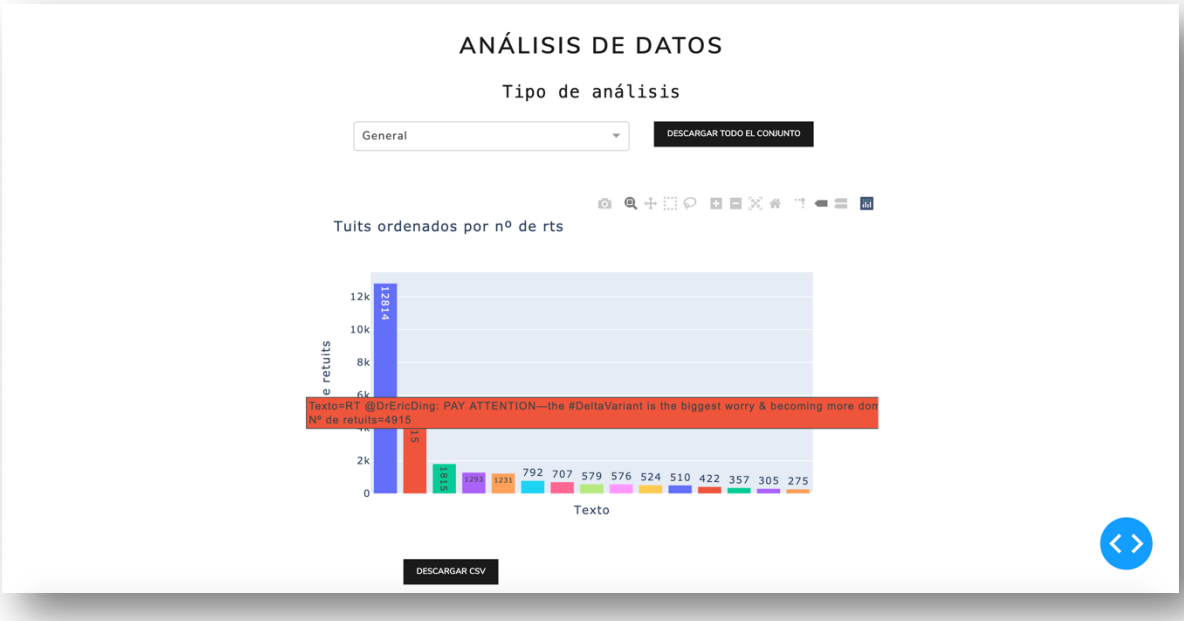


Figura 6.6 Gráfico de tuits ordenados por nº de retuits.

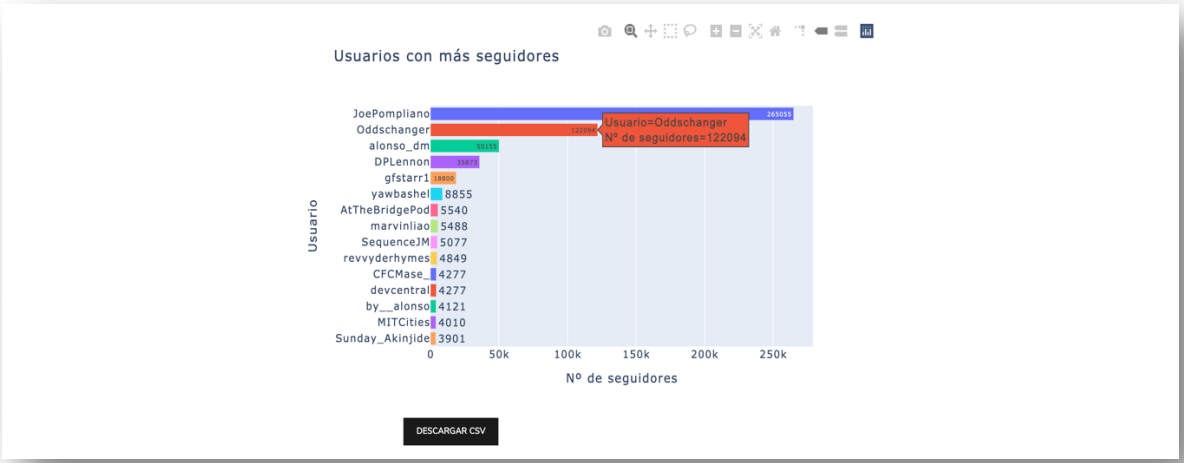


Figura 6.7 Gráfico de usuarios ordenados por nº de seguidores.

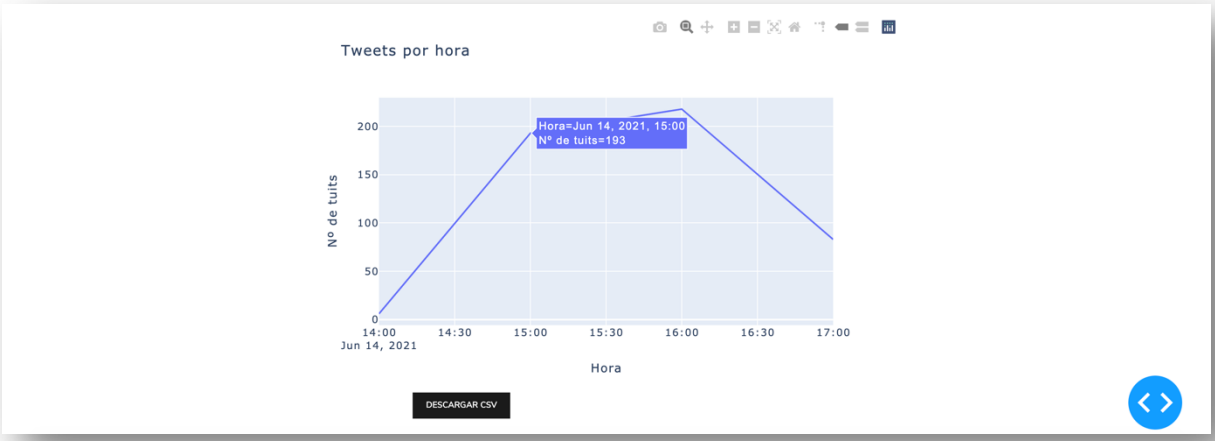


Figura 6.8 Gráfico de tuits recogidos por cada hora.

Para el gráfico de tuits ordenados por número de retuits se sigue el siguiente procedimiento:

- Transformar el conjunto de datos en un *pandas dataframe*.
- Aplicar la función sort_values() sobre el *dataframe* indicando que la columna a tener en cuenta es *retweet_count*.
- Seleccionar las columnas *text* y *retweet_count*, que son las que se van a utilizar para generar el grafo y su conjunto de datos.
- Obtener los 15 resultados con mayor número de retuits y generar un gráfico *plotly.bar* con ellos, de forma que se represente en un gráfico de barras.

Para el gráfico de usuarios ordenados por número de seguidores se sigue el siguiente procedimiento:

- Transformar el conjunto de datos en un *pandas dataframe*.
- Eliminar usuarios duplicados (un usuario puede haber escrito varios tuits de los recolectados).
- Aplicar la función sort_values() sobre el *dataframe* indicando que la columna a tener en cuenta es *user_followers*.

- Seleccionar las columnas *username* y *user_followers,* que son las que se van a utilizar para generar el grafo y su conjunto de datos.

- Obtener los 15 resultados con mayor número de retuits y generar un gráfico *plotly.bar* con ellos, de forma que se represente en un gráfico de barras.

Para el gráfico de tuits recogidos por hora se sigue el siguiente procedimiento:

- Transformar el conjunto de datos en un *pandas dataframe.*

- Transforma el formato de fecha que tienen los tuits por defecto en un formato válido.

- Obtener la hora a partir de la fecha y guardarla en otra columna del *dataframe* llamada *date.*

- Aplicar la función value_counts() sobre el *dataframe* indicando que la columna a tener en cuenta es *date,* de forma que se guarde el número de tuits por cada hora en una nueva columa llamada *date_count.*

- Seleccionar las columnas *date* y *date_count,* que son las que se van a utilizar para generar el grafo y su conjunto de datos.

- Obtener los 15 resultados con hora más reciente y generar un gráfico *plotly.line* con ellos, de forma que se represente en un histograma.

6.6 Sentiment.py

Es el archivo encargado de la página que representa los gráficos a partir de datos obtenidos del análisis de sentimientos de los tuits. Se encarga de cargar los diferentes gráficos a partir de los conjuntos de datos obtenidos previamente y de permitir la descarga de cada conjunto de datos mediante un botón debajo de cada gráfico. Los gráficos que se muestran son:

- Nº de tuits por cada sentimiento.

- Palabras más usadas en tuits positivos.

- Palabras más usadas en tuits negativos.

- Palabras más usadas en tuits neutrales.

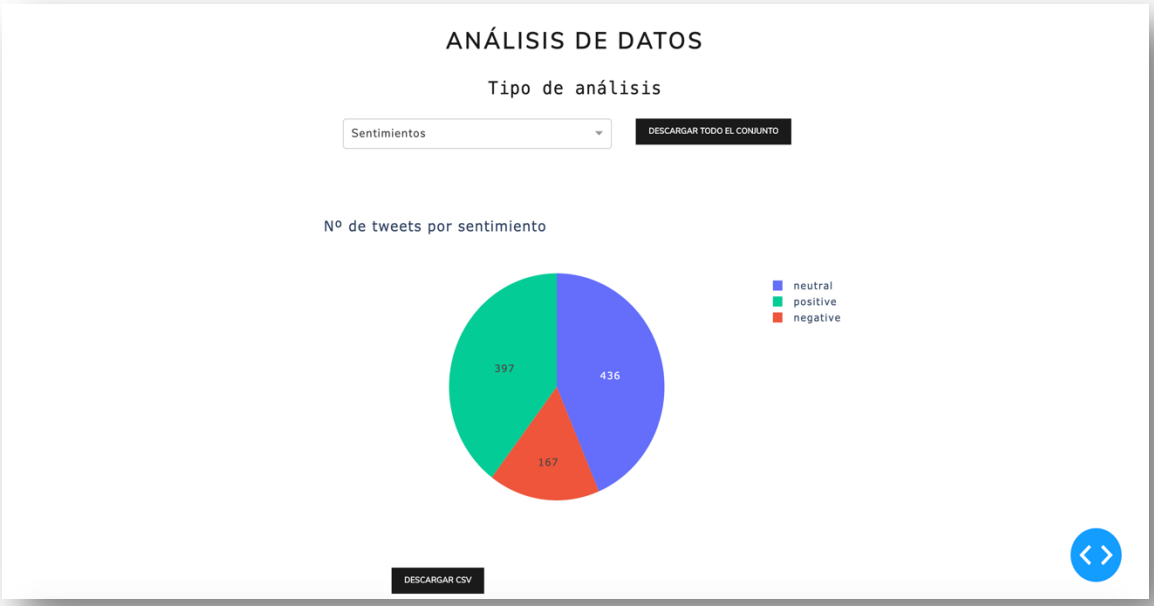


Figura 6.9 Gráfico de nº de tuits por sentimiento.

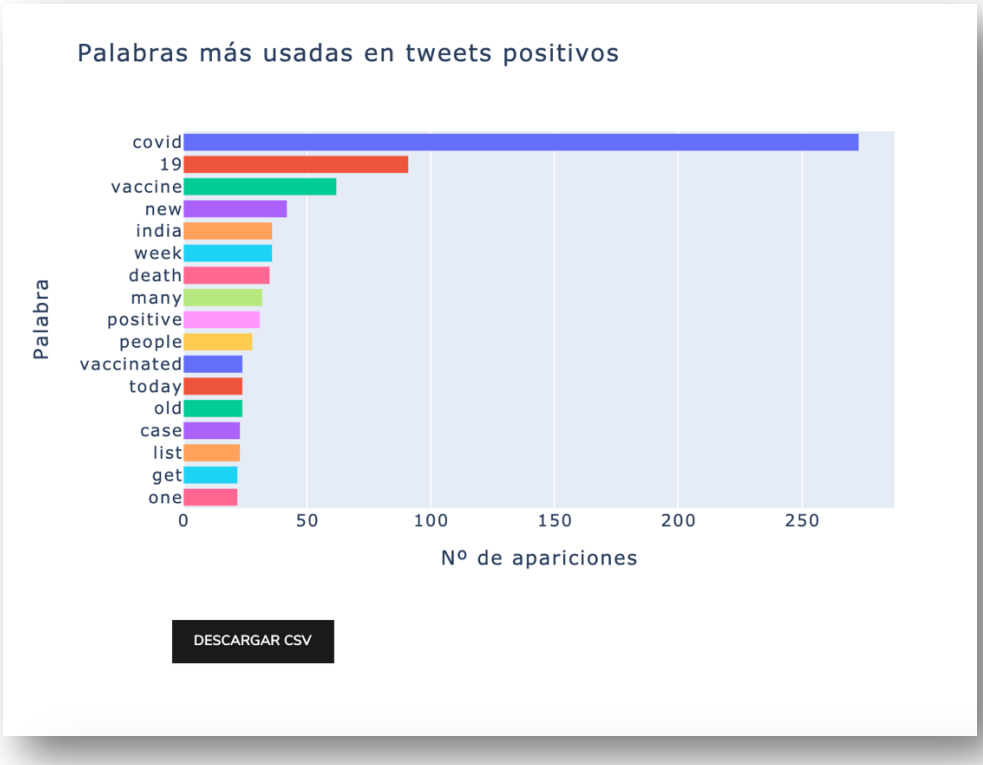


Figura 6.10 Gráfico de palabras más usadas en tuits positivos.

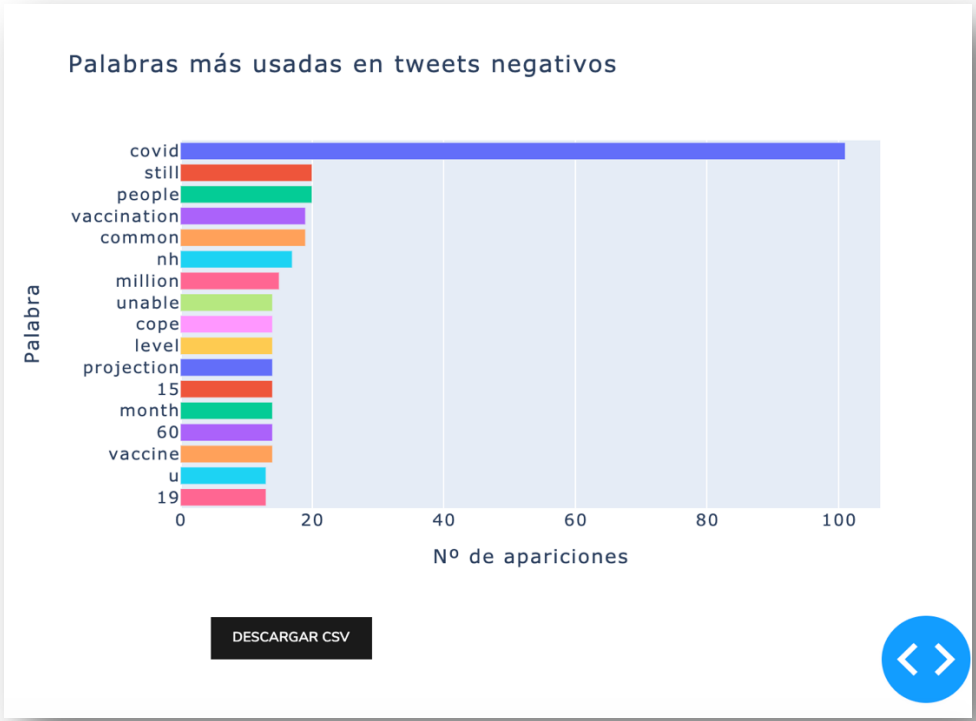


Figura 6.11 Gráfico de palabras más usadas en tuits negativos.

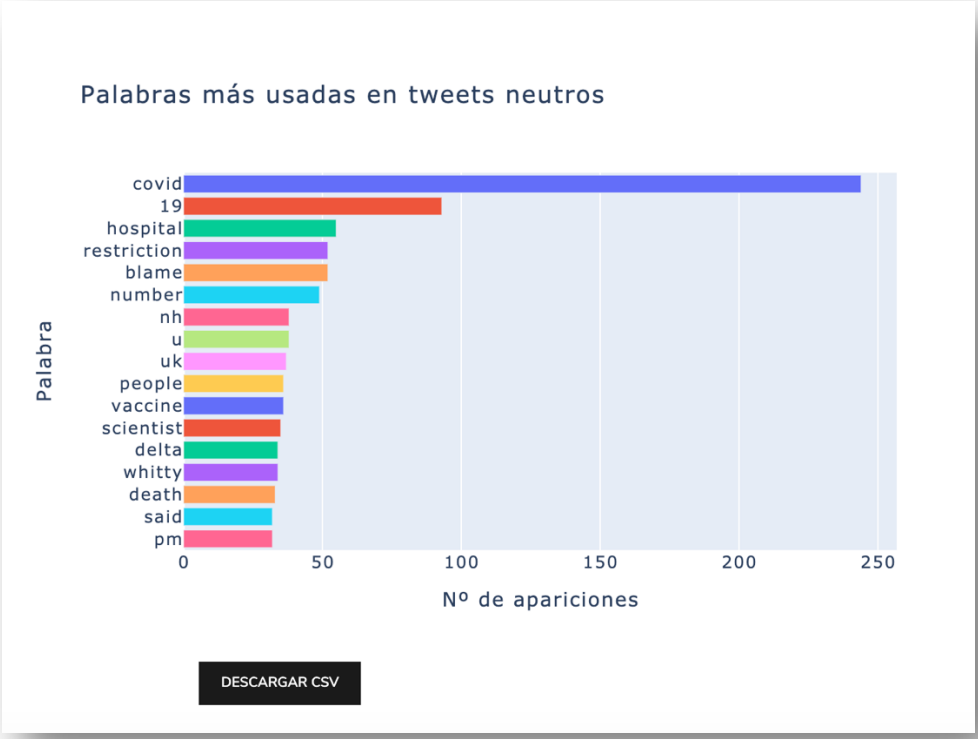


Figura 6.12 Gráfico de palabras más usadas en tuits neutrales.

Para el gráfico de número de tuits por sentimiento se sigue el siguiente procedimiento:

- Transformar el conjunto de datos en un *pandas dataframe*.

- Aplicar la función value_counts() sobre el *dataframe* indicando que la columna a tener en cuenta es *sentiment,* de forma que se guarde el número de tuits por sentimiento en una nueva columa llamada *counts.*

- Seleccionar las columnas *sentiment* y *count,* que son las que se van a utilizar para generar el grafo y su conjunto de datos.

- Obtener los resultados y generar un gráfico *plotly.pie* con ellos, de forma que se represente en un gráfico de sectores.

Para los gráficos de palabras más usadas por sentimiento se sigue el siguiente procedimiento, solo cambiando el sentimiento sobre el que buscar palabras:

- Transformar el conjunto de datos en un *pandas dataframe*.

- Seleccionar solo las filas en las que el sentimiento sea el deseado (columna *sentiment*).

- Recorrer todos los textos de los tuits y guardar una lista formada por todas las palabras.

- Eliminar de esa lista las palabras "URL", "USER" y "rt", ya que son las que genera el preprocesamiento y no queremos que se utilicen.

- Utilizar la función *Counter.most_common(18)* de la librería *collections* para obtener un nuevo *dataframe* con las 18 palabras que más aparecen y su frecuencia.

- Establecer el nombre de las columnas del *dataframe*, en este caso *word* y *frequency.*

- Aplicar la función value_counts() sobre el *dataframe* indicando que la columna a tener en cuenta es *sentiment,* de forma que se guarde el número de tuits por sentimiento en una nueva columa llamada *counts.*

- Seleccionar las columnas *sentiment* y *count,* que son las que se van a utilizar para generar el grafo y su conjunto de datos.

- Obtener los resultados y generar un gráfico *plotly.pie* con ellos, de forma que se represente en un gráfico de sectores.

7

Pruebas

En este capítulo se explica cómo se han realizado las pruebas sobre la aplicación y los resultados obtenidos, de forma que se pueda validar que el funcionamiento de la aplicación es el esperado y los objetivos han sido cumplidos.

Para realizar las pruebas del sistema se han utilizado dos tipos:

- Validación mediante Plan de Pruebas.
- Validación mediante Pruebas de usabilidad.

7.1 Plan de Pruebas.

Un plan de pruebas [13] es un producto formal que define los objetivos de las pruebas del sistema, establece y coordina las estrategias de trabajo y proporciona un marco apropiado para desarrollar un plan paso a paso para las actividades de prueba del sistema.

Es un documento extenso, en el que se incluyen: las personas involucradas, los objetivos del plan de pruebas, la metodología a usar, módulos involucrados, las pruebas realizadas y los resultados.

Al ser tan extenso se ha dedicado un apéndice para él (véase ***Apéndice C: Plan de Pruebas***), por lo que solo se indicará aquí los resultados obtenidos.
El resultado de las pruebas fue **Satisfactorio**, en el que solo se detectó un fallo medio (externo al sistema) y un fallo leve.

7.2 Pruebas de usabilidad.

Las pruebas de usabilidad son un tipo de prueba muy útil en el diseño de interfaces gráficas de usuario para evaluar la usabilidad de una aplicación probando con los propios usuarios.
Existen diferentes métodos de evaluación: inspección, indagación y test.
El método elegido para probar esta aplicación es el método de **indagación**, en el que se trabaja hablando con los usuarios, observándolos, usando el sistema en el trabajo real, obteniendo respuestas a preguntas verbalmente o por escrito, etc.
A su vez, dentro de este tipo de método existen varios métodos concretos, de los cuales se ha elegido el **método de cuestionarios**.

La evaluación de la usabilidad a través de cuestionarios es un método menos flexible que las otras opciones, pero puede abarcar un grupo más grande y se puede analizar de manera más rigurosa, por lo que es muy útil para este tipo de proyectos. El método utiliza principalmente tres tipos de cuestionarios:
- **Cuestionario "Pre-test":** Es un cuestionario pensado para obtener información personal y el tipo de perfil que se ajusta al usuario antes de comenzar la prueba.
- **Cuestionario "Post-tarea":** En este cuestionario se hacen preguntas relacionadas con la tarea realizada para obtener opiniones y valoraciones sobre la tarea . Como en la aplicación solo hay una tarea principal, **no se hará este tipo de cuestionario**, ya que está pensando para sistemas en los que existan muchas tareas.

- **Cuestionario "Post-test":** Es el cuestionario pensado para obtener las opiniones de los usuarios una vez que hayan completado todas las tareas. Para realizar este cuestionario se va a utilizar el estándar *SUS* [14]. Este estándar utiliza las siguientes preguntas:

1. Creo que me gustaría visitar con frecuencia este website.
2. Encontré el sitio web innecesariamente complejo.
3. Pensé que era fácil utilizar el sitio web.
4. Creo que necesitaría del apoyo de un experto para recorrer el sitio web.
5. Aprecié que las diversas funciones del sitio web estaban bastante bien integradas.
6. Pensé que había demasiada inconsistencia en el sitio web.
7. Imagino que la mayoría de las personas aprenderían muy rápidamente a utilizar el sitio web.
8. Encontré el sitio web muy incómodo/pesado al recorrerlo.
9. Me sentí muy confiado en el manejo del sitio web.
10. Necesité aprender muchas cosas antes de manejarme en el sitio web.

Una vez realizadas las preguntas, se calcula el resultado de este forma:
"Primero suma las contribuciones de puntuación de cada elemento. La contribución de cada puntaje de cada pregunta variará de 0 a 4. Para las preguntas 1, 3, 5, 7 y 9, la contribución de puntaje es la posición de escala menos 1. Para las preguntas 2, 4, 6, 8 y 10, la contribución es 5 menos la posición de la escala. Multiplique la suma de los puntajes por 2.5 para obtener el valor total de SUS" [14].
La puntuación total varía de 0 a 100, siendo 100 el mejor valor obtenible.

7.2.1 Cuestionario "Pre-test".

Las preguntas elegidas para realizar el cuestionario "Pre-test" son las siguientes:

1. ¿Te consideras un usuario básico, intermedio o avanzado de aplicaciones web? (Básico, Intermedio, Avanzado)
2. ¿Sabes cómo funciona la red social Twitter? (Sí/No)
3. ¿Sabes qué es un análisis de datos? (Sí/No)
4. ¿Sabes qué es un análisis de sentimientos? (Sí/No)
5. ¿Alguna vez has interpretado resultados en gráficos? (Sí/No)

El cuestionario se ha realizado a 4 usuarios, los resultados son los siguientes:

	Usuario 1	Usuario 2	Usuario 3	Usuario 4
Pregunta 1	Avanzado	Intermedio	Intermedio	Básico
Pregunta 2	Sí	No	Sí	Sí
Pregunta 3	Sí	Sí	Sí	No
Pregunta 4	Sí	No	No	No
Pregunta 5	Sí	Sí	Sí	No

Tabla 6.1 Resultados cuestionario "Pre-test".

7.2.2 Cuestionario "Post-test".

Los usuarios son los mismos que en el test anterior. Los resultados siguiendo el estándar SUS son los siguientes:

	Usuario 1	Usuario 2	Usuario 3	Usuario 4
Pregunta 1	5	4	4	4
Pregunta 2	1	1	1	2
Pregunta 3	5	4	5	4
Pregunta 4	2	1	2	3
Pregunta 5	4	4	5	4

Pregunta 6	1	2	1	1
Pregunta 7	5	4	5	4
Pregunta 8	1	1	1	1
Pregunta 9	5	3	4	3
Pregunta 10	1	2	2	2
Total SUS	95	80	90	75

Tabla 6.2 Resultados cuestionario "Post-test".

7.2.3 Análisis de resultados obtenidos.

Los perfiles obtenidos en los cuestionarios de "Pre-test" son los siguientes:

- Un usuario avanzado que conoce cómo funciona Twitter, sabe lo que es tanto un análisis de datos como de sentimientos y está familiarizado con la interpretación de gráficos.

- Un usuario intermedio que no sabe cómo funciona Twitter ni un análisis de sentimientos pero sí sabe qué es un análisis de datos y ha interpretados gráficos.

- Un usuario intermedio que sabe cómo funciona Twitter, los análisis de datos y está familiarizado con la interpretación de gráficos pero que no sabe qué es un análisis de sentimientos.

- Un usuario básico que sabe cómo funciona Twitter pero desconoce por completo qué es un análisis de datos o de sentimientos y tampoco ha interpretado gráficos en detalle.

Los resultados de los cuestionarios "Post-test" dan lugar a una media de 85 en el estándar SUS, lo que se podría calificar como una muy buena interfaz llegando a ser casi excelente. Este resultado es muy positivo ya que uno de los principales objetivos al realizar la aplicación era que la interfaz de usuario fuera lo mejor posible y este resultado lo corrobora.

8

Conclusiones

El último capítulo de la memoria describe los problemas y limitaciones encontradas durante el desarrollo de la aplicación, posibles mejoras que podrían añadirse y un análisis global de los conocimientos adquiridos y el resultado final de la aplicación.

8.1 Limitaciones y problemas encontrados.

Durante el desarrollo del sistema han surgido varios problemas y limitaciones que han afectado en mayor o menor medida al desarrollo, en este apartado se describen los más importantes.

1. **Ubicación de los tuits**: En la primera concepción del proyecto, se pensó hacer varios análisis en los que se utilizara la ubicación de los tuits, como por ejemplo: países en los que más tuits se han enviado, países que envían más tuits positivos,... Pero en el desarrollo se descubrió que Twitter no facilita la ubicación desde la que se envían todos los tweets, así que la única forma de obtener la ubicación es a través de la ubicación que tiene el usuario establecida en su perfil. El problema es que esta ubicación no tiene ningún control exhaustivo (se puede poner un texto cualquiera y no es obligatorio) por lo que los resultados que se obtuvieron no eran nada satisfactorios, y se optó por descartar el uso de la ubicación de los tuits.

2. **Limitaciones de Dash**: Dash facilita muchas de las tareas necesarias para desarrollar la aplicación, en gran medida gracias a la gran cantidad de elementos que se acoplan para unificar funciones. El problema surge cuando intentas personalizar alguno de estos elementos de alguna forma que Dash no contempla. En estos casos se pierde esa unión de funciones que presentan los elementos que incluye, dando lugar a tener que hacer cambios en CSS personalizados o añadir clases extra que no se ajustan al resto del desarrollo y la mayoría de las veces es mejor intentar cambiar la idea que se quiere realizar.

3. **API de Twitter**: El uso de la API de Twitter es fundamental en la aplicación, ya que la recolección de tuits depende completamente de la velocidad y disponibilidad que tenga. Teniendo en cuenta esto, se ha apreciado que la API de Twitter puede ser bastante inestable. Esto es inapreciable cuando se obtiene una cantidad de tuits no demasiado grande, pero que sí tiene mucha importancia cuando se recolecta un gran número de tuits, ya que es un proceso que tarda

bastante minutos y resulta muy tedioso que haya que reiniciar la recolección de tuits por fallos en la disponibilidad de la API.

8.2 Posibles mejoras.

Durante el desarrollo y al término de la aplicación se han apreciado posibles mejoras que pueden resultar interesantes e incrementar el posible valor del sistema:

- Añadir mayor interacción con los gráficos.
- Nuevos tipos de análisis.
- Más gráficos para esos nuevos tipos de análisis.
- Uso de una base de datos premium que pueda hacer que la velocidad de análisis se reduzca.

8.3 Conclusiones finales.

En general, ha sido bastante alto el nivel de satisfacción del autor con respecto al proyecto ya que se han conseguido la mayoría de objetivos y las expectativas que se tenían en el proyecto.

También a nivel de desarrollo personal del autor ha sido bastante beneficioso el desarrollo de una aplicación utilizando Python, ya que no lo había usado durante las asignaturas de la carrera de Ingeniería de Software a un nivel tan avanzado, y el aplicar muchos de los conocimientos y metodologías aprendidos durante la carrera en un trabajo realizado por cuenta propia y en los que se aprecia su gran utilidad en un proyecto real.

Referencias

[1] *What is Scrum?* (2021). Scrum.Org. https://www.scrum.org/resources/what-is-scrum

[2] *Dash.* (2021, 9 abril). PyPI. https://pypi.org/project/dash/

[3] *Introducing Dash - Plotly.* (2019, 7 agosto). Medium. https://medium.com/plotly/introducing-dash-5ecf7191b503

[4] *Plotly JavaScript Graphing Library.* (s. f.). JavaScript | Plotly. Recuperado 9 de junio de 2021, de https://plotly.com/javascript/

[5] *React.* (s. f.). React. Recuperado 9 de junio de 2021, de https://es.reactjs.org/

[6] Shah, H. (2021, 10 mayo). *12 Popular Websites that use React in Production.* Insights on Latest Technologies - Simform Blog. https://www.simform.com/websites-use-react/

[7] *Flask Documentation.* (s. f.). Flask. Recuperado 9 de junio de 2021, de https://flask.palletsprojects.com/en/2.0.x/

[8] MongoDB. (s. f.). *La base de datos líder del mercado para aplicaciones modernas.* Recuperado 9 de junio de 2021, de https://www.mongodb.com/es

[9] *Pymongo.* (2021, 5 mayo). PyPI. https://pypi.org/project/pymongo/

[10] *Tweepy.* (s. f.). Tweepy. Recuperado 9 de junio de 2021, de https://www.tweepy.org/

[11] Loria, S. (s. f.). *TextBlob: Simplified Text Processing.* TextBlob. Recuperado 10 de junio de 2021, de https://textblob.readthedocs.io/en/dev/

[12] Rustam, F. (2021, febrero). *A performance comparison of supervised machine learning models for Covid-19 tweets sentiment analysis.* https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0245909

[13] *829-2008 - IEEE Standard for Software and System Test Documentation.* (2008, 18 julio). IEEE Standard | IEEE Xplore. https://ieeexplore.ieee.org/document/4578383

[14] Brooke, J. (1986). *SUS - A quick and dirty usability scale.* https://hell.meiert.org/core/pdf/sus.pdf

Apéndice A

Manual de Instalación

Requerimientos:

- Python 3: https://www.python.org/downloads/

- Dash Plotly: https://dash.plotly.com/installation

- Dash Core Components: https://dash.plotly.com/dash-core-components

- Dash Bootstrap Components: https://dash-bootstrap-components.opensource.faculty.ai/

- Pandas: https://pandas.pydata.org/pandas-docs/stable/getting_started/install.html

- Pymongo: https://pypi.org/project/pymongo/

Instalación:

Una vez instaladas las tecnologías necesarias solo hay que ejecutar el fichero index.py mediante Python3 para arrancar el servidor. Una vez arrancado el servidor, la aplicación es completamente funcional accediendo a la URL establecida para el acceso al servidor.

Actualmente la base de datos se encuentra en un clúster en la nube y accesible públicamente, si se quiere utilizar otra base de datos solo hay que crear la base de datos y cambiar la URL de acceso a la base de datos en el fichero home.py.

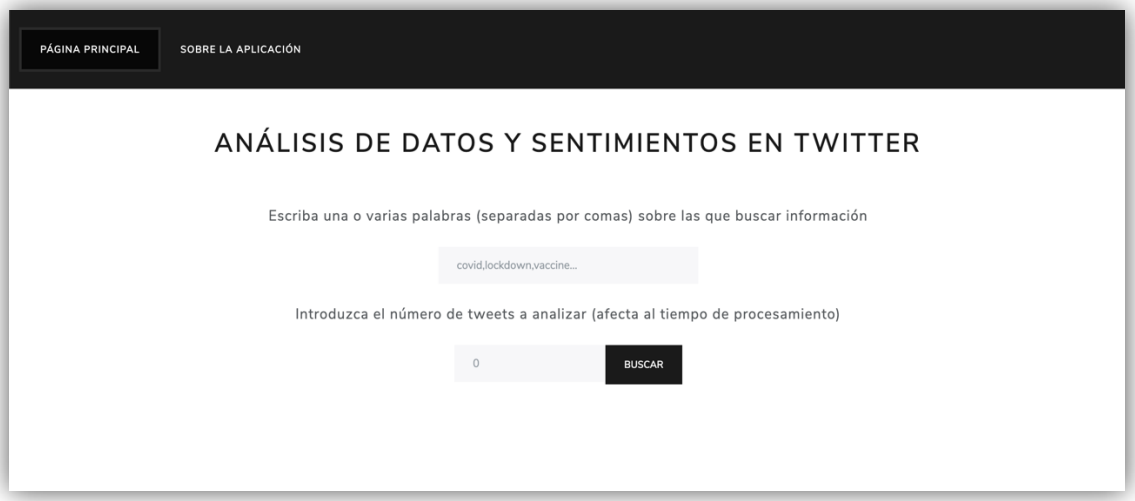
Las credenciales de la API de Twitter también son propias por lo que se pueden cambiar en el fichero collecting.py introduciendo las credenciales oportunas.

Apéndice B

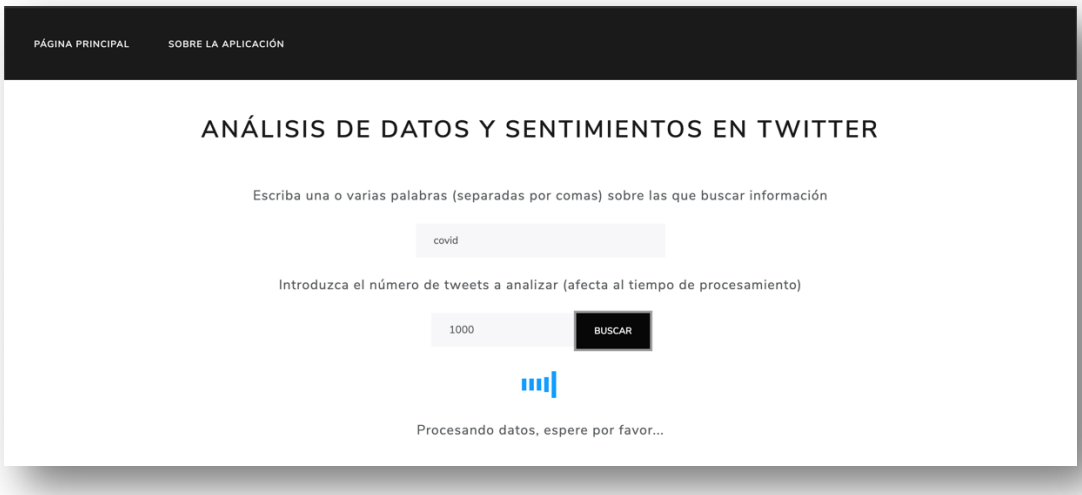
Manual de Usuario

Página principal:

Para empezar con el análisis es necesario introducir las palabras a buscar en los tuits (en ingles) y el número de tuits a buscar, cada uno en su campo correspondiente.



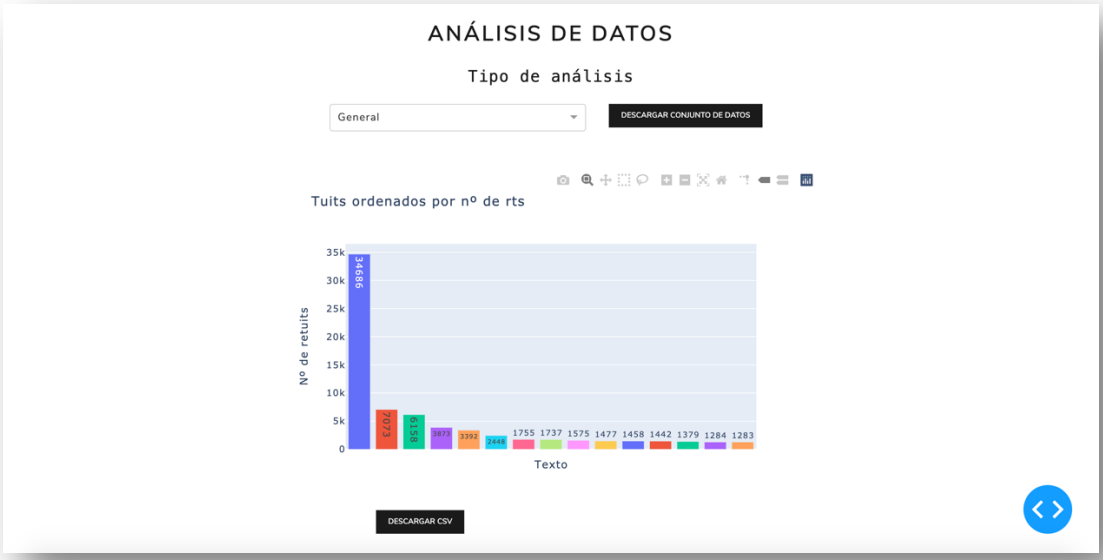
Una vez ambos datos están introducidos, se debe pulsar el botón “BUSCAR” para empezar la recogida y análisis de los datos. Mientras se realiza esta operación aparece un grafismo y un texto indicando que la operación se está realizando.



Página de análisis:

Una vez se ha completado el procesamiento de los tuits, la aplicación redirige a la página de análisis de los datos, en la que se puede elegir el tipo de análisis que se puede observar y descargar el conjunto de datos global generado (en formato .csv) mediante el botón "DESCARGAR CONJUNTO DE DATOS".

Cada vez que se elige el tipo de análisis se cambian los gráficos a mostrar, y para cada gráfico se puede descargar su conjunto de datos mediante el botón "DESCARGAR CSV".


ANÁLISIS DE DATOS
Tipo de análisis
General
DESCARGAR CONJUNTO DE DATOS
Tuits ordenados por nº de rts
Nº de retuits
35k
30k
25k
20k
15k
10k
5k
0
34686
7023
6159
3873
3392
2448
1755
1737
1575
1477
1458
1442
1379
1284
1283
Texto
DESCARGAR CSV

Apéndice C

Plan de Pruebas

PLAN DE PRUEBAS

Eugenio López Porras

VERSIÓN 3.0

HOJA RESUMEN DE MODIFICACIONES

VERSIÓN	FECHA	TIPO	PREPARADO POR	APROBADO POR
1.0	25/4/2021	Versión inicial	Eugenio López Porras	Eugenio López Porras
2.0	20/5/2021	Versión intermedia	Eugenio López Porras	Eugenio López Porras
3.0	14/6/2021	Versión final	Eugenio López Porras	Eugenio López Porras

ÍNDICE

1. INTRODUCCIÓN

Proyecto(s)		Tipo de Proyecto	
TFG – Eugenio López Porras		Proyecto de Desarrollo de Software Académico.	
Documentos Evaluación relacionados			
Eugenio López Porras Memoria.pdf			
Equipo de Proyecto			
Responsable del proyecto	Eugenio López Porras	Tutor	Eduardo Guzmán de los Riscos

1.1. OBJETIVOS DEL PLAN DE PRUEBAS

Este documento, tiene como finalidad entregar las pautas y definir la estrategia que se seguirá para llevar a cabo la certificación del software desarrollado.

El objetivo general del plan es establecer la cronología y condiciones para la aplicación de las pruebas de manera de obtener, un sistema que pueda ser completado con una recepción total de los interesados y entrar en operación con la totalidad de las funcionalidades requeridas para su funcionamiento.

1.2. DOCUMENTOS RELACIONADOS

Nombre	Descripción
Eugenio López Porras Memoria.pdf	Memoria del proyecto en el que se describen los requisitos del proyecto

2. ALCANCE DE LAS PRUEBAS

Mediante los siguientes cuadros se describen los requerimientos de pruebas de la página web a partir de los requisitos y alcances establecidos en la memoria del TFG.

2.1. CUADRO RESUMEN DE LAS PRUEBAS

Módulos del producto a ser probados:	Módulos: - Navegación entre páginas - Ciclo de funcionamiento de la aplicación. - Representación de resultados.
Objetivos de las Pruebas	En estos Módulos se realizarán pruebas para validar: Una correcta navegación entre las distintas páginas del sitio web. El ciclo de funcionamiento de la aplicación es el esperado. La representación de los resultados es la esperada.
Detalle del orden de ejecución de los módulos	Los módulos se deben ejecutar de forma independiente utilizando la herramienta Selenium.
Responsabilidad de la Prueba	Las pruebas son responsabilidad del Testing Operacional del equipo de proyecto, que al estar formado por un único usuario (el autor) cae íntegramente en su figura.

2.1. BASES DE DATOS DE PRUEBAS

Base de Datos: MongoDB

Servidor BD: demo.tweet_collecion

2.2. CRITERIOS DE APROBACIÓN / RECHAZO

Errores Graves: información crítica presentada erróneamente, información mal registrada en la base de datos, caídas de programas, incumplimiento de objetivos en funciones principales, etc.

Errores Medios: errores en presentación de datos, incumplimiento de objetivos en funciones secundarias, caídas de programas auxiliares, etc.

Errores Leves: errores en presentación de datos secundarios, no adecuación a estándares, comportamientos correctos pero diferentes en situaciones similares, dificultades de operación, etc.

Se aprobará el proyecto con un 100% de las pruebas ejecutadas, pero con un 90% de aceptación. Esto quiere decir el 80% de las pruebas deben ser exitosas y sin errores. El restante 20% pueden existir errores medios o bajos, pero **no** graves.

En caso de ocurrir que el proyecto no cumpla con el nivel exigido, el proyecto se rechaza completo en su etapa de certificación.

2.3. ORDEN DE EJECUCIÓN DE PRUEBAS

Las pruebas se llevarán a cabo de la siguiente forma:

Secuencias de pasos para la Configuración

1. Actualización de tecnologías y librerías auxiliares.
2. Arranque del servidor de la aplicación y de la Base de Datos.
3. Acceso a la aplicación mediante un navegador (Google Chrome).

2.4. CASOS DE PRUEBAS INCLUIDOS

Título del caso	Descripción de la prueba a realizar	Resultado esperado	Requisitos involucrados	Resultado de la prueba
Navegabilidad entre páginas	Se accede a las páginas disponibles de la aplicación web a través de la barra de navegación.	La página resultante es la esperada.	RNF-1, RNFF-2	Satisfactorio.
Navegabilidad entre páginas no válidas.	Se intenta acceder mediante URL a una página de la aplicación que no existe.	Se muestra un mensaje de error indicando que esa página no existe.	RNF-1	Satisfactorio.
Búsqueda de los tuits	Se introduce una o varias palabras separadas por comas en el campo habilitado para ello, el número de tuits a buscar y se pulsa el botón de buscar.	Aparece un grafismo y un texto indicando que la búsqueda se está realizando. Cuando se completa, pasa a la página principal.	RF-1.1, RF-1.2, RF-2.1, RF-2.2, RF-2.3, RF-2.4, RF-2.5, RNF-1, RNF-2, RNF-3	Fallo medio, en una de las ejecuciones se produjo un error de la API de twitter.
Búsqueda de los tuits	Se intenta pulsar el botón de buscar	Se muestra un mensaje de error	RF-1.1, RF-1.2, RF-2.1,	Satisfactorio.

incorrecta (campos sin rellenar)	sin haber rellenado los campos necesarios.	indicando que es necesario rellenar los campos obligatorios.	RNF-1, RNF-2	
Búsqueda de los tuits incorrecta (formato incorrecto	Se intenta pulsar el botón de buscar habiendo introducido un formato de texto no válido en el campo de texto.	Se muestra un mensaje de error indicando que el formato que se debe utilizar.	RF-1.1, RF-1.2, RF-2.1, RNF-1, RNF-2	Satisfactorio.
Búsqueda de los tuits incorrecta (número menor que 0)	Se intenta pulsar el botón de buscar introduciendo un número menor 0	El campo numérico impide introducir un número menor que 0.	RF-1.1, RF-1.2, RF-2.1, RNF-1, RNF-2	Satisfactorio.
Página principal	Al ingresar en la página principal, se selecciona “General” o “Sentimientos” en el desplegable.	Se ve por defecto la opción “General”, y al cambiar de opción, la página se actualiza con la opción correspondiente.	RF-4.1, RNF-1, RNF-2, RNF-3	Satisfactorio.
Gráficos general	Se selecciona la opción “General” en el desplegable de la página principal.	Aparecen los siguientes gráficos: - Gráfico de barras de los tuits de mayor a menor número de retuits. - Gráfico de barras de los usuarios de los tuits de mayor a menor número de seguidores.	RF-4.2, RF-4.3, RF-4.4, RNF-1, RNF-2, RNF-3	Fallo leve, algunos gráficos se pueden ver de una forma no atractiva (histograma si los tuits son todos de

		- Histograma del número de tuits por hora.		la misma hora).
Gráficos sentimientos	Se selecciona la opción "Sentimientos" en el desplegable de la página principal.	Aparecen los siguientes gráficos: - Gráfico de sectores con el número de tuits por cada sentimiento. - Gráfico de barras de las palabras con más apariciones en tuits positivos. - Gráfico de barras de las palabras con más apariciones en tuits negativos. - Gráfico de barras de las palabras con más apariciones en tuits neutrales.	RF-4.5, RF-4.6, RF-4.7, RF-4.8, RNF-1, RNF-2, RNF-3	Satisfactorio.
Botones de descarga	Se selecciona análisis general o de sentimientos.	Siempre está disponible el botón para descargar el conjunto de datos total en la parte superior. Debajo de cada gráfico está el botón de descarga del conjunto de datos	RF-5.1, RF-5.2, RNF-2, RNF-3	Satisfactorio.

		que usa para su representación.		
Accesibilidad	Se navega por las distintas páginas webs.	La información incluida en el sitio web se encuentra descrita con etiquetas.	RNF-3	Satisfactorio.
Facilidad de uso	Se presenta el sistema a un nuevo usuario.	Un usuario novel entiende las acciones del sitio web en menos de 5 minutos de uso.	RNF-3	Satisfactorio.
Resultado del plan de pruebas	Satisfactorio (16% fallo, 1 medio y 1 leve)			

2.5. EQUIPO DE PRUEBAS Y RESPONSABILIDADES

Nombre	Responsabilidad
Eduardo Guzmán de los Riscos	Tutor del TFG, responsable de evaluar las condiciones de término para el proceso de pruebas junto al Responsable de Calidad.
Eugenio López Porras	Jefe de Proyecto, Tester de la Solución, Responsable de Calidad y de la generación del Plan de Pruebas.

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga

E.T.S. DE INGENIERÍA INFORMÁTICA