UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

SOCIAL-HEALTH: ANÁLISIS SOCIAL SOBRE EL SISTEMA SANITARIO

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Chichell Ruiz, David

Tutor: Botti Navarro, Vicente Juan

Directora experimental: Del Val Noguera, Elena

Curso 2017/2018

Resumen

La toma de decisiones en el ámbito sanitario para la mejora del mismo es continua en el día a día. Gracias al aumento del uso de las redes sociales, y con un buen tratamiento de la información generada por los usuarios de las mismas, es posible generar respuestas que faciliten la toma de decisiones. Para generar estas respuestas, como objetivo principal, se ha propuesto el desarrollo de una aplicación web que analice los tuits relacionados con la sanidad. El usuario puede visualizar gráficas sobre temáticas frecuentes, palabras recurrentes, análisis de sentimiento, regiones activas y diagramas temporales. Para llevar acabo el proyecto ha sido necesaria una metodología que incluya el análisis del problema, análisis de la fuente de datos y medidas de protección de datos; el diseño de la respuesta al problema, haciendo uso de diagramas y especificaciones, y el desarrollo de la solución, empleando como tecnologías principales Django y MongoDB. Tras las pruebas, la herramienta permite realizar un análisis del sistema sanitario que el usuario indique.

Palabras clave: Twitter, análisis social, sanidad, web, análisis de datos

Resum

La presa de decisions en l'àmbit sanitari per a la millora del mateix és contínua en el dia a dia. Gràcies a l'augment de l'ús de les xarxes socials, i amb un bon tractament de la informació generada pels usuaris d'aquestes, és possible generar respostes que faciliten la presa de decisions. Per a generar aquestes respostes, com a objectiu principal, s'ha proposat el desenvolupament d'una aplicació web que analitze els tuits relacionats amb la sanitat. L'usuari pot visualitzar gràfiques sobre temàtiques freqüents, paraules recurrents, anàlisis de sentiment, regions actives i diagrames temporals. Per a portar terme el projecte ha sigut necessària una metodologia que incloga l'anàlisi del problema, anàlisi de la font de dades i mesures de protecció de dades; el disseny de la resposta al problema, fent ús de diagrames i especificacions, i el desenvolupament de la solució, utilitzant com a tecnologies principals Django i MongoDB. Després de les proves, l'eina permet realitzar una anàlisi del sistema sanitari que l'usuari indique.

Paraules clau: Twitter, anàlisi social, sanitat, web, anàlisi de dades

Abstract

The decision making in the health field for the own improvement is continuous in the day to day. Thanks to the increased use of social networks, and with a good treatment of information generated by the users, it is possible to generate answers that facilitate decision making. To generate these answers, as a main objective, It has been proposed to develop a web application that analyzes tweets related to health. The user can view graphs about frequent topics, recurring words, sentiment analysis, active regions and time diagrams. To carry out the project it has been necessary a methodology that includes the analysis of the problem, analysis of the source of data and data protection measures; the design of the answer to the problem, making use of diagrams and specifications, and the development

of the solution, using as main technologies Django and MongoDB. After testing, the tool allows an analysis of the sanitary system that the user indicates.

Key words: Twitter, social analysis, health, web, data analysis

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

1.1 Motivación

El auge del uso de las redes sociales en los últimos años [1], ha hecho que la sociedad utilice estas herramientas para estar en contacto con la gente, comunicarse y compartir ideas u opiniones sobre temática relevante para el usuario. Son muchas las funciones a realizar gracias a las redes sociales, pero no cualquier usuario es capaz de extraer el máximo rendimiento a la información que estas nos aportan.

La visión que pueda tener un usuario de estas plataformas de interacción, es limitada, sin embargo, la visión tecnológica que puede darse en los proyectos cuya premisa es el tratamiento de datos, es una visión que se está expandiendo en el denominado campo de análisis de los medios sociales. Aquí radica la idea del proyecto, presentar al usuario un conocimiento más exhaustivo del área que más interese al usuario haciendo uso de datos extraídos de las redes sociales.

Son muchas las áreas de conocimiento que abarcan este tipo de proyectos, ya que se requiere de estudio y técnica, no solo a nivel de información del dato sino a nivel contextual en el campo estudiado. El campo en el que se centra este proyecto es la sanidad, así que, se trabaja con datos pertenecientes a la sanidad pero que son de naturaleza social. Extrayendo información sobre el sistema sanitario, tratándola y presentándola al usuario, esta adquiere un valor añadido, ya que se ha creado información útil a partir de un amalgama de datos. Además, la radiografía del sistema sanitario no solo es útil para ciudadanos, también es una herramienta que puede servir como retroalimentación tanto a las distintas administraciones públicas como a los profesionales del sector para poder mejorar y tener una visión de las opiniones del sector. Esta retroalimentación actúa de forma positiva agilizando y facilitando la toma decisiones en el sector.

1.2 Objetivos

El principal objetivo de este proyecto es el desarrollo de una plataforma que permita obtener un análisis y visualización del estado actual del sistema sanitario desde el punto de vista social. Esta radiografía obtenida vendrá dada por el análisis de tuits generados por los usuarios de la red social Twitter.

Para la correcta consecución de este objetivo se plantean los siguientes subobjetivos:

- Extracción, limpieza y almacenamiento de información de la red social Twitter
- Tratamiento y análisis de la información relevante en el contexto sanitario.
- Propuesta de métricas e indicadores para establecer el estado del sistema sanitario.
- Presentación de los datos de una manera intuitiva y visual
- Desarrollo de una plataforma web que integre el proceso de análisis y su visualización.

1.3 Metodología

La metodología que se va a emplear consiste en un híbrido entre las etapas del ciclo de vida de un sistema de información y una selección de las etapas necesarias de un proyecto de explotación de datos [2] [3]. Dada la naturaleza del proyecto, se ha optado por esta combinación de metodologías porque nos permite tener un mayor control del seguimiento del proyecto.

Etapas del ciclo de vida de un sistema de información

1. Análisis: Fase de identificación de necesidades y requisitos del sistema.
2. Diseño: Representación de las características y organización de los componentes del sistema en base a los resultados obtenidos en la fase de análisis.
3. Implementación: Desarrollo de las distintas unidades que componen el sistema.
4. Pruebas: Realización de pruebas sobre el sistema para comprobar que cumple con las funcionalidades descritas

Etapas necesarias propias de un proyecto de explotación de datos

1. Definición: Análoga a la fase de análisis.
2. Búsqueda: Fase de búsqueda de la fuente de extracción de datos. Descripción de la fuente de los datos.
3. Obtención: Fase de recolección de los datos.
4. Verificación: Fase que consiste en la verificación de que los datos obtenidos de la fuente son los necesarios para lograr los objetivos definidos.

5. Limpieza: En esta fase se debe conseguir un conjunto de datos limpio que permita al sistema operar sin emplear tiempo en comprobar si el dato necesario está corrupto.

6. Análisis y presentación de los datos: Una vez desarrollada la herramienta es necesario hacer un análisis de los resultados obtenidos, así como una visualización de los datos.

Las etapas combinadas de ambos tipos de proyectos darían como resultado las siguientes fases:

Etapas de la metodología híbrida empleada

1. Análisis + Definición
    a) Búsqueda
2. Diseño
3. Implementación
    a) Obtención
    b) Verificación
    c) Limpieza
4. Pruebas
5. Análisis y presentación de los datos

1.4 Estructura de la memoria

Considerando la motivación y los objetivos de este trabajo, así como la metodología, que marcará la línea a seguir en la estructura de la memoria, el resto del documento se organiza como se muestra a continuación. El capítulo dos presenta una revisión del estado del arte de propuestas similares relacionadas con el lema del proyecto y se introduce la propuesta que se presenta en el proyecto. El capítulo tres describe en profundidad el problema que se pretende abordar presentando la propuesta, en el capítulo se procede al análisis de las diferentes áreas que afectan al proyecto y se analizan las soluciones posibles para resolver el problema. En el capítulo cuatro se presentan las características y requisitos a cumplir por la solución utilizando el estándar IEEE 830. El capítulo cinco presenta el diseño de la solución seleccionada, diseño estructural y tecnológico que se fundamenta en diagramas y análisis de las tecnologías empleadas. El capítulo seis se adentra en el desarrollo de la solución, el mismo capítulo fragmenta el desarrollo en los distintos módulos que componen la aplicación. En el capítulo siete se describen las tareas de que se deben llevar a cabo para la correcta implantación del sistema. En el capítulo ocho se detallan las pruebas realizadas para la validación del sistema. En el capítulo nueve se presentan las conclusiones obtenidas y la relación del trabajo con los estudios cursados. Finalmente, en el capítulo diez se indican las futuras líneas de trabajo.

CAPÍTULO 2
Estado del arte

2.1 Twitter como herramienta de análisis y predicción

El aumento del uso de las redes sociales comentado en la introducción, ha modificado el modelo clásico de emisión en directo que siempre se ha tenido en otros medios de comunicación como la televisión y la radio. Es por esto que muchas empresas, entidades y administraciones, se están lanzando al contacto directo con el público utilizando para ello las redes sociales, logrando establecer un vínculo que permita obtener una mejor percepción del público al que llega esa organización. Entre las distintas redes sociales existentes, este proyecto se va a basar en tuits recopilados de la red social Twitter.

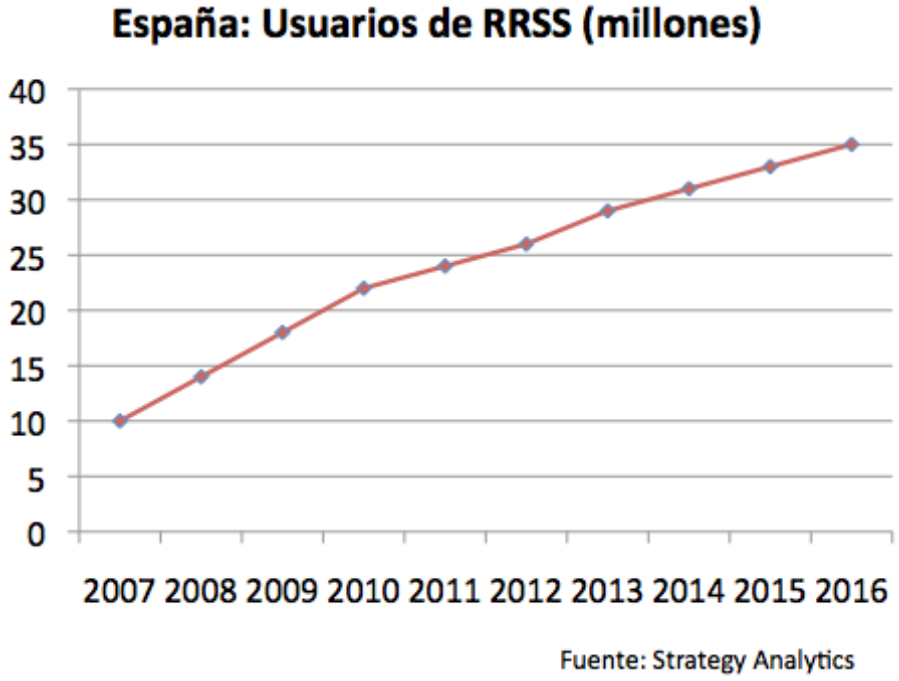

Figura 2.1: Evolución del número de usuarios de redes sociales en España

En la figura 2.1 se aprecia la perspectiva de aumento de los usuarios de las redes sociales entre 2007 y 2016 [4].

Twitter es una red social que proporciona una interacción constante entre usuarios de la plataforma mediante mensajes cortos llamados tuits. Estos mensajes en el momento de de creación de la plataforma, tenían una longitud de 140 caracteres, pero con las últimas actualizaciones a finales del año 2017, esta longitud aumentó hasta los 280 caracteres.

Los usuarios pueden utilizar la red social para expresar opiniones, apoyar lemas de personas influyentes, mostrar rechazo o aprobación a acontecimientos que se den en la sociedad, seguir a medios e instituciones relevantes, etc. Esta interacción entre usuarios es muy dinámica, ya que el límite de caracteres establecido, obliga a que el flujo de tuits sea constante para obtener una interacción fluida. Se ha calculado que estos mensajes tienen una vida media[1] de entre 15 minutos y 1 hora [5].

La generación de este gran volumen de información abre la puerta a nuevas líneas de investigación, que gracias al uso de herramientas de explotación de datos, son capaces de establecer nuevos métodos de análisis y predicción. Se debe tener en cuenta el aumento del número de estudios de la población ante eventos destacados, como puedan ser elecciones presidenciales, atentados, reformas legales y cualquier otro tema de relevancia que tenga un impacto directo en la sociedad [6]. No es de extrañar que Twitter sea un valioso campo de estudio de la sociedad.

Entre los distintos tipos de análisis que se pueden obtener gracias al uso de Twitter y herramientas de explotación de datos, así como otros tipos de herramientas y técnicas ligadas al lenguaje, destacan los siguientes tipos:

- Análisis de actividad: en este tipo de análisis se puede extraer la actividad de tuits relacionados a distintos acontecimientos, pudiendo medir el impacto del evento en la sociedad [7].

- Análisis de sentimiento: utilización de técnicas de procesamiento del lenguaje para realizar una búsqueda y clasificación, estableciendo así, un grado de emotividad a cada uno de los tuits analizados [8].

- Caracterización del público objetivo: realizar análisis de usuarios en base a datos geográficos basados en la localización, datos demográficos referentes al sexo, edad y ocupación, así como datos conductuales relativos a la personalidad, gustos e intereses del usuario [9].

Todo el conjunto de análisis que se realice en base a datos extraídos de Twitter, aporta una información que con una correcta visión, puede ser utilizada para la predicción de indicadores en distintas áreas como pueden ser la política, educación o la sanidad.

[1] Se entiende como tiempo de vida de un tuit el tiempo transcurrido desde que es publicado hasta que ya no hay interacción del mismo con otros usuarios.

2.2 Análisis social en diferentes ámbitos

El análisis social se puede aplicar a distintos campos, en esta sección se va a hacer énfasis en el área de aplicación a la empresa y en ámbito político.

Comenzando con el campo relativo al mundo empresarial, con el avance de la tecnología y la aparición de nuevos métodos de análisis, las empresas se han visto obligadas a evolucionar. Se ha llevado a cabo una modificación de sus modelos de negocio, que han tenido que dejar atrás rudimentarias técnicas de análisis y seguimiento para dar paso a nuevas herramientas que buscan aumentar el rendimiento en el proceso de recolección y análisis de información con el objetivo de facilitar la toma de decisiones, ya que el día a día de una empresa se basa en la toma de decisiones de todo el conjunto empresarial

Con esta evolución, aparecen términos como *Big Data*, que haría referencia al análisis y gestión de grandes cantidades de información de datos, en relación a este término, tendríamos *Business Intelligence*, que serían las herramientas que manejan la información interna de la empresa, donde se incluirían registros de ventas y evolución de producción, entre otros indicadores. También, las citadas herramientas de *Business Intelligence* proporcionan una visualización para que esta información sea fácilmente interpretable y ayude a la toma de decisiones.

Una vez presentados estos dos términos, entra en juego el denominado análisis social, y con su entrada surge el término *Business Analytics*. Lo que se pretende es realizar un cruce de datos entre datos internos a la asociación y datos externos, como pueden ser datos de redes sociales, datos demográficos, datos de administraciones públicas, etc., con el objetivo de dar una vuelta de tuerca al análisis, ya que la inserción de nuevos conjuntos de datos aporta una visión más global y realista del entorno, la situación y la sociedad, que al fin y al cabo, son factores que afectarán al negocio. Estos tipos de análisis pueden aportar perspectivas y predicciones para que la empresa pueda adaptarse y anticiparse a los acontecimientos en su entorno.

Si ponemos atención al uso del análisis social en el campo político, el empleo de las nuevas tecnologías puede convertirse en una verdadera herramienta de ayuda. La recopilación de información de las redes sociales, tiene sentido cuando es cruzada con otros conjuntos de datos, ya que se obtendrá como solución una información que hasta el momento era inexistente.

Partidos políticos pueden realizar sus propios sondeos de cara a elecciones, hacer una seguimiento de sus militantes y obtener un análisis de la opinión pública al momento [6].

Administraciones Públicas pueden realizar estudios rutinarios y medir el impacto de la aplicación de distintas políticas entre los ciudadanos. Se consigue obtener una retroalimentación para poder orientar las distintas políticas de una Administración en el rumbo apropiado [9].

Todos estos estudios, análisis y espectros son simplemente una muestra de todo lo que se pueden llevar a cabo gracias a la constante evolución tecnológica de los últimos tiempos, la aparición de las redes sociales y las nuevas líneas de investigación que han surgido en áreas como la inteligencia artificial, redes e integración de aplicaciones.

2.3 Análisis social en el ámbito sanitario

El ámbito sanitario es un campo fundamental en la vida de la sociedad. Cada vez más, año tras año se ha intentado extraer la opinión del paciente tras una estancia hospitalaria o alguna gestión en el ámbito sanitario, esto es debido a que es fundamental el conocimiento de la opinión para mejorar el servicio.

A día de hoy y con referencia al sistema sanitario en España, es el Centro de Investigaciones Sociológicas (CIS) el que realiza un barómetro sanitario anual cuyos objetivos vienen recogidos en el propio informe ordenado por el Sistema Nacional de Salud (SNS). Se presenta a continuación la lista de objetivos recogidos por el informe del barómetro sanitario del 2017 elaborado por el SNS.

OBJETIVOS

- Conocer cómo perciben y valoran los ciudadanos el funcionamiento de los Servicios Sanitarios Públicos.
- Saber la opinión de los ciudadanos acerca de determinadas medidas de política sanitaria.
- Recoger la valoración y experiencias de los usuarios de los servicios sanitarios.
- Conocer la penetración real de las estrategias informativas de las autoridades de salud.
- Obtener información sobre el grado de conocimiento o sobre las actitudes de los ciudadanos hacia los problemas de salud o actuaciones específicas del S.N.S.

[10]

En el informe del SNS se recogen una serie de gráficos acompañando a la descripción del estudio. Estos gráficos evalúan los distintos indicadores que se han definido para realizar el estudio. A continuación se muestran tres gráficos a modo de ejemplo (ver figuras 2.2, 2.3 y 2.4).

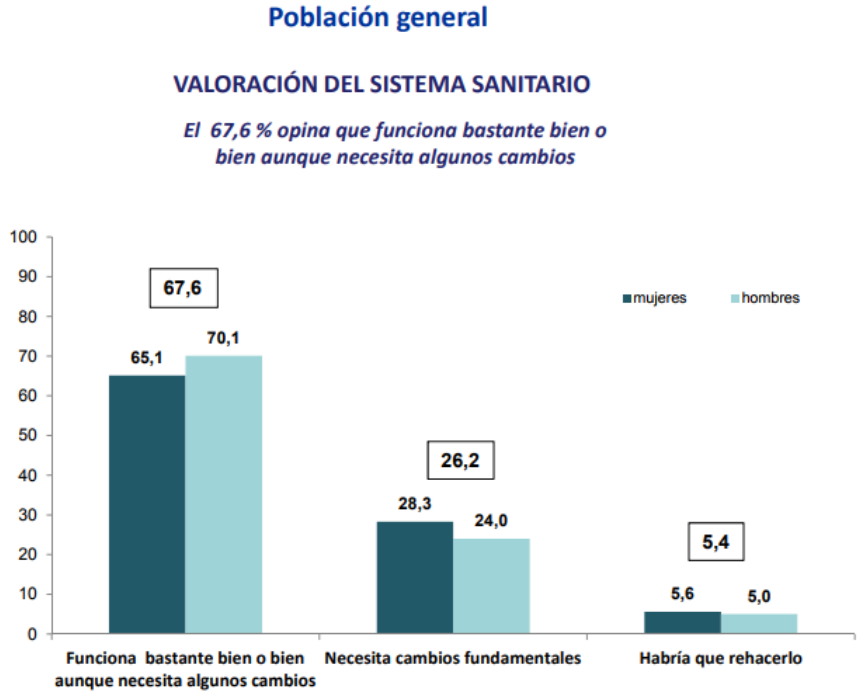

Figura 2.2: Valoración del sistema sanitario

Figura 2.3: Satisfacción con el funcionamiento del sistema sanitario público

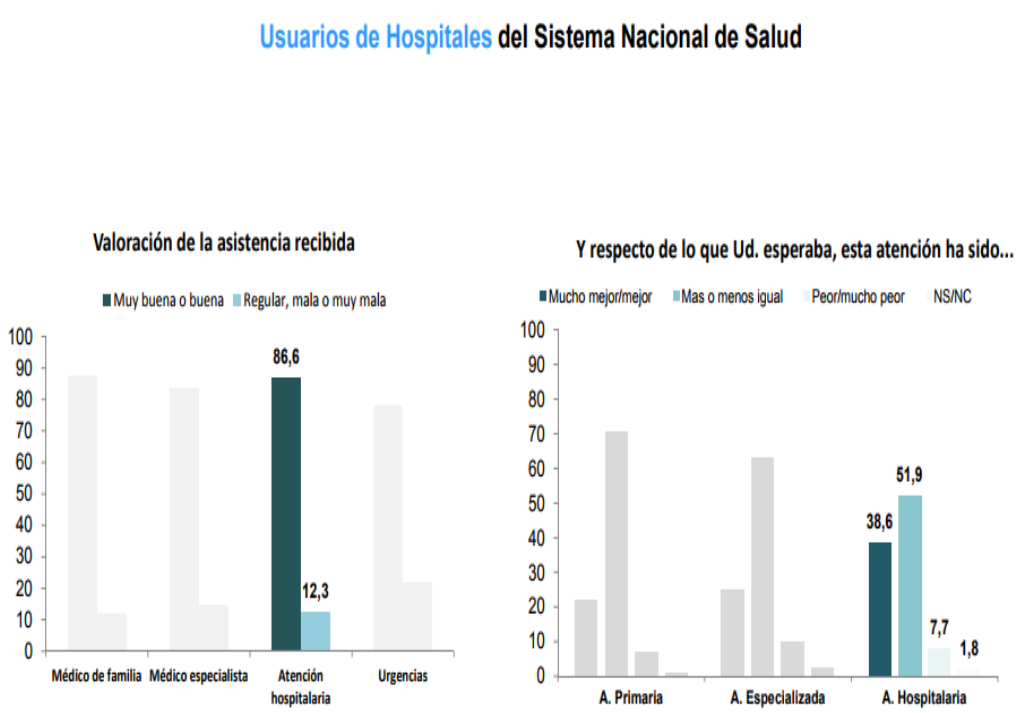

Figura 2.4: Usuarios de hospitales del Sistema Nacional de Salud

Estas son algunas de las figuras que acompañan a los análisis del informe [10]. Según la ficha técnica del informe se han realizado 7800 entrevistas. Teniendo en cuenta el número de la muestra, encontramos una de las principales limitaciones. El número de habitantes en España según el INE a día 1 de julio de 2017 era de 46.549.045 habitantes, en este caso el tamaño de la muestra, que recogía la opinión de 7800 personas, representa un 0.017 % de la población total. Si se utilizara el análisis social como herramienta para recoger las opiniones de los ciudadanos, se podría complementar con los resultados del informe y en algunos casos recoger información de un mayor número de usuarios.

El análisis social en este ámbito es una vía que está aún en sus periodos iniciales. Los avances en procesamiento de lenguaje y análisis de sentimiento abre la puerta a la aparición de estudios que intenten recoger esa opinión del paciente [11]. Los objetivos de estos nuevos estudios serían la recoleccion de las distintas opiniones que puedan dejar los pacientes en redes sociales, entradas de blogs o comentarios que puedan dejar estos usuarios en webs de centros médicos u hospitalarios. Una vez recolectada la información sería necesario aplicar un tratamiento adecuado para medir los indicadores adecuados en los que se está interesado. Gracias a estos estudios se conseguirá visualizar un panorama del sistema sanitario analizado, que permitiría ejecutar medidas de respuesta para mejorar la opinión pública del servicio. A su vez, estas medidas también podrían ser analizadas para calcular el impacto sobre los pacientes y facilitarían el desarrollo de nuevas políticas sanitarias que permitiesen ir refinando el sistema sanitario para que fuera mejorando cada día.

2.4 Herramientas similares

2.4.1. ObservaTICs

ObservaTICs [2] es un proyecto impulsado por la Universidad Politécnica de València (UPV) a través del Instituto de Aplicaciones de las Tecnologías de la Información y de las Comunicaciones Avanzadas (ITACA) del grupo de Innovaciones Tecnológicas para la Salud y el Bienestar (SABIEN). Este proyecto nace de la necesidad de adaptación del entorno sanitario a las nuevas tecnologías, por esto ObservaTICs plantea el desarrollo una web que actúa a modo de observatorio. Esta web ofrece por un lado una visualización del uso de las redes sociales en España por parte de hospitales y clínicas, además, por otro lado ofrece información estadística sobre la interacción de las propias cuentas de los centros con los ciudadanos, así como información básica sobre el hospital que se visualizará en las próximas figuras.

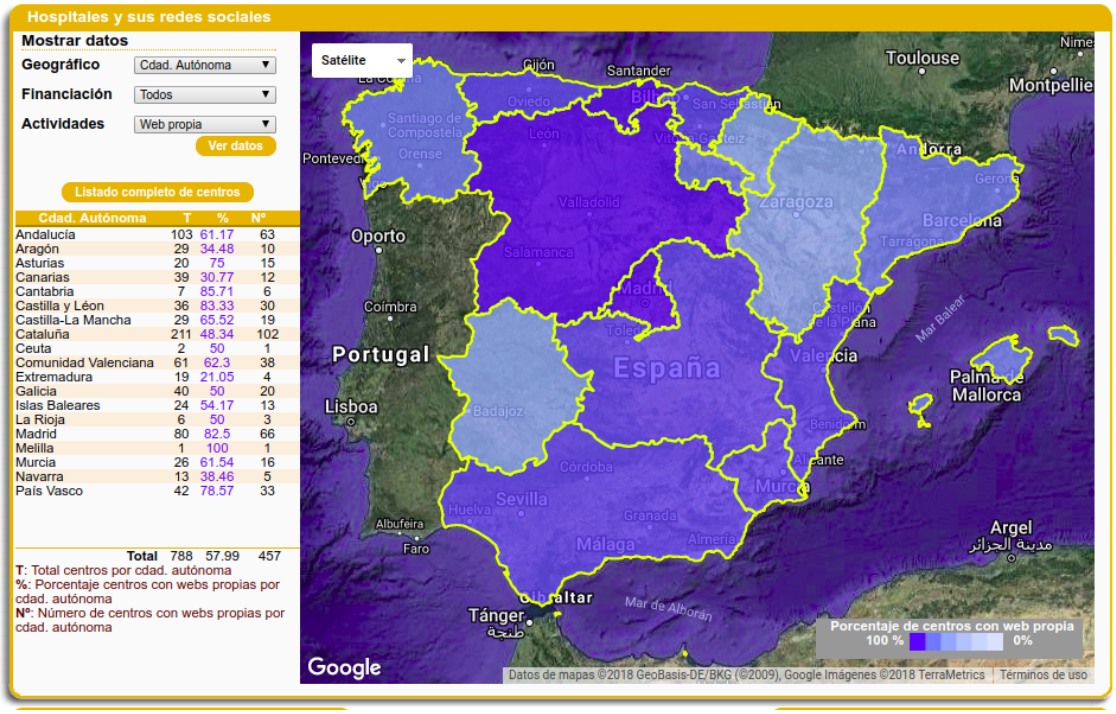

Figura 2.5: Vista general del observatorio de ObservaTICs

En la figura 2.5 se muestra el cuadro de mandos principal del observatorio. Puede verse un mapa de densidad que muestra el porcentaje de centros hospitalarios con web propia por comunidades autónomas. También se pueden modificar los criterios de visualización para mostrar densidad de número de cuentas en redes sociales, hacer la distinción entre provincias en vez de comunidades autónomas y distinguir entre centros con financiación pública o privada. Junto al mapa aparece una tabla con el número de centros registrados por cada comunidad autónoma y el porcentaje que representa en el mapa.

[2] http://www.observatics.com

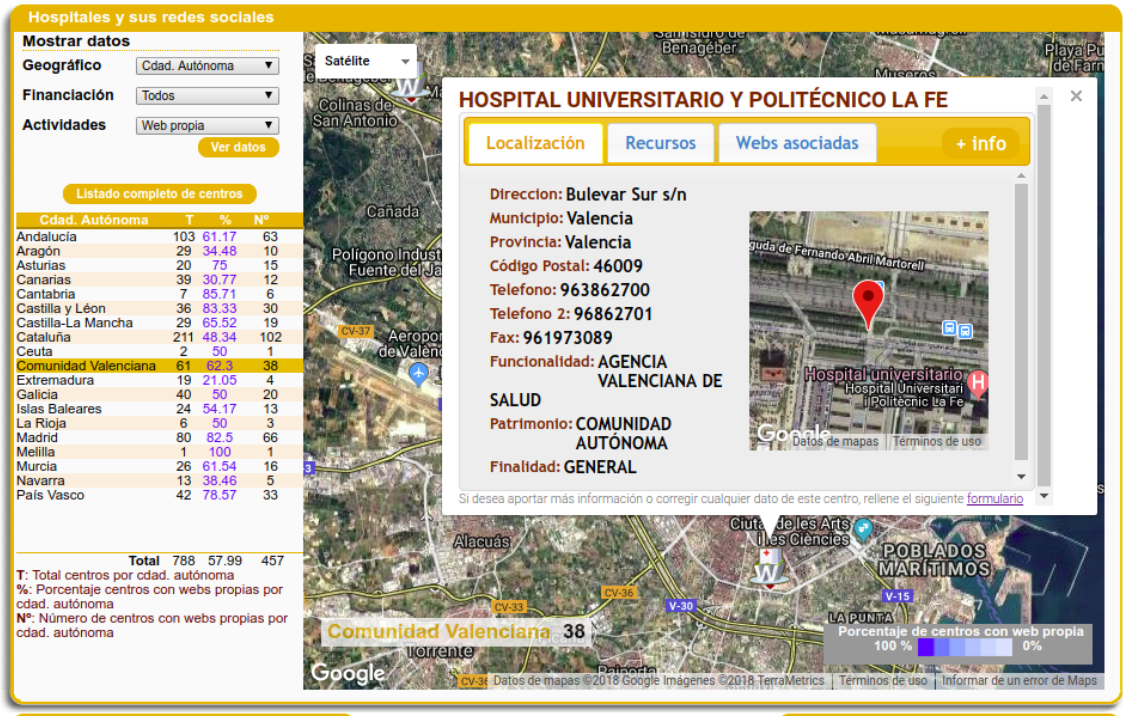

Figura 2.6: Vista de la información sobre la localización de un hospital

El mapa es interactivo y permite la visualización de datos sobre cualquier centro hospitalario que esté registrado. En la figura 2.6 se puede ver información básica del hospital como puede ser la ubicación, contacto y el propósito.

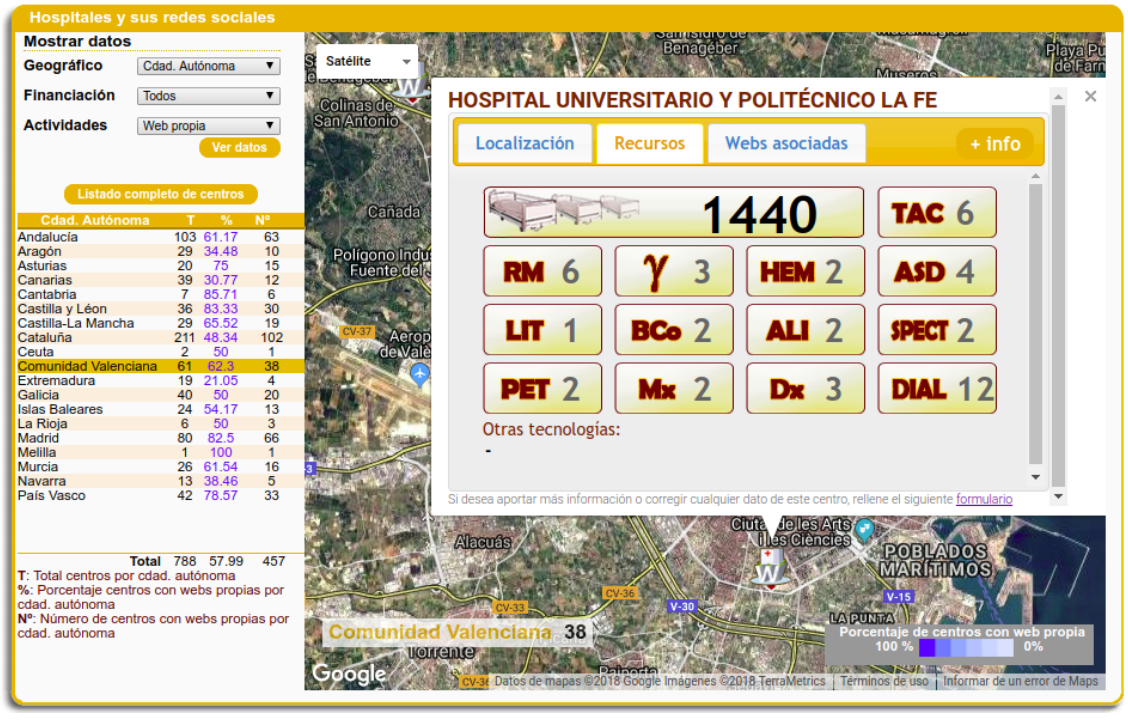

Figura 2.7: Vista de los recursos disponibles en el hospital

La web ofrece información sobre el número de camas totales, así como las tecnologías y el número de equipos disponibles en el hospital para el tratamiento de enfermedades, investigación y pruebas a pacientes (figura 2.7).

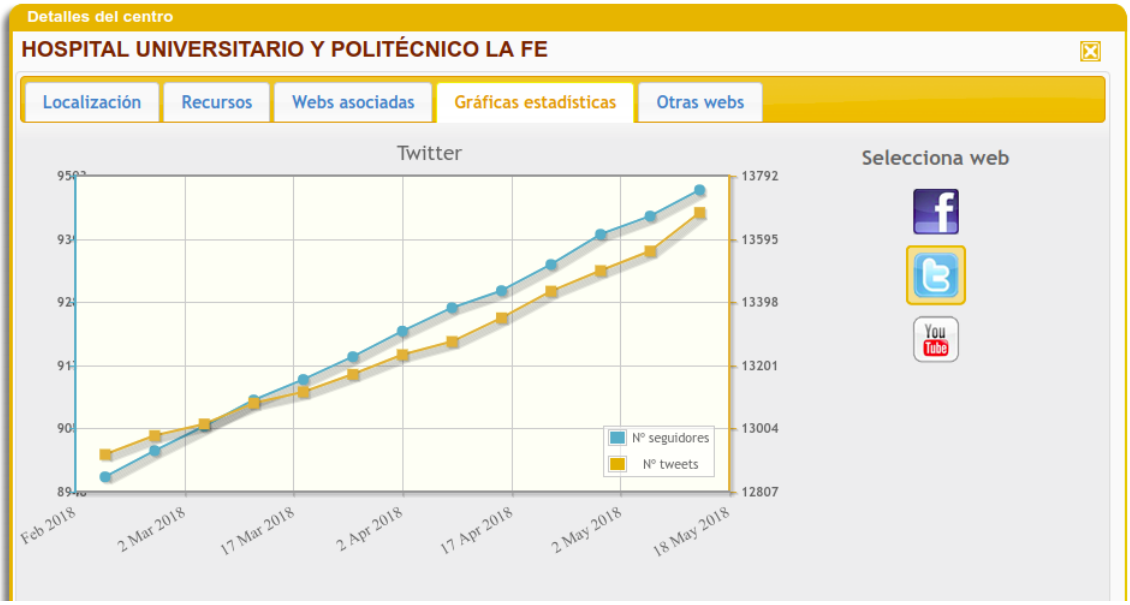

Figura 2.8: Vista de estadísticas de redes sociales

En la figura 2.8 se observa que hay posibilidad de generar gráficos de uso para Facebook, Twitter y YouTube. Estos gráficos permiten observar la evolución del uso de las redes sociales por parte del centro. En la figura se ha elegido generar la visualización referente a Twitter y se muestra el número de seguidores y el número de tuits generados por la cuenta verificada del centro hospitalario.

2.4.2. U-tool

U-Tool es una herramienta desarrollada por investigadores del Grupo de Tecnología Informática-Inteligencia Artificial (GTI-IA) de la UPV [12] [13]. Esta herramienta hace uso de los tuits geolocalizados recogidos por ciudadanos para medir el dinamismo de la ciudad. El objetivo es el de facilitar la toma de decisiones de los organismos públicos de la ciudad. Para ello, U-tool ofrece unas visualizaciones que permitan guiar al usuario a lo largo del análisis.

Las áreas de aplicación son diversas, la herramienta permite entre otras funciones, detectar grandes acumulaciones de gente, detectar patrones de movimiento de circulación de los ciudadanos, establecer puntos de interés en función de la densidad de personas y analizar espacios de la ciudad que podrían dedicarse para construcción de edificios, parques o espacios publicitarios.

En la línea de U-tool, la herramienta a desarrollar en el trabajo de fin de grado (TFG) persigue objetivos similares pero restringiendo el área de aplicación a la sanidad.

2.4.3. Otras herramientas

Durante la inspección de TFG presentados en años anteriores, se han encontrado dos trabajos relacionados con el presente proyecto debido a que en estos trabajos se hace uso de las redes sociales como medio de análisis.

El primero de ellos consiste en el desarrollo de una aplicación móvil para medir el estado de ánimo de alumnos durante las sesiones de clase [14]. Este proyecto utiliza tuits generados por la aplicación del alumno para que el profesor obtenga una retroalimentación sobre el estado anímico de los asistentes a la clase, también genera visualizaciones de estadísticas. Este proyecto se centra en el desarrollo de la aplicación y la publicación de tuits (lado utilizado por el alumno). Con el proyecto que se está presentando se pueden aportar técnicas de recolección y manejo de datos que podrían complementar al TFG mencionado agilizando las tareas del profesor que es el que luego debe descargar los datos de Twitter.

El otro TFG analizado consiste en el desarrollo de una plataforma web que permita al profesor evaluar la actividad de sus alumnos, así como el estado en el que se encuentran durante las clases [15]. El objetivo es que el profesor pueda obtener un análisis del efecto de su metodología de enseñanza y si cabe alguna modificación en la misma para la mejora del alumnado. Todo ello empleando un grupo de Facebook del que se analizarán los comentarios extrayendo los datos de la interfaz proporcionada por Facebook.

2.5 Crítica al estado del arte y propuesta

Tras el análisis del estado del arte, se ha visto que el proyecto ObservaTICs no se ciñe al análisis social en su totalidad, pero desarrolla unas herramientas que pueden servir como comienzo para llevar a cabo ese análisis. Las herramientas del proyecto ObservaTICs se podrían complementar con las herramientas que se desarrollan en este proyecto, pudiendo enriquecer la visualización del análisis de redes sociales generando gráficas de opinión sobre el centro que se está consultando, opinión generada a partir del análisis de tuits de pacientes del mismo centro y control de cuentas de Twitter del ámbito sanitario. Este proyecto también aporta un nuevo aire al análisis social ya que hasta el momento no se ha desarrollado un gran número de proyectos que aplique el análisis al sistema sanitario. Del mismo modo se puede apuntar que los TFG analizados recogen aplicaciones orientadas a pequeños grupos.

La línea más marcada y que más se asemeja a los objetivos de la herramienta del TFG será U-Tool. U-Tool lleva una orientación al análisis genérico y análisis de datos geoposicionados, sin embargo en la herramienta a desarrollar en este TFG la orientación camina hacia temas específicos como es la sanidad y el uso de datos no geoposicionados.

En conclusión, el presente proyecto aúna en un sistema datos que pueden provenir de cualquier parte del mundo, el volumen de información es elevado y el desarrollo de una correcta metodología en el tratamiento de datos será la clave para obtener un sistema eficaz, con una buena base de información y que tenga como propósito la mejora de algo tan esencial como es la sanidad. Se propone una

herramienta de análisis para que gobiernos, ciudadanos y profesionales del sector sanitario colaboren en la mejora diaria de la sanidad gracias a la explotación de la información.

CAPÍTULO 3
Análisis del problema

Dado que el objetivo final del proyecto es realizar una plataforma web con acceso para usuarios, en la que puedan obtener una visualización de indicadores referentes a los tuits sobre sanidad. Es necesario realizar en primer lugar un análisis de la fuente de datos (Twitter).

Twitter permite a los usuarios redactar y publicar mensajes de una longitud máxima de 280 caracteres, estos mensajes se llaman tuits y si la cuenta de usuario es pública son visibles para cualquier usuario de Twitter. Los usuarios de Twitter pueden seguir a otros usuarios de la red social para consultar directamente los tuits del usuario al que siguen a modo de suscripción, los usuarios que siguen a otros usuarios se llaman seguidores. Además, la red social permite retuitear un tuit de otro usuario, un símil en la vida real a retuitear podría encontrarse en la situación en la que una persona dice algo y otra persona en otro momento dice las mismas palabras que esa persona.

Como se puede deducir, el objeto principal en torno al que gira la red social y las interacciones entre usuarios es el tuit. Según estadísticas de uso, se mandan 500 millones de tuits diarios, unos 6000 tuits cada segundo [16]. No es de extrañar que el proyecto que se está presentando requiera de una metodología propia de un proyecto de explotación de datos, ya que es necesario un gran volumen de información para obtener unos resultados fiables.

El problema reside en la cantidad de tuits generados. Ante tal volumen de información se hace necesario la creación de una herramienta que automatice la recogida de datos y que permita al usuario obtener un análisis social del sistema santario en tiempo real.

3.1 Análisis de la naturaleza de los datos

Los tuits son devueltos por la API de Twitter en formato *JavaScript Object Notation* (JSON) [17]. JSON es utilizado en este caso para el intercambio de información entre un cliente y un servidor ya que la información transmitida únicamente puede ser texto. No es el único formato de transmisión de datos existente, se pueden emplear otros formatos como *Comma-Separated Values* (CSV) [18] o *eXtensible Markup Language* (XML) [19].

CSV

Como su acrónimo inglés implica, se trata de una lista de elementos separados por comas (el delimitador entre elementos puede variar). Este formato representa la información en forma de tabla separando cada columna con un separador. La primera línea suele denominarse línea de cabecera y tiene como función la de dar nombre a la columna en la que se encuentran los elementos.

```
city,lat,lon
Valencia,39.466667,-0.375
Reggane,26.720278,0.172778
```

JSON

Promovido a principios de los años 2000 pero especificado oficialmente en 2013, se trata un formato utilizado para el intercambio de datos que consta de texto escrito estructuralmente como si fuera un objeto *JavaScript* (JS). Esta estructuración otorga una mayor agilidad en el procesamiento de la información en este formato ya que se facilita la conversión a objetos JS.

```
 {
  "cities": [
    {
      "name": "Valencia",
      "location": [39.466667, -0.375]
    },
    {
      "name": "Reggane",
      "location": [26.720278, 0.172778]
    }
  ]
}
```

XML

Estandarizado en 1998, es un metalenguaje útil para la definición de lenguajes de marcas que requieran un uso específico. Permite el almacenamiento estructurado de datos que pueden ser transmitidos y procesados, también al igual que JSON sigue un modelo jerárquico.

```
<?xml version="1.0"?>
<catalog>
  <city id="001" name="Valencia">
     <coordinates lat="39.466667" lon="-0.375"></coordinates>
  </city>
  <city id="002" name="Reggane">
     <coordinates lat="26.720278" lon="0.172778"></coordinates>
  </city>
</catalog>
```

Se ha visto la estructuración de la información para cada uno de estos tres formatos. A continuación se muestran unas tablas resumen para analizar los puntos fuertes (tabla 3.1) y débiles (tabla 3.2) de cada uno de estos formatos y extraer una conclusión sobre por qué JSON es el formato elegido por los desarrolladores de la API de Twitter para estructurar los tuits.

	Aspectos positivos
CSV	- Es el más compacto - Útil para grandes volúmenes de información con la misma estructura
JSON	- Estructuración jerárquica más ligera que XML - Fácil traducción a objeto JS, útil para aplicaciones web
XML	- Soporta estructuras de datos jerárquicas - Fácil lectura humana

Tabla 3.1: Relación de aspectos positivos con CSV, JSON y XML

	Aspectos negativos
CSV	- La compactación limita el análisis eficiente - Necesario un analizador para cada estructura - No es jerárquico
JSON	- Menor soporte que XML en servicios web debido a que es relativamente nuevo
XML	- Tres veces más pesado que CSV debido a las etiquetas

Tabla 3.2: Relación de aspectos negativos con CSV, JSON y XML

En base a la información que aparece en las tablas, se puede afirmar que JSON es un intermedio entre ligereza y eficiencia [20]. Es un formato adecuado para el objeto tuit porque se trata de un objeto que no tiene una misma estructura en todos los casos ya que puede haber campos vacíos, además es un objeto estructurado jerárquicamente y JSON facilita el anidamiento de campos relacionados, por estas dos causas se descarta el formato CSV. Finalmente para descartar el formato XML se puede decir que el objeto tuit va a formar parte de tráfico mayoritario en aplicaciones web, si se quiere obtener el máximo rendimiento es recomendable utilizar JSON ya que es más ligero que XML y los tiempos de procesamiento en formato JSON son menores.

3.1.1. Estructura de un tuit

El objeto tuit devuelto es un objeto en formato JSON, un tuit no está formado únicamente por un objeto, el anidamiento que permite JSON hace el que tuit esté compuesto por múltiples objetos que contienen atributos de esquema clave-valor. Estos atributos son accesibles siguiendo la jerarquía hasta el atributo de interés. En la figura 3.1 se muestra el esquema de un tuit. Tras el análisis estructural, se han resaltado los campos que resultan de interés para el proyecto.

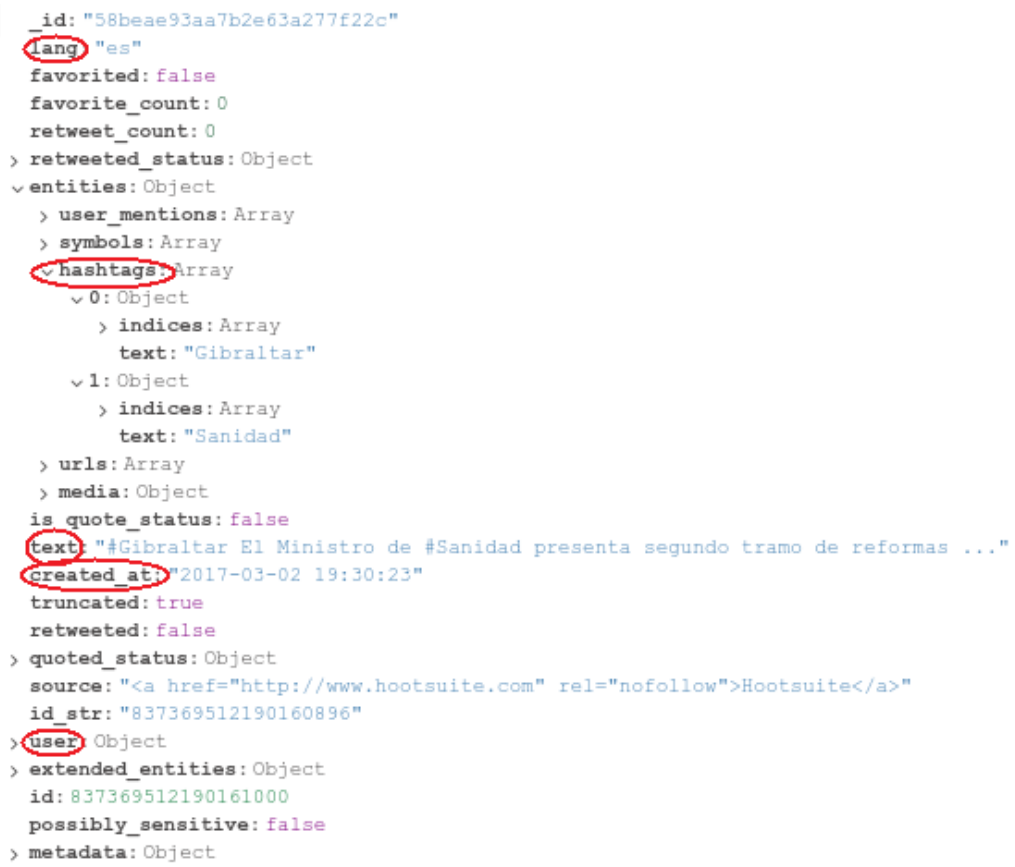

Figura 3.1: Esquema de objeto tuit con campos destacados

El atributo *lang* tiene como valor el idioma en el que está escrito el tuit, resulta interesante porque de cara a la aplicación a desarrollar permite identificar el idioma del tuit y poder tener una clasificación de los tuits dependiendo del idioma. El atributo *hashtags*, que se trata de un *array* que se encuentra dentro del objeto *entities*, contiene una colección de etiquetas que sirven para clasificar el texto del propio tuit. Este atributo es útil para realizar clasificaciones de los tuits en base a estas etiquetas. El atributo *text*, es el grueso de la parte visible de un tuit, su valor se corresponde con el texto escrito por el usuario. Este texto resulta útil para realizar el análisis de contenido de un tuit. El atributo *created_at* indica la fecha y hora en la que fue creado el tuit, este atributo permite tener un registro de las fechas de publicación para observar si se producen picos de actividad a lo largo del tiempo. Finalmente se ha resaltado dentro del tuit el objeto *user*, se trata de un objeto complejo que entre sus atributos aparece *screen_name*, este atributo indica el nombre del usuario y permitirá obtener un registro de los usuarios.

3.2 Análisis de la protección de datos

Como se ha podido observar, los datos con los que se trabaja en este proyecto son datos que han sido escritos por usuarios de una red social, estos usuarios mayoritariamente son personas y es conveniente comentar qué uso y tratamiento se les va a dar a estos datos para no vulnerar la privacidad de los usuarios.

Con el creciente uso de las nuevas tecnologías, la introducción de procesos como *Big Data* o la creciente transparencia que se le exige a las instituciones públicas, es inevitable el choque y la consecuente búsqueda de equilibrio entre el derecho a la información y el derecho a la protección de datos personales. Para conseguir este equilibrio y evitar la vulneración de derechos, los datos que se muestren o que se presenten en algún análisis, deben ser anonimizados.

Atendiendo al marco legislativo, del reglamento europeo [21] [22] se extrae que el término anonimización es referente al tratamiento irreversible de datos personales para evitar la identificación del origen. En la ley española no se recoge el término anonimización pero aparece el dato disociado como aquel que no permite la identificación. De igual modo en el artículo 3.F de la Ley Orgánica de Protección de Datos de Carácter Personal (LOPD)[23] se define el proceso de disociación como todo tratamiento de datos personales de modo que la información que se obtenga no pueda asociarse a persona identificada o identificable.

En el presente proyecto los datos mostrados en el análisis están anonimizados ya que únicamente se muestran palabras, temas, polaridad de palabras e idiomas que no van asociados a los nombres de usuario. También es importante destacar que el análisis no se centra en el individuo, el individuo forma parte de un gran conjunto utilizado para el análisis en busca de patrones y ocurrencias.

CAPÍTULO 4
Especificación de requisitos

4.1 Introducción

Una vez analizadas las herramientas y trabajos actuales relacionados con el proyecto, y analizada la fuente de extracción de datos y su naturaleza. En esta sección se deben especificar las requisitos *software*. Para la definición se hace uso del estándar IEEE 830.

4.1.1. Propósito

La especificación de requisitos servirá de guía para el desarrollo del sistema, aportándose las características y funcionalidades del sistema a desarrollar.

4.1.2. Ámbito del sistema

Este sistema, cuyo nombre es 'Health Analytics', permitirá obtener un análisis vía web sobre tuits relacionados con la sanidad.

El sistema hará:

- Búsqueda de tuits para poblar la base de datos (vía *script*).
- Registro de usuarios.
- Inicio y cierre de sesión controlados.
- Establecer parámetros de análisis que podrán ser modificados *a posteriori*.
- Descargar las visualizaciones del análisis.

El sistema no hará:

- Recolección de tuits en tiempo real.
- Geolocalización de tuits.

Una vez especificado lo que hará y no hará el sistema se muestra una tabla con las características más detalladas del sistema.

Característica	Descripción
CA01	Vista tipo cuadro de mandos interactivo
CA02	Objetos a analizar: tuits
CA03	Control de acceso
CA04	Posibilidad de modificar criterios de análisis
CA05	Aplicación web
CA06	Extracción de datos y población de base de datos
CA07	Guardar análisis

Tabla 4.1: Características del sistema

4.1.3. Definiciones, acrónimos y abreviaturas

A continuación, en la tabla 4.2 se encuentran los términos recurrentes en el proyecto junto a una breve explicación del mismo.

Término	Definición
API	Conjunto de subrutinas, funciones y métodos que ofrece el módulo Twitter de Python para ser utilizado por el sistema.
Usuario	Persona que está registrada para el uso de algún sistema. En el proyecto se distinguen los usuarios de Twitter y los usuarios de Health Analytics.
Tuit	Mensajes de los usuarios de Twitter. Objeto de estudio del proyecto.
Hashtag	Etiqueta que sirve para clasificar el texto del propio tuit.
Geolocalización	Coordenadas desde las que se ha publicado un tuit.
Localización	Región, ciudad o país que el usuario de Twitter indica en su perfil.
Sistema	Aplicación web a desarrollar.
LDA	Modelo utilizado para la clasificación de temas en los textos del tuit
PyCharm	Entorno de desarrollo utilizado

Tabla 4.2: Tabla de términos

4.2 Descripción general

4.2.1. Funciones del producto

Para tener una visión clara se ha realizado un diagrama de casos de uso que se muestra en la figura 4.1. El diagrama sirve para identificar el contexto en el que se encuentra el sistema, los usuarios y las acciones que pueden desarrollar estos últimos en el sistema.

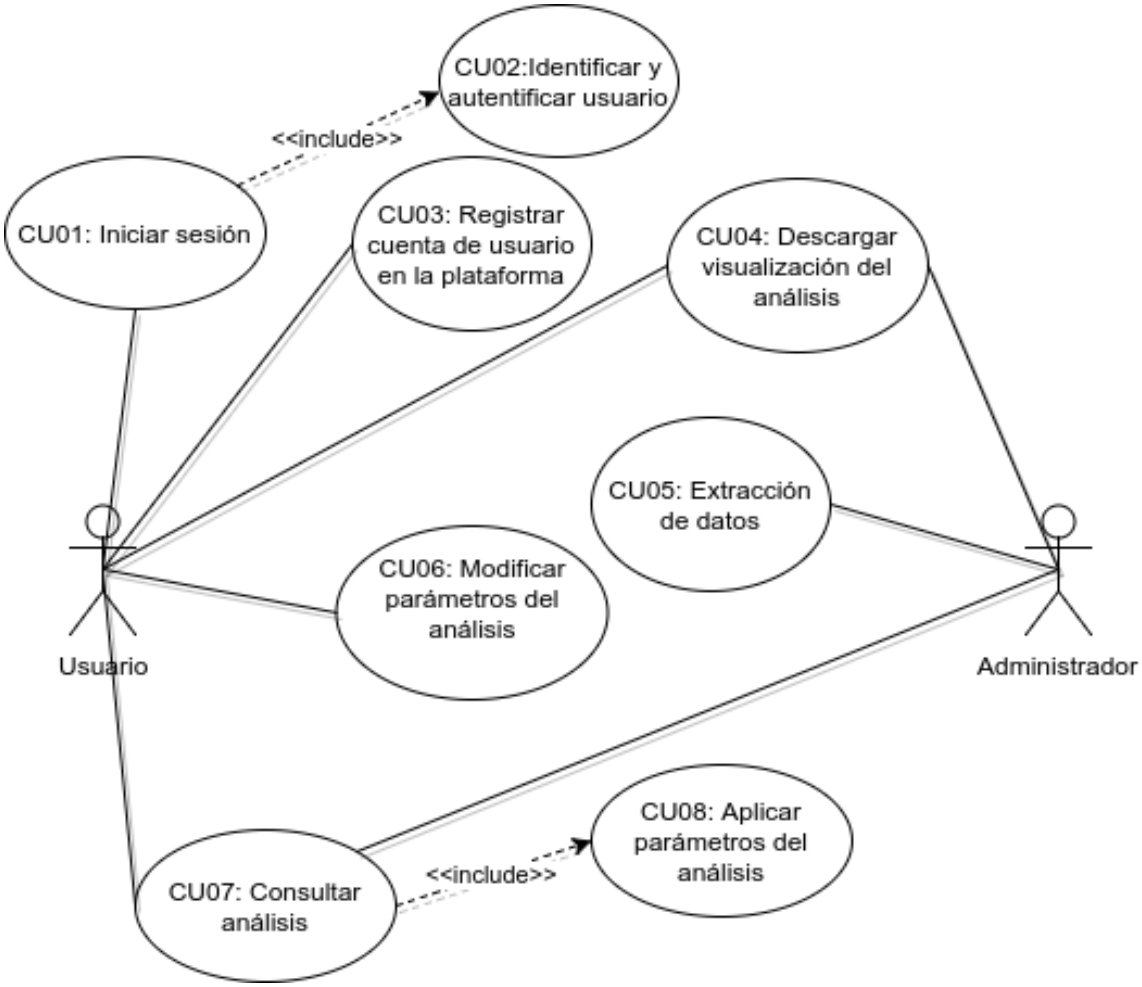

Figura 4.1: Diagrama casos de uso

A continuación, se muestra una tabla donde se relacionan los casos de uso con las características especificadas en la tabla 4.1.

Caso de uso	Descripción	Característica
CU01	Iniciar sesión	CA03
CU02	Identificar y autentificar usuario	CA03
CU03	Registrar cuenta de usuario	CA03
CU04	Descargar visualización de análisis	CA07
CU05	Extracción de datos	CA02 y CA06
CU06	Modificar parámetros análisis	CA04
CU07	Consultar análisis	CA01
CU08	Aplicar parámetros análisis	CA04

Tabla 4.3: Relación casos de uso y características

4.2.2. Características de los usuarios

El sistema es utilizado por dos roles de usuario: el usuario del servicio que se registra para obtener las visualizaciones del análisis y el usuario administrador que se encarga de la recolección de tuits y el poblado de la base de datos.

El usuario que se registra no debe por qué tener experiencia previa en utilización de herramientas de análisis de datos.

El usuario administrador es caracterizado por tener un control sobre las distintas herramientas existentes de análisis de datos. También es importante que el administrador tenga experiencia en tratamiento de datos, así como experiencia en bases de datos para el almacenamiento de la información.

4.2.3. Restricciones

Las restricciones del proyecto se pueden clasificar en restricciones de desarrollo y restricciones de uso. A continuación se listan las restricciones de cada grupo

Restricciones de desarrollo

1. Únicamente se recolectará la información de la red social Twitter.
2. Se utilizará la versión gratuita de la API, el límite establecido es de 180 peticiones cada 15 minutos.
3. Servidor web y servidor de base de datos correrán sobre un mismo equipo.

Restricciones de uso

1. Datos no actualizados en tiempo real.
2. Necesario el registro en la web.
3. Necesaria conexión a la red para acceder a la web.

4.2.4. Supuestos y dependencias

Se supone que el dato analizado (tuit) no será reemplazado por otro tipo de dato. El cambio de dato supondría un cambio estructural del sistema. También se debe destacar que el cambio del tipo de análisis a uno distinto al que se propone en el proyecto, podría afectar a la especificación de requisitos del sistema. Finalmente, para llevar a cabo un análisis de la forma más rigurosa será necesaria la recolección de un gran volumen de información. El número máximo de tuits a analizar (dada la restricción número tres de **Restricciones de desarrollo**) será de 2000 objetos recogidos en el periodo de un mes. Por lo que no se podrá sobrepasar los 24 000 tuits anuales para evitar colapsos.

4.2.5. Requisitos futuros

En futuros proyectos de mejora del sistema se podrían abordar los siguientes requisitos:

- Adición de equipos en la infraestructura de servidores.
- Utilización de la API de Twitter en tiempo real.

- Geolocalización de tuits almacenados.
- Analizador de sentimientos propio.

4.3 Requisitos específicos

Finalmente en esta sección de especifican los requisitos a un amplio nivel de detalle que se deberán abordar durante el desarrollo del proyecto.

4.3.1. Interfaces externas

RI01: Navegabilidad del sistema

- Descripción: El sistema proporcionará todos los mecanismos posibles al usuario para que sea utilizada la mayor proporción del sistema.
- Prioridad: Media
- Caso de uso: CU07

RI02: Integración del sistema

- Descripción:

  El sistema debe integrarse con la API de Twtiter para extraer los datos.

  El sistema debe integrarse con el sistema gestor de base de datos MongoDB.
- Prioridad: Alta
- Caso de uso: CU05

RI03: Modificación del estado del sistema

- Descripción: El sistema deberá interactuar de manera que el usuario pueda realizar cambios para obtener los resultados requeridos.
- Prioridad: Media
- Caso de uso: CU06

RI04: Visualizaciones

- Descripción: Las visualizaciones se dispondrán en un cuadro de mandos al que el usuario tendrá acceso tras identificarse.
- Prioridad: Alta
- Caso de uso: CU07

4.3.2. Requisitos de funcionalidad

RF05: Extracción de datos

- Descripción: El administrador deberá tener una cuenta de Twitter para proceder a la extracción de los datos.
- Prioridad: Alta
- Caso de uso:CU05

RF06: Registro de usuario

- Descripción: El sistema deberá permitir al usuario proceder al registro para acceder al análisis de los tuits. Para el registro, el usuario accederá a un formulario para ingresar nombre, contraseña y correo. Estos datos serán almacenados por el sistema.
- Prioridad: Media
- Caso de uso: CU03

RF07: Inicio de sesión

- Descripción: El sistema proporcionará un formulario de inicio de sesión en el que el usuario introducirá nombre y contraseña.
- Prioridad: Media
- Caso de uso: CU01 y CU02

RF08: Parámetros de análisis

- Descripción: El usuario será capaz de establecer el idioma y el intervalo temporal de los tuits que se requieran para el análisis.
- Prioridad: Media
- Caso de uso: CU06

RF09: Análisis-Temas frecuentes

- Descripción: El sistema proporcionará una visualización sobre los temas de los que se habla en los tuits.
- Prioridad: Alta
- Caso de uso: CU07

RF10: Análisis-Frecuencia de palabras

- Descripción: El sistema proporcionará una visualización sobre las palabras más frecuentes en los tuits.
- Prioridad: Alta
- Caso de uso: CU07

RF11: Análisis-Línea de tiempo

- Descripción: El sistema proporcionará una línea de tiempo para mostrar el número de tuits recogidos en el tiempo.
- Prioridad: Alta
- Caso de uso: CU07

RF12: Análisis-Idioma

- Descripción: El sistema proporcionará una visualización sobre los idiomas más frecuentes en los tuits.
- Prioridad: Alta
- Caso de uso: CU07

RF13: Análisis-Sentimiento

- Descripción: El sistema proporcionará una visualización sobre el análisis de sentimiento de los tuits analizados y la polaridad de las palabras.
- Prioridad: Alta
- Caso de uso: CU07

RF14: Análisis-Regiones geográficas

- Descripción: El sistema proporcionará una visualización sobre las regiones geográficas que más actividad registran en los tuits analizados.
- Prioridad: Media
- Caso de uso: CU07

4.3.3. Requisitos de rendimiento

RR15: Número de equipos

- Descripción: El sistema se desarrollará en un equipo donde residirá el servidor web y la base de datos en adición al sistema.
- Prioridad: Alta
- Caso de uso:-

RR16: Tiempo de respuesta

- Descripción: El sistema no debe tardar más de un minuto en procesar la información y generar las visualizaciones.
- Prioridad: Alta
- Caso de uso:-

RR17: Datos

- Descripción: El número de tuits para el análisis será de entre 1000 y 24 000 tuits.
- Prioridad: Alta
- Caso de uso:-

CAPÍTULO 5
Diseño de la solución

5.1 Arquitectura del sistema

El sistema está basado en el modelo-vista-controlador (MVC). Se crean tres módulos que interactúen entre ellos para crear una uniformidad en el sistema.

A continuación, en la figura 5.1 se muestra la relación existente entre los elementos del MVC [24]. Se debe destacar que lo que se pretende mostrar en la figura es la funcionalidad del modelo, sin aplicarlo aún al sistema que se está diseñando.

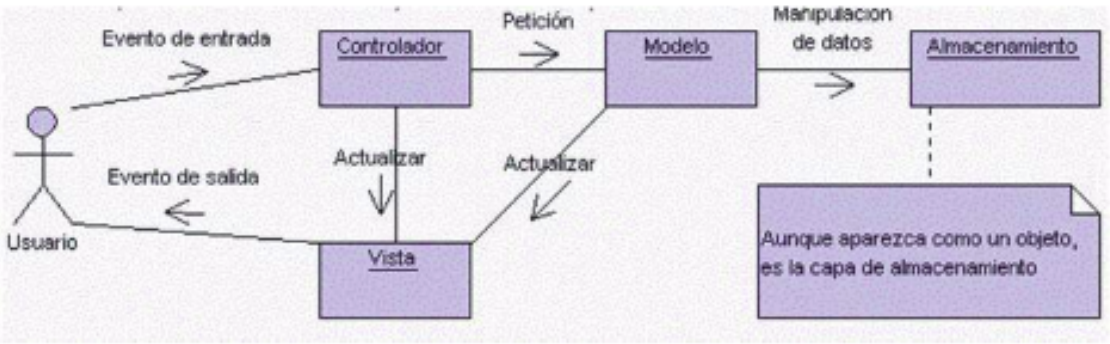

Figura 5.1: Relación existente entre los elementos del MVC

Modelo
Para llevar a cabo esta herramienta será necesario un módulo que conecte con la API de Twitter que será la que nos proporcione los datos para crear nuestro conjunto de datos. Estos datos recogidos deberán ser almacenados y el modelo se encargará de actualizaciones, consultas y búsquedas en la base de datos.

Controlador
Recibirá eventos de entrada por parte del usuario, dependiendo del evento, ejecutará parte del modelo o la vista necesario para generar respuestas.

Vista
Recibe datos por parte del controlador y el modelo para crear una visualización que será procesada mediante un evento de salida. Es el módulo encargado de la visualización de los datos

La elección de este patrón de diseño se basa en que el sistema a desarrollar se compone de una interfaz dinámica que varía dependiendo del valor que recojan diversas variables a lo largo de la sesión iniciada por el usuario. Se logra un desacople de las partes que permite que sean utilizadas en nuevos sistemas con el mínimo nivel de cambio posible.

El patrón MVC es frecuentemente utilizado en aplicaciones web cuya vista está compuesta de código *HyperText Markup Language* (HTML) y por código generador de datos dinámicos mientras que el modelo interactúa con el Sistema de Gestión de Bases de Datos (SGBD) y la lógica de negocio.

5.2 Tecnologías empleadas

5.2.1. API de Twitter

El acceso a estos tuits ya se ha remarcado que es público para cualquier usuario, siempre y cuando no se traten de tuits protegidos (tuits redactados por perfiles privados). Por esto, Twitter pone a disposición una API de búsqueda que permite la integración del servicio con otras aplicaciones o servicios web [25]. En la documentación de la API se puede leer que tras recibir una petición con determinados parámetros de búsqueda se proporciona una lista de tuits que cumplen cada uno de esos parámetros.

Hay multitud de parámetros, todos son opcionales menos uno, el parámetro 'q' en el que se indicará la consulta, en este parámetro se pueden especificar palabras o usuarios que queremos que aparezcan en los tuits que se devuelvan. Los parámetros utilizados, así como sus valores necesarios para la recolección de tuits del proyecto se detallarán en la sección de implementación. Otros parámetros de búsqueda útiles son: el parámetro de lenguaje para filtrar por idioma, parámetros de localización para filtrar por ubicación y el parámetro de conteo para indicar cuantos tuits deben devolverse.

Se presenta a continuación un ejemplo de consulta a la API especificando que se quiere realizar una búsqueda de tuits que contengan la palabra sanidad (parámetro q), que se hayan publicado cerca de Valencia (operador *near*), con un radio de búsqueda en millas (operador *within*) y cuyo idioma sea el Español (parámetro lang).

```
q=sanidad%20near%3A"Valencia%2C%20España"%20within%3A15mi&lang=es
```

5.2.2. Django

Antes de entrar a ver en profundidad a qué tecnología nos referimos cuando hablamos de Django, es necesario definir el concepto de *framework* ya que Django es un *framework* utilizado para el desarrollo de aplicaciones web.

Un *framework* no es únicamente utilizado en el ámbito del desarrollo web, suelen ser utilizados en la mayoría de campos relacionados con la informática como

puede ser el desarrollo de *software* sanitario, desarrollo de videojuegos, reconocimiento de imágenes, etc. Los *frameworks* otorgan el esqueleto de una aplicación estándar creada en base a módulos compuestos de clases, archivos, configuraciones y descriptores que pueden ser editados para componer la aplicación que se quiere desarrollar. La mayoría de los *frameworks* siguen el patron MVC explicado en la sección 5.1. [26]

Son muchos los *frameworks* para desarrollo de aplicaciones web que existen, a continuación se muestra tabla con el nombre de alguno de ellos asociado al lenguaje de programación empleado.

Framework	**Lenguaje**
Django	Python
Ruby on Rails (RoR)	Ruby
Laravel	PHP
Spring MVC	Java

Tabla 5.1: *Frameworks* web

Si comparamos RoR y Spring MVC con Django presentan similitudes. Por ejemplo, los tres se basan en lenguajes de programación orientados a objetos y siguen el patrón MVC. El factor que hace que nos decantemos por Django en este proyecto es debido al uso que se va a hacer de técnicas de aprendizaje automático y procesado de texto. Para el uso de estas técnicas, Python dispone de multitud de librerías cuyo objetivo es agilizar la implementación. También dispone de librerías que facilitan la integración con la API de Twitter y librerías de visualización [27].

Laravel es descartado debido a que PHP resulta ser un lenguaje cuya sintaxis provoca un aumento del número de líneas de código aumentando el tiempo de depuración de errores. El estilo de programación de Python concede al código una tabulación que hace que sea fácilmente entendible y ayuda a la ubicación dentro de un desarrollo complejo además de poseer reglas que ayudan a la identificación de errores.

Finalmente se debe destacar que en el ámbito de la seguridad Django proporciona mecanismos de protección. Serán empleadas las técnicas necesarias en el proyecto para prevenir las siguientes vulnerabilidades:

- *Cross site scripting* (XSS): Esta técnica que consiste en la inserción de código JS para corromper el sistema puede ser solventada debido a que en Django es posible utilizar un mecanismo para evitar caracteres maliciosos que puedan alterar el resultado de la vista. En la sección de implementación se describirá el mecanismo de solución de esta vulnerabilidad.

- *Cross site request forgery* (CSRF): Los ataques CSRF permiten ejecutar acciones usando credenciales de otro usuario sin que este se percate de la situación. En la sección de implementación se describirá el mecanismo de solución de esta vulnerabilidad.

Finalizando esta sección quedan expuestas las razones de la elección de Django como *framework* para el desarrollo de la aplicación de análisis de tuits relacionados con la sanidad.

5.2.3. MongoDB

Como SGBD para la aplicación del proyecto se opta por la utilización de MongoDB. MongoDB es un sistema de base de datos NoSQL que orienta el almacenaje de datos en documentos JSON, alejándose de las típicas bases de datos realacionales que almacenan la información en tablas. Para una mayor velocidad de almacenamiento, MongoDB hace uso del formato *Binary* JSON (BSON) que es una representación binaria de las estructuras de datos, sin embargo esos documentos BSON pueden tener un mayor tamaño que los documentos JSON.

MongoDB, como sistema gestor de base de datos, mantiene una jerarquía en su estructura (ver figura 5.2). El sistema se compone del número necesario de bases de datos que el usuario requiera. Estas bases de datos se componen de colecciones cuyo símil serían las tablas en los sistemas relacionales y estas colecciones agrupan documentos que podrían ser el símil de los registros en los sistemas clásicos. La gran diferencia es que estas colecciones no deben contener necesariamente documentos que representen mismos objetos JSON. La estructura de cada documento puede ser distinta y coexistir con otros documentos distintos en la misma colección [28].

En la práctica para obtener una mayor organización se suelen agrupar documentos similares dentro de una misma colección evitando por ejemplo tener en una misma colección tuits mezclados con credenciales de usuarios.

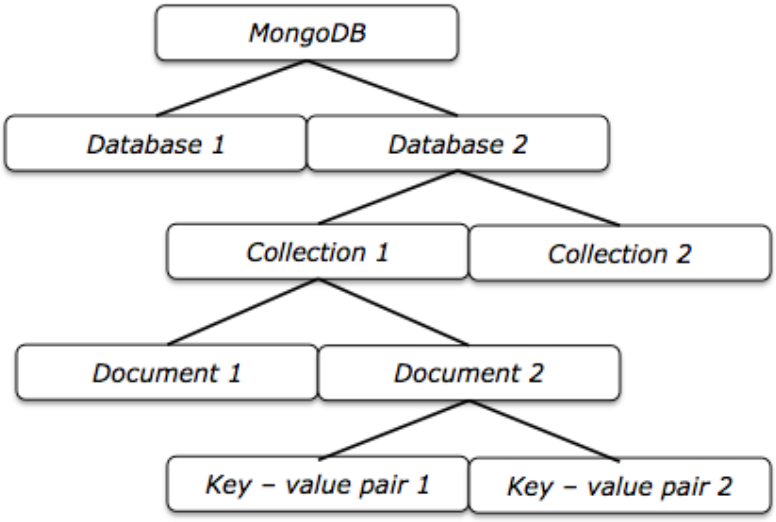

Figura 5.2: Organización estructural de MongoDB

MongoDB aporta esa flexibilidad de poder insertar objetos con campos distintos, flexibilidad y dinamismo que no es alcanzado por los sistemas de tipo relacional, donde son los datos los que se deben adaptar al sistema y no el sistema el que se debe adaptar a los datos. En términos de escalabilidad, los sistemas de tipo NoSQL como MongoDB ofrecen una escalabilidad horizontal, permitien-

do añadir cuantos servidores se requieran para poder servir a los usuarios del sistema, además, esta escalabilidad horizontal implica una alta disponibilidad ya que si por alguna razón algún servidor falla los demás siguen funcionando, cosa que en los sistemas SQL es más complejo ya que debido a la rigidez de estructura, suelen otorgar una escalabilidad vertical donde únicamente se van aumentando las prestaciones de un único nodo primario.

	SQL	**NoSQL**
Tipo	Relacional	No relacional
Información	Estructurada en tablas	Desestructurada Almacenada en ficheros JSON
Esquema	Estático	Dinámico
Escalabilidad	Vertical	Horizontal
Flexibilidad	Esquema rígido con dependecias	Alta flexibilidad

Tabla 5.2: Tabla resumen comparación bases de datos SQL vs. NoSQL

Para concluir esta sección se debe decir que MongoDB ha sido elegido como SGBD para el proyecto por dos principales razones. En primer lugar porque la fuente de datos nos otorga los datos en formato JSON, formato en el que trabaja MongoDB, y se aprovecha esta característica para evitar trabajo extra en la conversión y tratamiento de los datos. Y en segundo lugar por la flexibilidad que otorga, ya que los tuits recogidos no disponen de los mismos atributos y estas variaciones añadirían complejidad al almacenaje de los datos.

5.3 Diseño detallado

Django, basado en el MVC, hace uso de tres elementos clave que se deben tener en cuenta a la hora de realizar el diseño del sistema.

En primer lugar tenemos el modelo (capa de persistencia). Esta capa se encarga de realizar una abstracción sobre la base de datos y contiene todos los mecanismos para la creación, lectura, actualización y eliminación de componentes de la base de datos. Esta abstracción nos proporciona una reutilización del modelo sobre distintas bases de datos.

En segundo lugar tenemos las plantillas (capa de presentación). En esta capa se estructura la organización de la información en la página web.

Finalmente se pueden identificar las vistas (capa lógica de negocio). Esta capa se podría definir como un puente entre la capa de persistencia y la capa de presentación debido a que se encarga de establecer comunicaciones entre ambos para definir la presentación de los datos en la web.

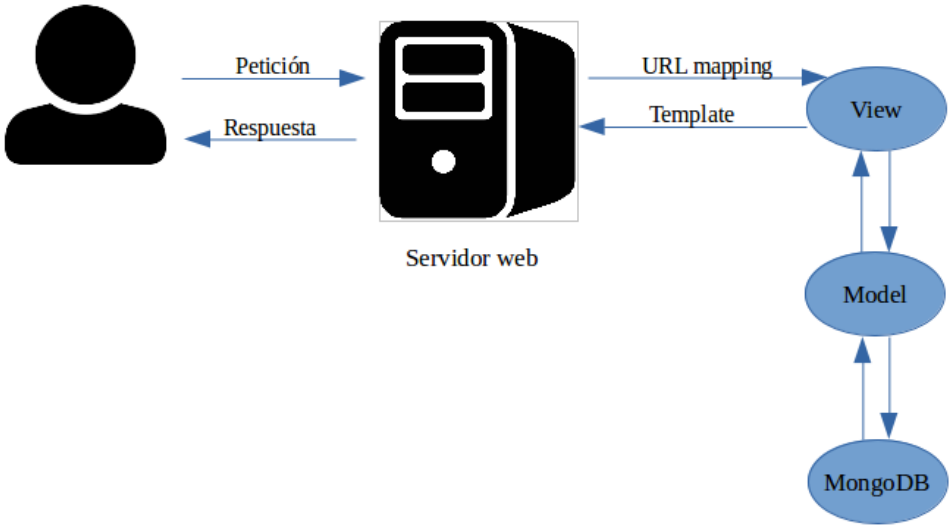

Figura 5.3: Esquema del sistema a desarrollar

En la figura 5.3 se puede ver el esquema que tendrá la aplicación.

5.3.1. Capa de persistencia

En el sistema se pueden identificar tres elementos diferenciados. En primer lugar tenemos la capa de persistencia que hará uso de mongoDB como ya se ha comentado anteriormente. En la base de datos se almacenarán los tuits requeridos en una colección y las credenciales de usuario en otra colección, obteniendo así una separación de los datos en función de su naturaleza. El fichero models.py contendrá las funciones necesarias para conectar a la API de Twitter, descargar y almacenar los tuits, además tendrá funciones de consulta sobre la base de datos o sobre ficheros JSON que contengan tuits. En la descarga de tuits se podrán indicar distintos parámetros como pueden ser lenguaje, palabras clave o *hashtags*.

La colección de MongoDB que contiene los tuits está formada por el objeto tuit que contiene los campos que se muestran en la figura 5.4. Debido a que se trata de un objeto muy complejo con multitud de campos, la herramienta hará uso de los campos que se especificaron en la la sección 3.1.1. Sin embargo se opta por almacenar el tuit entero por los campos que puedan resultar de interés para futuras investigaciones.

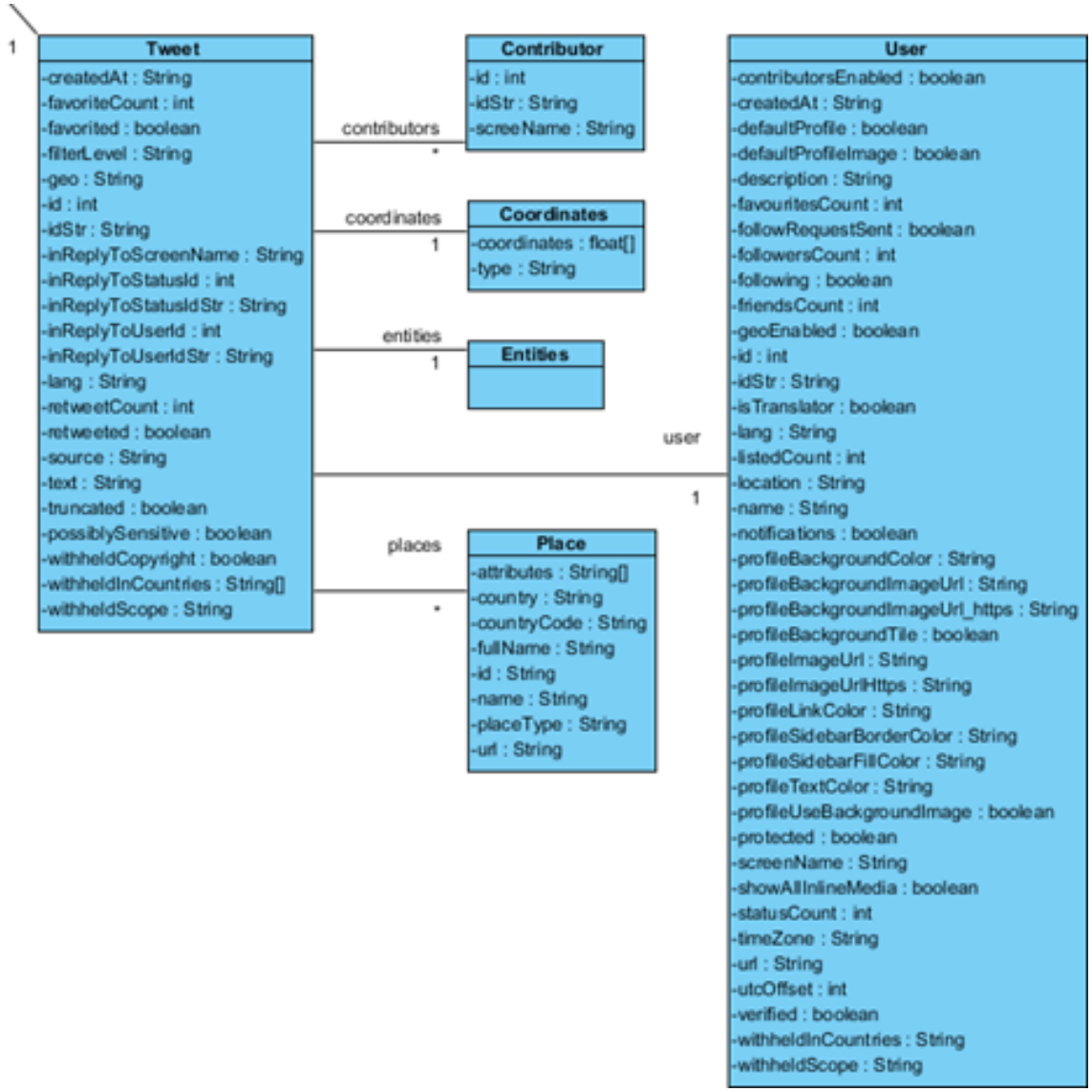

Figura 5.4: Campos de la colección tuit

La segunda colección de la que hace uso el sistema es la colección usuario (no confundir con el objeto *user* interno del tuit). Se trata de una colección sencilla que almacena objetos de tipo usuario en formato JSON que previamente se han registrado en la plataforma. Los campos son los siguientes:

- *Nickname*: cadena de texto que contiene el nombre de usuario que se ha registrado.
- *Pass*: cadena de texto que contiene la contraseña del usuario que se ha registrado.
- *Email*: cadena de texto que contiene el correo electrónico del usuario que se ha registrado.

Asociado a cada uno de estos objetos de tipo usuario MongoDB asocia un identificador único para cada uno de estos. Por seguridad, a estos valores del usuario que se almacenarán en la base de datos, se les aplica una función *hash* del

grupo SHA-2. En concreto se aplica la versión de 256 bits (SHA-256). Aplicando esta función, se logra transformar los datos introducidos por el usuario en un único valor y aquella persona que tenga acceso a la base de datos no podrá ver los campos en claro, verá los valores codificados. Cuando el usuario acceda a la web, los datos introducidos se transformarán mediante la función *hash*, se comparará con el *hash* almacenado en la base de datos y si el resultado coincide, el usuario tendrá acceso a la web.

La elección del algoritmo criptográfico se basa en la no existencia de colisiones en la serie SHA-2. Se dice que hay colisiones cuando al aplicarse un algoritmo criptográfico a dos conjuntos de datos distintos, genera el mismo código *hash*. Las colisones aumentan la probabilidad de que un usuario malicioso desarrolle conjuntos de datos para acceder a la información. Hasta la fecha las colisiones que se han conocido se han dado en lo siguientes algoritmos: MD5, SHA-0 y SHA-1.

5.3.2. Capa lógica de negocio

Para detallar la interacción lógica del sistema, se han desarrollado diagramas de secuencia asociados a los casos de uso identificados.

En la figura 5.5 se muestra el diagrama de secuencia relativo al caso de uso 5. El administrador para proceder a la extracción de datos, ejecuta en terminal un *script* que interactúa con el sistema. Este *script* realiza la conexión con la API, que devolverá los tuits de la búsqueda. Finalmente el sistema recibe los tuits, los almacena en la base de datos y notifica al usuario de que la acción se ha realizado correctamente.

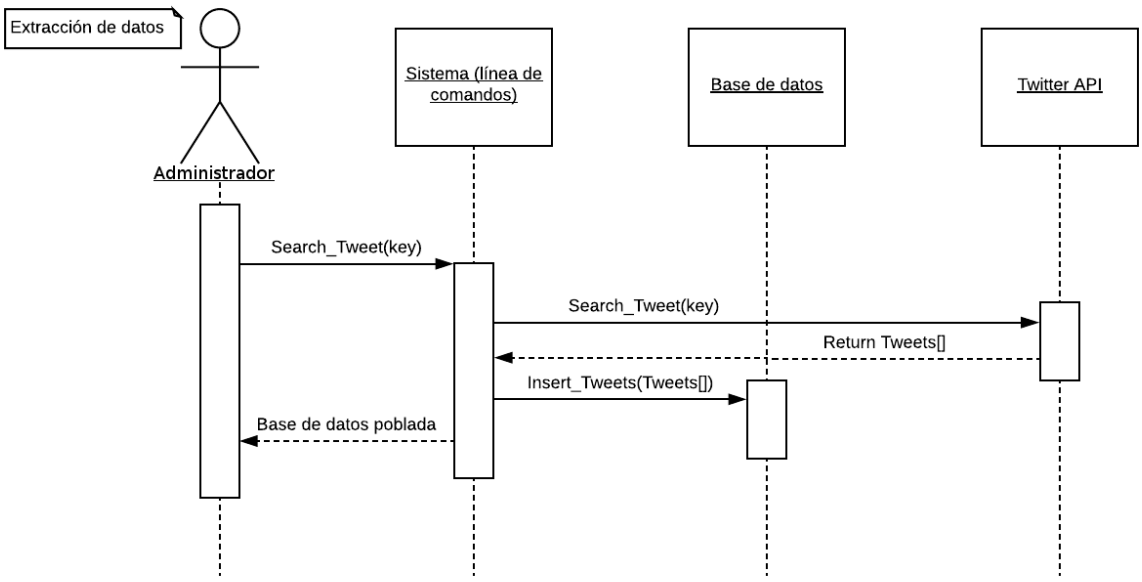

Figura 5.5: Diagrama secuencia recogida de datos (CU05)

En la figura 5.6 se muestran los pasos a seguir en el registro de usuario. Una vez rellenados los campos del formulario de registro, el sistema procede a comprobar que la cuenta de correo electrónico proporcionada es correcta. Si la com-

probación resulta satisfactoria, el sistema procede al almacenaje de los datos en la base de datos y la notificación del éxito del registro, de lo contrario el sistema informa del error al usuario y aborta el registro.

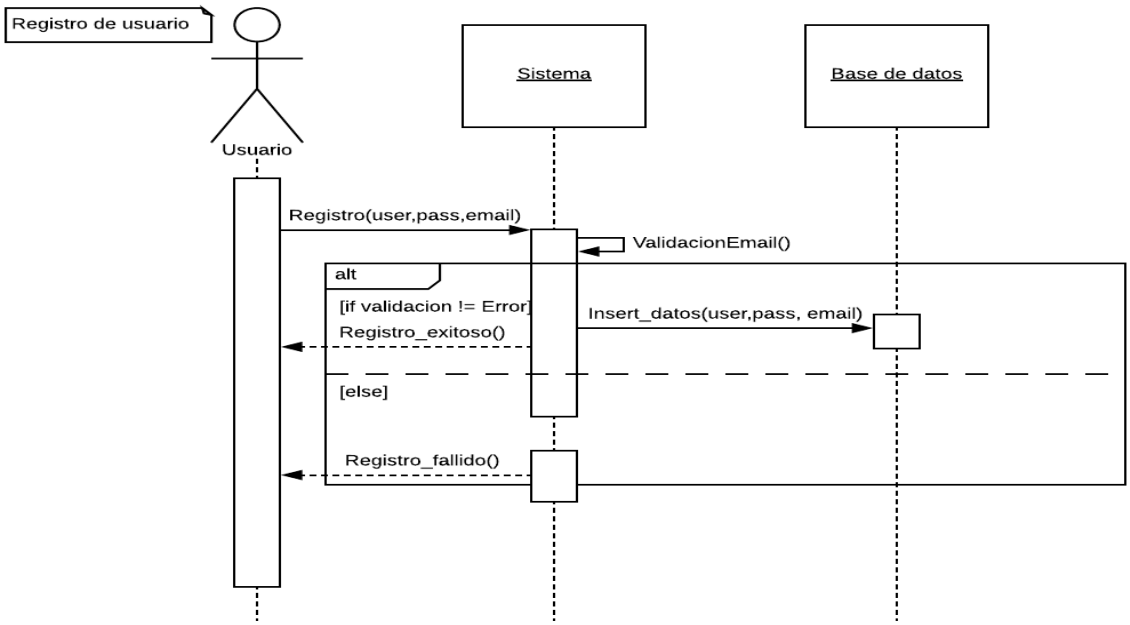

Figura 5.6: Diagrama secuencia registro de usuario (CU03)

En la figura 5.7 se muestra el diagrama de secuencia que sigue el sistema cuando un usuario inicia sesión. El usuario ingresa sus credenciales de acceso. A continuación el sistema recoge los datos de la base de datos para compararlos. Si la comparación resulta con éxito, el sistema da la bienvenida al usuario, en el caso contrario se hace saltar un error que informe al usuario de los sucedido.

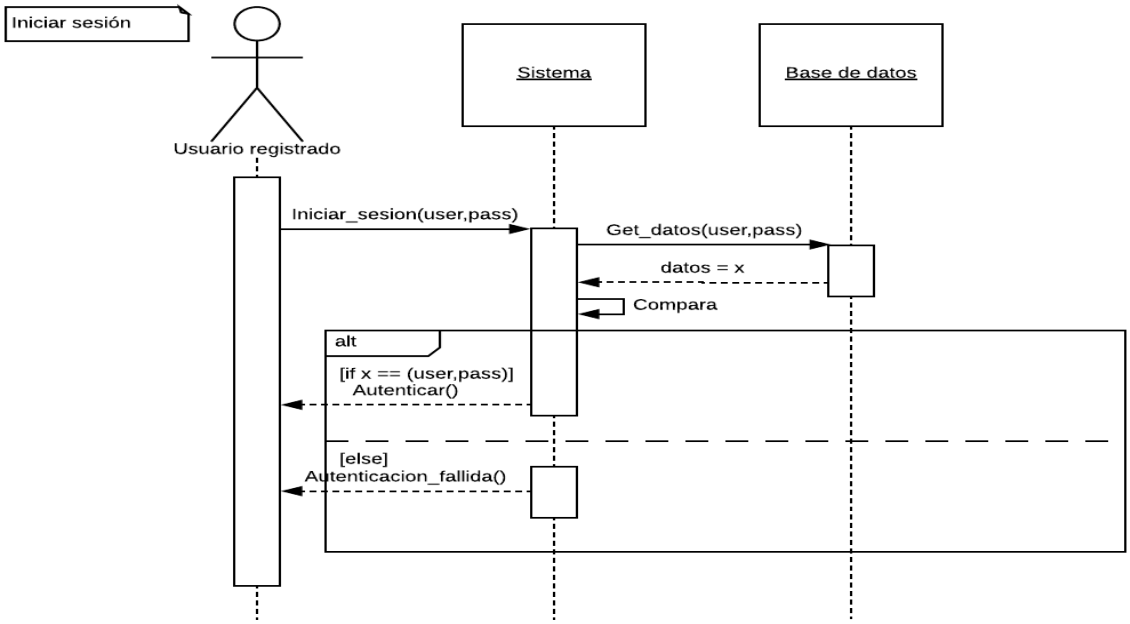

Figura 5.7: Diagrama secuencia inicio sesión (CU01 y CU02)

Finalmente, en la figura 5.8 se muestra el proceso de elección de parámetros y visualización del análisis generado. El usuario una vez ha accedido al sistema debe establecer unos parámetros para el análisis. Una vez establecidos estos parámetros, el sistema hará uso de las funciones de acceso a la base de datos para recoger los tuits necesarios. Una vez el sistema tiene la información, procederá al tratamiento y al renderizado de las visualizaciones. Estas visualizaciones serán presentadas al usuario en un cuadro de mandos con el que podrá interactuar.

Las funciones de tratamiento de datos y renderizado de las visualizaciones se tratarán en profundidad en el apartado 6.2.

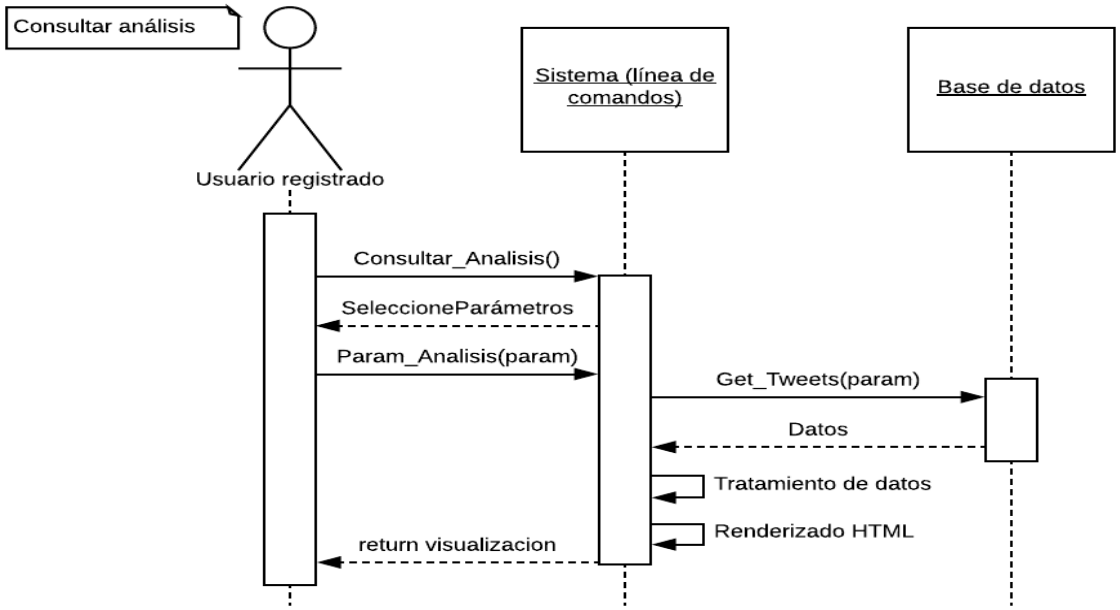

Figura 5.8: Diagrama secuencia consulta análisis (CU6, CU8 y CU7)

5.3.3. Capa de presentación

En esta sección se mostrarán prototipos primarios con el objetivo de presentar una primera idea sobre cómo será la interfaz del sistema, qué opciones tendrá y cómo el usuario podrá navegar entre las distintas pantallas.

Figura 5.9: Página bienvenida

En la figura 5.9 se muestra la página principal de inicio. En esta página el usuario puede ingresar al sistema de análisis de tuits mediante una cuenta que deberá crear con anterioridad. Se especifica mediante un enlace el lugar al que acceder en caso de que no se tenga una cuenta.

Figura 5.10: Página registro

En la figura 5.10 se observa la pantalla que ofrece al usuario los campos necesarios para llevar a cabo el registro. A esta página se accederá desde el enlace de la página de bienvenida. Una vez registrado con éxito el usuario será redirigido a la página principal para que proceda a iniciar sesión y acceder a la herramienta.

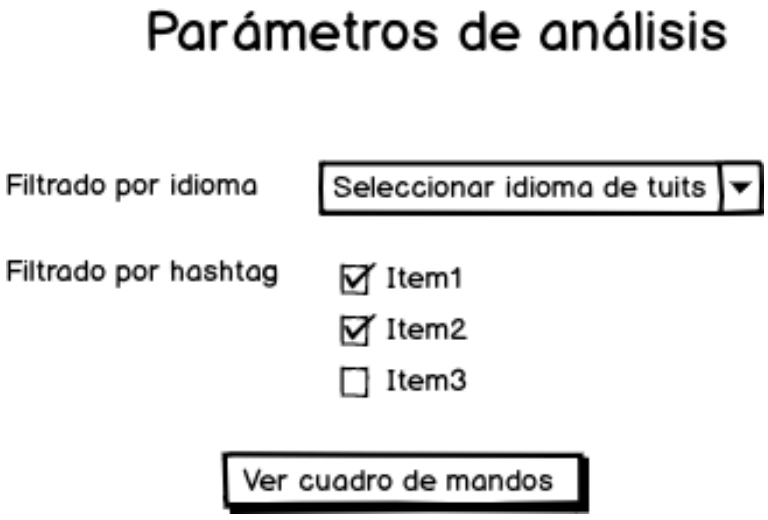

Figura 5.11: Página opciones de análisis

En la figura 5.11 se muestra la vista en la que el usuario podrá escoger el idioma de los tuits que generarán las estadísticas, así como realizar un filtrado en base a los *hashtags* seleccionados. Estas opciones no son de especificación obligatoria, si no se indica ninguna opción en los parámetros del análisis se realizará el

análisis de todos los tuits que haya en la base de datos. Una vez seleccionadas las opciones de visualización se podrá acceder al cuadro de mandos generado.

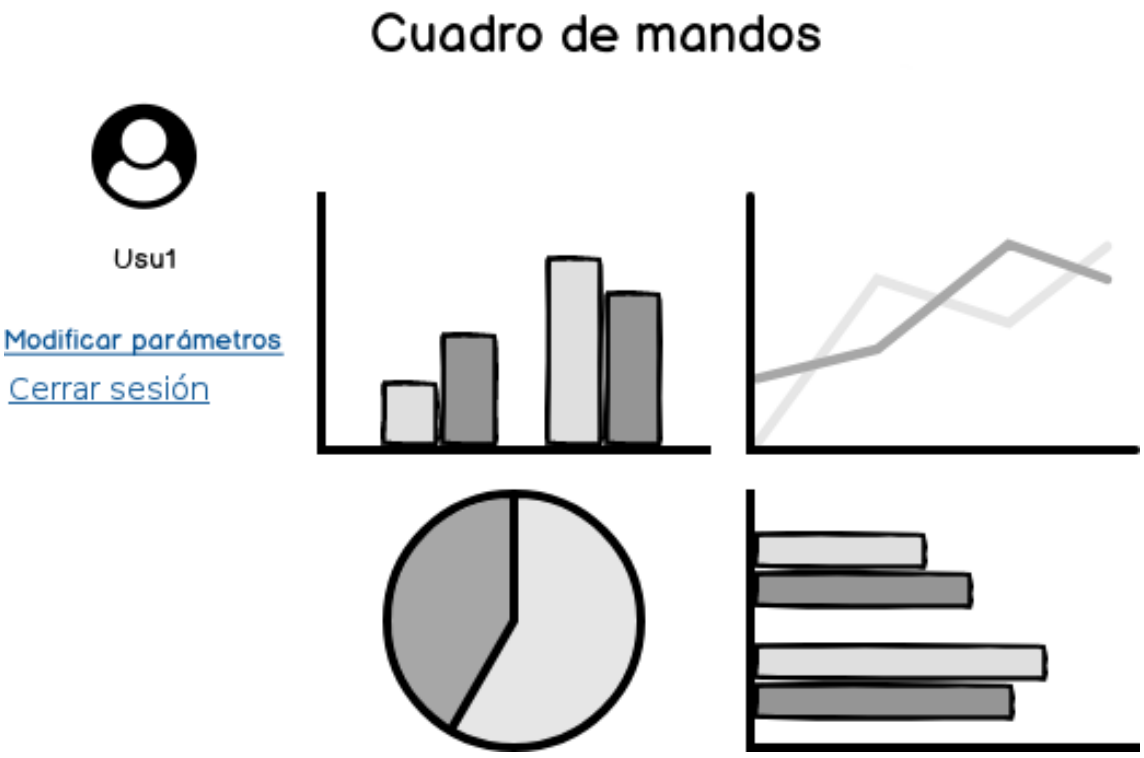

Figura 5.12: Cuadro de mandos del usuario

En la figura 5.12 se muestra el cuadro de mandos que obtendría el usuario en función de los parámetros seleccionados. Desde esta vista el usuario podrá interactuar con los distintos gráficos, descargar las visualizaciones, editar el valor de los parámetros de análisis y cerrar sus sesión.

Las visualizaciones que se mostrarán serán las siguientes:

- Diagrama de barras con palabras frecuentes: se establecerá una clasificación de las palabras más recurrentes en los tuits. Este diagrama variará en función del idioma de los tuits analizados. En relación directa a esta visualización también se aportará una nube de palabras donde se mostrará en distintos tamaños los términos más recurrentes.
- Temática: se aplicará modelo *Lineal Dirichlet Allocation* (LDA) para establecer qué palabras suelen aparecer en conjunto y tienen relación entre ellas.
- Visualización temporal: linea de tiempo que mostrará el número de tuits recogidos durante el tiempo. El objetivo es visualizar si a lo largo del año se producen picos de actividad de tuits relacionados con la sanidad en épocas de epidemias, aumento de infecciones, aplicación de medidas políticas en el ámbito sanitario, huelgas, etc.

- Análisis de sentimiento de los tuits: se presentará un diagrama de tarta con el porcentaje de tuits cuyo texto ha sido evaluado como positivo, negativo o neutro. También se generará una clasificación de las palabras más comunes en la que se mostrará la polaridad de cada palabra.
- Análisis de idiomas: se presentará un diagrama de tarta que mostrará el número de tuits recogidos por cada idioma que se tenga en la base de datos.
- Análisis de regiones activas: se establecerá una clasificación de las regiones geográficas que más actividad registran en los tuits analizados.

CAPÍTULO 6
Desarrollo de la solución propuesta

6.1 Creación de una aplicación en Twitter

Ya que la herramienta a desarrollar necesita hacer uso de la API de Twitter, es necesario proceder a la creación de una aplicación en Twitter para obtener el acceso a la API.

En primer lugar, para registrar una aplicación en Twitter es necesario tener una cuenta de usuario en Twitter. A continuación, una vez creada la cuenta de usuario es necesario acceder al enlace[1] correspondiente para proceder a la creación de la aplicación. Una vez creada la aplicación esta es asociada a la cuenta de usuario de Twitter del desarrollador, entre las opciones de la aplicación creada aparecen cuatro parámetros necesarios que deberán ser incluidos en el código de la aplicación para poder realizar las consultas a la API de Twitter.

- *Consumer Key (API Key)*: es la clave API asociada con la aplicación (Twitter). Esta clave identifica al cliente. En nuestro caso el cliente es la aplicación que se ha desarrollado ya que se trata de un servicio que intenta acceder a los recursos de un usuario.

- *Consumer Secret (API Secret)*: es la contraseña del cliente que se utiliza para autenticarse con el servidor de autenticación de Twitter.

- *Access Token*: es lo que se emite al cliente una vez que el cliente se autentica correctamente (utilizando API Key y API Secret). Este *token* de acceso define los privilegios del cliente (a qué datos puede y no puede acceder el cliente).

- *Access Token Secret*: Cada vez que el cliente desea acceder a los datos del usuario final, el *token* secreto de acceso se envía con el token de acceso como contraseña.

Una vez generados estos parámetros, la inclusión de los valores dentro del código proporcionará a la aplicación la funcionalidad de poder utilizar la API de Twitter y extarer los tuits necesarios.

[1] https://apps.twitter.com/

6.2 Estructura e interacción con el sistema

A la estructura presentada del modelo MVC se debe añadir un fichero clave para el acceso a la web, este es el fichero urls.py. En este fichero se indican las rutas a las que accede el cliente. Para cada ruta se indica la función del fichero views.py (lógica de negocio) que se ejecutará (ver figura 6.1).

```
urlpatterns = [
    url(r'^grafico/static/$',views.getJson),
    url(r'^final/static/$',views.getJson),
    url(r'^signup/$', views.signupform),
    url(r'^login/$', views.login),
    url(r'^login/result.html$', views.result),
    url(r'^login/config.html$', views.config),
    url(r'^login/registro.html$', views.mostrarRegistro),
    url(r'^final/$', views.graficos),
]
```

Figura 6.1: Configuración del fichero urls.py

El funcionamiento general del sistema es el siguiente: con el servidor en funcionamiento se accede a una dirección web, dado ese evento se disparará la función asociada a esa URL, esa vista por un lado podrá comunicarse con la capa de persistencia (models.py) para obtener datos y por otro lado manejará los datos para crear visualizaciones que las embeberá en las páginas HTML que debe renderizar en último lugar.

A lo largo de esta sección se seguirá el flujo de interacción con el sistema y se irán detallando los aspectos más relevantes de las funciones que se vayan ejecutando. Se mostrará el código de la función cuando se precise ilustrar el funcionamiento de algún aspecto muy concreto.

En el fichero models.py se han desarrollado todas las funciones relativas a la capa de datos. A continuación se va a detallar el desarrollo de las funciones que se van ejecutando durante la interacción con el sistema.

6.2.1. Recolección de datos

La recolección de datos en el primer uso es necesaria para poblar la base de datos y que los usuarios puedan proceder al análisis de los tuits. Para cumplir este requisito se ha desarrollado una función propia de la capa de persistencia que establece conexión con la API de Twitter. Una vez establecida la conexión realiza una búsqueda en función de los parámetros que se especifiquen en el *query* de búsqueda. La API devuelve un número concreto de tuits, que son almacenados en la colección indicada a la que previamente se ha conectado.

6.2.2. Registro de usuario

Cuando un usuario rellena el formulario de registro se procede al registro y es disparada la vista correspondiente. En la figura 6.2 se muestra el código correspondiente a la vista que se activa cuando el usuario se registra y se resalta la función **insertUser** necesaria para almacenar los datos.

```
def login(request):
    if request.method == 'POST':
        template = loader.get_template('index.html')
        context = {}
        email = request.POST['email']
        username = request.POST['username']
        password = request.POST['password']

        email = hashlib.sha256(email.encode('utf-8')).hexdigest()
        username = hashlib.sha256(username.encode('utf-8')).hexdigest()
        password = hashlib.sha256(password.encode('utf-8')).hexdigest()


        models.insertUser(email, username, password)

        return HttpResponse(template.render(context, request))
    else:
        template = loader.get_template('index.html')
        context = {}
        return HttpResponse(template.render(context, request))
```

Figura 6.2: Vista disparada en el registro

Esta vista hace uso de la función **insertUser**, propia de la capa de persistencia. La función **insertUser** proporciona los mecanismos para insertar un usuario que se ha registrado en el sistema en la base de datos. Para ello, en primer lugar es necesario realizar una conexión al SGBD MongoDB haciendo uso del módulo Pymongo. Este módulo proporciona las funciones necesarias para establecer conexión con el SGBD, realizar consultas, insertar datos, eliminar y actualizar. En esta función, una vez realizada la conexión a la base de datos, dado un usuario, una contraseña y un correo electrónico se crea un documento JSON con estos valores y se inserta en la colección especificada. En la figura 6.3 se puede ver el proceso de conexión indicando la ruta del servidor, creación del objeto e inserción del mismo.

```
def insertUser(email,username,password):
    connection = pymongo.MongoClient("mongodb://localhost:27017")
    db = connection.tweets
    record = db.usu
    record.insert({"email":email,"user":username,"pass":password})
    return 0
```

Figura 6.3: Función insertUser (models.py)

6.2.3. Inicio de sesión

Una vez registrado, el usuario se dispone a iniciar sesión, para ello se ha desarrollado un formulario en el que se debe especificar el nombre de usuario y la

contraseña. La vista que se ejecuta al enviar los datos vía formulario es la función **config** del archivo views.py. Esta función que se muestra en la figura 6.4 accede a las variables del formulario de inicio de sesión, el valor de las variables es pasado a la función **checkPassUser** (models.py) para que compruebe el usuario y dependiendo del valor de retorno renderiza la página HTML de un modo u otro. Si el usuario está registrado se renderiza la página HTML perteneciente a la elección de los parámetros de análisis.

```
def config(request):
    if request.method == "POST":
        template = loader.get_template('config.html')
        username = request.POST['username']
        password = request.POST['password']
        check = models.checkPassUser(username, password)
        if check:
            global USER
            USER = request.POST['username']
            context = {'username': username}
            return HttpResponse(template.render(context, request))
        else:
            template = loader.get_template('index.html')
            context = {'error': True}
            return HttpResponse(template.render(context, request))
    else:
        template = loader.get_template('config.html')
        context = {'username':USER}
        return HttpResponse(template.render(context, request))
```

Figura 6.4: Función config (views.py)

En la función de comprobación de usuario, cuyo código fuente se encuentra en el apéndice A, dado un usuario y una contraseña se ejecuta la función **find** del módulo Pymongo. Con ello se establece una búsqueda del objeto JSON y se cuenta el número de resultados de la búsqueda. Si el resultado es 1, es que se ha encontrado en la base de datos el usuario y la contraseña asociada entonces devuelve una variable con el valor *True* que será devuelta a la función **config** del proyecto para que el usuario acceda a la plataforma.

```
...
if record.find({"user":username,"pass":password}).count() == 1:
   userExists = True
...
```

En el caso de que no se encuentre ningún usuario y contraseña que coincida con los pasados a la función se procederá a ejecutar una alerta de error.

Se puede observar en la figura 6.4 como en el primer *else* se crea una variable de contexto con un valor. Las variables generadas en las vistas que se adjuntan al contexto que es renderizado junto al HTML, son variables cuyo valor viaja al HTML que debe renderizarse, con el objetivo de que una misma página HTML pueda tener múltiples visualizaciones y pueda mostrar datos distintos. En este ejemplo del que se está hablando, si observamos el código del fichero index.html (apéndice C) , que es el que se renderiza en el primer *else*, se puede observar lo siguiente.

```
...
 {% if error %}
     <script>
         alert('Fallo de autenticación, inténtelo de nuevo o regístrese');
     </script>
 {% endif %}
...
```

Con este desarrollo se comunica al sistema que, si el valor de la variable error es verdadero, se debe ejecutar el fragmento de código indicado.

La idea de la variable contexto es la idea clave de funcionamiento de Django. El contexto es el que permite el flujo de información entre la capa de la lógica de negocio y la capa de presentación. Todas y cada una de las funciones que se han desarrollado contienen esta variable contexto que permite, por ejemplo, que tras el manejo de datos y creación de un gráfico, este gráfico se pueda enviar a la capa superior y pueda ser embebido en el código HTML.

6.2.4. Elección de parámetros de análisis

Iniciada la sesión, el usuario es redirigido a una página donde puede especificar el idioma de los tuits que quiere analizar. Para ello se ha optado por realizar un formulario HTML en el que el usuario pueda elegir entre analizar todos los idiomas o seleccionar tuits en español, inglés o portugués. Una vez hecha esta elección, el evento hace saltar la función **graficos**. Esta función es muy compleja y extensa así que se mostrará pseudocódigo para tener una visión general (ver figura 6.5). En el apéndice B se puede visualizar el código de la función completo.

```
graficos(idioma, estacion):
    si idioma = todos
        tweets = funcionBusquedaBD(idioma, estacion)
        LlamadafunciónLDA(tweets)
        LlamadafunciónFrecuenciaPalabras(tweets)
        LlamadafunciónIdiomasFrecuentes(tweets)
        LlamadafunciónWordCloud(tweets)
        LlamadafunciónAnálisisSentimientos(tweets)
        LlamadafunciónGráficaTemporal(tweets)
        LlamadacreaciónVariablesContexto
    si idioma = inglés
        tweetsIdioma = funcionBusquedaBD(idioma, estacion)
        LlamadafunciónLDA(tweetsIdioma)
        LlamadafunciónFrecuenciaPalabras(tweetsIdioma)
        LlamadafunciónWordCloud(tweetsIdioma)
        LlamadafunciónAnálisisSentimientos(tweetsIdioma)
        LlamadafunciónGráficaTemporal(tweetsIdioma)
        LlamadacreaciónVariablesContexto
    si idioma = español
        ...
        idem que caso en inglés
        ...
    si idioma = portugués
        ...
        idem que caso en inglés
        ...
 return (renderHTML + variables de contexto)
```

Figura 6.5: Pseudocódigo función graficos (views.py)

Como se puede deducir, la mayor carga computacional se encuentra en esta función que se ha presentado. En esta función se accede a los datos, se leen, se limpian y se aplican diversos algoritmos para crear las visualizaciones deseadas.

A continuación se va a detallar como se procede a la limpieza de los datos y a explicar los distintos algoritmos que se han aplicado para obtener datos estadísticos y presentarlos. Todas las funciones que se van a desarrollar a partir de este punto del documento, se encuentran el el fichero views.py, ya que se parte de que ya se han recogido los datos por parte de la capa de persistencia.

Limpieza de texto

Para analizar el campo texto del objeto tuit es necesario hacer un tratamiento del mismo. El objetivo de este tratamiento será el de eliminar signos de puntuación, emoticonos, símbolos como los *hashtags* y todos aquellos elementos que puedan provocar excepciones y dificultar la tarea del análisis.

Para hacer más cómodo el manejo de los objetos JSON se utiliza la librería pandas. Esta librería nos permite organizar todos los campos del objeto tuit en columnas para que el acceso a los mismos sea directo. Dada la complejidad del objeto tuit es necesario acceder a campos que están anidados, así que con el uso de pandas se consigue una disminución del código fuente.

Una vez seleccionado el campo texto se guarda en una lista el texto en bruto del tuit. Tras esto, a cada elemento de la lista (que es un tuit) se le aplica la función **textClean**. El código de esta función se encuentra en el apéndice D. En esta función, mediante el uso del módulo Re, que permite establecer expresiones regulares para que se adecuen a los patrones que siguen los elementos que se quieren eliminar, se realizan los siguientes pasos.

1. Elimina caracteres y símbolos HTML.
2. Elimina enlaces que puedan aparecer en el texto.
3. Elimina almohadillas de los *hashtags* y arrobas que haya en cuerpo del tuit.
4. Elimina signos de puntuación.
5. Elimina siglas.
6. Reduce el tamaño de palabras que contengan letras seguidas repetidas.
7. Elimina emoticonos que se puedan haber formado con paréntesis, coma, punto y coma, etc.
8. Elimina conjunciones.

Una vez el texto del tuit ha pasado por este tratamiento, el tuit queda listo para proceder al análisis.

Modelo LDA

LDA viene del inglés *Lineal Dirichlet Allocation* y es un modelo empleado en aprendizaje automático para descubrir categorías. El modelo es utilizado en el sistema que se ha desarrollado para establecer una clasificación de las palabras que por su aparición en el texto suelen ir juntas.

Para llevar a cabo la clasificación se ha hecho uso del módulo Scikit-learn. Este módulo contiene múltiples funciones para agilizar el aprendizaje automático. Se ha utilizado una función que emplea el modelo LDA con algoritmos de inferencia bayesiana en línea. Estos algoritmos se utilizan para deducir la probabilidad de que un supuesto pueda ser verdadero haciendo uso de evidencias. A esta función se le pasan como parámetros el número de temas, el número de iteraciones máximas que se desea realizar y el método de aprendizaje, que en nuestro caso se como valor del parámetro de aprendizaje *online*.

Frecuencia de palabras

Para realizar este tipo de análisis, la función encargada de realizar el conteo de palabras recibe en primer lugar la lista de texto limpio, cada elemento corresponde al texto de un tuit. Con esto se obtiene una matriz con las ocurrencias de las palabras, una vez desarrollada la matriz se procede a realizar la suma de ocurrencias por cada columna para extraer el número de veces que aparece cada palabra, para llevar a cabo el cálculo de ocurrencias, esta matriz irá ligada a un vocabulario que servirá para asociar vocabulario con columnas. A modo explicativo se expone a continuación un ejemplo.

Imaginemos que tenemos un vocabulario de tres palabras y se pretende analizar la frecuencia de estas palabras en un conjunto de frases (en el ejemplo cuatro frases). Aplicando las funciones determinadas se obtiene la matriz resultante.

```
 >>> vocabulario=['frío', 'sueño', 'calor']
 >>> vocabulario.fit_transform(['hace frio', 'tengo sueño', 'hay comida',
    'hace calor']).toarray()
array([[1, 0, 0],
       [0, 1, 0],
       [0, 0, 0],
       [0, 0, 1]])
```

Como se puede observar si sumamos cada columna, que se correspondería con cada elemento del vocabulario, se obtiene el número de veces que aparece.

En nuestro sistema el vocabulario estaría formado por todas las palabras de todos los tuits que se estén analizando y las oraciones del ejemplo se corresponderían con el texto de cada tuit.

Una vez se tienen los valores se establece una clasificación de las palabras y se desarrollan dos visualizaciones. Una de ellas se trata de un diagrama de barras donde se muestran las palabras más comunes, este diagrama es generado haciendo uso de la librería Bokeh, que permite la creación de gráficos interactivos. La otra visualización que se genera es una nube de palabras donde las palabras más comunes ocuparán un mayor espacio en la imagen que se genere, esta nube de palabras es generada haciendo uso de la librería WordCloud.

Idiomas frecuentes

Para determinar el volumen de tuits analizados en los distintos idiomas, se ha hecho uso nuevamente de la librería pandas para acceder en este caso al campo de idioma del objeto tuit. El idioma en los tuits viene determinado por un código, el diccionario generado donde se encuentran relacionados los códigos y los idiomas se encuentra en el apéndice E.

Una vez accedido al campo idioma de cada tuit se recupera el código del idioma y se hace un *matching* con el diccionario. Si se encuentran ocurrencias se genera un contador para cada uno de los idiomas presentes. En el caso de que algún código no se encuentre en el diccionario se creará una variable auxiliar que indique los idiomas indefinidos. Finalmente, haciendo uso de la librería Matplotlib de Python se procede a realizar un diagrama de tarta donde se muestre la proporción de idiomas que se han encontrado entre el análisis.

Análisis de sentimiento

Para relizar el análisis de sentimiento sobre el texto que contiene un tuit se ha desarrollado una función que emplea la librería **TextBlob**. **TextBlob** ofrece múltiples funciones que permiten llevar a cabo tareas de procesado de lenguaje natural. En el sistema que se ha desarrollado se obtiene una polaridad para el texto de cada tuit. La polaridad se encuentra en el rango [-1.0, 1.0], siendo menor que 0 el valor asociado a una palabra negativa y mayor que 0 el valor asociado a una palabra positiva.

En nuestra clasificación indicaremos que los tuits que obtengan una polaridad menor a cero serán negativos, los tuits que obtengan una polaridad igual a cero serán neutros y los tuits que tengan una polaridad mayor a cero serán positivos. A su vez, se ha utilizado la misma función para analizar la polaridad de las palabras más frecuentes, presentando una tabla con las palabras, el número de apariciones y el valor de la polaridad.

Una vez se han analizado todos los tuits, la función genera un diagrama de tarta indicando el porcentaje de tuits relativos a cada estado.

Una de las limitaciones que se contempla en el análisis de sentimiento es que **TextBlob** está orientado al inglés, por lo que los resultados que daría si se analizaran tuits en idioma distinto al inglés variaría. Para solucionarlo se ha optado por traducir la palabra de la que se quiere extrear la polaridad haciendo uso de la librería **Goslate**. Esta librería porporciona una función de conexión con un servicio web para traducir las palabras que se le envíen. Con esto conseguimos solventar el problema del idioma pero aparece el problema de la limitación, ya que al ser una API de traducción gratuita el número de llamadas está limitado por lo que el número de palabras de las que se puede extraer la polaridad cuando sea un idioma distinto al inglés es escaso.

Análisis de actividad

En esta función se hace uso de la fecha de publicación del tuit para establecer una gráfica con la actividad registrada. Nuevamente se hace uso de pandas para acceder al valor del campo de publicación. Una vez archivada la fecha de publicación se ordenan los tuits temporalmente de más antiguo a más reciente.

A continuación se dividen en grupos, cada grupo pertenece a un mes del año y por último se cuenta el número de tuits de cada grupo. Finalmente se crea una visualización donde el eje X indica el mes de publicación y en el eje Y se indica el número de tuits.

Análisis de las regiones geográficas más activas
Para analizar de dónde provienen los tuits y establecer una clasificación de las regiones más activas, se ha desarrollado una función que recoge el campo *location* del tuit. Para ello, una vez se ha accedido a los datos se iteran y se va añadiendo a una lista las múltiples localizaciones, tras esto se hace uso de un extracto de la función que mide la frecuencia de palabras para extraer la frecuencia de localizaciones. Así se obtiene un diagrama de frecuencias pero en este caso con las localizaciones de los usuarios que han redactado el tuit.

6.2.5. Conclusión

Estas son todas las funciones que se ejecutan cuando el usuario ha elegido los parámetros de análisis. Una vez ejecutadas todas estas funciones, se procede al renderizado del cuadro de mandos que aparece para el usuario. Todas las funciones que se han detallado generan unos gráficos que son enviados a la capa de presentación haciendo uso de las variables contexto. Así termina la respuesta al evento generado por el usuario al haber seleccionado los parámetros.

A modo de resumen, una vez el usuario ha elegido los parámetros de análisis y quiere acceder al cuadro de mandos, es redirigido a una URL que hace saltar la función **graficos** perteneciente a la capa de la lógica de negocio (cuyo pseudocódigo se ha mostrado en la figura 6.5). Esta función interactúa con la capa de persistencia para que las funciones de esta capa obtengan los datos y sean devueltos a la capa lógica. Una vez se tienen los datos, estos son consumidos por las funciones realizadas para las visualizaciones. Finalmente, la función **graficos** renderiza y presenta la visualización con el cuadro de mandos generado en HTML.

6.3 Desarrollo contra vulnerabilidades

En la sección 5.2.2 en la que se hablaba de las aportaciones que Django ofrecía frente a otros *Frameworks*, se comentaron un par de vulnerabilidades que podían ser resueltas con el uso de Django, estas vulnerabilidades eran ataques XSS y CSRF.

Como ya se explicó en el apartado correspondiente en qué consistían estas técnicas de ataque, en esta sección se van a presentar las partes del desarrollo que las combaten.

Como se muestra en la figura 6.6, tras haber introducido una información en un campo de un formulario, se utiliza esa información para dar la bienvenida. Esta variable es sometida a un tratamiento por parte de Django que transforma determinados caracteres. La transformación es la siguiente.

- < es convertido en <

- > es convertido en >
- ’ (comilla simple) es convertido en '
- "(comilla doble) es convertido en "
- & es convertido en &

Esta conversión anula cualquier código JavaScript que haya sido insertado en campo de formulario, evitando con esto los ataques XSS.

```
<form name="login-form" class="login-form" action="result.html" method="POST">
        {% csrf_token %}
        <div class="header">
        <h1>Bienvenido/a <b>{{ username }}</b></h1>

        </div>
        <div class="content">
            <span>Por favor, espere hasta que le redirijamos al panel del control</span>
        </div>
```

Figura 6.6: Ejemplo de variable sometida a tratamiento

Para no añadir una carga computacional extra a la ya existente generada por el gran volumen de datos que se maneja, esta conversión es desactivada para variables que han sido generadas por el propio sistema y se consideran seguras. Por ejemplo, a un gráfico que se haya generado y se embeba en la visualización, se le puede deshabilitar esta conversión añadiendo la palabra *safe*.

```
<div>{{variableSegura | safe}}</div>
```

Para evitar los ataques CSRF que se producen cuando un usuario hace una conexión segura con la aplicación web; más tarde es infectado al visitar otro servidor que referencia al servidor legítimo ; a continuación el cliente envía petición con código malicioso a la aplicacón web con la que ha establecido conexión segura (sin que el usuario se entere de lo ocurrido) y finalmente el atacante accede al servidor legítimo vía usuario. Se hace uso del *middleware* CSRF de Django.

En la misma figura 6.6, se puede observar {% csrf_token %}. Esto implica que la solicitud de esta página, debe ir acompañada de la variable *token*, que es un número aleatorio acordado en la conexión entre cliente y servidor lo suficientemente grande como para que el atacante no pueda generar una petición con un *token* válido y esta sea rechazada.

CAPÍTULO 7
Implantación

Una vez desarrollado el sistema es el momento de proceder a la implantación. En esta fase se deben describir los mecanismos necesarios para que el sistema esté operativo en el entorno en el que se quiera exportar para hacer uso del mismo.

Para llevar a cabo una correcta implantación del sistema se deben seguir los pasos que se detallan en la lista que se muestra a continuación.

1. Instale en su ordenador el entorno de desarrollo PyCharm. Este entorno de desarrollo es compatible con la creación de aplicaciones en Django que nos proporcionará un servidor web.

2. Instale el sistema de base de datos MongoDB tal como se indica en https://docs.mongodb.com/manual/installation/.

3. Cree una aplicación en Twitter y obtenga las credenciales de acceso tal como se indicó en la sección 6.1. También puede encontrar información sobre cómo crear una cuenta en Twitter en https://help.twitter.com/en/create-twitter-account.

4. Descargue el proyecto comprimido de https://github.com/dachirui/HealthAnalytics.

5. Al iniciar PyCharm vaya a *File* > *Open...* (ver figura 7.1) y en la ventana emergente que le aparecerá introduzca la ruta del proyecto descargado.

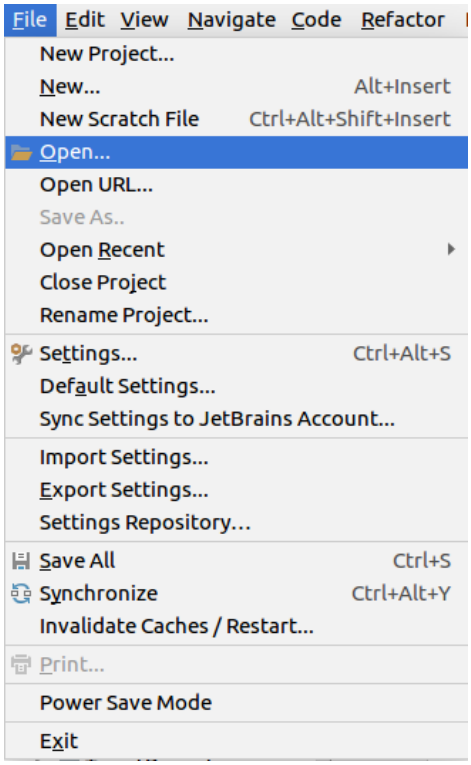

Figura 7.1: Opción Open

6. Una vez cargado el proyecto en PyCharm, vaya a *File* > *Settings...* y en la ventana emergente que le aparecerá escriba la palabra Django en el lugar que se indica en la imagen 7.2. Configure los campos tal como se muestran en la imagen.

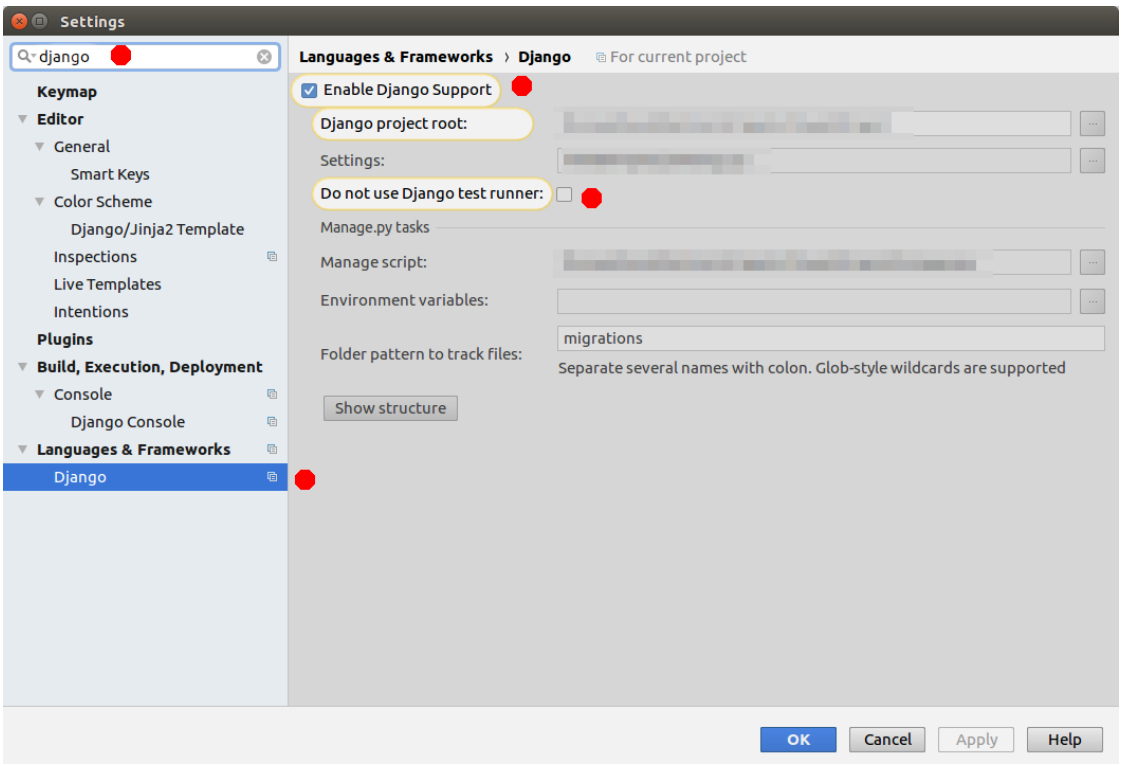

Figura 7.2: Configuración Django en PyCharm

7. A continuación, abra la consola de PyCharm (Alt+F12) para instalar los paquetes que se indican en el apéndice F. Para la instalación de los paquetes se utilizará el siguiente comando.

```
$ pip install <nombreDelPaquete>
```

8. En este punto se deberá arrancar el servicio que levantará la base de datos. Para arrancar el servicio deberá ejecutarse el siguiente comando.

```
$ sudo service mongod start
```

9. Antes de ejecutar el *script* poblarMongo.py, deberá indicar las credenciales de la aplicación de Twitter (que podrá obtener en https://apps.twitter.com/) en el código fuente del *script*, en las siguientes variables: *consumer_key*, *consumer_secret*, *access_token_key* y *access_token_secret*. El fragmento de código a editar se muestra en la figura 7.3.

```
api = twitter.Api(consumer_key='INSERTE_SU_CONSUMER_KEY',
    consumer_secret='INSERTE_SU_CONSUMER_SECRET',
    access_token_key='INSERTE_SU_ACCESS_TOKEN_KEY',
    access_token_secret='INSERTE_SU_ACCESS_TOKEN_SECRET')
```

Figura 7.3: Inserción de credenciales

Para poblar la base de datos ejecute el *script* poblarMongo.py que se encuentra en el proyecto que ha descargado. El *script* descargará tuits que contengan las palabras clave que se indiquen. Para ejecutar el *script* ejecute en consola la siguiente orden.

```
$ python poblarMongo.py [<palabrasClave>]
```

10. A continuación ejecute el proyecto en PyCharm clicando sobre el triángulo verde que se muestra en la figura 7.4

Figura 7.4: Arranque del servidor

11. Finalmente, tras la ejecución del paso 9 el servidor web estará listo para recibir peticiones y el sistema estará accesible. Para acceder al sistema ingrese la dirección http://127.0.0.1:8000/TweetApp/login en el navegador web y así acceder a la pantalla de inicio.

CAPÍTULO 8
Pruebas

Para evaluar que el sistema cumple con los requerimientos y que la solución que se ha desarrollado funciona correctamente, se han diseñado tres tipos de pruebas o análisis que se ejecutarán sobre el sistema de forma manual. Entre las pruebas realizadas, encontramos pruebas específicas para comprobar el comportamiento de cada módulo ante los casos de uso que se han detallado, también se procederá a realizar un análisis de usabilidad web y por último se detallará un caso real de análisis en el que se utilizará la aplicación desarrollada para analizar la información.

8.1 Pruebas modulares

Las pruebas modulares se centrarán en la satisfacción de los casos de uso. A lo largo de esta sección, se indicará la relación entre la prueba, los casos de uso que englobe y el resultado de la prueba.

Prueba 1

- CU03: Registrar cuenta de usuario en la plataforma.

- Prueba: Rellenar formulario de registro.

- Resultado: Los valores que se han introducido en los campos han sido almacenados en la base de datos.

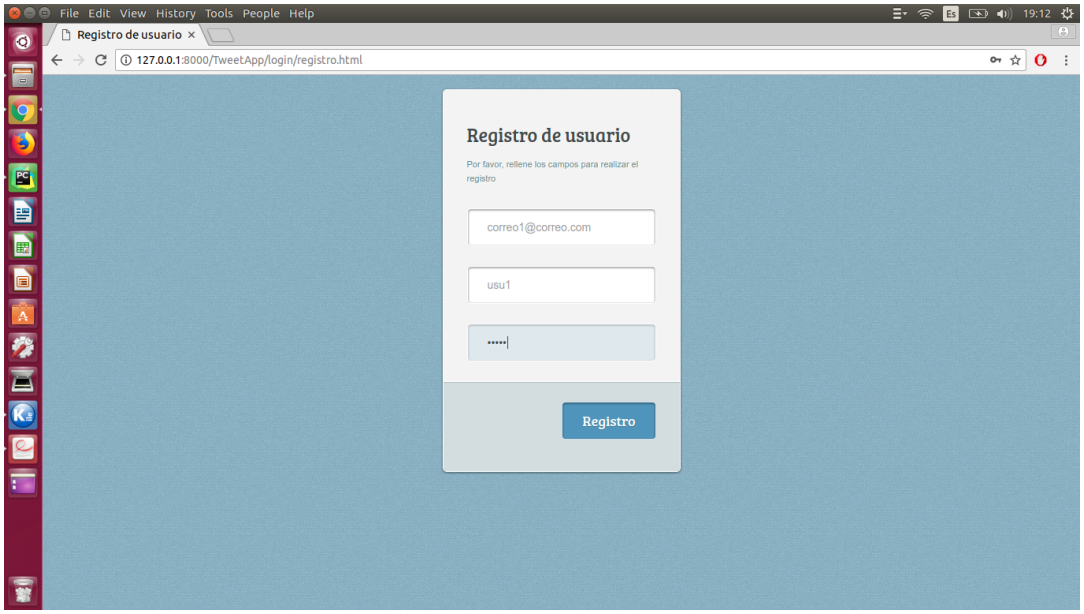

Figura 8.1: Registro del usuario

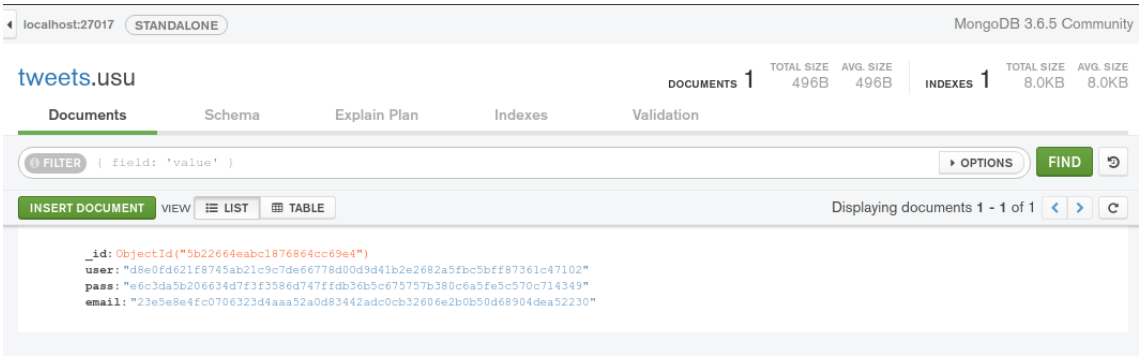

Figura 8.2: Campos cifrados del usuario registrado en la base de datos

En la figura 8.1 se muestra el formulario de registro de usuario desarrollado. Una vez completado y enviado el formulario, el sistema almacena en la base de datos los campos, tal como se muestra en la figura 8.2.

Prueba 2

- CU01: Iniciar sesión y CU02: Identificar y autentificar usuario.
- Prueba: Inicio sesión con usuario no registrado e inicio de sesión con usuario registrado.
- Resultado: Al intentar acceder con un usuario no registrado salta una alerta de error indicando que debe registrarse (figura 8.3). Al intentar acceder con un usuario registrado, el sistema redirige al usuario a la página de elección de parámetros (figura 8.4). El sistema interactúa con la base de datos para acceder a la información de usuario almacenada.

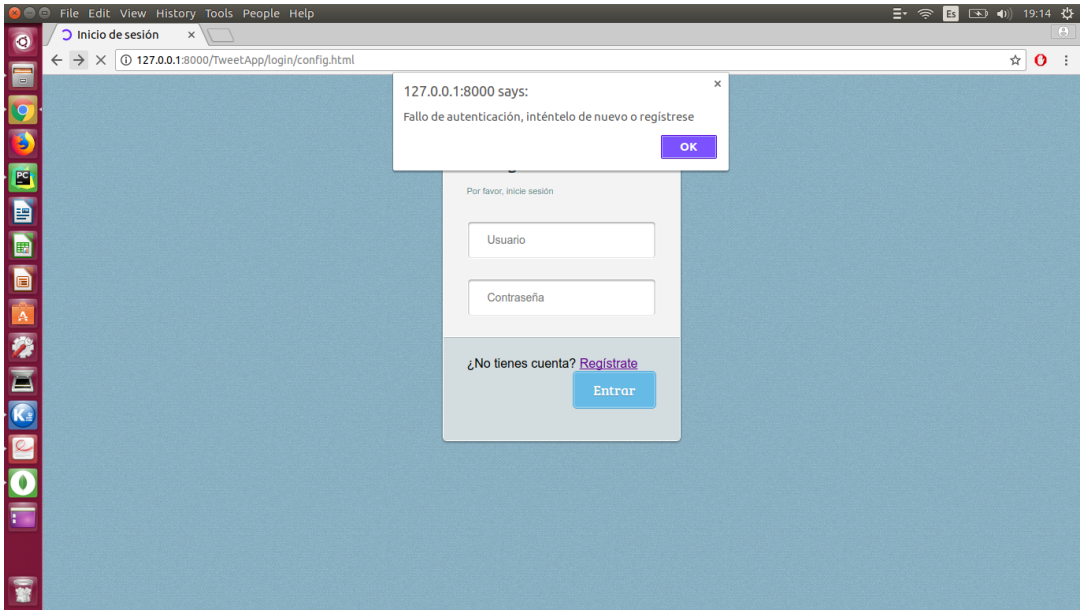

Figura 8.3: Error autenticación

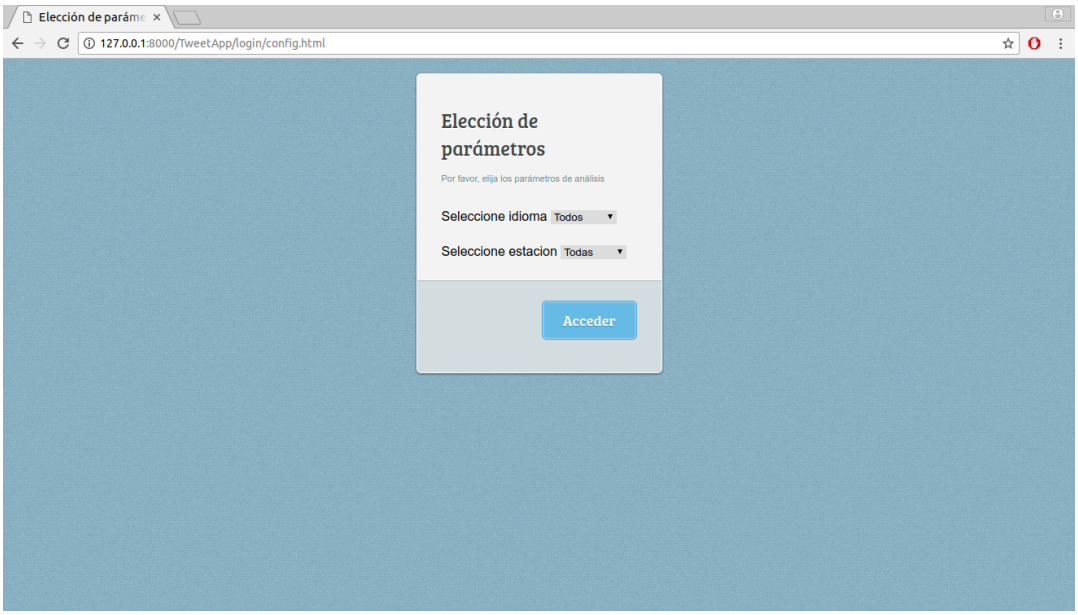

Figura 8.4: Página de elección de parámetros

Prueba 3

- CU05: Extracción de datos
- Prueba: Ejecución del *script* poblarMongo.py.
- Resultado: La base de datos ha sido poblada con los tuits que el *script* ha descargado de la API de Twitter.

Prueba 4

- CU07: Consultar análisis y CU08: Aplicar parámetros de análisis.
- Prueba: Se hace una prueba para cada una de las combinaciones de análisis de tuits (idioma,estación). (todos, todas); (español, invierno); (español, primavera); (español, todas); (inglés, invierno); (inglés, primavera); (inglés, todas);(portugués, invierno); (portugués, primavera); (portugués, todas).
- Resultado: Para todos los casos el usuario es redirigido al cuadro de mandos donde se muestran los gráficos en función de los parámetros elegidos (ver figura 8.5).

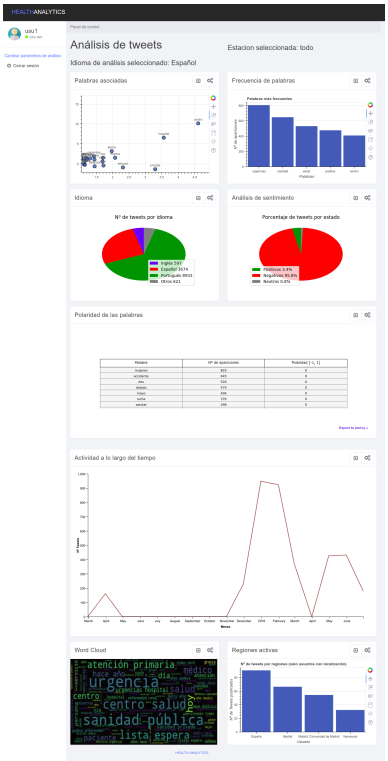

Figura 8.5: Vista general del cuadro de mandos

Prueba 5

- CU04: Descargar visualización del análisis.
- Prueba: Utilización del menú interactivo de las visualizaciones para descargar el gráfico.
- Resultado: El gráfico es descargado como una imagen en formato Portable Network Graphics (PNG).

Prueba 6

- CU06: Modificar parámetros de análisis.
- Prueba: Utilización de la opción 'Cambiar parámetros de análisis' que aparece en el cuadro de mandos.
- Resultado: El usuario es redirigido a la página de elección de parámetros donde puede establecer unos nuevos parámetros.

8.2 Informe de usabilidad web

Debido a que el sistema está compuesto por una web como herramienta principal de interacción con el usuario, es necesario que la web se ajuste a unos criterios de usabilidad para obtener una mejor experiencia de usuario. La usabilidad del sistema se considera un factor clave ya que el desarrollo de la herramienta resultaría en vano si finalmente esta no fuera útil debido a que no se han seguido unos criterios de usabilidad.

Para establecer estos criterios se partirá de los 10 principios heurísticos de la usabilidad que Jakob Nielsen enunció [29]. A continuación se mostrará una lista con cada uno de estos principios y para cada principio se indicarán los elementos de la web que satisfacen el mismo.

1. **Visibilidad del estado del sistema**

   Este principio se satisface en todas la interfaces de las pantallas de inicio de sesión o de registro, en las que se indica en la parte superior qué es lo que el sistema está haciendo en esa pantalla (inicio de sesión o registro de usuario). También se debe destacar que cuando se quiere registrar el usuario el sistema devuelve un mensaje de que el registro se ha realizado con éxito. Finalmente indicar que ante tiempos de espera elevados debido a la carga de datos y renderizado de las visualizaciones, el sistema alerta al usuario pidiendo que espere y el motivo de la espera.

2. **Relación entre el sistema y el mundo real**

   El idioma es el adecuado, las frases y el vocabulario es el idóneo para el usuario, la temática del contenido también se adecúa a la aplicación.

3. **Control y libertad del usuario**

El usuario tiene un control total sobre toda la página web, puede moverse libremente entre las pantallas. También se le da la opción al usuario de modificar algún campo ya rellenado en el formulario de registro e inicio de sesión. El conjunto de estos elementos proporciona al usuario una sensación de control sobre el sistema.

4. **Coherencia y estándares**

   La coherencia también es un aspecto clave en toda la página web. La disposición de las páginas es la misma en todas las pantallas. También la combinación de colores se mueve entre los colores azul y blanco durante el proceso de registro, inicio y visualización de resultados.

5. **Prevención de errores**

   En la ventana de elección de los parámetros de análisis se limitan los idiomas a los existentes, también ocurre en la elección de *hashtags*. El propósito es que el usuario no obtenga un error del sistema por algún error ortográfico o algún valor que no esté incluido en el registro del sistema.

6. **Reconocer frente a memorizar**

   Las visualizaciones interactivas contienen un menú de símbolos que son facilemente identificables. Por ejemplo, si se quiere guardar una visualización, la opción que ofrece esta acción se encuentra clicando en el icono del disco. En otro ejemplo se puede encontrar el caso de querer realizar zoom sobre el gráfico, para ello se ofrece el icono de la lupa que es fácilmente identificable.

7. **Flexibilidad y eficiencia de uso**

   La interfaz de usuario está preparada tanto para usuarios expertos como inexpertos mediante textos que indican la acción que el usuario debe realizar. Se persigue la secuencialidad en las acciones para que no resulte complejo el manejo.

8. **Estética y diseño minimalista**

El texto es casi inexistente porque la navegación es muy intuitiva. Se ha optado por una interfaz sencilla que cumpla las funciones deseadas.

9. **Ayudar a los usuarios a salir de los errores**

   Si el usuario intenta acceder a la web sin haberse registrado con anterioridad el sistema devuelve un mensaje indicando que el usuario no está registrado y que previamente al inicio de sesión debe registrarse.

10. **Ayuda y documentación**

    En las distintas pantallas se detallan los pasos a realizar para obtener los análisis que se deseen. También se dispone de una guía del usuario en el repositorio https://github.com/dachirui/HealthAnalytics

8.3 Caso real: análisis social del sistema sanitario

A modo de prueba, se propone un caso real para hacer uso de la herramienta y realizar un análisis social del sistema sanitario.

Como caso de estudio se pretende realizar un análisis social del sistema sanitario en España desde junio de 2017 hasta junio de 2018. En la base de datos se encuentran almacenados un total de 18 741 tuits que contienen alguno de los siguientes *hashtags* (el signo '+' significa que tienen que aparecer las dos palabras en el tuit): SanidadPublica, SanidadPrivada, *Sanitat*, *saúde*, lista+de+espera, centro+de+salud, *Centre+de+Salut*, atención+primaria, *atenció+primària*, urgencias, *urgències*, *urxencias*, *Llista+d'espera*.

Tras la puesta en marcha de la herramienta, se indica el idioma (todos) y las estaciones a analizar (todas). Obtenemos el primer acceso al cuadro de mandos con las visualizaciones de todo el conjunto de tuits. En este cuadro de mandos podemos fijarnos en que de los 18 741 tuits, 12 433 son en portugués, 4811 son en español, 610 son en inglés y los 860 restantes no tienen un idioma definido (ver figura 8.6).

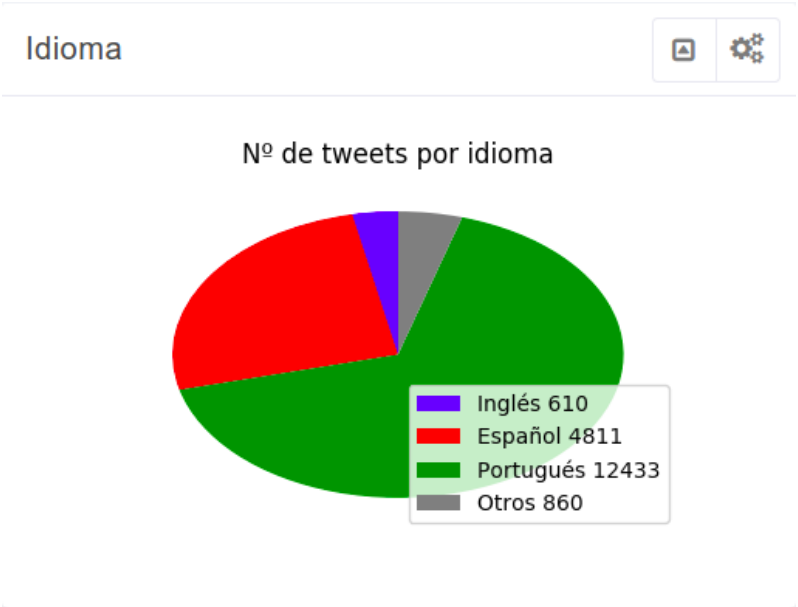

Figura 8.6: Recuento de tuits por idioma

Para focalizar el análisis, debido al distinto funcionamiento de los diferentes sistemas sanitarios de los que se pueda hablar en los tuits analizados, nos centraremos en el análisis de los tuits en Español. A partir de este punto en el documento, el número de ocurrencias de las palabras se indicará entre paréntesis.

Cambiando los parámetros de análisis e indicando ahora el idioma (español) y las estaciones (todas), en el cuadro de mandos generado se puede apreciar en la gráfica de actividad de tuits en el tiempo, unos picos de actividad entre otoño e invierno y en verano. El pico de actividad más alto se recoge en invierno (ver figura 8.7).

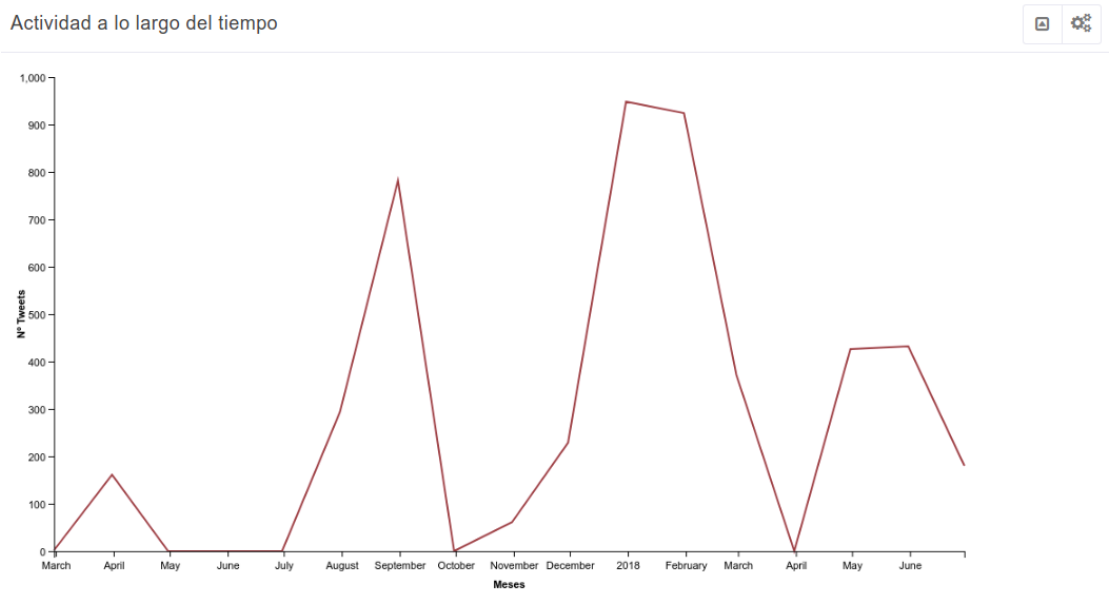

Figura 8.7: Nº de tuits en cada mes de marzo de 2017 a junio de 2018

También llama la atención, como entre las palabras que se suelen asociar, encontramos la atención primaria y los centros de salud ligados, pero sin embargo cuando se habla de sanidad pública, suele ir acompañado de las palabras 'hospital', 'pacientes', etc., tal como se muestra en la figura 8.8.

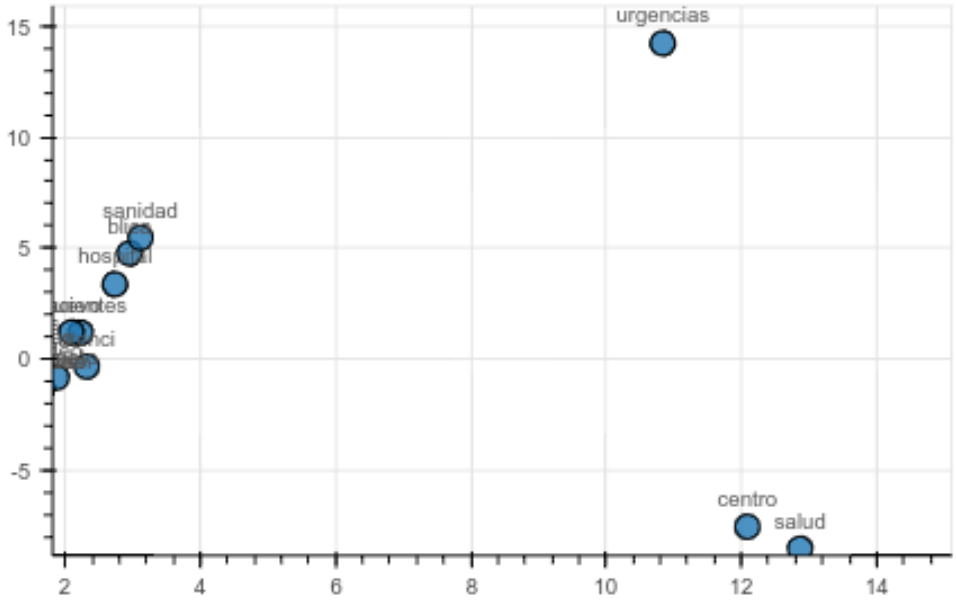

Figura 8.8: Asociación de palabras de marzo de 2017 a junio de 2018

Para realizar un análisis más exahustivo de los picos de actividad, se va a proceder a realizar un estudio de los tuits en español para cada estación.

Verano de 2017

En el periodo estival se han recogido 1166 tuits en español. Como hemos dicho, en verano se observaba un pico de actividad, menor al de invierno, pero era notable. Entre el análisis de palabras frecuentes se pueden encontrar las palabras 'inmigrante' e 'inmigración'. No es casual la aparición de estos términos, ya que dada la posición geográfica de la península y el buen estado que presenta el mar en los meses estivales, los movimientos migratorios entre costas de África y Europa se acentúan, provocando esto un aumento de actividad en las redes sociales en ese tema. Sin embargo, mirando la nube de palabras (ver figura 8.10) que se muestra para esta estación, llama la atención la aparición de las palabras 'grupos armados'. Este término puede relacionarse con la gráfica de regiones activas donde aparece Chile en tercer lugar (ver figura 8.9). Teniendo en cuenta el factor verano, el país Chile y la aparición de grupos armados, es muy probable que se esté indicando actividad referente a la Resistencia Ancestral Mapuche, una presunta organización guerrillera Chilena que tuvo enfrentamientos en Argentina en agosto de 2017, parte de la actividad registrada en la herramienta proviene de los tuits generados que referenciaban los ataques [30].

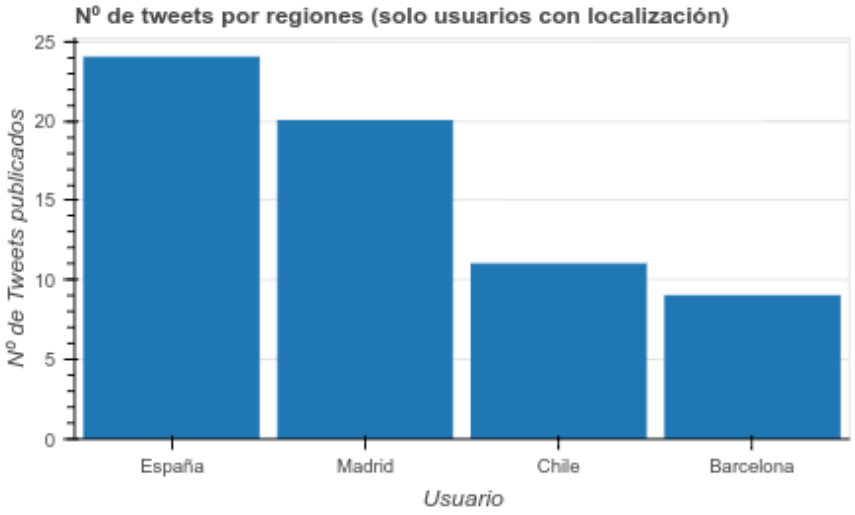

Figura 8.9: Chile en el tercer puesto de regiones activas en verano

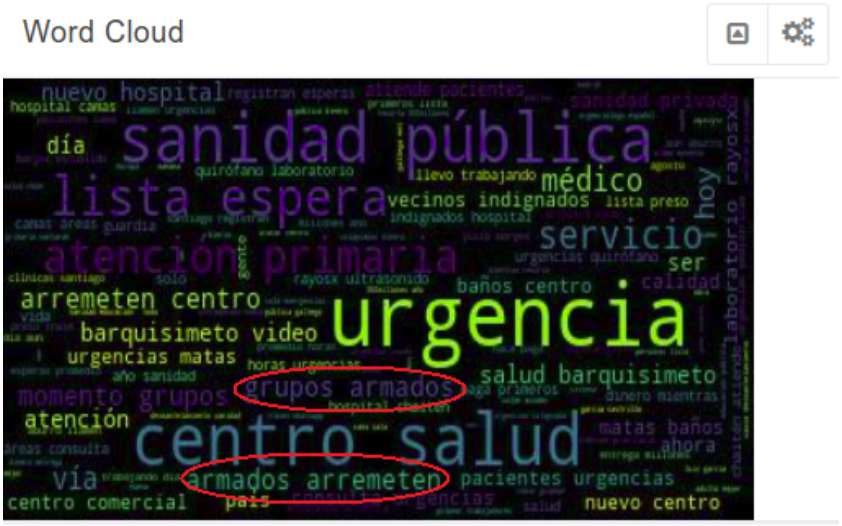

Figura 8.10: Word Cloud de la estación estival

Entre los términos con polaridad positiva, encontramos las palabras 'familiar' y 'social'(ver figuras 8.11 y 8.12 respectivamente).

Figura 8.11: Palabra, frecuencia y polaridad de la palabra 'familiar'

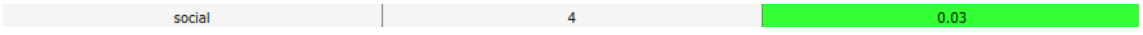

Figura 8.12: Palabra, frecuencia y polaridad de la palabra 'social'

Como se ha comentado en la sección 6.2.4, el número de palabras de las que se puede extraer la polaridad cuando el idioma es distinto al inglés es escaso. Es por esto por lo que en este análisis de los tuits en español se muestran únicamente dos palabras para que se aprecie a modo de ejemplo el funcionamiento de la clasificación polar.

Otoño de 2017

Durante el otoño se han recogido 774 tuits relativos a la sanidad en español. Como se apreciaba en el gráfico de actividad anual (figura 8.7), otoño era una estación con menor actividad que verano e invierno. Sin embargo es un precedente de un pico, por lo que se debe analizar en profundidad.

Entre las palabras recurrentes que aparecen en los tuits, se puede encontrar 'sanidad privada', 'sanidad pública', y 'listas de espera' (ver figura 8.13).

Figura 8.13: Word Cloud de otoño

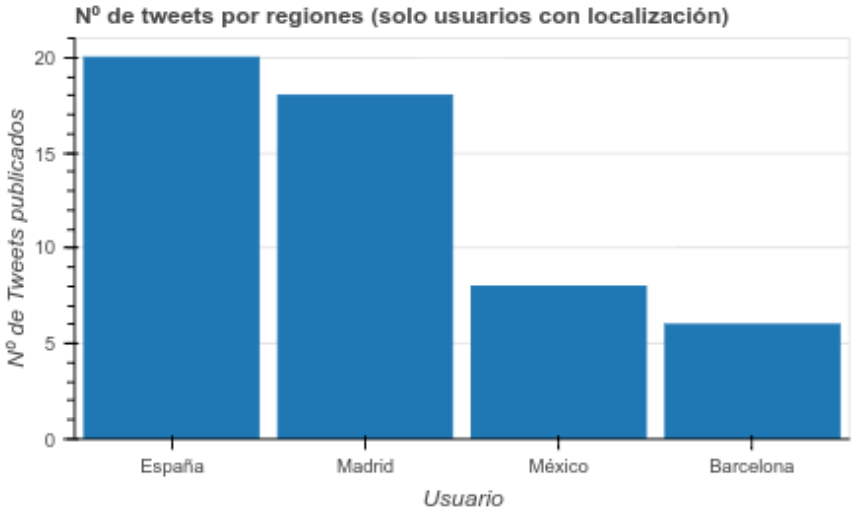

Figura 8.14: Regiones activas en otoño

Mirando las localizaciones frecuentes (8.14), Chile ya no aparece en esta época del año y la mayoría de tuits provienen de España, y más concretamente muchos de los usuarios que han tuiteado indican su localización en la Comunidad de Madrid. No es de extrañar que las palabras recurrentes sean las que se han indicado, ya que el otoño de 2017 fue un periodo de protestas en el ámbito sanitario, las denominadas 'mareas blancas' recorrían las ciudades a lo largo y ancho del país y con mucho impacto en la Comunidad de Madrid. En adición, se produjo una

huelga continuada de sanidad privada en el mismo intervalo de tiempo y el 18 de octubre se publicó que en Madrid se había alcanzado el récord con el tiempo más largo de espera en la listas de espera de sanidad [31].

Invierno de 2017/2018

En los meses de invierno, se analizan un total de 2473 tuits en español. Las palabras recurrentes siguen siendo muy parecidas a las del otoño, además hay que añadir la palabra 'colapso'(618). Con este periodo, ya son dos los periodos temporales consecutivos donde las palabras 'protestas' y 'sanidad pública' van ligados. Las protestas continúan y se desata el pico de actividad de tuits. Este pico es muy probable que sea derivado a que los días 13 y 14 de diciembre se vivió en la Comunidad de Madrid una huelga del personal sanitario (ver figura 8.15).

La figura 8.7 muestra el aumento de actividad entre otoño e invierno y se muestra cómo se estaba tuiteando cada vez más sobre la sanidad, en gran parte debido a las protestas que se sucedían cada semana.

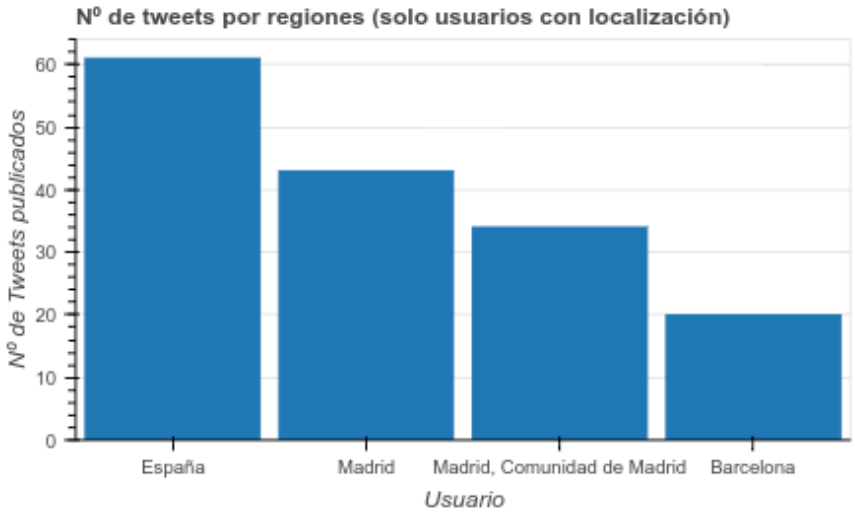

Figura 8.15: Madrid como núcleo de la generación de actividad durante el invierno

Primavera de 2018

En el periodo primaveral se han recogido 1038 tuits. Entre las palabras frecuentes (ver figura 8.16) encontramos 'enfermera'(127), la aparición de este término es muy posible que se deba a que el 12 de mayo fue el día internacional de la enfermería. También se puede observar que 'empleo' aparece 75 veces muy probablemente debido a que el 4 de junio de 2018 fue publicado el dato de que el empleo en el ámbito sanitario en España había aumentado un 4 %.

Palabra	Nº de apariciones	Polaridad [-1, 1]
hospitalario	191	0
comunidad	179	0
seguro	178	0
mejorar	150	0
enfermera	127	0
pudieron	97	0
empleo	75	0

Figura 8.16: Palabras frecuentes y polaridad de las palabras en primavera

En la figura 8.17 se puede observar el realce de dos grupos temáticos. El primero de ellos se corresponde con las palabras 'hospital', 'urgencias' y 'hoy', por lo que estos términos tienden a aparecer en las mismas frases y hacen referencia a tuits opinando en relación al servicio ofrecido cuando un paciente acude a urgencias. En contraposición tenemos el otro grupo que va más encaminado hacia la sanidad desde el punto de vista político, ya que aparecen palabras como 'universal', 'espera', 'nuevo', 'gobierno', 'sociales', 'calidad', 'defensa', 'mejor', etc. Estas palabras van en la línea de la renovación del sistema sanitario por parte del nuevo gobierno que entró en junio de 2018.

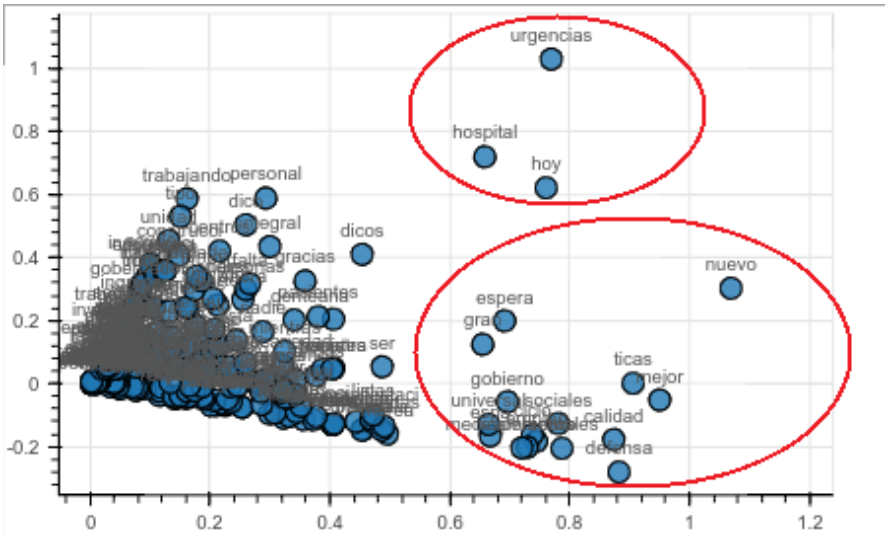

Figura 8.17: Asociación entre palabras en primavera

Por último, para tratar un poco más en profundidad la parte de análisis de sentimientos, se procederá a realizar un análisis de los tuits en inglés que se tienen almacenados.

De los 610 tuits en inglés almacenados, en la figura 8.18 se muestra el porcentaje de tuits en los que se ha clasificado.

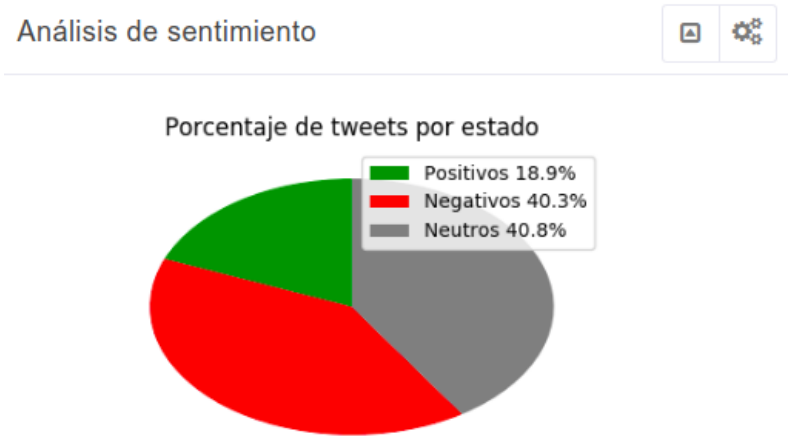

Figura 8.18: Porcentaje de tuits positivos, negativos y neutros en inglés

Palabra	Nº de apariciones	Polaridad [-1, 1]
thanks	18	0.2
fit	11	0.4
right	8	0.29
poor	7	-0.4
dead	7	-0.2
safe	3	0.5
sick	2	-0.71

Tabla 8.1: Tabla de polaridades de las palabras de los tuits en inglés

Figura 8.19: A la izquierda nube de palabras de tuits en inglés, a la derecha localizaciones más activas de tuits en inglés

Las palabras negativas que se pueden leer en la nube de palabras (ver figura 8.19) *'military'*, *'isis'*, *'fake doctors'*, *'attack'*, provienen de tuits referidos a un atentado en un hospital militar de Kabul en el que unos terroristas del Estado Islámico se hicieron pasar por médicos para llevar a cabo el ataque el 8 de marzo de 2017 [32].

Con este estudio se concluye la prueba de aplicación real del funcionamiento de la herramienta. El uso de la herramienta nos ha proporcionado las visuali-

zaciones, la información y los datos extraídos de los tuits para poder realizar la radiografía del sistema sanitario. Para lograr un mayor rendimiento de la herramienta se recomienda hacer uso de un buscador de noticias con intervalo de tiempo personalizable para buscar acontecimientos en base a las palabras recurrentes que aparezcan y la fecha del periodo analizado.

CAPÍTULO 9
Conclusiones

En este proyecto se ha desarrollado una herramienta para la extracción, análisis y visualización de la actividad en la red social Twitter relacionada con el ámbito sanitario.

Respecto a los objetivos y subobjetivos formulados en la sección 1.2, se puede afirmar que estos se han alcanzado de manera satisfactoria. Durante el desarrollo del TFG se han ido cumpliendo los subobjetivos indicados para llevar a cabo el desarrollo del sistema.

El primero de los subobjetivos alcanzados es el de extracción, limpieza y almacenamiento de los tuits. Para el cumplimiento del mismo se ha hecho uso de la API de Twitter para la extracción, uso de expresiones regulares en Python y las librerías Numpy, Nltk, Pandas, Scikit-learn y Stopwords para la limpieza y el tratamiento de la información. Por último la integración del sistema con MongoDB para el almacenamiento de la información. Los subobjetivos de realizar tratamiento y análisis de la información relevante en el contexto sanitario y del desarrollo de métricas e indicadores para establecer el estado del sistema sanitario, han quedado superados al realizar un análisis del objeto tuit y establecer visualizaciones y clasificaciones en base a los campos del objeto. La presentación de los datos de una manera intuitiva y visual, se ha cumplido gracias al uso de las librerías Matplotlib, Plotly, Bokeh y WordCloud que han aportado el entorno para generar las visualizaciones en base a la información previamente tratada.

Finalmente, el desarrollo de una plataforma web que integre el proceso de análisis y su visualización ha resultado satisfecho gracias a que se han seguido las etapas en las que se estructura el TFG para conseguir el objetivo. Primero, se ha realizado: (**i**) análisis para la evaluación de las distintas opciones que hay para la resolución del problema, especificación formal de requisitos siguiendo el estandar IEEE 830; (**ii**) diseño basado en el análisis; (**iii**) requisitos y usabilidad web haciendo uso de diagramas UML. Después, el desarrollo se llevó a cabo haciendo uso de Django como *framework*, HTML5, CSS y JavaScript para el desarrollo de la interfaz web. Por último, se realizaron pruebas para el análisis del sistema desde el punto de vista de la usabilidad y se desarrolló un caso real de uso del sistema.

La combinación de distintas áreas de conocimiento, unido el uso de múltiples y diversas tecnologías, han hecho de este proyecto, el trabajo más completo que he podido realizar hasta la fecha.

Entre las dificultades que se han encontrado, la primera de ellas ha sido el tener que aprender un nuevo lenguaje de programación. El uso de Python y Django ha requerido de un tiempo de aprendizaje antes de ponerse a trabajar directamente en la aplicación en cuestión. Tras el periodo de aprendizaje se ha adquirido la soltura suficiente como para poder llevar a cabo el desarrollo. La siguiente dificultad a destacar es el acceso a los tuits. En este proyecto se ha hecho uso de la API de Twitter gratuita que establecía limites sobre el número de tuits que se pueden descargar, una vez llegado al límite la API no está operativa durante 15 minutos. Para el tratamiento de este caso se desarrollaron excepciones para que el sistema de recolección de datos esperase el tiempo establecido. La tercera dificultad que debe destacarse es el volumen de información con el que se trabaja. En el caso de estudio se utilizaron 20 000 tuits que deben analizarse y generar distintas visualizaciones en base a estos. Para un único ordenador, ha supuesto una sobrecarga continua a la hora de computar, por lo que no se ha podido hacer uso de todo el conjunto de información por colapsos. La cuarta dificultad reside en el balanceo entre la visualización a mostrar y la protección de datos, ya que debido al uso del objeto usuario del tuit se han tenido que desarrollar visualizaciones que no permitiesen la identificación de los usuarios que hayan generado los tuits. Finalmente, destacar que en cuanto a tratamiento del lenguaje, el principal problema a la hora de encontrar módulos es el idioma. La mayoría de módulos que permiten el tratamiento de lenguaje están orientados al inglés, por lo que supone un a limitación a la hora de realizar dicho análisis.

Para concluir, con este TFG se ha aprendido a trabajar con el *framework* Django. Se han aprendido otros lenguajes de programación como Python. Se ha podido conocer el funcionamiento de sistemas de bases de datos no relacionales que no han sido tratados durante los estudios (MongoDB). Se ha podido analizar en profundidad los formatos que se utilizan en la transmisión de información en la red (JSON, XML, CSV). Se ha trabajado con lenguajes para el desarrollo del *front end*: HTML5 y JavaScript. Se ha podido desarrollar un proyecto completo con un inicio y un fin, un esfuerzo temporal que ha sido planificado, gestionado y evaluado en cada una de las fases que lo han compuesto. Se ha podido descubrir cómo aplicar los conocimientos adquiridos a lo largo del grado en trabajos y estudios que bien podrían formar parte del ámbito profesional. Por último, se ha descubierto una pequeña parte de la investigación que se desarrolla en el ámbito de la ingeniería informática gracias a la lectura de artículos publicados y aplicaciones desarrolladas por grupos de investigación.

9.1 Relación del trabajo desarrollado con los estudios cursados

Para la realización del proyecto ha sido necesaria la combinación de los conocimientos adquiridos en distintas áreas que se han tratado a lo largo del Grado en Ingeniería Informática. Se han requerido conocimientos de programación, conocimientos estadísticos, funcionamiento de protocolos de red a nivel de aplicación, *Hypertext Transfer Protocol* (HTTP), conocimientos sobre seguridad informática, lectura de legislación, conocimientos sobre desarrollo de interfaces, conocimien-

tos sobre bases de datos, ingeniería del software, gestión de proyectos, tratamiento de datos, tratamiento de lenguaje, desarrollo web e integración de aplicaciones.

En conclusión, para la realización de este proyecto ha sido necesario el conocimiento adquirido en la mayoría de las asignaturas cursadas. Por esto queda probado que el trabajo ha sido desarrollado conforme a los estudios cursados. Los conocimientos adquiridos cobran valor cuando son extraídos del ámbito de una asignatura específica y pueden ser utilizados junto a otros conocimientos distintos para formar una solución, un producto o un desarrollo con el mismo objetivo, el de proporcionar una solución real a un problema real. Con este desarrollo se ha conseguido proporcionar la solución, proporcionando un análisis de un área alejada de la informática, unida por el uso de las tecnologías y que responde a preguntas sobre el sistema sanitario.

Por último, para finalizar esta sección se indican algunas de las competencias transversales que se han ido evaluando a lo largo del grado y que se considera que han ido apareciendo a lo largo del desarrollo del TFG.

- CT_01. Comprensión e integración: la integración de distintas áreas de conocimiento ha hecho posible la realización del TFG.
- CT_03. Análisis y resolución de problemas: en el TFG se han analizado tecnologías y formatos como opciones ante la resolución de problemas.
- CT_05. Diseño y proyecto: el seguimiento de una metodología proporciona las claves del buen diseño y consecución de un proyecto.
- CT_07. Responsabilidad ética, medioambiental y profesional: uso de información generada por usuarios de Twitter y almacenamiento de credenciales de los usuarios que se registren en la web que se ha desarrollado. La responsabilidad del uso, tratamiento y custodia de los datos recae sobre el desarrollador.
- CT_10. Conocimiento de problemas contemporáneos: con el análisis social realizado se ha adquirido un conocimiento de los problemas contemporáneos en el ámbito sanitario.
- CT_11. Aprendizaje permanente: se ha requerido aprender nuevos conceptos y tecnologías para la realización del TFG.
- CT_12. Planificación y gestión del tiempo: ha sido necesario necesaria una gestión del tiempo para poder finalizar el trabajo a tiempo.

CAPÍTULO 10
Trabajos futuros

Para concluir, en esta sección se van a presentar las mejoras que se podrían desarrollar a raíz del proyecto para lograr una mejora del sistema y algunas líneas de investigación en las que se puede hacer uso del sistema.

En primer lugar se podría trabajar en el desarrollo de un mapa de calor que represente los sentimientos de los tuits en función de la región geográfica. Para ello sería necesario el uso de tuits geolocalizados, que combinados con las localizaciones de los usuarios permitirían el desarrollo del mapa. En segundo lugar, para que el desarrollo del mapa fuera satisfactorio, sería necesario desarrollar un analizador de sentimientos propio para otros idiomas distintos al inglés. Con este analizador, el mapa podría extrapolarse a nivel mundial. Otro aspecto que en el TFG no se ha podido abordar, ha sido el tema de eficiencia en la computación. Trabajar en el uso de técnicas de computación paralela para lograr una disminución de los tiempos de cómputo tendría repercusiones positivas para el sistema.

En cuanto a líneas de investigación, la mejora de indicadores y el desarrollo de nuevas visualizaciones es un campo extenso del que se podrían realizar diferentes estudios que mejorasen las técnicas empleadas, así como hacer uso de las cuentas que siguen los usuario que tuitean sobre sanidad para lograr una radiografía del público y diferenciar distintos perfiles (pacientes, profesionales del sector sanitario, periodistas, etc.).

Finalmente, debido al desarrollo modular del sistema, resulta ser un sistema fácilmente modificable que podría ser utilizado para el análisis de tuits en distintas áreas sin realizar un complejo cambio estructural. Las áreas que podrían abordarse con el sistema podrían ser educación, política, consumo, deportes y sociedad entre otras muchas áreas. Esta herramienta desarrollada podría ser utilizada en un proyecto que aunara distintas herramientas de análisis social sobre distintas redes sociales.

Bibliografía

[1] A. Lenhart, K. Purcell, A. Smith, and K. Zickuhr, "Social media & mobile internet use among teens and young adults. millennials.," *Pew internet & American life project*, 2010.

[2] L. Garcés and L. M. Egas, "Evolución de las metodologías de desarrollo de la ingeniería de software en el proceso la ingeniería de sistemas software.," *Revista Científica y Tecnológica UPSE*, vol. 1, no. 3, 2015.

[3] S. of Data, "Methodology." https://schoolofdata.org/methodology/, 2016 (accedido 15 mayo, 2018).

[4] A. Vegas, "2016: Dos mil millones de usuarios de redes sociales." http://www.andresvegas.es/2011/12/social-media/2016-dos-mil-millones-de-usuarios-de-redes-sociales, 2011 (accedido 8 mayo, 2018).

[5] Ma. Inés Fernández, "Importancia de la duración del ciclo de vida de un tuit." http://codendigital.com/importancia-de-la-duracion-del-ciclo-de-vida-de-un-tuit/, 2014 (accedido 8 mayo, 2018).

[6] M. L. C. Martínez and P. Aragón, "Twitter, del sondeo a la sonda: nuevos canales de opinión, nuevos métodos de análisis," *Más poder local*, no. 12, pp. 50–56, 2012.

[7] T. Inc., "Panel de actividad de tweets." https://business.twitter.com/es/help/campaign-measurement-and-analytics/tweet-activity-dashboard.html, 2016 (accedido 10 mayo, 2018).

[8] H. Saif, Y. He, and H. Alani, "Semantic sentiment analysis of twitter," in *International semantic web conference*, pp. 508–524, Springer, 2012.

[9] D. Chichell, "Caracterización ideológica de los seguidores de Twitter de la cuenta del Ayuntamiento de Valencia." https://github.com/dachirui/AAPP-Tweet, 2018 (accedido 10 mayo, 2018).

[10] S. N. de Salud., "Barómetro sanitario 2017." https://www.msssi.gob.es/estadEstudios/estadisticas/BarometroSanitario/Barom_Sanit_2017/RESUMENGRAFICO_BS2017.pdf, 2018 (accedido 11 mayo, 2018).

[11] F. Greaves, D. Ramirez-Cano, C. Millett, and L. Darzi, Ara Donaldson, "Harnessing the cloud of patient experience: using social media to detect poor quality healthcare," *BMJ Qual Saf*, vol. 22, no. 3, pp. 251–255, 2013.

[12] E. del Val, J. Palanca, and M. Rebollo, "U-tool: A urban-toolkit for enhancing city maps through citizens' activity," in *Advances in Practical Applications of Scalable Multi-agent Systems. The PAAMS Collection*, pp. 243–246, Springer, 2016.

[13] E. Vivanco, J. Palanca, E. del Val, M. Rebollo, and V. Botti, "Using geo-tagged sentiment to better understand social interactions," in *International Conference on Practical Applications of Agents and Multi-Agent Systems*, pp. 369–372, Springer, 2017.

[14] J. M. Martínez, "Desarrollo de una aplicación para el análisis de los sentimientos de los alumnos durante una clase," 2017.

[15] A. Yatsyk, "Medición de la actividad de los grupos de facebook para la evaluación del aprendizaje de los alumnos," 2017.

[16] Kit Smith, "44 estadísticas de Twitter para 2016." https://www.brandwatch.com/es/blog/44-estadisticas-twitter-2016/, 2016 (accedido 18 mayo, 2018).

[17] S. ECMA-404, "The JSON data interchange format," 2013.

[18] Y. Shafranovich, "Common format and mime type for comma-separated values (csv) files," 2005.

[19] T. Bray, J. Paoli, C. Sperberg-McQueen, Y. Mailer, and F. Yergeau, "Extensible markup language (xml) 1.0 5th edition, w3c recommendation, november 2008."

[20] E. Rowell, "CSV vs XML vs JSON – Which is the Best Response Data Format?." https://applerepairstation.co.uk/csv-vs-xml-vs-json-which-is-the-best-response-data-format/, 2017 (accedido 19 mayo, 2018).

[21] D. O. d. l. C. Europeas, "Directiva 95/46/CE del Parlamento Europeo y del Consejo de 24 de octubre de 1995 relativa a la protección de las personas físicas en lo que respecta al tratamiento de datos personales ya la libre circulación de estos datos," 1997.

[22] D. O. d. l. C. Europeas, "Directiva 2002/58/CE del Parlamento Europeo y del Consejo, de 12 de julio, relativa al tratamiento de los datos personales y a la protección de la intimidad en el sector de las comunicaciones electrónicas (Directiva sobre la privacidad y las comunicaciones electrónicas)," 2002.

[23] B. O. del Estado, "Ley orgánica 15/1999, de 13 de diciembre," 1999.

[24] Y. D. y Yenisleidy Fernández, "Patrón modelo-vista-controlador.," *Revista Telem@ tica*, vol. 11, no. 1, pp. 47–57, 2012.

[25] T. Inc., "Search Tweets." https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets, 2016 (accedido 18 mayo, 2018).

[26] E. Rowell, "¿qué es un framework web?." http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf, 2014 (accedido 24 mayo, 2018).

[27] M. Poczwardowsk, "Ruby on Rails vs. Django: Two Great Web Development Frameworks." https://www.netguru.co/blog/ruby-on-rails-vs-django-two-great-web-development-frameworks, 2017 (accedido 25 mayo, 2018).

[28] P. Balasubramani, "MongoDB Structure." https://www.quora.com/What-is-the-MongoDB-architecture, 2016 (accedido 25 mayo, 2018).

[29] J. Nielsen, *Designing web usability: The practice of simplicity*. New Riders Publishing, 1999.

[30] N. Niebieskikwiat, "La justicia ya registró más de 70 actos violentos de la resistencia ancestral mapuche," *Clarín*.

[31] M. González, "La lista de espera para ver al especialista en madrid alcanza su récord." https://www.elboletin.com/noticia/154760/nacional/la-lista-de-espera-para-ver-al-especialista-en-madrid-alcanza-su-record.html, 2017 (accedido 23 junio, 2018).

[32] M. Safi, "Isis militants disguised as doctors kill 38 in kabul hospital attack." https://www.theguardian.com/world/2017/mar/08/gunmen-dressed-as-doctors-attack-military-hospital-in-kabul, 2017 (accedido 23 junio, 2018).

APÉNDICE A
Código fuente - función checkPassUser

```
def checkPassUser(username,password):
    username = hashlib.sha256(username.encode('utf-8')).hexdigest()
    password = hashlib.sha256(password.encode('utf-8')).hexdigest()
    connection = pymongo.MongoClient("mongodb://localhost:27017")
    db = connection.tweets
    record = db.usu
    try:
        userExists = False
        if record.find({"user":username,"pass":password}).count() == 1:
            userExists = True

            return userExists
    except Exception as e:
        return userExists
```

APÉNDICE B
Código fuente - función graficos

```
def graficos(request):
        a = IDIOMA
        b = ESTACION
        if (b=="primavera"):
            tweets = models.searchTweetPrimavera()
        elif (b == "invierno"):
            tweets = models.searchTweetInvierno()
        elif (b == "verano"):
            tweets = models.searchTweetVerano()
        elif (b == "otono"):
            tweets = models.searchTweetOtono()
        else:
            tweets = models.searchTweet()

        idiom = clean = tweets

        if (a == "all"):


            texts, image2 = clean_tweet(clean)
            textUser = obtain_user(a,b)
            script1, div1 = lda(texts)
            script2, div2 = word_frequency(texts)
            script3, div3 = user_frequency(textUser)
            image3 = wordcloud(texts)
            image = sentimental2(texts,a)
            graphLine(b,"")
            imagePieSentimental = pieSentimental(texts)

            contexto = {'script3': script3,'div3':div3,'idioma': "Todos",
            'username': USER, 'script1': script1, 'div1': div1, 'script2': script2,
                        'div2': div2, 'image2': image2,
                        'image': image, 'image3': image3,'estacion':b,
                        'pieSentimental':imagePieSentimental}

        elif (a == "en"):
```

```
        texts, image2 = clean_tweet_languaje(clean,a)
        textUser = obtain_user(a, b)
        script1, div1 = lda(texts)
        script2, div2 = word_frequency(texts)
        script3, div3 = user_frequency(textUser)
        image3 = wordcloud(texts)
        image = sentimental2(texts,a)
        graphLineIdioma(a,b)
        imagePieSentimental = pieSentimental(texts)

        contexto = {'script3': script3,'div3':div3,'idioma': "Inglés",
        'username': USER, 'script1': script1, 'div1': div1, 'script2': script2,
                    'div2': div2, 'image2': image2,
                    'image': image, 'image3': image3,'estacion':b,
                    'pieSentimental':imagePieSentimental}


    elif (a == "es"):
        texts, image2 = clean_tweet_languaje(clean,a)
        textUser = obtain_user(a, b)
        script1, div1 = lda(texts)
        script2, div2 = word_frequency(texts)
        script3, div3 = user_frequency(textUser)
        image3 = wordcloud(texts)
        image = sentimental2(texts,a)
        graphLineIdioma(a,b)
        imagePieSentimental = pieSentimental(texts)

        contexto = {'script3': script3,'div3':div3,'idioma': "Español",
        'username': USER, 'script1': script1, 'div1': div1, 'script2': script2,
                    'div2': div2, 'image2': image2,
                    'image': image, 'image3': image3,'estacion':b,
                    'pieSentimental':imagePieSentimental}


    elif (a == "pt"):
        texts, image2 = clean_tweet_languaje(clean,a)
        textUser = obtain_user(a, b)
        script1, div1 = lda(texts)
        script2, div2 = word_frequency(texts)
        script3, div3 = user_frequency(textUser)
        image3 = wordcloud(texts)
        image = sentimental2(texts,a)
        graphLineIdioma(a,b)
        imagePieSentimental = pieSentimental(texts)

        contexto = {'script3': script3,'div3':div3,'idioma': "Portugués",
        'username': USER, 'script1': script1, 'div1': div1, 'script2': script2,
                    'div2': div2, 'image2': image2,
```

```
                    'image': image, 'image3': image3,'estacion':b,
                    'pieSentimental':imagePieSentimental}

    return render(request, 'graph.html', contexto)
```

APÉNDICE C
Código fuente - index.html

```
 {% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Inicio de sesión</title>
    <script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
    <script src="http://d3js.org/topojson.v1.min.js"></script>
    <script src="http://d3js.org/d3.geo.projection.v0.min.js" charset="utf-8">
    </script>
    <script src="http://trifacta.github.com/vega/vega.js"></script>
    <link href="{% static 'css/style.css' %}" rel="stylesheet">
    {% if error %}
        <script>
            alert('Fallo de autenticación, inténtelo de nuevo o regístrese');
        </script>
    {% endif %}
</head>
<body>
    <br>

  <form name="login-form" class="login-form" action="config.html" method="POST">
        {% csrf_token %}
<div class="header">
<h1>Bienvenido a Health Analytics</h1>
<span>Por favor, inicie sesión</span>
</div>

<div class="content">
<input name="username" type="text" class="input username" placeholder="Usuario" />
<div class="user-icon"></div>
<input name="password" type="password" class="input password"
placeholder="Contraseña" />
<div class="pass-icon"></div>
</div>
```

```
<div class="footer">
            <p>¿No tienes cuenta? <a href="registro.html">Regístrate</a></p>
        <input class="button" type="submit" value="Entrar">
</div>

</form>
</body>

</html>
```

APÉNDICE D
Código fuente - función textClean

```
def textClean(tweet):
   tweet_no_special_entities = re.sub(r'\&\w*;', '', tweet)
   tweet_no_hyperlinks = re.sub(r'https?:\/\/.*\/\w*', '', tweet_no_tickers)
   tweet_no_hashtags = re.sub(r'#\w*', '', tweet_no_hyperlinks)
   tweet_no_punctuation = re.sub(r'[' + punctuation.replace('@', '') + ']+',
   ' ', tweet_no_hashtags)
   tweet_no_https = re.sub(r'http', '', tweet_no_punctuation)
   tweet_no_emojis = ''.join(c for c in tweet_no_whitespace if
                             c <= '\uFFFF')
   tknzr = TweetTokenizer(preserve_case=False, reduce_len=True,
                          strip_handles=True)
   tw_list = tknzr.tokenize(tweet_no_emojis)
   list_no_stopwords = [i for i in tw_list if i not in cache_english_stopwords]
   list_no_stopwords = [i for i in list_no_stopwords if i not in
   cache_spanish_stopwords]
   tweet_filtered = ' '.join(list_no_stopwords)
   return (tweet_filtered)
```

APÉNDICE E
Diccionario relación código de idiomas

```
[('af', 'Afrikaans'),
('ar', 'Arabic'),
('ast', 'Asturian'),
('az', 'Azerbaijani'),
('bg', 'Bulgarian'),
('be', 'Belarusian'),
('bn', 'Bengali'),
('br', 'Breton'),
('bs', 'Bosnian'),
('ca', 'Catalan'),
('cs', 'Czech'),
('cy', 'Welsh'),
('da', 'Danish'),
('de', 'German'),
('dsb', 'Lower Sorbian'),
('el', 'Greek'),
('en', 'English'),
('en-au', 'Australian English'),
('en-gb', 'British English'),
('eo', 'Esperanto'),
('es', 'Spanish'),
('es-ar', 'Argentinian Spanish'),
('es-co', 'Colombian Spanish'),
('es-mx', 'Mexican Spanish'),
('es-ni', 'Nicaraguan Spanish'),
('es-ve', 'Venezuelan Spanish'),
('et', 'Estonian'),
('eu', 'Basque'),
('fa', 'Persian'),
('fi', 'Finnish'),
('fr', 'French'),
('fy', 'Frisian'),
('ga', 'Irish'),
('gd', 'Scottish Gaelic'),
```

```
('gl', 'Galician'),
('he', 'Hebrew'),
('hi', 'Hindi'),
('hr', 'Croatian'),
('hsb', 'Upper Sorbian'),
('hu', 'Hungarian'),
('ia', 'Interlingua'),
('id', 'Indonesian'),
('io', 'Ido'),
('is', 'Icelandic'),
('it', 'Italian'),
('ja', 'Japanese'),
('ka', 'Georgian'),
('kab', 'Kabyle'),
('kk', 'Kazakh'),
('km', 'Khmer'),
('kn', 'Kannada'),
('ko', 'Korean'),
('lb', 'Luxembourgish'),
('lt', 'Lithuanian'),
('lv', 'Latvian'),
('mk', 'Macedonian'),
('ml', 'Malayalam'),
('mn', 'Mongolian'),
('mr', 'Marathi'),
('my', 'Burmese'),
('nb', 'Norwegian Bokmål'),
('ne', 'Nepali'),
('nl', 'Dutch'),
('nn', 'Norwegian Nynorsk'),
('os', 'Ossetic'),
('pa', 'Punjabi'),
('pl', 'Polish'),
('pt', 'Portuguese'),
('pt-br', 'Brazilian Portuguese'),
('ro', 'Romanian'),
('ru', 'Russian'),
('sk', 'Slovak'),
('sl', 'Slovenian'),
('sq', 'Albanian'),
('sr', 'Serbian'),
('sr-latn', 'Serbian Latin'),
('sv', 'Swedish'),
('sw', 'Swahili'),
('ta', 'Tamil'),
('te', 'Telugu'),
('th', 'Thai'),
('tr', 'Turkish'),
('tt', 'Tatar'),
```

```
('udm', 'Udmurt'),
('uk', 'Ukrainian'),
('ur', 'Urdu'),
('vi', 'Vietnamese'),
('zh-hans', 'Simplified Chinese'),
('zh-hant', 'Traditional Chinese')]
```

APÉNDICE F
Paquetes a instalar

Package	Version
aspy.yaml	1.1.1
aylien-apiclient	0.6.0
beautifulsoup4	4.6.0
bokeh	0.12.15
branca	0.2.0
cached-property	1.4.3
certifi	2018.1.18
cfgv	1.1.0
chardet	3.0.4
click	6.7
colorlover	0.2.1
cycler	0.10.0
decorator	4.3.0
Django	2.0.2
donut	0.2.2
folium	0.5.0
futures	3.1.1
googletrans	2.3.0
goslate	1.5.1
httplib2	0.11.3
identify	1.1.0
idna	2.6
ipython-genutils	0.2.0
Jinja2	2.10
jsonschema	2.6.0
jupyter-core	4.4.0
kiwisolver	1.0.1
lxml	4.2.1
MarkupSafe	1.0
matplotlib	2.2.2
nbformat	4.4.0
nltk	3.2.5
nodeenv	1.3.1
numpy	1.14.2

packaging	17.1
pandas	0.22.0
Pillow	5.1.0
pip	9.0.3
plotly	2.7.0
pluggy	0.6.0
pre-commit	1.10.2
py	1.5.3
pymongo	3.6.1
pyparsing	2.2.0
python-dateutil	2.7.2
pytz	2018.3
PyYAML	3.12
requests	2.18.4
scikit-learn	0.19.1
scipy	1.0.1
setuptools	38.5.1
six	1.11.0
stopwords	0.1.3
textblob	0.15.1
toml	0.9.4
tornado	5.0.2
tox	3.0.0
traitlets	4.3.2
translate	3.5.0
translator	0.0.9
twitter	1.18.0
urllib3	1.22
vincent	0.4.4
virtualenv	16.0.0
wheel	0.30.0
wordcloud	1.4.1