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

Capítulo 1

Marco Teórico

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

1.1. 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.

1.1.1. 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.

1.1.2. Modelo Vista Presentador (MVP)

Es una derivación del patrón Modelo Vista Controlador (MVC) que se utiliza generalmente para construir interfaces de usuario [2]. 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 [3].

1.1.3. *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 [4]. Esto es útil cuando se necesita únicamente un objeto para coordinar las acciones en el sistema [5].

1.1.4. *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 [4].

Capítulo 2

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.

2.1. 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 [6]. Se caracteriza por ser una metodología ligera, fácil de entender y difícil de dominar, que permite entregar incrementos de producto potencialmente productivos [7].

2.1.1. 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 [8].

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 [8]. 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 [8]. 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 [8]. El equipo de desarrollo estuvo integrado únicamente por la pasante Susana Charara.

2.1.2. 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*) [8].

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* [8].

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* [8].

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* [8].

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 [8].

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 [8].

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* [9].

2.1.3. Artefactos

Dentro de Scrum existen dos herramientas o artefactos que permiten mantener un seguimiento del proyecto: el *product backlog* y el *sprint backlog* [9].

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 [9].

Sprint backlog

Es una lista donde se presenta un subconjunto de los elementos del *product backlog* divididos en tareas más pequeñas [9].

Capítulo 3

Desarrollo de la aplicacion

En este capítulo se describe el trabajo realizado para el desarrollo de la aplicación. Este proceso se dividió en dos fases: investigación y concepción,y el desarrollo del sistema que incluye tanto la aplicación móvil como el componente servidor.

3.1. Investigacion

El objetivo de esta primera fase era familiarizarse con el entorno de trabajo. En primer lugar se investigó lo necesario para conocer la metodología ágil *Scrum* y poder implementarla en el desarrollo del proyecto. También se aprendió a trabajar con el manejador de versiones Git y se presentaron los lineamientos para el uso del mismo dentro de la empresa. Por otra parte, se estudiaron diversos patrones de diseño y cómo podrían ser empleados para la estucturación de la aplicación.

Por último, se realizó una aplicación de prueba para

3.2. 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.

3.2.1. Sprint 1

Objetivos

- Mostrar información del balance general
- 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

3.2.2. Sprint 2

Objetivos

- Mostrar la lista de categorías
- Permitir la creación de un nuevo ingreso
- Tomar fotos del *entry*

- Asociar un gasto o ingreso 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

3.2.3. Sprint 3

Objetivos

- Mostrar la lista de gastos e ingresos del mes actual
- Mostrar los detalles de los gastos e ingresos existentes
- Permitir editar y borrar un gasto o ingreso 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*
- 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*

3.2.4. 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

3.2.5. Sprint 5

Objetivos

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

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

3.2.6. 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.

3.2.7. Sprint 7

Objetivos

Resultados

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] "Model-view-presenter." https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93presenter, consultado el 10 de julio de 2016.

[3] "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.

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

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

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

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

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

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