Escuela Politécnica Superior

Aplicación móvil de generación de resúmenes automáticos

Grado en Ingeniería Multimedia

Trabajo Fin de Grado

Autor:
Antonio Jiménez Pérez

Tutor/es:
Elena Lloret Pastor, José Manuel Gomez Soriano

Universitat d'Alacant
Universidad de Alicante

Junio 2015

Justificación y Objetivos

El trabajo de fin de grado de ingeniería multimedia es un proyecto individual en el que se sintetizan e integran las competencias adquiridas a lo largo de toda la titulación.

Un titulado en Grado en Ingeniería Multimedia debe ser capaz de adaptarse a trabajar con sistemas informáticos, abarcando perfiles diferentes y no estando enfocado a un único perfil. Además es capaz de participar y dirigir, de tal forma que puede ser tanto un programador como un diseñador, al igual que puede dirigir a estos e incluso ser el nexo de unión entre los mismos, también está capacitado para aprender y adaptarse a nuevas tecnologías existentes. Así mismo, es capaz de realizar el diseño y la implementación de interfaces que ayuden a los usuarios en la tarea de uso de los productos creados.

Este proyecto trata de un sistema de gestión y difusión de contenidos digitales para la plataforma móvil (Android) que servirá para demostrar todo lo aprendido, enfocándolo al itinerario de Gestión y Difusión de Contenidos Digitales.

Concretamente, el ***objetivo general*** de este proyecto es la realización de un sistema de gestión y difusión de contenidos digitales orientado al usuario, en el que se incluyan conceptos aprendidos a lo largo del grado cursado, produciendo un proyecto profesional que muestre las habilidades que un ingeniero multimedia es capaz de desarrollar. Este proyecto se ha materializado en el desarrollo de GPLSI Compendium App.

GPLSI Compendium App es una aplicación móvil que hará de un sistema de gestión y difusión de contenidos digitales, en la que el usuario podrá realizar resúmenes de textos de diferentes webs, tales como artículos o noticias, y posteriormente compartir esos resúmenes en sus redes sociales o a través del correo. Se le proporcionarán varios métodos de resumen, así como las descripciones y un pequeño tutorial de cómo utilizar la aplicación.

Al ser este un proyecto del itinerario de Gestión y Difusión de Contenidos Digitales, y por tanto, al haber cursado el mismo… ¿qué habilidades o conocimientos del Grado en Ingeniería Multimedia se relacionan con la creación de un sistema de gestión de contenidos digitales?, es una pregunta de difícil respuesta. Con este trabajo se han puesto en práctica conocimientos de planificación, desarrollo, diseño y estudio de sistemas digitales para extraer información, así como habilidades de diseño de interfaz centrado en el usuario, extracción y tratamiento de la información de sistemas digitales, y generación de resúmenes.

Dedicatoria

Este trabajo, por decirlo de algún modo, resume mi paso por esta ingeniería y por cuatro años originales, auténticos, y sobre todo, apasionantes. Por ello me gustaría dedicárselo a todas esas personas que desde que comencé en este grado estuvieron apoyándome, independientemente de no conocer esta ingeniería o parecerles del todo extravagante.

En primer lugar, a mis padres, y sobre todo a mi madre, que sin dudar me dijo: "Si es lo que te gusta, adelante", y sin ese apoyo nada de esto hubiera sido posible.

A mi abuela, que siempre ha estado ahí y siempre lo estará, sin ni siquiera conocer esta titulación ni de qué iba, lo único y primero que siempre le sale es: "Que tengas mucha suerte hijo".

A toda mi familia en general, sobre todo a la más cercana o con la que comparto más momentos, como mi tía Beatriz, mi "hermana" y prima Bea, y su padre.

Y por último pero no menos importante, a mis amigos, esa familia que está para todo, y que sin querer, te hacen superar cualquier obstáculo.

A todos ellos, gracias de corazón.

Índice de contenidos

Índice de figuras

Índice de tablas

1 Introducción

La gestión y difusión de contenidos digitales está implantada en todas las plataformas desde hace muchos años, pero desde la explosión de los *smartphones* y demás dispositivos (como tablets, etc) ha cobrado más importancia que nunca (IDC, 2015).

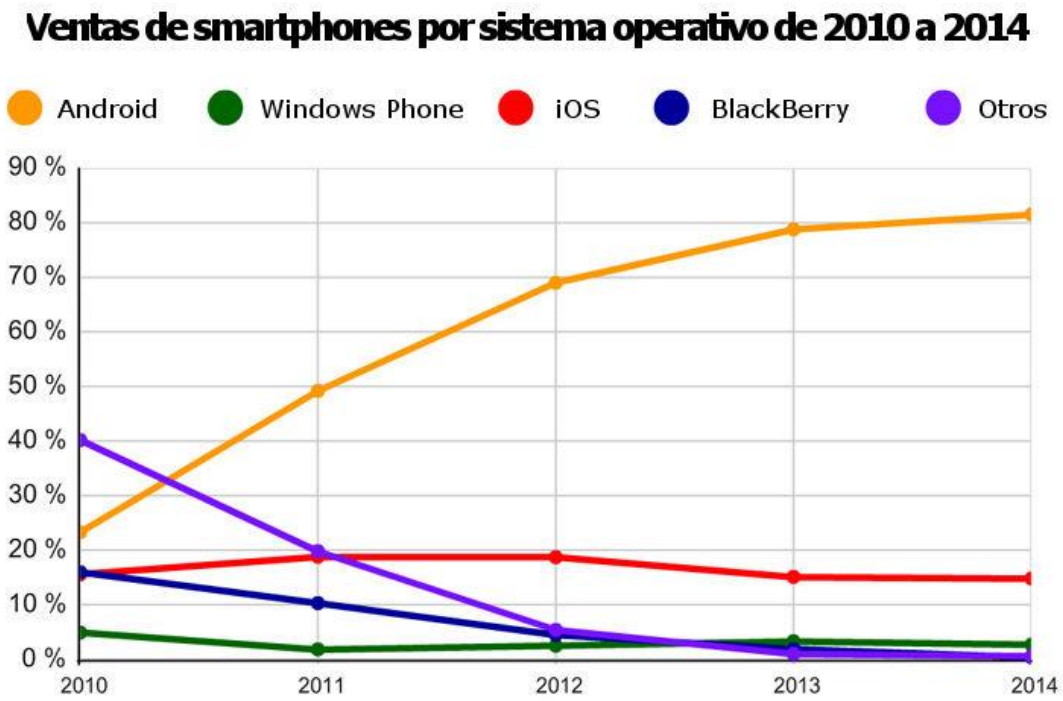


Figura 1: Ventas de smartphones por SO de 2010 a 2014

Ya no se nos hace raro ver al día muchos vídeos e imágenes por internet o que nos mandan nuestros amigos, y ni que decir tiene que todo el día estamos enganchados mandando mensajes de texto. Vivimos en una espiral de consumo y difusión de contenidos digitales a diario.

Actualmente Android es el sistema más utilizado por las personas, delante de IOS, sistema de Apple, y su gran rival durante años, como se puede apreciar en la figura anterior.

Este sistema lleva creciendo exponencialmente desde hace años, su funcionamiento, su facilidad de uso, su soporte de desarrollo, todo, y por ello apenas se dudó al querer realizar un proyecto para este sistema.

El sistema Android es una plataforma que cuenta con gran cantidad de usuarios como se ha mencionado anteriormente, que pone a disposición de los usuarios toda esta información digital en forma de aplicaciones, como navegadores de internet y aplicaciones de Google Play Store.

Por otra parte, a medida que pasa el tiempo encontramos más cantidad de información digital, que se hace difícil de abarcar y consumir, por ello surgen herramientas que nos ayudan a procesarla, concretamente a partir de técnicas y herramientas de Procesamiento del Lenguaje Natural (Boronat, 1999) y que son muy útiles en la sociedad actual.

Por todo esto nació el proyecto GPLSI Compendium App, un gestor y difusor de contenidos digitales para Android, pero más que un proyecto, es la culminación a varios años de aprendizaje, por lo que aparte de ser un proyecto, es un producto en el que mostrar todo lo aprendido en esta etapa, y la muestra de que es posible realizar un producto de este tipo a este nivel de experiencia.

En el siguiente apartado, 1.1 Motivación, se dará una descripción de la motivación que conlleva la realización de este proyecto y su justificación. En segundo lugar, en el apartado 1.2 Finalidad, se explicará la finalidad de este proyecto, y los objetivos más importantes que se persiguen con la realización del mismo. Por último, en el 1.3 Estructura del documento, se describirán brevemente los diferentes apartados de este documento, para hacer una idea del contenido de cada uno.

1.1 Motivación

¿Por qué es necesaria una aplicación de estas características? ¿Y por qué para Android? Para empezar, GPLSI Compendium App te ayuda en la síntesis de textos y noticias de diferentes periódicos digitales (por el momento *Marca.com, BBC.co.uk, AS.com, WhasingtonPost.com*), y hoy en día, con la cantidad de contenido digital que se puede encontrar, resulta de gran utilidad para personas tenga un tiempo limitado para leer todos los artículos. En segundo lugar, el sistema operativo imperante hoy día es Android, y por ello se ha creído conveniente realizarlo para esta plataforma, y de esta forma, llegar a gran cantidad de usuarios.

1.2 Finalidad

La finalidad fundamental de este proyecto es crear una aplicación que permita resumir noticias, que sea completa, estable y usable para los usuarios sin equipo, es decir, desarrollarla de forma individual, tanto aspectos de diseño como de programación de funcionalidades, y de esta forma poner en práctica todas las competencias adquiridas a lo largo del grado.

Otra finalidad es la de crear un sistema de gestión y difusión de contenidos digitales para plataforma móvil, y terminar con un producto estable que se pueda mostrar.

Así mismo, y por ello no menos importante, otra de las finalidades es que este trabajo sirva como acreditación a la hora de poder trabajar en una empresa, para que esta misma pueda valorar de forma positiva el trabajo realizado y que sea un valor añadido a la hora de presentarte en un puesto de trabajo. Este punto es clave ya que el trabajo de fin de grado debe ayudar a la incorporación a alguna empresa relacionada con el tema del trabajo.

1.3 Estructura del documento

La estructuración de este documento se ha realizado siguiendo el libro de estilo para la presentación de memorias del Trabajo Fin de Grado (Grado en ingeniería multimedia, 2014). Aparte de esto, dentro de cada apartado se han incluido los subapartados oportunos que se han considerado necesarios para explicar todas las partes del proyecto, el cual está en la rama de gestión y difusión de contenidos digitales.

Teniendo en cuenta estas consideraciones, la estructura de la que se compone este documento es la siguiente:

En primer lugar encontramos en este mismo capítulo 1. Introducción, donde se realiza una primera toma de contacto con el tema del que trata el proyecto, explicando la motivación, cuál es su finalidad y lo que nos ocupa, la estructura del mismo.

Tras la introducción se hace un repaso a lo largo de 2. Marco teórico de la historia de los sistemas de gestión de contenidos, así como del presente y futuro de los mismos. Otros temas que se tratan en este capítulo son el de los tipos de gestores de contenido actualmente y una introducción al Procesamiento del Lenguaje Natural (PLN) y generación de resúmenes, así como aplicaciones similares a esta que se encuentran en el mercado. Se acabará con la relación de todo con esta aplicación, GPLSI Compendium App.

Seguidamente tenemos en capítulo 3. Objetivos, donde se explican los objetivos a alcanzar con la realización de este proyecto.

En el capítulo 4. Metodología, se explica en un primer apartado el tema del control de versiones para este proyecto, y seguidamente se encuentra el apartado de metodología de desarrollo del proyecto, donde se explica cuál es la metodología que se va a seguir para realizar el proyecto.

Tras este, encontramos la parte fundamental del documento, 5. Cuerpo del trabajo, capítulo que consta del **análisis, el funcionamiento y el diseño de la aplicación**. En el apartado 5.1 Análisis

se realiza una explicación del contenido principal de la aplicación, también se plasman los requisitos funcionales y no funcionales del proyecto. Por último se tratan las herramientas con las que se va a realizar el proyecto, el ámbito y público y el tipo de modelo de negocio planteado para el futuro. Después tenemos 5.2 Funcionamiento de GPLSI Compendium App, donde se explican todas las funcionalidades de GPLSI Compendium App. Finalmente tenemos el apartado 5.3 Diseño de la aplicación, donde se explican los pasos seguidos para la elección del diseño de la aplicación, así como del logo de la misma.

Seguidamente encontramos en capítulo 6. Pruebas y evaluación, donde se explicarán los resultados de una encuesta hecha a usuarios de la aplicación, y se terminará explicando para qué sirven y en qué consisten los indicadores Google Play Store.

Luego tenemos el capítulo 7. Valoración personal, donde se plasman las sensaciones sobre este proyecto.

En el capítulo 8. Conclusiones y trabajo futuro, donde se hace un repaso de las sensaciones de la realización de este proyecto y las ideas de futuro que se tienen para el mismo.

Para acabar tenemos los capítulos 9. Bibliografía y referencias y 10. Anexos, donde en primer lugar se citan las fuentes que han ayudado a la realización del proyecto, y por último, en los anexos se incluyen los contenidos pertinentes para completar la información de este proyecto, como es el caso de 10.1 Guía de desarrollo.

2 Marco teórico

Este proyecto se construye en base a dos pilares, los Gestores de contenidos y el Procesamiento del Lenguaje Natural, por ello a continuación se explicarán cada uno de los mismos.

2.1 Gestores de contenidos

Un sistema de gestión y difusión de contenidos (CMS) es un software o programa que permite crear una estructura de soporte para la creación y administración de contenidos, ya sea, texto, imágenes, vídeos, etcétera (Sistemas de gestión de contenidos, 2015). Teniendo en cuenta esto, en los siguientes apartados se describirá y se proporcionará información sobre este tipo de sistemas, cómo han evolucionado y su mercado hoy en día, además claro está con la relación en este proyecto.

2.1.1 Historia

Nos basamos en (Hernández, 2014) para recopilar y destacar la información más relevante sobre la historia de los Gestores de Contenido y los ejemplos existentes.

A principios de los años noventa, el concepto de sistemas de gestión de contenidos era desconocido. Algunas de sus funciones se realizaban con aplicaciones independientes: editores de texto y de imágenes, bases de datos y programación a medida.

Ya el año 1994 Illustra Information Technology utilizaba una base de datos de objetos como repositorio de los contenidos de una web, con el objetivo de poder reutilizar los objetos y ofrecía a los autores un entorno para la creación basado en patrones. La idea no cuajó entre el público y la parte de la empresa enfocada a la Web fue comprada por AOL, mientras que Informix adquirió la parte de bases de datos.

RedDot es una de las empresas pioneras que empezó el desarrollo de un gestor de contenidos el año 1994. No fue hasta a finales del año siguiente que presentaron su CMS basado en una base de datos.

Entre los CMS de código abierto uno de los primeros fue Typo 3, que empezó su desarrollo el año 1997, en palabras de su autor, Kasper Skårhøj, “antes de que el término gestión de contenidos fuera conocido sobradamente”.

PHPNuke, la herramienta que popularizó el uso de estos sistemas para las comunidades de usuarios en Internet, se empezó a desarrollar el año 2000. La primera versión supuso tres semanas de trabajo al creador, rescribiendo el código de otra herramienta, Thatware.

En la actualidad, aparte de la ampliación de las funcionalidades de los CMS, uno de los campos más interesantes es la incorporación de estándares que mejoran la compatibilidad de componentes, facilitan el aprendizaje al cambiar de sistema y aportan calidad y estabilidad.

Algunos de estos estándares son CSS, que permite la creación de hojas de estilo; XML, un lenguaje de marcas que permite estructurar un documento; XHTML, que es un subconjunto del anterior orientado a la presentación de documentos vía web; WAI, que asegura la accesibilidad del sistema; y RSS, para sindicar contenidos de tipo noticia.

También las aplicaciones que rodean los CMS acostumbran a ser estándar, como los servidores web Apache y ISS (Internet Information Services); los lenguajes PHP, Perl y Python; y las bases de datos MySQL y PostgreSQL. La disponibilidad para los principales sistemas operativos de estas aplicaciones y módulos, permite que los CMS puedan funcionar en diversas plataformas sin muchas modificaciones.

Algunos ejemplos de CMS más usados son Drupal y Wordpress, gestores de contenido que permiten el manejo y gestión de la información de forma sencilla, usando todos los estándares nombrados anteriormente (CSS, XML…).

En relación al futuro de los CMS, se apunta a que se convertirán en un artículo de consumo, lo que provocará una disminución de los precios en los productos comerciales y una mayor consistencia en las funcionalidades que ofrecen. Esto podrá repercutir en que empresas que implementen webs tengan más complicado su futuro, y que muchos proyectos no puedan llevarse a cabo por no ajustarse a los estándares y no aplicar conceptos como usabilidad. Por último, se prevé una fusión entre gestión de contenidos, gestión de documentos y gestión de registros.

2.1.2 Tipos de Gestores de contenidos

Actualmente encontramos diferentes criterios para clasificar a los gestores de contenido (Duran Cruz, 2014):

- Por sus características: dentro de este grupo tenemos dos subgrupos:

    - **Según el lenguaje de programación que se emplee:** Java, PHP, etcétera.
    - **Según la licencia:**
        - Código abierto: el código fuente está disponible a todo el que quiera modificarlo.
        - Software propietario: código fuente perteneciente a empresas que restringen el acceso a terceros.

- Por su uso y funcionalidad:

    - **Blogs:** para páginas personales.
    - **Foros:** para compartir opiniones.
    - **Wikis:** para desarrollo colaborativo.
    - **Enseñanza electrónica:** para enseñanza en línea.
    - **Comercio electrónico:** para gestión de usuarios, compras y pagos.
    - **Publicaciones digitales:** para publicaciones en formato digital.
    - **Difusión de contenido multimedia:** para difundir contenido multimedia.
    - **Propósito general:** sobre temas generales.

2.2 Procesamiento del lenguaje natural y generación de resúmenes

El Procesamiento del Lenguaje Natural (PLN) se encarga de procesar el lenguaje humano de forma automática. Este área de investigación es una subdisciplina de la Inteligencia Artificial que investiga y formula mecanismos computacionalmente efectivos para facilitar la interrelación hombre-máquina, permitiendo una comunicación mucho más fluída y menos rígida que los lenguajes formales (Boronat, 1999).

Esta área se compone a su vez de varias disciplinas, y en la vamos a hacer hincapié es en la de generación de resúmenes automáticos, en la que está basada la GPLSI Compendium App.

Para la siguiente parte, correspondiente a la generación de resúmenes, nos basamos en (Lloret, 2011) para recopilar la información correspondiente. La generación de resúmenes no es una tarea nueva, ya que los primeros intentos de producir resúmenes automáticos se llevaron a cabo a finales de los años 50. Sin embargo, ha experimentado una gran evolución en la última década, sobre todo desde el rápido crecimiento de Internet. La cantidad de información disponible en formato electrónico crece de manera exponencial, dando lugar a millones de documentos cuya magnitud dificulta en gran medida su manejo. Debido a esto, la generación de resúmenes es de gran utilidad para procesar dicha información y presentarla de forma resumida y sencilla, de modo que ofrezca al usuario la posibilidad de gestionar la información de una forma más eficiente.

El objetivo de la tarea de generación de resúmenes es obtener una versión reducida del documento o documentos fuente, reduciendo su contenido de tal forma que se seleccionen y queden presentes en el resumen los conceptos más importantes de dichos documentos. Por lo tanto, de la definición de esta tarea se deduce que un resumen debe contener la información más significativa de uno o varios documentos, teniendo un tamaño considerablemente inferior al del documento(s) fuente.

Los sistemas de generación de resúmenes se pueden clasificar en base a múltiples factores. Por ejemplo, relacionado con los factores de entrada, podemos producir un resumen a partir de un solo documento (mono-documento) o a partir de un conjunto de ellos (multidocumento). Además, no siempre debe tratarse de documentos textuales, ya que podemos realizar resúmenes a partir de otros tipos de texto, como páginas web, blogs, imágenes, vídeos, reuniones, etc. En lo que respecta al resumen generado, éste se puede conseguir siguiendo dos posibles estrategias: extractiva o abstractiva. Si se sigue una estrategia extractiva, se seleccionarán y extraerán, literalmente, las frases más importantes sin realizar ninguna modificación sobre ellas. Sin embargo, si se opta por llevar a cabo una estrategia abstractiva para producir un resumen, será necesario realizar algún tipo de transformación en las frases seleccionadas o en los conceptos que se deseen que formen parte del resumen, de tal manera que la información que aparezca en el resumen estará expresada de una forma distinta a la que aparece en el documento original.

Este proyecto se basa en una generación de resúmenes mono-documento extractivo, es decir, se generaran resúmenes de un solo documento extrayendo las frases más importantes del mismo sin realizar modificaciones sobre ellas.

2.2.1 GPLSI Compendium

Compendium es el servicio web realizado en el que se va a apoyar en parte este proyecto para generar algunos tipos de resumen y mostrarlos al usuario de la aplicación. Este sistema fue el resultado de una tesis doctoral realizada en la Universidad de Alicante por Elena Lloret, tutora de este proyecto. (GPLSI Compendium, 2015).

2.3 Aplicaciones similares

Existen aplicaciones de generación de resúmenes automáticos para Android como Squash (https://play.google.com/store/apps/details?id=com.software995.squash) que realiza un solo tipo de resumen, sin posibilidad de dar al usuario la opción de qué tipo de resumen quiere para su texto. Está en inglés.

Otro ejemplo es SumIt (https://play.google.com/store/apps/details?id=com.karimo.sumit_final), una aplicación en inglés que permite introducir texto y resumirlo por número de líneas, y después te da la opción de compartir el resumen.

En la misma línea encontramos Text Summarizer (https://play.google.com/store/apps/details?id=com.adanvita.android.textSummarizer) que hace lo mismo que SumIt.

Para iOS encontramos Summly, una aplicación para el idioma inglés que selecciona y recoge distintas noticias de distintitos temas y las resume en un párrafo de menos de 400 caracteres (Blagdon, 2013).

Por su parte GPLSI Compendium App incorpora la novedad de proporcionar varios métodos de resumen que soportan tanto el idioma inglés como el español, aparte de un buscador de palabras y las opciones que comparten los anteriores como compartir los resúmenes y resumir según la longitud deseada.

2.4 Relación con proyecto GPLSI Compendium App

Aunque es cierto que este proyecto no es en sí mismo un sistema de gestión de contenidos, ya que, entre otras cosas no trabaja con ninguna base de datos, sí que se basa en sus mismos conceptos, y por ello se han utilizado como base los sistemas de gestión de contenidos para la investigación y realización del mismo.

GPLSI Compendium App es más bien un sistema que analiza, trata y difunde contenido digital, en este caso texto en forma resumida, obteniendo lo más relevante, y por ello corresponde al itinerario de gestión de contenidos y no al de ocio digital.

3 Objetivos

Como objetivo general, está el de demostrar todas las competencias aprendidas a lo largo de la titulación aplicándolas al trabajo, así como el crear un producto estable, completo y funcional, como si se tratara de un trabajo profesional.

Por otra parte, los objetivos específicos planteados son los siguientes:

- Crear una aplicación para plataforma móvil (Android).
- Realizar una aplicación Android compatible con la mayoría de dispositivos que existen en el mercado actualmente.
- Enfrentarse a herramientas no vistas ni usadas anteriormente y aprender a manejarlas y aplicarlas al proyecto.
- Capacidad de organización a la hora de desarrollar el proyecto paralelamente a la realización del curso de la titulación.
- Investigación de técnicas de procesamiento de lenguaje natural (PLN) y generación de resúmenes automáticos, junto con su posterior implementación en el proyecto.
- Abarcar y realizar las distintas fases en el desarrollo de un proyecto de importancia, como son la investigación, planificación y posterior desarrollo del mismo.
- Integrar técnicas de generación de resúmenes en aplicación real.

4 Metodología

En este capítulo se explicará la forma de gestionar el proyecto y sus respectivas copias de seguridad para no perder la información con la que se está trabajando, así como la metodología de desarrollo seguida para la realización del proyecto.

4.1 Control de versiones y repositorio

Para la gestión de este proyecto y sus respectivas versiones se empleará Subversion (svn), una herramienta que permite el control de versiones y además es open source. Este tipo de herramientas dan mucho soporte y ayuda para no perder datos de la aplicación o poder volver a una versión anterior sin problemas, por si algo falla.

Además, se pueden acceder a los datos de la aplicación desde cualquier ordenador que tenga permisos para ello.

Otra de las ventajas de esta herramienta es la comparación de diferentes versiones del mismo archivo, para que elijas el que más te convenga para tu proyecto actual.

Dicho esto, queda claro que se utilizará como contenedor de nuestro proyecto, pudiendo guardar una copia de seguridad continua del mismo, aunque es recomendable guardar copia de seguridad en algún disco duro externo por si el programa fallara o perdiera archivos.

Para el caso de GPLSI Compendium App se harán copias de seguridad en intervalos de una semana, para ir guardando los cambios y el avance en el desarrollo del proyecto.

El programa que se utilizará para hacer uso de Subversion será TortoiseSVN, programa de fácil uso y aprendizaje que permite de forma fácil y rápida gestionar las diferentes versiones y updatear nuestro proyecto. Es una herramienta multi-lenguaje y proporciona una interfaz intuitiva y usable para el usuario.

4.2 Metodología de desarrollo de software

La metodología de desarrollo seguida para la realización de este proyecto se corresponde mayormente con las pautas de las metodologías ágiles, donde no hay fases estrictas y los procesos son más flexibles (Scrum adaptado a iteraciones de 15 días). Haber utilizado otro tipo de metodología más estricta en cuanto a planificación no hubiera sido adecuado en nuestro caso, al no estar dedicado en exclusiva a la realización de la aplicación.

De esta forma, el proceso que se seguirá para el desarrollo de GPLSI Compendium App es el siguiente:

- Se establecerán las diferentes partes del desarrollo del proyecto, que son: creación de diferentes pantallas de la aplicación con sus funciones de resumen, establecer estilo para la aplicación, adición de funcionalidades y pruebas de funcionamiento.
- Establecimiento de tareas para las diferentes partes del desarrollo.
- Separación de tareas más críticas y menos críticas para decidir las prioridades ordenándolas de más prioritarias a menos.
- Una vez se tengan claras las tareas y las prioridades, se seleccionarán en el orden que se crea oportuno, teniendo en cuenta tanto la prioridad como las preferencias personales, para que una persona no sienta que está realizando siempre la misma parte, de esta forma se hace el desarrollo más rápido y agradable, más ágil.
- Se marcará la tarea que se está realizando, así como el tiempo invertido en la misma y el porcentaje completado.

No se pueden planificar tareas y tiempos concretos ya que pueden surgir contratiempos en cualquier momento y no se podrían realizar ambas cosas en un periodo de tiempo tan corto.

El proyecto, se irá completando realizando primero las tareas más prioritarias, intentando trabajar de una en una, es decir, mientras no esté completada una no pasar a la siguiente, por lo que es idóneo separar tareas en subtareas más pequeñas. Siempre se podrán modificar las tareas, ya que no es una planificación cerrada.

Este tipo de desarrollo permite ir creando pequeñas versiones o prototipos de la aplicación que se enviarán al tutor del proyecto y servirán para ver los avances del mismo y proponer mejoras y cambios, junto con reuniones periódicas de seguimiento para corregir errores e ir completando la realización del proyecto.

5 Cuerpo del trabajo

5.1 Análisis

Este apartado va a constar de un detallado análisis del proyecto. Este análisis será la base para una buena planificación y gestión de las tareas a realizar en el proyecto. Además sirve de fuente para recurrir cuando surgen dudas en el proyecto sobre qué debía realizarse y de qué forma. También sirve de guía para conocer los pasos a seguir en el desarrollo.

A continuación se explica el contenido de cada uno de los apartados de este capítulo:

En el apartado 5.1.1 Identificación de los requisitos se hace un repaso de los requisitos que debe cumplir nuestro sistema, como es el *contenido de la aplicación*, es decir, elementos como menús, pantallas, botones y demás; tras esto tendremos los *requisitos funcionales*, que indican las funcionalidades que debe tener el sistema para que los usuarios puedan usarla, y finalmente los *requisitos no funcionales*, que no son directamente funciones del sistema sino propiedades que debe tener.

El apartado 5.1.2 Herramientas Hardware y Software se repasan las herramientas que se utilizarán para el desarrollo de este proyecto.

A lo largo del apartado 5.1.3 Ámbito y público objetivo se explica cuál es el perfil de usuarios que pueden llegar a utilizar GPLSI Compendium App.

Finalmente, el apartado 5.1.4 Negocio y futuro incluye cuál es la idea de negocio que se planea seguir con GPLSI Compendium App y cuál será su futuro.

5.1.1 Identificación de los requisitos

A lo largo de este apartado se analizarán los requisitos que debe tener nuestro sistema, desde aspectos visuales hasta aspectos funcionales y no funcionales.

5.1.1.1 **Contenido de la aplicación**

GPLSI Compendium App constará de cuatro pantallas por las que el usuario podrá navegar y hacer uso de la misma:

- **Inicio:** Primera pantalla que se encuentra el usuario. En esta pantalla habrá un texto introductorio a modo de pequeño tutorial para una primera toma de contacto del usuario con la aplicación. Tendrá el botón de "Entendido" para acceder a la pantalla Principal donde se realizan la mayoría de funciones de la aplicación, así como el menú (que se encontrará en la parte superior de la pantalla, o bien en la parte inferior, en el botón físico del dispositivo si es que lo tiene) que podrá llevarlo a las páginas de "Forma de Uso" y "Acerca De", que ahora se explicarán.

Figura 2: Pantalla de Inicio

- **Principal:** Pantalla donde se encuentra la mayor parte de la funcionalidad de la aplicación. En esta pantalla encontramos un pequeño texto a modo de consejo, seguido de dos menús desplegables, uno con la selección de los tipos de resumen (Elemental, Básico, Avanzado, Experto, Inteligente y Manual) y otro con la forma de resumen (por línea o por párrafos). Dependiendo de la forma de resumen seleccionada aparecerá un selector de porcentaje de resumen o no, este selector permite elegir la cantidad de texto que quiere obtener como resumen. En la parte superior encontramos la opción de buscar en el texto que se obtiene para resumir, así como el botón de compartir el resumen generado con las diferentes redes sociales o correo, aparte de esto tenemos el menú en la parte superior derecha como en Inicio, pero este con más opciones, como son las de "Añadir Url", "Copiar al portapapeles", "Ver contenido del portapapeles", y el enlace a las páginas de "Forma de Uso" y "Acerca De".

Figura 3: Pantalla Principal

- **Forma de Uso:** Pantalla en la que se detallan los métodos de resumen para que el usuario seleccione el que más se adecue a sus necesidades.

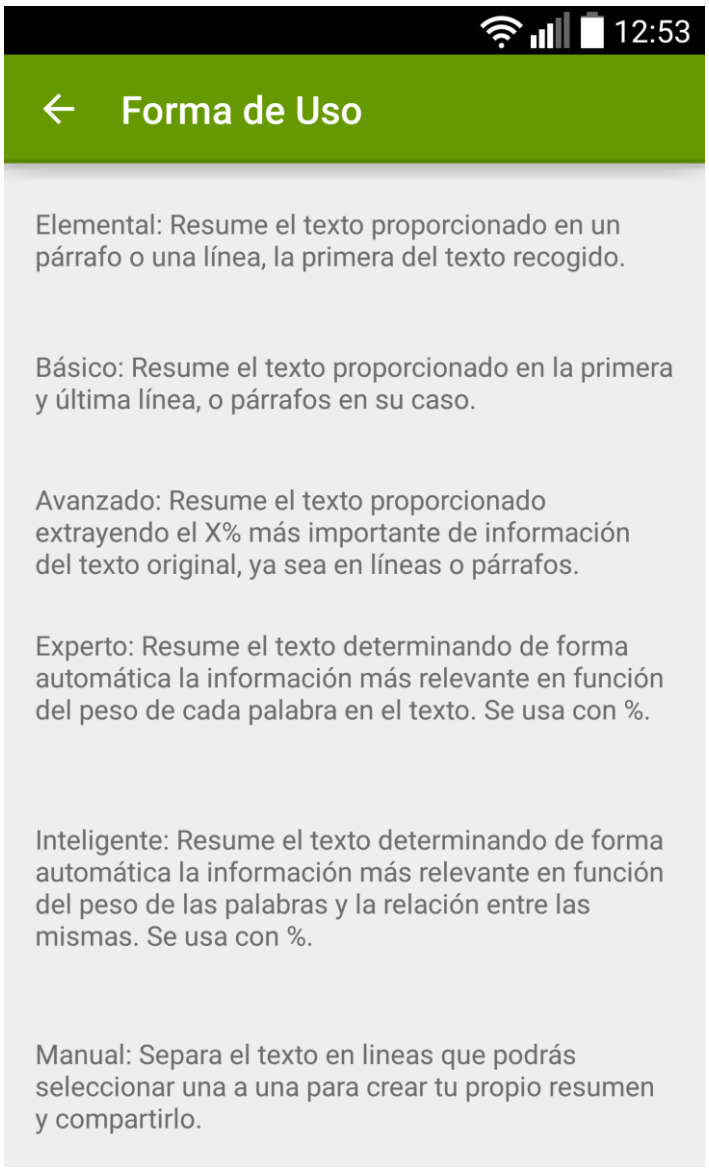


Figura 4: Pantalla Forma de Uso

- **Acerca De:** Pantalla con la información de contacto con el desarrollador de la aplicación.

Figura 5: Pantalla Acerca de

5.1.1.2 **Requisitos funcionales**

Los requisitos funcionales describen lo que el sistema debe poder hacer, es decir, sus funcionalidades o comportamiento.

Los atributos que emplearemos serán:

- ***Identificador***: identifica de forma única a cada requisito funcional (RF), para ello se emplearán las siglas RF y un número único entre los requisitos funcionales.
- ***Necesidad***: Esencial o deseable. Especifica la prioridad del mismo.
- ***Título***: se trata de una frase que sirve de nombre para el requisito.
- ***Descripción***: se expondrá el significado del requisito a modo de descripción.

Los requisitos funcionales del sistema son los siguientes:

Identificador	RF-1
Título	Iniciar la aplicación
Necesidad	Esencial
Descripción	El sistema debe iniciar sin problemas y cargar los recursos necesarios

Tabla 1: Requisito funcional 1.

Identificador	RF-2
Título	Mostrar pantalla de inicio
Necesidad	Esencial
Descripción	El sistema debe mostrar la pantalla de inicio de la aplicación una vez que ésta se ha cargado

Tabla 2: Requisito funcional 2.

Identificador	RF-3
Título	Mostrar pantalla principal
Necesidad	Esencial
Descripción	Una vez se pulse el botón Entendido el sistema debe pasar a la pantalla principal de la aplicación

Tabla 3: Requisito funcional 3.

Identificador	RF-4
Título	Mostrar pantalla forma de uso
Necesidad	Esencial
Descripción	El sistema debe mostrar la pantalla de Forma de Uso de la aplicación una vez se seleccione del menú superior derecho esa opción

Tabla 4: Requisito funcional 4.

Identificador	RF-5
Título	Mostrar pantalla acerca de
Necesidad	Esencial
Descripción	El sistema debe mostrar la pantalla de Acerca De de la aplicación una vez se seleccione del menú superior derecho esa opción

Tabla 5: Requisito funcional 5.

Identificador	RF-6
Título	Mostrar menú
Necesidad	Esencial
Descripción	Ya sea en la pantalla de Inicio o Principal, el sistema deberá mostrar al usuario el menú una vez pulse en el icono que corresponde, con las opciones que corresponda

Tabla 6: Requisito funcional 6.

Identificador	RF-7
Título	Interactuar con botones y menús de pantalla
Necesidad	Deseable
Descripción	El sistema debe permitir al usuario interactuar con todos los elementos de los que se componga la pantalla, como botones, menús, etcétera

Tabla 7: Requisito funcional 7.

Identificador	RF-8
Título	Actualización de texto a resumir en pantalla principal
Necesidad	Esencial
Descripción	El sistema debe actualizar la información en forma de texto de la pantalla principal al recoger texto de alguna web o del portapapeles para permitir resumir el mismo

Tabla 8: Requisito funcional 8.

Identificador	RF-9
Título	Transición entre pantallas
Necesidad	Esencial
Descripción	El sistema debe realizar una correcta transición entre las pantallas de la aplicación

Tabla 9: Requisito funcional 9.

Identificador	RF-10
Título	Buscar texto
Necesidad	Deseable
Descripción	El sistema debe buscar el texto que el usuario desee cuando éste introduzca en el buscador cualquier palabra o letras

Tabla 10: Requisito funcional 10.

Identificador	RF-11
Título	Compartir texto con aplicaciones externas
Necesidad	Deseable
Descripción	El sistema debe compartir el texto que se encuentre en ese momento con cualquier aplicación de la lista de aplicaciones que se proporcione

Tabla 11: Requisito funcional 11.

Identificador	RF-12
Título	Guardar último método de resumen seleccionado
Necesidad	Deseable
Descripción	El sistema debe guardar el último método de resumen seleccionado por el usuario para que cuando vuelva a abrir la aplicación esté seleccionado

Tabla 12: Requisito funcional 12.

Identificador	RF-13
Título	Realizar resumen
Necesidad	Esencial
Descripción	El sistema debe realizar el resumen del texto proporcionado por el usuario

Tabla 13: Requisito funcional 13.

Identificador	RF-14
Título	Añadir url
Necesidad	Esencial
Descripción	El sistema debe capturar el texto de la url dada por el usuario y actualizar el texto a resumir en la pantalla principal

Tabla 14: Requisito funcional 14.

Identificador	RF-15
Título	Copiar texto al portapapeles
Necesidad	Deseable
Descripción	El sistema debe copiar al portapapeles del dispositivo el texto que se encuentre en la pantalla principal

Tabla 15: Requisito funcional 15.

Identificador	RF-16
Título	Diseño independiente del dispositivo
Necesidad	Deseable
Descripción	El sistema debe mostrar la información independientemente del tamaño de la pantalla del dispositivo y de su orientación

Tabla 16: Requisito Funcional 16.

Identificador	RF-17
Título	Mostrar texto del portapapeles
Necesidad	Deseable
Descripción	El sistema debe mostrar el texto que se encuentre en el portapapeles del dispositivo como texto a resumir en la pantalla principal

Tabla 17: Requisito funcional 17.

Identificador	RF-18
Título	Salir de la aplicación
Necesidad	Esencial
Descripción	El sistema debe salir de la aplicación correctamente al pulsar la tecla de atrás en el dispositivo.

Tabla 18: Requisito funcional 18.

5.1.1.3 **Requisitos no funcionales**

Los requisitos no funcionales son aquellos que se refieren a las propiedades que un sistema debe de tener, como fiabilidad, respuesta en un tiempo determinado, usabilidad, etcétera.

A continuación se nombrarán aquellos que resultan más importantes en el desarrollo de un producto software como el que nos ocupa, es decir, una aplicación.

Al igual que con los requisitos funcionales, los atributos que emplearemos serán:

- ***Identificador*:** identifica de forma única a cada requisito no funcional (RNF), para ello se emplearán las siglas RNF y un número único entre los requisitos no funcionales.
- ***Necesidad*:** Esencial o deseable. Especifica la prioridad del mismo.
- ***Título*:** se trata de una frase que sirve de nombre para el requisito.
- ***Descripción*:** se expondrá el significado del requisito a modo de descripción.

A continuación se mostrarán los requisitos no funcionales:

Identificador	RNF-1
Título	Usabilidad
Necesidad	Esencial
Descripción	La usabilidad hace referencia a todo lo que hace que nuestra aplicación sea usable, es decir, que cualquier persona pueda hacer uso de ella de una manera sencilla y sin requerir de muchas indicaciones.

Tabla 19: Requisito no funcional 1.

Identificador	RNF-2
Título	Fiabilidad
Necesidad	Esencial
Descripción	Referido a lo fiable que es nuestra aplicación, en relación a fallos inesperados o mal funcionamiento que entorpezca la experiencia de usuario.

Tabla 20: Requisito no funcional 2.

Identificador	RNF-3
Título	Mantenibilidad
Necesidad	Esencial
Descripción	Propiedad que se refiere a lo fácil de mantener que es un sistema, lo fácil que es modificarlo o actualizarlo sin invertir demasiado coste temporal y económico.

Tabla 21: Requisito no funcional 3.

5.1.2 Herramientas Hardware y Software

Para el desarrollo de esta aplicación se ha hecho uso tanto de herramientas software como hardware. El software que se ha utilizado ha sido del tipo libre.

- **Software utilizado:** se hará uso de la herramienta Android Studio, un programa que permite a desarrolladores Android crear sus proyectos de forma cómoda y fácil. Este programa es un editor de código que permite crear, importar, exportar proyectos y muchas cosas más. Aparte de esto, se utilizará la plataforma InTime, con GPLSI Compendium, un sistema de generación de resúmenes automáticos desarrollado como tesis doctoral.
- **Hardware utilizado:** de la parte hardware se utilizará un portátil Sony Vaio (procesador i5 y 3gb de ram) y un ordenador de sobremesa HP (procesador i5 y 4gb de ram). Además de esto se hará uso para probar la aplicación de un dispositivo móvil LG G3. También se usarán periféricos comunes como el teclado y ratón.

5.1.3 Ámbito y público objetivo

No existen requisitos mínimos para el uso de esta aplicación ya que está realizada para cualquier dispositivo Android y no contiene funcionalidades que requieran más potencia o más capacidad de almacenamiento como algunas otras.

El campo de la gestión y la difusión de contenidos actualmente llega a personas de todas las edades, por ello el público objetivo de esta aplicación son todas aquellas personas que:

- Utilicen dispositivos móviles (tablets, smartphones) de tipo Android.
- Estén interesadas en noticias de prensa.
- Quieren estar actualizadas de lo que pasa, pero no disponen del tiempo suficiente para leer todas las noticias de prensa.

5.1.4 Negocio y futuro

El primer camino a tomar por parte de un desarrollador de este tipo de aplicación, es decir, un desarrollador Android, es hacer uso de la plataforma Google Play Store para subir la aplicación y que cualquier usuario de esta plataforma pueda descargarla y usarla.

Pero antes de ponerla en esta plataforma, se ha de pensar en la forma de comercializarla. Para GPLSI Compendium App se tiene pensado subirla de forma gratuita y que los usuarios puedan disfrutar de ella con el fin de proporcionar a los usuarios una aplicación que les permita ahorrar tiempo en la lectura y procesamiento de la información, y al mismo tiempo mantenerlos informados y actualizados, y de este modo poder coger algo de reputación como desarrollador para futuras aplicaciones.

GPLSI Compendium App se encontrará en la plataforma Google Play Store a disposición de los usuarios que quieran descargarla.

Como se ha dicho, GPLSI Compendium App será gratuita en una primera versión, y a partir de ahí se analizará el impacto de la misma en Google Play Store para futuras actualizaciones o aplicaciones. Dicho esto, si la aplicación fuera un éxito y tuviera mucha demanda popular, se tomaría el camino de insertar publicidad mediante Google AdSense y de esta forma poder conseguir algo de dinero por el trabajo realizado. Otra forma de conseguir ingresos sería la de hacer algunas funcionalidades de pago, como por ejemplo algún método de resumen o el tema de compartir con redes sociales.

Se plantea registrar la aplicación como producto software para mantener los derechos de propiedad intelectual.

5.2 Funcionamiento de GPLSI Compendium App

En esta parte del documento se va a tratar todo lo relacionado con el funcionamiento de la aplicación, explicando todas y cada una de las funcionalidades de las que se compone.

Antes de explicar ninguna de ellas, indicar que en el anexo de este documento se encontrará la guía de desarrollo de GPLSI Compendium App, que comprenderá todo lo necesario para que cualquier persona que quiera continuar desarrollando esta aplicación o quiera ver cómo funcione a nivel interno pueda hacerlo, desde Activitys (pantallas de la aplicación) hasta los tipos de resumen y/o funcionalidades más importantes.

En los siguientes apartados se van a explicar las funcionalidades de GPLSI Compendium App.

5.2.1 Compartir texto de internet con GPLSI Compendium App

Funcionalidad básica y sobre la que se construye la aplicación. Está pensada para cuando un usuario accede a una noticia de un periódico digital, que directamente pueda obtener el resumen y leerla y de esta forma no tener que leer toda la noticia si no lo desea.

GPLSI Compendium App se encargará de recoger el texto contenido en el HTML de la página web que el usuario proporcione. En un principio solamente funcionará con ***Marca.com, BBC.co.uk, AS.com, WhasingtonPost.com***, esto es porque cada web tiene una estructura diferente que hay que tratar, por ello si se intenta compartir un artículo de un periódico digital que no sea que no esté entre los mencionados anteriormente, el sistema informará al usuario de que no puede procesar ese texto ya que no se corresponde con los periódicos soportados por la aplicación.

Para extraer este texto se desarrollará un método que recoja el contenido HTML de la url que se proporcione, lo tratará y extraerá el texto de las etiquetas que corresponda en cada caso. Una vez se obtenga el texto se le mostrará al usuario para que pueda realizar los resúmenes que quiera.

Ejemplo con artículo de Marca.com

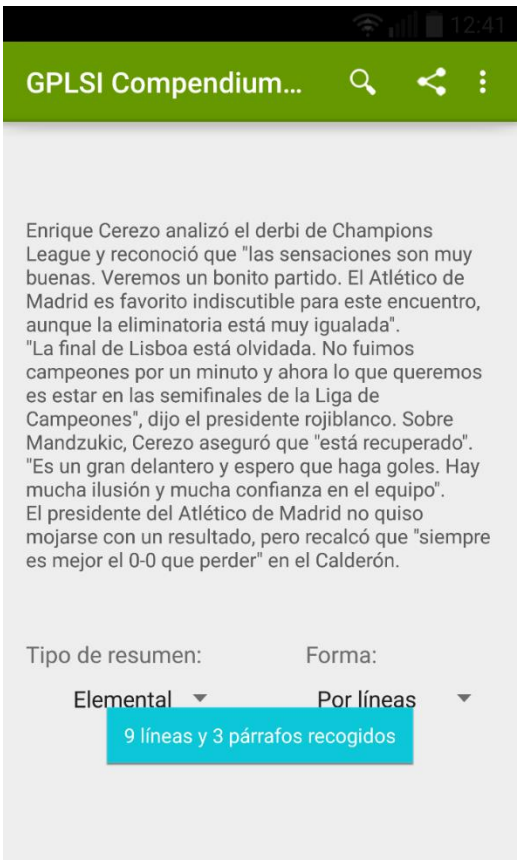


Figura 6: Recogida de datos de Marca.com

5.2.2 Tipos de resumen

GPLSI Compendium App en su versión inicial cuenta con seis métodos de resumen que se le proporcionaran al usuario de la misma. El procedimiento a realizar es el siguiente: el usuario podrá o bien compartir un artículo de un periódico digital como Marca o BBC, de tal forma que el sistema se lo recoja y lo introduzca para su posterior resumen, o bien el usuario copiará la url del artículo deseado y el sistema hará la misma operación que antes, de esta forma el texto quedará a disposición del usuario.

En la guía de desarrollo situada en el anexo de este documento se encontraran explicados los métodos de resumen desde el punto de vista del código realizado, con sus explicaciones pertinentes. El procesamiento que se hace del texto de los artículos es: descargar la página HTML, analizar su estructura y obtener los elementos relevantes asociados a la noticia. Esto se explica más detalladamente en la guía de desarrollo.

Cabe destacar que GPLSI Compendium App en principio sólo estará preparada para recoger artículos de cuatro periódicos digitales: *Marca.com, BBC.co.uk, AS.com, WhasingtonPost.com*. Esto se debe a que cada sitio web tiene su propia estructura, de forma que para extraer la información se tiene que realizar un análisis de la misma y tratar esa información. Esto no es objetivo principal de este proyecto, por ello no se han incorporado más periódicos, aunque en futuras versiones de la aplicación no se descarta esta opción.

Se han elegido estos periódicos para abarcar varios idiomas, como es el inglés y el español, y para llegar a noticias tanto deportivas como de cualquier tema. De este modo, en la versión inicial se abarcan varios idiomas y tipos de noticias diferentes.

Dicho esto, a continuación se van a explicar los tipos de resúmenes que contendrá esta aplicación, los cuales se dividen en resúmenes de longitud fija y resúmenes de longitud variable, donde el usuario podrá especificar el % de información relevante que quiere que recoja el resumen.

5.2.2.1 **Elemental**

Este tipo de resumen será el más sencillo, y consistirá en dejar como resumen del texto la primera línea del texto a resumir. Es de longitud fija.

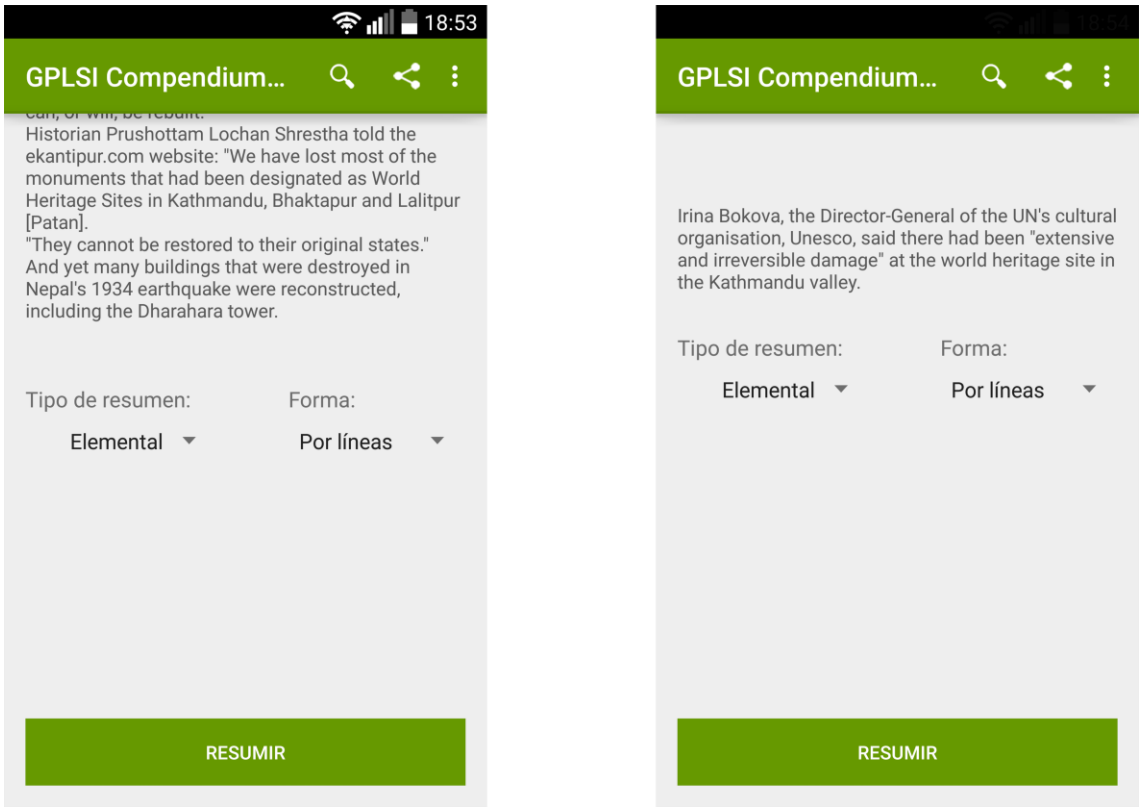


Figura 7: Ejemplo de resumen Elemental.

5.2.2.2 **Básico**

El resumen básico dejará como texto resumido la primera y la última línea del texto introducido por el usuario. Es de longitud fija.

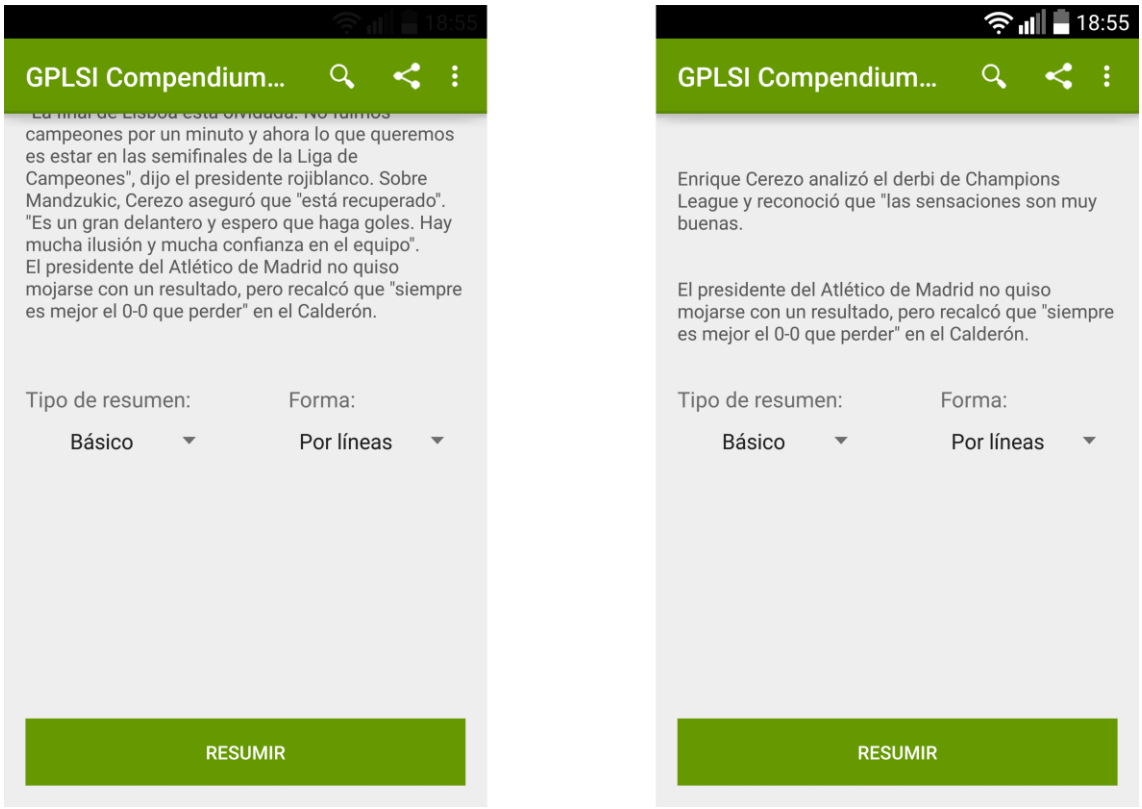


Figura 8: Ejemplo de resumen Básico.

5.2.2.3 **Avanzado**

Consistirá en mostrar las n primera líneas o párrafos del texto como resumen, para realizar esto se le proporcionará al usuario un selector de número que va de 0 a 100 en intervalos de 10, y representará el porcentaje de información que se desea como resumen. Por ejemplo, un valor de 50 para un texto de 20 líneas dará como resultado un texto resumido de las 10 primeras líneas. Es de longitud variable.

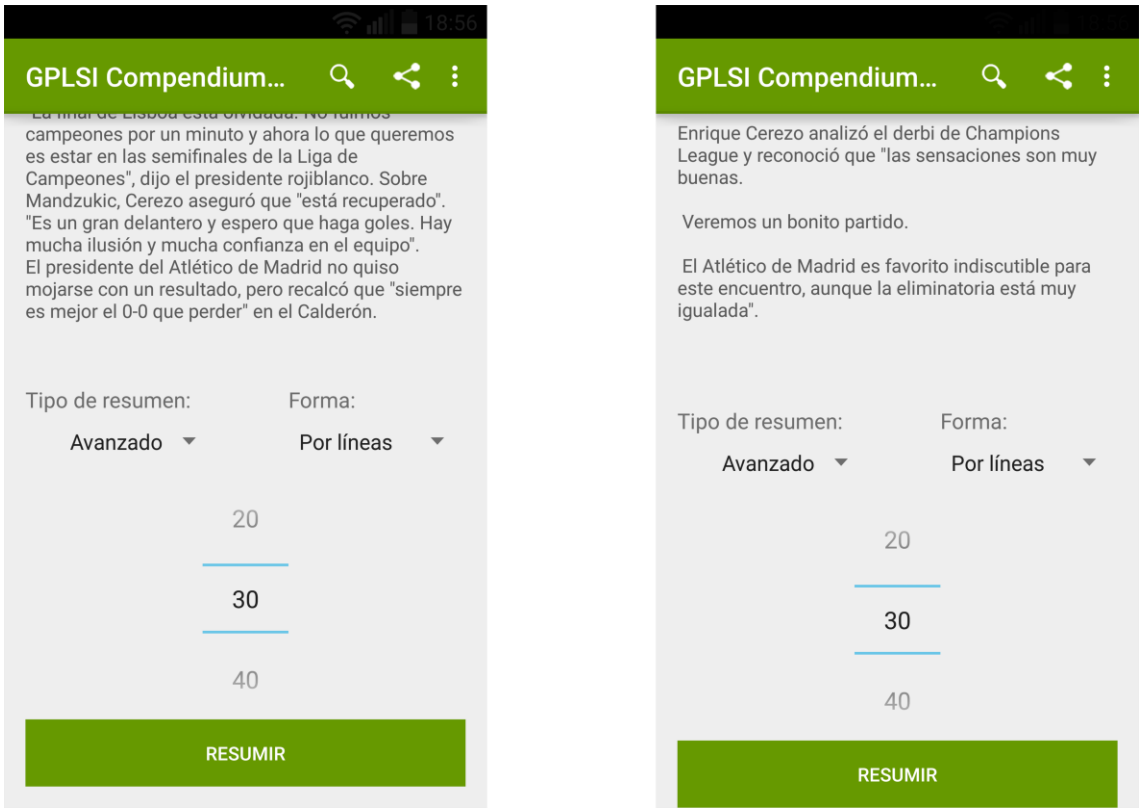


Figura 9: Ejemplo de resumen Avanzado.

5.2.2.4 **Experto**

Este tipo de resumen consistirá en analizar y procesar el texto y sacar como resumen la información más relevante en función del peso de cada palabra en el texto, esta funcionalidad se encuentra en el servicio web Compendium de la plataforma InTime. Se tendrá que elegir un porcentaje como en el resumen anterior ya que es de longitud variable.

Cabe destacar que este tipo de resumen será proporcionado por la herramienta GPLSI Compendium a través del método "topic".

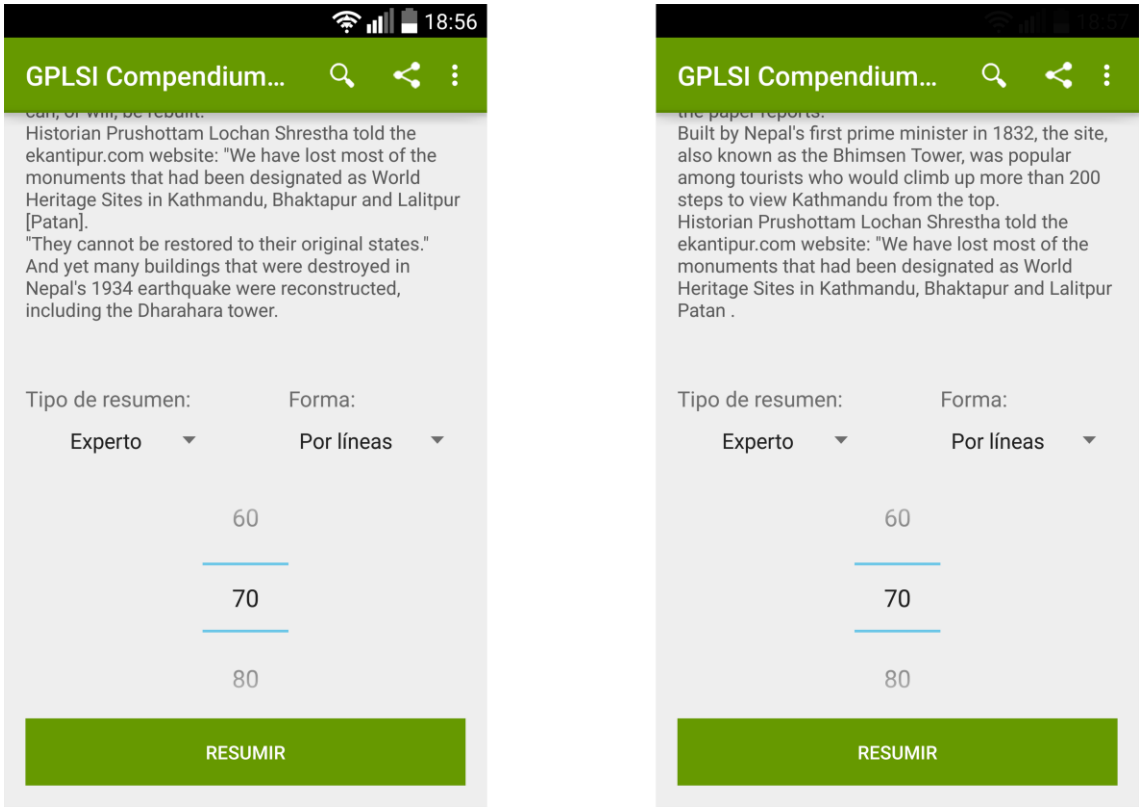


Figura 10: Ejemplo de resumen Experto.

5.2.2.5 **Inteligente**

Este tipo de resumen consistirá en analizar el texto y sacar como resumen la información más relevante en función del peso de cada palabra en el texto y la relación entre las mismas, esta funcionalidad se encuentra en el servicio web Compendium de la plataforma InTime. Se tendrá que elegir un porcentaje como en el resumen anterior ya que es de longitud variable.

Cabe destacar que este tipo de resumen será proporcionado por la herramienta GPLSI Compendium, a través del método "relevance + topic".

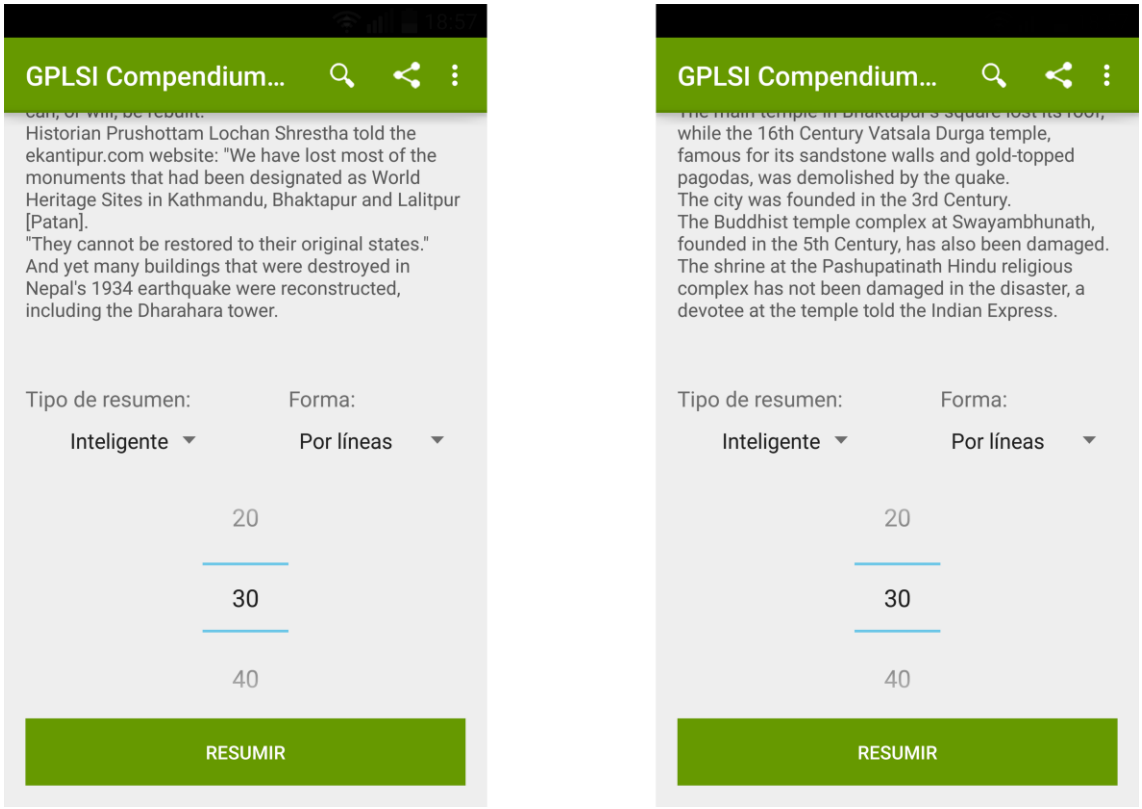


Figura 11: Ejemplo de resumen Inteligente.

5.2.2.6 **Manual**

Consistirá en que el usuario pueda realizar su propio resumen, para lo cual se segmentará el texto a resumir en líneas que serán separadas y el usuario podrá seleccionar las que guste. De esta forma se creará un resumen propio en base a sus necesidades.

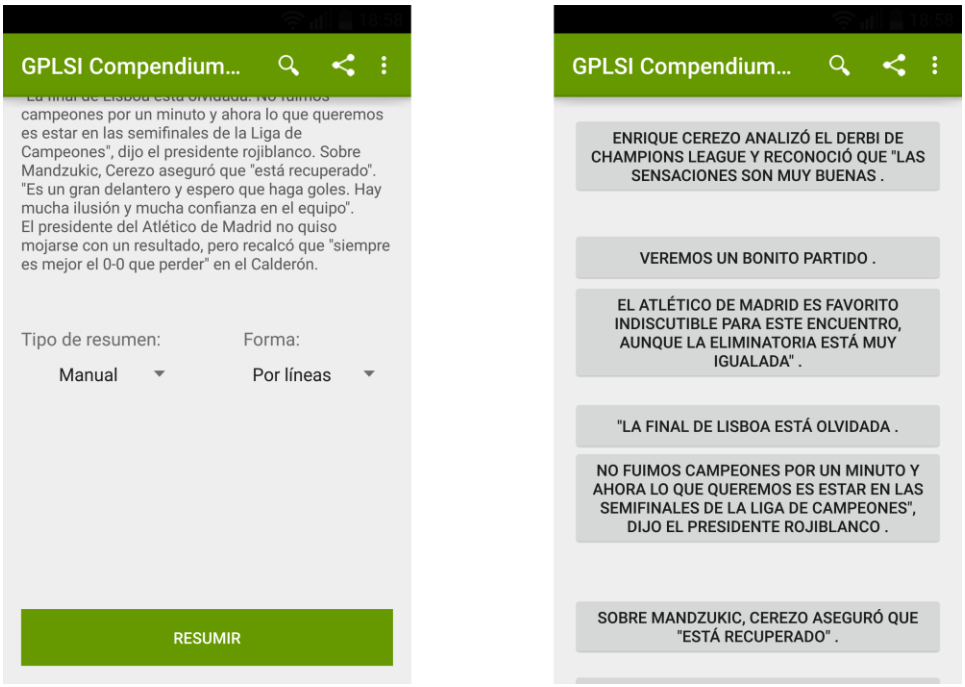


Figura 12: Ejemplo de resumen Manual.

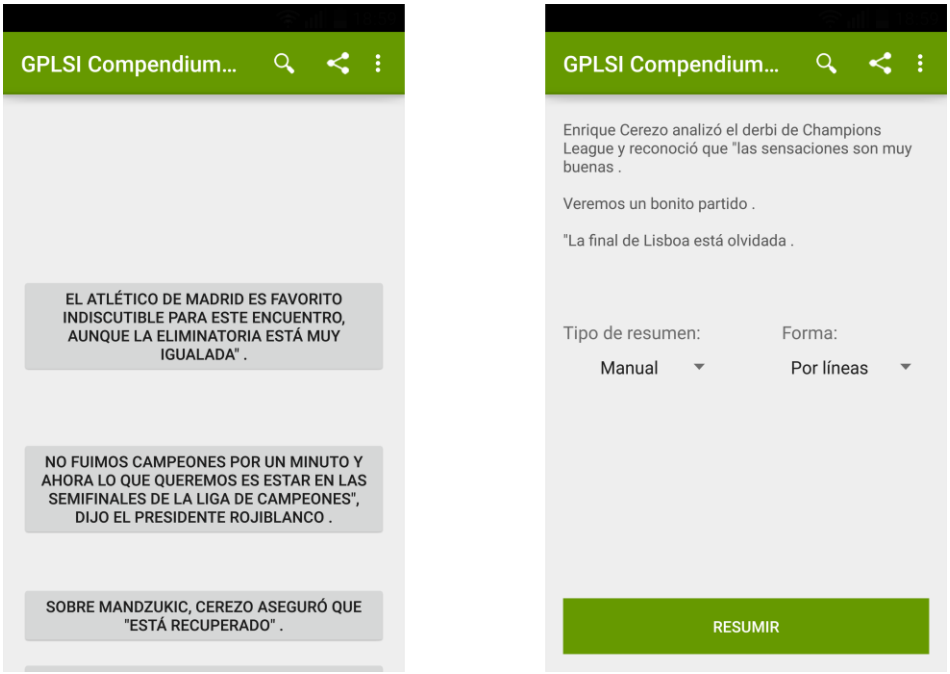


Figura 13: Ejemplo de resumen Manual 2.

5.2.3 Difundir resúmenes realizados en GPLSI Compendium App

Existirá un botón en el menú superior de la aplicación que permitirá compartir los textos que en ese momento se encuentren en GPLSI Compendium App con el resto de aplicaciones del dispositivo (Twitter, Gmail, WhatsApp…). Al seleccionar esa opción se desplegará una lista con las posibles opciones de compartir.

Remarcar que se creará para Twitter un sistema de forma que recorte el texto a compartir a 140 caracteres, ya que ese es el límite de caracteres para twittear.

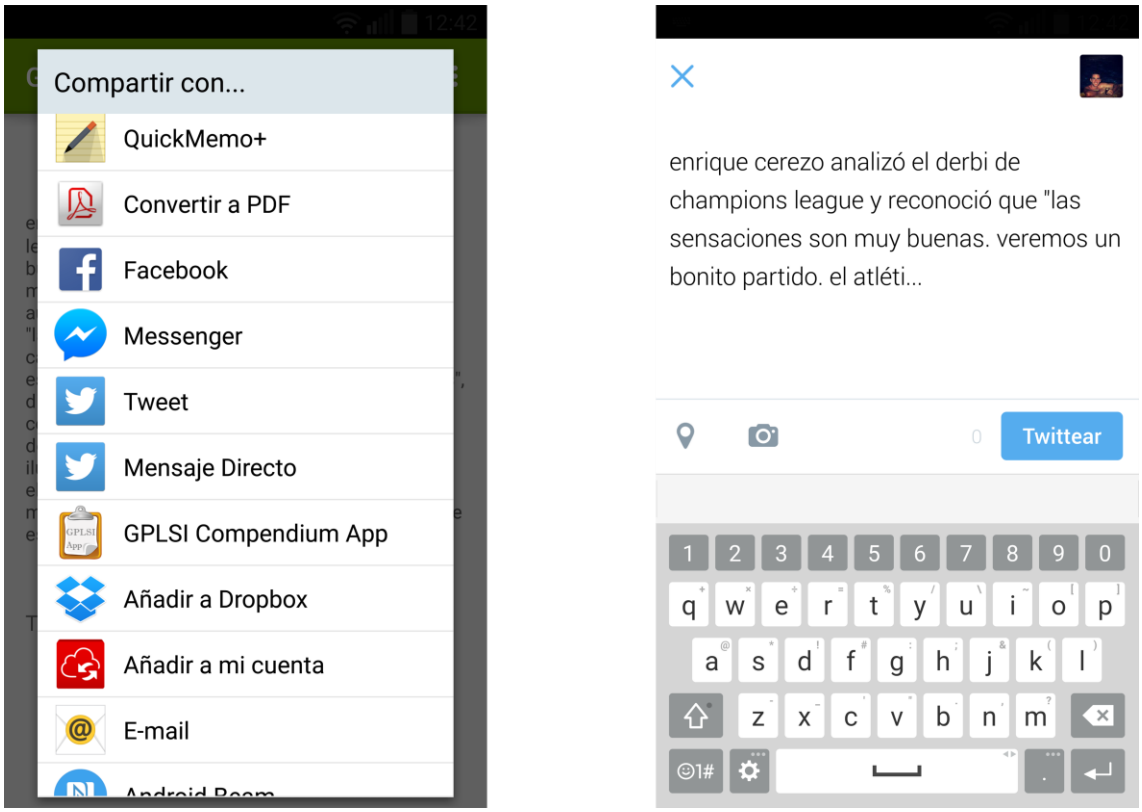


Figura 14: Compartir resumen con twitter.

5.2.4 Buscador de palabras

Se ha implementado un buscador de palabras que permite al usuario buscar palabras clave que puedan resultar importantes para el resumen que necesite. Esta opción se encontrará en el menú, de forma que cuando se seleccione se mostrará un cuadro para introducir el texto a buscar.

Cuando se vayan introduciendo caracteres, los que coincidan en el texto se irán remarcando de un color diferente y con otro tamaño.

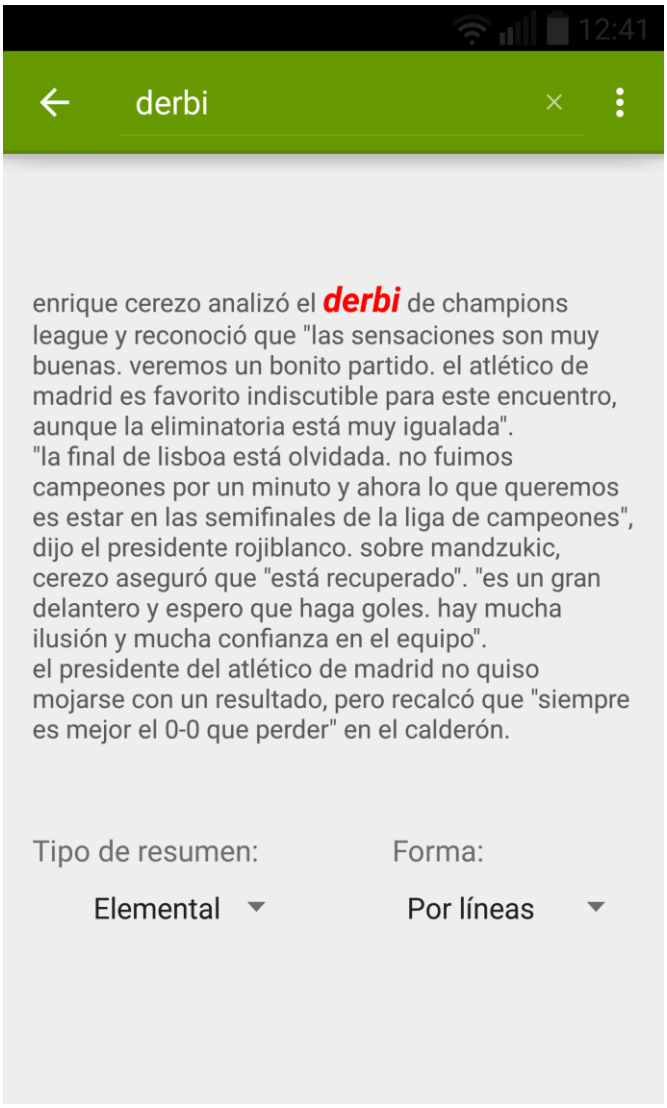


Figura 15: Buscador de palabras.

5.2.5 Añadir url

Existirá una opción de añadir url en el menú desplegable de la página principal de la aplicación donde se le mostrará al usuario un cuadro de diálogo donde podrá introducir la url del artículo que quiera resumir.

Una vez introducida el sistema recogerá el texto de esa dirección y se lo mostrará al usuario para su posterior resumen.

Como se ha comentado anteriormente el sistema está preparado para una serie de periódicos debido a la estructura de las webs, por ello si se inserta una url de un periódico digital que no sea **Marca.com, BBC.co.uk, AS.com, WhasingtonPost.com**, el sistema informará al usuario de que no puede procesar ese texto ya que no se corresponde con los periódicos soportados por la aplicación.

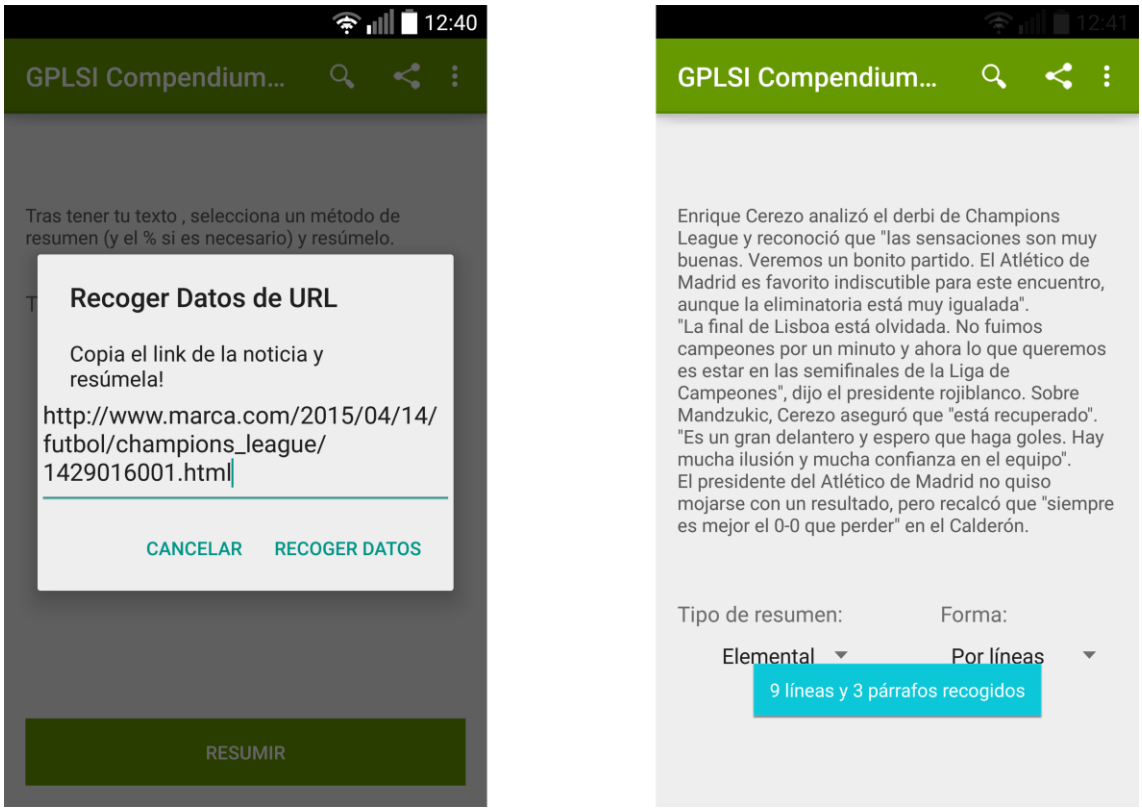


Figura 16: Añadir url y recogida de datos.

5.2.6 Copiar al portapapeles

GPLSI Compendium App pondrá a disposición del usuario copiar el contenido del texto que se encuentre en esos momentos al portapapeles del dispositivo Android.

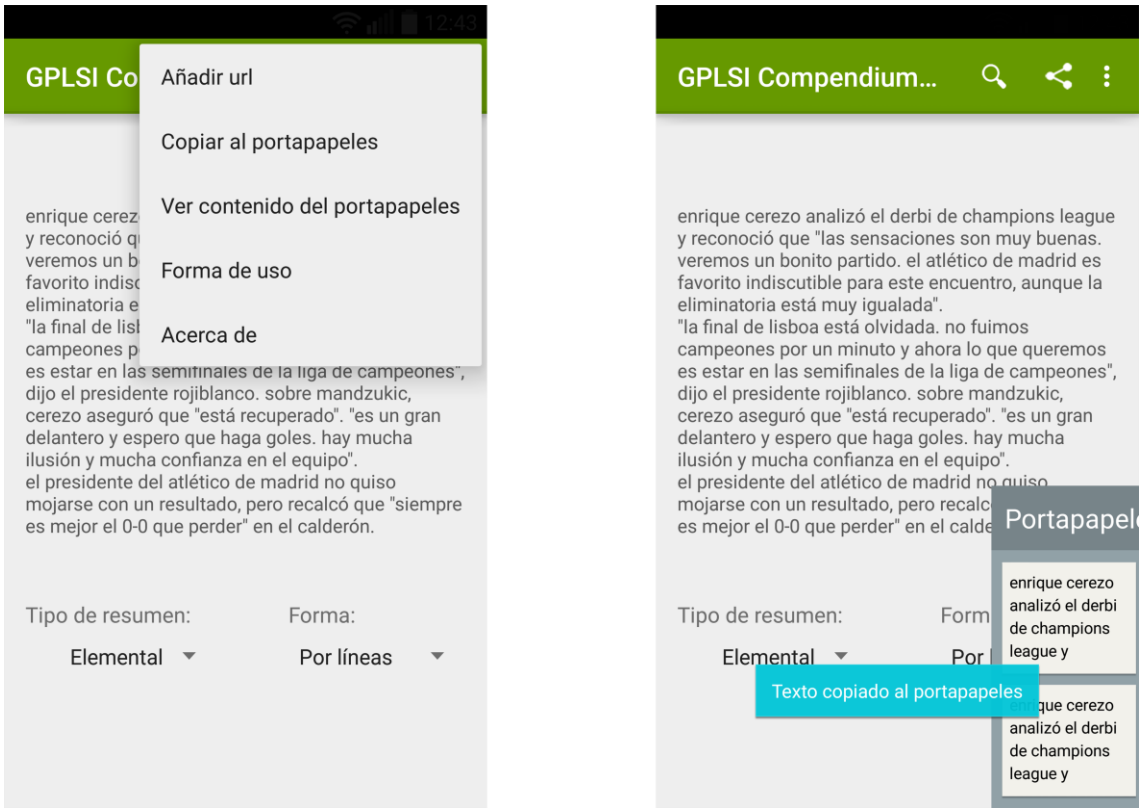


Figura 17: Copiar contenido al portapapeles.

5.2.7 Ver contenido del portapapeles

Así como se podrá copiar al portapapeles, también se podrá recuperar la información del mismo para mostrarla en la aplicación y poder resumirla si el usuario lo desea.

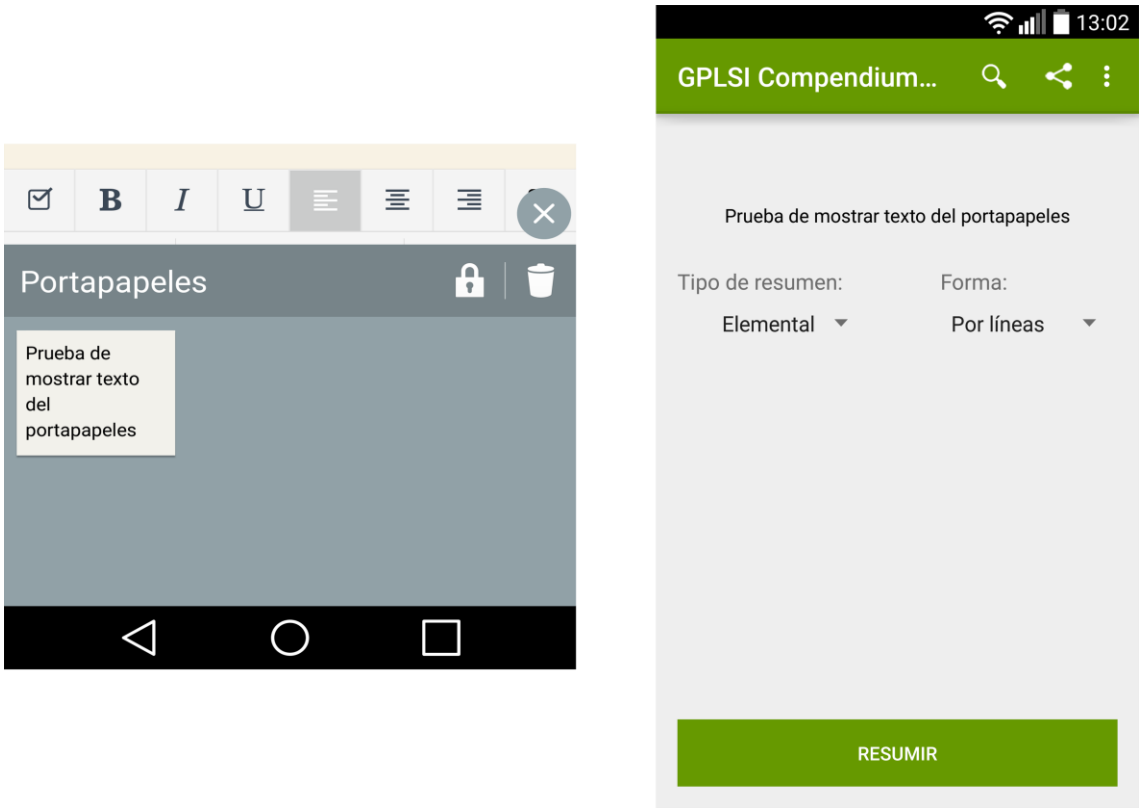


Figura 18: Ver contenido del portapapeles.

5.2.8 Guardar último método de resumen seleccionado

Por último, otra de las funcionalidades importantes que GPLSI Compendium App tiene es la de guardar la última configuración utilizada por el usuario.

Esto es importante para guardar las preferencias del usuario y que no tenga que cambiar siempre que entre a la aplicación el resumen que más adecuado le parezca. De esta forma se ayuda al usuario a controlar la aplicación y a sentirla más usable.

Se implementará con una activity llamada Configuración que recogerá el tipo de resumen seleccionado y lo almacenará, de forma que cuando el usuario vuelva a ejecutar la aplicación se comprobará si existe esa variable almacenada anteriormente, y si existe cambiará las opciones de resumen, poniendo por defecto la última utilizada.

5.3 Diseño de la aplicación

En este apartado se va a tratar el diseño del estilo para GPLSI Compendium App.

Se pretende crear un diseño atractivo y sencillo de cara al usuario, con tonos alegres.

Teniendo en cuenta esto, se va a utilizar una herramienta web que ayudará a la realización de este estilo, Android Action Bar Generator (Gilfelt, 2015). Con la ayuda de esta herramienta seleccionaremos los colores de los menús y partes más importantes en relación al estilo que queramos para nuestra aplicación. Una vez escogidos los colores se creará una carpeta donde se encuentran todos los recursos disponibles para incorporarlos a nuestra aplicación.

Una vez se tengan, se incorporarán al proyecto cambiando los archivos pertinentes, como es el fichero styles.xml y la inclusión de las imágenes a las distintas carpetas drawable.

Finalmente en el fichero AndroidManifest.xml se le indicará a la aplicación que deberá usar el tema que queramos, en este caso el realizado con la herramienta Android Action Bar Generator.

5.3.1 Diseño independiente del dispositivo

GPLSI Compendium App tendrá un diseño independiente del dispositivo, es decir, toda la información se mostrará siempre correctamente, como textos, menús, etcétera con independencia del dispositivo donde se ejecute, así como la orientación (vertical o horizontal).

Por ejemplo, se verá la información de la misma forma en un dispositivo con pantalla de 4 pulgadas que en uno de 6 pulgadas, sin pérdidas de información.

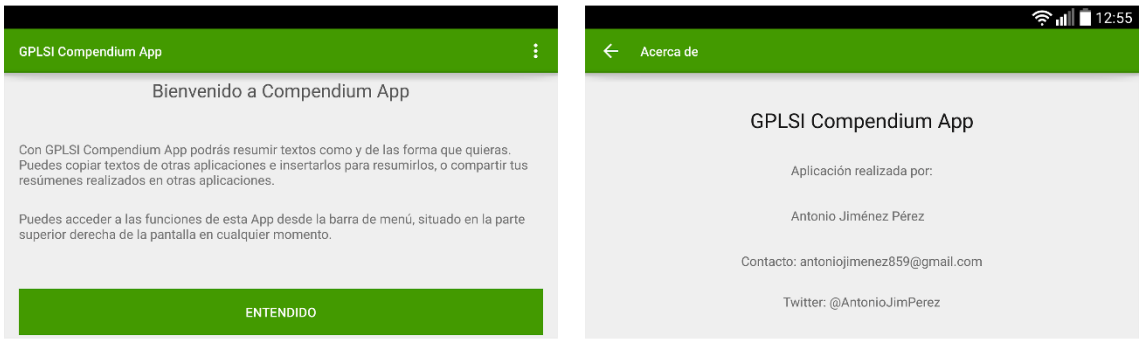


Figura 19: Pantallas en orientación horizontal.

5.3.2 Diseño del logo

Respecto al logo de la aplicación, se diseñará un logo simple como en la mayoría de apps que se encuentran en Google Play Store, un logo que a simple vista te diga de qué trata la aplicación.

Este logo se va a basar en colores blancos y marrones con texto grisáceo, será de forma cuadrada e irá en relación con el texto en general.

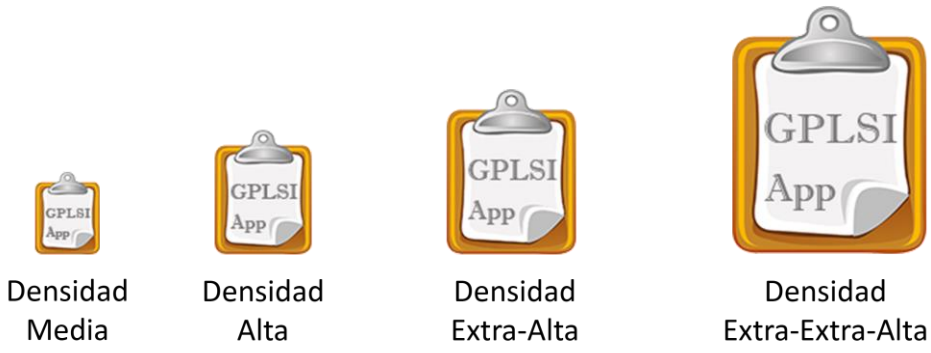


Figura 20: Logo para diferentes resoluciones de pantalla.

6 Pruebas y evaluación

Las pruebas constituyen un aspecto fundamental en el desarrollo de cualquier aplicación. Gracias a ellas se puede verificar el correcto funcionamiento del sistema. Hay que decir que las siguientes pruebas no entran a valorar si los resúmenes generados son más o menos correctos, ya que esto no era objetivo de este proyecto.

Se han realizado dos tipos de evaluación, una por parte de los usuarios, correspondiente a una evaluación relacionada con aspectos de la aplicación, como usabilidad y facilidad de uso, y otra por parte de Google Play Store, relacionada con estadísticas como las instalaciones diarias de GPLSI Compendium App.

6.1 Encuesta de usuarios

Con el objetivo de obtener feedback por parte de los usuarios de la aplicación y mejorar aspectos de la misma, se ha diseñado un formulario con Google Forms con 9 preguntas y se ha contactado con usuarios de la aplicación para pedirle que lo rellenen. Además se ha dado la opción de añadir comentarios sobre su experiencia de uso.

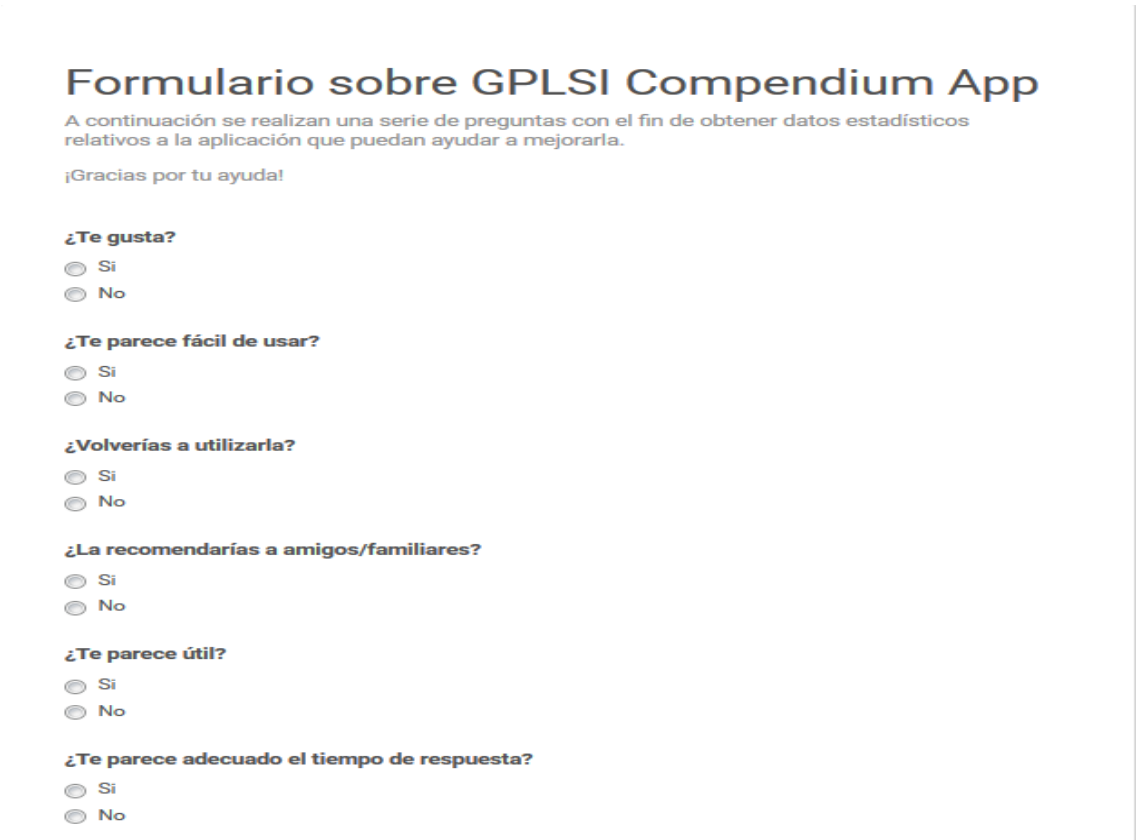


Figura 21: Formulario sobre GPLSI Compendium App

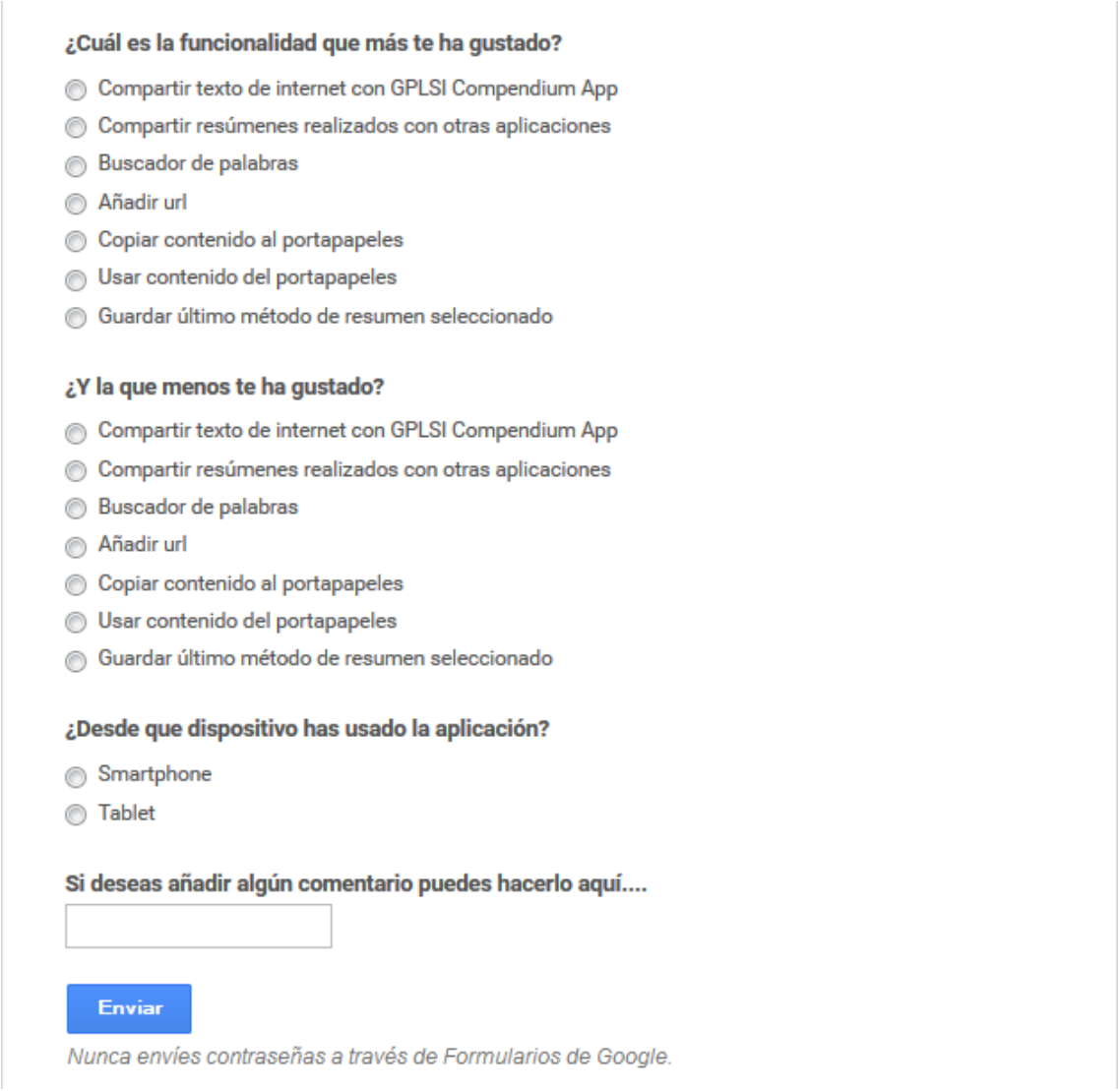


Figura 22: Formulario sobre GPLSI Compendium App. Parte 2

Las impresiones acerca de la evaluación de 24 usuarios de distinto perfil han sido en líneas generales buenas, como a continuación se demostrará. Atendiendo a estas evaluaciones, se puede observar que la funcionalidad que más ha gustado ha sido la de Compartir texto de internet con GPLSI Compendium App, funcionalidad sobre la que se construye la aplicación como ya se mencionó anteriormente en el apartado 5.2.1 Compartir texto de internet con GPLSI Compendium App, del mismo modo, en el apartado de funcionalidad que menos no destaca ninguna sobre el resto, si no que varias personas han elegido la misma en diferentes casos. En las siguientes imágenes se puede ver el resultado de la evaluación por parte de los usuarios:

	A	B	C	D	E	F	G
1	Marca temporal	¿Te gusta?	¿Te parece fácil de usar?	¿Volverías a utilizarla?	¿La recomendarías a ami	¿Te parece útil?	¿Cuál es la funcionalidad que más te ha gustado?
2	15/06/2015 17:10:56	Si	Si	Si	Si	Si	Compartir resúmenes realizados con otras aplicaciones
3	15/06/2015 17:13:11	Si	No	Si	Si	Si	Compartir resúmenes realizados con otras aplicaciones
4	15/06/2015 17:14:01	Si	Si	Si	Si	Si	Compartir resúmenes realizados con otras aplicaciones
5	15/06/2015 17:14:37	Si	Si	Si	Si	Si	Compartir resúmenes realizados con otras aplicaciones
6	15/06/2015 17:15:05	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
7	15/06/2015 17:18:34	Si	Si	Si	Si	Si	Copiar contenido al portapapeles
8	15/06/2015 17:20:45	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
9	15/06/2015 17:20:57	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
10	15/06/2015 17:26:19	Si	Si	Si	Si	Si	Buscador de palabras
11	15/06/2015 17:27:21	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
12	15/06/2015 17:28:57	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
13	15/06/2015 17:29:02	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
14	15/06/2015 17:29:11	Si	Si	Si	Si	Si	Buscador de palabras
15	15/06/2015 18:15:42	Si	Si	Si	Si	Si	Buscador de palabras
16	15/06/2015 19:05:46	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
17	15/06/2015 19:08:11	Si	Si	Si	Si	Si	Guardar último método de resumen seleccionado
18	15/06/2015 19:20:03	Si	Si	Si	Si	Si	Usar contenido del portapapeles
19	15/06/2015 19:22:28	Si	Si	Si	Si	Si	Compartir resúmenes realizados con otras aplicaciones
20	15/06/2015 19:25:24	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
21	15/06/2015 20:36:09	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
22	15/06/2015 20:36:50	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
23	15/06/2015 20:45:41	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
24	15/06/2015 22:52:25	Si	Si	Si	Si	Si	Guardar último método de resumen seleccionado
25	16/06/2015 13:23:48	Si	Si	Si	Si	Si	Compartir texto de internet con GPLSI Compendium App
26							

Figura 23: Resultado de la evaluación

	H	I	J	K	L	M
1	¿Y la que menos te ha gustado?	Si deseas añadir algún comentario puedes hacerlo aquí....	Si deseas añadir algún co	¿Te parece adecuado el tiempo de respuesta?		
2	Añadir url		Smartphone	Si		
3	Añadir url		Smartphone	No		
4	Guardar último método de resumen seleccionado		Smartphone	Si		
5	Buscador de palabras	Muy útil	Smartphone	Si		
6	Buscador de palabras		Smartphone	Si		
7	Compartir resúmenes realizados con otras aplicaciones		Smartphone	Si		
8	Guardar último método de resumen seleccionado		Smartphone	Si		
9	Guardar último método de resumen seleccionado		Smartphone	Si		
10	Buscador de palabras		Smartphone	Si		
11	Buscador de palabras	Muy útil para trabajar	Smartphone	Si		
12	Buscador de palabras	Muy útil para trabajar y fácil	Smartphone	Si		
13	Buscador de palabras	Muy útil para trabajar y fácil	Smartphone	Si		
14	Añadir url		Smartphone	Si		
15	Guardar último método de resumen seleccionado		Smartphone	Si		
16	Copiar contenido al portapapeles	La aplicación es genial! La recomiendo👍🏻 👍🏻 👍🏻 👍🏻	Smartphone	Si		
17	Añadir url	Muy util,recomendable :)	Smartphone	Si		
18	Guardar último método de resumen seleccionado		Smartphone	Si		
19		Muy práctico	Smartphone	Si		
20	Compartir resúmenes realizados con otras aplicaciones		Smartphone	Si		
21	Usar contenido del portapapeles	Me parece una aplicación excelente de primera	Smartphone	Si		
22	Usar contenido del portapapeles	Me parece una aplicación excelente de primera	Smartphone	Si		
23	Usar contenido del portapapeles	De primera excelente	Smartphone	Si		
24	Añadir url	Me ha parecido muy interesante y muy buen trabajo. Además resulta muy útil.	Smartphone	Si		
25	Añadir url		Smartphone	Si		
26						

Figura 24: Resultado de la evaluación. Parte 2

En las siguientes imágenes se verán los resultados en formato gráfico de las preguntas realizadas en el formulario cuya respuesta podía ser si/no:

PREGUNTA	SI	NO
¿Te gusta?	100,00%	0,00%
¿Te parece fácil de usar?	95,83%	4,17%
¿Volverías a utilizarla?	100,00%	0,00%
¿La recomendarías a amigos/familiares?	100,00%	0,00%
¿Te parece útil?	100,00%	0,00%
¿Te parece adecuado el tiempo de respuesta?	95,83%	4,17%

Figura 25: Estadística de preguntas si/no

Como se puede observar, en casi todas las preguntas la totalidad de los usuarios han dado valoración positiva, excepto en la relacionada con la usabilidad del producto y la del tiempo de respuesta, en la que existe alguna opinión negativa. Con todo esto se han obtenido evaluaciones positivas en líneas generales, como se comentaba anteriormente.

A partir de esta evaluación, se incluyó en la pantalla principal de la aplicación un cuadro de texto para insertar texto a resumir y de esta forma mejorar la usabilidad de GPLSI Compendium App.

6.2 Indicadores Google Play Store

Al subir la aplicación a Google Play Store, esta misma herramienta proporciona indicadores o gráficas que pueden ayudar a realizar un análisis del impacto que está teniendo tu app de cara a los usuarios.

Por el momento, el número de descargas tras 16 días en el sitio es de 30, como se puede observar en la imagen:

Figura 26: Número de descargas actuales

Por otra parte, en la parte de estadísticas relativas a la aplicación, podemos encontrar indicadores de diversos factores, como este, que plasma las desinstalaciones diarias por dispositivo:

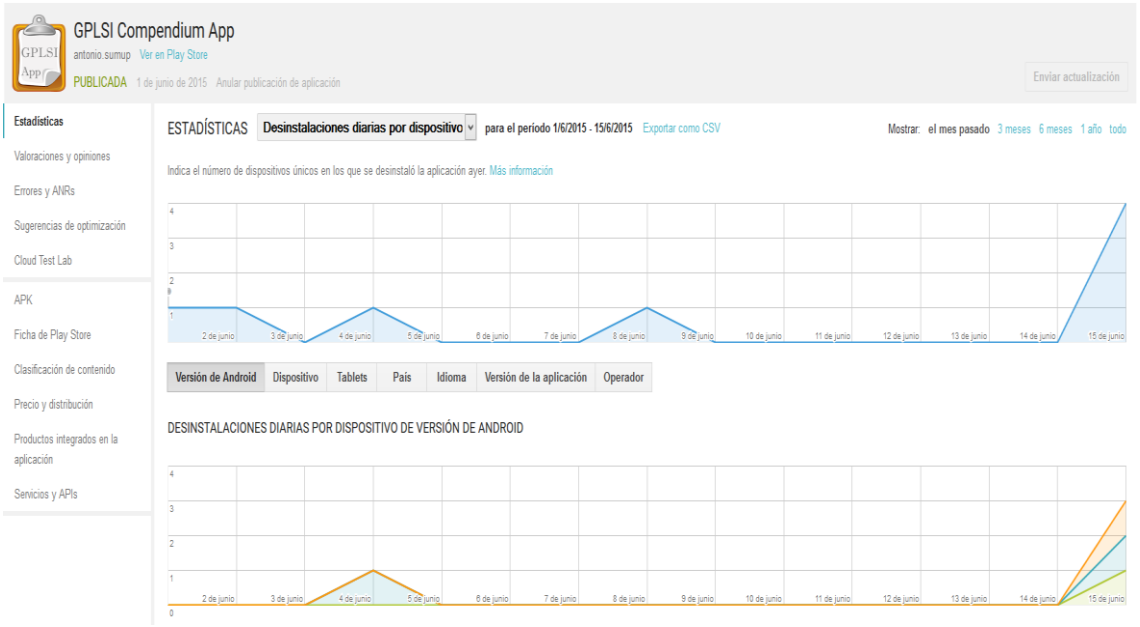


Figura 27: Indicador de desinstalaciones diarias por dispositivo.

Atendiendo a este gráfico podemos observar algunos picos de desinstalaciones en ciertos días, quizás provocados por usuarios que probaron la aplicación y posteriormente por falta de espacio tuvieron que desinstalarla, o porque la aplicación no fuera de su interés y sólo quisieran probarla por curiosidad.

Todos estos indicadores ayudarán en el análisis y posteriores propuesta de mejora para la aplicación.

7 Valoración personal

La sociedad de hoy día no podría vivir sin los servicios web que se ofrecen con contenidos digitales, y por ello, la decisión de realizar un proyecto enfocado a la gestión y difusión de contenidos digitales me pareció lo más apropiado.

Otro aspecto a destacar para la realización de este proyecto fue que era para plataforma móvil (Android).

También suponía un desafío la idea de desarrollar una aplicación por cuenta propia y sin ayuda de un equipo de desarrollo, como hace años siempre se hacía. El realizar una aplicación y "pelearte con ella" día tras día, te hace, adquirir muchísimos conocimientos, y esa experiencia no tiene precio.

Por otra parte, he adquirido conocimientos en el campo de la programación para dispositivos Android, en la estructura de las aplicaciones, de cómo se gestionan los permisos de las aplicaciones... y técnicas de extracción de información para la posterior generación de resúmenes automáticos. Considero todos estos conocimientos adquiridos fundamentales para mi futuro, ya que la recogida de información es una tarea actualmente muy demandada, las empresas buscan la recogida de datos para poder analizar y saber dónde enfocar sus esfuerzos de mejora.

Por todo esto se puedo decir que ha supuesto un orgullo enorme realizar este proyecto, y que de cara al futuro puede ayudar y proporcionar mucha confianza el saber hasta dónde se ha llegado en este trabajo de fin de grado.

8 Conclusiones y trabajo futuro

Como primera idea se ha de decir que se han cumplido todos los objetivos propuestos inicialmente, por lo que se puede decir que se ha creado la aplicación con éxito.

A continuación se van explicar en el apartado 8.1 Resultado del proyecto el resultado de los objetivos conseguidos al realizar este proyecto, así como las conclusiones finales relativas al aprendizaje en la realización de este proyecto. Para terminar en este mismo apartado se analizará el trabajo futuro a desarrollar con la aplicación, en base al análisis de los usuarios en el formulario.

8.1 Resultado del proyecto

En este apartado se va a hacer un repaso del resultado de los objetivos iniciales propuestos para el proyecto, así como un repaso de los resultados del desarrollo del mismo.

8.1.1 Resultado de objetivos iniciales propuestos

Como se ha dicho anteriormente, se han alcanzado todos los objetivos propuestos inicialmente en el apartado 3. Objetivos.

A continuación se van a explicar estos objetivos:

- ***Crear una aplicación para plataforma móvil (Android):*** se ha conseguido crear una aplicación para Android completa y funcional que puede ser puesta a disposición de los usuarios a través de la plataforma Google Play Store.

- ***Realizar una aplicación Android compatible con la mayoría de dispositivos que existen en el mercado actualmente:*** se ha trabajado en el diseño independiente del dispositivo y se ha conseguido que funcione para múltiples tamaños de pantalla, y para ambas orientaciones (vertical y horizontal).

- ***Enfrentarse a herramientas no vistas ni usadas anteriormente y aprender a manejarlas y aplicarlas al proyecto:*** se ha conseguido dominar herramientas como Android Studio para la realización de la aplicación de forma satisfactoria.

- ***Capacidad de organización a la hora de desarrollar el proyecto paralelamente a la realización del curso de la titulación:*** se ha conseguido compaginar la realización del proyecto con la del curso, de forma que se ha conseguido sacar hacia delante el curso y el proyecto antes de lo planificado.

- ***Investigación de técnicas de procesamiento de lenguaje natural (PLN) y generación de resúmenes automáticos, junto con su posterior implementación en el proyecto:*** se han investigado y aprendido técnicas de procesamiento de lenguaje natural, para implementarlas en la aplicación a modo de métodos de resumen automáticos.

- ***Abarcar y realizar las distintas fases en el desarrollo de un proyecto de importancia, como son la investigación, planificación y posterior desarrollo del mismo:*** objetivo fundamental que se ha alcanzado de forma óptima, se ha realizado el proyecto en menos tiempo del esperado, se ha conseguido aprender de las investigaciones realizadas y se ha realizado un desarrollo completo del mismo, dando como resultado una aplicación funcional y completa.

- ***Integrar técnicas de generación de resúmenes en aplicación real:*** se ha conseguido realizar hasta seis métodos de resumen automático en la primera versión de GPLSI Compendium App.

8.1.2 Resultado del desarrollo del proyecto

Las conclusiones a destacar al desarrollar este proyecto son las siguientes:

- ***Adquirir conocimientos:*** se han adquirido cantidad conocimientos de realización de aplicaciones para el sistema Android, desde cómo manejar los permisos de la aplicación hasta la implementación de funcionalidades como el buscador o la de compartir textos. También se han aprendido técnicas de tratamiento de texto y tipos de resúmenes para el mismo, fundamental para el desarrollo de esta aplicación.

- ***Demostrar capacidades adquiridas:*** este proyecto ha sido un buen ejemplo para demostrar capacidades adquiridas durante la realización de esta titulación.

- ***Mejorar la aplicación:*** siempre se puede mejorar, añadiendo nuevas funcionalidades o refinando aún más si cabe las que ya se tienen, pero no queda ninguna pena por el resultado final de la aplicación.

8.2 Trabajo futuro

Al observar los datos de evaluación por parte de los usuarios en el apartado 6.1 Encuesta de usuarios, podemos pensar en mejorar la aplicación en la parte de usabilidad y tiempo de respuesta, esto se hará en futuras versiones de GPLSI Compendium App.

Por otra parte, analizando periódicamente los indicadores que proporciona Google Play Store para desarrolladores, se pueden sacar conclusiones y enfocar esfuerzos a otros aspectos de la aplicación, y de este modo ir haciendo cada vez más completa y eficaz GPLSI Compendium App.

9 Bibliografía y referencias

Albes, Luis. 2013. Action Bar y Tabs en Android. Recuperado el 6 de Enero de 2015, del Sitio web de El baúl de Android: http://elbauldeandroid.blogspot.com.es/2013/10/actionbar-android-en-construccion.html#cuatro

Grado en Ingeniería Multimedia. (n.d.). Recuperado el 30 de Noviembre de 2014, del Sitio web de la Escuela Politécnica Superior de la Universidad de Alicante: http://www.eps.ua.es/es/ingenieria-multimedia/presentacion.html

Francho. 2011. Truco Android: menú compartir. Recuperado el 15 de Enero de 2015, del Sitio web de Francho(Lab): http://francho.org/2010/04/29/truco-android-menu-compartir/

Pereira García, José Manuel. 2012. Consumir datos de una URL en Android. Recuperado el 5 de Diciembre de 2014, del Sitio web de Androcode: http://androcode.es/2012/05/consumir-datos-de-una-url-en-android/

Receiving Simple Data from Other Apps. (n.d.). Recuperado el 10 de Enero de 2015, del Sitio web de Android Developers: http://developer.android.com/intl/es/training/sharing/receive.html

Gilfelt, Jeff. 2012. Android Action Bar Style Generator. Recuperado el 18 de Marzo de 2015, del Sitio web de jgilfelt: http://jgilfelt.github.io/android-actionbarstylegenerator/

Ferrer Delgado, Daniel. 2011. Cómo aplicar estilos a los controles de Android. Recuperado el 15 de Marzo de 2015, del Sitio web de nosinmiubuntu: http://www.nosinmiubuntu.com/como-aplicar-estilos-los-controles-de/

Leiva Gordillo, Antonio. 2013. Búsqueda en la ActionBar. Recuperado el 19 de Febrero de 2015, del Sitio web de LimeCreativeLabs: http://www.limecreativelabs.com/busqueda-en-la-actionbar-como-implementarla/

Ferrer Delgado, Daniel. 2011. Cómo guardar datos en Android. Recuperado el 5 de Abril de 2015, del Sitio web de nosinmiubuntu: http://www.nosinmiubuntu.com/como-guardar-datos-en-android/

2012. Botones dinámicos. (n.d.). Recuperado el 25 de Marzo de 2015, del Sitio web de stackoverflow: http://stackoverflow.com/questions/7195056/how-do-i-programmatically-add-buttons-into-layout-one-by-one-in-several-lines

IDC. 2015. Venta de smartphones por sistema operativo de 2010 a 2014, (n.d.). Recuperado el 13 de Junio de 2015, del Sitio web de idc: http://www.idc.com/getdoc.jsp?containerId=prUS25450615

GPLSI Compendium, (n.d.). Recuperado el 17 de Enero de 2015, del Sitio web de intime: http://intime.dlsi.ua.es:8080/compendium/

Duran Cruz, Laura. 2014. Sistemas de gestión de contenidos. Recuperado el 10 de Diciembre de 2014, del sitio web de Slideshare: http://es.slideshare.net/laurayasmindurancruz/sistemas-de-gestin-de-contenidos-content-management-system-o-cms

Hernandez, Alberto. 2014. Sistemas de gestión de contenidos: Historia y evolución. Recuperado el 14 de Diciembre de 2014, del sitio web de OCCFormacion: http://www.cursodemarketingdigitalocc.es/sistemas-de-gestion-de-contenidos-historia-y-su-evolucion/

Sistemas de gestión de contenidos. (n.d.). Recuperado el 15 de Enero de 2015, del sitio web de Wikipedia: https://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_contenidos

Lloret, Elena. 2011. Text summarisation based on human language technologies and its applications. Tesis Doctoral, Universidad de Alicante.

Moreno Boronat, Lidia. 1999. Introducción al procesamiento del lenguaje natural. Universidad de Alicante.

Blagdon, Jeff. 2013. Yahoo's latest Android app integrates Summly for faster news scanning. Recuperado el 17 de Junio de 2015, del sitio web de The Verge: http://www.theverge.com/2013/4/30/4285068/yahoo-android-app-integrates-summly-for-faster-news-scanning

10 Anexos

10.1 Guía de desarrollo

Introducción

GPLSI Compendium App está desarrollada para el trabajo de fin de grado de Ingeniería Multimedia del alumno Antonio Jiménez Pérez (ajp19@alu.ua.es), en la Universidad de Alicante, bajo la supervisión y tutorización de Elena Lloret Pastor y José Manuel Gómez Soriano.

La aplicación se basa en el resumen de textos pasados por el usuario, y está destinada a resumen de noticias de periódicos y artículos de webs como BBC, Marca, etc…

Los métodos de resumen que proporciona GPLSI Compendium App corresponden a la plataforma Compendium y a otros desarrollados por el creador de la aplicación.

El usuario podrá compartir los artículos que le interesen con la aplicación y seleccionar el método de resumen que más le parezca, así como compartir en las redes sociales o guardar los resúmenes que realice en la aplicación.

La aplicación se ha desarrollado con Android Studio en su totalidad, mientras que para el diseño se ha utilizado la herramienta web Android Action Bar Style Generator (http://jgilfelt.github.io/android-actionbarstylegenerator/). Aparte de esto se ha tenido acceso a la herramienta GPLSI Compendium (http://intime.dlsi.ua.es:8080/compendium/) para poder realizar los tipos de resumen Experto e Inteligente.

Activitys

Las activitys en Android son la base de la aplicación. Cada activity se puede comparar con cada una de las pantallas de la aplicación, y cada una consta del layout (interfaz de usuario), y la activity.java, donde se implementan los métodos y funcionalidades de los componentes de esa interfaz.

GPLSI Compendium App consta de cinco activitys, de las cuales una es la que lleva todo el peso de la funcionalidad en la app:

1. **AcercaDe**

   En esta actividad se encuentra la información sobre el desarrollador y enlaces al correo del mismo por si algún usuario tiene alguna duda o quiere contactar con él.

2. **Configuración**

   Actividad encargada de guardar la última opción de resumen seleccionada por el usuario.

3. **FormaUso**

   En esta actividad se describe de forma detallada los métodos de resumen y cómo funciona cada uno para que el usuario pueda seleccionar el que más le guste, y se explican las demás funciones que puede realizar la aplicación (p.e. el uso de las opciones del menú superior).

4. **Inicio**

   Actividad con la que se inicia la aplicación, en ella se presenta un pequeño tutorial o resumen de lo que la aplicación es capaz de hacer.

5. **MainActivity**

   Actividad más importante dentro de la aplicación, en ella se encuentran implementados todos los métodos de resumen, así como las demás funcionalidades que se explicarán a continuación.

Funcionalidades

A continuación se van a detallar las funcionalidades más importantes y se explicará el proceso de desarrollo de cada una de ellas:

En primer lugar, para el desarrollo de los tipos de resumen se crean dos tareas asíncronas:

1. **AsyncTaskRunner (Compendium)**

   Esta tarea asíncrona se encarga de realizar la llamada a compendium, y los métodos de resumen que se derivan de la misma son el ***Experto*** y el ***Inteligente***.

A ella se le pasa por parámetro el método de resumen y el porcentaje de resumen deseado, y se realiza una petición POST al servicio web de compendium (http://intime.dlsi.ua.es:8080/compendium/) pasándole como “body” en texto a resumir.

```
private class AsyncTaskRunner extends AsyncTask<String, String, String> {

    private String resp;
    private ProgressDialog progress = new ProgressDialog(MainActivity.this);
    private String tipo;
    private int porcen;

    public AsyncTaskRunner(String t, int n){

        tipo=t;
        porcen=n;

        String fin = "http://gplsi.dlsi.ua.es:80/services/pln/rest/v1/gplsi/compendium/"+tipo+"/"+porcen;
    }

    @Override
    protected String doInBackground(String... params) {

        try {

             HttpClient httpclient = new DefaultHttpClient();
            /*Creamos el objeto de HttpClient que nos permitira conectarnos mediante peticiones http*/
              //HttpPost httppost = new HttpPost("http://gplsi.dlsi.ua.es:80/services/pln/rest/v1/gplsi/compendium/n/30");
            HttpPost httppost = new HttpPost("http://gplsi.dlsi.ua.es:80/services/pln/rest/v1/gplsi/compendium/"+tipo+"/"+porcen);

            StringEntity se = new StringEntity(editText.getText().toString());

            /*El objeto HttpPost permite que enviemos una peticion de tipo POST a una URL especificada*/
                 //AÑADIR PARAMETROS
              List<NameValuePair> parametros = new ArrayList<>();
            parametros.add(new BasicNameValuePair("body",editText.getText().toString()));
            //parametros.add(new BasicNameValuePair("info","Otro mensaje"));
               /*Una vez añadidos los parametros actualizamos la entidad de httppost, esto quiere decir en pocas palabras anexamos los

            //httppost.setEntity(new UrlEncodedFormEntity(parametros));
            httppost.setEntity(se);

                        /*Finalmente ejecutamos enviando la info al server*/
              HttpResponse resp = httpclient.execute(httppost);
                 HttpEntity ent = resp.getEntity();/*y obtenemos una respuesta*/
                             String text = EntityUtils.toString(ent);

                 return text;

          }catch(Exception e) {
               return "error";
          }
    }
}
```

Figura 28: Código desarrollado para AsyncTaskRunner.

```
    @Override
    protected void onPostExecute(String result) {
        // execution of result of Long time consuming operation
        // In this example it is the return value from the web service
        if (progress.isShowing()) {
            progress.dismiss();
        }
        editText.setText(result);
        //textView.setText(request.getName());
    }


    @Override
    protected void onPreExecute() {
        // Things to be done before execution of long running operation. For
        // example showing ProgessDialog
        progress.setMessage("Espere unos instantes");
        progress.setTitle("Resumiendo...");
        progress.show();
    }


    @Override
    protected void onProgressUpdate(String... text) {
        editText.setText(text[0]);
        // Things to be done while execution of long running operation is in
        // progress. For example updating ProgessDialog
    }
}
```

Figura 29: Código desarrollado para AsyncTaskRunner. Parte 2.

2. **AsyncTaskRunner2 (métodos de Desarrollador)**

   Esta tarea asíncrona se encarga de recoger el html del artículo de donde proviene la noticia, lo trata y en función de la web que se trate (que se sabe mirando la url pasada) se miran unas etiquetas html u otras para extraer el texto de la noticia y no otra información no relevante como los créditos o el pie de página. Los métodos de resumen que se derivan de la misma son el ***Elemental***, el ***Básico*** y el ***Avanzado***.

   Por el momento esta tarea es capaz de recoger datos de **Marca.com, BBC.co.uk, AS.com, WhasingtonPost.com.**

   En las imágenes se pone un único ejemplo, ya que las demás es el mismo método pero buscado etiquetas html diferentes. Metemos en un array todo el contenido html de la página pasada, y vamos mirando letra a letra buscado la etiqueta deseada, cuando la tenemos insertamos en un StringBuilder los datos que nos interesan, y finalmente cuando acabamos pasamos el valor del texto que muestra la aplicación como el valor de ese StringBuilder que hemos construido anteriormente. De la misma forma se cuentan el número de párrafos y líneas que se van insertando en el texto a resumir.

```
private class AsyncTaskRunner2 extends AsyncTask<String, String, String> {

    private String resp;
    private ProgressDialog progress = new ProgressDialog(MainActivity.this);

    @Override
    protected String doInBackground(String... params) {
        //publishProgress("Obteniendo datos de URL..."); // Calls onProgressUpdate()

        //MIRAMOS MARCA, BBC Y WASHINGTON POST

        int marca = url.indexOf("www.marca.com");
        int bbc = url.indexOf("m.bbc.com");
        int washington = url.indexOf("www.washingtonpost.com");
        int as = url.indexOf(".as.com");

        //CONSUMIR DATOS URL

        StringBuilder builder = new StringBuilder(); //builder para datos HTML
        StringBuilder builder2 = new StringBuilder(); //builder para datos HTML procesados (texto final)
        HttpClient client = new DefaultHttpClient();
        HttpGet httpGet = new HttpGet(editText.getText().toString()); //pasamos url del metodo  handleText
        //HttpGet httpGet = new HttpGet("https://www.ingens-networks.com/secure/vinosyanadas.services/webser

        try {
            HttpResponse response = client.execute(httpGet);

            StatusLine statusLine = response.getStatusLine();

            int statusCode = statusLine.getStatusCode();

            if (statusCode == 200) {
                HttpEntity entity = response.getEntity();
                InputStream content = entity.getContent();
                BufferedReader reader = new BufferedReader(new InputStreamReader(content));

                //OBTENEMOS DATOS HTML
                String line;
                while ((line = reader.readLine()) != null) {
                    builder.append(line);
                }

                //-----------------------------------------------------------------------------------------

                //PROCESAMIENTO DE DATOS HTML PARA OBTENER TEXTO FINAL

                //-----------------------------------------------------------------------------------------

                //COMPROBAMOS DE QUE PAGINA PROVIENE EL ENLACE
```

Figura 30: Código desarrollado AsyncTaskRunner2.

```
if(bbc != -1){

    String aux = builder.toString(); //cogemos string del edittext
    String[] aux2 = new String[aux.length()]; //array de cadenas con longitud de aux
    int contador=0;
    int contador2=1;
    int contador3=0; //contador de letras
    boolean salir = false; //Boolean para salir al encontrar letra igual

    for(int i=0;i<aux2.length;i++){
        if(contador2 <= aux2.length){
            aux2[i]=aux.substring(contador, contador2); //metemos letra a letra en array
            //aux2[i].toLowerCase();//transformamos a minusculas
            contador++; //incrementamos contadores para la siguiente letra
            contador2++;
        }
    }

    for(int i=0;i<aux2.length;i++){ //recorremos string del edittext

        if(aux2[i].equalsIgnoreCase("<") && aux2[i+1].equalsIgnoreCase("p") && aux2[i+2].equalsIgnoreCase(">") && !aux2[i+3].equalsIgnoreCase("<") ){ //si encontramos <p> añadimos a partir de ahí al stringbuilder

            i+=3; //para pasar y añadir letra de contenido HTML (<p>)
            builder2.append(aux2[i]); //añadimos caracter a la cadena resultado
            parrSinResumir++; //anadimos linea sin resumir
            salir=true; //ponemos salir a true para que no entre hasta la siguiente letra

        }else if(salir == true){ //ya hemos encontrado <p> en el string, y metemos letras restantes en builder hasta encontrar </p>  que cierra el contenido

            if((aux2[i].equalsIgnoreCase("<") && aux2[i+1].equalsIgnoreCase("a")) || (aux2[i].equalsIgnoreCase("<") && aux2[i+1].equalsIgnoreCase("s")) ){ //eliminamos <a y <strong>

                int aux1 = contador3;

                for(int j=aux1;j<aux2.length;j++){
                    if(!aux2[i].equalsIgnoreCase(">")){ //hasta fin de etiqueta
                        i++;
                    }
                }

            }else if((aux2[i].equalsIgnoreCase("<") && aux2[i+1].equalsIgnoreCase("/") && aux2[i+2].equalsIgnoreCase("a")) || (aux2[i].equalsIgnoreCase("<") && aux2[i+1].equalsIgnoreCase("/") && aux2[i+2].equalsIgnoreCase("s")) ){ //eliminamos </a y </strong>

                int aux12 = contador3;

                for(int j=aux12;j<aux2.length;j++){
                    if(!aux2[i].equalsIgnoreCase(">")){ //hasta fin de etiqueta
                        i++;
                    }
                }
```

Figura 31: Código desarrollado AsyncTaskRunner2. Parte 2.

```
        }else if(aux2[i].equalsIgnoreCase("<") && aux2[i+1].equalsIgnoreCase("/") && aux2[i+2].equalsIgnoreCase("p") && aux2[i+3].equalsIgnoreCase(">")){
            //i+=4; //para pasar </p>
            builder2.append("\n");
            //builder2.append("\n");
            salir = false;
        }else{
            builder2.append(aux2[i]);
            if(aux2[i].equalsIgnoreCase(".")){ //añadimos linea a contador
                linSinResumir++;
            }
        }

    }

    contador3++;
}
```

Figura 32: Código desarrollado AsyncTaskRunner2. Parte 3.

```
                //------------------------------------------------------------------------------------------------------------
                // FIN DE PROCESAMIENTO DE DATOS HTML PARA OBTENER TEXTO FINAL
                //------------------------------------------------------------------------------------------------------------

            } else {
                //Log.e(JsonTestActivity.class.getName(), "Failed to download data");
                Toast toast1 =
                        Toast.makeText(getApplicationContext(),
                                "Fallo al descargar datos", Toast.LENGTH_SHORT);
                toast1.show();
            }
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        resp=builder2.toString();
        return resp;
    }

    @Override
    protected void onPostExecute(String result) {

        if (progress.isShowing()) {
            progress.dismiss();
        }
        editText.setText(result);
        gen2=result;

        if(parrSinResumir>0 && linSinResumir>0){
            Toast toast1 =
                    Toast.makeText(getApplicationContext(),
                            linSinResumir+" líneas y "+parrSinResumir+" párrafos recogidos", Toast.LENGTH_SHORT);
            toast1.show();
        }
    }

    @Override
    protected void onPreExecute() {
        progress.setMessage(url);
        progress.setTitle("Obteniendo datos de url...");
        progress.show();
    }

    @Override
    protected void onProgressUpdate(String... text) {
        editText.setText(text[0]);
    }
}
```

Figura 33: Código desarrollado AsyncTaskRunner2. Parte 4.

3. Compartir texto de internet con GPLSI Compendium App

Para compartir el texto de páginas de internet con la aplicación, en primer lugar debemos insertar en el Manifest.xml de la misma un **intent-filter** que le indique a la actividad que queremos que debe estar preparada para recibir texto:

```
<activity
    android:name=".MainActivity"
    android:label="SumUp" >
    <intent-filter>
        <action android:name="android.intent.action.PRINCIPAL" />

        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.SEND" />

        <category android:name="android.intent.category.DEFAULT" />

        <data android:mimeType="text/plain" />
    </intent-filter>
</activity>
```

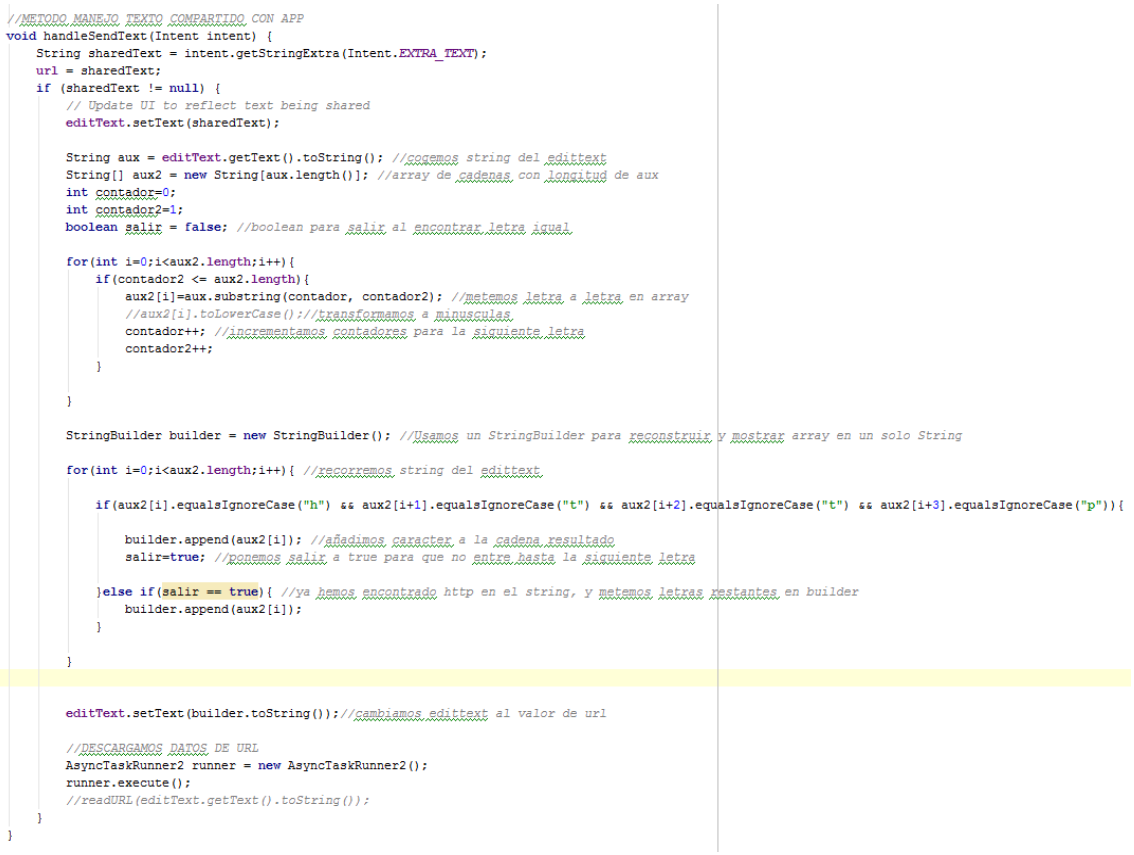
Figura 34: Código de activity en AndroidManifest.xml.

Después de esto solo queda el código en la activity.java, que será comprobar cuando nos está siendo enviada información, y se trata en el OnCreate() de la actividad, y tras esto llamar al método que maneje la información pasada, en este caso handleSendText(intent), en este método tras obtener la url que se nos pasa de la noticia se ejecuta la tarea asíncrona **AsyncTaskRunner2(),** para descargar y recoger la información de esa url pasada.

```
// Get intent, action and MIME type
Intent intent = getIntent();
String action = intent.getAction();
String type = intent.getType();

if (Intent.ACTION_SEND.equals(action) && type != null) {
    if ("text/plain".equals(type)) {
        handleSendText(intent); // Handle text being sent
    }
} else {
    // Handle other intents, such as being started from the home screen
}
```

Figura 35: Obtención de url de noticia.



```
//METODO MANEJO TEXTO COMPARTIDO CON APP
void handleSendText(Intent intent) {
    String sharedText = intent.getStringExtra(Intent.EXTRA_TEXT);
    url = sharedText;
    if (sharedText != null) {
        // Update UI to reflect text being shared
        editText.setText(sharedText);

        String aux = editText.getText().toString(); //cogemos string del edittext
        String[] aux2 = new String[aux.length()]; //array de cadenas con longitud de aux
        int contador=0;
        int contador2=1;
        boolean salir = false; //boolean para salir al encontrar letra igual

        for(int i=0;i<aux2.length;i++){
            if(contador2 <= aux2.length){
                aux2[i]=aux.substring(contador, contador2); //metemos letra a letra en array
                //aux2[i].toLowerCase();//transformamos a minusculas
                contador++; //incrementamos contadores para la siguiente letra
                contador2++;
            }

        }

        StringBuilder builder = new StringBuilder(); //Usamos un StringBuilder para reconstruir y mostrar array en un solo String

        for(int i=0;i<aux2.length;i++){ //recorremos string del edittext

            if(aux2[i].equalsIgnoreCase("h") && aux2[i+1].equalsIgnoreCase("t") && aux2[i+2].equalsIgnoreCase("t") && aux2[i+3].equalsIgnoreCase("p")){

                builder.append(aux2[i]); //añadimos caracter a la cadena resultado
                salir=true; //ponemos salir a true para que no entre hasta la siguiente letra

            }else if(salir == true){ //ya hemos encontrado http en el string, y metemos letras restantes en builder
                builder.append(aux2[i]);
            }

        }


        editText.setText(builder.toString());//cambiamos edittext al valor de url

        //DESCARGAMOS DATOS DE URL
        AsyncTaskRunner2 runner = new AsyncTaskRunner2();
        runner.execute();
        //readURL(editText.getText().toString());
    }
}
```

Figura 36: Obtención de url de noticia. Parte 2.

4. Compartir resúmenes de GPLSI Compendium App con otras apps

Simplemente debemos realizar un intent indicando el tipo de dato a compartir y el contenido del mismo, esto se realiza en la opción del menú superior compartir, por lo que se implementa en el **onOptionsItemSelected(MenuItem Item)**

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {

        case R.id.menu_share:

            //Compartimos contenido del resumen con otras apps
            Intent intent = new Intent (Intent.ACTION_SEND);
            intent.setType ("text/plain");
            intent.putExtra(Intent.EXTRA_TEXT,editText.getText().toString());
            startActivity (Intent.createChooser (intent, "Compartir con..."));

            return true;
```

Figura 37: Compartir resumen con otras apps.

5. Buscador de palabras

Para el buscador se le dice a la app primero de que clase va a ser el botón del menú, en este caso SearchView.

```
<item android:id="@+id/search"
    android:title="Busqueda"
    android:icon="@drawable/ic_busc22"
    app:showAsAction="collapseActionView|ifRoom"
    app:actionViewClass="android.widget.SearchView" />
```

Figura 38: Código de botón SearchView para buscador de palabras.

Tras esto se implementa en la activity el buscador mediante un SearchView y su correspondiente Listener, y una vez se le pulse a buscar se crea un **SpannableString** para resaltar la palabra buscada por el usuario, cambiándole el color y el tamaño de texto. Se splitea el texto por la palabra buscada y después se inserta tras cada parte del texto la palabra modificada con color u otra propiedad.

```java
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.menu_main, menu);

    MenuItem searchItem = menu.findItem(R.id.search);
    mSearchView = (SearchView) searchItem.getActionView();
    mSearchView.setQueryHint("Buscar...");
    mSearchView.setOnQueryTextListener(this);

    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {

        case R.id.search:

            mSearchView.onActionViewExpanded();
            mSearchView.requestFocus();
            mSearchView.requestFocusFromTouch();
            InputMethodManager imm = (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
            imm.toggleSoftInput(InputMethodManager.SHOW_FORCED,0);

            return true;

        default:
            return super.onOptionsItemSelected(item);
    }
}
```

Figura 39: Código de buscador de palabras en el texto.

```java
//////////////////////////// BUSCADOR ////////////////////////////////

public boolean onQueryTextChange(String text) {

    SpannableString spannable = new SpannableString(text);

    String gen = editText.getText().toString().toLowerCase();

    String b[]=gen.split(text);

    SpannableStringBuilder c= new SpannableStringBuilder();

    if(text.length()==0){ //optimizamos busqueda, si no se peta con mucho texto

        if(parrSinResumir==0 && linSinResumir==0){ //texto por defecto
            editText.setText(gen3);
        }else{ //texto a resumir
            editText.setText(gen2);
        }


    }else{

        for(int i=0;i<b.length;i++){

            c.append(b[i]);
            if(i<b.length-1){
                spannable.setSpan(new ForegroundColorSpan(Color.RED), 0, text.length(), Spannable.SPAN_EXCLUSIVE_EXCLUSIVE); //letra roja
                spannable.setSpan(new StyleSpan(Typeface.BOLD_ITALIC), 0, text.length(), Spannable.SPAN_EXCLUSIVE_EXCLUSIVE); //bold
                spannable.setSpan(new AbsoluteSizeSpan(50), 0, text.length(), Spannable.SPAN_EXCLUSIVE_EXCLUSIVE);//resize size
                c.append(spannable);
            }
        }

        editText.setText(c);

    }
    return false;
}

public boolean onQueryTextSubmit(String text) {

    Toast.makeText(this, "Busqueda realizada", Toast.LENGTH_LONG).show();

    mSearchView.clearFocus();

    return false;
}
```

Figura 40: Código de buscador de palabras en el texto. Parte 2.

Tipos de resumen

Para los tipos de resumen creados por el desarrollador se podrá trabajar por líneas o por párrafos, mientras que para los de Compendium sólo por líneas.

1. **Elemental**

   Este tipo de resumen devuelve la primera línea del texto a resumir. Para hacer esto se crea un StringBuilder al que se le añaden los caracteres comenzando por el principio del texto hasta encontrar un punto (“.”) que indique el final de línea. Para los párrafos igual pero buscando un salto de línea (“\n”).

```
public void tit_alternativo(String str){ //metodo para recoger primera frase del documento a resumir

    forma_seleccionada=spinner2.getSelectedItemPosition(); //cogemos tipo de resumen seleccionado por el usuario
    int lin=0;

    if(forma_seleccionada==0){ //resumen por lineas

        if(parrSinResumir>0){

            str=editText.getText().toString();

            StringBuilder builder2 = new StringBuilder(); //builder
            String aux = str; //cogemos string del edittext
            String[] aux2 = new String[aux.length()]; //array de cadenas con longitud de aux
            int contador=0;
            int contador2=1;
            boolean salir = false; //boolean para salir al encontrar letra igual

            for(int i=0;i<aux2.length;i++){
                if(contador2 <= aux2.length){
                    aux2[i]=aux.substring(contador, contador2); //metemos letra a letra en array
                    //aux2[i].toLowerCase();//transformamos a minusculas
                    contador++; //incrementamos contadores para la siguiente letra
                    contador2++;
                }

            }

            for(int i=0;i<aux2.length;i++){ //recorremos string del edittext

                if(!aux2[i].equalsIgnoreCase(".")) { //si encontramos un punto añadimos a partir de ahí al stringbuilder

                    builder2.append(aux2[i]); //añadimos caracter a la cadena resultado
                    salir=true;
                }else if(salir){
                    builder2.append("."); //añadimos punto a la cadena resultado
                    lin++;
                    break;
                }

            }

            editText.setText(builder2.toString()); //obtenemos primera linea como resumen
            gen2=editText.getText().toString();
            linResumidas=lin;
            Toast toast1 =
                    Toast.makeText(getApplicationContext(),
                            "Resumen de "+lin+" lineas", Toast.LENGTH_SHORT);

            toast1.show();
```

Figura 41: Código de resumen Elemental.

```
}else{ //resumen por parrafos

    if(parrSinResumir>0){

        str=editText.getText().toString();

        StringBuilder builder2 = new StringBuilder(); //builder
        String aux = str; //cogemos string del edittext
        String[] aux2 = new String[aux.length()]; //array de cadenas con longitud de aux
        int contador=0;
        int contador2=1;
        boolean salir = false; //boolean para salir al encontrar letra igual

        for(int i=0;i<aux2.length;i++){
            if(contador2 <= aux2.length){
                aux2[i]=aux.substring(contador, contador2); //metemos letra a letra en array
                //aux2[i].toLowerCase();//transformamos a minusculas
                contador++; //incrementamos contadores para la siguiente letra
                contador2++;
            }

        }

        for(int i=0;i<aux2.length;i++){ //recorremos string del edittext

            if(!aux2[i].equalsIgnoreCase("\n")) { //si encontramos \n añadimos a partir de ahí al stringbuilder

                builder2.append(aux2[i]); //añadimos caracter a la cadena resultado
                salir=true;
            }else if(salir){
                lin++;
                break;
            }

        }

        editText.setText(builder2.toString()); //obtenemos primera linea como resumen
        gen2=editText.getText().toString();
        parrResumidos=lin;
        Toast toast1 =
                Toast.makeText(getApplicationContext(),
                        "Resumen de "+lin+" párrafos", Toast.LENGTH_SHORT);

        toast1.show();
    }

}
```

Figura 42: Código de resumen Elemental. Parte 2.

2. Básico

En este tipo de resumen se devuelve como resumen la primera o última línea del texto (o párrafos). Para obtener la primera línea el procedimiento es el mismo que para el método Elemental, mientras que para obtener la última, se crea otro String donde se almacena desde el final hasta el primer punto que encuentre (en caso de líneas) o salto de línea (en caso de párrafos). Después se invierte esta cadena ya que está insertada desde final al principio, y se le añade al StringBuilder resultado.

```
public void basico(String str){ //metodo para recoger primera y ultima frase del documento a resumir

    forma_seleccionada=spinner2.getSelectedItemPosition(); //cogemos tipo de resumen seleccionado por el usuario
    int lin=0;

    if(forma_seleccionada==0) { //resumen por lineas

        if(linResumidas!=2){

            if(parrSinResumir>0){

                str=editText.getText().toString();
                int ultima=str.lastIndexOf("\n");

                StringBuilder builder2 = new StringBuilder(); //builder para primera linea

                StringBuilder builder3 = new StringBuilder(); //builder para ultima linea
                String cadenaInvertida="";

                String aux = str; //cogemos string del edittext
                String[] aux2 = new String[aux.length()]; //array de cadenas con longitud de aux
                int contador=0;
                int contador2=1;
                boolean salir = false; //boolean para salir al encontrar letra igual
                boolean salir2 = false; //boolean para salir al encontrar letra igual

                for(int i=0;i<aux2.length;i++){
                    if(contador2 <= aux2.length){
                        aux2[i]=aux.substring(contador, contador2); //metemos letra a letra en array
                        //aux2[i].toLowerCase();//transformamos a minusculas
                        contador++; //incrementamos contadores para la siguiente letra
                        contador2++;
                    }

                }

                for(int i=0;i<aux2.length;i++){ //recorremos string del edittext

                    if(!aux2[i].equalsIgnoreCase(".")) { //si encontramos \n añadimos a partir de ahí al stringbuilder

                        builder2.append(aux2[i]); //añadimos caracter a la cadena resultado
                        salir=true;
                    }else if(salir){
                        builder2.append(".");
                        lin++;
                        break;
                    }

                }
```

Figura 43: Código de resumen Básico.

```
        if(linSinResumir>1 && linResumidas >1){

            //builder3.append(editText.toString().substring(ultima, aux2.length));

            for(int i= ultima-1; i>=0; i--){ //recorremos string al revés desde ultima y saltamos ultimo punto con el -1

                if(!aux2[i].equalsIgnoreCase(".")) { //si encontramos \n añadimos a partir de ahí al stringbuilder

                    builder3.append(aux2[i]); //añadimos caracter a la cadena resultado
                    salir2=true;
                }else if(salir2){
                    lin++;
                    break;
                }
            }

            for (int x=builder3.toString().length()-1;x>=0;x--) { //invertimos cadena
                cadenaInvertida = cadenaInvertida + builder3.toString().charAt(x);
            }

            editText.setText(builder2.toString()+"\n\n"+cadenaInvertida+"."); //obtenemos primera linea como resumen
            gen2=editText.getText().toString();

        }else{
            editText.setText(builder2.toString()+"\n"); //obtenemos primera linea como resumen
            gen2=editText.getText().toString();
        }

        linResumidas=lin;
        Toast toast1 =
                Toast.makeText(getApplicationContext(),
                        "Resumen de "+lin+" líneas", Toast.LENGTH_SHORT);

        toast1.show();
    }

}
```

Figura 44: Código de resumen Básico. Parte 2.

3. Avanzado

Consiste en mostrar las n primera líneas o párrafos del texto como resumen, para realizar esto se le proporciona al usuario un NumberPicker que va de 0 a 100 en intervalos de 10, y representan el porcentaje de resumen deseado. Se realizar la operación correspondiente con el número de líneas o párrafos y se muestra del mismo modo que el Elemental, pero en este caso hasta llegar al valor n.

```
public void customizado(String str) { //metodo para recoger primera y ultima frase del documento a resumir

    forma_seleccionada=spinner2.getSelectedItemPosition(); //cogemos tipo de resumen seleccionado por el usuario
    int current = 0; //contador de lineas resumidas dinamico

    if(forma_seleccionada==0) { //resumen por lineas

        if(np.getValue()!=0) {
            float lin_finales2 = ((float) linSinResumir * ((float) np.getValue() * 10)) / 100;
            int lin_finales = Math.round(lin_finales2);

            if (parrSinResumir > 0) {

                str = editText.getText().toString();

                StringBuilder builder2 = new StringBuilder(); //builder
                String aux = str; //cogemos string del edittext
                String[] aux2 = new String[aux.length()]; //array de cadenas con longitud de aux
                int contador = 0;
                int contador2 = 1;
                boolean salir = false; //boolean para salir al encontrar letra igual

                for (int i = 0; i < aux2.length; i++) {
                    if (contador2 <= aux2.length) {
                        aux2[i] = aux.substring(contador, contador2); //metemos letra a letra en array
                        //aux2[i].toLowerCase();//transformamos a minusculas
                        contador++; //incrementamos contadores para la siguiente letra
                        contador2++;
                    }

                }

                for (int i = 0; i < aux2.length; i++) { //recorremos string del edittext

                    if (!aux2[i].equalsIgnoreCase(".")) { //si encontramos punto añadimos a partir de ahí al stringbuilder

                        if (current != lin_finales) {
                            builder2.append(aux2[i]); //añadimos caracter a la cadena resultado
                        }

                        if (current == lin_finales) {
                            salir = true;
                        }

                    } else if (salir) {
                        break;
                    } else {
                        current++;
                        builder2.append(".\n\n");
                    }
```

Figura 45: Código de resumen Avanzado.

```
                }

                editText.setText(builder2.toString()); //obtenemos primera linea como resumen
                gen2=editText.getText().toString();

                if (current == lin_finales) {

                    Toast toast1 =
                            Toast.makeText(getApplicationContext(),
                                    "Resumen de " + current + " líneas", Toast.LENGTH_SHORT);

                    toast1.show();
                }

            }
        }
```

Figura 46: Código de resumen Avanzado. Parte 2.

4. **Experto**

   Este resumen lo hace Compendium, por lo que sólo hemos de ejecutar la **AsyncTaskRunner** y pasarle los parámetros deseados, en este caso (-i, porcentaje).

5. **Inteligente**

   Este resumen lo hace Compendium, por lo que sólo hemos de ejecutar la **AsyncTaskRunner** y pasarle los parámetros deseados, en este caso (-i, porcentaje).

6. **Manual**

   En este tipo de resumen se mostrará al usuario las líneas del texto recogido del artículo por separado, y el propio usuario elegirá las líneas que desee para crear su propio resumen. Cada línea del texto se insertarán en botones, que el usuario al pulsar hará que se añadan al resumen. Una vez la añada, esa línea (botón) desaparecerá de las líneas posibles para añadir al resumen.

   Para crear este método de resumen, se separa el texto por líneas, y se crean tantos botones dinámicos como líneas tenga el resumen. Cada botón dinámico se añade al layout correspondiente a la activity en el mismo bucle. Se implementa el OnClickListener de cada botón, para que desaparezcan al pulsarlos y se añadan al resumen mediante el método ***anadirManual(String str, boolean terminar)***, en este método se van añadiendo las líneas seleccionadas a un StringBuilder y cuando se pulse el botón *Finalizar*, cargamos el layout original poniendo visibles los elementos originales y mostrando el contenido del resumen elegido por el usuario.

```
public void res_manual(String str) { //metodo para construir resumen propio

    bt1.setText("FINALIZAR");

    StringBuilder builder2 = new StringBuilder(); //builder

    str = editText.getText().toString();

    str = str.replaceAll("\n", " ");
    str = str.replaceAll("[\\.!\\?]\\s", " .\n");

    final String[] sentences = str.split("\n");

    final RelativeLayout layout = (RelativeLayout) findViewById(R.id.relmain);

    int contadorBoton= 0;

    LinearLayout.LayoutParams parametros = new LinearLayout.LayoutParams(
            LinearLayout.LayoutParams.FILL_PARENT, LinearLayout.LayoutParams.WRAP_CONTENT);
    parametros.setMargins(30, 20, 30, 0);

    int alturabtn = 0;

    Paint p = new Paint();
    Rect bounds = new Rect();

    spinner.setVisibility(View.INVISIBLE);
    spinner2.setVisibility(View.INVISIBLE);
    text4.setVisibility(View.INVISIBLE);
    text5.setVisibility(View.INVISIBLE);

    for (int i = 0; i < sentences.length; i++) {
        final LinearLayout row = new LinearLayout(this);
        row.setLayoutParams(new LinearLayout.LayoutParams(LinearLayout.LayoutParams.FILL_PARENT, LinearLayout.LayoutParams.WRAP_CONTENT));

        final Button btnTag = new Button(this);

        p.getTextBounds(sentences[i], 0, sentences[i].length(), bounds);

        ViewGroup.MarginLayoutParams params = (ViewGroup.MarginLayoutParams) row.getLayoutParams();
        params.width = LinearLayout.LayoutParams.FILL_PARENT; params.leftMargin = 0; params.topMargin = alturabtn;

        btnTag.setLayoutParams(params);
        btnTag.setText(sentences[i]);
        btnTag.setId(i);
        row.addView(btnTag);
```

Figura 47: Código de resumen Manual.

```
        layout.addView(row, i);

        alturabtn+=(p.measureText(sentences[i])/2)+(bounds.height()*8);
        contadorBoton++;

        btnTag.setOnClickListener((v) -> {

                int var = btnTag.getId();

                anadirManual(sentences[var], false);

                btnTag.setVisibility(View.INVISIBLE);

                Toast toast1 =
                        Toast.makeText(getApplicationContext(),
                                "Línea añadida a resumen", Toast.LENGTH_SHORT);

                toast1.show();
        });
    }

    editText.setText(""); //obtenemos primera linea como resumen

    final int finalContadorBoton = contadorBoton;
    contadorBoton=0;

    bt1.setOnClickListener((v) -> {

            //ELIMINAR BOTONES DINAMICOS

            if(bt1.getText()=="FINALIZAR") {
                anadirManual("",true);
                layout.removeViews(0, finalContadorBoton);
            }
    });

}
```

Figura 48: Código de resumen Manual. Parte 2.

```
public void anadirManual(String str, boolean terminar){

    if(terminar != true){

        buildermanual.append(str);
        buildermanual.append("\n\n");

    }else{
        editText.setText(buildermanual.toString());
        gen2=editText.getText().toString();
        bt1.setText("RESUMIR");
        spinner.setVisibility(View.VISIBLE);
        spinner2.setVisibility(View.VISIBLE);
        text4.setVisibility(View.VISIBLE);
        text5.setVisibility(View.VISIBLE);
        buildermanual.setLength(0); //vaciamos stringbuilder para siguiente vez
    }

}
```

Figura 49: Código de resumen Manual. Parte 3.