UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADUADO EN INGENIERÍA DE SOFTWARE

UN ENTORNO PARA ANÁLISIS DE SENTIMIENTOS A PARTIR DE UNA IMAGEN

AN ENVIRONMENT FOR IMAGE SENTIMENT ANALYSIS

Realizado por
John Carlo Purihin Enriquez

Tutorizado por
Eduardo Guzmán De los Riscos

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

MÁLAGA, JULIO DE 2018

Fdo. El/la Secretario/a del Tribunal

Resumen

La era digital está transformando la interacción entre las empresas y sus clientes. El espacio de los usuarios en la Web es cada vez más incluyente. Las redes sociales se han convertido en una herramienta que permite a los clientes expresar opiniones acerca de los productos de una empresa. Sin embargo, un estudio realizado por *IBM institute for Business Value*, confirma que más de la mitad de los Directores de Marketing (CMO) no se encuentran preparados para enfrentar esta explosión de datos.

El proyecto "Un Entorno para Análisis de Sentimientos a partir de una Imagen" se ha desarrollado para ayudar los diferentes tipos de profesionales de márketing en sus tareas de extracción de información y análisis de sentimientos a través de datos obtenidos de las redes sociales.

Para acometer este objetivo, el entorno automatiza el uso de técnicas propias de análisis y visualización de datos que garantizan la calidad de las conclusiones extraídas de los datos, para así poder contribuir en la toma de decisiones de la empresa.

Palabras clave: Análisis de Sentimientos, Análisis de Datos, Procesamiento de Lenguaje Natural, Márketing, Análisis de Imagen

Abstract

The Digital Age is revolutionizing customer-business relationship. The web currently has more room for users than ever before. Social media has become a tool that allows customers to express their opinions about consumer goods. However, a study conducted by *IBM institute for Business Value* stresses that more than half of Chief Marketing Officers (CMO) are not prepared to face this explosion of data.

The project "An Environment for Image Sentiment Analysis" was developed to help Marketing Professionals in fulfilling their tasks of generating insights from social media through data mining and sentiment analysis.

To achieve this goal, the environment automates data analysis and visualization techniques that guarantees the achievement of high quality business intelligence and provides support in companies' decision-making.

Keywords: Sentiment Analysis, Data Analysis, Natural Language Processing, Marketing, Image Analysis

Índice

Resumen .....

Abstract .....

Índice .....

Introducción 1
- **1.1 Motivación** 1
- **1.2 Objetivos** 2
- **1.3 Estructura de la memoria** 2

Resumen Ejecutivo 5
- **2.1 Contexto** 5
- **2.2 Objetivos del Sistema** 5
- **2.3 Suposiciones y Restricciones** 6
- **2.4 Justificación y Análisis de Mercado** 7
- **2.5 Análisis de Riesgos** 10
- **2.6 Estudio de viabilidad** 11
- **2.7 Criterios de Éxito** 12

Definición de técnicas de análisis y visualización 15
- **3.1 Técnicas de extracción de información y análisis de sentimientos** 15
- **3.2 Técnicas de visualización** 22

Fases de Trabajo y Metodología 27
- **4.1 Fases de Trabajo** 27
- **4.2 Metodología** 30

Especificación de Requisitos 33
- **5.1 Requisitos Funcionales** 33
- **5.2 Requisitos no funcionales** 37
- **5.3 Criterios de validación** 38

Modelado y Diseño del Sistema 39
- **6.1 Modelo Estructural** 39
  - 6.1.1 Diagrama de clases 39
  - 6.1.2 Diagrama de componentes 44
- **6.2 Modelo de Comportamiento** 49
  - 6.2.1 Diagrama de Casos de Uso 49
  - 6.2.2 Diagrama de Flujo de Procesos 52
- **6.3 Modelo de Interacción** 56
  - 6.3.1 Diagrama de Secuencias 56

Implementación 59
- **7.1 Tecnologías** 59

7.2 Tecnologías Alternativas Consideradas **63**
7.3 Herramientas Utilizadas y Lenguajes de Programación **64**
7.4 Aspectos Interesantes de la implementación **66**
7.4.1 El Servicio Web 66
7.4.2 Análisis de datos 68
7.4.3 Aplicación Android 68
Mantenimiento y Pruebas **73**
8.1 Pruebas de Unidad **73**
8.2 Pruebas de Usabilidad **75**
8.3 Mantenimiento de Software **80**
Conclusiones **83**
9.1 Resultados **83**
9.2 Conocimientos adquiridos **84**
9.3 Dificultades encontradas **84**
9.4 Posibles mejoras **85**
Referencias **87**
Planificación **89**
Manual de Instalación **93**
Instalación del Servidor Web: **93**
Instalación de la Aplicación Móvil: **95**
Manual de Usuario **97**
¿Cómo me registro? **97**
Y si ya tengo una cuenta ¿Cómo puedo iniciar de sesión? **99**
¿Cómo puedo realizar un análisis de imagen? **100**
Seleccionando una imagen de la galería de imágenes 100
Sacando una foto del objeto que desea analizar 102
¿Cómo puedo ver las representaciones gráficas que permiten visualizar los resultados del análisis? **104**
A partir del “Dashboard” 104
A partir de la lista de análisis 106
¿Cómo analizar un usuario? **107**

1

Introducción

En este capítulo de introducción se describe la motivación del proyecto, el objetivo que se pretenden alcanzar en esta memoria y se da a conocer la estructura del documento. La finalidad de este capítulo es que el lector tenga una idea general del proyecto y del contenido de la memoria.

1.1 Motivación

El proyecto "Un entorno para análisis de sentimientos a partir de una imagen" surge como una propuesta de idea para desarrollar una línea de trabajo del tutor de Trabajo Fin de Grado (TFG) del autor.

La motivación principal de este entorno es apoyar el estudio de viabilidad comercial de los productos y/o servicios de una empresa mediante la sugerencia de información útil acerca del mercado. Esta información abarca desde las posibles competencias del producto, sus características más destacadas, hasta el perfil de los clientes actuales y potenciales de la empresa.

Esta motivación dio lugar a la creación de una herramienta centrada en el análisis de sentimientos capaz de generar inteligencia empresarial mediante técnicas de análisis de datos que esté al alcance de todos.

1.2 Objetivos

En esta memoria se da a conocer toda la información relativa al proyecto software elaborado.

Para lograr esta finalidad, la memoria comienza con la definición del proyecto, las necesidades que debe satisfacer, los fundamentos y conceptos en los que se basa, y los beneficios que se pueden obtener del mismo.

Posteriormente, define un plan de desarrollo del sistema utilizando conceptos, técnicas, herramientas y metodologías propias de la ingeniería de software. Esto implica la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software.

Este plan de consecución abarca todos los aspectos de la producción del software, desde las etapas iniciales de la especificación del producto, la implementación del sistema, hasta el mantenimiento del mismo.

La memoria termina exponiendo una serie de conclusiones relativas a la ejecución del plan de proyecto y las dificultades encontradas.

1.3 Estructura de la memoria

Esta memoria está constituida por 9 capítulos. A continuación, se describe brevemente los contenidos de cada uno de ellos.

Capítulo 2: Resumen Ejecutivo

En este capítulo se realiza la contextualización del proyecto para explicar la problemática y las necesidades que pretende cubrir el sistema. Seguidamente, se explican los principales beneficios del sistema y se lleva a cabo una comparación del mismo con las herramientas existentes en el mercado. Posteriormente, se identifican los riesgos que podría afectar el desarrollo del proyecto, y se elaboran mecanismos de mitigación y contingencia para reducir sus consecuencias. Finalmente, se definen los criterios de éxito del proyecto.

Capítulo 3: Definición de técnicas de análisis y visualización

En este capítulo se define todos los conceptos necesarios para llevar a cabo la extracción de información y el análisis de sentimientos. Esto incluye tanto las técnicas de análisis como los diferentes gráficos para visualizar sus resultados.

Capítulo 4: Fases de Trabajo y Metodología

En el capítulo 4, se justifica la metodología de trabajo que se utilizará para llevar a cabo el proyecto y se expone en detalle todas las fases de elaboración del proyecto.

Capítulo 5: Especificación de Requisitos

El capítulo 5 explica todos los requisitos funcionales y no funcionales del proyecto en profundidad, y establece la relación existente entre ellos a través de una matriz de trazabilidad. Por último, este capítulo también detalla los criterios de validación del software que se elaborará en este proyecto.

Capítulo 6: Modelado y diseño del sistema

En este capítulo se desarrollan los modelos necesarios para implementar el sistema y entender correctamente sus funcionalidades. El capítulo 6 comienza con el modelado estructural para definir la estructura estática de los objetos del sistema. Seguidamente, presenta el modelo de comportamiento con el fin de explicar el comportamiento dinámico de todos elementos del sistema. Por último, lleva a cabo el modelado de interacción para mostrar cómo los objetos interactúan entre sí y el orden en que se producen esas interacciones.

Capítulo 7: Implementación

El capítulo 7 explica todas las tecnologías y herramientas utilizadas para elaborar el sistema. En este capítulo se justifica todas las decisiones de implementación que se han llevado a cabo para desarrollar el servicio web y la aplicación móvil.

Capítulo 8: Mantenimiento y pruebas

En este apartado de la documentación se explica en detalle los resultados de las pruebas de realizadas sobre el sistema. Para ello, se expone información acerca de los resultados de las pruebas de unidad del servidor, y se describe los pasos acometidos para llevar a cabo las pruebas de usabilidad sobre la interfaz de usuario. Por último, se describe cómo se recomienda llevar a cabo la tarea de mantenimiento software del sistema.

Capítulo 9: Conclusiones

En el último capítulo de la memoria se identifican cuáles son los objetivos cumplidos en el proyecto, y se enumeran todas las dificultades encontradas durante el proceso de elaboración del proyecto.

Anexos

Planificación

En el anexo de planificación se muestra una tabla en el que se enumeran todas las tareas que hay que llevar a cabo y el tiempo estimado necesario para llevarlos a cabo.

Manual de Instalación

En este anexo se describen los pasos que hay que seguir para instalar el sistema.

Manual de Usuario

Este anexo explica detalladamente cómo se puede utilizar la aplicación y enumera cuáles son las diferentes interacciones que puede llevar a cabo el usuario.

2

Resumen Ejecutivo

2.1 Contexto

Los resultados de un estudio realizado por la famosa empresa de consultoría McKinsey & Company acerca de la compraventa de productos/servicios nos indica que más del 85% de los clientes compran más del mismo producto después de una experiencia positiva. Y en cambio, el 75% de los clientes que han tenido una experiencia negativa compran menos. Este estudio demuestra un hecho evidente: proporcionar un producto/servicio que satisfaga a nuestros clientes es una clave del éxito de nuestra empresa. De ahí surgen las preguntas: ¿Existe alguna forma de conocer los gustos del cliente sin tener que recurrir a suposiciones? ¿Hay alguna herramienta para conocer los gustos del cliente? ¿Esta herramienta está al alcance de todos? ¿Tengo los conocimientos necesarios para manejar esta herramienta?

2.2 Objetivos del Sistema

Para satisfacer a las necesidades planteadas en el apartado anterior se desarrollará un sistema que permite realizar un estudio de los gustos y los sentimientos del cliente automatizando técnicas de extracción de información y de análisis sentimental y de tendencias. Para ello, se implementará una aplicación “Back-End” con un servicio web que, al recibir una imagen como parámetro de petición al servidor, devuelva un conjunto

de resultados obtenidos, a partir de un análisis realizado sobre dicha imagen alimentándose de información obtenida en redes sociales.

El resultado puede responder a las siguientes preguntas: ¿Qué es lo que piensa los clientes acerca del producto? ¿Quiénes son las personas más interesadas en el producto/servicio? ¿Qué características del producto son las que más le interesa al cliente? ¿Qué es lo que no le interesa del producto/servicio? ¿Qué características del producto está siendo discutido? ¿Cuál es el principal competidor del producto? ¿Qué otro producto/servicio suele comprar dichas personas? ¿En qué momentos del día se suele hablar del producto?

Para acceder este servicio con más facilidad, se aprovecharán las cámaras que tienen los teléfonos móviles y se desarrollará una aplicación móvil de tal manera que si un usuario desea utilizar el servicio, únicamente tendrá que sacar una imagen del producto que desea analizar mediante la aplicación, y esta se encargará de trabajar con el servicio web para obtener los resultados del análisis y mostrarlo al usuario mediante diferentes tipos de gráficos que facilitarán la comprensión de los mismos.

En resumen, el sistema tiene la siguiente *lógica de negocio*:
- Capacidad para reconocer el contenido de una imagen.
- Mecanismo para identificar el público objetivo de un producto.
- Posibilidad de poder determinar las competencias que tiene el producto.
- Identificación de las características más interesantes del producto
- Aplicación móvil que facilite la interpretación de los resultados
- Herramienta para analizar los gustos de un usuario.

2.3 Suposiciones y Restricciones

Para llevar a cabo este proyecto hay que tener en cuenta la disponibilidad de herramientas, métodos y mecanismos necesarios para implementar el servicio web, llevar a cabo el análisis y representar los diferentes gráficos. Por un lado, hay que destacar el hecho de que existen muchos recursos,

documentación y marcos de trabajo para implementar un servicio web. Por otro lado, también se dispone de múltiples lenguajes, entornos para computación científica y diversos algoritmos y métodos para la minería y el análisis de datos. Por último, hay que decir que existe una gran variedad de librerías para la visualización de datos, sin embargo, solamente una pequeña porción de ellas está a la disponibilidad de aplicaciones móviles nativas. Teniendo esto en cuenta, es muy probable que no haya considerable problema a la hora de implementar el servicio, dada la gran variedad de herramientas y técnicas disponibles y, quizás, la única preocupación es la de encontrar librerías o marcos de trabajo que hagan representaciones gráficas muy adecuadas para el resultado de nuestro análisis.

Las principales limitaciones o restricciones que afectan al desempeño de este proyecto son el tiempo, puesto que hay que cumplir la limitación temporal del trabajo fin de grado (296 horas); el alcance del proyecto, dado que el éxito del mismo dependerá fundamentalmente de las propias habilidades del desarrollador del mismo tanto en el ámbito de ciencias de datos e ingeniería como en su capacidad de llevar a cabo planes y procesos; y el costo, que principalmente se apreciará en el uso de diversos servicios existentes que utilizaremos para realizar nuestro proyecto (i.e. uso de API de terceros).

2.4 Justificación y Análisis de Mercado

El principal beneficio que ofrece este sistema con respecto a los demás es la automatización de análisis a partir del concepto extraído de una imagen. Habitualmente, la minería de datos y extracción de información implican recopilar, almacenar y analizar datos con el fin de descubrir nueva información acerca de la misma. Sin embargo, mediante esta aplicación todo este proceso se realiza de forma transparente al cliente. Por otro lado, hay que reconocer que existen algunas herramientas que permiten realizar una tarea similar (i.e. *sentiment viz*). Aunque dichas herramientas necesitan tener un conocimiento del concepto que deseamos analizar, por lo que la gran ventaja que ofrece nuestro sistema es, por una parte, el reconocimiento del contenido de una imagen y, por otra, la visualización de los datos mediante

una aplicación móvil. Este último aspecto es una característica que la diferencia del resto de las aplicaciones que actualmente existen en el mercado, ya que hoy día no existe una aplicación móvil que proporcione la misma cantidad y calidad de información como lo hace este sistema.

A continuación, se menciona algunos ejemplos de uso:

- Marketing

El sistema puede ayudar a las empresas de marketing a crear modelos basados en datos históricos para predecir quién responderá a las nuevas campañas de marketing como el correo directo, la campaña de marketing en línea, etc. Mediante esta información, los especialistas en marketing serán capaces de tener un mejor enfoque de las características del mercado y, con ello, tienen la posibilidad de vender productos que pueden generar beneficio a su empresa. Por ejemplo, si a través del análisis que proporciona el entorno determinan que si una persona compra tabaco existe mucha posibilidad de que compre una lata de cerveza, entonces se puede lanzar una oferta tabaco-cerveza con el fin de mejorar los ingresos de la empresa.

- Estructura de una tienda

Las tiendas minoristas utilizan los hábitos/detalles de compra de los clientes para optimizar el diseño de sus tiendas a fin de mejorar la experiencia del cliente y aumentar las ganancias. Dichos hábitos/detalles de compra se pueden determinar a través de este entorno. Por ejemplo, si llegan a la conclusión de que las personas que compran frutas también compran pescado, entonces puede organizar los productos de su tienda de tal manera que hay bastante proximidad entre pescado y frutas.

- Industria (Fabricación)

La extracción de información que proporciona este entorno de análisis permite a las fábricas determinar los principales productos que deben fabricar teniendo en cuenta las necesidades del mercado y, por supuesto, la eficiencia de su trabajo y los beneficios que aportan dichos productos. Por ejemplo: Si una fábrica determina que el componente electrónico *SG90 Micro Servo* tiene mucha demanda, entonces se podrá generar una planificación

más eficiente del proceso de producción orientado producir más cantidad de dicho componente para satisfacer las demandas.

Para terminar este apartado, se analizan los principales competidores del entorno de análisis destacando cuáles sus principales características y las ventajas que pretende mejorar el sistema que se desarrolla.

Nombre	Características	Desventajas
Sentiment Viz	- Aplicación web que proporciona muchos tipos de análisis y presenta buenas técnicas de visualización: sentiment graph, heatpmap, topics, tag cloud, timeline, map, etc	-El usuario tiene que conocer el concepto que está analizando. -Es una aplicación web que solamente se puede utilizar mediante un navegador de escritorio (no se usa fácilmente en el tfno. móvil).
Sentiment Analysis for Twitter	- Aplicación móvil que permite obtener un análisis en tiempo real devolviendo los resultados a través de un gráfico de geoespacial.	- Proporciona un solo tipo de resultado.
HappyGrumpy	- HappyGrumpy es una herramienta de análisis de emociones y sentimientos que le otorga la calificación de "Feliz / Gruñón" (y una variedad de otras estadísticas) para su cuenta de Twitter, sus celebridades favoritas o cualquier otra persona. Incluso puede analizar menciones de marcas y temas en Twitter. Simplemente escriba cualquier cuenta de Twitter o nombre de marca y aparecerá una gran cantidad de datos. - Aplicación móvil	- Está muy orientado al análisis de las relaciones existentes entre personas. - No tiene soporte. No ha recibido una actualización desde 31 de julio de 2015.
FinSentS	FinSentS significa Financial News y Sentiment Screener, es una herramienta de análisis de noticias de vanguardia que cubre más de 40,000 acciones, Forex, materias primas, así como los principales índices (Nasdaq, SP500, Euro Stoxx, Dax, CAC, STI, Nikkei, Ftse , HSI, Sensex ...)	- Es una herramienta muy orientada al análisis financiero

Tabla 2.1 Análisis de la competencia

Como se puede observar, todas las herramientas analizadas suelen tener un propósito muy específico y proporcionan resultados que solamente puede tener validez para un determinado tipo de usuarios. Quizás el principal competidor del sistema es Sentiment Viz, ya que es la única herramienta capaz de proporcionar un análisis general. Y desde ese punto de vista, podemos decir que nuestro entorno tiene los medios necesarios para llevar a cabo una buena competencia, sobre todo en el aspecto de la experiencia de usuario (UX).

2.5 Análisis de Riesgos

En este apartado, identificaremos los posibles problemas que pueden ocurrir durante el proceso de desarrollo e intentaremos ofrecer un mecanismo de tratamiento o plan de contingencia.

Riesgo	Probabilidad de Ocurrencia	Consecuencias	Tratamiento/Plan de Contingencia	Mecanismos de Monitorización
No cumplir la planificación de desarrollo software	Media	Media - Retraso en el desarrollo del proyecto	-Diseñar una planificación efectiva. -Determinar el camino crítico del proyecto -Tener holgura libre	-Realizar informes acerca de las actividades (o hitos) del proyecto
Problemas en el desarrollo debido a la falta de experiencia	Media	Alta - Posible degradación de la calidad del sistema	-Estudiar todos los conceptos necesarios antes de comenzar el proyecto	-Identificar si los conceptos estudiados son suficientes para llevar a cabo una actividad
Problema con la tecnología usada en el desarrollo (bugs, errores de configuración...)	Baja	Baja - Retraso en el desarrollo del proyecto	-Usar tecnologías alternativas	-Visitar durante un periodo de tiempo los foros oficiales de la tecnología para identificar los posibles bugs.
Necesidad de cambiar los requisitos para llevar a cabo adecuadamente el proyecto	Baja	Baja - Modificación del alcance	-Disponer de holgura libre -Analizar adecuadamente las necesidades del proyecto	

Límites o cuotas en el uso de APIs de terceros	Media	Media - Modificación del alcance	-Tener diferentes medios para asegurar que el usuario final no se vea afectada por límites de solicitudes a la API	
Subestimar el tamaño de la Especificación de Requisitos (ERS).	Baja	Baja - Retraso en el desarrollo del proyecto	-Disponer de holgura libre	-Realizar informes acerca del progreso del proyecto

Tabla 2.2 Análisis de Riesgos

2.6 Estudio de viabilidad

En este apartado se estudia cuatro áreas principales de interés: la viabilidad operativa, la viabilidad técnica, la viabilidad de fechas y la viabilidad económica (García, 2008).

a) Viabilidad Operativa

El principal interés de los usuarios del sistema es generar valor a partir de los resultados de análisis. Por este motivo, si se garantiza que el sistema siempre proporcione información de alta calidad se puede decir con certeza que dicho sistema satisface las necesidades de los usuarios.

b) Viabilidad Técnica

Teniendo en cuenta las restricciones, suposiciones y los riesgos que se han analizado en el aparto anterior se puede afirmar que se dispone de los medios necesarios para implementar todas las funcionalidades del sistema, y que realmente el principal incentivo de este proyecto son los conocimientos del desarrollador y la disponibilidad de las herramientas que se van a utilizar.

c) Viabilidad de fechas

Teniendo en cuenta las diferentes funcionalidades del sistema que hay que implementar y el tiempo disponible para llevar a cabo el trabajo fin de grado, 296 horas, se puede afirmar que se dispone de tiempo suficiente para llevar a cabo el proyecto. Esto es, porque se va a diseñar un sistema para realizar el proyecto en ese

plazo de tiempo y, por supuesto, también se han tenido en cuenta que algunas actividades del proyecto pueden generar retraso.

d) Viabilidad Económica

A pesar de que ser trabajo académico, no se puede evitar las diferentes consecuencias económicas que tiene el proyecto si realmente se desea ponerlo en marcha.

Los principales costos del proyecto se corresponden con el uso de la API Vision de Google:

Número de Llamadas	Precio
1000	gratis (hasta las 1000 primeras peticiones)
1001 a 5.000.000	1,26€
5.000.001 a 20.000.000	0,84€

Tabla 2.3 Coste del uso de la API VIsion

2.7 Criterios de Éxito

Hay una tradición en la gestión de proyectos software que establece que un proyecto se considera exitoso cuando se cumple en plazo, sin pasarse del presupuesto y con las funciones requeridas implementadas. Sin embargo, un estudio realizado por The Standish Group llamado informe Chaos señala que solamente el 29% de los proyectos software tiene éxito. Este informe indica que, aunque casi el 52% de los proyectos consiguen implementar todas (o casi todas) las funcionalidades solamente un cierto porcentaje de estos son los que finalmente se consideran exitosos, esto es porque dichos proyectos no atraían usuarios potenciales de la aplicación, con lo que no implicaron beneficios económicos para los desarrolladores del proyecto.

Según este informe las principales causas que llevan los proyectos al fracaso son: la falta de participación del usuario, los requisitos incompletos y los cambios frecuentes de requisitos. Mientras que los principales factores de éxito son: la integración de usuarios finales en el proceso de desarrollo, el soporte de personas con experiencia y una adecuada definición de requisitos.

Teniendo en cuenta esto, para realizar este proyecto se va a asegurar que se definen adecuadamente las características del proyecto para así describir adecuadamente

los requisitos del mismo. Esto siempre se llevará a cabo teniendo en cuenta la opinión de un experto, para así, asegurar que el proyecto se esté llevando adecuadamente.

Por otra parte, hay que reconocer que es difícil tener en cuenta la perspectiva de los usuarios del sistema durante el proceso de desarrollo, pues los asuntos que preocupan más al usuario son el uso del sistema y el valor que esto podría generar para su empresa, mientras que para el desarrollador lo más importante es finalizar el proceso de desarrollo. De ahí, que después de implementar todas las funcionalidades del sistema realizaremos una evaluación de la utilidad del sistema, asegurando que tanto el servicio web cómo la aplicación Móvil son capaces de generar valor para una empresa.

3

Definición de técnicas de análisis y visualización

3.1 Técnicas de extracción de información y análisis de sentimientos

A continuación se definen todos los análisis que llevará a cabo el sistema y se enumeran los posibles casos de uso de los mismos:

a) Usuarios más implicados: indica quiénes son los usuarios más relacionados con la imagen.

Ejemplo:

Usuario	Frecuencia de Participación
@JohnSmith	10
@MarySmith	6
@HannahSmith	5

Tabla 3.1 Ejemplo de usuarios más implicados

Este análisis puede ser interesante para conocer el tipo de público interesado al producto, pues permitiría la segmentación de clientes y, a su vez, permite

conocer el tamaño del mercado. Incluso, da la posibilidad de definir la proyección de ventas, conocer las decisiones de compra, las principales características de los clientes, sus gustos, etc.

b) Frecuencia de palabras: proporciona el número de veces que una palabra relacionada con la imagen aparece en los textos que se han extraído de las redes sociales. Este análisis a su vez permitirá crear una red de términos por asociación de palabras.

El resultado que devuelve este análisis es un "Document-term matrix" que es una matriz que describe la frecuencia en la que ocurre una palabra dentro de un documento que, en este caso, serán datos extraídos de *Twitter*.

El análisis multivariado del "Document-term matrix" permite identificar los tópicos del corpus y la relaciones entre las palabras. Esta técnica es lo que habitualmente se utiliza para realizar estudios de comercio teniendo en cuenta los efectos de todas las variables en las respuestas de interés. Se podría afirmar que el resultado de este análisis es la base del sistema.

Ejemplo:

Palabra	Ocurrencia
iPhone	50
iOS	33
Camera	30
Nexus	30
Galaxy	28

Tabla 3.2 Ejemplo de frecuencia ocurrencia de palabras

Este análisis nos permitiría conocer las características más interesantes del producto e incluso proporciona la posibilidad de comprender en profundidad nuestros principales competidores, para así poder elaborar una estrategia de marketing.

c) Palabras relacionadas "n-gram" y correlaciones: construye un modelo de relaciones entre palabras analizando la frecuencia con que una palabra Y

antecede a la palabra X, siendo n el número de palabras que se quiere capturar en cada "n-gram" (Broder, Glassman, & Zweig, 1997).

Un modelo "n-gram" es un tipo de modelo probabilístico que permite predecir el siguiente elemento en dicha secuencia en la forma de un modelo de Markov (n-1), proceso estocástico discreto en el que la probabilidad de que ocurra un evento depende solamente del evento inmediatamente anterior.

Ejemplo de un bigrama:

Palabra 1	Palabra 2
iPhone	X
Galaxy	Note
Pixel	2

Tabla 3.3 Ejemplo de "n-gram"

El conjunto de *bigrama* que se ha obtenido también se puede tratar como términos en un documento, lo cual nos permitiría mejorar el resultado de nuestro análisis.

Ejemplo:

Palabra	Ocurrencia
iPhone X	30
Galaxy Note	25
Pixel 2	24

Tabla 3.4 Ejemplo de frecuencia de ocurrencia de "n-gram"

d) Análisis de Sentimientos ("Opinion Mining"): consiste en clasificar la polaridad de un texto – si la opinión expresada en un texto es positiva, muy positiva, negativa, muy negativa, sorna o neutra (Rouse, 2010). También se realizará análisis de sentimiento más avanzado, "Análisis más allá de la polaridad" en busca de estados emocionales como "enfado", "tristeza", "ira", "miedo" …

El resultado de este análisis es una tupla que contiene por un lado un texto y por otro lado el sentimiento que contenga ese texto. Para realizar este análisis se usará el modelo de "Bolsas de palabras", que es un método que se

utiliza en el procesado del lenguaje para representar documentos/textos ignorando el orden de las palabras.

Para llevar a cabo esta tarea, se utilizarán diccionarios que asocian a cada palabra con un sentimiento, de tal manera que el sentimiento de un texto viene dado por el sentimiento más preponderante.

Estos diccionarios, en análisis de datos, es lo que se suele denominar como "lexicons". Los diccionarios que se van a usar son: *RSentiment* de **Subhasree Bose** y *nrc* lexicons creado por **Saif Mohammad** y **Peter Turney**. Ambos "lexicons" contienen muchas palabras en inglés y tiene una puntuación asociada según si son palabras negativas, positivas, etc. El "lexicón" *RSentiment* clasifican las palabras en positivo, negativo, sorna, muy positiva, muy negativa y neutral. En cambio, el "lexicón" *nrc* clasifican las palabras en función de si son: positivo, negativo, enojo, anticipación, disgusto, miedo, alegría, tristeza, sorpresa y confianza. Estos "lexicons" permitirían evitar los "stop words" o palabras vacías (artículos, determinantes, …) en los textos.

Ejemplo usando *RSentiment*:

Texto	Sentimiento
Good morning Steve!	Positive
I dislike loud people	Negative
I love apples	Very positive

Tabla 3.5 Ejemplo de Análisis de Sentimientos usando RSentiment

Ejemplo usando *nrc*:

Texto	Sentimiento
I hate when people stand close to me on the train.	Anger
I'm scared of heights	Fear
I miss my grandmother	Sadness

Tabla 3.6 Ejemplo de Análisis de Sentimientos usando nrc

La minería de opinión se puede utilizar de muchas formas. Puede ayudar a los especialistas en marketing a evaluar el éxito de una campaña publicitaria o el lanzamiento de un nuevo producto, determinar qué versiones de un producto o servicio son populares, identificar cuáles son las características

demográficas de su público objetivo o "target", y conocer las características más interesantes del producto que antes no se conocía.

Por ejemplo, si la empresa desea conocer cómo se puede mejorar las características del teléfono móvil que están comercializando y dicho teléfono ha recibido muchas reseñas, lo cual dificulta conocer cuáles son sus puntos débiles. La minería de opiniones permitirá saber con exactitud qué funcionalidades de dicho teléfono hay que mejorar: la batería, el System on Chip (SOC) …

Sin lugar a duda, el análisis de sentimiento es el punto más importante de este trabajo, pues proporciona toda información necesaria para posteriormente elaborar una estrategia de marketing, asegurar el éxito de una campaña de negocio, mejorar el servicio de atención al cliente, identificar los indicadores clave del rendimiento y conocer/generar nuevos targets o personas interesadas en el producto.

e) Clustering – Topic Modelling: es un método que clasifica las palabras, de manera que cada las palabras que están interrelacionadas aparecen agrupadas y cada uno de estos agrupamientos representan un tópico diferente.

"Topic modelling" proporciona formas de organizar, comprender y resumir enormes colecciones de información textual. Mediante esta técnica se puede:
- Descubrir temas ocultos que están presentes en la colección de datos.
- Clasificar los textos en función de su tema
- Organizar y resumir textos.

Este análisis encuentra un grupo de palabras que mejor represente la información que aparece en un conjunto de datos, es decir, es una forma de obtener patrones recurrentes de palabras en material textual.

Usaremos el método **Latent Dirichlet Allocation (LDA)** para ajustar un *Topic Model*, pues esta técnica permitirá tratar cada tópico de un texto como

una combinación de palabras, es decir, LDA considera que cada palabra en el documento es atribuible a uno de los temas del documento.

Ejemplo:

Número	Texto
1	I love pineapple on pizza
2	Cats and Dogs are mortal enemies
3	I had neapolitan pizza for lunch
4	Dogs are man's best friend
5	Dogs loves to eat Pizza

Tabla 3.6 Ejemplo de "Topic Modelling"

Tópico 1: pizza, pineapple, neapolitan...

Tópico 2: dogs, cats…

Texto 1 y 3: 100% Tópico 1

Texto 2 y 4: 100% Tópico 2

Texto 5: 50% Tópico 1 y 50% Tópico 2

Este método se implementa mediante el método de "k-means", que es un método de agrupamiento, que tiene como objetivo la partición de un conjunto de n observaciones en k grupos en el que cada observación pertenece al grupo cuyo valor medio es más cercano. Este algoritmo permite clasificar cada texto de nuestro conjunto de datos en k temas diferentes. A continuación, se explica paso a paso el funcionamiento del algoritmo (KDnuggets, 2016):

Paso 1: Se recorre todos los textos y se asigna aleatoriamente cada palabra del texto a uno de los K temas (se elige K de antemano).

Esta asignación aleatoria proporciona una representación temática de todos los textos y distribuciones de palabras de todos los temas, a pesar de que dichos temas no estén adecuadamente asignados.

Paso 2: Para mejorar las asignaciones:

-Para cada texto x, se revisa cada palabra y se calcula:

-p (tema t | texto x): proporción de palabras en el texto x que están asignadas al tema t.

-p (palabra w | tema t): proporción de asignaciones al tema t, sobre todos los textos x, que provienen de la palabra w.

Paso 3: Se reasigna la palabra w a un nuevo tema t ', donde se el tema t' con probabilidad p (tema t '| texto x) * p (palabra w | tema t').
Este modelo generativo predice la probabilidad de que el tema t' genere la palabra w.

Si se repite el último paso una gran cantidad de veces, se alcanzará un estado estable donde las asignaciones de temas son bastante buenas. Estas asignaciones se usarán posteriormente para determinar las mezclas temáticas que tiene cada uno de nuestros textos.

Al igual que en el caso anterior, "Topic Modelling" cumple un papel importante en el sistema. Gracias a ello, los usuarios serán capaces de determinar: cuáles son las características más interesantes de su producto, cuáles son las peores características de su producto, qué características deberían implementar en el futuro, qué características deberían eliminar …

Cada uno de los análisis seleccionados devolverán resultados que puedan ser fácilmente interpretadas por los usuarios. Y a partir de estos resultados, los mismos serán capaces de extraer una serie de conclusiones: público objetivo de su producto, tamaño del mercado, opinión de sus clientes acerca de determinadas características de su producto, principales competidores, qué otros productos suelen comprar sus clientes, etc.

Para terminar este apartado, hay que indicar que se ha intentado recoger los análisis que mejor se adapta a las posibles entradas del programa para proporcionar la mayor cantidad de información acerca del producto/servicio que el usuario desea conocer. Esto es teniendo en cuenta que cuanta más información le proporcionamos al cliente, mayor valor aportará la herramienta, y cuanto más valor aportamos, más usuario tendría el sistema.

Dicho esto, también hay que reconocer que no todos los patrones inducidos mediante técnicas de análisis son necesariamente válidos. Es común que los algoritmos de minería de datos encuentren patrones en la muestra que nos están presentes en otro conjunto de datos. Esto es lo que se denomina sobreajuste. Para ello, se llevará a cabo el siguiente pre-procesado de datos (Botía, s.f.):

- Limpieza de datos("Data cleaning"): que es un proceso que nos permite eliminar datos con ruido o incorrectos eliminando datos que faltan, suavizando el ruido, eliminando datos fuera de rango y corrigiendo inconsistencias (i.e. datos nulos).

- Integración de datos ("Data integration"): trata de integrar diferentes fuentes de datos en un almacén coherente y homogéneo como, por ejemplo, un *data.set* de R.

- Transformación de datos ("Data transformation"): se transforman los datos en datos equivalentes, y se escala un atributo a un conjunto de valores (Normalización).

- Reducción de datos ("Data reduction"): se reduce el tamaño del conjunto de datos mediante la eliminación de datos redundantes (i.e. retweets).

3.2 Técnicas de visualización

Para comprender el resultado de los análisis se ofrece una serie de gráficos para la interpretación, contrastación y comparación de datos que permiten obtener un conocimiento profundo y detallado de los mismos. Los gráficos sirven como una herramienta para ayudar al análisis y son capaces de resaltar las diferencias o correlaciones existentes en nuestro conjunto de datos.

"*En el contexto del marketing digital, la visualización de datos nos ayuda a elaborar mejores cuadros de mando, y en general a comunicar el significado de*

los datos de la manera más adecuada para cada interlocutor. Así mismo, en ámbitos como las redes sociales, que tratan con grandes cantidades de datos, la visualización nos ayuda a generar conocimiento" – **Pere Rovira, experto en visualización de datos**

La visualización de datos constituye un aspecto muy importante en el desarrollo de este proyecto. Por este motivo, se procura que los gráficos cumplan los principios propuestos por **Edward Tufte** en su libro *The Visual Display of Quantitative Information* (Tufte, 1983) . Los gráficos…

- Han de mostrar los datos.
- Inducir a pensar sobre la sustancia más que sobre metodología, diseño gráfico, tecnología de producción gráfica u otra cosa.
- Evitar distorsionar lo que dicen los datos.
- Presentar muchos números en un espacio pequeño.
- Hacer que los conjuntos de datos grandes sean coherentes.
- Alentar al ojo para comparar los diferentes componentes de datos.
- Revelar los datos en varios niveles de detalle, desde una visión general amplía hasta la estructura final.
- Servir un propósito razonablemente claro: descripción, exploración, tabulación o decoración.
- Estar estrechamente integrado con las descripciones estadísticas y verbales de un conjunto de datos.

Los gráficos que se utilizan para presentar la información a los usuarios se corresponden con los gráficos clásicos usados en la minería de datos, esto es, porque la mayoría de los posibles usuarios de nuestro sistema están más familiarizados con estos tipos de gráficos. A continuación, se comentan los detalles más importantes de cada gráfico:

a) Tabla: sirven para mostrar números, siempre se pueden trabajar y hacerlas más fáciles de descodificar a primera vista (con colores, explicación previa).

b) Gráfico de Barra: un gráfico de barra o columna hace énfasis en la comparación entre elementos en un período de tiempo específico.

c) Gráficos de Línea: un gráfico de línea muestra las relaciones de los cambios en los datos en un período de tiempo.

d) Gráfico de sectores: se utiliza para mostrar cómo diferentes partes representan un total.

e) Gráficos de Dispersión: los gráficos de Dispersión o "Scatter Plot" son útiles para mostrar la relación entre diferentes puntos de datos. Este tipo de gráfico utiliza valores numéricos para ambos ejes en lugar de utilizar categorías en alguno de los ejes como en los gráficos anteriores.

g) Nube de palabras: es una representación visual de las palabras que conforman un texto, en donde el tamaño es mayor para las palabras que aparecen con más frecuencia. Estos gráficos nos permiten representar tendencias.

f) "Text Network": es un grafo en el que cada vértice aparece una palabra y cada arista nos indica que hay una relación entre dos palabras en el texto que estamos analizando.

La cognición se refiere a procesos como percepción, atención, aprendizaje, memoria, pensamiento, formación de conceptos, lectura y resolución de problemas. El procesamiento visual humano es eficiente para detectar cambios y hacer comparaciones entre cantidades, tamaños, formas y variaciones en la ligereza. Cuando las propiedades de los datos simbólicos se asignan a las propiedades visuales, los seres humanos pueden navegar a través de grandes cantidades de datos de manera eficiente. Se estima que 2/3 de las neuronas del cerebro pueden estar involucradas en el procesamiento visual. Una técnica de visualización de datos adecuada es capaz de proporcionar un enfoque diferente para mostrar posibles conexiones, relaciones, etc. que no son tan obvias en los datos cuantitativos no

visualizados. La visualización puede convertirse en un medio de exploración de datos. Esto precisamente es el motivo por el cual se han elegido estas técnicas de representación gráfica.

4

Fases de Trabajo y Metodología

4.1 Fases de Trabajo

Las fases de trabajo de este proyecto sigue todo el ciclo de vida del desarrollo de software (SDLC en adelante), que es una secuencia estructurada y bien definida de las etapas de Ingeniería de software para desarrollar el producto software deseado (Veracode, s.f.).

La finalidad de SDLC es diseñar y desarrollar un software de alta calidad. Para ello, define un plan detallado que proporciona un marco de trabajo para desarrollar software y administrarlo a lo largo de todo su ciclo de vida.

SDLC no define una única forma de desarrollar software, pues existen diversas metodologías que sirven para abordar diferentes tipos de problemas. Dichas metodologías generalmente están basadas en el estándar *internacional ISO/IEC 12207*, que establece pautas para el desarrollo y mantenimiento de sistemas de información.

A continuación, se describe las diferentes etapas del desarrollo del proyecto siguiendo un modelo de la SDLC:

Nota: Como se puede observar las fases del proyecto se corresponden con la estructura de la memoria.

a) Especificación de Requisitos (SRS):

Esta fase recoge la definición de todas las características que presenta el sistema basando en las necesidades descritas en el capítulo **2. Resumen Ejecutivo**. Estas características se clasifican en:

-Requisitos funcionales: describe las funcionalidades del sistema.

-Requisitos no funcionales: describe las restricciones de las funcionalidades.

La fase de especificación de requisitos también recoge los criterios de validación del sistema, que establecen la calidad de satisfacción de los requisitos en función de medidas establecidas.

b) Diseño y Modelado de Software

Describe tanto la estructura del sistema como el comportamiento estático y dinámico de los elementos del mismo.

Modelo Estructural:

-Diagrama de Clases: describe las clases del sistema, sus atributos, operaciones y relaciones.

-Diagrama de Componentes: define los diferentes componentes del sistema y las dependencias entre dichos componentes.

Modelo de Comportamiento:

-Casos de Uso: establece las actividades que puede realizar un usuario del sistema.

-Diagrama de Actividades: representa la secuencia de pasos para realizar una tarea.

Modelo de Interacción:

-Diagrama de Secuencias: describe cómo los elementos del sistema interactúan entre sí y el orden en que se producen esas interacciones.

En este proyecto se usa el *Lenguaje Unificado de Modelado (UML)* para elaborar los diagramas.

c) Planificación de las actividades de desarrollo

Para planificar el desarrollo se elabora una estructura de desglose de trabajo (EDT) que contiene una descomposición jerárquica de las actividades del proyecto (Véase el **ANEXO Planificación** de la memoria).

Para realizar el seguimiento del proyecto se utiliza una tabla que contiene las el estado de la actividad y los horas invertidas de cada actividad del proyecto.

d) Implementación

En esta fase se realiza todas las actividades de programación del proyecto siguiendo la metodología de trabajo (que se explicará en el siguiente apartado). En esta fase se desarrollan todas funcionalidades del sistema siguiendo los modelos elaborados.

e) Pruebas

En esta fase se realizan pruebas de unidad sobre las principales funcionalidad del sistema con el fin de conocer su correcto funcionamiento. En esta fase también se realiza una prueba de usabilidad de la interfaz siguiendo la encuesta *System Usability Scale* (SUS).

f) Mantenimiento

La fase de mantenimiento de este proyecto se corresponde con la corrección de errores, la mejora del rendimiento y la refactorización del código.

4.2 Metodología

La metodología de trabajo que se utilizará para seguir el ciclo de vida de desarrollo software es la metodología ágil, *Scrum*. Esto se debe a que el autor del proyecto tiene experiencia en el uso de dicha metodología y la necesidad que tiene el proyecto de tener flexibilidad. A continuación, se describen la correspondencia de las características de la metodología con los aspectos de este proyecto:

a) Product backlog: son las funcionalidades que ha de presentar el sistema. En este proyecto, las *historias de usuario* se corresponden con las funcionalidades que se describen en el capítulo de **5. Especificación de Requisitos**, dichas funcionalidades son el resultado de las ideas propuestas por el autor del proyecto y su tutor de Trabajo Fin de Grado.

b) Sprint Planning: son reuniones para planificar las siguientes actividades del proyecto.
Para llevar a cabo la planificación de las iteraciones del proyecto habrá reuniones cada 1 o 2 semanas. El objetivo de estas reuniones es analizar el estado del proyecto (realizar un *Sprint Retrospective*) y determinar qué ajustes hay que realizar para la siguiente iteración planificada.

c) Product Owner: es la persona que define las características del producto software.
El rol de *product owner* será desempeñado tanto por el autor como por su tutor de Trabajo Fin de Grado.

d) Scrum Master: es la figura que lidera las reuniones del proyecto.
El rol de *scrum master* será desempeñado por el tutor.

e) The Scrum Team: es el equipo de desarrollo del proyecto.
En este proyecto realmente no habrá equipo de desarrollo y el papel de desarrollar el software será desempeñado por el autor .

f) Agile Metrics: son medidas que se utilizan para comprender el progreso del proyecto.
La métrica que se utilizará en este proyecto es *Scrum Velocity* que mide la cantidad de características implementadas del sistema ("Done Features").

Como se puede observar, la metodología Scrum se ha adaptado para las características de este proyecto.

5

Especificación de Requisitos

5.1 Requisitos Funcionales

FR-1. Detección de Imagen: el sistema recibe una imagen y debe ser capaz de reconocer su contenido.

FR-1.1. Almacenamiento temporal: la imagen recibida se deberá almacenar temporalmente en un directorio del servidor hasta finalizar el análisis del contenido de dicha imagen.

FR-1.2. Sistema de reconocimiento de imágenes: deberá existir una forma de reconocer el contenido de la imagen.

FR-2. Análisis de concepto: un componente del sistema se encargará de llevar a cabo el proceso de extracción de información y análisis de sentimientos.

FR-2.1. Obtención de tweets: el sistema debe obtener textos a analizar utilizando la API de *Twitter*.

FR-2.2. Pre-procesado de datos: deberá existir un componente del sistema que haga el pre-procesado de los tweets.

FR-2.3. Usuarios más implicados: un módulo del sistema ha de encargarse de analizar quienes son los usuarios de Twitter más implicados en el concepto que se está analizando.

FR-2.4. Frecuencia de palabras: una parte del sistema deberá analizar la frecuencia que tiene las palabras que aparecen en los tweets.

FR-2.5. Palabras relacionadas "n-gram" y correlaciones: deberá existir un componente del Sistema capaz de establecer la relación existente entre las palabras.

FR-2.6. Análisis de Sentimientos: un componente del sistema llevará a cabo el proceso de análisis sentimental.

FR-2.7. Clustering: esta parte del sistema realizará las tareas necesarias para llevar a cabo el "clustering" de los conceptos que aparecen en los tweets.

FR-2.8. Selección de análisis: el usuario deberá tener la posibilidad de seleccionar los análisis que desea lleva a cabo.

FR-3. Resultado: el sistema deberá devolver el resultado del análisis en formato JSON.

FR-4. Representación de los resultados: se desarrollará un aplicación móvil para representar gráficamente los resultados del análisis.

FR-4.1. Tabla: se mostrará una tabla para indicar los usuarios más implicados en relación con la cantidad de tweets que han generado.

FR-4.2. Nube de palabras: la aplicación mostrará una nube de palabras para indicar al usuario cuáles son los conceptos más relacionados con la imagen que desea analizar.

FR-4.3. Gráfico de Red: la aplicación deberá mostrar un gráfico de red para representar la relación que existente entre las palabras.

FR-4.4. Gráfico de barras: la aplicación mostrará un gráfico de barras para representar los resultados del análisis sentimental.

FR-4.5. Gráficos de dispersión: la aplicación usará gráficos de dispersión para representar la polaridad de los tweets.

FR-4.6. Gráfico de barras apiladas: la aplicación mostrará un gráfico de barras apiladas en el que cada barra aparece representado un tema y cada elemento de la pila representa un concepto perteneciente a dicho tema.

FR-4.7. Gráfico de radar: la aplicación deberá mostrar un gráfico de radar que represente el análisis de sentimientos y permita determinar el sentimiento más predominante.

FR-5. Autenticación del usuario: para acceder a los servicios del sistema el usuario tendrá que estar registrado y deberá acceder los servicios mediante sus credenciales.

FR-5.1. Inicio de sesión: el usuario final del sistema accederá el sistema mediante su correo electrónico y contraseña.

FR-5.2. Registro del usuario: la aplicación pedirá los siguientes datos personales: nombre, apellidos, email y contraseña.

FR-5.3. Almacenamiento de los análisis realizados por el usuario: el principal objetivo del mecanismo de autenticación es almacenar los diferentes análisis realizados por los usuarios.

La finalidad que tiene el almacenamiento del análisis es la de proporcionar un mecanismo para reutilizar los datos extraídos de *Twitter*.

FR-6. Análisis de usuario: el sistema deberá presentar un mecanismos para aplicar los análisis descritos en **FR.2** sobre los "tweets" de un usuario particular de *Twitter*.

En la Tabla 5.1, se presenta una matriz de trazabilidad para garantizar que toda la lógica de negocio aparece contemplado en los requisitos funcionales del sistema.

	Lógica de Negocio	Servidor reconoce el contenido de una imagen	Identificar el público objetivo del producto	Conocer la opinión del público	Determinar la competencia del producto	Identificar las características más interesantes del producto	Aplicación Móvil que facilite la interpretación de los resultados	Almacenamiento de los resultados de análisis	Analizar los gustos de un usuario de
Requisitos Funcionales									
FR-1. Detección de Imagen		X							
FR-1.1. Almacenamiento temporal		X							
FR-1.2. Sistema de reconocimiento de imágenes		X							
FR-2. Análisis del concepto			X	X	X	X			
FR-2.1. Obtención de tweets			X	X	X	X			
FR-2.2. Preprocesado de datos			X	X	X	X			
FR-2.3. Most Implicated users			X						
FR-2.4. Word Frequency				X	X	X			
FR-2.5. Word Relationships: n-gram and correlations				X	X	X			
FR-2.6. Sentiment Analysis				X					
FR-2.7. Clustering				X	X	X			
FR-2.8 Selección de Análisis									
FR-3. Resultado							X		
FR-4. Representación gráfica							X		
FR-4.1. Tabla							X		
FR-4.2. Wordcloud							X		
FR-4.3. Gráfico de Red							X		
FR-4.4. Gráfico de barras							X		
FR-4.5. Gráficos de dispersión							X		
FR-4.6. Gráfico de Barras							X		
FR-4.7. Gráfico de radar							X		
FR-5. Autenticación del usuario							X		
FR-5.1. Inicio de sesión							X		
FR-5.2. Registro del usuario							X		
FR-5.3. Almacenamiento de los análisis realizados por el usuario							X	X	
Fr-6. Análisis de Usuario									X

Tabla 5.1 Matriz de Trazabilidad entre Requisitos Funcionales y Lógica de Negocio

5.2 Requisitos no funcionales

NFR-1. Rendimiento: el sistema tiene que proporcionar respuestas rápidas para que los usuarios tengan una experiencia cómoda. Dado que el sistema realiza un trabajo de procesamiento intenso se considera adecuado un tiempo de respuesta inferior a 40s.

NFR-2. Portabilidad: el sistema ha de proporcionar resultados que sean fácilmente interpretados por otro lenguaje de programación en caso de reutilizar las características del sistema o en caso de crear un nuevo tipo de cliente.

NFR-3. Diseño de las operaciones: los clientes del sistema deberán ser capaces de realizar cualquier tipo de operación sin ningún tipo de dificultad.

NFR-4. Diseño de la interfaz de usuario: las tareas que tiene que realizar el usuario han de ser cómodas, intuitivas y visuales, para que tengan una experiencia correcta y pueda usar la aplicación en cualquier momento.

NFR-5. Seguridad: el sistema deberá de ser capaz de gestionar el acceso a las distintas operaciones que proporciona. Deberá impedir cualquier tipo de ataque de seguridad que pueda dañar no solamente el normal funcionamiento del sistema, sino también los usuarios.

En la Tabla 5.2, se expone una matriz de trazabilidad entre la lógica de negocio y los requisitos no funcionales. Esta matriz tiene como objetivo describir la relación que existe entre las distintas funcionalidades especificadas como lógica de negocio con los factores que afecten el rendimiento y la calidad del sistema.

	Lógica de Negocio	Servidor reconoce el contenido de una imagen	Identificar el público objetivo del producto	Conocer la opinión del público	Determinar la competencia del producto	Identificar las características más interesantes del producto	Aplicación Móvil que facilite la interpretación de los resultados	Almacenamiento de los resultados de análisis	Analizar los gustos de un usuario de
Requisitos No Funcionales									
NFR-1. Rendimiento		X	X	X	X	X		X	X
NFR-2. Portabilidad			X	X	X	X			X
NFR-3. Diseño de las operaciones			X	X	X	X			X
NFR-4. Diseño de la interfaz de usuario							X		
NFR-5. Seguridad		X	X	X	X	X		X	X

Tabla 5.2 Matriz de Trazabilidad entre Requisitos No Funcionales y Lógica de Negocio

5.3 Criterios de validación

Por último, comentaremos los diferentes indicadores que nos permitirán conocer la calidad del sistema en cualquier momento del proceso de desarrollo:

- Número de funcionalidades implementadas: tenemos que implementar todas las funcionalidades.

- Tratamiento de errores: el sistema ha de proporcionar mecanismo para manejar los diferentes tipos de errores que puede ocurrir en cualquier momento de ejecución del sistema.

- Porcentaje de fallo: el porcentaje de fallo de acceso al sistema ha de ser inferior que 20% (fallo/(acierto+fallo)).

- Tiempo de respuesta: el sistema ha de ser capaz de proporcionar los resultados antes de 40s.

- Recursos del sistema: el sistema tiene un límite de acceso a los recursos hardware. Ejemplos: Límite de espacio que ocupa en HDD, Límite de acceso a memoria RAM y Límite de uso del CPU.

- El intercambio de información entre la aplicación y el sistema se de realizar mediante un sistema de **cifrado de información**.

6

Modelado y Diseño del Sistema

6.1 Modelo Estructural

6.1.1 Diagrama de clases

El diagrama de clases del sistema define con un alto nivel de abstracción cada una de las entidades que constituye el sistema y las relaciones existentes entre cada una de ellas.

Para definir las entidades del sistema, se hace una descripción de los atributos que constituye los mismos identificando el tipo de información que debe almacenar, su visibilidad y multiplicidad. Asimismo, se realiza una explicación de las operaciones que puede llevar a cabo cada elemento del sistema.

Por otro lado, para definir las relaciones existentes entre las entidades, se realización una descripción de los mismos identificando los roles y las multiplicidades de cada extremo de la relación.

Diagrama de clases del servidor:

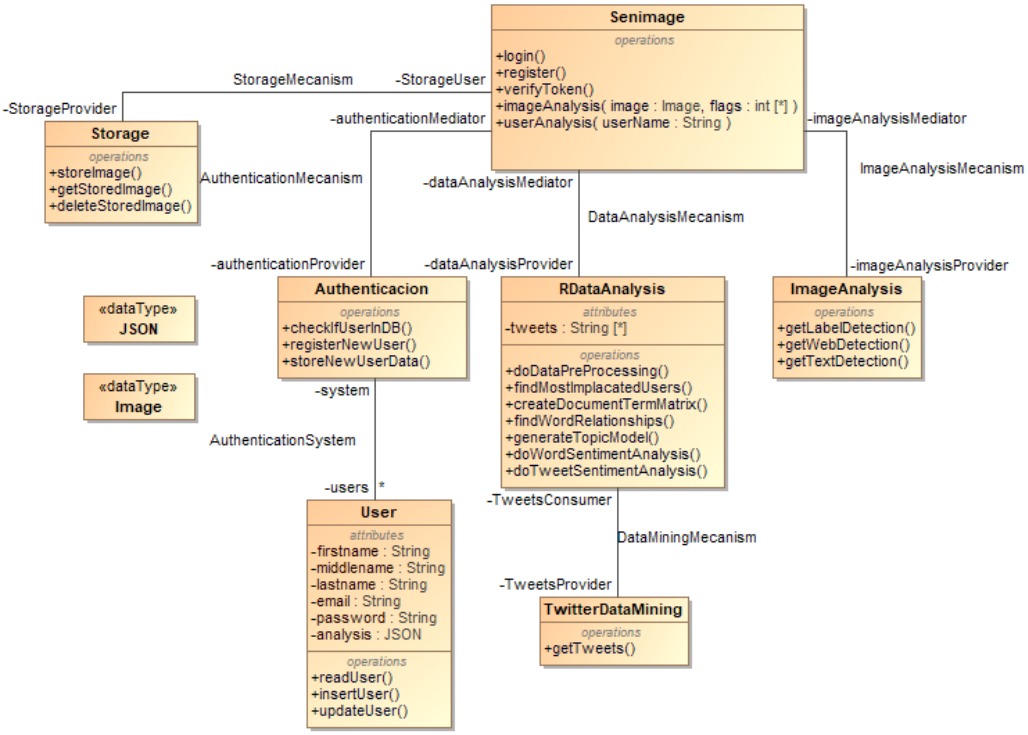


Figura 6.1 Diagrama de clases del servidor

a) "Senimage": es el "director de orquesta" del entorno de análisis. Es la clase que atiende todas las peticiones que se realizan sobre el servidor y, como se puede observar, presenta todas las operaciones que puede realizar un usuario del sistema:

- *verifyToken()*: es un método que garantiza que los usuarios que desean acceder las operaciones *imageAnalysis()* y *userAnalysis()* son usuarios registrados en la base de datos del sistema.
Esta operación hace una llamada al método: checkIfUserInDB() de la clase Authentication.

- *imageAnalysis(image: Image, flags: int[*])*: es una operación que recibe como parámetro la imagen a analizar y los "flags" para determinar qué tipo de análisis hay que ejecutar. Devuelve los análisis como resultado.
Este método hace una llamada a las funciones de la clase *RDataAnalysis* y a la operación *storeNewUserData()* de la clase Authentication.

- *userAnalysis(twitterUserName: String)*: es una función que recibe el nombre de un usuario de *Twitter* como parámetro. Devuelve el análisis de sus tweets como resultado.
Este método hace llamadas a los métodos de la clase *RDataAnalysis*.

- *login(email, password: String)*: permite inicio de sesión del usuario.
Esta operación hace una llamada al método: *checkIfUserInDB()* de la clase *Authentication*.

- *register(firstname, middlename,lastname,email,password: String)*: es una función que registra el usuario dentro del sistema.
Esta operación hace una llamada a los método: *checkIfUserInDB()* y *registerNewUser()* de la clase *Authentication*.

b) "Authentication": es la clase que ejecuta las operaciones CRUD sobre la entidad usuario. Esta clase permite conocer si un usuario está en el sistema y, en tal caso, extraer toda la información que se encuentra almacenado en la base de datos sobre dicho usuario.

c) "User": es una clase que mapea los objetos usuarios con los entradas de usuario que se encuentra almacenado dentro de la base de datos. En ella se almacena toda la información que se le pide al usuario cuando este se registra: nombre, apellidos, correo y contraseña. Además, a través de esta clase también se almacena los resultados de análisis (en formato JSON) que ha realizado el usuario utilizando el sistema.

d) "RDataAnalysis": es una clase que permite realizar todos los análisis que se han descrito en el capítulo **5. especificación de requisitos**. Cada una de las operaciones que presenta esta clase está implementado según lo definido en el apartado **3.1 Técnicas de extracción de información y análisis de requisitos**.

e) "ImageAnalysis": es una clase que interacciona con la *API Vision* de *Google* para conocer el contenido de una imagen. Las imágenes se pueden analizar por:

- Detección de etiquetas - *getLabelDetection()*: permite conocer categorías acerca del contenido de la imagen.
- Detección web - *getWebDetection()*: realiza una búsqueda en la web de imágenes similares a la recibida como parámetro.
- Detección de textos - *getTextDetection()*: analiza qué textos podría contener la imagen.

e) "TwitterDataMining": es una clase que permite extraer "tweets" utilizando la API de Twitter.

f) "Storage": es una clase que permite almacenar una imagen en el sistema de ficheros del servidor.

Diagrama de clases de la aplicación móvil

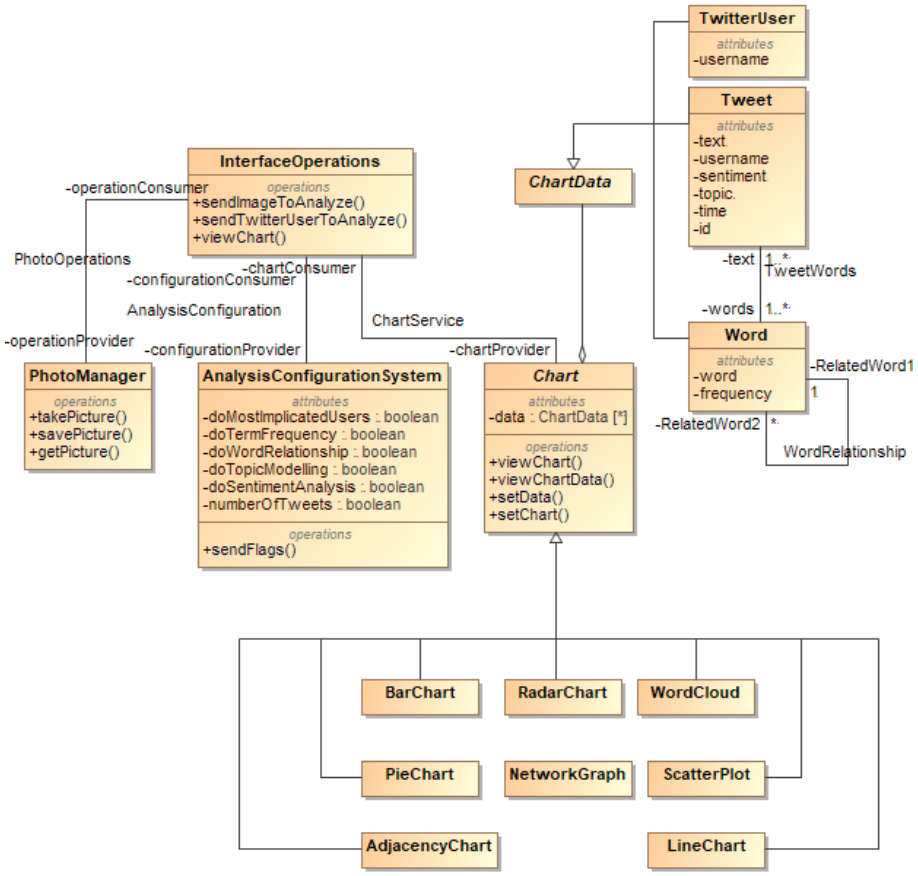


Figura 6.2 Diagrama de clases de la aplicación móvil

a) **InterfaceOperations**: es la clase que se encarga de responder los eventos ocasionados por la interacción del usuario con la aplicación. Para ello, define las siguientes operaciones:

   - *sendImageToAnalyze()*: este método hace la llamada al método *imageAnalysis()* de la clase *Senimage* del servidor.

   - sendTwitterUserToAnalyze(): este método hace la llamada al método userAnalysis() de la clase *Senimage* del servidor.

   - viewChart(): crea una instancia de las clases que heredan de la clase *Chart*. Es decir, puede instanciar un objeto de la clase *BarChart*, *PieChart*…

b) **PhotoManager**: esta clase gestiona todas las operaciones que tiene que realizar el teléfono móvil sobre la imagen que desea enviar al servidor. Estas operaciones incluye:

   - takePicture(): que permite lanzar la pantalla de la cámara para poder sacar un foto.

   - savePicture(): ofrece la posibilidad de almacenar la imagen en la memoria interna del teléfono.

   - getPicture(): permite obtener una imagen que está almacenada en la memoria.

c) **AnalysisConfigurationSystem**: es la clase que elabora el cuerpo de la petición que se realiza sobre el servidor web. Esta clase permite definir qué tipos de análisis se pedirán cuando la aplicación utilice el método *sendImageToAnalyze()* de la clase *InterfaceOperations*.

d) **Chart**: es una clase abstracta que permite realizar todas las operaciones e interacciones de usuario sobre las representaciones gráficas de la aplicación móvil. De esta clase, heredan todas las representaciones gráficas del sistema: *BarChart*, *WordCloud*, *PieChart*…

e) **ChartData**: es una clase abstracta que permite trasladar los resultados de análisis sobre las representaciones gráficas (son los datos de los gráficos).

6.1.2 Diagrama de componentes

En este apartado se realiza una descripción/representación a alto nivel de la estructura del sistema. Tiene como principal objetivo describir los diferentes componentes que integran el sistema, mostrar las interacciones entre ellos, exponer los patrones que supervisan su composición y así como las restricciones que se tienen que aplicar sobre dichos patrones.

El sistema está compuesto principalmente por dos componentes: el "Back-End" y el "Front-End". Esto tiene como objetivo la separación de intereses entre una capa de presentación (aplicación móvil) y una capa de acceso a datos (servicio web).

Diagrama de componentes del "Back-End":

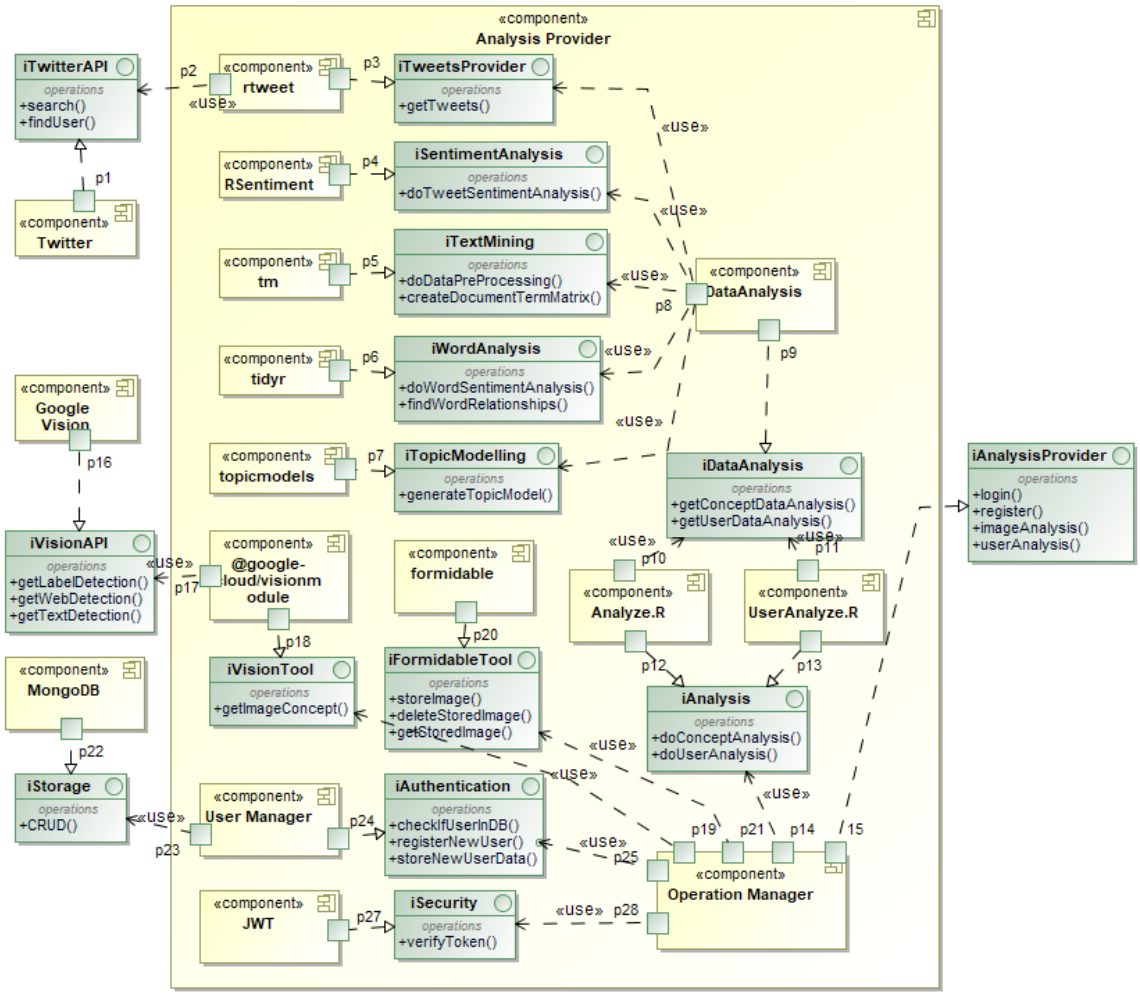


Figura 6.3 Diagrama de componentes del servidor

En el diagrama de componentes del servidor se describen todos los componentes del mismo, las operaciones que proporciona cada uno de ellos y las dependencias existentes entre dichos componentes.

Nótese que las operaciones que proporciona cada uno de los componentes se corresponden con las operaciones descritas en el apartado de **6.1.1 Diagrama de Clases**.

Los componentes del servidor que se definen a continuación son los archivos, bibliotecas, módulos, ejecutables y paquetes que constituyen el servicio de análisis:

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

- *rtweet*: es una librería de R que facilita el acceso a la API de *Twitter* y proporciona diferentes mecanismos para realizar peticiones HTTP. Esta librería incorpora herramientas para mantener sesión con el servidor OAuth de Twitter, manejo de tokens, funciones para hacer llamadas a las distintas rutas de la API.

- *Google Vision*: es una API que proporciona detección de imagen. Esta API permitirá determinar el concepto que hay que filtrar en Twitter, y consecuentemente será responsable de determinar la bondad del análisis, puesto que si realiza una detección imprecisa, todo el análisis realizado por el sistema sería acerca de un concepto que no está completamente relacionado con la imagen.

- *@google-cloud/vision*: es un módulo de *NodeJS* que proporciona funciones para acceder a los distintos servicios que proporciona *Google Vision*.

- *tm*: es una librería de R que permite realizar "text mining". Esta herramienta será fundamental para el preprocesamiento de datos y la generación de la matriz de términos.

- *RSentiment*: es una librería de R que permite indicar qué tipos de sentimientos se expresan en cada uno de los tweets que se está analizando. Esta herramienta tiene un conjunto de palabras asociadas a un sentimiento de tal manera que la puntuación (el sentimiento) de una frase se genera a partir de la suma de sentimientos de las palabras que se encuentran en dicha frase.

- *tidyr/tidytext*: son módulos que permite generar n-gramos, es decir, red de palabras que establecen las relaciones existentes entre dichas palabras en función de su cercanía física en los textos que estamos analizando. Dichos módulos también permiten generar otro tipo de análisis de sentimientos, pero esta vez no se trata de un análisis frase por frase sino de un análisis palabra por palabra y utilizando otros tipos de adjetivos para calificar a dichas palabras.

- *topicmodels*: es un módulo de R que permite generar un modelo de tópicos a partir de la matriz de términos. Los tópicos se elaboran a partir de las palabras más usadas en los textos. De esta manera, el resultado que proporciona este módulo es un conjunto de palabras correspondientes a un posible tema del corpus.

-*DataAnalysis*: es un componente que permite orquestar la llamada a los diferentes componentes del sistema que llevan a cabo algún tipo de análisis. Estos componentes son: *rtweet*, *RSentiment*, *tm*, *tidyr* y *topicmodels*.

-*Analyze.R*: se corresponde con la parte principal del sistema de análisis. Este componente es el que engloba las llamadas a las distintas librerías de R y es en definitiva el responsable de generar la estructura de datos que contiene el análisis del concepto correspondiente a la imagen que ha enviado el usuario.

-*UserAnalyze.R*: permite realizar el análisis de usuario. Su funcionalidad se asemeja al componente anterior. Sin embargo, este módulo está centrado en proporcionar análisis sobre los últimos tweets de un usuario particular de *Twitter*.

- *MongoDB*: es una base de datos NoSQL orientado a documentos que usamos para almacenar la información del usuario, así como los resultados de análisis que dicho

usuario realiza sobre nuestro sistema. Esta base datos tendrá una "colección" de usuarios y, por consecuencia, cada uno de sus "documentos" contendrá datos de un usuario. Para establecer la relación entre un usuario y sus análisis (relación uno a mucho), quizás lo más adecuado es generar un "atributo" correspondiente a una lista de análisis que ha realizado el usuario, es decir, incrustar el documento que contiene el análisis dentro del documento del usuario en la base de datos, tal y como se recomienda en la documentación de esta base de datos.

- *User Manager*: contiene una representación del esquema del usuario en la base de datos. Es un fichero que engloba todas las funcionalidades relacionadas con las operaciones que tiene que hacer el sistema a la base de datos. Este componente utiliza el módulo *Mongoose* de *NodeJS* para llevar a cabo el mapeo objeto relacional de los usuarios en la base de datos.

-JWT: nos permite generar tokens de acceso al usuario usando el cifrado *Hmac Sha256*. Este componente se encarga de "securizar" el acceso a las distintas rutas del servidor.

Diagrama de Componentes de la herramienta de Visualización:

En el diagrama de componentes de la herramienta de visualización se representan los componentes que constituyen la aplicación móvil. Estos componentes pueden ser de 4 tipos: componentes de representación gráfica, componentes para realizar autenticación, componentes para trabajar con la cámara del teléfono y componentes que permiten llamar las operaciones del servidor.

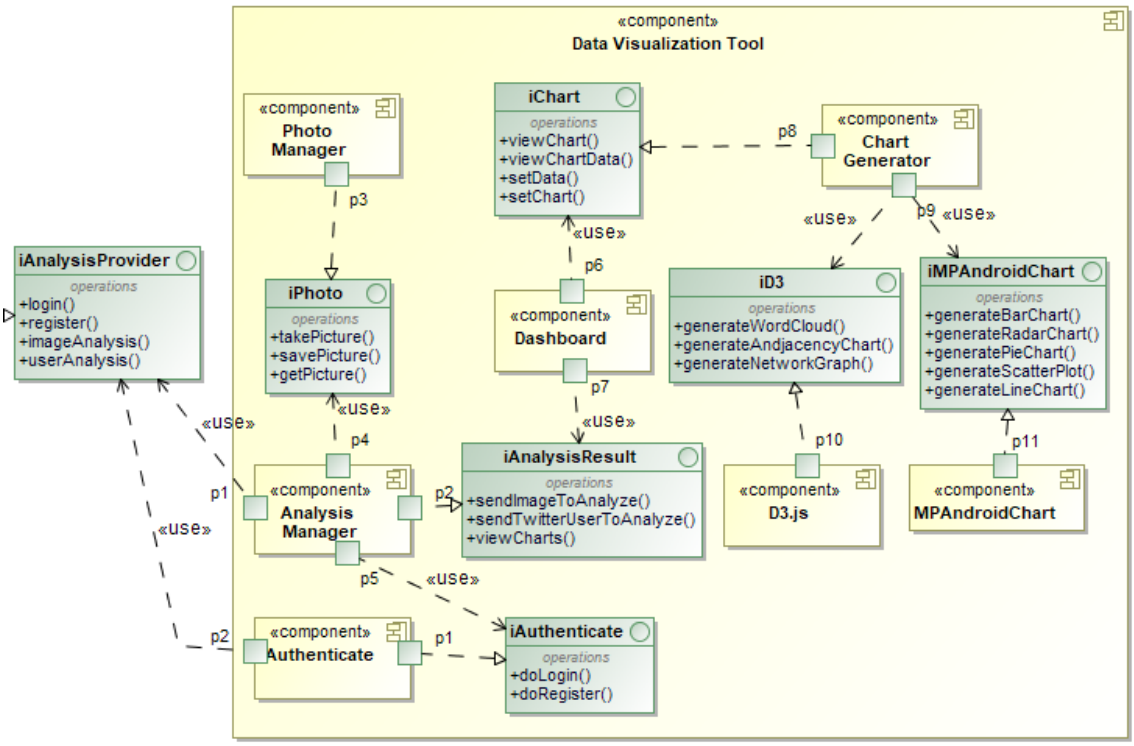


Figura 6.4 Diagrama de componentes de la aplicación móvil

Hay que destacar en este punto que la aplicación móvil utiliza la interfaz *IAnalysisProvider* para poder hacer una llamada a las operaciones del servidor.

-*Photo Manager*: es un componente que permite trabajar con la cámara del teléfono y el sistema de ficheros del mismo.

-*AnalysisManager*: permite identificar que tipos de análisis dese realizar el usuario (Análisis Sentimental, "Topic Modelling", Frecuencia de palabras…).

-*Authenticate*: permite llamar las operaciones login() y register() del servidor a través de la aplicación móvil.

-*Dashboard*: muestra un resumen de los resultados de análisis. Esto se corresponde con lo que se suele denominar como "story telling".

-*ChartGenerator*: es un componente que permite realizar las diferentes representaciones gráficas usando las librerías *MPAndroidChart* y *D3.js*.

-MPAndroidChart: es una librería de Android que permite representar Diagrama de Barras, Diagrama de sectores, Diagrama de Líneas, Diagram de radares y Gráficos de Dispersión.

-*D3.js*: es una librería de Javascript que permite realizar muchos tipos de gráficos. En este proyecto, dicha librería se usará para representar nube de palabras, diagramas de redes y matrices de adyacencia.

6.2 Modelo de Comportamiento

6.2.1 Diagrama de Casos de Uso

Casos de Uso del Servicio Web

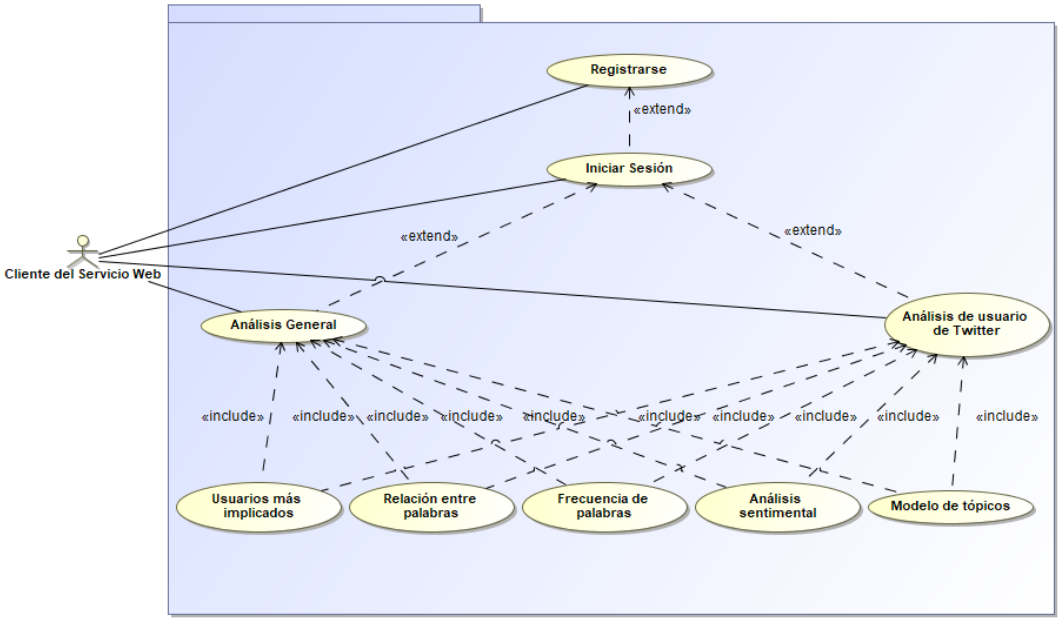


Figura 6.5 Diagrama de casos de uso del servidor

- Análisis general: permite realizar todos los análisis del sistema. Esta operación tiene como objetivo facilitar todas las operaciones que permite nuestro servicio web de forma más eficiente.

- Análisis de Usuario de Twitter: el sistema presenta una la posibilidad de realizar el análisis de los “tweets” de un usuario específico.

- Usuarios más implicados: esta operación permite al cliente conocer los usuarios más implicados sobre el producto que está analizando.

- Relación entre palabras: esta ruta del servicio web proporciona un listado de todas las palabras relacionadas con el producto y, así mismo, las relaciones existentes entre dichas palabras.

- Frecuencia de las palabras: este método permite al usuario obtener una lista de las palabras más relacionadas con su producto.

- Análisis sentimental: proporciona al cliente información acerca de qué es lo que piensa su público acerca de su producto.

- Modelo de Tópicos: es una operación que permite al usuario entender qué otros conceptos están relacionados con su producto. Esto podría ser útil para entender cuáles son las principales características del producto y cuál es la competencia existente en el mercado.

- Iniciar sesión: operación en la que un usuario proporcionar su nombre de usuario y contraseña para recibir un token de acceso a la aplicación.

- Registrarse: un usuario final proporciona información personal para poder tener acceso a nuestro sistema.

Casos de Uso de la herramienta de visualización

La aplicación móvil presenta las siguientes características:
- Sacar una Foto: la aplicación permite sacar una imagen del producto que el usuario de la aplicación desea analizar.

- Elegir tipo de análisis: el usuario puede elegir el tipo de análisis a realizar, ya sea un análisis general, un análisis de usuario de *Twitter* o una selección de análisis más específicos.

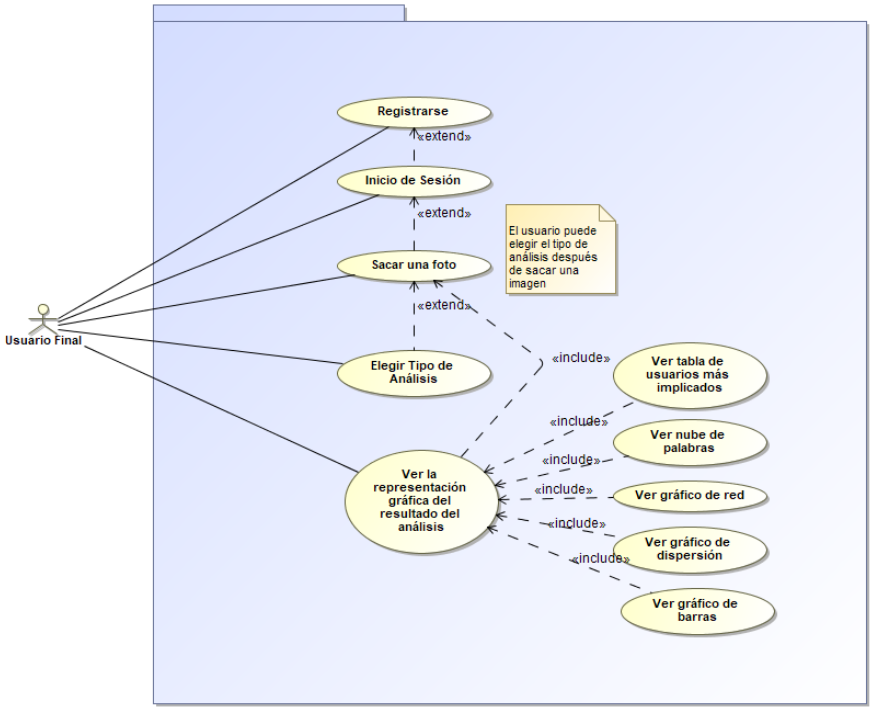


Figura 6.6 Diagrama de casos de uso de la aplicación móvil

-Ver la representación gráfica del resultado del análisis: tal y como su nombre indica, esta operación permitirá al usuario final visualizar el resultado del análisis mediante diferentes representaciones gráficas. Este caso de uso incluye las siguientes:

- Ver tabla de Usuarios más implicados
- Ver nube de palabras
- Ver gráfico de Red
- Ver gráfico de dispersión
- Ver gráfico de barras

- Inicio de sesión: el usuario (registrado) indica su correo electrónico y su contraseña en los campos dedicados a ello en el formulario.

- Registrarse: un usuario rellena un formulario con sus datos personales para acceder las distintas funcionalidades del sistema.

6.2.2 Diagrama de Flujo de Procesos

En este apartado se realiza el modelo de flujo de eventos de cada caso de uso que se ha identificado en el apartado anterior. Esto servirá como una herramienta para especificar la implementación de las operaciones más complejas del sistema.

Los diagramas de flujo elaborados para describir las diferentes operaciones del sistema se describen mediante un alto nivel de abstracción con el fin de enfatizar los aspectos más interesantes del flujo de ejecución del mismo.

Análisis de Imagen (Análisis General y Específico)

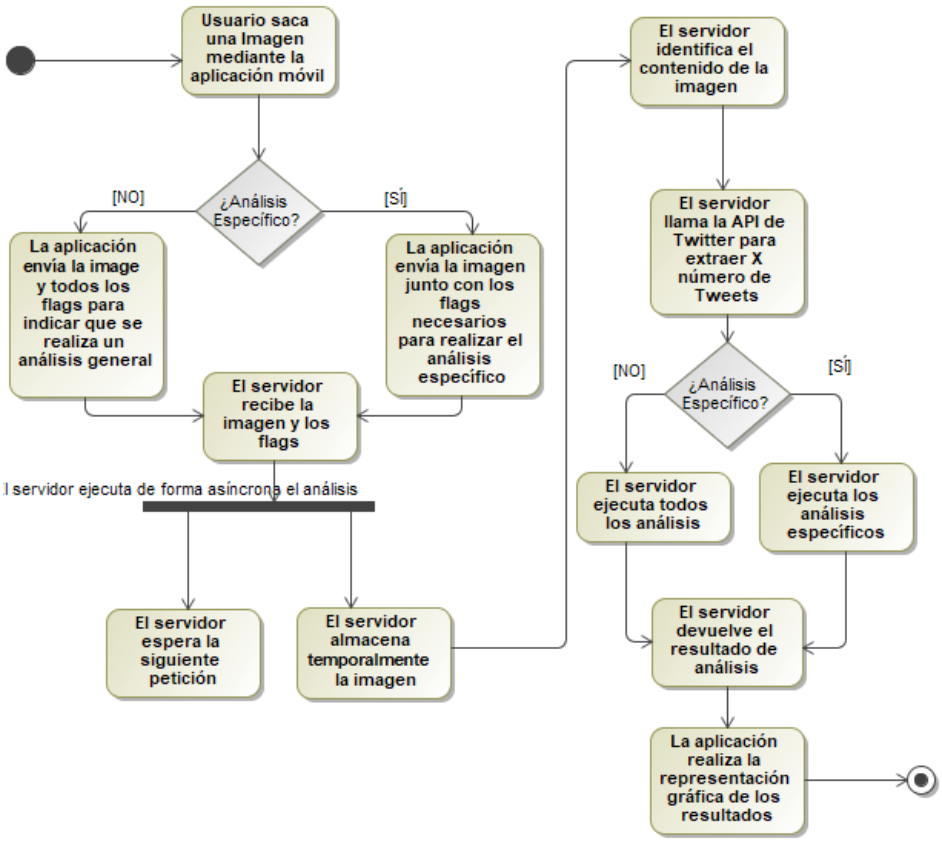


Figura 6.7 Diagrama de flujo de procesos del análisis de imagen

En los siguientes párrafos se comentan las principales actividades (en orden cronológico) que realizan los distintos componentes del sistema para llevar a cabo la operación *análisis a partir de una imagen*:

1. El usuario utiliza la cámara del teléfono para obtener una imagen del concepto que desea analizar.
2. La aplicación pregunta por los tipos de análisis que el usuario desea realizar. Asimismo pregunta por la cantidad de muestra de datos que se desea analizar (número de tweets). Estos parámetros determinan el tiempo de respuesta del sistema ya que, cuánto más análisis desea realizar el usuario y cuánto más muestra de datos, el servidor tardará más o menos en generar los resultados.

3. El servidor recibe la imagen y los "flags" y después crea una hebra que se ejecuta en segundo plano, para llevar a cabo todo el proceso de análisis. De tal manera, que el servidor sea capaz de esperar la siguiente petición.
4. La hebra que ejecuta el análisis almacena temporalmente la imagen (lo elimina después de realizar el análisis).
5. El servidor extrae el concepto que contiene la imagen usando la API Vision. Para ello, realiza detección de etiquetas, detección web y detección de textos (operaciones previamente comentadas en el Diagrama de Clases del servidor).
6. El servidor utiliza la función search() de Twitter para extraer X tweets, siendo X la cantidad de muestra de datos que ha pedido el usuario.

7. El servidor ejecuta los distintos tipos de análisis según los flags que ha recibido: Análisis de sentimientos, Frecuencia de términos…
8. El servidor devuelve el resultado del análisis en formato JSON.
9. La aplicación móvil realiza las representaciones gráficas en función de los resultados obtenidos.

Análisis de un Usuario de *Twitter*

El análisis de un usuario concreto de *Twitter* sigue un proceso similar análisis de una imagen. Sin embargo, en este caso no se puede configurar el análisis dado que los tipos de análisis que se realizan sobre el usuario son ligeramente más específicas y más eficientes, lo cual elimina la necesidad de optimizar manualmente los análisis. Esto es porque en esta operación no se realiza una petición a la *API Vision* ni tampoco hay que almacenar una imagen en el sistema de ficheros del servidor.

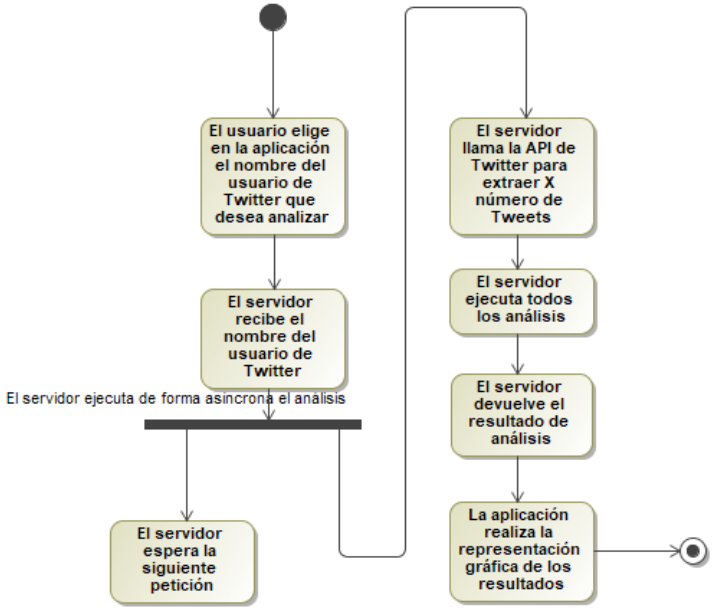


Figura 6.8 Diagrama de flujo de procesos del análisis de un usuario de *Twitter*

Registro e Inicio de sesión

Tanto el registro de un usuario como el inicio de sesión siguen un flujo de actividad relativamente sencillo. Pues dichas operaciones se basan en las técnicas tradicionales de autenticación al servidor.

La validación de los datos del formulario se realizan tanto en la aplicación móvil como en el servidor con el fin de implementar un sistema de autenticación robusto.

El registro de un usuario de nuestra aplicación se corresponde a esquema habitual de este proceso: cumplimentación de un formulario, validación y almacenamiento de los datos personales del usuario.

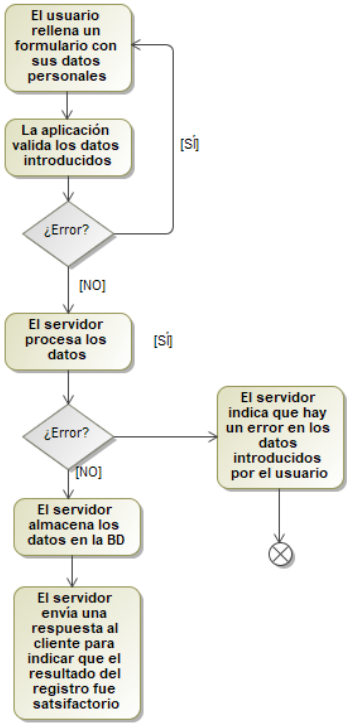


Figura 6.9 Diagrama de flujo de procesos del registro de usuario

El inicio de sesión es bastante peculiar ya que tras proporcionar las credenciales del usuario, el servidor genera un token de acceso a las diferentes rutas del servidor para mantener una sesión del usuario a nuestro servidor.

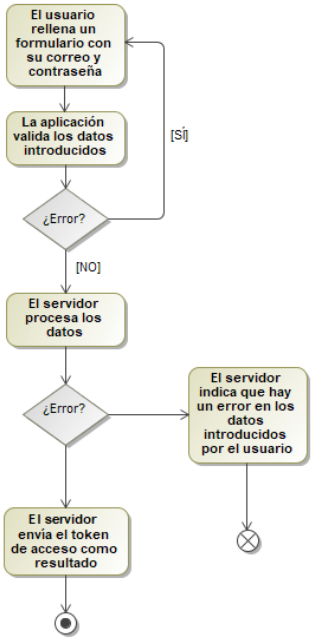


Figura 6.10 Diagrama de flujo de procesos del inicio de sesión

Después de realizar este proceso, la aplicación móvil usará el token que ha recibido del servidor para realizar las peticiones al resto de las operaciones del sistema.

6.3 Modelo de Interacción

6.3.1 Diagrama de Secuencias

El objetivo de los diagramas de secuencia que se definen en este apartado es describir claramente la interacción que existe entre los diferentes componentes del sistema cuando se realiza algún tipo de operación.

Análisis de Imagen

En el diagrama de secuencia del análisis general se realiza una mejor descripción de la interacción de la hebra que crea el servidor para llevar a cabo todo el proceso de análisis.

Nótese que en este diagrama se han omitido algunos detalles descritos en las diagramas de apartados anteriores, como por ejemplo la posibilidad de seleccionar el tipo de análisis ya que se encuentra descrito en el diagrama de actividad.

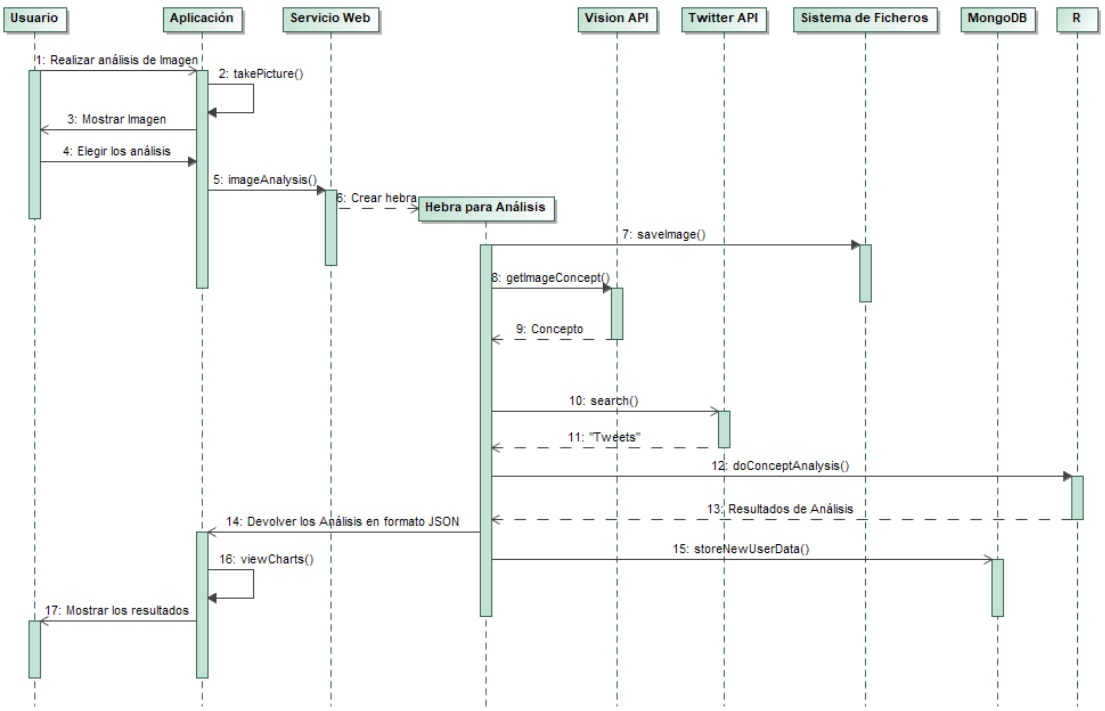


Figura 6.11 Diagrama de secuencia del análisis de imagen

A continuación se describirán los aspectos más interesantes del diagrama, especialmente aquellos que pueden llevar a la confusión.

-*Hebra para análisis*: es un proceso asíncrono que se encarga de llevar a cabo toda la actividad que hay que realizar para analizar una imagen.

-*Sistema de ficheros*: es una entidad que permite almacenar la imagen en el sistema de ficheros del sistema operativo. En el programa, el componente *formidable* realiza de interfaz al sistema de ficheros y permite la operación saveImage().

-*R*: es el motor del sistema de análisis del sistema. Se puede acceder sus servicios a través del componente *DataAnalysis* del sistema que es el encargado de gestionar todos los componentes (*rtweet*, *tm*, *topicmodels*…) de análisis del sistema que trabaja con *R*.

-*MongoDB*: es la base de datos del sistema (como ya descrito previamente). Para trabajar con esta entidad se utiliza el componente *User Manager* que permite llevar a cabo la operación storeNewUserData() con el fin de almacenar el resultado del análisis.

Análisis de un Usuario de *Twitter*

Como se ha comentado en apartados anteriores el análisis de usuario de Twitter sigue un proceso muy similar al análisis de Imagen. Esto también ocurre con el diagrama de secuencias.

Hay que destacar, no obstante, que el análisis de usuario es un proceso de menor complejidad en cuanto a interacción de componentes. Esto es porque no requiere el uso de *Vision* API ni del Sistema de Ficheros del sistema operativo del servidor.

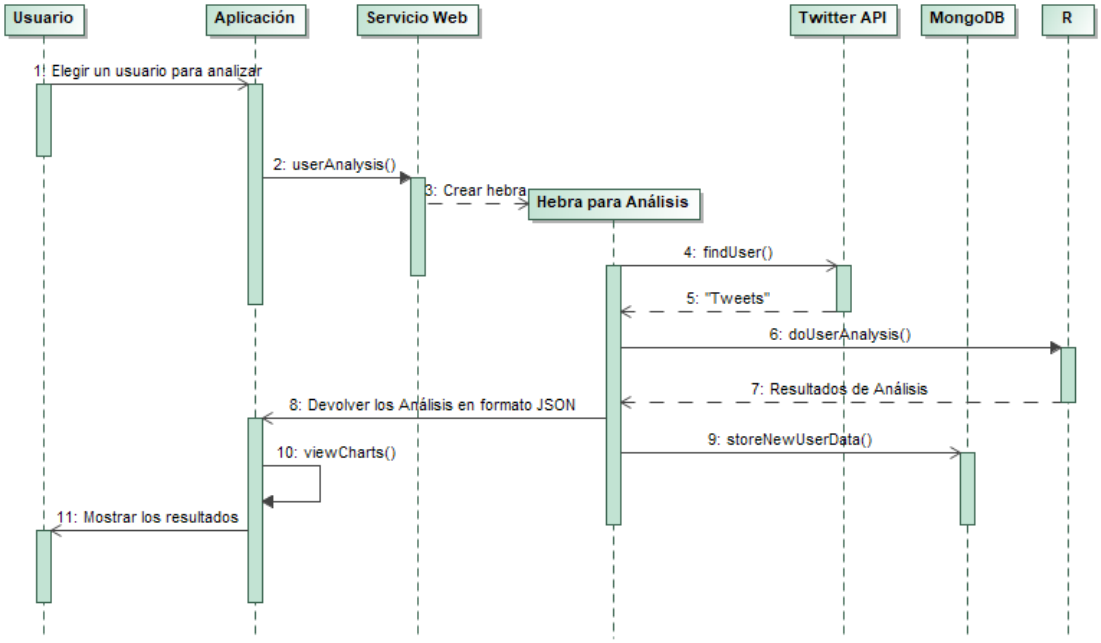


Figura 6.12 Diagrama de secuencia de un usuario de *Twitter*

La principal diferencia de este diagrama con respecto al diagrama de Análisis de Imagen es la llamada a la función findUser(), que es una operación que permite obtener los últimos "tweets" realizados por un usuario. También hay que destacar el uso del método doUserAnalysis(), que como se ha comentado anteriormente, es una operación que proporciona un análisis más especializado para un usuario concreto de *Twitter*.

7

Implementación

7.1 Tecnologías

En este subcapítulo se definen todas las tecnologías, tanto del lado del servidor como del lado del cliente, y herramientas utilizadas para desarrollar los distintos componentes del sistema. El objetivo principal de este apartado de la documentación es describir el uso que se hace de dichas tecnologías.

En este apartado se comentan todas aquellas tecnologías que desempeñan un papel fundamental en el sistema, que no forman parte de la descripción de la arquitectura del sistema expuesto en el apartado **6.1.2 Diagrama de Componentes,** con el fin de reducir la redundancia de información.

Tecnologías del Servicio Web

-*Node.js*: es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. (Node.js Foundation, s.f.)

Esta tecnología se utiliza para desplegar el servidor web (back-end de la aplicación) y constituye uno de los pilares fundamentales del sistema.

-*npm*: es el manejador de paquetes por defecto de Node.js. Se ejecuta a través de la línea de comandos y maneja todas las dependencias de una aplicación implementada mediante node.js.

Esta tecnología se encarga de instalar y actualizar todos los paquetes que necesita el servidor a través del comando *npm install* y *npm update*.

Este gestor de paquetes se ha utilizado para manejar las siguientes dependencias: *express*, *body-parser*, *path*, *fs*, *jsonwebtoken*, *mongoose*, *formidable*, *r-script* y *@google-cloud/vision*.

-*Express.js*: es un marco de trabajo de Javascript que permite gestionar todas las peticiones *HTTP* que se realiza sobre el servidor. Se encarga del enrutamiento de los puntos finales de la aplicación ("endpoints") y de proporcionar los resultados al cliente.

Express.js es el manejador de eventos del servicio web del sistema. Pues permite definir las funciones que se ejecutarán cuando el servidor reciba alguna petición del tipo GET o POST.

Esta tecnología se utiliza para implementar las funciones que atienden las siguientes peticiones o rutas del servidor: "*/login*", "*/register*", "*/imageAnalysis*" y "*/userAnalysis*".

-*body-parser*: es un "middleware", dependencia que define funciones que tienen acceso al objeto de solicitud y al objeto de respuesta, que permite "parsear" los resultados en formato *application/json*.

-*path*: es una dependencia que permite trabajar con las "rutas" del servidor. Se utiliza principalmente para importar las funciones definidas en otros archivos y para acceder las imágenes que se almacena en el servidor durante el proceso de análisis.

-*fs*: es una dependencia que facilita la realización de operaciones sobre el sistema de ficheros del servidor. Ofrece la posibilidad de almacenar la imagen que envía el cliente en una carpeta del servidor. Asimismo, permite eliminar dicho archivo después de realizar el análisis.

-*jsonwebtoken*: es un "middleware" que permite generar un JSON Web Tokens (de ahí su nombre) basando en el estándar *RFC 7519*.

La principal finalidad de este módulo es ofrecer autenticación al usuario. Se utiliza para definir las funciones "callbacks", una función que se pasa como parámetro y permite generar una respuesta para un evento del sistema, de las siguientes rutas del servidor: "/*login*" y "/*register*".

-*mongoose*: es un middleware capaz de realizar el mapeo objeto relacional de los elementos de la base de datos a una un clase del servidor.

En el proyecto este middleware se utiliza para mapear un usuario de la base de datos a un usuario definido como una clase del servidor con el objetivo de realizar operaciones CRUD ("Create", "Read", "Update" y "Delete") sobre dicha entidad.

-*formidable*: este módulo permite trabajar con peticiones *HTTP* que tienen como cuerpo de la llamada contenido del tipo *multipart/form-data*,

En concreto, este módulo se utiliza para extraer la imagen enviada por el usuario de la petición *HTTP*.

-*r-script*: es una dependencia que permite ejecutar "scripts" de R desde *Node.js*. Esta dependencia no sólo permite la ejecución sino también el paso de parámetro.

La dependencia *r-script* permite se utiliza para hace una llamada a los dos ficheros de análisis del servidor "Analyze.R" y "UserAnalyze.R".

Esta librería se utiliza para llevar a cabo la detección de etiquetas, la detección web y la detección de textos de una imagen.

-*R*: es un entorno software orientado al análisis estadístico y la representación gráfica. (The R Foundation, s.f.)

Este entorno se utiliza para llevar a cabo la minería y el análisis de datos. Este entorno tiene un gestor de paquetes que permite instalar todas las librerías necesarias para los servicios de análisis del sistema: *httr*, *dplyr*, *plyr*, *rtweet, tm, RSentiment*, *tidytext*, *tidyr* y *topicmodels*. Nótese que estas últimas librerías ya fueron comentadas en el apartado de **6.1.2 Diagrama de Componentes**, pues constituyen partes fundamentales de la arquitectura del sistema. Por lo tanto, para evitar información redundante solamente se explicarán el uso de httr, dplyr y plyr.

-*httr*: es una librería de *R* que permite realizar peticiones HTTP. Se trata de una dependencia necesaria para trabajar con la librería *rtweet*.

-*dplyr*: es una librería de *R* que permite transformar y resumir filas y columnas. (Love, s.f.)

-*plyr*: es un conjunto de herramientas que resuelve un conjunto de problemas. Para ello, divide un problema grande en problemas más pequeñas y después resuelve cada uno de dichos sub-problemas con el fin de construir la solución al problema. (Wickham, s.f.)

Tanto *dplyr* como *plyr* son herramientas necesarias para la librería las otras librerías de análisis de datos.

-JSON: es un formato de texto ligero para el intercambio de datos. Permite definir el contenido de un texto utilizando meta-lenguaje. Es el formato utilizado para responder todas las peticiones del cliente del servicio web.

Tecnologías de la herramienta de visualización

-*Android*: es un sistema operativo diseñado para dispositivos móviles con pantalla táctil. Es el sistema operativo móvil más utilizado del mundo (su cuota de mercado es del 80%) y su principal lenguaje de programación es *Java*.

La decisión de implementar una aplicación basado en este sistema operativo frente a la desarrollo de una aplicación web tradicional se debe principalmente a las características que ofrece las aplicaciones nativas: rendimiento, mayor interactividad, diseño de la interfaz de usuario predefinido (*Material Design*), Mejora la experiencia del usuario, y la facilidad de acceso al hardware del dispositivo (Cámara y Acceso al Sistema de Ficheros del Dispositivo). En este sentido, también hay que destacar que en el sistema operativo Android no existe ninguna aplicación que represente una competencia directa de la aplicación que se desarrolla en este proyecto, tanto en la cantidad y calidad de análisis que proporciona como en las características que mejorar la accesibilidad del usuario.

Para la representación gráfica en *Android* se han utilizado las siguientes librerías: *MPAndroidChart* y *D3.js*. Ambas tecnologías fueron explicadas en el modelado estructural del sistema.

-*OkHttp*: es una librería de Android (de Java) que permite realizar peticiones HTTP. Esta librería se utiliza para hacer llamadas a los "endpoints" del servicio web.

-*Picasso*: es una librería de Android que optimiza la renderización de imagen en una aplicación Android. Asimismo presenta mecanismos para almacenar una imagen en la memoria *caché* del dispositivo.

-*link_builder*: es una librería de Android que permite asociar enlaces (URLs) a los textos que aperecen en una aplicación Android.

-*wordcloud*: es una librería de *Android* basada en la librería *D3.js* para realizar la representación gráfica de nube de palabras.

7.2 Tecnologías Alternativas Consideradas

En este apartado se explican todas las tecnologías que fueron consideradas en algún momento durante el proceso de implementación del sistema.

-*twitteR*: es una librería que permite realizar minería de datos en *Twitter*. Hasta el año previo a la realización de este proyecto twitteR era la tecnología de ciencia de datos más usada en *R*. Sin embargo, *rtweet*; tecnología usada en este proyecto; constituye la segunda versión de esta librería y, por lo tanto, representa una mejor opción.

Se ha comentado el uso de *twitteR* puesto que las primeras versiones del componente de análisis de datos fue desarrollada mediante dicha tecnología.

-*sentiment140*: es una librería utilizada para realizar análisis de sentimientos de texto. Fue descartado, por las restricciones que presenta *r-script*. Pues la librería sentiment140 utiliza la herramienta *DevTools*, que facilita la creación de paquetes de R, y el módulo *r-script* de *Node.js* no permite el uso de dicha herramienta.

-*android-network-graph*: es una librería de *Android* que permite llevar a cabo la representación de grafos en una "actividad", una vista de la interfaz, de *Android*. Sin embargo, la librería está desactualizada.

-*C3.js*: está librería de JavaScript permite una amplia variedad de representaciones gráficas. Está basado en D3.js. Sin embargo, esta última presenta una documentación más amplia y una comunidad de mayor tamaño.

-*HoloGraphLibrary*: es una librería de Android que permite representar gráfico de líneas, gráfico de barras y gráfico de sectores. Fue descartada por las mejores prestaciones que ofrece *MPAndroidChart* y la gran labor de mantenimiento software que realiza el autor de este último, pues recibe actualizaciones, mejoras y corrección de errores cada 1 o 2 semanas.

7.3 Herramientas Utilizadas y Lenguajes de Programación

En este apartado se describe las distintas herramientas para el desarrollo del sistema.

-*Atom.io*: es el principal editor de texto utilizado para desarrollar el servicio web. Ofrece una gran variedad de funcionalidades como por ejemplo: auto-completado de código, resaltado de errores, y resaltado sintáctico y semántico de código. También ofrece mecanismos para trabajar con control de versiones.

-*Android Studio*: entorno de desarrollo integrado oficial para la plataforma Android. Ofrece las mismas funcionalidades que Atom.io pero especializado en Java y XML. Además, también ofrece mecanismos para depurar el programa Android.

-*Postman*: herramienta utilizada para comprobar la ejecución de las rutas del servidor. Permite configurar la cabecera y el cuerpo de una petición HTTP, para después poder probarlo contra el servidor.

-*Virtualbox*: es un software de virtualización para arquitecturas x86/amd64. Esta herramienta se ha utilizado para ejecutar una máquina virtual. Pues este último se utiliza para mantener una copia de seguridad del servidor.

-*Java*: es un lenguaje de programación de propósito general, concurrente, orientado a objetos. Es lenguaje de programación de Android y, por tanto, es lenguaje utilizado para diseñar la lógica de la aplicación móvil.

-Lenguaje *R*: es el lenguaje de programación del *entorno R*. Es uno de los lenguajes más utilizados en el campo de la minería de datos, la investigación biomédica, la bioinformática y las matemáticas financieras. Este lenguaje se utiliza principalmente para codificar todas las funcionalidades relacionadas con el análisis de datos del sistema.

-*Javascript*: es el lenguaje de programación de *Node.js*. Es un lenguaje de programación orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Este lenguaje se utiliza para definir el servicio web.

-*Mocha*: marco de trabajo de *Javascript* que permite ejecutar pruebas en *Node.js*. *Mocha* es la herramienta más usada para llevar a cabo las pruebas de unidad y de

integración sobre las diferentes rutas que ofrece una aplicación web desarrollada en *Node.js*.

-*chai.js*: es una librería de asertos para Node.js que se utilizar junto con cualquier marco de trabajo orientado. En el proyecto, esta librería se ha utilizado como una herramienta complementaria a *Mocha*.

-*supertest*: es una librería de *Node.js* que proporciona un alto nivel de abstracción para llevar a cabo pruebas HTTP. Esta librería también se utiliza como un complemento a *Mocha*.

7.4 Aspectos Interesantes de la implementación

En este subcapítulo del documento se explican las principales decisiones de implementación que se han tomado durante el proceso de desarrollo del entorno. En ella, se explica tanto la filosofía de diseño de los componentes más importantes del sistema como aspectos particulares del código del servicio web y de la aplicación.

7.4.1 El Servicio Web

Para implementar el servicio web se han considerado dos tipos de arquitectura orientada a servicios (*SOA*).

La primera de ellas es *Simple Object Access Protocol* (SOAP), un protocolo de intercambio de datos que utiliza XML como principal medio de comunicación. Es una arquitectura formada por varias capas que define una serie de funciones que se expone a través de *Web Services Description Language* (WSDL), y está orientado a la creación de sistemas robustos.

La segunda (que es la opción elegida) es la arquitectura REST, una arquitectura de transferencia de información que define una serie de restricciones y propiedades a través del uso del protocolo HTTP (utiliza los verbos POST, PUT, PATCH, GET, DELETE de este protocolo). Define lo que se denomina servicio web *RESTful*, y utiliza JSON como formato de intercambio de datos, aunque también permite usar XML.

La decisión de utilizar esta última se debe al uso del formato JSON, un formato más ligero de datos que permitiría reducir la información recibida por el cliente del servicio web. También hay que destacar su enorme compatibilidad con el entorno en el que se define el servidor, pues JSON contiene objetos de Javascript y el servidor se codifica usando dicho lenguaje.

Por otro lado, para hacer la llamada a los diferentes servicios web en los que se basa el sistema, se ha decidido utilizar el patrón *mediador*. Puesto que este patrón de diseño software permite encapsular cómo interaccionan los diferentes elementos del sistema, lo cual permite resolver el problema de complejidad de comunicación que presenta el entorno.

La base de datos

La decisión de usar la base de datos NoSQL MongoDB se debe a que sus características principales (rendimiento, flexibilidad de esquema y escalabilidad) se ajustan adecuadamente con las necesidades del sistema.

- El rendimiento de las consultas es una de las principales necesidades del sistema dado que cada análisis supone el almacenamiento de una gran cantidad de datos.

- La flexibilidad del esquema también es bastante importante, pues existe una posibilidad de cambiar los datos que se almacenan en ella, pues la estructura en la que se devuelve los tweets puede cambiar y los resultados de análisis también.

- La escalabilidad del sistema también es muy importante dado que, tal y como se ha comentado previamente, se almacenan grandes volúmenes de datos.

Para definir el esquema de la base de datos se utiliza la dependencia *Mongoose*, un sistema que permite no solamente definir el esquema sino que también ofrece los mecanismos de conexión con la base datos.

La definición del modelo se encuentra en un fichero llamado *user.js* y, en ella, aparece el esquema de la información que se tiene que almacenar del usuario: *datos personales* y *análisis realizados*.

"Callback"

Una de las características más interesantes de la implementación de las funciones que se utiliza para el análisis es el apilamiento de funciones "callbacks". Esto se debe a que en el sistema existe la necesidad de recoger los resultados de una función asíncrona (segundo plano) para poder ejecutar otra función asíncrona.

En concreto, este sistema de llamadas se cumple cuando el sistema tiene que realizar el análisis de datos después de realizar el análisis de imagen, y este último después de almacenar la imagen en el sistema de ficheros del sistema. Es decir, cuando realiza una llamada asíncrono tras otra.

7.4.2 Análisis de datos

Todas las técnicas de análisis de datos presentados en el **capítulo 3.1** se corresponden exactamente con los diferentes análisis llevados a cabo para implementar el componente de análisis del entorno. Esto es la razón por la cual en este apartado solamente se explicarán dos aspectos de la implementación del análisis de datos:

- En el preprocesamiento de datos se han eliminado los caracteres especiales: RT, salto de línea, @... y todas aquellas palabras que son consideradas palabras vacías ("stopwords").

-El uso de "flags" es una consecuencia de la customización de los análisis que puede realizar los usuarios. Esto consiste en usar booleanos para especificar si se tiene que realizar un tipo específico de análisis. Dicho "flags" se pasan como parámetro en la petición http y después llega al servidor, el cual se encarga de enviarlo al programa R a través del módulo *r-script* de *Node.js*.

7.4.3 Aplicación Android

La aplicación de *Android* presenta cuatro elementos fundamentales: actividad, fragmento, tareas asíncronas y la representación de la interfaz a través de XML.

Actividades Android

Una actividad de Android no es más que una descripción lógica de una pantalla de la aplicación. En ella, se describe qué datos se representarán en la pantalla y cómo se van a representar estos datos. En la aplicación desarrolla para el sistema existen muchas actividades pero todos ellos se pueden clasificar en seis tipos:

- Actividades de autenticación: son actividades que permiten el registro y el inicio de sesión al sistema. Representan el punto de entrada de la aplicación, ya que si no se realiza la autenticación, no se podrá ejecutar ninguna funcionalidad del sistema.

-Actividad de Cámara: tal y como dice su nombre contiene toda la lógica de la cámara del dispositivo. Esta actividad es la que hace la petición al servicio web, y dicha tarea lo realiza después de sacar una imagen y almacenarlo en la memoria del dispositivo.

-Actividades de representación gráfica: son actividades que permiten llevar a cabo la representación gráfica de los resultados de análisis. Esta actividad utiliza las siguientes funciones para llevar a cabo dicha tarea:

a) *getData()*: permite obtener los datos del conjunto de datos que devuelve el servidor.

b) setData(): permite añadir los datos al gráfico.

c) configureChart(): permite configurar el diseño de las representaciones gráficas. Asimismo, también permite añadir interactividad en estos gráficos.

-Actividades que muestra los datos de las gráficas: se corresponden con las actividades que muestran más información acerca de los datos que se presentan en las representaciones gráficas. Se pueden distinguir los siguientes:

a) Información acerca de los “tweets”: muestra todas las palabras analizadas de este tweet, el sentimiento asociado a este tweet y su tópico.

b) Información acerca de las “palabras”: muestra los sentimientos asociados a esta palabra, las palabras relacionadas con el mismo y los “tweets” con los que está relacionado.

c) Información acerca de los “usuarios”: indica información acerca del usuario y sus tweets.

Fragmentos Android

Los fragmentos son una porción de la interfaz de una actividad que tiene su propio ciclo de vida. Al igual que las actividades, estos presentan mecanismos para presentar información y añadir interactividad a los diferentes elementos de la interfaz de usuario. Los fragmentos utilizados en esta aplicación se corresponden con aquellas que mejoran la navegabilidad del sistema. Estos fragmentos son:

- Fragmento que muestra el "dashboard": es un fragmento que presenta un resumen de los análisis realizados en el sistema. Es la pantalla que inicia el "storytelling" del sistema, forma de presentar la información de tal manera que el usuario sea capaz de comprenderlo completamente.

- Fragmento que muestra el listado de todos los gráficos: tal y como dice su nombre presenta todas las representaciones gráficas que puede realizar la aplicación móvil.

- Fragmento "acerca de": muestra información acerca de la aplicación móvil.

- Fragmento "perfil de usuario": muestra información acerca del usuario que ha iniciado sesión.

Tareas Asíncronas

Las tareas asíncronas son hebras que se ejecutan en segundo plano. Su principal utilidad en el desarrollo de la aplicación es hacer peticiones HTTP al servidor. Las tareas asíncronas que presenta el sistema sirven para el inicio de sesión, el registro de usuario, el análisis de una imagen y el análisis de un usuario de *Twitter*. Dado que todos estos requieren la realización de una llamada a las funciones del servidor.

Descripción de la interfaz a partir de XML

La descripción de la interfaz de usuario en una aplicación Android se realiza a través de XML. Dicho lenguaje de marcado se utiliza de una forma similar a HTML (en las páginas web). En XML se puede definir tanto los elementos que aparecen en la interfaz como las características que presenta estos elementos: márgenes, "padding", color, tamaño de letra…

Uso de D3.js

La aplicación usa webviews para poder realizar representaciones gráficas usando la herramienta "D3.js". Para ello, se ha almacenado una versión "minimizada" de la librería en la carpeta "assets" de la aplicación.

Las representaciones gráficas que se realizan usando esta herramienta son: diagrama de red, nube de palabras y matriz de adyacencia. Todas estas gráficas se han realizado con el fin de que la representación gráfica sea lo más eficiente posible dado que una solución no nativa, como ya se comentó en apartados anteriores, presenta problemas de rendimiento.

La necesidad de usar una solución no nativa quizás sea el punto débil de la aplicación. Sin embargo, la falta de una alternativa mejor para realizar dichos tipos de gráficos ha obligado el uso de esta herramienta. Pues, actualmente no existe una librería nativa de Android que sea capaz de llevar a cabo una representación gráfica de una calidad similar que *D3.js*.

Diseño de la interfaz de Usuario

Para finalizar este apartado de implementación se expone a continuación el uso de la guía de diseño "Material Design" para desarrollar diseñar la interfaz de usuario de la aplicación:

-Uso de "Responsive Grid Layout": la aplicación presenta la información del "dashboard" a través de un sistema de "grid" que se adapta a la cantidad de datos que tiene que presentar la interfaz de usuario.

-Uso de "Navigation Drawer": la aplicación presenta su principal herramienta de navegación a través de una barra de navegación lateral que se puede acceder pulsando el icono ☰.

-Uso de colores de "Material Design": los colores utilizados en la aplicación móvil se ha extraído de una herramienta de *Google*, autor de esta guía de diseño, que permite conocer qué colores pueden combinares para presentar la interfaz de usuario de la aplicación móvil.

8

Mantenimiento y Pruebas

En este capítulo se describen las pruebas realizadas sobre el servicio web y la aplicación móvil. Asimismo, se presenta una serie de pautas sobre cómo se recomienda realizar la labor de mantenimiento necesario para preservar el valor del sistema sobre el tiempo.

Por un lado, se han realizado pruebas de unidad sobre las funciones que atienden las peticiones realizadas sobre el servidor. Y, por otro lado, se han realizados encuestas para llevar a cabo una prueba de usabilidad de la interfaz de usuario de la aplicación.

8.1 Pruebas de Unidad

Las pruebas unitarias tienen como objetivo comprobar el correcto funcionamiento de todos los módulos (unidad de código) del sistema con el fin de identificar, analizar y corregir los defectos.

Estos tipos de pruebas lo suele llevar a cabo el desarrollador del programa, y ofrece como ventaja la posibilidad de capturar los errores en una fase temprana del

proceso de desarrollo, la reducción de defectos y la mejora del diseño del código, lo cual facilita la refactorización del mismo.

Las pruebas unitarias que se realizan para el entorno tiene como fin asegurar el correcto funcionamiento del Servicio Web. Para este fin, se lleva a cabo *pruebas funcionales sistemáticas de caja negra*, es decir, pruebas basadas en "especificaciones" teniendo en cuenta las posibles entradas y salidas más relevantes del programa. Este tipo de pruebas habitualmente se lleva a cabo siguiendo los siguientes pasos:

- Descomposición de la especificación, que consiste en analizar las características probables.
- Selección de representantes (valores de entrada)
- Producción de especiaciones (combinación de los valores de entrada)
- Ejecución de las pruebas

En la primera fase, se ha identificado las principales características del servicio web: inicio de sesión, registro, análisis de imagen y análisis de usuario. Cada una de estas funcionalidades se corresponden respectivamente con las siguientes rutas del servidor: "*/login*", "*/register*", "*/imageAnalysis*" y "*/userAnalysis*".

En la fase de selección de representantes se han podido identificar los valores de entrada que aparece en la siguiente tabla:

Rutas	Valores de entrada
/login	Email y Contraseña(registrado, no registrado o nulo)
/register	Nombre, Apellidos, Correo y Contraseña (no nulo o nulo)
/imageAnalysis	Imagen (no nulo o nulo)
/userAnalysis	Twitter (usuario real o usuario inexistente)

Tabla 8.1 Identificación de los valores de entrada

Teniendo en cuenta los valores obtenidos en la fase anterior se han identificado los siguientes casos de prueba:

- POST */login*:
  - a) Iniciar sesión usando credenciales válidos (usuario registrado).
  - b) Iniciar sesión usando credenciales no válidos (usuario no registrado).

c) Iniciar sesión sin credenciales.

- POST /*register*:

a) Registrarse con todos los campos obligatorios.

b) Registrarse sin cumplimentar algunos campos obligatorios.

- POST /imageAnalysis:

a) Analizar una imagen.

b) Enviar una petición sin imagen.

-POST /userAnalysis:

a) Analizar un usuario registrado en *Twitter*.

b) Analizar un usuario no registrado.

Por último, en la fase de ejecución de pruebas se puede decir que se han obtenido resultados satisfactorios dado que hay 83.52% de cobertura de líneas de código, 84.44% de cobertura de funciones y 83.7% de cobertura de sentencias. Esto es teniendo en cuenta que las dependencias que utiliza el servidor presenta posibles errores que son muy difíciles de alcanzar. Los errores detectados y corregidos gracias a esta fase de prueba de unidad son las siguientes:

- No se realiza una captura ("try-catch") de los errores que puede ocasionar el análisis de un usuario de *Twitter* no registrado.

- No se almacena correctamente los resultados del análisis de usuario en la base de datos.

8.2 Pruebas de Usabilidad

Las pruebas de usabilidad es una técnica usada en el diseño de interfaces gráficas de usuario para evaluar la usabilidad de aplicación haciendo pruebas con los usuarios mismos. Entendiendo por usabilidad "la medida en la que un producto se puede usar por determinados usuarios para conseguir objetivos específicos con efectividad, eficiencia y satisfacción en un contexto de uso especificado (ISO)". Existen diferentes métodos de evaluación: inspección, indagación y "test" basado en medidas de usabilidad. En concreto, se utilizará para evaluar la aplicación móvil el método de indagación y, más concretamente, la evaluación de usabilidad por medio de cuestionarios.

La evaluación de usabilidad por medio de cuestionarios es un método poco flexible (las preguntas no se adaptan a cada situación en la que se encuentra el usuario) pero puede llegar a un grupo más numeroso y se puede analizar con más rigor. Este método presenta 3 partes: cuestionarios "Pre-Test", "Post-Task" y "Post-Test", y se suele llevar a cabo con una muestra de 5 o más participantes.

Antes de realizar la prueba de usabilidad se explica la finalidad que tiene la aplicación a los participantes con el fin de que conozcan cuáles son los resultados que pueden obtener de la misma.

a) Cuestionario "Pre-Test"

Estos cuestionarios se realizan al principio de la prueba de usabilidad con el fin de conocer el perfil del participante. Las preguntas que aparecen en estos cuestionarios suelen ser muy específicos al tipo de aplicación que se ha implementado. A continuación, se presentan las preguntas realizadas en el cuestionario "Pre-test" de la aplicación móvil y, seguidamente, se expone una tabla con los resultados.

1. ¿Te consideras un usuario básico, intermedio o avanzado de las aplicaciones móviles? (Básico, Intermedio, Avanzado)
2. ¿Tienes un perfil relacionado con "marketing" o mercadotecnia? (Sí/No)
3. ¿Sabes en qué consiste el análisis de datos? (Sí/No)
4. ¿Tienes facilidad para interpretar las representaciones gráficas? (Sí/No)

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

Tabla 8.2 Resultados del Cuestionario "Pre-test"

b) Cuestionario "Post-Task"

En esta segunda fase, se recogen opiniones y valoraciones de cada tarea que se puede realizar sobre la aplicación. Las preguntas que aparecen en estos

cuestionarios suelen ser genéricas. Se expone, a continuación, las preguntas utilizadas en el cuestionario "Post-Task" de la aplicación:

1. ¿Ha sido fácil completar la tarea? (Muy fácil/Fácil/Normal/Díficil/Muy Díficil)
2. ¿Has usado el manual para completar la tarea?(Sí/No)
3. Si has utilizado el manual ¿La información ha sido fácil de encontrar? (Muy fácil/Fácil/Normal/Díficil/Muy Díficil)
4. ¿La información que encontraste en el manual ha sido fácil de utilizar? (Muy fácil/Fácil/Normal/Díficil/Muy Díficil)

Las tareas que se han elegido para llevar a cabo esta encuesta se corresponden con las tareas generales que el usuario de la aplicación puede llevar a cabo. Presentan un carácter general puesto la mayoría de las actividades que se puede realizar usando la aplicación se pueden a ajustar a estos tipos de tareas.

Las tareas que se analizan en este apartado son las siguientes:

1. Analizar una Imagen
2. Analizar un Usuario de Twitter
3. Interpretar los resultados del "dashboard"
4. Interpretar una gráfica

En las tablas que se presentan a continuación se puede encontrar el resultado de los cuestionarios "Post-Task" en función de las tareas que tiene que realizar el usuario:

Analizar una imagen

	Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 4
Usuario 1	Normal	No	-	-
Usuario 2	Fácil	No	-	-
Usuario 3	Fácil	No	-	-
Usuario 4	Normal	No	-	-
Usuario 5	Normal	No	-	-

Tabla 8.3 Resultados del Cuestionario "Post-Task" - Analizar una imagen

Analizar un usuario de *Twitter*

	Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 4
Usuario 1	Normal	Sí	Fácil	Normal
Usuario 2	Muy Difícil	Sí	Normal	Muy Difícil
Usuario 3	Normal	Sí	Normal	Normal
Usuario 4	Muy Difícil	Sí	Difícil	Normal
Usuario 5	Muy Difícil	Sí	Difícil	Normal

Tabla 8.4 Resultados del Cuestionario "Post-Task" - Analizar un usuario de *Twitter*

Interpretar los resultados del "dashboard"

	Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 4
Usuario 1	Fácil	No	-	-
Usuario 2	Normal	No	-	-
Usuario 3	Fácil	No	-	-
Usuario 4	Normal	No	-	-
Usuario 5	Fácil	No	-	-

Tabla 8.5 Resultados del Cuestionario "Post-Task" - Interpretar los resultados del "dashboard"

Interpretar una gráfica

	Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 4
Usuario 1	Normal	No	-	-
Usuario 2	Fácil	Sí	Normal	Normal
Usuario 3	Fácil	Sí	Fácil	Normal
Usuario 4	Normal	No	-	-
Usuario 5	Fácil	No	-	-

Tabla 8.6 Resultados del Cuestionario "Post-Task" – Interpretar una gráfica

c) Cuestionario "Post-Test"

En los cuestionarios "Post-Test" se recogen opiniones y valoraciones después de que los participantes completen todas las tareas. Para realizar estos cuestionarios se suelen utilizar las preguntas del estándar *SUS* y *CSUQ*. En concreto, para las pruebas de esta aplicación se utilizará la plantilla del primero:

1. Creo que me gustará usar con frecuencia esta aplicación. (1/2/3/4/5)
2. Encontré la aplicación innecesariamente complejo. (1/2/3/4/5)
3. Pensé que la aplicación era fácil de usar. (1/2/3/4/5)

4. Creo que necesitaría del apoyo de un experto para utilizar la aplicación. (1/2/3/4/5)
5. Encontré las diversas posibilidades de la aplicación bastante bien integradas. (1/2/3/4/5)
6. Pensé que había demasiada inconsistencia en la aplicación. (1/2/3/4/5)
7. Imagino que la mayoría de las personas aprenderían muy rápidamente a utilizar la aplicación. (1/2/3/4/5)
8. Encontré la aplicación muy grande (pesado) al recorrerlo. (1/2/3/4/5)
9. Me sentí muy confiado en el manejo de la aplicación. (1/2/3/4/5)
10. Necesito aprender muchas cosas antes de manejarme en la aplicación. (1/2/3/4/5)

Según el estándar el cálculo del resultado de la encuesta se realiza de la siguiente forma:

> *"Primero suma las contribuciones de puntuación de cada elemento. La contribución de cada puntaje de cada ítem variará de 0 a 4. Para los ítems 1,3,5,7 y 9, la contribución de puntaje es la posición de escala menos1. Para los ítems 2,4,6,8 y 10, la contribución es 5 menos la posición de la escala. Multiplique la suma de los puntajes por 2.5 para obtener el valor total de SUS".* (Brooke, s.f.)

En la tabla que se presenta a continuación se presenta los resultados obtenidos:

	Usuario 1	Usuario 2	Usuario 3	Usuario 4	Usuario 5
Pregunta 1	5	4	4	4	5
Pregunta 2	3	4	3	3	3
Pregunta 3	4	3	4	5	4
Pregunta 4	2	3	2	3	4
Pregunta 5	4	3	2	4	3
Pregunta 6	3	4	3	3	4
Pregunta 7	3	4	4	3	3
Pregunta 8	4	5	3	4	3
Pregunta 9	4	3	4	3	3
Pregunta 10	3	4	3	2	4
Resultados	62.5	42.5	60	60	50

Tabla 8.7 Resultados del Cuestionario "Post-Test"

d) Conclusiones

En los cuestionarios "Pre-Test" se ha podido determinar el perfil de los participantes de la prueba de usabilidad. Hay un usuario avanzado, dos usuarios intermedios y 2 usuarios básicos. El usuario avanzado tiene todas las habilidades para usar correctamente la aplicación. Un usuario intermedio no sabe en qué consiste el análisis de datos, pero tiene un perfil relacionado con la mercadotecnia. El otro usuario intermedio no dispone de conocimientos de "marketing" ni de análisis de datos, pero tiene la facilidad para interpretar representaciones gráficas. Los dos usuarios básicos tienen un perfil relacionado con la mercadotecnia, no saben en qué consiste el análisis de datos, pero tienen facilidad para interpretar los gráficos.

En la segunda fase de la prueba de usabilidad, cuestionarios "Post-Task", se puede observar cómo la mayoría de las tareas del sistema se puede realizar sin la necesidad de consultar el manual. Esto es cierto salvo para la tarea "Analizar un usuario de Twitter", ya que para llevar a cabo esta tarea hay que realizar una serie de pasos que complica la realización de la tarea.

Por último, en la tercera fase de la prueba, cuestionarios "Post-Test" se puede observar que la media de los resultados es 55. Esto significa que la interfaz desarrollada y, en general, la aplicación ha obtenido resultados suficientes para superar los criterios básicos de calidad del estándar SUS. Sin embargo, las puntuaciones SUS tienen un rango de entre 0 y 100 y que, por tanto, la interfaz de la aplicación requiere una considerable mejora.

8.3 Mantenimiento de Software

El mantenimiento de software se corresponde con aquellas tareas asociadas con la modificación del producto software después de su entrega y que van destinadas a cumplir nuevos requisitos/prestaciones, a corregir fallos o a adaptar al producto a un nuevo entorno (Durán Muñoz).

La mayoría de los proyectos software no suelen considerar el mantenimiento como una parte del ciclo de vida del desarrollo software. Sin embargo, dado las

características que presenta el sistema, el mantenimiento es indudablemente una fase importante de este proyecto.

En este apartado, se presentan las diferentes técnicas y los diferentes métodos que se deben usar para llevar a cabo la tarea de mantenimiento del sistema. En los siguientes párrafos se describe un plan para llevar a cabo el mantenimiento del sistema:

- Comprensión del sistema: esta tarea consiste en comprender todas las características del sistema. Los métodos que se propone para llevar a cabo esta tarea son: la lectura completa de este documento y la Visualización Software, que consiste en realizar representaciones visuales de la información de los sistemas con el objetivo de mejorar su comprensión.

- Reingeniería de software: esta tarea consiste en mejorar una funcionalidad del sistema a partir de la introducción de nuevas tecnologías y prácticas. Se recomienda hacer esta tarea usando las siguientes técnicas:

a) "Revamping": consiste en llevar a cabo cambios a la interfaz de usuario con el fin de mejorar su usabilidad. Esta técnica es fundamental dado que los resultados obtenidos de la prueba de usabilidad han demostrado una clara necesidad de mejorar la interfaz.

b) Uso de Componentes comerciales (COTS): consiste en sustituir una parte del sistema por componentes comerciales desarrollados por terceras partes. Esto puede muy interesante para el "revamping", pues existen muchas librerías para mejorar la interfaz de la aplicación y las representaciones gráficas.

c) Reducción de código: consiste en eliminar métodos y trozos de código redundantes del sistema. Existen muchas herramientas para llevar a cabo esta tarea como por ejemplo: *CloneDR*. Esta técnica es muy importante dado que el sistema presenta miles de líneas de código.

d) Transformaciones funcionales: consiste en mejorar la estructura del programa a través de cambios estructurales del código, la redefinición de la estructura de los módulos y la modificación del tratamiento de datos. Las transformaciones funcionales pueden ser muy interesantes ya que el sistema presenta muchos componentes y un cambio en el tratamiento de la información puede hacer más eficiente el proceso de análisis del sistema.

Para determinar qué técnica y cómo y cuándo hay que aplicarla, se recomienda trabajar en función del flujo de proceso de mantenimiento tradicional cuyas etapas se corresponde con:

- Identificación de cambio.
- Análisis de impacto.
- Planificación de versiones (corrección de errores, adaptación de plataforma y perfeccionamiento del sistema).
- Implementación de cambios.

9

Conclusiones

En este último capítulo de la memoria se exponen los resultados del proyecto, los conocimientos adquiridos durante su proceso de desarrollo, las dificultades encontradas y las posibles mejoras que se pueden incorporar en el sistema.

9.1 Resultados

Tras el seguimiento en el presente documento del ciclo de desarrollo del proyecto "Un entorno para el análisis de sentimientos a partir de una imagen", es conveniente destacar las conclusiones objetivas que se han alcanzado a su conclusión.

Partiendo desde el análisis de requisitos inicial y siguiendo por el conjunto de requisitos funcionales y no funcionales que se debían cumplir para alcanzar los criterios de aceptación del sistema, se puede concluir que se ha alcanzado un nivel satisfactorio de desarrollo.

Este nivel de satisfacción respecto a los criterios requeridos para la entrega de este proyecto se alcanza gracias al siguiente conjunto de afirmaciones:

- Se presentan las necesidades que pretenden cubrir el sistema.
- Se concreta una solución para cubrir dichas necesidades.
- Se explican las técnicas necesarias para llevar a cabo dicha solución.
- Se define la consecución del sistema como un proyecto.
- Se fija el proceso que se tiene que utilizar para llevar a cabo el proyecto.

- Se especifican todas las características que ha de presentar el sistema.
- Se exponen los planos que definen el comportamiento y la estructura del sistema.
- Se detalla la implementación del sistema.
- Se presentan las pruebas realizadas para garantizar el correcto funcionamiento del sistema.
- Se elabora un plan para el mantenimiento del entorno.

9.2 Conocimientos adquiridos

La elaboración de una aplicación capaz de realizar el conjunto de funcionalidades descrito anteriormente, siempre basado en un análisis de sentimientos, en un análisis de datos, profundo y que caracterice la descripción informativa de su idea origen ha conllevado un fuerte aprendizaje a nivel de desarrollo.

En esta etapa académica no solo se han conocido nuevas herramientas y conocimientos de programación, sino que también se han adquirido diversas e importantes competencias a nivel de gestión de proyectos, desarrollo software con seguimiento de toda su línea de vida y, sobre todo, capacidades para ser capaz de eludir los distintos obstáculos que se puedan presentar en el mismo.

Independientemente de este conocimiento adquirido a nivel de ingeniería de software, se pueden mencionar las siguientes características técnicas aprendidas:
- Aprendizaje de técnicas necesarias para llevar a cabo las tareas de análisis de datos.
- Aprendizaje del uso de librerías de R que implementan las técnicas de análisis.
- Aprendizaje del uso de herramientas para llevar a cabo representaciones gráficas.
- Adquisición de conocimientos sobre la base de datos *NoSQL* orientado a documentos, *MongoDB*.
- Adquisición de un conocimiento sólido del entorno Android.

9.3 Dificultades encontradas

Desarrollar el sistema no fue una tarea fácil. A lo largo todo el proceso de consecución del proyecto se han presentado problemas que no solamente perjudican el plan del proyecto, sino que también afecta las características más esenciales del sistema.

En los párrafos que se presentan a continuación, se exponen dichos problemas junto con las soluciones que se han adoptado para resolverlos.

- Uso de *rtweet*: esta librería tuvo un problema con el uso de token de la API de *Twitter*. Dicho consiste en usar un mismo token de la aplicación para la versión de desarrollo de la librería y la versión que utilizan los usuarios finales.
La solución más inmediata a este problema es la utilización de la versión anterior de la librería *TwitteR*. Aunque, finalmente se optó por utilizar un token personal del autor del proyecto para volver a trabajar con la librería *rtweet*.

- Uso de *r-script*: esta librería permite la comunicación del servidor Node.js con el entorno R. Sin embargo, dicha librería sólo era capaz de recoger resultados de hasta 300 caracteres y el sistema requería mucho más que dicha cantidad para producir los resultados de análisis.
La solución adoptada para resolver este problema es la modificación de la librería para que esta pueda permitir más de 300 caracteres de resultado.

- Búsqueda de una herramienta para representar gráficos avanzados: como ya se comentó en apartados anteriores se tuvo que utilizar una solución no nativa al sistema operativo Android por la falta de librerías capaces de llevar a cabo representaciones gráficas más avanzadas.

- Uso de la *wordcloud*: esta librería Android no dispone de mecanismos para hacer más interactivo la representación gráfica de nube de palabras.
Para resolver este problema, se realizó una modificación de la librería con el fin de agregar métodos que permiten interaccionar con los elementos de la gráfica.

9.4 Posibles mejoras

A lo largo del desarrollo del proyecto, tanto el autor del proyecto como su tutor han ideado una serie de posibles mejoras que pueden incrementar el valor del sistema.
- La incorporación de nuevos tipos de análisis.
- La mejora de la calidad de los resultados de análisis.
- La utilización de librerías nativas para implementar las representaciones gráficas.

- La mejora del diseño de la interfaz de usuario.
- La Incorporación un análisis de mayor duración.

De entre estas posibles líneas futuras, se ha prestado especial atención al último. Pues, implementar dicha funcionalidad permitiría incrementar la calidad de los resultados de análisis del sistema.

Referencias

ALTONIVEL. (2013, 07 30). *9 puntos de la era digital que han cambiado al marketing*. Retrieved from https://www.altonivel.com.mx/empresas/37259-9-cambios-de-la-era-digital-que-han-cambiado-al-marketing/

Auth0. (2017). *Introduction to JSON Web Tokens*. Retrieved from https://jwt.io/introduction/

Botía, J. A. (n.d.). *Preprocesado de Datos*. Retrieved from http://webs.um.es/juanbot/miwiki/lib/exe/fetch.php?id=tiia0809&cache=cache&media=tiia0809_slides_prep.pdf

Broder, A. Z., Glassman, S. C., & Zweig, G. (1997). *Syntactice clustering of the web.*

Brooke, J. (n.d.). *SUS: A Quick and Dirty Usability Scale*. Retrieved from https://hell.meiert.org/core/pdf/sus.pdf

Charc-concepts. (2012, 10 16). *Data Mining - Learn all about datamining*. Retrieved from http://charc-concepts.org/the-benefits-of-data-mining/

Durán Muñoz, F. J. (n.d.). Mejora de los Procesos de Ingeniería del Software en Tecnatom.

García, P. (2008). *Estudio de Viabilidad*. Retrieved from http://revistaselectronicas.ujaen.es/index.php/pruebas/article/download/2/3

Google Android Developers. (2017). *Android API guides*. Retrieved from https://developer.android.com/guide/index.html

Google Cloud Platform Developers. (2017). *Google Cloud Vision API*. Retrieved from https://cloud.google.com/vision/docs/

KDnuggets. (2016, 07). *Topic Modelling*. Retrieved from https://www.kdnuggets.com/2016/07/text-mining-101-topic-modeling.html

L. Bass, P. C. (1998). *Software Architecture in Practica.* Addison Wesley.

Love, R. I. (n.d.). *dplyr tutorial*. Retrieved from http://genomicsclass.github.io/book/pages/dplyr_tutorial.html

MongoDB Inc. (n.d.). *MongoDB Manual*. Retrieved from https://docs.mongodb.com/manual/

Mozilla y colaboradores individuales. (2017). *Guía de Javascript*. Retrieved from https://developer.mozilla.org/es/docs/Web/JavaScript/Guide

Node.js Foundation. (n.d.). *Node js*. Retrieved from https://nodejs.org/es/

Pereira, J. E. (2012, 12 30). *Transformación del Marketing en la Era Digital*. Retrieved from http://www.mercadeo.com/blog/2012/12/transformacion-del-marketing-en-la-era-digital/

PhilJay. (2017, 03 23). *MPAndroidChart Wiki*. Retrieved from https://github.com/PhilJay/MPAndroidChart/wiki

R Development Core Team. (2017, 09 28). *The R Manuals*. Retrieved from https://cran.r-project.org/manuals.html

Robinson, J. S. (2017, 05 07). *A Tidy Approach*. Retrieved from http://tidytextmining.com/
Rouse, M. (2010, 07). *Opinion Mining*. Retrieved from http://searchbusinessanalytics.techtarget.com/definition/opinion-mining-sentiment-mining
StrongLoop, Inc. (n.d.). *Express API 4.X*. Retrieved from http://expressjs.com/es/4x/api.html
The R Foundation. (n.d.). *The R Project for Statistical Computing*. Retrieved from https://www.r-project.org/
Tufte, E. (1983). *The Visual Display of Quantitative Information.* Kansas City, Misuri: Graphics Press.
Twitter Developers. (2017). *Twitter Docs*. Retrieved from https://developer.twitter.com/en/docs
Veracode. (n.d.). *Software Development Lifecycle (SDLC)*. Retrieved from https://www.veracode.com/security/software-development-lifecycle
Vialcanet, G. (2017). *Visualización de Datos ¿Qué dicen los Expertos?* Retrieved from http://dbi.io/es/blog/visualizacion-de-datos-que-dicen-los-expertos/
Wickham, H. (n.d.). *cran.r*. Retrieved from plyr: Tools for Splitting, Applying and Combining Data: https://cran.r-project.org/web/packages/plyr/index.html
Zhao, Y. (n.d.). Retrieved from RDatamining.com: R and Data Mining: http://www.rdatamining.com/home

Apéndice A

Planificación

FASE	HORAS ESTIMADAS	HORAS DESMPEÑADAS
1. Elaboración de Documentación Inicial	**Total (fase): 41**	**Total (fase): 37.5**
1.1. Describir el contexto del proyecto	**2**	**1**
1.2. ¿Qué problemas resuelve este sistema?	**2**	**1**
1.3. Indicar el propósito del proyecto	**2**	**1**
1.4. Realizar una descripción general del proyecto	**7**	**6**
1.5. Enumerar las suposiciones y restricciones del trabajo	**2**	**1**
1.6. Identificar los posibles riesgos que pueden surgir y proponer soluciones	**3**	**2**
1.7. Listar las tecnologías utilizadas	**1**	**2**
1.8. Realizar un análisis del mercado	**2**	**1.5**
1.9. Hacer un estudio de viabilidad	**2**	**1**
1.10. Proponer criterios de éxito	**2**	**1**
1.11. Realizar Ingeniería de requisitos a partir de la descripción general del proyecto	**5**	**8**
1.12. Establecer los distintos casos de uso del sistema	**3**	**3**
1.13. Elaborar un diagrama para describir la estructuración del sistema	**5**	**5**
1.14. Describir el flujo de funcionamiento del programa	**3**	**4**

2. Desarrollo de la primera versión del backend	**Total (fase): 40**	**Total (fase): 36**
2.1. Estudio de las tecnologías que vamos a utilizar	**10**	**8**
2.2. Creación del Servidor Node js con Express js	**1**	**1**
2.3. Creación de Servicio RESTful	**Total (sub-fase): 9**	**Total (sub-fase): 9**
2.3.1. Obtención de la imagen recibida en la petición HTTP	**1**	**1**
2.3.2. Almacenamiento temporal de la Imagen usando la dependencia Formidable	**2**	**2**
2.3.3. Análisis de la Imagen recibida mediante la API visión de Google	**3**	**3**
2.3.4 Incorporación de R al servidor de Node JS mediante la dependencia rscript	**3**	**3**
2.4. Análisis de Datos usando R	**Total (sub-fase): 17**	**15**
2.4.1. Obtención de Tweets trabajando con la API de Twitter	**2**	**2**
2.4.2. Usuarios más implicados	**1**	**1**
2.4.3. Análisis Sentimental	**5**	**3**
2.4.4. Frecuencia de términos	**3**	**3**
2.4.5. Tokenización n-grams y Correlaciones	**3**	**3**
2.4.6. Topic Modelling	**3**	**3**
2.5. Parsear los data frames de R y devolver los Resultados en formato JSON.	**3**	**3**
3. Desarrollo de la segunda versión del backend	**Total (fase): 19**	**Total (fase): 15**
3.1. Estudio de mecanismos para securizar las rutas del servidor	**5**	**4**
3.2. Proteger las rutas del servidor usando JSON Web Tokens	**3**	**2**
3.3. Creación de la base de datos	**3**	**3**
3.4. Incorporación del módulo Mongoose y creación del esquema de usuario para su mapeo en la base de datos	**3**	**1**
3.5. Implementación del sistema de login y registro	**2**	**2**
3.6. Código para almacenar los resultados del análisis en la base de datos	**3**	**3**

4. Desarrollo de la primera versión de la aplicación Android	**Total (fase): 36**	**Total (fase): 37.5**
4.1. Configuración del proyecto Android	**1**	**1**
4.2. Estructuración de la interfaz gráfica de usuario (GUI)	**2**	**2**
4.3. Representación gráfica	**Total (sub-fase): 26**	**Total (sub-fase): 29.5**
4.3.1. Estudio de librerías para realizar representaciones gráficas	**5**	**5**
4.3.2. Diagrama de barras para los usuarios más implicados	**1**	**0.5**
4.3.3. Nube de palabras para expresar la frecuencia de términos	**2**	**4**
4.3.4. Diagrama de red para representar la relación entre palabras	**5**	**9**
4.3.5. Elaborar un diagrama de sectores del resultado del análisis de sentimientos	**5**	**1**
4.3.6. Representar los sentimientos en función del tiempo	**5**	**4**
4.3.7. Representación gráfica del modelo de tópicos	**3**	**6**
4.4. Creación de la actividad de la cámara	**3**	**3**
4.5. Menú para configurar los resultados del análisis	**4**	**2**
5. Desarrollo de la tercera versión del backend	**Total (fase): 40**	**Total (fase): 32**
5.1. Elaborar características para mejorar el resultado del análisis. Estas características dependen de las carencias observadas en las iteraciones anteriores.	**10**	**12**
5.2. Añadir mecanismos para permitir la configuración de los resultados del análisis	**10**	**5**
5.3. Hacer más eficiente los análisis. Intentar reducir el tiempo de respuesta. Introducir procesamiento paralelo	**10**	**5**
5.4. Añadir nuevos tipos de análisis. Es muy probable que se tenga que añadir nuevos tipos de análisis después de ver la primera versión del sistema (con backend y frontend)	**10**	**10**

6. Desarrollo de la segunda versión de la aplicación Android	**Total (fase): 32**	**47**
6.1. Mejorar las representaciones gráficas	**8**	**5**
6.2. Añadir más parámetros de configuración en el menú	**8**	**4**
6.3. Añadir nuevas representaciones gráficas	**8**	**8**
6.4. Reestructurar el interfaz de usuario de la aplicación para que sea más amigable para el usuario	**8**	**30**
7. Refinar el funcionamiento del sistema	**Total (fase): 45**	**Total (fase): 54**
7.1. Mejorar la estructuración del código (refactorizar el código)	**10**	**10**
7.2. Mejorar la interactividad de la aplicación	**6**	**25**
7.3. Realizar pruebas para asegurar el funcionamiento correcto del sistema	**15**	**8**
7.4. Corregir los errores detectados	**8**	**8**
7.5. Asegurar que todo el sistema funciona correctamente	**6**	**3**
8. Finalizar la documentación	**Total (fase): 42**	**Total (fase): 41**
8.1. Describir los resultados de la planificación	**4**	**4**
8.2. Indicar las características finales del sistema elaborado	**10**	**15**
8.3. Enumerar los problemas que han surgido durante el proceso de desarrollo	**5**	**5**
8.4. Indicar cuáles son las ideas descartadas y cuáles son las mejoras introducidas respecto a la planificación inicial	**8**	**6**
8.5. Resultado de las pruebas	**8**	**4**
8.6. Indicar posibles mejoras	**4**	**4**
8.7. Conclusiones	**3**	**3**
	Total Estimado: 296	**Total Desempeñado: 300**

Apéndice B

Manual de Instalación

Instalación del Servidor Web:

En este apartado se describe los pasos que hay que seguir para instalar el servidor:

1. Instale Node.js a través de su página web oficial: https://nodejs.org/es/

2. Copie la carpeta "senimage" del disco en un directorio del servidor.

3. Utilizando la línea de comando ejecute el comando *npm install* dentro de la carpeta "senimage" que acaba de copiar del disco.

4. Instale MongoDB a través de su página oficial: https://docs.mongodb.com/manual/installation/

5. Cree la base de datos "senimage" en MongoDB. Para ello, ejecute la sentencia *use senimage* en la línea de comando de Mongodb (para acceder la línea comandos de MongoDB hay que ejecutar el comando *mongo* en la línea de comando de su sistema operativo). Asegúrese que la base de datos se puede acceder a través de la ruta: mongodb://localhost/senimage (normalmente esto es la ruta que se crea cuando se ejecute el comando use senimage).

6. Modifique el fichero "/node_modules/r-script/index.js" para que la función *R.prototype.call* (se encuentra a partir de la línea 27) tenga el siguiente aspecto:

```
R.prototype.call = function(_opts, _callback) {
  var callback = _callback || _opts;
  var opts = _.isFunction(_opts) ? {} : _opts;
  this.options.env.input = JSON.stringify([this.d, this.path, opts]);
  var child = child_process.spawn("Rscript", this.args, this.options);
  var stdout = "";
  var stderr = "";
  child.stderr.on("data", function(d){
    stderr+=d.toString();
  });
  child.stdout.on("data", function(d) {
    stdout += d;
  });
  child.on('close',function(code){
    callback(
      stderr?stderr:null;
      stdout?JSON.parse(stdout) : {}
    );
  });
};
```

7. Configure la variable de entorno para la API Vision de Google. Para ello:

-En Windows Powershell ejecute el siguiente comando:

$env:GOOGLE_APPLICATION_CREDENTIALS="[PATH]".

- En Linux y de macOS ejecute el siguiente comando:

export GOOGLE_APPLICATION_CREDENTIALS="[PATH]".

Nota: [PATH] es la ruta de la API Key de la API Vision de Google. Puede obtener una API Key siguiente los pasos que aparecen en la siguiente página: https://cloud.google.com/vision/docs/auth

8. Instale la versión más actualizada de R a través de su página oficial: https://www.r-project.org/

9. En la línea de comando de R (que se accede ejecutando el comando *R* en la línea de comando de su sistema operativo), tiene que instalar los paquetes que utiliza el entorno de análisis utilizando el comando *install.packages("nombre de la librería")*. Las librerías que hay que instalar son:

-httr

-rtweet

-tm

-dplyr

-plyr

-RSentiment

-tidytext

-tidtr

-topicmodels

10. Obtenga un token de la API de Twitter siguiendo los pasos que aparecen en la página oficial de los desarrolladores de Twitter: https://developer.twitter.com/en/docs/basics/authentication/guides/access-tokens.html

Nota: Para realizar este paso necesita una cuenta de *Twitter*.

11. Modifique el fichero "/scripts/Analyze.R" y "/scripts/UserAnalyze.R" para que la sentencia:

Token <- readRDS('/home/john/twitter_token.rds')

Tenga el siguiente aspecto:

Token <- readRDS('[PATH]')

Nota: [PATH] en este caso es la ruta de la API Key de *Twitter*.

12. Para ejecutar el servidor instale el módulo forever ejecutando el comando *npm install forever -g*. En Linux probablemente tenga que utilizar *sudo*.

13. Ejecute la base de datos. Para ello, siga los pasos (cambia en función del sistema operativo) de la documentación de MongoDB: https://docs.mongodb.com/manual/tutorial/manage-mongodb-processes/

14. Ejecute el servidor utilizando ejecutando el comando *forever start index.js* en la línea de comando de su servidor. Asegúrese que la línea de comando se encuentra en la ruta raíz de la carpeta index.js.

Instalación de la Aplicación Móvil:

Para instalar la aplicación móvil solamente hay que descargar en el móvil el archivo ".apk" que se encuentra dentro del disco.

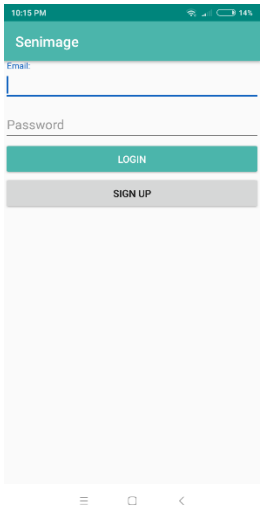
Apéndice C

Manual de Usuario

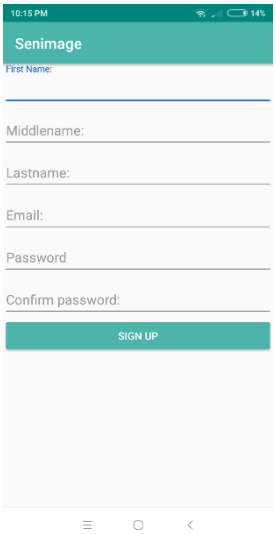
¿Cómo me registro?

Para empezar a utilizar los servicios de la aplicación hay que seguir los siguientes pasos:

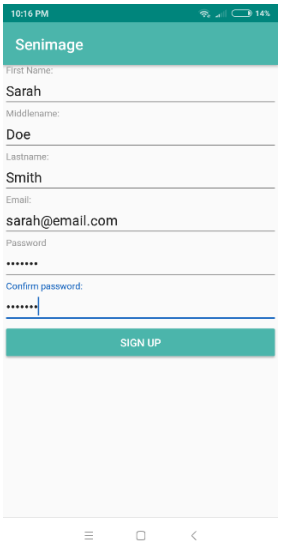
1. Al ejecutar la aplicación se le muestra la siguiente pantalla:



2. Pulse el botón “Sign up” para acceder el formulario del registro a la aplicación:



3. Rellene todos los campos del formulario.

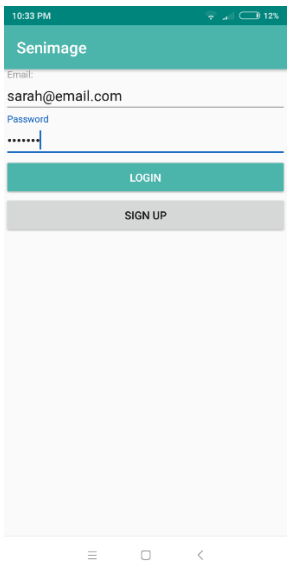


4. Pulse de nuevo el botón “Sign up” tras rellenar el formulario.

5. Si todo ha ido correcto, se le mostrará la galería de imágenes de la aplicación.

Y si ya tengo una cuenta ¿Cómo puedo iniciar de sesión?

Si dispone de una cuenta, para iniciar de sesión únicamente tendrá que acceder la primera pantalla de la aplicación y rellenar el formulario con sus credenciales: correo electrónico y contraseña.



Tras rellenar el formulario, pulse el botón “Login”. Si las credenciales son correctas accederá la galería de imágenes de la aplicación.

¿Cómo puedo realizar un análisis de imagen?

Existen dos formas de llevar a cabo un análisis de imagen:

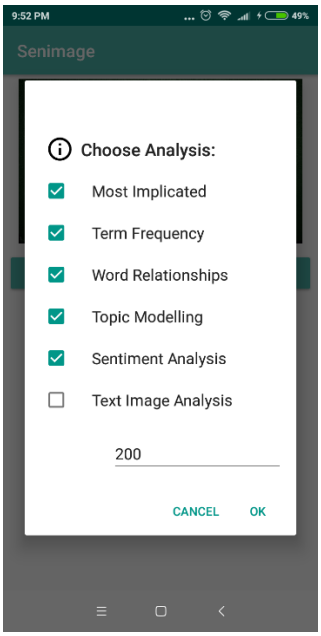
Seleccionando una imagen de la galería de imágenes

1. Después de iniciar sesión o de registrarse, se encontrará en la galería de imágenes de la aplicación.

2. Seleccione la imagen que desea analizar.

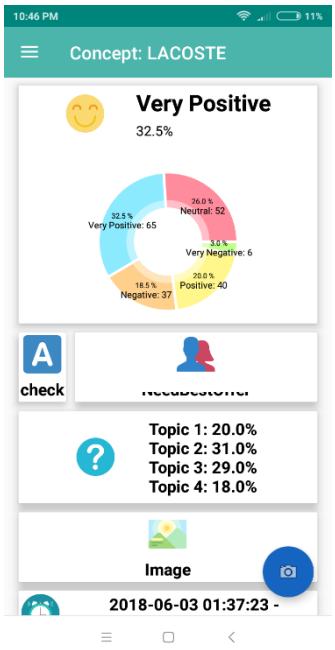
3. Pulse el botón “Analyze Image”.

4. Seleccione las características del análisis que desea realizar.



5. Para continuar el análisis pulse el botón “Ok”.

6. Si el proceso se ha realizado correctamente, se le mostrará la siguiente pantalla que contiene un resumen del análisis ejecutado:



Sacando una foto del objeto que desea analizar

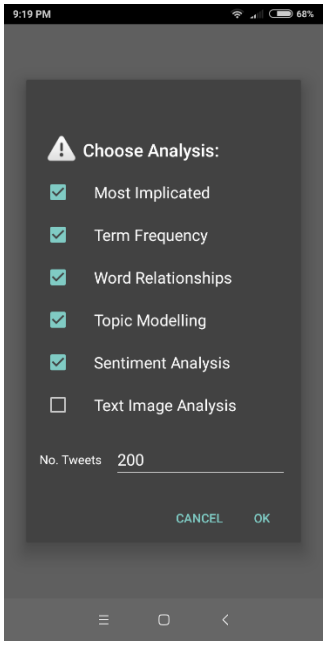
1. Después de iniciar sesión o de registrarse se encontrará en la galería de imágenes de la aplicación.

2. Pulse el botón que tiene el icono de una cámara (botón azul) situado en la parte inferior derecha de la pantalla.

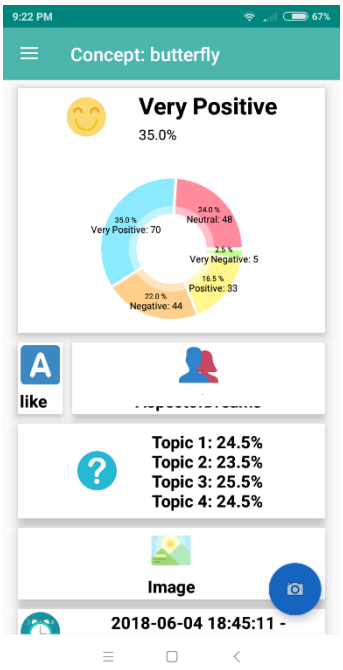
3. Al pulsar el botón le saldrá una pantalla para capturar imagen usando su dispositivo.

4. Capture una imagen del objeto que desea analizar (en este caso, se analizará la imagen de una mariposa).

5. Seleccione los tipos de análisis que desea realizar.



6. Si el proceso se ha realizado correctamente, se le deberá aparecer una pantalla que contiene el resultado del análisis realizado.

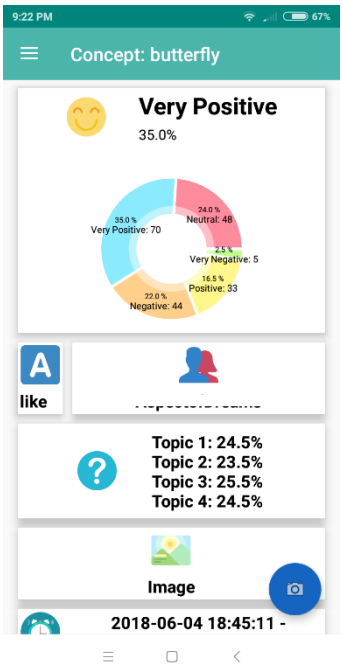


¿Cómo puedo ver las representaciones gráficas que permiten visualizar los resultados del análisis?

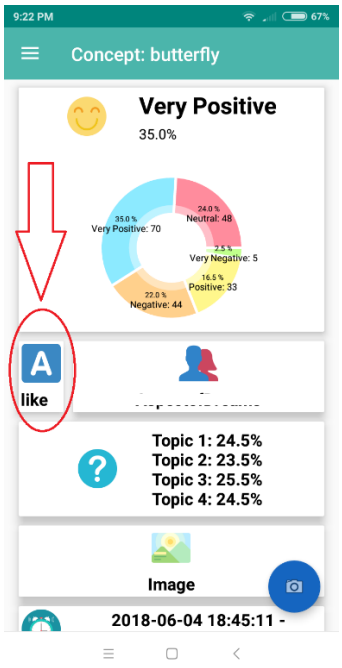
Existen dos formas de acceder las representaciones gráficas:

A partir del "Dashboard"

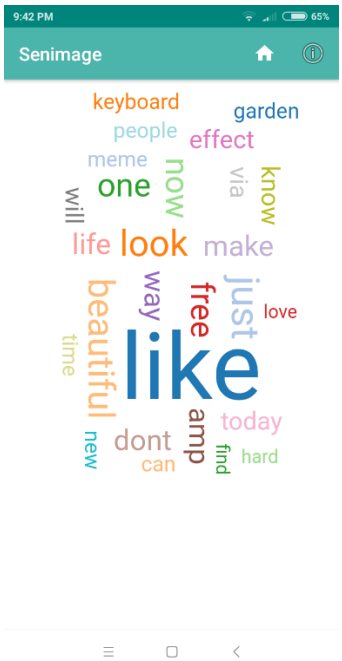
1. Partiendo de la pantalla de resumen (el "Dashboard").



2. Seleccione el resultado de análisis que desea conocer con más profundidad. Por ejemplo: en este caso, se desea conocer cuáles son las palabras más frecuentemente usadas acerca del concepto que se está analizando.



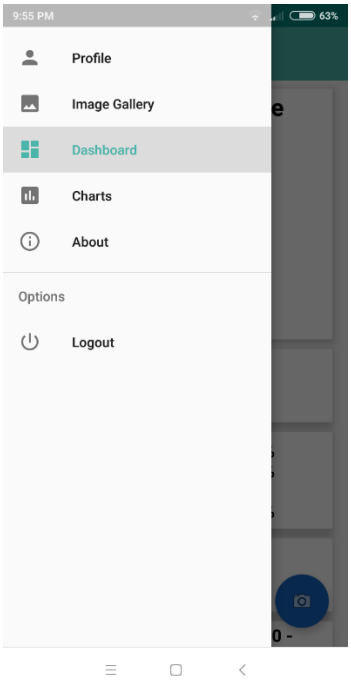
3. Si todo se realizado correctamente podrá visualizar los resultados de análisis a partir de una gráfica. En este caso, se le mostrará una nube de palabras.



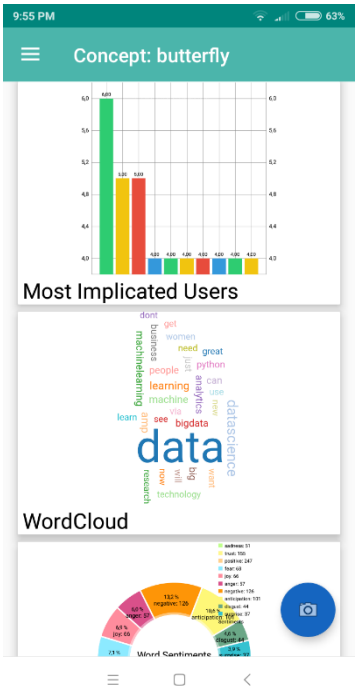
A partir de la lista de análisis

1. Para acceder de la lista de análisis tendrá que abrir la barra de navegación lateral de la aplicación desde la pantalla de resumen de los resultados de análisis.

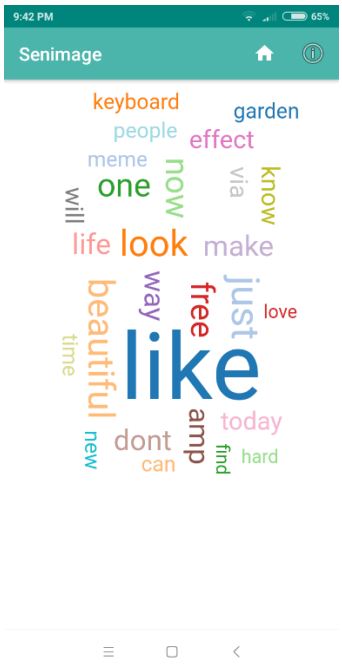
2. Seleccione la opción “Charts” para poder acceder todas las representaciones gráficas.



3. Seleccione el gráfico que desea visualizar. En este caso se seleccionará el gráfico “WordCloud”.



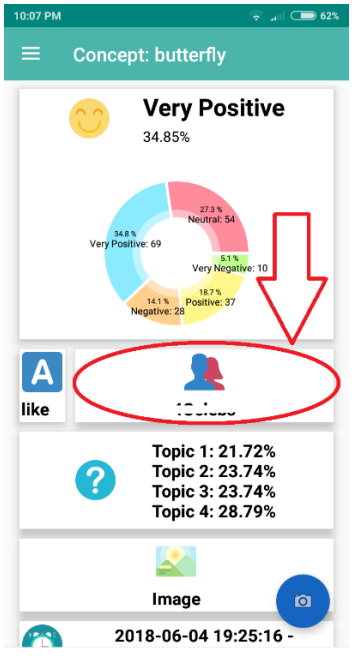
4. Si todo correctamente, podrá visualizar la nube de palabras.



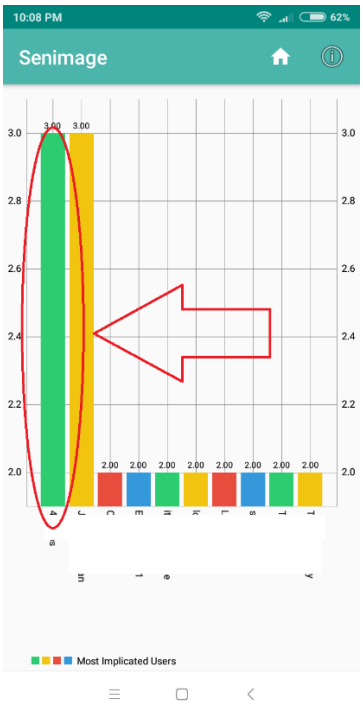
¿Cómo analizar un usuario?

Para llevar a cabo un análisis de usuario, primero deberá acceder la pantalla de información del usuario. Una de las formas de conseguir dicha tarea es a través de la pantalla de los usuarios más implicados.

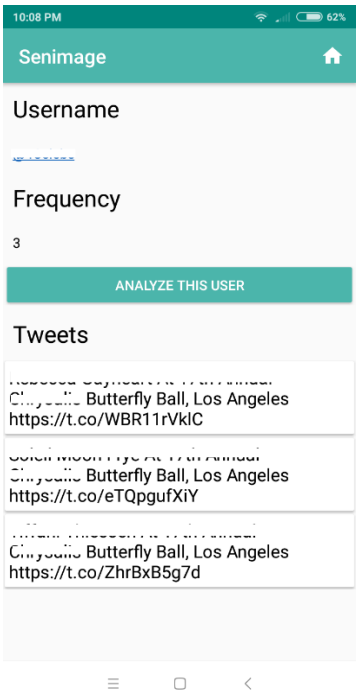
1. Para ello, seleccione el resumen de los usuarios más implicados en el "Dashboard" de la aplicación.



2. En el gráfico seleccione la barra del diagrama que se corresponde con el usuario que desea analizar.



3. Pulse el botón “Analyze this user” para ejecutar el análisis.



4. Si los pasos se han realizado satisfactoriamente, se le mostrará una pantalla con el resultado del análisis del usuario.