eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Grado en Ingeniería Informática

Ingeniería del Software

Trabajo de Fin de Grado

Analítica web con Google Analytics: el caso de la extensión FRAMEndeley

Autor

Guillermo Arenales Muñoz

Director

Oscar Diaz García

Resumen

FRAMEndeley se concibe como una capa sobre Mendeley mediante una extensión del navegador que nos permite realizar análisis temáticos a pequeña escala . Esto puede ser útil para determinar el alcance de las Preguntas de investigación de manera iterativa o para comparar diferentes artículos de investigación . Sin embargo, ¿ Cómo utilizan los usuarios de internet esta herramienta ? ¿ Qué fallos suelen cometer al usarla ?

Para averiguarlo, en este trabajo se recoge cómo se han introducido una serie de inyecciones de código a la extensión, que nos han permitido conectarla con el servicio de google analytics, que gracias a su recolección de datos de usuarios y dashboards nos ayudar a analizar en detalle cómo los usuarios interactúan con ella y para así finalmente poder sacar conclusiones de si realmente los usuarios usan la extensión como se preveía en un principio o si se deberían de aplicarse cambios ya que o los usuarios no entienden el uso correcto o no está claramente explicado.

Índice

Índice general

Índice de figuras

Índice de tablas

1. CAPÍTULO : Introducción y antecedentes

Capítulo en el que nos centramos en los antecedentes del proyecto. Se aporta una pequeña introducción de lo que se ha trabajado en este proyecto, además hablaremos de cómo se gestó la idea del trabajo y finalizamos hablando de la metodología de trabajo que vamos a llevar durante este proyecto.

1.1. Introducción

El análisis web es el proceso de recopilar, medir y analizar datos sobre el tráfico y el uso de un sitio web con el fin de comprender y optimizar la experiencia del usuario. Esto incluye el monitoreo de las fuentes de tráfico, la navegación del usuario, las conversiones y las métricas de rendimiento. El análisis web se realiza generalmente mediante el uso de herramientas de análisis web, como Google Analytics.

Que es precisamente la herramienta en la que nos centraremos en este proyecto, una plataforma gratuita de análisis web ofrecida por Google que permite a los dueños de sitios web y aplicaciones móviles monitorizar y analizar el tráfico de sus sitios. Con Google Analytics, los usuarios pueden recopilar información detallada sobre el tráfico de sus sitios, incluyendo la fuente de tráfico, la ubicación geográfica de los visitantes, el comportamiento de navegación, las conversiones y mucho más. Esta información puede ser utilizada para mejorar la experiencia del usuario y optimizar la estrategia de marketing en línea.

La aplicación escogida para ser analizada es la extensión de FRAMEndeley , una utilidad de Scoping para Mendeley. Arquitectónicamente, FRAMEndeley se concibe como una capa sobre Mendeley mediante una extensión del navegador En general, FRAMEndeley “simplemente” ayuda con la organización y el seguimiento de extractos y trabajos relacionados. Esta herramienta es la que se va a analizar en este trabajo.

El presente documento se organiza de la siguiente manera: Primero, se presentará el estudio que se ha realizado para establecer la nueva infraestructura del sistema para poder inyectar el código y poder conectar con google de manera eficiente, así como todas las variables que se han tenido en cuenta. Después, se explicará qué tipo de estudio se ha decidido realizar, explicando qué tipos de variables se han decidido visualizar y las razones. Más tarde, veremos que dichas variables seleccionadas nos permiten ver sobre la extensión. Finalmente, recopilaremos todos los datos proporcionados por google analytics y procederemos a realizar las conclusiones necesarias sobre la extensión así como proponer cambios en la extensión con tal de reducir el número de fallos o errores.

1.2. Primeros pasos del proyecto

Mi director, Oscar Diaz, tenía claro que este trabajo debía ser complementario a los contenidos que he trabajado y estudiado durante la carrera y cuando me presentó la idea, desde un primer momento me pareció interesante lo que me proponía, tenía claro que quería que estuviera relacionado con el desarrollo web y aunque hasta ese momento no había trabajado con extensión de Google Chrome me atrajo la idea. Además también me habló de la segunda parte de este proyecto, Google Analytics, herramienta de la que había oído hablar e incluso había leído algún

que otro artículo pero con la cual tampoco había trabajado todavía.

1.3. Metodología de trabajo

En cuanto a la metodología de trabajo que vamos a implementar en este proyecto básicamente realizaremos primeros una fase de investigación de las herramientas y una desarrollado la parte de programación volveremos a una fase más teórica en la que realizaremos un estudio para poder determinar que queremos analizar exactamente una vez conectemos nuestra extensión con los servicios de Google Analytics.

2. CAPÍTULO : Planificación del proyecto

Este capítulo se centra en la planificación del proyecto y todo el trabajo que se realizó antes de empezar con el proyecto. Está dividido en tres subapartados, comenzaremos hablando del alcance que se marcó en un principio y de los objetivos que se esperan alcanzar durante el desarrollo del proyecto. Por otro lado, hablaremos de cómo hemos dividido los diferentes paquetes de trabajo así como las horas que se estiman para su dedicación. Finalmente, terminaremos hablando de los sistemas de información que se han usado durante el trabajo.

2.1 Alcance

El alcance del proyecto es por medio de la extensión de Framendeley crear la estructura necesaria para lograr conectarla con los servicios de Google Analytics y así poder obtener los resultados de cómo sus usuarios la utilizan a través de los “dashboards” de Google Analytics.

Y una vez obtenidos dichos resultados se analizarán para poder generar una reflexión sobre qué cambios podría realizar la extensión para mejorarla.

2.2. Objetivos del proyecto

2.2.1. Objetivo general

El objetivo general de este proyecto es conseguir inyectar código en la extensión de FRAMEndeley de tal manera que nos permita conectarnos con la API de google analytics, y gracias a esta podamos empezar a evaluar cómo se usa dicha extensión, de tal manera que una vez obtenidos los resultados del uso de la extensión podamos presentar una serie de dashboards que nos ayuden a hacernos una idea de cómo se usa la extensión/ que fallos tiene etc. y podamos sacar conclusiones, para añadir cambios a la extensión.

2.2.2. Objetivos específicos

Pero aparte del objetivo general del proyecto también se desean alcanzar ciertos objetivos más específicos

- Aprender de todas las tecnologías usadas durante el proyecto: Como ya he comentado, desde un principio se deseó que este trabajo sirviera para aprender contenido extra al estudiado durante la carrera.

- Plantear un modelo o estructura siguiendo un patrón de diseño que nos ayude a conseguir nuestro objetivo principal de manera eficiente.

- Conectar con google analytics, obtener resultados y sacar conclusiones.

2.3. Estructura de descomposición del trabajo (EDT)

La estructura de descomposición del trabajo es un esquema en el que se dividen todas las actividades a realizar para poder conseguir todos los objetivos y entregables previstos, ayuda a ver cuánto se va a dedicar a cada una de estos tipos de tareas.
Se han estructurado las actividades principales a realizar en paquetes de trabajo.

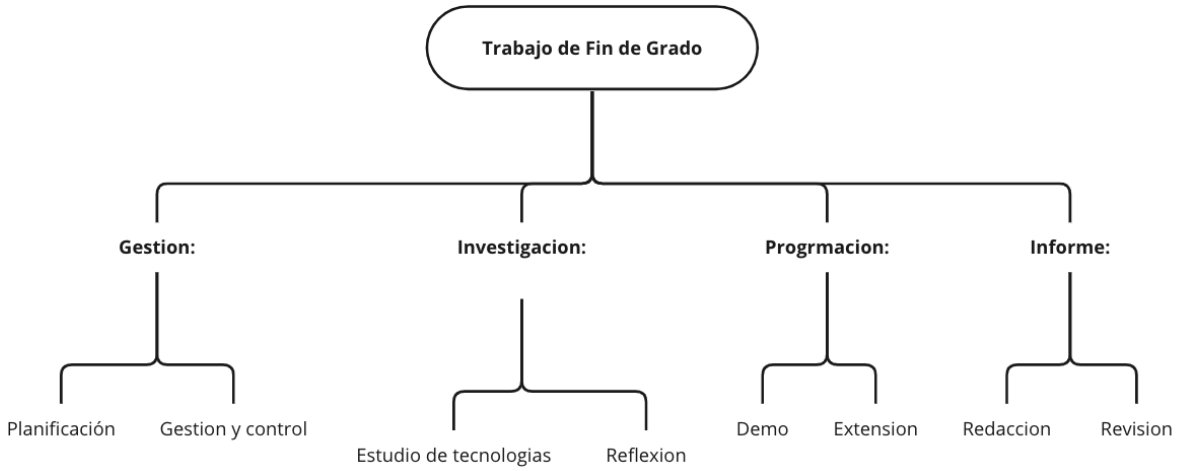


Ilustración 1 EDT del proyecto

Se ha dividido el proyecto en cuatro etapas o paquetes principales: **Gestión (G)**, **Investigación (I)**, **Programación (P)** y por último **Redacción del informe (R)** .

A continuación, se detallarán todas las subactividades que componen las diferentes fases del trabajo.

Comenzaremos con el paquete de la **gestión**, obviamente esta fase fue la primera con la que se empezó a trabajar, un proyecto de estas magnitudes necesita de una buena planificación previa para poder ejecutarlo con éxito, además hay que contar que durante el desarrollo del trabajo se tuvo que compaginar con las asignaturas que he estado cursando durante este cuatrimestre, luego era totalmente necesaria una buena previsión del tiempo que se iba a dedicar a cada tarea.

En este proyecto se iba a trabajar con tecnologías que no conocíamos, o al menos no dominábamos lo suficiente como para un trabajo de este calibre, luego era necesario realizar un trabajo previo para poder comenzar a utilizar todas las herramientas que deseábamos utilizar.

Unido a esto también era completamente necesario conocer bien el ámbito sobre el que íbamos a trabajar para poder analizar y elaborar una estructura que resultara lo más eficiente posible para las características del proyecto.

Por todas estas razones designamos que una de las principales fases del proyecto iba a ser la de **investigación**, en la que se dedicó tiempo tanto a estudiar las tecnologías que íbamos a utilizar tanto la aplicación sobre la que íbamos a aplicarlas.

No se habla solo de lectura de artículos, también se refiere a visualización de vídeos, lectura de libros relacionados con el tema e incluso se reunió con el desarrollador de la extensión, Jeremías Pérez, para entrevistarlo. Esta tarea sirve para evaluar correctamente como se quiere afrontar el proyecto y que tipo de medidas han de tomarse según se avanza.

Por otro lado, ya que gran parte del proyecto iba a suponer a la de implementar código, se decidió que esta fase de **programación** era lo suficientemente relevante como para que tuviera su propio paquete de trabajo.

Por último, al igual que para todos los trabajos de fin de grado es necesaria una buena memoria en el que se recojan toda la información necesaria para entender claramente en lo que se ha estado trabajando, por lo tanto se definió un paquete de **redacción** en los que únicamente se dedica a la redacción de la memoria y a su posterior revisión.

2.4. Tareas por paquete

Habiendo explicado en que consiste las diferentes fases del trabajo, ahora vamos a desgranar las actividades a realizar por cada paquete de trabajo.

(G) Planificación:

La planificación es la tarea en la que se realiza una planificación inicial con un EDT, una definición de paquetes de trabajo y tareas, una estimación de dedicación y un diagrama de Gantt.

(G) Gestión y control:

En esta tarea se ha llevado a cabo un proceso de control a lo largo del proyecto, se han tenido en cuenta las horas reales que se han dedicado a cada tarea y como se ha llevado a cabo el orden de las tareas para una vez acabado compararlo con la planificación inicial que se propuso previo al inicio.

(I) Estudio de las herramientas:

Las dos herramientas principales con las que hemos trabajado han sido por un lado JavaScript, como lenguaje en el que está desarrollada la gran parte de nuestra extensión, y por otro lado está Google Analytics.

Para estudiar JavaScript nos hemos apoyado sobre todo en los video tutoriales que podemos encontrar en plataformas como YouTube que nos han ayudado para aprender sobre (...) que después hemos podido aplicar a nuestro código.

En cambio para familiarizarnos con Google Analytics hemos podido aprender mucho gracias al contenido que proporciona el propio Google, que ofrece una gran cantidad de artículos gratuitos en los que se explica detalladamente cómo funciona esta herramienta.

(I) Análisis:

Este apartado solo está dedicado a, una vez conocida la información, reflexionar acerca de lo leído, visto o escuchado y reflexionar sobre ello. Aquí se analiza cómo vamos a aplicar todo lo que hemos aprendido y de qué manera vamos a usar las tecnologías.

En esta sección se dedicó gran parte del tiempo a diseñar la estructura final de la extensión y a escoger los "dashboards" (paneles de Google Analytics) de manera que nos brindaran la mayor cantidad de información posible.

(P). Extensión:
Paquete de trabajo dedicado a la creación de código necesario para crear la infraestructura del código que vamos a utilizar para así después poder conectarla con Google Analytics y así poder analizarla en profundidad.

(P). Demo:
Paquete de trabajo dedicado para una demostración a pequeña escala de todo lo que queríamos hacer con la extensión. Como se ha dado a entender en anteriores párrafos, había tecnologías con las que no estábamos familiarizados a priori, por lo que se pensó que antes de aplicar todos los cambios deseados a la propia extensión, se trabaja a pequeña escala, de modo que el "salto" a la gran escala no fuera tan pronunciado.

(R). Redacción:

Este paquete de trabajo estará dedicado solamente a escribir. Aquí se harán borradores sobre la información extraída al momento, y más adelante se pasará a limpio.

(R). Revisión:

Este paquete de trabajo se usará para corregir lo redactado en los borradores y en el propio documento en limpio para comprobar que todo esté escrito como es deseado.

Después de haber explicado en qué consistían las tareas de cada paquete de trabajo ahora nos toca estimar cuántas horas dedicaremos a cada una de ellas. La dedicación que se propuso al inicio del proyecto fue la siguiente:

TFG		
	%	Horas
Gestion	15	45
Planificacion	8,33	25
Gestion y Control	6,67	20
Investigacion	25	75
Estudio Tecnologias	16,67	50
Reflexion	8,33	25
Programacion	30	90
Demo	6,67	20
Extension	23,33	70
Informe	30	90
Redaccion	20	60
Revision	10	30
Suma	100	300

Tabla 1 Horas estimadas para cada paquete

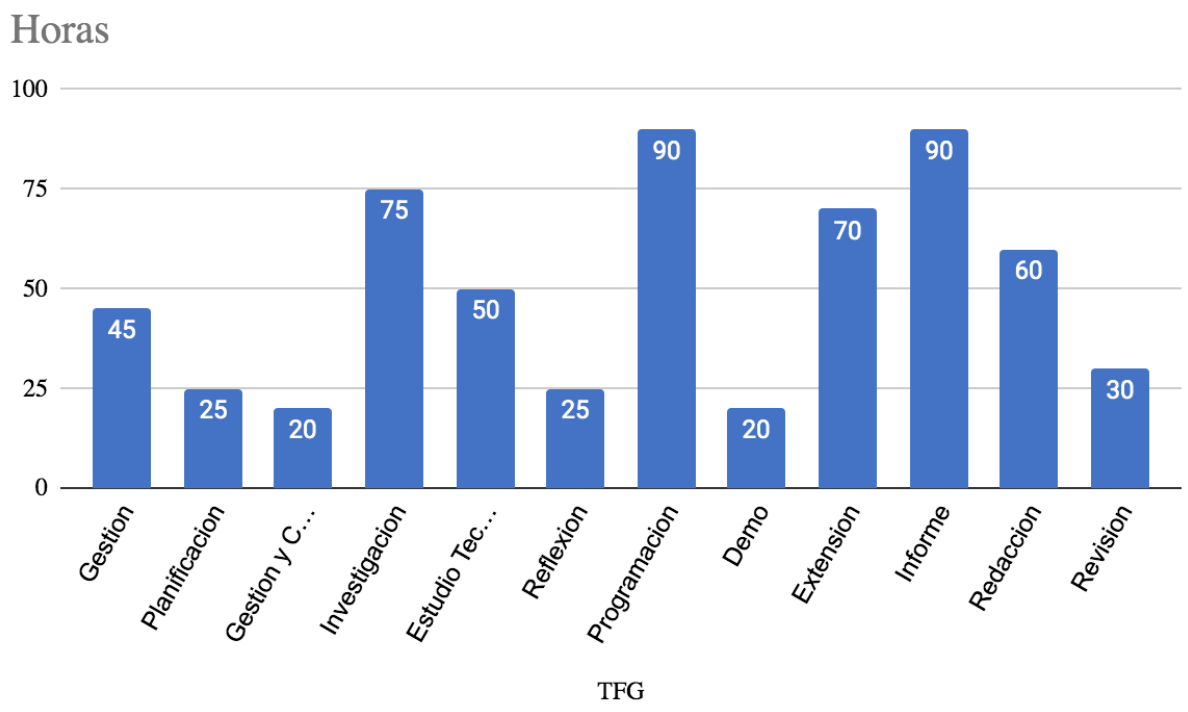


Ilustración 2 Grafico de las horas estimadas

2.5. Diagrama de Gantt

A continuación se presenta el diagrama de Gantt de la planificación de las tareas:

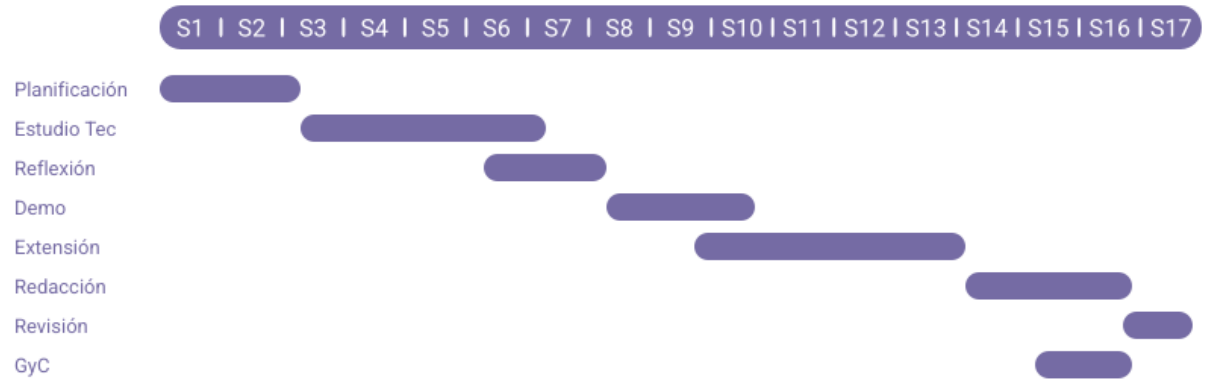


Ilustración 3 Diagrama de Gantt inicial

2.6. Herramientas utilizadas

A lo largo del proyecto se han utilizado varias herramientas, a continuación se presentan las más relevantes:

JavaScript:

JavaScript (abreviado comúnmente JS) es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como objetos, basado, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas y JavaScript del lado del servidor (Server-side JavaScript o SSJS). Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.

Google Analytics:

Google Analytics es una herramienta de analítica web de la empresa Google lanzada el 14 de noviembre de 2005. Ofrece información agrupada del tráfico que llega a los sitios web según la audiencia, la adquisición, el comportamiento y las conversiones que se llevan a cabo en el sitio web. Es una herramienta utilizada en marketing digital.1

Se pueden obtener informes como el seguimiento de usuarios exclusivos, el rendimiento del segmento de usuarios, los resultados de las diferentes campañas de marketing en línea, las sesiones por fuentes de tráfico, tasas de rebote, duración de las sesiones, contenidos visitados, conversiones (para e-commerce), etcétera. Este producto se desarrolló basándose en la compra de Urchin (hasta entonces, la mayor compañía de análisis estadístico de páginas web) por parte de Google.

Gulp:
Gulp es una herramienta, en forma de script en NodeJS, que te ayuda a automatizar tareas comunes en el desarrollo de una aplicación, como pueden ser: mover archivos de una carpeta a otra, eliminarlos, minificar código, sincronizar el navegador cuando modificas tu código, validar sintaxis y un largo etcétera.

Intellij:
IntelliJ IDEA es un entorno de desarrollo integrado (IDE) para lenguajes JVM diseñado para maximizar la productividad del desarrollador. Realiza las tareas rutinarias y repetitivas por usted al proporcionar finalización de código inteligente, análisis de código estático y refactorizaciones, y le permite concentrarse en el lado positivo del desarrollo de software, lo que lo convierte no solo en una experiencia productiva sino también agradable.

NodeJS:

Node.js es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) basado en el lenguaje de programación JavaScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como por ejemplo, servidores web. Fue creado por Ryan Dahl en 2009 y su evolución está apadrinada por la empresa Joyent, que además tiene contratado a Dahl en plantilla.

Node.js es similar en su propósito a Twisted o Tornado de Python, Perl Object Environment de Perl, libevent o libev de C, EventMachine de Ruby, vibe.d de D y Java EE de Java existe Apache MINA, Netty, Akka, Vert.x, Grizzly o Xsocket. Al contrario que la mayoría del código JavaScript, no se ejecuta en un navegador, sino en el servidor. Node.js implementa algunas especificaciones de CommonJS. Node.js incluye un entorno REPL para depuración interactiva.

GitHub:
GitHub es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador. El software que opera GitHub fue escrito en Ruby on Rails. Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. Anteriormente era conocida como

Logical Awesome LLC. El código de los proyectos alojados en GitHub se almacena generalmente de forma pública.

Google Drive:
Google Drive es un servicio de alojamiento y sincronización de archivos desarrollado por Google. Lanzado el 24 de abril de 2012, el servicio permite a sus usuarios almacenar archivos en la nube (en los servidores de Google), sincronizar archivos entre dispositivos y compartir archivos. Cada usuario cuenta con 15 gigabytes (GB) de espacio gratuito para almacenar sus archivos, ampliables mediante diferentes planes de pago. Es accesible a través del sitio web desde computadoras y disponen de aplicaciones para Android e iOS que permiten editar documentos y hojas de cálculo.

2.7. Gestión de Riesgos

Por muy precisos que queramos ser con la planificación también debemos tener en cuenta que a lo largo de este tipo de proyectos, cuya duración oscila entre tres y cuatro meses, se pueden producir riesgos que retrasen o hagan variar la realización de las tareas o que incluso lleven a la cancelación de este. Estos son los posibles riesgos que se valoraron a la hora de realizar la planificación:

Actualización de los requisitos de Google: El mundo de las extensiones avanza diariamente y cada poco tiempo se producen actualizaciones que pueden hasta dejar nuestras aplicaciones inservibles. Especialmente estos últimos meses en los que Google está tratando que todos sus usuarios migren sus extensiones de las versiones a la versión 4, hecho que ha repercutido en este proyecto y de lo cual hablaremos más adelante ya que la extensión está desarrollada para la versión 2.

Por suerte Google ha avisado a todos sus usuarios que las versiones antiguas seguirán disponibles hasta julio del 2023 y se ha preocupado de crear tanto videotutoriales como artículos para poder migrar a las nuevas versiones. Luego la medida que se tomó para mitigar este riesgo fue ir actualizando poco a poco las partes más antiguas para procurar tener los menos problemas posibles con Google.

Pérdida de información: Un tipo de riesgo que afecta a todos los proyectos es el de la pérdida de información, es posible que por algún motivo el dispositivo en donde tenemos guardada información relevante para el desarrollo del trabajo quede inoperativo provocando que perdamos toda esa información tan valiosa.

Sin embargo, hoy en día, afortunadamente, se puede mitigar fácilmente, gracias a herramientas

como GitHub, en caso de que se hicieran avances importantes en la implementación del código rápidamente se procuraba guardar esos cambios en el repositorio personal. Por otro lado, en cuanto a los documentos generados, estos se guardaban rápidamente online gracias a Google Drive por lo que en caso de pérdida siempre existía una copia de seguridad en la nube.

Problemas con las asignaturas del curso: Como ya se ha mencionado durante la realización del TFG he estado cursando varias asignaturas optativas del último año de carrera y podría ocurrir que debido a un examen o a un aumento de la carga de trabajo en dichas asignaturas no se pudiera dedicar el tiempo estimado a las tareas.

Aun así, este riesgo no debería suponer un grave problema ya que al cursar estas asignaturas en la modalidad de evaluación continua, la carga de trabajo se divide equitativamente a lo largo de toda la asignatura y en caso de un aumento en dicha carga se compensa con una bajada en las siguientes semanas, por lo cual el tiempo que no se podría dedicar al TFG se podrá recuperar posteriormente.

2.8. Sistemas de Información

El sistema de información principal será en local, en el dispositivo en el que se desarrolla todo el proyecto, sin embargo, como se ha destacado en el apartado anterior también se hará uso de herramientas como GitHub o Google Drive que actuarán como un respaldo al sistema local guardando copias de seguridad que podrán ser usadas en caso de fallo en el sistema local. En el caso de GitHub incluso se guardan todas las versiones del trabajo por lo que si una versión falla y se ha de volver a una versión anterior la cual funcionaba correctamente, se podría valorar esta opción.

También hay que destacar que a lo largo de todo el desarrollo de la extensión se han leído y estudiado artículos y referencias que han resultado útiles, y que gracias a la propia aplicación de Mendeley se han podido administrar para que en caso de volver a necesitarlos en un futuro estuvieran accesibles.

3. Capítulo: Caso de estudio: FRAMEndeley

En este capítulo se analizará la extensión FRAMEndeley antes de aplicarle ningún cambio, con la intención de saber sobre qué tipo de herramienta estamos trabajando antes de ponernos manos a la obra. Nos fijaremos en su estructura y su funcionamiento analizando el código.

3.1. FRAMEndeley

Enriquece a Mendeley con la funcionalidad de análisis temático. FRAMEndeley amplía el sitio web de Mendeley con soporte para realizar análisis temáticos en pequeño. Esto puede ser útil para determinar el alcance de las Preguntas de investigación de manera iterativa o para comparar diferentes artículos de investigación.

3.2. ¿Y qué es Mendeley?

Mendeley es una aplicación de gestión de referencias y colaboración científica. Se utiliza para organizar y compartir artículos de investigación, tesis, libros y otros documentos relacionados con la investigación académica. Con Mendeley, los usuarios pueden importar automáticamente documentos desde diferentes fuentes, organizarlos en bibliotecas personales, etiquetarlos, resaltar fragmentos y tomar notas. También se pueden compartir bibliotecas públicas o privadas con otros investigadores y colaboradores, y se puede utilizar para generar automáticamente listas de referencias y citas en diferentes formatos de estilo.

Un sistema de gestión de referencias es un software o una aplicación que ayuda a los usuarios a organizar, almacenar y compartir información bibliográfica. Los sistemas de gestión de referencias permiten a los usuarios importar, almacenar y etiquetar información bibliográfica de artículos, libros, tesis, y otros documentos de investigación. También proporcionan herramientas para generar automáticamente listas de referencias y citas en diferentes formatos de estilo, lo que facilita el proceso de escribir trabajos académicos y científicos. Los sistemas de gestión de referencias también permiten compartir bibliotecas con otros investigadores y colaboradores, lo que facilita la colaboración en proyectos de investigación.

3.3. ¿Cómo funciona FRAMEndeley? ¿Para qué sirve?

Mendeley es un software de gestión de referencias desarrollado por Elsevier . Se utiliza para administrar y compartir trabajos de investigación y generar bibliografías para artículos académicos.

Ilustración 4 Página principal de Mendeley

El software puede realizar un seguimiento de los recuentos de lectores, una estadística de lectores que se ha afirmado que predice el impacto de las citas. También está disponible una extracción automática de metadatos de archivos PDF. La plataforma puede integrarse con Microsoft Word , OpenOffice y otras plataformas. La plataforma tiene la intención de cumplir con los derechos de autor al permitir que los usuarios compartan archivos en grupos privados.

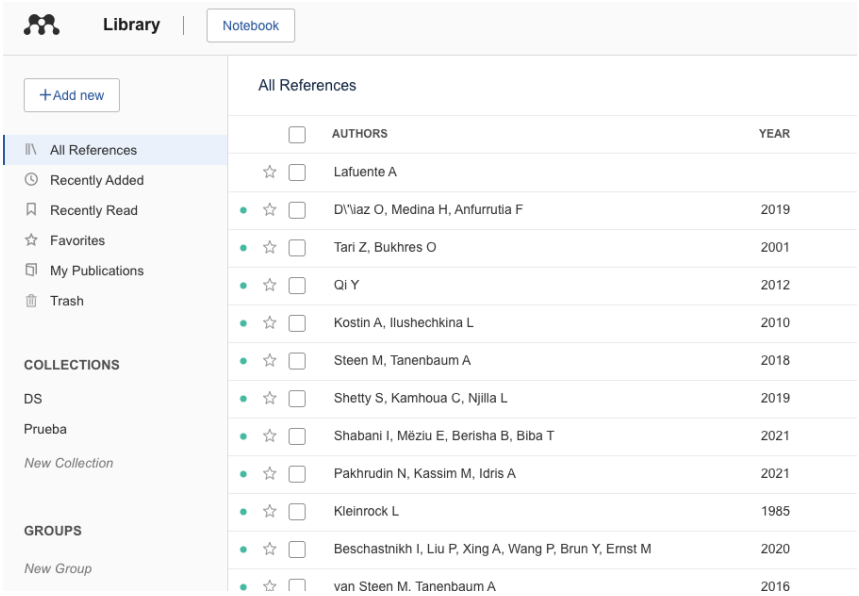


Ilustración 5 Librería de Framendeley

Como ya hemos mencionado anteriormente FRAMEndeley se concibe como una capa sobre Mendeley mediante una extensión del navegador, y las funcionalidades[1] que ofrece la extensión son las siguientes:

Botón FRAMEndeley en la página de la biblioteca de Mendeley:

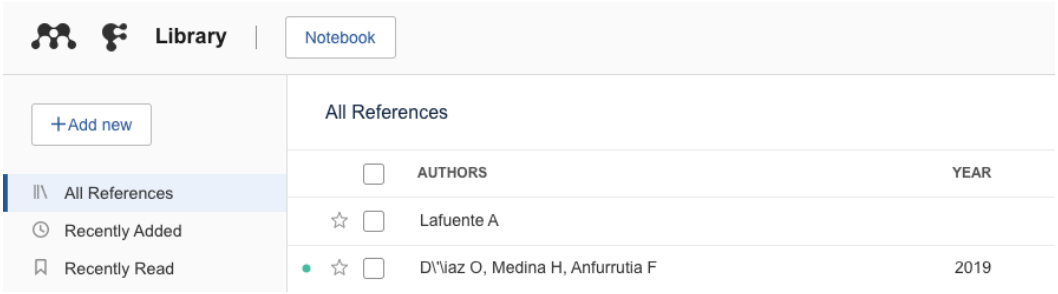


Ilustración 6 Botón Framendeley en la librería

Al hacer clic, aparece el lienzo de alcance:

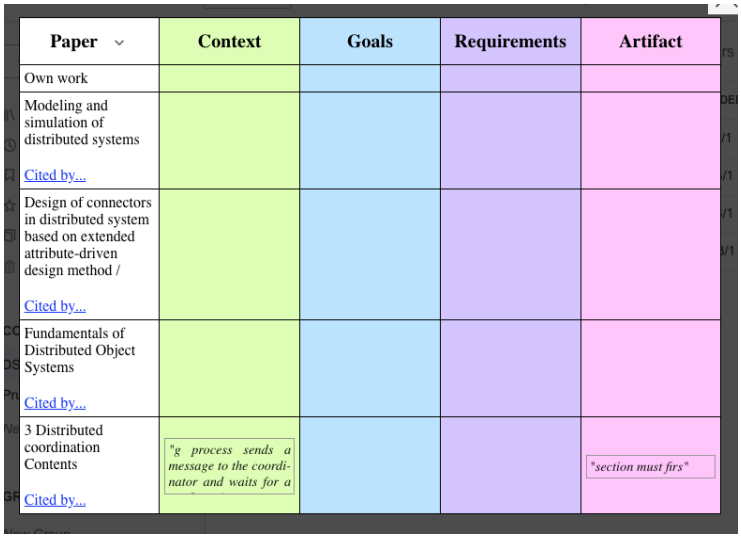


Ilustración 7 Open Canvas de Framendeley

[1] Video en el que se muestran todas las funcionalidades de la extensión: https://www.youtube.com/watch?v=1XuFb_cZIQY&t=1s&ab_channel=OnekinSanSebasti%C3%A1n

Este lienzo contiene la información que hemos ido subrayando en nuestros textos, y esa es precisamente otra de las características de FRAMEndeley, la de subrayar.

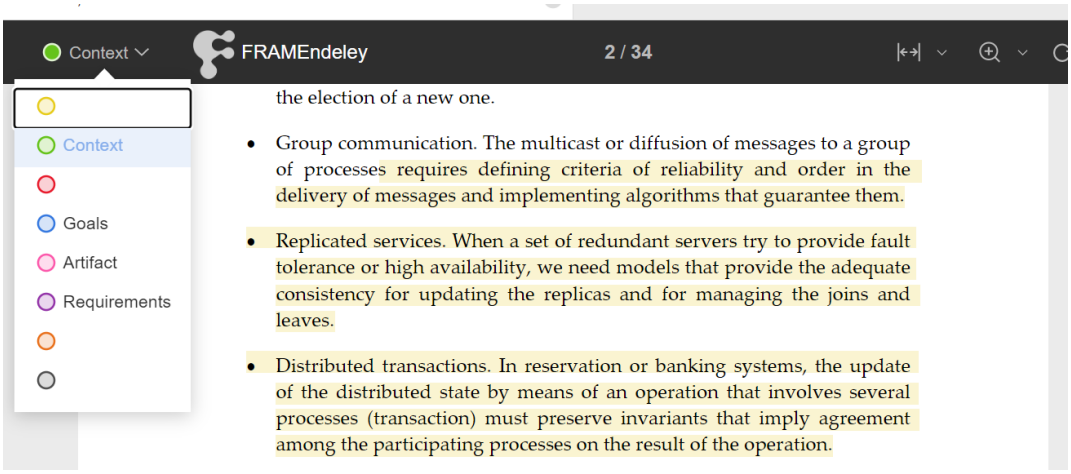


Ilustración 8 Subrayado en FRAMEndeley

Se ofrecen diferentes colores de subrayado al usuario, el usuario elegirá el color dependiendo a que columna de la tabla de la figura quiera rellenar , por ejemplo, si se quisiera resaltar un fragmento del texto ya que refleja los objetivos que se quieren lograr pues se seleccionara el fragmento y se subrayara de azul. A medida que vayamos subrayando contenido en nuestros textos la tabla se ira rellenando.

Por último está el "Article Canvas" una tabla similar a la anterior pero en este caso representa la información exclusiva de un texto en particular.

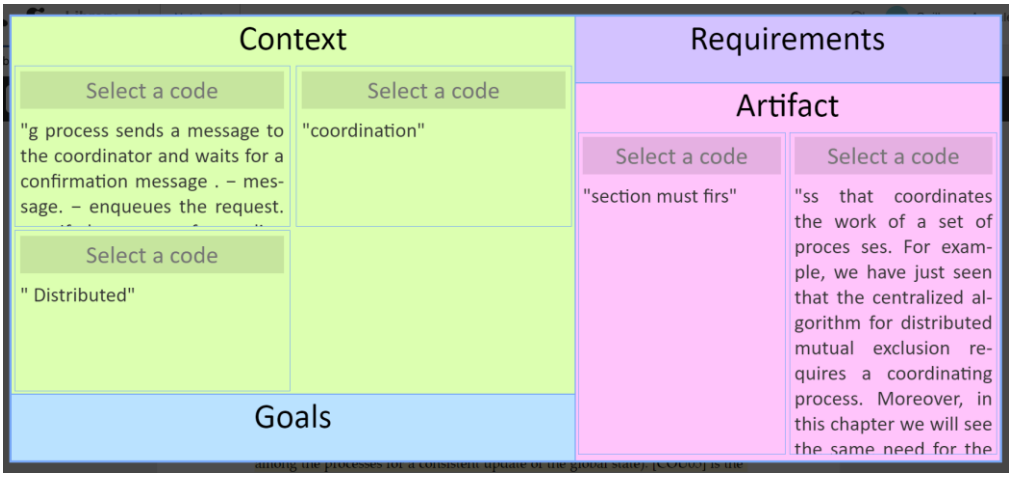


Ilustración 9 Article Canvas

3.4. ¿Cómo funcionan las extensiones?

Podríamos pasar directamente al análisis de la extensión, sin embargo para poder comprender el porqué del tipo de diseño de la extensión y poder analizar en mayor profundidad la estructura de la herramienta es necesario aclarar ciertas características de las extensiones:

3.4.1. Manifest.json:

El manifiesto de una aplicación web es un archivo en formato JSON que informa al navegador sobre la Aplicación Web Progresiva y cómo debe proceder cuando se instala en el escritorio o en el dispositivo móvil del usuario. Un archivo anexo habitual incluye el nombre de la aplicación, los iconos que debe utilizar y la URL que debe abrir cuando se inicie la aplicación.

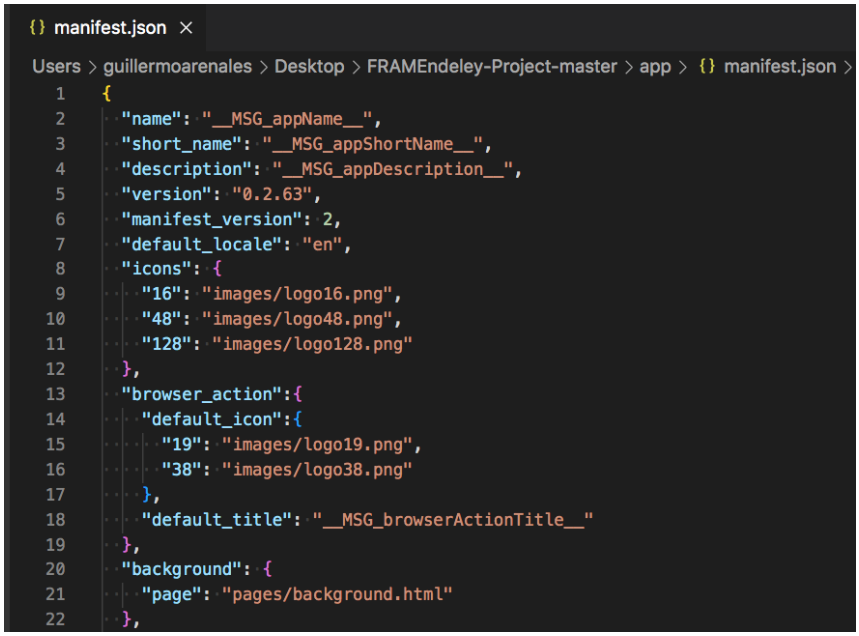


Ilustración 10 Manifest de la extensión

El manifiesto también permite a los desarrolladores declarar una orientación de pantalla predeterminada para su aplicación web, además de brindar la posibilidad de establecer el modo de visualización de la aplicación (por ejemplo, en pantalla completa). Además, el manifiesto permite a un desarrollador "alcanzar" una aplicación web a una URL. Esto restringe las URL a las que se aplica el manifiesto y proporciona un medio para "enlace profundo"

Con estos metadatos, los agentes de usuario pueden proporcionar a los desarrolladores los medios para crear experiencias de usuario que sean más comparables a las de una aplicación nativa.

Comenzamos analizando el Manifest.json que usa nuestra extensión y sus principales componentes:

Lo primero en lo que nos fijamos es la **versión** de manifiesto, en este caso toda la extensión está desarrollada en la versión 2, ahora mismo esto no nos aporta ningún cambio, sin embargo según avance el proyecto veremos la importancia de la versión.

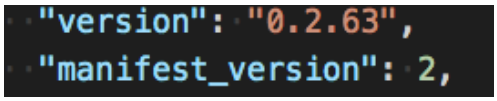


Ilustración 11Versión del Manifest

En lo siguiente en lo que nos fijamos es en la sección **background**, clave para incluir una o más secuencias de comandos de fondo y, opcionalmente, una página de fondo en su extensión. Los scripts de fondo son el lugar para colocar el código que necesita mantener un estado a largo plazo o realizar operaciones a largo plazo, independientemente de la vida útil de cualquier página web o ventana del navegador en particular.

Los scripts de fondo se cargan tan pronto como se carga la extensión y permanecen cargados hasta que la extensión se deshabilita o desinstala. Puede usar cualquiera de las API de Web Extensión en el script, siempre que haya solicitado los permisos necesarios .

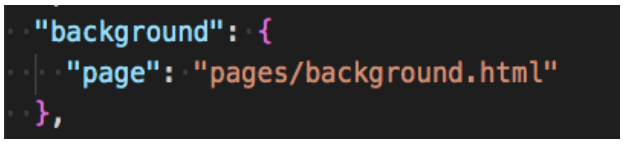


Ilustración 12 Página background de la extensión

Por otro lado tendríamos el **content_scripts**, que indican al navegador que cargue secuencias de comandos de contenido en páginas web cuya URL coincida con un patrón determinado: si la URL del documento coincide con la especificación de la clave, se adjuntará la secuencia de comandos.

Ilustración 13 Content_scripts de la extensión

Seguimos con **content_security_policy**, las extensiones tienen una política de seguridad de contenido aplicada de manera predeterminada. La política predeterminada restringe las fuentes desde las que las extensiones pueden cargar código (como recursos <script>) y no permite prácticas potencialmente inseguras. Este apartado no es obligatorio añadirlo en el manifiesto de todas las extensiones, sin embargo, en nuestro caso es totalmente necesario, ya que de esta forma le podremos dar permisos a google analytics para que empiece a recolectar datos sobre nuestra extensión.

```
"content_security_policy": {
  "extension_pages": "script-src 'self' https://ssl.google-analytics.com; object-src 'self'",
},
```

Ilustración 14 Política de seguridad de la extensión

Finalmente, tenemos el apartado de **permissions**, para solicitar poderes especiales para su extensión. Esta clave es una matriz de cadenas, y cada cadena es una solicitud de permiso.

Si solicita permisos usando esta clave, entonces el navegador puede informar al usuario en el momento de la instalación que la extensión está solicitando ciertos privilegios y pedirles que confirmen que están dispuestos a otorgar estos privilegios. El navegador también puede permitir que el usuario inspeccione los privilegios de una extensión después de la instalación. Dado que la solicitud para otorgar privilegios puede afectar la voluntad de los usuarios de instalar su extensión, vale la pena considerar cuidadosamente la solicitud de privilegios.

```
"permissions": [
  "storage",
  "unlimitedStorage",
  "https://api.mendeley.com/*",
  "webNavigation"
],
```

Ilustración 15 Permisos de la extensión

3.4.2. Background y Content Script:

Las extensiones funcionan sobre las pestañas de nuestro navegador web, y todas las pestañas, sin importar el tipo de página web a la que accedemos, siguen la misma estructura.

Para simplificarlo, podríamos dividir esta estructura en dos partes principales, una parte “interior”, content script, que como su propio nombre indica abarca todo el contenido de la web que estamos accediendo en ese momento.

Por otro lado tenemos la parte “exterior”, background, que nos da un contexto del navegador en general, más allá de la pestaña sobre la que estamos actualmente, esta parte más exterior abarca componentes del navegador como la propia URL o la lista de las otras pestañas que tenemos abiertas en ese momento.

¿Pero por qué es importante esta explicación?

Para comprender que este tipo de estructura tiene una limitación y es que el Content Script puede acceder y modificar contenido al que el Background no puede llegar y necesita , y lo mismo sucede al revés, el Background está en contacto con contenidos al que el Content Script no tiene acceso y también necesita.

¿Y cómo se afronta esta limitación?

La manera más sencilla y común entre los programadores es hacer uso de los mensajes, ambas partes están comunicadas entre sí y tienen tanto la capacidad como para recibir mensajes como para mandarlos, además ambos están a la espera de mensajes en todo momento, de manera que en el momento que una parte necesite acceder al contenido al que no puede acceder y necesita la otra parte se encargará de mandar la información que necesite al momento y viceversa.

En el siguiente apartado hemos preparado un ejemplo sencillo para poder ver cómo se trabaja en una extensión la comunicación entre la parte externa y la parte interna.

3.5. Estructura FRAMEndeley

La extensión de FRAMEndeley es un proyecto bastante complejo y que contiene muchas clases involucradas, sin embargo, con la intención de ayudar al lector para que comprenda mejor cómo funciona esta herramienta hemos diseñado un esquema simplificado de la estructura que se usa:

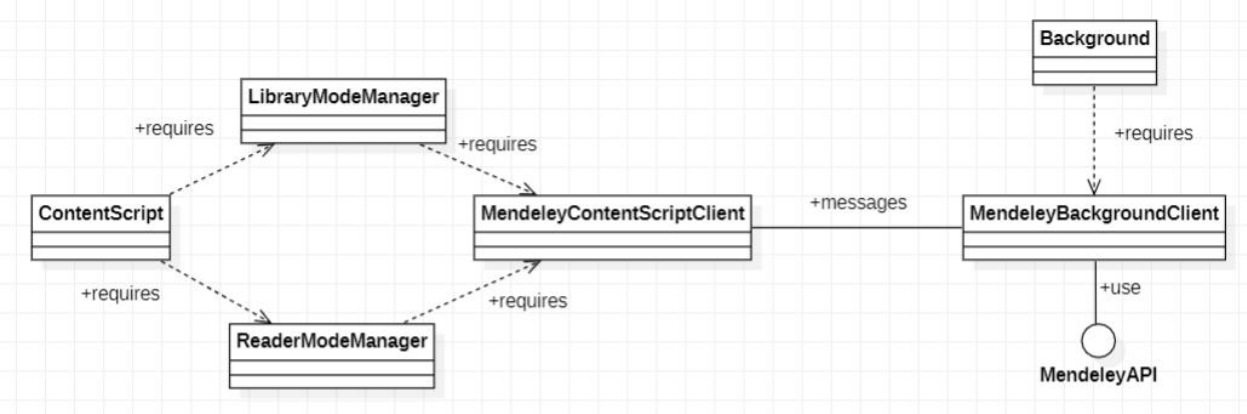


Ilustración 16 Gráfico de la estructura de Framendeley

Estas clases son las encargadas de realizar las principales funciones de FRAMEndeley y como ya hemos explicado en el apartado anterior esta extensión también sigue la típica estructura de intercomunicación entre interior y exterior, como se puede diferenciar en el dibujo entre la parte izquierda y la parte derecha. Además podemos apreciar como una de las clases de la parte externa es la encargada de conectarse directamente con la API de Mendeley.

Por un lado tenemos la parte interna formada por las 4 clases de la izquierda:

Content Script:

Actúa como una clase mediadora entre los dos tipos de modos en los que se puede acceder a la extensión, lector o librería. Analiza la URL y decide si crear una instancia de LibraryModeManager o de ReaderModeManager.

LibraryModeManager:

Una vez accedemos a la sección de librería en la que almacenamos todos nuestros documentos de interés, esta clase se encarga de implementar todas las funciones que se ofrecen en esta pantalla:

ReaderModeManager:

Una vez accedemos a un documento dentro de nuestra librería , esta clase se encarga de implementar todas las funciones que se ofrecen en esta pantalla:

MendeleyContentScriptClient:

Como podemos ver en el esquema esta clase es la encargada de realizar las comunicaciones con el Background. También está en contacto con las dos clases de los dos roles que se utilizan en la aplicación, de manera que si en algún momento la clase LibraryModeManager o la clase ReaderModeManager necesita algún contenido fuera del propio Content Script se lo pedirá a esta clase que a su vez se lo solicitara al Background.

Y por el otro la parte externa formada por las dos clases restantes:

MendeleyBackgroundclient:

Realiza un trabajo muy parecido al de la clase MendeleyContentScriptClient, aunque esta lo hace desde el otro lado, en caso de que el Background necesite información de la parte interna de la extensión esta se comunicará con el MendeleyContentScriptClient y una vez la haya obtenido se la proporcionará al Background.

Además, esta es la única parte que está directamente conectada con la API de Mendeley por lo que también se encargará de comunicarse con la propia API, luego si alguna parte necesitará contenido de Mendeley tendrá que solicitarlo a través de esta clase.

Background:

Esta clase está atenta a todos los cambios que se realizan el exterior como pueden ser las pestañas o la URL, en caso de que se realice algún cambio esta se lo comunicará a MendeleyBackgroundclient inmediatamente para que se transmita el cambio a todas las partes y estas actúan dependiendo del tipo de cambio.

4. Capítulo : Web Analytics para FRAMEndeley

En este capítulo hablaremos sobre el análisis que se ha realizado para crear la arquitectura del proyecto basándonos en qué patrón debería implementar la extensión para poder sacar su máximo provecho. Finalmente se presentará el diseño del producto, y el resultado final.

4.1. Casos de uso

Se han divido las métricas en cuatro grupos, cada uno de ellos tiene como objetivo medir una propiedad:

4.1.1. Agilidad

Habilidad de una extensión para adaptarse rápidamente a los cambios en los requisitos o en el entorno de trabajo. Esto incluye la capacidad de ser modificada o actualizada fácilmente, así como la capacidad de trabajar de manera eficiente en diferentes sistemas o plataformas. La agilidad también se refiere a la facilidad de uso y al bajo impacto en el rendimiento del sistema. En resumen la agilidad es la capacidad de una extensión para ser flexible y adaptarse fácilmente a los cambios y necesidades del usuario. Se han planteado las siguientes preguntas:

Pregunta	Métrica
¿Cuántos documentos abren los usuarios?	Número de documentos que se abre por sesión/día/semana
¿Cada cuánto tiempo crean los usuarios una nueva temática?	Números de temáticas creadas por sesión/día/Semana
¿Cada cuánto tiempo renombran los usuarios las temáticas?	Número de temáticas renombradas por sesión/día/semana

Tabla 2 Estudio de la agilidad

4.1.2. Rendimiento

Se refiere a cómo una extensión funciona en términos de su capacidad para realizar las tareas para las cuales fue diseñada. Esto incluye la precisión y fiabilidad con la que realiza sus tareas, así como la capacidad de cumplir con los estándares y requisitos establecidos. La actuación también se refiere a la facilidad de uso y al nivel de satisfacción del usuario con la extensión. En resumen, la actuación es la medida en la cual una extensión cumple con sus objetivos y requerimientos, y brinda una experiencia satisfactoria al usuario. Se han planteado las siguientes preguntas:

Pregunta	Métrica
¿Cuántos subrayados realizan los usuarios?	Número de subrayados realizados por sesión/día/semana/documento
¿Cuántos subrayados no amarillos realizan los	Numero de subrayados amarillos realizados

usuarios	por sesión/día/semana/documento
¿Cada cuánto tiempo los usuarios crean un nuevo código?	Número de códigos creados por sesión/día/semana

Tabla 3 Estudio del rendimiento 1

4.1.3. Engagement

Se refiere a la medida en la que una extensión logra involucrar y retener a los usuarios. Esto puede incluir la facilidad con la que los usuarios pueden interactuar con la extensión, la personalización y la capacidad de la extensión para adaptarse a las necesidades y preferencias individuales de los usuarios. También se refiere a la capacidad de la extensión para proporcionar contenido y funciones relevantes y valiosas para los usuarios. En resumen, el Engagement es la medida en la que una extensión logra involucrar y retener a los usuarios, proporcionando una experiencia personalizada y relevante. Se han planteado las siguientes preguntas:

Pregunta	Métrica
¿Cada cuánto tiempo leen los usuarios?	Número de lecturas de sesión por día/semana/mes
¿Cuánto tiempo están los usuarios leyendo?	Duración de la sesión de lectura

Tabla 4 Estudio del Engagement

4.1.4. Interacción con las características de FRAMEndeley

Por último, hemos creado una sección para las métricas que no tienen que ver con las propiedades que podría poseer cualquier extensión y nos hemos centrado en algunas que afectan directamente a nuestra extensión. Se han planteado las siguientes preguntas:

Pregunta	Métrica
¿Cada cuánto tiempo abren los usuarios el esquema de Scoping?	Número de veces que un usuario abre el esquema de Scoping por sesión/día/semana
¿Cada cuánto tiempo abren los usuarios el esquema de temáticas?	Número de veces que un usuario abre el esquema de temáticas por sesión/día/semana
¿Cada cuánto tiempo abren los usuarios el	Número de veces que un usuario abre el

esquema de artículos?	esquema de artículos por sesión/día/semana?

Tabla 5 Estudio de la característica 1

4.2. Elección de patrón

Antes de ponernos a implementar todos los cambios fue necesario hacer una reflexión para valorar cómo queríamos que fuera la estructura de esta nueva parte de la extensión:

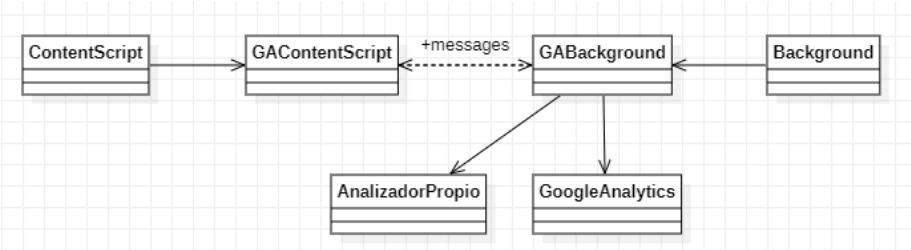


Ilustración 17 Primer boceto de la nueva es de la nueva estructura de la extensión

Este fue un primer boceto de cómo queríamos que fuera la parte de la extensión encargada de conectarse con Google Analytics.

Se basa principalmente en la estructura de la aplicación original

Una vez más siguiendo la estructura externa/interna típica de las extensiones que ya hemos explicado previamente.

Después de pensar un poco en cómo queríamos enfocar la nueva herramienta el siguiente paso era pensar en qué patrón de diseño nos íbamos a basar. Durante este proceso tuvimos bastantes dudas y hasta comenzamos trabajando con un patrón para después cambiarlo.

Teníamos claro que los dos patrones que mejor se adaptan a nuestros objetivos eran el Mediator y el Observer, sin embargo primero nos decantamos por la opción del Mediator:

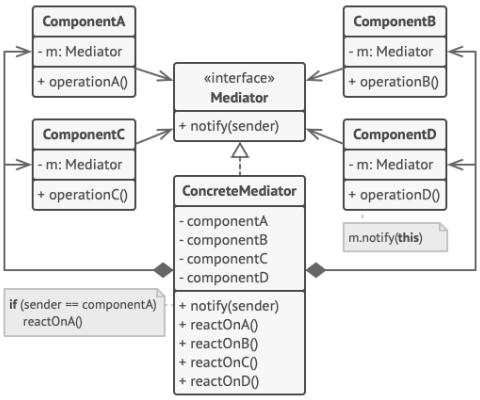


Ilustración 18 Esquema del patrón Mediator

Es un patrón que te permite reducir las dependencias caóticas entre objetos. El patrón restringe las comunicaciones directas entre los objetos, forzándolos a colaborar únicamente a través de un objeto mediador.

Viendo cómo se trabajaba con este patrón pensamos que era la opción ideal para manejar las comunicaciones entre la parte interior y la parte exterior : liberaba de la responsabilidad de comunicarse con otras partes a clases que ya tenían una función definida, haciéndolas más simples y eficientes.

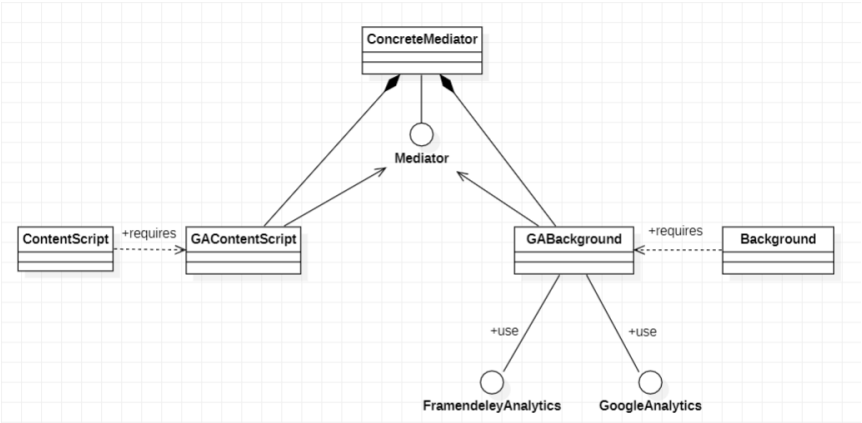


Ilustración 19 Posible esquema de la extensión aplicando el patrón Mediator

Sin embargo a la hora de implementar comenzaron los problemas: la estructura se volvió más enrevesada provocando que fuera más difícil de entender, por otro lado, tampoco ayudaba a que las comunicaciones fueran más sencillas, se tenían que intercambiar muchos más mensajes y hasta clases que antes del cambio no debían ocuparse en enviar mensajes ahora lo requerían.

Además, este cambio tampoco liberó a las principales funciones de responsabilidades, es verdad que ya no tenían que intercambiar mensajes entre ellas pero ahora lo tenían que hacer con el mediador, luego seguían igual de sobrecargadas de tareas que antes.

En conclusión, la estructura se volvió más compleja que la del principio y tampoco ayudó a repartir las tareas entre más clases para que estuvieran más compensadas.

Por lo tanto decidimos que lo mejor sería dejar a un lado el patrón Mediator y probar con nuestra otra opción el patrón Observer:

Observer es un patrón de diseño de comportamiento que te permite definir un mecanismo de suscripción para notificar a varios objetos sobre cualquier evento que le suceda al objeto que están observando.

Al fin y al cabo con las modificaciones que planeábamos hacer a la extensión íbamos a jugar un rol de observador frente a los cambios que se realizaban en la extensión original, luego resultaba un patrón que nos venía como "anillo al dedo".

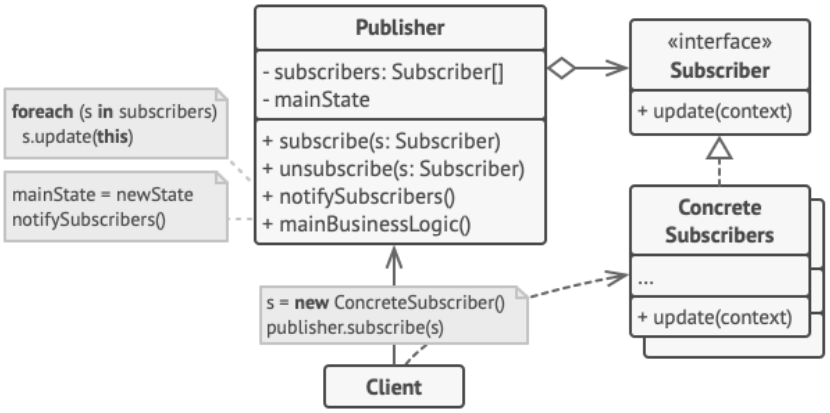


Ilustración 20 Esquema del patrón Observer

Este es un esquema genérico de cómo se podría aplicar este patrón a una aplicación cualquiera.

Finalmente después de todas las variaciones que hemos realizado durante el proceso de definir la nueva estructura de la extensión, este ha sido el resultado:

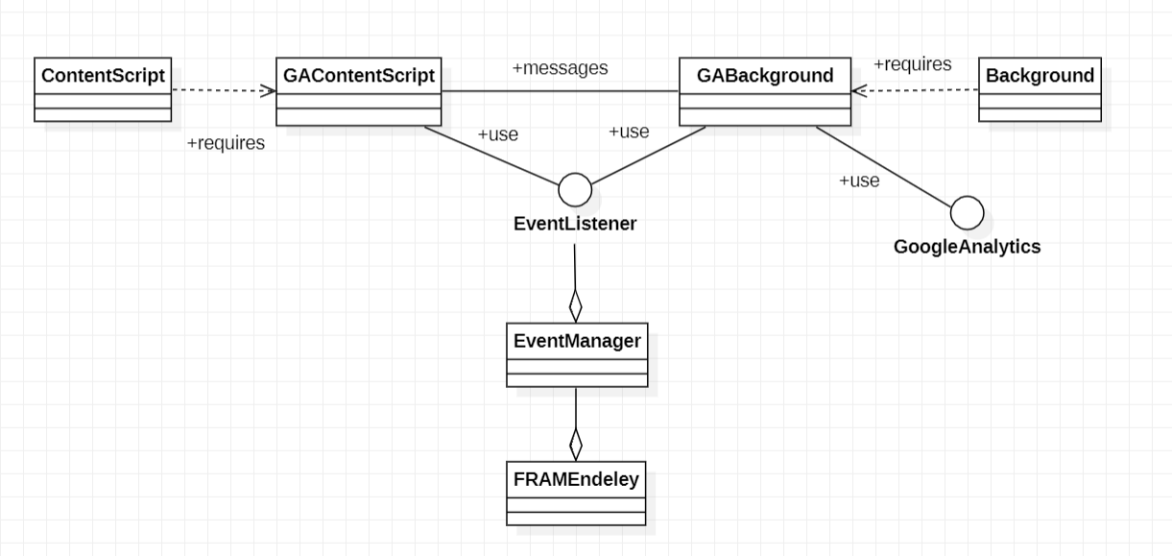


Ilustración 21Esquema final con el patrón Observer

Como podemos ver ambas partes, tanto interna como externa, implementan la extensión ya que el patrón permite a varias clases actuar como observadores a los que se notifican los cambios que se realizan, en este caso, desde la extensión de FRAMEndeley.

5. Capítulo : Implementación

Este capítulo se centrará en la implementación del proyecto. Esto incluye la preparación del entorno de desarrollo y las funciones usadas a destacar.

5.1. Herramienta de desarrollo

En un principio debido a la familiaridad con el entorno se escogió a Visual Studio Code como entorno de programación sobre el que se desarrollaría toda la parte de programación, además ya había realizado más de un proyecto de JavaScript en dicho entorno. Sin embargo, después de hablar con el desarrollador, Jeremías, me recomendó que para la implementación de extensiones web usará el entorno de desarrollo Intellij IDEA, no solo porque a la hora de desarrollar su extensión lo hizo así sino porque es el más idóneo para este tipo de proyectos. Por lo que finalmente, a pesar de no tener ningún tipo de experiencia con la herramienta desarrollada por JetBrains se escogió como el entorno de programación.

Por otro lado, también se decidió que también se trabajaría con Node.js ya que permite la creación de servidores web y herramientas de red utilizando JavaScript y una colección de "módulos" que manejan varias funcionalidades básicas.

5.2. Funciones que destacar

Para realizar este proyecto se han utilizado muchas funciones de cada una de las herramientas que hemos usado. Hay algunas que no tienen mucha relevancia, sin embargo a continuación presentaremos aquellas que podrían ser de interés:

5.2.1. Gulp:

Dentro del fichero de **gulpfile.babel.js**, el cual es un fichero que se encarga de gestionar las tareas, para este proyecto hemos utilizado sobre todo dos:

Build:

Ilustración 22 Código de la función Build

La cual básicamente compila nuestro código de la extensión.

Pack:

```
function getPackFileType () {
  switch (args.vendor) {
    case 'firefox':
      return '.xpi'
    default:
      return '.zip'
  }
}

gulp.task('pack', ['build'], () => {
  let name = packageDetails.name
  let version = packageDetails.version
  let filetype = getPackFileType()
  let filename = `${name}-${version}-${args.vendor}${filetype}`
  return gulp.src(`dist/${args.vendor}/**/*`)
    .pipe(zip(filename))
    .pipe(gulp.dest('./packages'))
    .on('end', () => {
      let distStyled = colors.magenta(`dist/${args.vendor}`)
      let filenameStyled = colors.magenta(`./packages/${filename}`)
      log(`Packed ${distStyled} to ${filenameStyled}`)
    })
})
```

Tarea que se encarga de "construir" nuestro código de la extensión y lo deja comprimido en la carpeta /task listo para una vez descomprimido ser subido a Google Chrome como extensión

Ilustración 23 Código de la función Build

5.2.2. JavaScript:

Al estar desarrollada la mayoría de la extensión en JavaScript es obvio que hemos usado muchas funciones de este lenguaje, destacaremos un par:

El primero es un método de JavaScript que se utiliza para obtener una colección de elementos del documento HTML que tengan el nombre de la etiqueta especificado.

m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;

He decidido resaltarlo ya que es un método que usamos en la creación del objeto **ga**, que actúa básicamente como una instancia de Google Analytics, y el cual es imprescindible para poder llamar a cualquiera de las funciones de esta herramienta.

El otro método del que hablaremos también es usado a la hora de la creación del objeto Google Analytics y es el siguiente:

m.parentNode.insertBefore(a,m);

Es un método de JavaScript que pertenece al API de Document Object Model (DOM). Inserta un nodo antes del nodo de referencia como hijo de un nodo padre indicado. Si el nodo hijo es una referencia a un nodo ya existente en el documento, insertBefore() lo mueve de la posición actual a la nueva posición .

5.2.3. Google Analytics:

Como es obvio hemos tenido que usar varias funciones de Google Analytics para obtener los resultados deseados, pero la función básica con la que hemos obtenido los principales hits (información enviada por el código javascript a Google) es la siguiente:

Con la siguiente función lo que tratamos es de contabilizar todas las visitas que se obtienen para una página determinada, en este caso es la de background.html:

ga('send', 'pageview', '/background.html')

Aunque podemos usar estas funciones para conseguir otros objetivos como con “anonymize IP” que permite a los propietarios de sitios web solicitar que todas las direcciones IP de los usuarios permanezcan enmascaradas dentro del producto.
ga('set', 'anonymizeIp', true)

5.2.4. Extensiones de Chrome:

También se han usado varias funciones de Chrome que nos han ayudado a implementar el código, aunque en este caso a pesar de no ser una función como tal, resaltare las variables globales:

Las variables globales son aquellas que son declaradas fuera de cualquier función y están disponibles en todo el ámbito del documento, tanto dentro como fuera de las funciones. En el contexto de una aplicación de Chrome, las variables globales estarían disponibles en todas las páginas y scripts de esa aplicación. Es por eso que métodos como **chrome.storage.sync.set** son tan útiles:

chrome.storage.sync.set({“GA_ENABLED”, :false});

De la misma forma que en el párrafo anterior establecemos una variable "global", también la podemos obtener usando **chrome.storage.sync.get.** El valor guardado se devuelve en el parámetro options del callback.

chrome.storage.sync.get(["GA_ENABLED"], function (options) ;

6. Capítulo : Pruebas realizadas

En este capítulo hablaremos de las distintas pruebas que se han realizado durante el desarrollo del TFG además hablaremos de la versión Demo del proyecto que ha servido para testear avances en los que no se tenía mucha certeza de cómo actuaría o si funcionaran.

6.1. Demo

Como ya hemos venido comentando a lo largo de esta memoria una de las medidas que se tomó durante la parte de implementación, debido al desconocimiento y/o falta de práctica en algunas herramientas, fue la de probar los avances en una versión mucho más sencilla a la extensión original de modo que no se aplicarían estos cambios a la versión final hasta que no se probaran con éxito en esta versión demo.

Esta demo es infinitamente más simple que la extensión de Framendeley pero contiene los elementos necesarios para probar nuestras ideas. Empezamos fijándonos en el Manifest.json:

Como podemos ver hay un elemento que es completamente necesario para poder conectarse con Google Analytics y es el apartado de "content_security_policy":

Tenemos que especificar a la extensión que el siguiente script es seguro y que nos de permiso para poder cargarlo en tiempo de ejecución "script-src 'self' https://ssl.google-analytics.com; object-src 'self'",

Lo siguiente en lo que nos fijamos es en la estructura que se ha usado en la demo para conectar por primera vez con Google Analytics :

```
var _gaq = _gaq || [];
_gaq.push(['_setAccount', _AnalyticsCode]);
_gaq.push(['_trackPageview']);

(function() {
  var ga = document.createElement('script');
  ga.type = 'text/javascript';
  ga.async = true;
  ga.src = 'https://ssl.google-analytics.com/ga.js';
  var s = document.getElementsByTagName('script')[0];
  s.parentNode.insertBefore(ga, s);
})();
```

Ilustración 24 Código de la demo para conectarse con GA

Y como podemos ver cuando lo ejecutamos en el navegador esta extensión simplemente es un conjunto de botones:

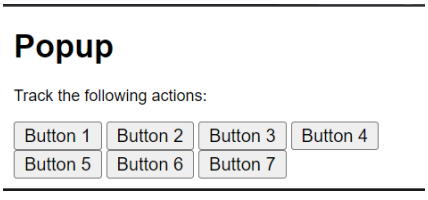


Ilustración 25 Interfaz de la demo 1

Explicadas las partes más relevantes de esta pequeña extensión, pasaremos a analizar las pruebas que se hicieron con ella:

Ejecutamos el siguiente código de Google Analytics el cual nos permitirá contabilizar tanto las visitas de los usuarios como las interacciones que han tenido con los eventos, que en este caso simplemente será clicar un botón:

Y estos han sido los resultados que hemos obtenido:

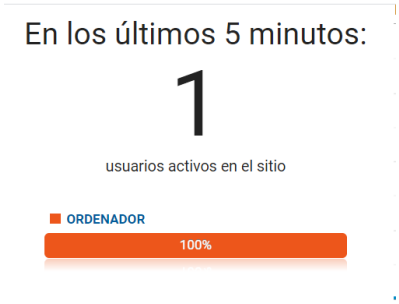


Ilustración 26 Usuarios conectados a la extension DEMO

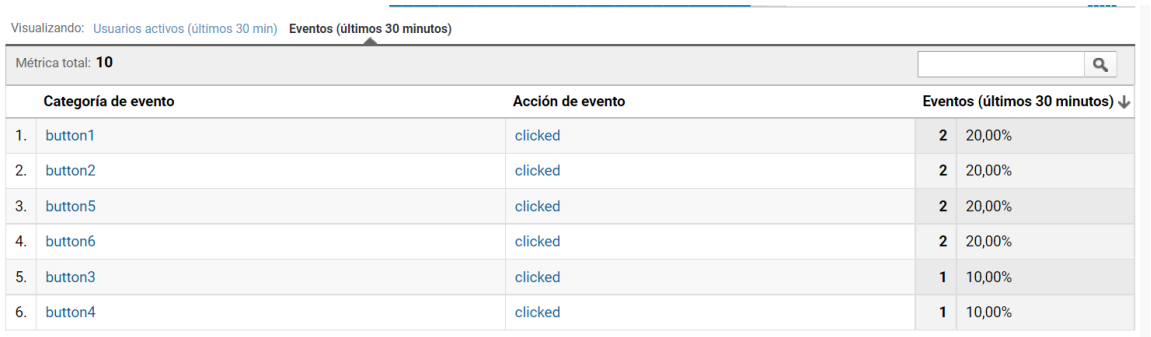
En este caso como este código lo he diseñado yo y no lo he subido a ninguna web pues como era de esperar solo se registra a un usuario que ha podido visitar la extensión.

Por otro lado podemos ver también cómo se han registrado los eventos en el tiempo

Ilustración 27 Eventos registrados en la versión DEMO

Aquí podemos apreciar cómo ha habido un pico de eventos durante los últimos segundos.

Incluso Google Analytics nos permite distinguir entre los diferentes botones, contabilizando incluso el número de clics que ha recibido cada botón específicamente.



	Categoría de evento	Acción de evento	Eventos (últimos 30 minutos)	
1.	button1	clicked	2	20,00%
2.	button2	clicked	2	20,00%
3.	button5	clicked	2	20,00%
4.	button6	clicked	2	20,00%
5.	button3	clicked	1	10,00%
6.	button4	clicked	1	10,00%

Ilustración 28 Captura de la tabla de los eventos de DEMO

Pueden parecer unas pruebas un tanto burdas sin embargo la dificultad que plantea Google Analytics es conectarse por primera vez y enviar los primeros hits, una vez realizado este proceso los siguientes pasos son muy similares a estos.

6.2. Pruebas realizadas

Desde que se consiguió conectar la extensión con Google Analytics se han ido recogiendo los datos de cómo han usado los usuarios nuestra herramienta, estos son algunos “dashboards” que hemos obtenido a lo largo de este tiempo:

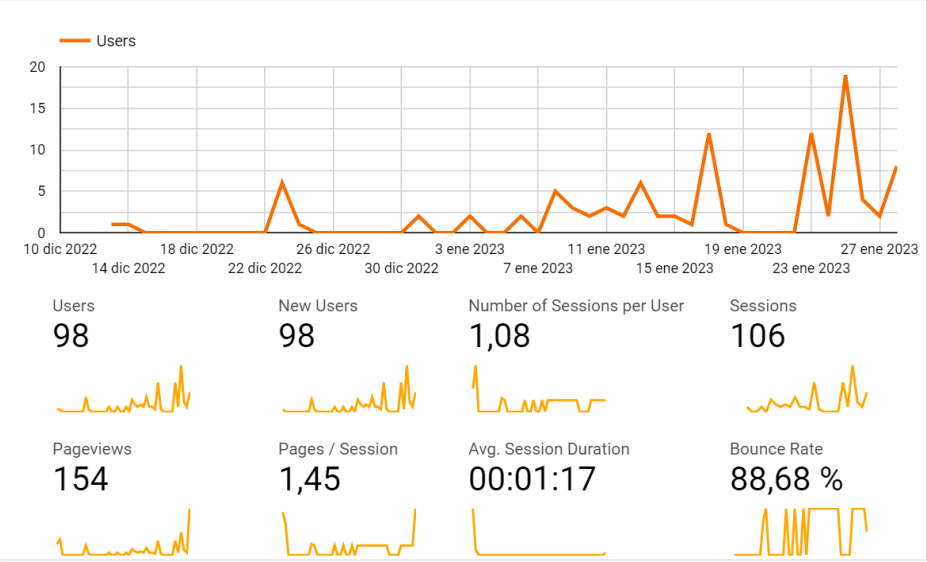
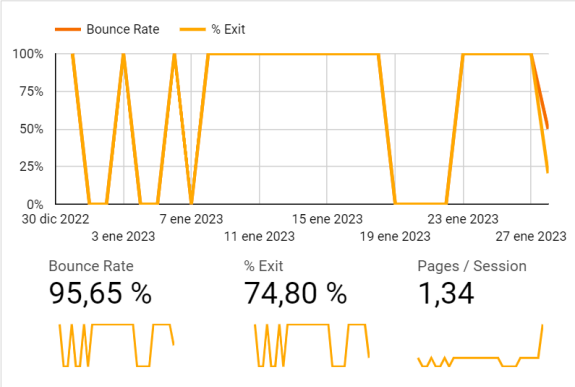
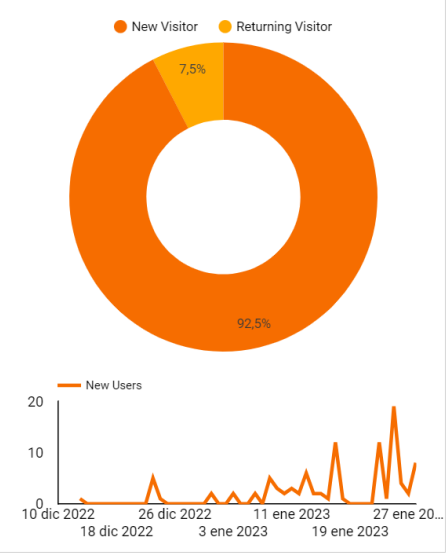


Ilustración 29 Captura del dashboards de GA

Ilustración 30 Gráficos sobre los usuarios que han accedido a la extensión





A continuación se mostrarán los resultados de las pruebas que se han realizado para medir al menos un aspecto de entre las cuatro propiedades que he mencionado en el apartado de estudio:

Comenzando por la agilidad, responderemos a la pregunta de cuantos documentos abren los usuarios, para ello medimos el número de documentos que abre un usuario en una sesión. Para un usuario en concreto obtenemos los siguientes resultados:

Ilustración 31Captura de los eventos en una sesión

Esta es la sesión del último usuario que se ha conectado hace un rato, podemos observar cómo ha abierto hasta 3 documentos.

Ahora nos centraremos en el rendimiento, comprobaremos cuantos subrayados realizan los usuarios a lo largo de una semana:

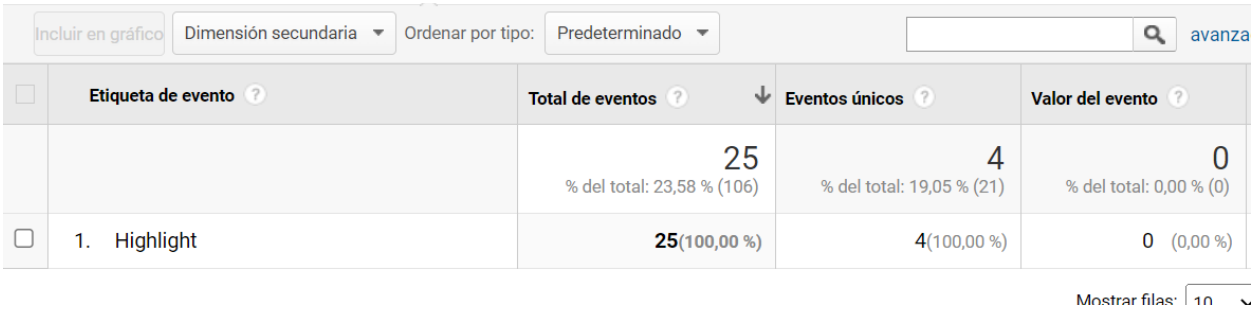


Ilustración 32 Núm. veces Highlight

En cuanto al “Engagement” o retención de usuarios debido a que la extensión está desarrollada por Jeremías, que forma parte del grupo de investigación de Onekin, comprobaremos como de interesados están los usuarios con el grupo que se ha hecho cargo de desarrollar la extensión. Para ello mediremos los accesos a los diferentes links y/o logos que conectan directamente con la página principal de Onekin:

Ilustración 33 Captura de las visitas al enlace de Onekin

En esta gráfica podemos observar el número de veces totales que se han accedido a links que llevaban directamente a la página de Onekin en el último mes. Para poner un poco más en contexto estos resultados diremos que en el último mes el número de visitas que ha recibido la página ha sido de aproximadamente 90.

Por último, veremos cómo es la interacción con las características y elementos de Framendeley, por ejemplo ¿ cada cuanto abren los usuarios el Scoping Canvas ? que es una de las principales funcionalidades que ofrece Framendeley, para ello, igual que en los casos anteriores, “trackearemos” a los usuarios y veremos cuantas veces abren este esquema durante una semana:

Etiqueta de evento	Total de eventos	Eventos únicos	Valor del evento
	47 % del total: 44,34 % (106)	4 % del total: 19,05 % (21)	0 % del total: 0,00 % (0)
1. Open Canvas	**47**(100,00 %)	4(100,00 %)	0 (0,00 %)

Mostrar filas: 10

Ilustración 34 Núm. veces Open Canvas

7. Capítulo : Control y Seguimiento

A lo largo del desarrollo de este proyecto se ha ido controlando el número de horas que se han invertido para cada tarea especificada en el EDT. En este capítulo se mostrará el seguimiento que se ha realizado del proyecto durante el desarrollo de este, principalmente en el número de horas y en la fecha en la que se ha completado cada tarea, y se comparara con la planificación inicial.

7.1. Horas estimadas frente a horas reales

TFG				
	%	Horas	Horas Reales	Diferencia
Gestion	15	45	50	5
Planificacion	8,33	25	25	0
Gestion y Control	6,67	20	25	5
Investigacion	25	75	80	5
Estudio Tecnologias	16,67	50	50	0
Reflexion	8,33	25	30	5
Programacion	30	90	100	10
Demo	6,67	20	20	0
Extension	23,33	70	80	10
Informe	30	90	105	15
Redaccion	20	60	75	15
Revision	10	30	30	0
Suma	100	300	335	35

Tabla 6 Comparativa entre las horas estimadas y las reales

Como podemos apreciar las mayores desviaciones se producen en los paquetes de programación y en el del informe. En el paquete de programación se preveía que una vez se acabará con las pruebas en la demo el "salto" a la versión final no supondría un gran desafío, sin embargo no fue así, algunos avances que se había probado en la demo no funcionaban como se pensaban en la extensión luego se tuvo que dedicar un tiempo extra a la corrección de estos fallos.

En cuanto a la redacción del informe simplemente se realizó una mala predicción en la planificación y se estimaron menos horas a las que finalmente se necesitaron.

7.2. Desviación horas

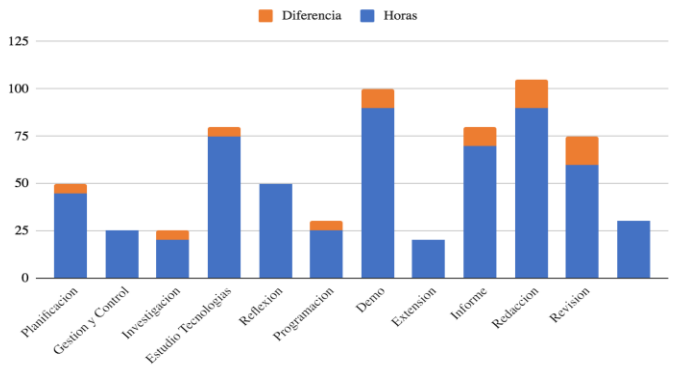


Ilustración 35 Gráfico con las horas extras

Esquema que nos permite observar mejor en qué fases se han producido las desviaciones, la barra azul representa las horas estimadas mientras que las naranjas representan las horas extras que se han invertido en cada tarea.

7.3. Esquema final

En la siguiente imagen podemos apreciar como ha quedado el diagrama de Gantt aplicando las horas que realmente se necesitaron para completar cada tarea y tomando en cuenta cuando se ha finalizado realmente una tarea:

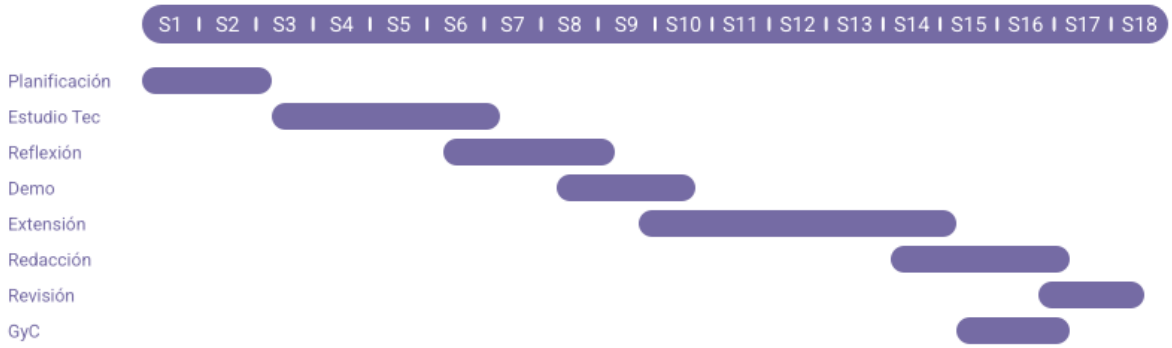


Ilustración 36 Diagrama de Gantt final

8. Capítulo: Conclusiones

En este capítulo que utilizaremos a modo de cierre de la memoria hablaremos de los conocimientos que me ha ayudado el TFG a adquirir y también de mi experiencia personal realizando un proyecto de este tamaño. Finalmente también propondremos un par de mejoras que una vez acabado todo he contemplado.

8.1. Conclusión del proyecto

Una vez acabado el proyecto y prácticamente redactada la memoria en su totalidad se podría decir que el trabajo se ha realizado con un nivel de calidad aceptable.

Aunque, hay que resaltar que se han cometido desviaciones en las últimas etapas, sobre todo en la fase de implementación que una vez terminada y probada la versión demo se pensaba que aplicarla a la versión final no supondría una dificultad, sin embargo se necesitó unas cuantas horas extra y una tutoría con Jeremías para lograr el objetivo. También hay que resaltar que se estimaron menos horas de las que finalmente resultaron a la hora de redactar la memoria. Hechos que provocaron que se apurara demasiado con los plazos de entrega. Aun así, y como se ha comentado al principio del párrafo se ha completado el proyecto.

En cuanto a la tecnología, la extensión no ha sufrido ningún tipo de deterioro tras haberle aplicado todos los cambios inyectando los scripts de JavaScript. Además, la extensión sigue conectada con Google y este sigue recolectando datos de los usuarios que recibe a diario.

En cuanto al trabajo como tal a pesar de parecer un simple trabajo de desarrollo de software también ha implicado mucho tiempo y esfuerzo de investigación para familiarizarse con las herramientas y analizar los resultados.

Para finalizar con esta conclusión quería resaltar también que no solo se ha logrado el objetivo general sino que también los objetivos específicos que se plantearon en el apartado de planificación.

8.2. Conocimientos y experiencia adquiridos

Quiero destacar sobre todo los conocimientos adquiridos en cuanto tecnologías, en JavaScript por ejemplo, cuando se empezó el proyecto tenía un nivel bastante básico de haberlo visto ligeramente en un par de asignaturas, y aunque no creo que mi nivel actualmente sea de avanzado sí que es verdad que he ganado en soltura y en práctica con este lenguaje. Ocurre algo similar con Google Analytics, era una herramienta de la que había oído hablar y tenía una idea de qué funciones podría ofrecer, pero ahora es verdad que comprendo mucho mejor cómo funciona esta herramienta y he visto el verdadero potencial de la misma.

Por otro lado, en ámbitos como el de Node.js o Gulp era la primera vez que trabajaba con ellos y ahora por lo menos puedo decir que al menos tengo un nivel básico con estas herramientas.

En cuanto a mi experiencia, he de agradecer a mi tutor Oscar que ha ofrecido una gran accesibilidad en todo momento y me ha ofrecido toda la ayuda y el apoyo que ha podido, a pesar

de que yo no fuera el alumno más ordenado ni el más eficiente. Agradecer también a Jeremías que me ha ayudado bastante con la parte de implementación, tanto explicándome como funcionaban las diferentes herramientas como aclarándome el funcionamiento del código de su extensión.

Finalmente, he de decir que a pesar de tratar sobre temas que no conocía y con los que no había trabajado a penas me han sorprendido gratamente, me ha parecido un trabajo interesante en el que me he encontrado cómodo trabajando en él.

8.3. Posibles mejoras

Por el tipo de TFG que se planteó, en el que parte de él dependía de la participación de los usuarios hubiera sido muy interesante poder subir la extensión al mercado de la tienda de Chrome y experimentar con usuarios reales, no porque no se pueda realizar sino porque hubiera sido muy difícil captar la atención de nuevos usuarios a una extensión que ya existía y que a estos usuarios no hubieran notado ninguna mejora luego el volumen de participantes hubiera sido muy bajo o directamente nulo y no habría servido de demasiado. Aunque es verdad que frente a este problema, de esperar que una gran cantidad de usuarios añada nuestra extensión al navegador, no hay mucho que podamos hacer al respecto.

Por otro lado, otra posible mejora, la cual se planteó como plausible al principio del proyecto y que no se ha acabado llevando a cabo debido a la falta de tiempo fue la de realizar una conexión a un servidor que actuará como Google Analytics pero que estuviera controlado por nosotros de manera que pudiéramos recibir todos los datos en “crudo”, no como para algunos parámetros de Analytics en los que Google los aplica sobre un gráfico y no se puede obtener el dato exacto.

ANEXOS

ANEXO A. Ejemplo gráfico de cómo interactúan la parte externa e interna de una extensión.

Esta extensión funciona como un marcapáginas pero para los videos de YouTube, es decir, en caso de que no tengas tiempo de ver un video entero y quieras acabar de verlo en otro momento y no quieras perder tiempo buscando exactamente en qué momento dejaste de ver el video, puedes dejar una marca en la barra de reproducción del video para poder acceder exactamente al segundo donde lo dejaste de ver sin perder tiempo, o si simplemente quieres destacar un momento del video.

La acción de destacar un video se realiza mediante un botón que se posiciona al lado de los controles de reproducción de YouTube (pausa/play, avanzar/retroceder, control de volumen etc.).

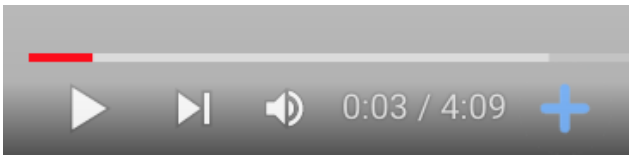


Ilustración 37 Reproductor de YouTube con la extensión

La marca de la cruz azul es el botón de marcapáginas.

Sin embargo, si accedemos a otra página que no sea YouTube no nos interesa colocar el botón de marcapáginas, pero ¿cómo sabrá la extensión si estamos en YouTube?
Aquí es donde entra la comunicación entre exterior e interior.

```
chrome.tabs.onUpdated.addListener((tabId, tab) => {
 if (tab.url && tab.url.includes("youtube.com/watch")) {
  const queryParameters = tab.url.split("?")[1];
  const urlParameters = new URLSearchParams(queryParameters);
  chrome.tabs.sendMessage(tabId, {
    type: "NEW",
    videoId: urlParameters.get("v"),
  });
 }
});
```

El background al tener acceso a todas las pestañas y la propia URL comprueba si la pestaña actual sobre la que está el usuario es YouTube, en caso afirmativo envía un mensaje al Content Script avisando que debe actuar y añadir el botón de marcapáginas.

Una vez entendidos estos conceptos acerca de las extensiones ya estamos preparados para comprender mejor el funcionamiento y la estructura de la herramienta FRAMEndeley.

ANEXO B. Estructura de los ficheros del proyecto

Los ficheros del proyecto se han dividido en varias carpetas, cada una con su respectiva función:

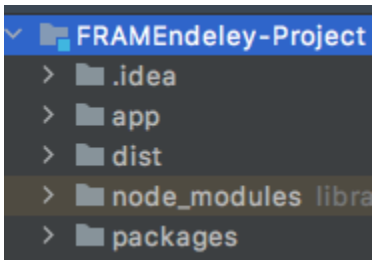


Ilustración 38 Estructura de archivos general

.idea: La carpeta .idea en la raíz de la solución contiene los archivos de configuración específicos del proyecto de IntelliJ. Estos incluyen detalles por proyecto, como el mapeo de VCS y las configuraciones de ejecución y depuración, así como detalles por usuario, como los archivos abiertos actualmente, el historial de navegación y la configuración seleccionada actualmente.

app: contiene todas las carpetas que hemos utilizado para crear y modificar la extensión según queramos.

dist: significa *distribución* , y es la versión minimizada/concatenada , que en realidad se usa en los sitios de producción, por ejemplo el directorio que pasamos a Google para que añada la extensión al navegador posee el contenido de dist .

node_modules: contiene el código que se genera cuando se compila la aplicación.

packages: contiene la versión comprimida de la aplicación una vez se ha compilado y construido.

Pero a pesar de contar con distintas carpetas la que verdaderamente nos interesa es la de app, en la que hemos trabajado y desarrollado toda la extensión:

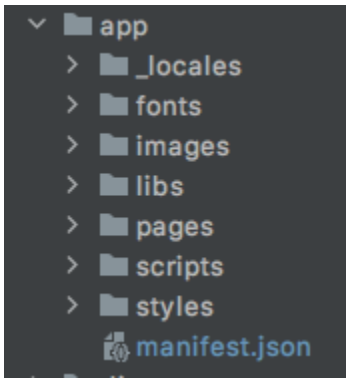


Ilustración 39 Estructura archivos carpeta app

_locales: esta carpeta está destinada a gestionar los mensajes que muestra la extensión en distintos idiomas, aunque actualmente solo está en inglés.

fonts: gestiona la fuente del tipo de letra que se utiliza.

images: aquí se almacenan todas las imágenes que se han usado para la extensión, mayoritariamente son las imágenes de los logos tanto de Mendeley como Framendeley.

libs: se guardan las librerías que se han utilizado para desarrollar el trabajo.

pages: aquí se almacenan los archivos .HTML de las diferentes páginas que puede acceder la extensión mientras es utilizada. No solo sirven para la parte visual del proyecto también algunas páginas se encargan de cargar los scripts JavaScript según el usuario interaccione con los elementos que se ven en pantalla.

scripts: contiene toda la lista de scripts de JavaScript tanto de la extensión inicial como el de la extensión una vez modificada para conectarse con Google Analytics. Como parece obvio esta es la carpeta en la que más se ha trabajado y donde se ha dedicado la mayor parte del desarrollo del proyecto.

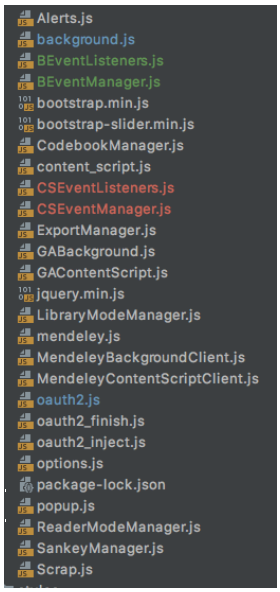


Ilustración 40 Estructura archivos scripts

Por último dentro de la carpeta app podemos encontrar el fichero **Manifest.json** del cual ya hemos hablado y sabemos de su importancia en la extensión.

Referencias

Developing research questions in conversation with the literature: operationalization & tool support:
https://link.springer.com/article/10.1007/s10664-022-10204-8

¿Cómo escribir la sección de Trabajo Relacionado? Una herramienta para Mendeley:
https://biblioteca.sistedes.es/submissions/descargas/2019/JISBD/2019-JISBD-069.pdf

Mendeley, Wikipedia, The Free Encyclopaedia , 2022:
https://en.wikipedia.org/wiki/Mendeley

Refactoring Guru, Design Patterns, Observer:
https://refactoring.guru/es/design-patterns/observer

Refactoring Guru, Design Patterns, Mediator:
https://refactoring.guru/es/design-patterns/mediator

2023, W3C, Web application Manifest:
https://www.w3.org/TR/appmanifest/

2022, JetBrains, Running and Debugging Node.js:
https://www.jetbrains.com/help/idea/running-and-debugging-node-js.html

2016, Gisela Torres ,¿Qué es Gulp y para qué sirve?:
https://www.returngis.net/2016/07/que-es-gulp-y-para-que-sirve/#:~:text=Gulp%20es%20una%20herramienta%2C%20en,sint%C3%A1xis%20y%20un%20largo%20etc%C3%A9tera.

2022, Gulp , JavaScript and Gulpfiles:
https://gulpjs.com/docs/en/getting-started/javascript-and-gulpfiles/

2023, MDN web Docs, Manifest.json:
https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/manifest.json

2021, TechTarget Contributor, Wesley Chai, Google Analytics:
https://www.techtarget.com/searchbusinessanalytics/definition/Google-Analytics

2023, Google Support, Enmascaramiento de IP en Universal Analytics:
https://support.google.com/analytics/answer/2763052

2023, Google Developers, Medición de eventos:
https://support.google.com/analytics/answer/2763052

2023, Chrome Developers, Tutorial: Google Analytics:
https://developer.chrome.com/docs/extensions/mv3/tut_analytics/

2014, David Walsh, Track JavaScript Errors with Google Analytics:
https://davidwalsh.name/track-errors-google-analytics

2023, Chrome Developers, Publish in the Chrome Web Store:
https://developer.chrome.com/docs/webstore/publish/

2018, Tomer Ben Rachel ,How to publish your Chrome Extension:
https://www.freecodecamp.org/news/how-to-publish-your-chrome-extension-dd8400a3d53/

2020, Whatagraph, Wendy, Which Kinds of Hits Does Google Analytics Track?:
https://whatagraph.com/blog/articles/which-kinds-of-hits-does-google-analytics-track

2021, David Escolano, Que es un hit en Google Analytics:
https://www.davidescolano.es/glosario-analitica-web/hit-en-google-analytics.html

2022, Google Developers, Measurement Protocol (Google Analytics 4):
https://developers.google.com/analytics/devguides/collection/protocol/ga4?hl=es

2023, Google Developers, Medir vistas de pagina y vistas de pantalla:
https://developers.google.com/analytics/devguides/collection/ga4/views?hl=es&technology=websites

2018, Ritwik B,The Ultimate Guide To Google Analytics Hits: What Are Hits & Its Limitations?
https://www.digishuffle.com/blogs/google-analytics-hits/

2019,Aaron Mandelbaum, Which kind of hits does google analytics track? what to know:
https://paradoxmarketing.io/capabilities/knowledge-management/insights/which-kind-of-hits-does-google-analytics-track-what-to-know/

2022, Google Analytics Event Tracking Tutorial:
https://www.optimizesmart.com/event-tracking-guide-google-analytics-simplified-version/