Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática - Mención Computación

APKFalcon: Servicio de usuarios para la evaluación y comprensión del impacto sobre la privacidad de aplicaciones móviles

Autor:
D. Javier Crespo Guerrero
Tutora:
Dña. Mercedes Martínez González

Agradecimientos

A mis profesores del Grado en Informática, por formarme todos estos años y transmitirme sus conocimientos y pasión por la informática. A Mercedes y Amador, por su gran dedicación en este trabajo, su invaluable ayuda y cuidado a la hora de trabajar, de la que he podido aprender mucho. También agradezco a Yania y a José Manuel, por su inestimable ayuda en el análisis y diseño de este sistema.

A mi familia, en especial a mis padres y mi hermano, por su apoyo, comprensión y paciencia. Gracias por enseñarme a afrontar los retos y no tirar nunca la toalla.

A Paula, por estar a mi lado siempre en los buenos y malos momentos. Gracias por todos estos años.

Por último, a todos los amigos y compañeros que me han acompañado durante estos últimos años, gracias por todo.

Resumen

En este trabajo se realiza un servicio web que tiene como objetivo proporcionar una herramienta que permita a los usuarios evaluar y comprender el impacto sobre su privacidad de aplicaciones para dispositivos moviles. La medición de este impacto está fundamentada en la implementación de una métrica de privacidad desarrollada en el marco de la investigación doctoral de D. Amador Aparicio de la Fuente, basada en el análisis estático de los permisos que declara una aplicación. Dentro de este servicio, el usuario puede buscar aplicaciones y obtener un informe detallado del riesgo que éstas suponen para su privacidad, así como subir sus propias aplicaciones para que sean analizadas. Este informe propociona resultados gráficos, resultados desarrollados y puede ser exportado en varios formatos. El sistema está pensado tanto para usuarios con perfil tecnológico como usuarios de dispositivos móviles sin conocimientos tecnológicos, por lo que se ha realizado un proceso de pruebas de usabilidad para asegurarlo.

El servicio se compone de un servidor web, que da respuesta a las peticiones HTTP de los clientes y un sistema integrador, que se encarga de descargar las aplicaciones de diversas fuentes y analizarlas para extraer sus metadatos. Además, este servicio se integra con el trabajo realizado en otros dos TFG, formando un sistema global de integración de información de aplicaciones móviles.

Abstract

In this work, a web service is developed with the aim of providing a tool that allows users to assess and understand the impact on their privacy of mobile applications. The measurement of this impact is based on the implementation of a privacy metric developed within the framework of D. Amador Aparicio de la Fuente's doctoral research, which is based on a static analysis of the permissions declared by an application. Within this service, users can search for applications and obtain a detailed report on the privacy risks posed by these applications, as well as upload their own applications to be analyzed. This report provides both graphical and detailed results, and can be exported in various formats. The system is designed for both technically inclined users and mobile device users without technical knowledge, therefore usability testing has been carried out to ensure its effectiveness.

The service consists of a web server that responds to client's HTTP requests, and an integration system that is responsible for downloading applications from various sources and analyzing them to extract their metadata. Additionally, this service integrates with the work carried out in two other final degree projects, creating a comprehensive system for integrating information about mobile applications.

Índice general

Capítulo 1

Introducción

1.1. Contexto

La protección de la privacidad de los usuarios de aplicaciones móviles está relacionada con el acceso de éstas a la información privada que se almacena en los dispositivos móviles. En el entorno Android estos accesos se controlan a través del sistema de permisos [9], donde los usuarios pueden conceder y denegar algunos de estos permisos, lo cual les da control sobre su privacidad. Este control supone empoderar a los usuarios de las aplicaciones móviles, lo cual se alinea con los objetivos expresados por la Unión Europea (UE) en su Estrategia para una Sociedad Digital [11].

Sin embargo, varias investigaciones demuestran que la comprensión por parte de los usuarios del sistema de permisos y de su impacto sobre la privacidad es deficitaria [16]. Es más, se da lo que se denomina *Paradoja de la Privacidad*, esto es, usuarios que afirman estar preocupados por su privacidad hacen una gestión de los permisos de sus aplicaciones contradictoria con esta afirmación, de tal modo que su privacidad está desprotegida de modo evidente [16]. Se han propuesto diversas explicaciones para esta realidad, que abarcan el desconocimiento del sistema de permisos [21], la falta de cultura digital y bajo nivel de los conocimientos técnicos de los usuarios, o la carencia de información realmente comprensible para los usuarios sobre la repercusión de cada permiso sobre sus datos personales [17]. Probablemente ninguna de ellas explica por si sola este fenómeno, pero es evidente que la comprensión correcta de un sistema, situación o fenómeno sea de la naturaleza que sea, ayuda a los seres humanos a hacer gestiones más eficaces.

En este contexto se planteó desde el grupo de investigación al que pertenecen los tutores de este trabajo una métrica para medir el impacto de la privacidad de las aplicaciones móviles [3]. Esta métrica basa sus cálculos en los metadatos extraídos de las aplicaciones móviles.

1.2. Motivación

Como se ha comentado, en muchos casos los usuarios de aplicaciones móviles están desinformados y carecen de herramientas de acceso rápido y sencillo con resultados comprensibles que digan de una forma clara qué riesgos están asumiendo al utilizar ciertas aplicaciones móviles. Además, la mayoría de herramientas existentes (véase, por ejemplo, Exodus Privacy[1] o Privacy Grade[2], presentada en [15]) son frecuentemente difíciles de comprender por parte del usuario inexperto, y en muchos casos se limitan a hacer una recopilación de los permisos que declaran las aplicaciones y datos a los que acceden, proporcionando resultados parciales que no sirven al usuario final para

[1] `https://exodus-privacy.eu.org/`

[2] `https://android-network-tracing.herokuapp.com/privacygrade`

empoderarse y proteger mejor su privacidad, sino que simplemente se limita a instalar o no instalar una aplicación.

En nuestra opinión, el desarrollo de una métrica de privacidad debe de ir acompañado de un servicio que permita a cualquier usuario acceder a ella, de forma que la información no quede aislada sobre el papel, sino que sea utilizada para empoderar a los usuarios y ayudarles a tomar un papel activo en su protección.

Por todo ello, en el marco de la investigación que se realiza en el departamento de Informática de la Universidad de Valladolid sobre la seguridad y privacidad en dispositivos móviles se plantea como elemento esencial de mejora la facilitación de herramientas que faciliten a los usuarios una mejor comprensión del ecosistema de permisos presente en sus dispositivos. Y que, en consecuencia, les ayude a tomar mejores decisiones en su gestión, de modo que puedan restringir los permisos que conceden a aquellos que realmente se adecúen con su voluntad. En la literatura existen propuestas destinadas a un perfil técnico, que busca evaluar el nivel de seguridad de las aplicaciones o realizar evaluaciones de impacto de tipo profesional [4]. Sin embargo, es difícil encontrar propuestas que avancen en la creación de herramientas como la que aquí se pretende, destinadas a empoderar a los usuarios de aplicaciones de cualquier perfil, no necesariamente técnico.

En una sociedad cada vez más digitalizada uno de los retos es obtener la información necesaria para proporcionar servicios de calidad. Este es el conocido como problema de la integración de información, el cual se ocupa de construir herramientas capaces de extraer información de fuentes de datos heterogéneas, distribuidas y no controladas [10]. Este es también el caso en el problema que se aborda: en ocasiones las aplicaciones y sus metadatos están dispersos en diversas fuentes. Esto se debe a que las medidas de protección que los grandes proveedores de mercados de aplicaciones ponen en funcionamiento para proteger sus intereses comerciales dificultan progresivamente el acceso automatizado a los metadatos de sus aplicaciones. Este es el caso de Google y su mercado Android [3], por ejemplo, donde no es posible obtener los metadatos de una aplicación de forma sencilla. Por ello es conveniente recurrir a fuentes alternativas donde algunos desarrolladores han extraído y puesto a disposición de la comunidad tecnológica los metadatos de las aplicaciones Android, de forma que puedan ser fácilmente utilizados para realizar el análisis deseado.

1.3. Objetivos

Este Trabajo Fin de Grado (en adelante TFG) tiene como objetivo **proporcionar una herramienta que permita a los usuarios evaluar y comprender el impacto sobre su privacidad de aplicaciones para dispositivos móviles**.
Los objetivos específicos son:

- Diseño e implementación de un servicio web que permita obtener el grado de intrusividad de una aplicación Android a partir de la métrica dada por [3].

- Diseño e implementación de un sistema de integración de la información que permita la obtención de información de aplicaciones móviles de fuentes heterogéneas.

- Asegurar la comprensibilidad y usabilidad de la herramienta por parte de los usuarios finales.

[3] https://play.google.com/store/

1.4. Organización del documento

En el capítulo 2 se muestra la planificación del proyecto, junto con el detalle de la metodología empleada y todo lo relacionado con actividades, riesgos y presupuesto. A continuación, en el capítulo 3, serán revisados los conceptos fundamentales de Android y su sistema de permisos, así como las distintas propuestas para cuantificar la privacidad de las aplicaciones móviles y la métrica desarrollada por el grupo de investigación.

En el capítulo 4 se realiza el análisis completo del sistema a desarrollar. En primer lugar se mostrará el sistema global que se pretende desarrollar en el departamento, a continuación se detalla el análisis de requisitos, casos de uso, modelo de dominio, fuentes de datos y usuarios que utilizarán el sistema. Posteriormente, en el capítulo 5 se realiza el diseño del sistema, comenzando por los patrones de diseño empleados y siguiendo por el diseño del software que compondrá el Servidor Web y Sistema Integrador del sistema.

La implementación es descrita en el capítulo 6, donde se expondrán los cambios con respecto al diseño inicial, la organización del código y partes clave de la implementación como la comunicación con las fuentes de datos. Las pruebas de este sistema se presentan en el capítulo 7, en primer lugar las pruebas software y en segundo las de aceptación por parte de los usuarios. El seguimiento de la planificación del proyecto, así como los riesgos materializados, se detallarán en el capítulo 8

Finalmente, en el capítulo 9 se presentan las conclusiones obtenidas tras la realización del TFG y las líneas futuras de trabajo.

Capítulo 2

Planificación

2.1. Características del proyecto

El proyecto tiene dos componentes principales: un Servidor Web y un Sistema Integrador. El Servidor Web se encargará de procesar las peticiones HTTP que realicen los usuarios, dando respuesta a dichas peticiones a través de documentos HTML que serán vistos desde un navegador web. El Sistema Integrador encapsula el acceso a las fuentes de datos, recabando y procesando metadatos de aplicaciones móviles en un formato compatible con el modelo de dominio que maneja el Servidor Web. La funcionalidad básica del sistema es consultar la métrica de privacidad asociada a una aplicación Android. A los resultados de esta métrica se añaden explicaciones, gráficos y comparaciones con aplicaciones similares que aumentan la comprensibilidad de los resultados. Además, se incluye la posibilidad de exportar los resultados y consultar aplicaciones subiendo el archivo fuente, lo que completará la funcionalidad básica de consulta de la métrica.

Este sistema, debido a los posibles cambios en la métrica durante el desarrollo y la disponibilidad de las fuentes de datos, está sujeto a la continua posibilidad de cambios en las fuentes que se utilizan. Es por ello que la revisión, modificación y actualización de partes ya desarrolladas del sistema puede ser necesaria a lo largo de todo el proceso de desarrollo.

2.2. Metodología empleada

Dadas las características comentadas, **se empleará una metodología incremental**: el sistema se irá construyendo por etapas en las que se va añadiendo una nueva funcionalidad o cambios que faciliten la usabilidad del sistema. Los incrementos (o iteraciones) se definen por medio de hitos que indican la funcionalidad a añadir o las acciones a realizar. Además, nos permite gestionar los riesgos de una manera local a cada incremento (por ejemplo, si una fuente de datos dejase de estar disponible al inicio de un incremento la metodología nos da flexibilidad para solucionar el problema).

Sin embargo, en contraposición al modelo incremental clásico, todos los requisitos de funcionalidad del sistema se reunirán al principio y a partir de ellos se diseñarán los incrementos. Esto es debido a que el sistema no tiene previsto un crecimiento fuera del tiempo establecido en este proyecto, por ello no se prevén cambios en los requisitos. La única parte del sistema que está sujeta a la posibilidad de cambios mayores a lo largo del desarrollo del proyecto son la fuentes de datos utilizadas. Cabe destacar que el último incremento se diseñará como una etapa de evaluación de los usuarios de la aplicación con toda la funcionalidad implementada, por tanto el resultado de ese último incremento no será la adición de nueva funcionalidad al sistema, sino cambios en la presentación de la información que faciliten la usabilidad del mismo.

2.3. Planificación inicial

El proyecto se lleva a cabo a lo largo de **300h**[1] **entre el 13 de febrero y el 4 de junio de 2023**, entre los cuales hay 110 días (15 semanas) de diferencia. Esto nos da una media de trabajo de 3h por día (21h por semana) dejando 10 días (30h) de margen por posibles inconvenientes.

2.3.1. Plan de actividades

En la figura 2.1 se puede ver el desglose de las actividades con la duración y relaciones de dependencia. La primera actividad que se llevará a cabo es el estudio del funcionamiento y algoritmo de la métrica a implementar, a partir de la información dada por los tutores. En paralelo a este estudio se realizará el análisis del sistema, que incluirá los requisitos y casos de uso del mismo, así como el estudio preliminar de las fuentes de datos disponibles y los datos a extraer de éstas.

A continuación se procederá a realizar el diseño software del sistema, tanto del Servidor Web como del Sistema Integrador. Posteriormente se pondrá el marcha el entorno de desarrollo y se seleccionarán las fuentes de datos a utilizar, todo ello en función de las necesidades y limitaciones recogidas en el análisis y diseño del sistema. Con todo ello, se procederá a implementar el sistema de acuerdo con la metodología descrita: comenzando por un sistema completo que implemente la funcionalidad básica y posteriormente añadiendo nueva funcionalidad como resultado de posteriores incrementos.

Por último, una vez tengamos la funcionalidad completa, se realizarán las pruebas de usuario que permitirán detectar aspectos de mejora, cuya modificación será el resultado del último de los incrementos. La redacción de la presente memoria se ha realizado a lo largo de todo el desarrollo del proyecto.

2.3.2. Hitos

En la tabla 2.1 se muestran los hitos propuestos y su fecha de compleción estimada.

ID	Hito	Fecha
1	Análisis del sistema	21/02/2023
2	Diseño del sistema	01/03/2023
3	Implementación del servicio básico con una fuente de datos	19/04/2023
4	Implementación de todas las fuentes de datos	26/04/2023
5	Implementación de la funcionalidad completa	09/05/2023
6	Tests de usabilidad realizados	30/05/2023
7	Fin del proyecto	05/06/2023

Cuadro 2.1: Hitos del proyecto y fechas de compleción.

A partir de estos hitos podemos definir los siguientes incrementos:

Incremento 1: Implementación de la funcionalidad básica del sistema, que incluye las fases pre-

[1]Este TFG está englobado dentro de un periodo de colaboración con el Departamento de Informática bajo una Beca de colaboración destinada a estudiantes universitarios para realizar tareas de investigación en departamentos universitarios del Ministerio de Educación. Se han dedicado más horas a lo largo del primer cuatrimestre, pero no se reflejan en esta memoria al no pertenecer estrictamente a los objetivos de este TFG

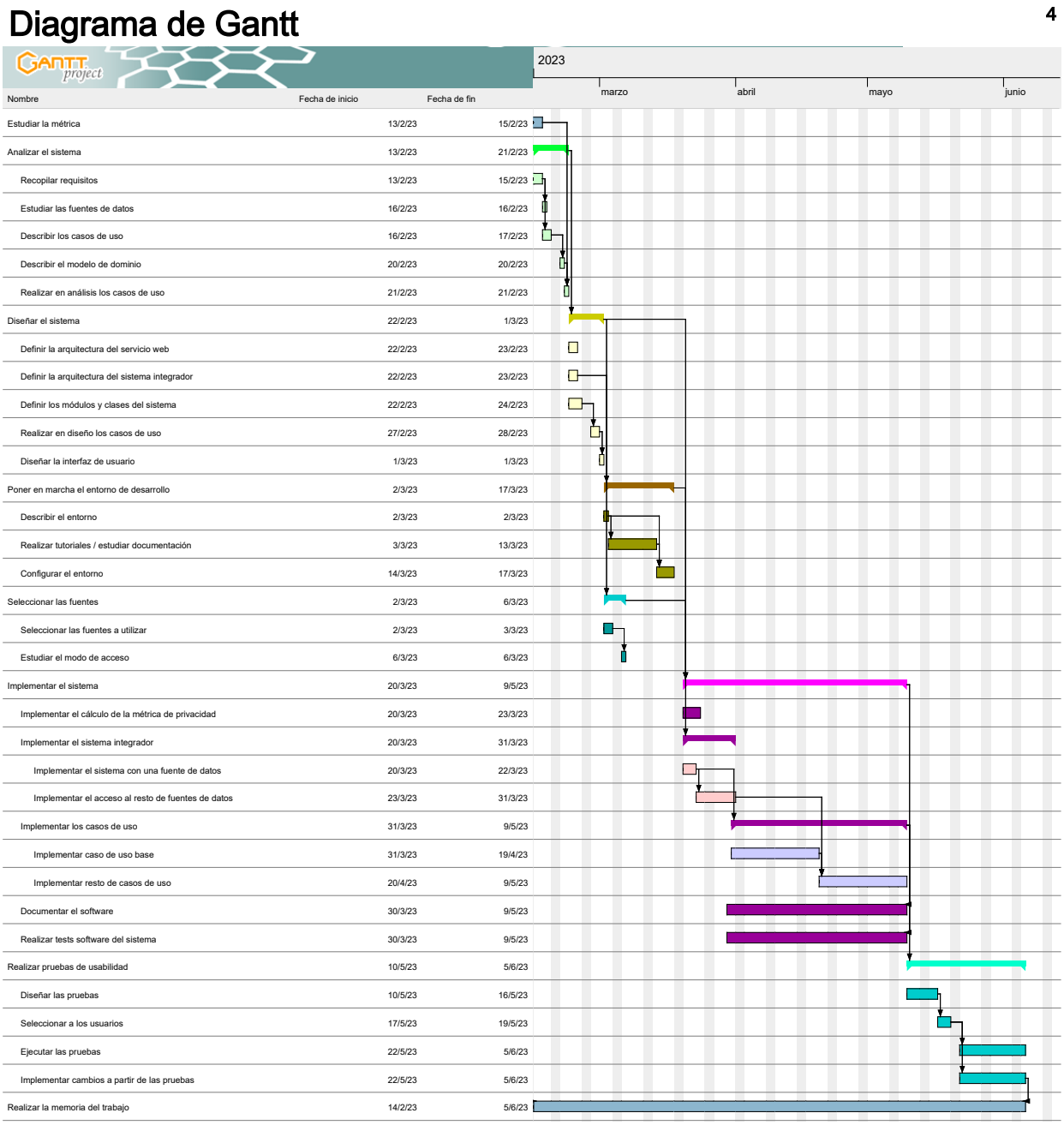


Figura 2.1: Plan de actividades del proyecto: descripción, fechas de inicio y fin y diagrama de Gantt con relaciones de dependencia.

vias de análisis y diseño del sistema completo[2]. El servidor web permitirá consultar la métrica asociada a una aplicación web y el sistema integrador utilizará una sola fuente de datos. Fecha de finalización: 19/04/2023.

Incremento 2: Implementación del sistema integrador completo, que utilizará todas las fuentes de datos. Fecha de finalización: 26/04/2023.

[2]Aunque en un desarrollo basado en incrementos no sea habitual realizar el análisis y diseño completos al principio (sino hacerlo por partes en cada incremento), en este caso se hará así, ya que al tener los requisitos fijos, un análisis y diseño completo llevarán a una implementación de la funcionalidad básica que será más fácilmente ampliable.

Incremento 3: Implementación de la funcionalidad completa, entre la que se incluye exportar resultados, subir archivos fuente y presentaciones gráficas de los resultados. Fecha de finalización: 09/05/2023.

Incremento 4: Realización de los tests de usuario y cambios en la presentación de los resultados a la luz de los resultados de las pruebas. Fecha de finalización: 05/06/2023.

Como corresponde a un diseño incremental, los resultados de los hitos serán sistemas completos en funcionamiento.

2.4. Plan de riesgos

En esta sección se analizan los distintos riesgos que podremos encontrar en la ejecución del proyecto y que puedan afectar a la planificación. Estos se caracterizan de acuerdo a la probabilidad de ocurrencia y al impacto que puedan causar en caso de materializarse, ambos en una escala del 1 al 4 [14] creando así una *matriz de riesgos* que se muestra en la figura 2.2 que nos permite conocer la exposición (o daño) de cada riesgo.

Impacto	Alto	Significativo	Moderado	Alto	Alto
	Significativo	Significativo	Moderado	Moderado	Alto
	Moderado	Bajo	Significativo	Moderado	Moderado
	Bajo	Bajo	Bajo	Significativo	Significativo
		Baja	Moderada	Significativa	Alta
		Probabilidad			

Figura 2.2: Matriz de riesgos, imagen propia basada en la información mostrada en [13].

A partir de la exposición de cada riesgo se plantean las posibles las acciones de mitigación y contención necesarias, pudiéndose simplemente aceptar el riesgo en caso de que tenga una exposición baja. En la figura 2.2 se muestra la "línea de la tolerancia", marcada con un trazo más grueso,

que marca las categorías donde tenemos que prestar especial atención a los riesgos, diseñando cuidadosamente los planes de mitigación/contención.

En la tabla 2.2 se muestran las descripciones de los riesgos identificados junto con su probabilidad, impacto y exposición.

ID	Riesgo	Probabilidad	Impacto	Exposición
R1	Estimaciones demasiado optimistas del trabajo	Moderada	Significativo	Moderada
R2	Retrasos por el uso de tecnologías desconocidas	Moderada	Significativo	Moderada
R3	**Modificación/adición de requisitos**	Baja	Significativo	**Significativa**
R4	**Máquina virtual / entorno de desarrollo inaccesible**	Baja	Alto	**Significativa**
R5	**Inexistencia de fuentes de datos**	Baja	Alto	**Significativa**
R6	Caída de una o más fuentes de datos	Significativa	Significativo	Moderada
R7	Imposibilidad de acceder a las fuentes de datos	Moderada	Alto	Moderada
R8	**No disponibilidad de usuarios para realizar pruebas**	Baja	Alto	**Significativa**
R9	**Enfermedad / no disponibilidad del desarrollador**	Baja	Significativo	**Significativa**

Cuadro 2.2: Identificación de riesgos.

En la tabla 2.3 se muestran las acciones de mitigación para cada riesgo.

Riesgo	Acciones de mitigación
R1	Contrastar la planificación con los tutores y hacer revisiones contínuas de ésta.
R2	Estudio cuidadoso de documentación y manuales previo al desarrollo.
R3	Estudio exhaustivo de requisitos previo al desarrollo, contrastado con tutores.
R4	Creación de entornos de desarrollo y despliegue locales que puedan ser usados a modo de "backup".
R5	Consulta con tutores y especialistas del sector.
R6	Uso de fuentes de datos alternativas. Estudio previo de fuentes principales y fuentes alternativas.
R7	Consulta con especialistas. Uso de fuentes alternativas.
R8	Ampliar el espectro de usuarios a considerar.
R9	Asegurar márgenes de tiempo en la planificación.

Cuadro 2.3: Mitigación de riesgos.

2.4.1. Riesgos de seguridad

Al tratarse de un servicio expuesto al público, a continuación se muestran los distintos riesgos que afectan a la seguridad del sistema, junto con sus acciones de mitigación. En la tabla 2.4 se puede ver la descripción de los riesgos junto con su exposición y en la tabla 2.5 las acciones de mitigación.

ID	Riesgo	Probabilidad	Impacto	Exposición
RS1	Componentes software no actualizados	Moderada	Significativo	Moderada
RS2	**Se muestran errores en tiempo de ejecución no tratados que puedan revelar información sensible**	Alta	Alto	**Alta**
RS3	El sistema tiene habilitado el módulo de depuración, por lo que se revela información sensible	Baja	Significativo	Significativa
RS4	**El sistema tiene puntos de entrada donde el usuario puede introducir información**	Alta	Alto	**Alta**
RS5	El sistema tiene una configuración de permisos que permite la descarga de ficheros a cualquier usuario	Baja	Alto	Significativa

Cuadro 2.4: Identificación de riesgos de seguridad.

Riesgo	Acciones de mitigación
RS1	Actualizar componentes para eliminar vulnerabilidades.
RS2	Encapsular los errores en tiempo de ejecución, mostrando mensajes de error.
RS3	Deshabilitar el modo depuración al lanzar el servicio.
RS4	Validar la información recibida por el usuario para comprobar que no se compromete el sistema.
RS5	Configurar de manera adecuada los permisos del sistema de ficheros.

Cuadro 2.5: Mitigación de riesgos de seguridad.

2.5. Presupuesto

El gasto en productos hardware será de 0€, ya que solamente es necesaria una máquina de prestaciones medias para el desarrollo, que será el ordenador personal del desarrollador.

En cuanto a productos software, se utilizará una máquina virtual para el desarrollo y despliegue que será proporcionada por la Escuela, por lo que el coste también será de 0€. Sin embargo, el coste de un servidor de pequeñas prestaciones se puede estimar en 70€ mensuales. Podríamos estimar el coste software en 70€ * 4 = 280€, suponiendo que alquilásemos un servidor durante 4 meses.

El gasto de personal será también de 0€, ya que no se cobrará por el desarrollo del proyecto. El coste de contratación de un programador junior es de, aproximadamente, 11€/hora, por lo que un proyecto de 300h tendría un coste de 3300€.

En un sistema real esto supondría un coste total de, aproximadamente, 3580€.

Capítulo 3

Fundamentos

3.1. Aplicaciones Android

Las aplicaciones Android se presentan y distribuyen como archivos con extensión `.apk` [6] (Android Package), un archivo comprimido que contiene todos los recursos y componentes necesarios para la instalación y ejecución de una aplicación en un dispositivo Android. El archivo `.apk` se compone de varios elementos, entre los que se incluyen:

1. Manifest: un archivo XML (*AndroidManifest.xml*) con los metadatos de la aplicación: el nombre del paquete (que suele seguir la estructura `com.desarrollador.nombre`), la versión, los permisos requeridos, las actividades, servicios y receptores de la aplicación.

2. Código fuente: el código fuente de la aplicación, escrito en Java/Kotlin y compilado en código ejecutable de máquina virtual.

3. Recursos: los recursos necesarios para la aplicación tales como imágenes, sonidos, vídeos, archivos de configuración e interfaz de usuario.

4. Archivos nativos: si la aplicación utiliza bibliotecas nativas, como accesos a C o C++, se incluyen en el archivo `.apk`.

5. Certificados: cada archivo `.apk` debe estar firmado digitalmente por el desarrollador para asegurar su integridad y autenticidad. Cada desarrollador tiene un certificado digital de firma único que puede utilizar en varias aplicaciones. Cabe destacar que el estudio de estos certificados también es interesante desde el punto de vista de la seguridad y la privacidad de una aplicación, pero en este trabajo no serán considerados.

6. Otros archivos tales como software de conexión a servicios en la nube, modelos 3D o software para trabajar con realidad aumentada, por ejemplo.

Los archivos `.apk` pueden ser descargados e instalados directamente en el dispositivo móvil, pero lo más común es descargarlos e instalarlos a través de *markets* como *Google Play*, de Google, o *Galaxy Store*, de Samsung.

3.2. Sistema de permisos en Android

El sistema de permisos de Android controla tanto el acceso a datos sensibles (contactos, SMS, imágenes,...) como la ejecución de acciones restringidas (tales como grabar audio/vídeo o acceder

al almacenamiento interno) mediante la declaración de permisos[9]. Ésta se hace el el *Manifest* de la aplicación mediante elementos de tipo `<permission>`.

El grado de restricción de las acciones o información a la que se accede se define a partir del "nivel de protección" (*protection level*) del permiso, que puede ser *normal, dangerous, signature* o *appop*.

Los permisos se caracterizan en dos dependiendo del grado de restricción de los datos/acciones de que hagan uso: en tiempo de instalación, en tiempo de ejecución y permisos especiales.

3.2.1. Permisos en tiempo de instalación

Estos permisos acceden a información y acciones que apenas afectan al sistema o a otras aplicaciones. Estos permisos se muestran en las páginas de detalles de los distintos markets y el sistema otorga el permiso a la aplicación automáticamente en la instalación. Los permisos en tiempo de instalación se caracterizan en dos en función del nivel de protección que se les asigne.

1. Nivel de protección *normal*: permisos que permiten el acceso a datos y acciones que suponen un riesgo mínimo para el funcionamiento de otras aplicaciones y la privacidad del usuario. Son ejemplos de estos permisos [8] `ACCESS_NETWORK_STATE`, que permite conocer la información sobre las redes disponibles, y `BLUETOOTH_ADMIN`, que permite descubrir y vincular dispositivos Bluetooth.

2. Nivel de protección *signature*: asignado a permisos que solo se otorgan a aplicaciones que están firmadas con el mismo certificado que la aplicación que los solicita. Son ejemplos de estos permisos `MANAGE_DOCUMENTS`, que permite a una aplicación leer y escribir documentos en nombre del usuario, y `BIND_ACCESSIBILITY_SERVICE`, que permite a una aplicación interactuar con servicios de accesibilidad.

3.2.2. Permisos en tiempo de ejecución

Estos permisos acceden a información o acciones restringidas del dispositivo, pudiendo contener datos especialmente sensibles que afectan a la privacidad del usuario y al funcionamiento del dispositivo. Éstos han de ser solicitados de forma explícita al usuario en tiempo de ejecución y éste puede revocarlos en cualquier momento desde la configuración del dispositivo, por ello se recomienda que al solicitar estos permisos la aplicación explique correctamente su propósito. Todos los permisos en tiempo de ejecución tienen asignado un nivel de protección *dangerous*. Ejemplos de estos permisos son `CAMERA`, que permite a una aplicación acceder a la cámara del dispositivo, y `READ_CONTACTS`, que permite a una aplicación leer los contactos almacenados en el dispositivo.

3.2.3. Permisos especiales

Los permisos especiales son aquellos que sólo pueden ser otorgados por el fabricante o el sistema operativo. Están diseñados para ser utilizados por aplicaciones del sistema o desarrolladas por el fabricante del dispositivo para acceder a funciones muy concretas. La mayoría de las aplicaciones de terceros no pueden acceder a estos permisos. Ejemplos de estos permisos son `READ_PHONE_NUMBERS`, que permite a una aplicación leer los números de teléfono asociados a la tarjeta SIM, y `ANSWER_PHONE_CALLS`, que permite a una aplicación responder a llamadas telefónicas.

3.2.4. Grupos de permisos

Los permisos que declara una aplicación se categorizan en grupos de permisos, declarados en el *Manifest* como elementos de tipo `<permission-group>` [7]. En la tabla 3.1 se muestran los grupos de permisos disponibles y su descripción.

Grupo	Descripción
android.permission-group.CALENDAR	Permisos en tiempo de ejecución relacionados con el calendario.
android.permission-group.CALL_LOG	Permisos asociados a funciones de telefonía.
android.permission-group.CAMERA	Permisos asociados con el acceso a la cámara y la captura de imagenes/vídeos en el dispositivos.
android.permission-group.CONTACTS	Permisos en tiempo de ejecución relacionados con el acceso a contactos y perfiles del dispositivo.
android.permission-group.LOCATION	Permisos relacionados con el acceso a la ubicación.
android.permission-group.MICROPHONE	Permisos relacionados con el acceso al micrófono.
android.permission-group.NEARBY_DEVICES	Permisos necesarios para descubrir y conectarse con dispositivos Bluetooth cercanos.
android.permission-group.NOTIFICATIONS	Permisos asociados con la creación de notificaciones.
android.permission-group.PHONE	Permisos asociados con funciones de telefonía.
android.permission-group.READ_MEDIA_AURAL	Permisos necesarios para leer archivos de audio del almacenamiento compartido.
android.permission-group.READ_MEDIA_VISUAL	Permisos necesarios para leer archivos de imagen/vídeo del almacenamiento compartido
android.permission-group.SENSORS	Permisos relacionados con el acceso a sensores corporales o del entorno.
android.permission-group.SMS	Permisos relacionados con el acceso a mensajes SMS.
android.permission-group.STORAGE	Permisos en tiempo de ejecución que acceden al almacenamiento compartido externo.

Cuadro 3.1: Grupos de permisos en Android (extraidos de [7] a fecha 20-04-2023).

Todos los permisos de tipo *dangerous* deben estar asignados a un grupo de permisos. Esto se puede hacer explicitamente en el *Manifest* mediante el atributo `android:permissionGroup` de los elementos `<permission>` . Si no se hace una asignación explícita, se asignan a grupos pre-establecidos. Google proporciona una recomendación [1] sobre qué grupo asignar a cada permiso, pero la decisión final recae sobre el desarrollador. Cuando la app solicita consentimiento explícito, en realidad lo está haciendo para un grupo entero de permisos. Éste es un dato que muchos usuarios finales desconocen y es necesario enfatizar a la hora de explicar el impacto sobre la privacidad de una aplicación.

3.3. Privacidad y aplicaciones Android

Medir el impacto sobre la privacidad de una aplicación Android de forma estática (a través del análisis del *Manifest*, sin llegar a ejecutar la aplicación) resulta complicado, pues no hay forma de saber el uso que se le da a los permisos declarados sin recurrir a un estudio en profundidad de la política de privacidad (estudio difícilmente realizable de forma automática). Por ejemplo, una aplicación puede necesitar autenticación en dos pasos con SMS para iniciar sesión y usar el permiso de tipo *dangerous* `android.permission.READ_SMS` para capturar automáticamente el SMS de la autenticación, no utilizando los SMS en ningún otro caso. Por otro lado, puede haber alguna aplicación que también solicite acceso a los SMS y sí que lo utilice de forma maliciosa. En ambos casos la información del *Manfiest* será la misma, pero no así el uso que se le está dando al permiso.

Sin embargo, se ha demostrado [21] que el *malware* suele solicitar más permisos (y de carácter más intrusivo) que las aplicaciones benignas. Esta información se ha utilizado para rankings de permisos y así poder cuantificar el riesgo de una aplicación en base a los permisos que ésta declara. Recientemente se han hecho intentos de detección de *malware* en base a estos rankings en [19] y en [20] utilizando además el tráfico de red.

Se han propuesto varias métricas de privacidad en [21], [19], [16] y [5]. Sin embargo, todas ellas están más enfocadas en la detección de *malware* que en hacer una métrica que sirva para medir en impacto en la privacidad de todo tipo de aplicaciones. Tampoco se incluye información sobre los grupos de permisos en estas propuestas.

3.3.1. Métrica desarrollada

En el grupo de investigación al que pertenecen los tutores de este TFG se está desarrollando una métrica (presentada en [3]) para puntuar la privacidad de las aplicaciones Android. El servicio desarrollado en este TFG pretende servir tanto como una herramienta de usuario final como un entorno de pruebas fácilmente accesible y editable para la métrica. Esta métrica se determina de forma estática, es decir, directamente sobre el *Manifest* de una aplicación.

La métrica parte las siguientes premisas [2]:

- Sólo se consideran los permisos de tipo *dangerous*.
- Los grupos de permisos están formados por permisos de tipo *dangerous*.
- Las aplicaciones del mismo sector usan grupos de permisos parecidos.
- El *malware* tiende a pedir más permisos (y más peligrosos) que las aplicaciones benignas.

A continuación se formula la métrica. Se reproduce la formulación que aparece en [2].

$$M(a_i) = \frac{\sum_{j=1}^{m}(e_j \sum_{k=1}^{q} p_{jk})}{max\{I\}}$$

donde:

- A: conjunto de aplicaciones de la misma categoría tal que $A = \{a_1, a_2, ..., a_n\}$.
- n: número de aplicaciones presentes en la categoría A.
- a_i: aplicación dentro de la categoría tal que $a_i \in A$.
- $M(a_i)$: impacto sobre la privacidad de la aplicación a_i.
- m: número de grupos de permisos utilizados por la aplicación a_i.
- G_{a_i}: conjunto de grupo de permisos utilizados por la aplicación a_i tal que $G_{a_i} = \{g_1, g_2, ..., g_m\}$.
- q: número de permisos dentro de un grupo de permisos. utilizados por la aplicación a_i.
- P_{a_i}: conjunto de permisos utilizados por la aplicación a_i tal que $P_{a_i} = \{p_{jk}/j = \{1, ..., m\}, k = \{1, ..., q\}\}$
- p_{jk}: peso del permiso *k-ésimo* dentro del grupo de permisos *j-ésimo* utilizado por la aplicación $a_i \in A$.
- e_j: estado del grupo de permisos *j-ésimo* $\in \{0, 1\}$. 0 indica que el grupo de permisos está inactivo y 1 que está activo.
- I: conjunto con los valores de los impactos máximos para cada una de las apps analizadas dentro de la categoría A.
- P_D: conjunto de todos los permisos tipo Dangerous existentes en Android.

En la fecha de redacción de este documento aún no se cuenta con los valores máximos de cada categoría I, por lo que vamos a considerar I como el impacto máximo de una aplicación que utilice todos los permisos existentes de tipo Dangerous. Para los pesos p_{jk} vamos a utilizar las puntuaciones del ranking dado en [21], donde los permisos no recogidos en el ranking tendrán un peso de $1/|P_D|$. De esta forma se implementará una versión inicial de la métrica, que sirva para explorar los resultados, a partir de los cuales se plantearán mejoras posteriores.

Capítulo 4

Análisis

4.1. Sistema Global

Este TFG forma parte de la propuesta "Herramientas para la evaluación y comprensión del impacto sobre la privacidad de aplicaciones móviles por parte de los usuarios finales a través del análisis de los permisos", financiada con una Beca de Colaboración con el Departamento de Informática de la Universidad de Valladolid bajo la dirección de la profesora M. Mercedes Martínez González dentro del programa *Becas de colaboración destinadas a estudiantes universitarios para realizar tareas de investigación en departamentos universitarios*, del Ministerio de Educación y Formación Profesional.

El objetivo del Departamento es crear un sistema global que recopile y presente información sobre la privacidad aplicaciones móviles, dentro del cual se validará y se dará a conocer la métrica desarrollada. Esto se realiza mediante el desarrollo de tres Trabajos Fin de Grado en paralelo. El primero de ellos es este, que proporciona un servicio web que implemente la métrica de privacidad y presente los resultados a los usuarios finales de un modo comprensible. El segundo, desarrollado por el alumno Alejandro Pérez, proporciona un Warehouse de aplicaciones móviles que recopile datos sobre aplicaciones y privacidad que sirva como fuente de datos del servicio web y como repositorio sobre el que validar la métrica desarrollada. El tercero, realizado por Alejandro de la Cruz Garijo, proporcionará un estudio completo sobre el sistema de permisos Android, a través del desarrollo de una app que permita obtener datos como, por ejemplo, qué grupos de permisos se asignan por defecto. Este último trabajo se utilizará para realizar cambios y ajustes en la métrica de acuerdo con las conclusiones obtenidas.

En la figura 4.1 se muestra un diagrama con la arquitectura del sistema global. En él se puede ver cómo el servicio web desarrollado en este TFG será la "cara visible" del sistema. Este servicio tomará datos del Warehouse de aplicaciones móviles, que a su vez será utilizado para validar la métrica que incorporará las conclusiones obtenidas en el estudio sobre los permisos en Android.

4.2. Requisitos

A continuación se muestran los requisitos del sistema desarrollado en este TFG, desglosados en funcionales y no funcionales.

4.2.1. Requisitos funcionales

[RF1] El sistema permitirá obtener el impacto de privacidad para una aplicación especificada.

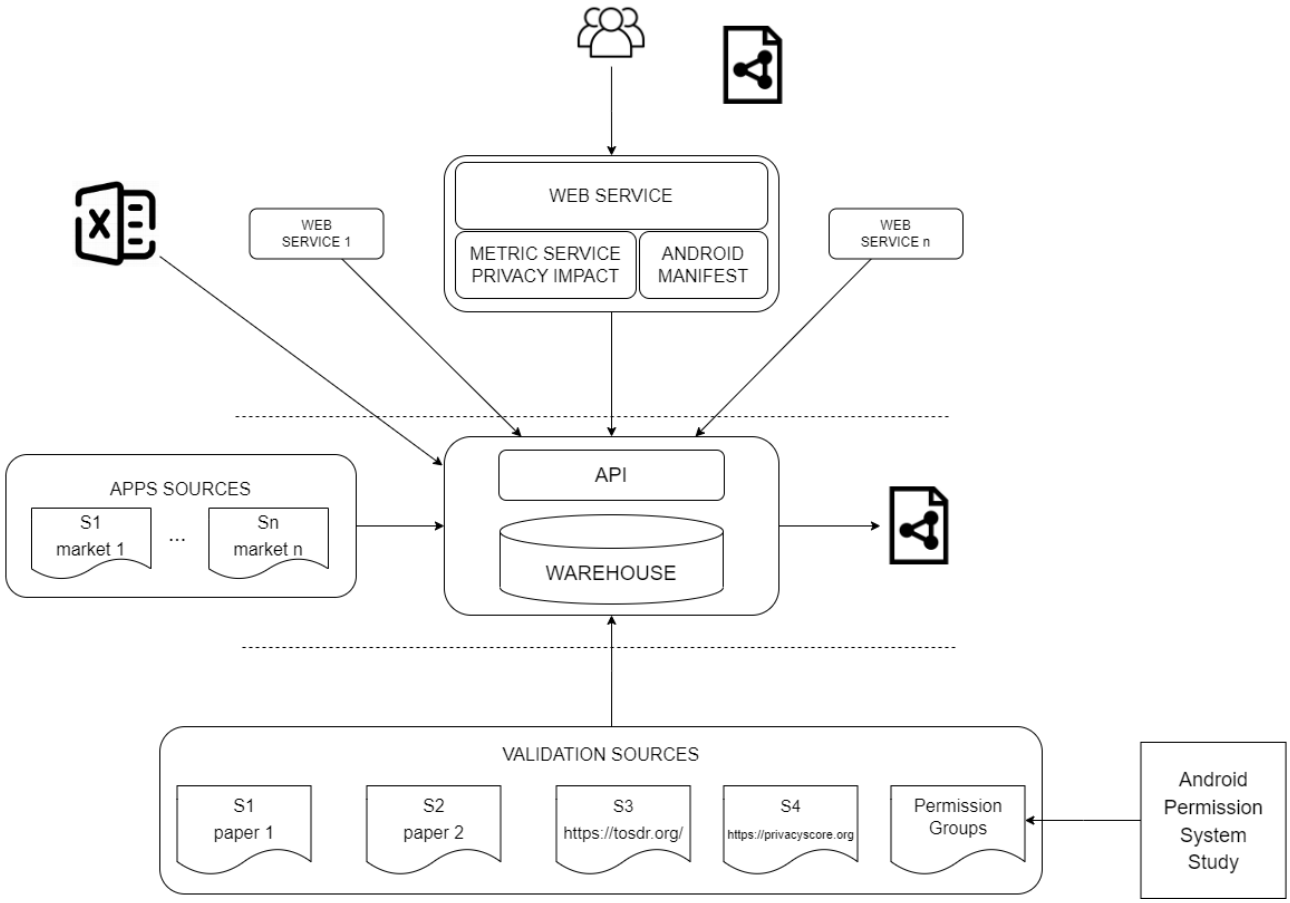


Figura 4.1: Diagrama con la arquitectura del sistema global dentro del cual se enmarca este TFG. Proporcionado por la tutora.

[RF2] El sistema permitirá simular distintos escenarios de concesión de permisos, que tendrán impacto en el valor de la métrica.
[RF3] El sistema permitirá evaluar archivos fuente proporcionados por los usuarios.
[RF4] El sistema permitirá exportar la puntuación de privacidad, junto con la información que fundamenta su cálculo.
[RF5] El sistema descargará y analizará las aplicaciones automáticamente en tiempo real al realizar la consulta del impacto de privacidad.
[RF6] El sistema almacenará la información de las aplicaciones consultadas en un repositorio de datos externos (Warehouse).

4.2.2. Requisitos no funcionales

[RNF1] El sistema permitirá especificar la aplicación a consultar mediante el nombre del paquete, la URL del market o el archivo .apk.
[RNF2] El sistema se implementará como un servicio web.
[RNF3] El sistema será robusto frente a la posible alteración de fuentes de datos externas.
[RNF4] El sistema será robusto ante caídas en las fuentes de datos.
[RNF5] El sistema desacoplará el cálculo de la métrica del resto de funcionalidades, pudiéndose actualizar en un futuro sin afectar a otras partes del servicio.
[RNF6] El sistema desacoplará los tipos de permisos considerados en la métrica del resto de funcionalidades, pudiéndose actualizar en un futuro sin afectar a otras partes del servicio.

[RNF7] El sistema soportará la caída del repositorio de datos externo, utilizando un repositorio de datos local como *backup*.
[RNF8] El sistema podrá ser desplegado en un servidor comercial.

Eficiencia

[RNF9] El sistema obtendrá la métrica asociada a una aplicación en un tiempo igual o inferior a 3 minutos en el 90 % de los casos para archivos fuente de tamaño menor o igual a 200MB y el servidor dedicado.

Seguridad

[RNF10] El sistema comprobará la extensión y formato de las aplicaciones subidas por los usuarios.
[RNF11] El repositorio de datos interno estará securizado ante intentos de intrusión.
[RNF12] El sistema utilizará las últimas versiones del software necesario para su implementación.
[RNF13] El sistema encapsulará los errores, de forma que no se revele información interna.

Experiencia de usuario

[RNF14] El sistema mostrará los datos de forma comprensible para un usuario no familiarizado con el sistema de permisos de Android, de modo que un 90 % de los usuarios encuestados en las pruebas de aceptación valoren la comprensibilidad de los resultados con una nota de, al menos, 3/5.
[RNF15] El sistema mostrará los resultados de forma gráfica adaptada al usuario no especializado.
[RNF16] El sistema permitirá obtener un informe de privacidad más detallado adaptado al usuario especializado.
[RNF17] El sistema mostrará las puntuaciones de aplicaciones similares a las que se está consultando a modo de comparación.
[RNF18] Al menos el 90 % de los usuarios serán capaces de consultar el impacto de privacidad de una aplicación (especificada mediante nombre de paquete, URL o archivo fuente) en un tiempo inferior a 30s (sin contar tiempos de carga).

4.3. Casos de uso

El sistema tiene un único caso de uso, consultar el impacto de privacidad de una aplicación, donde el único actor que interviene es el Usuario. A continuación se detalla su secuencia.

4.3.1. Consultar impacto de privacidad

Actor: Usuario
Precondición: Ninguna
Postcondición: La puntuación de la aplicación es mostrada al usuario

Secuencia principal:

1. El usuario introduce el nombre del paquete de la aplicación.
2. El sistema comprueba que la aplicación existe, la obtiene y muestra su puntuación de privacidad al usuario con todos los permisos concedidos.

3. El sistema solicita si se quiere simular la concesión/revocación de permisos, exportar los resultados o finalizar.

4. El usuario indica que desea finalizar.

5. El caso de uso finaliza.

Alternativas:

(1a) El usuario indica que desea introducir la URL de la aplicación.

1. El sistema solicita la URL.

2. El usuario envía la URL.

3. El sistema comprueba que la URL es válida y se corresponde con una aplicación existente.

4. El caso de uso continúa en el paso 2.

(1b) El usuario indica que desea introducir el archivo fuente de la aplicación.

1. El sistema solicita el archivo fuente.

2. El usuario envía el archivo fuente.

3. El sistema comprueba que el archivo fuente es válido y lo almacena.

4. El caso de uso continúa en el paso 2.

(3a) El usuario indica que desea simular la concesión/revocación de uno o varios permisos.

1. El sistema muestra una lista con los permisos y solicita los permisos a conceder/revocar.

2. El usuario indica los permisos que desea conceder/revocar.

3. El sistema muestra la puntuación actualizada.

4. El caso de uso continúa en el paso 3.

(3b) El usuario indica que desea exportar los resultados.

1. El sistema muestra una lista con los formatos de exportación disponibles y solicita el formato en que se desean exportar los datos.

2. El usuario indica el formato en que desea exportar los datos.

3. El sistema proporciona al usuario el archivo con los resultados.

4. El caso de uso continúa en el paso 3.

Excepciones:

(3, 1a.2, 1b.2, 3a.2, 3b.2) Si el usuario cancela, el caso de uso queda sin efecto.
(1a.3) Si la URL no es válida, el sistema informa del error y se vuelve al paso 1.
(1b.3) Si el archivo fuente no es válido, el sistema informa del error y se vuelve al paso 1.
(2) Si la aplicación solicitada no existe, el sistema informa del error y se vuelve al paso 1.
(2) Si el sistema no es capaz de obtener la aplicación se informa del error y finaliza el caso de uso.

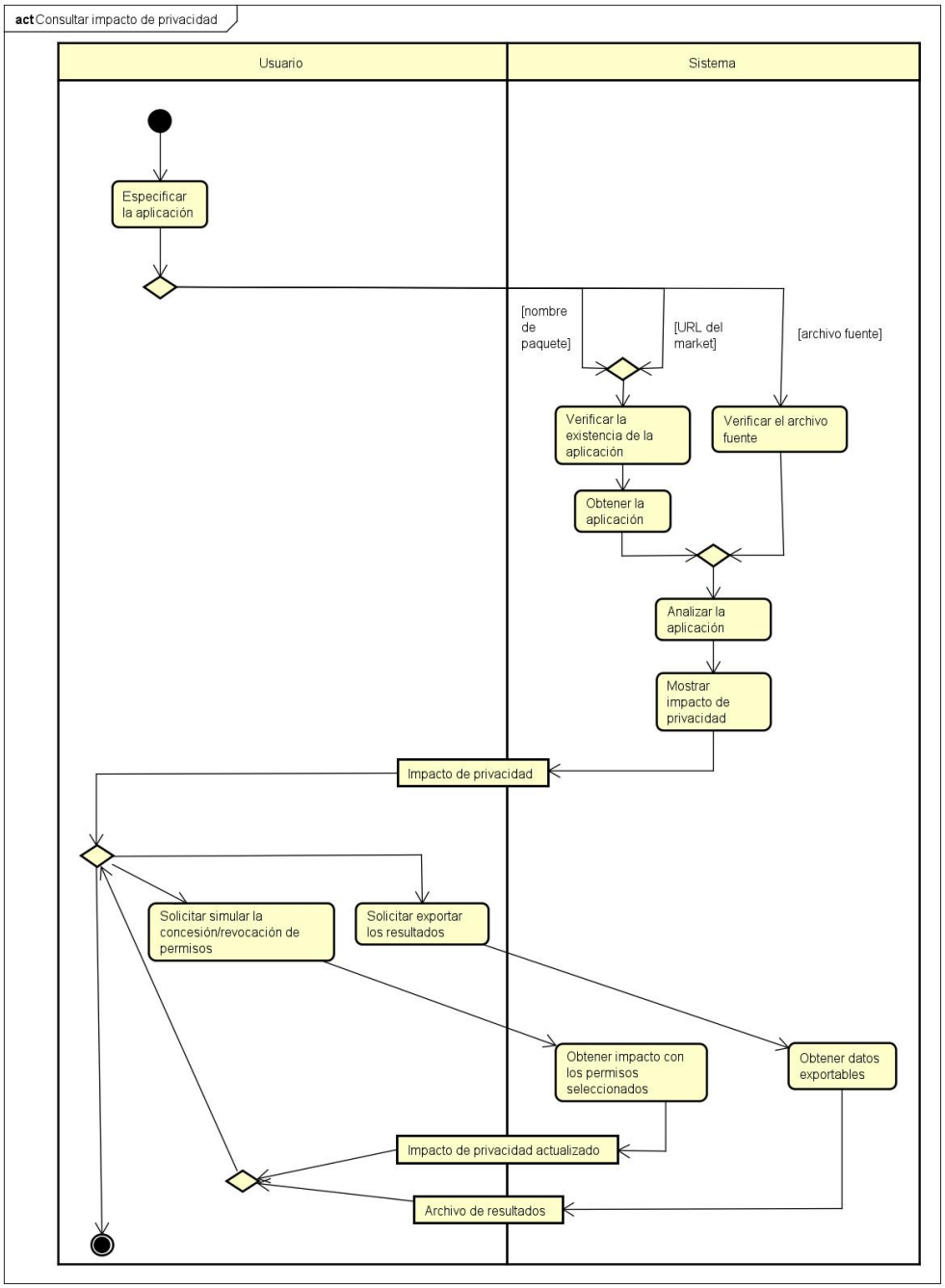


Figura 4.2: Diagrama de actividad del caso de uso Consultar métrica de privacidad.

Diagrama de actividad

En la figura 4.2 se muestra el diagrama de actividad del caso de uso Consultar métrica de privacidad.

4.4. Modelo de dominio

En la figura 4.3 se muestra el modelo de dominio del sistema. En él, la clase *App* es la representación de una aplicación móvil, con un nombre de paquete, versión y categoría asociados. Las clases *Permission* y *PermissionGroup* son representaciones de los permisos y grupos de permisos

de *Android*. Los permisos disponen de un grupo por defecto al que son asignados, pero también pueden ser asignados dentro de una aplicación a otro grupo distinto, eso es lo que se representa en la clase *PermissionAssignment*.

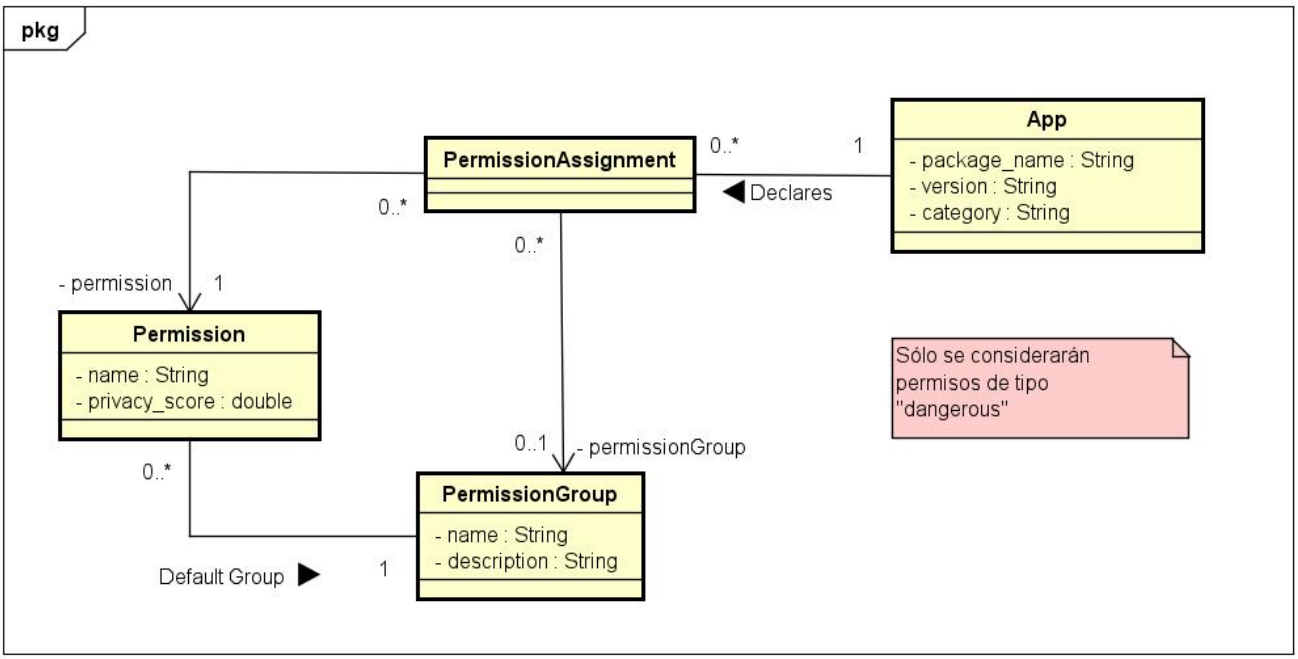


Figura 4.3: Modelo del dominio en análisis.

4.5. Realización en análisis de los casos de uso

En las figuras 4.4 4.6, 4.7 y 4.5 se muestra el diagrama de secuencia en análisis correspondiente al caso de uso Consultar impacto de privacidad.

4.6. Análisis de las fuentes de datos

Al estar realizando un Sistema Integrador de la información, es necesario un análisis previo de las fuentes de datos que se describe a continuación.

4.6.1. *Warehouse* de aplicaciones móviles

Este proyecto se realiza en paralelo al TFG del alumno Alejandro Pérez de la Fuente, que tiene como objetivo la creación de un Warehouse de integración de la información de aplicaciones móviles. En él se almacenará, entre otros datos, la información sobre los permisos declarados por las distintas aplicaciones móviles. Este proyecto utilizará el Warehouse como fuente principal de información, de forma que no sea necesario descargar la aplicación (lo que resultaría lento y costoso) si se puede consultar la información directamente en el Warehouse. Sin embargo, es necesario que el servicio desarrollado funcione de manera independiente al Warehouse, por lo que es necesario el uso de más fuentes de datos.

4.6.2. *Markets* de descarga de aplicaciones

En caso de que el Warehouse falle, se recurrirá a la descarga directa de la aplicación. La fuente de datos natural para ello sería *Google Play*[1], que es el mayor mercado de aplicaciones Android. Sin embargo, se requiere de autenticación y dispositivos vinculados para poder descargar aplicaciones, por lo que su uso automatizado desde un servidor resulta costoso y complicado. Aunque hay repositorios públicos (por ejemplo, `https://github.com/ClaudiuGeorgiu/PlaystoreDownloader`, en *Github*) que usan la API de Google Play para obtener los archivos `.apk`, estos resultan complicados de instalar y configurar, amén de que son repositorios sin ningún tipo de garantía de seguridad en los que hay que introducir datos personales. Algo parecido ocurre en *Amazon Store*[2] y *Galaxy Store*[3], otros dos markets de aplicaciones móviles muy populares. Por tanto, hay que buscar fuentes alternativas de las que descargar las apps. Estas fuentes alternativas presentan dos problemas principales. El primero es que son muy propensas a cambios de formato y/o de interfaz. El segundo es que, al tratarse en muchos casos de fuentes creadas por equipos pequeños e independientes, pueden ser inestables y dejar de estar disponibles de un momento a otro. Es por ello que utilizaremos una fuente principal para la descarga de aplicaciones y una de refuerzo en caso de que falle la principal. En cualquier caso, el sistema se diseñará de forma que las fuentes de datos sean fácilmente intercambiables.

La primera fuente a utilizar en caso de que el Warehouse no esté disponible será *APKPure*[4], un market alternativo en el que no se necesita autenticación y se descargan directamente los archivos `.apk`, sin necesidad de vincular un dispositivo como en los markets oficiales.

La segunda fuente a utilizar será *Evozi Apk Downloader*[5], de funcionamiento y características análogas a *APKPure*.

Por último, se utilizará *Google Play* como fuente auxiliar para obtener el nombre comercial y la categoría asociados a una aplicación, así como la imagen de su logotipo. En caso de que este fuente falle no se recurrirá a una alternativa, ya que la categorización de las apps varía mucho entre fuentes de datos, sino que se dejará vacía.

4.7. Modelo de usuario

Este servicio es público y de código abierto, por lo que es accesible para todo el mundo que disponga de conexión a internet. Sin embargo, se espera que el usuario promedio sea un usuario consciente del riesgo en la privacidad de los dispositivos móviles o simplemente curioso respecto al tema, por tanto se le presupone cierta familiaridad con el lenguaje asociado las aplicaciones móviles (que entienda términos como permiso en Android, intrusividad o privacidad). En cualquier caso, se buscará que los primeros resultados que se muestren sean comprensibles para el público general, recurriendo a gráficos y otros recursos visuales que apoyen los resultados sin necesidad de entrar en un lenguaje técnico. Como extensión, se proporcionara un informe más detallado de los resultados de cara a un usuario más especializado con cierto conocimiento técnico. En este caso el género u otras condiciones personales son irrelevantes en el uso del servicio.

[1] `https://play.google.com/store/`
[2] `https://www.amazon.es/mobile-apps/`
[3] `https://www.samsung.com/es/apps/galaxy-store/`
[4] `https://apkpure.com/es/`
[5] `https://apps.evozi.com/apk-downloader/`

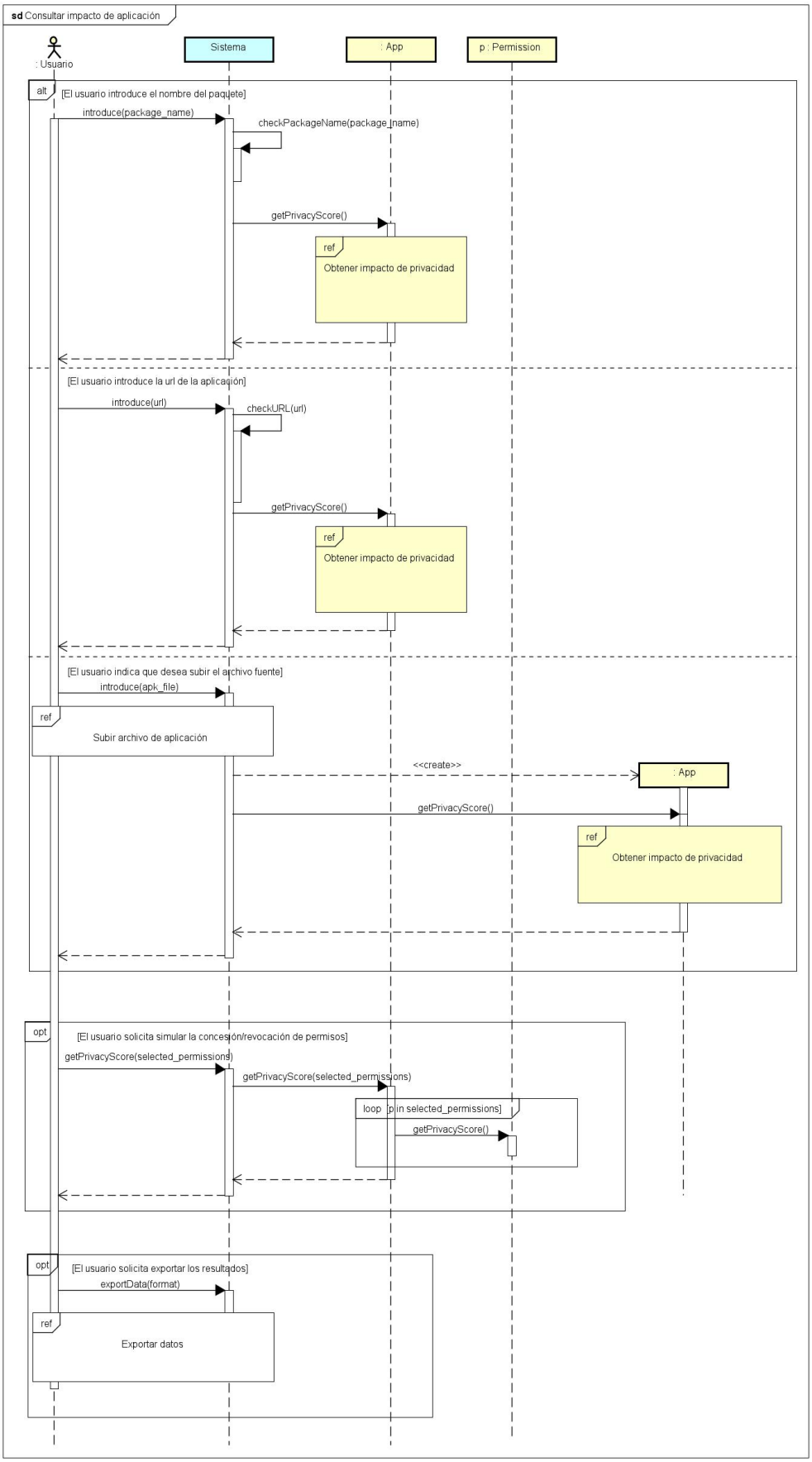


Figura 4.4: Diagrama de secuencia en análisis del caso de uso Consultar métrica de privacidad.

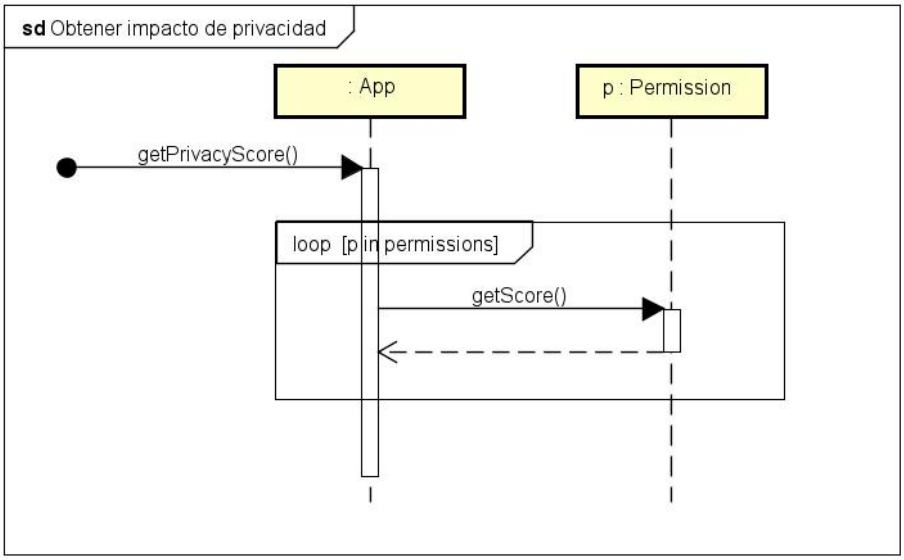


Figura 4.5: Diagrama de secuencia en análisis correspondiente a subir obtener el impacto de privacidad de una aplicación.

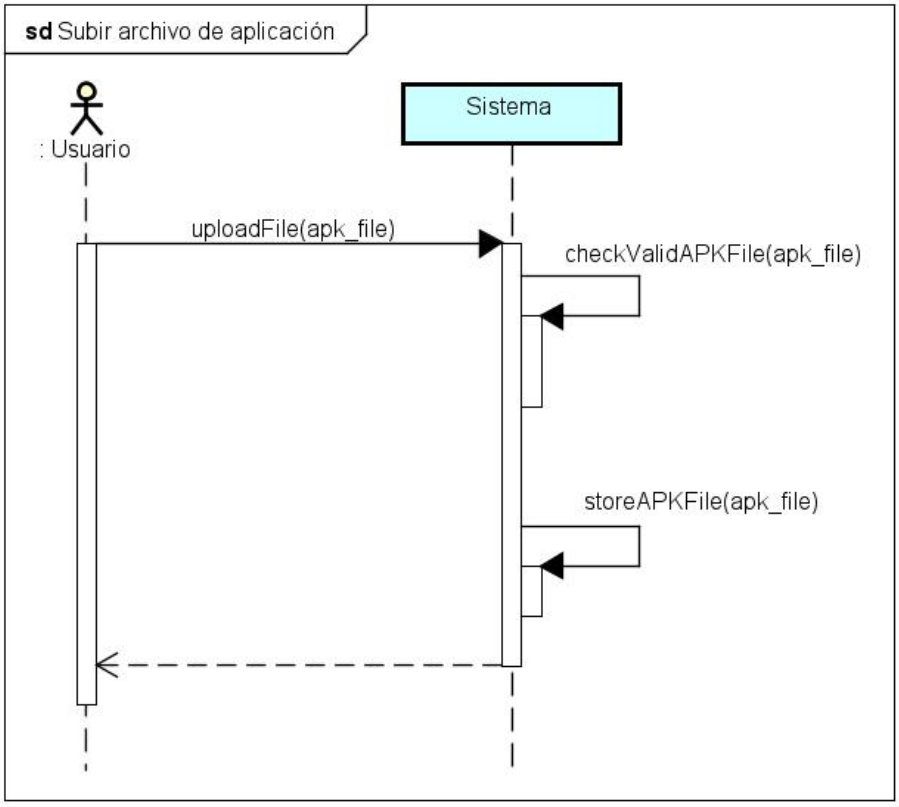


Figura 4.6: Diagrama de secuencia en análisis correspondiente a subir un archivo de aplicación.

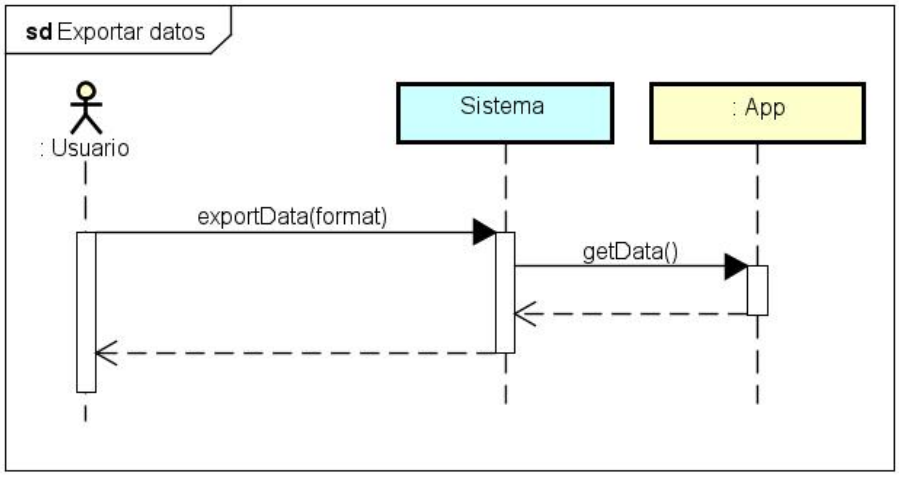


Figura 4.7: Diagrama de secuencia en análisis correspondiente a exportar datos.

Capítulo 5

Diseño

5.1. Entorno tecnológico

A continuación se listan las tecnologías que se utilizan para el desarrollo del proyecto.

1. *Python y Django*: Se utilizará Python como lenguaje de desarrollo, debido a su simplicidad y versatilidad. En concreto, se utilizará el framework web Django para realizar el servidor web. Este framework trae implementado el patrón Modelo-Vista-Plantilla de forma nativa y proporciona muchas facilidades para el desarrollo y pruebas del sistema.

2. *SQL y SQLite*: La persistencia se manejará a través de SQL, en concreto el repositorio local de datos se implementará con una base de datos SQLite, que implementa Django por defecto.

3. *Selenium*: Esta biblioteca de Python se utilizará para el Web scraping a la hora de obtener las aplicaciones de las fuentes. En concreto se utilizará el navegador web Firefox en su modo *headless*.

4. *HTML, CSS y JavaScript*: Debido a la naturaleza del sistema, se utilizarán plantillas (*templates*) HTML para realizar el front-end en forma de páginas web dinámicas. Django proporciona un renderer que nos permite incorporar variables en el front-end de forma muy sencilla. Se utilizará CSS y JavaScript para los elementos de estilo y la funcionalidad que tenga que ser implementada en el front-end (por ejemplo, barras de progreso o tooltips).

5. *D3.js*: Esta biblioteca gráfica de JavaScript se utilizará para realizar los gráficos y visualizaciones que se presentan cuando el usuario solicita el informe de privacidad de una aplicación.

6. *Bootstrap*: Un framework de diseño web que utiliza HTML, CSS y JavaScript para crear sitios web responsivos con una apariencia atractiva y consistente en diferentes dispositivos. Se utilizará para la organización de los elementos del front-end.

7. *Git* y *GitLab*: Se utilizará Git para el control de versiones y GitLab para crear el repositorio en la nube donde se alojará el código, así como para llevar un control del tiempo empleado en el desarrollo y las tareas a realizar.

8. *Poetry*: Se utilizará la librería Poetry de Python para la gestión de dependencias del proyecto. Esta librería permite gestionar las dependencias de modo sencillo y fácilmente exportable a otras máquinas.

9. Despliegue: Se utilizará una máquina virtual Linux proporcionada por la Escuela para el despliegue del servicio.

5.2. Patrones de diseño

A continuación se detallan los patrones de diseño en los que se fundamenta el servicio. No se pretende realizar una descripción exhaustiva de todos aquellos presentes en Django, sino que solamente se incluyen los que afectan directamente al servicio y aquellos que los desarrolladores podemos controlar y personalizar.

5.2.1. Servidor Web: Patrones MVC y MVT

El patrón de diseño Modelo-Vista-Controlador (MVC) es una arquitectura de software que separa una aplicación en tres componentes principales: el modelo, la vista y el controlador. El modelo representa los datos y la lógica de negocio, la vista se encarga de la presentación de la información al usuario y el controlador maneja las solicitudes del usuario y actualiza el modelo y la vista según sea necesario. El objetivo de este patrón es separar la lógica de presentación de la lógica de negocio para facilitar la comprensión, el mantenimiento y la escalabilidad del código. En Django, se implementa una variante de este patrón conocida como *Model-View-Template* (MVT). En este patrón (ver figura 5.1), el modelo representa los datos y la lógica de negocio, la vista maneja las solicitudes del usuario y la generación de respuestas, y el template se encarga de la presentación de la información al usuario. La vista se comunica con el modelo para obtener o actualizar los datos y utiliza el template para generar la respuesta. No existe un controlador explícito, la vista hace vez de controlador y de controlador de la vista en el sentido en que se encarga de "rellenar" los templates con la información necesaria.

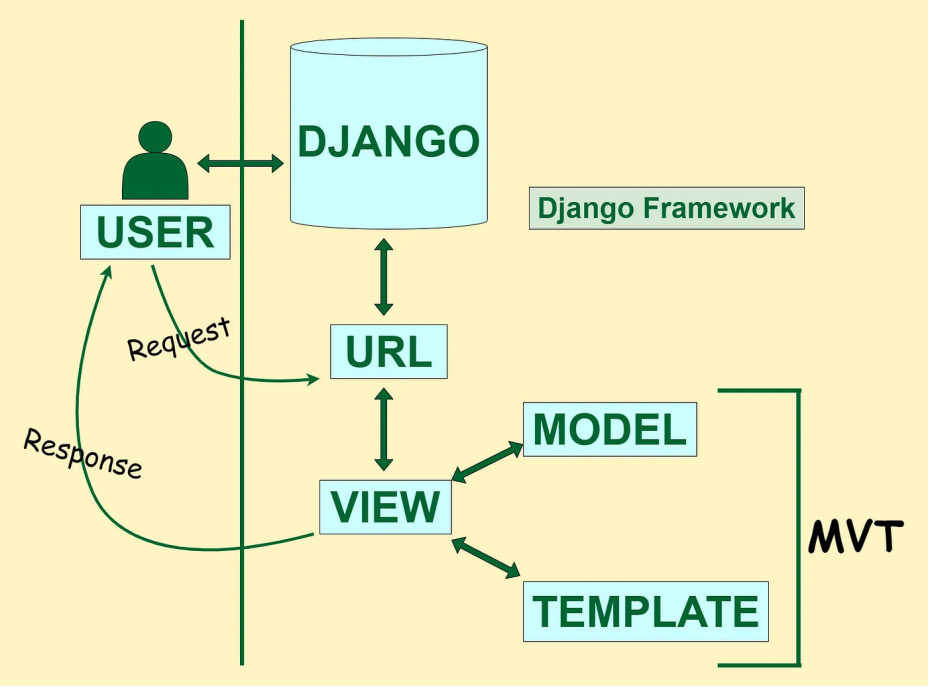


Figura 5.1: Diagrama del patrón MVT, extraída de [12].

Para este proyecto, debido a que tenemos que incorporar un Sistema Integrador, realizaremos una modificación a este patrón (ver figura 5.2), añadiremos un controlador entre la Vista y el Modelo, de forma que este nuevo controlador sea el que acceda al Sistema Integrador y no lo haga la propia Vista, que sólo se encarga de renderizar los templates y ser en punto de entrada por el

que se comunica con el controlador. De esta forma, estaremos siendo más fieles al patrón MVC original.

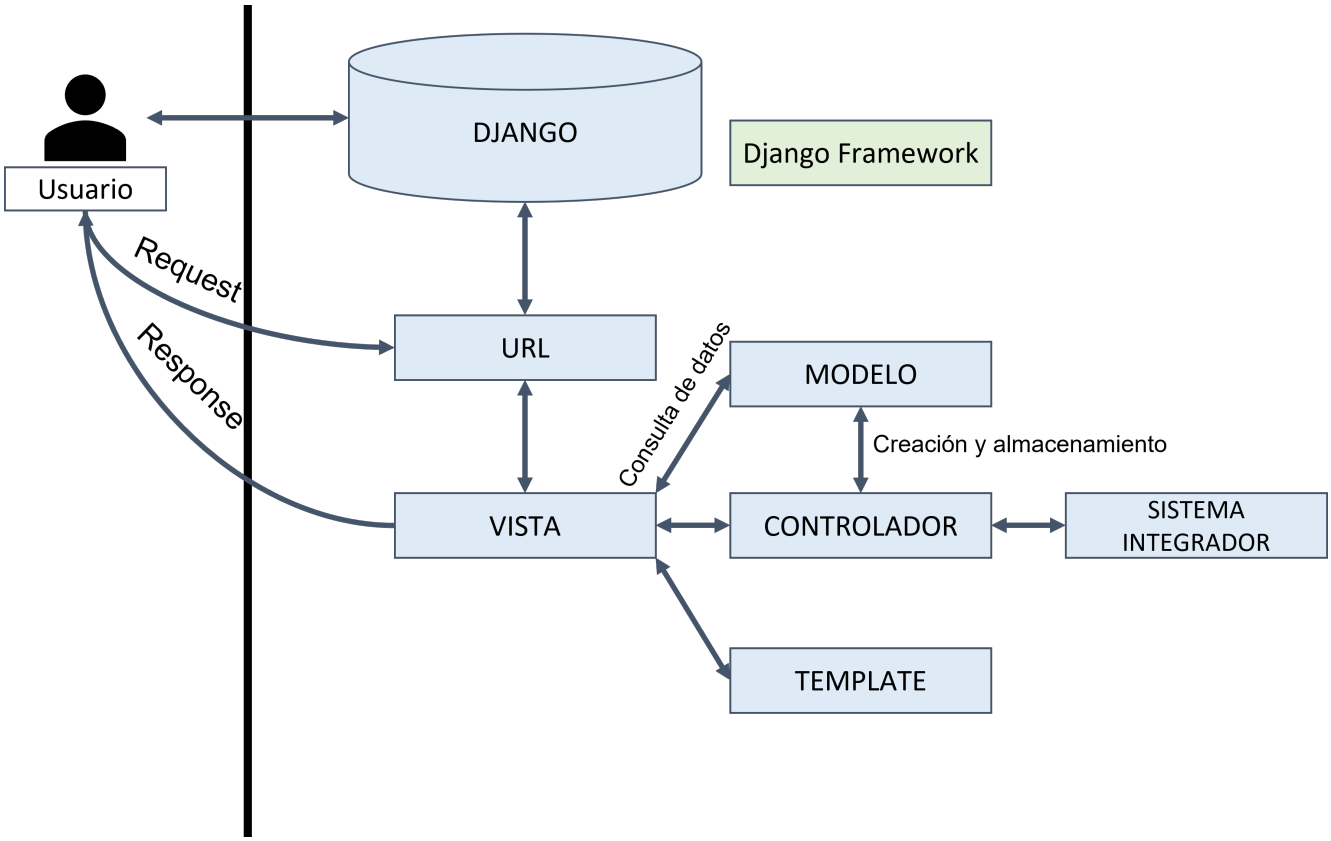


Figura 5.2: Diagrama de la adaptación del patrón MVT a nuestro problema, diagrama basado en [12].

Al tratarse de un servicio web, se implementará la variante pasiva del patrón MVC, pues el usuario solicita cambios en el Modelo mediante peticiones al servidor, por lo que los cambios en el modelo a partir del *input* del usuario en la vista los hace el Controlador. El Modelo del sistema almacenará la información relativa a aplicaciones, permisos, grupos, etc. Así como la lógica del cálculo de la métrica de privacidad. La Vista estará formada por el conjunto de páginas web dinámicas (y sus controladores, entendidos como controladores de vistas que interactúan con el Controlador principal) que se crearán a partir de la información del modelo que proporciona el Controlador.

Requests en Django

En la figura 5.3 se muestra el diagrama de secuencia correspondiente al procesamiento de una request HTTP en Django, que fundamenta el resto de diagramas de secuencia que se mostrarán. Se puede ver que el servidor determina la vista que tiene que usar como respuesta a una petición haciendo una búsqueda en el directorio de patrones de URL disponibles (dado por `ROOT_CONF_URL`) buscando aquel que cuadre con la URL que recibe en la petición. En caso de que no haya ninguna vista que concuerde con la URL, se responde con una vista de error por defecto.

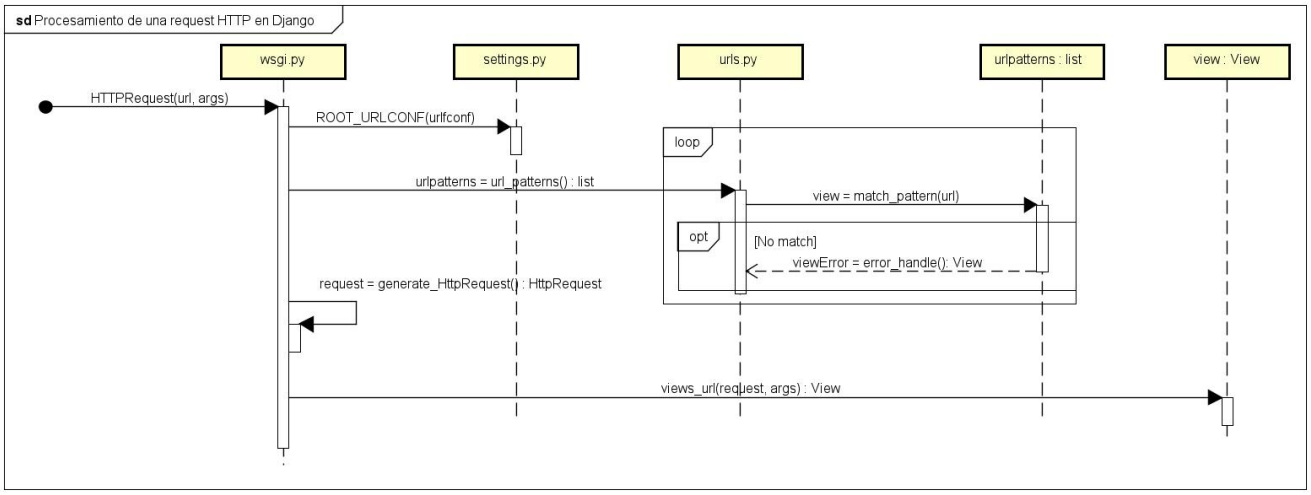


Figura 5.3: Procesamiento de una *request HTTP* en Django.

5.2.2. Vistas y templates: Patrón *Page Controller*

El patrón *Page Controller* se utiliza muy comúnmente en el desarrollo de aplicaciones web. En este patrón, cada página o vista del servicio tiene su propio controlador, que se encarga de manejar las interacciones con el usuario y coordinar la lógica de presentación y negocio asociada con esa página específica. En el contexto de Django se sigue este enfoque mediante el uso de vistas basadas en plantillas HTML y funciones como controladores de página. Cada controlador de página se encarga de recibir las solicitudes del usuario, procesar los datos enviados y generar la respuesta correspondiente. El controlador de página interactúa con el modelo de datos y otros componentes del sistema para realizar operaciones como validar datos, acceder al Sistema Integrador y generar contenido dinámico (como mensajes de error) para ser presentado al usuario. El patrón *Page Controller*, como se puede ver en la figura 5.4, proporciona una separación clara de las responsabilidades y permite un diseño modular, lo que facilita el mantenimiento y escalado de aplicaciones web.

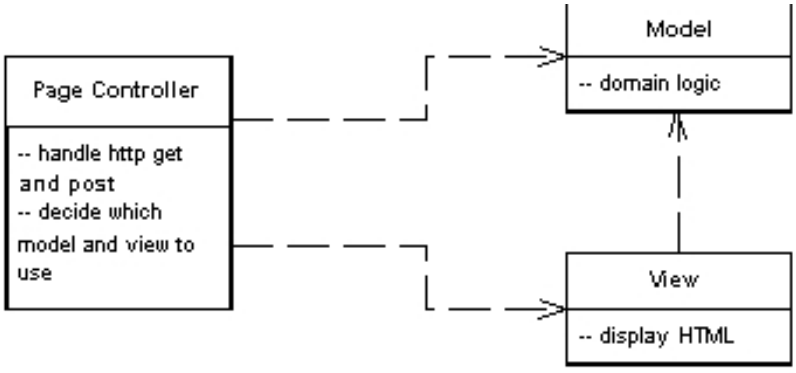


Figura 5.4: Esquema del patrón *Page Controller*, extraído de [13].

5.2.3. Persistencia local: Patrón *Active Record*

En Django, el almacenamiento persistente de objetos del modelo se realiza mediante el patrón *Active Record*. Este patrón de diseño se centra en la representación de objetos del dominio como

registros de una base de datos relacional. En el contexto de Django, cada modelo definido en la aplicación actúa como una clase *Active Record*, que encapsula tanto los datos como la lógica de negocios asociada. Esto significa que los desarrolladores pueden interactuar con la base de datos utilizando métodos y propiedades directamente en los objetos del Modelo, sin tener que escribir consultas SQL manualmente. Django proporciona una capa de abstracción que se encarga de la persistencia de los objetos en la base de datos, realizando automáticamente operaciones de creación, lectura, actualización y eliminación (*CRUD*) de los registros. A este patrón se le han hecho algunas críticas, pues son los objetos del modelo los que se encargan de almacenarse a sí mismos en la base de datos, violando así la separación entre lógica de negocio y persistencia, ya que los objetos del modelo tienen que disponer de información sobre cómo está implementado el almacenamiento persistente. Sin embargo, en este caso no supone ningún problema para el proyecto que se aborda.

5.2.4. Sistema Integrador: Patrón Fachada y DAO

El patrón fachada se utiliza para proporcionar un punto de entrada único a una parte del sistema, encapsulando su funcionamiento de forma que se permita al cliente abstraerse de la implementación. Esta fachada se utilizará para el Sistema Integrador, que proporcionará dos funciones de entrada en un módulo main: `retrieve_app_info` y `handle_uploaded_file`. Estas serán las únicas funciones que el controlador del servidor web conoce, aunque internamente el Sistema Integrador esté compuesto de muchos módulos que separan las funcionalidades. Cabe recordar que entre los requisitos del sistema está el ser flexible ante cambios en el modo de computar la métrica o el tipo de permisos considerados en ésta, por lo que mediante el patrón fachada estamos desacoplando acciones como la extracción de los permisos que declara una aplicación que nos permite cambiar su funcionamiento sin afectar al resto del sistema.

El patrón DAO (*Data Access Object*) es un patrón de diseño que separa la lógica de acceso a datos de la lógica de negocio en una aplicación. Esto se logra mediante la creación de una capa de abstracción entre la lógica de negocio y el almacenamiento de datos, lo que permite que el almacenamiento de datos sea cambiado sin afectar el resto de la aplicación. El DAO proporciona una interfaz para acceder a los datos de forma coherente, independientemente del tipo de almacenamiento utilizado. Este patrón se utilizará para la obtención de datos de una nueva aplicación, el Sistema Integrador hará de DAO (aunque no en el sentido estricto del patrón, ya que se accede mediante funciones y no mediante un objeto de acceso a datos) y proporcionará la información sobre una aplicación en formato `JSON` (un formato "neutral" que no tiene nada que ver con el modelo del sistema) al controlador, que se encargará de crear el objeto del modelo correspondiente a partir de esta información. De esta forma, evitamos acoplar el modelo con el Sistema Integrador.

5.2.5. *Logging*: Patrón Singleton

El patrón de diseño Singleton es un patrón de creación que garantiza que una clase tenga solo una instancia y proporciona un punto de acceso global a ella. Esto se logra mediante la creación de una única instancia de la clase y la restricción de su creación a través de un constructor privado. Este patrón se utilizará para el *logging* del sistema, donde crearemos un único objeto *logger* al que accederán el resto de componentes del sistema de forma que los *logs* se unifiquen y uniformicen en un único objeto que será el encargado de mostrar la información y definir el formato de los mensajes.

5.3. Arquitectura

El sistema se compone de dos partes claramente diferenciadas: un Servidor Web y un Sistema Integrador. A estas dos partes se les suman dos repositorios de datos: uno local y otro externo. El repositorio de datos externo es un Warehouse de Integración de la Información de aplicaciones móviles elaborado por el alumno Alejandro Pérez de la Fuente en el marco de su Trabajo Fin de Grado. El repositorio de datos locales funcionará como "backup" del repositorio remoto, se utilizará en caso de que éste no esté disponible. En la figura 5.5 se muestra la arquitectura del servicio completo.

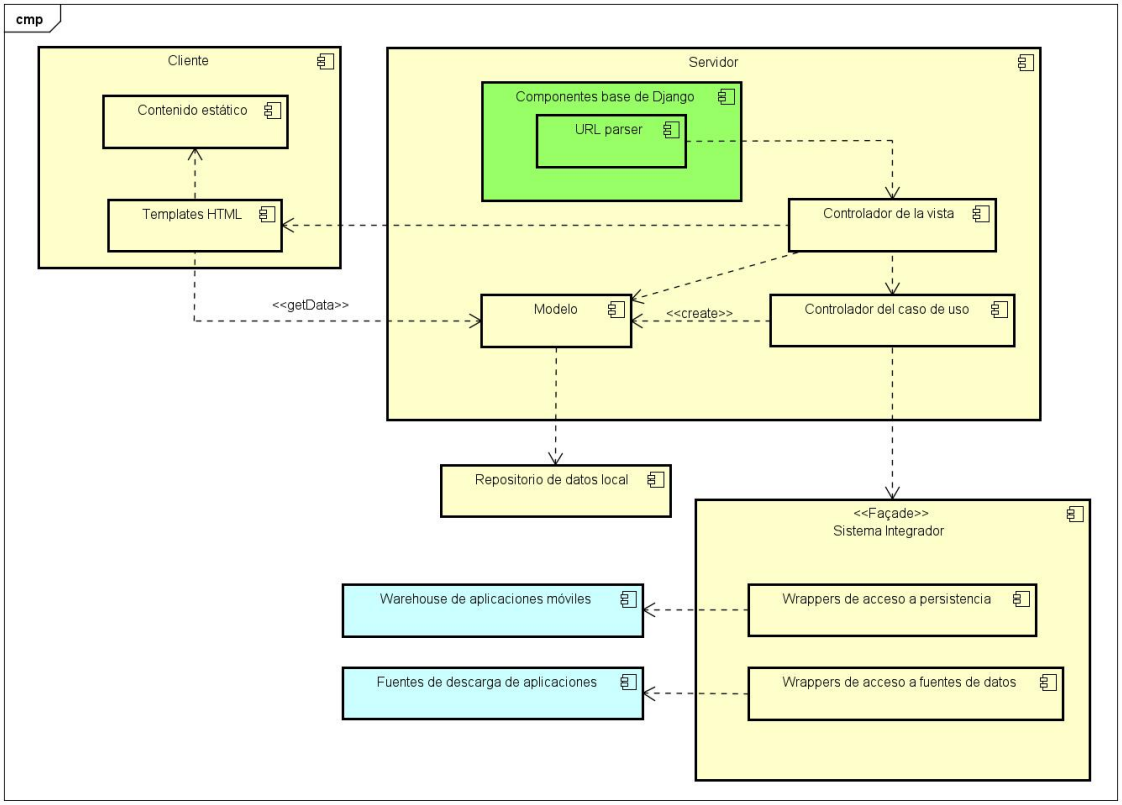


Figura 5.5: Diagrama de arquitectura del servicio.

El servidor web dará respuesta a las peticiones en forma de documentos HTML (o documentos json, xml, etc. En caso de que se solicite la exportación de datos). Éste utilizará el patrón de diseño Modelo, Vista, Controlador donde el controlador se encargará de solicitar al Sistema Integrador la información para crear objetos del Modelo, así como el almacenamiento persistente de éstos. El Sistema Integrador proporciona una fachada de acceso al Servidor, de forma que éste último no tenga que conocer la implementación del Sistema Integrador.

5.3.1. Arquitectura del Sistema Integrador

El Sistema Integrador tendrá tres funciones: la primera será abstraer al resto del servicio de la comunicación con el Warehouse de aplicaciones móviles, tanto para la obtención de metadatos de las aplicaciones como para la inserción de nuevas aplicaciones en el Warehouse. La segunda será la de descargar aplicaciones (archivos `.apk`) de las fuentes de datos de descarga y procesar estos archivos fuente para obtener los metadatos de las aplicaciones adaptados al modelo de dominio que maneja el servidor. La última función, en parte solapada con la segunda, es la de procesar los

archivos fuente de aplicaciones subidos por los usuarios al servidor, proporcionando los metadatos de las aplicaciones. En la figura 5.6 se puede ver un diagrama con las fuentes a las que accede el Sistema Integrador (el diseño de la comunicación con éstas se detalla en la sección 5.16).

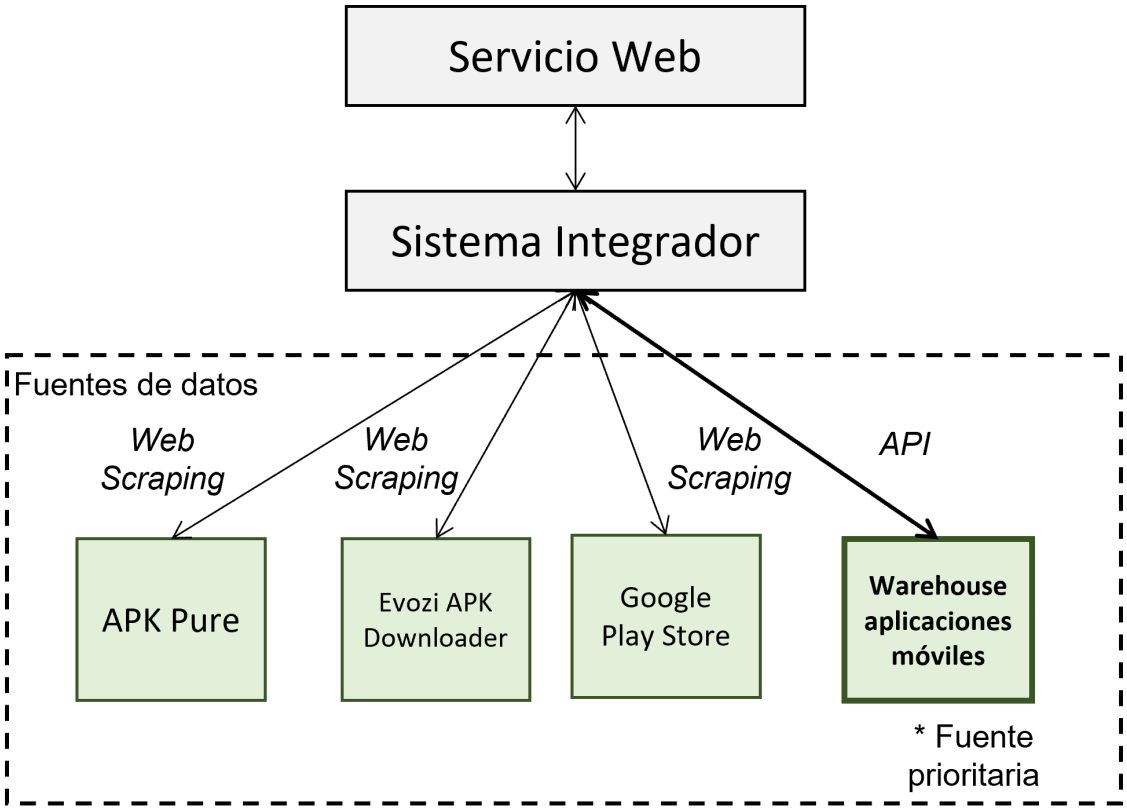


Figura 5.6: Diagrama de arquitectura del Sistema Integrador.

5.4. Descomposición en módulos

En la figura 5.7 se muestra la descomposición del sistema en módulos. Se omiten aquellos módulos que trae Django de base, pues no son específicos de este proyecto y su contenido y función ya se ha explicado en secciones previas.

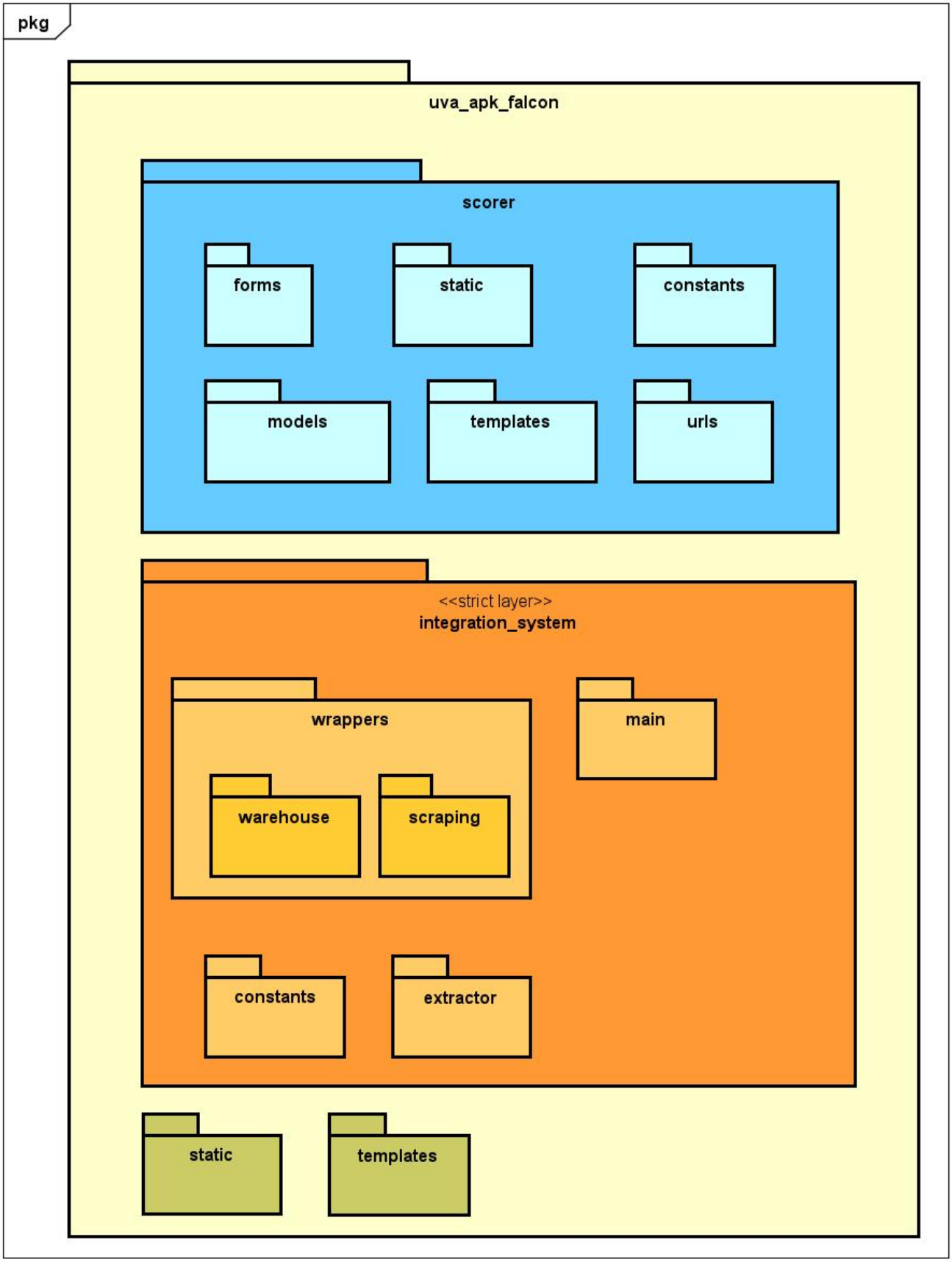


Figura 5.7: Descomposición del sistema en módulos.

5.5. Diagrama de clases del Servidor

En la figura 5.8 se muestra el diagrama de clases del Servidor para los componentes relacionados con el back-end[1].

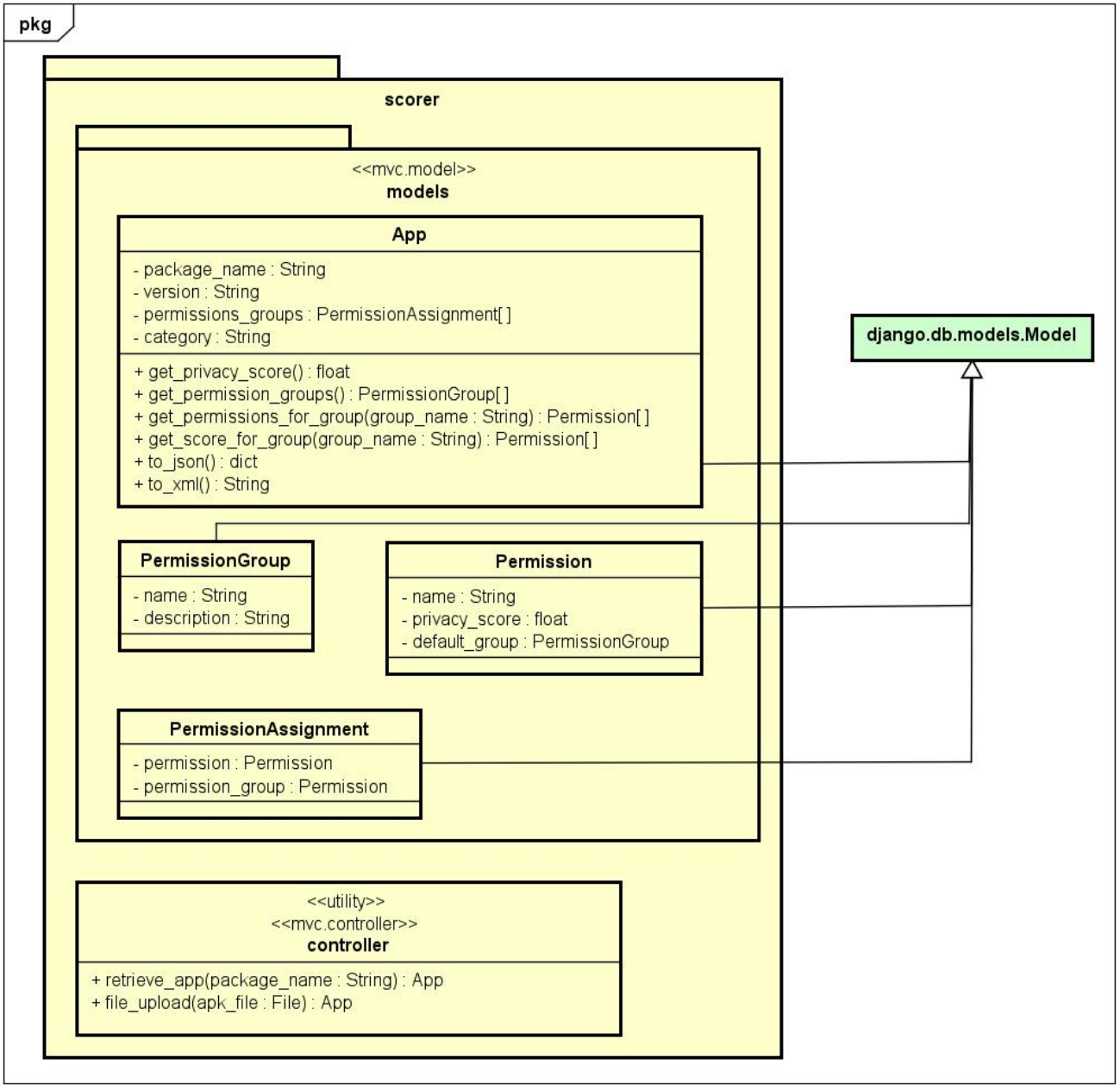


Figura 5.8: Diagrama de clases para los componentes del servidor relacionados con el back-end.

En la figura 5.9 se muestra el diagrama de clases del Servidor para los componentes más relacionadas con el front-end.

[1]Todos los componentes se nombran usando la notación estándar de Python[18]

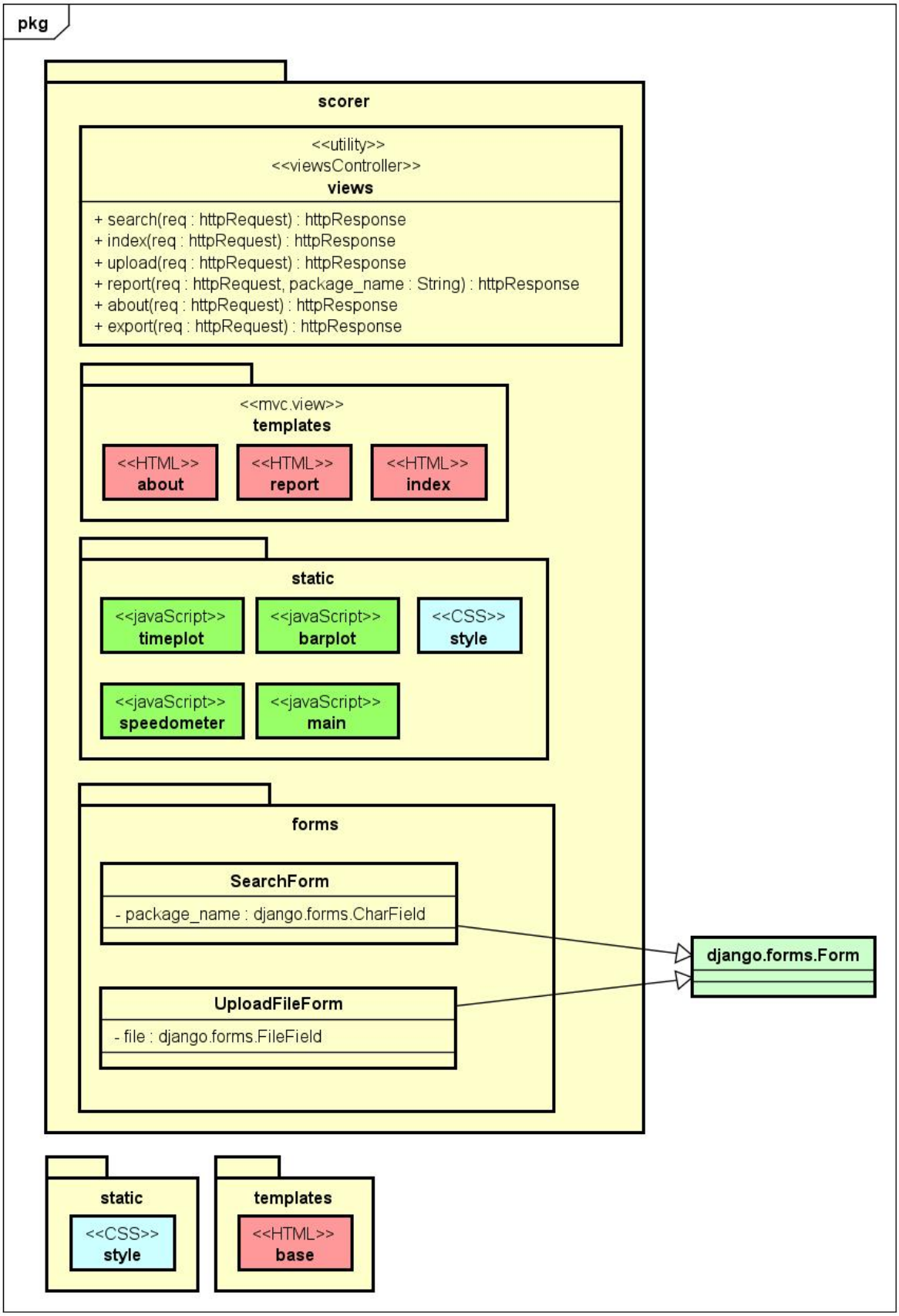


Figura 5.9: Diagrama de clases para los componentes del servidor relacionados con el front-end.

5.6. Diagrama de clases del Sistema Integrador

En la figura 5.10 se muestra el diagrama con las clases del Sistema Integrador.

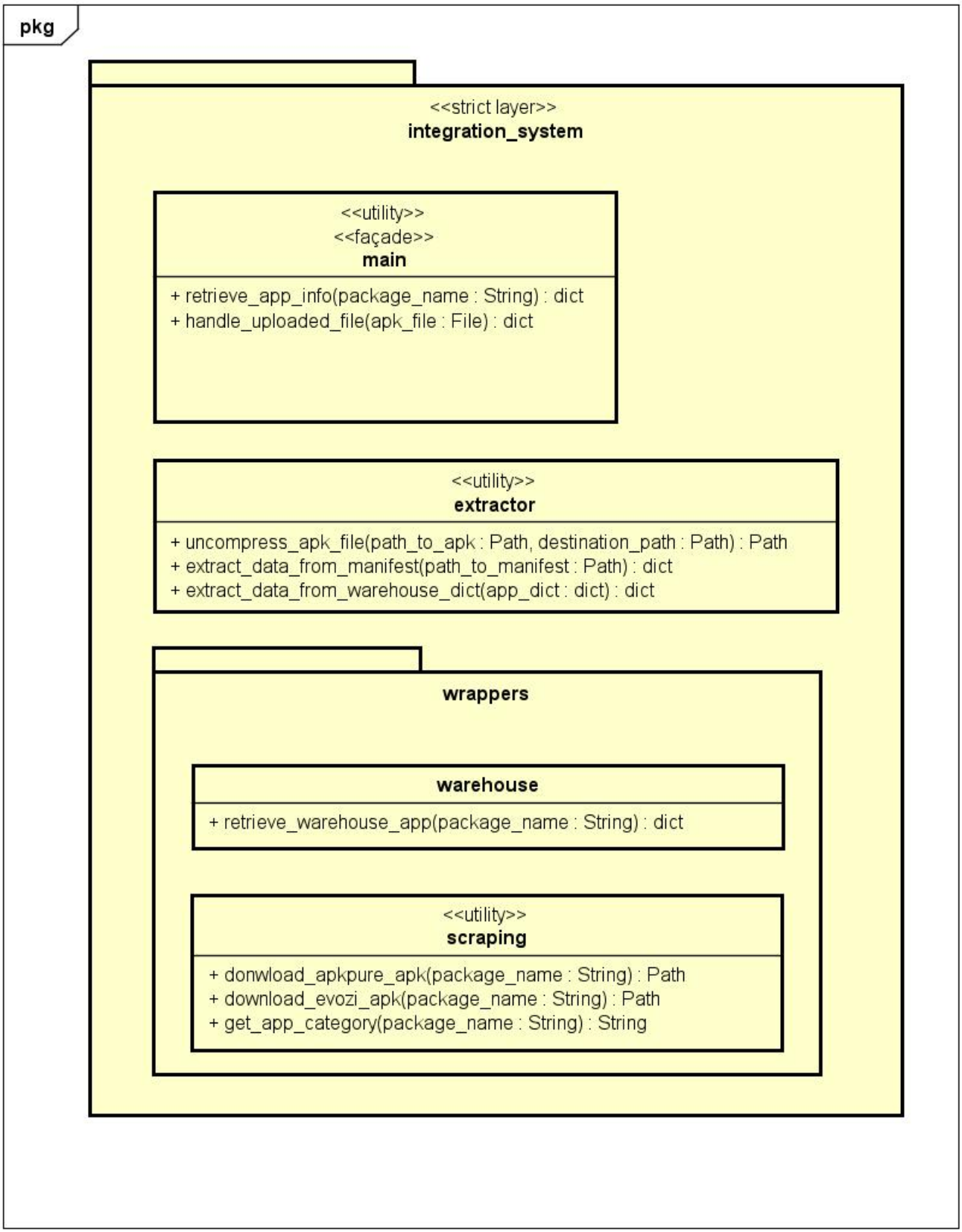


Figura 5.10: Diagrama de clases para los componentes del Sistema Integrador.

5.7. Interacción entre los componentes

A continuación se muestra el modo en que interactúan los distintos componentes del sistema de acuerdo con la arquitectura y patrones de diseño propuestos. Debido a su complejidad, se ha dividido en tres diagramas.

5.7.1. Interacción en el front-end

En la figura 5.11 se muestra la interacción entre los componentes del Servidor que guardan relación con el front-end del sistema.

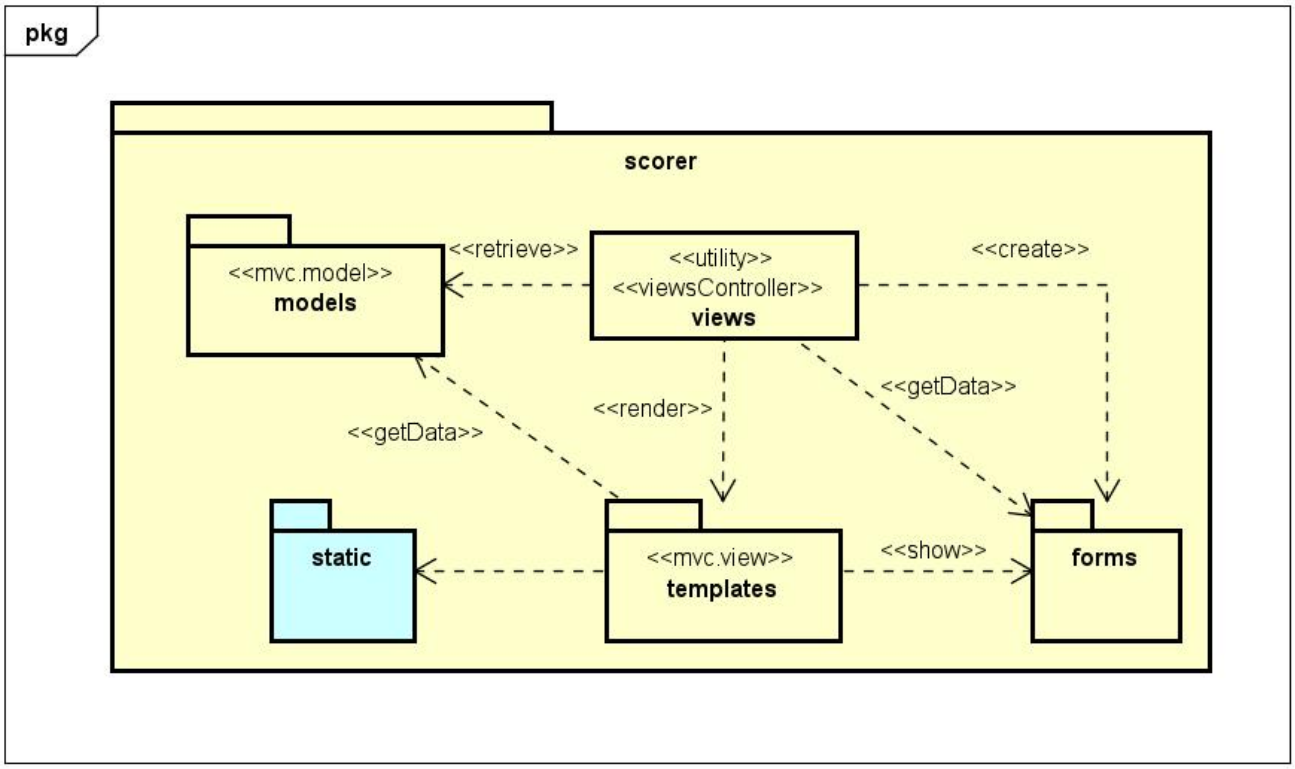


Figura 5.11: Diagrama de interacción de los componentes del front-end.

5.7.2. Interacción en el back-end

En la figura 5.12 se muestra la interacción entre los componentes del Servidor que guardan relación con el back-end del sistema.

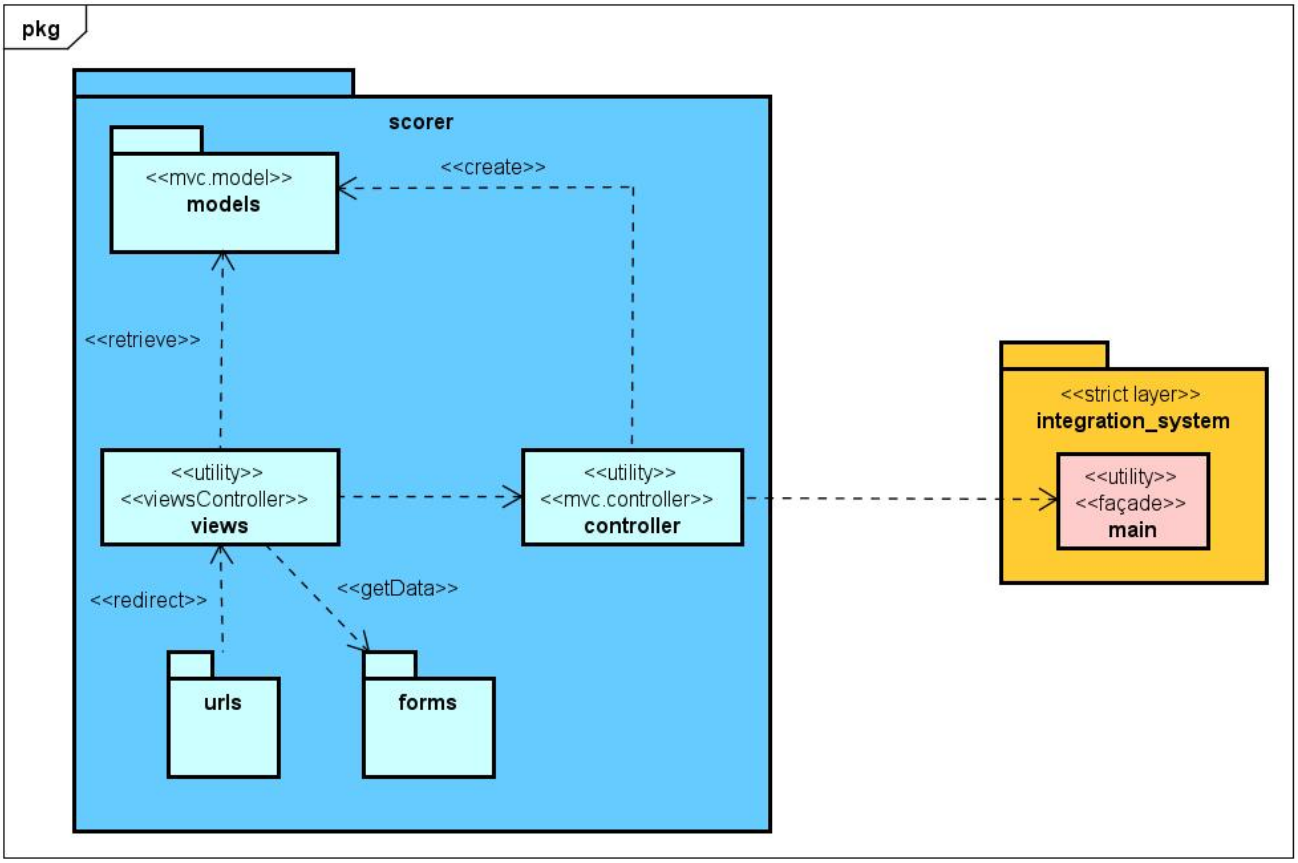


Figura 5.12: Diagrama de interacción de los componentes del back-end.

5.7.3. Interacción en el Sistema Integrador

En la figura 5.13 se muestra la interacción entre los componentes del Sistema Integrador.

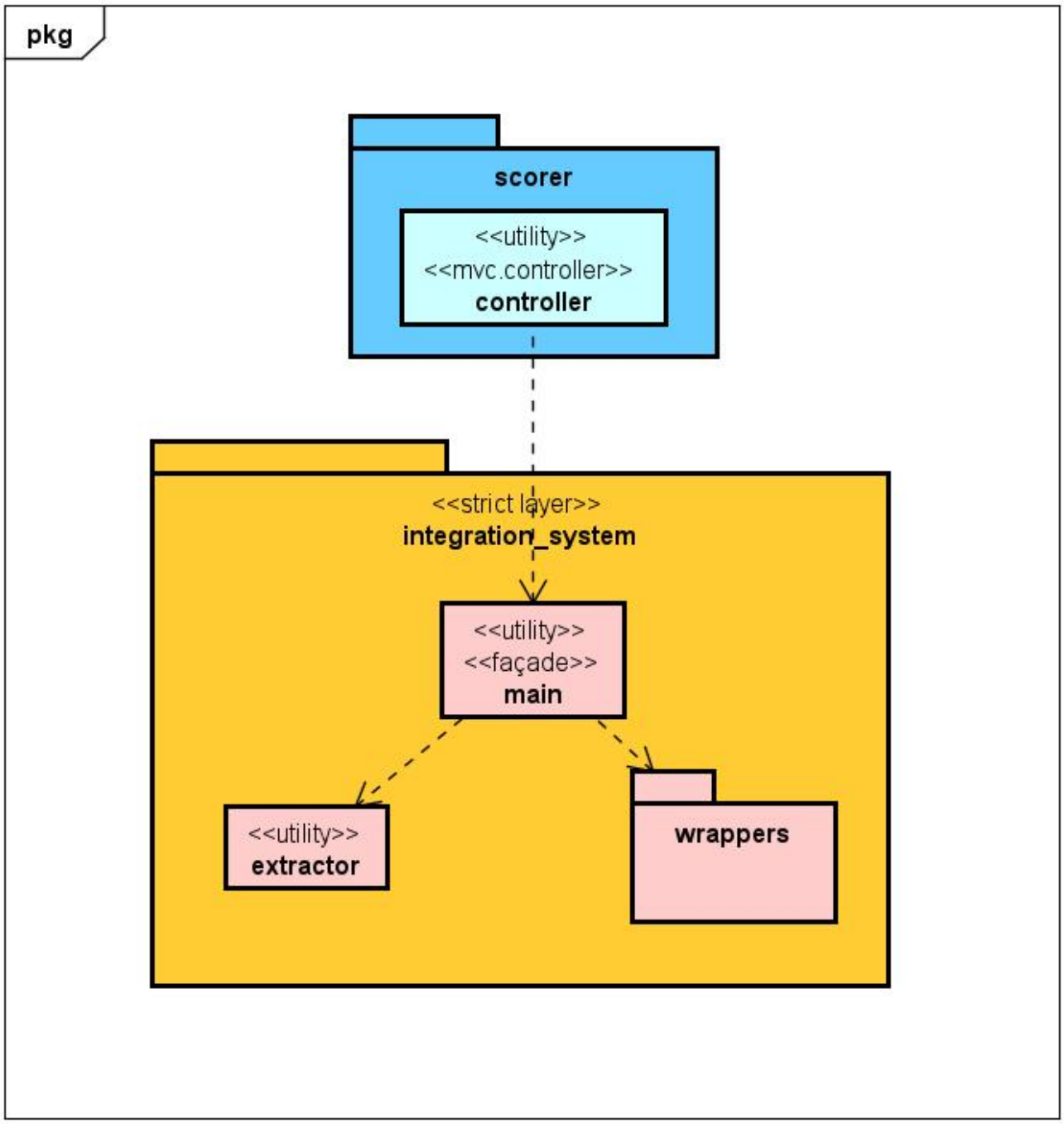


Figura 5.13: Diagrama de interacción de los componentes del Sistema Integrador.

5.8. Realización en diseño de los casos de uso

A continuación de detalla la realización en diseño del caso de uso. Cabe destacar que el primer paso de procesar la request HTTP se omite, pues en todos los casos es completamente análogo al que se ha mostrado en la figura 5.3. En este caso, aunque el exportar los resultados y el subir un archivo fuente sean alternativas dentro del mismo caso de uso, se separan en diagramas distintos al corresponderse a URL separadas para que se entiendan mejor.

5.8.1. Consultar impacto de aplicación

En la figura 5.14 se detalla el caso de uso Consultar métrica de aplicación.

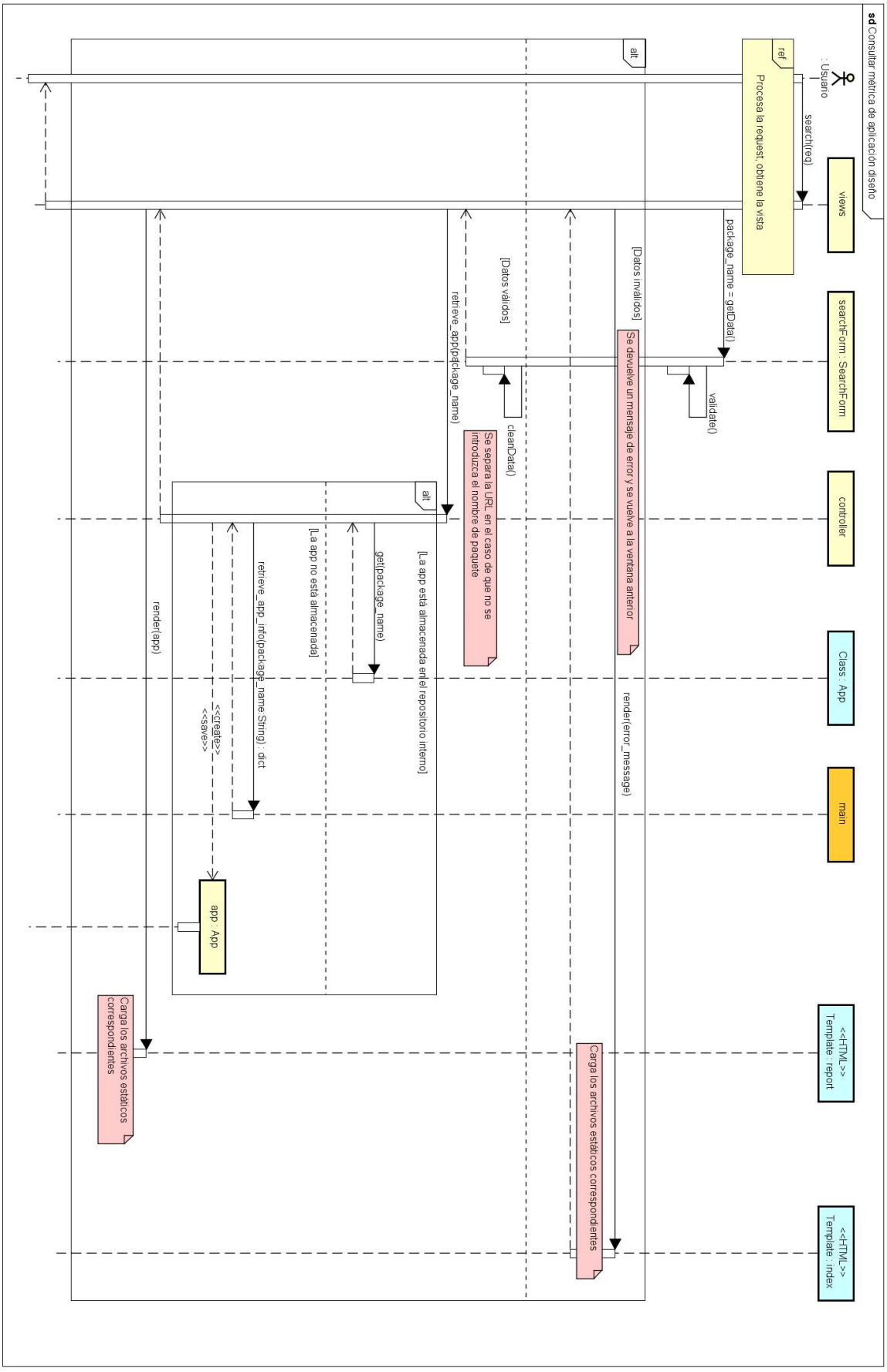


Figura 5.14: Realización en diseño del caso de uso Consultar métrica de aplicación.

Subir archivo de aplicación

En la figura 5.15 se detalla la secuencia correspondiente subir el archivo fuente de una aplicación.

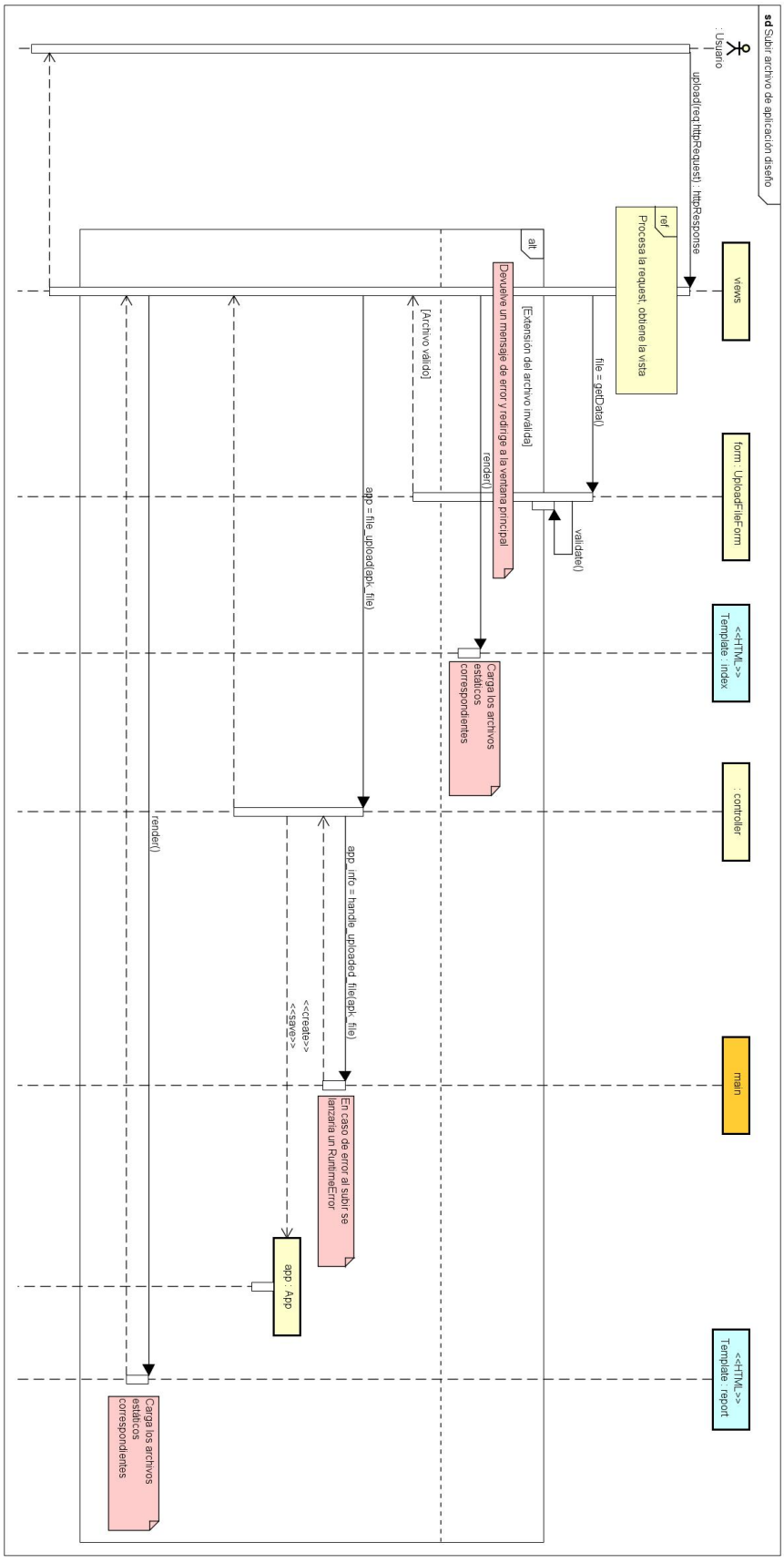


Figura 5.15: Detalle de la secuencia de subir el archivo fuente de una aplicación.

Exportar datos

En la figura 5.16 se detalla la secuencia correspondiente a exportar resultados.

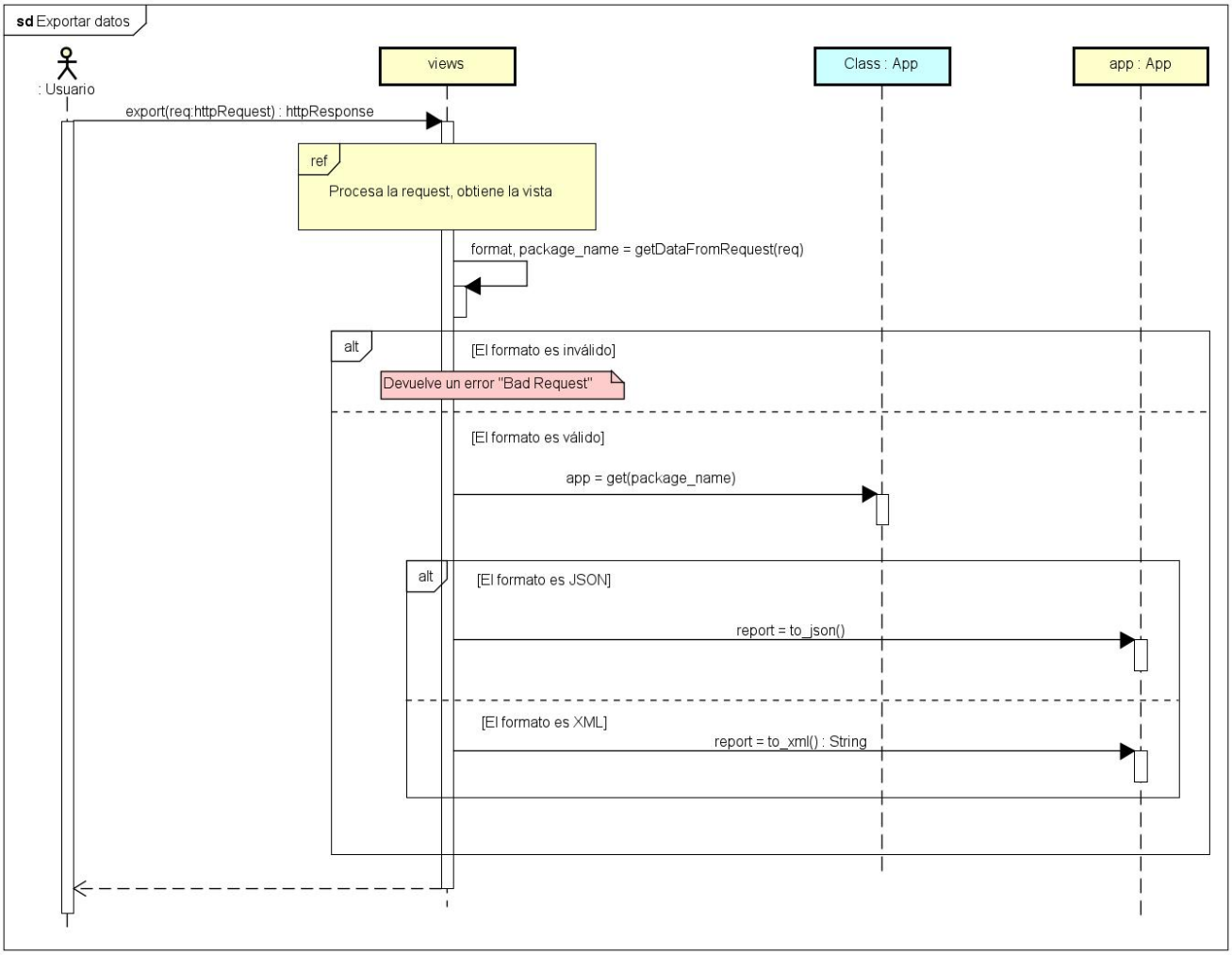


Figura 5.16: Detalle de la secuencia de exportar resultados.

5.9. Flujo de trabajo del Sistema Integrador

En esta sección se mostrarán los diagramas de flujo de proceso y de datos del Sistema Integrador. Como el sistema se va a basar en una serie de módulos que implementan funciones en vez de utilizar clases y objetos, se ha optado por mostrar su funcionamiento a través de diagramas de flujo, en los que cada paso se corresponderá con una función.

5.9.1. Obtención de metadatos de aplicaciones

La primera función del sistema es la obtención de los metadatos necesarios para calcular la métrica y mostrar el informe para aquellas aplicaciones móviles que no se encuentren almacenados de forma local en el servidor. El sistema utilizará el Warehouse externo como fuente de datos preferente. En caso de que la app no esté disponible en el Warehouse, el sistema realizará el cálculo utilizando el resto de fuentes de datos, después se subirán los datos calculados al Warehouse externo. En la figura 5.17 se muestra el flujo de proceso al obtener los metadatos de una aplicación.

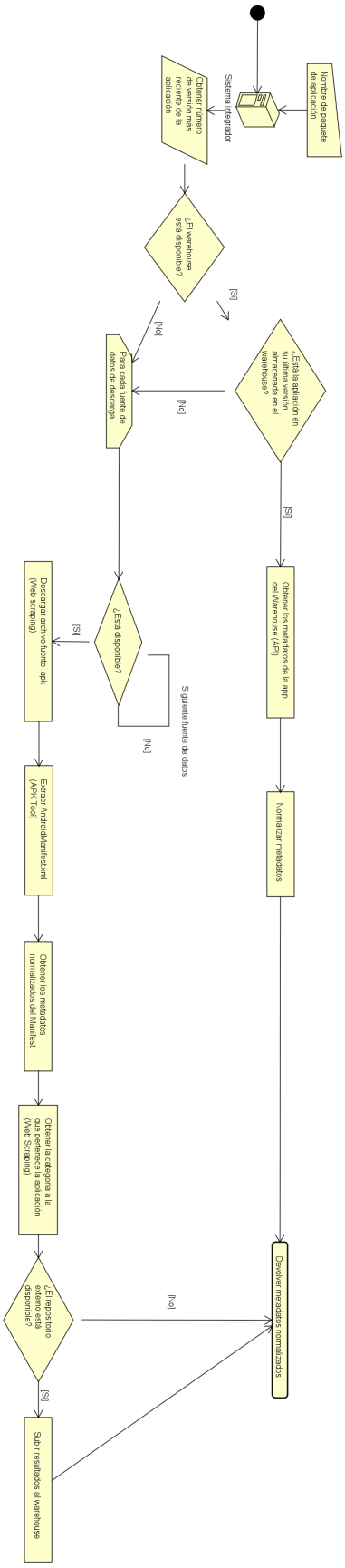


Figura 5.17: Flujo del proceso de obtención de metadatos de una aplicación.

Se puede ver que si la aplicación está disponible en el Warehouse, la información se obtendrá

de ahí. En cambio, si la aplicación no está descargada, se recurrirá al resto de fuentes de datos para descargar y procesar el archivo fuente y obtener de esta manera los metadatos de la aplicación. Una vez se hayan obtenido, se subirán los resultados al Warehouse para que queden almacenados de forma persistente.

En la figura 5.18 se puede ver cómo será el flujo de datos para el proceso descrito. En este caso el Servidor Web pasará el nombre de paquete de la aplicación y el Sistema Integrador devolverá los metadatos de dicha aplicación adaptados al modelo de dominio del Servidor.

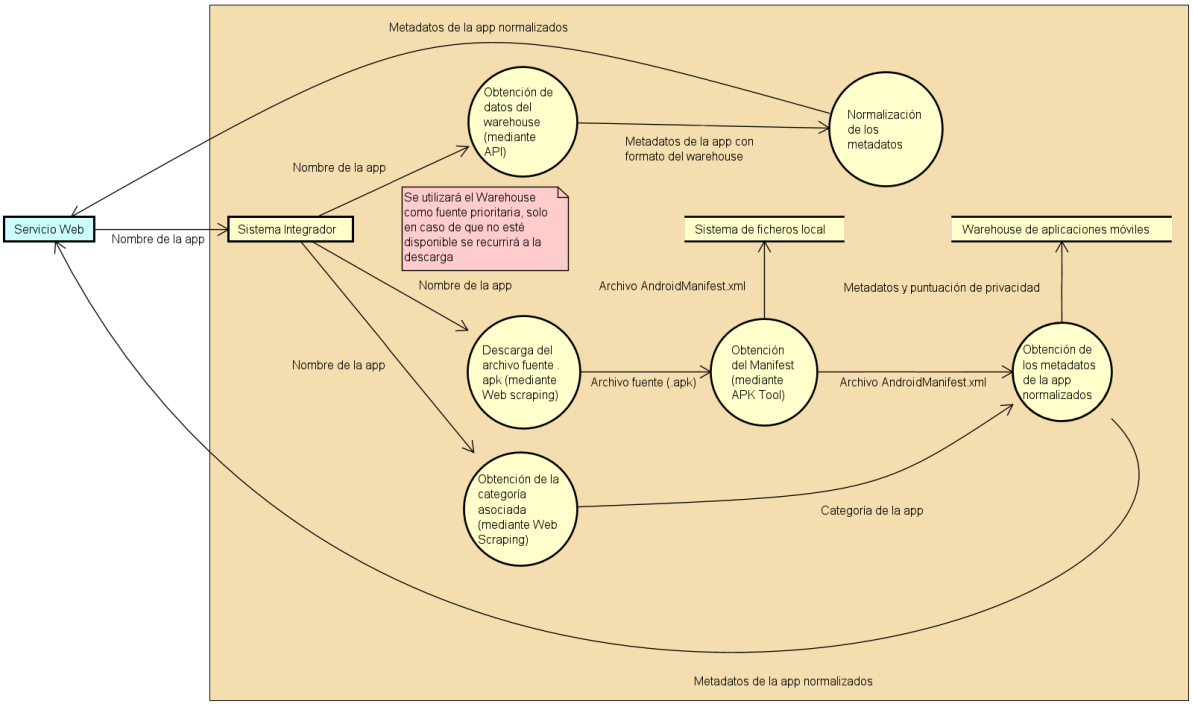


Figura 5.18: Flujo de datos en la obtención de metadatos de una aplicación.

5.9.2. Obtención de metadatos a partir de un archivo fuente

Otra función del Sistema Integrador es la de procesar los archivos fuente subidos por los usuarios, obteniendo los metadatos de la aplicación adaptados al modelo de dominio del Servidor. En la figura 5.19 se muestra el flujo de procedo del Sistema Integrador al procesar un archivo de aplicación subido. Al subir un archivo, por motivos de seguridad, no se almacenará la información de la aplicación en ninguno de los repositorios, sino que se extraerá la información y a continuación se destruirán los archivos fuente.

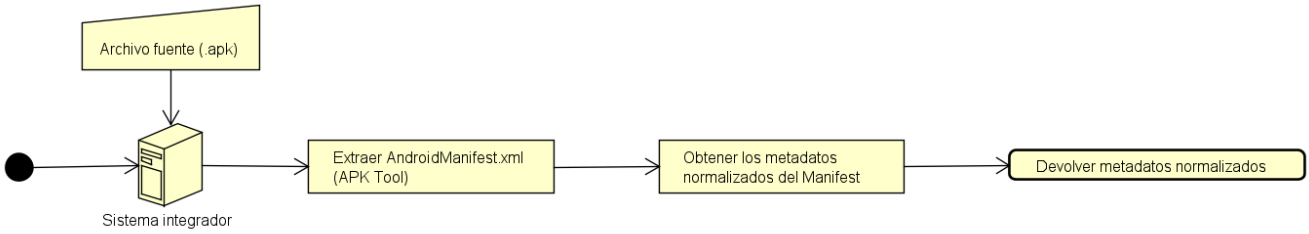


Figura 5.19: Flujo del proceso al subir un archivo fuente.

En la figura 5.20 se muestra el flujo de datos para el proceso descrito. En este caso, el Servidor proporciona el archivo fuente al Sistema Integrador. Éste le devuelve los metadatos adaptados al modelo de dominio.

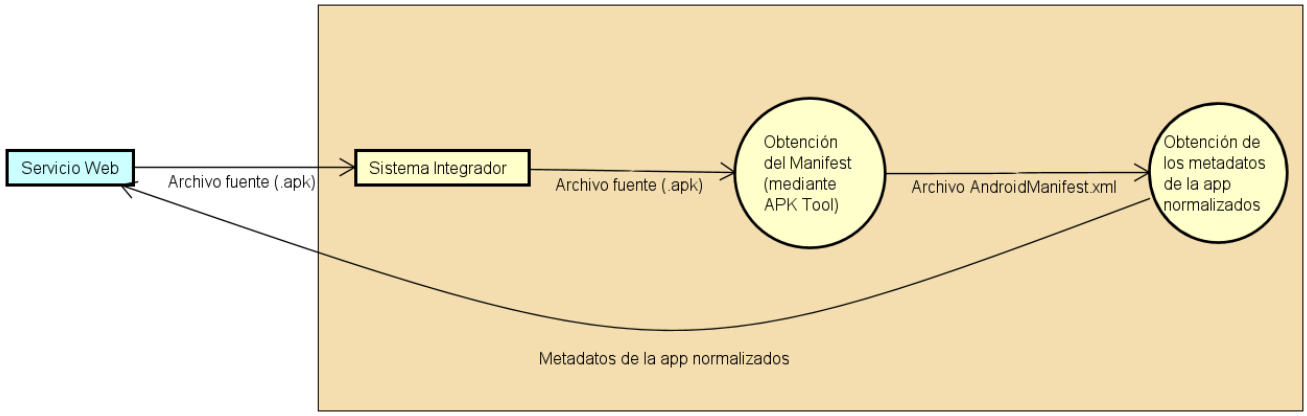


Figura 5.20: Flujo del proceso al subir un archivo fuente.

5.10. Almacenamiento persistente

5.10.1. Almacenamiento de los archivos fuente

Con el objetivo de hacer el servicio más rápido, se almacenan los archivos `AndroidManifest.xml` de forma persistente (para no tener que descargar las aplicaciones de nuevo, ya que éste es el cuello de botella que más ralentiza el servicio), siendo innecesario el almacenamiento persistente de los archivos `.apk` debido a que todos los datos pueden ser recogidos directamente del Manifest. Para ello, se define un directorio base en una variable de entorno `APK_FALCON_ROUTE_TO_APK_STORAGE_DIR`, ese directorio base contiene subdirectorios con los nombres de paquete de las aplicaciones (ver figura 5.21) donde se almacenarán los archivos.

```
javier@LAPTOP-7U64GK3H:~/apk$ ls
app.grotinou.sushi              com.facebook.orca          com.spotify.music          com.zzkko
com.ai.chat.bot.aichat          com.kingsfantasy           com.squareup.cash          es.davidpob99.ContadorMus
com.amazon.avod.thirdpartyclient com.netflix.mediaclient   com.twitter.android        me.pou.app
com.disney.disneyplus           com.ryanair.cheapflights   com.us.zoom.videomeetings  org.telegram.messenger
com.facebook.katana             com.snapchat.android       com.whatsapp               uploaded
javier@LAPTOP-7U64GK3H:~/apk$ ls com.whatsapp/
AndroidManifest.xml
```

Figura 5.21: Ejemplo de la estructura del sistema de ficheros.

De cara a la descarga de nuevas aplicaciones, se creará un directorio temporal con el nombre `<NOMBRE DE PAQUETE>.download`, de forma que se almacene ahí el archivo `.apk` descargado. A continuación este archivo se descomprimirá y el Manifest se moverá al directorio "definitivo", eliminándose el resto de archivos resultantes de la descompresión, así como el directorio temporal.

5.10.2. Almacenamiento de los metadatos

Los metadatos de las aplicaciones móviles serán almacenados en dos sitios: en el Warehouse de aplicaciones móviles y en la base de datos local del Servidor, en un formato reducido de forma que esta base de datos local actúe como caché de acceso rápido para hacer más eficientes las consultas. En la figura 5.22 se muestra el modelo de datos del repositorio de datos local, éste es una adaptación directa del modelo de dominio, tal y como marca el patrón de diseño *Active Record* explicado en 5.2.3. Como marca el requisito [RNF11], este almacén estará securizado ante intentos de intrusión y ataques contra la integridad de los datos.

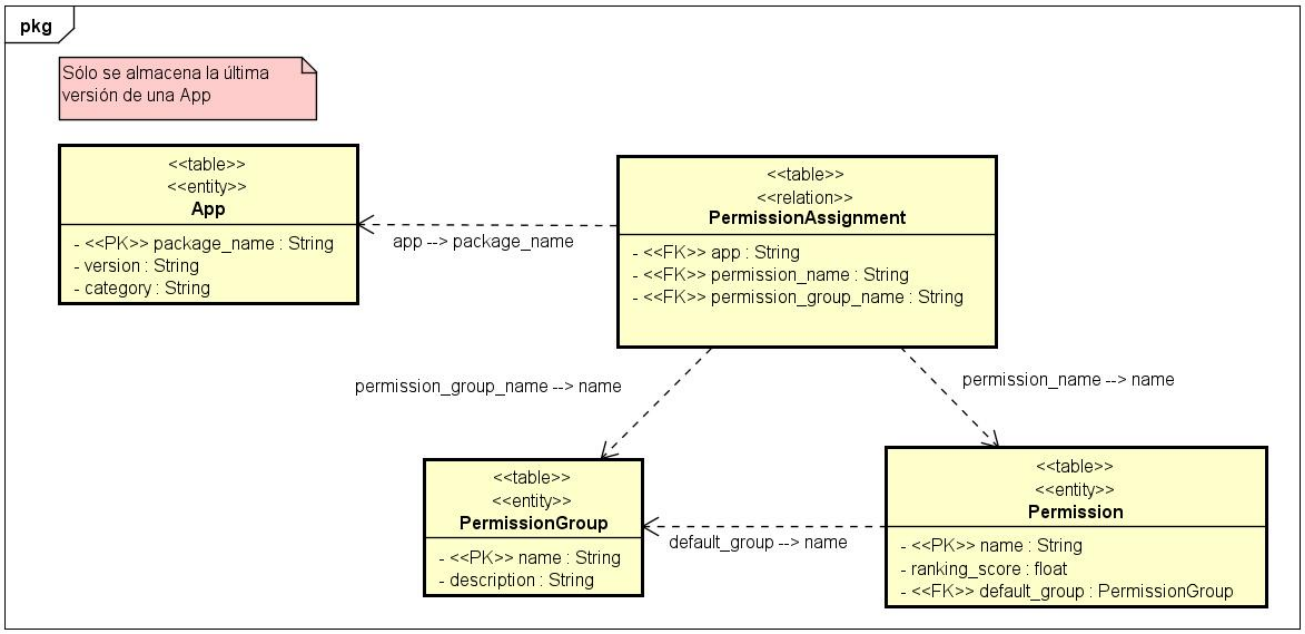


Figura 5.22: Modelo de datos de la base de datos local.

En la figura 5.23 se muestra el modelo de datos del Warehouse de aplicaciones móviles, proporcionado por Alejandro Pérez de la Fuente. Obsérvese que el modelo de datos que maneja el Servidor Web puede ser visto como un "subconjunto" del modelo de datos del Warehouse, de forma que dentro del Servidor Web solamente se encuentran únicamente los datos necesarios para calcular la métrica y mostrar los resultados. El Warehouse incluye información más detallada y completa de las aplicaciones.

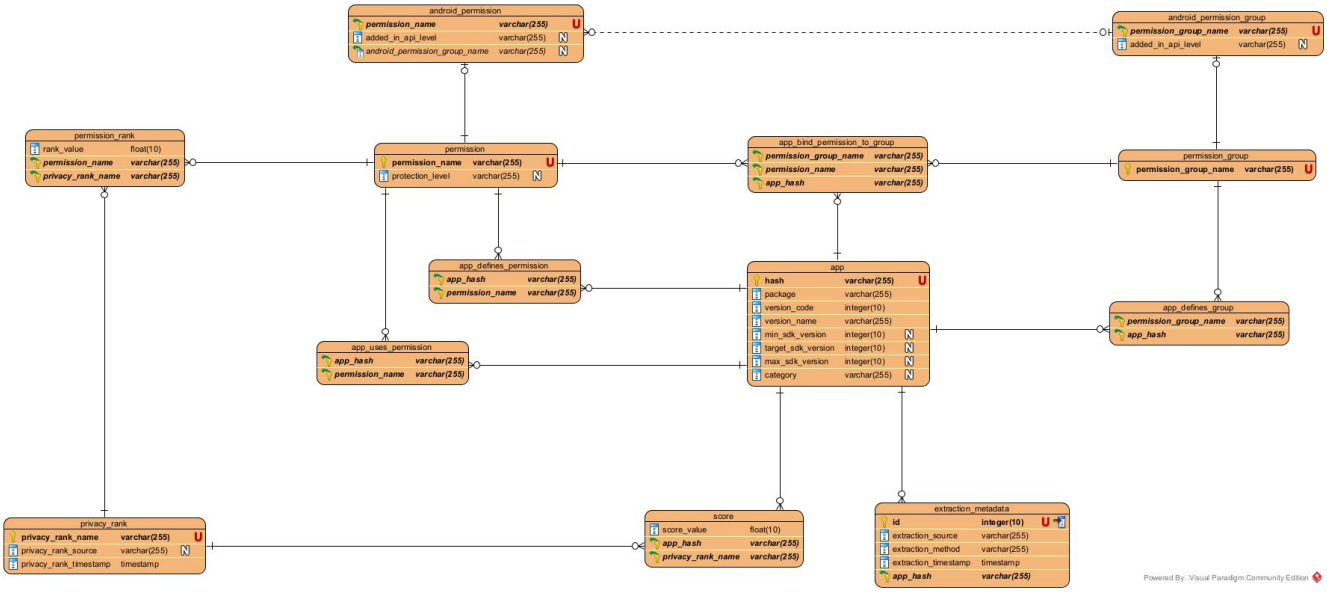


Figura 5.23: Modelo de datos del Warehouse de aplicaciones móviles, proporcionado por Alejandro Pérez de la Fuente.

5.11. *Mock-ups* de la interfaz de usuario

A continuación se muestran los bocetos de la interfaz de usuario, dividida en ventanas.

5.11.1. Ventana principal

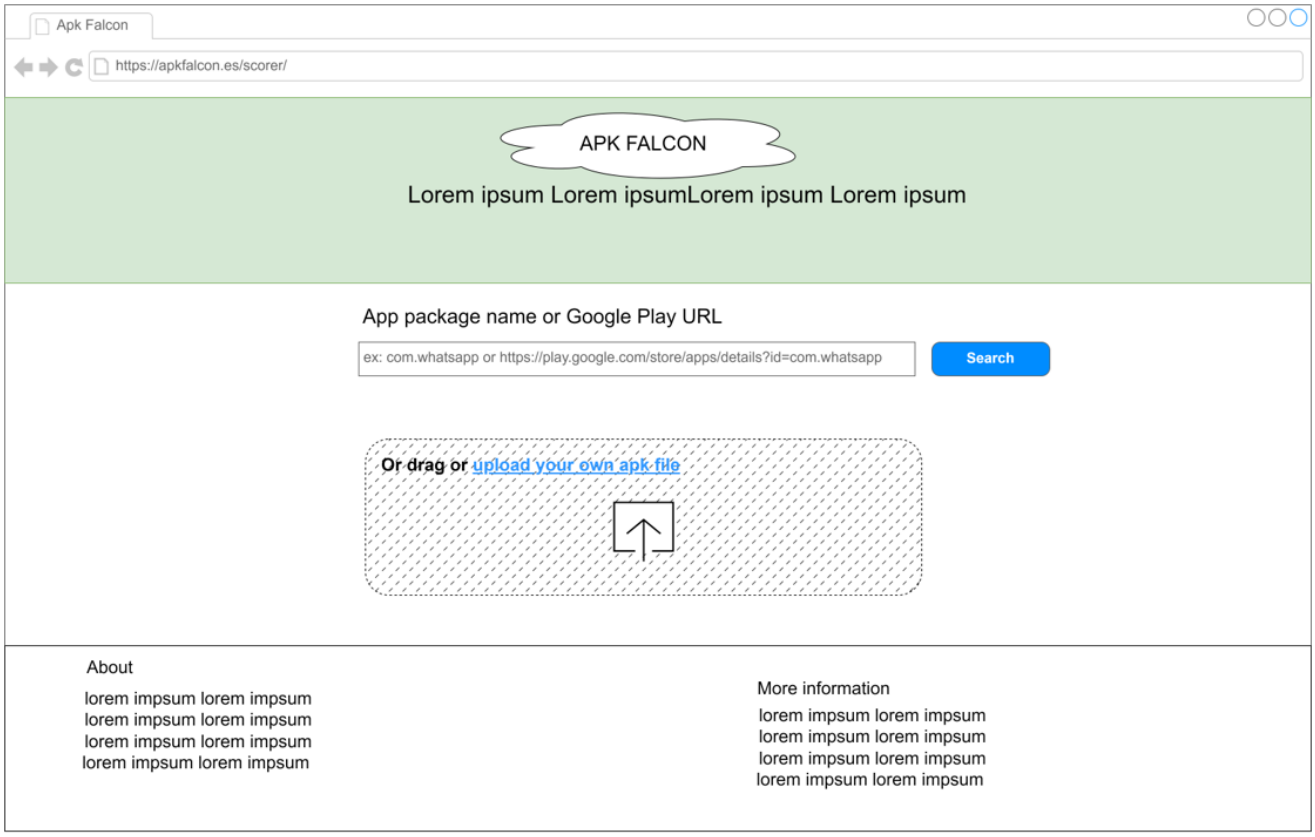


Figura 5.24: Boceto de la ventana principal.

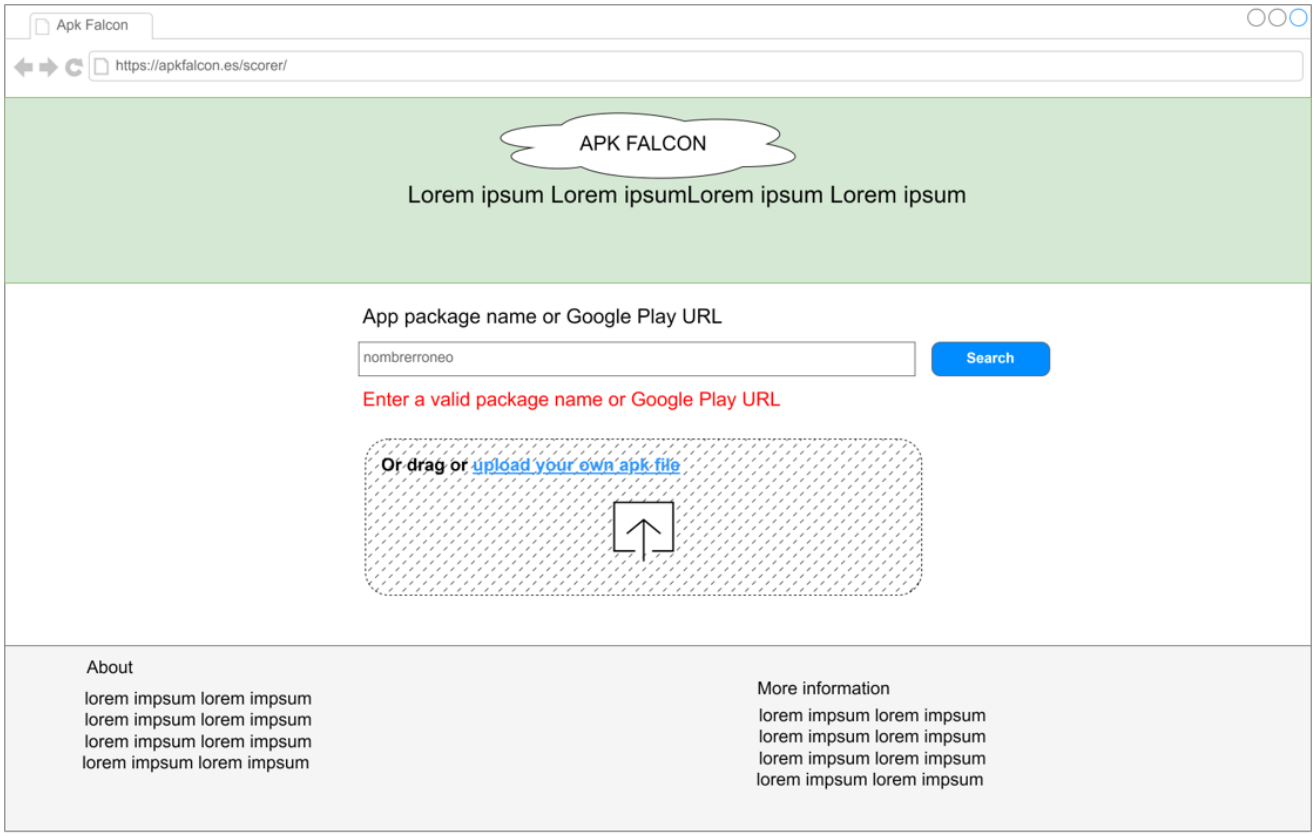


Figura 5.25: Boceto de la ventana principal mostrando un error de nombre de paquete incorrecto.

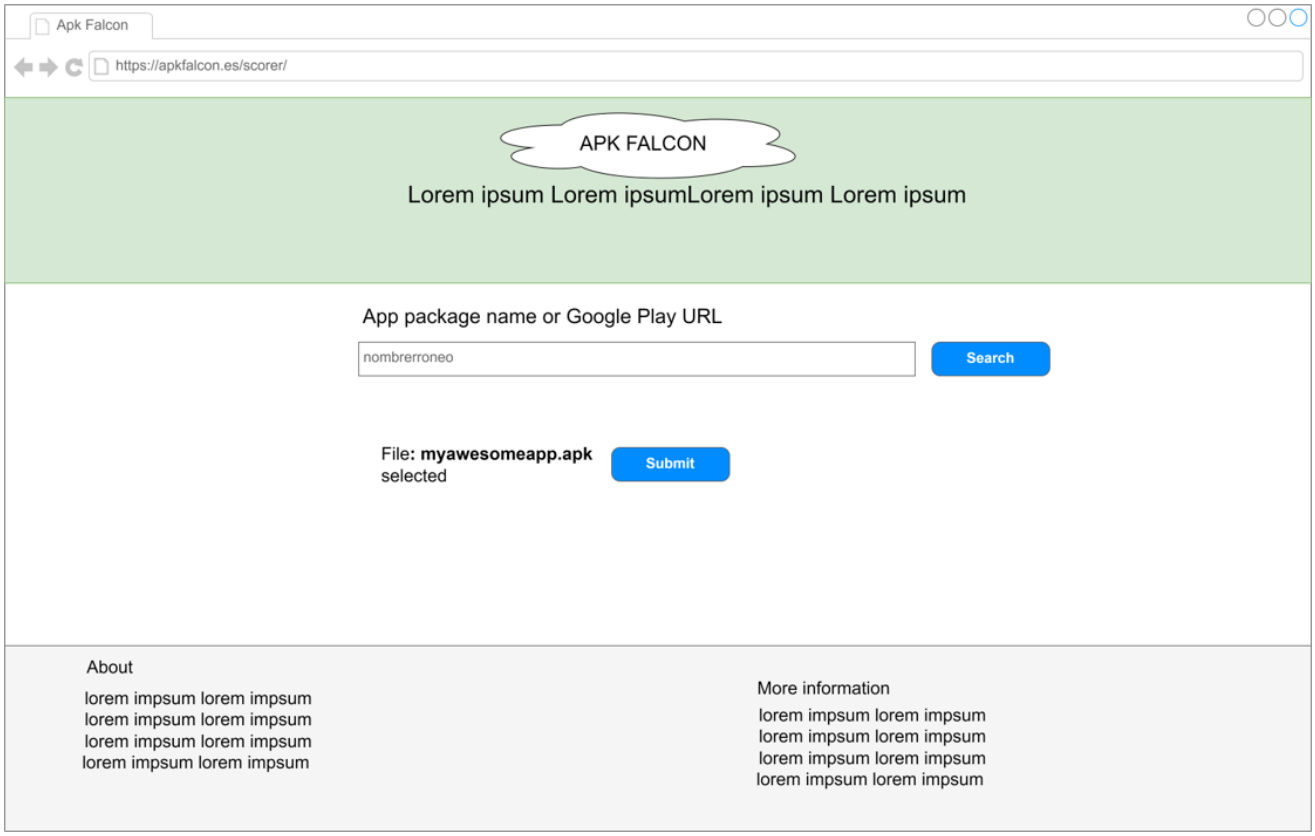


Figura 5.26: Boceto de la ventana principal al seleccionar un archivo para subir.

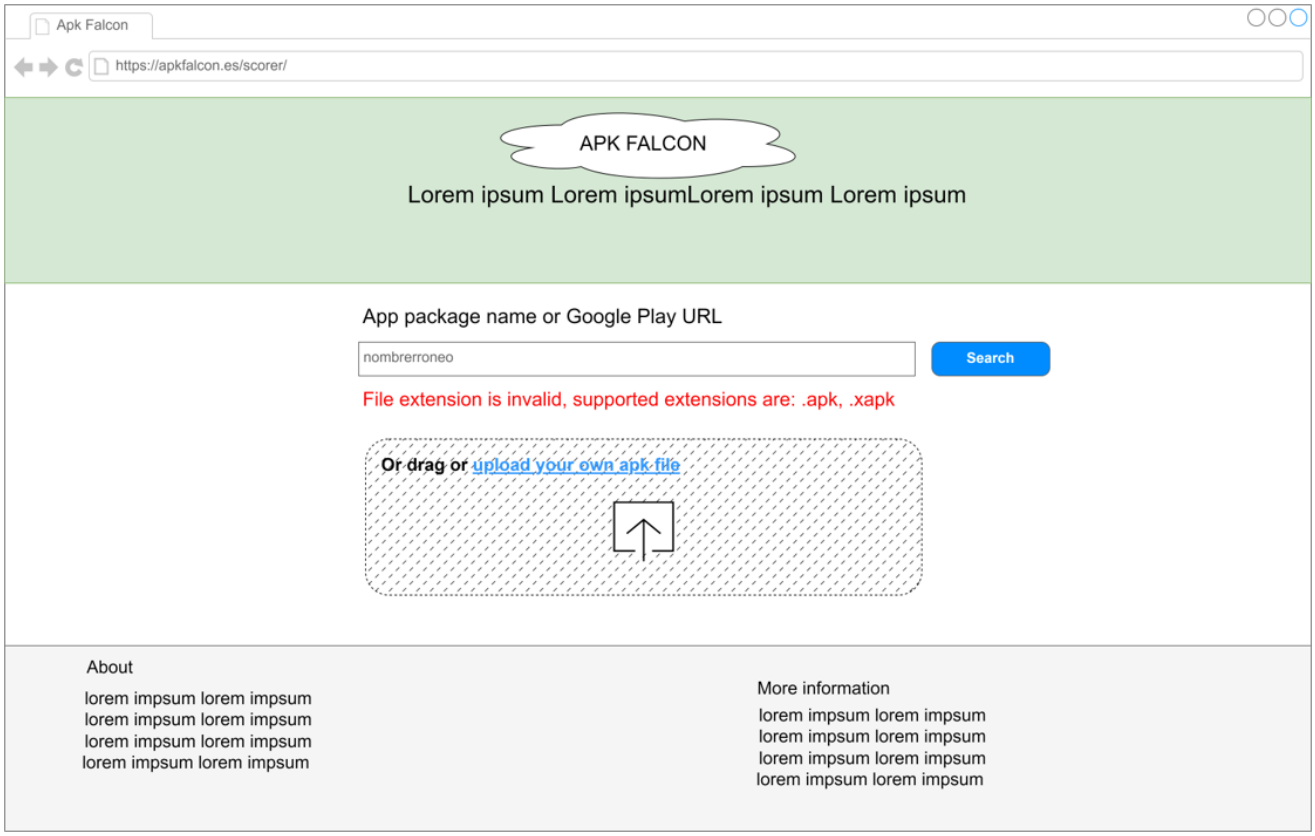


Figura 5.27: Boceto de la ventana principal mostrando un error de formato de archivo a subir.

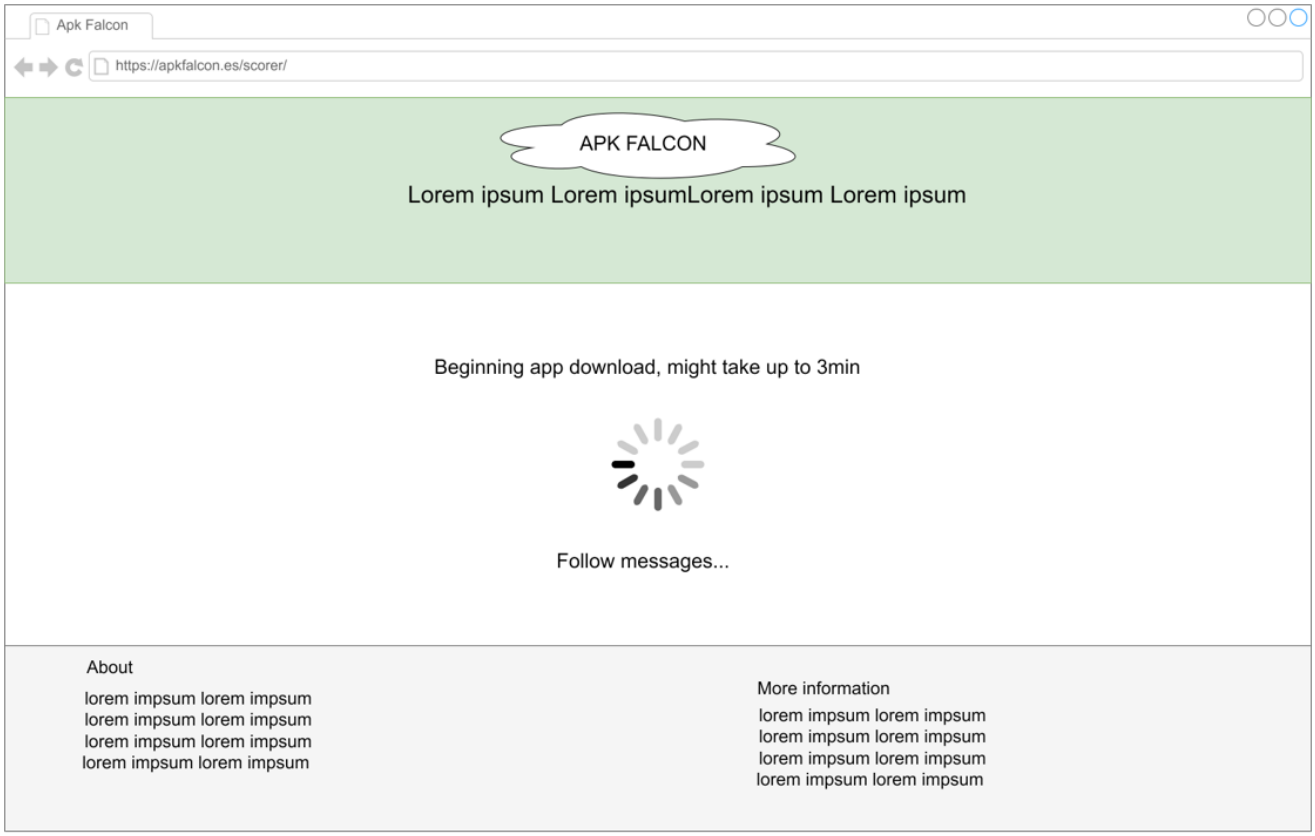


Figura 5.28: Boceto de la ventana principal mostrando el mensaje de carga al buscar.

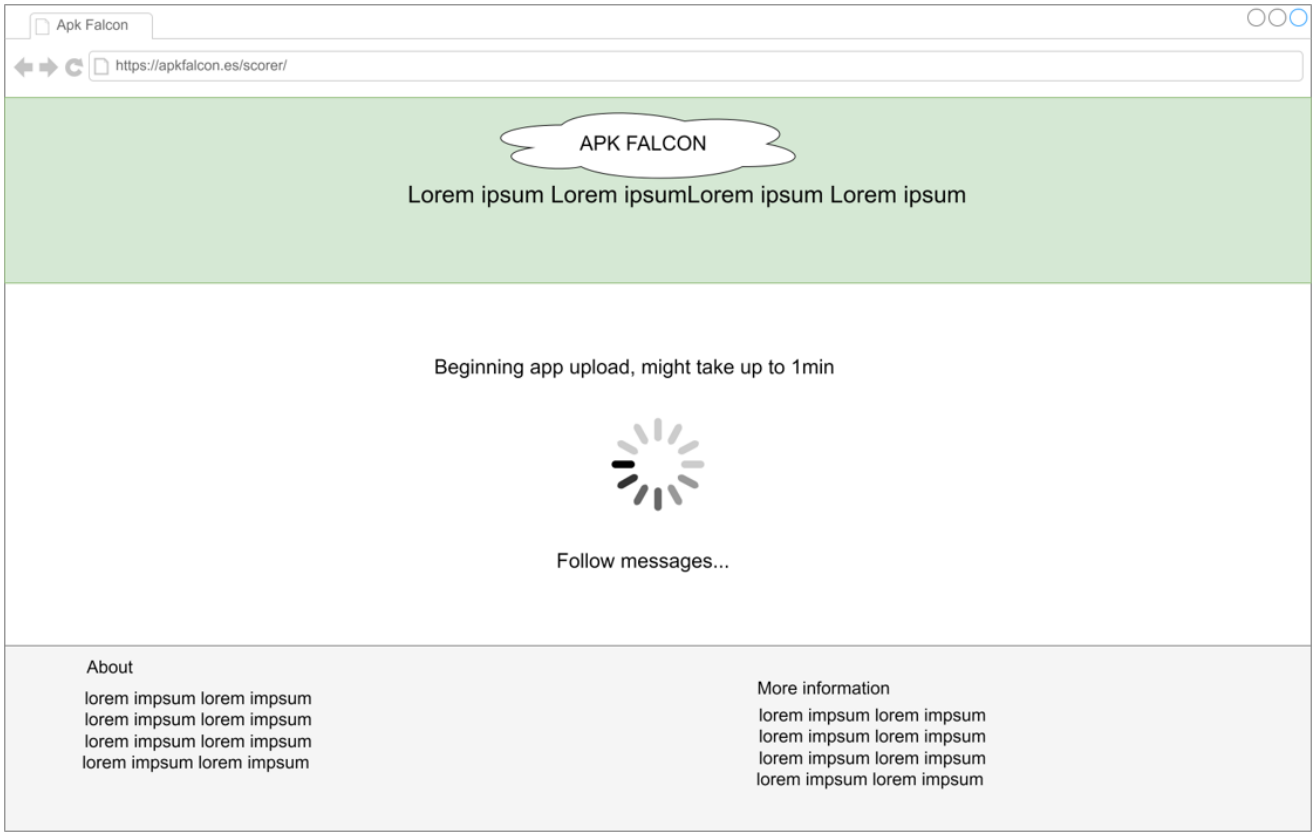


Figura 5.29: Boceto de la ventana principal mostrando el mensaje de carga al subir una aplicación.

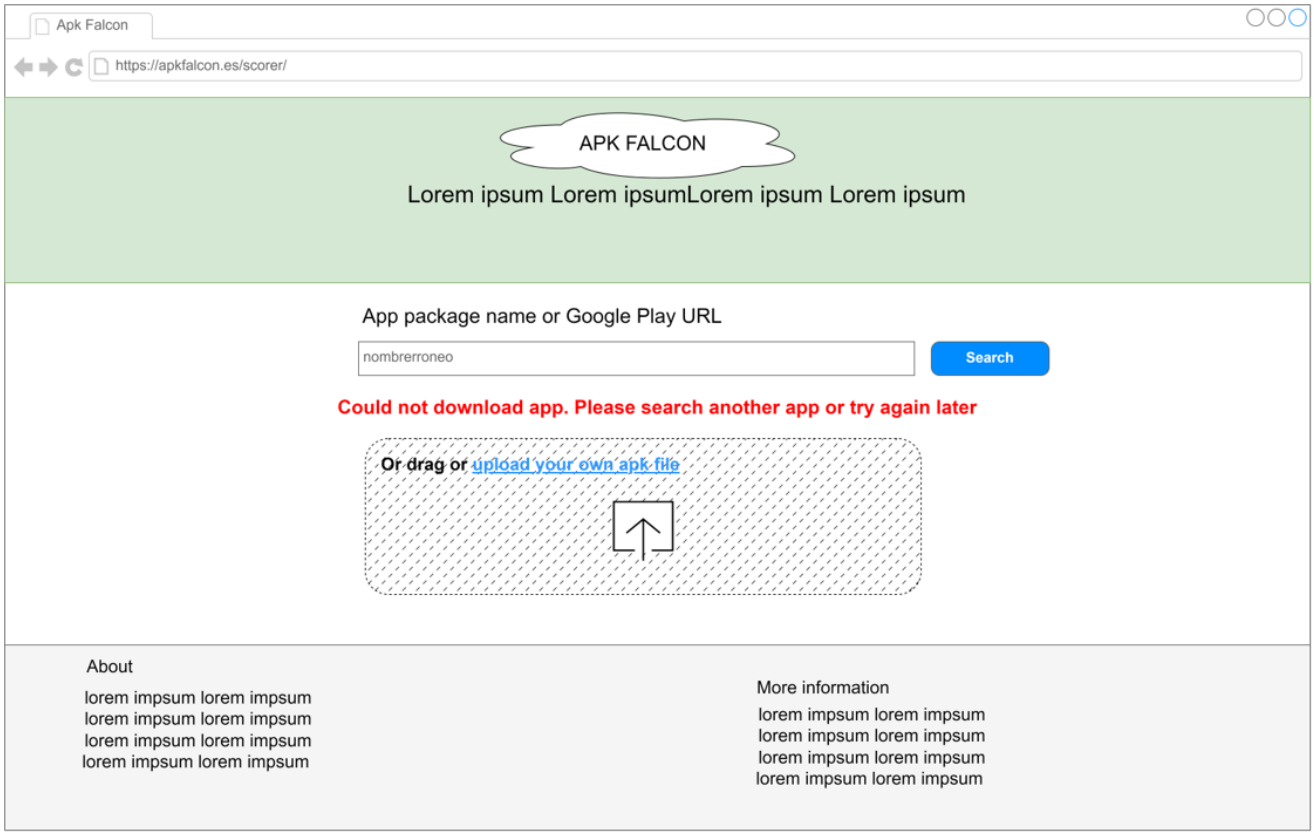


Figura 5.30: Boceto de la ventana principal mostrando un error al descargar la aplicación.

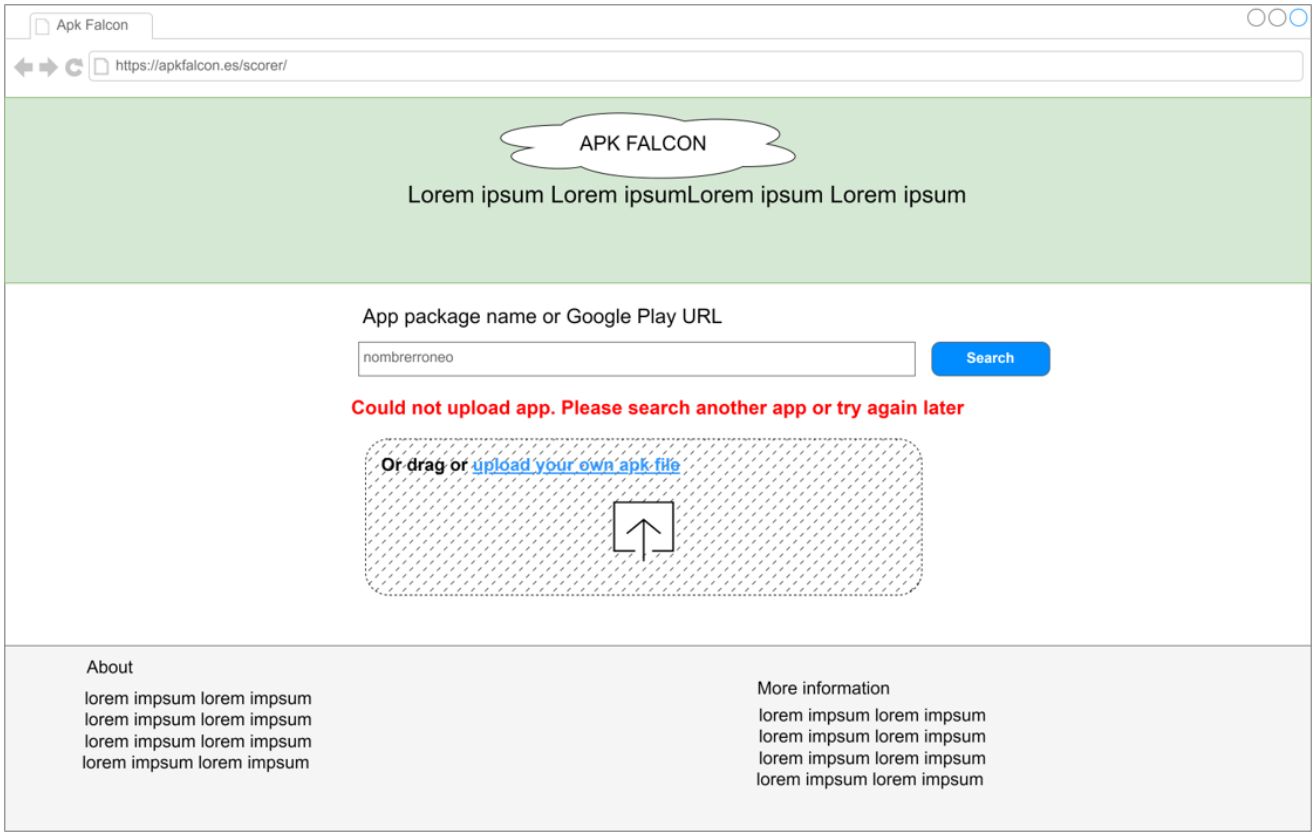


Figura 5.31: Boceto de la ventana principal mostrando un error al subir una aplicación.

5.11.2. Ventana de informe de privacidad

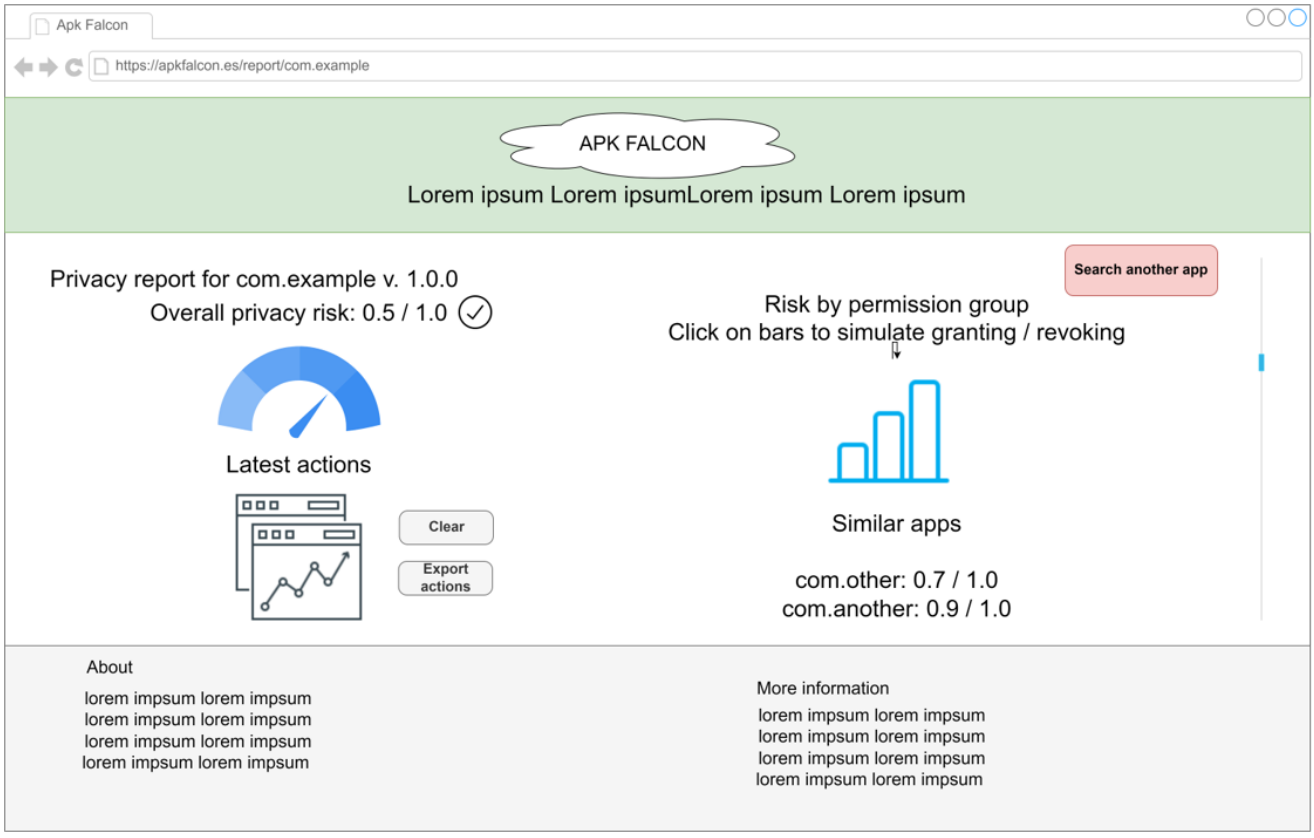


Figura 5.32: Boceto de la ventana de informe de privacidad mostrando los gráficos resumen.

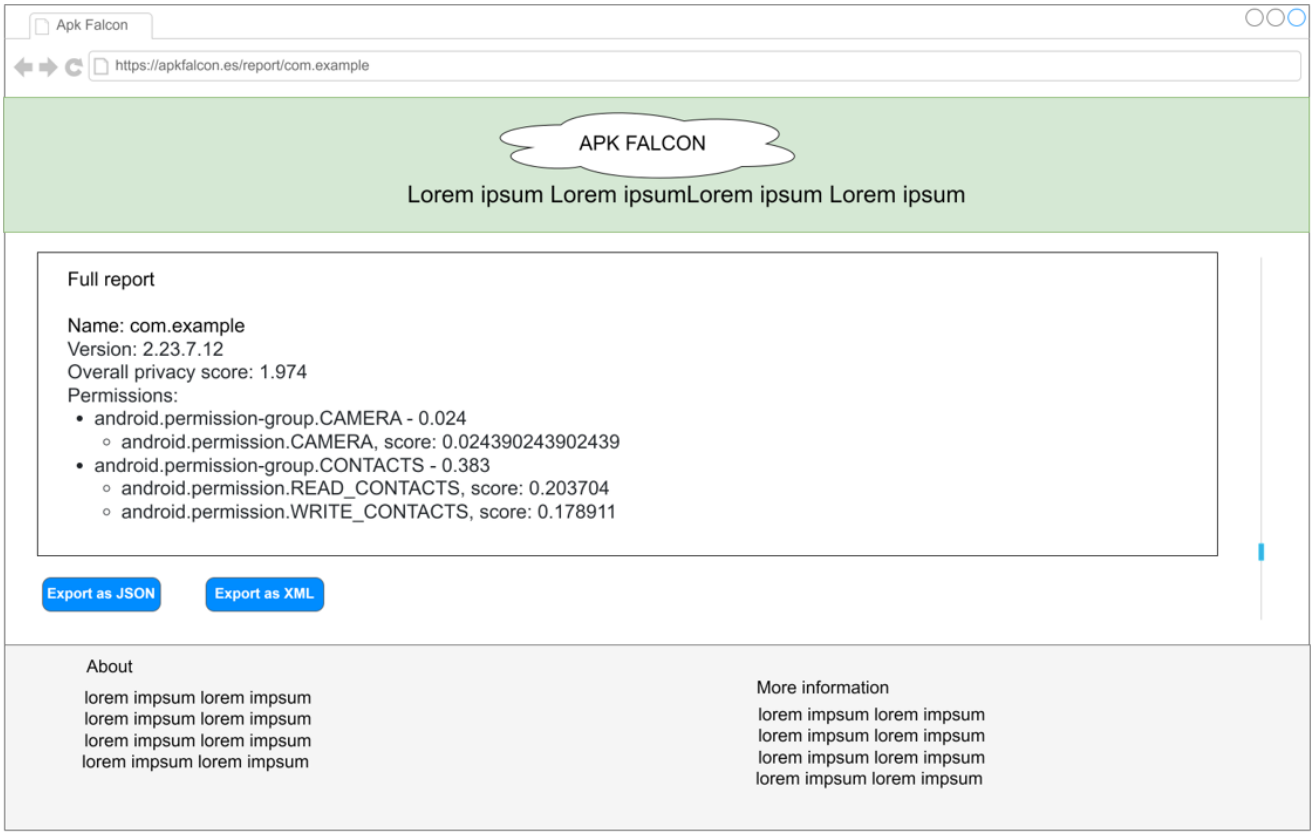


Figura 5.33: Boceto de la ventana de informe de privacidad mostrando el informe detallado.

5.11.3. Ventana de información

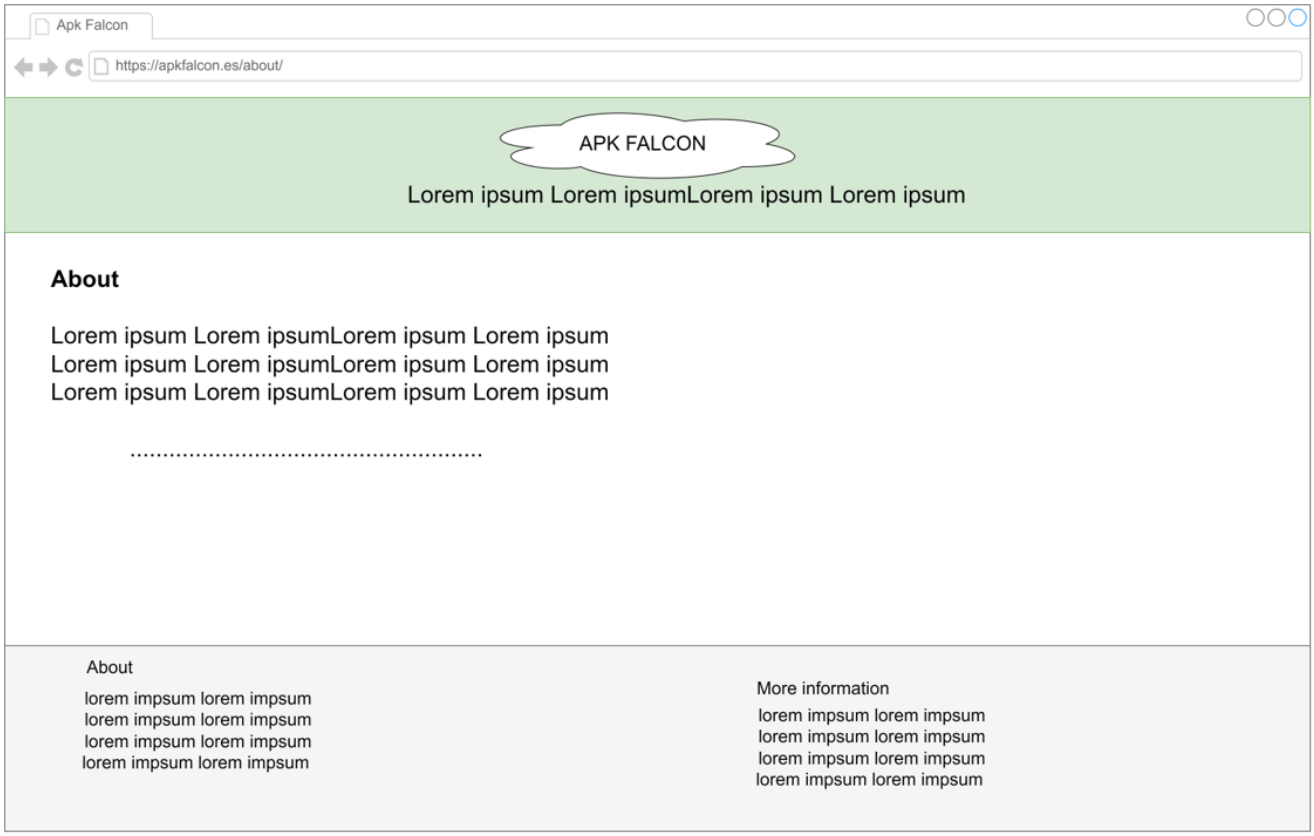


Figura 5.34: Boceto de la ventana de información.

5.12. Flujo de la interfaz y nombre de las URL

En la figura 5.35 se muestra el flujo del sistema desde el punto de vista del usuario y los nombres de las URL asociados a cada ventana.

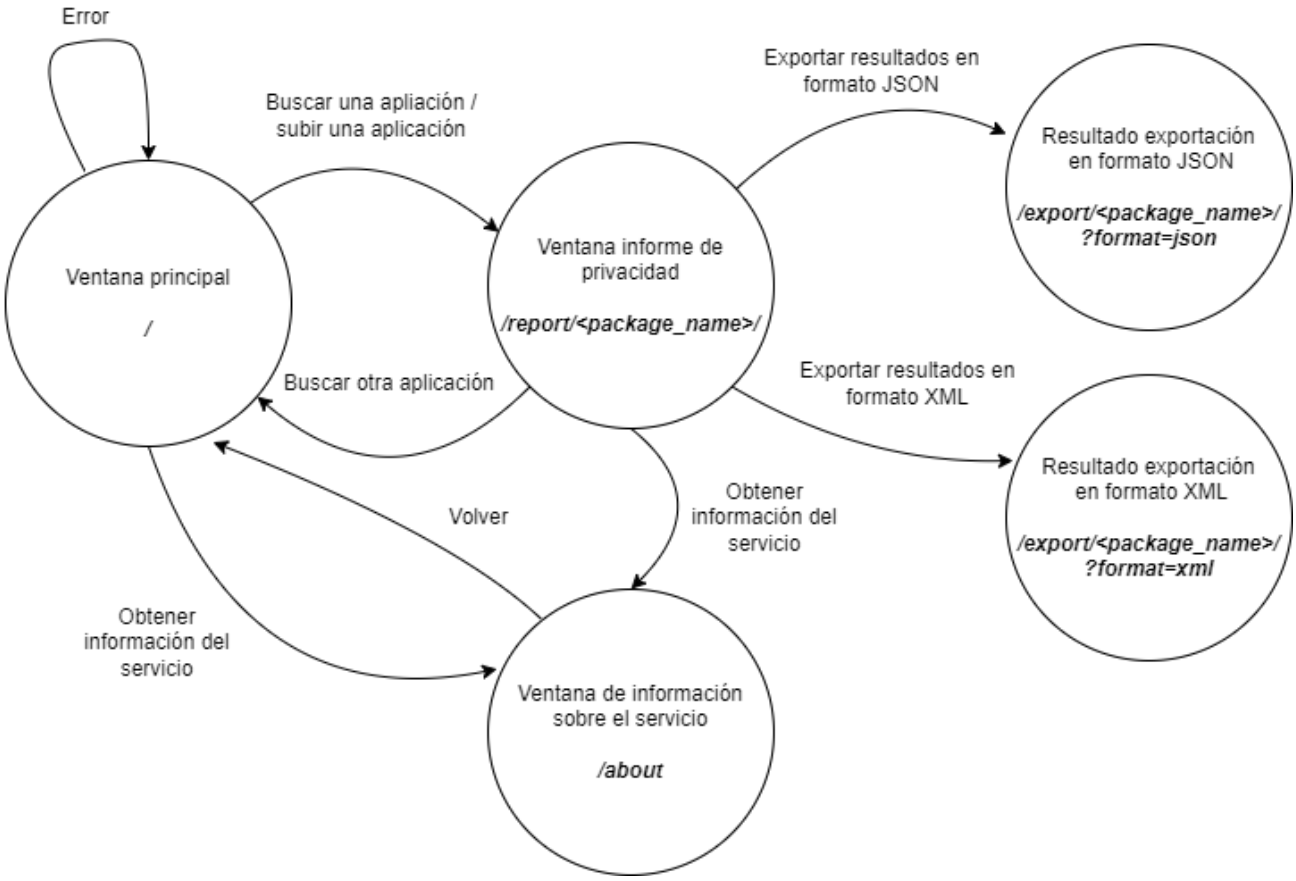


Figura 5.35: Flujo de ventanas en la interfaz de usuario.

5.13. Visualizaciones de datos e interactividad

Como se puede ver en los *mock-ups* de la interfaz de usuario, el informe de privacidad incluirá una primera sección gráfica e interactiva destinada a ser comprendida por el público más general. En esta sección se detalla su diseño, en la figura 5.36 se puede ver un boceto de la visualización.

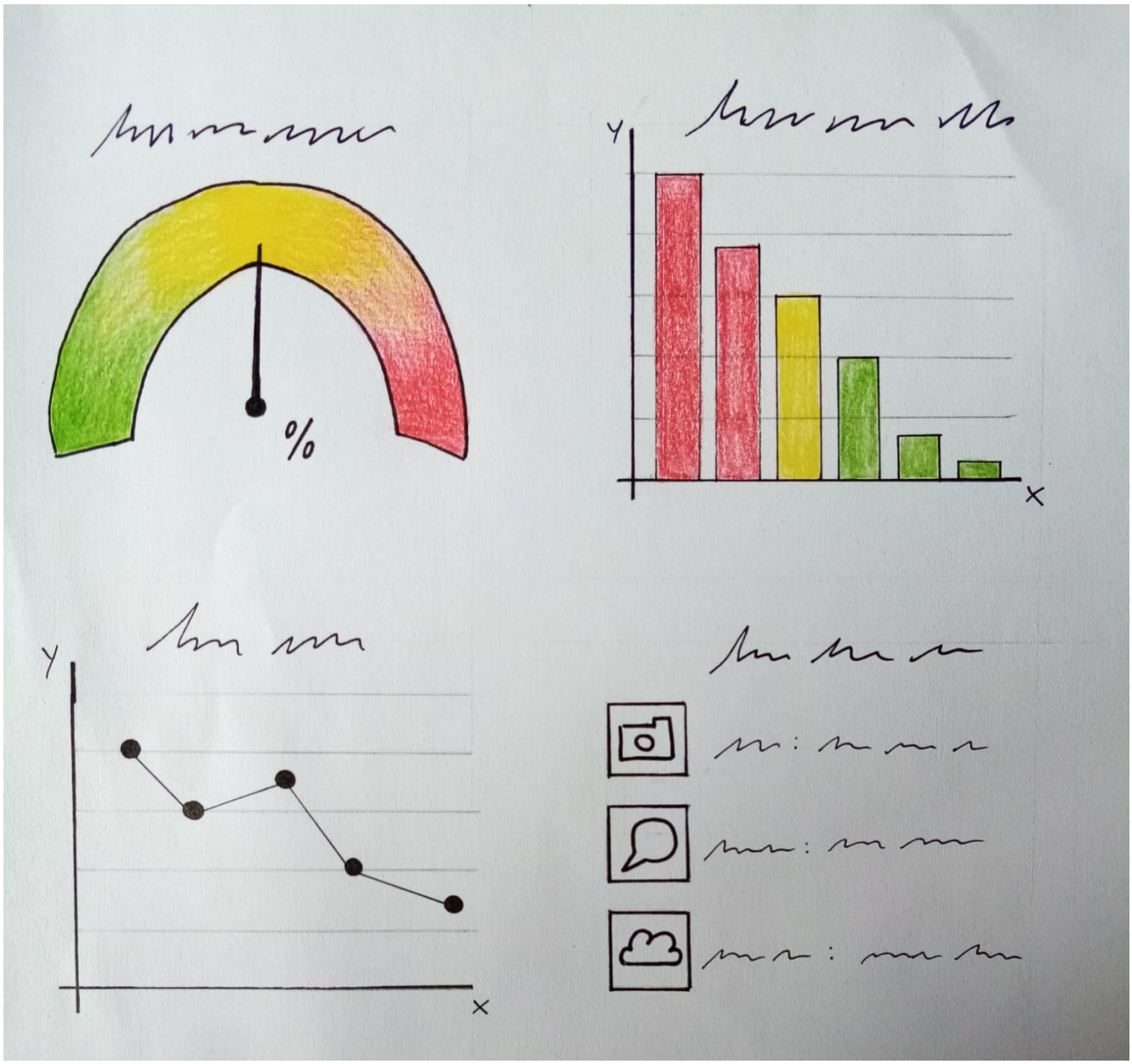


Figura 5.36: Boceto de la visualización.

La visualización se compone de tres gráficos fundamentales:

1. Gráfico de velocímetro: la puntuación global de privacidad se mostrará mediante un gráfico de velocímetro, ya que ésta es una medida normalizada y por tanto tiene sentido representarla como "parte de un total". Al lado de la puntuación de privacidad se mostrará un icono indicando el grado de intrusividad de la aplicación con respecto a la privacidad. Se utilizarán iconos de "tick", "warning" y "danger", bien conocidos por la mayoría de usuarios, para categorizar a las aplicaciones en poco intrusivas, medianamente intrusivas y muy intrusivas, respectivamente. Del mismo modo, se utilizará un degradado de verde a rojo dentro del velocímetro, para reforzar la idea anterior. Este gráfico irá variando la posición de la aguja y el porcentaje en función de cómo se vayan revocando/concediendo los permisos.

2. Gráfico de barras: se mostrarán los grupos de permisos usados por la aplicación mediante un gráfico de barras mostrando la puntuación de privacidad asociada al grupo (se calcula como

la suma de las puntuaciones de cada permiso). Al pasar el ratón por encima, se mostrará un recuadro con la información detallada de la descripción del grupo y la puntuación de privacidad asociada. La concesión y/o revocación de permisos se realizará pulsando en las barras del gráfico (que se mostrará jugando con la transparencia en el color de las barras), lo que repercutirá en la puntuación mostrada en el gráfico de velocímetro. Se utilizará una clave de tres colores: rojo, amarillo y verde para mostrar el grado de intrusividad con la privacidad de un grupo de permisos.

3. Gráfico de líneas: se utilizará un gráfico de líneas para mostrar los efectos en la puntuación de privacidad de las acciones realizadas (concesión/revocación de permisos) en la puntuación global y de este modo poder comparar distintos escenarios sin tener que memorizar resultados previos. Este gráfico se irá poblando según se vaya pulsando en las barras del gráfico anterior. Del mismo modo que en el anterior, se mostrará un recuadro con la información detalla de la acción que se acaba de realizar al pasar el ratón por encima de un punto. Se incluirá una opción para restablecer este gráfico y otra para exportar las acciones que se han realizado.

Aunque estrictamente hablando no forme parte de la visualización, también se mostrará un conjunto de aplicaciones similares a la aplicación buscada (en base a la categoría). Para ello se mostrará el logo de estas aplicaciones junto con su puntuación de privacidad en verde si es menor y en rojo si es mayor. De esta forma el usuario puede hacer una comparación rápida de varias aplicaciones.

5.14. Exportación de los resultados

Por un lado, se permitirá la exportación del informe de privacidad en dos formatos: `XML` y `JSON`. Por otro, se permitirá exportar los resultados de la simulación de concesión/revocación de permisos en formato `JSON`.

5.14.1. Formato de exportación `JSON`

A continuación se muestra el formato de exportación del informe de privacidad en `JSON` a través de un ejemplo.

```
{
    "app" : {
        "package_name" : "com.example",
        "version" : "1.0.0",
        "category" : "EXAMPLES",
        "privacy_score" : "1.62",
        "max_privacy_score" : "2.0",
        "permissions_groups" : [
            {
                "name": "android.permission.example",
                "score": 0.5,
                "group": "android.permission-group.example"
            }
        ]
    }
}
```

5.14.2. Formato de exportación XML

A continuación se muestra el formato de exportación del informe de privacidad en XML a través de un ejemplo.

```
<app>
    <package_name>com.example</package_name>
    <version>1.0.0</version>
    <category>EXAMPLE</category>
    <privacy_score>1.62</privacy_score>
    <max_privacy_score>2.0</max_privacy_score>
    <permissions_groups>
        <name>android.permission.example</name>
        <score>0.5</score>
        <group>android.permission-group.example</group>
    </permissions_groups>
    <permissions_groups>
        <name>android.permission.example2</name>
        <score>0.5</score>
        <group>android.permission-group.example2</group>
    </permissions_groups>
</app>
```

5.15. Despliegue del sistema

En la figura 5.37 se muestra el diagrama de despliegue del sistema. El cliente accederá a través de un navegador web al servicio web, que se desplegará en una máquina virtual con el Sistema Integrador y la base de datos local incorporadas. El Warehouse de aplicaciones móviles se desplegará en otra máquina virtual y se accederá mediante peticiones HTTP a través de una API.

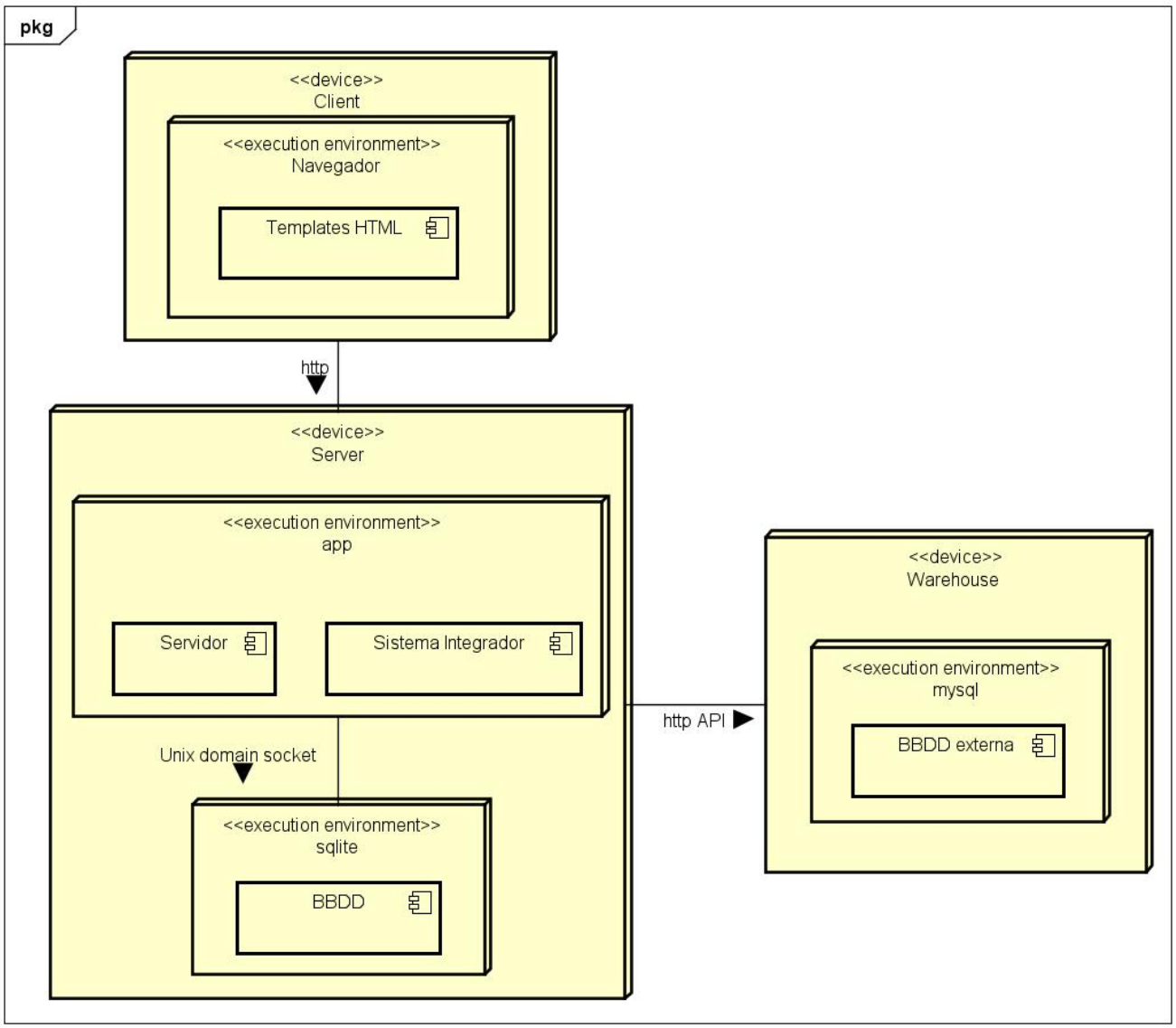


Figura 5.37: Diagrama de despliegue del sistema

5.16. Comunicación con las fuentes de datos

Como se ha explicado, el servicio tomará información de 3 fuentes de datos: un Warehouse de aplicaciones móviles (desarrollado de forma paralela en otro TFG) y dos fuentes de descarga de aplicaciones: *APKPure* y *Evozi APK Downloader*, a continuación se describe el diseño de la comunicación con estas fuentes.

5.16.1. Warehouse de aplicaciones móviles

La comunicación con el Warehouse se realizará en dos contextos: al buscar la información asociada a una aplicación de la que no se dispone en el almacenamiento local y al descargar una nueva aplicación que no existe en el Warehouse. En ambos casos la comuniación se realizará por mensajes `HTTP` a través de una `API` que proporcionará la fuente de datos. De este modo, el Warehouse utilizará el patrón Fachada para proporcionar una interfaz de acceso única, abstrayendo al Sistema Integrador del detalle de las operaciones que realiza tras las peticiones.

Consulta de aplicaciones

Cuando no se disponga de la información de una aplicación solicitada, se usará esta fuente de datos como fuente prioritaria para obtener la información que nos permita construir el correspondiente objeto del modelo. Para ello, se realizará una petición `GET` a la fuente de datos con el nombre de paquete de la aplicación como parámetro y ésta devolverá un `JSON` con la información. Cabe destacar que la fuente almacena más información sobre las aplicaciones de la que se almacena en el repositorio local, por lo que será necesaria una etapa de procesamiento tras obtener la respuesta a la petición `HTTP`.

Descarga de una nueva aplicación

Cuando el Warehouse no disponga de la información de una aplicación, el Sistema Integrador se verá obligado a descargar una nueva aplicación de las fuentes de descarga de aplicaciones. Una vez descargada, realizará una petición `POST` al Warehouse indicándo el nombre de paquete de la aplicación. Por motivos de eficiencia, el Sistema Integrador no esperará a que la descarga termine en el Warehouse, sino que simplemente informará a éste de la aplicación que debe descargar, a modo de "recado" despreocupándose del proceso de descarga interno.

Validación del nombre introducido por el usuario

Aunque no estuviese contemplado originalmente en los requisitos funcionales, por temas de usabilidad se permite especificar el nombre de la aplicación como una subcadena del nombre de paquete completo, de forma que, por ejemplo, en vez del el nombre completo `com.whatsapp`, se pueda especificar simplemente `whatsapp`. Se hará una petición de la subcadena al Warehouse y éste devolverá la app más parecida a ese nombre de la subcadena que tenga almacenada. Posteriormente, se hará una validación del nombre del paquete completo mediante una petición HTTP a la URL de detalles de Google Play `https://play.google.com/store/apps/details?id=<nombredelpaquete>`, si ésta devuelve el código de respuesta `200`, es que la aplicación existe en Google Play. Si por el contrario devuelve un error `404`, esto significará que la aplicación no existe, por tanto se considerará el nombre de paquete como no válido.

5.16.2. Fuentes de descarga de aplicaciones

Cuando la información no esté disponible ni en el repositorio local ni en el Warehouse, se recurrirá a las fuentes de descarga de aplicaciones, donde se descargarán los archivos fuente (`.apk`) que posteriormente serán descomprimidos y procesados. Las dos fuentes de datos son accesibles a través de la Web y no requieren de ningún tipo de autentificación para acceder a ellas, por lo que se descargarán las aplicaciones por medio de Web Scraping, cuyo proceso se detallará en el capítulo 6.

5.16.3. *Google Play*, obtención de informacion comercial

Para obtener el nombre comercial y categoría de una aplicación, así como la URL con la imagen con el logotipo, se va a acceder a la página web de Google Play, donde éstos se obtendrán mediante Web Scraping.

Capítulo 6

Implementación

6.1. Cambios con respecto al diseño inicial

6.1.1. Soporte de archivos `.xapk`

El cambio más importante con respecto al diseño inicial ha sido el dar soporte a archivos `.xapk`. Tras realizar las primeras pruebas, nos dimos cuenta de que en muchos casos las aplicaciones descargadas de las fuentes de datos, en especial de APK Pure, se encontraban en formato `.xapk` en vez de en el formato "original" `.apk`. Los archivos `.xapk` incoporan un paso más de compresión que puede hacer a los archivos más eficientes en memoria y ayudar en algunos procesos de instalación.

Para tratar con estos archivos, tanto en la descarga como en la subida, se requiere añadir un paso más de descompresión, que se realiza con el comando `unzip`, que extrae, entre otros archivos, el archivo fuente `.apk` que ya se puede tratar como el resto de archivos fuente. Esto da mayor flexibilidad al servicio y permite consultar muchas más aplicaciones.

6.1.2. Últimas versiones de las apps y forzado de descarga

En el capítulo de diseño, se especificó (ver figura 5.17) que el primer paso a la hora de consultar una aplicación sería buscar la última versión disponible. Esto requiere de la puesta en marcha de Web Scraping para consultar cuál es la última versión de la aplicación disponible en las fuentes. Esta puesta en marcha se hace bastante costosa (especialmente en la máquina virtual en que se ha lanzado el servicio) y ralentiza considerablemente la consulta, amén de que como no se están descargando las aplicaciones del market oficial Google Play, cuando una aplicación se actualiza en Google Play habría que esperar a que esté disponible en APK Pure o Evozi APK Downloader, lo que suele tardar cierto tiempo. Por todo ello, se decidió optar por un sistema periódico de actualización de las aplicaciones.

Para ello, se ha incluido la opción de forzar la descarga y análisis de la aplicación aunque ya haya sido consultada anteriormente. Esto se hace mediante un argumento `?f=true` dentro de la URL principal, de forma que si se quiere forzar la descarga, hay que acceder a la URL `http://<ip>/scorer/?f=true`. De este modo se re-ejecuta la descarga y análisis de la aplicación en su última versión disponible. De este modo, cada cierto tiempo, se puede forzar la actualización de las aplicaciones. Además, resulta muy sencillo programar un pequeño script que actualice las aplicaciones dentro del servidor, sin recurrir a la consulta manual de las URL, basta con especificar el argumento `force_download = true` dentro de la función `retrieve_app_info` del controlador.

6.1.3. Adiciones a la interfaz gráfica

Disclaimer

Aunque no estaba considerado en el bocetaje inicial, se decidió añadir un pequeño *disclaimer* en la parte inferior de las ventanas (ver apéndice A), con el siguiente contenido.

While every effort has been made to provide accurate and up-to-date information, APK Falcon makes no guarantees about its accuracy or completeness, and is not responsible for any actions taken by users based on this information. Users should always act with caution and carefully review the permissions requested by apps before granting access.

La función de este disclaimer es "descargar de responsabilidades" sobre la interpretación o acciones que los usuarios puedan realizar a la vista de los resultados que se muestran, así como avisar a los usuarios de que los resultados en todo caso deberían ser tomados con precaución.

Tooltips

También se decidió añadir *tooltips* que se muestren al pasar el ratón por encima de los distintos iconos que se utilizan en la interfaz gráfica, de forma que en todo momento se pueda saber qué representan los iconos mostrados sin saturar la interfaz gráfica de información complementaria. Una descripción de estos tooltips se puede ver en el manual de usuario, en el apéndice A.

6.2. Organización del código

El código del servicio, así como la documentación de diseño, se encuentran disponibles en el repositorio de GitLab `https://gitlab.inf.uva.es/javcres/uva-apk-falcon`. A continuación se muestra la organización en ficheros y directorios del código del servicio.

```
uva_apk_falcon/
├── db.sqlite3
├── integration_system
│   ├── __init__.py
│   ├── constants.py
│   ├── extractor.py
│   ├── main.py
│   └── wrappers
│       ├── scraping.py
│       └── warehouse.py
├── integration_tests.ipynb
├── manage.py
├── manifest.json
├── populate_test_database.py
├── referenceCategoryNames.json
├── referencePermissionsDescriptions.csv
├── referencePermissionsGroupsScores.csv
├── scorer
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── constants.py
```

```
│   ├── controller.py
│   ├── forms.py
│   ├── models.py
│   ├── static
│   │   └── scorer
│   │       ├── barplot.js
│   │       ├── img
│   │       │   ├── danger_icon.png
│   │       │   ├── information_icon.png
│   │       │   ├── ok_icon.png
│   │       │   ├── star_icon.png
│   │       │   ├── warning_icon.png
│   │       │   └── worst_icon.png
│   │       ├── main.js
│   │       ├── speedometer.js
│   │       ├── style.css
│   │       └── timeplot.js
│   ├── templates
│   │   └── scorer
│   │       ├── about.html
│   │       ├── index.html
│   │       └── report.html
│   ├── templatetags
│   │   ├── __init__.py
│   │   └── scorer_extras.py
│   ├── tests.py
│   ├── urls.py
│   ├── validators.py
│   └── views.py
├── static
│   ├── falcon.png
│   ├── favicon.png
│   ├── logo.png
│   ├── logo_background.png
│   ├── logo_inf.png
│   ├── logo_uva.png
│   ├── logo_dpto.png
│   └── style.css
├── templates
│   └── base.html
└── uva_apk_falcon
    ├── __init__.py
    ├── asgi.py
    ├── settings.py
    ├── urls.py
    └── wsgi.py
```

6.3. Comunicación con las fuentes de datos

6.3.1. *Warehouse* de aplicaciones móviles

A continuación se describe el detalle de las peticiones `HTTP` que se realizan al Warehouse para consultar los metadatos de las aplicaciones y subir nuevas aplicaciones.

Consultar metadatos

Para consultar la información asociada a una aplicación se hará una petición `GET` a la URL `https://<ip>:<puerto>/get/app/package?package=<nombre del paquete>`. Esta petición utiliza una clave API para autentificarse que se pasa como parte de los parámetros de la petición:

```
{
    "Authorization": "Bearer <clave>"
}
```

El Warehouse responderá con código `200` si la consulta ha sido correcta, en el cuerpo de la respuesta serán los metadatos asociados a la app con formato `JSON`. Si el código devuelto es `401`, quiere decir que la clave API es incorrecta. Si es `404`, quiere decir que la app no existe en el Warehouse. Por último, un código `422` quiere decir que la consulta está mal formulada, típicamente porque falta el nombre de paquete como parámetro de la URL. En caso de que la consulta haya sido correcta y exitosa, se devolverán los metadatos con el siguiente formato:

```
{
    "App": {
        "hash": "00000000",
        "package": "com.example",
        "version_code": 100,
        "version_name": "1.0.0",
        "min_sdk_version": 21,
        "target_sdk_version": 33,
        "max_sdk_version": None,
        "category": "EXAMPLES",
        "uses_permission_list": [
            {
                "Permission": {
                    "name": "android.permission.EXAMPLE",
                    "protection_level": "dangerous|instant",
                    "declared_group_list": None,
                    "rank_list": [
                        {
                            "Rank": {
                                "value": 0.5,
                                "rank_name": "EXAMPLE",
                                "permission_name": "android.permission.EXAMPLE",
                            }
                        }
                    ],
                }
```

```
            },
        ],
        "defines_group_list": None,
        "extraction_metadata_list": [
            {
                "ExtractionMetadata": {
                    "source": "Example",
                    "method": "web scraping",
                    "timestamp": "2023-05-11T17:42:22",
                }
            },
        ],
        "score_list": [
            {
                "Score": {
                    "value": 5.00,
                    "rank_name": "EXAMPLE",
                    "app_hash": "000000000",
                }
            }
        ],
    }
}
```

Subir una nueva aplicación

En este caso se realizará una petición `POST` a la URL `https://<ip>:<puerto>/post/app/package`. La cabecera de la petición será la misma que en el `GET`, en este caso se incluirán los siguientes parámetros en el cuerpo de la petición:

```
{
    "package": <package_name>
}
```

En este caso la petición devolverá los mismos códigos de respuesta que la anterior, pero en el caso del código `200`, el cuerpo de la respuesta contiene {`"status": "requested"`} si la aplicación se ha puesto a la cola o {`"status": "busy"`} si la cola de aplicaciones en descarga está llena.

6.3.2. *APK Pure*

Esta fuente de datos y las siguientes se consultan mediante Web scraping, para ello se utiliza Firefox como navegador en su modo *headless*. En el caso de APK Pure se parte directamente de la ventana de resultados de búsqueda (figura 6.1, URL `https://m.apkpure.com/es/search?q=com.whatsapp`) debido a que es más eficiente que hacerlo desde la principal e introducir el término de búsqueda. Tenemos que seleccionar el primer resultado de la búsqueda y pulsar en el botón "Descargar". Para ello identificamos el botón mediante el selector `CSS a.first-info`, lo pulsamos y esperamos a que nos lleve a otra ventana.

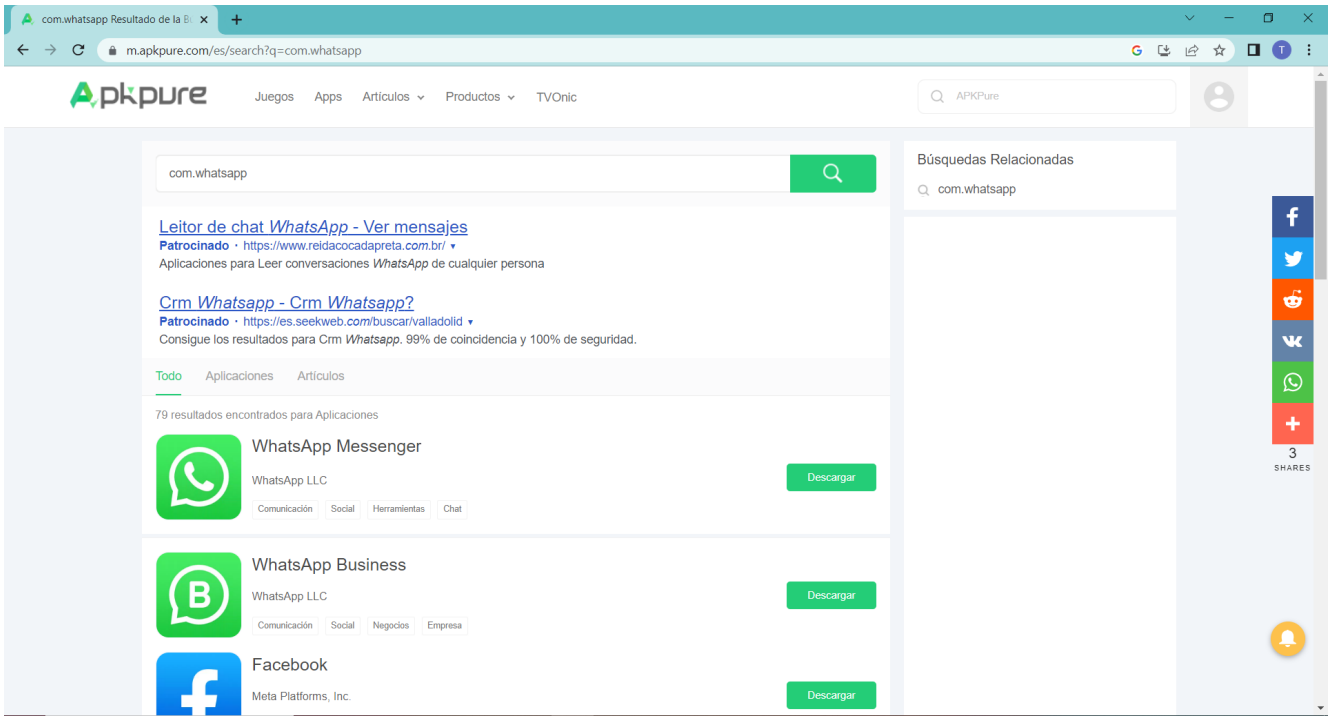


Figura 6.1: Ventana de búsqueda de APK Pure.

A continuación, se nos lleva a una ventana intermedia (figura 6.2, URL `https://m.apkpure.com/es/whatsapp-android/com.whatsapp`) donde tenemos que pulsar el botón "Descargar APK", identificado mediante el selector `a.download-start-btn`.

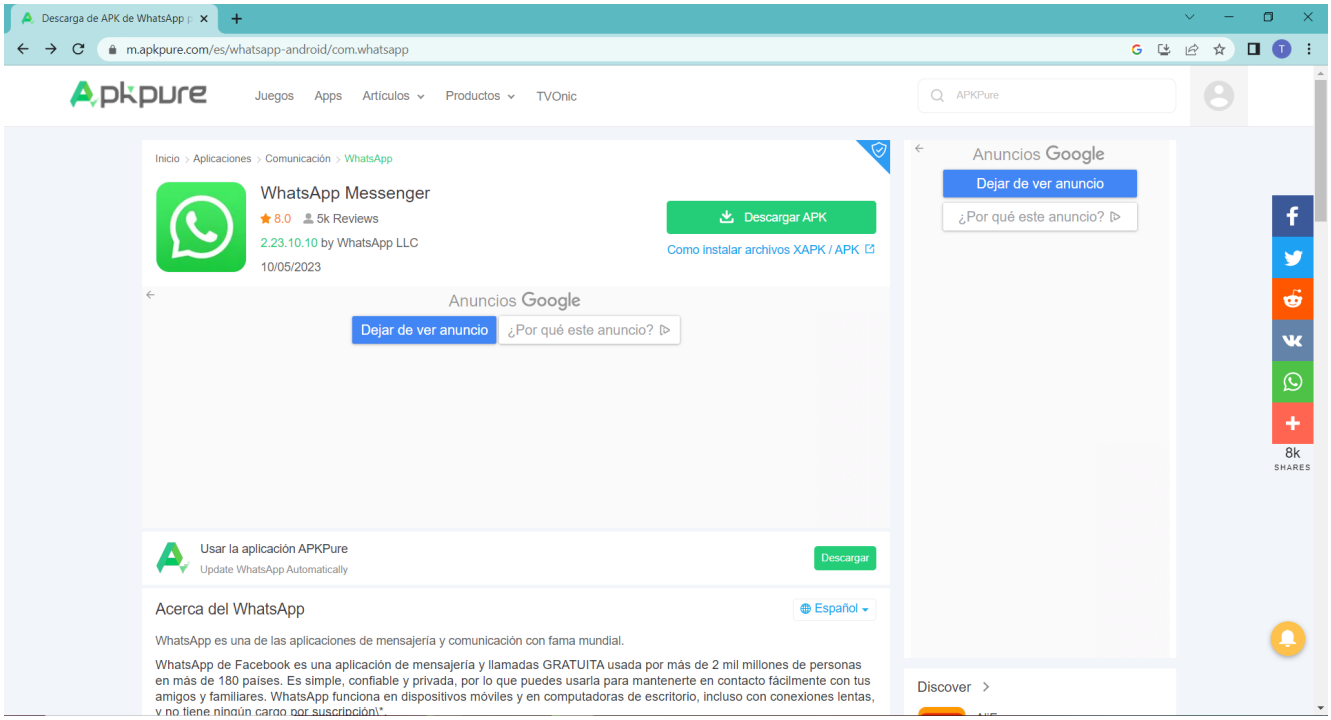


Figura 6.2: Ventana intermedia de APK Pure.

Por último, se nos lleva a la ventana de descarga mostrada en la figura 6.3 (URL `https://m.apkpure.com/es/whatsapp-android/com.whatsapp/download`) donde, de nuevo, tenemos

que pulsar el botón “Descargar APK”, identificado mediante el selector `a.download-start-btn`.

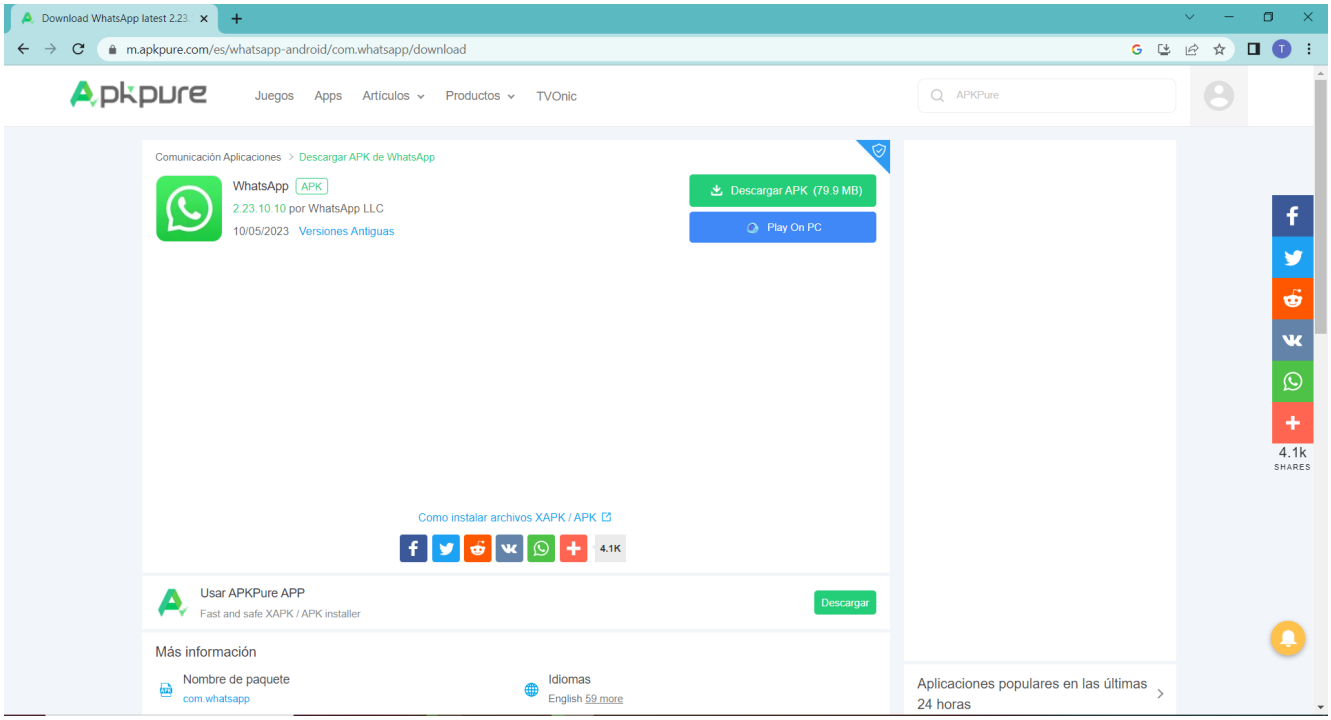


Figura 6.3: Ventana de descarga de APK Pure.

Una vez se ha pulsado el botón, comienza la descarga del archivo `.apk`. Se puede ver que se está descargando el archivo porque en el directorio de descarga aparecen dos: un archivo `.apk` y otro `.part`. Cuando acaba la descarga se elimina el `.part`, por lo que escaneando la existencia de archivos `.part` en el directorio de descarga podemos esperar a que termine la descarga, implementando también un sistema de timeout.

6.3.3. *Evozi APK Downloader*

En este caso se parte de la web que se muestra en la figura 6.4 (URL `https://apps.evozi.com/apk-downloader/`), en la que hay que introducir el nombre de paquete de la aplicación en la barra de búsqueda central y pulsar el botón que está debajo. Tras una primera inspección, se ve que los id se asignan de forma dinámica y no siguen ningún tipo de patrón consistente en el tiempo, por lo que hay que seleccionar los elementos basándonos en la clase o en la etiqueta `HTML` correspondiente. En este caso podemos identificar el `input` de búsqueda mediante el selector `CSS` `input.input-lg` y el botón de búsqueda mediante `button.btn-lg`.

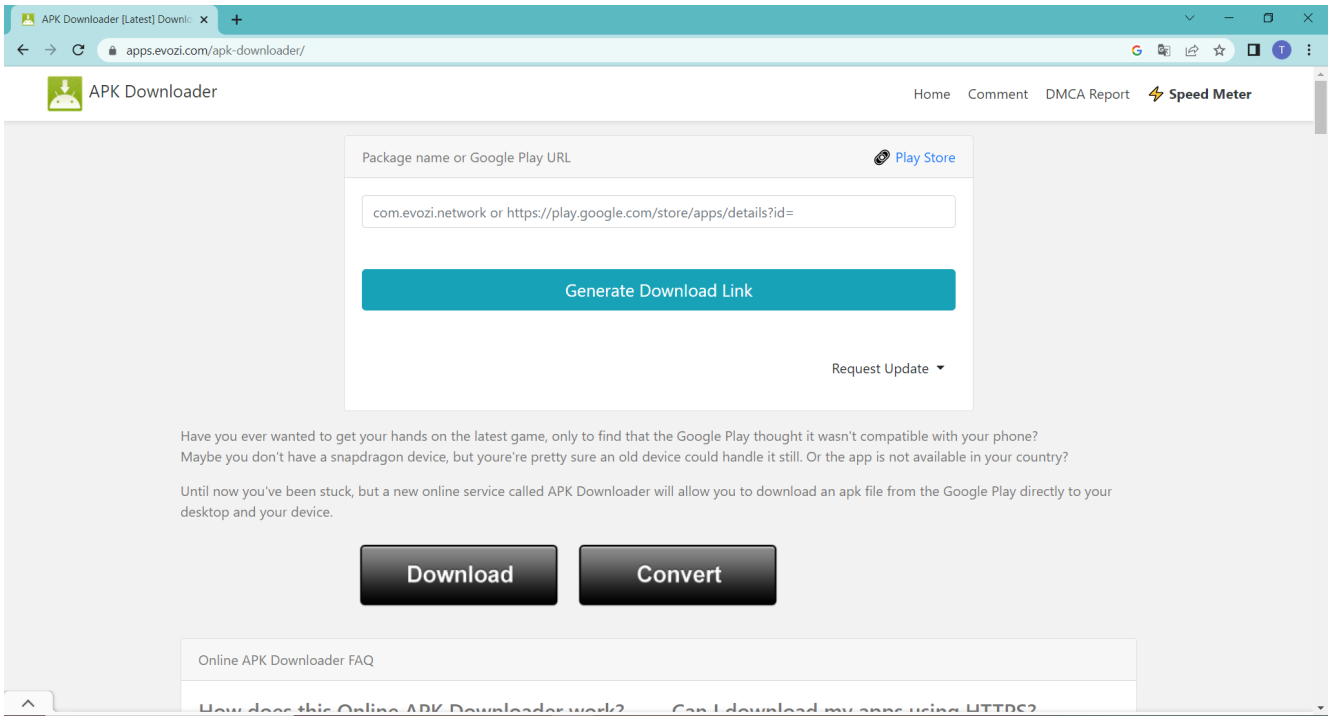


Figura 6.4: Ventana principal de *Evozi APK Downloader*.

Tras activar el proceso de búsqueda, se añade contenido a la misma ventana que se muestra en la figura 6.5, vemos que se muestra un botón de "descargar ahora" que tenemos que pulsar para descargar el archivo fuente, este botón se identifica mediante el selector `a.btn-success`. Una vez se pulsa en este botón, se inicia la descarga y se espera a que se complete, de nuevo con un sistema de timeout en caso de que la descarga falle o el archivo sea demasiado pesado.

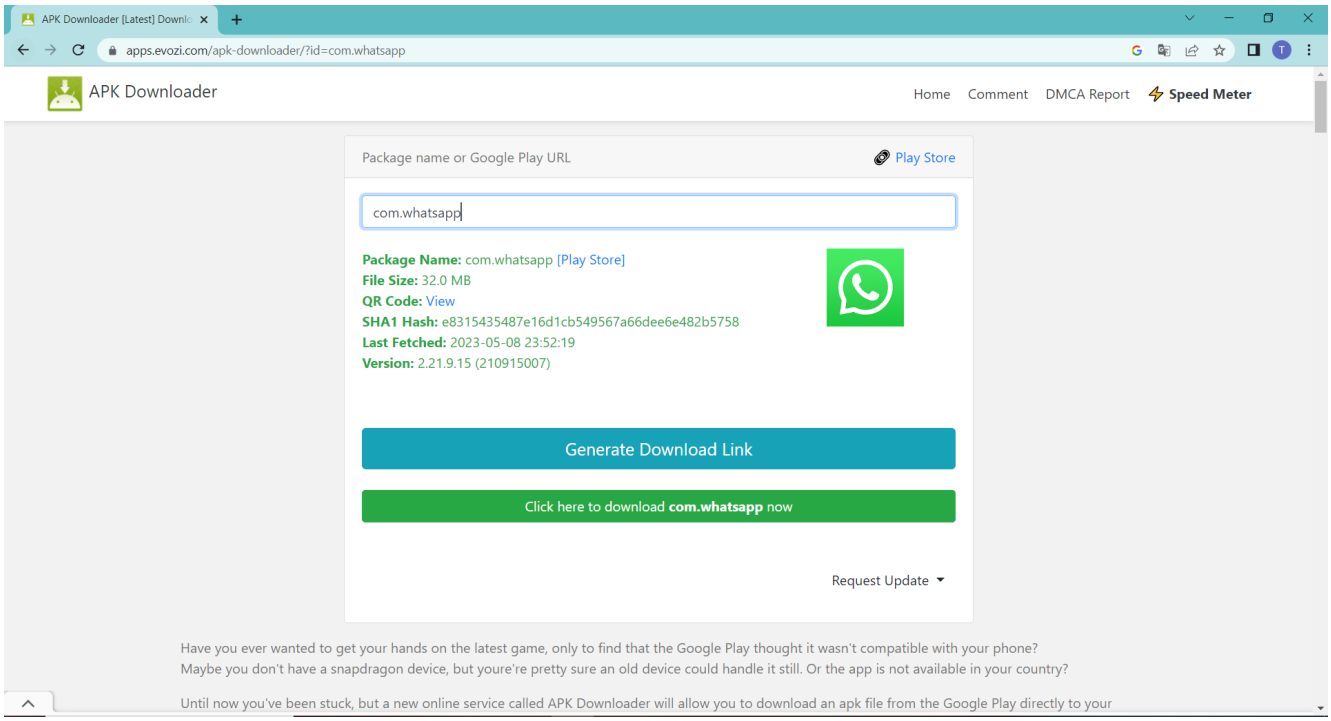


Figura 6.5: Ventana de descarga de *Evozi APK Downloader*.

6.3.4. *Google Play*

De Google Play se extren el nombre comercial, la categoría y el logotipo de las aplicaciones mediante Web Scraping. Cabe destacar que en la web de Google Play, los `id` y `class` de los elementos HTML se asignan de forma distinta (a cadenas de letras aleatorias) cada vez que se accede a la página, por lo que ha habido que buscar otras formas de acceder a los elementos. Dado el nombre de paquete completo de una aplicación, partimos de la URL de detalles `https://play.google.com/store/apps/details?id=<nombredepaquete>`, se ve un ejemplo en la figura 6.6.

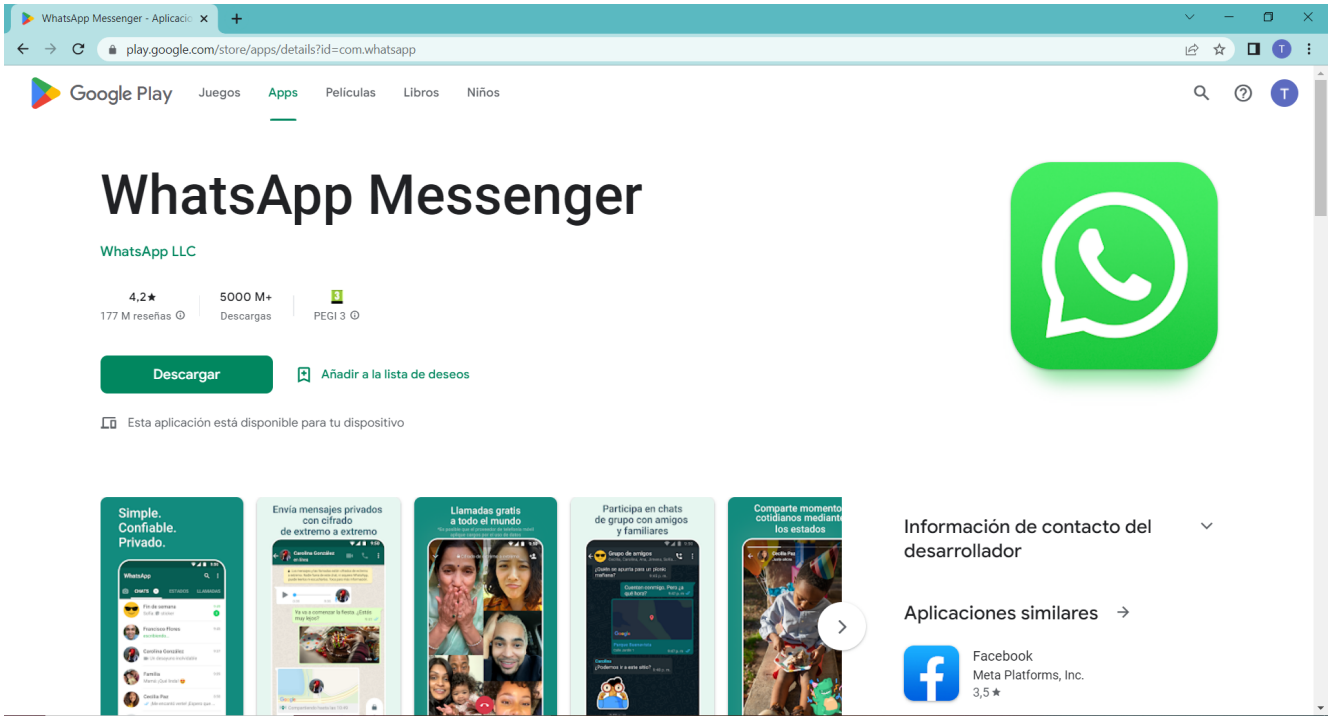


Figura 6.6: Parte superior de la ventana de detalles de una aplicación de Google Play.

A la hora de sacar el nombre comercial, si exploramos los elementos de la web vemos que este está dentro de una etiqueta `<span>` dentro de una etiqueta `<h1>`, por lo que se puede obtener fácilmente mediante una expresión regular. Al haber muchas imágenes en esta web, hay que tener en cuenta que las correspondientes a los logotipos empiezan por la URL `https://play-lh.googleuserconten` por lo que basta con escoger el primer elemento `<img>` cuyo atributo `href` comience por la URL mencionada.

La categoría de la aplicación se puede ver más abajo en la misma URL. Se ve un ejemplo en la figura 6.7, donde se puede ver que hay un botón con la categoría debajo de la información de la aplicación.

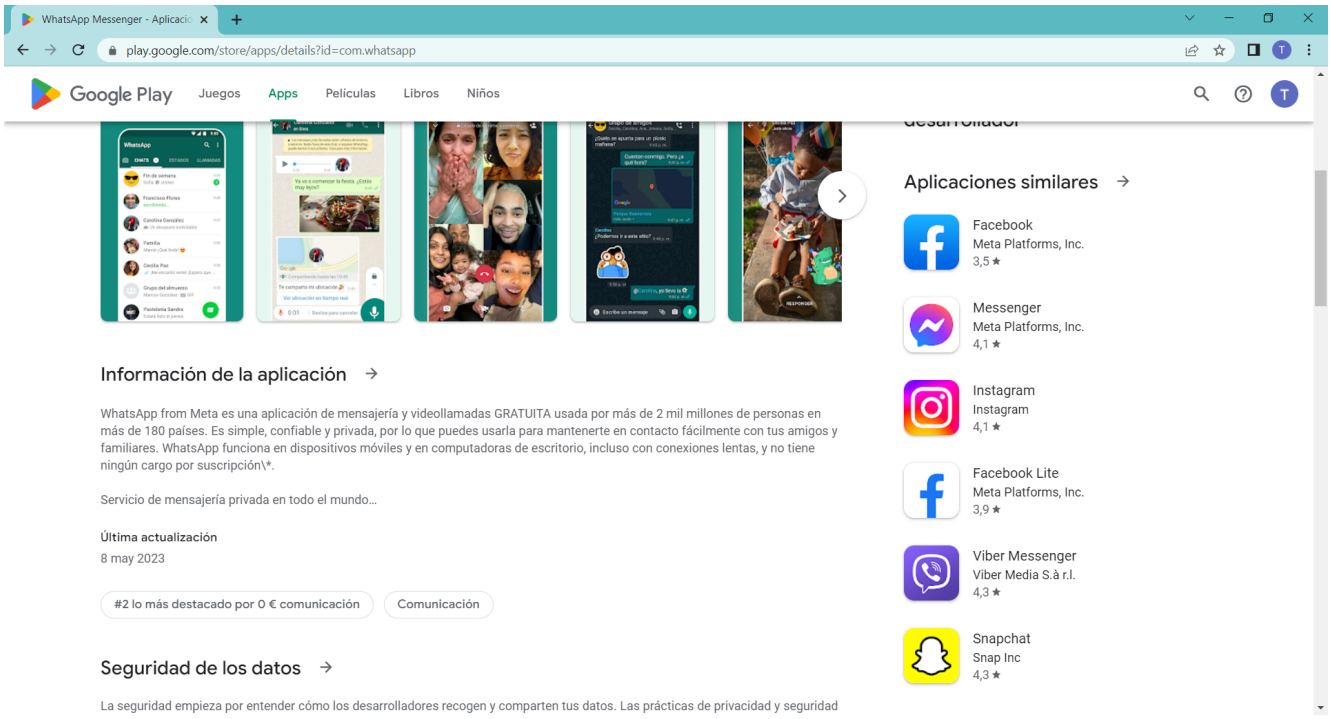


Figura 6.7: Parte inferior de la ventana de detalles de una aplicación de Google Play.

El núcleo de estos botones está en elmentos de tipo `<a>` cuyo atributo `href` toma la forma `https://play.google.com/store/apps/category/<nombredelacategora>`, siendo ejemplos de nombres de categoría `COMMUNICATION` o `PRODUCTIVITY`. Hay que tener cuidado porque no hay un único elemento que cumpla estas características, en la figura 6.6 se puede ver que en la barra de menú superior hay una etiqueta de "Niños", esto es un enlace a la URL `https://play.google.com/store/apps/category/FAMILY`, por lo que tenemos que seleccionar el último elemento que encontremos con estas características.

Capítulo 7

Pruebas

Para comprobar correcto cumplimiento de los requisitos expresados en 4.2, se realizan dos tipos de pruebas: de funcionamiento y de aceptación.

7.1. Pruebas de funcionamiento

En los sistemas informáticos existen tres tipos básicos de pruebas sobre el funcionamiento: tests unitarios, tests de integración y tests del sistema. Los tests unitarios prueban cada módulo software de forma aislada mientras que los tests de integración prueban los módulos que interactúan de forma conjunta. Los tests del sistema se encargan de probar los casos de uso en su conjunto sobre el sistema completo para verificar que se cumplen todos los requisitos.

7.1.1. Tests unitarios

Se han realizado tests unitarios para el modelo con el que trabaja el servidor, es decir, las clases `App, Permission, PermissionGroup` y `PermissionAssignment`, se encuentran en el archivo `uva_apk_falcon/scorer/tests.py`. Dentro del servidor, se pueden ejecutar con el comando `python manage.py test` y se puede ver que todas las funciones de estas clases están cubiertas.

No se han hecho tests unitarios para el resto del código, ya que el resto del código no puede funcionar de forma aislada, sino en conjunto con otros módulos, por ello estos tests se ven absorbidos por los tests de integración.

7.1.2. Tests de integración

Estas pruebas sirven para asegurar la correcta interacción entre módulos y el correcto funcionamiento de las operaciones más complejas. En este caso tenemos que probar la integración entre el Servidor y el Sistema Integrador. Los módulos a considerar son los siguientes:

Módulo 1 Servidor - Controlador: crea y recupera objetos del Modelo, accediendo al sistema integrador si fuese preciso para obtener los metadatos necesarios para la creación de objetos del Modelo.

Módulo 2 Servidor - Modelo: clases que implementan la lógica del servicio.

Módulo 3 SI - main: punto de entrada al sistema. Proporciona una fachada que abstrae al Controlador de los detalles de implementación. Implementa la lógica del Sistema Integrador.

Módulo 4 SI - extractor: se encarga de descomprimir un archivo de aplicación y extraer la información necesaria del *Manifest*, así como de normalizar los datos que llegan del Warehouse.

Módulo 5 SI - scraping: implementa el acceso a las fuentes de datos externas a las que se accede mediante web-scraping.

Módulo 6 SI - warehouse: implementa el acceso al Warehouse de aplicaciones móviles.

Para realizar estos tests se utilizará el *shell* que proporciona Django, que permite poner en marcha una consola de comandos con todos los ajustes y configuración del servidor cargados (usando la orden `python manage.py shell`). Se dejarán fuera de estos tests la parte correspondiente al front-end, puesto que estos tests ya se van a hacer de forma manual como parte de los tests del sistema. Los tests se pueden encontrar en el archivo `uva_apk_falcon/integration_tests.ipynb`, para utilizarlos primero se debe poblar la base de datos ejecutando para ello el archivo encargado de poblarla `uva_apk_falcon/populate_test_database.py`. Estos tests se realizarán llamando a las funciones del controlador (dentro del *Módulo 1*) `retrieve_app` y `file_upload`, donde el acceso (y sobre todo el no acceso) a los módulos que indica cada test se monitorizará mediante los logs del servicio, que nos permiten saber de forma muy rápida a qué módulos se está accediendo y, por ende, si el funcionamiento del sistema es la descrita en el diseño. Los tests están diseñados para ser realizados de forma secuencial en el orden en que se muestran a continuación.

ID	**Módulos**
1	1,2
Descripción	
Búsqueda de una aplicación presente en la base de datos local.	
Pre-requisitos	
Ninguno.	
Input	
package_name = “com.whatsapp”	
Resultado Esperado	
La información se extrae de la base de datos local, no se accede al sistema integrador.	

ID	Módulos
2	1, 2, 3, 4, 6
Descripción	
Búsqueda de una aplicación no presente en la base de datos local pero presente en el warehouse.	
Pre-requisitos	
Variable que controla si se usa el warehouse a True.	
Input	
package_name = “org.telegram.messenger”	
Resultado Esperado	
La información se intenta extraer de la base de datos local, luego se extrae del warehouse, no se hace scraping en ningún momento.	

ID	Módulos
3	1, 2
Descripción	
Comprobación de que la información obtenida de fuentes externas es guardada en la base de datos local.	
Pre-requisitos	
Se ha ejecutado el test de integración 2.	
Input	
package_name = “org.telegram.messenger”	
Resultado Esperado	
La información se extrae de la base de datos local, no se accede al sistema integrador.	

ID	Módulos
4	1, 2, 3, 4, 5, 6
Descripción	
Descarga de una aplicación no presente ni en la base de datos local ni en el warehouse. Fuente de descarga APK Pure.	
Pre-requisitos	
Se ha cargado APK Pure como la única fuente de desv	
Input	
package_name = “jp.naver.line.android”	
Resultado Esperado	
La información se intenta extraer de la base de datos local, posteriormente del warehouse. Finalmente se hace scraping solamente a la fuente APK Pure. Se sube la información de la aplicación al warehouse.	

ID	Módulos
5	1, 2, 3, 4, 5, 6
Descripción	
Descarga de una aplicación no presente ni en la base de datos local ni en el warehouse. Fuente de descarga Evozi APK Downloader.	
Pre-requisitos	
Se ha cargado Evozi APK Downloader como la única fuente de descarga del sistema.	
Input	
package_name = “com.viber.voip”	
Resultado Esperado	
La información se intenta extraer de la base de datos local, posteriormente del warehouse. Finalmente se hace scraping solamente a la fuente Evozi APK Downloader. Se sube la información de la aplicación al warehouse.	

ID	Módulos
6	1, 2, 3, 4
Descripción	
Subida de un archivo fuente (.apk).	
Pre-requisitos	
Ninguno.	
Input	
Archivo Fuente de aplicación “Whatsapp” (descargado manualmente de APK Pure).	
Resultado Esperado	
El archivo se descomprime y procesa. No se sube la información al warehouse ni se descarga ningún tipo de dato.	

ID	Módulos
7	1, 2, 3, 4, 5, 6
Descripción	
Forzar la descarga de una app.	
Pre-requisitos	
Ninguno.	
Input	
package_name = “com.whatsapp” force_download = True	
Resultado Esperado	
No se accede a la base de datos local ni al warehouse para la obtención de la información, sino directamente al scraping. Los resultados se suben al warehouse.	

ID	Módulos
8	1, 2, 3, 5, 6
Descripción	
Búsqueda de una app no presente ni en el warehouse ni en las fuentes de datos.	
Pre-requisitos	
Ninguno.	
Input	
package_name = “es.race.asistencia.genesis.assist”	
Resultado Esperado	
Se intenta obtener la información de la app primero del warehouse y luego de las dos fuentes de datos. En todos casos falla, no se descarga ningún tipo de archivo.	

7.1.3. Tests del sistema

Los tests del sistema se han hecho utilizando la base de datos reducida, cuyo script de población se puede encontrar en el archivo `uva_apk_falcon/populate_test_database.py` (hay que ejecutarlo utilizando el *shell* que proporciona Django para que se carguen correctamente los ajustes del mismo modo que los tests de integración). A continuación se muestra la tabla con todos los casos de prueba, describiendo el caso, las vistas iniciales y finales, sus entradas, su salida esperada y el resultado obtenido.

Descripción	Vista inicial	Vista final	Input	Resultado esperado
Búsqueda de una aplicación	Ventana principal */scorer*	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	package name: com.whatsapp	Se muestra una barra de carga y mensajes describiendo el proceso de descarga y análisis. Se muestra el informe de privacidad de la aplicación.
Búsqueda de una aplicación mediante enlace a Google Play	Ventana principal */scorer*	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	package name: https://play.google.com/store/apps/details?id=com.whatsapp	Se muestra una barra de carga y mensajes describiendo el proceso de descarga y análisis. Se muestra el informe de privacidad de la aplicación.
Búsqueda de una aplicación mediante enlace a Google Play inexistente	Ventana principal */scorer*	Ventana principal */scorer*	package name: https://play.google.com/store/apps/details?id=com.wasap	Se muestra la barra de carga, a continuación devuelve a la ventana principal donde se muestra un mensaje de error.
Búsqueda de una aplicación inexistente	Ventana principal */scorer*	Ventana principal */scorer*	package name: com.wasapp	Se muestra la barra de carga, a continuación devuelve a la ventana principal donde se muestra un mensaje de error.
Búsqueda de un nombre de paquete mal formado	Ventana principal */scorer*	Ventana principal */scorer*	package name: whatsapp	Se muestra la barra de carga, a continuación devuelve a la ventana principal donde se muestra un mensaje de error.
Subida de un archivo .apk	Ventana principal */scorer*	Ventana de informe de la aplicación */scorer/report/uploaded*	Archivo con extensión .apk válido	Se muestra una barra de carga y mensajes describiendo el proceso de subida y análisis. Se muestra el informe de privacidad de la aplicación.
Subida de un archivo .xapk	Ventana principal */scorer*	Ventana de informe de la aplicación */scorer/report/uploaded*	Archivo con extensión .xapk válido	Se muestra una barra de carga y mensajes describiendo el proceso de subida y análisis. Se muestra el informe de privacidad de la aplicación.
Subida de un archivo con extensión inválida	Ventana principal */scorer*	Ventana principal */scorer*	Archivo con extensión .pdf	Se muestra la barra de carga, a continuación devuelve a la ventana principal donde se muestra un mensaje de error.

Obtener información de un grupo de permisos	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Click en una de las barras del gráfico de barras	Se muestra un *tooltip* con la descripción y puntuación del grupo.
Revocación de un grupo de permisos	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Click en una de las barras del gráfico de barras	La barra cambia a un color más transparente. Se actualiza el gráfico de velocímetro. Se añade una acción al gráfico de últimas acciones.
Concesión de un grupo de permisos revocado	Ventana de informe de la aplicación */scorer/report/com.whatsapp*. Al menos un permiso ha sido revocado	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Click en una de las barras del gráfico de barras correspondiente a un permiso revocado	Se muestra un *tooltip* con la descripción y puntuación del grupo. La barra vuelve al color original. Se actualiza el gráfico de velocímetro. Se añade una acción al gráfico de últimas acciones.
Obtener información de una acción realizada	Ventana de informe de la aplicación */scorer/report/com.whatsapp*. Al menos un	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Pasar el ratón sobre uno de los puntos del gráfico de últimas acciones	Se muestra un *tooltip* con la información de la acción.

	permiso revocado.			
Reestablecer el gráfico de acciones realizadas	Ventana de informe de la aplicación */scorer/report/com.whatsapp*. Al menos un permiso revocado.	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Pulsar el botón "Clear" del gráfico de acciones realizadas	Se reestablece el gráfico, solamente se muestra un punto.
Exportar últimas acciones	Ventana de informe de la aplicación */scorer/report/com.whatsapp*. Al menos un permiso revocado.	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Pulsar el botón "Export actions as JSON"	Se descarga un archivo .json con las acciones realizadas.
Exportar informe de privacidad en JSON	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Ventana con texto en formato JSON. */scorer/export/com.whatsapp/?format=json*	Pulsar el botón "Export as JSON"	Se muestra una nueva ventana con el informe en formato JSON.
Exportar informe de privacidad en XML	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Ventana con texto en formato JSON. */scorer/export/com.whatsapp/?format=xml*	Pulsar el botón "Export as XML"	Se muestra una nueva ventana con el informe en formato XML.
Acceder a la información desde la ventana principal	Ventana principal */scorer*	Ventana de información */about*	Pulsar en el enlace que se muestra al final del texto About	Se muestra la ventana de información.
Acceder a la información desde la ventana de informe	Ventana de informe de la aplicación */scorer/report/com.whatsapp*	Ventana de información */about*	Pulsar en el enlace que se muestra al final del texto "About"	Se muestra la ventana de información.

Figura 7.1: Tests del sistema

7.2. Pruebas de aceptación

Para cerrar las pruebas del sistema, se han realizado tests de usabilidad a 10 usuarios potenciales del sistema para descubrir posibles fallos o puntos de mejora del mismo (así como evaluar los requisitos [RNF14] y [RNF18]). Los usuarios escogidos se clasifican de acuerdo de acuerdo con sus conocimientos sobre el sistema Android y la privacidad en tres grupos: usuarios inexpertos, usuarios curiosos y usuarios avanzados, estando estos últimos muy familiarizados con el sistema de permisos. Estos usuarios también serán clasificados según su rango de edad en tres grupos: jóvenes, mediana edad y mayores. En todo momento se garantizará el anonimato en las respuestas de los usuarios de prueba, posteriormente se muestra la hoja de consentimiento que se pedirá firmar a los usuarios con el motivo de informar sobre cómo se van a tratar sus datos y formalizar su consentimiento.

Las pruebas de usabilidad seguirán la siguiente estructura:

1. Se dará una breve introducción sobre el servicio que se está desarrollando y un breve contexto

sobre el sistema de permisos para aquellos usuarios inexpertos.

2. Se pedirá a los usuarios realizar una serie de tareas utilizando el servicio web que se detallan a continuación. Para cada prueba, se anotará si el usuario es capaz de completar la tarea por sí solo, el tiempo que tarda en hacerlo y observaciones adicionales (por ejemplo, aquellos puntos en los que el usuario se ha tomado más tiempo).

3. Se pedirá a los usuarios una valoración numérica y opinión del servicio que incluirá los siguientes aspectos (en una escala del 1 al 5, siendo 1 muy malo y 5 muy bueno): valoración global del servicio, claridad en la interacción, valoración estética y comprensibilidad del informe de privacidad.

4. Se pedirá a los usuarios su opinión sobre qué aspectos creen que se podrían mejorar del servicio o comentarios que tengan a mayores.

7.2.1. Englobación dentro del desarrollo basado en incrementos

Como el trabajo se realiza utilizando una metodología de trabajo basada en incrementos, está contemplada la posibilidad de realizar múltiples cambios menores de forma rápida que sigan resultando en un producto final perfectamente funcional. Es por ello que se ha optado por aplicar las mejoras que se descubran a partir de los tests de usuario de forma inmediata, antes de realizar el siguiente test, en contraposición a "fijar" una versión del servicio, realizar todos los tests de aceptación y luego ya al final realizar todos los cambios a la vez. Además, cabe destacar que es de esperar que la mayoría de usuarios coincidan en los cambios más grandes que faciliten la usabilidad, por lo que realizando estos cambios mientras se hacen los tests hará que los usuarios se centren en aspectos distintos, intentando no repetir problemas ya detectados. Sin embargo, corremos el riesgo de que los cambios que se hagan a partir del test de un usuario sean perjudiciales para los siguientes usuarios, por lo que habría que deshacer el trabajo con la correspondiente pérdida de tiempo. Éste es un riesgo que se ha asumido. Además, hay que tener en cuenta que mediante esta forma de trabajo los resultados van a tener una dependencia temporal, por lo que habrá que utilizar métodos que tengan en cuenta el tiempo a la hora de analizarlos.

7.2.2. Tareas a desarrollar en las pruebas

A continuación se listan las tareas que se pedirá realizar a los usuarios.

- **Tarea 1**: Obtener el informe de privacidad de la aplicación *Whatsapp*. En un primer momento no se dirá nada sobre que hay que introducir el nombre del paquete en vez del nombre comercial.

- **Tarea 1.1**: Una vez se ha obtenido el informe, simular la concesión y revocación de algunos permisos.

- **Tarea 1.2**: Una vez se han concedido y revocado algunos permisos, exportar las acciones que acaban de realizar.

- **Tarea 1.3**: Ver el informe de privacidad completo y descargar los resultados.

- **Tarea 2**: Volviendo a la página de inicio, obtener un informe completo para una aplicación de la que se tiene el archivo fuente (se proporcionará el archivo).

- **Tarea 3**: En la ventana principal, obtener más información sobre el servicio (ir a la ventana *about*).

Documento de recopilación de respuestas

A continuación se muestra el documento que se utilizará para recoger las respuestas de los usuarios de acuerdo con las especificaciones anteriores.

Evaluación de usabilidad: APK Falcon

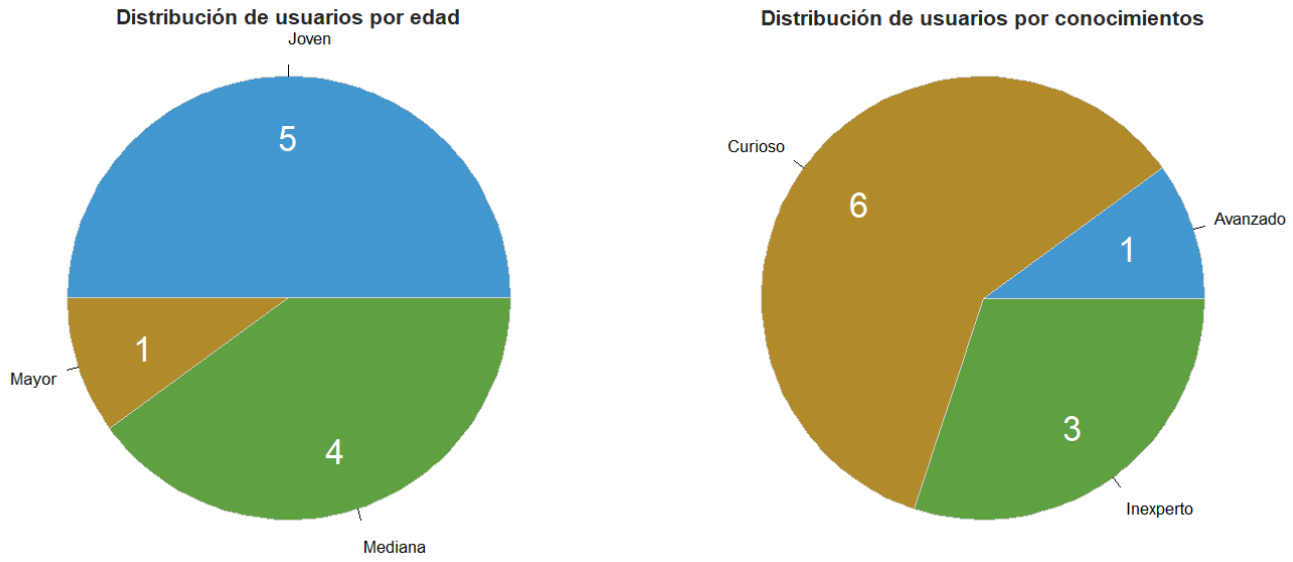
ID usuario		Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado

Tarea a realizar	1	Resultado		Tiempo empleado	
Observaciones					
Tarea a realizar	1.1	Resultado		Tiempo empleado	
Observaciones					
Tarea a realizar	1.2	Resultado		Tiempo empleado	
Observaciones					
Tarea a realizar	1.3	Resultado		Tiempo empleado	
Observaciones					
Tarea a realizar	2	Resultado		Tiempo empleado	
Observaciones					
Tarea a realizar	3	Resultado		Tiempo empleado	
Observaciones					

Valoración del sistema		1	2	3	4	5
	Global					
	Claridad en la interacción					
	Valoración estética					
	Comprensibilidad del informe					
Opinión y aspectos de mejora						

7.2.3. Resultados de las pruebas

Con las pruebas ya realizadas (se pueden ver las hojas de evaluación rellenas en el apéndice D), a continuación se procede a analizar los resultados. Este análisis se ha realizado usando el lenguaje de programación R (se pueden ver el código con la generación de tablas y gráficos en el archivo `analisis_usabilidad.R`). En primer lugar hemos de estudiar la distribución de edad y conocimientos de los 10 usuarios que se han podido evaluar. En la figura 7.2 se pueden ver *piecharts* con la distribución de los usuarios por edad y conocimientos. Debajo se puede ver la tabla con la distribución completa.



(a) (b)

Figura 7.2: Gráficos de tarta con la distribución de usuarios evaluados por (a) Edad y (b) Conocimientos previos.

	Conocimientos		
Edad	Avanzado	Curioso	Inexperto
Joven	0	4	1
Mayor	0	0	1
Mediana	1	2	1

Podemos ver que la mayoría de usuarios son Jóvenes y tienen ciertos conocimientos sobre los permisos en Android. Por limitaciones de tiempo, solamente se ha podido contar con una persona mayor y una persona con conocimientos avanzados. No hay que perder de vista que 10 usuarios es una población muy pequeña, por lo que en ningún caso podremos obtener conclusiones absolutas, sino simplemente posibles tendencias o sospechas sobre el grado de usabilidad del servicio.

Como se ha explicado en la sección 7.2.1, se ha decidido ir haciendo pequeños cambios para facilitar la usabilidad del servicio a medida que se han ido haciendo los tests de usabilidad, sin esperar hasta el final. Por ello, los resultados tienen una dependencia temporal con respecto al ID de usuario, por lo que a continuación se representarán los datos ordenados por ID de usuario en el eje X.

En primer lugar tenemos que estudiar el desempeño de los usuarios en las tareas, para ello realizamos gráficos (figura 7.4) donde en el eje X se representa el ID de usuario y en el eje

Y el tiempo que se tarda en realizar la tarea, siendo éste -1 en caso de que la tarea no haya sido completada. Aprovecharemos el color de los puntos y la forma para representar la edad y conocimientos previos tal y como se muestra en la figura 7.3.

Figura 7.3: Leyenda de colores y símbolos.

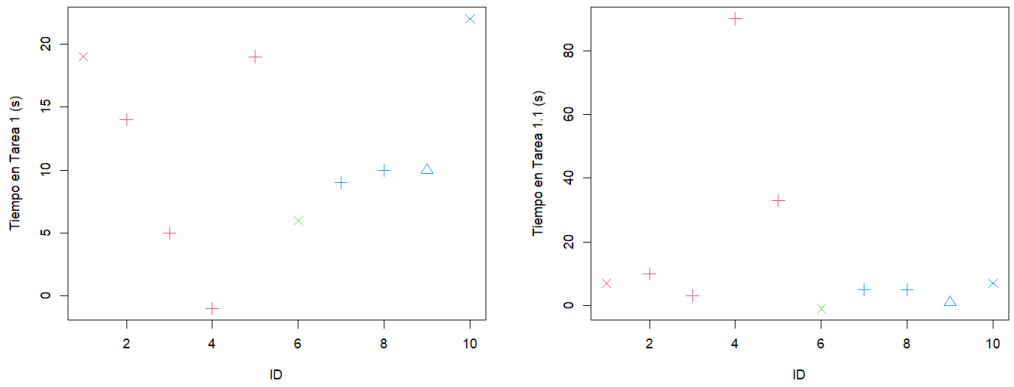
Podemos ver que en la mayoría de casos el tiempo de realización de las tareas es de menos de 30s, un valor bajo. La tarea más costosa parece ser la 1.1, simular la concesión/revocación de permisos, no siendo ésta muy evidente para los usuarios. Se pueden ver ligeras tendencias decrecientes en los gráficos (b), (c) y (d) que sugieren que los cambios están teniendo efecto positivo (de nuevo, no podemos afirmarlo rotundamente debido al número reducido de usuarios, solamente tenemos la sospecha de que esto ocurre). No parece haber grandes diferencias entre los distintos grupos de edad ni entre los distintos grupos de conocimiento, por lo que podríamos decir que el modo de realizar las tareas está siendo razonablemente evidente para todos los usuarios, independientemente de su edad o conocimientos previos.

A continuación realizaremos un análisis similar para la valoración que dan los usuarios a los distintos aspectos del servicio, en la figura 7.5.

Se puede ver que la valoración del servicio en general es buena, obteniéndose en la mayoría de casos valoraciones de 4 sobre 5. Se valora especialmente bien la comprensibilidad del informe, que es uno de los principales objetivos de éste TFG. Sin embargo, la estética de la aplicación tiene una valoración algo peor. En estos gráficos se puede ver más claranente una tendencia creciente con el ID de usuario, lo que de nuevo sugiere que los cambios realizados entre pruebas han servido para mejorar la usabilidad del servicio y por tanto la metodología escogida ha sido adecuada.

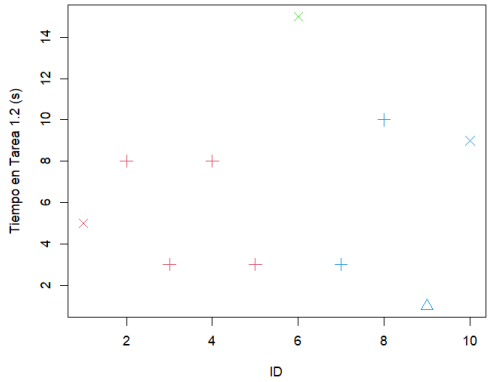
Con respecto a los comentarios realizados en las encuestas, se ha intentado realizar todos los cambios oportunos. Sin embargo, cabe destacar dos: la búsqueda de aplicaciones mediante el nombre comercial y la claridad en la acción de simular la concesión/revocación de permisos. Con respecto a la primera, no se ha podido implementar una búsqueda mediante el nombre comercial al ser ésta una tarea complicada que requiere tiempo, por lo que se deja como línea futura de trabajo. Con respecto a la segunda, se ha intentado resaltar el texto encima del gráfico de barras y animar la primera de las barras para que "parpadee" al mostrar el informe, pero al parecer no se ha logrado hacer más obvia esta acción de cara a los usuarios. A partir de los comentarios y observaciones recogidos, podemos ver que los usuarios no tienen una forma preferida de especificar la aplicación que se quiere buscar: algunos introducen el nombre de paquete que ven en el ejemplo, otros directamente introducen una subcadena de éste y otros se van a Google Play a buscar la URL asociada.

Como conclusión, vamos a comprobar si se han cumplido los requisitos de usabilidad [RNF14] y [RNF18]. El requisito [RNF14] dice que el sistema mostrará los datos de forma comprensible para un usuario no familiarizado con el sistema de permisos de Android, de modo que un 90 % de los usuarios encuestados en las pruebas de aceptación valoren la comprensibilidad de los resultados con una nota de, al menos, 3/5. No hay ninguna valoración menor que 4 para este aspecto, por lo que podemos considerar el requisito como completado. En segundo lugar, el requisito [RNF18] nos dice que al menos el 90 % de los usuarios serán capaces de consultar el impacto de privacidad de una aplicación (especificada mediante nombre de paquete, URL o archivo fuente) en un tiempo

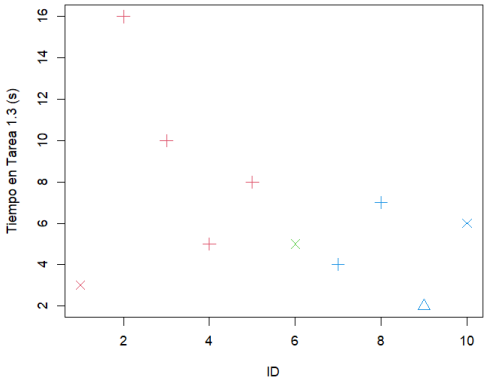


(a)

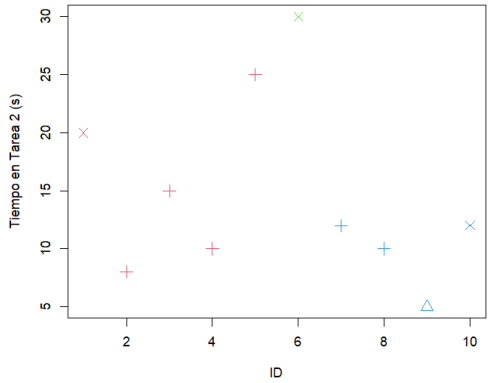
(b)



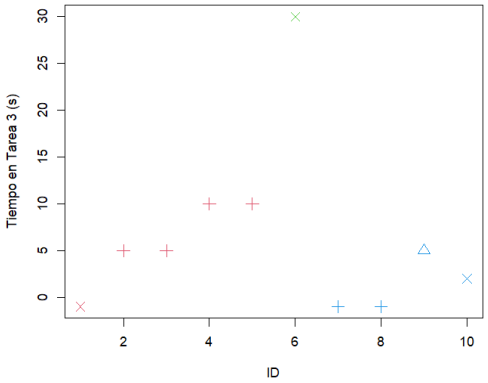
(c)



(d)



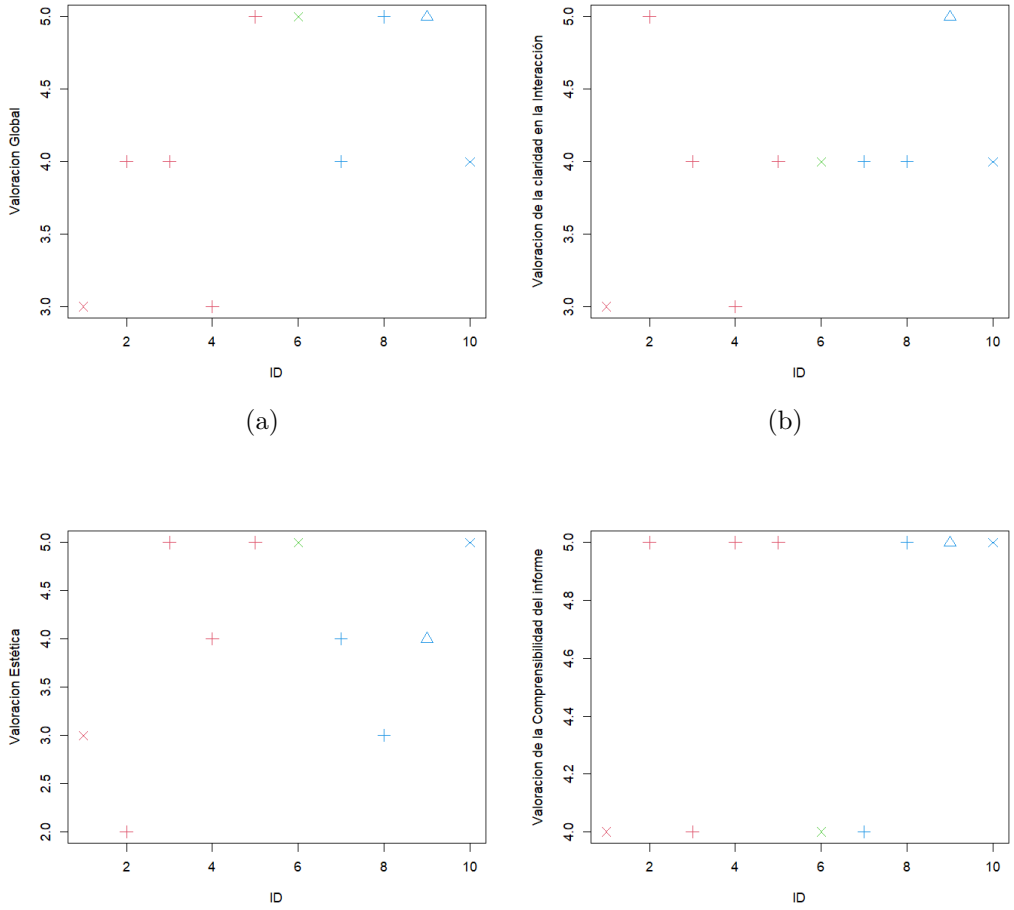
(e)



(f)

Figura 7.4: Tiempo de realización frente a ID de usuario de las tareas a realizar en las pruebas de usabilidad.

inferior a 30s (sin contar tiempos de carga). Esto corresponde a la tarea 1, podemos ver en el gráfico que el tiempo máximo apenas supera los 20s, y solamente ha habido 1 usuario de entre 10 que no ha podido completar la tarea, por lo que podemos dar por cumplido el requisito.



(a) (b)

(c) (d)

Figura 7.5: Valoración (a) global (b) de la interacción (c) de la estética (d) de la comprensibilidad del informe por parte de los usuarios.

Capítulo 8

Seguimiento del proyecto

8.1. Seguimiento de la planificación

Se ha utilizado una metodología incremental para este proyecto, por lo que a continuación se muestra el estado del proyecto a fecha de fin de cada uno de los incrementos.

Incremento 1: - 19/04/2023: Se ha implementado el sistema básico usando APKPure como fuente de datos. Sin embargo, la interfaz gráfica es algo más pobre de lo previsto para una primera versión, aunque se haya conseguido un sistema funcional, pues no se ha podido completar la cabecera y el pie de página. El diseño del sistema se ha demorado casi una semana más de lo previsto, teniendo que quitar algo de tiempo a la implementación.

Incremento 2: - 26/04/2023: Se ha logrado implementar el sistema integrador completo, a falta de la comunicación con el Warehouse de aplicaciones móviles, que todavía no está disponible. Por ello, han sobrado 3 días que se han dedicado a comenzar con el *Incremento 3*.

Incremento 3: - 09/05/2023: Se ha logrado completar el servicio completo sin mayores problemas ni retrasos considerables en las fechas previstas.

Incremento 4: - 05/06/2023: No se han conseguido realizar los tests de usabilidad a tiempo, debido a los cambios en la interfaz que se han realizado entre tests, que han ocupado más tiempo del previsto. A día 5 de junio, sólo se contaba con 6/10 tests, por lo que se ha tenido que emplear otra semana más para finalizarlos y analizar los resultados. En la planificación se había dejado un margen para imprevistos, por lo que se ha aprovechado este margen para completar los tests de usabilidad.

En la figura 8.1 se muestra el diagrama Gantt de seguimiento del proyecto, donde las tareas que se han completado antes de lo previsto se muestran con una banda verde debajo y las que se han retrasado con una banda roja.

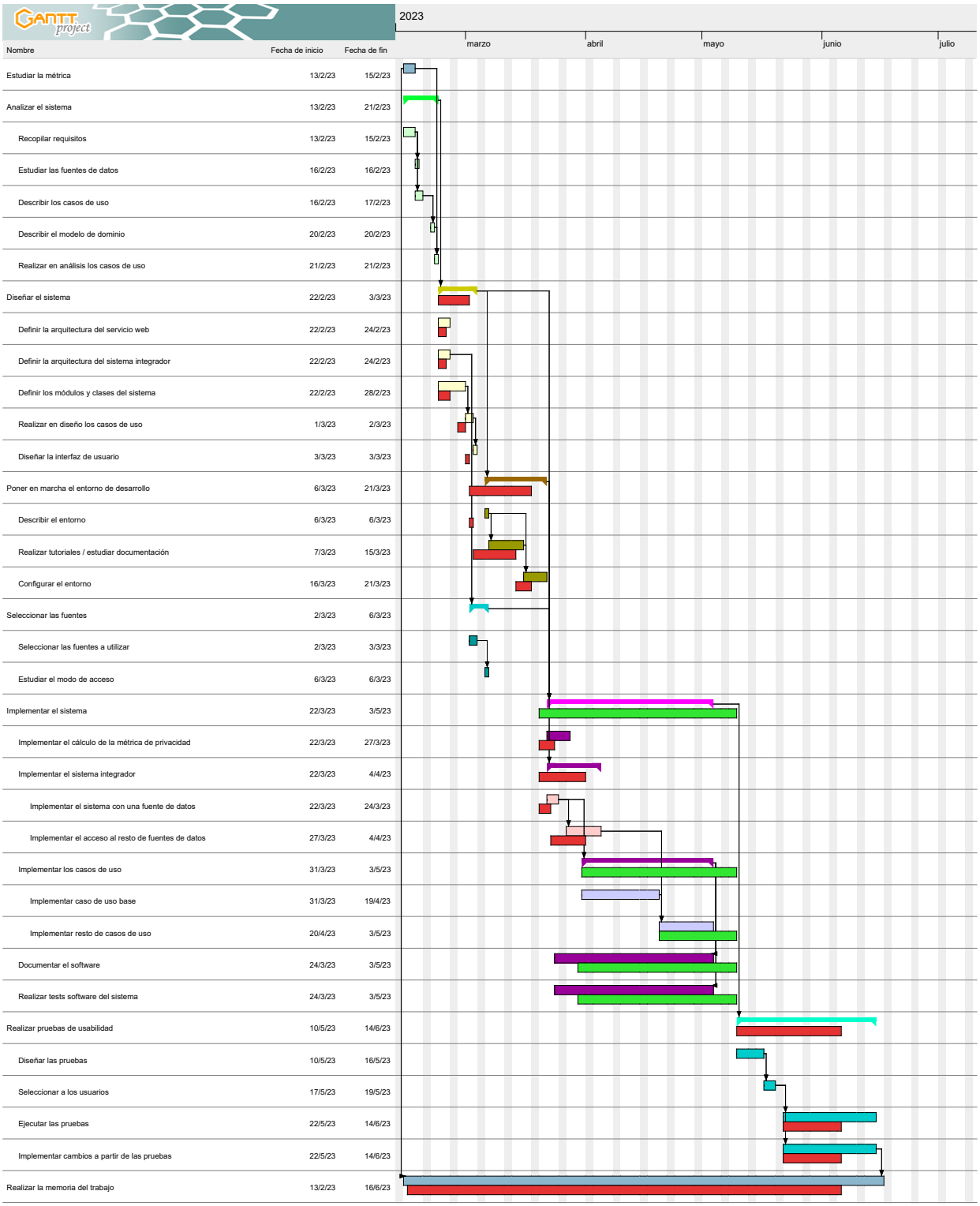


Figura 8.1: Diagrama Gantt de seguimiento del proyecto.

En general se ha cumplido con la planificación establecida, cumpliendo en su mayoría con las fechas previstas en los distintos incrementos. Sin embargo, podemos ver que se ha subestimado el tiempo de diseño y pruebas de usabilidad, mientras que se ha sobreestimado el de implementación. Como lectura podemos extraer la importancia de un buen diseño previo a la implementación, puesto que si éste está lo suficientemente trabajado, la implementación será mucho más sencilla. También podemos extraer como conclusión que es importante realizar una estimación holgada y

con cierto margen de las tareas que tengan que ver con usuarios, puesto que es necesario amoldarse a sus necesidades y el trabajar con usuarios reales es mucho más propenso a retrasos en la planificación.

8.2. Riesgos ocurridos

El primer riesgo que se ha materializado, aunque en pequeña medida, ha sido *R1 - Estimaciones demasiado optimistas del trabajo*. Esto ha ocurrido durante el *Incremento 1* del proyecto (a fecha de finalización dle Hito 2, 01/03/2023), al realizar el diseño del sistema se ha subestimado el tiempo que éste ha llevado. En consecuencia, se ha tenido que dedicar más tiempo, resultando en un sistema algo peor de lo que se esperaba al final del *Incremento 1*. Esto tampoco ha sido demasiado grave y se ha podido completar el trabajo con un poco de esfuerzo extra durante ese incremento.

El segundo riesgo que se ha materializado es el *R8 - No disponibilidad de usuarios para realizar pruebas*, al no tener mucha disponibilidad de usuarios mayores ni usuarios con conocimientos avanzados del sistema de permisos de Android. Este riesgo se ha materializado en el *Incremento 4* del proyecto (a fecha 05/06/2023), por lo que el fin del proyecto se ha tenido que retrasar una semana y se ha tenido que considerar un espectro más amplio de usuarios para cubrir los tests de usabilidad previstos.

Con respecto a los riesgos de seguridad, se ha dado el riesgo *RS3 - El sistema tiene habilitado el módulo de depuración, por lo que se revela información sensible*. Debido a que en la máquina virtual en la que se ha desplegado el servicio no se permite la conexión mediante HTTPS, el servicio no se puede ejecutar sin el módulo de depuración activado, ya que de otro modo solamente se aceptan conexiones a través de HTTPS. Por limitaciones de tiempo no se ha podido realizar ninguna acción para mitigar este riesgo, por lo que simplemente se ha asumido el riesgo.

Capítulo 9

Conclusiones y líneas futuras

9.1. Conclusiones

Se han cumplido los objetivos propuestos, se ha desarrollado un servicio web que permite a los usuarios consultar el riesgo de privacidad asociado a aplicaciones móviles Android y el grado de intrusividad de éstas. Las pruebas realizadas con usuarios han servido para confirmar que se ha cumplido el objetivo de comprensibilidad y usabilidad del sistema. No obstante, han permitido detectar algunas carencias en la interacción con los usuarios que se han podido corregir antes de finalizar este proyecto, mejorando así el servicio. A pesar de estas mejoras, persisten algunas limitaciones que no se han podido abordar (el esfuerzo que requieren desborda el previsto para un TFG) y que se han incluido como posibles mejoras en el Trabajo Futuro.

Como puntos fuertes del trabajo desarrollado, se ha conseguido desarrollar un sistema autónomo y robusto ante caídas en las distintas fuentes de datos que implementa la métrica de privacidad. Además, este sistema está completamente integrado con el Warehouse de aplicaciones móviles desarrollado en el TFG de Alejandro Pérez de la Fuente, de donde se consultan los metadatos de las aplicaciones y se suben nuevas aplicaciones no almacenadas. Otras ventajas de este trabajo son su capacidad para analizar aplicaciones que no están en el Warehouse, para generar informes en formatos interoperables, que se pueden descargar para su posterior tratamiento en sistemas externos, y la facilidad para realizar simulaciones. Estas simulaciones ofrecen la oportunidad de empoderar a los usuarios finales, y de validar la métrica. Esta capacidad de apoyo para la validación convierte este servicio en una herramienta de apoyo para la investigación asociada a la métrica de privacidad.

Aunque los objetivos planteados para este TFG se han cumplido, algunas limitaciones abren la oportunidad de posibles mejoras. No soportar búsquedas de aplicaciones por su nombre comercial es la más importante desde el punto de vista del usuario. Hacerlo requiere un trabajo adicional de desambiguación en casos donde los nombres de las aplicaciones son demasiado similares que excede el esfuerzo de un TFG. Por otro lado, el acceso mediante HTTPS a la máquina virtual donde se aloja el servicio limita su rendimiento. Existen otras opciones con mejor rendimiento, pero no son gratuitas, lo cual limita las garantías de perdurabilidad del servicio a medio plazo.

9.2. Líneas de trabajo futuras

Como trabajo futuro, que no se ha podido completar por limitaciones de tiempo, se plantean las siguientes propuestas:

- **Comparación entre aplicaciones**. Esta es la principal línea de mejora, permitir que

se puedan comparar aplicaciones más en detalle dentro de una misma ventana. Aunque de momento se pueden ver "apps similares" a las que se está consultando, no se puede seleccionar qué apps se muestran como similares o realizar una comparación de los permisos en detalle.

- **Buscador de nombres comerciales**. Una de las principales limitaciones que saltaron a la vista en los tests de aceptación fue que muchas veces el usuario intenta introducir el nombre comercial en vez de el de paquete. Esto se palió parcialmente permitiendo introducir subcadenas del nombre de paquete (como, por ejemplo, "whatsapp" en vez de "com.whatsapp" o "telegram" en vez de "org.telegram.messenger"). Sin embargo hay aplicaciones donde el nombre comercial no se corresponde con una subcadena del nombre de paquete (por ejemplo, la app "TikTok" tiene como nombre de paquete "com.zhiliaoapp.musically"). Una propuesta clara de mejora a futuro sería implementar un pequeño buscador de nombres comerciales, de forma que el usuario no tenga que acceder a Google Play para introducir la URL de la aplicación o encontrar su nombre de paquete.

- **Despliegue y conexión mediante HTTPS**. De momento, el servicio está desplegado en una máquina virtual proporcionada por la Escuela. Esta máquina virtual solamente dispone de posibilidad de conexión mediante HTTP, no dispone de lo necesario para realizar consultas HTTPS ni el uso de DNS para especificar la web mediante el nombre en vez de la dirección IP del Host. Es por ello que el servidor se ha tenido que lanzar en "modo desarrollo", con serias implicaciones sobre la seguridad como el revelar errores internos o parte del código del servicio. El siguiente paso a dar sería realizar un despliegue en un servidor que soporte HTTPS.

- **Realización de más tests de usuario**. Como se comentó en la sección 7.2.3, no se han podido evaluar todos los grupos de usuarios que se plantearon. Es por ello que una línea futura clara de trabajo es realizar estas pruebas. Del mismo modo, 10 usuarios son pocos para obtener conclusiones claras sobre la usabilidad del sistema, por lo que se podrían realizar más pruebas y de otros tipos (por ejemplo, pruebas no supervisadas recogidas a través de un formulario que realiza y evalúa el usuario de forma autónoma).

Bibliografía

[1] Android Open Source Project. AndroidManifest.xml. [Online]. https://github.com/aosp-mirror/platform_frameworks_base/blob/master/core/res/AndroidManifest.xml. Último acceso: 20-04-2023.

[2] A. Aparicio, M. M. M. González, and V. Cardeñoso. Definición de una métrica para conocer el impacto sobre la privacidad de las aplicaciones móviles. Unpublished.

[3] A. Aparicio, M. M. M. González, and V. Cardeñoso. Métrica basada en grupos de permisos para entender el impacto de las aplicaciones android sobre la privacidad. In *2022 17th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–5, 2022.

[4] S. Barth, M. D. de Jong, M. Junger, P. H. Hartel, and J. C. Roppelt. Putting the privacy paradox to the test: Online privacy and security behaviors among users with technical knowledge, privacy awareness, and financial resources. *Telematics and Informatics*, 41:55–69, 2019.

[5] K.-L. CHEN and C.-H. YANG. Design and implementation of privacy impact assessment for android mobile devices. *ZTE Communications*, 14(S0):37–43, 2019.

[6] Desarrolladores de Android. Android para desarrolladores. Recursos y Herramientas. [Online]. https://developer.android.com/. Último acceso: 20-04-2023.

[7] Desarrolladores de Android. API Reference - Manifest.permission-group. [Online]. https://developer.android.com/reference/android/Manifest.permission_group.Último acceso: 20-04-2023.

[8] Desarrolladores de Android. API Reference - Manifest.permission. [Online]. https://developer.android.com/reference/android/Manifest.permission.Último acceso: 20-04-2023.

[9] Desarrolladores de Android. Permisos en Android. [Online]. https://developer.android.com/guide/topics/permissions/overview.Último acceso: 20-04-2023.

[10] A. Doan, A. Ardalan, J. R. Ballard, S. Das, Y. Govind, P. Konda, H. Li, E. Paulson, P. S. G. C., and H. Zhang. Toward a system building agenda for data integration. *CoRR*, abs/1710.00027, 2017.

[11] European Comission. A Europe fit for the digital age. Empowering people with a new generation of technologies. [Online]. https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/europe-fit-digital-age_en. Último acceso: 21-05-2023.

[12] Fahadul Shadhin. The MVT Design Pattern of Django . [Online]. https://python.plainenglish.io/the-mvt-design-pattern-of-django-8fd47c61f582. Último acceso: 27-04-2023.

[13] M. Fowler, D. Rice, M. Foemmel, E. Hieatt, R. Mee, and R. Stafford. *Patterns of Enterprise Application Architecture*. Addison-Wesley Professional, 2002.

[14] B. Hughes and M. Cotterel. *Software Proyect Management*, pages 162–171. McGraw-Hill Education (UK), 5th edition, 2009.

[15] H. Jin, M. Liu, K. Dodhia, Y. Li, G. Srivastava, M. Fredrikson, Y. Agarwal, and J. I. Hong. Why are they collecting my data?: Inferring the purposes of network traffic in mobile apps. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2(4):173:1–173:27, 2018.

[16] A. Khatoon and P. Corcoran. Android permission system and user privacy — a review of concept and approaches. In *2017 IEEE 7th International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*, pages 153–158, 2017.

[17] Y. Liu and A. Simpson. On the trade-off between privacy and utility in mobile services: A qualitative study. page 261–278, Berlin, Heidelberg, 2019. Springer-Verlag.

[18] Python Developers. PEP 8 – Style Guide for Python Code. [Online]. `https://peps.python.org/pep-0008/`. Último acceso: 08-05-2023.

[19] M. S. Saleem, J. Mišić, and V. B. Mišić. Android malware detection using feature ranking of permissions. *arXiv preprint arXiv:2201.08468*, 2022.

[20] M. Upadhayay, A. Sharma, G. Garg, and A. Arora. Rpndroid: Android malware detection using ranked permissions and network traffic. In *2021 Fifth World Conference on Smart Trends in Systems Security and Sustainability (WorldS4)*, pages 19–24, 2021.

[21] Y. Wang, J. Zheng, C. Sun, and S. Mukkamala. Quantitative security risk assessment of android permissions and applications. In L. Wang and B. Shafiq, editors, *Data and Applications Security and Privacy XXVII*, pages 226–241, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

Apéndice A

Manual de usuario

A.1. Búsqueda del informe de privacidad de una aplicación

Para acceder a la ventana principal, tenemos que acceder a la URL `http://<ip>/scorer` mediante un navegador Web, donde se nos mostrará la ventana principal que se ve en las figuras A.1 y A.2.

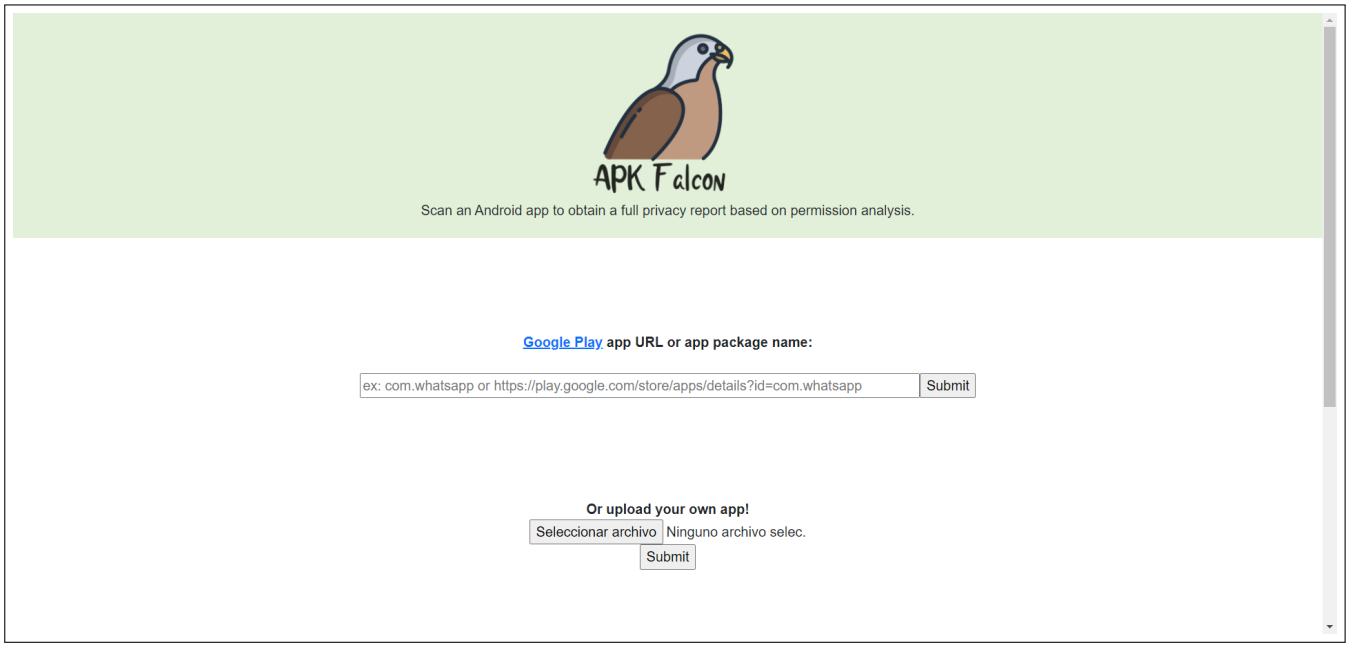


Figura A.1: Ventana principal del servicio web, parte superior.

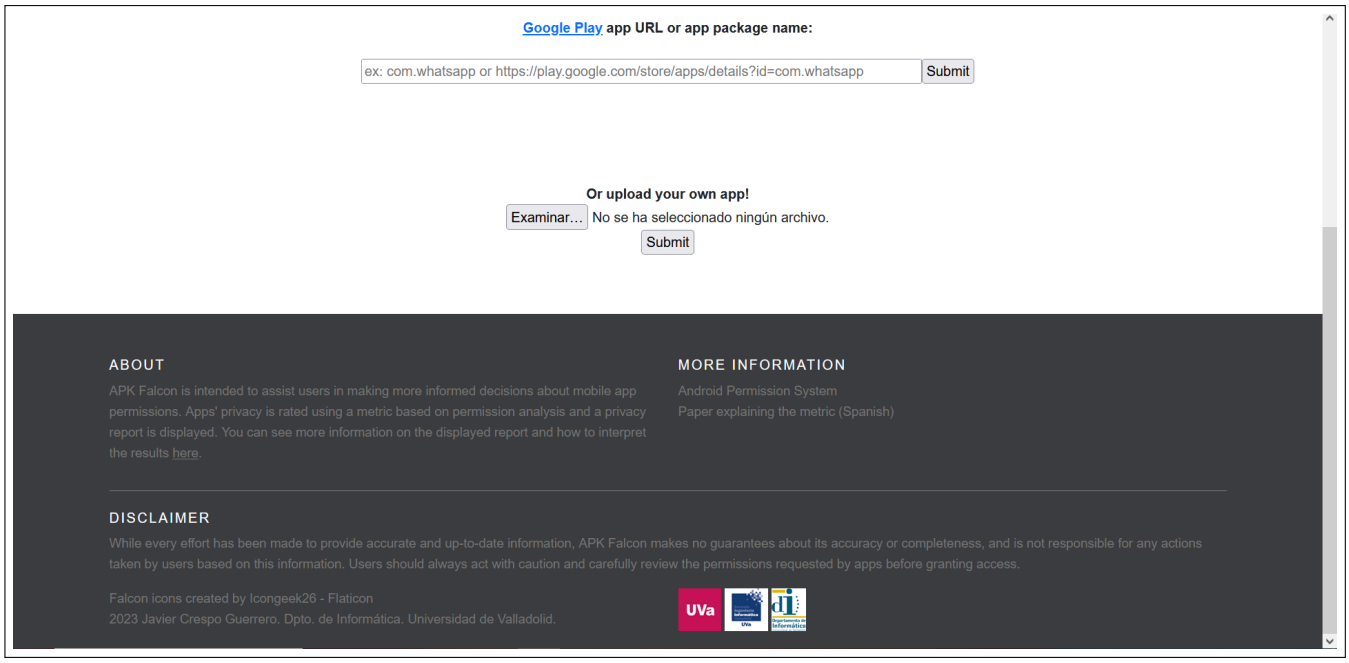


Figura A.2: Ventana principal del servicio web, parte inferior.

En esta ventana podemos especificar la aplicación, bien sea mediante el nombre de paquete, una subcadena de éste o la URL en la barra de búsqueda o bien pulsando en el botón "Seleccionar archivo" y cargando un archivo fuente. Una vez hayamos especificado la aplicación, pulsaremos en el botón "Submit" para ir al informe de esa aplicación.

Para facilitar el especificar una aplicación mediante la URL, se ha incluido un enlace a Google Play en el texto situado encima de la barra de búsqueda, que lleva a la ventana principal de Google Play donde se puede encontrar fácilmente la URL asociada a una aplicación.

A.1.1. Cabecera y pie de página

La cabecera y el pie de página son compartidos por todas las ventanas, cambiando solamente el contenido que se muestra entre ambos. Dentro de la cabecera (ver parte superior de la figura A.1), aparece el logotipo del servicio y un pequeño texto describiéndolo. Si se hace click en el logotipo, nos llevará a la ventana principal.

En el pie de página (ver parte inferior de la figura A.2) aparecen tres secciones:

- Una sección "About" donde se incluye una descripción más detallada del servicio y se resalta un texto que enlaza con la ventana "About", que será descrita posteriormente.
- Una sección "More Information" donde aparecen dos enlaces de interés, uno a la descripción del sistema de permisos de Android [8] y otro al paper [3].
- Una sección "Disclaimer" donde aparece el disclaimer del servicio y debajo se muestra información sobre el desarrollador, la Escuela y la Universidad. Se puede hacer click sobre el texto y los iconos de la Escuela y la Universidad para ir a las páginas web correspondientes.

A.1.2. Errores

En caso de que busquemos una aplicación mediante la barra de búsqueda superior que no exista en Google Play, se nos mostrará un error (ver figura A.3) debajo de la barra de búsqueda.

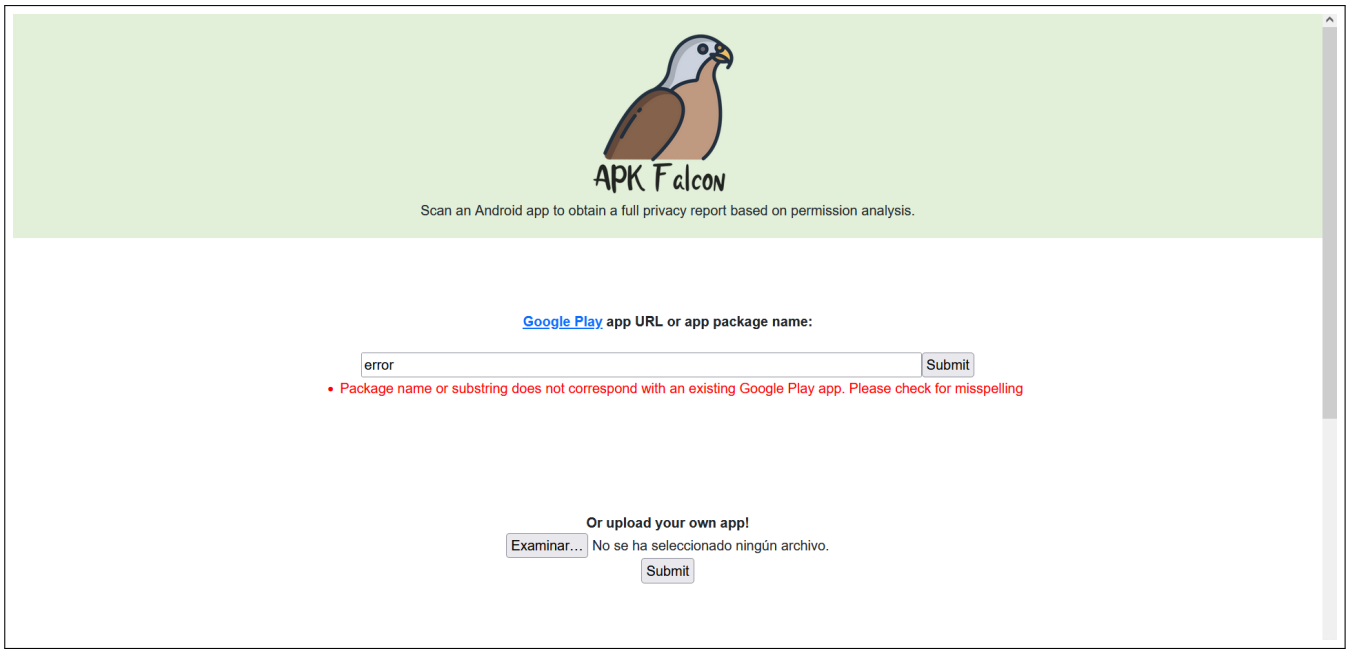


Figura A.3: Ventana principal del servicio web, error de aplicación incorrecta.

En caso de que la descarga de la aplicación falle (por ejemplo, porque la aplicación no se encuentre en ninguna de las fuentes de descarga), se mostrará un error como se ve en la figura A.4.

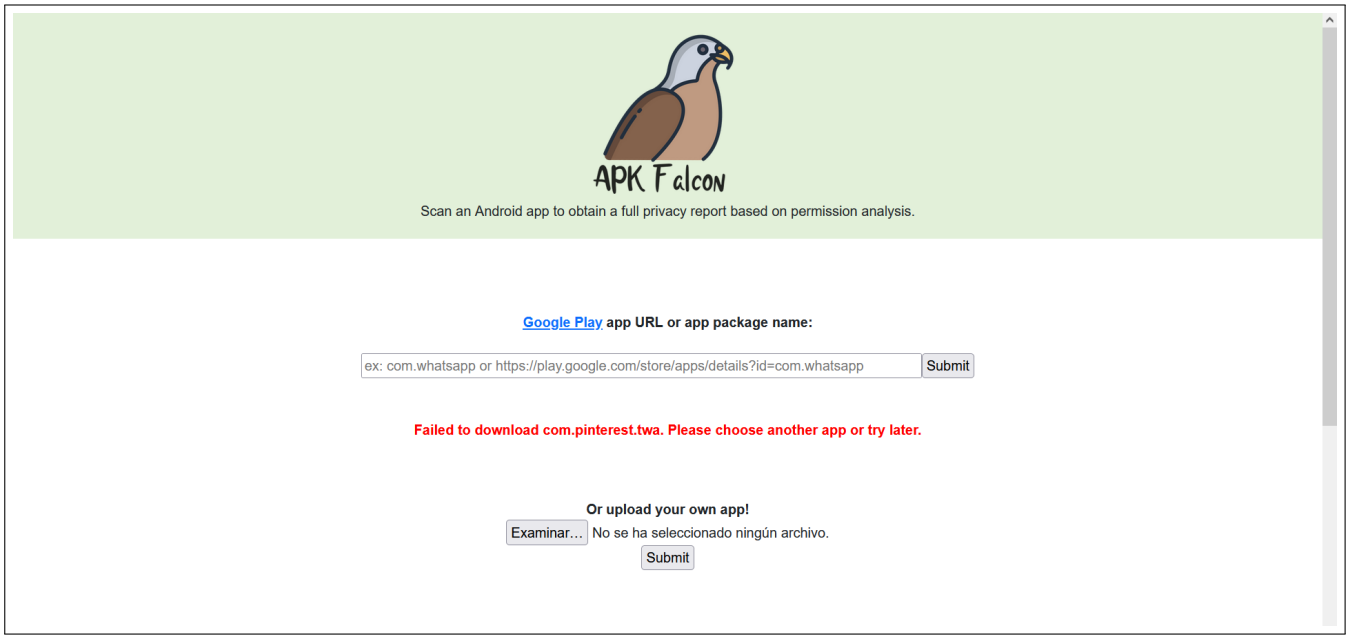


Figura A.4: Ventana principal del servicio web, error de descarga.

En caso de que se suba una aplicación con una extensión incorrecta, se mostrará un mensaje de error como se ve en la figura A.5.

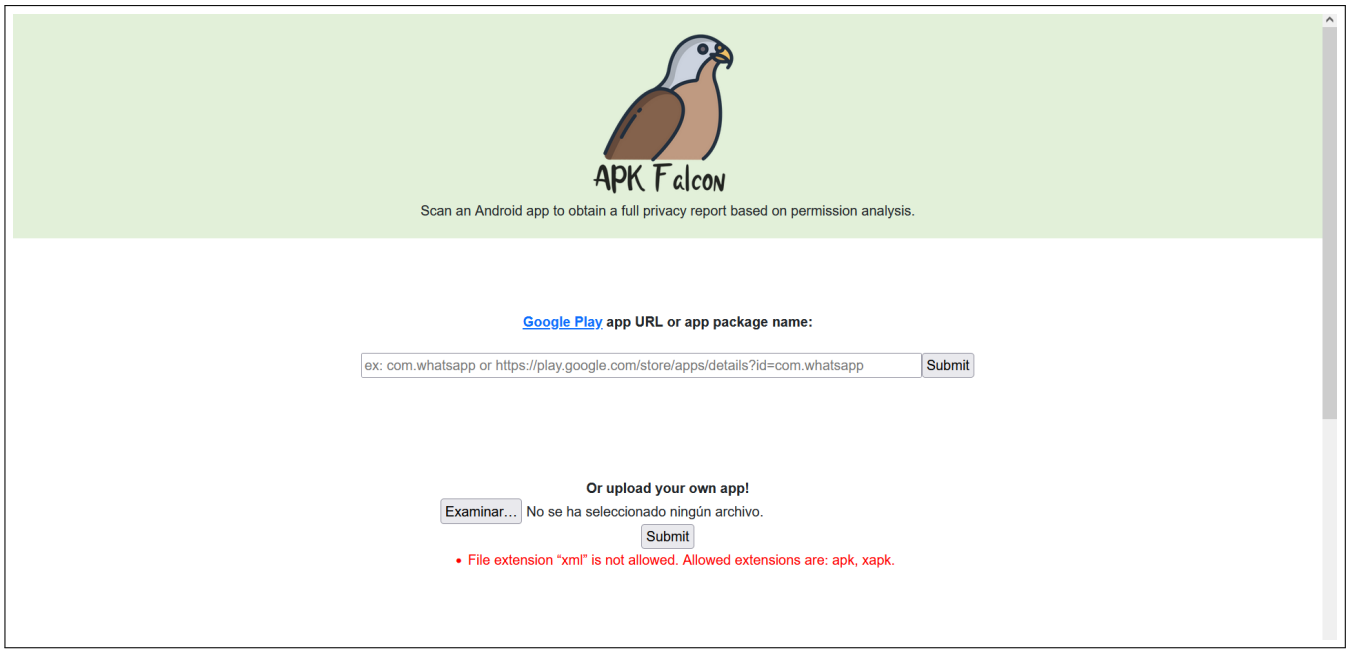


Figura A.5: Ventana principal del servicio web, error de formato de archivo fuente inválido.

En caso de que falle la subida y/o procesamiento de un archivo fuente, se mostrará un mensaje de error como se ve en la figura A.6.

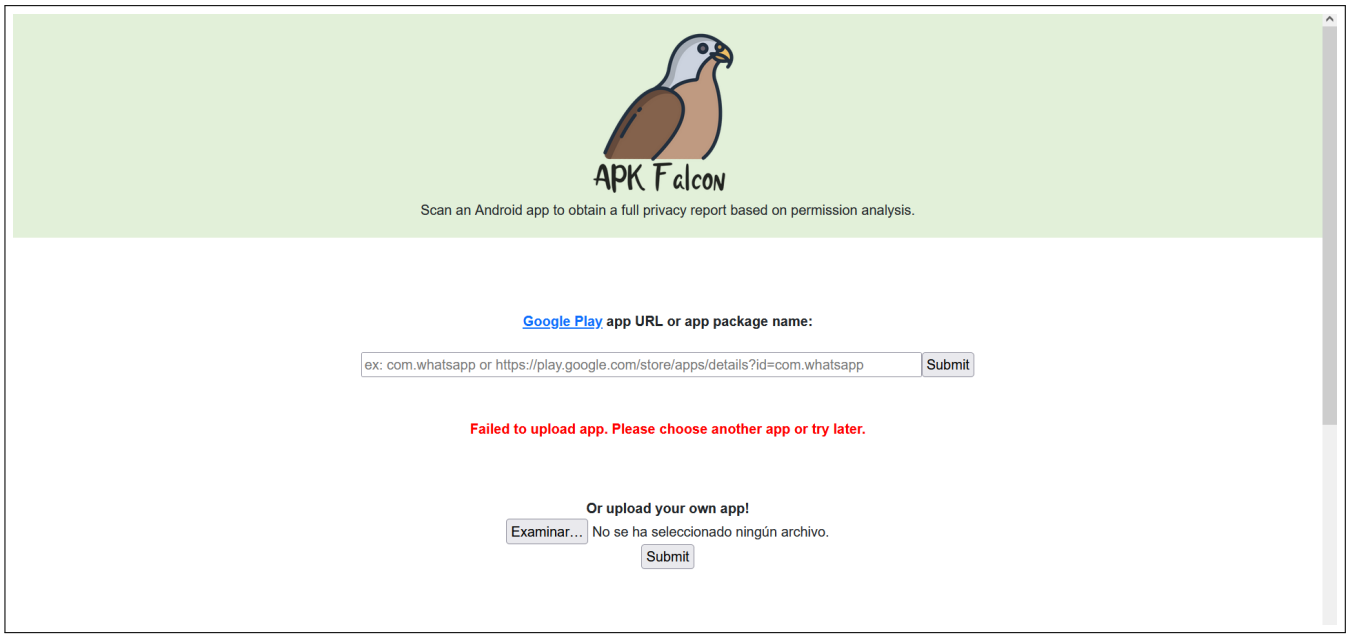


Figura A.6: Ventana principal del servicio web, error en la subida del archivo fuente.

A.2. Ventana de carga

Una vez hayamos buscado o subido una aplicación, se nos mostrará una pantalla de carga como se ve en la figura A.7, donde se nos irán mostrando mensajes que describen el proceso junto con una ruleta de carga.

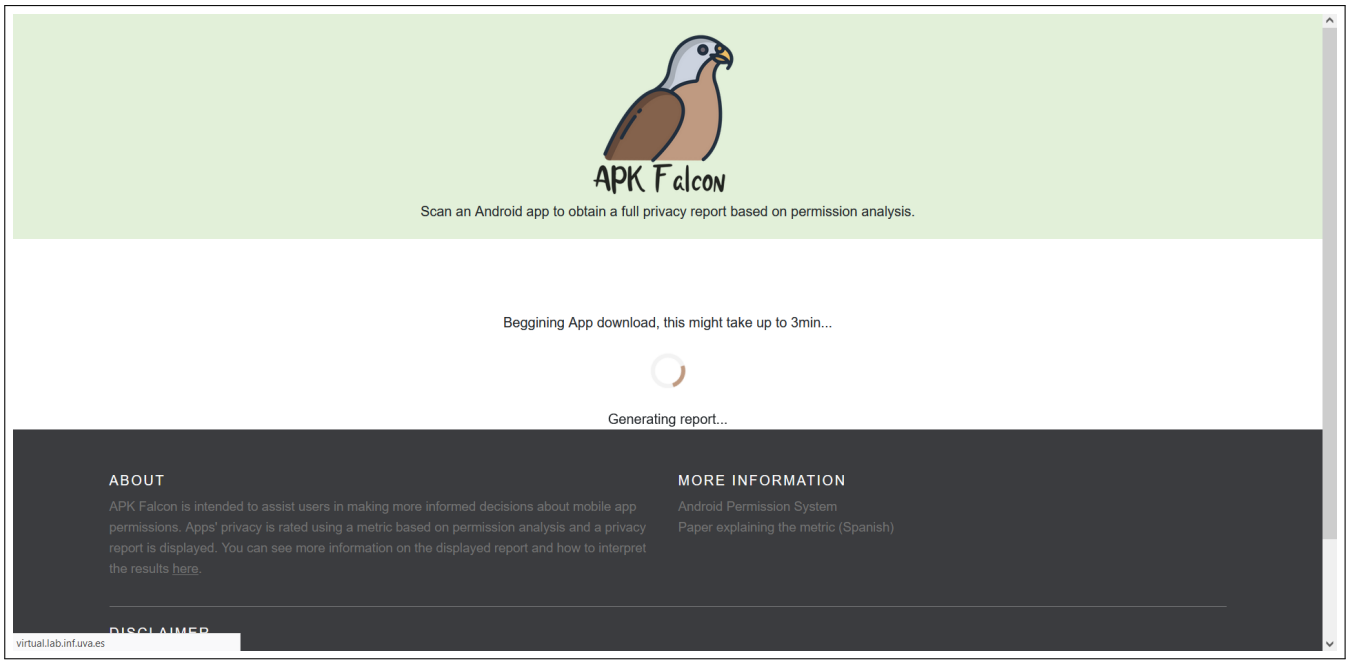


Figura A.7: Ventana de carga del servicio.

A.3. Informe de privacidad

Una vez se ha completado la descarga/subida de la aplicación, se muestra un informe de privacidad como se ve en las figuras A.8 A.9 y A.10.

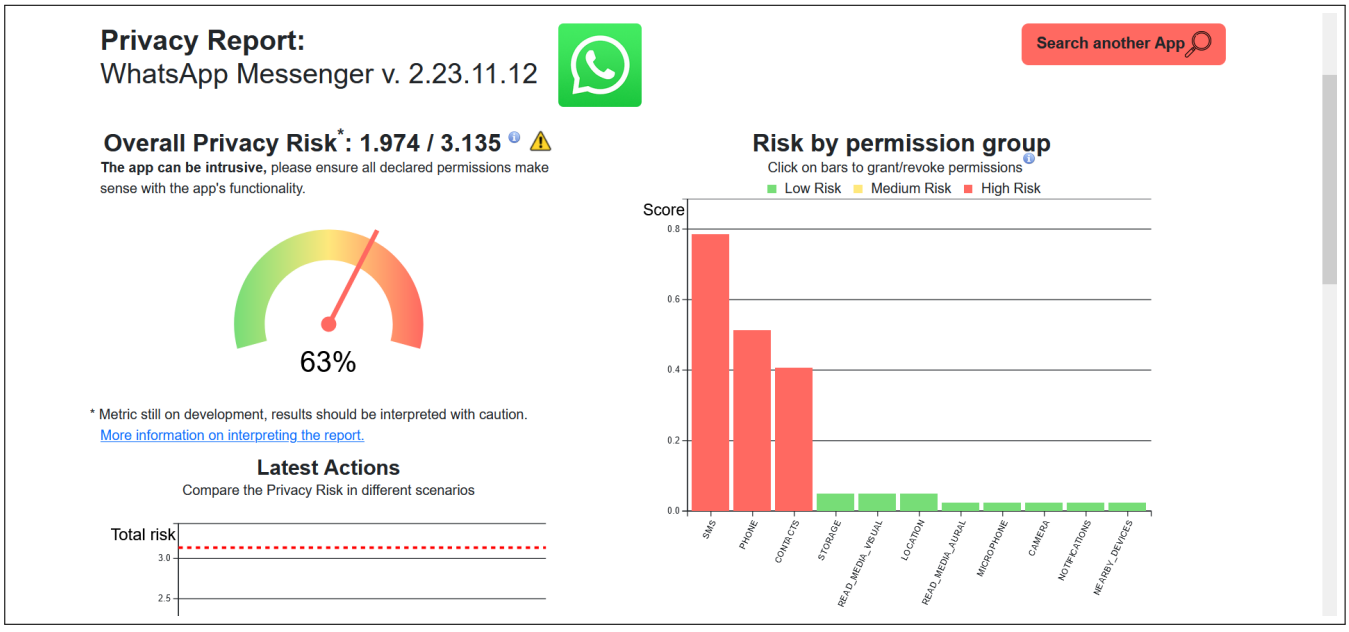


Figura A.8: Ventana de informe de privacidad del servicio, parte superior.

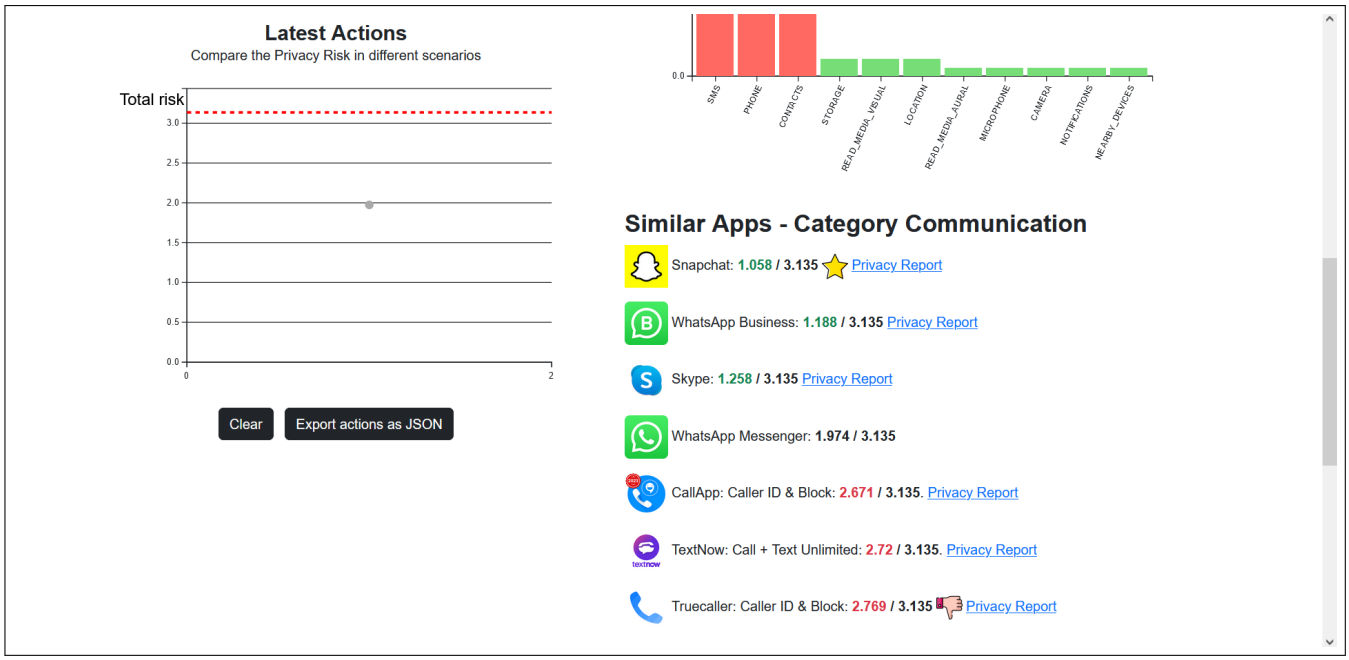


Figura A.9: Ventana de informe de privacidad del servicio, parte media.

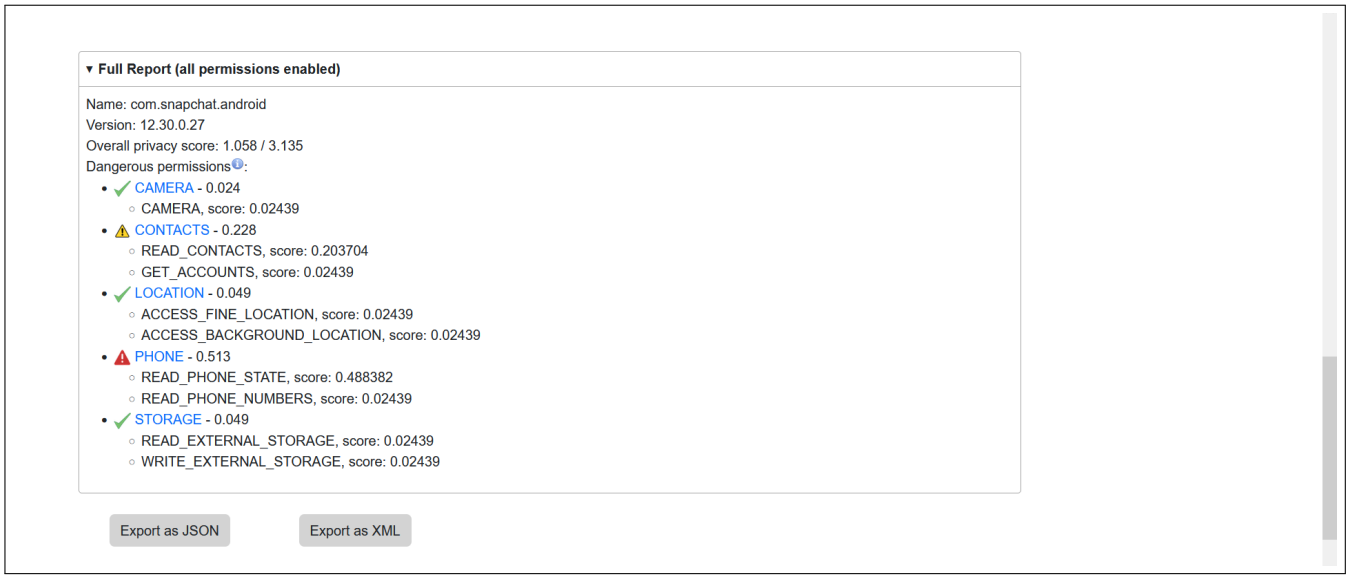


Figura A.10: Ventana de informe de privacidad del servicio, parte inferior.

En primer lugar, en la parte superior, podemos ver el nombre comercial de la aplicación y su logotipo (con un enlace a la página de Google Play), un botón “Search another App” que nos lleva a la ventana principal. Debajo podemos ver la parte gráfica, compuesta por un gráfico de velocímetro arriba a la izquierda, uno de barras a su derecha, uno temporal debajo y una sección de aplicaciones similares.

A.3.1. Gráficos e interactividad

En primer lugar se muestra un informe gráfico de los resultados, mediante un gráfico de velocímetro, uno de barras y otro temporal. En el gráfico de velocímetro se muestra la puntuación de

privacidad junto con una frase que indica el riesgo que esa aplicación supone y debajo un enlace a la ventana "About" para más información sobre cómo interpretar el informe. Se muestra un icono de "tick", "warning" y "danger" en función de si la aplicación tiene asociado un riesgo de privacidad bajo, medio o alto, respectivamente. En el gráfico de velocímetro se muestra una aguja con la puntuación de privacidad en porcentaje.

A la derecha, se muestra un gráfico de barras que indica la puntuación de privacidad asociada a cada grupo de permisos, con las barras coloreadas tal y como se indica en la leyenda que aparece encima. Si se pasa el ratón por encima, se puede ver la descripción del grupo de permisos. Si se hace click, se simula la concesión/revocación del grupo de permisos, que hace que se actualice el gráfico de velocímetro como se muestra en la figura A.11.

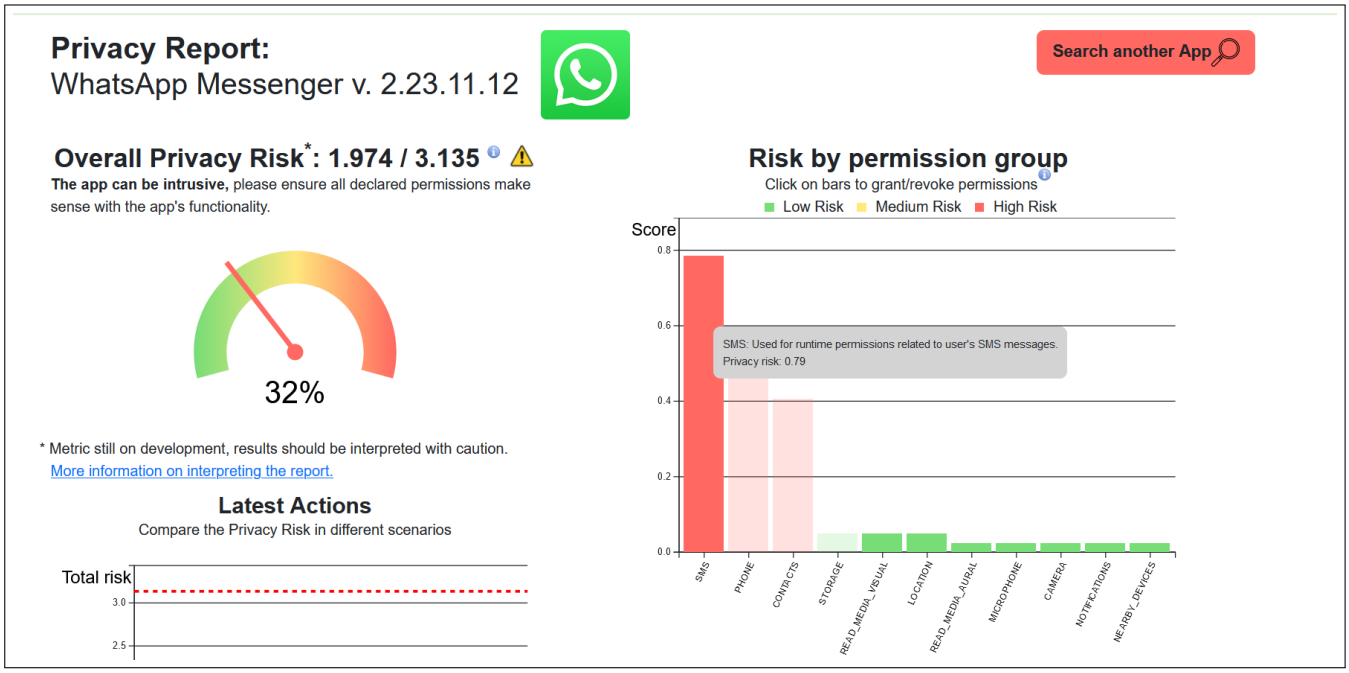


Figura A.11: Ventana de informe de privacidad del servicio, algunos permisos revocados.

Las acciones de simulación de concesión/revocación de permisos que realizamos hacen que se vaya creando un gráfico temporal con las acciones realizadas, que permite comparar distintos escenarios de privacidad como se ve en la figura A.12, la línea roja discontínua nos indica la puntuación máxima de privacidad posible. Si pulsamos el botón "Clear" se reiniciará el gráfico (con la activación/desactivación de permisos que se tenga en el gráfico de barras), si pulsamos el botón "Export actions as JSON" se descargará un archivo JSON con las acciones realizadas y su impacto en la métrica de privacidad.

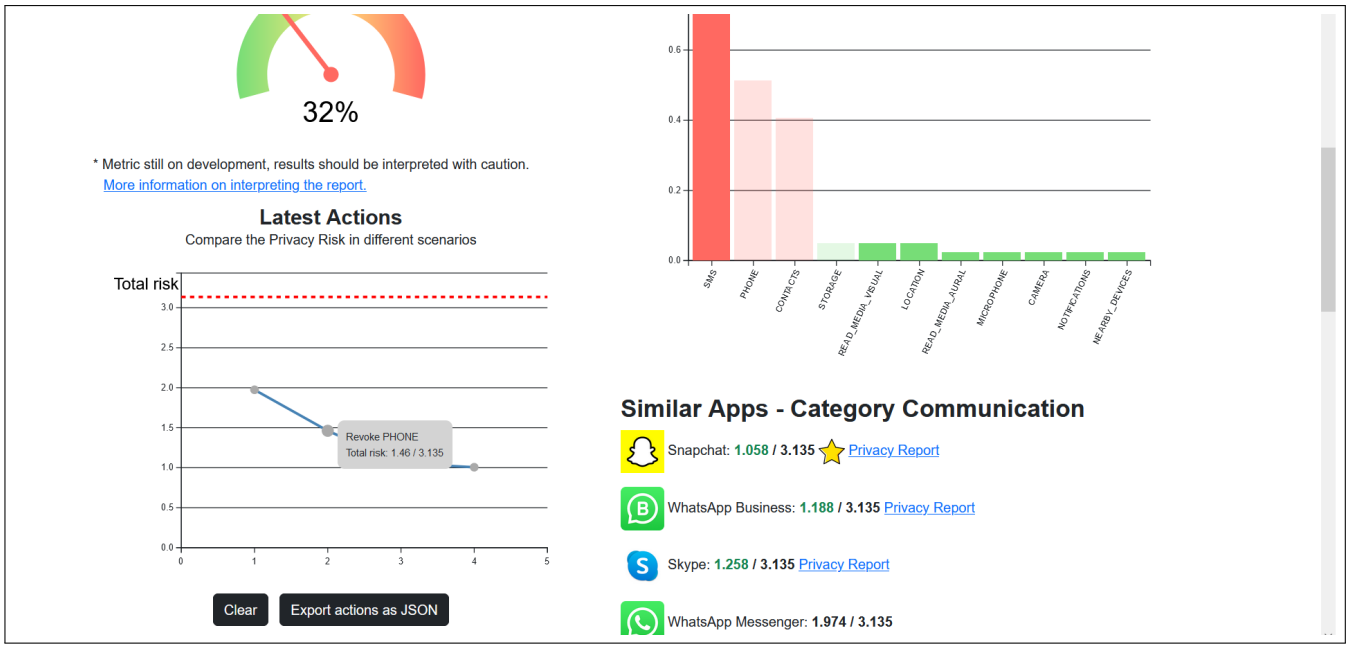


Figura A.12: Ventana de informe de privacidad del servicio, gráfico temporal con algunos permisos revocados.

A.3.2. Aplicaciones similares

Para concluir la parte gráfica, se muestran aplicaciones similares (con la misma categoría) a la que se está buscando, mostrándose los logotipos (enlazados a la correspondiente página de detalles de Google Play) y nombres comerciales de las aplicaciones junto con las puntuaciones de privacidad . en rojo si es peor que la que se está consultado y en verde si es mejor. La primera y última aplicación que se muestra se corresponde con la mejor y la peor aplicación almacenada en el repositorio local para esa categoría.

A.3.3. Informe detallado

Finalmente, en la parte inferior, se muestra el informe de privacidad detallado, mediante un desplegable que detalla el nombre de la aplicacióm, su puntuación de privacidad, los grupos de permisos y permisos que declara la aplicación, junto con los pesos asociados a cada uno. Se utilizan iconos de “tick”, “warning” y “danger” en función de si el grupo tiene asociado un riesgo de privacidad bajo, medio o alto, respectivamente.

Debajo de este informe se muestran los botones para exportarlo en formato `XML` y `JSON`.

A.4. Ventana About

Por último, en la ventana “About” (ver figuras A.13 A.14 y A.15) se puede obtener más información sobre APK Falcon: qué es el servicio, para qué se puede utilizar, etc.

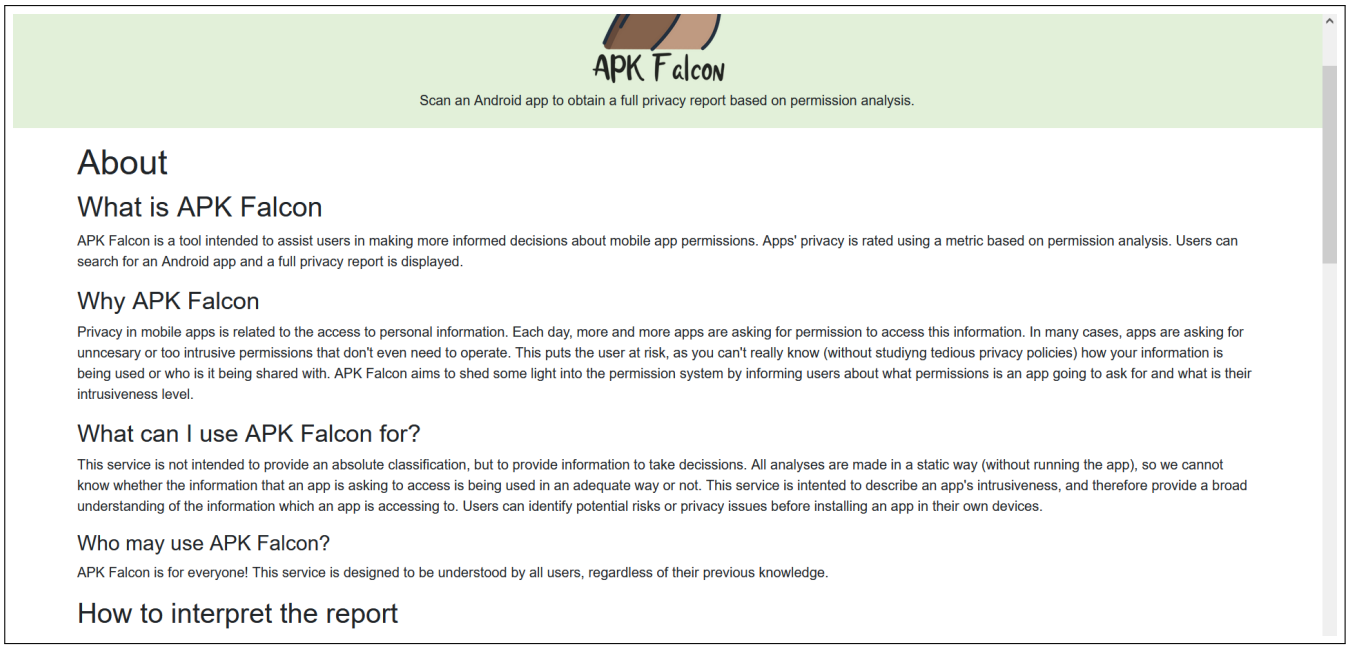


Figura A.13: Ventana de más información, parte superior.

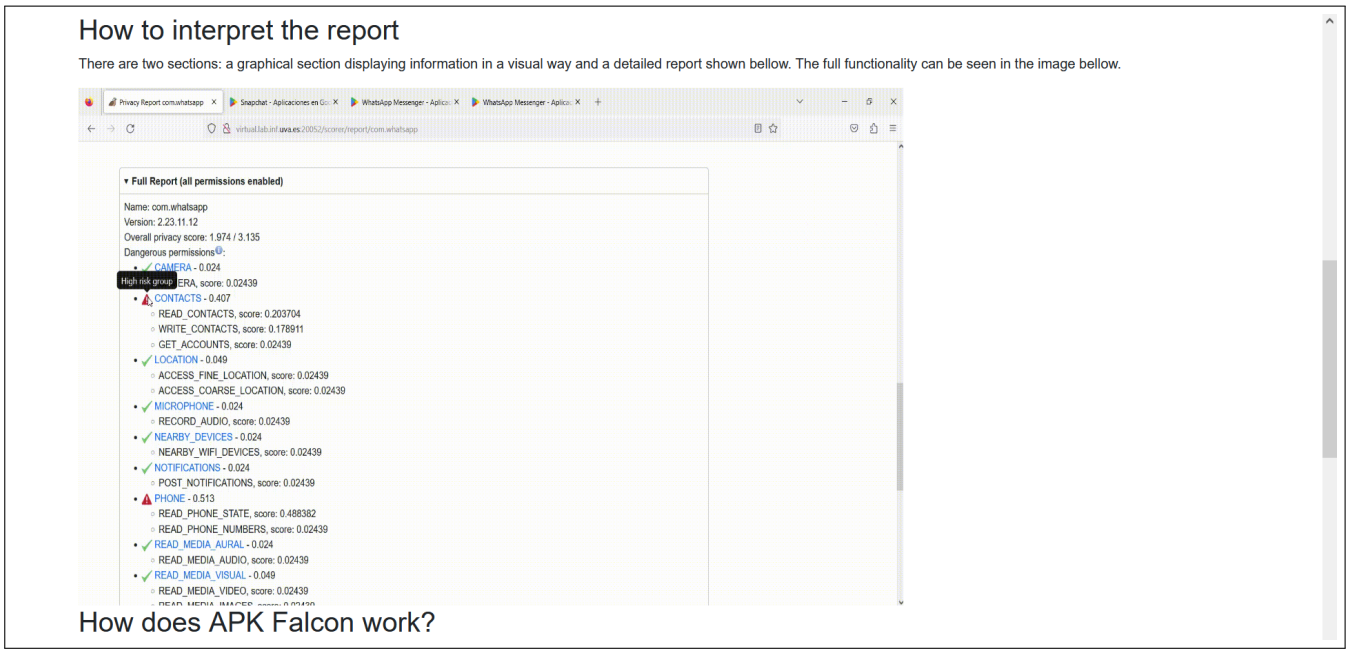


Figura A.14: Ventana de más información, parte media.

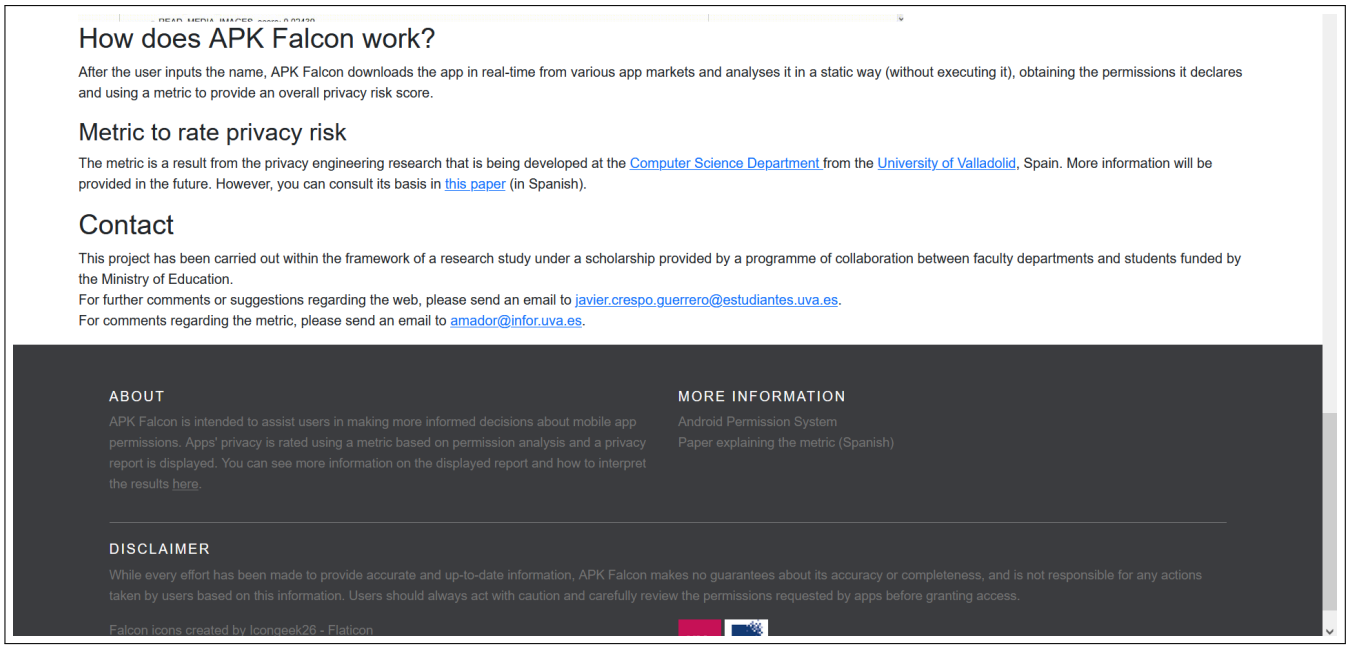


Figura A.15: Ventana de más información, parte inferior.

De entre toda la información que se muestra, cabe destacar la sección "How to interpret the report" (figura A.14), donde se muestra un GIF donde se realiza un pequeño recorrido por toda la funcionalidad y apartados del informe de privacidad. También, dentro de las secciones "Metric to rate privacy risk" y "Contact" (figura A.15) se puede ver más informarción sobre los fundamentos de la métrica de privacidad y la información de contacto.

Apéndice B

Manual de instalación y despliegue

A continuación se detalla el proceso necesario para realizar el despliegue del servicio.

1. El primer paso es descargar el código del servidor, esto se puede hacer mediante Git usando la siguiente orden.

   ```
   git clone https://gitlab.inf.uva.es/javcres/uva-apk-falcon.git
   ```

2. Una vez se tiene el código, se tienen que instalar todas las dependencias de Python (se requiere de Python 3, preferiblemente Python 3.10), para ello se utiliza el gestor de dependencias `poetry` mediante el siguiente código.

   ```
   cd uva_apk_falcon
   pip install poetry
   poetry install
   ```

   Con esto, el gestor de dependencias instalará todas las dependencias en sus versiones necesarias definidas en el archivo `poetry.lock`.

3. A continuación, se deben instalar una serie de programas auxiliares para el procesamiento y descarga de archivos fuente. En primer lugar se debe instalar la herramienta ApkTool. Para ello, en primer lugar instalamos Java mediante la orden:

   ```
   sudo apt install openjdk-11-jre-headless
   ```

4. A continuación, descargamos los archivos "Wrapper script" y el archivo fuente .jar de la URL `https://ibotpeaches.github.io/Apktool/install/`. A continuación movemos los dos archivos al directorio `/usr/local/bin` y cambiamos sus permisos con las órdenes:

   ```
   sudo mv apktool_2.7.0.jar /usr/local/bin/apktool.jar
   sudo mv apktool /usr/local/bin/apktool
   sudo chmod +x /usr/local/bin/apktool.jar
   sudo chmod +x /usr/local/bin/apktool
   ```

   Con todo esto ya tendríamos ApkTool instalado. A continuación necesitamos instalar el comando `unzip` para descomprimir los archivos `.xapk`, lo instalamos con:

```
sudo apt-install unzip
```

5. Por último, tenemos que instalar Firefox, necesario para el web scraping, primero tenemos que instalar Firefox[1]:

   ```
   sudo apt-install firefox
   ```

6. A continuación tenemos que descargar el driver necesario para el scraping, llamado `geckodriver`, lo descargamos de la página `https://github.com/mozilla/geckodriver/releases`. A continuación lo movemos a la carpeta `/usr/bin` y ajustamos los permisos con los siguientes comandos:

   ```
   sudo mv ./geckodriver /usr/bin/geckodriver
   sudo chmod 755 /usr/bin/geckodriver
   ```

7. Ya tenemos todo lo necesario para ejecutar el servidor, pero antes tenemos que añadir unas cuantas variables de entorno en nuestro sistema. Para ello se recomienda crear un archivo llamado `uva_apk_falcon/.env` en el que se definan las siguientes variables de entorno:

   - `APK_FALCON_PATH_TO_REFERENCE_TABLE`: string con el path a la tabla de referencia
   - `APK_FALCON_PATH_TO_DESCRIPTION_TABLE`: string con el path a la tabla de descripciones de los grupos
   - `APK_FALCON_PATH_TO_CATEGORY_TABLE`: string con el path al documento JSON con los nombres comerciales de cada categoría
   - `APK_FALCON_ROUTE_TO_APK_STORAGE_DIR`: string con path al DIRECTORIO donde se van a almacenar los AndroidManifest.xml
   - `APK_FALCON_WAREHOUSE_API_KEY`: string con la clave API para acceder al Warehouse

8. Con todo ello ya podemos ejecutar el servidor, lo podemos hacer con las órdenes:

   ```
   poetry shell
   python manage.py runserver 0.0.0.0:<puerto HTTP>
   ```

   Esto lanzará un proceso con el servidor, donde se mostrará el output por consola. Esto tiene la desventaja de que la terminal desde la que se ha lanzado el proceso no se puede cerrar. Si lo queremos lanzar de forma permanente, podemos utilizar la orden `nohup` como sigue:

   ```
   nohup python manage.py runserver
       0.0.0.0:<puerto HTTP> > server_output.log &
   ```

   Una vez ejecutada esta orden, se lanzará un proceso permanente y los logs serán redirigidos al archivo `server_output.log`, de forma que ahora podemos cerrar la consola y el servidor seguirá ejecutándose.

[1]Dependiendo de la versión de Linux, la instalación predeterminada de Firefox puede dar problemas, consultar `https://number1.co.za/ubuntu-22-04-firefox-selenium-geckodriver/` en caso de fallos del web scraping en tiempo de ejecución.

Apéndice C

Enlaces Adicionales

Los enlaces de interés en este TFG son:

Repositorio con el código completo del servicio y la documentación de diseño: `https://gitlab.inf.uva.es/javcres/uva-apk-falcon`

Servicio web APK Falcon desplegado: `http://virtual.lab.inf.uva.es:20052/scorer/`

Apéndice D

Resultados escaneados de las pruebas de usabilidad

A continuación se muestran los documentos de evaluación de usabilidad recogidos para cada uno de los 10 usuarios evaluados.

Evaluación de usabilidad: APK Falcon

ID usuario	1	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
			X				X		

Tarea a realizar	1	Resultado	OK.	Tiempo empleado	19s.
Observaciones	Quitar demasiado obvio.				
Tarea a realizar	1.1	Resultado	OK	Tiempo empleado	7s.
Observaciones					
Tarea a realizar	1.2	Resultado	OK	Tiempo empleado	5s.
Observaciones					
Tarea a realizar	1.3	Resultado	OK	Tiempo empleado	≃ 3s.
Observaciones	Primera impresión confusa.				
Tarea a realizar	2	Resultado	OK	Tiempo empleado	20s
Observaciones	Bien volver a la principal				
Tarea a realizar	3	Resultado	X	Tiempo empleado	X
Observaciones	Confunde con more information. No queda claro que hay una pest about.				

Valoración del sistema			2	3	4	5
	Global			X		
	Claridad en la interacción			X	X	
	Valoración estética			X		
	Comprensibilidad del informe				X	

Opinión y aspectos de mejora

- Función de latest actions no se entiende.
- Poner un link a Google Play en pestaña ppal, o un pantallazo.
- Quitar "android.permission." delante.
- Acción de pasar ratón por encima del informe no es obvia.

Poner iconos de report a la izda.

Evaluación de usabilidad: APK Falcon

ID usuario	2	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
			X					X	

Tarea a realizar	1	Resultado	OK	Tiempo empleado	14s.
Observaciones	Extraña con. whatsapp => mejor war.				
Tarea a realizar	1.1	Resultado	OK	Tiempo empleado	14s (10 carga)
Observaciones					
Tarea a realizar	1.2	Resultado	OK	Tiempo empleado	8s
Observaciones					
Tarea a realizar	1.3	Resultado	OK.	Tiempo empleado	16s.
Observaciones	Hace click en more info. on interpr.				
Tarea a realizar	2	Resultado	OK	Tiempo empleado	8s + carga.
Observaciones					
Tarea a realizar	3	Resultado	OK.	Tiempo empleado	5s.
Observaciones					

Valoración del sistema			2	3	4	5
	Global				X	X
	Claridad en la interacción					X
	Valoración estética		X			
	Comprensibilidad del informe					X

Opinión y aspectos de mejora	
	Poner límite de subida en tamaño Intenta buscar "tiktok" en vez de nombre de paquete. Usuario introduce URL de Google Search, no detalles. ~~le ha costado ver~~ Dice que demasiada info a la vez en report. Estaría bien explicar lo que es un nombre de paquete.

Evaluación de usabilidad: APK Falcon

ID usuario	3	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
			X					X	

Tarea a realizar	1	Resultado	OK.	Tiempo empleado	5s + carga.
Observaciones					
Tarea a realizar	1.1	Resultado	OK.	Tiempo empleado	3s
Observaciones					
Tarea a realizar	1.2	Resultado	OK.	Tiempo empleado	3s.
Observaciones					
Tarea a realizar	1.3	Resultado	OK.	Tiempo empleado	10s.
Observaciones					
Tarea a realizar	2	Resultado	OK	Tiempo empleado	15s + carga.
Observaciones					
Tarea a realizar	3	Resultado	OK	Tiempo empleado	5s.
Observaciones					

Valoración del sistema		2	3	4	5
	Global			X	
	Claridad en la interacción			X	
	Valoración estética				X
	Comprensibilidad del informe			X	

Opinión y aspectos de mejora

Dice que sería mejor Submit. auto.
Más cambiar en la interfaz al seleccionar archivo.
No la queda claro que se puede poner la url.
Quizás el resumen de privacidad debería estar mas claro (NEGRITA)
y logo al lado de frase

Evaluación de usabilidad: APK Falcon

ID usuario	4	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
			X					X	

Tarea a realizar	1	Resultado	NO.	Tiempo empleado	20s.
Observaciones	Pone whatsapp de primeras.				
Tarea a realizar	1.1	Resultado	OK.	Tiempo empleado	130s !!!
Observaciones	Dice que se ve poco!				
Tarea a realizar	1.2	Resultado	OK	Tiempo empleado	8s.
Observaciones					
Tarea a realizar	1.3	Resultado	OK	Tiempo empleado	5s.
Observaciones					
Tarea a realizar	2	Resultado	OK.	Tiempo empleado	10s + carga
Observaciones					
Tarea a realizar	3	Resultado	OK.	Tiempo empleado	80s.
Observaciones					

Valoración del sistema		2	3	4	5
	Global		X		
	Claridad en la interacción		X		
	Valoración estética			X	
	Comprensibilidad del informe				X

Opinión y aspectos de mejora

Cree que similar apps mejor debajo del todo. ← Es + curiosidad que funcionalidad.
Usar full report para cambiar.
Aclaración "Click on bars" muy escondida.
No entiende línea roja.
Pondría estrella y 4.3 al final.
Aclarar qué significa en 3.135.

Evaluación de usabilidad: APK Falcon

ID usuario	5	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
			X					X	

Tarea a realizar	1	Resultado	OK	Tiempo empleado	19s + carga.
Observaciones	No está del todo seguro.				
Tarea a realizar	1.1	Resultado	OK.	Tiempo empleado	33s.
Observaciones					
Tarea a realizar	1.2	Resultado	OK.	Tiempo empleado	3s.
Observaciones					
Tarea a realizar	1.3	Resultado	OK.	Tiempo empleado	8s.
Observaciones					
Tarea a realizar	2	Resultado	OK.	Tiempo empleado	25s + carga.
Observaciones	Va para atrás en vez de a botón.				
Tarea a realizar	3	Resultado	OK.	Tiempo empleado	10s.
Observaciones					

Valoración del sistema		2	3	4	5
	Global	.			X
	Claridad en la interacción			X	
	Valoración estética				X
	Comprensibilidad del informe				X

Opinión y aspectos de mejora	No es obvio que barras son interactivas. Tick verde ~~gaiva~~ confuso en caso de lavarna → Sólo el rojo para peligrosas No usar Rojo para Search Another + Poner ícono lupa al lado.

Evaluación de usabilidad: APK Falcon

ID usuario	6	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
					X		X		

Tarea a realizar	1	Resultado	OK	Tiempo empleado	6 + carga.
Observaciones					
Tarea a realizar	1.1	Resultado	NO	Tiempo empleado	+ 1 min
Observaciones	Mora en full report Pulsa				
Tarea a realizar	1.2	Resultado	≃ OK.	Tiempo empleado	15s.
Observaciones	Pulsa en export es JSON				
Tarea a realizar	1.3	Resultado	OK.	Tiempo empleado	5s.
Observaciones	Da para atrás				
Tarea a realizar	2	Resultado	OK.	Tiempo empleado	30s + carga.
Observaciones	Da para atrás				
Tarea a realizar	3	Resultado	OK	Tiempo empleado	30s.
Observaciones	Pulsa primero en More Info, luego se da cuenta que no				

Valoración del sistema		2	3	4	5
	Global				X
	Claridad en la interacción			X	
	Valoración estética				X
	Comprensibilidad del informe			X	

Opinión y aspectos de mejora	No le queda claro que haya que meter la URL. Gráfico latest actions dice que no le ve mucho interés.

Evaluación de usabilidad: APK Falcon

ID usuario	7	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
				X				X	

Tarea a realizar	1	Resultado	OK	Tiempo empleado	9s + carga.
Observaciones					
Tarea a realizar	1.1	Resultado	OK	Tiempo empleado	5s
Observaciones	No vio barra que parpadea.				
Tarea a realizar	1.2	Resultado	OK	Tiempo empleado	3s
Observaciones					
Tarea a realizar	1.3	Resultado	OK	Tiempo empleado	4s.
Observaciones					
Tarea a realizar	2	Resultado	OK	Tiempo empleado	12s.
Observaciones	Pulsa Search Another App.				
Tarea a realizar	3	Resultado	NO	Tiempo empleado	X
Observaciones	Ha pulsado en el resto de links salvo en about				

Valoración del sistema		1	2	3	4	5
	Global				X	
	Claridad en la interacción				X	
	Valoración estética				X	
	Comprensibilidad del informe				X	

Opinión y aspectos de mejora	En Firefox se ve logo y search pegados arriba. Borde circular en logo de App.

Evaluación de usabilidad: APK Falcon

ID usuario	8	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
				X				X	

Tarea a realizar	1	Resultado	OK	Tiempo empleado	30s + carpa
Observaciones					
Tarea a realizar	1.1	Resultado	OK	Tiempo empleado	5s.
Observaciones					
Tarea a realizar	1.2	Resultado	OK.	Tiempo empleado	9,5s
Observaciones					
Tarea a realizar	1.3	Resultado	OK.	Tiempo empleado	7s.
Observaciones					
Tarea a realizar	2	Resultado	OK.	Tiempo empleado	10s.
Observaciones					
Tarea a realizar	3	Resultado	NO	Tiempo empleado	X
Observaciones	Pulsa en resto links.				

Valoración del sistema			2	3	4	5
	Global					X
	Claridad en la interacción				X	
	Valoración estética			X		
	Comprensibilidad del informe					X

Opinión y aspectos de mejora	Here no es obvio, cambiar color. Pte edo pap. no le gusta el texto gris → última tarea (le ~~gr~~ + contraste)

Evaluación de usabilidad: APK Falcon

ID usuario	9	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
				X					X

Tarea a realizar	1	Resultado	OK	Tiempo empleado	10s + carga
Observaciones	De primeras				
Tarea a realizar	1.1	Resultado	OK..	Tiempo empleado	1s.
Observaciones					
Tarea a realizar	1.2	Resultado	OK	Tiempo empleado	1s.
Observaciones					
Tarea a realizar	1.3	Resultado	OK	Tiempo empleado	2s.
Observaciones					
Tarea a realizar	2	Resultado	OK	Tiempo empleado	5s + carga.
Observaciones					
Tarea a realizar	3	Resultado	OK.	Tiempo empleado	5s.
Observaciones					

Valoración del sistema		2	3	4	5
	Global				X
	Claridad en la interacción				X
	Valoración estética			X	
	Comprensibilidad del informe				X

Opinión y aspectos de mejora	Mejor que JSON y XML se descarguen en vez de nueva pestaña. Indicar que se puede arrastrar.

Evaluación de usabilidad: APK Falcon

ID usuario	10	Rango de edad	Joven	Mediana	Mayor	Conocimientos previos	Inexperto	Curioso	Avanzado
				X			X		

Tarea a realizar	1	Resultado	OK.	Tiempo empleado	22s + carga.
Observaciones	Va directo a Google Play				
Tarea a realizar	1.1	Resultado	OK.	Tiempo empleado	7s
Observaciones	Ha visto la animación.				
Tarea a realizar	1.2	Resultado	OK.	Tiempo empleado	9s.
Observaciones					
Tarea a realizar	1.3	Resultado	OK.	Tiempo empleado	6s.
Observaciones					
Tarea a realizar	2	Resultado	OK.	Tiempo empleado	32s + carga.
Observaciones	Pulsa en Icono, no Search Another.				
Tarea a realizar	3	Resultado	OK.	Tiempo empleado	2s.
Observaciones					

Valoración del sistema			2	3	4	5
	Global				X	
	Claridad en la interacción				X	
	Valoración estética					X
	Comprensibilidad del informe					X
Opinión y aspectos de mejora	No le gusta que no se descargue el archivo, sino que se abre en nueva pestaña. Preferiría no dar a Submit tras subir APK.					