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, SEPTIEMBRE 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, SEPTIEMBRE de 2016

Resumen

Introducción

Capítulo 1

Entorno Empresarial

Capítulo 2

Marco Teórico

En este capítulo se exponen los conceptos teóricos sobre los cuales se basan las tecnologías empleadas para el desarrollo del proyecto.

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 [1]. 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 que la clase sólo tiene una instancia y provee un acceso global a la misma [2]. 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 [2].

2.2. Patrones de arquitectura

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.

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 [**?**]. 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 [**?**].

Capítulo 3

Marco Tecnológico

Capítulo 4

Marco Metodológico

A continuación se describe el procedimiento seguido para el desarrollo del proyecto. Se decidió utilizar la metodología Scrum por ser el marco sobre el cual trabaja la empresa.

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 [4]. Se caracteriza por ser una metodología ligera, fácil de entender y difícil de dominar, que permite entregar incrementos de producto potencialmente productivos [5].

4.1.1. Roles

En Scrum el desarrollo se realiza por uno o más equipos de trabajo dentro de los cuales existen tres roles: *Product owner* (jefe del producto), *Scrum master* (jefe de Scrum) y el equipo de desarrollo [6].

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 [6]. Este rol fue asumido por el Ing. Chi Wang Zhong.

Scrum master

Actúa como facilitador tanto para el *product owner* 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 [6]. Este rol fue asumido por el Lic. Luis Augusto Peña.

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 [6]. El equipo de desarrollo estuvo integrado únicamente por la pasante Susana Charara.

4.1.2. Actividades

En Scrum, el trabajo se desarrolla en interaciones de una duración máxima de un mes, llamadas ***sprints***. Al final de cada *sprint*, 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:s *sprint planning* (planeación del *sprint*), *sprint execution* (ejecución del *sprint*), *daily scrum* (scrum diario), *sprint review* (revisión del *sprint*) y *sprint retrospective* (retrospectiva del *sprint*) [6].

Sprint planning

Para determinar qué funcionalidades del producto final son las más importantes y próximas a desarrollar, el equipo de trabajo (*product owner*, *Scrum master* y el equipo de desarrollo) realizan una reunión llamada *sprint planning* [6].

Durante la reunión, el *product owner* y el equipo de desarrollo establecen una meta que debe ser cumplida para el final del *sprint*. De acuerdo a esta meta, el equipo de desarrollo decide de una manera realista qué incrementos del producto final pueden entregarse al terminar el *sprint* [6].

Ejecución del *sprint*

Luego del *sprint planning*, el equipo de desarrollo desarrolla todas las tareas acordadas en la reunión. Esto es lo que se conoce como la ejecución del *sprint* [6].

Daily scrum

Cada día dentro de la ejecución del *sprint*, 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 su trabajo [6].

Sprint review

Al final de cada *sprint*, ocurre un evento que se conoce como *sprint review* o revisión del *sprint*. El objetivo de esta actividad es revisar el incremento y realizar las adaptaciones

necesarias al prouducto [6].

Sprint retrospective

Esta actividad ocurre generalmente luego del *sprint review* y antes del próximo *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 los proximos *sprints* [7].

4.1.3. Artefactos

Dentro de Scrum existen dos herramientas o artefactos que permiten mantener un seguimiento del proyecto: el *product backlog* y el *sprint backlog* [7].

Product backlog

Es una lista de los requerimientos funcionales del producto ordenados según su importancia. EL *product owner* 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 [7].

Sprint backlog

Es una lista donde se presenta un subconjunto de los elementos del *product backlog* divididos en tareas más pequeñas [7].

Capítulo 5

Desarrollo de la aplicacion

En este capítulo se describe el trabajo realizado para el desarrollo de la aplicación. Este proceso se dividió en tres fases: investigación, dise no y desarrollo del sistema que incluye tanto la aplicación móvil como el componente servidor.

El proyecto se desarrollo bajo el marco de trabajo de *Scrum*, descrito en el capitulo 1.

5.1. Investigacion

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

En primer lugar se estudiaron diversos patrones de diseño. Para esto, se investigo acerca de patrones ampliamente utilizados en el desarrollo de aplicaciones para Android y en la programacion orientada a objetos en general. Esto permitio estructurar la aplicacion de una manera mas ordenada y entendible, de manera que se pueda extender facilmente.

Luego de esto, se investigó acerca del desarrollo de aplicaciones para Android. Se aprendió a trabajar con el IDE Android Studio, así como los componentes y herramientas para el desarrollo bajo el sistema operativo.

Por último, 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, asi como el flujo de ejecución.

5.2. Diseno

5.3. Desarrollo

Durante esta fase se implementó progresivamente la versión alfa del prototipo funcional. A continuación se presentará los *sprints* realizados, sus objetivos y los resultados obtenidos al final de cada uno.

5.3.1. Sprint 1

Objetivos

- Mostrar información del balance de una cuenta
- Permitir la creación de un nuevo gasto

Resultados

- Se creó la interfaz para ver el total de ingresos, gastos y el balance de la cuenta
- Se crearon las consultas necesarias a la base de datos para obtener el total de ingresos, gastos y balance
- Se creó la interfaz para la creación de un nuevo gasto
- Se creó la consulta para insertar un nuevo gasto en la base de datos

Actividades

En la primera parte del *sprint* se implementó el modelo de datos del dispositivo móvil.

Se creó la vista principal de la aplicación móvil, donde se muestra el total de ingresos y el de gastos, así como el balance general de la cuenta desde su creación hasta la fecha. Para esto fue necesario crear las consultas a la base de datos para obtener el total de ingresos/gastos de una cuenta y la consulta para obtener el balance total de una cuenta.

También se implementó la vista para crear un nuevo gasto con su fecha, monto y descripción. Para esto fue necesario crear una consulta en la base de datos para guardar un nuevo gasto.

Como se expuso en el capítulo 2, la comunicación entre el modelo y la vista se hace a través de una capa intermedia: el presentador. Por esta razón, también fue necesaria la creación de un presentador para cada vista que permita hacer consultas a la base de datos.

5.3.2. Sprint 2

Objetivos

- Mostrar la lista de categorías
- Permitir la creación de un nuevo ingreso
- Permitir guardar fotos de un *entry*
- Asociar un *entry* a una categoría

Resultados

- Se creó en la base de datos del dispositivo una consulta para guardar la lista de categorías por defecto

- Se creó la interfaz para listar las categorías guardadas en la aplicación
- Se adaptó y reutilizó la vista existente para crear un gasto para permitir la creación de un ingreso
- Se agregó la funcionalidad de tomar fotos relacionadas a un *entry*
- Se creó la consulta para guardar en la base de datos el nombre de las fotos tomadas
- se agregó la funcionalidad para poder asociar un *entry* a una categoría existente

Actividades

En este *sprint* se creó la vista para mostrar las categorías presentes en el dispositivo.

Se creó una consulta para guardar en la base de datos una nueva categoría con su nombre y un ícono que la represente. Además, se creó una consulta para guardar en la base de datos una lista de categorías por defecto.

También se adaptó el *activity* asociado a la creación de un gasto, de manera que se pueda reutilizar esta vista para la creación de un ingreso.

Por otra parte, se implementó la funcionalidad para tomar y guardar fotos asociadas a un *entry*. Para esto se tuvo que crear un método que permita abrir la aplicación de la cámara del dispositivo desde la aplicación móvil. Luego, se tuvo que adaptar el *activity* para la creación de un *entry* para mostrar las miniaturas de las fotos tomadas. Se puso un límite por parte de la empresa para que el máximo de fotos que se puedan guardar por *entry* sean 3.

Por último, se modificó la vista para la creación de un *entry* para que se permitiera asociar una categoría. 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 categorias mas 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. En este caso, se guardó una lista con las categorías más recientes.

5.3.3. Sprint 3

Objetivos

- Mostrar la lista de *entries* del mes actual
- Mostrar los detalles de los *entries* existentes
- Permitir editar y borrar un *entry* existente
- Implementar una calculadora para guardar el monto de un *entry*

Resultados

- Se creó la interfaz para mostrar una lista con los gastos y otra con los ingresos del mes actual
- Se creó la consulta en la base de datos para obtener los *entries* del mes actual
- Se agregó la funcionalidad para ver los detalles de un *entry* ya existente, y poder editarlo
- Se agregó la funcionalidad para eliminar *entries*
- Se creó la interfaz para usar la calculadora que permita ingresar el monto de un *entry*

Actividades

En primer lugar, se adaptó la vista principal de la aplicación para mostrar el total de ingresos y gastos únicamente durante el mes actual. Para esto, se tuvo que modificar la consulta a la base de datos para obtener el total de ingresos/gastos dado un rango de fecha.

Se implementó la funcionalidad para navegar a la lista de ingresos o de gastos del mes desde la vista principal de la aplicación. Para esto, se creó una vista general donde se pueda mostrar una lista de *entries*, con su descripción (o categoría, en caso de que no tenga descripción), su fecha y su monto. Esta vista se reutilizó, y dependiendo del tipo de *entry* que se quiere mostrar (ingreso o gasto), se personaliza.

Por otra parte, se creó la funcionalidad para eliminar *entries* existentes desde la vista mencionada en el párrafo anterior.

También se creó la funcionalidad para editar un *entry* existente. Para esto no fue necesario crear una vista nueva, sino que se adaptó la que se tenía para la creación de un nuevo *entry*, de manera que se muestre la información relacionada adicho *entry* (fecha, monto, categoría, descripción y fotos).

Por último, se tenía como requerimiento mostrar una calculadora para ingresar el monto de un *entry*, en lugar de un teclado numérico común. Por esta razón, se tuvo que modificar la vista de creación de un *entry* 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. Se creó el formato del teclado, el cual incluye las teclas presentes en una calculadora básica. También se tuvo que manejar el uso de las teclas: detectar qué tecla fue presionada y realizar una acción en base a esta. Se utilizó una librería para evaluar fórmulas dada una cadena de caracteres.

Para la calculadora, se tenía 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 tuvo que obtener el país de configuración del teléfono para decidir qué símbolo decimal mostrar.

5.3.4. Sprint 4

Objetivos

- Permitir el manejo de nuevas cuentas

Resultados

- Se creó la interfaz para crear una nueva cuenta
- Se creó la consulta en la base de datos para persistir cuentas
- Se implementó la funcionalidad para listar las cuentas del usuario
- Se implementó la funcionalidad para cambiar de cuenta
- Se implementó la funcionalidad para editar el nombre de una cuenta existente
- Se implementó la funcionalidad para eliminar cuentas existentes
- Se implementó la funcionalidad para cambiar el estado de una cuenta: activa o archivada
- Se implementó la funcionalidad para ver la lista de *entries* de una cuenta desde su creacion hasta la fecha actual

Actividades

En la ejecución de los *sprints* anteriores se estuvo trabajando con una sola cuenta creada por defecto en la aplicación. Para este *sprint*, se decidió agregar la funcionalidad para poder manejar nuevas cuentas.

Para esto, se creó la vista que permite agregar una nueva cuenta, con su nombre y la moneda en la que van a estar sus *entries*.

Se creó una vista para mostrar la lista de cuentas guardadas. Se implementó la funcionalidad para cambiar el estado de una cuenta: activa o archivada. La lista de cuentas se muestra segun su estado, es decir, se muestra una lista para la lista de cuentas activas y una para las cuentas archivadas.

Además, se implementaron las funcionalidades para editar el nombre de una cuenta y eliminar cuentas ya existentes.

También se creó la funcionalidad para cambiar la cuenta que se está mostrando actualmente en el dispositivo.

Por último, se adapto la vista existente para mostrar la lista de *entries*, de manera que se pueda mostrar todos los *entries* (ingreos y gastos en una misma lista) asociados a la cuenta que se muestra actualmente.

5.3.5. Sprint 5

Objetivos

- Permitir el manejo de las fotos de un *entry*
- Permitir el manejo de las categorias

Resultados

- Se agregó la funcionalidad para ver las fotos de un *entry*
- Se agregó la funcionalidad para borrar las fotos de un *entry*
- Se agregó la funcionalidad para cambiar la ubicacion en el dispositivo de las fotos tomadas de un *entry*
- Se agregó la funcionalidad para crear una categoría

- Se agregó la funcionalidad para eliminar una categoría existente
- Se agregó la funcionalidad para editar una categoría existentes

Actividades

La primera parte del *sprint* se dedicó al manejo de las fotos. En primer lugar se creó la vista para poder ver en pantalla completa las fotos de un *entry* (esto se hace desde la vista de creación de un *entry*). También se implementó la funcionalidad para eliminar una foto.

Por otra parte, se creo la vista de configuraciones de la aplicación. En esta vista se agregó una opción para cambiar la ubicación en el dispositivo en la que se guardan los archivos de las fotos: memoria interna o memoria extraíble.

La egunda parte del *sprint* se dedicó al manejo de las categorías. Se creó la interfaz para agregar una nueva categoría. Por último, se agregaron las funcionalidades para editar y eliminar categorias existentes.

5.3.6. Sprint 6

Objetivos

- Mostrar un reporte con la lista de *entries* 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

Resultados

- Se implementó la funcionalidad para listar los *entries* de una cuenta en el rango de fecha ingresado por el usuario
- Se implementó la funcionalidad para listar el balance total de las categorías de una cuenta en el rango de fecha ingresado por el usuario
- Se creó la funcionalidad para compartir el reporte con la lista de *entries*,incluyendo las fotos, de la cuenta en el rango de fecha ingresado.
- Se creó la funcionalidad para compartir el reporte con la lista de *entries*, sin incluir las fotos, de la cuenta en el rango de fecha ingresado.
- Se creó la funcionalidad para compartir el reporte con el balance total de las categorías de la cuenta en el rango de fecha ingresado.

Actividades

Este *sprint* se dedicó exclusivamente al manejo de los reportes. Se tenía como requerimiento el manejo de dos tipos de reportes: reporte de balance general y reporte por categorías. Para la generación de ambos reportes, se puede especificar una cuenta y un rango de fechas. El reporte de balance general incluye la lista de todos los *entries* de la cuenta escogida, dentro del rango de fechas establecido. El reporte por categorías incluye únicamente el monto total de todos los *entries* (es decir, la suma de ingresos y gastos) divididos por categorías, de la cuenta y rango de fechas escogidos.

Se creó la vista para mostrar el reporte de balance general, que muestra una lista con los *entries* asociados, sus montos, fechas y descripción (o categoría, en caso de que la descripción sea nula). También se creó la vista para el reporte por categoría, donde se muestra una lista con con el nombre de cada categoría y la suma total de los ingresos y gastos correspondientes.

En este último caso, se omitieron las categorías que no tenían *entries* asociados (es decir, categorías cuyo monto total es cero).

Luego, se creó la funcionalidad para generar un archivo PDF con la información de los reportes descritos anteriormente. Para esto se usó una librería que provee la plataforma de Android para la creación de archivos PDF.

Se tenía como requerimiento que el usuario pueda escoger generar un archivo con el reporte de balance general incluyendo las fotos de los *entries* asociados, o sin incluirlas.

Luego 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. Para la escritura del PDF, se tuvo que lidiar manualmente con la paginación.

Por último, se creó la funcionalidad para compartir este archivo PDF con otras personas a través de otras aplicaciones instaladas en el dispositivo móvil. Dentro de estas aplicaciones se incluyen las de correo electrónico y mensajería móvil que soporten el envío de archivos.

Durante la creacion del archivo PDF se presentaron diferentes dificultades. En primer lugar, las librerias existentes en Java para generar archivos con dicha extension hacen uso de otras librerias, las cuales no son soportadas directamente en la plataforma de Android. Por esta razon, en un principio se decidio utilizar una adaptación de una librería de Java para poder ser utilizada en Android. Sin embargo, no se encontró documentación suficiente que permitiera entender su uso. Por esta razón, se decidió finalmente utilizar una librería nativa de Android que facilita la creación de archivos PDF, pero que sólo está disponible para versiones de Android a partir de la 4.4

5.3.7. Sprint 7

Objetivos

Resultados

Resultados

Capítulo 6

Conclusiones y Recomendaciones

Bibliografía

[1] "Patrón de diseño." https://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o, consultado el 9 de julio de 2016.

[2] E. Freeman, E. Freeman, K. Sierra, and B. Bates, *Head First: Design Patterns*. O'Reilly, 1 ed., 2004.

[3] "Singleton pattern." https://en.wikipedia.org/wiki/Singleton_pattern, consultado el 10 de julio de 2016.

[4] "Scrum (software development)." https://en.wikipedia.org/wiki/Scrum_%28software_development%29, consultado el 7 de julio de 2016.

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

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

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