UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA DE LA COMPUTACIÓN

DESARROLLO DE UNA APLICACIÓN MÓVIL DE CONTROL Y REPORTES DE GASTOS

Por:
SUSANA CHARARA CHARARA

Realizado con la asesoría de:
Tutor Académico: PROF. XIOMARA CONTRERAS
Tutor Industrial: LIC. LUIS AUGUSTO PEÑA PEREIRA

INFORME DE PASANTÍA LARGA
Presentado ante la Ilustre Universidad Simón Bolívar
como requisito parcial para optar al título de
Ingeniero en Computación

Sartenejas, OCTUBRE de 2016

UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA DE LA COMPUTACIÓN

DESARROLLO DE UNA APLICACIÓN MÓVIL DE CONTROL Y REPORTES DE GASTOS

Por:
SUSANA CHARARA CHARARA

Realizado con la asesoría de:
Tutor Académico: PROF. XIOMARA CONTRERAS
Tutor Industrial: LIC. LUIS AUGUSTO PEÑA PEREIRA

INFORME DE PASANTÍA LARGA
Presentado ante la Ilustre Universidad Simón Bolívar
como requisito parcial para optar al título de
Ingeniero en Computación

Sartenejas, OCTUBRE de 2016

Resumen

En esta pasantía se desarrolló, para la empresa Digitalica Group, C.A., una aplicación para dispositivos móviles con la cual se pueden realizar reportes de gastos. Actualmente, la empresa Digitalica Group, C.A. ofrece a sus trabajadores el beneficio de realizar reembolsos de gastos en ciertos rubros; el proyecto nació como solución al problema de agilizar el proceso en que los trabajadores hacen llegar a los supervisores la información de estos gastos. Con la aplicación desarrollada, se pueden registrar gastos e ingresos con un monto, fecha y descripción. Se permite también capturar y guardar fotos tanto de los ingresos como de los gastos, de manera que el usuario puede tomar fotos de los recibos de sus gastos. Además, estos gastos/ingresos pueden estar asociados a categorías, que indican el rubro al que pertenecen. La aplicación permite al usuario generar archivos con formato PDF que contienen un reporte con una lista de gastos, en un rango de fecha determinado; estos reportes se pueden enviar a un servidor web. Este servidor también fue desarrollado durante la pasantía. Se creó un servicio web que recibe credenciales de un usuario y las valida; éste es utilizado por la aplicación móvil para autenticar usuarios. Asimismo, se desarrolló un servicio web que recibe archivos PDF con reportes de gastos, el cual es utilizado por la aplicación para enviar reportes. Igualmente, se desarrolló una aplicación web, con la cual los supervisores de la empresa pueden revisar y descargar los archivos de los reportes recibidos por el servidor, aprobarlos y rechazarlos.

En este informe se describen los conceptos teóricos que ayudaron al diseño y desarrollo de la solución. Igualmente, se describe el proceso de desarrollo de los tres componentes principales desarrollados durante la pasantía: aplicación nativa para Android, servidor web y aplicación web. El desarrollo de estos involucró el diseño e implementación de los modelos de datos de la aplicación y el servidor, así como múltiples interfaces de usuario que permiten la interacción con los modelos.

Utilizando Scrum como marco metodológico, se implementaron los componentes mencionados anteriormente. Se empleó el lenguaje de programación Java para el desarrollo tanto de la aplicación móvil como del servidor; para este último, se utilizaron las herramientas AppFuse, Spring, Hibernate y Tapestry.

Índice general

Índice de cuadros

Índice de figuras

Lista de Símbolos y Abreviaturas

API: Application Programming Interface (en español Interfaz de Programación de Aplicaciones)

ART: Android Runtime (en español Entorno de Ejecución Android)

CEO: Chief Executive Officer (en español Director Ejecutivo)

COO: Chief Operations Officer (en español Director de Operaciones)

CTO: Chief Technology Officer (en español Director de Tecnología)

DAO: Data Access Object (en español Objeto de Acceso a Datos)

DVCS: Distributed Version Control System (en español Sistema de Control de Versiones Distribuido)

HAL: Hardware Abstraction Layer (en español Capa de Abstracción de Hardware)

HTML: Hypertext Markup Language (en español Lenguaje de Marcas de Hipertexto)

HTTP: Hypertext Transfer Protocol (en español Protocolo de Transferencia de Hipertexto)

IDE: Integrated Development Environment (en español Entorno de Desarrollo Integrado)

JSON: JavaScript Object Notation (en español Notación de Objetos de JavaScript)

JVM: Java Virtual Machine(en español Máquina Virtual de Java)

MVC: Model View Controller (en español Modelo Vista Controlador)

MVP: Model View Presenter (en español Modelo Vista Presentador)

ORM: Object-Relational Mapping (en español Mapeo Objeto-Relacional)

PDF: Portable Document Format (en español Formato de Documento Portátil)

POJO: Plain Old Java Object (en español Objeto de Java Plano Antiguo)

REST: Representational State Transfer (en español Transferencia de Estado Representacional)

SDK: Software Development Kit (en español Kit de Desarrollo de Software)

SMTP: Simple Mail Transfer Protocol (en español Protocolo de Transferencia de Correo Simple)

SQL: Structured Query Language (en español Lenguaje de Consulta Estructurada)

URL: Uniform Resource Locator (en español Localizador de Recursos Uniforme)

Introducción

En la actualidad, empresas como Digitalica Group, C.A., le ofrecen a sus empleados el beneficio de realizar cierta cantidad de gastos mensuales y posteriormente iniciar un proceso de reembolso. Este proceso puede tomar mucho tiempo y puede resultar difícil si se hace manualmente, pues implica que el trabajador debe guardar todas las facturas durante el mes; al finalizar el mes debe entregar las facturas a la empresa y esta última inicia un proceso de verificación para culminar con el proceso de reembolso. Igualmente, para que la empresa pueda mantener un control y registro de todos los gastos que han sido reembolsados, debe guardar todos los meses las facturas de todos sus trabajadores, para lo que es necesario contar con un espacio físico que lo permita.

Dado que Digitalica Group, C.A. es una empresa dedicada al desarrollo de *software*, decidió automatizar todo el proceso de reembolso, mediante la creación de una aplicación móvil[1] y un servidor. De esta manera, con la aplicación los trabajadores pueden mantener un registro de gastos dentro de un dispositivo móvil, y con el servidor web se mantiene organizada y centralizada la información de estos gastos.

Esta aplicación debe contar con las siguientes funcionalidades:

- Registrar gastos con su fecha, monto, una breve descripción, fotos y categoría a la que

[1]Para efectos de simplicidad, se utilizará a lo largo del libro el término *aplicación móvil* para hacer referencia a aplicaciones que se ejecutan en dispositivos móviles.

pertenecen.

- Organizar y consultar gastos con criterios de búsqueda pre-establecidos.
- Crear reportes personalizados con los gastos y enviar dichos reportes a un servidor.
- Enviar reportes mediante archivos con formato PDF a los supervisores.

En cuanto al servidor, éste debe proveer funcionalidades que permitan controlar el estado de aprobación de los reportes recibidos.

En este informe se describe el proceso de desarrollo de una aplicación móvil y de un servidor web que cumplen con los requisitos mencionados anteriormente.

El informe se estructura de la siguiente manera: En el Capítulo 1 se describe de una forma general la empresa; en el Capítulo 2 se definen los conceptos teóricos estudiados para el desarrollo del proyecto; en el Capítulo 3 se mencionan las herramientas y tecnologías que facilitaron el desarrollo; en el Capítulo 4 se describe brevemente el marco de trabajo utilizado, Scrum; en el Capítulo 5 se describen todas las fases involucradas tanto en el diseño como el desarrollo de la solución; en el Capítulo 6 se exponen las conclusiones y recomendaciones que surgieron luego de la investigación y desarrollo del proyecto; finalmente, se muestran las referencias bibliográficas consultadas.

Capítulo 1

Entorno Empresarial

1.1. Descripción de la empresa

Digitalica Group, C.A. es una empresa que se dedica al desarrollo de soluciones móviles, aplicaciones empresariales y *middleware*. Se caracteriza por estar a la vanguardia, utilizando tecnologías innovadoras [1]. Sus principales servicios son:

- Consultoría en el área de desarrollo de aplicaciones móviles: Durante los últimos años se ha brindado apoyo en el área de soluciones móviles a empresas del sector educativo.
- Desarrollo de *software* y aplicaciones móviles para clientes. Se han desarrollado aplicaciones para medios de comunicación nacionales e internacionales, así como para empresas del sector educativo.
- Ofrecer *software* como un servicio: Actualmente, los productos que ofrece la empresa están enfocados en el área de noticias, medios de comunicación y registro de gastos e ingresos.

1.2. Misión

"Proveer soluciones y servicios de consultoría basados en tecnolías móviles, utilizando la mejores herramientas, metodologías y estándares de calidad en la industria" [1].

1.3. Visión

"Ser el líder en Latinoamérica en brindar soluciones innovadoras basadas en tecnologías móviles, con un equipo creativo con los conocimientos más sólidos en tecnologías de información para móviles, para cometir en el mercado global" [1].

1.4. Estructura organizacional

La estructura organizacional de la empresa está comprendida principalmente por la Junta de Directores, como se muestra en la figura 1.1.

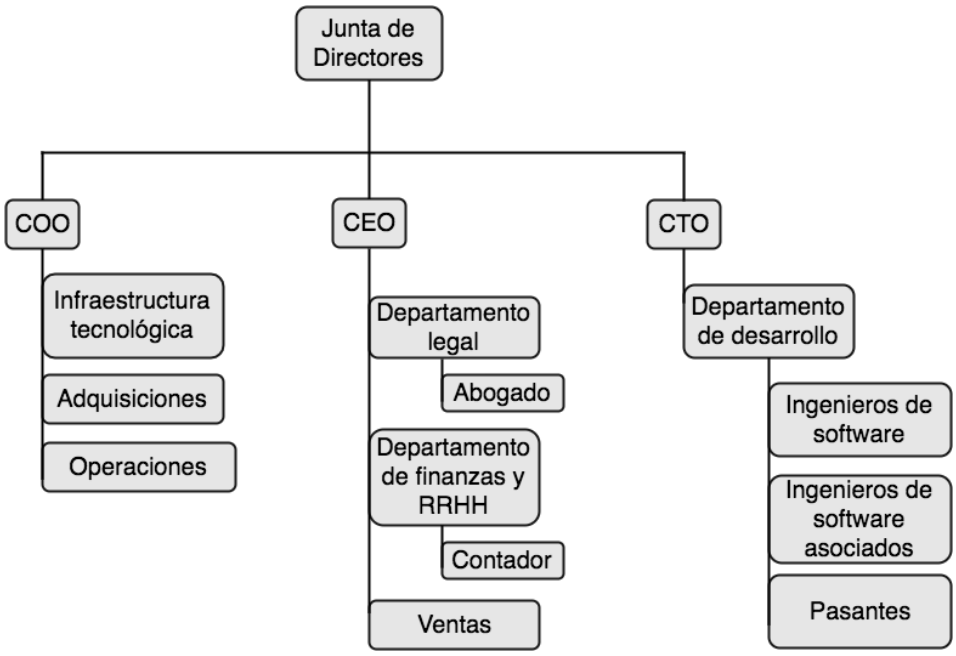

Figura 1.1: Estructura organizacional de la empresa

La Junta de Directores está compuesta por tres cargos: Director de Operaciones (COO por sus siglas en inglés), Director Ejecutivo (CEO por sus siglas en inglés) y Director de Tecnología (CTO por sus siglas en inglés).

El COO se encarga de gestionar las adquisiciones de la empresa y de mantener la infraestructura tecnológica.

El CEO se encarga de manejar la Dirección de Ventas, dirigir el Departamento Legal junto al abogado y dirigir el Departamento de Finanzas y Recursos Humanos junto al contador.

El CTO se encarga de dirigir el departamento de desarrollo, por lo que el equipo de desarrollo de *software* se comunica y le rinde cuentas a él.

1.5. Ubicación del pasante

Los pasantes se encuentran en el Departamento de Desarrollo, dirigido por el CTO, y donde se encuentan los especialistas de todas las áreas de desarrollo de *software*. Esto permite que los pasantes estén en constante contacto con personas que puedan asistirlos y solventar las dudas que puedan surgir durante el desarrollo.

Capítulo 2

Marco Teórico

En este capítulo se exponen los conceptos teóricos estudiados, tanto para entender las tecnologías utilizadas como para realizar el desarrollo de *software* requerido. Los conceptos estudiados formaron parte fundamental del proceso de diseño y desarrollo de la solución.

2.1. Patrones de diseño

Un patrón de diseño es una solución general y repetible a problemas que suelen presentarse en el proceso de diseño de software. Para que una solución pueda ser considerada como un patrón de diseño debe ser reutilizable, es decir, que se pueda aplicar a diferentes problemas de diseño en diferentes situaciones [2]. A continuación se describen los patrones utilizados en el diseño de la solución del proyecto.

2.1.1. *Singleton*

El *singleton* es un patrón de diseño que asegura la existencia de una sola instancia de la clase que lo implementa. Esto es útil cuando se necesita únicamente un objeto para coordinar las acciones en el sistema [3].

2.1.2. *Adapter*

Transforma la interfaz de una clase en otra interfaz que el cliente espera. Esto permite que una clase que no pueda utilizar la primera interfaz, sí pueda hacerlo a través de la otra [3].

2.1.3. *Facade*

Se utiliza para ofrecer una interfaz sencilla para operaciones complejas. Con este patrón se logra ofrecer una interfaz de alto nivel que hace que el sistema sea más fácil de usar para los clientes [3].

2.2. Patrones de arquitectura

Un patrón de arquitectura es una solución a problemas de arquitectura de *software*, ayudando a definir una estructura para sistemas de *software*. La diferencia entre los patrones de diseño y los de arquitectura es que estos últimos tienen un nivel de abstracción mayor [4].

2.2.1. Modelo Vista Controlador (MVC)

Es un patrón de arquitectura de *software* que divide una aplicación en tres partes que están interconectadas: los datos y la lógica de negocio (modelo), la interfaz de usuario (vista) y el módulo encargado de gestionar las comunicaciones (controlador). Esto se utiliza para separar la representación de la información de la forma en que dicha información es presentada al usuario [5].

2.2.2. Modelo Vista Presentador (MVP)

Es una derivación del patrón Modelo Vista Controlador (MVC) que se utiliza generalmente para construir interfaces de usuario [6]. En este patrón, la interacción entre el modelo

y la vista se logra únicamente a través del presentador, mientras que en el MVC la vista en ocasiones puede comunicarse directamente con el modelo. Otra diferencia con el patrón MVC es que en este último las peticiones son recibidas por el controlador. Éste, se comunica con el modelo para pedir los datos y luego se encarga de mostrar la vista adecuada. En el patrón MVP las peticiones son recibidas por la vista y delegadas al presentador, que es quien se comunica con el modelo para obtener los datos [7].

2.3. Arquitectura cliente-servidor

Es un modelo de arquitectura de sistema distribuido, donde existe un conjunto de procesos que proporcionan servicios (servidores) a otros procesos (clientes). Por lo general, el servidor no conoce la identidad ni el número de sus clientes, pero el cliente sí conoce la identidad del servidor [4]. Dentro de esta arquitectura existen cuatro tipos [8]:

- Cliente activo, servidor pasivo: El cliente procesa toda la información.
- Cliente pasivo, servidor pasivo: Tanto el cliente como el servidor pasan información.
- Cliente pasivo, servidor activo: El servidor procesa toda la información y el cliente presenta los datos.
- Cliente activo, servidor activo: Tanto el servidor como el cliente procesan la información.

2.4. Servicio web

Es un conjunto de funcionalidades diseñadas para permitir la comunicación entre diferentes dispositivos o aplicaciones a través de una red, utilizando un conjunto de protocolos y estándares [9].

2.5. HTTP (*Hypertext Transfer Protocol*)

Es un protocolo de comunicación que permite la transferencia de recursos en la web. Un recurso es cualquier información que puede ser identificada por un URL. Existen diversos métodos que permiten realizar peticiones con el protocolo HTTP, entre los que está POST [10].

Dentro de una petición HTTP suelen enviarse encabezados (*headers*) que proveen información de la misma, entre los que se encuentra el *Content-Type*. Con éste, se especifica el tipo de codificación que se usará en el cuerpo de la petición. Estos tipos pueden ser: application/xml, application/json, text/html, multipart/form-data, etc.

El tipo multipart permite enviar mensajes con varias partes combinadas en un solo cuerpo, y suele usarse también para el envío de archivos. [11]

2.5.1. HTTP POST

Es uno de los diferentes métodos de peticiones que soporta el protocolo HTTP, que permite enviar datos a un recurso, los cuales deben ser procesados. Se requiere que los datos sean enviados dentro del cuerpo del mensaje [12].

2.6. REST (*Representational State Transfer*)

Es un estilo de arquitectura de *software* utilizado para la creación de servicios web. Para que un sistema cumpla con esta arquitectura se deben cumplir diversas restricciones [13]:

- Cliente-Servidor: Debe existir una separación bien delimitada en cuanto a las ocupaciones del servidor y del cliente.
- Sin estados: Toda petición hecha de un cliente al servidor, debe contener toda la in-

formación necesaria para entenderla. De esta forma, la información de contexto de un cliente no puede ser almacenada en el servidor entre peticiones.

- Cache: La información de una respuesta a una petición debe ser marcada como *cacheable* o no. Si es *cacheable*, para ciertas peticiones del mismo estilo no es necesario realizar una nueva petición, sino que el cliente puede usar la respuesta previa.

- Interfaz uniforme: La característica principal que distingue la arquitectura REST del resto, es el énfasis que se da en tener una interfaz uniforme entre los componentes. Para lograr una interfaz uniforme, es necesario cumplir con las siguientes condiciones: todo recurso debe estar identificado, la manipulación de los recursos debe hacerse bajo representaciones, los mensajes deben ser autodescriptivos, y la hipermedia debe ser el motor del estado de la aplicación.

- Sistema en capas: Los componentes que conforman el sistema deben estar distribuidos en capas. Así, ningún componente podrá ver información que se encuentre en una capa diferente a la cual pertenece.

2.7. JSON (*JavaScript Object Notation*)

Es un formato de texto ligero utilizado para intercambiar datos. Un JSON puede estar compuesto por dos tipos de estructuras: una colección de pares clave/valor, o una lista ordenada de valores [14].

2.8. POJO (*Plain Old Java Object*)

Es una instancia de una clase que no extiende ni implementa ninguna otra [15]. Sirve para promover el uso de clases simples que no dependen de ningún *framework* [16].

2.9. DAO (*Data Access Object*)

Es un objeto que provee una interfaz abstracta para realizar operaciones sobre una base de datos u otro mecanismo de persistencia. Esconde todos los detalles de implementación a sus clientes, por lo que la interfaz expuesta por un DAO no cambia [17].

2.10. *Manager*

Es un objeto que se utiliza para actuar como puente entre la capa de persistencia (base de datos) y la capa web. Dentro de este componente se incluye la lógica de negocio de la aplicación [18].

Capítulo 3

Marco Tecnológico

En este capítulo se definen las diferentes herramientas utilizadas a lo largo de todo el proyecto. Dichas herramientas permitieron el desarrollo del *software* planteado como solución al problema.

3.1. Herramientas, entornos y lenguajes

3.1.1. Android

Es un sistema operativo de código abierto basado en el sistema operativo Linux, utilizado principalmente para dispositivos móviles [19]. Es la plataforma para la cual se desarrolló la aplicación.

La plataforma de Android está basada en una arquitectura que tiene diferentes capas que van desde servicios de bajo nivel del sistema operativo, que interactúan directamente con el *hardware*, hasta las aplicaciones. Además, Android provee un kit de desarrollo de *software* (SDK por sus siglas en inglés), que permite crear aplicaciones [20].

La capa de más bajo nivel es la del kernel de Linux, como se puede observar en la figura 3.1.

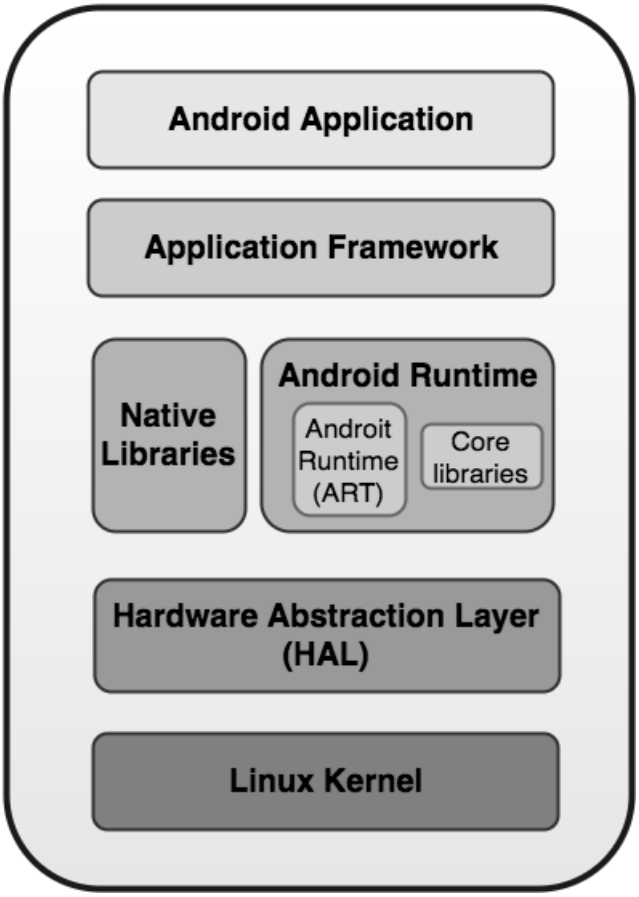

Figura 3.1: Arquitectura de la plataforma de Android

La plataforma de Android está basada en el kernel de Linux. En ella se proveen servicios del sistema operativo, como el manejo de memoria y procesos [20].

La siguiente capa es la de abstracción de *harwdare* (HAL por sus siglas en inglés). En ella se provee un nivel de abstracción para la comunicación con el *hardware* del dispositivo, y contiene los *drivers* esenciales para el funcionamiento de dicho *hardware*. El HAL está compuesto por múltiples módulos de librerías, cada uno de los cuales implementa una interfaz para un tipo específico de componente de *hardware*. Por ejemplo, se tienen módulos para la cámara y para el *bluetooth*.

Encima de esta capa se encuentra la de librerías nativas, dentro de las cuales se encuentra SQLite, que permite guardar datos de las aplicaciones [20].

En conjunto con la librerías, está la sección de Android *Runtime* en el tercer nivel. Dentro de ella se encuentra un componente llamado ART (*Android Runtime*), que es el entorno de ejecución de las aplicaciones [20].

La cuarta capa de abajo hacia arriba corresponde al *framework* de las aplicaciones (*Application Framework*). En ella se proveen servicios de alto nivel mediante APIs escritos en Java. Dentro de estos servicios destaca el Manejador de Actividades (*Activity Manager*), que se encarga de controlar todas las actividades de una aplicación [20]. Una actividad (*activity*) es un componente que provee una pantalla con la cual el usuario final puede interactuar. Generalmente, a cada actividad corresponde una interfaz de usuario [21]. El desarrollo de aplicaciones para Android se realiza dentro de esta capa, pues en ella se proveen los servicios necesarios [22].

Por último, se encuentra la capa de aplicaciones (*Android Application*), que es donde se encuentran instaladas las aplicaciones del dispositivo [20].

3.1.2. Java

Es un lenguaje de programación de alto nivel orientado a objetos, que puede correr virtualmente en cualquier computadora [23]. Es el lenguaje en el que están desarrolladas la mayoría de las aplicaciones de Android [24]. Se utilizó tanto para la programación de la aplicación móvil como del servidor.

3.1.3. MySQL

Es un sistema de manejo de base de datos relacional basado en el Lenguaje de Consulta Estructurado (SQL por sus siglas en inglés) [25]. Se utilizó para la gestión de la base de datos del servidor.

3.1.4. SQLite

Es una librería que implementa un motor de base de datos relacional, basado en el Lenguaje de Consulta Estructurado (SQL por sus siglas en inglés). Forma parte del programa principal, en lugar de ser un proceso independiente como ocurre con los sistemas de gestión de base de datos cliente-servidor [26]. Se utiliza para la gestión de la base de datos del dispositivo móvil.

3.1.5. Maven

Es una herramienta que permite compilar y manejar proyectos de *software* basados en Java. Se encarga de conectar las dependencias de los paquetes [27]. Se utilizó para manejar la dependencia entre las librerías usadas para el desarrollo del servidor.

3.1.6. Jetty

Es un servidor HTTP que puede funcionar independientemente por sus propios medios, o que puede correr dentro de otra aplicación [28]. Se utilizó para ofrecer los servicios web creados, mediante el protocolo HTTP.

3.1.7. Android Studio

Es el entorno de desarrollo integrado (IDE por su siglas en inglés) oficial para el desarrollo de aplicaciones nativas para Android [29]. Se utilizó para desarrollar la aplicación móvil.

3.1.8. Eclipse

Es un entorno desarrollo integrado (IDE por sus siglas en inglés) utilizado principalmente para desarrollar aplicaciones en Java [30]. Se utilizó para desarrollar el servidor.

3.1.9. Git

Es un sistema de control de versiones distribuido (DVCS por sus siglas en inglés). Es una herramienta que permite gestionar las distintas versiones de un proyecto de *software* [31]. Se utilizó para mantener un control entre las versiones del código que surgieron a lo largo del desarrollo tanto de la aplicación móvil como del servidor.

3.2. *Frameworks* y librerías

3.2.1. AppFuse

Es un *framework* utilizado para la creación de aplicaciones web en la máquina virtual de Java (JVM por sus siglas en inglés). Dentro de AppFuse, se pueden integrar otras tecnologías como Bootstrap, Maven, Hibernate, Spring, Tapestry, etc [32]. Se utilizó en el desarrollo del servidor mediante el uso de otros *frameworks* (Hibernate, Spring y Tapestry).

3.2.2. Hibernate

Es una herramienta de mapeo objeto-relacional (ORM por sus siglas en inglés), que permite el mapeo de objetos de Java a bases de datos relacionales [33]. Se utilizó para proveer una abstracción sobre los elementos de la base de datos del servidor.

3.2.3. Spring

Es un *framework* utilizado para el desarrollo de sistemas y aplicaciones basadas en la máquina virtual de Java (JVM por sus siglas en inglés). Provee funcionalidades para el desarrollo de servicios web basados en la arquitectura REST [34]. Se utilizó para la creación de los servicios web.

3.2.4. Tapestry

Es un *framework* para el desarrollo de aplicaciones web en Java. Toma un enfoque modular al relacionar la interfaz de usuario con clases de Java. El uso de Tapestry para la construcción de aplicaciones implica la creación de archivos HTML, y clases de Java para cada uno de ellos. [35]. Se utilizó para la creación de las vistas y controladores de la aplicación web.

3.2.5. Gson

Es una librería de Java que permite la conversión de objetos a JSON y viceversa [36]. Se utilizó en la aplicación móvil para transformar objetos de Java a archivos con formato JSON para ser enviados al servidor, así como para transformar archivos con formato JSON a objetos de Java en el servidor.

3.2.6. Retrofit

Es una librería de Android que permite realizar peticiones utilizando el protocolo HTTP, mediante una interfaz de Java [37]. Se utilizó para realizar peticiones al servidor desde la aplicación móvil.

Capítulo 4

Marco Metodológico

A continuación se describe el procedimiento seguido para el desarrollo del proyecto. Se utilizó el marco de trabajo Scrum dado que éste es utilizado por la empresa para el desarrollo de *software*.

4.1. Scrum

Scrum es un marco de trabajo que se basa en el desarrollo iterativo e incremental de un producto, en lugar del modelo clásico de planificación y ejecución completa [38]. Se caracteriza por ser una metodología ligera, fácil de entender y difícil de dominar, que permite entregar incrementos de producto potencialmente productivos [39].

4.1.1. Roles

En Scrum, el desarrollo se realiza por uno o más equipos de trabajo dentro de los cuales existen tres roles: dueño del producto, facilitador de Scrum y el equipo de desarrollo [38].

4.1.1.1. Dueño del producto (*Product owner*)

Es el representante de los clientes. Dentro del equipo de Scrum, es el líder principal del producto y el responsable de decidir qué funcionalidades serán desarrolladas y la prioridad que tendrá cada una de ellas. Debe comunicar al resto de los involucrados en el proyecto una visión clara de lo que se quiere lograr. Tiene la obligación de asegurar que siempre se entregue un producto con el máximo de valor, por lo que debe colaborar con el resto del equipo para responder cualquier duda que surja [38].

4.1.1.2. Facilitador de Scrum (*Scrum master*)

Actúa como facilitador tanto para el dueño del producto como para el equipo de desarrollo. Es el encargado de ayudar al resto del equipo a entender y cumplir con los principios y prácticas de Scrum. También tiene la responsabilidad de eliminar cualquier impedimento que el equipo no sea capaz de resolver y que afecte su productividad [38].

4.1.1.3. Equipo de desarrollo

Es el encargado de desarrollar el producto. Es un equipo que está compuesto por arquitectos, programadores, probadores, administradores de base de datos, diseñadores de interfaces, entre otros. Son los responsables de diseñar, desarrollar y probar el producto [38].

4.1.2. Actividades

En Scrum, el trabajo se desarrolla en iteraciones de una duración máxima de un mes, llamadas ***sprints***. Al final de cada iteración, se debe haber desarrollado una parte del producto final, la cual debe ser completamente funcional. Dentro de cada iteración existe una serie de eventos o actividades que se llevan a cabo: planeación de la iteración, ejecución de la iteración, reuniones diarias, revisión de la iteración y retrospectiva de la iteración [38].

4.1.2.1. Planeación de la iteración (*Sprint planning*)

Para determinar qué funcionalidades del producto final son las más importantes y próximas a desarrollar, el equipo de trabajo (dueño del producto, facilitador de Scrum y el equipo de desarrollo) realizan una reunión llamada *sprint planning* o planeación de la iteración [38].

Durante la reunión, el dueño del producto y el equipo de desarrollo establecen una meta que debe ser cumplida para el final de la iteración. De acuerdo a esta meta, el equipo de desarrollo decide de una manera realista qué incrementos del producto final pueden entregarse al terminar la iteración [38].

4.1.2.2. Ejecución de la iteración (*Sprint execution*)

Luego de la planeación de la iteración, el equipo de desarrollo desarrolla todas las tareas acordadas en la reunión. Esto es lo que se conoce como la ejecución de la iteración [38].

4.1.2.3. Reunión diaria (*Daily scrum*)

Cada día dentro de la ejecución de la iteración, los miembros del equipo de desarrollo se reúnen durante un máximo de 15 minutos con el fin de informar qué se hizo el día anterior, qué se tiene planificado realizar el presente día y qué impedimentos se han presentado durante el desarrollo de cada trabajo [38].

4.1.2.4. Revisión de la iteración (*Sprint review*)

Al final de cada iteración, ocurre un evento que se conoce como *sprint review* o revisión de la iteración. El objetivo de esta actividad es revisar el incremento y realizar las adaptaciones necesarias al producto [38].

4.1.2.5. Retrospectiva de la iteración (*Sprint retrospective*)

Esta actividad ocurre generalmente luego de la revisión de la iteración (*sprint review*) y antes de la próxima planeación de la iteración (*sprint planning*). Es una oportunidad para que todo el equipo se reúna para discutir qué ha funcionado y qué no acerca de Scrum. Al final de la retrospectiva, el equipo de Scrum habrá discutido qué acciones deberán tomarse para mejorar la dinámica en las próximas iteraciones [40].

4.1.3. Artefactos

Dentro de Scrum existen dos herramientas o artefactos que permiten mantener un seguimiento del proyecto: el documento de historias de usuario y la lista de tareas de la iteración. [40].

4.1.3.1. Documento de historias de usuario (*Product backlog*)

Es una lista de los requerimientos funcionales del producto ordenados según su importancia. El dueño del producto es el responsable de definir qué elementos serán incluidos en esta lista y de colocarlos según su prioridad, de manera que los elementos de mayor valor o prioridad aparezcan al principio de la lista, y los de menos valor al final de la misma [40].

Uno de los formatos en los que se pueden expresar los elementos que conforman el documento (*Product Backlog*), son las historias de usuario (*user stories*). Estas se caracterizan por especificar el rol del usuario, qué se quiere lograr con la funcionalidad y cuál es el beneficio [40].

4.1.3.2. Lista de tareas de la iteración (*Sprint backlog*)

Es una lista donde se presenta un subconjunto de los elementos del *product backlog* divididos en tareas más pequeñas [40].

Capítulo 5

Desarrollo de la solución

En este capítulo se describe el trabajo realizado para el desarrollo de la solución. Este proceso se dividió en tres fases: investigación, diseño y desarrollo del sistema, que incluye tanto la aplicación móvil como el componente servidor. Para las fases de diseño y desarrollo, se utilizó el marco de trabajo Scrum, descrito en el capítulo 4.

5.1. Investigación

El objetivo de esta primera fase consistió en familiarizarse con el entorno de trabajo.

Al inicio, se estudiaron diversos patrones de diseño, donde se investigó acerca de patrones ampliamente utilizados en el desarrollo de aplicaciones para Android y en la programación orientada a objetos en general. Esto permitió estructurar la aplicación de una manera ordenada y entendible, permitiendo así que ésta pueda ser extendida facilmente.

Seguidamente, se investigó acerca de los componentes y herramientas para el desarrollo de aplicaciones en Android, así como el uso del IDE Android Studio.

Finalmente, se realizó una aplicación de prueba para poner en práctica algunos conceptos básicos en el desarrollo de aplicaciones para Android. Con esto se aprendió la estructura

general de una aplicación, así como el flujo de ejecución.

5.2. Análisis y diseño de la solución

En esta fase tuvieron lugar dos etapas fundamentales para el desarrollo: análisis del problema y diseño de la solución.

Para comprender mejor el problema presentado, se realizó una reunión con el dueño del producto (*product owner*). En esta reunión se realizó el primer levantamiento de requerimientos. A partir de estos requerimientos, se diseñó una arquitectura base que permite solucionar el problema (ver sección 5.2.2.1).

5.2.1. Análisis del problema

Se necesita una aplicación móvil que permita a los empleados de la compañía registrar gastos personales que posteriormente serán reembolsados. Para estos gastos, se requiere mantener información de su fecha, monto y descripción. También se requiere que a través de la aplicación, los empleados puedan capturar fotos de las facturas de los gastos; un gasto puede tener un máximo de tres fotos. Dado que la empresa realiza el reembolso de gastos en determinadas áreas, es necesario que estos gastos puedan asociarse a categorías que los identifiquen

También se quiere realizar consultas de los gastos registrados en un rango de fechas y generar un reporte de dichos gastos. Este reporte debe ser enviado a un servidor como un archivo de formato PDF.

Se quiere que los supervisores de la compañía puedan revisar los reportes de gastos mensuales de los trabajadores. Los supervisores deben poder aprobar o rechazar dichos reportes. Además, el archivo PDF de cada reporte debe ser guardado en el servidor.

Por otra parte, se quiere que la aplicación, además de solucionar el problema de la empresa, pueda ser utilizada en un futuro por el público en general. Por esta razón, se quiere que,

además de gastos, se puedan registrar ingresos; los requerimientos para los ingresos son los mencionados anteriormente para gastos. Adicionalmente, la aplicación móvil debe permitir la creación de múltiples cuentas con diferentes monedas. Estas cuentas sirven de mecanismo para agrupar ingresos y gastos según desee el usuario. También se requiere que el usuario pueda manejar sus propias categorías, es decir, que pueda crear nuevas categorías y editar las ya existentes.

Para especificar cada uno de estos requerimientos, así como su prioridad, se realizó una primera versión del Documento de Historias de Usuario (*Product Backlog*). Éste fue modificándose a lo largo del desarrollo del proyecto (ver apéndice C).

Para entender mejor las principales funcionalidades que debe tener el sistema, se realizaron algunos diagramas de estados y actividades, que se pueden revisar en el apéndice D.

5.2.2. Diseño de la solución

5.2.2.1. Arquitectura de la solución

Dados los requerimientos anteriores, se diseñó una arquitectura base para solucionar el problema. Se dividió el sistema según sus componentes: servidor web, cliente móvil y cliente web.

Como se muestra en la figura 5.1, es una arquitectura cliente-servidor, donde el componente móvil y la aplicación web juegan el papel de cliente. En este caso, la aplicación móvil actúa como un cliente activo, pues es quien manejará los datos relevantes. El servidor solo recibirá esta información, sin poder modificarla. Esto corresponde a una arquitectura del tipo cliente activo-servidor pasivo, descrita en el capítulo 2.

Por medio de la aplicación móvil se podrá hacer el registro de los gastos personales. El servidor permitirá la revisión de los reportes de gastos, así como su aprobación o rechazo.

Toda la información relacionada a los gastos (fecha, monto, categorías, fotos) será creada y

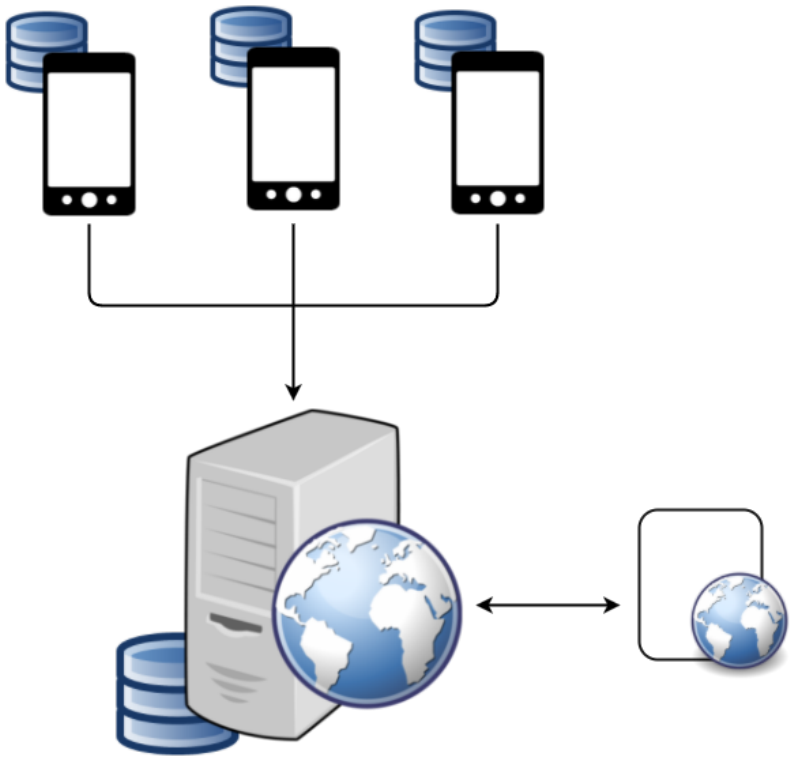

Figura 5.1: Arquitectura base de la solución

guardada localmente en una base de datos en cada dispositivo móvil. Cada dispositivo podrá enviar al servidor reportes de gastos en forma de archivos PDF. La comunicación entre la aplicación móvil y el servidor se hace por medio de servicios web que presta este último.

Estos reportes deberán ser guardados en la base de datos del servidor. Además, a través de una aplicación web, se podrá aprobar o rechazar los reportes recibidos.

La aplicación móvil es la que se encarga de crear y mantener los datos correspondientes a los reportes, mientras que el servidor contiene información únicamente de los reportes que son recibidos a través del servicio web correspondiente. Por esta razón, el cliente (en este caso, el usuario de la aplicación móvil) se encargará de mantener la consistencia entre los datos que se encuentran en el dispositivo móvil, y los que tiene el servidor. En este sentido, si en algún momento se hacen cambios locales (es decir, dentro del dispositivo móvil) a información que esté contenida dentro de un reporte que se haya enviado al servidor, es responsabilidad del

usuario de la aplicación enviar nuevamente el reporte, de manera que los datos que tiene el dispositivo sean consistentes con los que tenga el servidor.

5.2.2.2. Arquitectura de *software*

Para los componentes mencionados anteriormente, se definieron los patrones de arquitectura de *software* sobre los cuales se desarrollaron. Para el servidor se utilizó el patrón Modelo Vista Controlador (MVC) y para la aplicación móvil el Modelo Vista Presentador (MVP).

Tal como se explicó en el capítulo 2, MVC se usa para aislar la lógica de negocio de la interfaz de usuario. En este patrón, el modelo representa todos los datos de la aplicación y las reglas de negocio que se usan para manejar dichos datos.

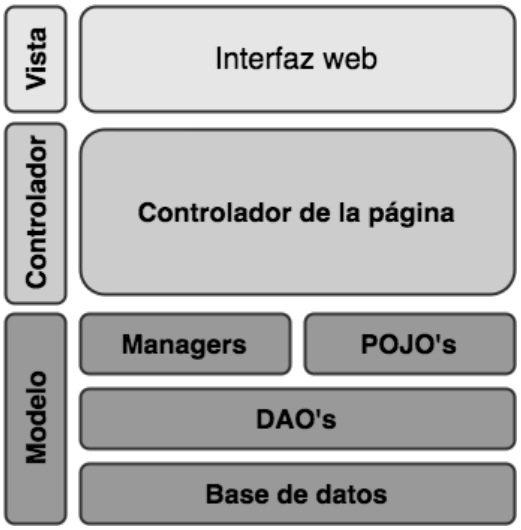

Figura 5.2: Modelo Vista Controlador

Como se muestra en la figura 5.2, el modelo incluye la base de datos, los POJO's que permiten darle un nivel de abstracción a las tablas de la base de datos, los DAO's que proveen una interfaz sencilla para realizar operaciones sobre la base de datos, y los managers que contienen la lógica de negocio y actúan como fachada para los DAO's (implementando el patrón de diseño *Facade* mencionado en el capítulo 2). La vista corresponde a todo lo que

son los elementos de la interfaz de usuario. En este caso, está compuesta por las interfaces de la aplicación web. El controlador es la capa que recibe las acciones del usuario y se encarga de manejarlas, comunicándose con el modelo o con la vista según sea necesario.

Por su parte, MVP es una derivación del patrón MVC, como se observa en la figura 5.3.

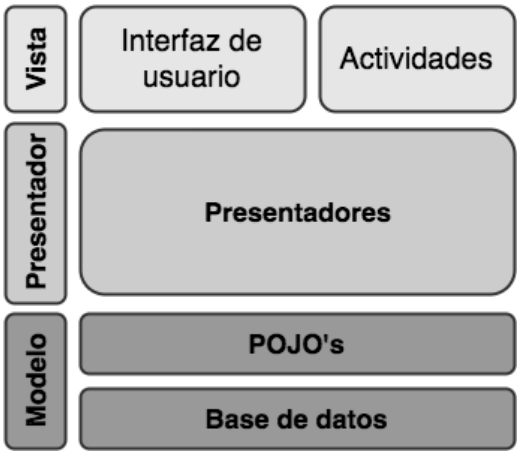

Figura 5.3: Modelo Vista Presentador

El modelo incluye tanto la base de datos como los POJO's, los cuales fueron utilizados para establecer una correspondencia entre objetos de la aplicación con tablas de la base de datos. La vista está formada por las actividades (*activities*) definidos en el capítulo 3, y las interfaces de usuario.

En este caso, las acciones de usuario son recibidas por la vista. Luego, la vista se comunica con el presentador, que pide los datos al modelo. Al obtener los datos necesarios, el presentador invoca a la vista a través de una interfaz para que ésta actualice su contenido.

5.2.2.3. Estructura de datos

Para dar lugar al desarrollo del sistema, fue necesario diseñar un modelo de datos que permita mantener la información necesaria en el dispositivo móvil, y otro para mantener información en el servidor.

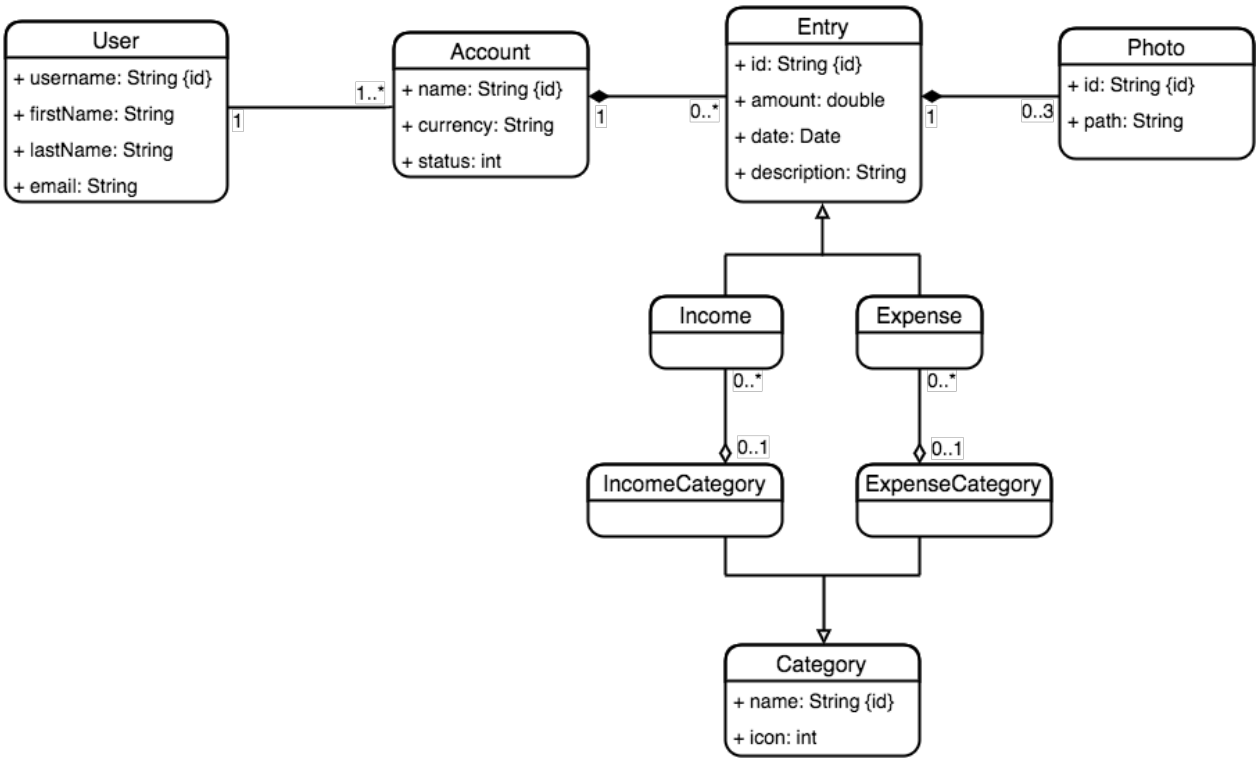

Figura 5.4: Diagrama de clases de la aplicación móvil

En la figura 5.4 se puede observar el diagrama de clases que representa el modelo de datos de la aplicación móvil.

A continuación se describirán cada una de estas clases, así como sus atributos:

- ***User***: representa un usuario de la aplicación móvil. Almacena información de su nombre de usuario, nombre, apellido y correo electrónico.

- ***Account***: representa un conjunto de gastos e ingresos. Almacena información de su nombre, estado (activa o archivada) y la moneda en la que estarán los montos de los gastos e ingresos.

- ***Entry***: representa un registro dentro de una cuenta, y puede ser de dos tipos: ingreso (*Income*) o gasto (*Expense*). Almacena información de su *id*, monto, fecha y descripción.

- ***Category***: representa las áreas a las que puede pertenecer un *entry*. Puede ser de dos tipos: categoría de ingresos (*IncomeCategory*) o categoría de gastos (*ExpenseCategory*). Almacena información de su nombre y un ícono que la representa.

- ***Photo***: representa las fotos asociadas a un *entry*. Almacena información de su *id*, y la ruta del archivo dentro del dispositivo móvil.

Por otra parte, en la figura 5.5 se puede observar la estructura de datos del servidor. En este caso, se tienen tres clases: *User* (que puede ser *Administrator*) y *Report*.

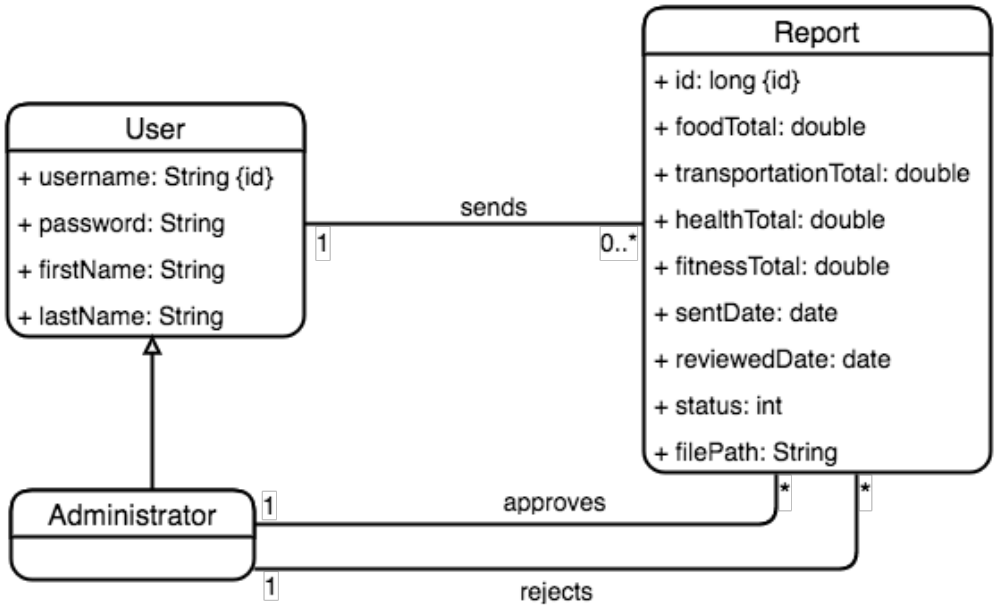

Figura 5.5: Diagrama de clases del servidor

A continuación se describirán cada una de estas clases, así como sus atributos:

- ***User***: representa los usuarios registrados en el sistema. Almacena información del nombre, apellido, nombre de usuario y contraseña.

- ***Administrator***: representa un usuario administrador del sistema. Los usuarios administradores son los únicos que pueden aprobar o rechazar un reporte.

- ***Report***: representa un reporte enviado por un usuario, que contiene información de la suma total de *entries* pertenecientes a ciertas categorías. Almacena información de su *id*, monto total de entries de cuatro categorías: comida (*food*), transporte (*transportation*), salud (*health*) y deporte (*fitness*). Además, también almacena la fecha en que se envió el reporte, su estado (aprobado, rechazado o sin revisar) y la fecha en que fue revisado. Por último, tiene la ruta del archivo PDF correspondiente al reporte.

Estos diagramas sirvieron como base para la creación del modelo tanto del servidor como de la aplicación móvil.

5.3. Desarrollo

Durante esta fase se implementó progresivamente la versión alfa de una aplicación móvil y servidor web, para dar solución al problema de registro y control de gastos.

Para cumplir con los requerimientos del componente móvil, se creó una aplicación para el sistema operativo Android, y se trabajó con el IDE Android Studio.

El desarrollo de la aplicación implicó la creación de múltiples vistas (tanto interfaces de usuario como la lógica para manejar acciones de usuario), y de diversas consultas a la base de datos, muchas de las cuales realizan modificaciones a la misma. Para un mayor detalle de las interfaces de usuario creadas, revisar apéndice A.

En primer lugar, se creó la vista principal de la aplicación, donde se muestra el total de ingresos, el total de gastos y el balance general (ver figura 5.6).

Se implementó una vista para crear y editar gastos (ver figura 5.7). Para estos gastos, se debe guardar su monto, fecha, descripción, fotos de recibos y categoría a la que pertenecen. Posteriormente, se extendió la funcionalidad de creación de gastos para permitir también el manejo de ingresos.

Figura 5.6: Interfaz principal de la aplicación

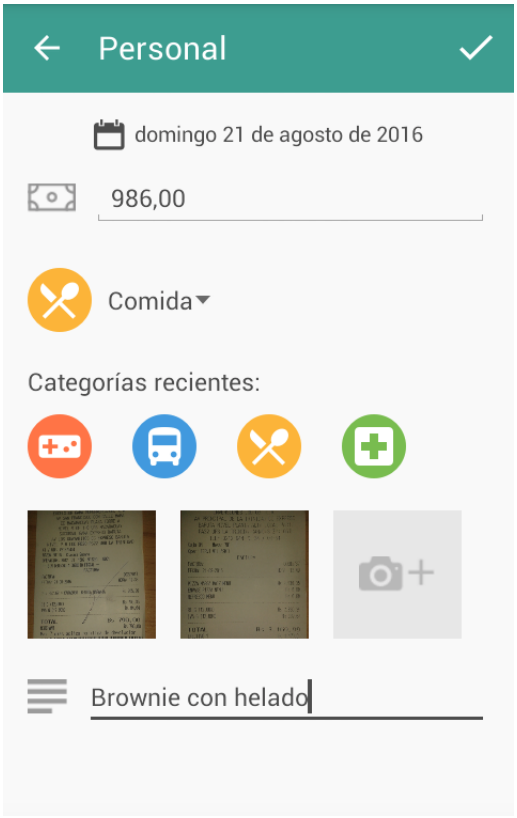

Figura 5.7: Interfaz para crear o editar gastos o ingresos

Para mostrar los gastos e ingresos guardados, se creó una vista para listarlos (ver figuras 5.8 y 5.9). En las listas se muestran la descripción/categoría a la que pertenece cada entrada, su monto y su fecha. A través de cada lista, se permite ver los detalles de cada gasto/ingreso, editarlos y eliminarlos. Además, también se permite filtrar las listas por palabras claves.

Para la creación de las listas, se utilizó un componente que provee Android para facilitar la tarea. Para hacer uso de él, es necesario crear una clase que actúe como intermediario entre el modelo de datos que se quiere mostrar en la interfaz de usuario, y el elemento de la interfaz que se encargará de mostrar dichos datos. Por esta razón, se utilizó el patrón de diseño *Adapter* (ver capítulo 2) para crear la clase en cuestión.

Figura 5.8: Interfaz para listar gastos

Figura 5.9: Interfaz para listar ingresos

También se crearon vistas para el manejo de categorías. Se creó una vista para la creación/edición de categorías, y otra para listarlas (ver figuras 5.10 y 5.11). Estas categorías pueden ser de dos tipos, categorías de gastos o categorías de ingresos.

Posteriormente, se implementaron funcionalidades que permiten el manejo de cuentas de ingresos/gastos. Por esta razón, también se creó una funcionalidad para agregar nuevas cuentas, y poder crear nuevos gastos e ingresos dentro de dichas cuentas. Estas cuentas pueden ser de dos tipos: activas o archivadas; la navegación entre cuentas se hace únicamente para las cuentas activas.

Además, se creó una vista para listar las cuentas creadas y guardadas en el dispositivo (ver figuras 5.12 y 5.13). A partir de esta vista, se puede editar, eliminar o cambiar el estado de las cuentas (activa o archivada).

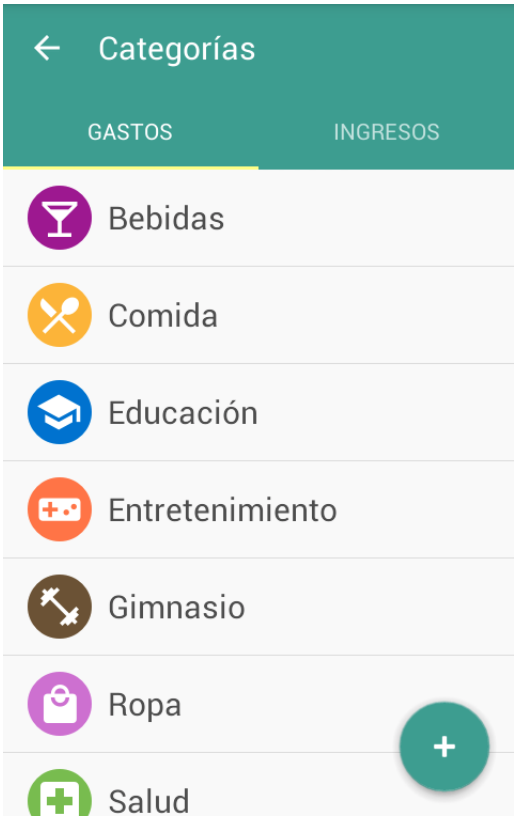

Figura 5.10: Interfaz para listar categorías de gastos

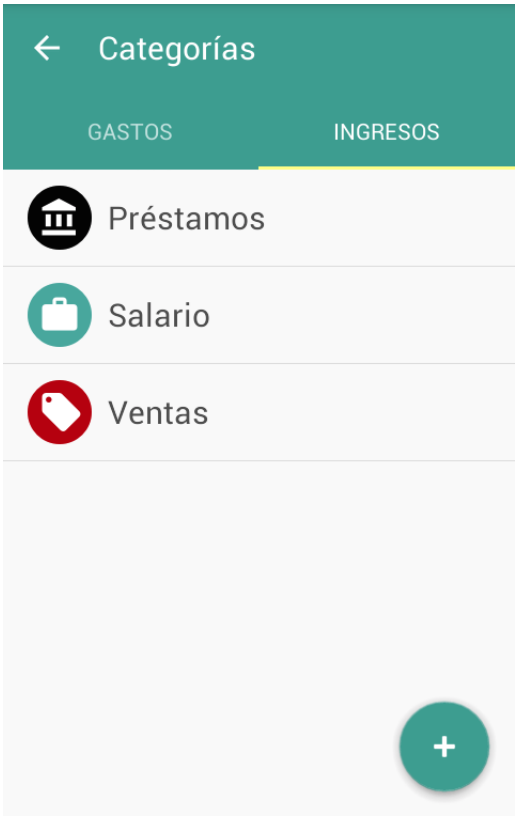

Figura 5.11: Interfaz para listar categorías de ingresos

Además, también se crearon dos vistas para reportes de gastos: una para reportes de balance general, y otra para reporte de gastos e ingresos agrupados por categorías (ver figuras 5.14 y 5.15).

La vista de reporte de balance general muestra una lista de gastos e ingresos filtrados por fecha. La vista de reporte por categorías muestra una lista de categorías con la suma total de los gastos e ingresos de dichas categorías, filtradas por fecha.

Como se explicó en la sección anterior, es necesario guardar toda esta información en la base de datos local de cada dispositivo. Para ello, se utilizó la librería SQLite.

Para mantener la lógica separada de la interfaz gráfica, se utilizó el patrón de arquitectura MVP o Modelo Vista Presentador, descrito en el capítulo 2. De esta manera, toda acción en la

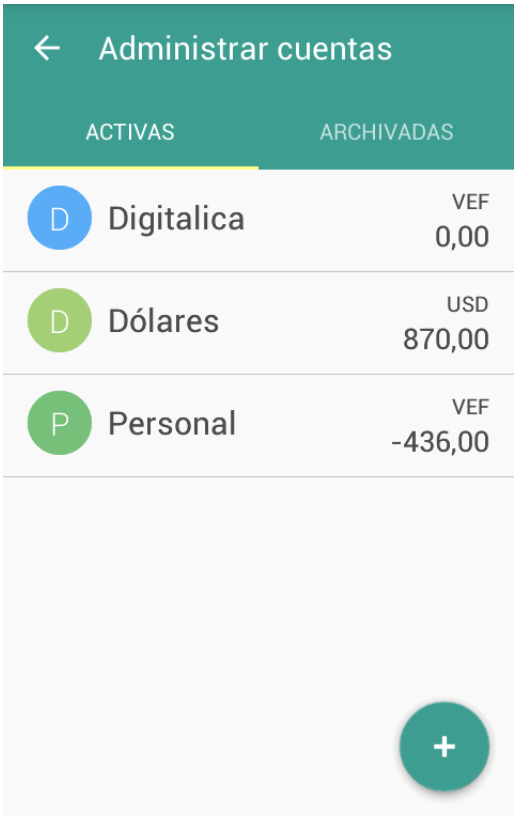

Figura 5.12: Interfaz para listar cuentas activas

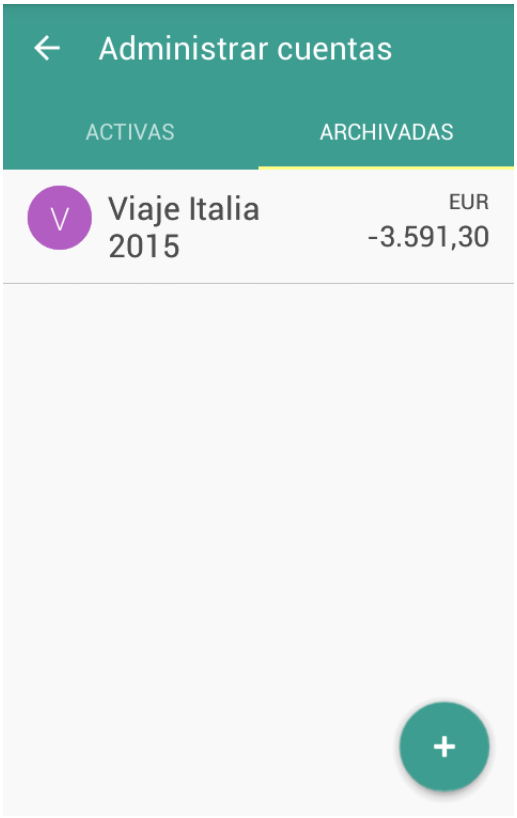

Figura 5.13: Interfaz para listar cuentas archivadas

vista que requiera realizar operaciones sobre la base de datos, se implementa por medio de una capa intermedia, conocida como presentador. Dado que el presentador es el que se comunica con el modelo para modificarlo, puede existir la necesidad de realizar estas operaciones en un hilo diferente al principal (que se encarga de actualizar la vista). Por esta razón, para la creación de las clases que actúan como presentadores, se utilizó el patrón de diseño *Singleton* (ver capítulo 2). Con esto, se asegura la existencia de una sola instancia de cada presentador.

Se implementó el modelo descrito en la figura 5.4. Se creó la base de datos con cada una de sus tablas, así como las clases (POJO's) que permitieron dar un nivel de abstracción mayor para el mapeo con la base de datos.

También se crearon las vistas necesarias, lo que incluyó tanto las interfaces gráficas como

Figura 5.14: Interfaz para mostrar reporte de balance general

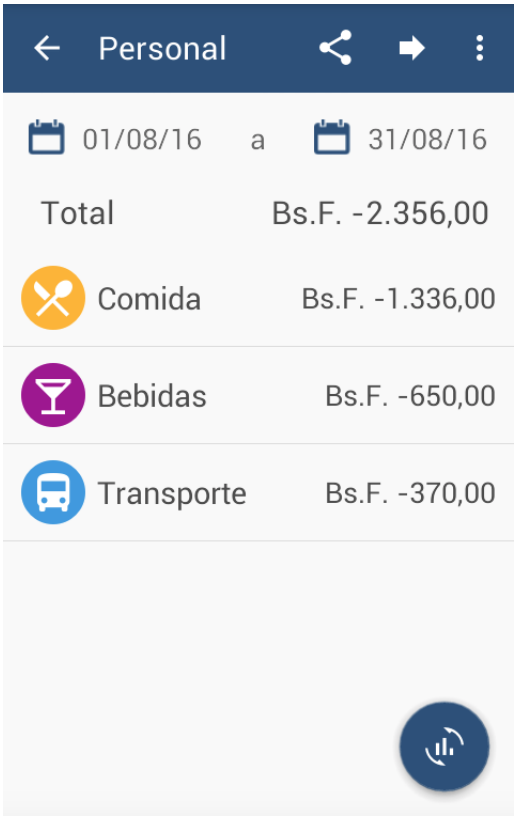

Figura 5.15: Interfaz para mostrar reporte por categorías

la lógica necesaria para manejar las acciones del usuario (actividades). Para cada vista, se creó una clase que actúa como la capa de presentador, que es invocado por la vista para realizar acciones sobre el modelo. La vista se encarga de la creación de los objetos correspondientes al modelo, para luego pedir alguna acción al presentador sobre dichos objetos. El presentador tiene la función de invocar al modelo para realizar los cambios necesarios en la base de datos. Para mayor detalle de la interacción entre la vista, el modelo y el presentador, ver figuras D.5 y D.6 del apéndice D.

Dada la naturaleza iterativa del marco de trabajo utilizado, muchas de las funcionalidades implementadas fueron hechas a lo largo de varias iteraciones (*sprints*). Por esta razón, las vistas, presentadores, así como el modelo, fueron creados y actualizados progresivamente en

cada iteración.

Por otra parte, se creó un servidor al que se puede enviar información desde la aplicación móvil. Para esto fue necesario implementar el modelo de datos mostrado en la figura 5.5. La herramienta utilizada para el desarrollo web crea por defecto una base de datos con tres tablas: *User*, *Role* y *UserRole*. Esta base de datos se pudo adaptar al modelo de datos diseñado para el servidor, por lo que únicamente fue necesario crear la tabla *Report*.

Todos los servicios web se crearon basándose en el estilo arquitectónico REST (ver capítulo 2).

Para permitir el acceso a la aplicación móvil, se creó un servicio web que permita hacer la autenticación del usuario. La aplicación envía al servidor un JSON con los datos del usuario (nombre de usuario y contraseña) por medio de una petición HTTP POST.

Se creó un servicio web que recibe un archivo PDF cuyo contenido es un reporte de gastos e ingresos. Además del archivo, recibe un JSON que contiene la suma de gastos por cada categoría de interés para la empresa, el mes y el año del reporte. Estos datos se enviaron por medio de una petición HTTP POST con Content-Type Multipart, cuyo cuerpo está compuesto por un archivo y un JSON. Para la conversión de objetos de Java a JSON, y viceversa, se utilizó la librería GSON tanto en el servidor como en la aplicación móvil. Para persistir esta información, se utilizó Hibernate y MySQL como manejador de base de datos.

También se desarrolló una aplicación web que permite aprobar o rechazar los reportes enviados al servidor, así como ver los reportes recibidos y ya revisados (ver figuras 5.16 y 5.17).

Se creó una funcionalidad para notificar a los supervisores vía correo electrónico cuando un nuevo reporte es recibido en el servidor; este correo electrónico incluye el archivo PDF del reporte. Asimismo, cuando un supervisor toma la decisión de aprobar o rechazar un reporte, se notifica por correo electrónico al usuario que envió dicho reporte.

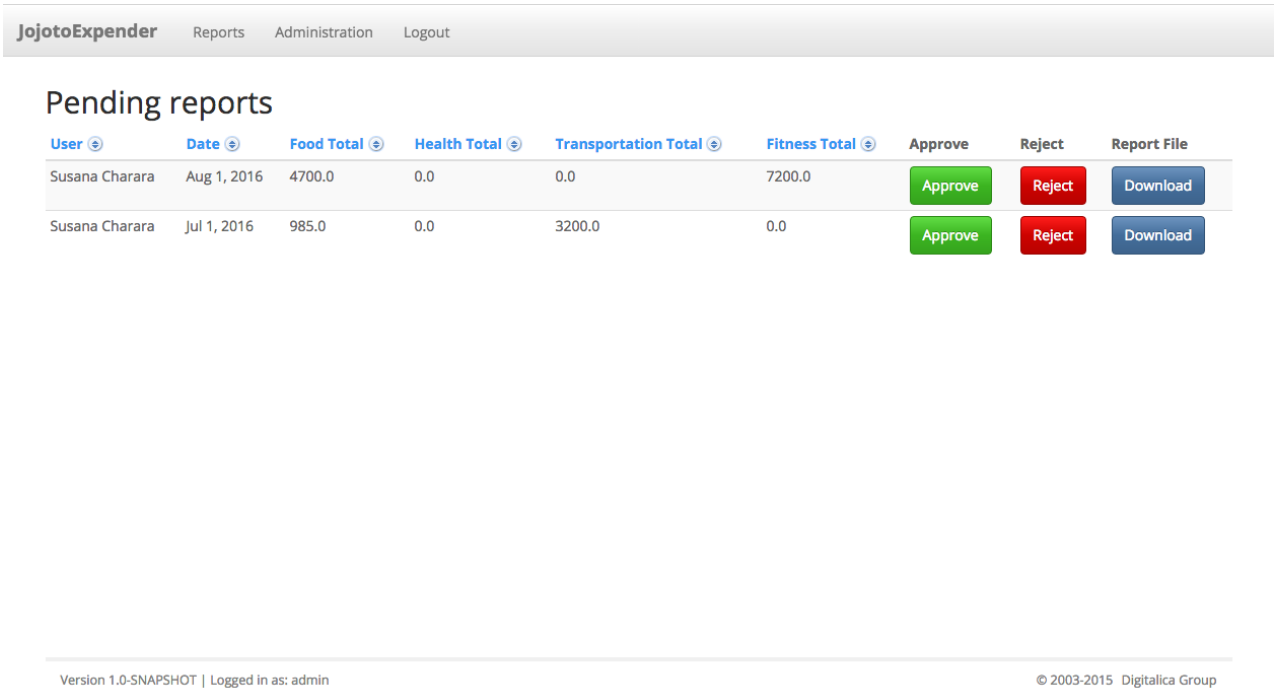

Figura 5.16: Interfaz para listar reportes pendientes de revisión

Además de esto, la herramienta utilizada para el desarrollo del servidor trae consigo algunas funcionalidades por defecto (incluyendo las vistas web necesarias) que permiten iniciar sesión, agregar un usuario, editar un usuario y eliminar un usuario.

El desarrollo del componente servidor se realizó con el IDE Eclipse. Se utilizó el *framework* AppFuse, que unifica a su vez otros *frameworks* que facilitan el desarrollo web. En este caso, se trabajó en conjunto con Spring y Tapestry.

Se utilizó Spring para la implementación de los servicios web que permiten recibir datos desde la aplicación móvil. Estos servicios se implementaron utilizando el patrón Fachada o *Facade* (ver capítulo 2); se implementó una clase *Manager* para exponer los servicios web, que actúa como una fachada para los DAO. Además, AppFuse también cuenta con el *framework* Hibernate para la persistencia de los datos, utilizando como manejador de base de datos MySQL.

Para crear la aplicación web, se utilizó Tapestry. Con Jetty se proveen los servicios, me-

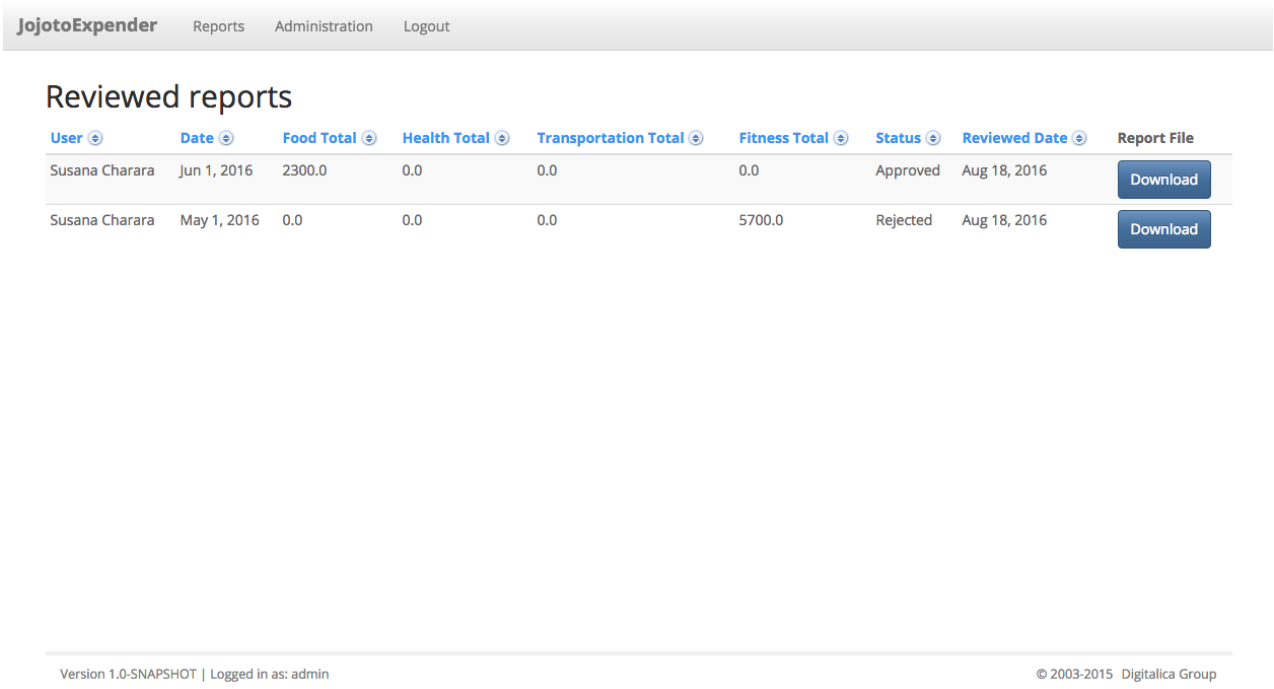

Figura 5.17: Interfaz para listar reportes aprobados/rechazados

diante el protocolo HTTP, que permiten la comunicación con la aplicación web y la aplicación móvil.

Por otra parte, a través de la librería Retrofit, la aplicación móvil se comunica con el servidor web utilizando el protocolo HTTP. La comunicación con la base de datos del dispositivo y con los servicios web se realiza en la capa del presentador. Por su parte, en la vista se maneja toda la interfaz gráfica a través de la cual el usuario interactúa con la aplicación móvil, y toda la lógica necesaria para la comunicación con la capa de modelo y el presentador.

El proyecto se desarrolló bajo el marco de trabajo de *Scrum*, descrito en el capítulo 4. Durante la ejecución de las iteraciones, se implementaron las funcionalidades planificadas. Para cada una de ellas, se realizaron las pruebas necesarias para validarlas (para más información, consultar el plan de pruebas en el apéndice E).

A continuación se presentarán las iteraciones (*sprints*), durante las cuales se desarrollaron las funcionalidades descritas en los párrafos anteriores. Para cada iteración, se mencionarán

sus objetivos y resultados, con una breve descripción de las actividades realizadas.

5.3.1. Iteración 1

5.3.1.1. Objetivos

- Implementar el modelo de datos de la aplicación móvil.
- Crear la vista principal de la aplicación.
- Permitir la creación de un nuevo gasto.

5.3.1.2. Resultados

- Se implementó el modelo de datos de la aplicación móvil, que permite persistir toda la información necesaria para la misma. Se creó la base de datos con la librería SQLite, las tablas necesarias para cumplir con el modelo de datos de la figura 5.4, así como los POJO's correspondientes.
- Se creó la vista principal de la aplicación, a través de la cual el usuario puede consultar de la base de datos el total de ingresos, gastos y balance general. Se creó una interfaz de usuario con tres elementos que muestran dichos montos (figura 5.6).
- Se creó una vista para permitir la creación de un nuevo gasto con su fecha, monto y descripción. Un gasto representa un registro dentro de la tabla correspondiente a la clase *Expense* (figura 5.4). Se creó una interfaz de usuario que incluye los campos necesarios para ingresar la fecha, el monto y la descripción del gasto. Además, se creó una consulta para agregar a la base de datos un nuevo gasto.

5.3.2. Iteración 2

5.3.2.1. Objetivos

- Mostrar la lista de categorías.
- Permitir la creación de un nuevo ingreso.
- Permitir guardar fotos de un ingreso/gasto.
- Asociar un ingreso/gasto a una categoría.

5.3.2.2. Resultados

- Se creó una vista para mostrar la lista de categorías a las que puede pertenecer un gasto. Una categoría es un registro dentro de la tabla *Category* (figura 5.4); en este sentido, la vista permite al usuario consultar de la base de datos del dispositivo los registros correspondientes a la tabla *Category*. Estas categorías pueden ser de dos tipos: ingresos o gastos. Por esta razón, se creó una interfaz que muestra dos listas de categorías, una para cada tipo (figuras 5.10 y 5.10). Para cada categoría, se muestra su nombre y su ícono. Además de esto, se creó una funcionalidad para guardar en la base de datos una lista de categorías por defecto. Esto implicó la creación de una consulta para agregar a la base de datos una nueva categoría, con su nombre y su ícono.
- Se adaptó la vista creada en la iteración 2, a través de la cual se puede crear un gasto, de manera que permitiera la creación de un ingreso. Un ingreso representa un registro dentro de la tabla correspondiente a la clase *Income* (figura 5.4). Como se observó en la figura 5.4, un ingreso y un gasto guardan la misma información; por esta razón, no fue necesario cambiar la interfaz ya creada, sino que se cambió la lógica detrás de ella para identificar si se está usando para crear un ingreso o un gasto.

- Se agregó una funcionalidad para capturar y guardar fotos relacionadas a un ingreso/gasto; esto permite que, por ejemplo, el usuario pueda guardar en el dispositivo móvil fotos de los recibos de los gastos. Para esto se creó un método que permite abrir alguna aplicación instalada en el dispositivo, a través de la cual se puedan capturar y guardar fotos. Una vez guardadas, se inserta dentro de la tabla *Photo* la ruta donde están ubicadas las fotos dentro del dispositivo. Se adaptó la vista para la creación de un ingreso/gasto para mostrar las miniaturas de las fotos tomadas.

- Se agregó una funcionalidad para poder asociar un ingreso/gasto a una categoría existente, lo que permite al usuario especificar a qué rubro pertenece el gasto o ingreso, según sea el caso. Con esta funcionalidad, el usuario puede insertar, dentro de las tablas correspondiente a las clases *Income* y *Expense*, la categoría a la que pertenece dicho ingreso o gasto. Se modificó la vista para la creación de un ingreso/gasto, pues se tuvo que agregar un nuevo elemento a la interfaz, a través del cual se muestran las categorías guardadas en la base de datos, y de las cuales se puede seleccionar una para asociarla al ingreso/gasto.

  Se tenía como requerimiento guardar las cuatro categorías que se utilizaron más recientemente. Para esto, se tuvo que crear los métodos necesarios para persistir en el dispositivo móvil las categorías más recientes. Dado que esta información no tiene una estructura compleja, se decidió utilizar un objeto que provee la plataforma de Android, llamado *SharedPreferences*, que permite guardar un conjunto de pares clave-valor. Este objeto permite persistir las categorías más recientes, aún cuando se cierre la aplicación.

5.3.3. Iteración 3

5.3.3.1. Objetivos

- Mostrar la lista de ingresos/gastos del mes actual.
- Mostrar los detalles de los ingresos/gastos existentes.
- Permitir editar y borrar un ingreso/gasto existente.
- Implementar una calculadora para guardar el monto de un ingreso/gasto.

5.3.3.2. Resultados

- Se implementó una funcionalidad para navegar a la lista de ingresos o de gastos del mes, desde la vista principal de la aplicación. Esta funcionalidad permite al usuario consultar los registros de las tablas que corresponden a las clases *Income* y *Expense*, según sea el caso. Dado que se guarda la misma información para ingresos y gastos, se creó una vista general donde se puede mostrar la lista deseada. Se creó una interfaz que permite mostrar una lista de ingresos/gastos con su descripción (o categoría, en caso de que no tenga descripción), su fecha y su monto. Esta vista se reutilizó para mostrar tanto los ingresos como los gastos, según sea necesario (ver figuras 5.8 y 5.9).
- Se agregó una funcionalidad para ver los detalles de un ingreso/gasto ya existente, y poder editarlo. Con esta funcionalidad, el usuario puede consultar la información relacionada con los ingresos/gastos guardados en la base de datos, así como actualizarla. Para esto no fue necesario crear una interfaz nueva, sino que se reutilizó la que se tenía para la creación de un nuevo ingreso/gasto. Se adaptó la vista correspondiente, de manera que se pueda detectar si se quiere crear un nuevo ingreso/gasto, o si por el contrario se quiere editar uno ya existente. En caso de que exista, se llenan automáticamente los campos de la fecha, monto, categoría, descripción y fotos.

- Se agregó una funcionalidad para eliminar ingresos/gastos. Esto permite al usuario eliminar ingresos y gastos de la base de datos del dispositivo, a través de la vista donde se listan los mismos. Además, se tuvo que realizar cambios a la lógica detrás de la interfaz creada, de manera que se permitiera seleccionar filas de la lista (en este caso las filas corresponden a ingresos o gastos que se deseen eliminar, ver figura A.5).

- Se creó una interfaz para usar una calculadora que permita ingresar el monto de un ingreso/gasto. Se modificó la vista de creación de un ingreso/gasto para incluir la nueva funcionalidad.

  Para este último requerimiento fue necesario crear un teclado virtual personalizado, pues el teclado numérico del dispositivo no incluye los operadores matemáticos básicos. Para esto se creó una interfaz en la que se especificó la disposición de las teclas de la calculadora (ver figura A.11). También se tuvo que manejar el uso de las teclas: dentro de la vista, se creó la lógica para detectar qué tecla fue presionada y realizar una acción en base a esto. Para realizar los cálculos, se utilizó una librería que evalúa fórmulas dada una cadena de caracteres. En este caso, la cadena de caracteres representa las teclas presionadas.

  Para la calculadora, se tiene que mostrar el símbolo decimal de acuerdo con el país para el cual está configurado el teléfono, ya que en algunos países se usa la coma (,) como separador y en otros el punto (.). Por esta razón, se debe obtener el país de configuración del teléfono y, en base a eso, decidir qué símbolo decimal mostrar.

5.3.4. Iteración 4

5.3.4.1. Objetivos

- Permitir el manejo de nuevas cuentas.

5.3.4.2. Resultados

- Se creó una vista para crear una nueva cuenta. Una cuenta representa un registro dentro de la tabla *Account*(figura 5.4). Así, esta vista permite insertar en la base de datos un registro dentro de la tabla *Account*. Se creó una interfaz de usuario, con campos para ingresar el nombre y la moneda en la que estará la nueva cuenta (ver figura A.7). También fue necesario crear una consulta para agregar a la base de datos una cuenta.

- Se implementó una funcionalidad para listar las cuentas del usuario. Esta vista le permite al usuario consultar de la base de datos del dispositivo todos los registros correspondientes a la tabla *Account*. Se creó una interfaz que contiene una lista, y las filas corresponden a cuentas. Para cada cuenta, se muestra su nombre, la moneda y el balance general de la misma. Fue necesario crear una consulta a la base de datos para obtener todas las cuentas guardadas en el dispositivo.

- Se implementó una funcionalidad para navegar entre cuentas, lo que permite cambiar la cuenta que se está mostrando actualmente en el dispositivo. Se creó una barra lateral (*sidebar*) en la vista principal de la aplicación (ver figura A.2) en la que se muestra la lista de cuentas activas en el dispositivo. Para mostrar dicha lista, se realiza una consulta a la base de datos para obtener todos los registros de la tabla *Account* cuyo *status* sea *activa*. De la lista de cuentas que se muestra, se puede seleccionar alguna para que sea la que se esté mostrando actualmente en el dispositivo. La cuenta seleccionada se persistió mediante un *SharedPreferences*, de manera que esta información queda guardada en caso de que se cierre la aplicación.

- Se implementó una funcionalidad para editar el nombre de una cuenta existente. Con esto, se permite al usuario modificar el campo *name* de un registro de la tabla *Account*(figura 5.4). Fue necesario adaptar la vista existente para la creación de una

cuenta, de manera que se permita editar una ya existente. Dentro de la vista, se creó la lógica necesaria para detectar si se está creando una nueva cuenta o si se está editando. En caso de que se esté editando, se llenan los campos del nombre y la moneda. Para persistir los cambios realizados, se creó una consulta a la base de datos que permite actualizar la información de una cuenta.

- Se implementó una funcionalidad para eliminar cuentas existentes. Esta funcionalidad permite al usuario eliminar registros de la tabla *Account*. Se modificó la vista de la lista de cuentas de manera que se puedan seleccionar las cuentas que se desean eliminar (ver figura A.8). También se creó una consulta a la base de datos para eliminar cuentas de la misma.

- Se implementó una funcionalidad para cambiar el estado de una cuenta: activa o archivada; esto permite que el usuario pueda modificar el campo *status* de un registro de la tabla *Account*. Fue necesario la creación de una consulta que permita cambiar el estado de una cuenta en la base de datos. La lista de cuentas se muestra según su estado, es decir, se muestra una lista para la lista de cuentas activas y una para las cuentas archivadas.

- Se implementó una funcionalidad que permite al usuario consultar de la base de datos todos los registros correspondientes tanto a ingresos como a gastos. Hasta ahora, desde la vista principal de la aplicación se podía navegar a la lista de ingresos y la lista de gastos del mes actual, por separado. Se agregó una nueva funcionalidad para navegar a una lista que contenga todos los ingresos y gastos de una cuenta, y que no esté limitada al mes actual. Esto no implicó la modificación de la interfaz, sino de la lógica a través de la cual se puede saber qué lista se está mostrando. Se creó una consulta a la base de datos para obtener una lista con todos los ingresos y gastos de una cuenta, sin estar

limitado a un rango de fechas.

5.3.5. Iteración 5

5.3.5.1. Objetivos

- Permitir el manejo de las fotos de un ingreso/gasto.
- Permitir el manejo de las categorías.

5.3.5.2. Resultados

- Se agregó una funcionalidad para ver las fotos de un ingreso/gasto. Esto permite al usuario hacer una consulta a la base de datos para obtener los registros de la tabla *Photo* asociados a un ingreso/gasto. Se creó una vista que muestra en pantalla completa las fotos de un ingreso/gasto (esto se hace desde la vista de creación de un ingreso/gasto). Esto implicó la creación de la interfaz y la lógica que permita seleccionar una foto, abrirla y mostrarla en pantalla completa (ver figura A.3).
- Se agregó una funcionalidad para borrar las fotos de un ingreso/gasto, a través de la cual el usuario puede realizar una consulta para eliminar de la base de datos registros de la tabla *Photo*, así como eliminar los archivos de las fotos dentro del dispositivo. Se agregó un elemento a la interfaz mencionada en el punto anterior, a través del cual se pueden eliminar las fotos(ver figura A.4).
- Se agregó una funcionalidad para cambiar la ubicación en el dispositivo móvil de las fotos tomadas de un ingreso/gasto. Para esto se creó una vista de configuraciones de la aplicación, a través de la cual se puede cambiar la ubicación donde se guardarán las fotos tomadas. Esto implica cambiar la ubicación de las fotos tomadas anteriormente, y que las nuevas fotos que serán tomadas se guardarán en la nueva ubicación.

Se creó una interfaz donde se muestra la ubicación actual (carpeta) de las fotos. Además, se agregaron dos opciones para cambiar la ubicación en el dispositivo en la que se guardan los archivos de las fotos: memoria interna o memoria extraíble (ver figura A.12). También fue necesario crear la lógica que se encarga de cambiar a la nueva ubicación las fotos tomadas anteriormente.

- Se agregó una vista para crear una categoría. A través de ella, se hace una consulta a la base de datos para insertar un nuevo registro dentro de la tabla *Category*. Se creó una interfaz de usuario con elementos que permiten ingresar el nombre de la categoría y un ícono (ver figura A.9). Las imágenes de los íconos a los cuales se puede asociar una categoría están guardadas por defecto dentro de la aplicación. Se creó un campo a través del cual se muestran los íconos que aún no han sido utilizados, y de los cuales se debe escoger uno. La consulta para agregar una nueva categoría a la base de datos fue creada en la iteración 2, como se mencionó anteriormente.

- Se agregó una funcionalidad para eliminar, a través de una vista, registros de la tabla *Category*. Para esto, se adaptó la interfaz creada en la iteración 2, a través de la cual se muestra la lista de categorías presentes en el dispositivo (ver figura A.10). Esta adaptación se hizo para permitir seleccionar las categorías que se desean eliminar. Se creó una consulta a la base de datos para eliminar de la misma una categoría. También fue necesario crear una consulta para verificar qué ingresos/gastos están asociados a las categorías que se quieren eliminar; a estos ingresos/gastos se les deja sin categoría, y posteriormente se eliminan las categorías deseadas de la base de datos.

- Se agregó una funcionalidad para editar una categoría existente; con ella, se permite al usuario realizar consultas sobre la base de datos del dispositivo para actualizar registros de la tabla *Category*. Se modificó la vista existente para la creación de una nueva

categoría, de manera que soportara también la edición. No se modificó la interfaz gráfica sino que se creó la lógica para verificar si se trata de una nueva categoría o de una ya existente que se quiere editar. En caso de que sea una existente, se muestra la información correspondiente (nombre e ícono). Se creó una consulta a la base de datos para persistir los cambios una vez editada una categoría.

5.3.6. Iteración 6

5.3.6.1. Objetivos

- Mostrar un reporte con la lista de ingresos/gastos asociados a una cuenta, en un rango de fecha dado.
- Mostrar un reporte con el balance total por categorías asociadas a una cuenta, en un rango de fecha dado.
- Permitir el envío de los reportes en un archivo PDF a otras personas.

5.3.6.2. Resultados

- Se implementó una funcionalidad que permite al usuario crear dos tipos de reportes: reporte de balance general y reporte por categoría. El reporte de balance general incluye la lista de todos los ingresos/gastos dentro del rango de fechas establecido. El reporte por categorías incluye únicamente el monto total de todos los ingresos/gastos (es decir, la suma de ingresos y gastos) separados por categorías, en el rango de fechas escogido. Esta funcionalidad permite al usuario realizar consultas a la base de datos para obtener los ingresos/gastos de una cuenta en un rango de fecha determinado, y otra lista con el balance total de cada categoría de una cuenta en dicho rango.

  Se creó una única vista para mostrar el reporte de balance general y el reporte por

categorías. Esto implicó la creación de una interfaz para mostrar la lista de ingresos/-gastos o de categorías, según sea el caso. Se agregó además un elemento que permite cambiar de un reporte a otro. Se creó la lógica necesaria para poder identificar qué tipo de reporte se debe mostrar (ver figuras 5.14 y 5.15).

- Se creó una funcionalidad para generar un archivo PDF con la información mostrada en los reportes mencionados en el punto anterior. Para esto se usó una librería que provee la plataforma de Android para la creación de archivos PDF.

  Se implementaron los métodos necesarios para crear un archivo PDF con el reporte del balance general. Esta funcionalidad luego se adaptó para permitir también la creación del reporte por categorías.

  Se tenía como requerimiento que el usuario pueda escoger generar un archivo con el reporte de balance general incluyendo las fotos de los ingresos/gastos asociados, o sin incluirlas. Por esta razón, se creó la lógica necesaria para incluir dentro del archivo PDF las fotos.

- Se creó una funcionalidad para compartir un reporte (de cualquiera de los dos tipos) con otras personas; esto se realiza a través de otras aplicaciones instaladas en el dispositivo móvil que permitan enviar archivos con formato PDF. Dentro de estas aplicaciones se incluyen las de correo electrónico y mensajería móvil que soporten el envío de archivos. Esto se hizo a través de una funcionalidad del sistema operativo Android, que permite mostrar las aplicaciones instaladas en el dispositivo móvil, a través de las cuales se pueden mandar archivos PDF.

5.3.7. Iteración 7

5.3.7.1. Objetivos

- Crear un servicio web para permitir la autenticación de un usuario.
- Crear un servicio web para recibir un reporte de gastos.
- Crear una aplicación web para mostrar los reportes recibidos.
- Permitir la aprobación o rechazo a través de la aplicación web de los reportes recibidos.

5.3.7.2. Resultados

- Se creó el proyecto en AppFuse, con Tapestry como *framework* para el desarrollo de la aplicación web. Se crearon los POJO's y, utilizando Hibernate con MySQL, se crearon las tablas necesarias para implementar el modelo de datos mostrado en la figura 5.5. Como se mencionó anteriormente, AppFuse crea por defecto una base de datos y las tablas *User*, *Role* y *UserRole*. La tabla *UserRole* fue utilizada para especificar qué usuarios son administradores, por lo que únicamente fue necesario implementar la clase *Report* del modelo de datos del servidor.
- Se creó un servicio web al cual la aplicación móvil se puede conectar para la autenticación de usuarios. Se utilizó Spring para crear el servicio, el cual recibe un archivo JSON con el nombre de usuario y una contraseña, y verifica en la base de datos del servidor esta información. Posteriormente, envía una respuesta a la aplicación indicando si las credenciales son válidas o no. Fue necesario crear una clase DAO y otra *Manager* (descritas en el capítulo 2) que permiten acceder a la base de datos para la verificación. A través de una librería de Java, se permitió exponer como servicio web el método creado en el *Manager* para la autenticación. También se implementó en la aplicación móvil

una funcionalidad que permite iniciar sesión en el dispositivo. Para esto fue necesario crear una vista con una interfaz de usuario con campos para ingresar el nombre y la contraseña (ver figura A.1). Se crearon los métodos necesarios para enviar un archivo JSON con el nombre de usuario y la contraseña al servicio web para la autenticación. Se utilizó la librería Retrofit (ver capítulo 3) para hacer la conexión mediante el protocolo HTTP, utilizando una petición POST.

- Se creó un servicio web a través del cual se reciben archivos PDF de reportes de gastos, lo que permite que la aplicación móvil pueda enviar al servidor reportes de gastos. Los archivos de estos reportes son guardados dentro del servidor, y dentro de la base de datos se guarda la ubicación de los mismos. Al igual que con la autenticación de usuario, fue necesario crear un DAO y un *Manager* que permiten guardar en la base de datos los reportes recibidos. Es necesario enviar tanto el archivo PDF con la información de los gastos, como un resumen de los totales por las categorías de interés para la empresa. Por esta razón, la comunicación entre la aplicación móvil y el servidor para el envío de reportes se hizo a través de una petición HTTP POST Multipart, que contiene el archivo PDF y un JSON con el resumen mencionado anteriormente.

- Se creó una aplicación web a través de la cual se permite listar, aprobar y rechazar reportes recibidos por el servidor. Mediante la aplicación web, los supervisores pueden hacer una consulta a la base de datos del servidor para obtener los reportes que han sido enviados. Además, a través de la misma, también se permite aprobar y rechazar reportes; esto quiere decir que utilizando la aplicación web, un supervisor puede realizar consultas para modificar el estado (aprobado o rechazado) de un reporte. Para esto, se creó una vista y un controlador que permitan el manejo de las acciones de usuario.

  Se creó una vista que muestra la lista de reportes pendientes por revisión (es decir, que aún no han sido aprobados ni rechazados). Se creó una interfaz de usuario que

muestra una lista con los reportes recibidos, y botones que permiten aprobar/rechazar un reporte (ver figura 5.16). En caso de que se decida rechazar, se le muestra una ventana emergente con un campo para ingresar la razón por la cual se tomó la decisión.

Finalmente, se adaptó la vista mencionada anteriormente de manera que se mostrara una lista de los reportes ya revisados (es decir, que ya fueron aprobados o rechazados). Esto no implicó hacer cambios en la interfaz sino en el controlador para detectar qué lista se desea mostrar. Se agregó también un botón que permite descargar el archivo PDF de cada reporte(ver figura 5.17).

5.3.8. Iteración 8

5.3.8.1. Objetivos

- Notificar por correo electrónico a los supervisores cuando se recibe un nuevo reporte.
- Notificar por correo electrónico al usuario cuando su reporte ha sido aprobado o rechazado.
- Filtrar por palabras clave la lista de ingresos/gastos en la aplicación móvil.

5.3.8.2. Resultados

- Se creó un módulo en el servidor para enviar notificaciones vía correo electrónico. Se crearon las clases y métodos necesarios para el envío de notificaciones mediante correos electrónicos. Se utilizó una clase que provee el *framework* AppFuse que facilita el envío de correos, a través de un servidor SMTP.

  Este módulo se utiliza para enviar un correo electrónico a los supervisores cuando un nuevo reporte llega al servidor; en este correo se adjunta el archivo PDF del reporte.

Asimismo, también se envía un correo electrónico al usuario correspondiente una vez su reporte ha sido aprobado o rechazado.

- Se creó una nueva funcionalidad en la aplicación móvil para permitir la búsqueda (filtrado) de gastos/ingresos por palabras clave (ver figura A.6). Esto se hizo únicamente para las vistas de las figuras 5.8 y 5.9. Para esto se creó un nuevo elemento a la interfaz, a través del cual se permite ingresar una palabra clave para realizar la búsqueda.

Capítulo 6

Conclusiones y Recomendaciones

En este proyecto de pasantía, se desarrolló una versión alfa de una aplicación móvil y un servidor web que permiten agilizar el proceso de reembolso de gastos de los trabajadores de la empresa Digitalica Group C.A. La aplicación fue desarrollada para el sistema operativo Android. Con ella, se permite registrar gastos, a los cuales se puede asociar un monto, fecha, descripción, fotos y categoría. El servidor web permite hacer la autenticación de usuarios, así como el envío de reportes compuestos por un conjunto de gastos. Igualmente, se desarrolló una aplicación web que permite a los supervisores de la empresa revisar, aprobar y rechazar reportes. También se implementaron funcionalidades dirigidas al público en general, y no únicamente a la empresa. Estas funcionalidades incluyen el manejo de ingresos, manejo de múltiples cuentas de gastos/ingresos y manejo de categorías de gastos/ingresos.

Se estudiaron conceptos que facilitaron el desarrollo de la solución propuesta al problema planteado. Utilizando los patrones de arquitectura MVC en el servidor y MVP en la aplicación móvil, se logró hacer una separación clara de las interfaces de usuario de los modelos de datos y controladores/presentadores. Con ello, se permite introducir cambios en algún componente sin afectar los demás.

El uso de Scrum como marco de trabajo permitió desarrollar el proyecto progresivamente,

y tener una parte del producto funcional luego de cada iteración. Esto fue de gran ventaja porque le facilitaba al dueño del producto (*Product Owner*) evaluar constantemente las funcionalidades y verificar que cumplieran con las necesidades del producto. Además, las reuniones diarias (*Daily Scrum*) le permitía a todo el equipo de trabajo estar informado de la evolución del producto, además de ser una oportunidad en que el equipo de desarrollo (en este caso integrado únicamente por el pasante) pudiera solventar cualquier duda o impedimento que surgiera.

Durante el desarrollo del proyecto de pasantía, se identificaron funcionalidades que pudieran ser añadidas o mejoradas, que se explicarán en los siguientes párrafos.

Actualmente, el servidor tiene únicamente información de los reportes que son enviados por la aplicación móvil, lo cual es un subconjunto muy limitado de todos los datos presentes en el dispositivo; toda la información es guardada localmente en el dispositivo móvil. Esto implica que si un usuario guarda su información en un dispositivo, y posteriormente inicia sesión en otro dispositivo, no podrá ver desde este último la información que había guardado anteriormente; para poder acceder a sus datos, debe iniciar sesión en el dispositivo desde donde los guardó. Se recomienda enviar al servidor todos los datos que almacena la aplicación móvil en los dispositivos, para permitir que un usuario pueda iniciar sesión y consultar sus datos en distintos dispositivos. De esta manera, al iniciar sesión en la aplicación móvil, se piden al servidor y se muestran los datos del usuario correspondiente.

Para la creación del archivo PDF se presentó una dificultad. Las librerías existentes en Java para generar archivos con dicha extensión hacen uso de otras librerías, que no están soportadas directamente en la plataforma de Android. Por esta razón, se decidió utilizar una librería nativa de Android que facilita la creación de archivos PDF; sin embargo, esta librería sólo puede ser utilizada por dispositivos que tengan una versión de Android a partir de la 4.4. Esta decisión se apoyó en el hecho de que más del 80 % de los usuarios de Android utilizan

una versión mayor o igual a 4.4, como se puede ver en el cuadro 6.1 (datos obtenidos del sitio web oficial para desarrolladores de Android) [42].

Versión	**Nombre**	**API**	**Uso**
2.2	Froyo	8	0.1 %
2.3.x	Gingergread	15	1.5 %
4.0.x	Ice Cream Sandwich	15	1.4 %
4.1.x	Jelly Bean	16	5.6 %
4.2.x		17	7.7 %
4.3		18	2.3 %
4.4	KitKat	19	27.7 %
5.0	Lollipop	21	13.1 %
5.1		22	21.9 %
6.0	Marshmallow	23	18.7 %

Cuadro 6.1: Porcentaje de uso de las versiones de Android. No se muestran versiones con menos de 0.1 % de distribución

Durante el proceso de investigación, se encontraron librerías que permiten la generación de archivos PDF para todas las versiones de Android. Sin embargo, se debe pagar para obtener una licencia comercial. Si se piensa lanzar al mercado la aplicación, se recomienda a la empresa evaluar si dejar de ofrecer la funcionalidad a un porcentaje de usuarios generará mayores pérdidas que la adquisición de una licencia comercial de alguno de los productos que pueda solucionar el problema.

Bibliografía

[1] "Digitalica — the technology partner of your disruptive ideas." http://digitalicagroup.com/, consultado el 30 de julio de 2016.

[2] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Dessign Patterns. Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1 ed.

[3] R. Fielding, *Architectural Styles and the Design of Network-based Software Architectures*. 2000.

[4] D. Garlan and M. Shaw, *An introduction to Software Engineer*. World Scientific, 1 ed., 1993.

[5] "Modelo-vista-controlador." https://msdn.microsoft.com/en-us/library/ff649643.aspx, consultado el 18 de agosto de 2016.

[6] "Mvc or mvp pattern - what's the difference?." http://www.codeproject.com/Articles/288928/Differences-between-MVC-and-MVP-for-Beginners, consultado el 18 de agosto de 2016.

[7] "Differences between mvc and mvp for beginners." http://www.codeproject.com/Articles/288928/Differences-between-MVC-and-MVP-for-Beginners, consultado el 10 de julio de 2016.

[8] "Tipos de arquitectura cliente servidor." https://www.ibiblio.org/pub/Linux/docs/LuCaS/Manuales-LuCAS/doc-curso-salamanca-LAMP/lamp-teoria-html/ch01s02.html, consultado el 2 de agosto de 2016.

[9] "What are web services." http://www.tutorialspoint.com/webservices/what_are_web_services.htm, consultado el 19 de agosto de 2016.

[10] "Http made really easy." http://www.jmarshall.com/easy/http/#whatis, consultado el 15 de agosto de 2016.

[11] "Rfc1341(mime): 7 the multipart content type." https://www.w3.org/Protocols/rfc1341/7_2_Multipart.html, consultado el 2 de agosto de 2016.

[12] "Http methods get vs post." http://www.w3schools.com/tags/ref_httpmethods.asp, consultado el 15 de agosto de 2016.

[13] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Architectural Styles and the Design of Network-based Software Architectures*. Doctoral dissertation, University of California, Irvine, 2000.

[14] "Json." http://www.json.org/, consultado el 9 de agosto de 2016.

[15] "Pojo." http://www.martinfowler.com/bliki/POJO.html, consultado el 17 de agosto de 2016.

[16] "Understanding pojo." https://spring.io/understanding/POJO, consultado el 17 de agosto de 2016.

[17] "Core j2ee patterns - data access object." http://www.oracle.com/technetwork/java/dataaccessobject-138824.html, consultado el 17 de agosto de 2016.

[18] "Services - appfuse - appfuse confluence." http://appfuse.org/display/APF/Services, consultado el 9 de agosto de 2016.

[19] "What is android os?." https://www.techopedia.com/definition/14873/android-os, consultado el 30 de julio de 2016.

[20] "Platform architecture — android developers." https://developer.android.com/guide/platform/index.html, consultado el 23 de agosto de 2016.

[21] "Activity — android developers." https://developer.android.com/reference/android/app/Activity.html, consultado el 23 de agosto de 2016.

[22] "Android interfaces and architecture — android open source project." https://source.android.com/devices/, consultado el 22 de agosto de 2016.

[23] "What is java? definition and meaning." http://www.businessdictionary.com/definition/Java.html, consultado el 26 de julio de 2016.

[24] "Application fundamentals — android developers." https://developer.android.com/guide/components/fundamentals.html, consultado el 23 de agosto de 2016.

[25] "What is mysql?." http://searchenterpriselinux.techtarget.com/definition/MySQL, consultado el 26 de julio de 2016.

[26] "About sqlite." https://sqlite.org/about.html, consultado el 30 de julio de 2016.

[27] "Maven - introduction." https://maven.apache.org/what-is-maven.html, consultado el 30 de julio de 2016.

[28] "Jetty - servlet engine and http server." http://www.eclipse.org/jetty/, consultado el 30 de julio de 2016.

[29] "Meet android studio — android studio." https://developer.android.com/studio/index.html, consultado el 19 de agosto de 2016.

[30] "Eclipse - the eclipse foundation open source community website." http://www.eclipse.org/home, consultado el 30 de julio de 2016.

[31] S. Chacon and B. Straub, *Pro Git: Everything you need to know about Git.* Apress, 2 ed., 2014.

[32] "Home - appfuse - appfuse concluence." http://appfuse.org/display/APF/Home, consultado el 26 de julio de 2016.

[33] "Hibernate orm." http://hibernate.org/orm/, consultado el 19 de agosto de 2016.

[34] "Spring." https://spring.io/, consultado el 25 de agosto de 2016.

[35] "Apache tapestry home page." http://tapestry.apache.org/index.html, consultado el 30 de julio de 2016.

[36] "Github - google/gson." https://github.com/google/gson, consultado el 26 de julio de 2016.

[37] "Retrofit." http://square.github.io/retrofit/, consultado el 30 de julio de 2016.

[38] K. Rubin, *Essential Scrum: A practical guide to the most popular agile process.* Addison-Wesley Professional, 1 ed., 2012.

[39] "Scrum basics." http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-ES.pdf, consultado el 7 de julio de 2016.

[40] "Artefactos en scrum: claves para una organización diaria." http://www.desarrolloweb.com/articulos/artefactos-scrum.html, consultado el 7 de julio de 2016.

[41] "Camera — android developers." https://developer.android.com/guide/topics/media/camera.html, consultado el 17 de agosto de 2016.

[42] "Dashboards — android developers." https://developer.android.com/about/dashboards/index.html?hl=es, consultado el 29 de julio de 2016.

Glosario

Android: Sistema operativo de código abierto basado en el núcleo de Linux, utilizado pricipalmente en dispositivos móviles.

***Application Programming Interface* (API):** Conjunto de funciones y protocolos que ofrece una librería como capa de abstracción, para ser utilizados por otro *software*.

Aplicación móvil: *Software* diseñado para ser ejecutado en dispositivos móviles, como teléfonos inteligentes y tabletas.

Aplicación web: Herramientas que los usuarios pueden utilizar para acceder a un servidor web a través de Internet.

Cliente: *Software* o usuario que realiza peticiones de tareas a otros ordenadores que actúan como servidores.

***Framework*:** Estructura conceptual y tecnológica que sirve de base para la organización e implementación de *software*.

***Hypertext Markup Language* (HTML):** Lenguaje utilizado para la elaboración de pági-

nas web.

***Integrated Development Environment* (IDE):** Aplicación informática que provee servicios que facilitan el desarrollo de *software* al usuario.

Librería: Conjunto de funciones, desarrolladas en un lenguaje de programación, que ofrecen una interfaz definida para la funcionalidad que se invoca.

Manejador de base de datos: Colección de *software* que sirve de interfaz entre la base de datos, el usuario y las aplicaciones utilizadas.

***Object-Relational Maping* (ORM):** Técnica de programación utilizada para convertir datos entre sistemas de tipos incompatibles en lenguajes orientados a objetos.

***Portable Dcument Format* (PDF):** Formato de almacenamiento de documentos digitales intependiente de plataformas de *software* y *hardware*.

Scrum: Marco de desarrollo ágil que se caracteriza por adoptar una estrategia de desarrollo incremental e iterativa.

***Software Development Kit* (SDK):** Conjunto de herramientas de desarrollo de *software* que le permiten al programador crear aplicaciones para un sistema en concreto.

Servidor: Computador en el que se ejecuta continuamente un *Software* que realiza tareas para atender peticiones de un cliente.

Servidor web: Computador en el que se ejecuta continuamente un *Software*, al cual se hacen peticiones a través de Internet.

***Simple Mail Transfer Protocol* (SMTP):** Protocolo de red que se utiliza para el intercambio de mensajes de correo electrónico entre dispositivos.

***Structured Query language* (SQL):** Lenguaje declarativo que permite realizar operaciones sobre bases de datos relacionales.

***Uniform Resource Locator* (URL):** Secuencia de caracteres que sigue un estándar y permite denominar recursos dentro del entorno de Internet para que pueden ser localizados.

Apéndice A

Capturas de pantalla de la aplicación móvil

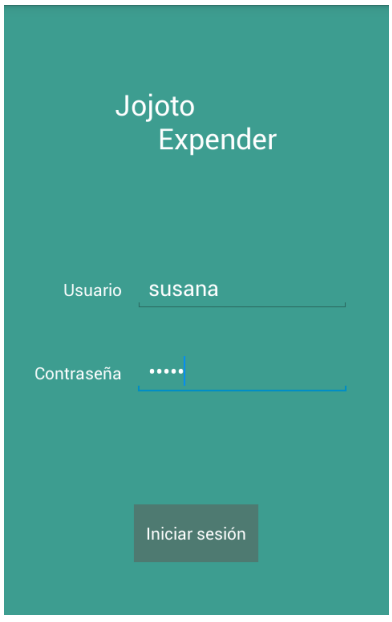

Figura A.1: Interfaz para iniciar sesión

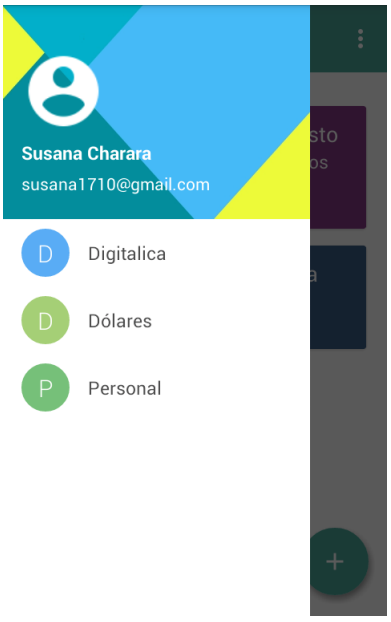

Figura A.2: Interfaz para cambiar de cuenta

Figura A.3: Interfaz para ver fotos en pantalla completa

Figura A.4: Interfaz para eliminar fotos

Figura A.5: Interfaz para eliminar gastos

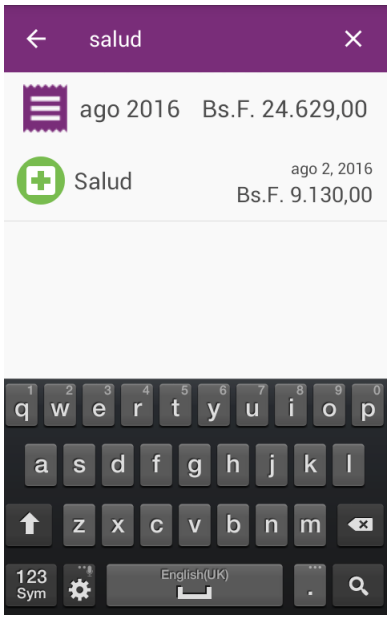

Figura A.6: Interfaz para filtrar gastos

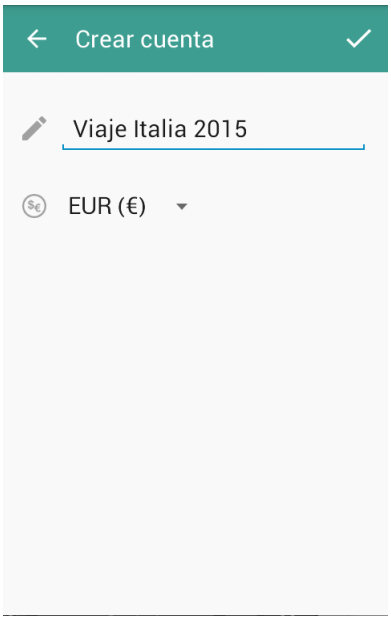

Figura A.7: Interfaz para crear una cuenta

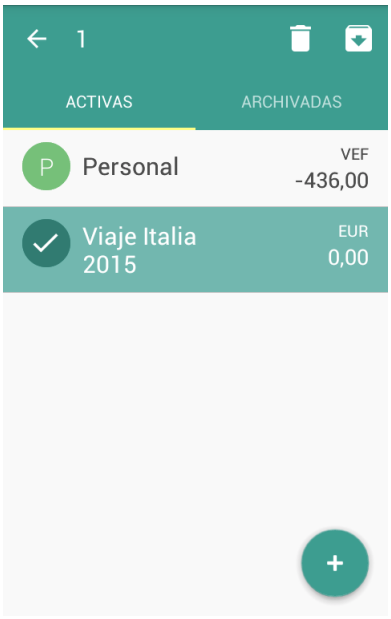

Figura A.8: Interfaz para eliminar cuentas

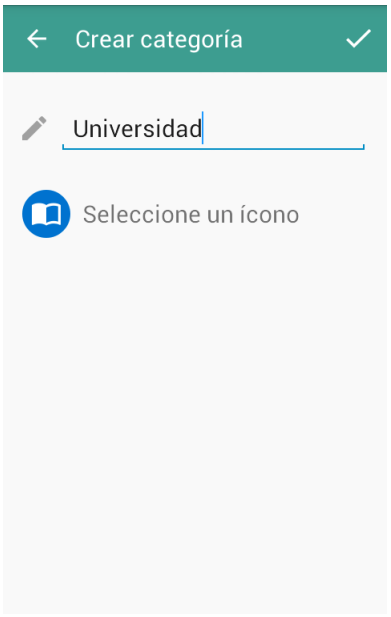

Figura A.9: Interfaz para crear una categoría

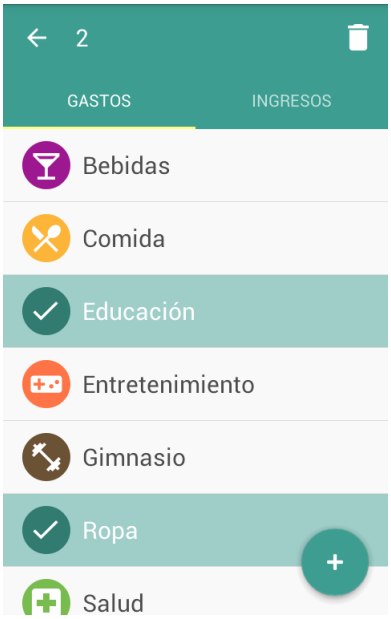

Figura A.10: Interfaz para eliminar categorías

Figura A.11: Interfaz de la calculadora

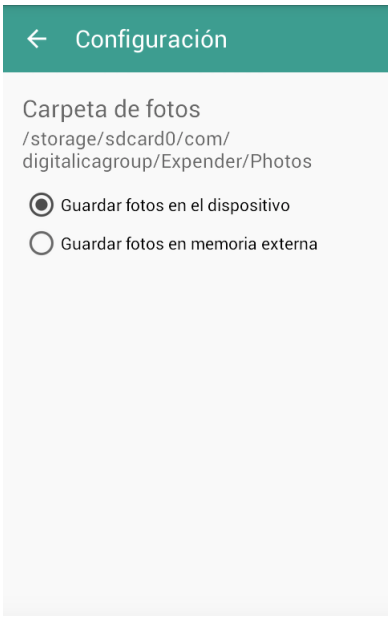

Figura A.12: Interfaz para cambiar la ubicación de las fotos

Apéndice B

Capturas de pantalla de la aplicación web

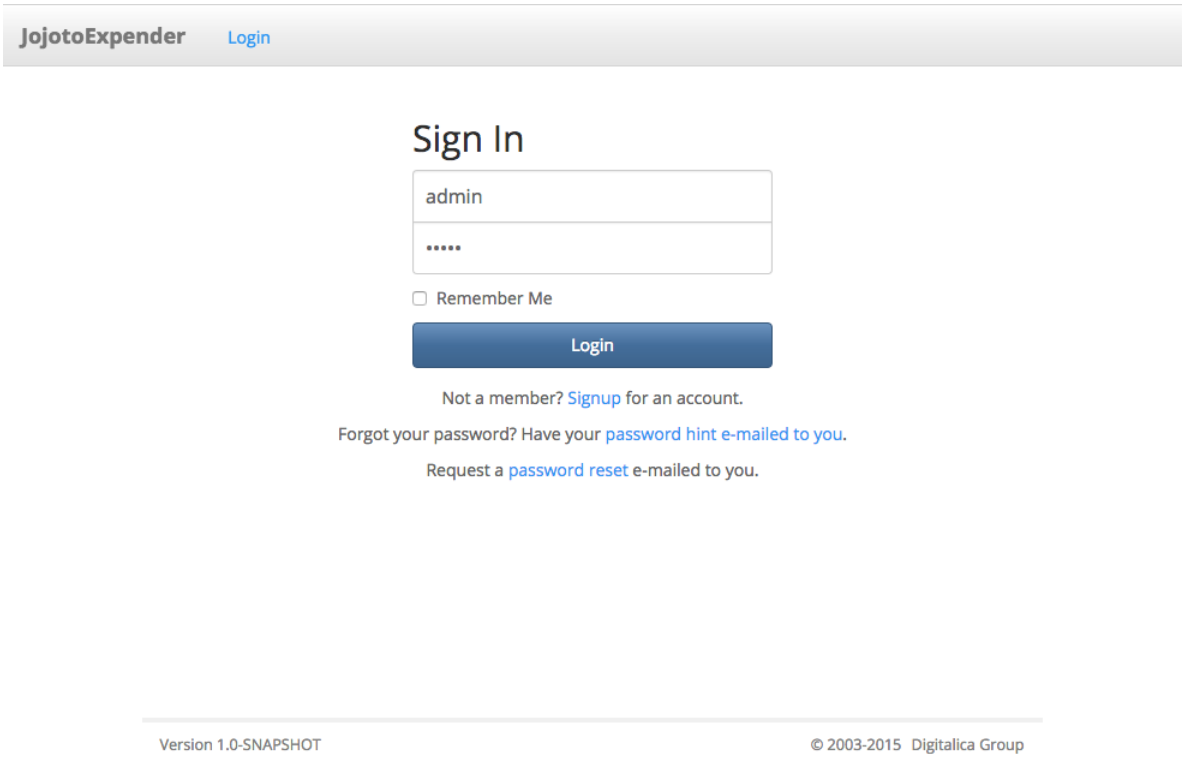

Figura B.1: Interfaz para iniciar sesión en la aplicación web

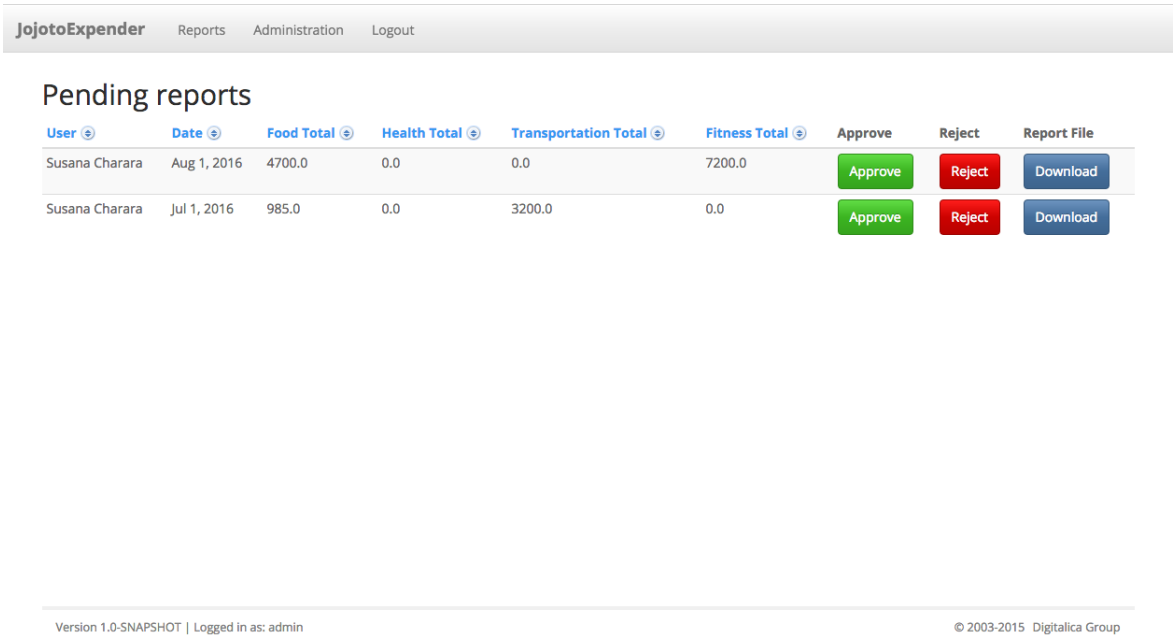

Figura B.2: Interfaz para listar reportes pendientes de revisión

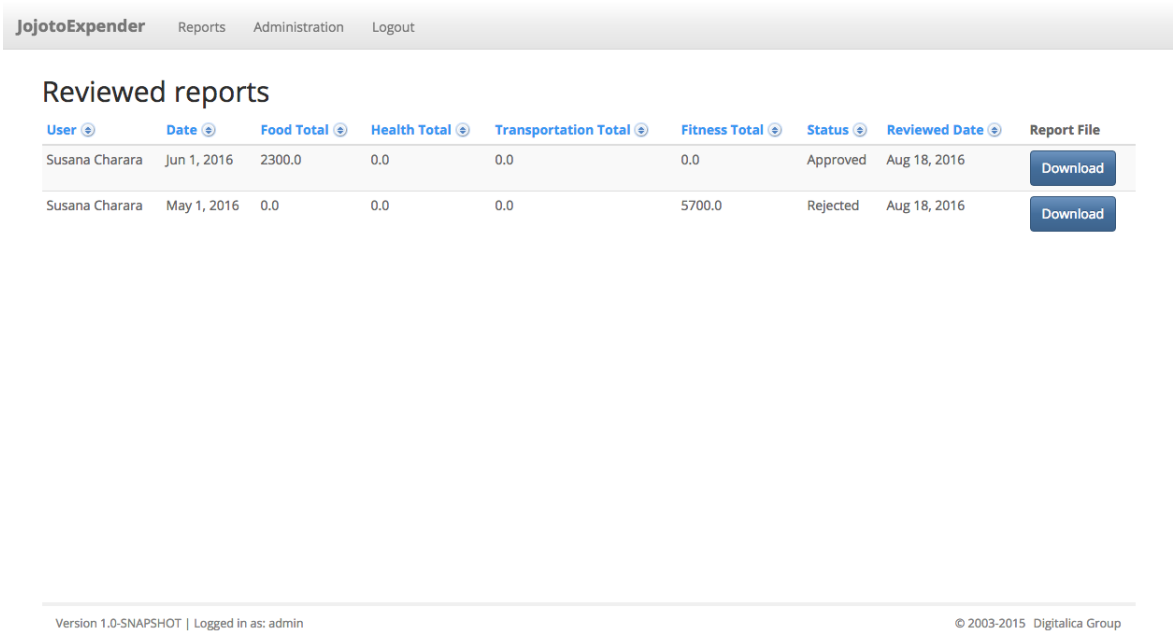

Figura B.3: Interfaz para listar reportes aprobados/rechazados

Apéndice C

Documento de Historias de Usuario (*Product Backlog*)

- Como USUARIO, quiero ver el Dashboard.
- Como USUARIO, quiero crear un gasto.
- Como USUARIO, quiero crear un ingreso.
- Como USUARIO, quiero tomar fotos de un ingreso o gasto.
- Como USUARIO, quiero listar mis categorías.
- Como USUARIO, quiero seleccionar una categoría al crear/editar un ingreso/gasto.
- Como USUARIO, quiero ver la lista de ingresos/gastos del mes actual.
- Como USUARIO, quiero ver los detalles de un ingreso/gasto en específico.
- Como USUARIO, quiero editar un ingreso/gasto.
- Como USUARIO, quiero eliminar ingresos/gastos.

- Como USUARIO, quiero tener una calculadora integrada.
- Como USUARIO, quiero ver mi lista de cuentas.
- Como USUARIO, quiero crear una nueva cuenta.
- Como USUARIO, quiero navegar entre cuentas.
- Como USUARIO, quiero editar mis cuentas.
- Como USUARIO, quiero eliminar cuentas.
- Como USUARIO, quiero archivar una cuenta.
- Como USUARIO, quiero desarchivar una cuenta.
- Como USUARIO, quiero ver la lista de ingresos/gastos de una cuenta.
- Como USUARIO, quiero ver las fotos de un ingreso/gasto.
- Como USUARIO, quiero eliminar una foto.
- Como USUARIO, quiero crear una nueva categoría.
- Como USUARIO, quiero eliminar una categoría.
- Como USUARIO, quiero editar una categoría.
- Como USUARIO, quiero cambiar la carpeta de mis fotos.
- Como USUARIO, quiero ver mi reporte de balance general.
- Como USUARIO, quiero ver mi reporte por categoría.
- Como USUARIO, quiero compartir mi reporte como un archivo PDF.

- Como USUARIO, quiero iniciar sesión con mis credenciales.
- Como USUARIO, quiero recibir reportes de gastos de un cliente.
- Como USUARIO, quiero ver la lista de reportes pendientes.
- Como USUARIO, quiero aprobar un reporte de gastos.
- Como USUARIO, quiero rechazar un reporte de gastos.
- Como USUARIO, quiero ver una lista de todos los reportes aprobados y rechazados.
- Como USUARIO, quiero descargar los detalles de un reporte de gastos en un archivo PDF.
- Como USUARIO, quiero filtrar las listas de ingresos/gastos por palabras claves.

Apéndice D

Artefactos complementarios del sistema de creación y control de reportes

D.1. Diagrama de componentes del sistema

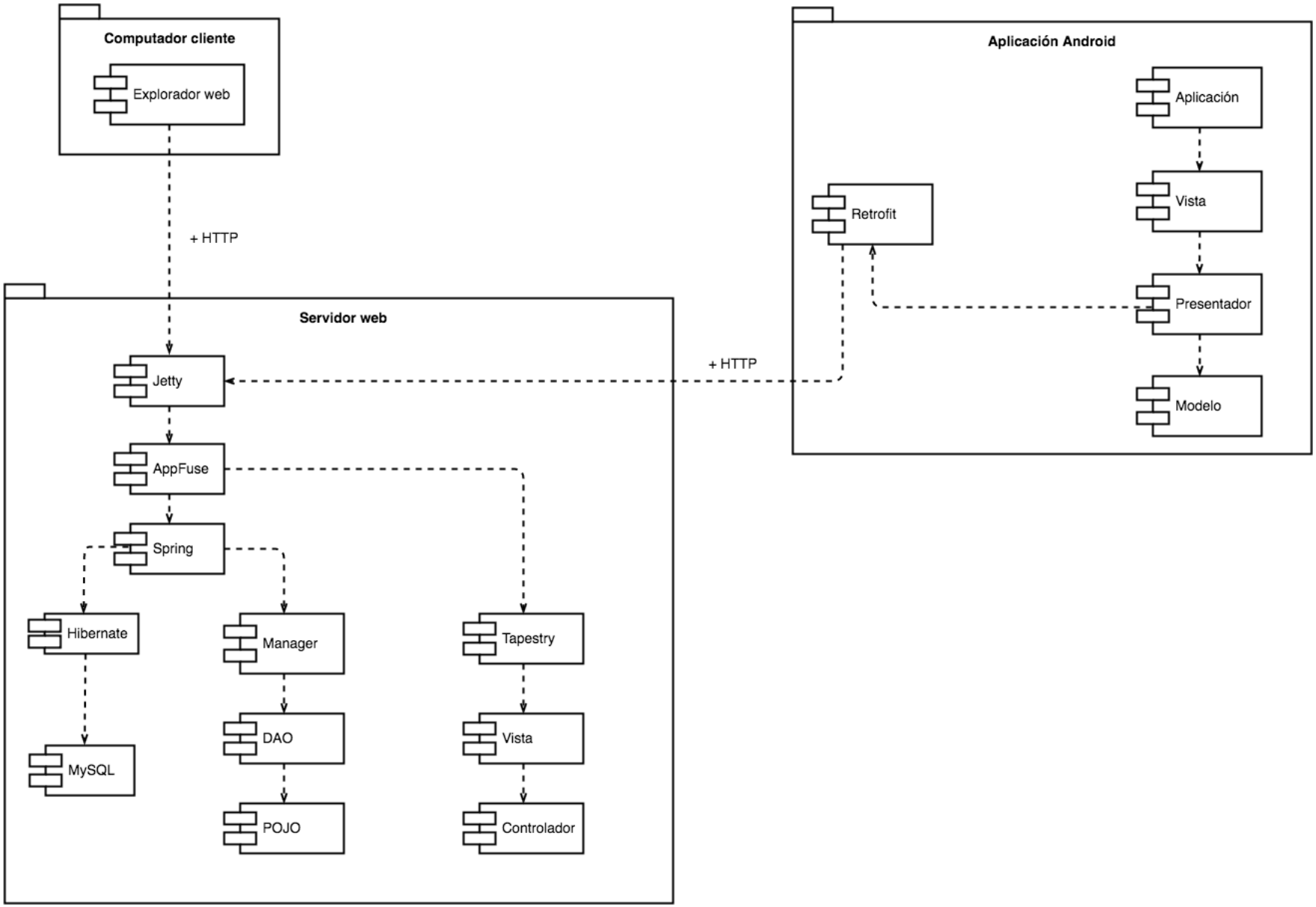

Figura D.1: Diagrama de componentes del sistema

D.2. Diagramas de actividades, estados y secuencia

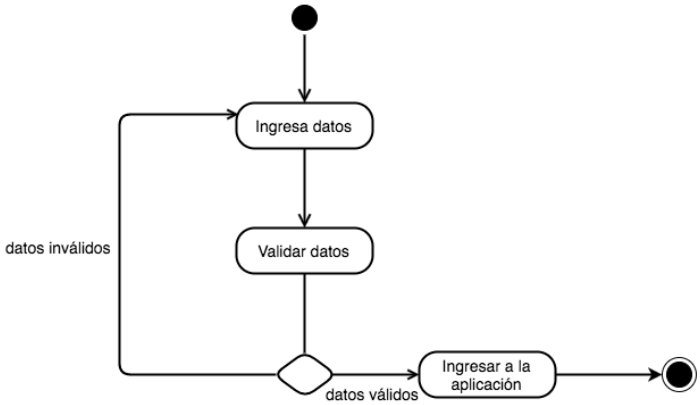

Figura D.2: Diagrama de actividades del proceso de inicio de sesión en la aplicación móvil

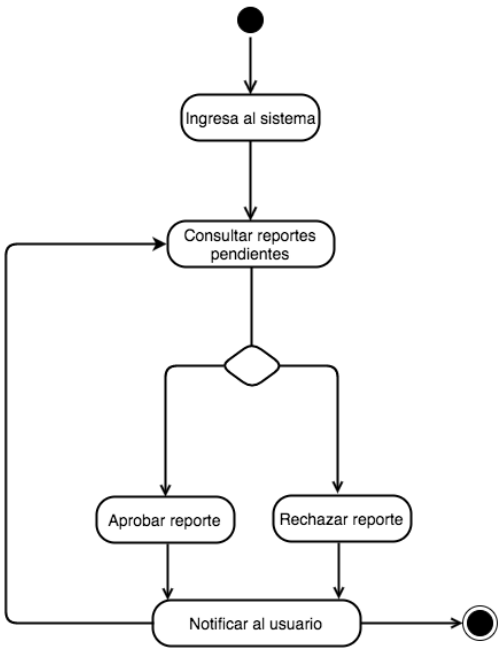

Figura D.3: Diagrama de actividades del proceso de revisión(aprobar/rechazar) de un reporte

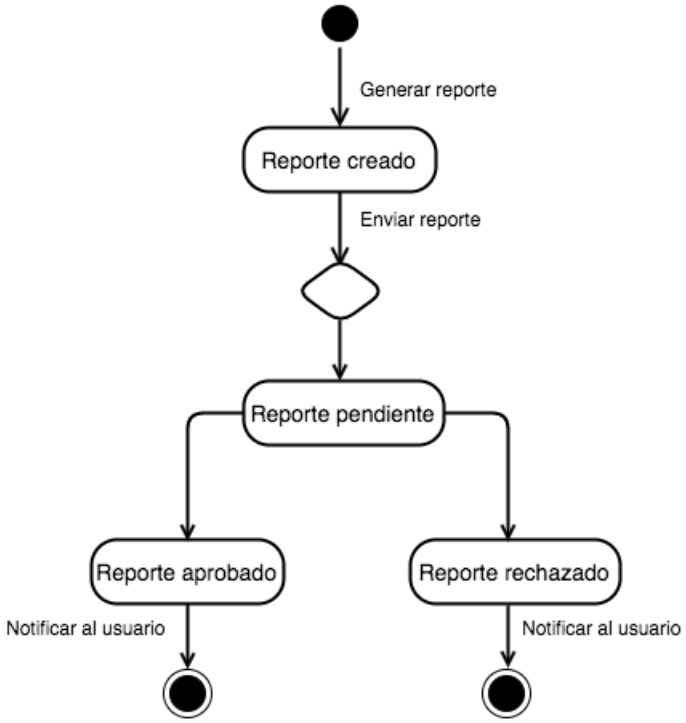

Figura D.4: Diagrama de estados de un reporte

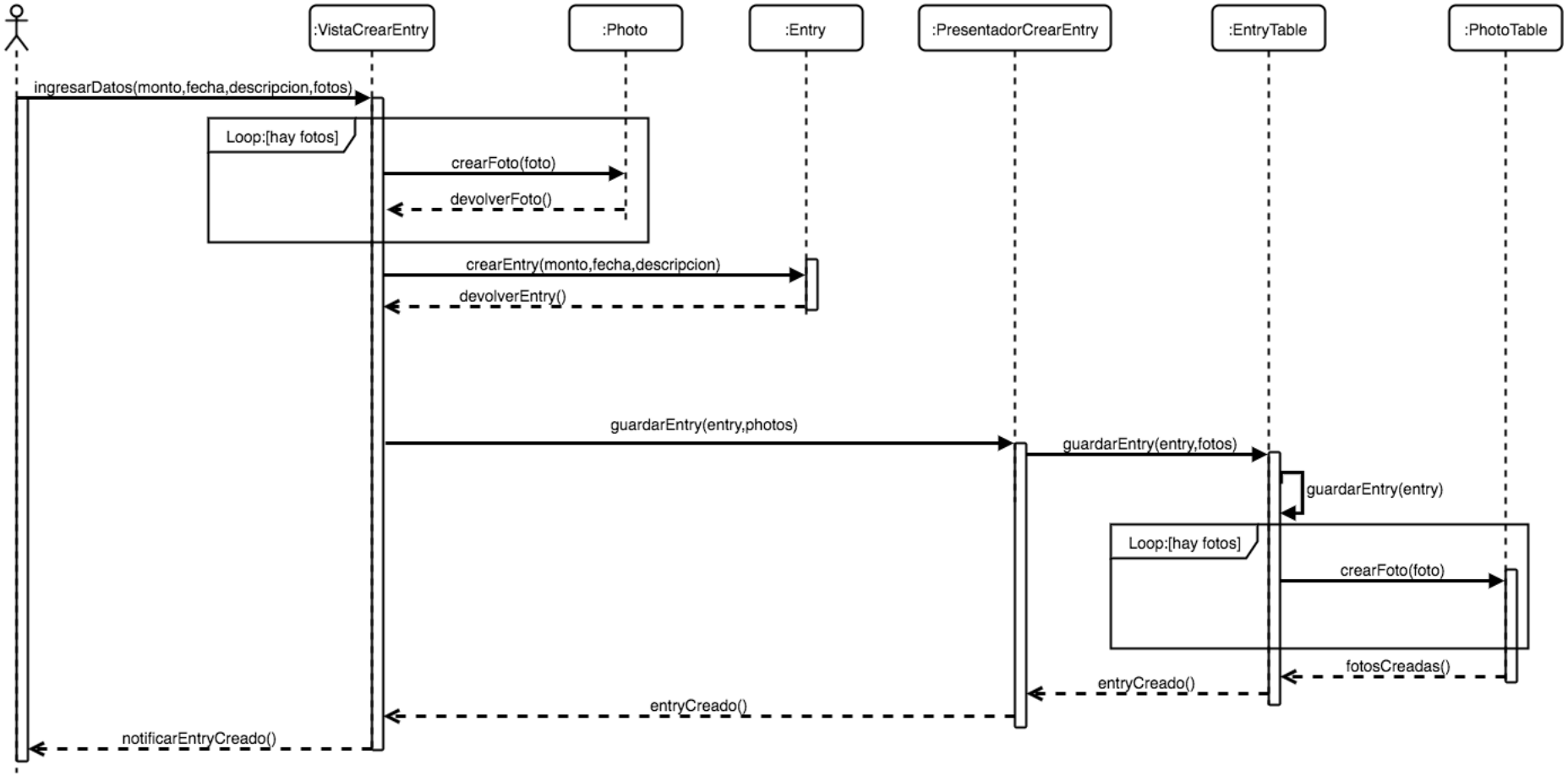

Figura D.5: Diagrama de secuencia del proceso de creación de un entry (ingreso/gasto)

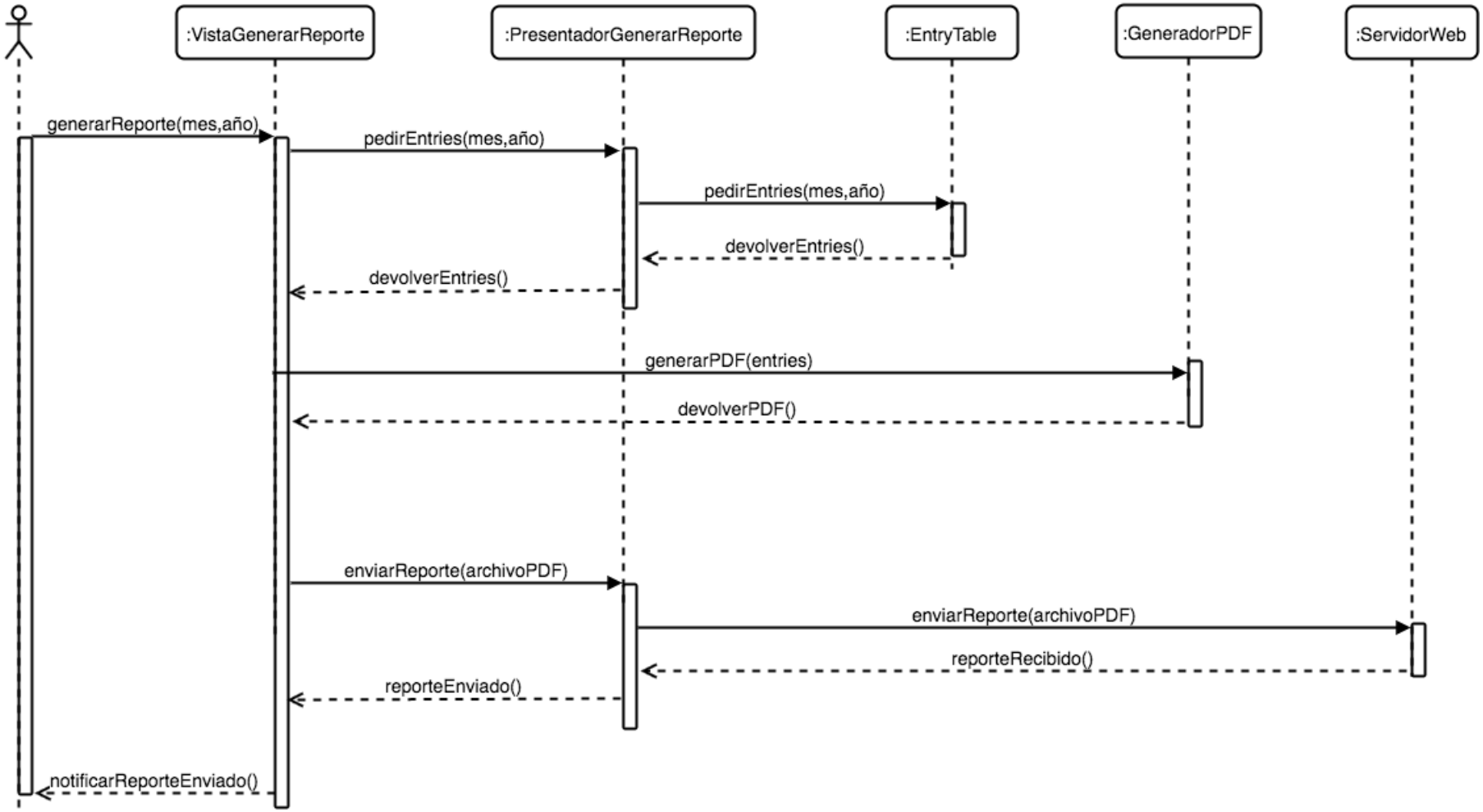

Figura D.6: Diagrama de secuencia del proceso de generación de un reporte

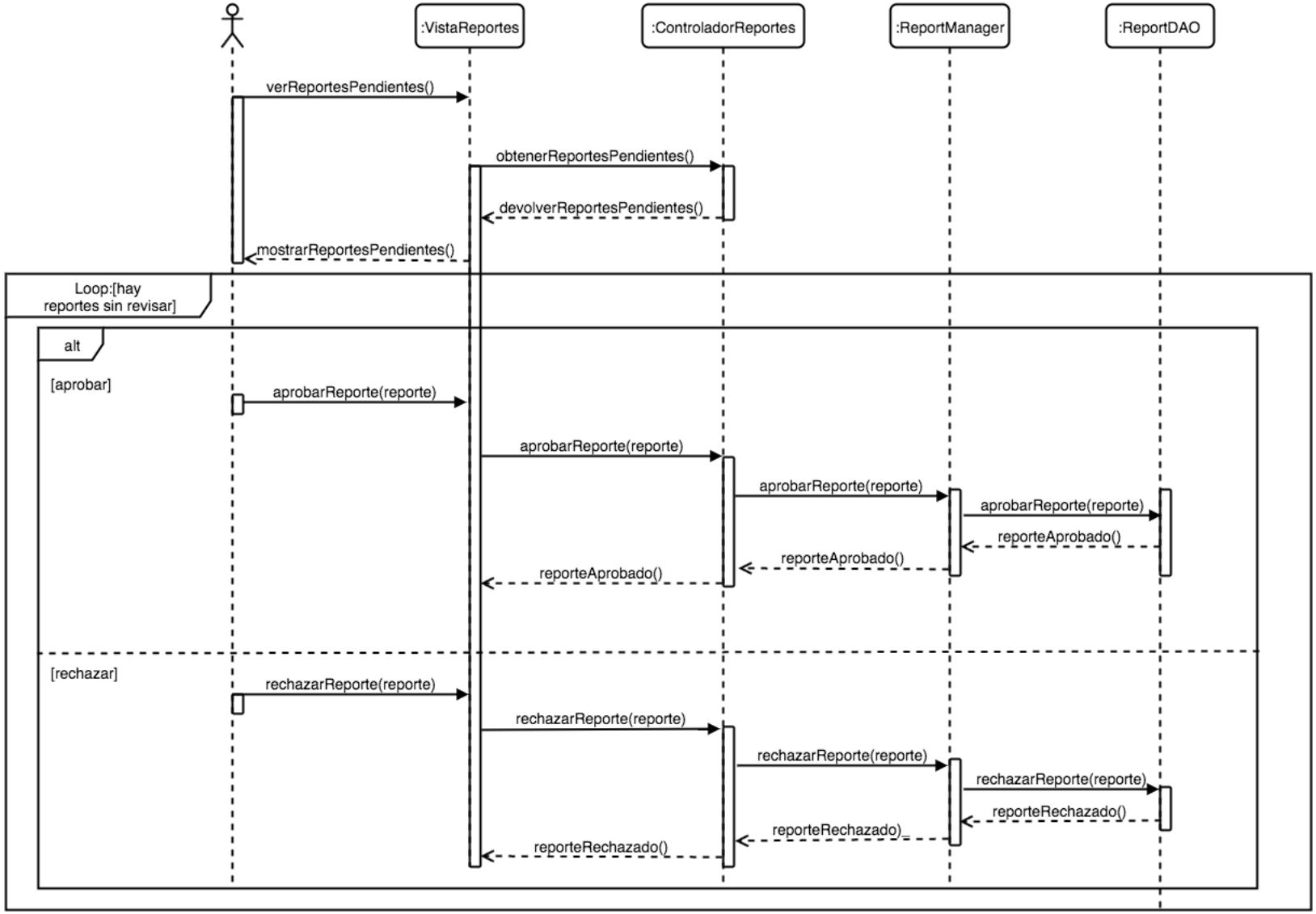

Figura D.7: Diagrama de secuencia del proceso de revisión(aprobar/rechazar) de un reporte

Apéndice E

Plan de pruebas

Casos de pruebas realizadas a la aplicación de control y reportes de gastos

Historia de Usuario:
Como USUARIO, quiero ver el Dashboard.

ID Caso de Prueba:
JJ-45

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil	La aplicación móvil muestra la vista principal de la aplicación, con el total de gastos, ingresos y balance general.	La aplicación móvil muestra la vista principal de la aplicación, con el total de gastos, ingresos y balance general.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero crear un gasto.

ID Caso de Prueba:
JJ-56

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar opción *Crear nuevo gasto*	La aplicación móvil muestra una vista con un formulario para crear un nuevo gasto. También se muestra un botón para guardar. Al presionarlo, se agrega el gasto.	La aplicación móvil muestra una vista con un formulario para crear un nuevo gasto. También se muestra un botón para guardar. Al presionarlo, se agrega el gasto.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero crear un ingreso.

ID Caso de Prueba:
JJ-86

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar opción *Crear nuevo ingreso*	La aplicación móvil muestra una vista con un formulario para crear un nuevo ingreso. También se muestra un botón para guardar. Al presionarlo, se agrega el ingreso.	La aplicación móvil muestra una vista con un formulario para crear un nuevo ingreso. También se muestra un botón para guardar. Al presionarlo, se agrega el ingreso

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero tomar fotos de un ingreso o gasto.

ID Caso de Prueba:
JJ-87

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar alguna de estas dos opciones: *Crear nuevo gasto* o *Crear nuevo ingreso*	La aplicación móvil muestra una vista con un formulario para crear un nuevo ingreso/gasto, con un botón para tomar una foto. Al presionarlo, se abre la cámara para poder capturar una foto.	La aplicación móvil muestra una vista con un formulario para crear un nuevo ingreso/gasto, con un botón para tomar una foto. Al presionarlo, se abre la cámara para poder capturar una foto.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero listar mis categorías.

ID Caso de Prueba:
JJ-93

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar opción del menú *Categorías*	La aplicación móvil muestra una vista con la lista de categorías guardadas.	La aplicación móvil muestra una vista con la lista de categorías guardadas.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero seleccionar una categoría al crear/editar un ingreso/gasto.

ID Caso de Prueba:
JJ-98

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar alguna de estas dos opciones: *Crear nuevo gasto* o *Crear nuevo ingreso.*	La aplicación móvil muestra una vista con un formulario para crear un nuevo gasto/ingreso, con un botón para seleccionar una categoría. Al presionarlo, se muestra una lista con las categorías presentes.	La aplicación móvil muestra una vista con un formulario para crear un nuevo gasto/ingreso, con un botón para seleccionar una categoría. Al presionarlo, se muestra una lista con las categorías presentes.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero ver la lista de ingresos/gastos del mes actual.

ID Caso de Prueba:
JJ-134

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar alguno de estos dos elementos de la vista principal: *Ingresos* o *Gastos.*	La aplicación móvil muestra una vista con la lista de ingresos/gastos, según sea el caso, del mes actual.	La aplicación móvil muestra una vista con la lista de ingresos/gastos, según sea el caso, del mes actual.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero ver los detalles de un ingreso/gasto en específico.

ID Caso de Prueba:
JJ-135

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar alguno de estos dos elementos de la vista principal: *Ingresos* o *Gastos.* - Seleccionar algún elemento de la lista.	La aplicación móvil muestra una vista con un formulario donde se muestran los datos del ingreso/gasto correspondiente.	La aplicación móvil muestra una vista con un formulario donde se muestran los datos del ingreso/gasto correspondiente.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero editar un ingreso/gasto.

ID Caso de Prueba:
JJ-136

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar alguno de estos dos elementos de la vista principal: *Ingresos* o *Gastos.* - Seleccionar algún elemento de la lista. - Editar los campos del formulario - Presionar botón para guardar	La aplicación móvil muestra una vista con un formulario donde se muestran los datos del ingreso/gasto correspondiente. Además, también hay un botón para guardar. Al presionarlo, se guardan los cambios hechos.	La aplicación móvil muestra una vista con un formulario donde se muestran los datos del ingreso/gasto correspondiente. Además, también hay un botón para guardar. Al presionarlo, se guardan los cambios hechos.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero eliminar ingresos/gastos.

ID Caso de Prueba:
JJ-137

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar alguno de estos dos elementos de la vista principal: *Ingresos* o *Gastos.* - Seleccionar los íconos correspondientes a los ingresos/gastos que se quieren eliminar. - Presionar botón para eliminar	La aplicación móvil muestra una vista con la lista de ingresos/gastos, según sea el caso. Al seleccionar algún ícono correspondiente a estos ingresos/gastos, se muestra un botón para eliminar. Al presionar este botón, se eliminan los ingresos/gastos correspondientes.	La aplicación móvil muestra una vista con la lista de ingresos/gastos, según sea el caso. Al seleccionar algún ícono correspondiente a estos ingresos/gastos, se muestra un botón para eliminar. Al presionar este botón, se eliminan los ingresos/gastos correspondientes.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero tener una calculadora integrada.

ID Caso de Prueba:
JJ-149

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar alguna de estas dos opciones: *Crear un nuevo gasto* o *Crear un nuevo ingreso* - Seleccionar el campo para ingresar el monto	La aplicación móvil muestra una vista con un formulario para crear un nuevo gasto/ingreso, según sea el caso. Dentro de este formulario se incluye un elemento para ingresar el monto. Al presionar este elemento, se muestra una calculadora.	La aplicación móvil muestra una vista con un formulario para crear un nuevo gasto/ingreso, según sea el caso. Dentro de este formulario se incluye un elemento para ingresar el monto. Al presionar este elemento, se muestra una calculadora.

Decisión de aprobación del Caso de Prueba :
Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)

Historia de Usuario:
Como USUARIO, quiero ver mi lista de cuentas.

ID Caso de Prueba:
JJ-195

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil	La aplicación móvil muestra una	La aplicación móvil muestra una

- Seleccionar del menú la opción *Administrar cuentas.*	vista con la lista de cuentas.	vista con la lista de cuentas.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero crear una nueva cuenta.		
ID Caso de Prueba: JJ-145		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar del menú la opción *Administrar cuentas* - Presionar botón para agregar una nueva cuenta	La aplicación móvil muestra una vista con un formulario para crear una nueva cuenta. También muestra un botón para guardar. Al presionarlo, se guarda la nueva cuenta.	La aplicación móvil muestra una vista con un formulario para crear una nueva cuenta. También muestra un botón para guardar. Al presionarlo, se guarda la nueva cuenta.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero navegar entre cuentas.		
ID Caso de Prueba: JJ-146		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar el botón para abrir la barra lateral - Seleccionar la cuenta a la que se desea navegar	La aplicación móvil muestra en la vista principal un elemento para mostrar una barra lateral, en la cual se listan las cuentas. Al seleccionar alguna cuenta de esta lista, se cambia la cuenta actual a la seleccionada.	La aplicación móvil muestra en la vista principal un elemento para mostrar una barra lateral, en la cual se listan las cuentas. Al seleccionar alguna cuenta de esta lista, se cambia la cuenta actual a la seleccionada.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero editar mis cuentas.		
ID Caso de Prueba: JJ-147		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**

- Ingresar a la aplicación móvil - Seleccionar la opción del menú *Administrar cuentas* - Seleccionar algún elemento de la lista	La aplicación móvil muestra una vista con la lista de cuentas presentes. Al seleccionar algún elemento de esta lista, se muestra un formulario con los datos de dicha cuenta. Además del formulario, se muestra un botón para guardar. Al presionar el botón, se guardan los cambios hechos.	La aplicación móvil muestra una vista con la lista de cuentas presentes. Al seleccionar algún elemento de esta lista, se muestra un formulario con los datos de dicha cuenta. Además del formulario, se muestra un botón para guardar. Al presionar el botón, se guardan los cambios hechos.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero eliminar cuentas.		
ID Caso de Prueba: JJ-148		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar la opción del menú *Administrar cuentas* - Seleccionar los íconos de las cuentas que se desean eliminar - Presionar botón para eliminar	La aplicación muestra móvil una vista con la lista de cuentas. Al seleccionar algún ícono de alguna cuenta, se muestra un botón para eliminar. Al presionar este botón, se eliminan las cuentas seleccionadas.	La aplicación móvil muestra una vista con la lista de cuentas. Al seleccionar algún ícono de alguna cuenta, se muestra un botón para eliminar. Al presionar este botón, se eliminan las cuentas seleccionadas.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero archivar una cuenta.		
ID Caso de Prueba: JJ-188		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar la opción del menú *Administrar cuentas* - Seleccionar los íconos de las cuentas que se desean archivar - Presionar botón para archivar	La aplicación móvil muestra una vista con la lista de cuentas. Al seleccionar algún ícono de alguna cuenta, se muestra un botón para archivar. Al presionar este botón, se archivan las cuentas seleccionadas.	La aplicación móvil muestra una vista con la lista de cuentas. Al seleccionar algún ícono de alguna cuenta, se muestra un botón para archivar. Al presionar este botón, se archivan las cuentas seleccionadas.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario:

Como USUARIO, quiero desarchivar una cuenta.		
ID Caso de Prueba: JJ-189		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar la opción del menú *Administrar cuentas* - Seleccionar los íconos de las cuentas que se desean desarchivar - Presionar botón para desarchivar	La aplicación móvil muestra una vista con la lista de cuentas. Al seleccionar algún ícono de alguna cuenta, se muestra un botón para desarchivar. Al presionar este botón, se desarchivan las cuentas seleccionadas.	La aplicación móvil muestra una vista con la lista de cuentas. Al seleccionar algún ícono de alguna cuenta, se muestra un botón para desarchivar. Al presionar este botón, se desarchivan las cuentas seleccionadas.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero ver la lista de ingresos/gastos de una cuenta.		
ID Caso de Prueba: JJ-190		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar el elemento de *Balance general* de la vista principal	La aplicación móvil muestra una lista con los ingresos y gastos de la cuenta.	La aplicación móvil muestra una lista con los ingresos y gastos de la cuenta.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero ver las fotos de un ingreso/gasto.		
ID Caso de Prueba: JJ-232		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar alguno de estos tres elementos de la vista principal: *Ingresos, Gastos* o *Balance general.* - Seleccionar algún elemento de la lista - Seleccionar alguna de las fotos	La aplicación móvil muestra una vista con una lista de ingresos/gastos, o ambos, según sea el caso. Al seleccionar algún elemento de esta lista, se muestran los detalles, incluyendo las fotos. Al seleccionar alguna foto, ésta se muestra en pantalla completa.	La aplicación móvil muestra una vista con una lista de ingresos/gastos, o ambos, según sea el caso. Al seleccionar algún elemento de esta lista, se muestran los detalles, incluyendo las fotos. Al seleccionar alguna foto, ésta se muestra en pantalla completa.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero eliminar una foto.		
ID Caso de Prueba: JJ-235		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar alguno de estos tres elementos de la vista principal: *Ingresos, Gastos* o *Balance general.* - Seleccionar algún elemento de la lista - Seleccionar alguna de las fotos - Seleccionar opción para eliminar la foto	La aplicación móvil muestra una vista con una lista de ingresos/gastos, o ambos, según sea el caso. Al seleccionar algún elemento de esta lista, se muestran los detalles, incluyendo las fotos. Al seleccionar alguna foto, ésta se muestra en pantalla completa. También se muestra una botón para eliminar dicha foto. Al presionarlo, se elimina la foto correspondiente.	La aplicación móvil muestra una vista con una lista de ingresos/gastos, o ambos, según sea el caso. Al seleccionar algún elemento de esta lista, se muestran los detalles, incluyendo las fotos. Al seleccionar alguna foto, ésta se muestra en pantalla completa. También se muestra una botón para eliminar dicha foto. Al presionarlo, se elimina la foto correspondiente.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero crear una nueva categoría.		
ID Caso de Prueba: JJ-239		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar opción *Categorías* - Seleccionar botón para agregar	La aplicación móvil muestra una vista con un formulario para crear una nueva categoría. Además, se muestra un botón para guardar. Al presionarlo, se guarda la categoría.	La aplicación móvil muestra una vista con un formulario para crear una nueva categoría. Además, se muestra un botón para guardar. Al presionarlo, se guarda la categoría.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero eliminar una categoría.		
ID Caso de Prueba: JJ-249		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación	La aplicación muestra una vista con	La aplicación muestra una vista con

- Seleccionar opción *Categorías* - Seleccionar los íconos correspondientes a las categorías que se desean eliminar. - Presionar botón para eliminar.	la lista de categorías. Al seleccionar el ícono de alguna categoría, se muestra un botón para eliminar. Al presionar este botón, se eliminan las categorías seleccionadas.	la lista de categorías. Al seleccionar el ícono de alguna categoría, se muestra un botón para eliminar. Al presionar este botón, se eliminan las categorías seleccionadas.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero editar una categoría.		
ID Caso de Prueba: JJ-250		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar opción *Categorías* - Seleccionar alguna categoría de la lista - Cambiar los campos deseados - Presionar botón para guardar	La aplicación móvil muestra una vista con un formulario, con los datos de la categoría seleccionada . Además, se muestra un botón para guardar. Al presionarlo, se guardan los cambios hechos.	La aplicación móvil muestra una vista con un formulario, con los datos de la categoría seleccionada . Además, se muestra un botón para guardar. Al presionarlo, se guardan los cambios hechos.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero cambiar la carpeta de mis fotos.		
ID Caso de Prueba: JJ-253		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar opción *Configuración* - Seleccionar la ubicación que se desea que tenga la carpeta de fotos	La aplicación móvil muestra una vista que muestra la ubicación actual de la carpeta de fotos. Además, muestra dos opciones de ubicación: memoria interna y memoria externa del dispositivo. Al seleccionar alguna de estas opciones, se cambian las fotos a la nueva ubicación y las fotos tomadas en lo siguiente se guardan en dicha ubicación.	La aplicación móvil muestra una vista que muestra la ubicación actual de la carpeta de fotos. Además, muestra dos opciones de ubicación: memoria interna y memoria externa del dispositivo. Al seleccionar alguna de estas opciones, se cambian las fotos a la nueva ubicación y las fotos tomadas en lo siguiente se guardan en dicha ubicación.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero ver mi reporte de balance general.		
ID Caso de Prueba: JJ-280		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar opción *Reporte*	La aplicación móvil muestra una vista con una lista de ingresos y gastos. Además, muestra dos campos para seleccionar el rango de fechas del reporte. Al ingresar las dos fechas del rango, se muestra una lista con los ingresos y gastos que están en ese rango de fechas.	La aplicación móvil muestra una vista con una lista de ingresos y gastos. Además, muestra dos campos para seleccionar el rango de fechas del reporte. Al ingresar las dos fechas del rango, se muestra una lista con los ingresos y gastos que están en ese rango de fechas.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero ver mi reporte por categoría.		
ID Caso de Prueba: JJ-281		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar opción *Reporte* - Seleccionar opción para cambiar de reporte	La aplicación móvil muestra una vista con el reporte de balance general. Además, muestra un botón para cambiar el tipo de reporte a reporte por categoría. Al presionarlo, se muestra el reporte por categoría para el rango de fechas especificado.	La aplicación móvil muestra una vista con el reporte de balance general. Además, muestra un botón para cambiar el tipo de reporte a reporte por categoría. Al presionarlo, se muestra el reporte por categoría para el rango de fechas especificado.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero compartir mi reporte como un archivo PDF.		
ID Caso de Prueba: JJ-284		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Seleccionar opción *Reporte* - Seleccionar alguno de los dos tipos de reporte	La aplicación móvil muestra una vista con el reporte de balance general/por categoría, según sea el caso. Se muestra un botón para	La aplicación móvil muestra una vista con el reporte de balance general/por categoría, según sea el caso. Se muestra un botón para

- Seleccionar opción para compartir - Seleccionar la aplicación a través de la cual se quiere compartir el reporte.	compartir. Al seleccionarlo, se muestra una lista de aplicaciones a través de las cuales se puede compartir el reporte. Al seleccionar una de las aplicaciones, se comparte el reporte.	compartir. Al seleccionarlo, se muestra una lista de aplicaciones a través de las cuales se puede compartir el reporte. Al seleccionar una de las aplicaciones, se comparte el reporte.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero iniciar sesión con mis credenciales.		
ID Caso de Prueba: JJ-332		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil	La aplicación móvil muestra una vista con un formulario para iniciar sesión. Al ingresar los datos, éstos se validan con el servidor. Si son correctos, se muestra la vista principal de la aplicación. De lo contrario, se muestra un mensaje de error.	La aplicación móvil muestra una vista con un formulario para iniciar sesión. Al ingresar los datos, éstos se validan con el servidor. Si son correctos, se muestra la vista principal de la aplicación. De lo contrario, se muestra un mensaje de error.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero recibir reportes de gastos de un cliente.		
ID Caso de Prueba: JJ-340		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación móvil - Iniciar sesión, de ser necesario - Seleccionar opción *Reporte* - Seleccionar opción para enviar reporte mensual - Seleccionar mes y año para el que se desea generar el reporte - Presionar enviar	La aplicación móvil genera el reporte correspondiente al mes y año, y luego lo envía al servidor. En el servidor se guarda el nuevo reporte recibido.	La aplicación móvil genera el reporte correspondiente al mes y año, y luego lo envía al servidor. En el servidor se guarda el nuevo reporte recibido.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero ver la lista de reportes pendientes.		
ID Caso de Prueba: JJ-335		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación web - Iniciar sesión como administrador - Seleccionar opción *Reportes pendientes*	La aplicación web muestra una vista con la lista de reportes que han sido recibidos y que no han sido revisados.	La aplicación web muestra una vista con la lista de reportes que han sido recibidos y que no han sido revisados.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero aprobar un reporte de gastos.		
ID Caso de Prueba: JJ-337		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación web - Iniciar sesión como administrador - Seleccionar opción *Reportes pendientes* - Seleccionar la opción *Aprobar* para el reporte que se quiere aprobar	La aplicación web muestra una vista con la lista de reportes que han sido recibidos y que no han sido revisados. Para cada uno de estos reportes se muestra una opción para aprobar y otra para rechazar. Al presionar el botón de aprobar, se elimina el reporte de la lista de pendientes y se marca como aprobado.	La aplicación web muestra una vista con la lista de reportes que han sido recibidos y que no han sido revisados. Para cada uno de estos reportes se muestra una opción para aprobar y otra para rechazar. Al presionar el botón de aprobar, se elimina el reporte de la lista de pendientes y se marca como aprobado.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero rechazar un reporte de gastos.		
ID Caso de Prueba: JJ-338		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
Ingresar a la aplicación web - Iniciar sesión como administrador - Seleccionar opción *Reportes pendientes* - Seleccionar la opción *Rechazar* para el reporte que se quiere	La aplicación web muestra una vista con la lista de reportes que han sido recibidos y que no han sido revisados. Para cada uno de estos reportes se muestra una opción para aprobar y otra para	La aplicación web muestra una vista con la lista de reportes que han sido recibidos y que no han sido revisados. Para cada uno de estos reportes se muestra una opción para aprobar y otra para

rechazar	rechazar. Al presionar el botón de rechazar, se elimina el reporte de la lista de pendientes y se marca como rechazado.	rechazar. Al presionar el botón de rechazar, se elimina el reporte de la lista de pendientes y se marca como rechazado.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero ver una lista de todos los reportes aprobados y rechazados.		
ID Caso de Prueba: JJ-339		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación web - Iniciar sesión como administrador - Seleccionar opción *Reportes revisados*	La aplicación muestra una lista con los reportes que ya han sido aprobados y rechazados.	La aplicación muestra una lista con los reportes que ya han sido aprobados y rechazados.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historial de Usuario: Como USUARIO, quiero descargar los detalles de un reporte de gastos en un archivo PDF.		
ID Caso de Prueba: JJ-349		
NRO. PASO/FLUJO	**RESULTADO ESPERADO**	**RESULTADO OBTENIDO**
- Ingresar a la aplicación web - Iniciar sesión como administrador - Seleccionar alguna de estas opciones: *Reportes pendientes* o *Reportes revisados*	La aplicación web muestra una lista con los reportes pendientes/revisados, según sea el caso. Para cada reporte, se muestra un botón para descargar el archivo correspondiente. Al presionarlo, se descarga y se guarda en el disco el archivo.	La aplicación web muestra una lista con los reportes pendientes/revisados, según sea el caso. Para cada reporte, se muestra un botón para descargar el archivo correspondiente. Al presionarlo, se descarga y se guarda en el disco el archivo.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Historia de Usuario: Como USUARIO, quiero filtrar las listas de ingresos/gastos por palabras claves.
ID Caso de Prueba: JJ-400

NRO. PASO/FLUJO	RESULTADO ESPERADO	RESULTADO OBTENIDO
- Ingresar a la aplicación móvil - Seleccionar alguno de estos tres elementos de la vista principal: *Gastos, ingresos* o *Balance general* - Seleccionar botón para buscar - Escribir la palabra por la cual se quiere filtrar la lista	La aplicación móvil muestra una lista con ingresos/gastos, o ambos, según sea el caso. Únicamente se muestran aquéllos que contienen en su descripción la palabra ingresada.	La aplicación móvil muestra una lista con ingresos/gastos, o ambos, según sea el caso. Únicamente se muestran aquéllos que contienen en su descripción la palabra ingresada.
Decisión de aprobación del Caso de Prueba : Aprobó __X__ Falló _____ (marque con una X el resultado obtenido)		

Cantidad de casos de prueba probados:

Se probaron 36 casos de prueba que representan las Historias de Usuario (*User Stories*).
Casos de Prueba Aprobados: 36
Casos de Prueba Rechazados: 0
Porcentaje de casos de prueba aprobados: 100%