Tweethawk
Monitorización avanzada en Twitter

Tweethawk
Advanced monitoring on Twitter

Alexandro Bindreiter Perez

Universidad Complutense de Madrid

Facultad De Infomática

Supervisor: Alberto Diaz

Trabajo de Fin de Grado

Grado en Ingeniería del Software

Curso académico 2020/21

«*Igitur qui desiderat pacem, praeparet bellum*»

«*Quien desea la paz, deberá preparase para la guerra..*»

Flavio Vegecio Renato

Índice general

Tweethawk server

`http://tweethawk.bindrei.com`

Palabras Clave

Twitter, análisis, monitorización, redes sociales, Social Media Intelligence (SOCMINT), Open Source Intelligence (OSINT)

Keywords

Twitter, analysis, monitoring, social networks, Social Media Intelligence (SOCMINT), Open Source Intelligence (OSINT)

Resumen

En lo que se tarda en leer esta línea se han publicado en Twitter 18.000 tweets. Al final de este día habrán sido 500 millones[1]. Con 300 millones de usuarios mensuales activos, Twitter es la tercera red social más usada del planeta, y la primera por excelencia en transmisión de noticias e información.

En la era digital en la que vivimos, la información es poder. Pero de lo que nadie habla es del factor tiempo. Reaccionar a tiempo a la información es tan importante como obtenerla.

Twitter nos permite recibir notificaciones cuando una cuenta publica un tweet. Sin embargo la posibilidad de que nos notifique cuando un tweet contiene una palabra en concreto o que cumpla una norma es muy limitada. Es un todo o nada.

El objetivo de Tweethawk es ser un sistema avanzado de notificaciones en tiempo real. Podremos crear grupos de usuarios, a los que monitorizará de forma automatizada y generará avisos en tiempo real basándose en unas normas lógicas que podremos definir: ¡Avísame si alguien en este grupo de usuarios publica un texto que incluya esta palabra! ¡O avísame si publica una imagen que contenga este contenido!

Abstract

In the time it takes to read this line, 18,000 tweets have been published on Twitter. At the end of this day it will have been 500 million [1]. With 300 million active monthly users, Twitter is the third most used social network on the planet, and the first par excellence in transmission of news and information.

In the digital age we live in, information is power. But what nobody talks about is the time factor. Reacting on time to information is as important as getting it.

Twitter allows us to receive notifications when an account publishes a tweet. However, the possibility of notifying us when a tweet contains a specific word or that meets a standard is very limited. It is all or nothing.

Tweethawk's goal is to be an advanced real-time notification system. We can create groups of users, which will be monitored in an automated way and will generate alerts in real time based on logical rules that we can define: Let me know if someone in this group of users publishes a text that includes this word! Or let me know if you post an image containing this content!

Capítulo 1

Introducción

1.1. Motivación

La cantidad de usuarios que hay en Twitter, unido con la cantidad de publicaciones que puede llegar a hacer cada uno a lo largo del día hace que esta red social sea una de las mayores fuentes de información de la historia. Sin embargo, la experiencia de usuario no ofrece una forma de recibir notificaciones en tiempo real sobre lo que hacen otros perfiles.

Hay determinadas situaciones en las que se desea estar pendiente de lo que dice un grupo de usuarios entorno a un tema en concreto. La única posibilidad que nos da Twitter de hacerlo es estar continuamente recargando la aplicación y cambiando entre perfiles comprobándolo manualmente.

A día de hoy existen ciertas aplicaciones que nos permiten generar analíticas acerca de esta red social, pero sus sistemas de filtrado y notificación suelen ser muy básicos. (Limitados al texto plano no asociado a un usuario)

1.2. Objetivos y solución tomada

El objetivo es realizar un software que pueda monitorizar usuarios o grupos de usuarios definidos en tiempo real y generar alertas o eventos en el momento que uno de estos haga una publicación que cumpla unas normas previamente definidas.

Los dos principales contenidos que encontramos en Twitter son texto y multimedia. Por lo tanto, habrás dos tipos de normas: Por un lado podremos generar alertas que nos avisen si un usuario ha publicado un String de texto en concreto o una expresión regular (ReGex). Por otro lado, podremos definir alertas que nos avisen si un usuario ha publicado un contenido multimedia específico mediante reconocimiento de imágenes. Estas opciones abarcan la detección de armas, droga, dinero, pornografía o contenido gore entre otros.

Por ejemplo. Definimos un grupo con 10 usuarios a los que queremos monitorizar, y definimos la siguiente norma: Detectar publicaciones con imágenes que contengan armas.En el momento que cualquiera de esos 10 usuarios twitee una imagen de un arma la aplicación generará una alerta que nos llegará al correo electrónico. (Si así lo hemos definido). Sin embargo, cualquier otro tweet no relacionado con la norma pasará inadvertido.

Poder delegar este trabajo a una aplicación, con la tranquilidad de saber que lo hará de forma automatizada y solo nos avisará cuando surja una publicación es algo que ahorrará mucho tiempo a quien tenga que realizar esta tarea. Por ejemplo, un departamento de orientación de un colegio que quiera saber en tiempo real si sus alumnos publican fotografías de armas o drogas. O un periodista que quiera ser de los primeros en ser notificado si ciertos personajes públicos hablan sobre un tema en concreto etc sin tener que estar constantemente revisando la aplicación.

1.3. Tweethawk: Resumen de la aplicación desarrollada

Tweethawk pretende ser la aplicación que cumpla los objetivos citados previamente. Esta plataforma SOCMINT (Social Media Intelligence) nos permite definir grupos, a los que añadir usuarios y reglas para monitorizar. Basado en las normas y publicaciones en Twitter de estos usuarios genera alertas y/o eventos. Estos eventos puede ser o bien simplemente almacenar el contenido del tweet en la plataforma, o bien enviarlo por Slack para recibirlo instantáneamente, ya sea en un ordenador o en un móvil.

Para mejorar la experiencia de usuario dispone de una interfaz gráfica que se ofrece como un servicio web. A través de esta interfaz, ‘podremos crear los mencionados grupos, a los que se añaden usuarios de Twitter y reglas de monitorización sobre estos. Además permite configurar las claves de API que son necesarias para la conexión con Twitter y otros servicios.

El frontend ha sido desarrollado en HTML + CSS, Bootstrap, Javascript y JQuery, mientras que el backend emplea Python3, Flask y MySQL junto con los servicios externos

de PicPurify para el reconocimiento de imágenes y Slack.

Capítulo 2

Estado del arte

2.1. Analizando aplicaciones existentes

En este apartado analizaremos algunas de las aplicaciones con mayor similitud al proyecto que pretendemos desarrollar.

2.1.1. Twilert.com

Twilert es una aplicación web de monitorización y búsqueda avanzada en Twitter. Ofrece un plan gratis que permite definir una único filtro y distintos planes de pago que ofrecen funciones más avanzadas. Nos centraremos en todas estas funciones.

La configuración es muy sencilla: Un menú de configuración nos permite seleccionar distintas normas lógicas, entre las que destacan:

- Si incluye / No incluye cierta palabra.
- Si es escrito por o destinado para cierto usuario.
- Si incluye links.
- Si tiene intención positiva / Negativa.

Posteriormente, tras configurar nuestras normas, nos redirigirá a un panel para configurar

el email o emails donde queremos que se nos notifique, y la frecuencia con la que queremos que se haga. Disponemos de las siguientes opciones:

- Cada hora
- Diariamente
- Semanalmente
- En tiempo real. Esta corresponde al plan PRO que supone un gasto de 99USD mensuales.

Al probar la aplicación hemos observado ciertas peculiaridades:

Las normas que definimos en los filtros siempre se aplican con el operador lógico AND y nunca con OR. Esto hace que si un usuario que quiera recibir todos los tweets de un perfil que sean o preguntas, o que contengan la palabra azul deberá crearse dos normas de monitorización distintas. Si lo hace en una sola, se aplicaría de la siguiente manera: "Todos los tweets del perfil X que contengan la palabra azul y que sean preguntas"

Como consecuencia de esto anterior, si quisiéramos monitorizar a 3 usuarios al mismo tiempo deberíamos crear 3 filtros distintas, aplicando las normas que queramos a cada uno de ellos, triplicando el trabajo. En vez de tener (UsuarioA OR UsuarioB OR UsuarioC) AND (Tweet contiene la palabra azul) tendríamos UsuarioA AND usuarioB AND UsuarioC AND (Tweet contiene la palabra azul). Este último escenario nunca se puede cumplir, ya que solo un usuario puede haber escrito el tweet, por lo que siempre por lo mínimo 2 de las 4 condiciones serian negativas y la proposición lógica sería siempre falsa y no generaría ninguna alerta.

Por otro lado, al tratar de analizar los tweets de un usuario, lo que la aplicación hace es, en vez de descargar todos los tweets en orden cronológico de un usuario específico y ver cuales coinciden (Entrar al perfil y comprobar), simplemente aplica una búsqueda sencilla en el buscador de Twitter con un filtro tipo user:x AND "palabra clave". Esto puede hacer que se pierda información, ya que la búsqueda genérica de twitter muchas veces omite información que no considera relevante. Lo ideal sería solicitar a la API todos los tweets del usuario mencionado y comparar con las normas.

2.1.2. Awario.com

Awario es una aplicación web de analítica en twitter. En ella podemos generar informes en tiempo real acerca de toda la actividad en torno a una o varias palabras.

El usuario define que palabras o keywords le interesa analizar, y esta aplicación genera gráfico en tiempo real entorno a ella, así como la recopilación de publicaciones en la que está contenida.

Sin embargo esta búsqueda se da sobre toda la actividad de twitter, y no permite realizarla sobre uno o varios usuarios en concreto. Si es cierto que sobre el resultado nos permite filtrar por autor, pero habremos procesado miles de tweets que no nos interesan.

Por lo que parece, está más destinada a analizar las modas y corrientes en redes sociales (Como por ejemplo un producto, lo que dice la gente sobre nuestra empresa o sobre el competidor, pero sin un publico objetivo en concreto). Esta aplicación proporciona una visión general.

2.1.3. Google alerts

No tiene nada que ver con Twitter, pero el concepto es bastante parecido a lo que se pretende hacer con la aplicación desarrollada. Google Alerts nos permite generar alertas en tiempo real ante la publicación de noticias en todo internet. El usuario selecciona uno o varios términos que está interesado en ser notificado y recibe alertas cuando esto ocurre.

De tal manera que si configuramos el término “Real Madrid”, recibiremos notificaciones por email cada vez que un periódico blog o web en internet publiquen la palabra “Real Madrid”.

Esta herramienta es muy similar a lo que se pretende hacer con Twitter, pero sin entrar en el reconocimiento de imágenes, solo que aplicado a contenido estático en todo internet.

2.1.4. PicPurify

PicPurify es una API de pago utilizada para el reconocimiento de imágenes.

Este es un servicio utilizado por diversas aplicaciones para moderar imágenes. Dispone

de una API, a la cual enviamos una URL de una imagen, junto con qué queremos que detecte en ella y nos devuelve si esa temática está presente en ella.

PicPurify es capaz de detectar las siguientes características en una imagen:

- Pornografía: Desnudos, actos sexuales etc.
- Contenido violento: Gore, cadáveres, heridas, accidentes etc.
- Armas: Pistolas, cuchillos, armas de calibre militar etc.
- Drogas: Todo tipo de drogas, desde marihuana a jeringuillas.
- Esvásticas.
- Dinero.
- Gestos obscenos
- Desnudez parcial.

Más adelante se explicará como nuestro proyecto emplea dicho servicio.

Capítulo 3

Tweethawk: Monitorización avanzada en Twitter.

En este capítulo hablaremos de la aplicación que hemos desarrollado y explicaremos su funcionamiento a alto nivel, centrándonos en la experiencia de usuario y funcionalidades sin hacer referencia a la parte técnica.

3.1. ¿Qué es?

Tweethawk pretende dar solución a los problemas planteados anteriormente. Permite monitorizar cualquier publicación de un grupo definido de usuarios en twitter, basándose en unas directrices que nos interesen y que tendremos que definir previamente.

En esta aplicación podemos crear grupos de monitorización, basados en reglas y usuarios. Introducimos en la aplicación qué usuarios queremos monitorizar, y añadimos unas reglas sobre estos usuarios. Cada vez que unos de estos usuarios publique un tweet que cumpla cualquiera de dichas reglas, la aplicación lo detectará en tiempo real y recibiremos una alerta. (A través de la plataforma o en el móvil a través de Slack).

Tweethawk tiene tres tipos de reglas o normas de monitorización:

- Texto plano: Detectar si un tweet contiene una palabra en concreto.
- ReGex: Detectar si un tweet cumple un patrón de expresión regular.

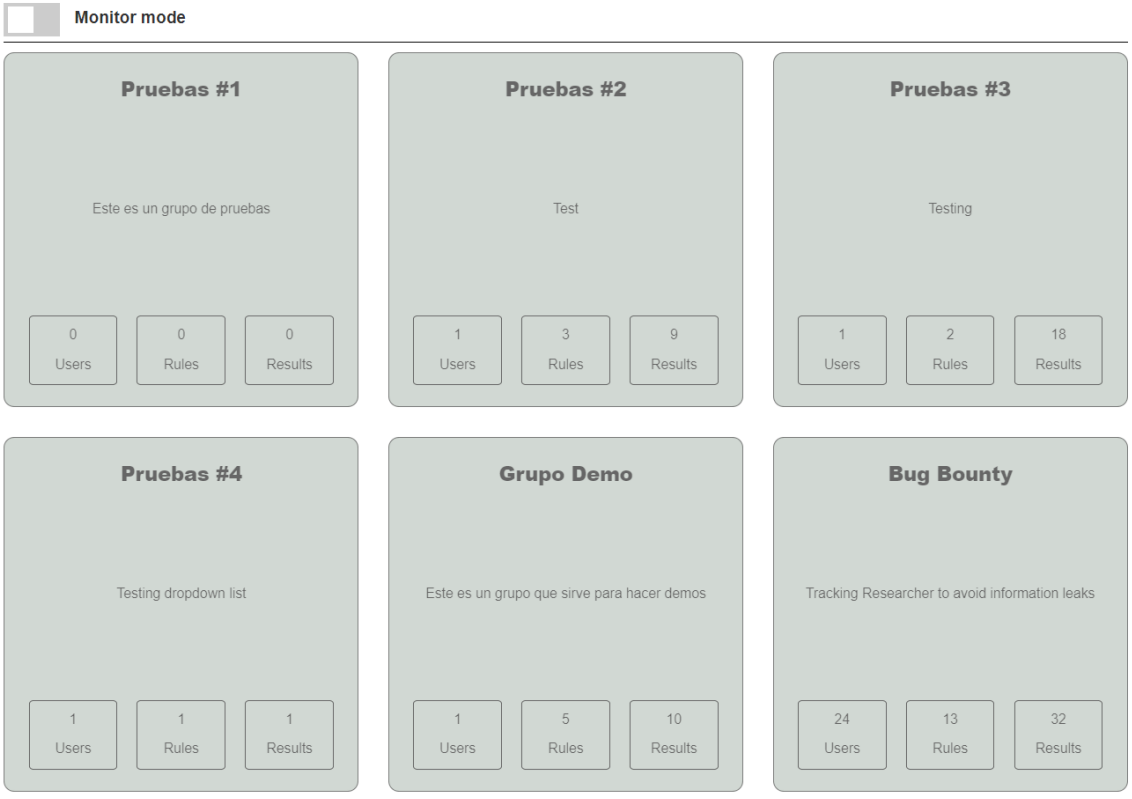
- Reconocimiento de imágenes: Detectar si un tweet contiene una imagen con una temática determinada (Ej: Armas)

Previamente hemos comentado que esta aplicación podría servir para detectar ciertas situaciones como el acoso escolar, los incumplimientos de acuerdos de confidencialidad o opiniones acerca de algo de un público determinado.

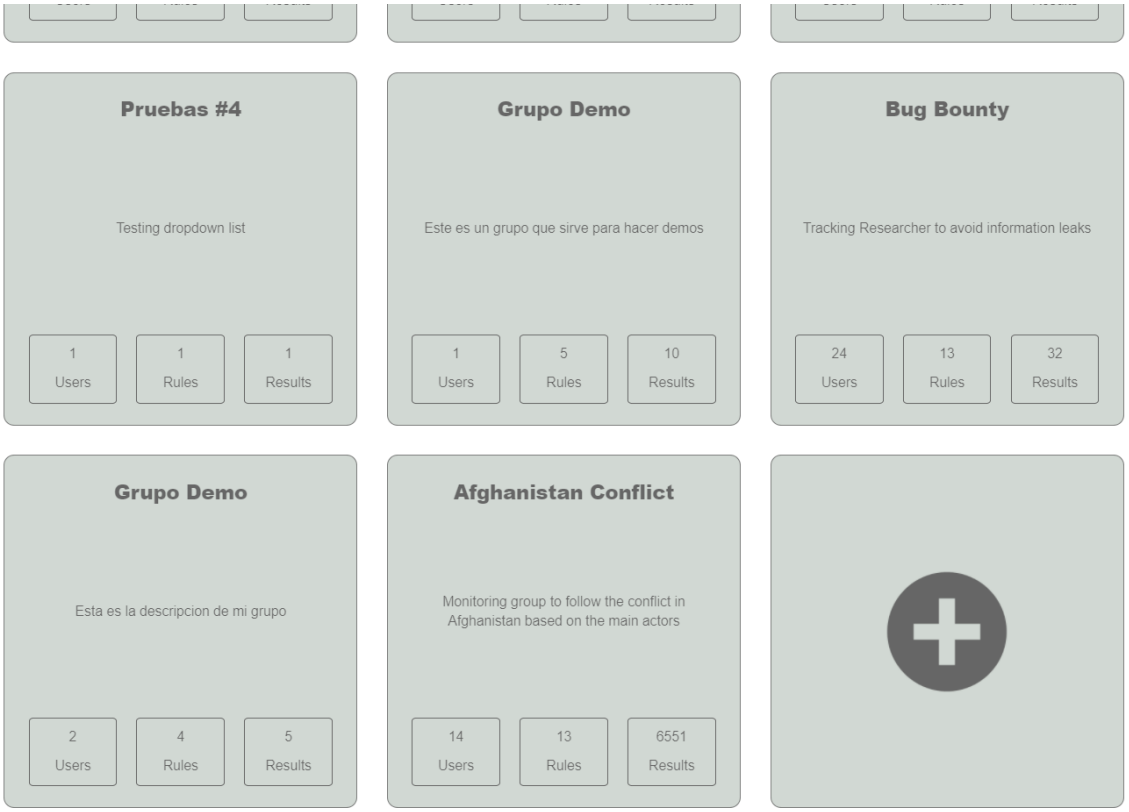
Sin embargo es importante recalcar que Tweethawk no pretende ser un software de detección de patrones o que agudice su búsqueda con el tiempo: Toda monitorización requiere previamente un trabajo de investigación y análisis por parte de quien la lleve a cabo. El éxito de esta dependerá de lo minucioso que haya sido dicho trabajo: Tweethawk no es una IA, si no una herramienta para la monitorización y alerta en tiempo real.

3.2. Cómo crear un grupo de monitorización.

Al acceder a la aplicación, se nos redirigirá automáticamente a un menú (/monitor/-groups/) con la lista de todos los grupos creados. Si queremos acceder a alguno de estos grupos, simplemente tenemos que hacer click encima de ellos.

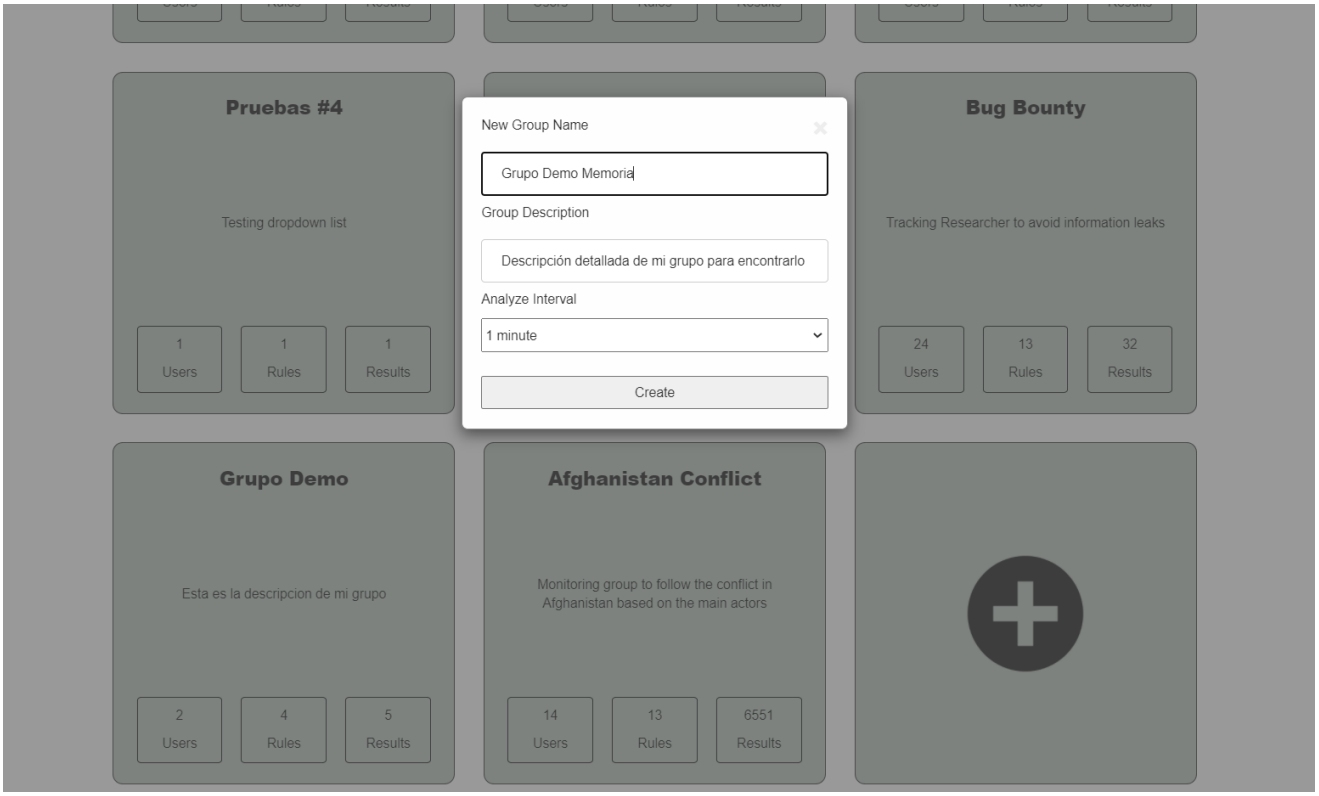


Si queremos crear un grupo nuevo, debemos buscar y clickar el símbolo (+) el cuál está en la última posición de la lista de grupos:



A continuación procederemos a rellenar los campos del formulario pop-up. Añadimos un nombre y una descripción y el intervalo de monitorización y pulsamos en “Create”.

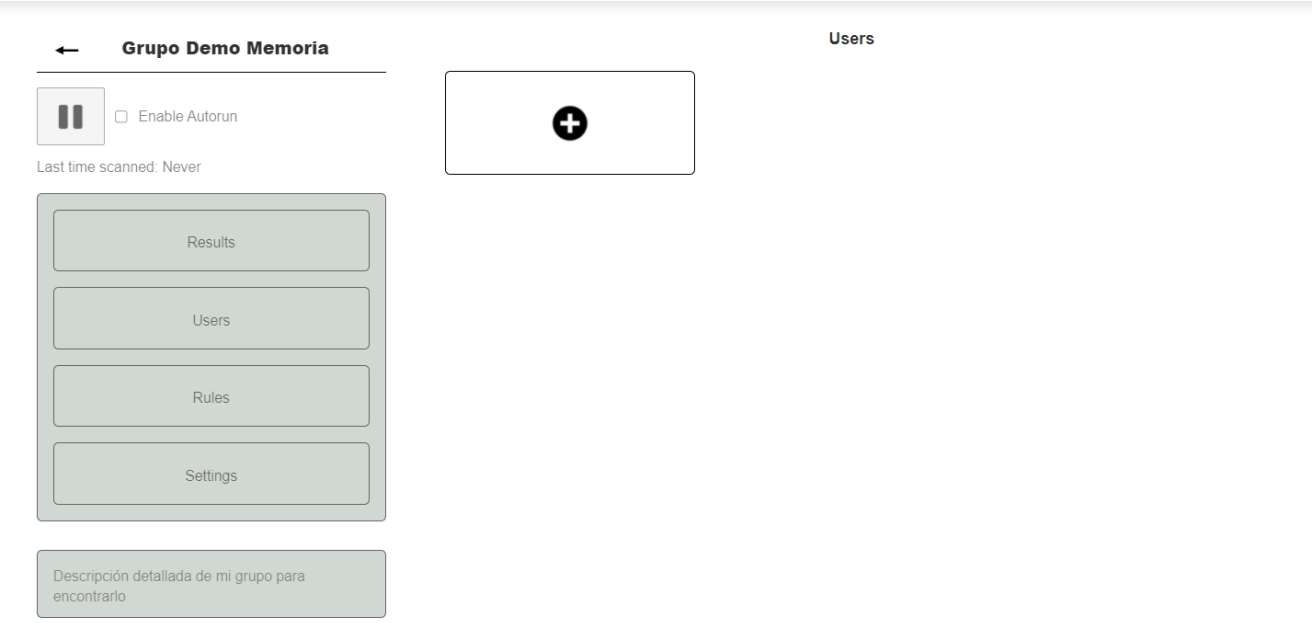
“Analyze interval” es si duda la casilla más importante: Define cada cuanto tiempo vamos a realizar la monitorización de las cuentas del grupo cuando el modo automático está activado. Por ejemplo, si ponemos 30 minutos, la aplicación comprobará cada 30 minutos si los usuarios de twitter del grupo de monitorización han añadido algo a esta red social. Si seleccionamos 1 minuto estaremos comprobándolo en tiempo real con un margen de 60 segundos de notificación.



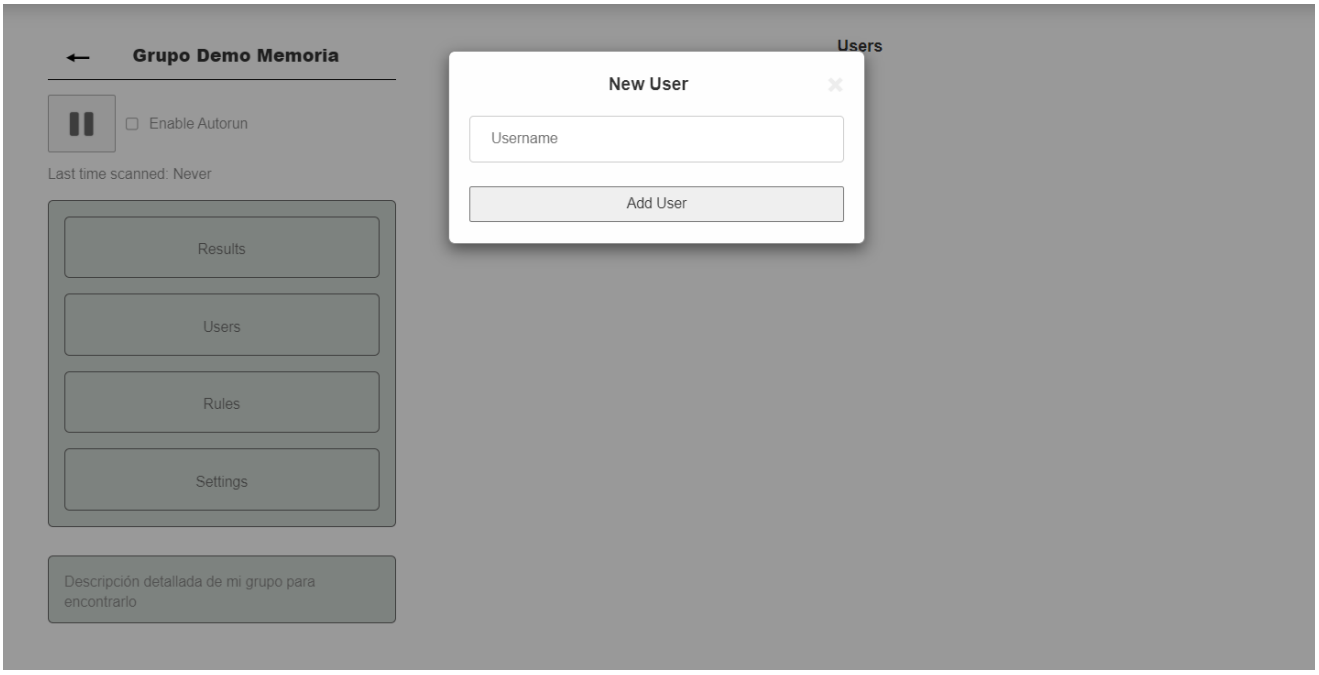
Una vez hemos creado el grupo, veremos como este aparece en la lista de grupos creados, con sus valores inicializados a 0, pues aún no tenemos usuarios, ni normas ni resultados.

3.3. Cómo añadir usuarios a un grupo de monitorización.

Una vez dentro del grupo de monitorización, pinchamos sobre el menú “Users” y sobre el símbolo (+).

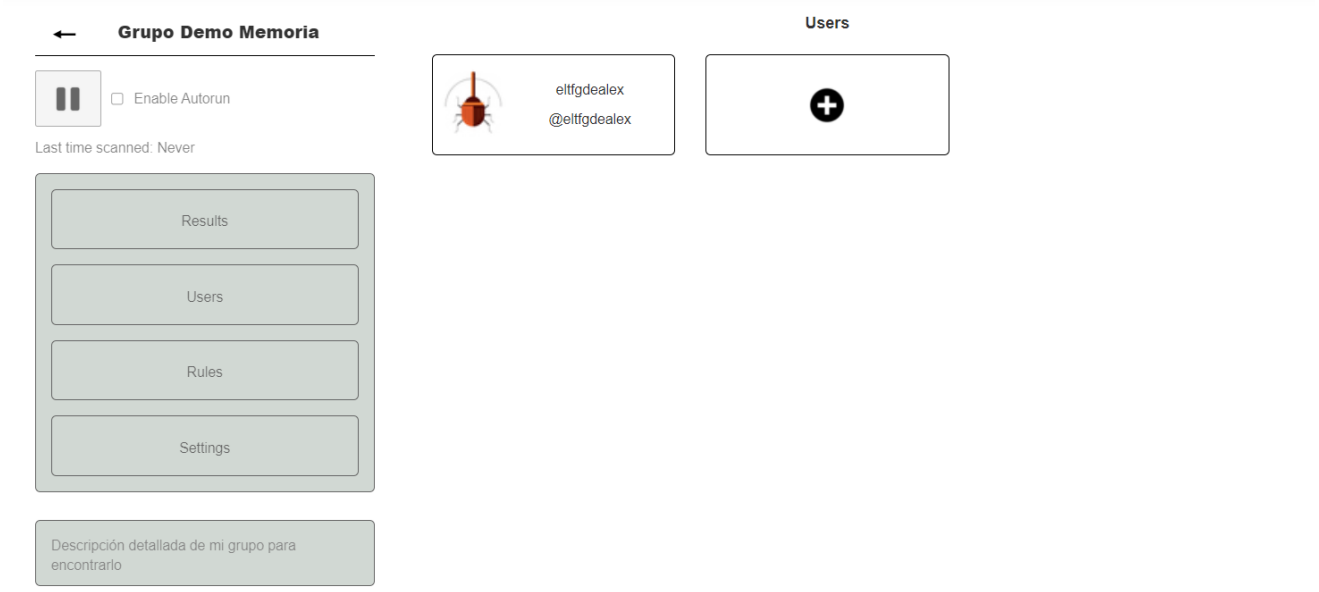


Al pulsar sobre este símbolo de añadir, aparecerá un formulario pop-up donde tendremos que introducir el nombre del usuario de twitter que queremos añadir a la monitorización.



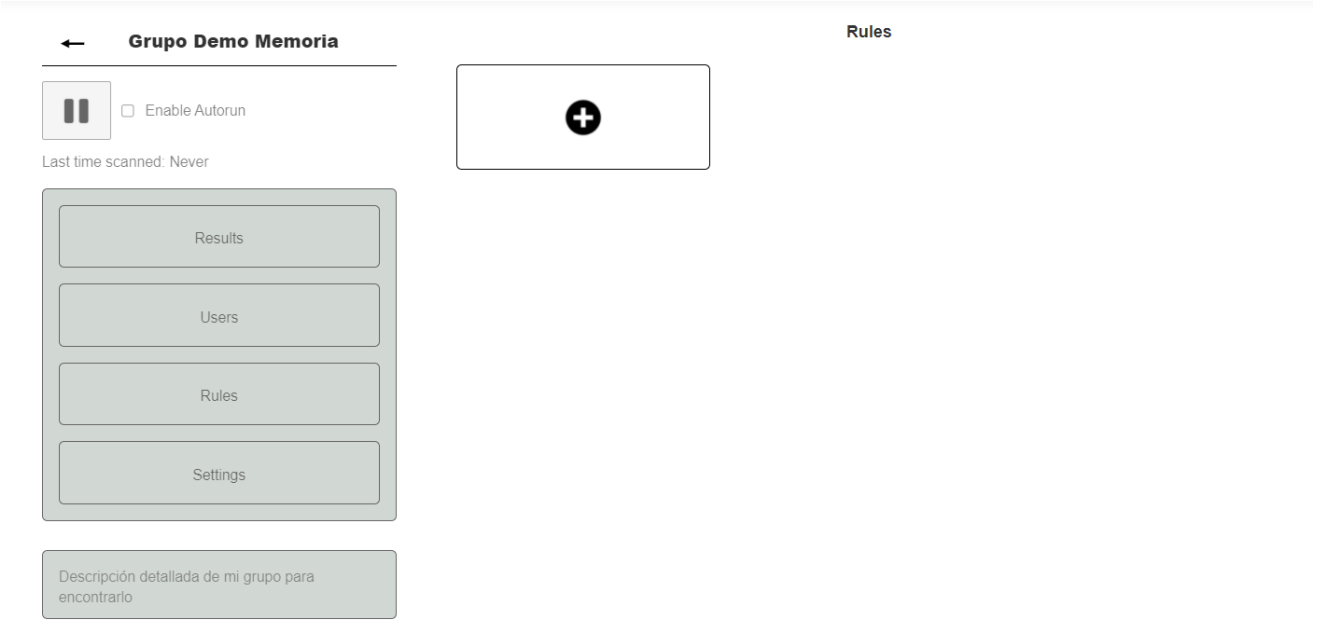
Tras introducir el nombre de usuario y pulsar sobre "Add user" podrá pasar lo siguiente:

- Si el usuario existe en twitter: Se añadirá y aparecerá como usuario del grupo
- Si el usuario no existe en twitter: No se añadirá ningún usuario
- Si el usuario ya está incluido: No se añadirá de nuevo, pues los usuarios duplicados no están permitidos.

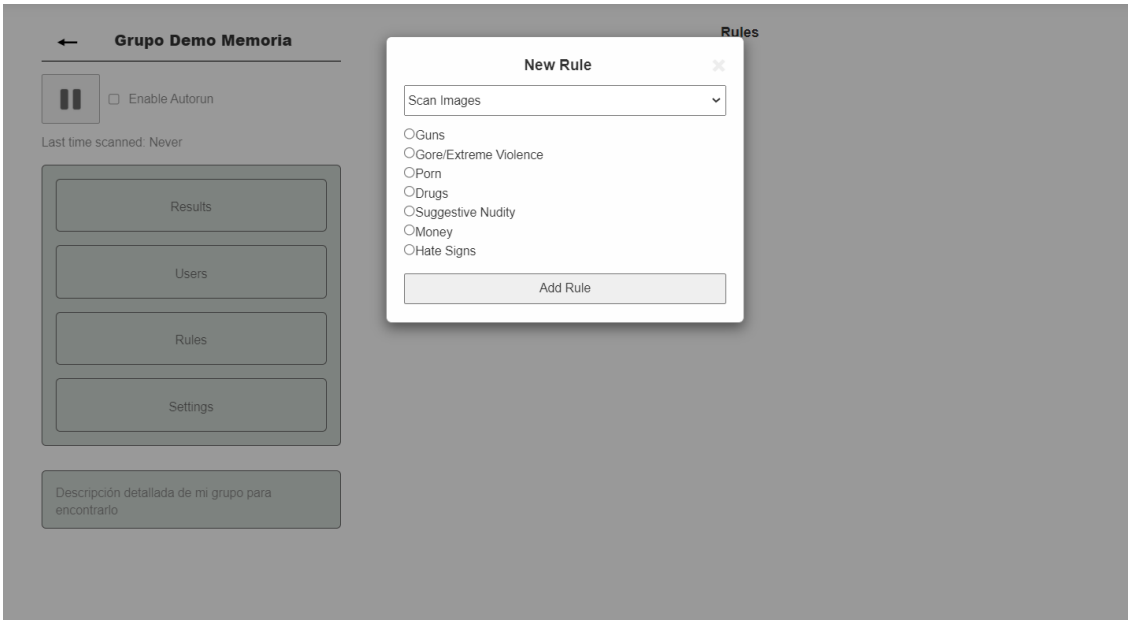


3.4. Cómo crear una regla de monitorización para un grupo.

Una vez dentro del grupo de monitorización, pinchamos sobre el menú “Rules” y sobre el símbolo (+).

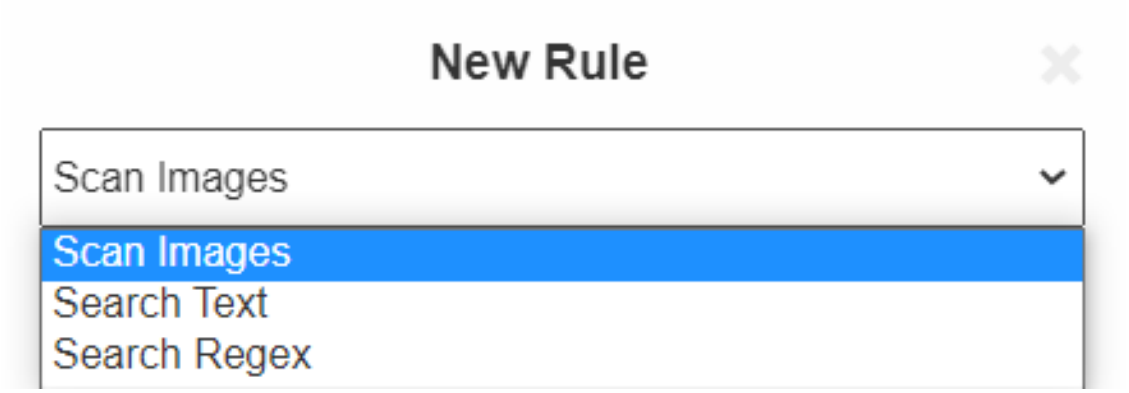


Al pulsar sobre este veremos como aparece un formulario Pop-up, con un menú de selección con las diferentes normas/reglas de monitorización que podemos crear:



Tras pulsar en "Add Rule" se añadirá la norma creada a la lista de normas existentes para el grupo de monitorización.

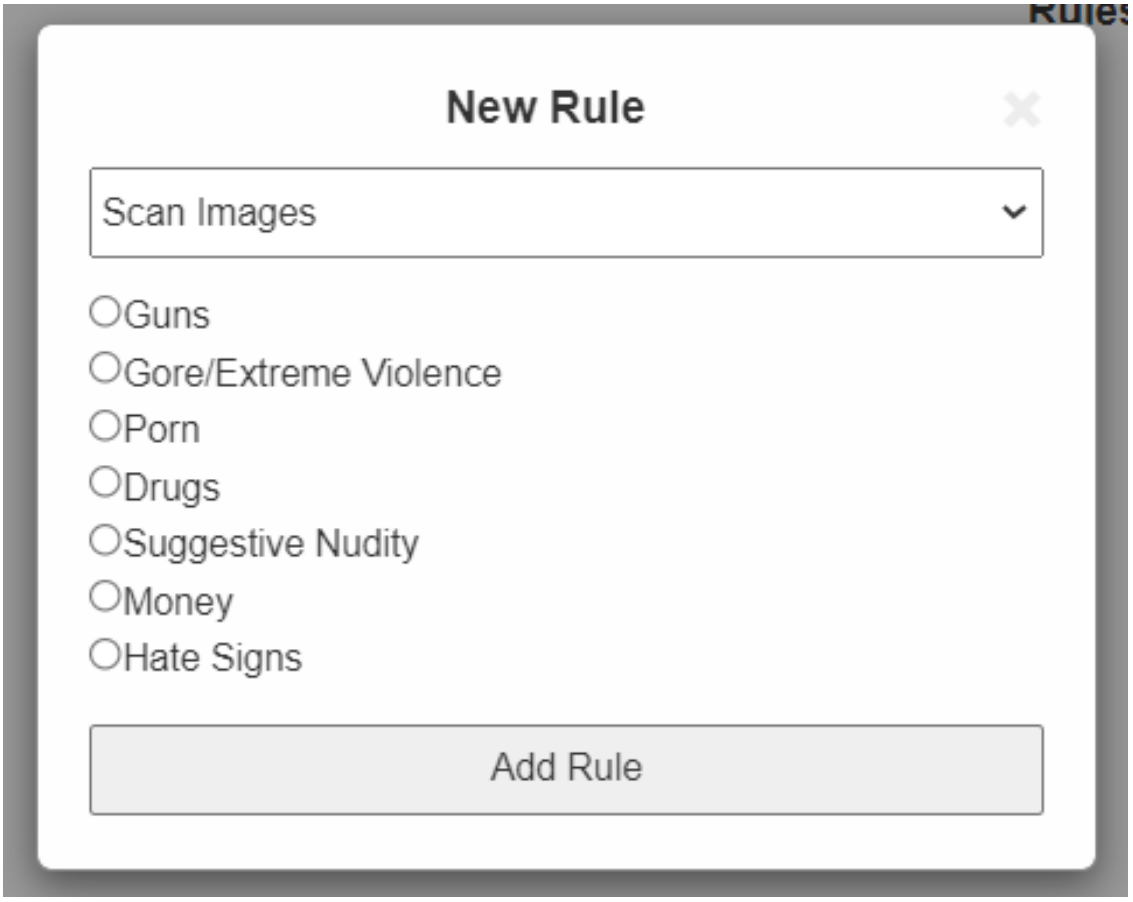
A continuación se explicarán los tres tipos de reglas de monitorización que podemos crear:



3.4.1. Reconocimiento de imágenes

Esta norma permite detectar los tweets publicados por los usuarios del grupo de monitorización que contengan imágenes con una temática específica.

La norma de reconocimiento de imágenes nos permite elegir entre armas, pornografía, drogas, desnudez, dinero y símbolos de odio.

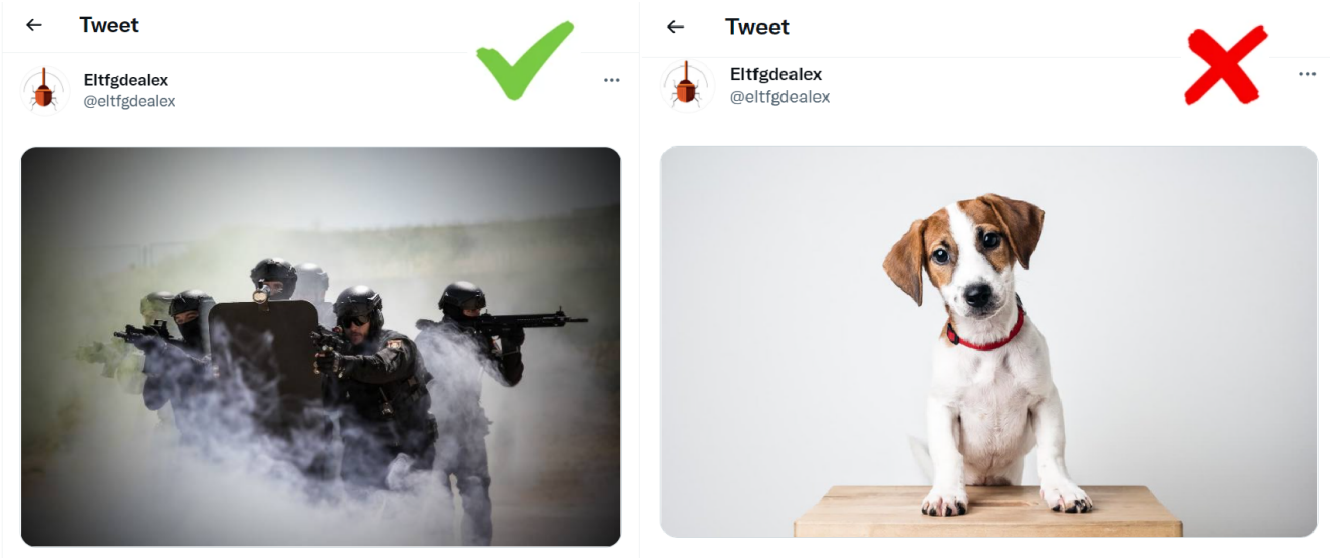


Si nos interesase que detectase por ejemplo, armas, drogas, y dinero, deberíamos crear tres normas de monitorización con reconocimiento de imágenes, una para cada una de ellas.

Un caso de uso sería el siguiente:

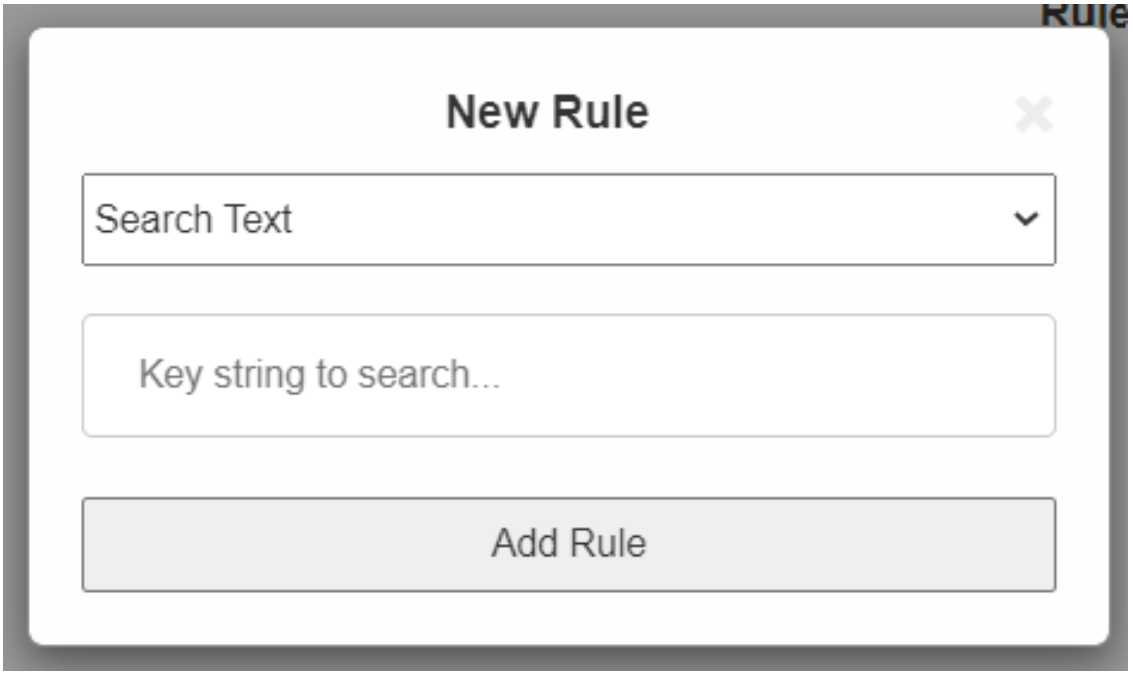
Supongamos que estamos monitorizando una cuenta de twitter en específico, y añadimos la norma de reconocimiento de imágenes para Armas:

Si este publica en twitter una imagen que contiene un rifle de asalto, y una imagen que contenga un perro y ningún arma, recibiremos un aviso por la primera, que es la que coincidirá con la regla de monitorización, pero no la segunda.



3.4.2. Texto plano

Esta norma permite detectar los tweets publicados por los usuarios del grupo de monitorización que contengan una cadena de caracteres en concreto, es decir texto plano.

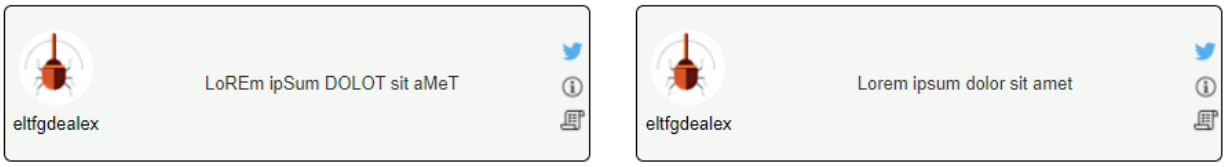


Un caso de uso sería el siguiente:

Supongamos que estamos monitorizando una cuenta de twitter en específico y queremos que se nos notifique acerca de todos los tweets con la siguiente cadena de texto: “lorem ipsum”.

Suponiendo los siguientes tweets, estos serían los que serían detectados y los que no:

Y estos serían los resultados reflejados en la aplicación:



3.4.3. ReGex. Expresiones regulares.

Esta norma permite detectar los tweets publicados por los usuarios del grupo de monitorización que contengan un patrón RegEx en concreto.

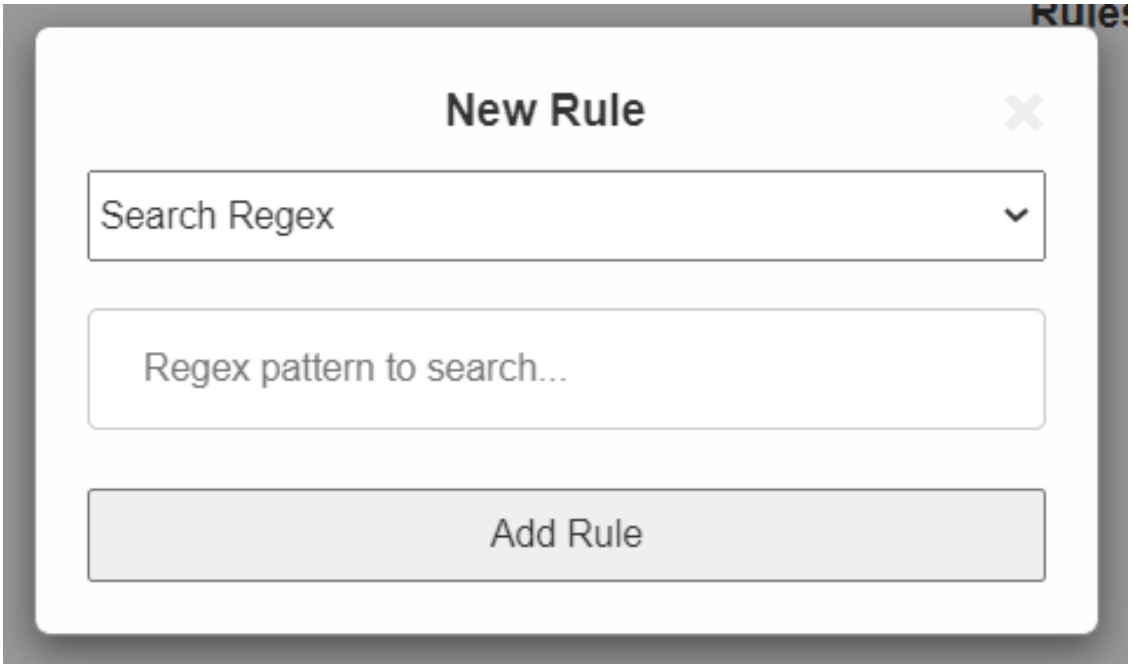
¿Que es ReGex?

Es una cadena de caracteres utilizado para definir un patrón de búsqueda en un texto y obtener substring que coincidan con este. Sirve para cuando queremos buscar algo, que no sabemos su forma literal, pero si su estructura.

Varios ejemplos serían:

- ([a-z0-9|-]+\.)*[a-z0-9|-]+\.[a-z]+ : Obtener dominios y subdominios de un texto.
- ∧Hola.*Adiós$: Cualquier texto que esté entre Hola y Adiós. Por ejemplo: “Hola, esto es un TFG, Adiós” sería detectado, pero “Esto es un TFG, Adiós” no, ya que no cumple el patrón de llevar un “Hola” delante.

Documentación de Mozilla acerca de ReGex: `https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Regular_Expressions` [2]



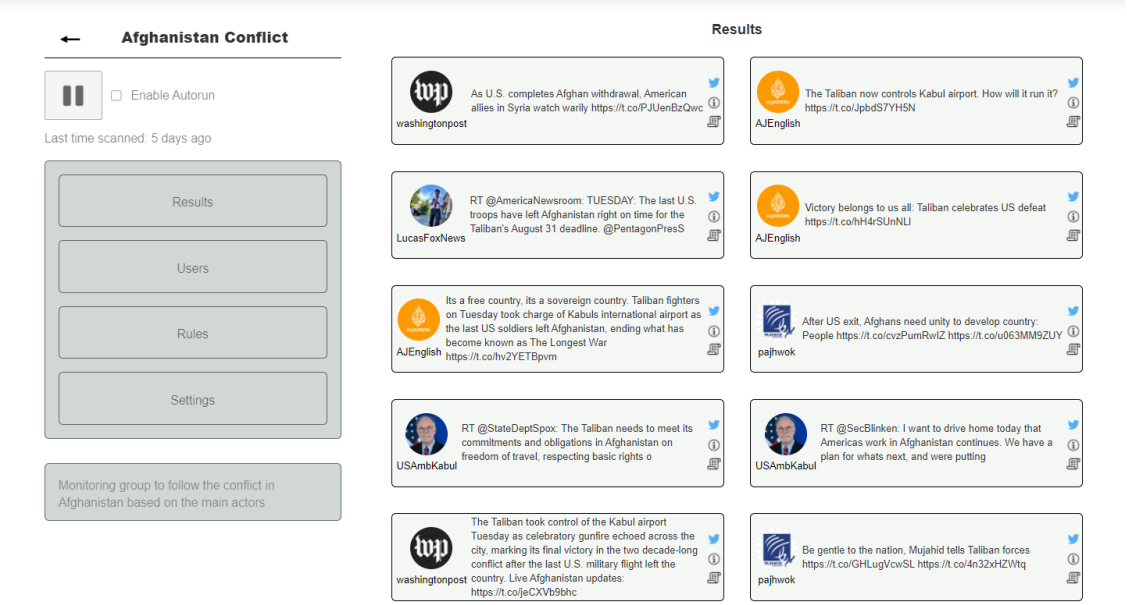
3.5. Visualización de resultados

Según nuestra aplicación va monitorizando a las cuentas de twitter añadidas, basándose en las normas dictaminadas, se van generando resultados que generan alertas (Como por ejemplo un envió de notificación).

Estos resultados se almacenan en la base de datos, con la intención de que aunque el usuario borre la publicación aún tengamos acceso a ella.

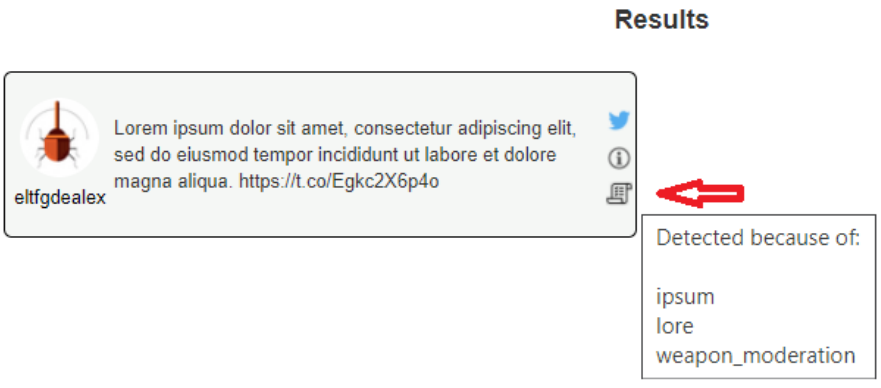
Para acceder a estos resultados, debemos pulsar en el menú en la sección de “Results”, y se mostrarán por orden de antigüedad. Los más recientes estarán primero. Aquellos resultados nuevos, estarán resaltados por un color distinto, para diferenciarlos de aquellos que ya se han visualizado.

Aquí tenemos la visualización de resultados de uno de los casos de uso que se presentarán en el capítulo 4.



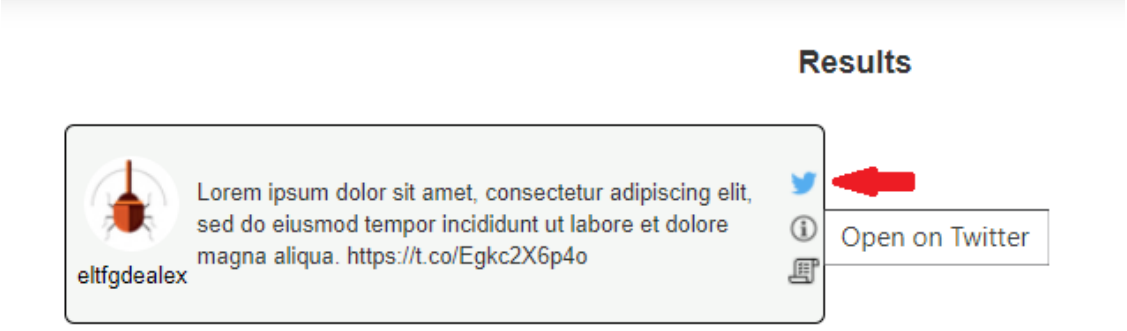
Cada elemento de resultado tendrá varios elementos accionables para acceder a más información:

- Si situamos el cursor sobre el siguiente icono, se desplegará una lista con las reglas por el cual ha sido capturado ese tweet, como podemos ver en la siguiente imagen:

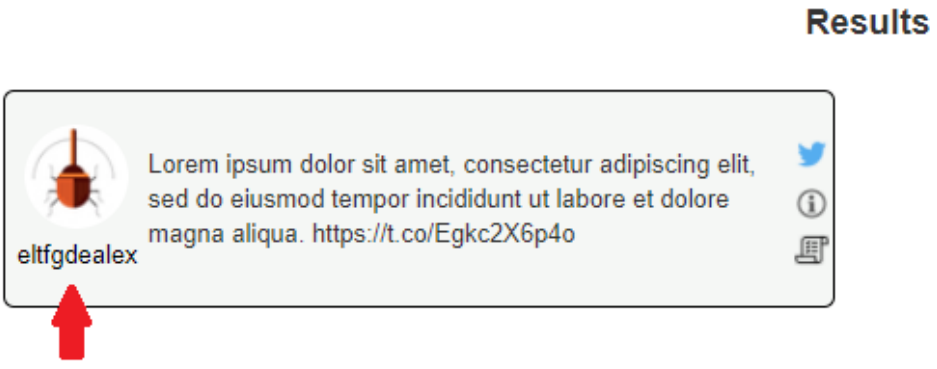


- Si situamos el cursor sobre el siguiente icono, se desplegará la fecha en la que ha sido publicado el tweet, y la fecha en la que ha sido escaneado:

- Si pulsamos sobre el icono de Twitter, situado junto al resultado, podremos abrir el tweet directamente en twitter, para acceder a el rápidamente en caso de que fuese necesario.



- Si pulsamos sobre el nombre del usuario que ha publicado el tweet, se abrirá su perfil directamente en la red social, para acceder a el rápidamente en caso de que fuese necesario.



3.6. Monitorización automática VS Monitorización manual.

Tweethawk permite activar o desactivar la monitorización continua para pasar a modo manual, en caso de que se quiera ahorrar recursos, o simplemente parar los escaneos durante un tiempo.

Monitorización Automática

La monitorización automática es aquella que se ejecuta constantemente, y que nos notifica en tiempo real. No necesita de un usuario dándole órdenes, si no que siempre está de fondo 24 horas al día.

Para comenzar a monitorizar, debemos activar el motor de monitorización de la plataforma. Una vez activado este motor, el escaneo estará ejecutándose, analizará qué grupos están activados.

Una vez está activado este motor, se escanearán todos los grupos que tengan activado el modo monitorización. Si este motor no está activado, por mucho que un grupo esté configurado para ser monitorizado, no lo hará.

Por ejemplo:

- Motor de la plataforma **activado**. Grupo 11 **activado** para modo automático. Grupo 11 **SI** se monitoriza automáticamente.
- Motor de la plataforma activado. Grupo 11 desactivado para modo automático. Grupo 11 **NO** se monitoriza automáticamente.
- Motor de la plataforma **desactivado**. Grupo 11 **activado** para modo automático. Grupo 11 **NO** monitoriza automáticamente.

En función del tiempo configurado para cada grupo, cada vez que se termine un escaneo, se configurará la próxima hora a la que se tiene que volver a realizar para determinado grupo. Es decir, si escaneamos el grupo 11 a las 14:00, el cual se monitoriza cada 5 minutos, al terminar esta tarea se reflejará que el próximo escaneo debe ser a las 14:05.

El motor va comparando el tiempo actual, con los tiempos de próximo escaneo, y si el primero supera al segundo para uno o varios grupos, los monitoriza y marca la próxima hora.

Si tenemos grupo 11, el cual se escanea cada 3 minutos, y el grupo 12, el cual lo hace cada 2, tendríamos una monitorización así:

- 14:00: Se escanea grupo 11 y grupo 12
- 14:01: No se escanea ningún grupo
- 14:02: Se escanea grupo 11
- 14:03: Se escanea grupo 12
- 14:04: Se escanea grupo 11
- 14:05: No se escanea ningún grupo
- 14:06: Se escanea grupo 11 y grupo 12

Monitorización Manual

A parte de esta monitorización que se realiza de forma periódica, también podemos realizar nosotros una monitorización en el momento que queramos, iniciándola de manera manual.

Cuando accedemos a un grupo, podemos observar un botón de play. Si pulsamos sobre este, se ejecutará de manera manual un nuevo escaneo, que no interfiere con el automático.

3.7. Alertas en el móvil: Configurando la aplicación para recibir notificaciones en Slack.

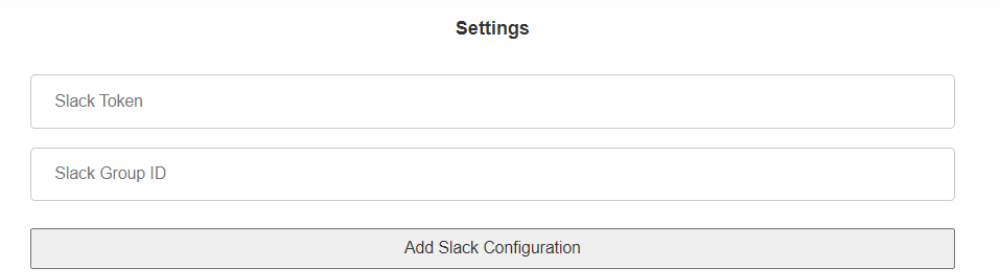
La aplicación se ha desarrollado de manera que si le facilitamos un token de Slack y un ID de grupo de Slack a un grupo determinado, usará un bot para enviarnos mensajes con la información una alerta cada vez que esta se genere a través de dicha aplicación.

Esto nos permitirá recibir toda la información en tiempo real en cualquier dispositivo, ya sea ordenador, tablet o teléfono móvil.

Al poder configurar un canal y token por cada grupo, podremos configurar un grupo de Slack por cada grupo de monitorización, para así clasificar todos los mensajes.

Para generar el token necesario la documentación oficial de Slack nos explica como crear un Bot y añadirlo a un canal aquí: `https://slack.com/intl/es-es/help/articles/115005265703` [3]

Para usar este token con Tweethawk, al entrar en un grupo de monitorización, presionamos sobre "Settings", lo cual nos llevará al siguiente menú:



3.8. Tweethawk vs Otras aplicaciones

En la siguiente tabla, podremos observar cuales son las diferencias principales entre Tweethawk y las otras principales aplicaciones de monitorización de Twitter de las que hemos hablado antes:

TweetHawk vs Otras Aplicaciones			
X	TweetHawk	Twilert	Awario
Monitorizar múltiples usuarios	✅	❌	❌
Filtrado por texto plano	✅	✅	✅
Filtrado por RegEx (REGular EXpression)	✅	❌	❌
Filtrado por temática en imágenes: Reconocimiento de imágenes	✅	❌	❌
Filtrado por preguntas	❌	✅	❌
Filtrado por intención positiva o negativa	❌	✅	❌
Notificación de evento a través de email	✅	✅	✅

Capítulo 4

Presentación de Resultados: Casos de uso y utilidad

A continuación se tratarán varios casos de uso en los que Tweethawk pudiera servir para estar alerta ante ciertos comportamientos y realidades, ya sea aplicados en escenarios reales, o bien en escenarios reconstruidos basándose en hechos pasados, o ficticios.

4.1. Geopolítica actual. Siguiendo en directo el conflicto en Afganistán

4.1.1. Introducción

En el siguiente caso de uso analizaremos como Tweethawk ha podido ser utilizado para seguir todos los acontecimientos al instante de la ofensiva Talibán que puso fin a la República Islámica de Afganistán durante agosto de 2021 y las posteriores consecuencias: La evacuación de la OTAN por el aeropuerto de Kabul, la resistencia anti-talibán en Panjshīr, el atentado del ISPK etc.

4.1.2. Contexto histórico

Tras el 11-S, el régimen Talibán se negó a entregar a Osama Bin Laden a EEUU, lo que originó el "casus beli" para la invasión de Afganistán por parte de Estados Unidos y otros miembros de la OTAN, en la llamada Operación Libertad Duradera, que comenzó el 7 de octubre de 2001 y llevó a la caída del régimen Talibán y a su retirada al norte del país y a la formación de un nuevo gobierno. [4]

EEUU y otros miembros aliados mantuvieron ahí su presencia tras haber conquistado el país, sin embargo, tras la muerte de Osama Bin Laden en Mayo de 2011, Barack Obama anunció la intención de retirar las tropas de dicho país.

En 2020, la administración Trump negoció con los talibanes en el llamado Acuerdo de Doha [5] la retirada de sus tropas de suelo afgano a cambio de prometer no dar cobijo ni a Al-Qaeda ni a ISIL.[6] La administración Biden llevó a cabo la retirada de tropas a lo largo de los primeros meses de 2021. De la mano de Estados Unidos, todos los países de la OTAN retiraron sus tropas paulatinamente. Una vez retiradas la 15.000 tropas de la coalición internacional, los Talibán comenzaron una rápida ofensiva de norte a sur, comenzando el 1 de Mayo de 2021 y terminando el 15 de agosto tras la caída de la capital: Kabul.

Los servicios de inteligencia calcularon 6 semanas para la caída de Kabul, sin embargo esta se rindió en 24 horas, lo cual generó una necesidad de respuesta inmediata para evacuar a los civiles estadounidenses, personal diplomático y colaboradores afganos con la coalición internacional, los cuales eran considerados "traidores". No solo EEUU tenía la necesidad de sacar a sus civiles, si no todas las naciones de la OTAN como Canadá, Francia, Alemania, etc.

La evacuación se llevó a cabo por el aeropuerto internacional de Kabul, siendo este el único punto seguro pactado con los Talibán, de manera improvisada y poco preparada. Dio lugar a miles de civiles afganos abarrotando el aeropuerto, trepando encima de aviones o invadiendo las pistas de despegue, con la intención de encontrar una plaza en un vuelo de evacuación que no estaba destinado para ellos.

Ante esta gran aglomeración de militares y personal diplomático, mezclado con civiles, sin apenas perímetros de seguridad, hizo que saltasen las alarmas, advirtiendo del riesgo potencial de un atentado llevado a cabo por el ISKP [7] (Estado Islámico del Gran Jorasán). El estado islámico es enemigo acérrimo de los Talibán y de la Coalición Internacional, por lo cual que estuviesen todos juntos en el mismo espacio suponía un gran riesgo.

5 días después de los primeros avisos, el día 26 de agosto de 2021 se produjo un atentado con coche bomba en el aeropuerto de Kabul, que se saldó con la vida de 183 personas, 13 marines estadounidenses entre ellos.

Tras este suceso, los distintos países terminaron rápidamente sus labores de evacuación, adelantando la fecha de retirada debido al extremo peligro de repetirse otro ataque. [8]

Para más información, se recomienda leer los siguientes enlaces:

`https://www.cfr.org/timeline/us-war-afghanistan` [9]

`https://es.wikipedia.org/wiki/Guerra_de_Afganistán_(2001-2021)` [4]

4.1.3. Monitorización

A continuación añadiré qué usuarios hemos decidido añadir a nuestro grupo de monitorización sobre el conflicto de Afganistán y por que, así como las normas de monitorización que hemos empleado:

- John Kirby (@PentagonPresSec): John Kirby es el secretario de prensa del pentágono. A través de su cuenta se emiten multitud de comunicados oficiales del departamento de defensa de estados unidos. Probablemente una de las cuantas más activas en lo que a decisiones y eventos militares se refiere: Despliegues, ataques, etc
- Washington Post (@washingtonpost): Periódico estadounidense enfocado en asuntos internacionales. Es uno de los mayores periódicos de la nación y destaca por su cobertura 24h en redes sociales.
- AlJazeera (@AJENews y @AJEnglish): Mayor medio de comunicación de oriente medio. Hemos introducido la cuneta en la que traducen las noticias al inglés.
- pajhwok (@pajhwok): Periódico local de Kabul. A pesar de que la mayoría de sus publicaciones son en árabe, traduce las más importantes al inglés. Resulta de gran utilidad debido a su cercanía con los hechos.
- Embajada de EEUU en Afganistán (@USEmbassyKabul): Cuenta oficial de la embajada de EEUU en Afganistán, con sede en la capital del país, Kabul. Emite comunicados de prensa sobre asuntos diplomáticos, así como el estado del personal y evacuación.

- Ross Wilson (@USAmbKabul): Embajador de los estados unidos en Afganistán. Sus tweets son similares a los de la embajada, pero transmiten situaciones más personales, y una narración desde su punto de vista de los eventos. Mientras que la embajada emite comunicados oficiales, el embajador añade fotografías de los sucesos que vive tomadas por él mismo.

- Ministerio de defensa de Gran Bretaña(@DefenceHQ y @DefenceHQPress): Publica comunicados de prensa oficiales, mayoritariamente sobre el estado y labores de evacuación de personal diplomático, civiles y colaboradores británicos.

- Joe Biden (@POTUS): 55º presidente de los Estados Unidos de América. Sus publicaciones sobre el conflicto, si bien son oficiales, denotan un claro tono de campaña electoral. Sin embargo, al ser el mayor representante del principal país beligerante en el conflicto, suele ser útil para seguir las decisiones respecto a este que seguirá su gobierno.

- Ministerio de defensa de España(@Defensagob): Publica comunicados de prensa oficiales, mayoritariamente sobre el estado y labores de evacuación de personal diplomático, civiles y colaboradores españoles.

- Ministerio de defensa de Alemania(@bmvgbundeswehr): Publica comunicados de prensa oficiales, mayoritariamente sobre el estado y labores de evacuación de personal diplomático, civiles y colaboradores alemanes.

  Nota: Nos hubiese gustado añadir cuentas representantes del otro lado del conflicto, sin embargo, todas ellas publican sus tweets en árabe. Nuestra aplicación no está preparada para traducir árabe, lo cual es una gran limitación y sin duda un punto de mejora.

Normas de Monitorización (Rules):

- "Afghan" y "afgan": Mediante este término queremos absorber cualquier publicación de las anteriores 13 cuentas que haga cualquier mención a Afganistán, tanto en inglés como en castellano: Afgano, afgana, Afghan, Afghanistan etc.

- "Talib": Hace referencia a los Talibán, principal grupo beligerante del conflicto. Cualquier mención a este término por parte de los usuarios monitorizados sin duda nos interesa.

- "ANA": Siglas empleadas para denominas al Ejército de Afganistán, el cual combate contra el movimiento Talibán. ANA son las siglas para Afghan National Army. Como la otra parte beligerante sin duda debe ser incluida al sistema de monitorización, con el riesgo de obtener falsos positivos con otras siglas iguales que no tengan nada que ver.

- "panjshir": Región de Afganistán [datos región - historia de la revuelta]. Tras el avance de los talibán, en dicha provincia el ANA se hizo fuerte. El anterior gobierno movió su sede ahí y se creó una resistencia anti-Talibán

- "Lion of Panjshir's": Alias de un señor de la guerra afgano llamado Ahmad Shāh Mas'ūd. Su hijo forma parte de la resistencia del ANA contra los talibanes en Panjshir. La prensa le ha puesto el mismo alias que a su padre. Lo añadimos como filtro tras investigar a los partícipes de dicha resistencia.

- "Herat, Kandahan y Jalalabad": Tres ciudades de Afganistán donde se han llevado a cabo diversos combates y situaciones importantes en el conflicto.

- "Kabul Airport" y "Aeropuerto de Kabul": Una vez se comunica que la evaciiación del país por parte de EEUU y otros países se llevará a cabo por este aeropuerto, decidimos añadir este término ya que la mayor parte de noticias se estaban dando allí.

- "daesh","ISIL","ISIS-K":Todos estos términos hacen referencia al estado islámico. Una vez supimos que este quería atentar contra talibanes y estadounidenses, decidimos que sería interesante añadirlo al filtro. Hemos introducido los tres nombres que tienen importancia para referirse al Estado Islámico en este conflicto: Daesh (Dawlah al-Islāmiyah fī 'l-ᶜIrāq wa-sh-Shām), ISIL (Islamic State of Iraq and the Levant), ISIS-K o ISKP .

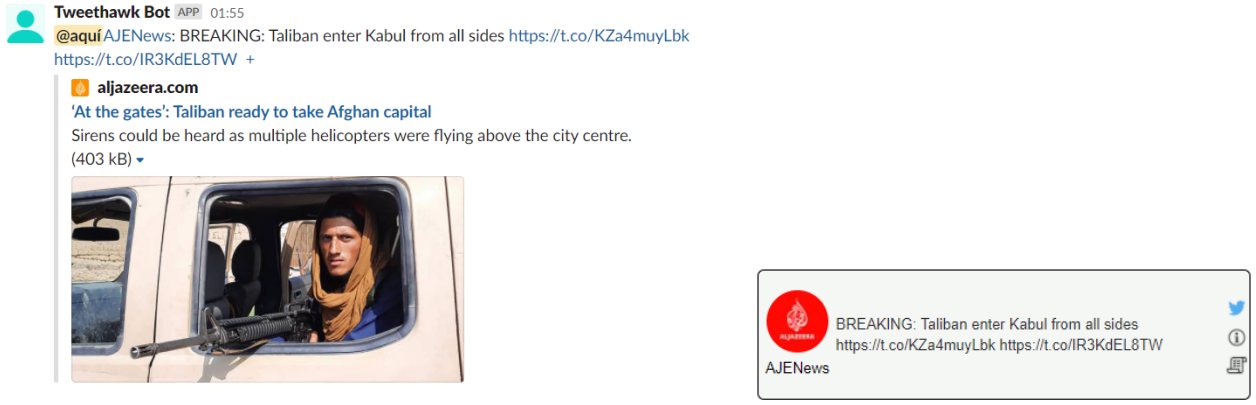
- "Abbey Gate": Puerta del aeropuerto donde se llevó a cabo el atentado por parte del ISKP. Vinculadas a este término aparecieron la mayor parte de noticias vinculadas a este suceso, por lo que lo añadimos a nuestra monitorización.

4.1.4. Resultados

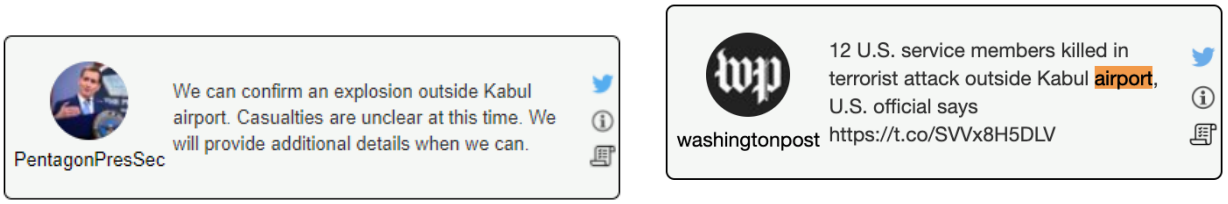
A continuación, se exponen algunos de los resultados más destacados de esas dos semanas de monitorización.

- Al Jazeera y Alarabiya confirmaron que la ciudad de Jalalabad (Una de las ciudades más grandes del país) había caído ante los Talibán.

- El día 15 de agosto el medio de comunicación Al Jazeera, tras un asedio de apenas 24h, comunicaba la caída de Kabul ante el régimen Talibán. Este es el mensaje que recibimos en Slack.



- El día 26 de junio a las 15:44 John Kirby, secretario de prensa del pentágono, publicó un tweet en el que informó de una explosión en el aeropuerto de Kabul, que se acabaría sabiendo que fue el atentado de Abbey gate. La aplicación tardó 36 segundos en detectarlo y enviar un mensaje a través de Slack y recibirlo en el móvil.



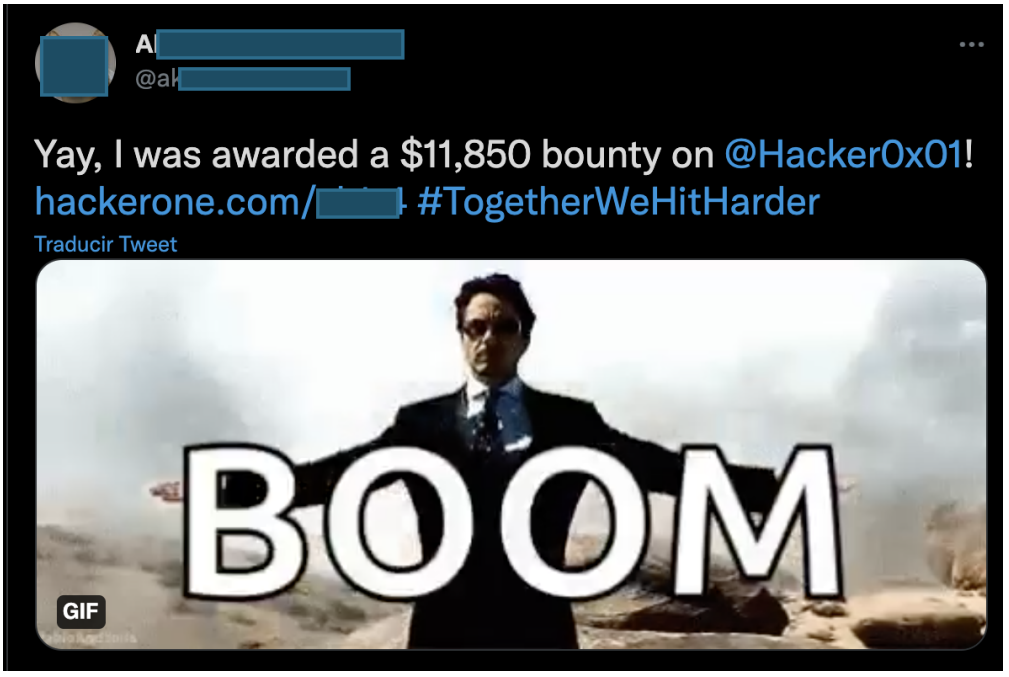
- El 31 de agosto, un reportero de la Fox News confirma que el último soldado estadounidense ha abandonado el país.

4.2. Caso real: Incumplimiento de acuerdos de confidencialidad

Imaginemos un "program manager" en un programa privado de Bug Bounty de una empresa del SP 100. Un bug bounty es un programa de ciberseguridad, en el cual se le dice a hackers freelance del mundo que se les recompensará si son capaces de encontrar brechas de seguridad en los productos y sistemas de dicha empresa.

En el programa que gestiona dicho program manager hay invitados a participar 150 hackers, que han ganado en total 8 millones de dolares en recompensas en los últimos 5 años. Sin embargo, dada la naturaleza y sensibilidad de la información, estos 150 hackers tienen un acuerdo de confidencialidad por el cual no pueden hablar ni confirmar acerca de la existencia de este programa.

Sin embargo, hemos detectado que muchos de ellos son muy activos en twitter, ya que forman parte de la comunidad de ciberseguridad en redes sociales (InfoSec). Muchos de ellos suelen hacer alarde cuando ganan una recompensa o publican en un blog explicando como han sido capaces de hackear a cierta empresa. Cuando se trata de empresas como Uber, Facebook o Google, que tienen un programa público, están en su derecho de hacerlo, ya que no hay de por medio un acuerdo de confidencialidad.



No obstante, tienen totalmente prohibido hacerlo en los programas privados, entre los que se incluye la empresa para la que trabajo. Ni para decir algo positivo, ni para quejarse: Para absolutamente nada.

Claramente es imposible que el equipo legal supervise 150 cuentas de Twitter para comprobar que en ningún momento se salten el acuerdo de confidencialidad. Sin embargo, hacer un grupo de monitorización con las cuentas de twitter de esos 150 hackers, añadiendo las palabras que tienen prohibido mencionar es totalmente factible. En tal caso, el equipo legal que supervisa esto recibiría con máximo un minuto de retraso un aviso la publicación de dicho post.

Para demostrar esta prueba de concepto, hemos creado un grupo de monitorización con las cuentas de twitter de los 40 investigadores de ciberseguridad más activos en cierto programa existente. A este grupo, le hemos añadido alertas con términos como el nombre de la empresa y sus producto más destacados, el nombre de usuario de las cuentas de los de analistas de seguridad del programa, directivos que llevan el programa etc.

Tras hacer un análisis, de cuales serían los términos que deberían llamar la atención , y que más relacionadas están con el programa, hemos procedido a introducirlos como reglas.

Tras 9 días monitorizando los nuevos tweets de todos estos usuarios cada 5 minutos, hemos obtenido aproximadamente 6 resultados de los haciendo mención en redes sociales acerca del programa. En este espacio de tiempo las cuentas publicaron 900 tweets.

Debido a la confidencialidad de dicho programa no podemos publicar los resultados. Sin embargo, si un equipo legal hubiese tenido que repasar estos 900 tweets de forma manual a través de Twitter hubiese tardado 7 horas (Asumiendo que tardase medio minuto en leer cada post), además del hecho de que no sería en tiempo real, con lo que no se pueden tomar medidas reactivas sin retraso.

4.3. Otros casos de uso

4.3.1. Opiniones públicas de los participantes a un evento

Siguiendo con el caso anteriormente propuesto, en el que no se puede romper el contrato de confidencialidad al hablar de un programa de recompensas de ciberseguridad, hablaremos de una situación opuesta: Los eventos de hacking y caza de recompensas. En estos, una compañía invita, con todos los gastos pagados a unos 30-40 hackers a un hotel, que dispone de sala de conferencias, durante dos semanas. En estas dos semanas, los hackers estarán día y noche hackeando a esta empresa y cobrando también recompensas por ello.

Mientras que en el caso anterior se fuerza el secretismo y la confidencialidad, los eventos tratan de dar publicidad a dicha empresa, y vender una imagen de seguridad a los clientes. Por ello se promueve que todos los participantes hablen de dicho evento en redes sociales. Crear un grupo de monitorización con todos los asistentes, y unas reglas que permitan identificar los tweets de estos que tienen relación con dicho evento, hará que los organizadores sepan en tiempo real, todas y cada una de las opiniones de estos, para poder reaccionar de forma rápida a cualquier comentario negativo y mejorar, o bien seguir la vía del feedback positivo.

Esto es solo un ejemplo familiar, pero sería aplicable a cualquier tipo de evento donde al usuario le interese saber las opiniones de ciertas personas en concreto, y no de forma masiva. Por ejemplo, sería útil para una conferencia o un estreno cinematográfico exclusivo (Donde incluiríamos a todos los invitados), pero no sería útil si queremos saber las opiniones de todas España respecto a un partido de fútbol.

4.3.2. Detección de acoso escolar y radicalización en redes sociales

Según UNICEF el 6'9 % de alumnos en España ha sufrido **ciberacoso**(O dice haberlo sufrido). Con el auge de las redes sociales, el acoso, que siempre ha existido, se mueve a

nuevos ámbitos. Si bien aquí es más sencillo demostrarlo, pues deja una huella literal, es igual de difícil detectarlo entre la gran cantidad de información que se produce diariamente.

Por otro lado, otra gran problema que afrenta la sociedad relacionada con las aulas y la infancia son los tiroteos escolares. Si bien en Europa es una realidad lejana, en Estados Unidos no lo es.

De 1970 a 2020 Estados Unidos ha reportado 1773 tiroteos escolares, que han dejado detrás 617 muertos y 1683 heridos. Estos datos son ofrecidos por el Centro para la defensa y seguridad nacional, que en su portal disponen de un muy elaborado mapa con estadísticas:

`https://www.chds.us/ssdb/data-map/` [10]

En numerosas ocasiones, los autores de estos crímenes han dejado una constancia previa de sus intenciones en redes sociales. Ya sea comentarios amenazadores contra sus centros educativos, o imágenes portando armas, o ambas.

Por ejemplo, el caso del tiroteo en 2018 en la escuela secundaria de Parkland que se saldó con 17 fallecidos, en el cual, su autor publicó numerosas imágenes en redes sociales en las que portaba armas y realizaba amenazas contra su centro. [11]

En otras numerosas ocasiones, el hecho de haberlo publicado ha alertado a las fuerzas de seguridad impidiendo tales ataques.

Tweethawk podría servir, tras un análisis e investigación de un equipo de orientación de un centro, monitorizar las publicaciones en redes sociales de los alumnos, buscando indicios de acoso o amenazas. En esta investigación se debe encontrar las palabras claves que interesen acerca de cada centro y sus entornos. Entre estas palabras estarían motes o insultos utilizados para acosar a estudiantes, amenazas, imágenes con armas o drogas etc.

Capítulo 5

Arquitectura del software

5.1. Back-end: Lógica de negocio y servicios externos

En este capítulo analizaremos como funciona el back-end y la lógica de Tweethawk, centrándonos en las cuestiones técnicas del desarrollo del software y su arquitectura.

5.1.1. Detalles generales

El proyecto consta de 2.500 lineas de código y está desarrollado en Python3 para el backend, HTML+CSS+JS para el front-end y MySQL para la base de datos.

El repositorio oficial es `https://github.com/bindrei/Tweethawk/`

Una versión operativa de Tweethawk está ejecutándose en `http://tweethawk.bindrei.com`

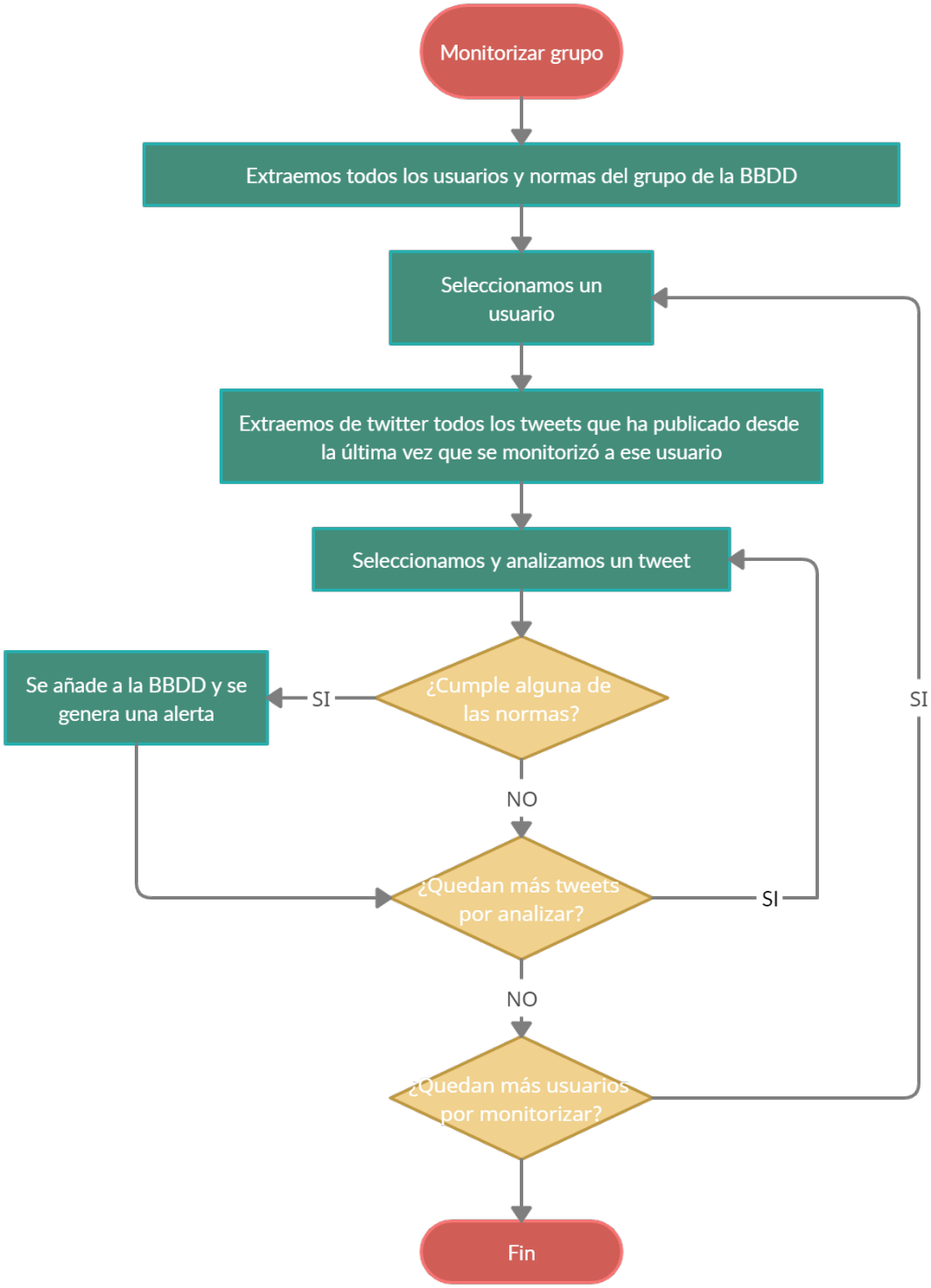
5.1.2. ¿Como funciona la monitorización de un grupo?

En esta sección trataremos de explicar cómo funciona por detrás cuando monitorizamos un grupo en concreto.

Existen tres tipos de listas fundamentales en un grupo: La lista de usuarios, la lista de tweets asociadas a cada usuario y la lista de reglas. Al monitorizar un grupo hacemos un

triple bucle: A cada usuario, extraemos de la API de Twitter los tweets nuevos desde la última vez que se monitorizó. Con cada uno de estos tweets, iteramos sobre la lista de normas. Por cada tweet, iremos extrayendo una norma tras otra para compararla con la publicación. En caso de cumplirse la norma, añadiremos a la BBDD. A cada tipo de norma se aplicará una lógica distinta (Texto plano, regex o reconocimiento de imágenes).

A continuación, veremos un diagrama de flujo que ilustra esto:


Monitorizar grupo
Extraemos todos los usuarios y normas del grupo de la BBDD
Seleccionamos un usuario
Extraemos de twitter todos los tweets que ha publicado desde la última vez que se monitorizó a ese usuario
Seleccionamos y analizamos un tweet
¿Cumple alguna de las normas?
SI
Se añade a la BBDD y se genera una alerta
NO
¿Quedan más tweets por analizar?
SI
NO
¿Quedan más usuarios por monitorizar?
SI
Fin

5.1.3. Base de datos: Diagrama Entidad-Relación

Como hemos comentado previamente, la base de datos está desarrollada en MySQL. Para el desarrollo de este proyecto hemos usado Xampp como servidor y PHPMyAdmin como gestor visual, pero se puede utilizar cualquiera siempre que sea MySQL.

A continuación haremos un repaso por las diferentes tablas de la base de datos, así como de los atributos de cada una de ellas.

monitorizeGroups

La tabla "monitorizeGroups" está destinada a almacenar los datos de los diferentes grupos de monitorización disponibles. Cada grupo estará asociado a un ID único que lo identificará como grupo. Los atributos de esta tabla son:

- **ID**: Entero. Identificador único de grupo. Clave primaria.
- **Name**: String. Nombre del grupo de monitorización. Empleado para identificarlo de forma sencilla.
- **Description**:String Descripción del grupo de monitorización. Empleada para clasificarlo y explicarlo.
- **looptime**:Entero Tiempo cada cuanto se ejecuta la monitorización, en caso de estar activado el modo automático.
- **nUsers**:Entero. Número de usuarios añadidos al grupo.
- **nRules**:Entero. Número de normas de monitorización añadidas al grupo.
- **nResults**:Entero. Número de resultados generados en el grupo.
- **lastTimeScanned**:Fecha y hora Última vez que se escaneó el grupo.
- **nextTimeScan**: Fecha y hora. Próxima vez que se debe escanear el grupo cuando está en modo automático. Este valor se obtiene de la suma entre lastTimeScanned y looptime.
- **autorun**:Booleano. Cuando está activado, significa que este grupo debe monitorizarse de forma automática.

monitorizeUsers

La tabla "monitorizeUsers" incluye a todos los usuarios que han sido añadidos a los grupos de monitorización. Puede haber un mismo usuario añadido a varios grupos de monitorización. Cada usuario está vinculado a un grupo a través del groupID.

- **userID**: Entero. Equivale al ID de dicho usuario en twitter. Este ID es el identificador único tanto en tweethawk como en dicha red social.
- **groupID**: Entero. El grupo de monitorización al que pertenece esa entrada de usuario.
- **username**: String. Es el nombre de usuario en Twitter del usuario con ese ID.
- **lastTweetCursor**: String. Es el ID del último tweet publicado por ese usuario que ha sido escaneado por Tweethawk. Ayuda a encontrar tweets posteriores a este, para no tener que repetir trabajo ya hecho. Por ejemplo, si lastTweetCursor es 1000, buscaremos en Twitter tweets publicados por este usuario y que tengan un ID mayor que 1000.
- **profilePicURL**: String. Enlace a la imagen de perfil del usuario, para mostrarla posteriormente en la interfaz gráfica.

monitorizeRules

La tabla "monitorizeRules" incluye a todas las normas que han sido añadidas a los grupos de monitorización. Puede haber una misma norma añadida a varios grupos de monitorización. Cada norma está vinculada a un grupo a través del groupID.

- **ID**: Entero. Identificador único de la norma de monitorización.
- **groupID**: Entero. El grupo de monitorización al que pertenece esa entrada de norma.
- **type**: Entero. Refleja el tipo de norma que es. Si es 1, es texto plano; si es 2, reconocimiento de imágenes; Si es 3, Regex.
- **rule**: String. Es la carga útil de la norma. Define lo que tenemos que buscar.
- **priority**: Entero. Prioridad que tiene haber encontrado esta norma. No se utiliza actualmente, pero pretende definir lo grave que es haber generado la norma. En el futuro podremos tomar una acción u otra en función de la prioridad.

monitorizeResults

La tabla "monitorizeResults" incluye a todas los resultados que han sido generados por los grupos de monitorización. Puede haber un mismo resultado añadido a varios grupos de monitorización. Cada norma está vinculada a un grupo a través del groupID y a un usuario a través de userID. Puede haber incluso varios resultados por el mismo tweet añadidos al mismo grupo, pues un post puede coincidir en varias normas, por lo que también está asociado a la norma que lo ha generado.

- **groupID**: Entero. Grupo al que pertenece el resultado.
- **userID**: Entero. Usuario por el que se ha generado el resultado.
- **ruleID**: Entero. Norma por la cual ha sido generado el resultado.
- **new**: Booleano. Indica si este resultado es nuevo o no. Es decir, si se ha visualizado antes o no a través de la plataforma.
- **tweetID**: String. Identificados del tweet que ha generado el resultado. Es el mismo identificador que en la plataforma de Twitter.
- **tweet**: String. Contenido literal del tweet publicado. Se almacena en la base de datos por si el usuario decidiese borrarlo.

monitorizeGroupSettings

La tabla "monitorizeGroupSettings" incluye las configuraciones específicas de un grupo en concreto. Es decir, lo que atañe al grupo. Actualmente, es la tabla que almacena las credenciales de Slack, ya que cada grupo puede enviar mensajes a una instancia de Slack distinta.

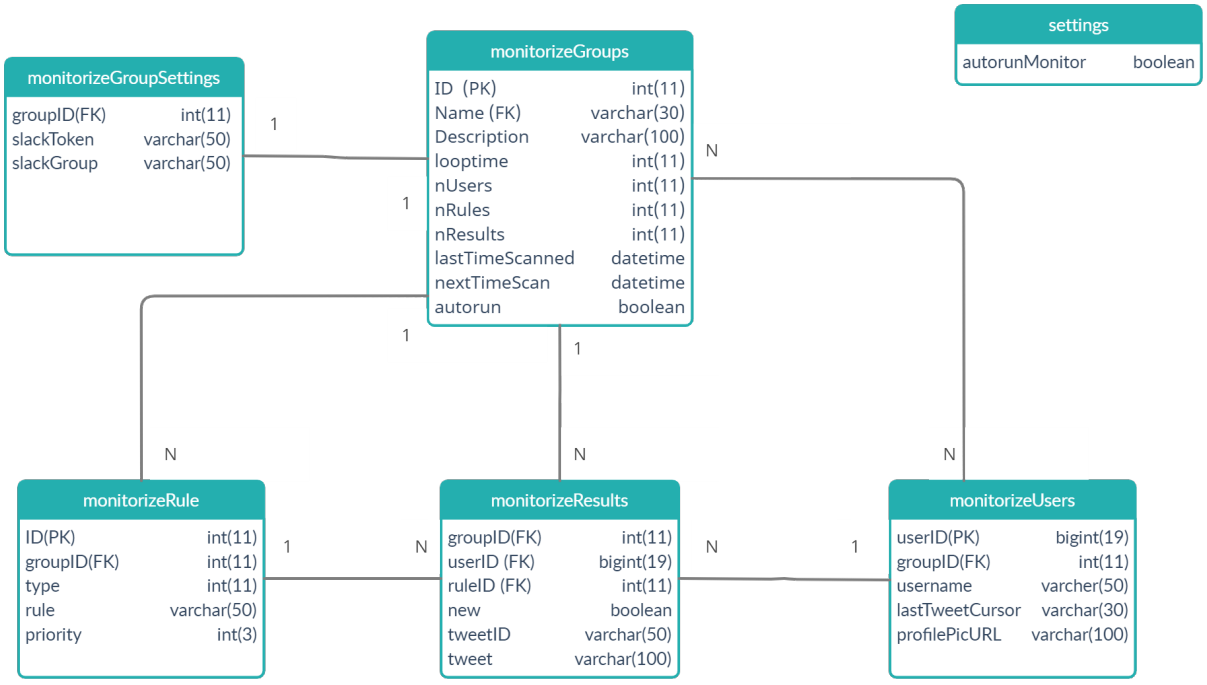
- **groupID**: Entero. Grupo al que pertenece la configuración.
- **slackToken**: String. Token de Slack que nos permite enviar mensajes a través de la aplicación.
- **slackGroup**: String. Identificador del grupo o canal de Slack por el que queremos enviar mensajes.

settings

La tabla "settings" incluye las configuraciones de todo el sistema de monitorización. Estas atañen a todos los grupos en general. Por el momento solo almacena si el motor de monitorización debe ejecutarse o no.

- **autorunMonitor**: Booleano. Indica si el motor de monitorización debe ejecutarse o no.

Diagrama de Entidad-Relación



Triggers

La tabla monitorizeGroups dispone de algunos atributos como nUsers, nRules y nResults que indican el numero de items asociados. Para mantener este número actualizado, hemos empleado triggers cada vez que se realiza un Insert.

5.1.4. Modelo DAO: Interacción con la base de datos

Para la interacción del software con la base de datos hemos empleado el patrón de arquitectura del software conocido como Data Access Object (DAO). El objetivo de este patrón es la de separar la lógica de acceso a datos, de la lógica de negocio de Tweethawk, aumentando así la modularidad del proyecto. [12]

DaoGroups.py contiene la clase que crea el objeto encargado de relacionarse de manera directa con la base de datos. Este objeto, será utilizado por la lógica de negocio para hacer las peticiones a esta, pero encapsulando la consulta SQL. De esta manera, nunca tendremos una Query en la lógica de negocio, si no llamadas a distintos métodos de Daogroups.

Por ejemplo:

monitor.py, encargado de la monitorización, el cual es lógica de negocio, tiene un método encargado de mostrar todos los usuarios pertenecientes a un grupo. En vez de realizar una consulta a la base de datos empleando una query del estilo "select * from users", lo que hace es lo siguiente:

```
1                @app.route('/monitor/groups/<int:groupID>/users/')
2                def showUsersFromMonitorGroup(groupID):
3
4                    daoGroup = DAOGroups(connection())
5                    userList = daoGroup.getGroupUsers(groupID)
6                    groupInfo = daoGroup.getGroupInfo(groupID)
7                    ...
8
```

Como se puede observar, crea un objeto DAO, del cual utiliza el método getGroupUsers(<ID>). De no emplear este patrón, cada vez que quisiésemos hacer una petición a la base tendríamos que programar repetitivamente todas las querys, haciendo el proyecto menos modular y pesado. El código de arriba, sin DAO resultaría en lo siguiente:

```
1                @app.route('/monitor/groups/<int:groupID>/users/')
2                def showUsersFromMonitorGroup(groupID):
3
4                    cursor = connectionPool.cursor()
5                    try:
6                        cursor.execute("SELECT userID, username,
   lastTweetCursor, profilePicURL FROM monitorizeUsers WHERE groupID = %s", (
   str(groupID),))
7                    except:
8                        # Handle exception
```

```
9                   userList = cursor.fetchall()
10
11                  try:
12                      self.cursor.execute("SELECT id, name, description,
    nUsers, nRules, nResults, lastTimeScanned  FROM monitorizeGroups WHERE  ID
     = %s", (groupID,))
13                  except:
14                      # Handle exception
15
16                  groupInfo = cursor.fetchall()
17                  ...
18
```

- **getGroupUsers(groupID):** Devuelve todos los usuarios que son monitorizados en un grupo dado un ID.
- **getAllGroups():** Devuelve todos los grupos de monitorización que existen.
- **getGroupstoRun():** Devuelve todos los grupos cuyo periodo de monitoriación es menor o igual a los minutos que han pasado desde la última ejecución. Estos son los grupos que están listos para escanearse de nuevo.
- **keepRunningMonitor():** Devuelve un booleano que determina si el servicio de monitorización está activado en la plataforma.
- **getGroupExecutionInterval(groupID):** Devuelve cada cuanto tiempo se debe escanear determinado grupo dado su ID.
- **getGroupInfo(groupID):** Devuelve la información de un grupo dado su ID.
- **groupExists(groupID):** Comprueba si existe un grupo con un ID dado.
- **getGroupMonitorizeRules(groupID):** Devuelve las distintas normas de monitorización de un grupo dado su ID.
- **getGroupMonitorizeResults(groupID):** Devuelve los resultados generados para un grupo dado su ID.
- **setTweetCursorFromUser(lastTweetCursor, userID, groupID):** Actualiza el ID del ultimo tweet escaneado de un usuario en concreto. Tiene la intención de permitir a la aplicación reanudar el análisis de los tweets de determinada cuenta desde el punto donde lo dejó la última vez.

- **insertRule(groupID, typeR, rule, priority):** Permite insertar una norma de monitorización asociada a un grupo.
- **setLastTimeScan(id):** Actualiza la última hora a la que se escaneó un grupo por última vez. Permite posteriormente calcular cuanto tiempo lleva un grupo sin escanearse para decidir si hay que escanearlo de nuevo.
- **setNextTimeScan(id, date):** Actualiza la próxima hora a la que se debe escanear un grupo.
- **insertResult(userID, groupID, ruleID, tweetID, tweet):** Añade un resultado a un grupo, asociado al usuario de twitter que lo ha generado. Incluye el cuerpo del tweet, la hora de publicación y los enlaces a las imágenes.
- **insertUser(userID, groupID, username, photoURL):** Añade un usuario a un grupo de monitorización. Incluye nombre de usuario y link a la foto de perfil.
- **insertGroup(name, description, loop):** Añade un grupo de monitorización a la plataforma. Inicializa todas las estadísticas a 0 y añade nombre, descripción y el tiempo que dictamina cada cuanto tiempo tiene que ser escaneado dicho grupo.
- **enableMonitor():** Activa el centinela de monitorización.
- **disableMonitor():** Desactiva el centinela de monitorización.
- **getRulesForResult(tweetID, groupID):** Dado el ID de un resultado, devuelve la norma de monitorización por la cual este tweet ha sido indexado.

5.1.5. Tweepy: Interacción con la API de Twitter

El servicio externo principal utilizado por el backend de esta aplicación sin duda es la API de twitter. La API de Twitter funciona a través del protocolo HTTPS. Cada funcionalidad, o tipo de dato que queremos extraer va acompañado de un endpoint y una serie de parámetros que queremos utilizar.

Documentación de la API de Twitter: `https://developer.twitter.com/en/docs/twitter-api/` [13]

Por ejemplo, si queremos solicitar a esta API todos los tweets de un usuario llamado @unicomplutense, debemos realizar la siguiente petición:

```
GET /2/users/2244994945/tweets
Host: api.twitter.com
Authorization: Bearer [TOKEN-DE-ACCESO]
```

Sin embargo, no sabemos cual es el ID del usuario @unicomplutense, antes debemos realizar una petición a la API para obtener el ID de una cuenta.

```
GET /2/users/by/username/unicomplutense
Host: api.twitter.com
Authorization: Bearer [TOKEN-DE-ACCESO]
```

Para evitar tener que desarrollar esto, hemos decidido emplear Tweepy. Tweepy es una librería de código abierto que ofrece una manera sencilla de interactuar con la API de Twitter, a través de clases y métodos que representan los distintos endpoints de la plataforma.

De esta manera, para realizar lo anteriormente planteado, simplemente tendremos que hacer:

```
import tweepy as twitter
twitter.user_timeline(screen_name="unicomplutense")
```

Documentación oficial de Tweepy: `https://docs.tweepy.org/en/stable/` [14]

5.1.6. Reconocimiento de imágenes: Interacción con la API de PicPurify

Como hemos visto anteriormente, Tweethawk es capaz no solo de encontrar texto plano o expresiones regulares, si no también tiene la capacidad de aplicar reconocimiento de imágenes.

Esta funcionalidad es posible gracias a PicPurify. Este servicio utilizado para moderar imágenes dispone de una API, a la cual enviamos una url de una imagen, justo con qué queremos que analice de esta, y nos devuelve si lo contiene o no, junto con el porcentaje de seguridad al acierto.

Según su White Paper, emplean un modelo de red neural entrenado con millones de imágenes para cada tipo de reconocimiento, que les permite más de un 95 por ciento de

efectividad.

Hemos decidido que es más óptimo para el proyecto emplear un servicio de terceros en vez de desarrollar nuestro propio sistema de reconocimiento de imágenes por diferentes motivos:

- Desarrollar un sistema de reconocimiento de imágenes implica un trabajo muy distinto y nada relacionado con la temática de este proyecto.
- Entrenar un modelo de IA para reconocer imágenes supone una gran cantidad de horas e investigación.
- Empleando un servicio conocido de un tercero nos asegura el correcto funcionamiento del modelo de reconocimiento de imágenes.
- Si bien esta API es de pago, nos ahorramos los costes del desarrollo del proyecto y el tiempo de computación del reconocimiento de imágenes.

PicPurify ofrece una documentación detallada sobre como usar su API en su plataforma web:

Documentación Oficial De PicPurify [15]

Cada vez que analizamos un tweet, iteramos sobre este con todas las normas del grupo. Si una norma es de reconocimiento de imágenes, comprobamos si el tweet tiene contenido multimedia.

En caso de que el tweet tenga contenido multimedia, extraemos la URL de este contenido y se lo enviamos a PicPurify de la siguiente manera:

```
1                POST  /analyse/1.1
2                Host: www.picpurify.com
3
4                API_KEY=XXXXX&task=gun_moderation&url_image=http://
   url_image_to_analyse
5
```

La respuesta de la API será así:

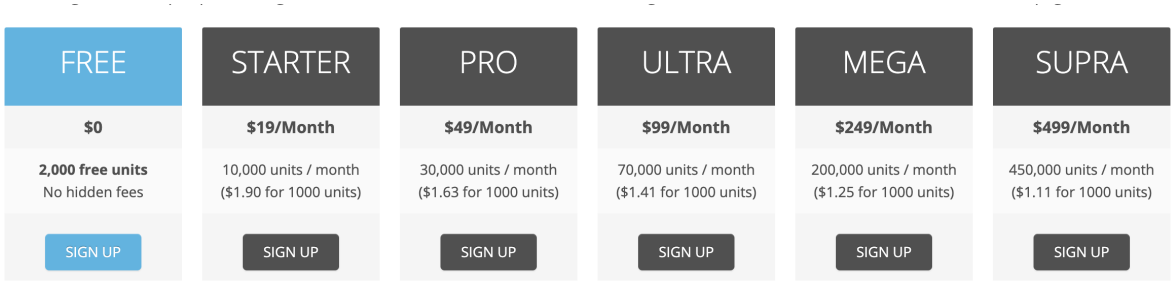
```
1                {
2                   "status":"success",
3                    "final_decision":"KO",
4                    "confidence_score_decision":0.99155,
```

```
                    "gun_moderation":{
                       "confidence_score":0.99921,
                       "compute_time":0.109,
                       "gun_content":true
                    },
                    "media":{
                       "url_image":"http://url_image_to_analyse",
                    },
                    "total_compute_time":0.109
                }
```


Como se puede observar, se ha llevado a cabo un escaneo buscando armas y el parámetro "final_decision" es "KO", lo que significa que estas han sido encontradas en la imagen proporcionada. De haberse encontrado "OK" significaría que no se han encontrado.

Mediante este parámetro es, como decidimos en el backend si un tweet cumple una norma de monitorización o no. En caso de "KO" se incluye como alerta y se notifica. En caso de OK el tweet no cumple con la norma por lo cual no se incluye ni hace saltar la alarma.

Picpurify es de pago, sin embargo, al registrarnos nos ofrece una prueba para escanear 2000 imágenes gratis, por lo que cualquiera podría integrarlo en Tweethawk para un uso moderado. En caso de necesitar más, podemos consultar los planes en Precios Picpurify



5.1.7. Integración con Slack para el envío de notificaciones a dispositivos móviles

Como hemos mostrado antes en la documentación de usuario, podemos enviar una notificación con el contenido del evento a un grupo de Slack, de tal manera que podamos recibir la notificación en otros dispositivos, como móviles, y pudiendo compartirlo con un grupo de

personas.

Tras seguir los pasos mencionados anteriormente en la documentación de usuario, en el cual creamos un Bot de Slack y recibimos un token, el cual insertamos en la configuración del grupo, empleamos dicho token para autenticarnos en la API y enviar mensajes.

Para realizar esto, hemos empleado el método chat.postMessage de la API de Slack. Cada vez que un tweet coincida con una de las reglas de monitorización, se hará la siguiente petición POST:

```
1
2                   POST https://slack.com/api/chat.postMessage
3                   Content-type: application/json
4                   Authorization: Bearer [TOKEN DE SLACK]
5                   {
6                       "channel": "[ID DEL CANAL]",
7                       "text": "[URL DEL PERFIL] - [CONTENIDO DEL TWEET] [
    ENLACE AL TWEET]"
8                   }
9
```

Enlace a la documentación de la API de Slack para enviar mensajes: https://api.slack.com/messaging/sending

Enlace a la documentación del método chat.postMessage: https://api.slack.com/messaging/sending

5.1.8. Creando el servidor

En este apartado explicaremos los pocos pasos que tiene crear el servidor de Tweethawk con el código fuente. Es una tarea realmente sencilla:

- Debemos añadir al fichero secrets.py las claves OAUTH de la API de Twitter y la API KEY de PicPurify, las cuales hemos explicado previamente de donde obtener.
- A continuación, instalaremos el fichero de requisitos con las librerías de Python:

```
1 python3 -m pip install -r requirements.txt
2
```

- Creamos una base de datos de nombre Tweethawk, en el puerto 3306 con el fichero añadido en la carpeta raíz “tweethawk.sql”.

```
cat tweethawk.sql | mysql -u root tweethawk

```

- Ejecutamos el servidor de flask desde el directorio en el que se encuentra app.py de la siguiente manera:

```
set FLASK_APP=app.py
python3 -m flask run

```

5.2. Front-end: interfaz gráfica y experiencia de usuario

5.2.1. La aplicación WEB

En este apartado, realizaremos un breve repaso por todas las particiones disponibles en la aplicación.

- **/monitor/groups/** - GET: Muestra todos los grupos de monitorización creados.
- **/monitor/groups/create** - POST: Crea un grupo de monitorización.
- **/monitor/groups/auto** - POST: Activa el motor de monitorización de la plataforma.
- **/monitor/groups/<int:groupID>/** - GET: Muestra los resultados de un grupo en concreto dado un ID.
- **/monitor/groups/<int:groupID>/auto** - POST: Activa el modo de monitorización automática para un grupo en concreto.
- **/monitor/groups/<int:groupID>/users/** - GET: Muestra los usuarios añadidos a un grupo en concreto dado un ID.
- **/monitor/groups/<int:groupID>/users/add** - POST: Añade un usuario a un grupo de monitorización en específico dado un ID.
- **/monitor/groups/<int:groupID>/rules/** - GET: Muestra las normas de monitorización añadidos a un grupo en concreto dado un ID.

- **/monitor/groups/<int:groupID>/rules/add** - POST: Añade una norma de monitorización a un grupo de monitorización en específico dado un ID.
- **/monitor/groups/<int:groupID>/run** - POST: Ejecuta una monitorización manual para un grupo en específico dado un ID.
- **/monitor/groups/<int:groupID>/settings** - GET: Muestra el panel de configuración de un grupo en concreto dado un ID.
- **/monitor/groups/<int:groupID>/settings** - POST: Añade el token de Slack a la configuración de un grupo dado un ID.

5.2.2. Flask, Html, Javascript: Añadiendo dinamismo a la UI

Hemos empledo las plantillas dinámicas de Flask para poder tener una web no estática. Pasámos como parámetros las listas de datos, y mediante pseudocódigo los colocamos a través de la plantilla que hemos creado previamente en HTML.

Este HTML utiliza CSS y JS para poder ofrecer una experiencia de usuario agradable. Hemos dedicado bastante tiempo a hacer las cosas sencillas y agradables a la vista. Tenemos numerosas utilidades que emplean Javascript para mostrar Pop-ups con formularios, o enviar peticiones al servidor si tener que recargar página constantemente. También empleamos Bootstrap, así como Google fonts para títulos y demás cabeceras.

Capítulo 6

Conclusión

6.1. Conclusión

Durante el desarrollo de este proyecto hemos comprobado la dificultad que supone seguir todas las novedades que nos interesan en Twitter de forma manual, debido a la gran cantidad de contenido que se publica en dicha red a cada momento. Sin embargo, automatizando este proceso no solo podemos ahorrar mucho tiempo de trabajo, si no que podemos disponer de información en tiempo real.

En la era de la información, disponer de esta en el momento en el que se produce es un aspecto crucial para numerosos campos: Periodismo, legal, marketing, militar etc.

Mediante Tweethawk hemos podido observar como en numerosos casos de uso hemos sido capaces de mejorar los procesos de obtención y procesamiento de información extraída de redes sociales. Nos han permitido tener una noción clara de diferente situaciones propuestas según se desenvolvían los hechos: Conflictos internacionales, ciberacoso y radicalización, opinión de un público específico.

Estos son tan solo un pequeño espectro de lo que se puede monitorizar, pero las posibilidades son tan infinitas como la imaginación del que use el software.

Conclusion

During the development of this project we have verified the difficulty of following all the interesting news on Twitter manually, due to the large amount of content that is published there each minute. However, by automating this process we can not only save a lot of time: we can also have information in real time in any device.

In the information age, having it at the moment it is produced is a crucial aspect for many fields: Journalism, legal, marketing, military, etc.

Through Tweethawk we have observed how in numerous use cases we have been able to improve the processes of obtaining and processing information extracted from social networks. It has allowed us to have a clear notion of different situations proposed as the events unfold: International conflicts, cyberbullying and radicalization and the opinion of a specific public, among others.

These are just a small spectrum of what we can monitored, but the possibilities are as endless as the imagination of the user of the software.

6.2. Trabajo Futuro

Durante el uso de la aplicación nos hemos encontrado ciertas limitaciones que de no haberse dado hubiésemos tenido incluso mejores resultados.

También hemos encontrado otros posibles avances que no han sido detectados por ser una limitación, si no por el hecho de que pueden ser útiles o arrojar resultados curiosos.

Poder detectar caracteres de abecedarios No Latinos

Ahora mismo, Tweethawk nos permite solo analizar tweets en UTF-8, es decir, alfabeto Latino. Una nueva versión debería poder permitir monitorizar otros alfabetos, como el alfabeto árabe, el cirílico, alfabeto hebreo etc.

Traducir tweets

Unido al punto anterior, proponemos que una de las posibles mejoras sea permitir seleccionar un idioma de traducción sobre el que trabajar. Nos hemos encontrado muchas situaciones en las que no podemos monitorizar tweets por el simple hecho de no conocer el idioma en el que hablan. Esto podría llevarse a cabo con la API de Google Translator.

Durante la monitorización del conflicto afgano, por barreras del lenguaje solo hemos podido añadir a cuentas angloparlantes.

Por ejemplo, si pudiésemos traducir un tweet escrito en árabe de forma automática al inglés, y una vez allí aplicar las normas podríamos saltarnos esta barrera del lenguaje.

Detectar preguntas, intención positiva o negativa.

El proyecto da pie a poder ser integrado con machine learning e interpretación del lenguaje. De realizarse, podríamos no solo detectar por texto o imagen, si no por forma de hablar, si transmite un mensaje positivo o negativo, o si hace una pregunta.

Esto en sí podría ser temática de un TFG entero debido a la complejidad del asunto.

Filtros, gráficas y generación de estadísticas

Sería conveniente poder filtrar por fechas, por usuarios, por normas, al igual que sería conveniente poder generar gráficas con estadísticas para representar los datos de manera visual.

Si bien es cierto que los tweets en una lista transmiten información, el formato no nos permite ver “el todo”. Mediante gráficas podríamos detectar tendencias, como picos de actividad, usuarios que más resultados han generado, normas más coincididas, etc que nos permiten extraer aún mas datos para un posible informe.

Capítulo 7

Bibliografía

Bibliografía

[1] Smith, Kit: *60 Incredible and Interesting Twitter Stats and Statistics*. https://www.brandwatch.com/blog/twitter-stats-and-statistics/.

[2] Corporation, Mozilla: *Expresiones Regulares*. https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Regular_Expressions.

[3] Help Center, Slack: *Cómo crear un bot para tu espacio de trabajo*. https://slack.com/intl/es-es/help/articles/115005265703.

[4] Wikipedia: *Guerra de Afganistán (2001-2021)*. https://es.wikipedia.org/wiki/Guerra_de_Afganistán_(2001-2021).

[5] Wikipedia: *Acuerdo de Doha (2020)*. https://es.wikipedia.org/wiki/Acuerdo_de_Doha_(2020).

[6] Mundo, BBC News: *EE.UU. y el Talibán firman un histórico acuerdo que prevé la retirada de todas las tropas estadounidenses de Afganistán*. https://www.bbc.com/mundo/noticias-internacional-51689432.

[7] Wikipedia: *Estado Islámico del Gran Jorasán*. https://es.wikipedia.org/wiki/Estado_Islámico_del_Gran_Jorasán.

[8] Wikipedia: *Retirada de tropas estadounidenses de Afganistán*. https://es.wikipedia.org/wiki/Retirada_de_tropas_estadounidenses_de_Afganistán.

[9] Foreign Relations, Council on: *Council on Foreign Relations*. https://www.cfr.org/timeline/us-war-afghanistan.

[10] CHDS: *K-12 School Shooting Database*. https://www.chds.us/ssdb/data-map/.

[11] Peyer, Robin de: *Nikolas Cruz Instagram: 'Disturbing' social media accounts show Parkland school shooting suspect's terrifying weapons arsenal*. https://www.standard.co.uk/news/world/

nikolas-cruz-disturbing-instagram-accounts-show-parkland-school-shooting-suspect-html.

[12] Blancarte, Oscar: *Data Access Object (DAO) Pattern.* https://www.oscarblancarteblog.com/2018/12/10/data-access-object-dao-pattern/.

[13] Center, Twitter Developer: *The U.S. War in Afghanistan.* https://developer.twitter.com/en/docs/twitter-api/.

[14] Roesslein, Joshua: *Tweepy Documentation.* https://docs.tweepy.org/en/stable/.

[15] PicPurify: *PicPurify Official Documentation.* https://www.picpurify.com/api-services.html.

[16] PicPurify: *PicPurify White Papers.* https://www.picpurify.com/white-papers.html.