



Universidad de Valladolid

**Escuela de Ingeniería Informática de
Valladolid**

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática
Mención Ingeniería de Software

**“Vitalboard” : Desarrollo de una
webapp-dashboard que proporcione
informes de marketing online**

Autora: Clara Morrondo Merino
Tutoras: Margarita Gonzalo Tasis
Jezabel González Díez

Agradecimientos

Quería agradecer y dedicar la realización de este proyecto a todas aquellas personas que me han estado apoyando durante estos años de carrera y sobre todo estos últimos meses, dejando que no me rindiera. También a Margarita Gonzalo por guiarme durante el trabajo.

También a mi madre que siempre me ha apoyado y animado y ha estado ahí para todo pero sobre todo, este proyecto, la carrera y todo lo que viene a partir de ahora, se lo dedico a mi padre, que siempre me animó y confió en mi y que era la persona que más ganas tenía de que llegara el día de hoy.

Esto es para ti, papá.

Resumen

En la época actual que estamos viviendo, donde se está llevando a cabo una importante evolución de Internet, el mundo del marketing digital ha evolucionado hacia lo que se conoce como Web 2.0 o Economía 2.0, donde todo se centra en los usuarios.

Por ello, en este trabajo se va a intentar crear una herramienta que nos permita controlar dichos usuario que interactúan con nuestras páginas web para así obtener informes y análisis y poder desarrollar una estrategia digital.

Abstract

In the current era we are living, where a major evolution of the Internet is taking place, the world of digital marketing has evolved into what is known as Web 2.0 or Economy 2.0, where everything is focused on users.

Therefore, in this work we will try to create a tool that allows us to control those users who interact with our web pages to obtain reports and analysis and to develop a digital strategy.

Índice

Índice de figuras	11
Índice de tablas	13
Capítulo 1 : INTRODUCCIÓN.....	17
1.1. Motivación y objetivos del proyecto.....	19
1.2. Estructura de la memoria	19
Capítulo 2 : CONTEXTO	21
2.1. Contexto teórico	23
2.2. Herramientas similares.....	23
2.3. Ventajas y desventajas	26
Capítulo 3 : DESARROLLO DEL PROYECTO.....	29
3.1. Proceso de Desarrollo.....	31
3.2. Gestión de Riesgos	32
3.3. Roles y Responsabilidades.....	38
3.4. Planificación del proyecto	39
3.5. Seguimiento del proyecto.....	39
Capítulo 4 : ANÁLISIS	45
4.1. Actores.....	47
4.2. Requisitos funcionales.....	47
4.3. Casos de uso.....	50
4.4. Reglas de negocio / Restricciones	61
4.5. Requisitos no funcionales.....	61
4.6. Requisitos de información	63
4.7. Diagramas de secuencia	65
4.8. Diagramas de actividad	68
4.9. Modelo del dominio	72
Capítulo 5 : DISEÑO.....	75
5.1. Arquitectura lógica	77
5.2. Arquitectura física	78
5.3. Diagramas de diseño	79
5.4. Diseño de la Base de Datos.....	81
5.5. Diccionario de datos.....	82
Capítulo 6 : IMPLEMENTACIÓN.....	85

6.1.	Creación de la estructura Django	87
6.2.	Registro y login de usuarios	88
6.3.	Conexión con las APIs	89
6.4.	Filtrado de datos y maquetación	89
Capítulo 7 : TECNOLOGÍA UTILIZADA		91
7.1.	Django.....	93
7.2.	SASS	95
7.3.	GULP	95
7.4.	JavaScript	96
7.5.	API de Google Analytics	96
7.6.	API de Twitter	97
7.7.	API de Facebook	98
7.8.	API de Instagram	99
7.9.	Char.js	99
Capítulo 8 : PRUEBAS		101
8.1.	Pruebas de caja blanca	103
8.2.	Pruebas de caja negra	103
Capítulo 9 : MANUAL DE USUARIO		111
Capítulo 10 : MANUAL DE INSTALACIÓN		121
Capítulo 11 : MANUAL DE ADMINISTRADOR		127
Capítulo 12 : CONCLUSIONES		131
12.1.	Líneas futuras	133
BIBLIOGRAFÍA.....		135
APÉNDICE A		139
	Contenido del CD.....	141

Índice de figuras

VISTA DEL TABLERO DE DUCKSBOARD.....	24
VISTA DEL TABLERO DE SPROUTSOCIAL.....	25
VISTA DEL TABLERO DE HOOTSUITE.....	26
ETAPAS DEL PROCESO RACIONAL UNIFICADO (UP).....	32
MATRIZ DE RIESGOS BASADOS EN LA PROBABILIDAD DE OCURRIR E IMPACTO.....	33
DIAGRAMA DE GANTT DE PLANIFICACIÓN DEL PROYECTO.....	39
DIAGRAMA DE GANTT DE LA PRIMERA FASE DEL PROYECTO.....	40
DIAGRAMA DE GANTT DE LA SEGUNDA FASE DEL PROYECTO.....	40
DIAGRAMA DE GANTT DE LA TERCERA FASE DEL PROYECTO.....	41
DIAGRAMA DE GANTT DE LA CUARTA FASE DEL PROYECTO.....	41
DIAGRAMA DE GANTT DE LA QUINTA FASE DEL PROYECTO.....	42
DIAGRAMA DE GANTT DE LA SEXTA FASE DEL PROYECTO.....	42
DIAGRAMA DE GANTT DE LA SÉPTIMA FASE DEL PROYECTO.....	43
DIAGRAMA DE GANTT DE LA OCTAVA FASE DEL PROYECTO.....	43
DIAGRAMA DE GANTT DE LA NOVENA FASE DEL PROYECTO.....	44
DIAGRAMA DE CASOS DE USO.....	50
DIAGRAMA DE SECUENCIA REGISTRARSE.....	65
DIAGRAMA DE SECUENCIA IDENTIFICACIÓN.....	66
DIAGRAMA DE SECUENCIA GESTIÓN DE USUARIOS.....	67
DIAGRAMA DE SECUENCIA AÑADIR PERFIL GA.....	67
DIAGRAMA DE SECUENCIA OBTENER INFORMES GA.....	68
DIAGRAMA DE ACTIVIDAD REGISTRO.....	69
DIAGRAMA DE ACTIVIDAD IDENTIFICARSE.....	70
DIAGRAMA DE ACTIVIDAD AÑADIR PERFIL GA.....	71
DIAGRAMA DE ACTIVIDAD OBTENER INFORMES GA.....	72
DIAGRAMA DEL MODELO DE DOMINIO.....	73
ESTRUCTURA DJANGO MTV.....	77
ESTRUCTURA DJANGO MTV.....	78
DIAGRAMA ESTRUCTURA DE DISEÑO.....	79
DIAGRAMA DE LA BASE DE DATOS.....	81
SETTINGS.PY DE DJANGO.....	88
URLS.PY DE DJANGO.....	88
VISTA DE LA APLICACIÓN EN ESCRITORIO Y MÓVIL.....	90
LOGO DEL FRAMEWORK DJANGO.....	93
ESTRUCTURA BÁSICA DE UN PROYECTO DJANGO.....	94
SINTAXIS DE SASS COMO SCSS.....	95
INICIALIZACIÓN DE LA BIBLIOTECA DE LA API DE INSERCIÓN DE GOOGLE ANALYTICS.....	97
AUTORIZACIÓN DEL USUARIO PARA EL USO DE LA API.....	97
INICIALIZACIÓN DE HELLO.JS.....	97
RESPONSE OBTENIDA DE LLAMADA A API DE TWITTER.....	98
PANTALLA INICIAL HERRAMIENTA INICIO SESIÓN.....	113
PANTALLA REGISTRO HERRAMIENTA.....	113
PANTALLA AUTORIZACIÓN DE PERMISOS.....	114
PANTALLA SELECCIÓN CUENTA GOOGLE.....	115
PANTALLA INFORMES GOOGLE ANALYTICS.....	116

PANTALLA INFORMES TWITER	117
PANTALLA INFORMES FACEBOOK	118
PANTALLA INFORMES INSTAGRAM	119
PANTALLA DESCARGA DE INFORMES.....	119
VISTA PREVIA INFORME EXPORTADO	120
VISTA CREDENCIALES GOOGLE APIS	124
VISTA CREDENCIALES FACEBOOK	124
VISTA CREDENCIALES TWITTER	125
VISTA CREDENCIALES INSTAGRAM	126
LOGIN DJANGO ADMINISTRADOR.....	129
LISTADO DE USUARIOS DESDE PERFIL ADMIN	129

Índice de tablas

RIESGO POR FALTA DE PERSONAL.....	33
RIESGO POR NO DISPONIBILIDAD DE EQUIPO INFORMÁTICO	34
RIESGO POR FALTA DE CONOCIMIENTO PARA EL DESARROLLO	34
RIESGO POR FALTA DE REVISIONES PERIÓDICAS POR EL EQUIPO	35
RIESGO POR CALIDAD BAJA EN EL SOFTWARE DESARROLLADO.....	35
RIESGO POR CAMBIOS CONTINUOS EN LOS REQUISITOS.....	36
RIESGO POR DESARROLLO DE FUNCIONES SOFTWARE INCORRECTAS.....	36
RIESGO POR INSUFICIENTE EXPERIENCIA PARA LA GESTIÓN DE PROYECTOS	37
RIESGO POR NO CUMPLIMIENTO EN LAS ENTREGAS	37
ROLES Y RESPONSABILIDADES DE JEZABEL GONZÁLEZ	38
ROLES Y RESPONSABILIDADES DE MARGARITA GONZALO	38
ROLES Y RESPONSABILIDADES DE CLARA MORRONDO	38
ACTOR ACT - 001 USUARIO FINAL	47
ACTOR ACT - 002 ADMIN.....	47
RF - 001 ROLES DE USUARIOS.....	47
RF - 002 REGISTRO.....	48
RF - 003 REGISTRO.....	48
RF - 004 GESTIONAR USUARIOS	48
RF - 005 AÑADIR PERFIL DE GOOGLE ANALYTICS	48
RF - 006 AÑADIR PERFIL DE FACEBOOK	48
RF - 007 AÑADIR PERFIL DE TWITTER.....	48
RF - 008 AÑADIR PERFIL DE INSTAGRAM	49
RF - 009 OBTENER ANALÍTICAS DE GOOGLE ANALYTICS	49
RF - 010 OBTENER ANALÍTICAS DE FACEBOOK	49
RF - 011 OBTENER ANALÍTICAS DE TWITTER	49
RF - 012 OBTENER ANALÍTICAS DE INSTAGRAM	49
RF - 013 ELEGIR WIDGETS QUE VISUALIZAR	49
RF - 014 VISUALIZAR PERFIL DE USUARIO	50
RF - 015 CERRAR SESIÓN	50
FRQ - 001 REGISTRARSE	51
FRQ - 002 IDENTIFICACIÓN	51
FRQ - 003 GESTIÓN DE USUARIOS.....	52
FRQ - 004 AÑADIR PERFIL DE GOOGLE ANALYTICS.....	53
FRQ - 005 AÑADIR PERFIL DE FACEBOOK.....	54
FRQ - 006 AÑADIR PERFIL DE TWITTER.....	55
FRQ - 007 AÑADIR PERFIL DE INSTAGRAM.....	56
FRQ - 008 OBTENER ANALÍTICAS DE GOOGLE ANALYTICS	56
FRQ - 009 OBTENER ANALÍTICAS DE FACEBOOK	57
FRQ - 010 OBTENER ANALÍTICAS DE TWITTER	57
FRQ - 011 OBTENER ANALÍTICAS DE INSTAGRAM	58
FRQ - 012 ELEGIR WIDGETS QUE VISUALIZAR	59
FRQ - 013 VISUALIZAR PERFIL DE USUARIO	59
FRQ - 014 CERRAR SESIÓN	60
FRQ - 015 IMPRIMIR/GUARDAR INFORMES	60
CRQ - 001 ACCESO AL SISTEMA	61

CRQ - 002 GESTIÓN DE USUARIOS	61
CRQ - 003 VISUALIZACIÓN DE PERFIL	61
CRQ - 004 VISUALIZACIÓN DE ANALÍTICAS	61
RNF - 001 ACCESIBILIDAD	62
RNF - 002 USO DE BASE DE DATOS	62
RNF - 003 EJECUCIÓN DE PYTHON/DJANGO	62
RNF - 004 COMPATIBILIDAD	62
RNF - 005 IDENTIFICACIÓN DE USUARIOS	62
RNF - 006 FIABILIDAD	62
RNF - 007 PERMISOS	63
RNF - 008 UX/USABILIDAD	63
RNF - 009 DISPONIBILIDAD	63
RNF - 010 NECESIDAD DE RED DE INTERNET	63
IRQ - 001 USUARIOS	63
IRQ - 002 PERFIL	64
IRQ - 003 PERFIL DE GOOGLE ANALYTICS	64
IRQ - 004 PERFIL DE FACEBOOK	64
IRQ - 005 PERFIL DE TWITTER	64
IRQ - 006 PERFIL DE INSTAGRAM	64
IRQ - 007 ROLES	64
DICCIONARIO DE DATOS AUTH_USER	82
DICCIONARIO DE DATOS AUTH_USER_GROUPS	82
DICCIONARIO DE DATOS AUTH_GROUP	82
DICCIONARIO DE DATOS AUTH_GROUP_PERMISSIONS	83
DICCIONARIO DE DATOS DJANGO_CONTENT_TYPE	83
DICCIONARIO DE DATOS DE AUTH_USER_USER_PERMISSIONS	83
DICCIONARIO DE DATOS DE AUTH_PERMISSION	83
PRUEBA CU REGISTRARSE	103
PRUEBA CU REGISTRARSE CORREGIDO	104
PRUEBA CU INICIAR SESIÓN	104
PRUEBA CU BORRAR USUARIO	104
PRUEBA CU EDITAR USUARIO	105
PRUEBA CU AÑADIR PERFIL GA	105
PRUEBA CU AÑADIR PERFIL FACEBOOK	105
PRUEBA CU AÑADIR PERFIL TWITTER	106
PRUEBA CU AÑADIR PERFIL INSTAGRAM	106
PRUEBA CU AÑADIR PERFIL INSTAGRAM CORREGIDO	106
PRUEBA CU OBTENER INFORME GA	107
PRUEBA CU OBTENER INFORME FACEBOOK	107
PRUEBA CU OBTENER INFORME TWITTER	107
PRUEBA CU OBTENER INFORME INSTAGRAM	108
PRUEBA CU ELEGIR WIDGETS QUE VISUALIZAR	108
PRUEBA CU CAMBIAR CUENTA GA	108
PRUEBA CU CERRAR SESIÓN RRSS	109
PRUEBA CU CERRAR SESIÓN RRSS SOLUCIONADO	109
PRUEBA CU CERRAR SESIÓN SISTEMA	110

Capítulo 1 : INTRODUCCIÓN

Este Trabajo de Fin de Grado, titulado “*Vitalboard: Desarrollo de una webapp-dashboard que proporcione informes de marketing online*” ha sido desarrollado por la alumna Clara Morrondo Merino y tutelado por Margarita Gonzalo Tasis, como tutora de la Universidad de Valladolid y por Jezabel González Díez, como tutora de la empresa Vital Innova.

Este capítulo servirá para presentar dicho trabajo y ver cuál ha sido la motivación para su desarrollo, los objetivos y la estructura de la memoria.

1.1. Motivación y objetivos del proyecto

La motivación que ha llevado a realizar dicho trabajo se centra en la creación de una herramienta potente capaz de monitorizar los datos proporcionados por diferentes herramientas de marketing para el uso y evaluación del departamento de marketing de la empresa Vital Innova.

El problema surge cuando el equipo de marketing realiza el trabajo de análisis y preparación de los informes de distintas aplicaciones ya que éstos deben consultar cada una de las páginas webs de dichas aplicaciones para visualizar los datos y realizar sus mediciones.

La aplicación desarrollada permite visualizar y obtener todos los informes en un mismo sitio web facilitando y ahorrando tiempo de trabajo. Además, da la posibilidad a los propios clientes de obtener sus analíticas sin tener que depender de terceras personas.

El objetivo de dicho proyecto es buscar una solución al problema que permita estudiar, analizar, diseñar y desarrollar una herramienta que satisfaga dichos problemas.

Se partirá de un conjunto de tecnologías, analizando cada una de ellas y eligiendo la que ofrezca mejor funcionalidad, estabilidad, resultados... Por último, la solución final en forma de herramienta y cuál ha sido el proceso para llegar a ella.

1.2. Estructura de la memoria

En los capítulos que se van a presentar ahora, se va a describir la realización del proyecto. Se comenzará explicando el contexto de la aplicación, el para qué y una pequeña comparativa con aplicaciones que ofrecen una funcionalidad parecida.

A continuación, se presentará la planificación llevada a cabo para la realización de las diferentes tareas que constituyen el trabajo. Después, se ahondará en el análisis, diseño e implementación del sistema pasando por las principales tecnologías utilizadas y la realización de pruebas que se han llevado a cabo.

Por último, los manuales de usuario, tanto de desarrollo como de administración, las conclusiones y líneas futuras y la bibliografía consultada para la realización del Trabajo de Fin de Grado.

Capítulo 2 : CONTEXTO

2.1. Contexto teórico

En el mundo en el que vivimos, donde Internet lo controla todo, es raro la empresa o profesional que no cuente con una página web para bien vender directamente sus productos, informar sobre el trabajo que realiza etc. Tan importante como realizar una buena página web es el medir los resultados que se están obteniendo a partir de ésta y así poder conseguir éxito en la estrategia de marketing online.

Los profesionales de marketing deben conocer la captación de tráfico, la optimización de la navegación, las técnicas de fidelización... y combinarlo con el análisis de los resultados obtenidos. A esto se le conoce como *marketing analytics*, una combinación del marketing digital y el análisis digital.

El *marketing analytics* analiza toda la actividad de la empresa y sus estrategias, basando dicho análisis en las medidas que toma.

El uso correcto del *marketing analytics* nos ofrece los siguientes beneficios:

- Creación de la estrategia de la empresa, conocer los puntos fuertes y débiles, definirla y ponerla en marcha
- Definir y analizar las métricas más importantes para la toma de decisiones
- Realizar un análisis predictivo para saber qué puede pasar en el futuro en base a los datos actuales
- Analizar el CRM Analytics, el cual toma en consideración la relación que tiene el cliente con la empresa para la toma de decisiones
- Conseguir leads y convertirlos en clientes que se mantengan a lo largo del tiempo

2.2. Herramientas similares

Como hemos visto en el punto anterior, el análisis de métricas para el desarrollo de marketing digital está en pleno auge. Por ello, existen en el mercado un conjunto de herramientas que facilitan dicho trabajo ofreciendo datos reales y significativos para los profesionales.

Cada una de las herramientas tienen una base común y es la obtención de datos de diferentes redes sociales aunque siempre con alguna que otra diferencia, tal y como veremos a continuación.

Si investigamos qué herramientas similares podemos encontrar en el mercado, podríamos obtener los siguientes resultados:

Ducksboard:

Creado por Diego Marino, Jan Urbaski y Aitor Guevara en el 2011, Ducksboard era un SaaS que permitía visualizar métricas de marketing online como número de visitas, impresiones de usuarios, tasas de clicks... de servicios externos tales como Facebook, Twitter, Google Analytics a tiempo real. Además, contaba con una API que permitía integrar dicho producto en otros sistemas.



Ilustración 1: Vista del tablero de Ducksboard

La lista de controles era configurable por el usuario y los datos se presentaban en un formato visual, facilitando la lectura al usuario.

En 2013, la herramienta estaba a punto de desaparecer debido a que la gente probaba el producto pero no pagaba por ello, los inversores, New Relic compraron Ducksboard.

Sprout Social:

“Es una herramienta de gestión de redes sociales, promoción de la marca y análisis para empresas.”

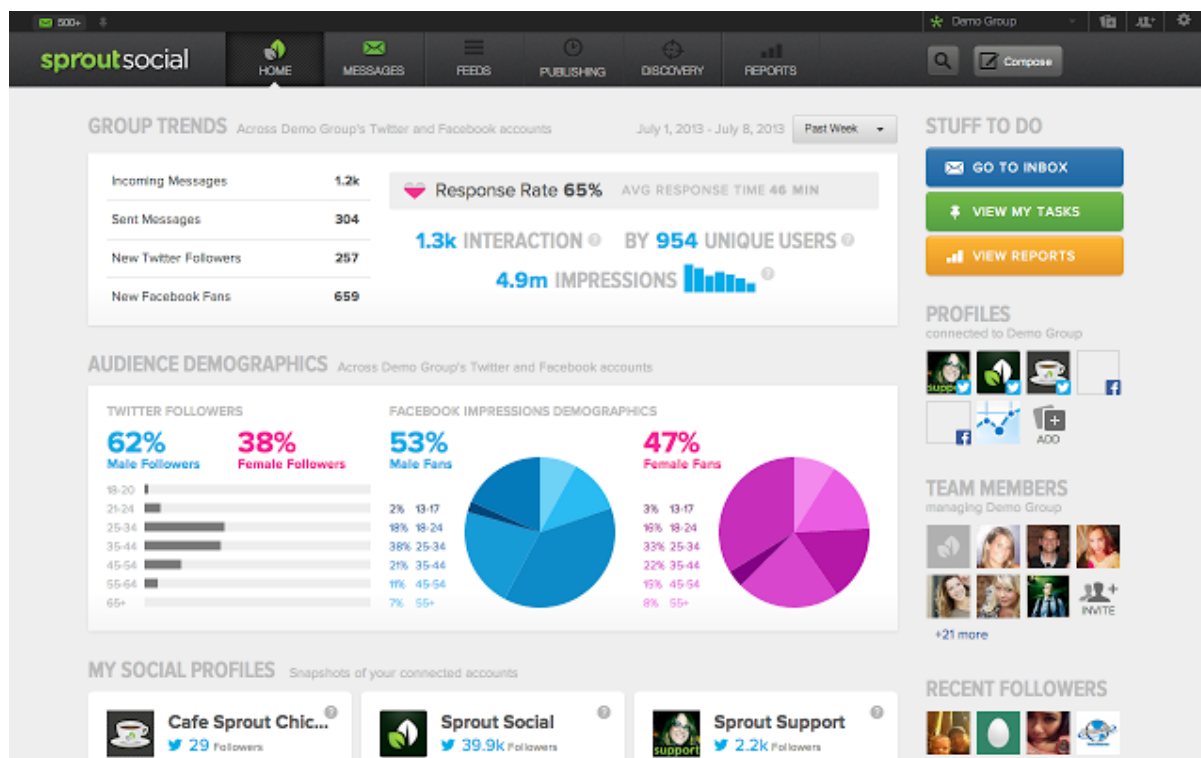


Ilustración 2: Vista del tablero de SproutSocial

Nace a finales del 2009 e integra un conjunto de redes sociales tales como Facebook, Twitter, Instagram, LinkedIn... Además permite la publicación de post con previa programación y ofrece la posibilidad de utilizar la plataforma como CRM de redes sociales.

Es una aplicación de pago, con diferentes tarifas dependiendo de las características necesarias para cada tipo de empresa/usuario.

HootSuite:

Es una aplicación web/móvil para la gestión de redes sociales por parte de empresas y/o personas de la empresa que quieran tener un control de los datos de sus sitios web.

La principal característica de HootSuite es que permite la gestión colaborativa, es decir, distintos miembros de un equipo pueden visualizar los informes desde la misma cuenta, a diferencia de, por ejemplo, Sprout Social que sólo permite un usuario conectado por sesión.

Permite la conexión con 35 redes entre las que destacan Facebook, Twitter, Instagram, LinkedIn, Youtube etc.

Es una herramienta que cuenta con diferentes tipos de planes, uno de ellos gratuito el cual te permite iniciar sesión con un usuario y la conexión a 10 redes sociales. El resto de planes son de pago, desde uno a 99€/mes hasta el Enterprise el cual te ofrece soluciones a medida.

El principal punto débil es que intenta unificar las publicaciones de contenidos en todas las redes sociales.



Ilustración 3: Vista del tablero de HootSuite

2.3. Ventajas y desventajas

Una vez vistas cuales son las ventajas y desventajas de las aplicaciones, se ha llegado a las siguientes conclusiones sobre qué aspectos debe tener la herramienta desarrollada y cuáles no:

- **Conexión simultánea de varios usuarios:** Sprout Social no permite que varios usuarios estén conectados a la vez en la misma aplicación. Además, si ya hay alguien conectado en la plataforma, le expulsa sin previo aviso. HootSuite permite la conexión de varios usuarios simultáneamente pero con la misma cuenta y en un plan de pago. Con Vitalboard suplimos este

inconveniente, facilitando que cada usuario tenga su propia cuenta y que éste sea el que seleccione las cuentas que desea visualizar.

- **Interfaz intuitiva y fácil de usar:** A pesar de ser, quizás la herramienta de este tipo más utilizada, HootSuite presenta muchos problemas de interfaz y además es poco intuitiva lo cual es una constante queja por parte de los usuarios.

- **Publicaciones desde la aplicación a diferentes redes sociales:** Las aplicaciones presentadas presentan la posibilidad de que el usuario, además de poder visualizar datos, pueda realizar publicaciones sin salirse de la herramienta

Capítulo 3 : DESARROLLO DEL PROYECTO

3.1. Proceso de Desarrollo

El proceso de desarrollo que se ha llevado a cabo para realizar el proyecto ha sido el Proceso Racional Unificado (UP).

Las principales características del Proceso Racional Unificado son las siguientes:

- Sigue una estrategia del **ciclo de vida iterativa e incremental**. Divide el proyecto en 4 fases, las cuales veremos a continuación, añadiendo en cada una de ellas una nueva funcionalidad.
- Se adapta al contexto y características del sistema mediante la creación de objetos, constituidos por datos y funciones. RUP está basado en los **casos de uso**, los cuales definen el qué, quién, cómo y cuándo de cada acción.
- Está centrado en la **arquitectura**, la cual surge a partir de los problemas de la empresa y se reflejan en los casos de uso. La forma la da la arquitectura y la función los casos de uso.
- Este tipo de proceso tiene **tantos artefactos como roles**.

Las fases principales de UP se dividen en cuatro:

1. Inicio. En esta primera fase se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos. Se obtiene una visión general del proyecto. Obtenemos un modelado del sistema, donde los integrantes del equipo se familiarizan con el proyecto y entienden el problema y la estructura. Además se obtienen los requisitos entre clientes y *skateholders*, se define una primera interfaz y se realiza una estimación de los costes.

2. Elaboración. Se realizar el plan de proyecto, se completan los casos de uso realizados en la fase anterior y se mitigan los riesgos. Se especifican los requerimientos y se describe cómo se va a realizar la implementación. Lo ideal es realizar una arquitectura óptima en esta fase.

3. Construcción. Como su propio nombre indica, en esta fase se desarrolla el producto obteniendo una versión que aunque no sea la definitiva, sea viable y usable por los usuarios. Se realizan las pruebas más significativas y se desarrollan los manuales de usuario.

4. Transición. En esta última fase se despliega el producto y se ve cómo los usuarios interactúan con ella, lo cual suele generar cambios en la aplicación. Esta fase incluye las siguientes acciones: manufactura, envío, entrenamiento, soporte y mantenimiento del producto.

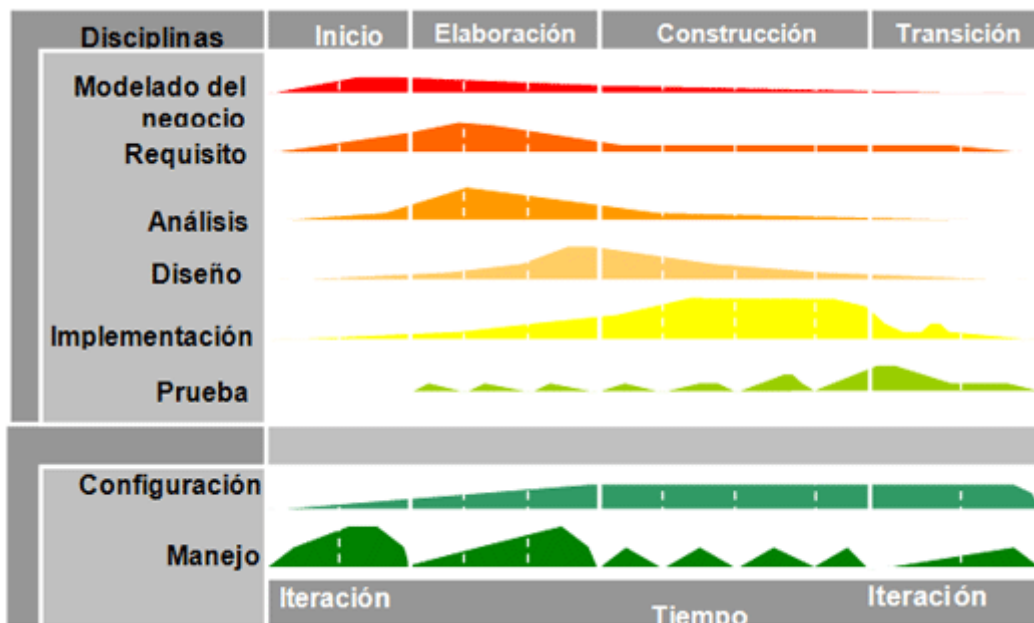


Ilustración 4: Etapas del Proceso Racional Unificado (UP)

3.2. Gestión de Riesgos

En el ámbito del software, un riesgo es una medida de la probabilidad y gravedad de sufrir efectos adversos inherentes al desarrollo de software que no cumpla con sus requerimientos funcionales y/o no funcionales.

Los riesgos se pueden clasificar en tres tipos:

- Riesgos de **Proyecto**: son aquellos que ponen en riesgo el plan y cuya solución supone un incremento del coste total del proyecto. Dentro de este tipo de riesgos están los relacionados con la planificación, el personal, los recursos, requisitos...
- Riesgos de **Producto**: afectan a la calidad del producto resultante e incrementan la complejidad del proyecto. Son los relacionados con la falta de experiencia, desconocimiento de las tecnologías, verificación de las pruebas...
- Riesgos de **Proceso**: ponen en peligro la realización del proyecto. Incluyen riesgos de estrategia, mala documentación...

Los riesgos se clasifican además en la probabilidad que tienen de ocurrir. Estos pueden tener una probabilidad "Muy Alta", "Alta", "Media" o "Baja". Además, el impacto que tiene el riesgo en el proyecto puede ser "Crítico", "Significativo", "Tolerante" e "Insignificante".

Probabilidad	Consecuencias				
	Insignificante 1	Menor 2	Moderada 3	Mayor 4	Catastrófica 5
Raro 1	Bajo	Bajo	Moderado	Alto	Alto
Improbable 2	Bajo	Bajo	Moderado	Alto	Extremo
Posible 3	Bajo	Moderado	Alto	Extremo	Extremo
Probable 4	Moderado	Alto	Alto	Extremo	Extremo
Casi seguro 5	Alto	Alto	Extremo	Extremo	Extremo

Ilustración 5: Matriz de riesgos basados en la probabilidad de ocurrir e impacto

Los riesgos que se han identificado en el proyecto realizado se distinguen en tres tipos de categoría: de proyecto, de proceso y de producto. Se presentan a continuación junto a sus planes de acción:

RIESGO-01	FALLO DE PERSONAL	
Fase: Todas	Probabilidad: Baja	Categoría: Proyecto
Descripción: Un miembro del equipo no se encuentra disponible durante un período de tiempo para realizar una tarea que le ha sido asignada.		
Consecuencias: Una tarea queda sin poder ser realizada por falta de personal.		
Contexto: Situación que se puede dar en cualquier fase del proyecto por motivos inherentes al mismo.		
Análisis: Puede provocar un alargamiento de la tarea y por lo tanto, el incumplimiento del plazo de entrega.		
Plan de acción: Reorganización del trabajo asignando otro trabajador a la tarea.		

Tabla 1: Riesgo por falta de personal

RIESGO-02	NO DISPONIBILIDAD DE EQUIPO INFORMÁTICO	
Fase: Todas	Probabilidad: Baja	Categoría: Proyecto
Descripción: El equipo informático de un miembro del equipo no se encuentra disponible durante un período de tiempo por lo que no puede realizar una tarea que le ha sido asignada de forma adecuada.		
Consecuencias: Una tarea queda sin poder ser realizada por falta de personal.		
Contexto: Situación que se puede dar en cualquier fase del proyecto por motivos inherentes al mismo.		
Análisis: Puede provocar un alargamiento de la tarea y por lo tanto, el incumplimiento del plazo de entrega.		
Plan de acción: Arreglo e intento de disposición de equipo informático efectivo y reorganización del trabajo asignando otro trabajador a la tarea.		

Tabla 2: Riesgo por no disponibilidad de equipo informático

RIESGO - 03	FALTA DE CONOCIMIENTO PARA EL DESARROLLO	
Fase: Elaboración y construcción	Probabilidad: Media	Categoría: Proyecto
Descripción: El equipo de desarrollo no conoce las herramientas/técnicas/software utilizado para el desarrollo de la aplicación.		
Consecuencias: Prolongación en el tiempo de desarrollo e incluso creación de código de mala calidad.		
Contexto: Situación que puede darse al comienzo de la etapa de desarrollo y pruebas aunque también puede darse una vez empezada la tarea si se introducen nuevos recursos desconocidos.		
Análisis: Tiene un impacto fuerte ya que puede provocar desde el incumplimiento del plazo de entrega hasta el fracaso total del proyecto.		
Plan de acción: El equipo de desarrollo debe conocer de antemano o aprender si lo desconoce todos los recursos que se utilizarán para el desarrollo, ya sea individualmente o con el apoyo del resto de compañeros del equipo.		

Tabla 3: Riesgo por falta de conocimiento para el desarrollo

RIESGO - 04	FALTA DE REVISIONES EFECTIVAS DE MIEMBROS DEL EQUIPO	
Fase: Elaboración y construcción	Probabilidad: Media	Categoría: Proceso
Descripción: El equipo no realiza la revisiones periódicas acordadas a lo largo de todo el proyecto.		
Consecuencias: Mal funcionamiento del equipo y realización de actividades de baja calidad.		
Contexto: Situación que puede darse en la fase de desarrollo debido a una mala organización del equipo.		
Análisis: Puede provocar la producción de software de mala calidad.		
Plan de acción: Los miembros del equipo de desarrollo deben estar concienciados de la necesidad de revisar las tareas de los demás.		

Tabla 4: Riesgo por falta de revisiones periódicas por el equipo

RIESGO - 05	CALIDAD BAJA DE SOFTWARE	
Fase: Análisis y elaboración	Probabilidad: Media	Categoría: Producto
Descripción: Se realizan tareas de manera no efectiva que tienen como consecuencia la producción de software de baja calidad.		
Consecuencias: Producto software que no cumple con los objetivos y requisitos planteados.		
Contexto: Situación que se puede dar en las fases de análisis y desarrollo debido a una mala organización y/o trabajo del equipo.		
Análisis: Tiene un fuerte impacto ya que no se consiguen los objetivos marcados del proyecto.		
Plan de acción: Investigación de los causas del evento para que no vuelva a ocurrir y búsqueda de errores para su posible corrección.		

Tabla 5: Riesgo por calidad baja en el software desarrollado

RIESGO - 06	SECUENCIA CONTINUADA DE CAMBIOS EN LOS REQUISITOS	
Fase: Todas	Probabilidad: Media	Categoría: Proceso
Descripción: La interfaz externa cambia de forma sistemática algún o algunos requisitos del proyecto.		
Consecuencias: Afecta a las tareas relacionadas con los requisitos que se han sido modificados.		
Contexto: Puede ocurrir en cualquier fase del proyecto debido a una falta de comunicación con la interfaz externa.		
Análisis: Tiene un fuerte impacto ya que implica una importante prolongación del tiempo de proyecto y el consecuente incumplimiento del plazo de entrega.		
Plan de acción: Los jefes de las fases afectadas deben realizar una reunión con la interfaz externa y dirigir actividades necesarias para llevar a cabo las modificaciones necesarias.		

Tabla 6: Riesgo por cambios continuos en los requisitos

RIESGO - 07	DESARROLLO DE FUNCIONES SOFTWARE INCORRECTAS	
Fase: Análisis y Elaboración	Probabilidad: Media	Categoría: Producto
Descripción: Interpretación errónea de un requisito del sistema que provoca una implementación no deseada.		
Consecuencias: Realización incorrecta del trabajo.		
Contexto: Situación producida en la fase de análisis que tiene un efecto negativo en el desarrollo, debida a una mala realización de una actividad.		
Análisis: Tiene un impacto fuerte pues si la funcionalidad no es la deseada, no cumplirá las expectativas del cliente.		
Plan de acción: Reunión del equipo con la interfaz externa para afianzar las funcionalidades requeridas y la realización de tareas destinadas a las modificaciones pertinentes.		

Tabla 7: Riesgo por desarrollo de funciones software incorrectas

RIESGO - 08	EXPERIENCIA INSUFICIENTE PARA LA GESTIÓN DE PROYECTOS	
Fase: Planificación	Probabilidad: Alta	Categoría: Producto
Descripción: La inexperiencia del equipo hace que la planificación no se realice de forma adecuada.		
Consecuencias: Afecta a la gestión del proyecto en general.		
Contexto: Situación producida en la fase de planificación.		
Análisis: Tiene un fuerte impacto ya que incluso puede provocar el fracaso del proyecto.		
Plan de acción: El equipo de trabajo y en concreto, el Jefe de Proyectos, debe formarse sobre la planificación de proyectos o buscar gestores de proyectos que les asesoren. Finalmente, deben replanificar el proyecto corrigiendo errores producidos anteriormente y pudiendo utilizar el proyecto anterior como plantilla.		

Tabla 8: Riesgo por insuficiente experiencia para la gestión de proyectos

RIESGO - 09	NO CUMPLIMIENTO DE LA ENTREGA DE UN HITO	
Fase: Todas	Probabilidad: Alta	Categoría: Proceso
Descripción: No se consiguen los objetivos planteados en la fecha de entrega del hito.		
Consecuencias: Afecta al plazo de entrega del hito actual y siguientes.		
Contexto: Situación que puede darse en cualquier fase del proyecto y se debe a una mala planificación del mismo.		
Análisis: Produce el retraso de la entrega actual y el de los hitos siguientes, por lo que se produce una importante prolongación del tiempo del proyecto.		
Plan de acción: Replanificación el siguiente hito para no retrasar el proyecto completo.		

Tabla 9: Riesgo por no cumplimiento en las entregas

3.3. Roles y Responsabilidades

Miembro del equipo: Jezabel González Díez	
Rol	Project Manager
Responsabilidades	Encargado de definir las metas del proyecto, organización de las tareas y toma de decisiones para garantizar que el proyecto se desarrolle con éxito.

Tabla 10: Roles y responsabilidades de Jezabel González

Miembro del equipo: Margarita Gonzalo Tasis	
Rol	Project Manager
Responsabilidades	Encargado de interpretar las necesidades de cada tarea para garantizar que el proyecto se ejecute de forma adecuada.

Tabla 11: Roles y responsabilidades de Margarita Gonzalo

Miembro del equipo: Clara Morrondo Merino	
Rol	Analista
Responsabilidades	Recoge los requisitos del proyecto y elabora el modelo de análisis y diseño, garantizando el cumplimiento de los objetivos del proyecto.
Rol	Diseñador
Responsabilidades	Realiza el diseño general del sistema el cual engloba la arquitectura, bases de datos etc. buscando soluciones que cumplan los requisitos de una forma fiable, rápida y eficaz.
Rol	Desarrollador
Responsabilidades	Implementa el sistema siguiendo los patrones desarrollados.
Rol	Probador
Responsabilidades	Encargado de probar la funcionalidad del sistema en busca de errores para su futura corrección y mejora.

Tabla 12: Roles y responsabilidades de Clara Morrondo

3.4. Planificación del proyecto

Teniendo en cuenta el esfuerzo esperado para la realización del proyecto, se planteó el siguiente plan de fases:

- 1 fase de Inicio, con una duración de 7 días
- 2 fases de Elaboración, una para el análisis y otra para el diseño, con una duración de 7 días cada una
- 4 fases de Construcción o Implementación:
 - 1 fase para la creación y estructura del proyecto con una duración de 10 días
 - 1 fase para el registro y *login* de usuarios con una duración de 7 días
 - 1 fase para la conexión con las APIs con una duración de 15 días
 - 1 fase de filtrado de datos y maquetación con una duración de 15 días
- 1 fase de transición con una duración de 15 días

En total, obtenemos una suma de 93 días, comenzando el 14 de febrero de 2017 y esperando finalizar el 8 de junio de 2017 , teniendo días de holgura suficiente por si surgiera alguno de los inconvenientes vistos en los riesgos y/o preparación de presentación y defensa final del proyecto.

A continuación se presentan dichos datos en un diagrama de Gantt:

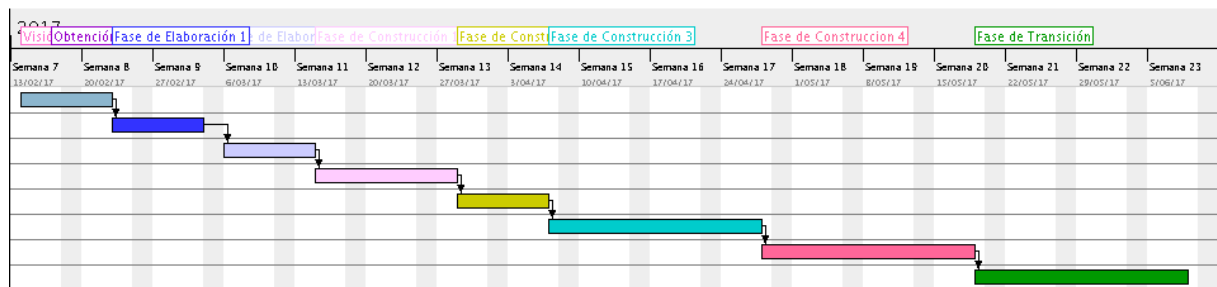


Ilustración 6: Diagrama de Gantt de planificación del proyecto

3.5. Seguimiento del proyecto

La duración de la fase de inicio tenía una duración prevista de 7 días y tuvo una duración real de 5 días. Esto fue así ya que el proyecto tenía unos objetivos muy concretos y fue fácil de

definir. El compartir los problemas que tenían los compañeros del departamento de Marketing hizo que fuera más fácil detectar los requisitos.

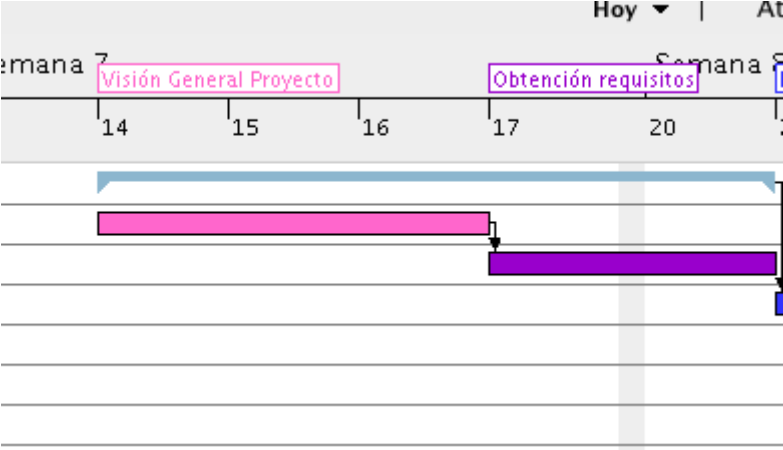


Ilustración 7: Diagrama de Gantt de la primera fase del proyecto

Las fases de análisis y diseño también sufrieron cambios con respecto a la planificación final. La fase de análisis tuvo una duración de 1 día más, es decir, 8 días y la de diseño 2 días más, es decir, 9 día en total. Esto ocurrió debido al uso de una arquitectura poco conocida y al cambio del tipo de desarrollo software, partiendo de *Scrum* al finalmente utilizado, *UP*.

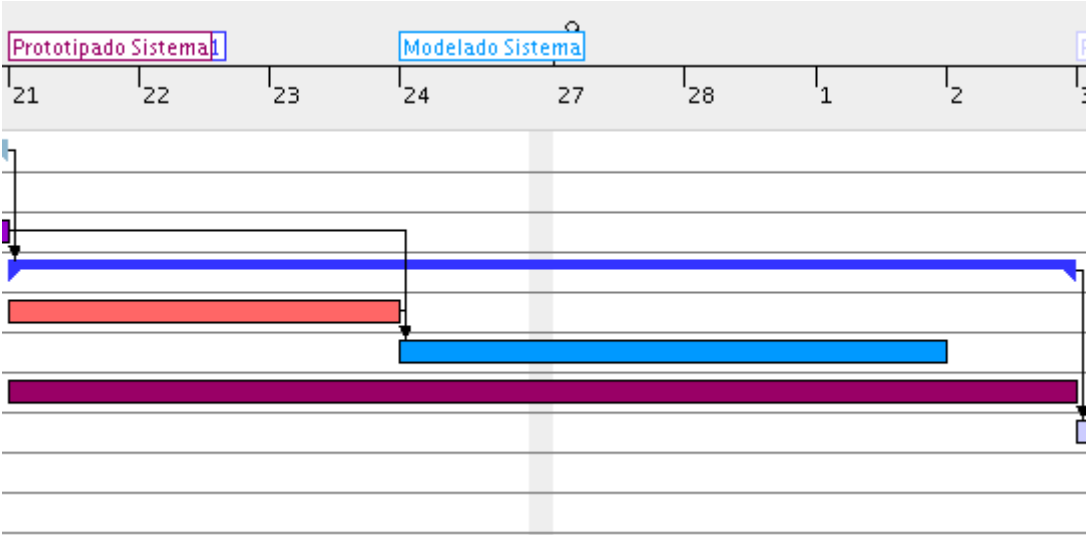


Ilustración 8: Diagrama de Gantt de la segunda fase del proyecto

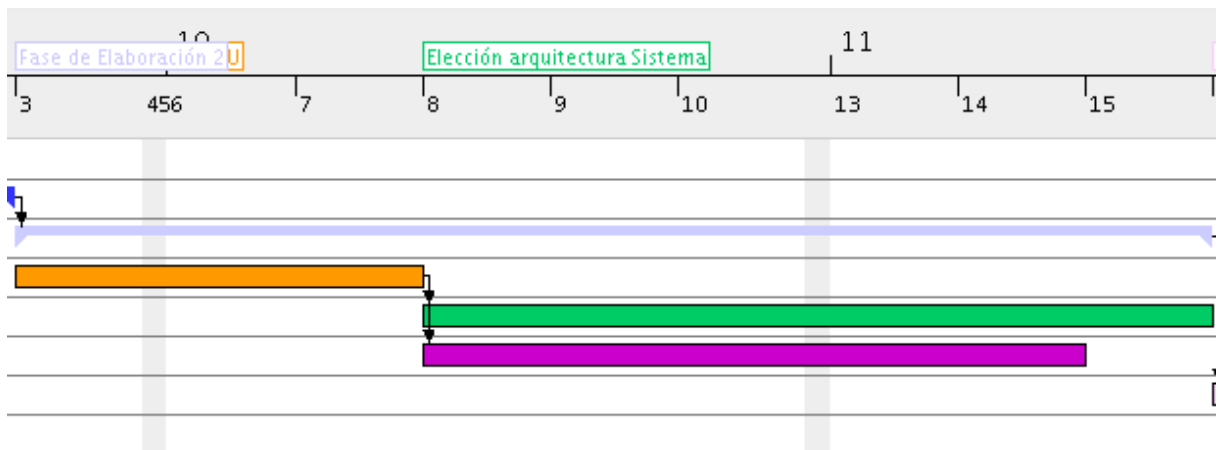


Ilustración 9: Diagrama de Gantt de la tercera fase del proyecto

Las fases de implementación sufrieron diversos cambios por diferentes factores como, por ejemplo, el desconocimiento del lenguaje de programación Django y problemas a la hora de realizar llamadas a las diferentes APIs, principalmente.

En la primera fase se desarrolló el esqueleto de la aplicación. En esta subfase se cumplió el tiempo establecido, es decir, se dedicó 10 días para su desarrollo.

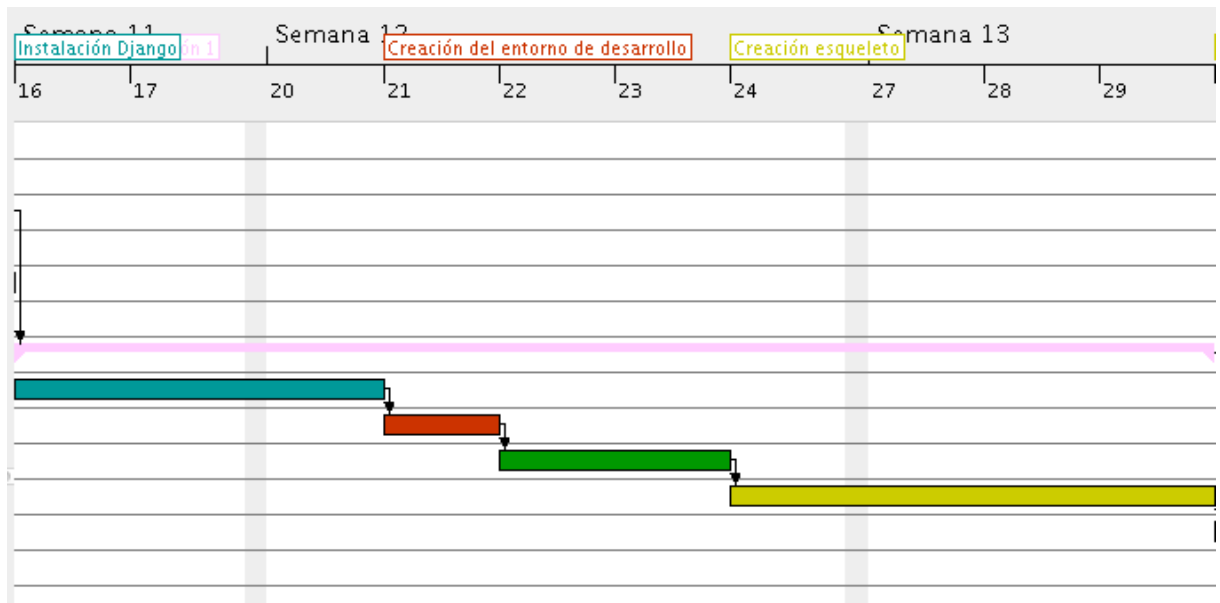


Ilustración 10: Diagrama de Gantt de la cuarta fase del proyecto

En la segunda etapa de la fase de implementación, se desarrolló el sistema de *login* y registro de la aplicación. En esta subfase se redujo la duración inicial, de 7 días a 5, debido a la facilidad que trae Django para realizar dicha funcionalidad.

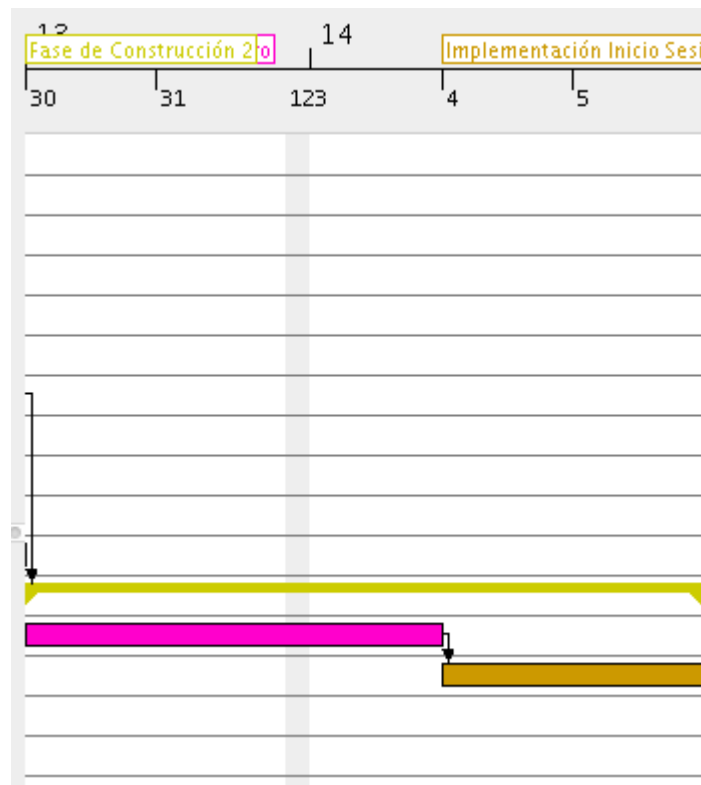


Ilustración 11: Diagrama de Gantt de la quinta fase del proyecto

La tercera etapa consistió en la conexión con las distintas APIs, la cual sufrió un aumento de los 15 días iniciales a 17 días reales.

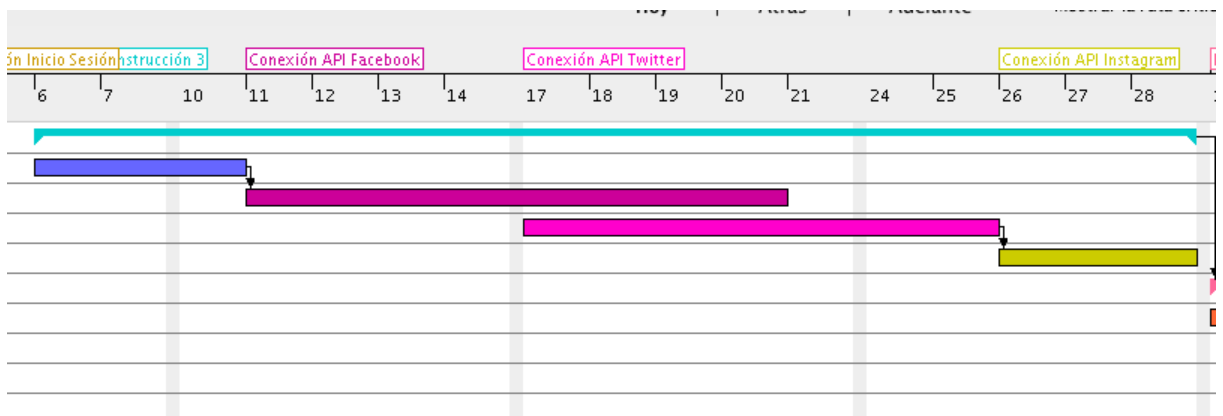


Ilustración 12: Diagrama de Gantt de la sexta fase del proyecto

La última subetapa de la fase de implementación consistió en el filtrado de los datos obtenidos por las APIs y la maquetación. La planificación estimada era de 15 días pero finalmente fue de 20 días.

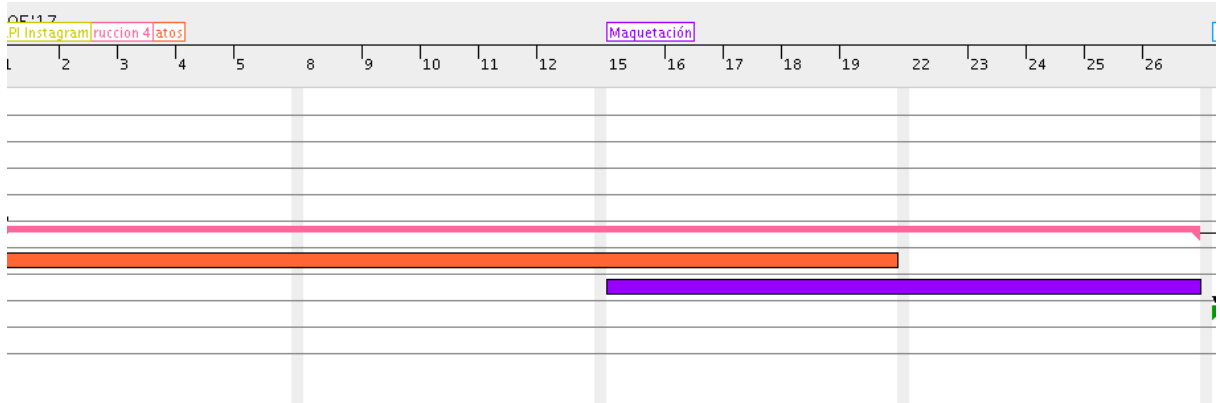


Ilustración 13: Diagrama de Gantt de la séptima fase del proyecto

Por último, en la fase de transición se cumplió lo estimado, es decir, 15 días, donde se realizaron las pruebas, se corrigieron bugs y se desarrollaron los manuales de usuario.

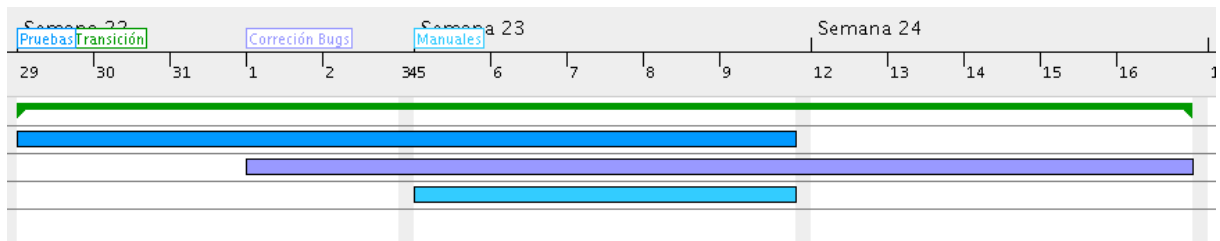


Ilustración 14: Diagrama de Gantt de la octava fase del proyecto

Finalmente, obtendremos un diagrama de Gantt de la planificación real que presenta la siguiente forma:

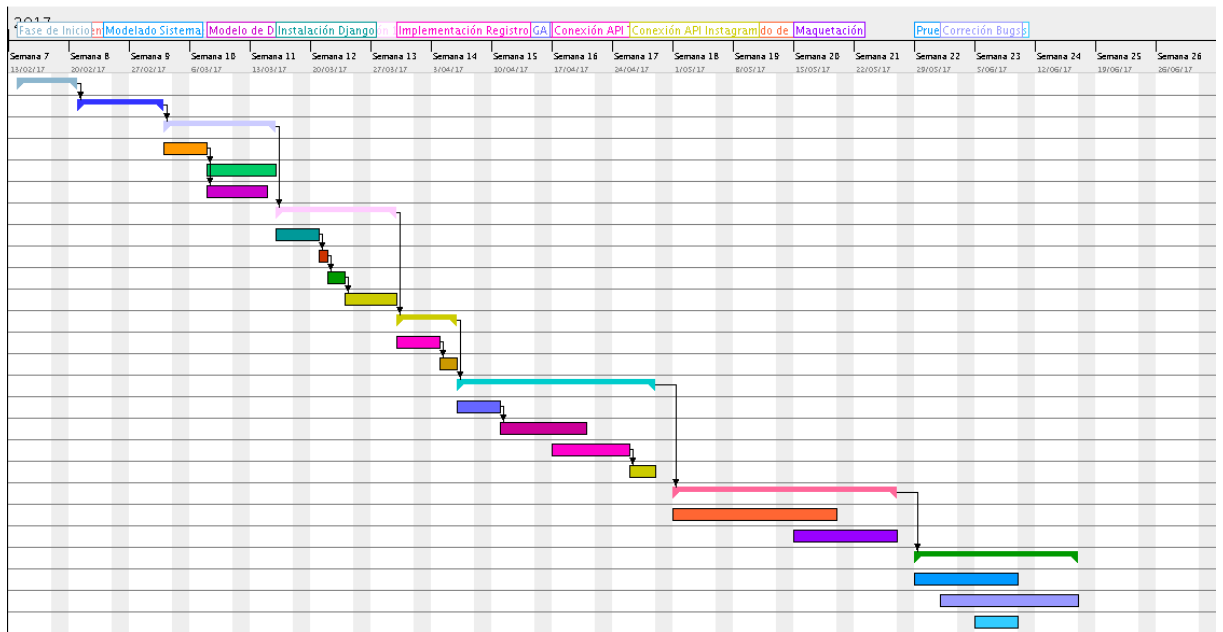


Ilustración 15: Diagrama de Gantt de la novena fase del proyecto

Capítulo 4 : ANÁLISIS

En este apartado se van a incluir todos los artefactos de análisis relacionados con la herramienta, tales como requisitos, casos de uso y las relaciones entre éstos.

4.1. Actores

Los actores representan a todos aquellos roles que interactúan con el sistema. En este caso existen dos actores diferentes, que veremos a continuación:

ACT - 001	Usuario final
Descripción	Su función es la de usuario final de la aplicación, es decir, representa al usuario del que se han obtenido los requisitos y para el que se ha realizado la herramienta.

Tabla 13: Actor ACT - 001 Usuario Final

ACT - 002	Administrador
Descripción	Tiene control total sobre todo el sitio. Puede ver los usuarios registrados en la página, los permisos que tiene cada uno de ellos, ver la última sesión y gestionarlos, es decir, puede modificar sus datos, cambiar los permisos, borrarlos...

Tabla 14: Actor ACT - 002 Admin

4.2. Requisitos funcionales

Los requisitos funcionales definen las acciones que debe poder desarrollar la herramienta para dar lugar a la funcionalidad deseada.

RF - 001	Roles de usuarios
Descripción	El sistema debe permitir que existan usuarios con dos roles distintos, administrador y cliente final

Tabla 15: RF - 001 Roles de Usuarios

RF - 002	Registro
Descripción	El sistema debe permitir que los usuarios se puedan registrar con rol cliente final

Tabla 16: RF - 002 Registro

RF - 003	Iniciar Sesión
Descripción	El sistema debe permitir que los usuarios puedan acceder una vez se hayan registrado

Tabla 17: RF - 003 Registro

RF - 004	Gestionar usuarios
Descripción	El sistema debe permitir que el administrador pueda gestionar usuarios desde su perfil

Tabla 18: RF - 004 Gestionar Usuarios

RF - 005	Añadir Perfil de Google Analytics
Descripción	El sistema debe permitir que el usuario añada su perfil de Google Analytics

Tabla 19: RF - 005 Añadir Perfil de Google Analytics

RF - 006	Añadir Perfil de Facebook
Descripción	El sistema debe permitir que el usuario añada su perfil de Facebook

Tabla 20: RF - 006 Añadir Perfil de Facebook

RF - 007	Añadir Perfil de Twitter
Descripción	El sistema debe permitir que el usuario añada su perfil de Twitter

Tabla 21: RF - 007 Añadir Perfil de Twitter

RF - 008	Añadir Perfil de Instagram
Descripción	El sistema debe permitir que el usuario añada su perfil de Instagram

Tabla 22: RF - 008 Añadir Perfil de Instagram

RF - 009	Obtener analíticas de Google Analytics
Descripción	El sistema debe permitir que el usuario obtenga analíticas de Google Analytics

Tabla 23: RF - 009 Obtener analíticas de Google Analytics

RF - 010	Obtener analíticas de Facebook
Descripción	El sistema debe permitir que el usuario obtenga analíticas de Facebook

Tabla 24: RF - 010 Obtener analíticas de Facebook

RF - 011	Obtener analíticas de Twitter
Descripción	El sistema debe permitir que el usuario obtenga analíticas de Twitter

Tabla 25: RF - 011 Obtener analíticas de Twitter

RF - 012	Obtener analíticas de Instagram
Descripción	El sistema debe permitir que el usuario obtenga analíticas de Instagram

Tabla 26: RF - 012 Obtener analíticas de Instagram

RF - 013	Elegir widgets que visualizar
Descripción	El sistema debe permitir que el usuario elija los widgets que desea visualizar de cada red social

Tabla 27: RF - 013 Elegir widgets que visualizar

RF - 014	Visualizar Perfil de Usuario
Descripción	El sistema debe permitir que el usuario visualice sus datos de usuario en el apartado "Mi Perfil"

Tabla 28: RF - 014 Visualizar Perfil de Usuario

RF - 015	Cerrar Sesión
Descripción	El sistema debe permitir que el usuario cierre sesión cuando desee para permitir que otros se conecten a la herramienta

Tabla 29: RF - 015 Cerrar Sesión

4.3. Casos de uso

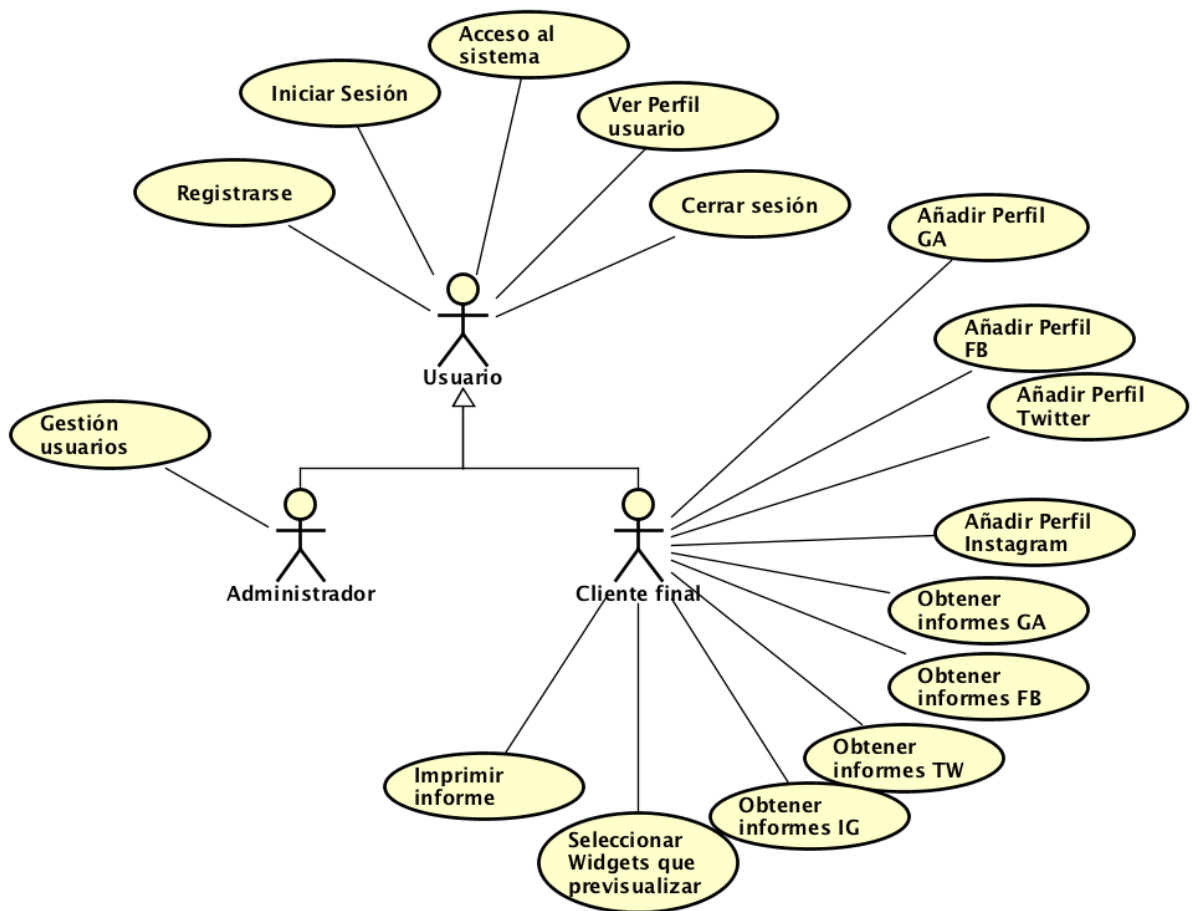


Ilustración 16: Diagrama de casos de uso

FRQ - 001	Registrarse
Versión	1.0
Dependencias	Ninguna
Precondición	
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando un usuario desee registrarse para obtener unas credenciales con las que acceder a la aplicación</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la página de identificación 2. El sistema solicita el nombre de usuario y contraseña 3. El usuario proporciona al sistema ambos datos 4. El sistema comprueba que los datos introducidos son correctos
Postcondición	El usuario accede al sistema y queda identificado en este

Tabla 30: FRQ - 001 Registrarse

FRQ - 002	Identificarse
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario debe haber sido dado de alta en el sistema
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando un usuario dado de alta en el sistema trate de identificarse</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a la página de identificación 2. El sistema solicita el nombre de usuario y contraseña 3. El usuario proporciona al sistema ambos datos 4. El sistema comprueba que los datos introducidos son correctos
Postcondición	El usuario accede al sistema y queda identificado en este
Excepciones	<ol style="list-style-type: none"> 4. El usuario ha proporcionado unos credenciales incorrectos: <ul style="list-style-type: none"> - El sistema informa al usuario y se vuelve al paso 2

Tabla 31: FRQ - 002 Identificación

FRQ - 003	Gestionar usuarios
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema y haber iniciado sesión
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario administrador desee gestionar los usuarios registrados en la aplicación</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario accede al panel de administrador 2. El sistema le muestra el panel de administración con los usuarios y grupos 3. El usuario selecciona usuarios o grupos 4. El sistema muestra el listado de usuarios o grupos dependiendo de la elección del paso 3 5. El usuario selecciona el usuario sobre el que desea realizar modificaciones/borrado
Postcondición	El usuario administrador ha realizado cambios en los usuarios registrados en la página
Excepciones	<ul style="list-style-type: none"> - 1. b) El usuario proporciona unos datos incorrectos - El sistema informa al usuario y se vuelve al paso 1 - 1. c) El usuario proporciona unos datos de un usuario que no es administrador - El sistema informa al usuario y se vuelve al paso 1

Tabla 32 *Tabla 32: FRQ - 003 Gestión de Usuarios*

FRQ - 004	Añadir Perfil Google Analytics
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema y haber iniciado sesión
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee agregar una cuenta de Google Analytics a su perfil</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a su perfil 2. El sistema muestra al usuario las redes sociales disponibles para iniciar sesión y si está o no logeado en cada una de ellas 3. El usuario solicita añadir una cuenta de Google Analytics 4. El sistema muestra el listado de cuentas de Google que el usuario tiene asociadas en dicha máquina para que seleccione una o bien añada una nueva 5. El usuario selecciona la cuenta con la que desea iniciar sesión 6. El sistema solicita los permisos necesarios para una correcta funcionalidad con dicha cuenta
Postcondición	El usuario ha añadido una cuenta de Google Analytics a las cuentas de su perfil
Excepciones	<ul style="list-style-type: none"> - 3. b) El usuario ya ha introducido una cuenta de Google Analytics - El sistema informa al usuario y el caso de uso queda sin efecto - 3. c) El usuario introduce unas credenciales no válidas de Google Analytics - El sistema informa al usuario y se vuelve al paso 2

Tabla 33: FRQ - 004 Añadir Perfil de Google Analytics

FRQ - 005	Añadir Perfil Facebook
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema y haber iniciado sesión
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee agregar una cuenta de Facebook a su perfil</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a su perfil 2. El sistema muestra al usuario las redes sociales disponibles para iniciar sesión y si está o no logeado en cada una de ellas 3. El usuario solicita añadir una cuenta de Facebook 4. El sistema solicita al usuario la aceptación de que <i>Vitalboard</i> utilice los datos de la cuenta de Facebook que el usuario tenga iniciada en dicha máquina
Postcondición	El usuario ha añadido una cuenta de Facebook a las cuentas de su perfil
Excepciones	<ul style="list-style-type: none"> - 3. b) El usuario ya ha introducido una cuenta de Facebook - El sistema informa al usuario y el caso de uso queda sin efecto - 3. c) El usuario introduce unas credenciales no válidas de Facebook - El sistema informa al usuario y se vuelve al paso 2

Tabla 34: FRQ - 005 Añadir Perfil de Facebook

FRQ - 006	Añadir Perfil Twitter
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema y haber iniciado sesión
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee agregar una cuenta de Twitter a su perfil</i>

Secuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a su perfil 2. El sistema muestra al usuario las redes sociales disponibles para iniciar sesión y si está o no logeado en cada una de ellas 3. El usuario solicita añadir una cuenta de Twitter 4. El sistema solicita al usuario la aceptación de que <i>Vitalboard</i> utilice los datos de la cuenta de Twitter que el usuario tenga iniciada en dicha máquina
Postcondición	El usuario ha añadido una cuenta de Twitter a las cuentas de su perfil
Excepciones	<ul style="list-style-type: none"> - 3. b) El usuario ya ha introducido una cuenta de Twitter - El sistema informa al usuario y el caso de uso queda sin efecto - 3. c) El usuario introduce unas credenciales no válidas de Twitter - El sistema informa al usuario y se vuelve al paso 2

Tabla 35: FRQ - 006 Añadir Perfil de Twitter

FRQ - 007	Añadir Perfil Instagram
Versión	1.0
Dependencias	
Precondición	El usuario administrador debe haber sido dado de alta en el sistema y haber iniciado sesión
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee agregar una cuenta de Instagram a su perfil</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario accede a su perfil 2. El sistema muestra al usuario las redes sociales disponibles para iniciar sesión y si está o no logeado en cada una de ellas 3. El usuario solicita añadir una cuenta de Instagram 4. El sistema solicita al usuario la aceptación de que <i>Vitalboard</i> utilice los datos de la cuenta de Instagram que el usuario tenga iniciada en dicha máquina
Postcondición	El usuario ha añadido una cuenta de Instagram a las cuentas de su perfil
Excepciones	<ul style="list-style-type: none"> - 3. b) El usuario ya ha introducido una cuenta de Instagram - El sistema informa al usuario y el caso de uso queda

	<p>sin efecto</p> <ul style="list-style-type: none"> - 3. c) El usuario introduce unas credenciales no válidas de Instagram - El sistema informa al usuario y se vuelve al paso 2
--	---

Tabla 36: FRQ - 007 Añadir Perfil de Instagram

FRQ - 008	Obtener analíticas de Google Analytics
Versión	1.0
Dependencias	
Precondición	El usuario administrador debe haber sido dado de alta en el sistema, haber iniciado sesión y haber añadido una cuenta de Google Analytics
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee obtener analíticas de la cuenta asociada de Google Analytics</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario solicita visualizar los datos de Google Analytics haciendo click en la pestaña asociada 2. El sistema comprueba que el usuario haya iniciado sesión con una cuenta de Google Analytics, hace las llamadas a la API correspondientes y muestra los resultados al usuario
Postcondición	El usuario ha obtenido informes sobre diferentes métricas de Google Analytics
Excepciones	<ul style="list-style-type: none"> - 2. b) El sistema no puede obtener los datos requeridos - El sistema informa al usuario y se vuelve al paso 1

Tabla 37: FRQ - 008 Obtener analíticas de Google Analytics

FRQ - 009	Obtener analíticas de Facebook
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema, haber iniciado sesión y haber añadido una cuenta de Facebook

Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee obtener analíticas de la cuenta asociada de Facebook</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario solicita visualizar los datos de Facebook haciendo click en la pestaña asociada 2. El sistema comprueba que el usuario haya iniciado sesión con una cuenta de Facebook, hace las llamadas a la API correspondientes y muestra los resultados al usuario
Postcondición	El usuario ha obtenido informes sobre diferentes métricas de Facebook
Excepciones	<ul style="list-style-type: none"> - 2. b) El sistema no puede obtener los datos requeridos - El sistema informa al usuario y se vuelve al paso 1

Tabla 38: FRQ - 009 Obtener analíticas de Facebook

FRQ - 010	Obtener analíticas de Twitter
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema, haber iniciado sesión y haber añadido una cuenta de Twitter
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee obtener analíticas de la cuenta asociada de Twitter</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario solicita visualizar los datos de Twitter haciendo click en la pestaña asociada 2. El sistema comprueba que el usuario haya iniciado sesión con una cuenta de Twitter, hace las llamadas a la API correspondientes y muestra los resultados al usuario
Postcondición	El usuario ha obtenido informes sobre diferentes métricas de Twitter
Excepciones	<ul style="list-style-type: none"> - 2. b) El sistema no puede obtener los datos requeridos - El sistema informa al usuario y se vuelve al paso 1

Tabla 39: FRQ - 010 Obtener analíticas de Twitter

FRQ - 011	Obtener analíticas de Instagram
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema, haber iniciado sesión y haber añadido una cuenta de Instagram
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee obtener analíticas de la cuenta asociada de Instagram</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario solicita visualizar los datos de Instagram haciendo click en la pestaña asociada 2. El sistema comprueba que el usuario haya iniciado sesión con una cuenta de Instagram, hace las llamadas a la API correspondientes y muestra los resultados al usuario
Postcondición	El usuario ha obtenido informes sobre diferentes métricas de Instagram
Excepciones	<ul style="list-style-type: none"> - 2. b) El sistema no puede obtener los datos requeridos - El sistema informa al usuario y se vuelve al paso 1

Tabla 40: FRQ - 011 Obtener analíticas de Instagram

FRQ - 012	Elegir widgets que visualizar
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema, haber iniciado sesión y haber añadido al menos una red social sobre la que visualizar los datos
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee seleccionar los widgets que desea visualizar de cada una de las redes sociales</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción "Ajustes de Visualización" del dashboard para elegir qué widgets desea ocultar/mostrar 2. El sistema muestra el listado de widgets disponibles para mostrar/ocultar 3. El usuario selecciona el widget que desea ocultar/mostrar
Postcondición	El usuario ha seleccionado el listado de widgets que desea visualizar de cada red social

Tabla 41: FRQ - 012 Elegir widgets que visualizar

FRQ - 013	Visualizar Perfil de Usuario
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema y haber iniciado sesión
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee visualizar sus datos de usuario de la aplicación desde su perfil</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario selecciona acceder a la vista de "Mi perfil" 2. El sistema comprueba los datos del usuario logeado y muestra sus credenciales
Postcondición	El usuario ha obtenido los datos sobre su perfil de usuario en la aplicación

Tabla 42: FRQ - 013 Visualizar Perfil de Usuario

FRQ - 014	Cerrar Sesión
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema y haber iniciado sesión
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee cerrar sesión en la herramienta</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción de Cerrar Sesión de la aplicación 2. El sistema cierra la sesión de dicho usuario y redirecciona a la página principal donde se permite un nuevo inicio de sesión y/o registro
Postcondición	El usuario ha cerrado sesión en la aplicación

Tabla 43: FRQ - 014 Cerrar sesión

FRQ - 015	Imprimir/guardar informe
Versión	1.0
Dependencias	Ninguna
Precondición	El usuario administrador debe haber sido dado de alta en el sistema y haber iniciado sesión
Descripción	<i>El sistema deberá comportarse como se describe el siguiente caso de uso cuando el usuario desee imprimir o guardar en PDF el informe de una red social</i>
Secuencia normal	<ol style="list-style-type: none"> 1. El usuario selecciona la opción de imprimir en una de las pestañas de la herramienta 2. El sistema recoge los datos escritos en HTML y crea una pantalla que muestra al usuario con las opciones de imprimir o guardar 3. El usuario selecciona si desea guardar o imprimir el fichero
Postcondición	El usuario ha obtenido un informe en PDF o lo ha impreso

Tabla 44: FRQ - 015 Imprimir/guardar informes

4.4. Reglas de negocio / Restricciones

Las restricciones o reglas de negocio describen un conjunto de operaciones y definiciones que se deben cumplir para el funcionamiento correcto del sistema.

A continuación se describen las restricciones del sistema:

CRQ - 001	Acceso al sistema
Descripción	Si el usuario no está registrado en el sistema no podrá acceder a éste

Tabla 45: CRQ - 001 Acceso al sistema

CRQ - 002	Gestión de usuarios
Descripción	Si el usuario no está loggeado como administrador no podrá gestionar usuarios

Tabla 46: CRQ - 002 Gestión de usuarios

CRQ - 003	Visualización de Perfil
Descripción	Los usuarios no podrán ver el perfil de otros usuarios que utilicen la plataforma

Tabla 47: CRQ - 003 Visualización de perfil

CRQ - 004	Visualización de analíticas
Descripción	Los widgets que muestran información de datos de cada una de las redes sociales sólo se mostrarán si tienen una cuenta de dicha red social asociada

Tabla 48: CRQ - 004 Visualización de analíticas

4.5. Requisitos no funcionales

Los requisitos no funcionales son restricciones que afectan al funcionamiento del sistema:

RNF - 001	Accesibilidad
Descripción	El sistema deberá contar con una interfaz web que le permita ser accesible desde cualquier navegador web

Tabla 49: RNF - 001 Accesibilidad

RNF - 002	Uso de una base de datos
Descripción	El sistema deberá utilizar una base de datos en la que guardar y recoger los datos que utilizará para su funcionamiento

Tabla 50: RNF - 002 Uso de base de datos

RNF - 003	Ejecución de Python/Django
Descripción	El sistema deberá permitir la ejecución y utilización del <i>framework</i> Django el cual está escrito en Python

Tabla 51: RNF - 003 Ejecución de Python/Django

RNF - 004	Compatibilidad
Descripción	El sistema deberá ser capaz de funcionar en los distintos navegadores existentes en el mercado así como en sus respectivas versiones

Tabla 52: RNF - 004 Compatibilidad

RNF - 005	Identificación de usuarios
Descripción	Los usuarios deberán acceder al sistema mediante unas credenciales creadas por éstos a la hora de realizar el registro, almacenadas en la base de datos del sistema y comprobadas por éste

Tabla 53: RNF - 005 Identificación de usuarios

RNF - 006	Fiabilidad
Descripción	El sistema deberá proporcionar una garantía de funcionamiento sin errores, respondiendo de forma rápida y fiable

Tabla 54: RNF - 006 Fiabilidad

NFR - 007	Permisos
Descripción	El sistema deberá distinguir entre los dos tipos de roles existentes en el sistema y realizar una acción u otra

Tabla 55: RNF - 007 Permisos

RNF - 008	UX/Usabilidad
Descripción	El sistema deberá ser fácil e intuitivo de utilizar para el usuario, facilitando si fuera necesario mensaje de ayuda para guiar a éste

Tabla 56: RNF - 008 UX/Usabilidad

RNF - 009	Disponibilidad
Descripción	El sistema deberá estar disponible para su uso 24/7

Tabla 57: RNF - 009 Disponibilidad

RNF - 010	Necesidad de Red de Internet
Descripción	El sistema deberá tener una red de Internet disponible para su funcionamiento correcto

Tabla 58: RNF - 010 Necesidad de red de Internet

4.6. Requisitos de información

Los requisitos de información son aquellos que describen la información que el sistema debe almacenar para su correcto funcionamiento.

IRQ - 001	Usuarios
Descripción	El sistema debe almacenar la información del usuario, en concreto su nombre, apellido, nombre de usuario, email y contraseña

Tabla 59: IRQ - 001 Usuarios

IRQ - 002	Perfil
Descripción	El sistema deberá almacenar el listado de las redes sociales que el usuario actual tenga asociadas y disponibles para su uso. Estas pueden ser Google Analytics, Facebook, Instagram y Twitter

Tabla 60: IRQ - 002 Perfil

IRQ - 003	Perfil de Google Analytics
Descripción	El sistema deberá almacenar información relacionada con el perfil de Google Analytics, en concreto, su email y contraseña de la cuenta y el <i>clientID</i>

Tabla 61: IRQ - 003 Perfil de Google Analytics

IRQ - 004	Perfil de Facebook
Descripción	El sistema deberá almacenar información relacionada con el perfil de Facebook, en concreto, su email y contraseña de la cuenta, el <i>pageID</i> y el <i>appID</i>

Tabla 62: IRQ - 004 Perfil de Facebook

IRQ - 005	Perfil de Twitter
Descripción	El sistema deberá almacenar información relacionada con el perfil de Twitter, en concreto, su email y contraseña de la cuenta y el <i>appToken</i>

Tabla 63: IRQ - 005 Perfil de Twitter

IRQ - 006	Perfil de Instagram
Descripción	El sistema deberá almacenar información relacionada con el perfil de Instagram, en concreto, su email y contraseña de la cuenta y el <i>appToken</i>

Tabla 64: IRQ - 006 Perfil de Instagram

IRQ - 007	Roles
Descripción	El sistema deberá almacenar el rol del usuario, es decir, si es administrador o usuario final

Tabla 65: IRQ - 007 Roles

4.7. Diagramas de secuencia

A continuación se presentan los diagramas de secuencia de los principales casos de uso presentados en el punto 4.5.

Como hemos podido ver, hay un conjunto de Casos de Uso que la única variante que tienen es la llamada a la API de la red social a la que se quieren conectar por lo que se ha simplificado la creación de los diagramas de secuencia en uno sólo.

- **Caso de uso 1: Registro**

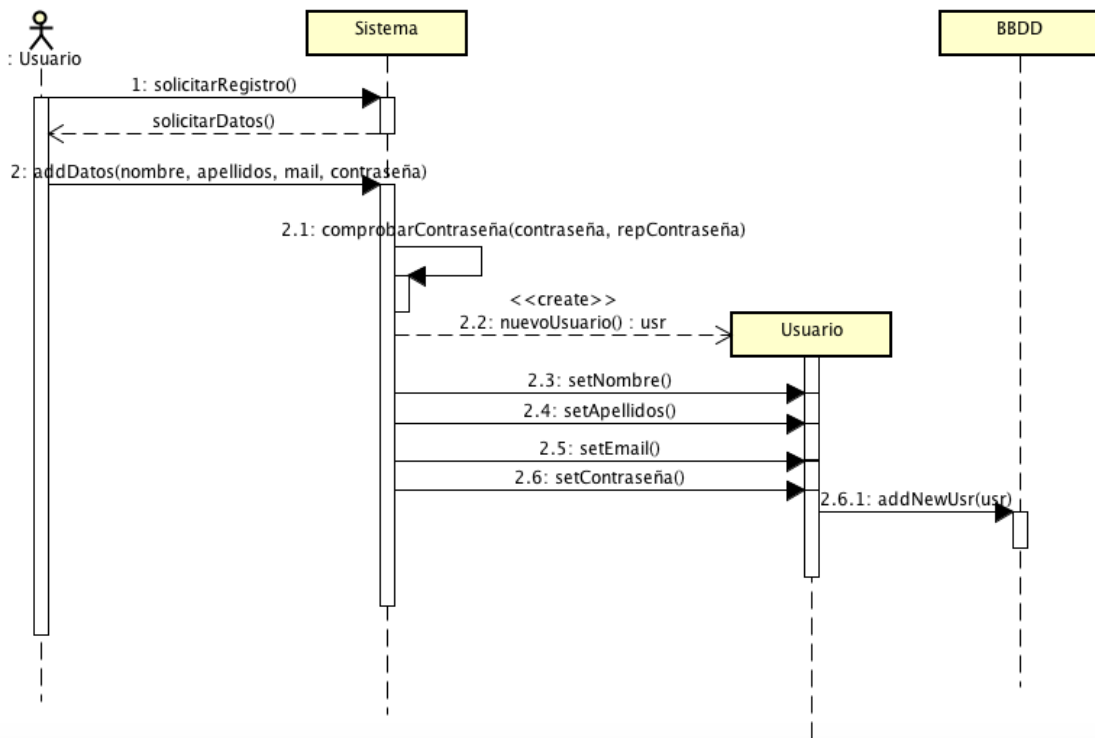


Ilustración 17: Diagrama de secuencia Registrarse

- **Caso de uso 2: Identificación**

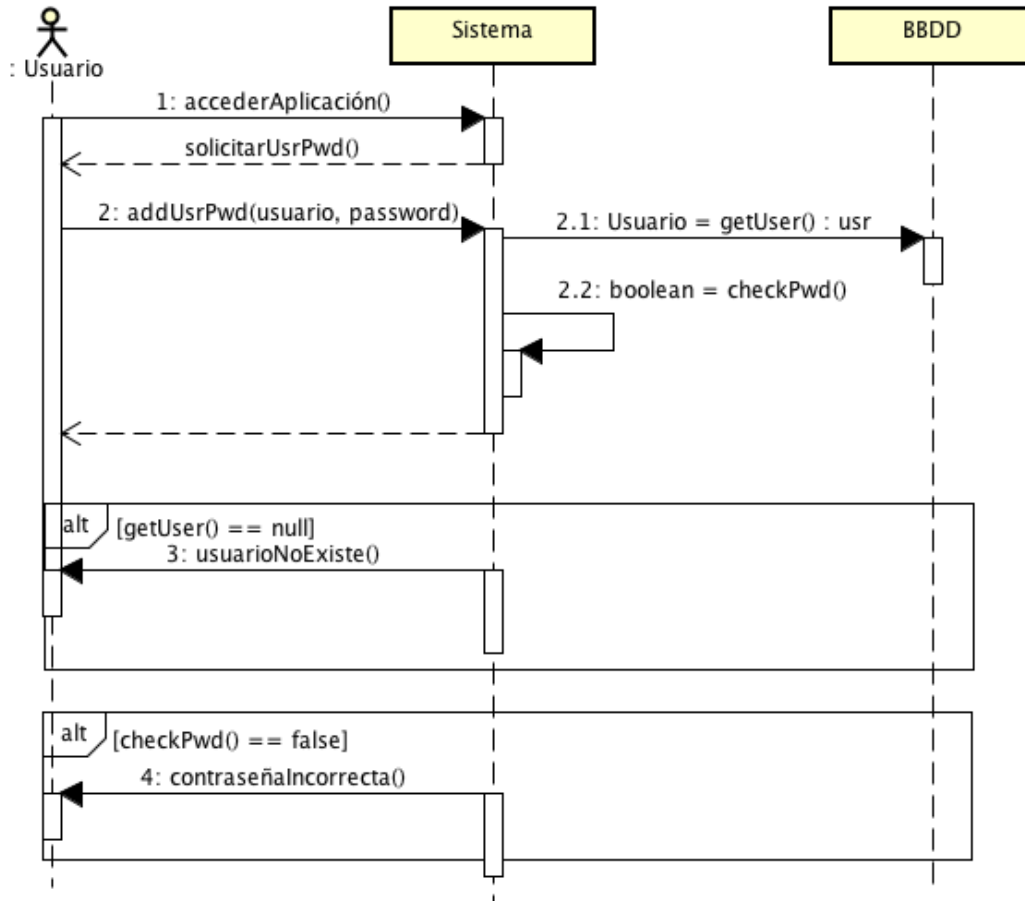


Ilustración 18: Diagrama de secuencia Identificación

- **Caso de uso 3: Gestión de usuarios**

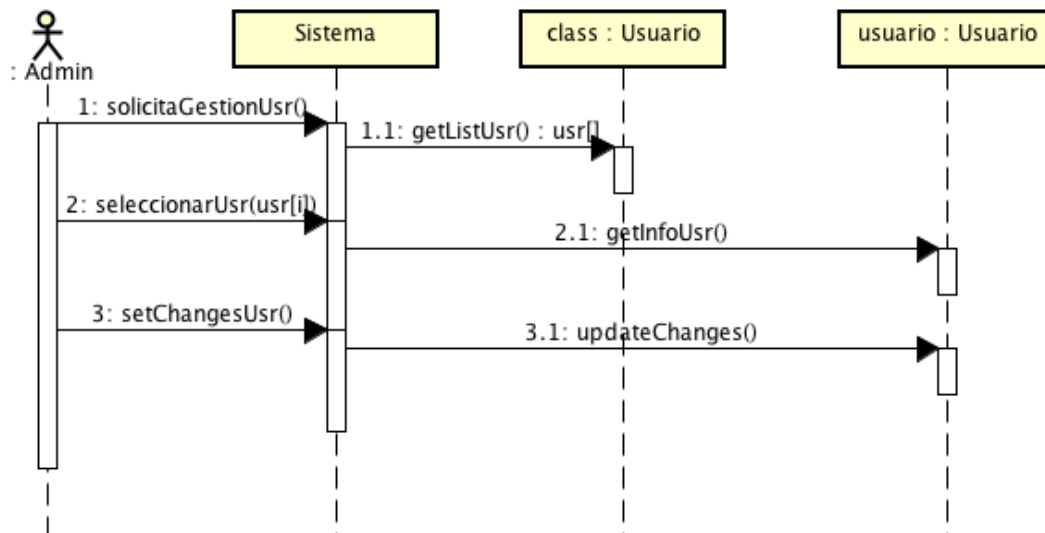


Ilustración 19: Diagrama de secuencia Gestión de Usuarios

- **Caso de uso 4: Añadir Perfil Google Analytics**

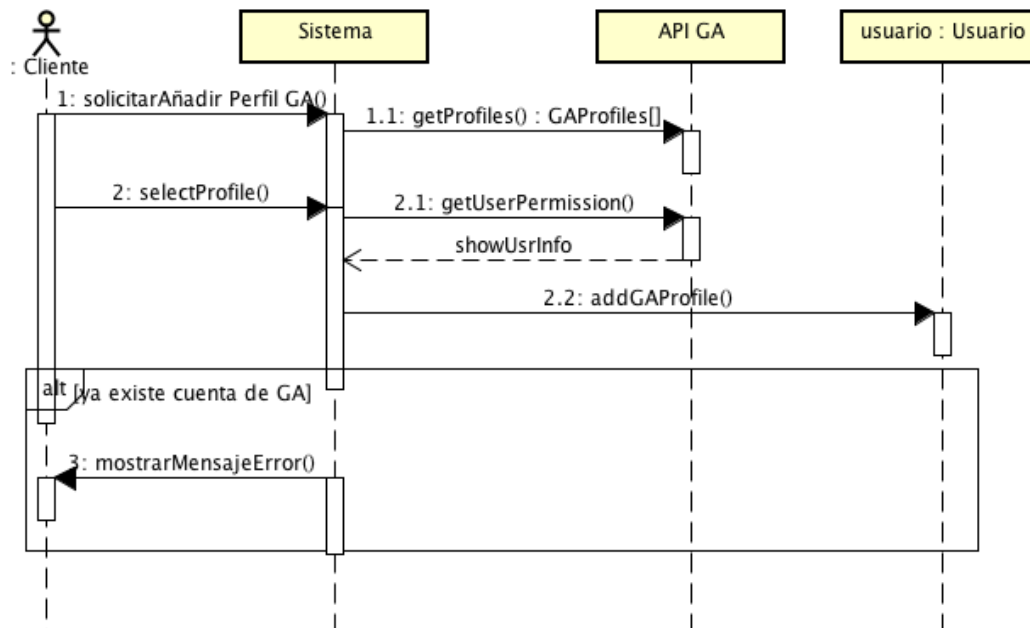


Ilustración 20: Diagrama de secuencia añadir perfil GA

- **Caso de uso 5: Obtener informes Google Analytics**

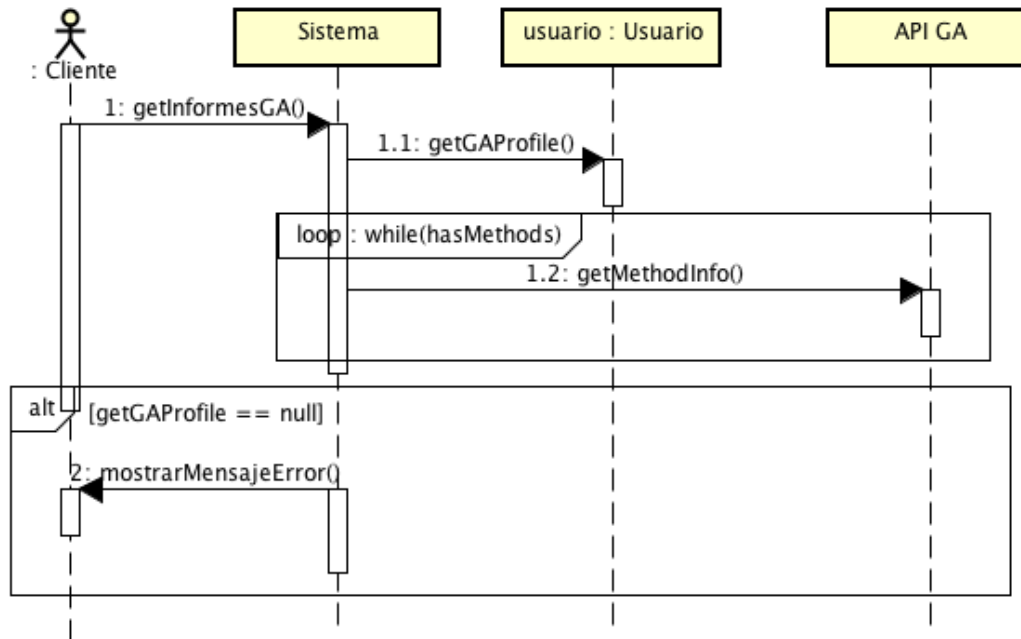


Ilustración 21: Diagrama de secuencia obtener informes GA

4.8. Diagramas de actividad

Los diagramas de actividades muestran procesos de negocio o procesos software en forma de flujo de trabajo, a través de una serie de acciones. Estas acciones pueden ser desarrolladas por personas, componentes software o equipos informáticos.

A continuación se presentan los diagramas de actividad de los casos de uso más importantes vistos en puntos anteriores.

- **Caso de uso 1: Registro**

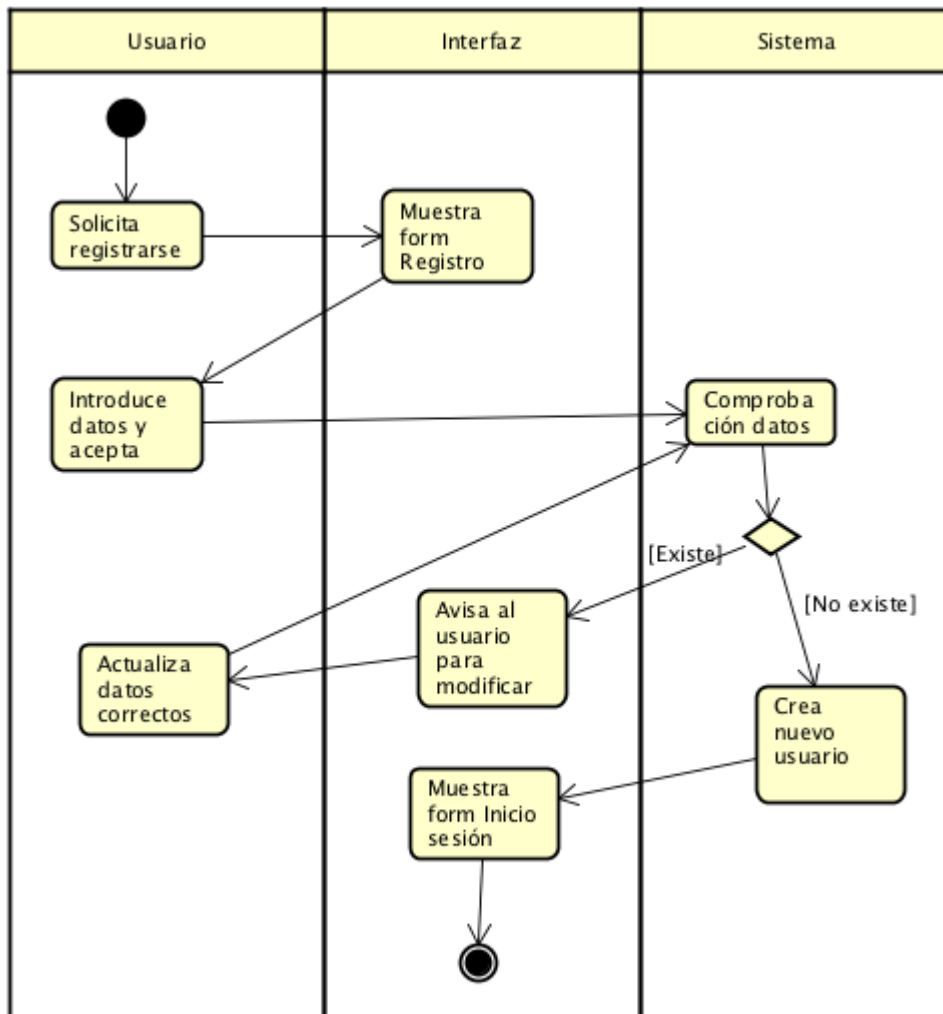


Ilustración 22: Diagrama de Actividad Registro

- **Caso de uso 2: Identificarse**

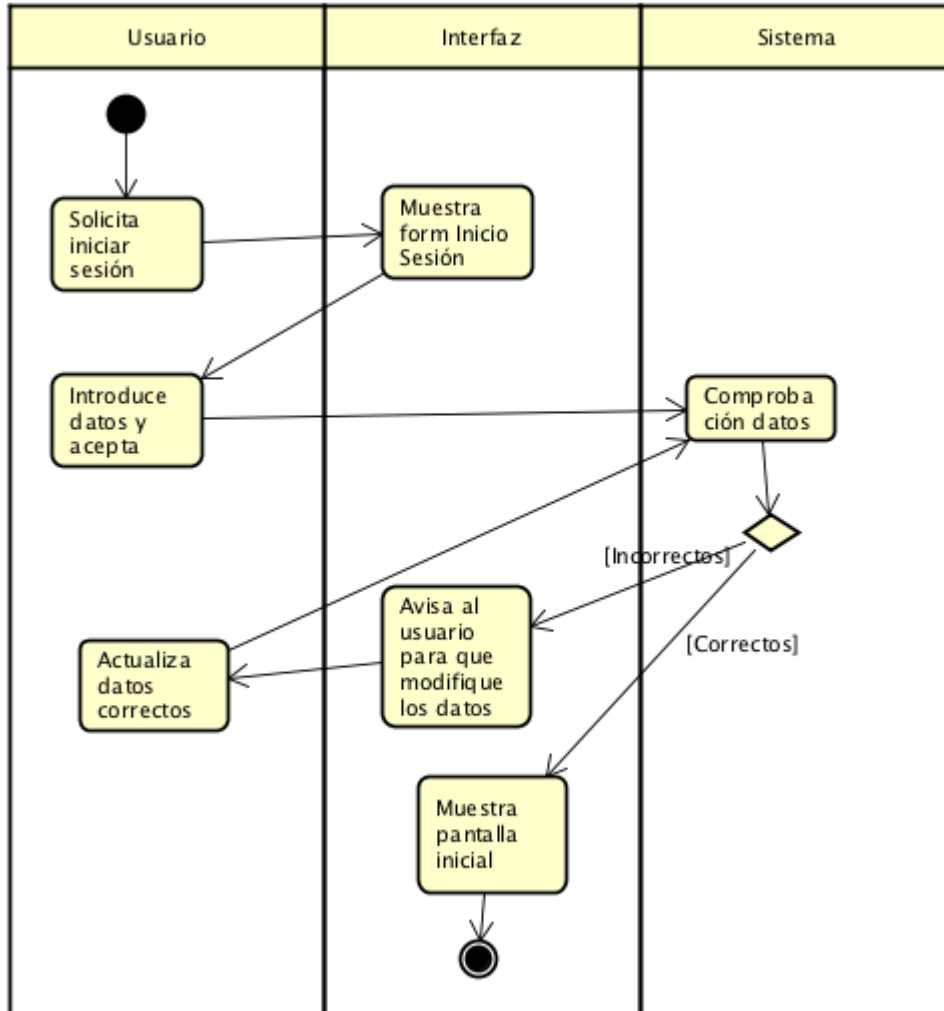


Ilustración 23: Diagrama de Actividad Identificarse

• **Caso de uso 3: Añadir Perfil Google Analytics**

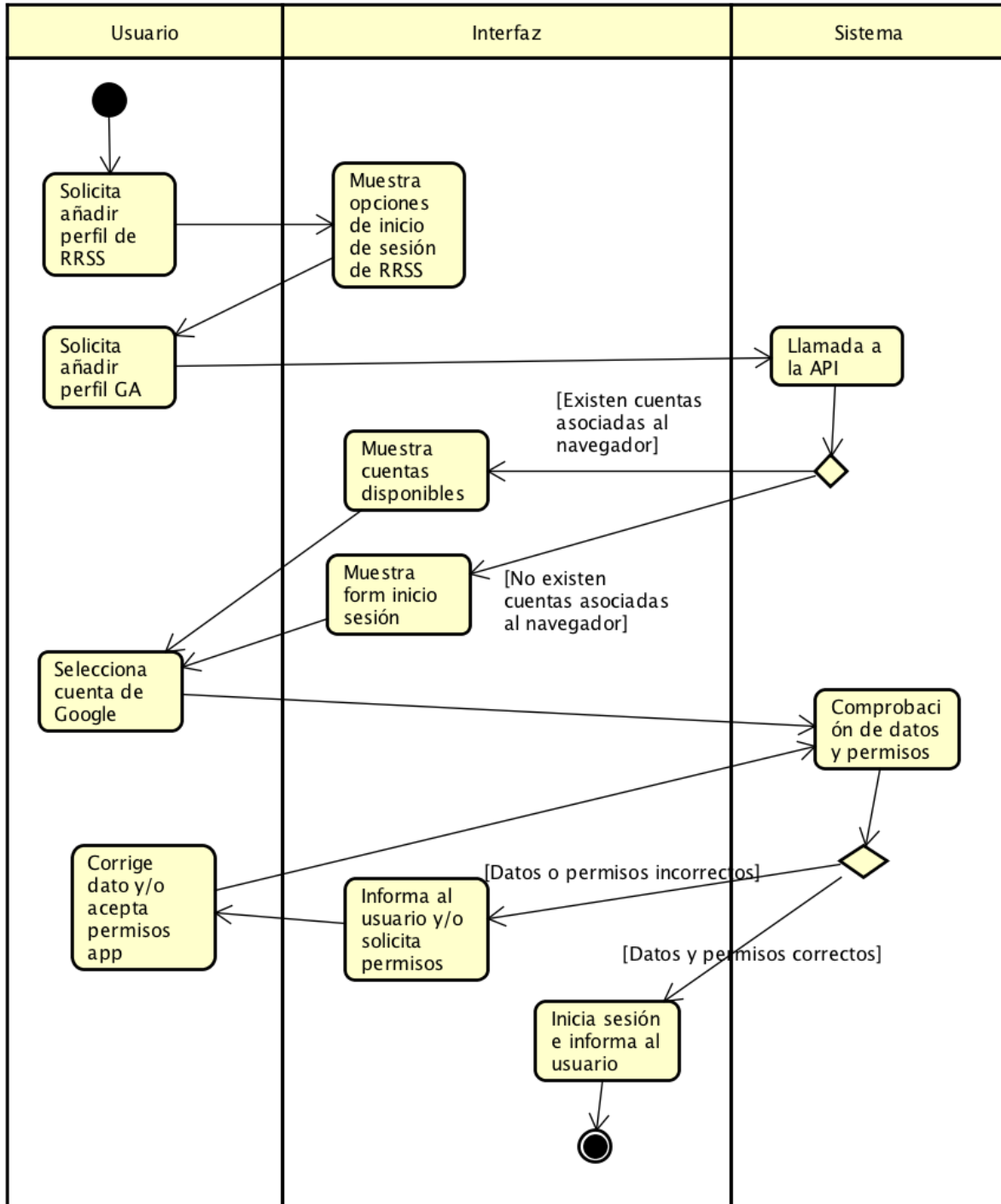


Ilustración 24: Diagrama de Actividad Añadir Perfil GA

- **Caso de uso 4: Obtener informes de Google Analytics**

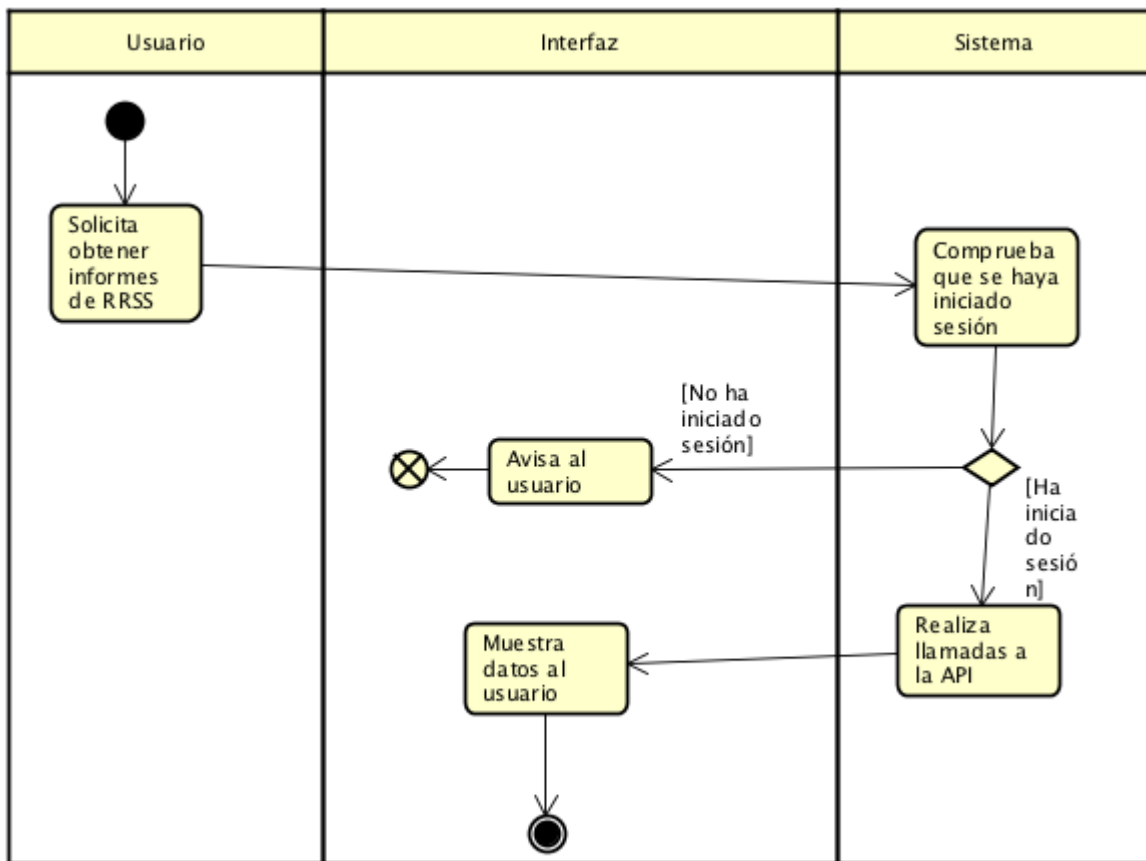


Ilustración 25: Diagrama de Actividad Obtener informes GA

4.9. Modelo del dominio

El modelo de dominio de la aplicación está formado por una serie de *Usuarios* que son los clientes finales de la herramienta. Los usuarios pueden tener dos roles cerrados diferentes. En principio, sólo hay un usuario *Administrador*, el cual puede realizar unas operaciones que el otro rol, *Cliente*, no puede.

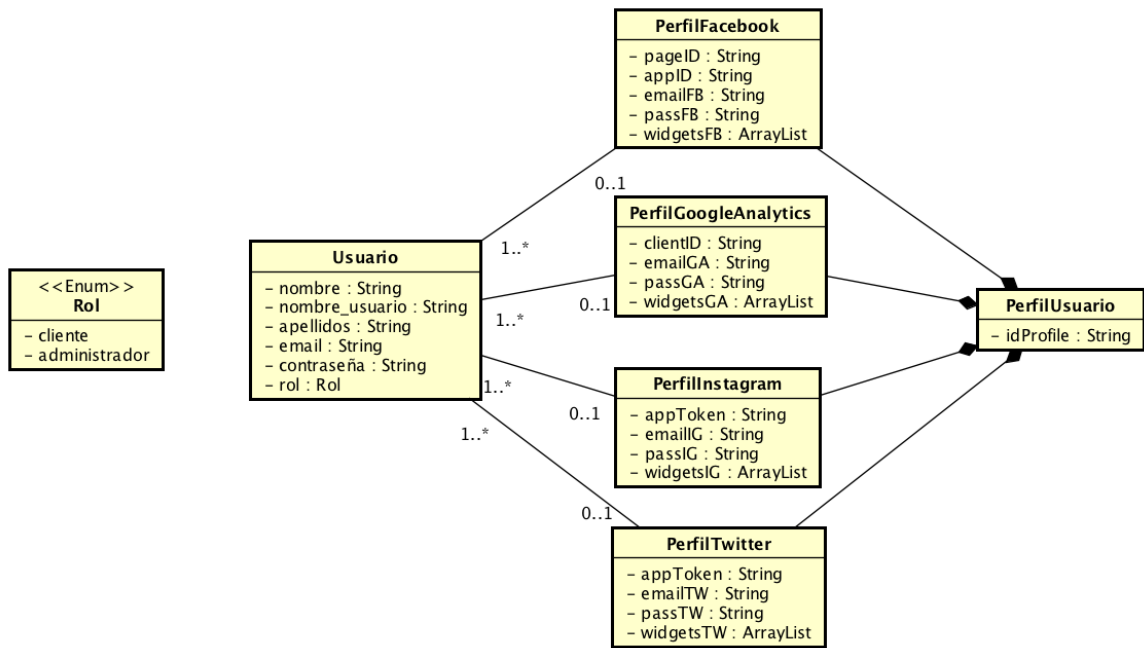


Ilustración 26: Diagrama del modelo de dominio

Cada uno de los usuarios *Cliente* tiene asociado un perfil de las distintas redes sociales a las que se puede conectar. Estas clases de los diferentes perfiles, almacenan un *ID* el cual sirve de “puente” para que nuestra herramienta se conecte con la API correspondiente y obtenga los datos necesarios.

Además, cada uno de los perfiles de las diferentes redes sociales tiene un mail y una contraseña, es decir, una cuenta y un listado de widgets en los que muestra el contenido consultado.

Estos perfiles, están englobados en un perfil de usuario de la propia aplicación, donde además de estos datos de los perfil, se almacena la información personal del usuario tal como nombre, apellidos, email...

Capítulo 5 : DISEÑO

En este apartado explicaremos todo lo relacionado con la arquitectura lógica y física del sistema así como el diagrama de diseño.

5.1. Arquitectura lógica

La arquitectura lógica de un sistema representa la estructura del sistema a un nivel de diseño lo más alto posible.

Cómo ya se explicó en el punto 1.2 en el cual se hablaba sobre Django, el esquema de arquitectura que siguen este tipo de proyectos es el de MVC, en concreto el patrón MTV.

Este diseño promueve el acoplamiento débil y la separación entre las distintas partes de la aplicación.

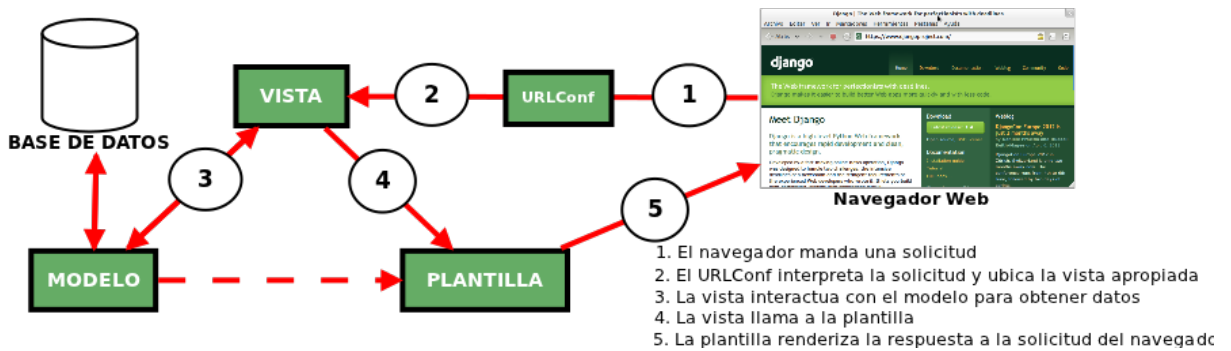


Ilustración 27: Estructura Django MTV

El patrón MTV está formado por las siguientes partes:

El **Modelo** es el encargado de definir y controlar el comportamiento de los datos que se van a utilizar dentro de la aplicación. Cada tipo de dato se almacena en una variable junto a sus parámetros y métodos.

La **Vista** se encarga de la gestión de datos, es decir, son un conjunto de funciones que permiten realizar determinadas funcionalidades tales como enviar un mail, hacer una consulta equivalente a una consulta SQL etc.

La **Plantilla** es el conjunto de ficheros HTML, CSS, JS... Estos archivos reciben los datos de la vista y se encarga de mostrarlos en el navegador. Django cuenta con un sistema propio de estructuras (bucles, condicionales, llamadas a ficheros estáticos etc.) escrita en Python por si es necesaria una presentación lógica de los datos. Estas estructuras son limitadas para permitir que la estructura lógica siga perteneciendo a la vista.

Además, tal y como se ve en la figura 2, Django posee un mapeo de URLs para controlar el despliegue de las vistas, es decir, dependiendo de cual sea la vista que está “operativa” en ese momento, Django utiliza las plantillas enviando unos parámetros u otros.

5.2. Arquitectura física

La arquitectura física hace referencia a los componentes físicos, es decir, cliente, servidor, servidor web, base de datos.. que participan en la herramienta y cómo interactúan entre ellos.

Nuestra aplicación consta de un cliente, que es el navegador y el cual se conecta al servidor, en este caso nuestra herramienta con estructura Django a través de Internet y del protocolo HTTP.

El servidor a su vez, necesita llamar a las APIs de cada uno de las redes sociales las cuales están alojadas en un server externo a nuestra aplicación.

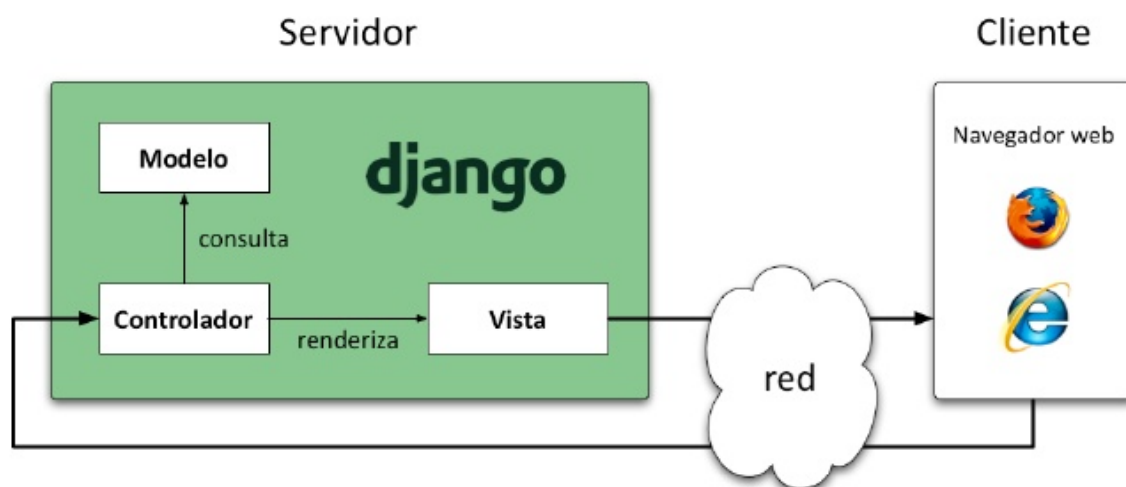


Ilustración 28: Estructura Django MTV

5.3. Diagramas de diseño

A continuación vamos a explicar la estructura de paquetes que sigue nuestra aplicación.

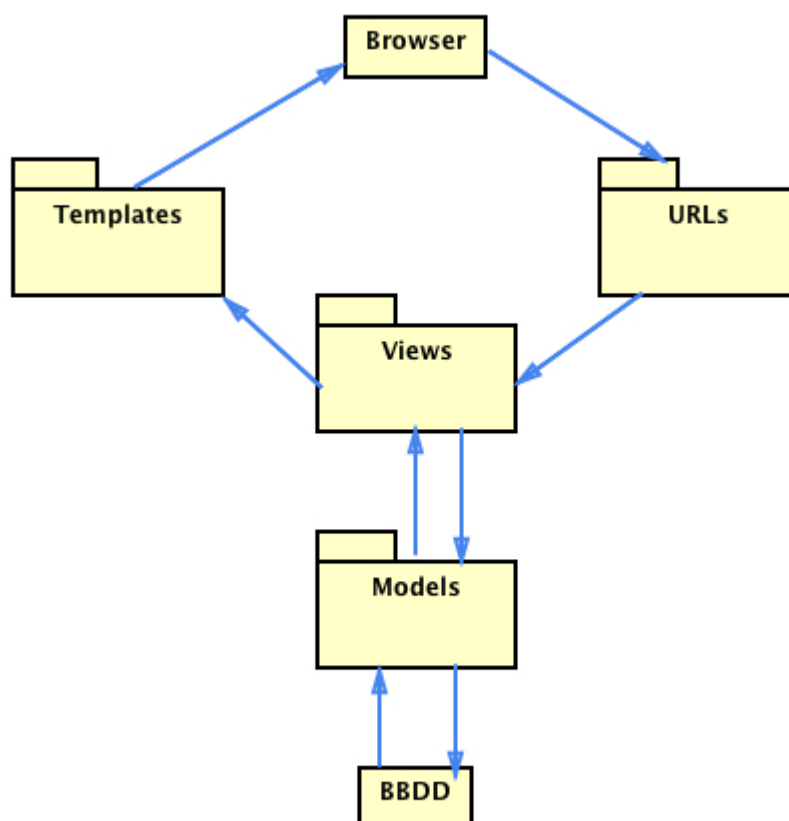


Ilustración 29: Diagrama estructura de diseño

Cada uno de los paquetes que se presentan en la Figura 29 contienen una serie de archivos y funcionalidad que se verá a continuación.

- **Templates:**

Este paquete contiene los ficheros de tipo *html* necesarios para poder mostrar los datos de nuestra aplicación. Hay una serie de ficheros que son comunes para todas las páginas y otros que son *template parts*, los cuales forman plantillas completas cuando se juntan con otras.

- **Base.html:** contiene el <head> general de la aplicación, con todos los ficheros que necesita para su buen comportamiento, carga el <header> y el bloque de contenido general.
- **Header.html:** construye la cabecera de la aplicación, dependiendo de si el usuario está *logeado* o no y mostrando una información u otra dependiendo del caso.
- **Home.html:** forma la pantalla principal que se muestra una vez el usuario haya iniciado sesión correctamente. Carga los nuevos ficheros necesarios para el buen funcionamiento, tanto ficheros JS internos como externos. Realiza llamadas a los *template-parts* de cada una de las pestañas de las redes sociales.
- **Login.html:** página principal al abrir la aplicación. Contiene un formulario donde el usuario debe indicar su nombre de usuario y contraseña para poder acceder a su *dashboard*.
- **Signup.html:** formulario para que el usuario pueda registrarse en la aplicación y poder acceder posteriormente.
- **Profile.html:** muestra al usuario la información sobre su perfil y sobre las cuentas disponibles de la aplicación, indicando si se ha realizado un inicio de sesión o no de cada una de ellas.
- **Template_analytics.html:** muestra un conjunto de *widgets* con los datos correspondientes a cada uno de ellos obtenidos tras las llamadas a la API de Google Analytics.
- **Template_facebook.html:** muestra un conjunto de *widgets* con los datos correspondientes a cada uno de ellos obtenidos tras las llamadas a la API de Facebook.
- **Template_twitter.html:** muestra un conjunto de *widgets* con los datos correspondientes a cada uno de ellos obtenidos tras las llamadas a la API de Twitter.
- **Template_instagram.html:** muestra un conjunto de *widgets* con los datos correspondientes a cada uno de ellos obtenidos tras las llamadas a la API de Instagram.

Todos estos ficheros pertenecientes al paquete *template* dependen de otro paquete a su mismo nivel llamado *static*. Este paquete contiene subdirectorios en los que se incluyen los ficheros JS, SASS, CSS, los archivos de las fuentes, las imágenes y el *favicon*.

Vamos a centrarnos en el subdirectorio que contiene los ficheros JavaScript pues contienen la funcionalidad principal de la aplicación.

- **Custom.js:** inicializa la llamada a la librería *hello.js* y contiene métodos genéricos, como por ejemplo los selectores de fechas utilizados en las diferentes secciones.
- **Profile.js:** realiza los inicios de sesión de las diferentes redes sociales
- **Analytics.js:** contiene las funciones necesarias para obtener los datos requeridos y enviárselos a la *template* adecuada.
- **Facebook.js:** contiene las funciones necesarias para obtener los datos requeridos y enviárselos a la *template* adecuada.
- **Twitter.js:** contiene las funciones necesarias para obtener los datos requeridos y enviárselos a la *template* adecuada.
- **Instagram.js:** contiene las funciones necesarias para obtener los datos requeridos y enviárselos a la *template* adecuada.

5.4. Diseño de la Base de Datos

Por defecto, Django trae configurado sqlite3 para el uso de bases de datos. Esta configuración la encontramos dentro del fichero *settings.py*, en la opción *DATABASES*.

La base de datos cuenta con nueve entidades que son las predeterminadas de Django, ya que en esta fase del desarrollo de la aplicación no han sido necesarias más. En el diagrama mostrado a continuación se pueden ver cuáles son y sus relaciones.

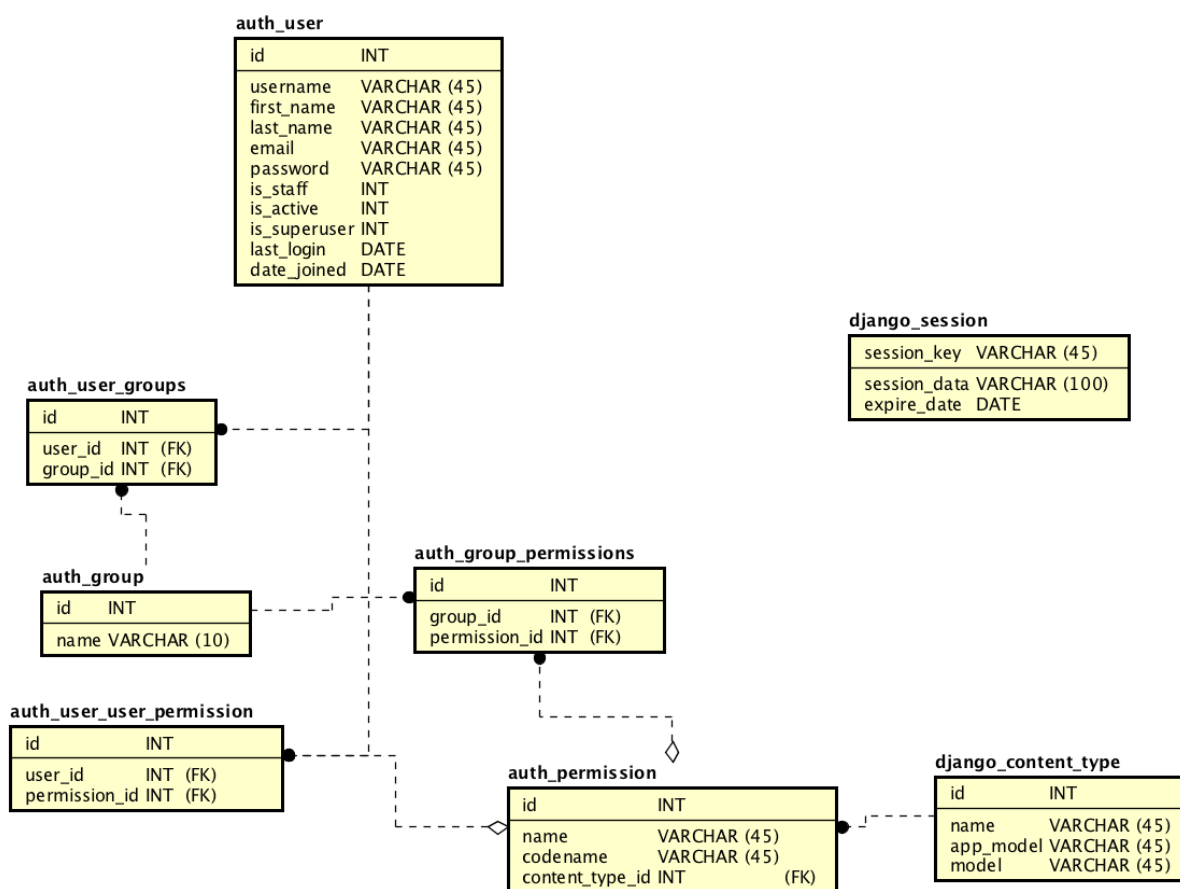


Ilustración 30: Diagrama de la base de datos

Como se puede ver en el diagrama, en la base de datos actual guardamos información sobre los usuarios registrados, los grupos a los que pertenecen y sus permisos. Además, dependiendo de estos permisos, el usuario podrá visualizar unas cosas u otras. En nuestro caso, sólo tenemos dos usuarios distintos, el administrador que sería el *superuser* y el resto de clientes, usuarios finales de la herramienta.

5.5. Diccionario de datos

A continuación se explica cada entidad mediante un diccionario de datos para comprender el uso y estructura de cada tabla.

Entidad auth_user					
Usuarios registrados en el sistema					
Columna	Tipo	Tamaño	Nulo	Restricción	Descripción
Id	INT		NO	PRIMARY KEY	Id del usuario
Username	VARCHAR	30	NO		Nombre de usuario
First_name	VARCHAR	30	NO		Nombre
Last_name	VARCHAR	30	NO		Apellido
Email	VARCHAR	75	NO		Email
Password	VARCHAR	128	NO		Contraseña
Id_active	INT		NO		Indica si está activo o no
Is_staff	INT		NO		Indica si el usuario puede acceder a la pantalla <i>admin</i> o no
Is_superuser	INT		NO		Indica si es el <i>superuser</i> o no
Last_login	DATE	10	NO		Última visita
Date_joined	DATE	10	NO		Fecha de creación

Tabla 66: Diccionario de datos auth_user

Entidad auth_user_groups					
Grupos de usuarios registrados en el sistema					
Columna	Tipo	Tamaño	Nulo	Restricción	Descripción
id	INT		NO	PRIMARY KEY	Id del grupo de usuarios
User_id	INT		NO	FOREIGN KEY	Id del usuario
Group_id	INT		NO	FOREIGN KEY	Id del grupo

Tabla 67: Diccionario de datos auth_user_groups

Entidad auth_group					
Grupos registrados en el sistema					
Columna	Tipo	Tamaño	Nulo	Restricción	Descripción
Id	INT		NO	PRIMARY KEY	Id del grupo
name	VARCHAR	80	NO		Nombre del grupo

Tabla 68: Diccionario de datos auth_group

Entidad auth_group_permissions					
Permisos que tienen los diferentes grupos					
Columna	Tipo	Tamaño	Nulo	Restricción	Descripción
Id	INT		NO	PK	Id del permiso del grupo
Group_id	INT		NO	FK	Id del grupo
Permission_id	INT		NO	FK	Id del permiso

Tabla 69: Diccionario de datos auth_group_permissions

Entidad django_content_type					
Tipo de contenido que puede ver el usuario/grupo dependiendo de los permisos					
Columna	Tipo	Tamaño	Nulo	Restricción	Descripción
Id	INT		NO	PK	Id del tipo de contenido
app_label	VARCHAR	100	NO		Etiqueta del tipo de contenido
model	VARCHAR	100	NO		Tipo de modelo visible

Tabla 70: Diccionario de datos django_content_type

Entidad auth_user_user_permissions					
Permisos de los usuarios					
Columna	Tipo	Tamaño	Nulo	Restricción	Descripción
Id	INT		NO	PK	Id del permiso del usuario
User_id	INT		NO	FK	Id del usuario
Permission_id	INT		NO	FK	Id del permiso

Tabla 71: Diccionario de datos de auth_user_user_permissions

Entidad auth_permission					
Permisos que puede tener un usuario/grupo registrado en el sistema					
Columna	Tipo	Tamaño	Nulo	Restricción	Descripción
Id	INT		NO	PK	Id del permiso
name	VARCHAR	50	NO		Nombre del permiso
content_type_id	INT		NO	FK	Id del contenido que es visible con dicho permiso
codename	VARCHAR	100			Nombre del permiso codificado

Tabla 72: Diccionario de datos de auth_permission

Capítulo 6 : IMPLEMENTACIÓN

En este apartado se va a explicar cómo se ha llevado a cabo la implementación de la aplicación.

Después de haber realizado el análisis y diseño, explicados en puntos previos, la implementación se ha llevado a cabo en varias etapas:

6.1. Creación de la estructura Django

La primera fase del proyecto consistía en construir la estructura base del proyecto, la cual se hizo con Django.

Una vez tengamos instalado Django y hayamos creado nuestro primer proyecto tal y como se explica en el punto 10, *Manual de Instalación*, obtendremos un árbol de estructura del proyecto.

La estructura contiene los ficheros `manage.py`, el cual permite la administración del sitio, el archivo de configuración `settings.py`, `urls.py` con los patrones que necesita para resolver las `urlconfig...`

```
vitalboard
├── manage.py
└── mysite
    ├── settings.py
    ├── urls.py
    ├── wsgi.py
    └── __init__.py
```

Por defecto, Django utiliza como base de datos `sqlite3`, lo cual ya viene configurado en el fichero `settings.py`. Para crear una base de datos sólo necesitamos ejecutar lo siguiente.

```
(myvenv) ~/djangogirls$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, contenttypes, auth, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying sessions.0001_initial... OK
```


6.2. Registro y login de usuarios

Una vez construida la estructura base, se procedió a crear las primeras plantillas para el registro y *login* de usuarios. Como se explicará en el punto 7.1 , una de las ventajas que nos ofrece Django es la gestión de usuarios, la cual viene preparada de base.

En el fichero *settings.py* hay que indicar cual es la url de *login* y *logout* de la aplicación, así como la url de inicio.

```
92
93 LOGIN_URL = 'login'
94 LOGOUT_URL = 'logout'
95 LOGIN_REDIRECT_URL = 'home'
96
```

Ilustración 31: *Settings.py* de Django

Además, en el fichero *urls.py* debemos indicar cual es la plantilla correspondiente a cada una de estas urls. El fichero *url* está formado por expresiones regulares facilitando la creación de éstas y reduciendo el proceso.

```
1 from django.conf.urls import url, include
2 from django.contrib.auth import views as auth_views
3
4 from mysite.core import views as core_views
5 from django.contrib import admin
6
7 urlpatterns = [
8     url(r'^admin/', include(admin.site.urls)),
9     url(r'^$', core_views.profile, name='home'),
10    url(r'^login/$', auth_views.login, {'template_name': 'login.html'}, name='login'),
11    url(r'^logout/$', auth_views.logout, {'next_page': 'login'}, name='logout'),
12    url(r'^signup/$', core_views.signup, name='signup'),
13    url(r'^profile/$', core_views.profile, name='profile'),
14 ]
15
```

Ilustración 32: *Urls.py* de Django

Por último, se deben crear las plantillas e incluir ambos formularios, e indicar a Django qué es el formulario de registro de usuarios y de *login*.

6.3. Conexión con las APIs

Una vez realizada la estructura del proyecto y creados los formularios de registro y login de la aplicación, se procedió a la conexión con las distintas APIs.

Para ellos, se creó un fichero común a todos que es el *profile.js* donde se realiza la primera conexión, bien sea directamente o mediante la biblioteca *hello.js* para posteriormente crear el inicio de sesión de cada de ellas.

Después, se creó un fichero JavaScript para cada una de las plataformas, donde se realizan las llamadas a los métodos necesarios y una vez obtenida la *response* de dicha llamada, se capturan los datos y se filtran para su mostrado, tal y como se explica en el punto siguiente.

6.4. Filtrado de datos y maquetación

Como se ha explicado en el punto anterior, en cada uno de los ficheros JavaScript de las APIs se realiza un conjunto de operaciones para la obtención de datos. Estos datos se obtienen en un formato JSON los cuales hay que filtrar para su posterior visualización.

Cada uno de los JSON devuelve una serie de datos pero los que tienen valor por el contexto son los *data*.

En el ejemplo que vemos en la imagen inferior, se ha hecho una llamada a la API de Facebook para obtener el número de impresiones que ha tenido nuestra página en un periodo de tiempo determinado.

```

/*****
IMPRESIONES DE LA PÁGINA
*****/
facebookApi.api(
  '/' + pageId + '/insights/page_impressions?since=' + dateStart + '&until='
+ dateEnd, 'GET', {},
  function(response) {
    var index = response.data[0].values.length;
    $('#fb_page-impressions .widget-content').html('<i class="fa fa-eye"
style="color:#5bc0de"></i> ' + response.data[2].values[index - 1].value);
    $('#fb_total_impresions').html('<i class="fa fa-eye"
style="color:#2196F3"></i> <span class="badge" style="background-color:#2196F3">'
+ response.data[2].values[index - 1].value);
  }
);

```

Podemos ver como la llamada nos devuelve una respuesta la cual se trata posteriormente. Si nos fijamos en el tratamiento, nos quedamos con los valores de *data*, tal y como se ha explicado anteriormente.

Una ventaja de este tipo de APIs es que nos dan los datos de tres formas distintas si hablamos de espacio temporal. El JSON contiene 3 arrays principales, uno con los datos de nuestra petición día a día, otro acumulativo semanal y otro mensual, representados como *data[0]*, *data[1]* y *data [2]*.

Como podemos ver en este caso, mostramos el dato acumulativo mensual, ya que al estar buscando información dentro de un periodo de tiempo concreto, obtenemos el dato directamente sin tener que estar haciendo un acumulativo por días.

Finalmente decir, que la maquetación de la herramienta se ha hecho para dispositivos de escritorio así como móviles, es decir, un desarrollo *responsive*, tal y como se puede ver en la figura 33.

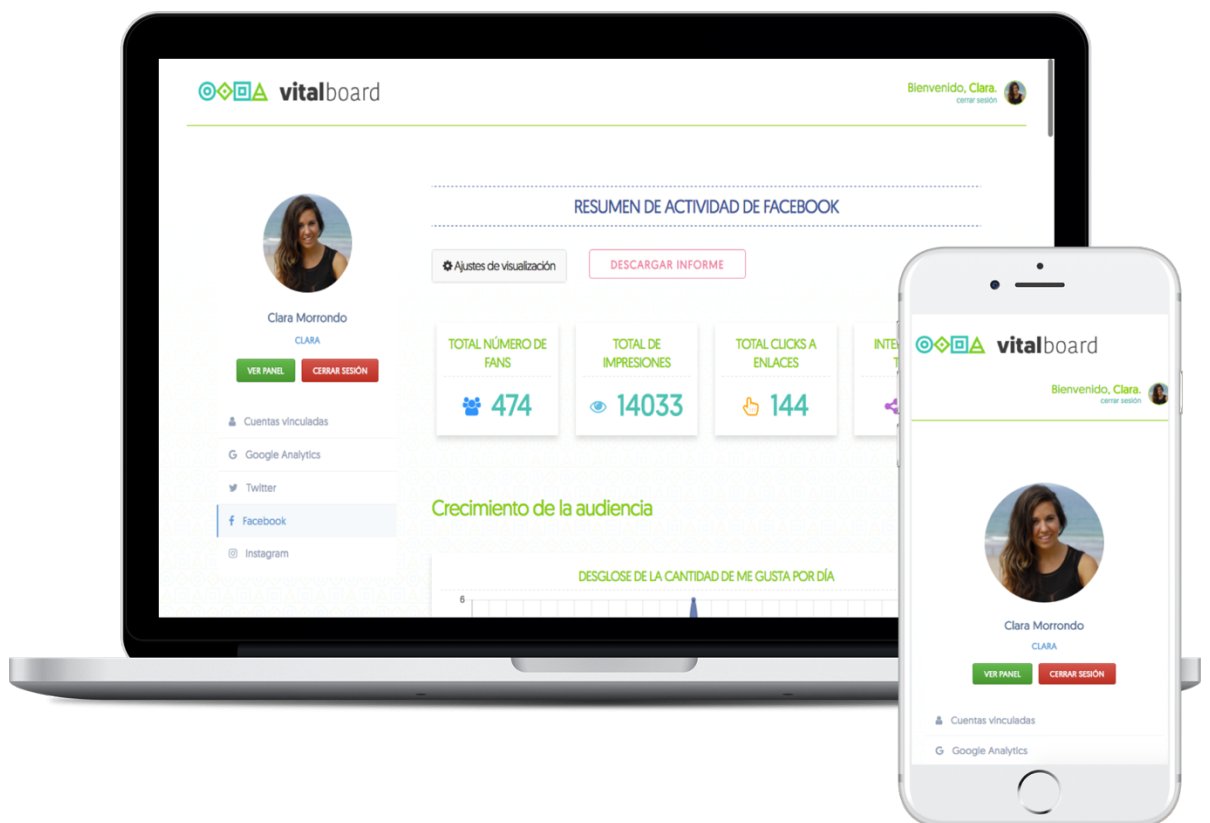


Ilustración 33: Vista de la aplicación en escritorio y móvil

Capítulo 7 : TECNOLOGÍA UTILIZADA

En este apartado se va a explicar la tecnología utilizada para la implementación del proyecto, tanto lenguajes de programación como *frameworks* y llamadas a APIs.

7.1. Django

Para entrar en el contexto es importante comenzar diciendo que la estructura de la aplicación se ha realizado con Django, un *framework* web de alto nivel escrito en Python, con mucha potencia y el cual te permite realizar aplicaciones web a medida y de una forma muy ágil.

Además, se ajusta al patrón de diseño MVC el cual permite crear aplicaciones web extensibles, escalables y fáciles de mantener.



Ilustración 34: Logo del framework Django

Una de las ventajas más importantes de Django es que proporciona una interfaz automática de administrador la cual puede incluirse en cualquier página desarrollada con dicho *framework*. Esto permite la creación, actualización y borrado de contenido así como de usuarios, llevando un registro en todo momento de los cambios realizados.

Como hemos dicho anteriormente, Django sigue la estructura MVC. En realidad, el nombre que se le da es MTV ya que la vista se llama *Template* y el controlador es la *Vista*.

Por defecto, Django trae unos archivos predeterminados. Estos son los siguientes:

- **__init__.py**: es un fichero vacío que le dice a Python que el directorio en el que se encuentra es un paquete de Python. Puede incluir código de inicialización o la variable `__all__` la cual funciona como un `import*` y anula el comportamiento predeterminado de anular todo lo que empiece con un *underscore*.
- **manage.py**: es un script el cual contiene código que permite interactuar con el proyecto como por ejemplo iniciar un servidor web en nuestro ordenador.
- **settings.py**: contiene la configuración del proyecto.
- **urls.py**: contiene el listado de rutas y patrones para que *URLConf* pueda utilizarlos.

Dentro de la aplicación, en la figura 3 sería el subdirectorio *vitalboard* encontramos otro conjunto de archivos predeterminados:

- **__init__.py**: misma funcionalidad que la explicada en las líneas superiores.
- **models.py**: se definen todos los objetos llamados Models con sus respectivas variables y métodos.
- **views.py**: contiene la lógica de la aplicación, es decir, el conjunto de funciones que utilizarán los modelos para pasárselo a las plantillas.
- **test.py**: se declaran las pruebas necesarias para la aplicación, es decir, es una suite de test para automatizar las pruebas.

```
vitalboard_project
├── mysite
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   ├── manage.py
│   └── vitalboard
│       ├── migrations
│       ├── __init__.py
│       ├── admin.py
│       ├── models.py
│       ├── tests.py
│       └── views.py
```

Ilustración 35: Estructura básica de un proyecto Django

Además, es importante decir que Django utiliza ORM, es decir, un mapeador objeto-relacional, un interfaz que convierte el sistema de tipos del lenguaje de programación y la base de datos relacional con motor de persistencia.

Con esto se consigue una simplificación en el proceso de desarrollo al utilizar clases y objetos, abstracción e independencia con la base de datos utilizada, posibilidad de realizar acciones típicas de las bases de datos como por ejemplo un *trigger*...

Por último decir que es OpenSource, un punto muy interesante para todas aquellas personas que nos dedicamos al mundo del desarrollo porque podemos modificar y estudiar el código fuente sin ningún problema.

7.2. SASS

Sass (*Syntactically Awesome StyleSheets*) es una extensión del lenguaje de estilos CSS el cual añade características muy potentes tales como:

- Permite el uso de variables, lo que hace que el código sea más limpio
- Anidación de reglas, evitando así la repetición de código
- *Mixins* los cuales funcionan como funciones definiendo estilos reutilizables admitiendo argumentos en sus llamadas

Sass tiene dos tipos de sintaxis, una de ellas conocida como SCSS la cual es una extensión de CSS3 y la otra es la propia de Sass o sintaxis indentada donde las llaves propias de CSS se sustituyen por indexación.

Para el desarrollo del proyecto se ha seguido el primer tipo de sintaxis con ficheros con extensión `.scss`

```
1 @media(max-width: 720px){
2   .tab{
3     flex-wrap: wrap;
4   }
5   .tab-btn-container{
6     width: 100%;
7     text-align: center;
8     button{
9       width: 300px;
10      margin-bottom: 1em;
11      color: $green_color;
12    }
13  }
14 }
15
```

Ilustración 36: Sintaxis de Sass como SCSS

7.3. GULP

Gulp es un automatizador de tareas escrito en JavaScript que nos permite empaquetar y minificar ficheros SCSS, JavaScript, optimiza imágenes...Es fácil de instalar y configurar ya que con un par de comandos podemos obtener toda la potente funcionalidad.

En resumen, nos ofrece estas características:

“Automatización: Gulp es un conjunto de herramientas que nos ayuda a automatizar las tareas pesadas o que consumen mucho tiempo en nuestro flujo de trabajo de desarrollo.

Plataforma-agnóstica: Las integraciones están incluidas en la mayor parte de los IDEs y la gente lo combina con PHP, .NET, Node.js, Java y otras plataformas.

Ecosistema fuerte: Usar módulos de npm para hacer lo que queramos con más de 2000 plugins para la transformación de ficheros de streaming

Simple: Proporcionando una API mínima, gulp es fácil de aprender y simple de utilizar.”

7.4. JavaScript

JavaScript es un lenguaje interpretado y orientado a objetos utilizado principalmente como lenguaje para la creación de scripts aunque también es la base para otras plataformas como Node.js.

Principalmente se usa del lado del cliente aunque también existe una forma de usarlo de parte del servidor. Su uso en aplicaciones es muy extendido, por ejemplo, para la implementación de widgets.

Otra ventaja que ofrece JavaScript es la gran cohesión que tiene con la mayoría de APIs desarrolladas en el mercado, con una alta documentación y facilidad de integración.

7.5. API de Google Analytics

La API de Google Analytics nos ofrece un conjunto de funciones y procedimientos para obtener datos de aquellos sitios que tengamos alojados en Google Analytics.

Google Analytics nos ofrece diferentes tipo de APIs, pero en la realización del proyecto se ha utilizado la API insertada la cual está escrita en JavaScript y nos permite *logear* al usuario y crear e insertar widgets en nuestro sitio web.

Para conectarnos con la API de Google Analytics, lo primero que tenemos que hacer es conseguir un ID de cliente. Simplemente accediendo a la consola de la API de Google e indicando nuestro sitio web obtendremos dicho código.

Posteriormente, en nuestra aplicación debemos cargar la biblioteca para seguidamente invocar los *return* de las funciones que utilizaremos.

```
<script>
(function(w, d, s, g, js, fjs){
  g=w.gapi||(w.gapi={});g.analytics={q:[], ready:function(cb){this.q.push(cb)}};
  js=d.createElement(s);fjs=d.getElementsByTagName(s)[0];
  js.src='https://apis.google.com/js/platform.js';
  fjs.parentNode.insertBefore(js, fjs);js.onload=function(){g.load('analytics')};
})(window,document, 'script');
</script>
```

Ilustración 37: Inicialización de la biblioteca de la API de inserción de Google Analytics

El siguiente paso es autorizar al usuario mediante la creación de un botón que permita el inicio de sesión y los requisitos necesarios para dicha aplicación. Esto se realiza mediante *OAuth 2.0*, el cual es un protocolo para la autenticación segura mediante APIs en sitios móvil, web etc.

```
gapi.analytics.auth.authorize({
  container: 'auth-button',
  clientid: CLIENT_ID,
});
```

Ilustración 38: Autorización del usuario para el uso de la API

Por último, faltaría la creación de los contenedores y la llamada a cada una de las funciones que nos permite la API [11].

7.6. API de Twitter

Para la implementación de la API de Twitter se ha utilizado una librería llamada *hello.js*. Esta librería escrita en JavaScript facilita la autenticación mediante *OAuth2* a determinadas API Rest, como por ejemplo Twitter.

La instalación es tan sencilla como incluir el fichero *hello.js* [15] e inicializarlo, indicando a qué API queremos llamar y asociando el app ID indicado:

```
hello.init({
  facebook: FACEBOOK_CLIENT_ID,
  windows: WINDOWS_CLIENT_ID,
  google: GOOGLE_CLIENT_ID
}, {redirect_uri: 'redirect.html'});
```

Ilustración 39: Inicialización de *hello.js*

Una vez realizado esto, ya sólo tendremos que crear los contenedores y realizar las llamadas necesarias para mostrar en dichos contenedores.

Cada llamada devuelve una response con una serie de datos los cuales se deben filtrar hasta obtener el buscado. Este filtro se realiza con JavaScript y se manda a la plantilla adecuada para ser mostrado en el navegador.

```
▼ Object {data: Array(3), paging: Object} ⓘ
  ▼ data: Array(3)
    ▼ 0: Object
      description: "Daily: The number of impressions seen of any content associated with your Page. (Total Count)"
      id: "87650822978/insights/page_impressions/day"
      name: "page_impressions"
      period: "day"
      title: "Daily Total Impressions"
      ▼ values: Array(3)
        ▼ 0: Object
          end_time: "2017-04-21T07:00:00+0000"
          value: 342
          ▶ __proto__: Object
        ▶ 1: Object
        ▶ 2: Object
          length: 3
        ▶ __proto__: Array(0)
      ▶ __proto__: Object
```

Ilustración 40: Response obtenida de llamada a API de Twitter

Esta librería ofrece una gran cantidad de métodos y una muy buena documentación, la cual facilita enormemente el trabajo [25]

Además, la propia API de Twitter nos ofrece el listado de todos los métodos *Get* y *Post* disponibles con dicha API (es importante diferenciar entre al API de Twitter y la de Twitter Ads).

7.7. API de Facebook

La API de Facebook se basa en un SDK que proporciona al usuario un gran número de funciones y simplificando la obtención de datos mediante la API Graph.

La API Graph está formada por tres elementos principales:

- **Nodos:** son los objetos sobre los que obtendremos las métricas como puede ser el usuario, una página, fotos...
- **Perímetros:** conectan los distintos nodos entre sí
- **Campos:** devuelven información sobre un nodo en concreto

Se va a utilizar la librería *hello.js* explicada en el apartado anterior, donde la inicialización se realiza de la misma manera que como se hizo con Twitter.

Una vez inicializado nuestro entorno, sólo debemos buscar qué llamadas debemos realizar de todas las posibilidades que nos ofrece la API Graph [12] para obtener los datos deseados.

En concreto, se han utilizado los métodos que parten de *insights*, el cual es un término utilizado en marketing que lo define como puntos que permiten buscar un camino para encontrar la solución a lo que estamos buscando.

7.8. API de Instagram

Para la conexión de la API de Instagram se ha utilizado la librería explicada en el punto anterior, *hello.js*.

La instalación y conexión es igual que la que se realiza para el resto de APIs como Twitter. La única diferencia es el conjunto de métodos, ya que cada uno de ellos es específico de la propia red social.

Los métodos disponibles en la documentación de *developers* [2] nos proporciona información sobre el usuario, sus archivos de media, relaciones etc.

7.9. Char.js

Chart.js es una librería escrita en JavaScript que nos permite dibujar gráficos con el elemento *canvas* de HTML de una manera más visual y agradable.

Es tan sencillo como crear el elemento *canvas* en la plantilla que deseemos y posteriormente, mediante un sencillo código escrito en JavaScript, elegir el tipo de gráfico que queremos plasmas y cuáles son los datos de dicho gráfico. Chart.js nos permite realizar gráficos de barras, lineales, en forma de radar, de porciones etc. así hasta 8 tipos y además son adaptables a la pantalla.

Es de código abierto y podemos encontrar tanto el código como la documentación en GitHub [4].

Capítulo 8 : PRUEBAS

En este capítulo se va a explicar el conjunto de pruebas que se han llevado a cabo a lo largo del desarrollo del proyecto. A pesar de ser un proceso costoso, es un proceso necesario ya que con él podemos descubrir y llegar a eliminar los errores que se hayan podido cometer en la etapa de desarrollo. Existen varios tipos de pruebas pero nosotros nos centramos en las de caja blanca y caja negra, tal y como veremos a continuación.

8.1. Pruebas de caja blanca

Son pruebas de tipo estructural, es decir, evalúan la estructura de control del diseño procedimental de la aplicación, garantizando que todos los flujos funcionen correctamente.

Este tipo de pruebas se han llevado a cabo durante todo el proceso de desarrollo. Cada vez que se añadía una nueva funcionalidad se realizaba una prueba de este tipo al igual que a la finalización del proyecto, con una prueba global del funcionamiento.

8.2. Pruebas de caja negra

Las pruebas de caja negra son pruebas funcionales donde se evalúa el comportamiento del software el cual está definido en requisitos y casos de uso y en la interacción con el usuario.

Se definen a partir de funciones, características e interoperabilidad y se puede ejecutar en cualquier nivel. Con las funciones establecemos “lo que el sistema hace”.

Este tipo de pruebas se ha realizado en base a los casos de uso, comprobando que cuando el usuario introdujera o interactuara con el sistema, la salida fuera la esperada, valorando si se han producido fallos en el proceso, interfaz etc.

Registrarse	
PCN - 001	Registrarse
Versión	1.0
Descripción	El usuario solicita registrarse en la app
Resultado Esperado	El sistema crea un usuario nuevo
Aplicación	Versión final
Valoración	INCORRECTO -> fallo de UI/UX . Botón no visible

Tabla 73: Prueba CU Registrarse

Registrarse (Solucionado)	
PCN - 002	Registrarse (solucionado)
Versión	1.0
Descripción	El usuario solicita registrarse en la app
Resultado Esperado	El sistema crea un usuario nuevo
Aplicación	Versión final
Valoración	CORRECTO

Tabla 74: Prueba CU Registrarse Corregido

Iniciar Sesión	
PCN - 003	Iniciar sesión
Versión	1.0
Descripción	El usuario solicita iniciar sesión
Resultado Esperado	El sistema inicia la sesión del usuario
Aplicación	Versión final
Valoración	CORRECTO

Tabla 75: Prueba CU Iniciar Sesión

Borrar usuario (Admin)	
PCN - 004	Borrar usuario
Versión	1.0
Descripción	El usuario admin desea borrar un usuario
Resultado Esperado	El sistema borrar el usuario especificado
Aplicación	Versión final
Valoración	CORRECTO

Tabla 76: Prueba CU Borrar Usuario

Editar usuario (Admin)	
PCN - 005	Editar usuario
Versión	1.0
Descripción	El usuario admin desea editar un usuario
Resultado Esperado	El sistema edita el usuario especificado
Aplicación	Versión final
Valoración	CORRECTO

Tabla 77: Prueba CU Editar Usuario

Añadir Perfil Google Analytics	
PCN - 006	Añadir perfil Google Analytics
Versión	1.0
Descripción	El usuario desea añadir una cuenta de GA
Resultado Esperado	El sistema asocia la cuenta al usuario
Aplicación	Versión Final
Valoración	CORRECTO

Tabla 78: Prueba CU Añadir Perfil GA

Añadir perfil Facebook	
PCN - 007	Añadir perfil FB
Versión	1.0
Descripción	El usuario desea añadir su perfil de FB
Resultado Esperado	El sistema asocia el perfil de FB al usuario
Aplicación	Versión final
Valoración	CORRECTO

Tabla 79: Prueba CU Añadir perfil Facebook

Añadir Perfil Twitter	
PCN - 008	Añadir perfil Twitter
Versión	1.0
Descripción	El usuario desea añadir su perfil de Twitter
Resultado Esperado	El sistema asocia el perfil de Twitter al usuario
Aplicación	Versión final
Valoración	CORRECTO

Tabla 80: Prueba CU Añadir perfil Twitter

Añadir Perfil Instagram	
PCN - 009	Añadir perfil Instagram
Versión	1.0
Descripción	El usuario desea añadir su perfil de Instagram
Resultado Esperado	El sistema asocia el perfil de Instagram al usuario
Aplicación	Versión final
Valoración	INCORRECTO -> Fallo con sandbox

Tabla 81: Prueba CU Añadir perfil Instagram

Añadir Perfil Instagram (Solucionado)	
PCN - 010	Añadir perfil Instagram (solucionado)
Versión	1.0
Descripción	El usuario desea añadir su perfil de Instagram
Resultado Esperado	El sistema asocia el perfil de Instagram al usuario
Aplicación	Versión final
Valoración	CORRECTO

Tabla 82: Prueba CU Añadir perfil Instagram Corregido

Obtener informe GA	
PCN - 011	Obtener informes de GA
Versión	1.0
Descripción	El usuario desea obtener informes de GA
Resultado Esperado	El sistema devuelve datos de GA
Aplicación	Versión final
Valoración	CORRECTO

Tabla 83: Prueba CU Obtener informe GA

Obtener informe Facebook	
PCN - 012	Obtener informes de Facebook
Versión	1.0
Descripción	El usuario desea obtener informes de Facebook
Resultado Esperado	El sistema devuelve datos de Facebook
Aplicación	Versión final
Valoración	CORRECTO

Tabla 84: Prueba CU Obtener informe Facebook

Obtener informe Twitter	
PCN - 013	Obtener informes de Twitter
Versión	1.0
Descripción	El usuario desea obtener informes de Twitter
Resultado Esperado	El sistema devuelve datos de Twitter
Aplicación	Versión final
Valoración	CORRECTO

Tabla 85: Prueba CU Obtener informe Twitter

Obtener informe Instagram	
PCN - 014	Obtener informes de Instagram
Versión	1.0
Descripción	El usuario desea obtener informes de Instagram
Resultado Esperado	El sistema devuelve datos de Instagram
Aplicación	Versión final
Valoración	CORRECTO

Tabla 86: Prueba CU Obtener informe Instagram

Elegir Widgets que visualizar	
PCN - 015	Elegir widgets que visualizar
Versión	1.0
Descripción	El usuario desea modificar la vista de visualización
Resultado Esperado	El sistema oculta o muestra los widgets
Aplicación	Versión final
Valoración	CORRECTO

Tabla 87: Prueba CU Elegir Widgets que visualizar

Cambiar cuenta Google Analytics	
PCN - 016	Cambiar cuenta de analytics
Versión	1.0
Descripción	El usuario desea cambiar la cuenta de GA para obtener otros datos
Resultado Esperado	El sistema cambia de cuenta y muestra nuevos datos
Aplicación	Versión final
Valoración	CORRECTO

Tabla 88: Prueba CU Cambiar cuenta GA

Cerrar sesión RRSS	
PCN - 017	Cerrar sesión de RRSS
Versión	1.0
Descripción	El usuario desea cerrar la sesión de una red social
Resultado Esperado	El sistema cierra la sesión de la red social especificada
Aplicación	Versión final
Valoración	INCORRECTO -> Fallo de UI/UX. El usuario va a la pantalla de la RRSS en vez de a la de cuentas vinculadas

Tabla 89: Prueba CU Cerrar sesión RRSS

Cerrar sesión RRSS (Solucionado)	
PCN - 018	Cerrar sesión de RRSS (solucionado)
Versión	1.0
Descripción	El usuario desea cerrar la sesión de una red social
Resultado Esperado	El sistema cierra la sesión de la red social especificada
Aplicación	Versión final
Valoración	CORRECTO

Tabla 90: Prueba CU Cerrar Sesión RRSS Solucionado

Cerrar sesión sistema	
PCN - 019	Cerrar sesión del sistema completo
Versión	1.0
Descripción	El usuario desea cerrar la sesión del sistema completo
Resultado Esperado	El sistema cierra la sesión del sistema
Aplicación	Versión final
Valoración	CORRECTO

Tabla 91: Prueba CU cerrar sesión sistema

Capítulo 9 : MANUAL DE USUARIO

Una vez hayamos accedido a la aplicación, la primera pantalla nos muestra el formulario de inicio de sesión y la opción de registro si aún no tenemos cuenta.

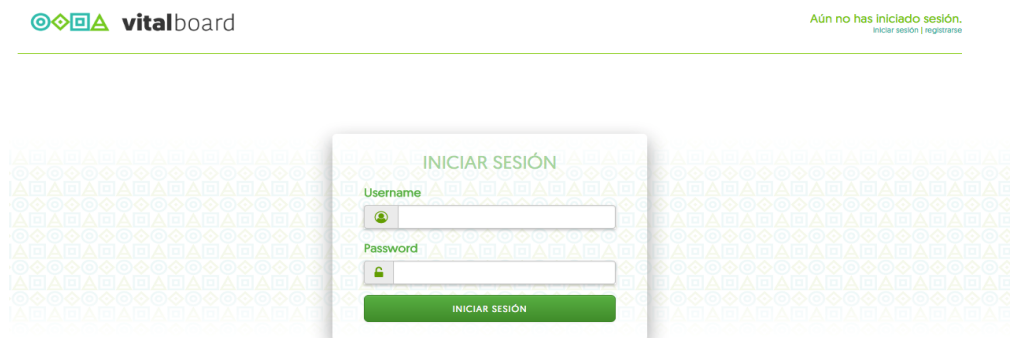


Ilustración 41: Pantalla Inicial herramienta inicio sesión

La primera vez que accedemos, debemos crear una cuenta, haciendo click en la opción de registrarse en la parte superior derecha de la pantalla, donde nos aparecerá un nuevo formulario para introducir nuestros datos y obtener los de acceso.

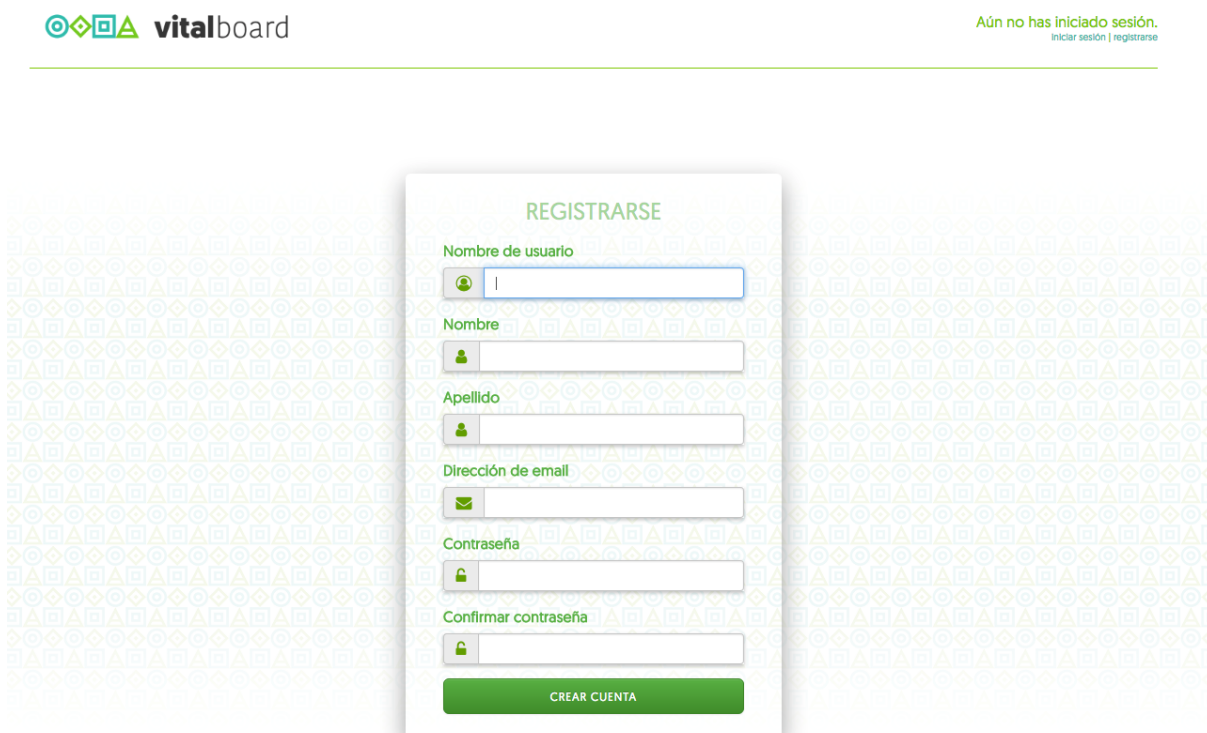


Ilustración 42: Pantalla Registro herramienta

Una vez hayamos iniciado sesión, debemos proceder a añadir nuestras cuentas de las distintas redes sociales para visualizar los datos.

Como se ve en la figura inferior, sólo tenemos que dar a cada uno de los botones de iniciar sesión. La herramienta busca si ya hay una sesión iniciada de dicha red social; si es así, iniciará sesión automáticamente, preguntándonos antes si aceptamos dar los permisos necesarios para la aplicación. Si no, deberemos iniciar sesión en dicha red social para recoger los datos necesarios.

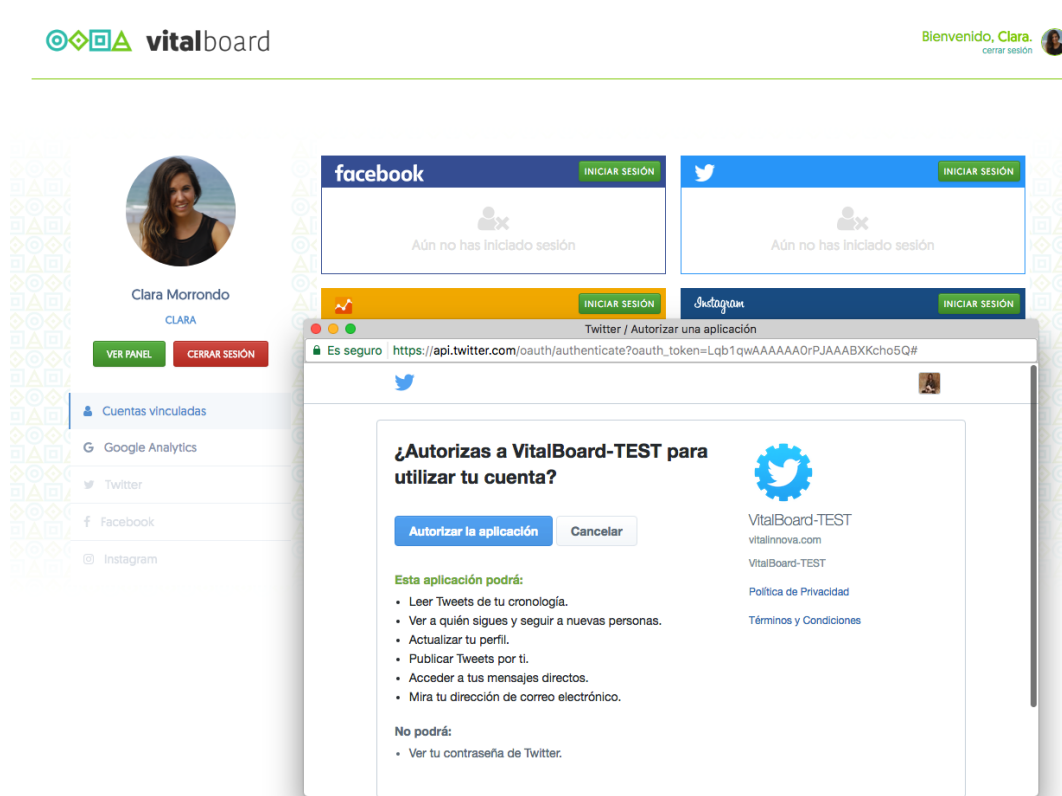


Ilustración 43: Pantalla autorización de permisos

En el caso de Google, es común tener varias sesiones iniciadas simultáneamente donde en dicho caso, la herramienta nos preguntará con qué cuenta deseamos iniciar sesión.

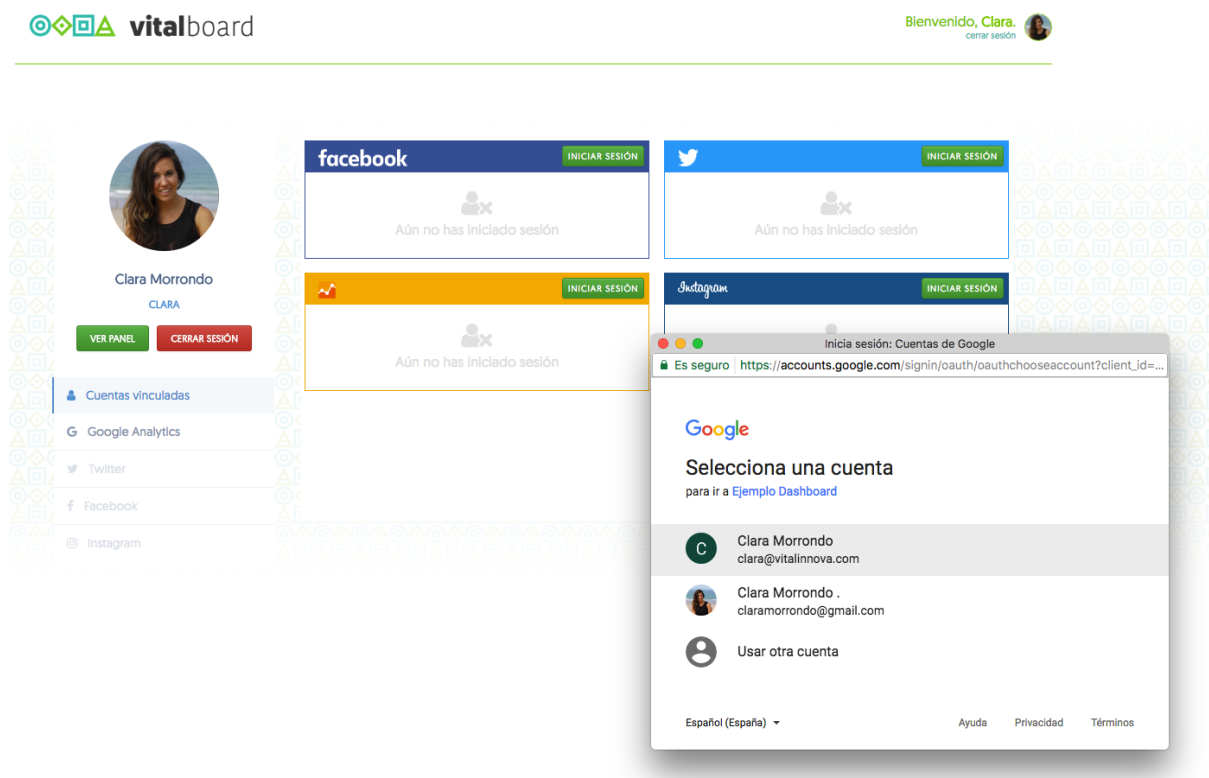


Ilustración 44: Pantalla selección cuenta Google

Una vez hayamos realizado todos los pasos, podemos ir a cada una de las opciones del menú lateral izquierdo para la visualización de los datos.

En el caso de Google Analytics, hay una opción de menú en la parte superior de la pantalla llamada *Ajustes de Visualización*, donde podemos elegir la cuenta, propiedad y vista de Google Analytics que queramos visualizar en dicho momento.

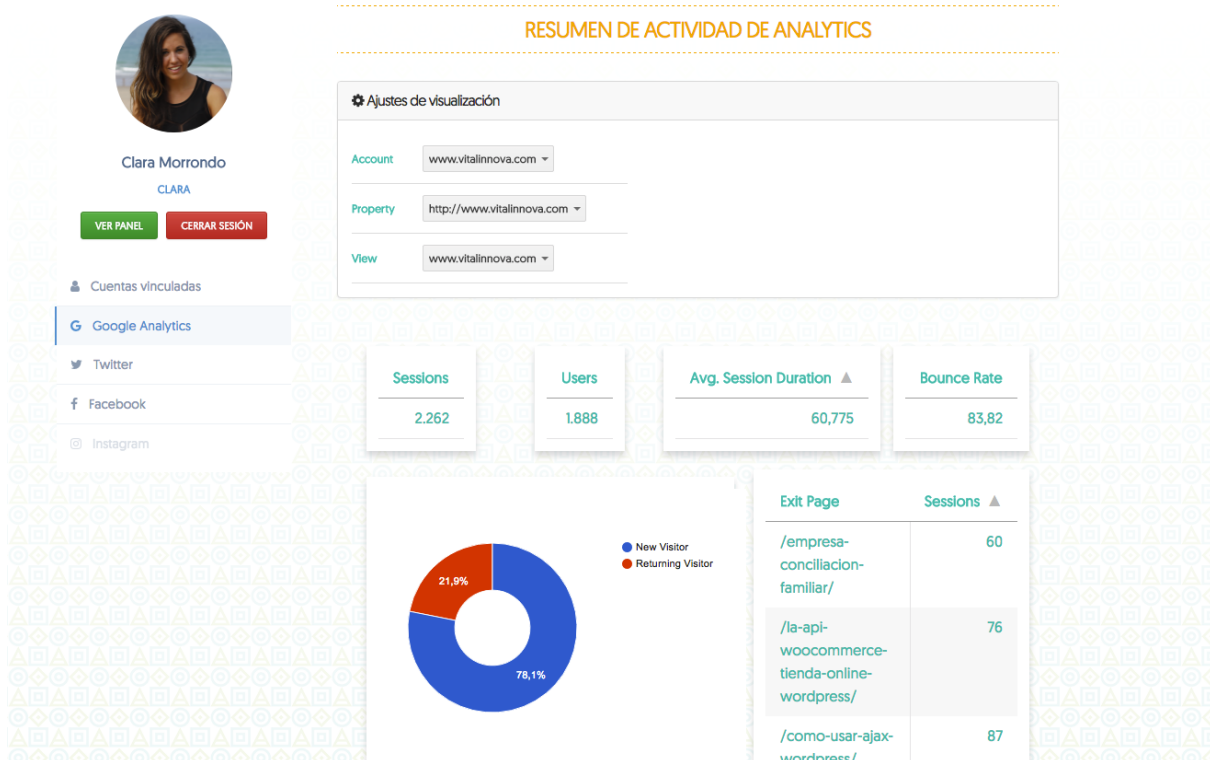


Ilustración 45: Pantalla informes Google Analytics

En Twitter, podemos obtener datos sobre nuestros seguidores, tweets con más interacciones por parte de usuarios, tipo de contenido etc.

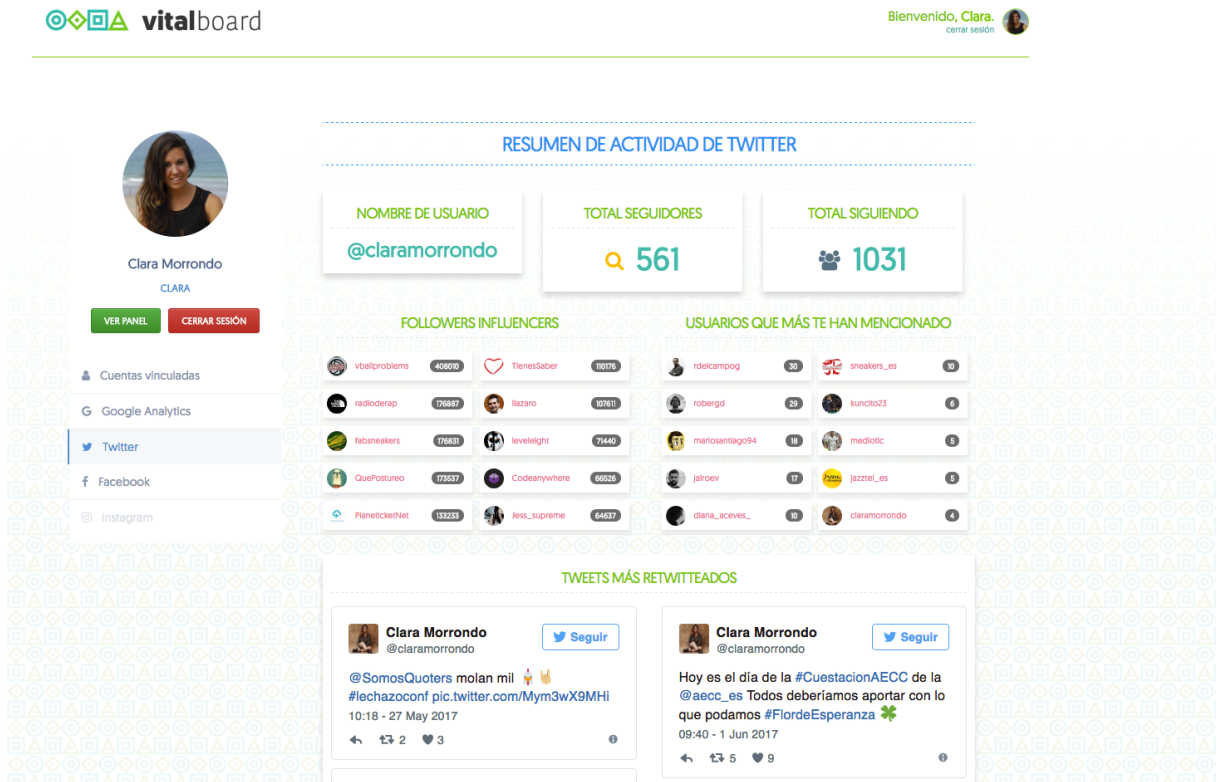


Ilustración 46: Pantalla informes Twiter

En el caso de Facebook, la opción de *Ajustes de Visualización*, nos permite seleccionar qué controles deseamos visualizar y además, nos permite cambiar el periodo de visualización de los datos. Esto es una característica de la API propia de Facebook, la cual no está disponible en las demás.



Ilustración 47: Pantalla informes Facebook

En el caso de Instagram al igual que en Twitter, obtendremos datos sobre nuestros *followers*, tipo de contenido, fotos con más interacciones etc.



Ilustración 48: Pantalla informes Instagram

Por último, existe la posibilidad de descargar cada uno de los informes o bien guardarlos en PDF. Para ello, sólo tenemos que pulsar en botón que se encuentra en la parte superior de cada una de nuestras pantallas de la herramienta, con el texto “*Descargar informe*”.



Ilustración 49: Pantalla descarga de informes

Nos aparecerá una pantalla, con el aspecto que tendrá el fichero y con las opciones disponibles para que el usuario elija lo que desea hacer.

The image shows a side-by-side comparison of a PDF export interface and the content of the report. On the left is a 'Imprimir' (Print) dialog box with the following options:

- Total: 2 páginas
- Buttons: Cancelar, Guardar
- Destino: Guardar como PDF, with a 'Cambiar...' button.
- Páginas: Radio buttons for 'Todo' (selected) and 'p. ej. 1-5, 8, 11-13'.
- Diseño: Vertical (dropdown menu)
- + Más opciones
- Imprimir utilizando el cuadro de diálogo del sistema (⌘P)
- Vista previa de PDF

On the right is a preview of an Instagram activity report titled 'Resumen de actividad de Instagram' (Instagram Activity Summary) for the user 'VitalBland' on 2017-6-19. The report includes:

- A 'DESCARGAR INFORME' (Download Report) button.
- Summary statistics:

SEGUIDORES EN TOTAL	SIGUIENDO EN TOTAL	ME GUSTA RECIBIDOS	COMENTARIOS RECIBIDOS
1892	1724	2678	60
- Additional statistics:

TOTAL PUBLICACIONES	TOTAL PARTICIPACIONES	MEDIA POR SEGUIDOR	MEDIA POR PUBLICACIÓN
402	2738	1.59	6.81
- Section: 'LOS HASHTAGS MÁS USADOS' (Most Used Hashtags) with tags: #picoftheday, #photooftheday, #love, #like, #summer.
- Section: 'TOP 3 FOTOS' (Top 3 Photos) showing a photo of a woman sitting on a bench in front of a blue door, with the URL: <https://www.instagram.com/p/B5gSNgqB8Ro/>.
- Section: 'TIPOS DE PUBLICACIÓN' (Post Types).

Ilustración 50: Vista previa informe exportado

Capítulo 10 : MANUAL DE INSTALACIÓN

Lo primero que tenemos que hacer es tener instalado en nuestro equipo Python para poder realizar la instalación de Django.

La realización de Python en Windows y/o Mac OS X, es tan sencilla como ir a la página oficial de Python, descargarnos el instalador e ir siguiendo los pasos.

En Linux, primero comprobaremos si tenemos instalado ya Python con `$ python3 --version`.

Si no está instalado, ejecutamos el siguiente comando:

```
sudo apt-get install python3.4
```

Una vez instalado Python, podemos proceder a la instalación de Django. Para la instalación, previamente se creó un entorno virtual, el cual es una herramienta que permite aislar nuestros proyectos realizados con Django/Python permitiendo mantener nuestro entorno de desarrollo más ordenado.

```
~/claramorrondo$ python3 -m venv myvenv
```

Una vez creado nuestro entorno virtual, debemos activarlo con `source myvenv/bin/activate` donde aparecerá el prefijo (myvenv) en la terminal, instalamos Django mediante pip.

```
(myvenv) ~$ pip install django==1.8
Downloading/unpacking django==1.8
Installing collected packages: django
Successfully installed django
Cleaning up...
```

Para la creación de nuestro proyecto, ejecutamos el siguiente comando, el cual crea en el directorio actual, un nuevo proyecto con los ficheros predeterminados de un proyecto Django.

```
(myvenv) ~/claramorrondo$ django-admin startproject vitalboard .
```

Para la conexión de las APIs, debemos obtener un ID que asocie dicha red social con nuestra aplicación.

Comenzando por el *clientId* de Google Analytics, debemos acceder a la consola de desarrolladores [8] y añadir nuestro proyecto.

Una vez añadido, accedemos al apartado de credenciales y crearemos una nueva credencial, es decir, solicitamos la creación de un ID para nuestra página, donde obtendremos el identificador buscado que podremos utilizar en nuestra herramienta.

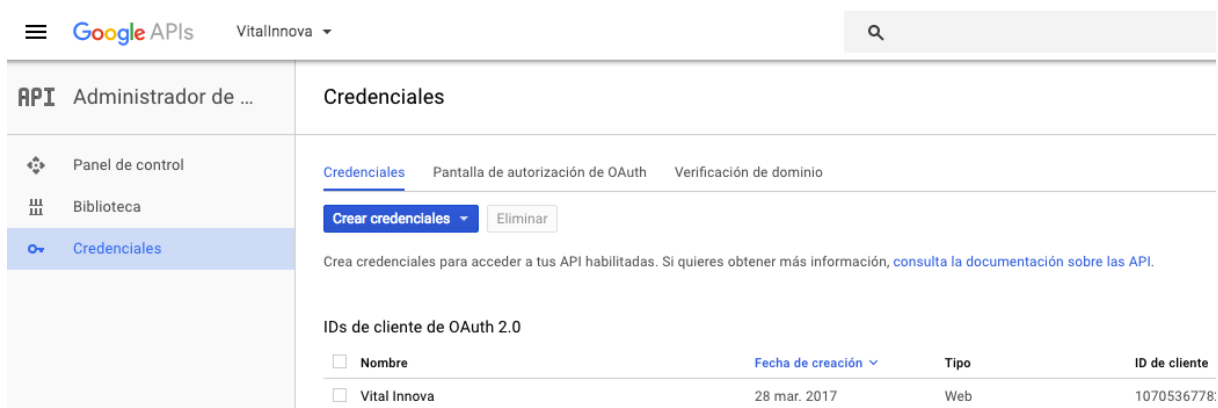


Ilustración 51: Vista credenciales Google APIs

Para la obtención del ID de Facebook, el proceso es similar al comentado anteriormente. Accedemos a [7] y creamos una nueva aplicación. En el panel de la nueva aplicación creada, podemos ver el identificador de la aplicación.

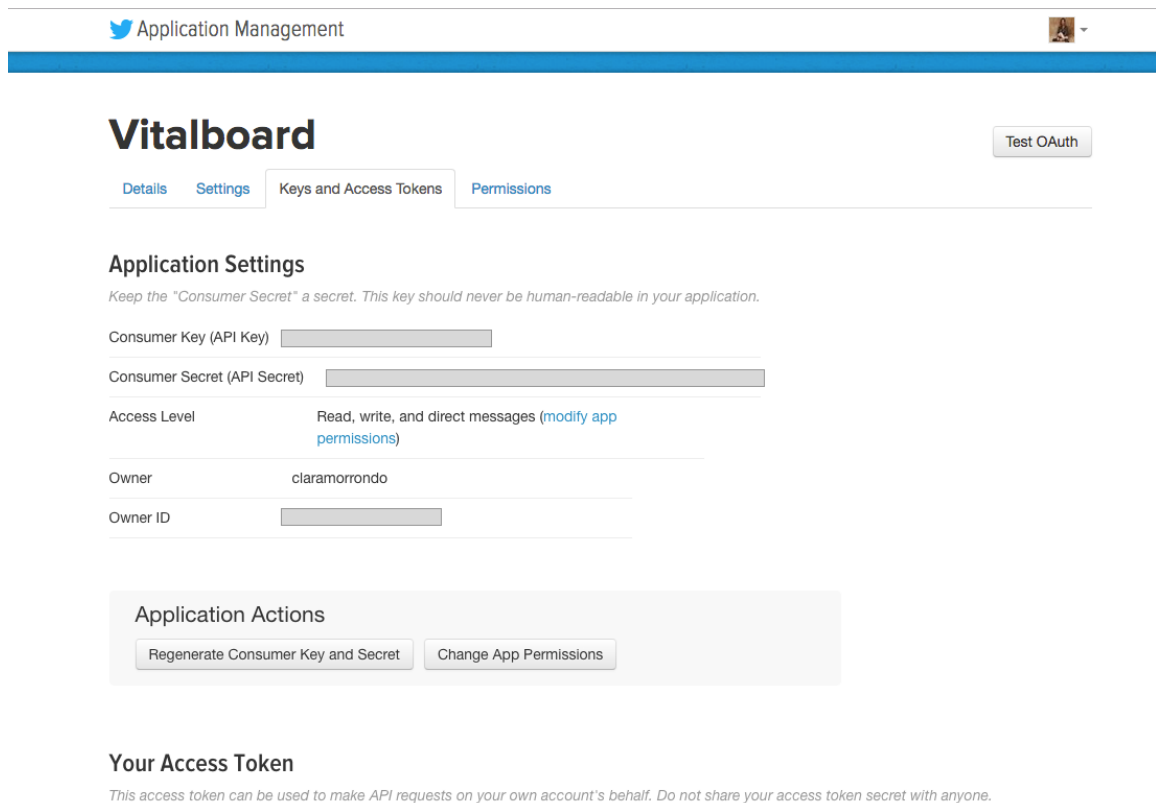


Ilustración 52: Vista credenciales Facebook

Posteriormente, debemos dar de alta la aplicación en [3] introduciendo el identificador que acabamos de obtener para así poder llamar a la librería y que ésta, junto con el *clientID* haga las peticiones por nosotros.

En Twitter, el proceso es similar. Debemos acceder a [29] y crear una nueva aplicación, donde debemos indicar el nombre de la aplicación, la dirección etc.

Una vez realizado esto, obtenemos las claves y tokens asociados a nuestra nueva aplicación. En nuestro caso, el identificador que necesitamos es el *Access Token*.



The screenshot shows the Twitter Application Management interface for an application named 'Vitalboard'. The page has a blue header with the Twitter logo and 'Application Management' text. Below the header, the application name 'Vitalboard' is displayed in large bold letters, with a 'Test OAuth' button to its right. A navigation bar contains tabs for 'Details', 'Settings', 'Keys and Access Tokens', and 'Permissions'. The 'Settings' tab is active, showing 'Application Settings'. A warning message states: 'Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.' Below this, there are input fields for 'Consumer Key (API Key)' and 'Consumer Secret (API Secret)'. The 'Access Level' is set to 'Read, write, and direct messages (modify app permissions)'. The 'Owner' is listed as 'claramorrondo' and the 'Owner ID' is shown in a greyed-out field. An 'Application Actions' section contains two buttons: 'Regenerate Consumer Key and Secret' and 'Change App Permissions'. At the bottom, the 'Your Access Token' section is visible with a warning: 'This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone.'

Ilustración 53: Vista credenciales Twitter

Por último, para obtener el id de Instagram, debemos acceder a la web [18] y acceder a *Manage Clients*. Ahí, registraremos un nuevo cliente introduciendo datos sobre nuestra aplicación y una vez registrado, éste nos devolverá nuestro app ID.

Instagram Sandbox Invites Manage Clients claramorrondo

Search Documentation

- Overview >
- Authentication >
- Login Permissions >
- Permissions Review >
- Sandbox Mode >
- Secure Requests >
- Endpoints >
- Rate Limits >
- Subscriptions >
- Embedding >
- Mobile Sharing >
- Libraries >
- Support >
- Changelog >
- Platform Policy >

Register new Client ID

Details Security

Application Name:

*Do not use **Instagram, IG, insta** or **gram** in your app name. Make sure to adhere to the [API Terms of Use and Brand Guidelines](#).*

Description:

Company Name:

Website URL:

Valid redirect URIs:

The `redirect_uri` specifies where we redirect users after they have chosen whether or not to authenticate your application.

Privacy Policy URL:

Contact email:

An email that Instagram can use to get in touch with you. Please specify a valid email address to be notified of

Ilustración 54: Vista credenciales Instagram

Tanto el app ID de Twitter como de Instagram debemos añadirlos a nuestra librería hello.js tal y como hicimos con Facebook.

Capítulo 11 : MANUAL DE ADMINISTRADOR

Como ya se ha explicado, Django trae una sección de administración del sitio.

Para acceder a dicho perfil, deberemos tener un *superusuario*, el cual crearemos desde una consola introduciendo los siguientes comandos.

```
(myvenv) ~/claramorrondo$ python manage.py createsuperuser
Username: admin
Email address: admin@admin.com
Password:
Password (again):
Superuser created successfully.
```

Una vez realizado este paso, accedemos al sitio en un navegador web, poniendo la url de nuestro sitio seguido de */admin/*.

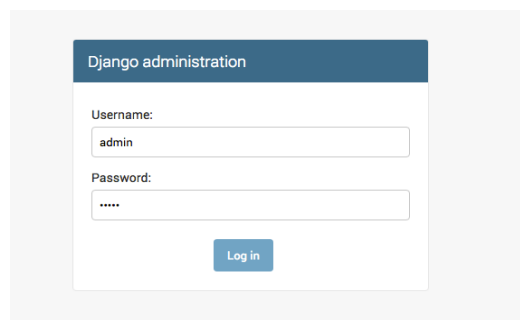


Ilustración 55: Login Django administrador

Dentro del panel, podemos obtener un listado de todos los usuarios registrados en la aplicación, visualizar sus datos, registros, y realizar modificaciones de sus datos, así como borrar dicho usuario. Es importante saber que desde el perfil de administrador, podemos ver y cambiar los permisos de cada uno de los usuarios.

USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
Ana				○
Litzeidy	litzeidy@gmail.com	Litzeidy	Carrasco	○
admin	admin@admin.com			●
clamorr				○
clara	claramorrondo@gmail.com	Clara	Morrondo	○
jairo	jairo@jairovelasco.com	Jairo	Velasco	○
ruben.delcampo	yo@rubendelcampo.es	Rubén	del Campo	○

Ilustración 56: Listado de usuarios desde perfil admin

Capítulo 12 : CONCLUSIONES

Después de la finalización del proyecto, puedo concluir con que se han cumplido los objetivos presentados inicialmente.

Se ha desarrollado una web-app con las funcionalidades presentadas, que conecta con una serie de redes sociales y sobre las que se recogen datos y se desarrollan informes que se pueden exportar. A esto, hay que sumarle una interfaz agradable y fácil de usar para el usuario tras haber realizado un estudio de usabilidad y haber realizado la maquetación de tal forma que permita que la aplicación se visualice perfectamente en cualquier dispositivo.

Además, se han aprendido unos conceptos y tecnologías desconocidas, como pueden ser la utilización de Django, las llamadas a APIs y su filtrado de datos, la exportación de vistas de páginas webs a ficheros etc.

Por último, se ha podido descubrir cuál es el proceso de creación de un proyecto nuevo, pasando por todas las fases y tomando la posición de diferentes roles.

12.1. Líneas futuras

A pesar de haber cumplido con los objetivos, hay una serie de mejoras que se pueden desarrollar para mejorar y/o añadir nuevas funcionalidades útiles para la herramienta:

- Almacenado de los datos de las APIs en base de datos para mejorar el rendimiento de la aplicación sin tener que realizar todas las llamadas cada vez que el usuario requiera dicha información.
- Realizar la exportación de los informes de una manera más visual y gráfica.
- Permitir al usuario que añada varias cuentas de determinadas redes sociales para la visualización conjunta de informes.
- Realizar un histórico de los datos recogidos de las diferentes redes sociales. Debido a que determinadas APIs sólo permiten la visualización de los últimos 20 post, sería recomendable ir almacenando información a lo largo del tiempo para crear históricos.
- Realizar publicaciones en las redes desde la propia herramienta, facilitando y ahorrando trabajo y tiempo al usuario
- Recuperación de contraseña si el usuario olvida cuáles son sus datos de acceso

BIBLIOGRAFÍA

[1] A Javascript RESTFUL API library for connecting with OAuth2 services, such as Google+ API, Facebook Graph and Windows Live Connect [En Línea] Enlace: <https://github.com/MrSwitch/hello.js>

[2] API Endpoints - Instagram Developer Documentation [En Línea] Enlace: <https://www.instagram.com/developer/endpoints/>

[3] Auth-Server Herokuapp [En Línea] Enlace: <https://auth-server.herokuapp.com/#-auth-server>

[4] Chart.js | Open source HTML5 Charts for your website [En Línea] Enlace: <http://www.chartjs.org/>

[5] CONCEPTOS BÁSICOS DE INGENIERÍA DE SOFTWARE [En Línea] Enlace: <http://ingenieriasoftware2014udec.blogspot.com.es/>

[6] De la ruina al éxito en un año: la increíble historia de la 'startup' española Ducksboard [En Línea] Enlace: <http://www.elmundo.es/tecnologia/2014/10/17/5440c5d7ca4741904a8b4572.html>

[7] Django Edge Documentation [En Línea] Enlace: <https://django-edge.readthedocs.io/en/latest/>

[8] Facebook para desarrolladores [En Línea] Enlace: <https://developers.facebook.com/>

[9] Google API Console [En Líneas] Enlace: <https://console.developers.google.com>,

[10] Google Developers [En Línea] Enlace: <https://developers.google.com/analytics/?hl=es-419>

[11] Google Developers [En Línea] Enlace: <https://developers.google.com/analytics/devguides/reporting/embed/v1/component-reference?hl=es-419>

[12] Graph API Reference - Documentation - Facebook for Developers [En Línea] Enlace: <https://developers.facebook.com/docs/graph-api/reference/v2.9/>

[13] gulp.js [En Línea] Enlace: <http://gulpjs.com/>

[14] Gulp.js, una herramienta para automatizar las tareas de desarrollo web [En Línea] Enlace: <http://www.vitalinnova.com/automatizacion-tareas-frontend-gulp-js/>

[15] hello.js - JavaScript API for OAuth2 authentication and RESTful services [En Línea] Enlace: <https://adodson.com/hello.js/#quick-start>

[16] Hootsuite - Su plataforma para interactuar, escuchar y compartir en redes [En Línea] Enlace: <https://hootsuite.com/es/>

- [17] Instagram Developer Documentation [En Línea] Enlace: <https://www.instagram.com/developer/>
- [18] Instagram Developer Manage Opciones [En Línea] Enlace: <https://www.instagram.com/developer/register/>
- [19] JavaScript | MDN [En Línea] Enlace: <https://developer.mozilla.org/es/docs/Web/JavaScript>
- [20] jQuery [En Línea] Enlace: <https://jquery.com/>
- [21] Las 5 mejores herramientas de gestión de redes sociales [En Línea] Enlace: <https://www.agorapulse.com/es/blog/mejores-herramientas-gestion-redes-sociales>
- [22] Maestros del Web [En Línea] Enlace: <http://www.maestrosdelweb.com/images/2012/04/esquema-mtv.png>
- [23] ¿Qué es Django? · Django Girls Tutorial [En Línea] Enlace: <https://tutorial.djangogirls.org/es/django/>
- [24] ¿Qué es Marketing Analytics y por qué deberías aplicarlo?[En Línea] Enlace: <http://tristanlosegui.com/2015/03/25/que-es-marketing-analytics-y-por-que-deberias-aplicarlo/>
- [25] REST APIs --- Twitter Developers [En Línea] Enlace: <https://dev.twitter.com/rest/public>
- [26] RUP (Rational Unified Process) Proceso Unificado Racional [En Línea] Enlace: <http://proceso-unificado-racional.blogspot.com.es/>
- [27] Software de gestión de redes sociales | Sprout Social [En Línea] Enlace: <https://es.sproutsocial.com/>
- [28] Software Project Management Fifth Edition. Autor: B. Hughes - M. Cotterell . Ed. McGraw-Hill
- [29] The Web framework for perfectionists with deadlines | Django. [En Línea] Enlace: <https://www.djangoproject.com/>
- [30] Twitter Apps [En Línea] Enlace: <https://apps.twitter.com/>
- [31] Uso del ORM de Django en tus programas CUI/GUI | Tux Rincon [En Línea] Enlace: <http://www.tuxrincon.com/blog/uso-del-orm-de-django-en-tus-programas-cuigui/>

APÉNDICE A

Contenido del CD

La memoria presente contiene un CD-ROM adjunto con toda la información presentada en dicho documento sobre el proyecto. El CD-ROM está compuesto por los siguientes elementos:

- Memoria.pdf: versión digital del fichero actual de la memoria
- Código fuente de la herramienta: contiene el proyecto con todos los ficheros, bibliotecas y elementos para su correcto funcionamiento.
- Manuales en formato PDF de usuario, administrador y desarrollador (también adjuntos en éste fichero)

MANUAL DE USUARIO

Una vez hayamos accedido a la aplicación, la primera pantalla nos muestra el formulario de inicio de sesión y la opción de registro si aún no tenemos cuenta.

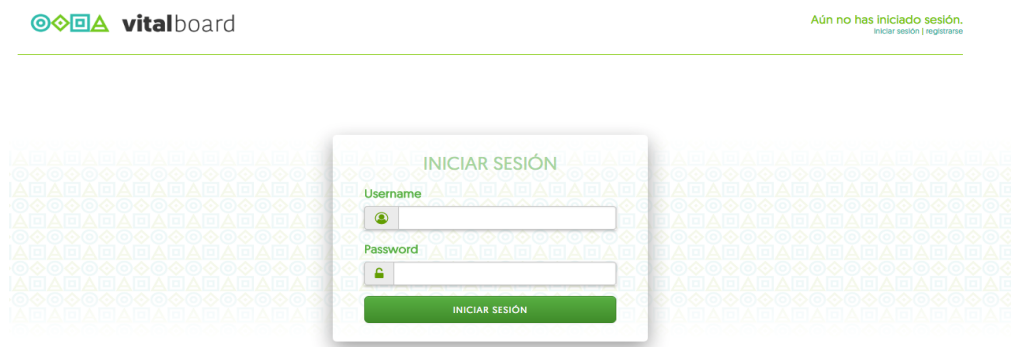


Ilustración 1: Pantalla Inicial herramienta inicio sesión

La primera vez que accedemos, debemos crear una cuenta, haciendo click en la opción de registrarse en la parte superior derecha de la pantalla, donde nos aparecerá un nuevo formulario para introducir nuestros datos y obtener los de acceso.

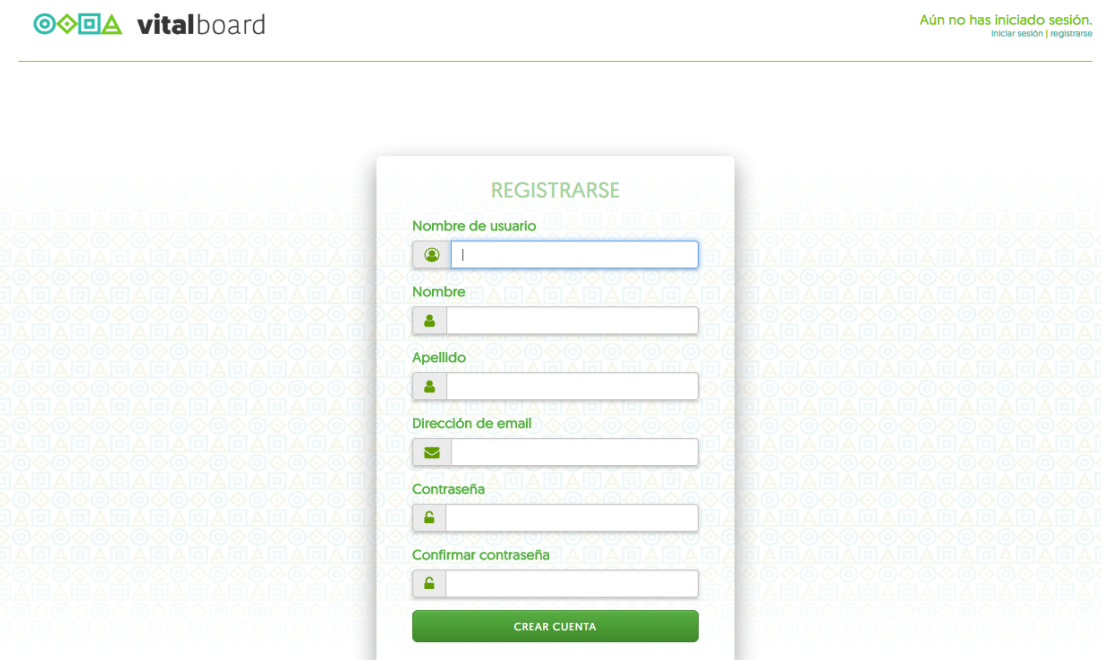


Ilustración 2: Pantalla Registro herramienta

Una vez hayamos iniciado sesión, debemos proceder a añadir nuestras cuentas de las distintas redes sociales para visualizar los datos.

Como se ve en la figura inferior, sólo tenemos que dar a cada uno de los botones de iniciar sesión. La herramienta busca si ya hay una sesión iniciada de dicha red social; si es así, iniciará sesión automáticamente, preguntándonos antes si aceptamos dar los permisos necesarios para la aplicación. Si no, deberemos iniciar sesión en dicha red social para recoger los datos necesarios.

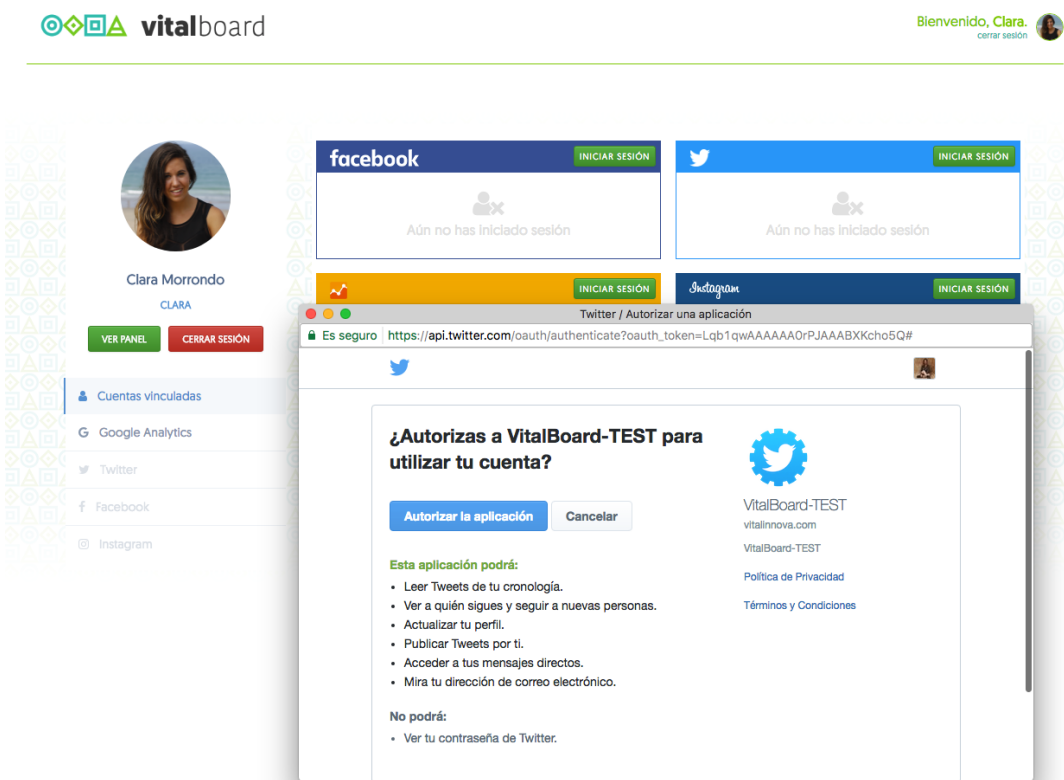


Ilustración 3: Pantalla autorización de permisos

En el caso de Google, es común tener varias sesiones iniciadas simultáneamente donde en dicho caso, la herramienta nos preguntará con qué cuenta deseamos iniciar sesión.

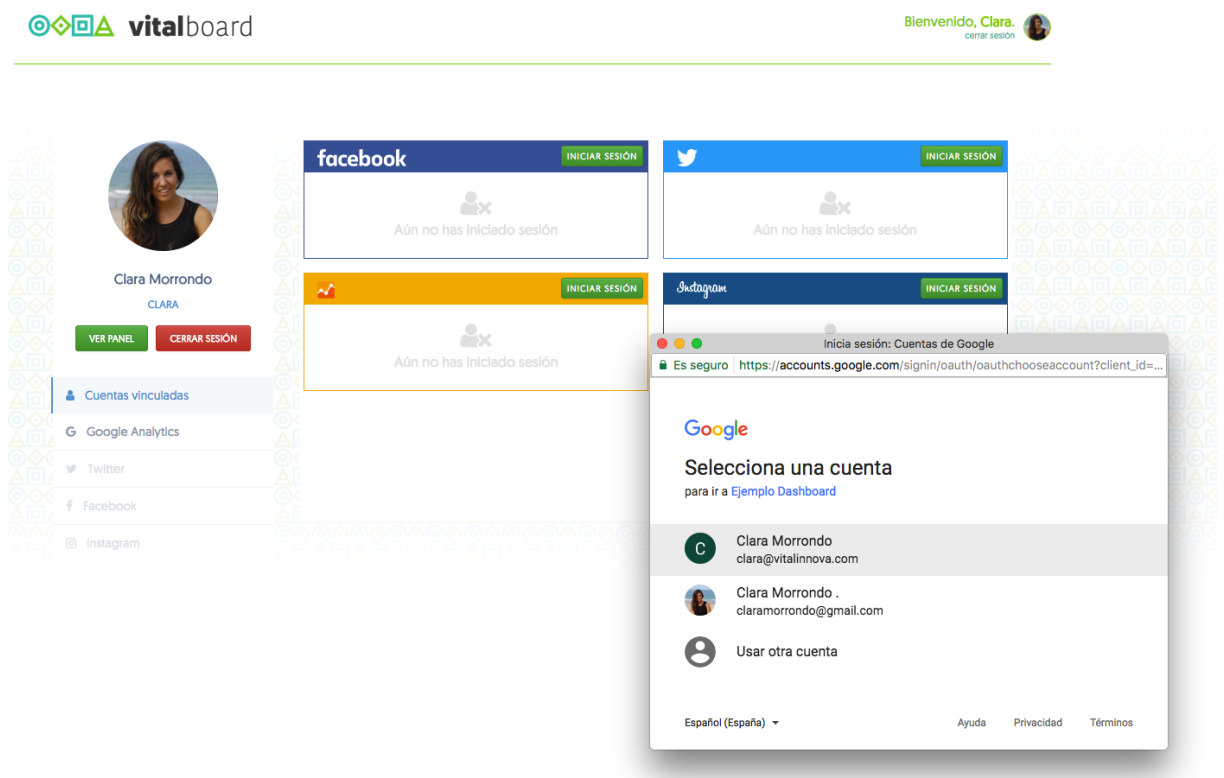



Ilustración 4: Pantalla selección cuenta Google

Una vez hayamos realizado todos los pasos, podemos ir a cada una de las opciones del menú lateral izquierdo para la visualización de los datos.

En el caso de Google Analytics, hay una opción de menú en la parte superior de la pantalla llamada *Ajustes de Visualización*, donde podemos elegir la cuenta, propiedad y vista de Google Analytics que queramos visualizar en dicho momento.



Clara Morrondo
CLARA

VER PANEL CERRAR SESIÓN

Cuentas vinculadas

- Google Analytics
- Twitter
- Facebook
- Instagram

RESUMEN DE ACTIVIDAD DE ANALYTICS

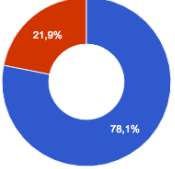
Ajustes de visualización

Account:

Property:

View:

Sessions	Users	Avg. Session Duration ▲	Bounce Rate
2.262	1.888	60,775	83,82



● New Visitor
● Returning Visitor

Exit Page	Sessions ▲
/empresa-conciliacion-familiar/	60
/la-api-woocommerce-tienda-online-wordpress/	76
/como-usar-ajax-wordpress/	87

Ilustración 5: Pantalla informes Google Analytics

En Twitter, podemos obtener datos sobre nuestros seguidores, tweets con más interacciones por parte de usuarios, tipo de contenido etc.

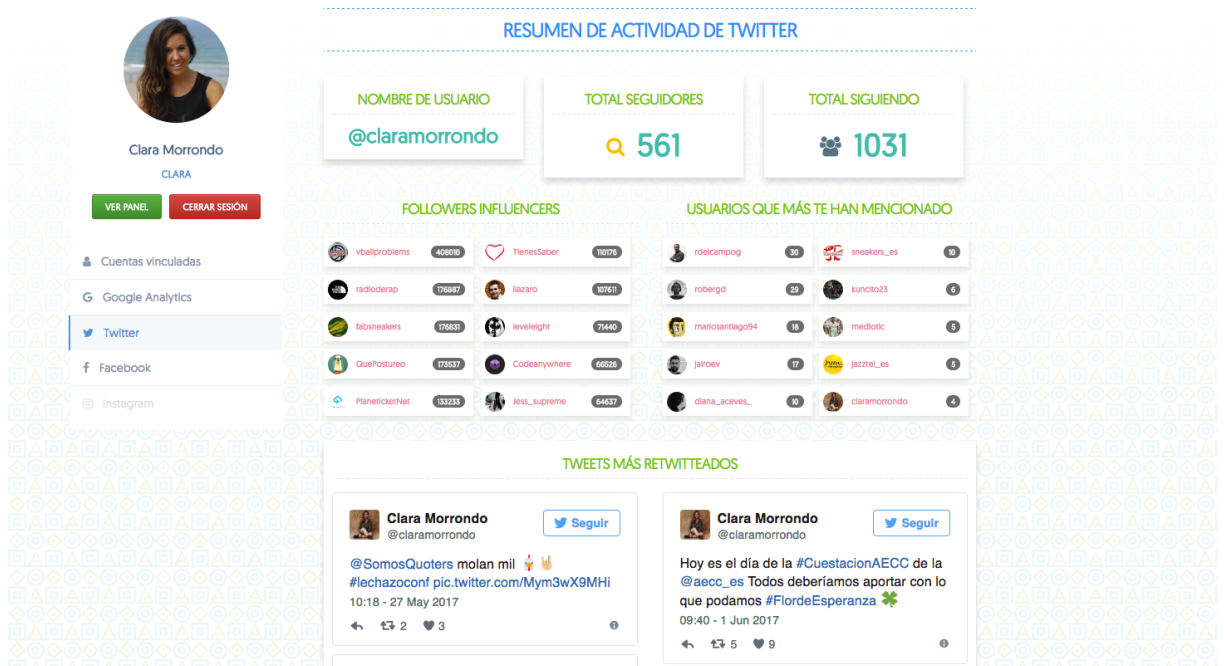


Ilustración 6: Pantalla informes Twitter

En el caso de Facebook, la opción de *Ajustes de Visualización*, nos permite seleccionar qué controles deseamos visualizar y además, nos permite cambiar el periodo de visualización de los datos. Esto es una característica de la API propia de Facebook, la cual no está disponible en las demás.



Clara Morrondo
CLARA

VER PANEL CERRAR SESIÓN

Cuentas vinculadas

Google Analytics

Twitter

Facebook

Instagram

RESUMEN DE ACTIVIDAD DE FACEBOOK

Ajustes de visualización

Impresiones

Clicks a enlaces

Interacciones

Tráfico por países

Tráfico por ciudades

Crecimiento audiencia

Likes/día

Fans

Posts/día

Métricas publicación

Rendimiento del vídeo

Selectores de fecha:

miércoles 7 de junio de 2

sábado 10 de junio de 2

TOTAL NÚMERO DE FANS

471

TOTAL DE IMPRESIONES

19350

TOTAL CLICKS A ENLACES

207

INTERACCIONES TOTALES

271

Crecimiento de la audiencia

Ilustración 7: Pantalla informes Facebook

En el caso de Instagram al igual que en Twitter, obtendremos datos sobre nuestros *followers*, tipo de contenido, fotos con más interacciones etc.



Ilustración 8: Pantalla informes Instagram

Por último, existe la posibilidad de descargar cada uno de los informes o bien guardarlos en PDF. Para ello, sólo tenemos que pulsar en botón que se encuentra en la parte superior de cada una de nuestras pantallas de la herramienta, con el texto “*Descargar informe*”.



Ilustración 9: Pantalla descarga de informes

Nos aparecerá una pantalla, con el aspecto que tendrá el fichero y con las opciones disponibles para que el usuario elija lo que desea hacer.

Imprimir
Total: 2 páginas

Destino Guardar como PDF

Páginas ● Todo
 p. ej. 1-5, 8, 11-13

Diseño Vertical ▼

[+ Más opciones](#)

[Imprimir utilizando el cuadro de diálogo del sistema \(⌘P\)](#)

[Vista previa de PDF](#)

2017-0-19 Vista previa

Resumen de actividad de Instagram


DESCARGAR INFORME

SEGUIDORES EN TOTAL	SIGUIENDO EN TOTAL	ME GUSTA RECIBIDOS	COMENTARIOS RECIBIDOS
👤 1892	🔍 1724	❤️ 2678	💬 60
TOTAL PUBLICACIONES	TOTAL PARTICIPACIONES	MEDIA POR SEGUIDOR	MEDIA POR PUBLICACIÓN
📄 402	↔️ 2738	📊 1.59	📊 6.81

LOS HASHTAGS MÁS USADOS

#picoftheday
#photooftheday
#love
#like
#summer

TOP 3 FOTOS



[https://www.instagram.com/p/BSg5NqdB8Ro/]

TIPOS DE PUBLICACIÓN

http://localhost:3000/ 1/2

Ilustración 10: Vista previa informe exportado

MANUAL DE INSTALACIÓN

Lo primero que tenemos que hacer es tener instalado en nuestro equipo Python para poder realizar la instalación de Django.

La realización de Python en Windows y/o Mac OS X, es tan sencilla como ir a la página oficial de Python, descargarnos el instalador e ir siguiendo los pasos.

En Linux, primero comprobaremos si tenemos instalado ya Python con `$ python3 -version`.

Si no está instalado, ejecutamos el siguiente comando:

```
sudo apt-get install python3.4
```

Una vez instalado Python, podemos proceder a la instalación de Django. Para la instalación, previamente se creó un entorno virtual, el cual es una herramienta que permite aislar nuestros proyectos realizados con Django/Python permitiendo mantener nuestro entorno de desarrollo más ordenado.

```
~/claramorrondo$ python3 -m venv myenv
```

Una vez creado nuestro entorno virtual, debemos activarlo con `source myenv/bin/activate` donde aparecerá el prefijo (myenv) en la terminal, instalamos Django mediante pip.

```
(myenv) ~$ pip install django==1.8
Downloading/unpacking django==1.8
Installing collected packages: django
Successfully installed django
Cleaning up...
```

Para la creación de nuestro proyecto, ejecutamos el siguiente comando, el cual crea en el directorio actual, un nuevo proyecto con los ficheros predeterminados de un proyecto Django.

```
(myenv) ~/claramorrondo$ django-admin startproject vitalboard .
```

Para la conexión de las APIs, debemos obtener un ID que asocie dicha red social con nuestra aplicación.

Comenzando por el *clientid* de Google Analytics, debemos acceder a la consola de desarrolladores [8] y añadir nuestro proyecto.

Una vez añadido, accedemos al apartado de credenciales y crearemos una nueva credencial, es decir, solicitamos la creación de un ID para nuestra página, donde obtendremos el identificador buscado que podremos utilizar en nuestra herramienta.

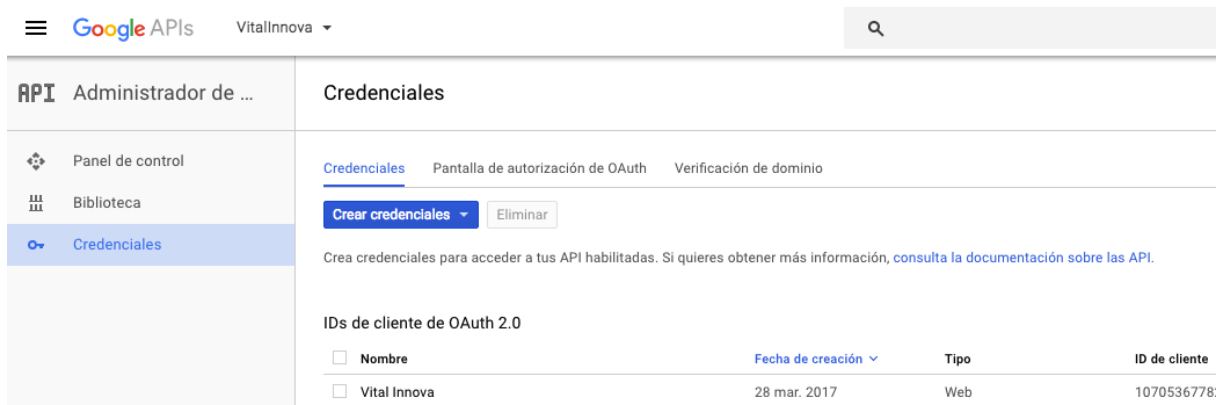


Ilustración 1: Vista credenciales Google APIs

Para la obtención del ID de Facebook, el proceso es similar al comentado anteriormente. Accedemos a [7] y creamos una nueva aplicación. En el panel de la nueva aplicación creada, podemos ver el identificador de la aplicación.



Ilustración 2: Vista credenciales Facebook

Posteriormente, debemos dar de alta la aplicación en [3] introduciendo el identificador que acabamos de obtener para así poder llamar a la librería y que ésta, junto con el *clientID* haga las peticiones por nosotros.

En Twitter, el proceso es similar. Debemos acceder a [29] y crear una nueva aplicación, donde debemos indicar el nombre de la aplicación, la dirección etc.

Una vez realizado esto, obtenemos las claves y tokens asociados a nuestra nueva aplicación. En nuestro caso, el identificador que necesitamos es el *Access Token*.



Vitalboard

[Test OAuth](#)[Details](#) [Settings](#) [Keys and Access Tokens](#) [Permissions](#)

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	<input type="text"/>
Consumer Secret (API Secret)	<input type="text"/>
Access Level	Read, write, and direct messages (modify app permissions)
Owner	claramorrondo
Owner ID	<input type="text"/>

Application Actions

[Regenerate Consumer Key and Secret](#) [Change App Permissions](#)

Your Access Token

This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone.

Ilustración 3: Vista credenciales Twitter

Por último, para obtener el id de Instagram, debemos acceder a la web [18] y acceder a *Manage Clients*. Ahí, registraremos un nuevo cliente introduciendo datos sobre nuestra aplicación y una vez registrado, éste nos devolverá nuestro app ID.

Ilustración 4: Vista credenciales Instagram

Tanto el app ID de Twitter como de Instagram debemos añadirlos a nuestra librería hello.js tal y como hicimos con Facebook.

MANUAL DE ADMINISTRADOR

Como ya se ha explicado, Django trae una sección de administración del sitio.

Para acceder a dicho perfil, deberemos tener un *superusuario*, el cual crearemos desde una consola introduciendo los siguientes comandos.

```
(myvenv) ~/claramorrondo$ python manage.py createsuperuser
Username: admin
Email address: admin@admin.com
Password:
Password (again):
Superuser created successfully.
```

Una vez realizado este paso, accedemos al sitio en un navegador web, poniendo la url de nuestro sitio seguido de */admin/*.

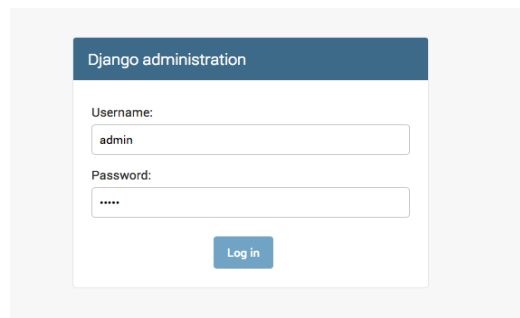


Ilustración 1: Login Django administrador

Dentro del panel, podemos obtener un listado de todos los usuarios registrados en la aplicación, visualizar sus datos, registros, y realizar modificaciones de sus datos, así como borrar dicho usuario. Es importante saber que desde el perfil de administrador, podemos ver y cambiar los permisos de cada uno de los usuarios.

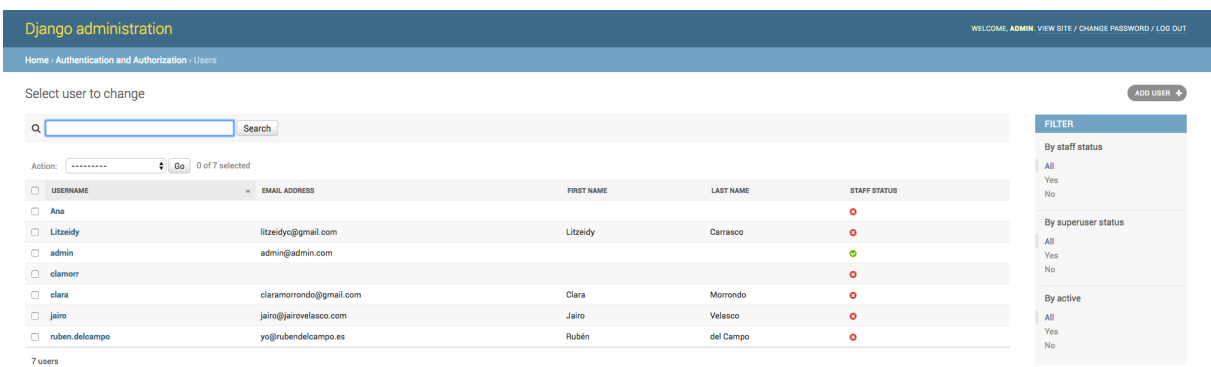


Ilustración 2: Listado de usuarios desde perfil admin