INSTITUTO TECNOLÓGICO SUPERIOR DE MISANTLA

DISEÑO DE UNA ARQUITECTURA PARA LA ADAPTACIÓN DE NEGOCIOS AL COMERCIO MÓVIL

TESIS

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN SISTEMAS COMPUTACIONALES

P R E S E N T A

NOEL RUDECINO CAMPILLO

DIRECTOR:
MSC. JOSÉ ANTONIO HIRAM VÁZQUEZ LÓPEZ

CODIRECTORES:
MIA. ROBERTO ÁNGEL MELÉNDEZ ARMENTA
DR. SIMÓN PEDRO ARGUIJO HERNÁNDEZ

MISANTLA, VERACRUZ JUNIO, 2019

INSTITUTO TECNOLÓGICO SUPERIOR DE MISANTLA
DIVISIÓN DE ESTUDIOS PROFESIONALES
AUTORIZACIÓN DE IMPRESIÓN DE TRABAJO DE TITULACIÓN

FECHA: 14 de Junio de 2019.

ASUNTO: **AUTORIZACIÓN DE IMPRESIÓN DE TESIS PROFESIONAL.**

A QUIEN CORRESPONDA:

Por medio de la presente hago constar que el (la) C:

NOEL RUDECINO CAMPILLO

pasante de la carrera de INGENIERÍA EN SISTEMAS COMPUTACIONALES con No. de Control 142T0201 ha cumplido satisfactoriamente con lo estipulado por el **Manual de Procedimientos para la Obtención del Título Profesional de Licenciatura** bajo la **opción Titulación Integral (Tesis Profesional)**

Por tal motivo se **Autoriza** la impresión del **Tema** titulado:

"DISEÑO DE UNA ARQUITECTURA PARA LA ADAPTACIÓN DE NEGOCIOS AL COMERCIO MÓVIL"

Dándose un plazo no mayor de un mes de la expedición de la presente a la solicitud del Acto de Recepción para la obtención del Titulo Profesional.

ATENTAMENTE

ING. GERBACIO TLAXALO ESPINOZA
DIVISIÓN DE ESTUDIOS PROFESIONALES

DIVISION DE ESTUDIOS PROFESIONALES

Archivo.

Resumen

Los negocios de la región Misantla en su mayoría no están automatizados, por lo que en un mundo globalizado tecnológicamente estos negocios ofrecen poca competencia ante el comercio electrónico, la principal problemática es que los negocios de esta región no hacen uso de las tecnologías que tienen a su alcance para ofrecer mejores servicios y un mayor rendimiento en sus procesos de trabajo, para solventar esto se planteó el objetivo de desarrollar una aplicación móvil que ayude a los negocios a introducirlos al comercio móvil y así se puedan iniciar en las ventas online motivándolos a invertir más en estas tecnologías, el proyecto desarrollado cubre el manejo de inventarios, la entrada y salida de productos, reporte de ventas, devoluciones y la visualización de productos para realizar compras.

La importancia de desarrollar este proyecto para los negocios de la región de Misantla es la de otorgar una herramienta que permita competir tecnológicamente en el mercado actual, debido a que está incrementando el comercio electrónico debido a las facilidades que ofrece.

Para el desarrollo de este proyecto se escogió la plataforma móvil Android con la implementación de la metodología ágil RAD, aplicando el patrón de arquitectura de software MVVM y utilizando la plataforma Firebase para el almacenamiento de datos. Para una correcta implementación de la aplicación se diseñaron los modelos necesarios como el modelo de casos de uso, modelo de negocios y diagramas UML, la programación se realizó en el IDE Android Studio en el lenguaje de programación Kotlin.

Se obtuvo una aplicación que solventa las necesidades básicas de un negocio que apenas comienza en el e-commerce, la aplicación le permite al comerciante automatizar su negocio en algunos procesos como el manejo de inventario de sus productos o los reportes de ventas.

Al finalizar el trabajo se concluyó que es difícil desarrollar una aplicación que se adapte a todos los diferentes tipos de negocios que existen, debido a que las necesidades de cada uno son diferentes por lo tanto los requisitos que debe cumplir la aplicación se vuelven mucho más amplios y difícil de lograr.

Índice general.

Índice de figuras.

Índice de tablas.

Índice de gráficas.

Capítulo I. Generalidades del proyecto.

1.1 Introducción

En la actualidad la tecnología ha ayudado a las personas a simplificar sus tareas diarias, con el uso de Internet y los dispositivos móviles es fácil mantenerse comunicado en todo momento, las grandes empresas han sabido aprovechar estas tecnologías para poder ejercer una mayor competencia en el mercado y poder expandir su presencia a más lugares de forma nacional e internacional.

El comercio electrónico (e-commerce) es un avance importante para los empresarios, debido a que este nuevo paradigma de marketing les ofrece la posibilidad de llegar a más personas desde cualquier parte del mundo donde se cuente con un dispositivo con conexión a Internet, así los clientes pueden buscar, comprar, adquirir productos y/o servicios desde cualquier lugar donde se encuentren siempre y cuando tengan acceso a Internet.

Esta forma de vender trajo muchas ventajas al mercado debido a que se pueden encontrar productos y/o servicios a un mejor precio, agilizar varios procesos de ventas, llevar un mejor control del inventario de los negocios, el poder recibir los productos comprados en un domicilio acordado por el cliente.

Con las constantes innovaciones a los smartphones y el uso continuo de estos por parte de las personas dio parte a una nueva plataforma de ventas conocida como m-commerce, que consiste en aplicaciones móviles con todas las características del comercio electrónico pero adaptadas a los teléfonos inteligentes.

Sin embargo, el m-commerce no está siendo aprovechado por todos los empresarios y en especial las Pymes, sector comercial al cual va dirigido este trabajo, las razones por las que no hacen uso de estas plataformas de ventas son porque no saben cómo empezar en el mundo del comercio electrónico, no tienen confianza al utilizar estos métodos o simplemente no saben acerca de ellos.

Para el caso del trabajo realizado, con el objetivo de apoyar a estos pequeños negocios se decidió realizar el desarrollo de una aplicación móvil para el sistema operativo Android que realice las funciones de una plataforma de ventas en línea que sea amigable para las personas que apenas estén aprendiendo a utilizar este tipo de sistemas y así familiarizarse con ellas e ir generando confianza al utilizarlas.

Por lo que la idea principal es una aplicación que permita realizar ventas con total facilidad para los clientes como para los vendedores. Las características para los vendedores es ofrecer una mayor agilización en sus procesos de trabajo, ya que sólo tienen que realizar el registro de sus productos para llevar automáticamente el control de las ventas, esto permitirá realizar el despliegue de sus productos en el sistema, también hacer recomendaciones a los clientes de algunos artículos que les pueda interesar. así como llevar un registro de las ganancias, perdidas, las entradas o salidas de mercancía, la generación de tickets electrónicos mediante códigos QR, también la visualización en gráficas entendibles acerca de los reportes de ventas y porcentajes de ganancias.

Las funciones para los clientes es que les permita visualizar los productos que se encuentran disponibles en su región para no tener que recurrir a buscarlos en otros lugares, que permita al cliente realizar la compra desde cualquier lugar, facilitar la comparación de productos u ofertas para los clientes para facilitarles la decisión de que productos comprar y al realizar una compra poder pasar al establecimiento a recoger sus productos o que lo envíen a la dirección acordada por el cliente.

Para lograr esto se decidió utilizar todos los beneficios que otorgan las tecnologías móviles, así como la implementación de técnicas de la ingeniería de software para obtener un producto final estable, sostenible, escalable y de calidad como, el uso de arquitecturas de software, patrones de diseño, la creación de distintos modelos que permitiesen representar el funcionamiento y comportamiento de esta aplicación.

Este sistema está enfocado a la plataforma Android por ser el sistema operativo más usado para móviles, haciendo uso de Firebase que es una plataforma para el desarrollo de aplicaciones web y móviles, por lo que se utilizará para el almacenamiento de datos, se optó por elegir esta plataforma debido a que es gratuito y cubre las necesidades del proyecto.

El proyecto se planteó con el fin de desarrollar herramienta tecnológica que ayude a los comerciantes de la ciudad de Misantla que no hacen uso del comercio electrónico, que tienen la necesidad de automatizar sus procesos de trabajo o de dar un impulso tecnológico a sus negocios para poder ofrecer una mayor competencia en el mercado, la aplicación cuenta con las funciones de registrar usuarios, agregar productos, llevar el inventario del negocio, mostrar los productos a las personas que usen la aplicación, visualizar los productos en un mapa dependiendo de la ubicación del usuario y simular la venta de los productos. La aplicación está diseñada de forma que se pueda ir actualizando para agregar más funciones conforme las necesidades del usuario y demanda del mercado.

Para el desarrollo de este proyecto en elcapítulo1 se describe la necesidad de realizar este trabajo, así como las bases y la justificación siguiendo una lista de objetivos que cumplan con los requisitos necesarios para que la aplicación cumpla sus funciones, en el capítulo 2 se mencionan conceptos claves que se hacen uso en este trabajo, así como las herramientas y tecnologías que se aplicaron, siendo en el capítulo 3 donde se habla acerca de todo el desarrollo realizado para llegar al producto final obtenido, especificando cada parte realizada y las herramientas utilizadas para lograrlo, en el capítulo 4 se muestran todos los resultados obtenidos del desarrollo.

1.2 Descripción de la problemática.

La principal problemática que se trata en este trabajo aborda el déficit de ventas en la región de Misantla, lo que ocasiona la reducción, pérdida de clientes, estancamiento comercial y económico de los negocios de la zona, por lo que algunas personas que dependen de ello se ven forzadas a tomar medidas drásticas como reducir el personal que labora o hasta cerrar sus negocios al no ser redituables para buscar otras oportunidades.

Una de las causas de este problema es que muchos negocios no cuentan con una optimización eficiente para la realización de sus procesos de trabajo, limitándose a hacerlos de forma manual, por lo que la información obtenida de forma tradicional que es necesaria para los dueños como la de los inventarios, las ganancias y/o perdidas no es precisa por lo que no se tiene un control exacto de la oferta y la demanda, esto conlleva a que no sean lo suficientemente competitivos en el mercado actual.

La falta de información precisa es un factor importante por el cual muchos comerciantes no logran obtener los ingresos esperados, lo que provoca que muchas veces se cancelen sus planes de negocios futuros por no tener una vista clara de los beneficios que obtendrán y mucho menos si serán capaces de sostener dichos planes ya que se requieren de grandes inversiones que muchos no se pueden dar el lujo de perder.

Por otra parte, la limitación geográfica es otro de los problemas a los que se enfrentan estos comercios, ya que sólo pueden dar a conocer sus productos a la población de la región donde se encuentran ubicados, con tanta competencia actualmente es cada vez más difícil atraer clientes a estos establecimientos, y para poder expandirse es necesario abrir más sucursales, lo que requiere de una gran inversión económica.

Otro de los problemas que afrontan los comerciantes de esta región es el desconocimiento o desconfianza hacia las ventas por Internet, el no hacer uso de estas plataformas implica perder oportunidades de crecimiento, debido a que no dan a conocer sus productos más allá de la región en la que se encuentran ubicados, y también a perder cuotas de mercado por los clientes que prefieren realizar compras en línea.

Por lo que, los negocios al no contar con herramientas que los ayuden a impulsarse están perdiendo bastante potencial de ventas y mucha cuota de mercado, todo esto por no utilizar el comercio electrónico que es una plataforma móvil de ventas por la que cada vez más usuarios prefieren realizar sus comprar, por consecuencia estos usuarios terminan comprando productos de fuera de la región, esto hace que la cantidad de ingresos en la región sea menor, generando más desabastos y desempleo.

1.3 Objetivos.

1.3.1 Objetivo general.

Crear una aplicación móvil para impulsar el crecimiento de los negocios de la región de Misantla adaptándolos al m-commerce.

1.3.2 Objetivos específicos.

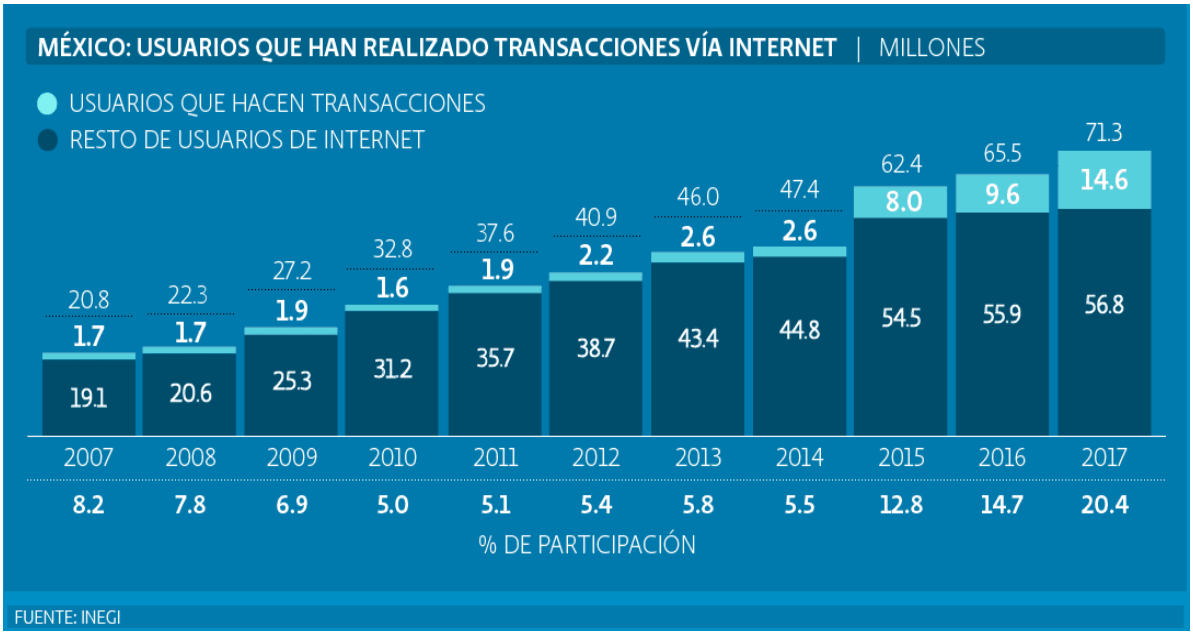
En la búsqueda de llegar a la meta planteada es necesario establecer una lista de objetivos específicos, en los que comprenden:

- Promover en los clientes el uso de aplicaciones de comercio móvil.
- Fomentar el uso del comercio móvil en los comerciantes.
- Fomentar que el negocio tenga un mayor alcance con sus clientes.
- Administrar el stock de productos automáticamente para los negocios.
- Facilitar la búsqueda de productos para los consumidores.
- Promover la inversión e implementación de nuevas tecnologías por parte de los comerciantes.

1.4 Justificación.

De acuerdo con un estudio realizado por el INEGI, el comercio electrónico representó el 4% del PIB de México en el 2016, esto significa que el e-commerce generó 803,103.00 millones de pesos en ese mismo año (Valor agregado bruto del comercio electrónico en México, a precios corrientes, 2018).

En el año2017 de acuerdo a la encuesta realizada por el INEGI (Encuesta sobre Disponibilidad y Uso de las Tecnologías de la Información en los Hogares 2017, 2018), uno de cada cinco usuarios de Internet realizó alguna transacción electrónica, lo que representó un avance respecto al 2016 (ver Gráfica 1), indicando una mayor aceptación en el uso de medios electrónicos con acceso a Internet para realizar compras online.



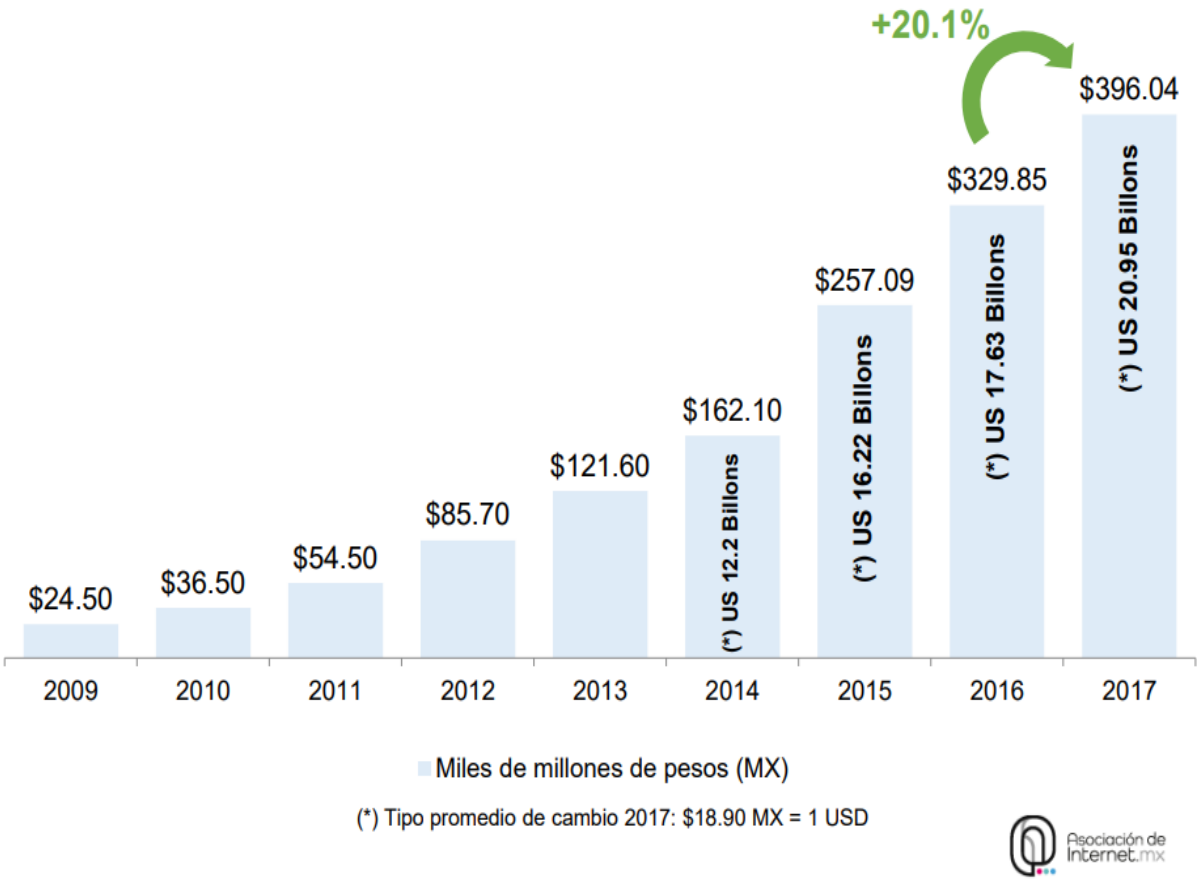
Gráfica 1. Usuarios que han realizado transacciones vía Internet.

Fuente: INEGI, 2017.

Esto conllevó a que los negocios que hicieron uso del e-commerce encontrar una nueva posibilidad para generar ganancias, debido a la comodidad que les otorga a los clientes las compras online y la facilidad que ofrecían los medios de pagos electrónicos, esto aunado al

crecimiento de los smartphones y las redes telefónicas, les permite estar conectados la mayor parte del tiempo lo que benefició aún más al comercio electrónico.

De acuerdo a los datos obtenidos de la Asociación de Internet, el comercio electrónico en México alcanzó un valor de 396 mil millones de pesos en el 2017, esto representa un crecimiento del 20% respeto al año anterior como se puede observar en la Gráfica 2 realizada por la Asociación de Internet, en promedio los usuarios en México compran cada mes algún producto en línea, esto debido a la comodidad y facilidad de compra.

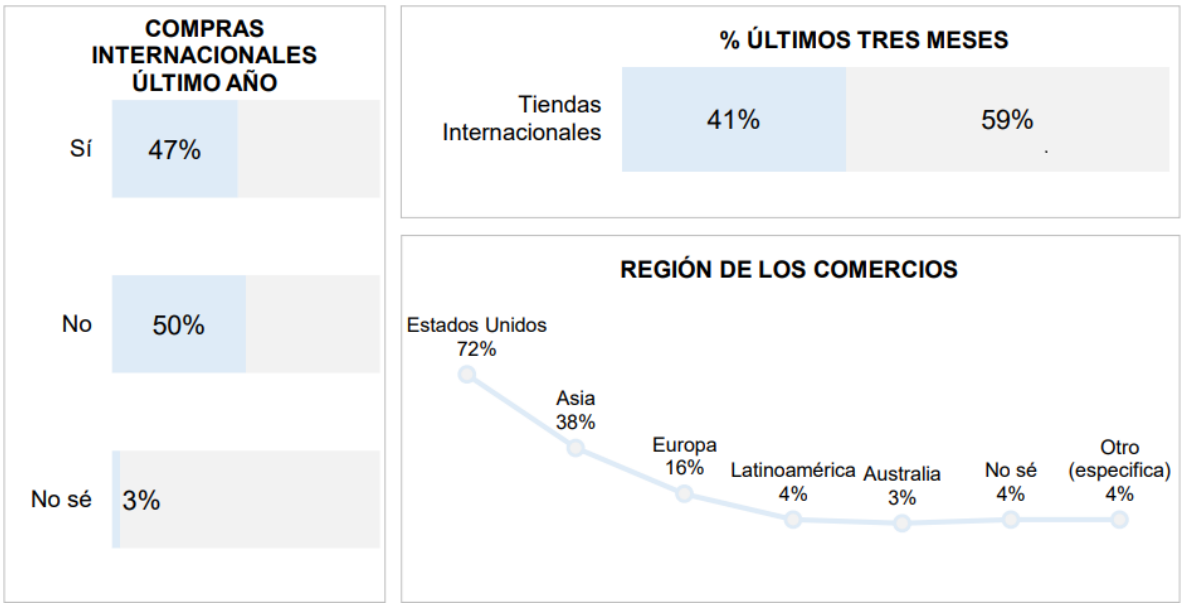


Gráfica 2. Crecimiento del comercio electrónico en México.

Fuente: Asociación de Internet, 2017.

Sin embargo, en el 2018 el 41% de los compradores realizaron una compra en algún comercio electrónico fuera de México lo que es una gran cuota de mercado que no está llegando a los negocios mexicanos (ver Gráfica 3). De todas las compras internacionales se destaca el

mercado de Estados Unidos con un 72%, estos datos obtenidos de la Asociación de Internet indican que 5 de cada 10 compradores realizaron una compra a nivel internacional (Estudio de Comercio Electrónico en México 2018, 2019).



Gráfica 3. Compras internacionales en el 2018.

Fuente: Asociación de Internet, 2018.

Teniendo en cuenta los datos anteriores, cada vez es más necesaria la automatización efectiva y la adaptación al e-commerce de cualquier negocio que quiera destacar y ser competitivo en el mercado, esto debido a que las compras en Internet van en aumento año tras año generando grandes ganancias para los que ya hacen uso de esta plataforma de ventas.

Es por esto que, es tiempo de implementar nuevas formas de vender y generar más presencia en la región de Misantla contra los competidores, ya que actualmente los comerciantes no hacen uso del comercio electrónico o sólo lo utilizan muy pocos, por lo tanto es importante darles a conocer una herramienta que les permita realizar las ventas de sus productos a través de medios electrónicos, para darles la oportunidad de adaptarse a las exigencias de los consumidores actuales, esto con el fin de atraer más clientes hacia los negocios ya que muchos de ellos prefieren visualizar un producto y sus características por medio de alguna aplicación o página web antes de hacer una compra.

Las facilidades que ofrecen las compras online para los usuarios son, la comparación de productos y precios, visualizar detalles en los productos que en un comercio físico no se puede y la seguridad de los métodos de pagos electrónicos con los que se cuenta actualmente entre otras cosas.

La razón de realizar una aplicación dedicada a la región de Misantla es que las e-commerce existentes no se adaptan completamente a las necesidades del negocio, hay muchos comercios que no cuentan con el capital suficiente para invertir en sistemas de ventas y en hardware específico, como sería el caso de máquinas registradoras o lectores de barras, por lo cual se planea hacer uso de una herramienta con la que la mayoría cuenta que es el smartphone, siendo posible con esta lograr cubrir varias de las necesidades que se plantean.

Esto es la razón por la cual se propone desarrollar un sistema informático móvil que se adapte a las necesidades y demandas de los diferentes establecimientos, la aplicación permitirá registrar sus productos existentes, mostrar sus productos a través de Internet, generar tickets de compras mediante códigos QR y generar reportes de ventas, todo esto con el fin de que el negocio sea más competitivo en el mercado.

La implementación de estas herramientas tecnológicas en los negocios de la región es necesaria para ofrecer sus productos de una forma novedosa y mantenerse frente a la competencia, además de mantener mejor control de sus inventarios y finanzas, por lo que se busca inculcarles el uso de estas tecnologías con el fin de que los comerciantes tengan mejores ganancias, negocios rentables y exitosos.

Los beneficios que se quieren otorgar a los negocios son la de facilitar sus métodos de ventas, llevar una información precisa de ganancias y/o perdidas, dar importancia del negocio a nivel regional aumentando su presencia en esta, mostrar a los consumidores que los productos que buscan están en su ciudad y puedan ser recibidos por ellos de la forma más cómoda que elijan, la generación de tickets electrónicos resulta en menos utilización de papel.

Capítulo II. Marco teórico.

2.1 Fundamentos teóricos

2.1.1 ¿Qué es el e-commerce?

Es una forma de vender u ofrecer servicios utilizando contenidos, aplicaciones y servicios en línea para que el cliente encuentre los productos u ofertas que busca de un distribuidor a través de un dispositivo conectado a internet (Fernandez & Medina, 2018).

El comercio electrónico es un nuevo paradigma en la forma de comprar y vender, esto debido a que las nuevas tecnologías de información y comunicación que se están innovando constantemente están generando nuevos canales de ventas para las empresas.

El comercio electrónico es una revolución tanto para empresas como para los consumidores de sus productos, esto se convirtió en una de las principales actividades de la economía mundial, debido a que una empresa puede tener presencia mundial y dar a conocer sus productos en cualquier parte del mundo donde exista un dispositivo con conexión a internet.

Los procesos del comercio electrónico en general (ver Figura 1) empiezan en el cliente accediendo a internet desde algún dispositivo, accede a una página web de ventas online y realiza la compra de lo que busca, sus datos son enviados a un servidor para realizar las transacciones necesarias con el banco, a la tienda le llega la notificación de la compra de alguno de sus productos para que luego se procese el producto y sea enviado al cliente.

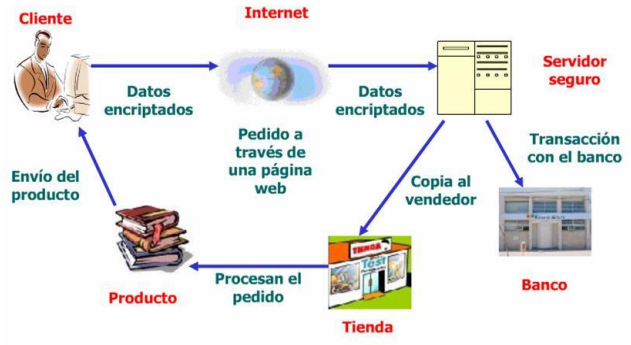


Figura 1: Proceso del comercio electrónico.

2.1.2 ¿Qué es el m-commerce?

De acuerdo los trabajos de (Clarke, 2001) y (Lee, 2007) se describe al comercio móvil como el proceso de realizar transacciones comerciales de productos y/o servicios a través de tecnologías móviles para facilitar a los clientes las compras en cualquier momento y lugar.

Por esto se llega a deducir que el comercio móvil es un derivado del comercio electrónico, debido a la escasez de trabajos acerca del comercio móvil que adopten los puntos de vista de los diferentes negocios resulta complicado comprender el proceso de aceptación de las empresas hacia el m-commerce.

En el caso de las empresas es difícil hacerse una idea de ellas al momento de plantearse la idea de adaptar el comercio móvil a sus planes de negocios, así como identificar en la literatura una serie de motivos y obstáculos que las empresas reconocen al momento de adentrarse al comercio móvil (ver Figura 2).

Figura 2: Comercio móvil.

El diseño y desarrollo de una aplicación de comercio móvil requiere de varias metodologías y técnicas de desarrollo para lograr un producto final estable, escalable y robusto, todo esto parte de diferentes conceptos de la ingeniería de software que son necesarios saber para comprender todo lo que se aplicó para llegar al producto final de este trabajo, los cuales se describen a continuación.

2.1.3 Modelo entidad-relación

El modelo de datos entidad-relación (E-R) está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre estos objetos.

Las entidades se describen en una base de datos mediante un conjunto de atributos. Una relación es una asociación entre varias entidades, el conjunto de todas las entidades del mismo tipo, y el conjunto de todas las relaciones del mismo tipo, se denominan respectivamente conjunto de entidades y conjunto de relaciones (Silberschatz, Korth, & Sudarshan, 2002).

La estructura lógica general de una base de datos se puede expresar gráficamente mediante un diagrama E-R.

2.1.4 Paradigmas de programación

Los paradigmas corresponden a los modelos matemáticos que subyacen a una determinada forma de resolver un problema y que, en la actualidad, involucra en gran medida la participación de tecnología informática, computadores y herramientas de desarrollo sin que estos elementos sean absolutamente imprescindibles. Los lenguajes de programación corresponden a conjuntos de instrucciones que permiten construir programas a la luz de determinados paradigmas de programación e, incluso, como combinación de algunos de ellos. Estos conjuntos de instrucciones llamados Lenguajes de Programación son aceptados por la comunidad tecnológica internacional y cuentan con recursos como compiladores, ejecutores, editores y ambientes integrados de desarrollo que, en algunos casos, están

disponibles libremente y se puede acceder a ellos a través de la web y, en otros casos, corresponden a la línea de software privativo (Trejos Buriticá, 2014).

Programación Orientada a Objetos

Una de las ideas fundamentales del paradigma de programación orientada a objetos es el concepto de objeto como una entidad que engloba datos y funciones (Gallardo López & Promares Puig, 2008).

En programación funcional el elemento principal que determina el funcionamiento de un programa son las funciones. Las funciones transforman datos, los cuales sólo tienen posibilidad de existir como valores que se pasan a una función o que son devueltos por ella. Pero ahí termina su existencia. No tienen vida fuera de las funciones (Gallardo López & Promares Puig, 2008).

En programación imperativa, por otra parte, los datos se pueden considerar los elementos fundamentales de un programa. Existen por sí mismos y son usados o modificados por las funciones. La posibilidad de la programación imperativa de mantener y modificar un cierto estado (datos, valores de las variables) la hace muy potente para representar y modelar procesos del mundo real que serían complicados de expresar en forma de programación funcional. Las funciones son utilizadas para modificar el estado del programa (Gallardo López & Promares Puig, 2008).

Sin embargo, es posible un enfoque distinto que agrupe datos y funciones. Se trata del utilizado en el paradigma de Programación Orientada a Objetos (POO). En este enfoque, el concepto fundamental es el de objeto. Un objeto es una entidad con un estado (datos o variables de instancia) y unas funciones (métodos) que pueden acceder y modificar este estado. Para evaluar las funciones hay que enviar un mensaje al objeto solicitando que se ejecute alguno de sus métodos. Sólo es posible consultar el estado de un objeto mediante alguno de sus métodos. De esta forma, en POO se refuerza la filosofía de la barrera de abstracción y de la ocultación de información (Gallardo López & Promares Puig, 2008).

El enfoque de la POO permite modelar un dominio (problema a programar, un simulador de deportes, por ejemplo) de una forma muy cercana a la realidad. Los objetos del programa simulan los objetos (sustantivos) del dominio (por ejemplo, bicicleta, marchas, carretera, etc.). Y los métodos de los objetos permiten modelar perfectamente las acciones (verbos, por ejemplo, cambiar de marcha, pedalear, etc.) que pueden realizar (Gallardo López & Promares Puig, 2008).

2.1.5 Ingeniería de software

Según (Sommerville, 2005), la ingeniería de software es una disciplina de la ingeniería que comprende todos los aspectos de la producción de software desde las etapas iniciales de la especificación del sistema, hasta el mantenimiento de éste después de que se utiliza. En esta definición, existen dos frases clave:

- **Disciplina de la ingeniería:** Los ingenieros hacen que las cosas funcionen. Aplican teorías, métodos y herramientas donde sean convenientes, pero las utilizan de forma selectiva y siempre tratando de descubrir soluciones a los problemas, aun cuando no existan teorías y métodos aplicables para resolverlos. Los ingenieros también saben que deben trabajar con restricciones financieras y organizacionales, por lo que buscan soluciones tomando en cuenta estas restricciones.

- **Todos los aspectos de producción de software:** La ingeniería del software sólo comprende los procesos técnicos del desarrollo de software, sino también con actividades tales como la gestión de proyectos de software y el desarrollo de herramientas, métodos y teorías de apoyo a la producción de software.

2.1.6 Modelo de procesos del software

Un proceso se puede definir como la colección de actividades de trabajo acciones y tareas que se realizan cuando va a crearse algún producto terminado. Cada una de las actividades, acciones y tareas se encuentra dentro de una estructura o modelo que define su relación tanto con el proceso como entre sí (Pressman, 2005).

RAD

El término Rapid ApplicationDevelopment o RAD (Desarrollo Rápido de Aplicaciones o DRA en español) es modelo de procesos de software, desarrollado inicialmente por James Martin. Aumenta la productividad por la extensa participación del usuario en el desarrollo. El desarrollo de prototipos tiene muchos beneficios, estos reducen los malentendidos y aclaran los requerimientos. La creación de prototipos efectivos incrementa la calidad del software, en la Tabla 1 se listan las ventajas y desventajas.

Ventajas	Desventajas
Fácil implementación.	Riesgo de sistemas de mala calidad.
Mejora de la satisfacción del usuario.	Necesita más experiencia de desarrollo que otros modelos.
Reduce el tiempo de desarrollo.	Requiere una fuerte administración y control del proyecto.
Incrementa la calidad del sistema.	Necesidad de normas y procedimientos documentados.

Tabla 1: Ventajas y desventajas del modelo RAD

Para el uso de RAD se consideran al menos los siguientes requerimientos:

- Una metodología sólida.
- Un repositorio central para toda la información reunida del proyecto.
- Reutilización de código.
- Uso de librerías de terceros.

2.1.7 Metodologías ágiles

Las metodologías ágiles son las que permiten la adaptación de la forma de trabajar a las condiciones que el proyecto necesita, al aplicar estas metodologías se logra obtener una flexibilidad e inmediatez en el desarrollo con las circunstancias específicas requeridas.

Estas metodologías mejoran la calidad del producto final debido a la continua interacción entre los desarrolladores y los clientes por lo que es más probable que el resultado sea exactamente lo que el cliente había solicitado.

Programación extrema (Extreme Programming, XP)

XP (Beck, 2004) es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.

XP se basa en la alimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico (ver Figura 3).

Figura 3: Fases de la metodología Extreme Programming

2.1.8 Patrones de arquitectura de software

Un patrón de arquitectura de software permite separar los componentes del programa en capas y describe como interactúa cada una entre ellas. Existen varios, cada uno con sus desventajas y desventajas.

Patrón de diseño MVVM

Es un patrón de arquitectura de software que separa la aplicación en 3 componentes principales Modelo, Vista, Vista-Modelo, cada una con un propósito.

Modelo

Al igual que en modelos MVC o MVP, es la información o la lógica de negocio, también puede incluir modelos de objetos.

Vista

Se encarga de presentar los datos obtenidos a través de la Vista-Modelo, a su vez también se encarga de interactuar con el usuario y de detectar eventos de esto como el clic en un botón, desatando alguna acción como la petición y posterior carga de información, esto a través de mantenerse escuchando al Vista-Modelo asociada a ella.

En Android los elementos que forman parte de la vista son:

- Actividades
- Fragments
- Layouts (XML)

Vista-Modelo

Es el encargado de digerir la información obtenida del Modelo para que la Vista pueda digerirla fácilmente. Se encarga de proveer la funcionalidad a la Vista, en el modelo de

MVVM cuando un usuario genera un evento en la Vista como un clic en un botón la lógica que ocurre después de eso es trabajo de la Vista-Modelo, a su vez también se encarga de preservar la información durante la sesión del usuario.

Una de las ventajas de usar este tipo de arquitecturas es que tanto programadores y diseñadores gráficos pueden trabajar conjuntamente sobre una misma aplicación sin interferir en las actividades del otro ya que al ser la vista un elemento independiente de la cual se encarga el diseñador gráfico de la Vista-Modelo de la cual se encarga el programador que es la que proporciona funcionalidad a la Vista (ver Figura 4).

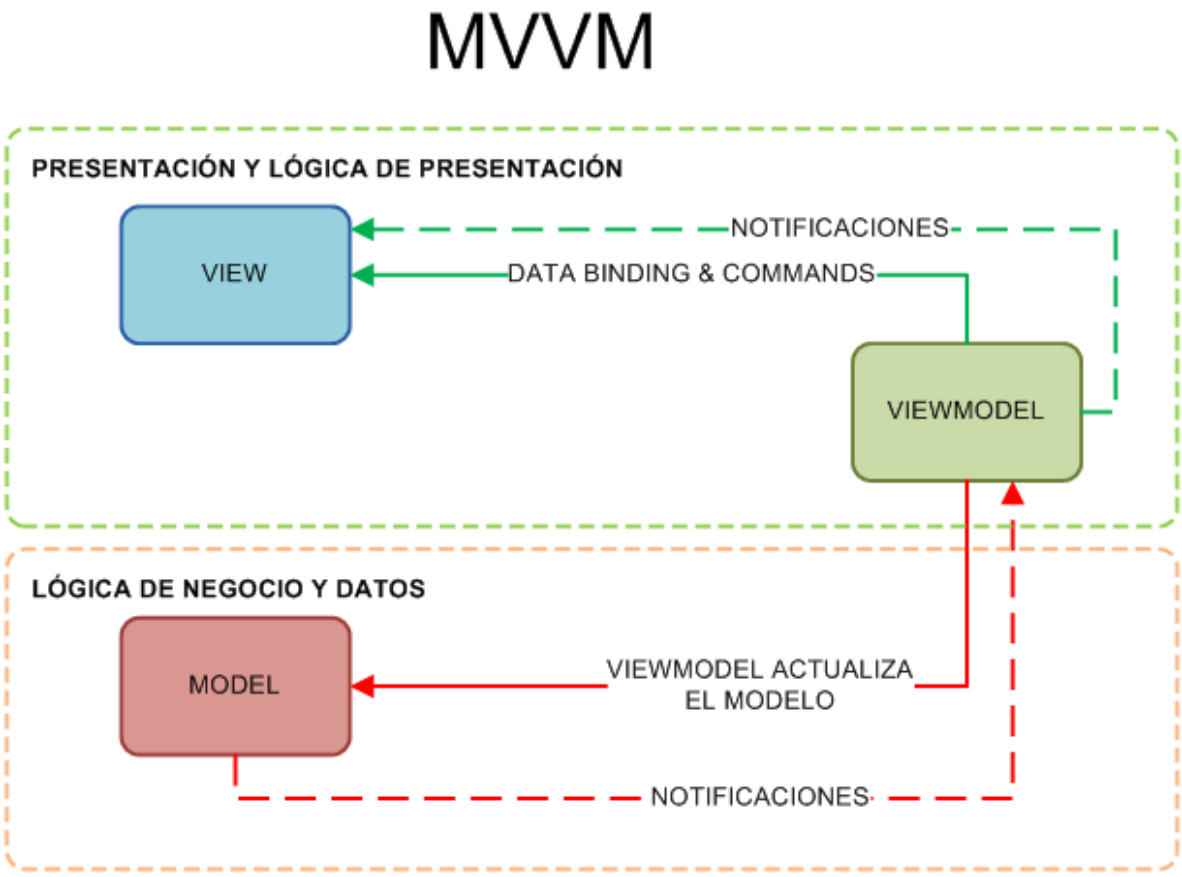


Figura 4. Diagrama patrón MVVM.

La importancia del uso de MVVM en Android es el poner toda la lógica y guardar información temporal dentro de la Vista puede significar en perdida de información, ya que esta suele ser algo “volátil”, por ejemplo una simple rotación de pantalla resulta en una

destrucción y creación de nuevo de una Vista perdiendo toda la información almacenada en ella, tal como se explica en la Figura 5, la Vista-Modelo permanece durante todo el ciclo de vida de la Vista asociada a ella.

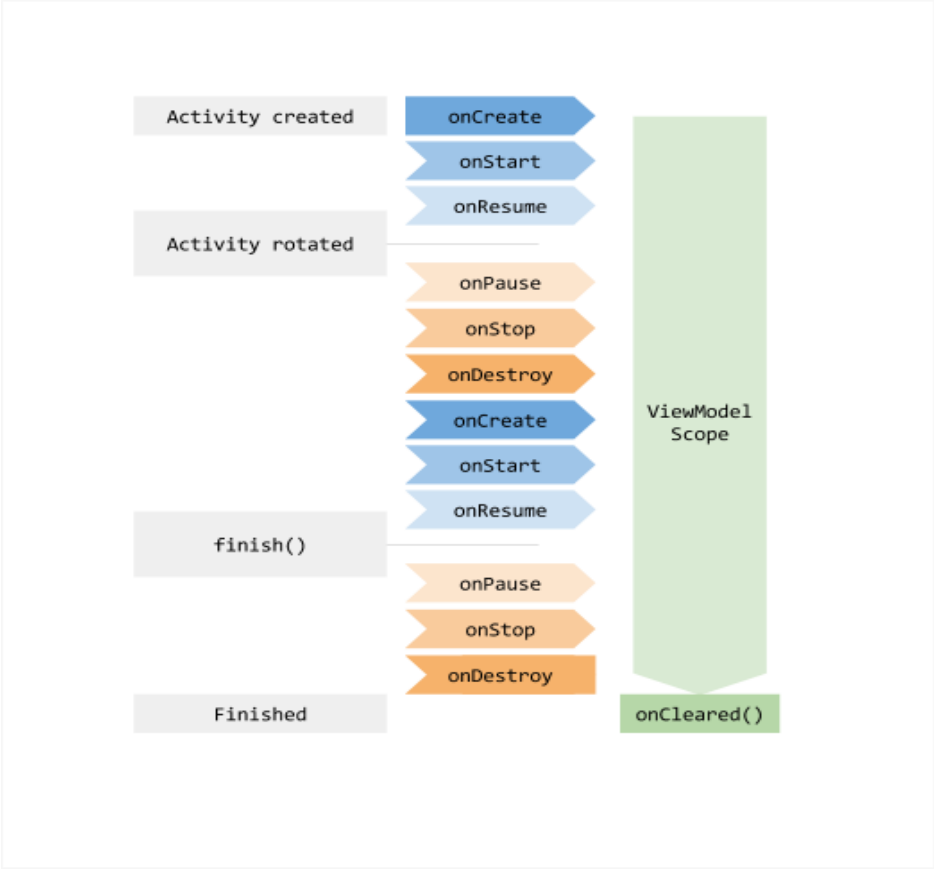


Figura 5: Ciclo de vida de una aplicación Android.

2.1.9 Kotlin

Kotlin es un lenguaje de tipado estático que compila a byteco de de la máquina virtual de Java o incluso a JavaScript

A diferencia de otros lenguajes Kotlin viene con un periodo de diseño, iteración y desarrollo muy largo para convertirlo en un producto muy maduro desde su primera versión estable, con el objetivo de resolver problemas que a día de hoy tienen todos los programadores de Java.

Otro de los objetivos importantes en su diseño es que sea 100% compatible e interoperable con Java, ejecutar código Java desde Kotlin o viceversa sin problemas.

Uno de los objetivos de Kotlin es minimizar la barrera de entrada permitiendo que no se tenga que esperar a desarrollar un nuevo producto para empezar a utilizarlo, sino que, se pueda empezar a utilizar en todo nuevo código que se desarrolle sin tener que preocuparse por la compatibilidad.

Por si esto fuera poco Kotlin no es solo un lenguaje, sino que también es un plugin para diferentes entornos de desarrollo. Entre las características que ofrece está el autocompletado, soporte completo para depuración, refactors e incluso un convertidor de Java a Kotlin.

Capítulo III. Desarrollo.

3.1 Descripción de método.

Para el desarrollo del trabajo se utilizó la metodología ágil de desarrollo de software XP (Extreme Programing) por ser la que mejor se adapta a las necesidades del proyecto junto con el modelo de procesos RAD (Rapid ApplicationDevelopment).

3.1.1 Planeación de requerimientos.

En la etapa de planeación se especificaron las actividades necesarias para la obtención de requerimientos de acuerdo con la ingeniería de software, todo esto con el fin de conocerlas necesidades del sistema, por lo cual se realizó una investigación de algunas e-commerce para identificar sus puntos fuertes y las razones por las que los usuarios prefieren utilizar este tipo de aplicaciones para realizar compras, también se acudió a diversos establecimientos de comercio para observar sus procesos de trabajo para encontrar similitudes y diferencias con el fin de plantear diversas soluciones que cubrieran las diferentes necesidades de los negocios, por lo tanto, a partir de la información recopilada fue posible determinar los requerimientos necesarios para el desarrollo de la aplicación, para posteriormente plantear una alternativa que dé solución a las necesidades generales que tienen todos los negocios que limitan su crecimiento en el mercado, esto con el fin de dar un impulso tecnológico a los diferentes tipos de comercio para que sean más competitivos.

3.1.2 Diseño del sistema.

Con una propuesta planteada teniendo claro el trabajo a desarrollar, se contempló realizar actividades enfocadas al diseño de modelos de casos de uso y modelos de negocios que determinaran la funcionalidad del sistema, así como el diseño general de la arquitectura que contempla todas y cada una de las necesidades comunes de la aplicación para representar todas las funciones del producto final con almacenamiento de datos en la nube a través de Firebase, también el diseño de diagramas UML para tener una ruta clara al momento de desarrollar la aplicación, para el diseño de las interfaces de la aplicación fue necesario realizar los diseños preliminares o bocetos de los componentes con las que contaría la aplicación.

Para el diseño del sistema en general se plantearon 14 módulos que componen el funcionamiento total de la aplicación. Estos módulos comprenden desde el registro de usuarios nuevos, el manejo de inicio de sesión, el registro de mercancía, manejo automático de inventario, la visualización de varios artículos en base a las preferencias del usuario, busque por localización geográfica de productos, la generación de tickets por códigos QR, el proceso de venta mediante pagos electrónicos y la opción de métodos de envíos viables.

Estos módulos que se encuentran especificados en el punto 3.2 se desarrollaron por separado uno independiente de los otros de acuerdo a las necesidades de programación del sistema, para al final al juntar todos generar un sistema robusto, de calidad, cumpliendo con todos los requisitos y necesidades que se plantearon en este trabajo.

3.1.3 Desarrollo.

Para el desarrollo se planearon las actividades a realizar, en la cual, se iniciaría con la creación del proyecto en Firebase que es la plataforma elegida para almacenar los datos y administrar el inicio de sesión o el registro de los usuarios, posteriormente, se desarrolló las vistas de la aplicación de acuerdo con el diseño de los bocetos del punto anterior, para el desarrollo del sistema se implementó el patrón de diseño MVVM con el fin de tener un producto final escalable y fácil de mantener.

Debido a la falta de tiempo, de recursos económicos y acceso a algunas tecnologías solo se desarrollaron 7 de los 14 módulos que se encuentran descritos en el punto 3.2, por lo tanto, los puntos alcanzados fueron los siguientes:

- Módulo 1. Registro de usuarios.
- Módulo 2. Inicio de sesión y validación de usuarios.
- Módulo 3. Despliegue de productos de inicio.
- Módulo 4. Agregar productos al inventario.
- Módulo 5. Modificación del inventario.
- Módulo 6. Carrito de compras.
- Módulo 7. Simulación del proceso de compra.

Estos módulos fueron suficientes para realizar las funciones básicas de una m-commerce como el manejo de usuarios, de productos e inventario y el proceso de compra simulado ya que no se cuenta con integración de pagos electrónicos.

El lenguaje elegido para la programación es Kotlin por las ventajas que ofrece, el proyecto se segmentó en diferentes módulos, de forma que se le estarían agregando funcionalidades a la aplicación dependiendo de las necesidades de programación, por lo tanto, el proyecto iniciaría con el manejo de usuarios, como el registro e inicio de sesión de estos, para posteriormente, continuar con el desarrollo de un módulo que se encargue de la administración del inventario, el cual permite llevar un control de los productos de los vendedores, de igual forma, sería necesario realizar el desarrollo de un módulo de venta, el cual funcionaria a la par con el de inventario.

3.1.4 Implementación.

Etapa en la cual se implementaría el sistema para realizar diversas pruebas con el fin de detectar errores o falta de optimización en algunos procesos, para ello se instalaría la aplicación en diferentes dispositivos y diferentes versiones de Android siendo la versión 4.4 Kitkat la mínima soportada.

Al realizar todas las correcciones pertinentes, se planeó implementar el producto final en unos pocos negocios para verificar su comportamiento en busca de posibles errores que pudieran ocurrir.

3.2 Procedimiento y descripción de las actividades realizadas.

Primero se investigó las necesidades comerciales, el alcance del proyecto, las restricciones y los requisitos del sistema de la aplicación, este punto inicial fue importante para un correcto desarrollo del proyecto, el resolver todas las preguntas que pudieran surgir durante este punto fue clave para tener una idea clara de lo que se quería realizar y así llegar a un resultado final deseado.

Se investigó acerca del comportamiento del e-commerce en México, para esto se revisaron los datos de un estudio que realizó la Asociación de Internet acerca del comercio electrónico en el país durante el 2018 (Estudio de Comercio Electrónico en México 2018, 2019), estos datos ayudaron a entender el panorama del comercio online en México para así tener una idea de cómo abordar la problemática planteada en el punto 1.3.

Se acudió a varios negocios para analizar sus necesidades tecnológicas y evaluar las posibles formas de automatizarlos, al ver la forma en que trabajaba el personal de estos negocios para realizar sus tareas diarias, se llegó a la conclusión de que tienen procesos de trabajo parecidos y otros muy diferentes por lo que se procedió a plantear una solución factible para estos comercios, partiendo de estos como base para crear una aplicación general que se adapte a la mayoría de los negocios de la región.

De acuerdo a los datos analizados del (Estudio de Comercio Electrónico en México 2018, 2019) se tiene que el comercio electrónico aumento un 20% en el 2018 en comparación con el año anterior y que el 65% de los compradores online se encuentran satisfechos con sus compras realizadas y un 17% muy satisfechos, por lo que el porcentaje de recompra en internet de estos usuarios es de un 91%.

Con los datos que se tenían hasta el momento se plantearon posibles soluciones del problema de la automatización y el de iniciar estos negocios en el comercio electrónico, la solución por la que se optó fue por la creación de una aplicación m-commerce optimizada principalmente para los negocios antes mencionados para solventar las necesidades de estos establecimientos.

Con los requisitos obtenidos y una solución planteada se procedió con la elaboración de los modelos de caso de uso, estos modelos son una descripción de una acción o actividad, una actividad que deberá realizar un actor para que se lleve a cabo.

Se crearon un total de 5 modelos de casos de uso para describir las acciones que se llevarán a cabo en la aplicación, un modelo de casos de uso general que describe las actividades del sistema donde interactúan el cliente y el vendedor, un modelo para el cliente donde describe sus acciones permitidas y 3 modelos para la descripción de las actividades del vendedor.

El diseño de estos modelos ayudó a entender las interacciones de los usuarios que harían uso de la aplicación y así se llegó a comprender las funcionalidades que le harían falta al sistema.

Posteriormente se empezaría a diseñar una arquitectura de software general para visualizar como sería los componentes con los que debería contar el sistema que se desarrollaría en este trabajo, la arquitectura se diseñó contemplando el patrón MVVM para obtener una mejor escalabilidad del proyecto.

Teniendo ya diseñada la arquitectura se crearon 8 diagramas UML para representar las clases que se programarían más adelante en la aplicación, estas clases solo están limitadas a cubrir ciertos módulos del alcance total del proyecto debido a la falta de tiempo y el gran esfuerzo de ingeniería que conlleva desarrollar todas las funcionalidades descritas, por lo tanto, solo cubren las funciones básicas de la aplicación que se mencionaran más adelante.

Posteriormente se diseñaron 2 modelos de negocios 1 muestra el alcance realizado del proyecto y otro muestra la idea general que se quiere alcanzar, esta parte sirvió de guía al momento de realizar la programación del sistema, por lo que fue importante tener bien definido este punto.

El modelo de negocios es una herramienta que permite definir con claridad lo que se ofrecerá en este proyecto, es una herramienta de análisis que permite saber el funcionamiento antes del desarrollo.

A continuación, se describen los módulos realizados en este proyecto para entender con mayor claridad los modelos de negocios realizados:

Módulo 1. Registro de usuarios.

Esta parte se desarrolló con el objetivo de registrar nuevos usuarios, validar la información ingresada al sistema como el correo electrónico, que la contraseña sea lo suficientemente segura, el número telefónico entre otros datos, estos datos al ser validados se almacenan en un servidor utilizando el servicio de Firebase Authentication.

Módulo 2. Inicio de sesión y validación de usuarios.

El siguiente módulo se encarga de que el usuario al ingresar sus credenciales estas sean correctas, haciendo uso del mismo servicio de Firebase que se usó en el módulo 1, este se encargará de mantener la sesión iniciada hasta que el usuario la cierre, por lo tanto, el usuario, aunque cierre la aplicación al abrirla de nuevo su cuenta de usuario estará activa.

Módulo 3. Despliegue de productos de inicio.

La función que se desarrolló en esta parte es la de recuperar la información de los productos registrados para mostrarlos en forma de lista, ordenando la información de forma tal que sea entendible para el usuario, esta lista mostrará productos aleatorios con el fin de mostrar la mayor cantidad de productos para los clientes cada que se acceda a la pantalla principal y así buscar productos de interés para ellos.

Módulo 4. Agregar productos al inventario.

Módulo hecho para que los usuarios puedan agregar productos a la aplicación y estos puedan ser vistos o buscados dentro de esta, la información que se ingrese para el registro de nuevos productos será validada y se comprobará si no existen productos iguales agregados por ese

usuario, si es así entonces solo se sumará la cantidad agregada al producto existente, sino entonces se agregará un nuevo producto.

Módulo 5. Modificación del inventario.

Etapa de desarrollada para que el usuario pueda modificar la información de los productos registrados, las características que se pueden modificar son el nombre, precio, cantidad o descripción, también se puede dar de baja algún artículo que ya no sea necesario con el fin de no mostrar más el producto dentro de la aplicación.

Módulo 6. Carrito de compras.

El carrito de compras permite a los usuarios generar una lista de los artículos que quiere comprar, esta lista se guarda mientras la sesión este activa, en caso de que el cliente no quiera algún artículo puede retirarlo de la lista o modificar la cantidad de productos que quiere, la cantidad disponible a agregar se validad con la existencia disponible, por lo tanto, no se pueden agregar o solicitar más de los existentes en el inventario.

Módulo 7. Simulación del proceso de compra.

Este módulo complementa al anterior, una vez el usuario decida realizar la compra de los artículos del carrito de compras, en esta parte se realizará el cálculo del coste total de los productos en la lista, también se modificará la existencia en el inventario de los artículos comprados automáticamente.

Estos módulos fueron los que se lograron desarrollar en este proyecto, esto se representa en el modelo de negocios que se encuentra el capítulo 4 de resultados (ver Figura 13).

A continuación, se mencionan los módulos que quedan por desarrollar:

Módulo 8. Generador de tickets mediante QR.

Con el fin de limitar el uso de papel en los tickets se plantea la opción de generar tickets digitales que se almacenan en el dispositivo del cliente, estos tickets estarán en formato QR y disponibles para el usuario al momento de necesitarlos, los vendedores podrán analizar el

código QR del cliente para ver los detalles de la venta proporcionando la información necesaria en caso de haber algún inconveniente.

Módulo 9. Asignación de roles a los usuarios.

Para una mejor función de la aplicación es necesario que los usuarios se les asignen roles según su función, estos roles son cliente, vendedor y administrador, esto con el fin de ofrecer las funciones necesarias para cada tipo de usuario registrado en el sistema.

Módulo 10. Registro de tiendas dentro de la aplicación.

Opción que es necesaria para los vendedores, al momento de dar de alta su usuario como comerciante será necesario que registre su negocio ingresando la información correspondiente como el nombre, ubicación, descripción de lo que ofrece, etc, cuando se registra un negocio, el que lo dio de alta puede agregar más usuarios como empleados a su negocio con funciones más limitadas.

Módulo 11. Localización geográfica de productos.

Opción para que los clientes puedan localizar los productos que buscan cerca de su ubicación si es que existen, esta función es para los usuarios que buscan algo en particular y lo necesitan a la brevedad, siendo esta la solución posible al indicarle en que establecimiento se encuentra el producto y pasar a comprarlo en ese momento.

Módulo 12. Métodos de envió o entrega de productos.

Este módulo es el que se encargará de mantener informado al cliente acerca de cómo se enviaron sus artículos comprados y el medio que se utilizó dando una fecha aproximada de cuándo será entregado en el domicilio acordado por el cliente, para esto primero se tienen que ver qué posibilidades para realizar envíos existen y cuales serían posibles implementar al sistema.

Módulo 13. Generador de reportes de ventas.

Esta función es necesaria para todos los vendedores, ya que les permitirá saber la entrada y salida de productos, las ganancias obtenidas, la oferta y la demanda entre otras cosas, los

reportes serán mostrados mediante gráficas de forma entendible para el usuario y así pueda usar la información otorgada por la aplicación para tomar mejores decisiones en el negocio.

Módulo 14. Notificaciones de ofertas de productos.

Esto permitirá que los dispositivos móviles permitan recibir notificaciones por parte de la aplicación para tener al cliente informado acerca de productos que le interese, o de productos parecidos a los que tenga en su lista de deseados, esto con el fin de dar a conocer más artículos que puedan ser interesantes o necesarios para los clientes.

Para la programación de los módulos antes mencionados se realizó lo que es la instalación de Android Studio que es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android. Ofrece un potente editor de códigos y las herramientas necesarias para crear aplicaciones Android. Este IDE potente y fácil de usar ofrece a los usuarios un entorno gráfico para la visualización del proyecto y emuladores con diferentes versiones de Android para testear las apps creadas.

Se descargó Android Studio desde su página oficial y una vez finalizó este proceso se instaló, configuró y ejecutó el IDE. Cuando se abrió Android Studio por primera vez pidió instalar los últimos ajustes necesarios para su uso.

Al abrir Android Studio se creó un proyecto nuevo con el nombre de Systock con las configuraciones por defecto, en la última parte de la creación del proyecto aparecen varias plantillas a escoger, para este caso no se escogió ninguna para empezar el proyecto completamente en blanco, al terminar se abre la interfaz de Android Studio donde hay que esperar un momento en lo que termina de copilar el proyecto y de hacer los ajustes necesarios.

Por último, antes de que se empezara a realizar la programación de la aplicación fue necesario agregar un emulador desde el AVD Manager de Android Studio, esta opción permite añadir un emulador de forma sencilla gracias a su asistente gráfico, el emulador recibió el nombre de “Tester” con un tamaño de pantalla de 5”, una resolución de 1080x1920 y se le asigno 4

GB de memoria RAM, la versión de Android instalada fue la versión 8.1 con el nombre de Android Oreo, al finalizar la instalación se ejecutó sin problemas.

Para la conexión de la base de datos con la aplicación Android se realizaron web Services en el lenguaje de PHP y se alojaron en el servidor para que se pudieran acceder a ellos, se crearon un total de 11 Web Services que se encargan de procesar la información de diferentes acciones de la aplicación.

Fue necesario la creación de Web Services para manejar el registro e inicio de sesión de los usuarios, el registro, la modificación y eliminación de los productos, la creación de un carrito de compras para la sesión activa y así poder agregar productos a su lista de compra o eliminarlos.

Al finalizar con los Web Services se comenzó la codificación de la aplicación en Android Studio, iniciando por las vistas en el lenguaje XML, en las clases de cada vista se programó la funcionalidad que tendrían esas vistas para actualizar la información mostrada en pantalla.

Se crearon diferentes clases para realizar la conexión con la base de datos utilizando Web Services para acciones diferentes de la aplicación como el registro de usuarios, el registro de productos, consulta y modificación de productos, posteriormente se crearon clases que comunicarán la vista con las clases que contenían la programación del modelo de negocios.

Se decidió utilizar el patrón de diseño Model View ViewModel (MVVM) por ser una arquitectura robusta que resuelve varias problemáticas que otros patrones de diseño no pueden, el MVVM permite tener separado en 3 capas el desarrollo del proyecto lo que permitió que el código pueda ser mantenido sin tantos problemas y escalable para trabajos futuros.

En el parte de Model se agregó toda la lógica de negocios de la aplicación como el registro de usuarios, el manejo de inventarios, la conexión con la base de datos, los procesos de venta,

etc. En esta capa se realizaron todas las validaciones necesarias para mantener la seguridad de la aplicación.

La capa View contiene todas las vistas que se le mostrarán al usuario con las que puede interactuar, esta capa se encarga solo de mostrar información al usuario al interactuar con la aplicación, por lo que tener separada las vistas de todo lo demás beneficia a la hora de hacer una actualización a las interfaces en un futuro.

ViewModel adquiere y mantiene la información necesaria para que la vista de la aplicación se mantenga activa. La View debe poder detectar los cambios en el ViewModel para poder actualizar la información que se muestra en pantalla, la única función de esta capa es administrar los datos de la interfaz.

Por último, se realizaron pruebas de validación para comprobar que funcionaran todos los aspectos de la aplicación, en estas pruebas se han encontrado errores de conexión y de almacenamiento de información en la base de datos, debido a esto no es posible realizar pruebas de integración.

Capítulo IV. Resultados y conclusiones.

4.1 Resultados.

Durante el desarrollo del este trabajo y hasta su finalización se obtuvieron varios resultados siguiendo la metodología eXtreme Programming, siguiendo los pasos que marca la XP los resultados se dividen en 4 puntos que son en las fases que se fueron obteniendo los resultados.

4.1.1 Planificación

Al realizar el análisis de los pros y contras de las e-commerce más usadas en la región y con la información obtenida se realizaron las siguientes tablas de requerimientos

La siguiente tabla de consultas muestra todos los requisitos necesarios para que la aplicación puedo tener un manejo de información y saber que datos son los que se tienen que manejar en cada módulo dentro de la aplicación (ver Tabla 2).

Lista de requerimientos			
Grupo	Clave	Nombre	Descripción
Consulta/Informes	RC1	Informe de ventas.	El vendedor podrá obtener informes de sus ventas realizadas.
	RC2	Consulta de productos.	Los usuarios podrán realizar la búsqueda de productos disponibles dentro de la aplicación.
	RC3	Información del negocio.	Los clientes podrán ver información del negocio al que le quieran comprar algún producto.
	RC4	Consulta de datos de contacto.	Al realizar una compra tanto el cliente como el vendedor recibirán los datos de contacto de ambas partes.

Tabla 2. Lista de requerimientos de consultas.

La tabla de almacenamiento muestra los datos que tienen que ser almacenados en el servidor en este caso en Firebase Cloud, en la tabla se puede apreciar los datos que deben ser almacenados dependiendo del procedimiento que se esté haciendo en la aplicación, estos datos al ser guardados pueden ser consultados después según sea necesario, es importante llevar a cabo el correcto almacenamiento de esta información, ya que hay información privada de los usuarios (ver Tabla 3).

Lista de requerimientos			
Grupo	Clave	Nombre	Descripción
Almacenamiento	RA1	Alta y/o baja de productos.	Fecha, hora, nombre, precio, cantidad.
	RA2	Alta de usuarios	Nombre, usuario, dirección, correo.
	RA3	Compra/ventas realizadas	Fecha, hora, precio, cantidad, cliente, vendedor.
	RA4	Información del cliente	Nombre, dirección, correo, contacto.
	RA5	Información del negocio	Ubicación, horarios de atención, producto, información de contacto.

Tabla 3. Lista de requerimientos de almacenamiento.

En la última tabla que es la de procesamiento se muestran las acciones que la aplicación realiza utilizando y consultando la información almacenada en el servidor para poder llevar a cabo los procesos requeridos, en la tabla se puede apreciar cada acción con la descripción de lo que debe de hacer (ver Tabla 4).

Lista de requerimientos			
Grupo	Clave	Nombre	Descripción
Procesamiento	RP1	Manejo de inventario	La aplicación permite el manejo de inventario de forma automatiza al realizar ventas.
	RP2	Proceso de compra/venta	La aplicación se encargará de verificar la información durante el proceso de compra/venta.
	RP3	Inicio de sesión	La aplicación permite identificar el tipo de usuario y darle acceso a la aplicación dependiendo de su rol.
	RP4	Cierre de sesión	Le permite al usuario cerrar la sesión activa en la aplicación.
	RP5	Registro de usuarios.	Solicita la información necesaria al usuario para crear una cuenta nueva.
	RP6	Envió de productos	La aplicación permite realizar un método de envió acordado por el cliente y el vendedor.

Tabla 4. Lista de requerimientos de procesamiento.

Después de obtener estas listas de requerimientos se procedió con la realización de los siguientes casos de uso para ver el comportamiento e interacción que debe cubrir la aplicación para la satisfacción de sus usuarios.

Se obtuvo un modelo de casos de uso general para representar las interacciones entre el cliente y el vendedor (ver Figura 6) con esto se planeó la funcionalidad que debía tener la aplicación.

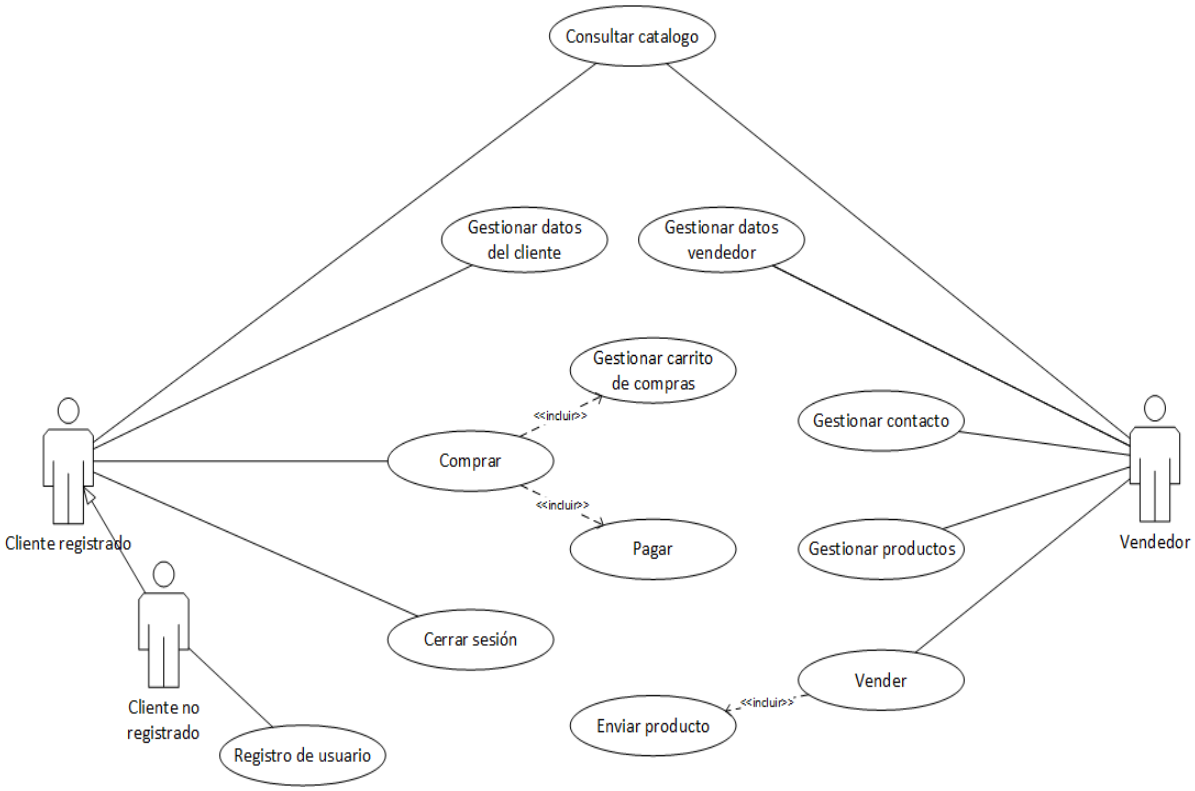


Figura 6. Caso de uso: Diseño general.

Este modelo se dio por la necesidad de representar las interacciones necesarias entre un cliente y un vendedor al momento de usar la aplicación, como se puede observar en la Figura 6, se detalla el funcionamiento del cliente como la gestión de sus datos, la vista de productos y el proceso de compra, mientras que el vendedor puede gestionar sus productos, sus datos, y realizar las ventas y envíos, la Tabla 5 muestra más detalles acerca de este modelo.

Descripción de caso de uso	
Nombre	Caso de uso general
Actores	Cliente registrado, cliente no registrado, vendedor.
Función	Proceso general de la aplicación, compra-venta de productos, administración de la información.
Descripción	Mostrar una vista general del funcionamiento de la aplicación, de la forma en la que interactúa el cliente con el vendedor.

Tabla 5. Descripción caso de uso general.

Después se obtuvieron 4 modelos de casos de uso específicos del cliente y el vendedor para representar sus actividades dentro de la aplicación.

En la gestión de los datos del cliente, este puede consultar sus datos y si encuentra algún error o tiene que actualizarlos puede modificar esos datos (ver Figura 7).

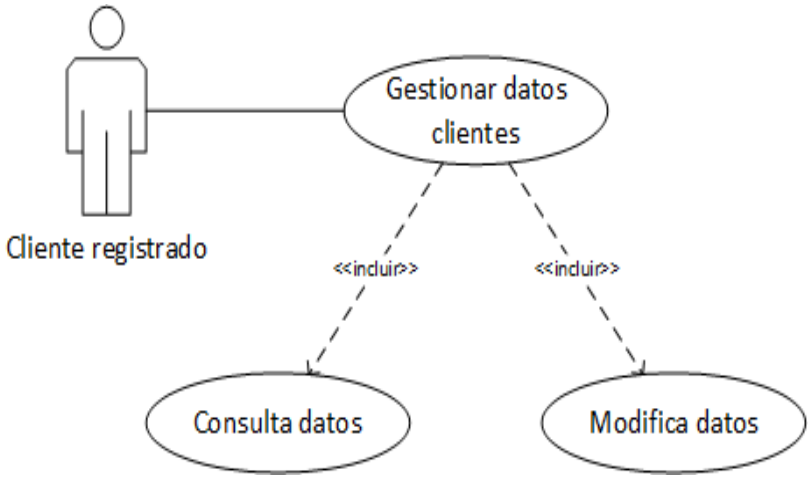


Figura 7. Caso de uso: Gestión de datos del cliente.

La Tabla 6 muestra la descripción y las funciones que debe realizar, este modelo cubre los requerimientos RC4 y RA4 de la lista de requerimientos.

Descripción de caso de uso	
Nombre	Gestión de datos del cliente.
Actores	Cliente registrado
Función	Consultar y modificar la información del cliente
Descripción	Mostrar una vista donde el usuario pueda consultar su información registrada y una opción para poder cambiar la información en caso de ser necesario.
Requerimientos	RC4, RA4 (ver Tabla 2 y Tabla 3Tabla 2. Lista de requerimientos).

Tabla 6.Descripción gestión de datos del cliente.

Posteriormente se diseñó el modelo de casos de uso del vendedor y sus interacciones que debe hacer. El vendedor puede consultar sus datos de contacto de la tienda que dio de alta en la aplicación, si lo cree necesario puede modificar su ubicación, el número telefónico o la descripción para que refleje de mejor manera el perfil de su negocio (ver Figura 8).

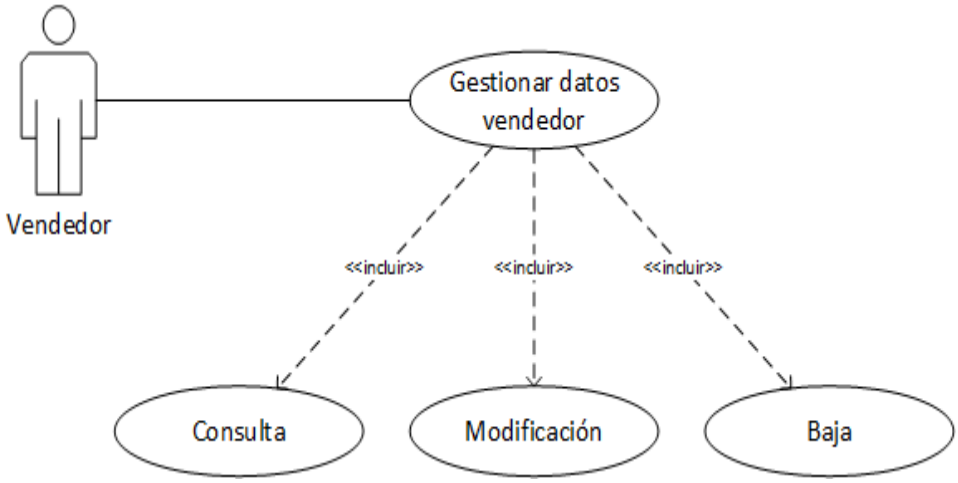


Figura 8. Caso de uso: Gestión de datos del vendedor.

En caso de que el vendedor ya no quiera seguir haciendo uso de la aplicación puede darse de baja y automáticamente toda la información del negocio será eliminada cancelando cualquier venta que este en proceso, ya no será visible la tienda ni los productos que tenía registrados, estos detalles se pueden ver en la Tabla 7.

Descripción de caso de uso	
Nombre	Gestión de datos de vendedor.
Actores	Vendedor.
Función	Consultar y modificar la información del vendedor o solicitar su baja de la aplicación.
Descripción	Mostrar una vista donde el vendedor pueda consultar su información registrada y una opción para poder cambiar la información en caso de ser necesario y la opción de dar de baja su cuenta.
Requerimientos	RC3 (ver Tabla 2).

Tabla 7.Descripción de gestión de datos del vendedor.

Luego se diseñó el modelo de gestión de productos en este caso de uso un vendedor puede dar de alta nuevos productos, ingresando la información necesaria como el nombre, descripción, precio y cantidad, también puede modificarlos o eliminarlos si es necesario, en caso de ya no contar con algún producto o que ya no lo quiera vender más dentro de la aplicación puede darlo de baja (ver Figura 9).

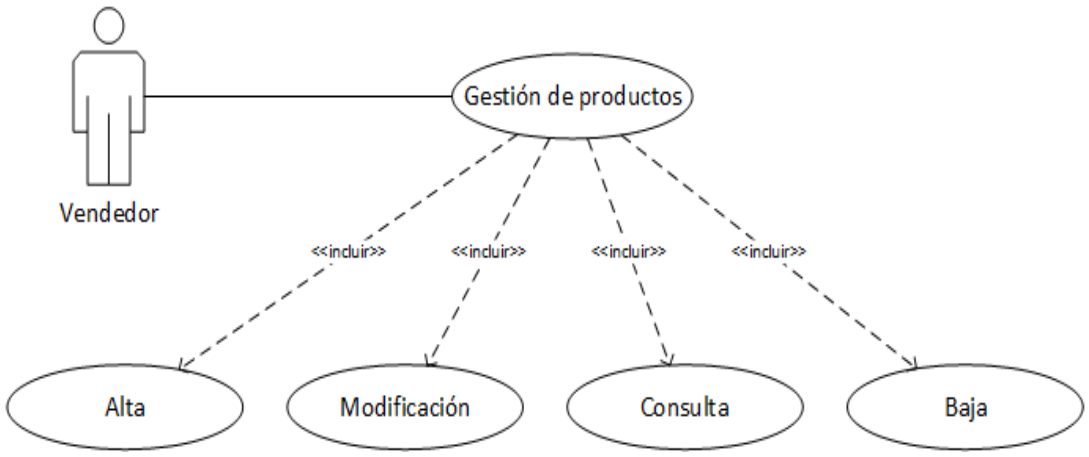


Figura 9. Casos de uso: Gestión de productos.

En la siguiente tabla (ver Tabla 8) se pueden observar las funciones que cubre este modelo y las características que debe cubrir así como los requerimientos de las Tabla 2 y Tabla 3.

Descripción de caso de uso	
Nombre	Gestión de productos.
Actores	Vendedor.
Función	Consultar, alta, modificación y baja de productos por parte del vendedor.
Descripción	El vendedor podrá agregar productos especificando sus características, como nombre, precio y cantidad, así también modificar la información de los productos o eliminarlos de su stock.
Requerimientos	RC2, RA1 (ver Tabla 2 y Tabla 3).

Tabla 8.Descripción gestión de productos.

El proceso de venta de los productos consiste primero en verificar que el pago realizado sea correcto, cuando el pago se realizó con éxito el vendedor recibe los datos de contacto del cliente para posteriormente enviar el producto (ver Figura 10).

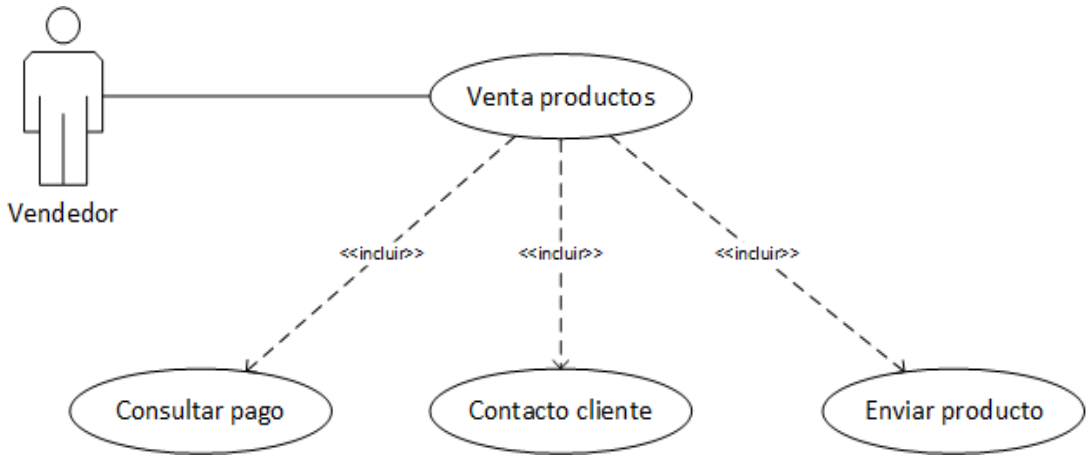


Figura 10. Casos de uso: Venta de productos.

La siguiente tabla muestra las características de la venta de productos y las funciones que debe cumplirse como se indica en las tablas de listas de requerimientos (ver Tabla 2, Tabla 3 y Tabla 4), aquí el vendedor puede consultar si el pago de una venta ha sido realizado o no, entonces le da la posibilidad de contactar al cliente para ver por qué no se ha acreditado el pago o si ya lo hizo acordar como se realizara él envió (ver Tabla 9).

Descripción de caso de uso	
Nombre	Venta productos.
Actores	Vendedor.
Función	Consultar el pago del cliente, ver información de contacto del cliente, envió del producto
Descripción	El vendedor puede realizar el proceso de venta de sus productos y realizar él envió de productos a sus clientes.
Requerimientos	RA3, RP1, RP2, RP6 (ver Tabla 2, Tabla 3 y Tabla 4).

Tabla 9.Descripción venta de productos.

Al terminar de realizar los modelos de casos de usos se planteó una arquitectura de software a seguir que se muestra a continuación (ver Figura 11).

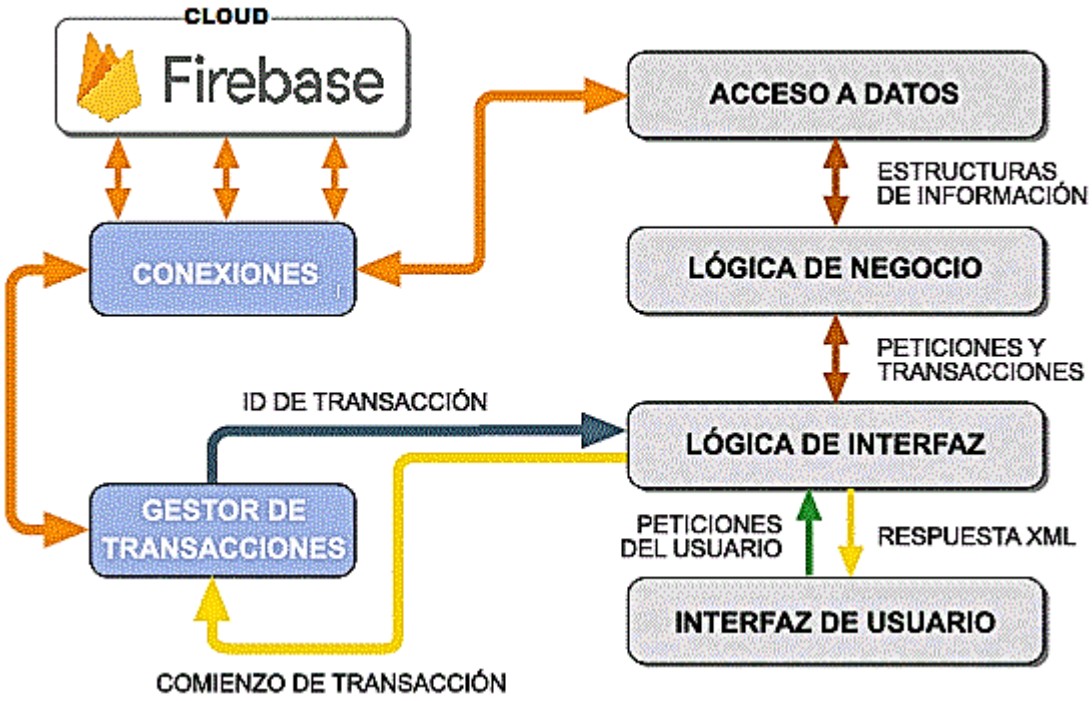


Figura 11. Arquitectura de software propuesta.

Se diseño esta arquitectura teniendo en cuenta el patrón de diseño MVVM en donde el desarrollo de código se divide en 3 capaz para poder dar un mejor mantenimiento a la aplicación y agregar funciones futuras de forma más sencilla al tener por separado la vista de la lógica de negocios.

Primero se tiene lo que es la interfaz de usuario en esta son todos los elementos con los que interactúa la persona, esta capa envía peticiones a la lógica de interfaz y esta le devuelve una respuesta que actualiza la vista y cambia los elementos que visualiza el usuario.

La lógica de interfaz se encarga de mediar entre la vista que es la interfaz de usuario y la lógica de negocios, esta capa envía y recibe peticiones para interpretarlas y mandar una respuesta en formato XML para actualizar la vista.

La lógica de negocios es la capa que contiene todo el código que hace funcionar a la aplicación, es donde se procesa la información que el usuario solicita y se procede a realizar las acciones necesarias, esta capa se encarga de preparar la información para almacenarla en un servidor o para recuperarla y manipularla dependiendo de las peticiones del usuario.

El acceso a datos tiene la función de manejar la información otorgada por el usuario ya sea para almacenarla en un servidor en este caso es Firebase o para encontrar la información que la aplicación necesita para seguir funcionando.

El gestor de transacciones se encarga de las funciones como el inicio o cierre de sesión, notificaciones de la aplicación, actualizaciones de interfaz entre otras cosas dependiendo de las funciones que se utilicen de Firebase, para este trabajo solo se usaron las mencionadas anteriormente.

Siguiendo con el diagrama de la arquitectura de software sigue la capa de conexiones, aquí solo hace referencia a los medios que se utilizan para conectarse a Internet ya sea a través de una red de telefonía móvil o de una red otorgada por un proveedor de servicios de Internet.

Por último, se tiene la nube (Cloud) que es el servicio contratado para almacenar toda la información que necesita la aplicación, en este caso se optó por utilizar el servicio de Firebase por las facilidades que ofrece y por tener funcionalidades gratuitas.

4.1.2 Diseño.

Siguiendo lo que es el diagrama de la arquitectura de software propuesto se procedió con la realización de los diagramas UML para llevar un desarrollo concreto dentro de la aplicación, se obtuvieron un total de 8 diagramas para el funcionamiento básico del sistema, estos se pueden observar en la siguiente imagen (ver Figura 12).

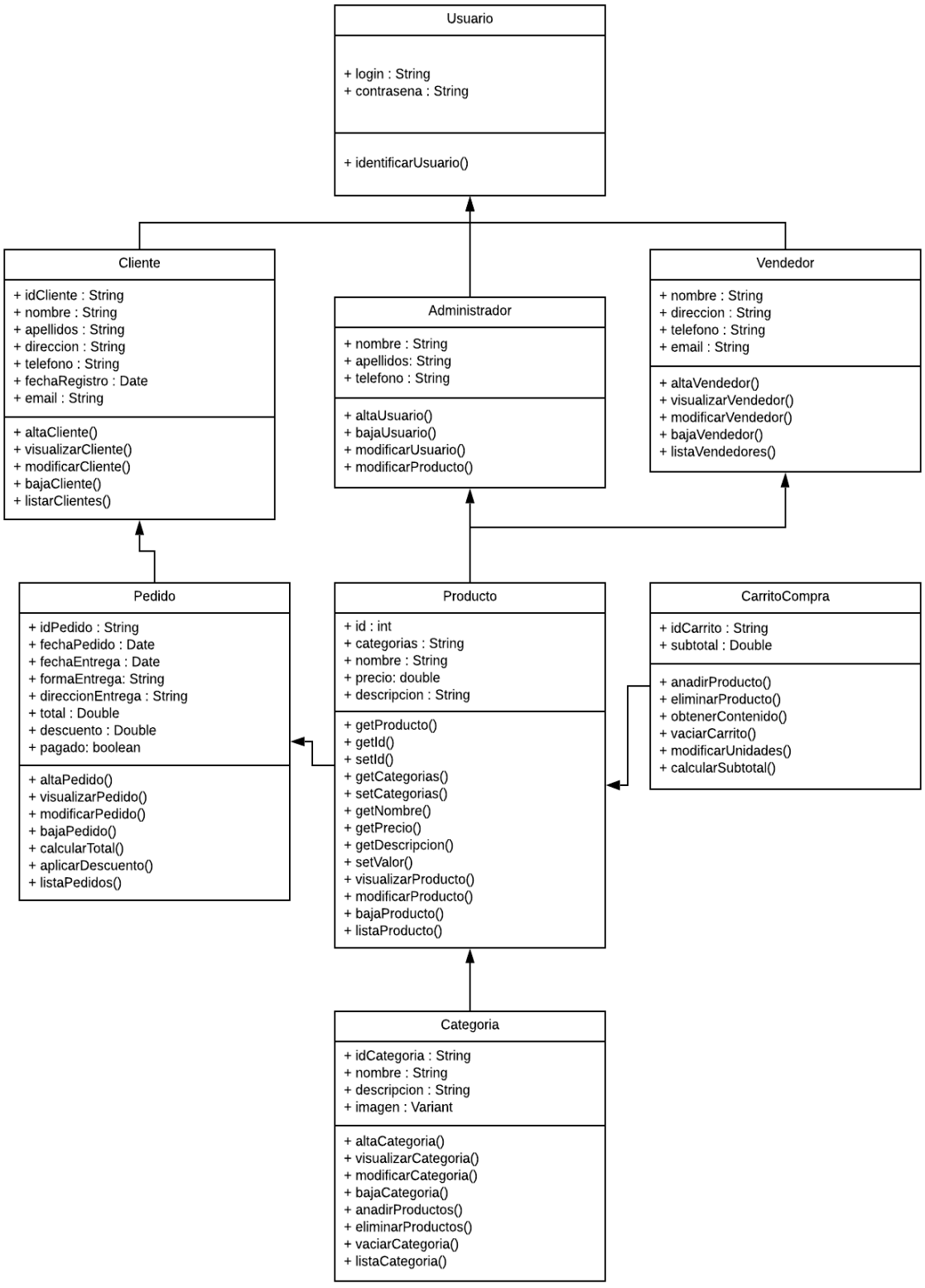


Figura 12. Diagramas UML de la aplicación.

Estos diagramas muestran las variables y métodos necesarios que deben tener las clases a desarrollar en la aplicación, empezando por la clase “Usuario” se tienen dos variables para almacenar el usuario y la contraseña, al tener estas 2 variables se accede al método identificar usuario para realizar las acciones necesarias que validen que las credenciales ingresadas son correctas.

Siguiendo las clases “Cliente”, “Vendedor” y “Administrador”, en la clase cliente se tienen variables para guardar información del cliente como su id, nombre, dirección, teléfono, fecha de registro y email, también cuenta con métodos como el alta del cliente para registrar usuarios nuevos, otro método para modificar la información de los usuarios o para darlos de baja si es necesario.

La clase administrador se encarga de dar de alta usuarios, modificarlos o darlos de baja según sea necesario o por criterios propios del administrador, también tiene la opción de cambiar la información de los productos en caso de que no esté correcta la información.

Mientras que la clase “Vendedor” permite agregar más vendedores, modificar la información de los que ya están registrados, visualizar su información ingresada al sistema, darse de baja en caso de que ya no quiera mostrar más sus productos en la aplicación y solo quedar como cliente.

Después se tiene la clase “Producto” que se encarga de agregar los productos por parte de los clientes, aquí se asignan los valores necesarios como es el id, el nombre del producto, precio, cantad o descripción, así también permite modificar la información de productos ya registrados o darlos de baja según las necesidades del vendedor, también muestra la información de un producto en concreto o una lista dependiendo de los criterios de búsqueda.

También se tiene la clase “Categoria”, esta se encarga de asignar a cada producto registrado una categoría, tiene métodos que permiten mostrar los productos dependiendo de a qué grupo fue asignado, también permite agregar más productos a diferentes categorías o quitarlos en caso de que no correspondan a ese grupo.

La clase “CarritoCompra” permite añadir productos a una lista temporal generada por el usuario, esta lista se guarda mientras el usuario tenga la sesión iniciada, la clase permite modificar la cantidad de productos en la lista, quitar los que el usuario ya no quiera y calcular el valor total de todos los artículos que se encuentran agregados.

Por último, se tiene la clase “Pedido” esta tiene métodos que se encargan de llevar a cabo las compras realizadas por los clientes, permite generar pedidos nuevos, modificarlos en caso de que ya no se requiera algún artículo o cancelarlo si ya no se quiere realizar la compra, permite la visualización en forma de lista de todos los artículos que se van a comprar así como calcular el total de todo los productos incluso si hay algún descuento valido, todo eso de forma automática.

Después de obtener la arquitectura de software y los diagramas UML se realizaron 2 modelos de negocios uno que representa el alcance del proyecto realizado y otro del modelo general de la aplicación.

El primer modelo describe las funciones de que un usuario puede registrarse e iniciar sesión en la aplicación, una vez dentro de esta puede ver los productos existentes y simular la compra de los productos, esto afectara a la cantidad de productos que existentes en el inventario, al agregar productos a la lista de compra es posible modificar la cantidad requerida o eliminar un artículo, el usuario también tiene la posibilidad de agregar productos y estos se muestren en la pantalla principal, al agregar productos debe ingresar la información necesaria y esta será verificada por el sistema para que sea un producto valido, también el usuario puede dar de baja los productos que el registrara anteriormente, puede ver una lista de todos los productos agregados y eliminar alguno si ya no lo quiere mostrar dentro de la aplicación, todo esto se puede observar en la siguiente imagen (ver Figura 13) que representa el desarrollo actual del proyecto.

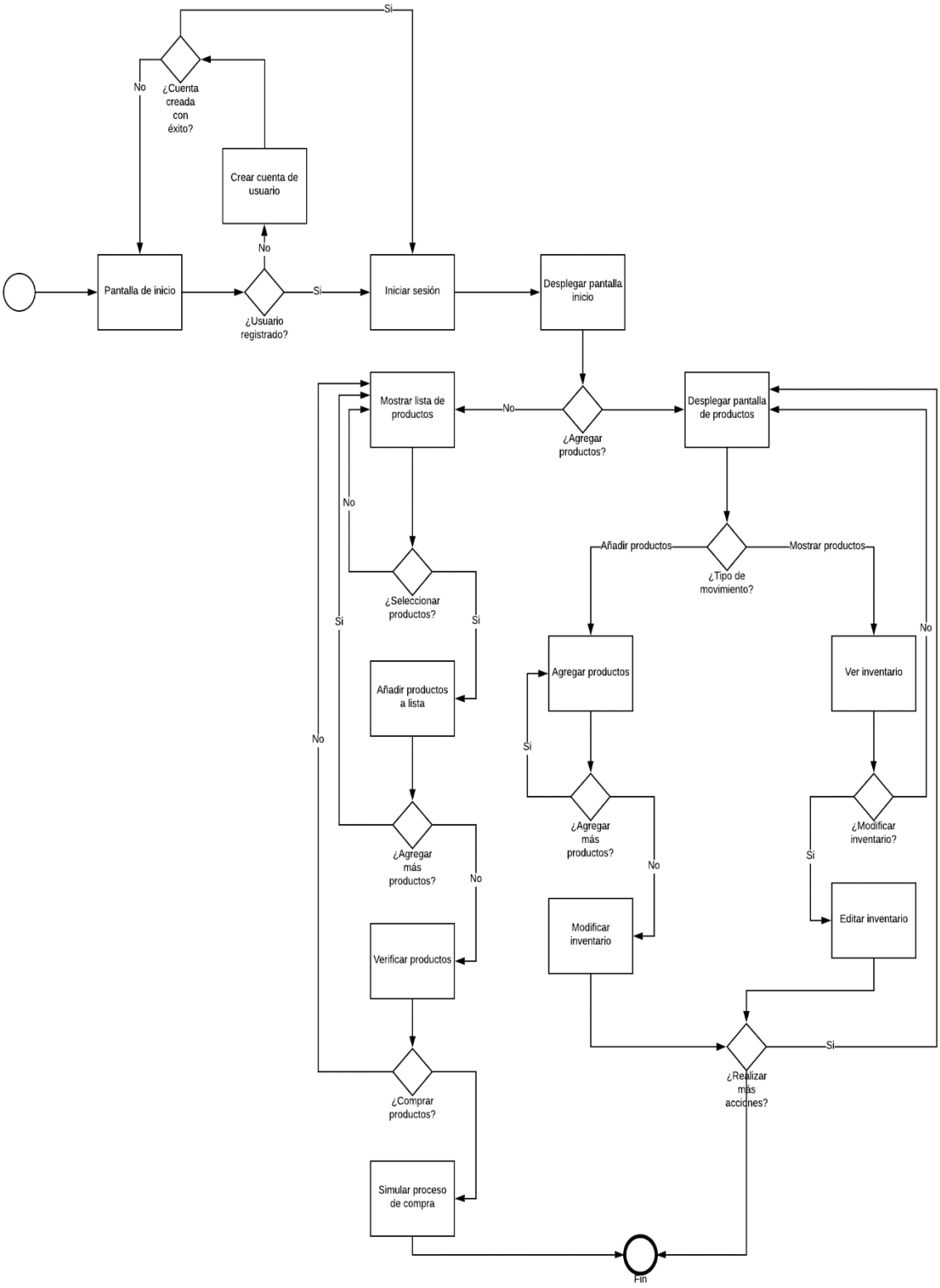


Figura 13. Modelo de negocios desarrollado.

El siguiente modelo de negocios se realizó para recopilar la información necesaria para visualizar el funcionamiento general que deberá cumplir la aplicación, para este proyecto se obtuvo el siguiente modelo de negocios (ver Figura 14).

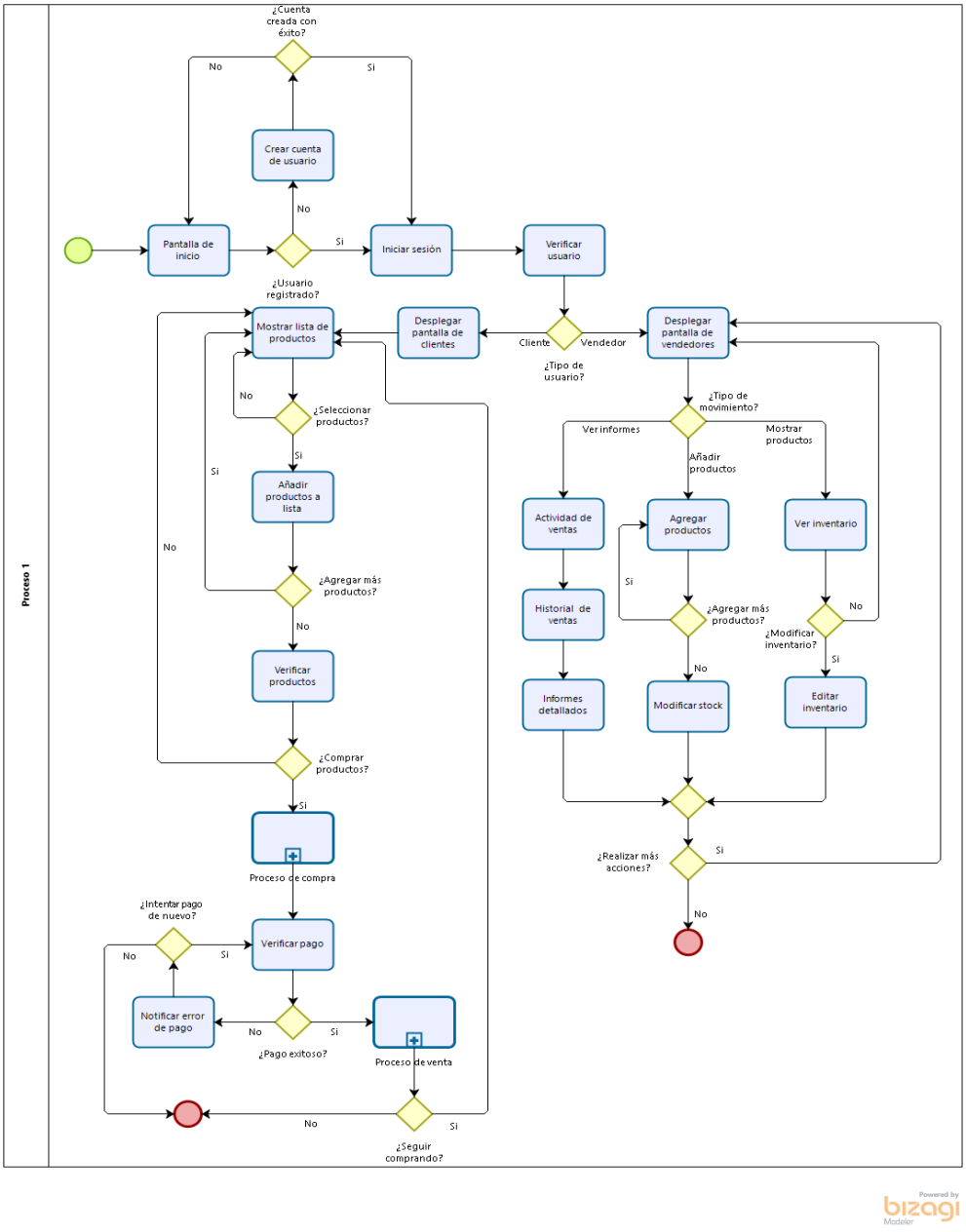


Figura 14. Modelo de negocios general

El modelo de negocios empieza en la pantalla de inicio de la aplicación donde le pedirá al usuario que inicie sesión y en caso de estar registrado le pedirá crear una cuenta nueva, en el proceso de crear cuenta nueva el usuario tendrá la opción de sólo poder comprar dentro de la aplicación o registrar su negocio para vender sus productos dentro de la app, después de registrarse el usuario podrá iniciar sesión y se verificará que tipo de usuario es ya sea cliente o vendedor para mostrar una interfaz optimizada para estos 2 tipos de usuarios.
Estos pasos se muestran en la siguiente imagen (ver Figura 15).

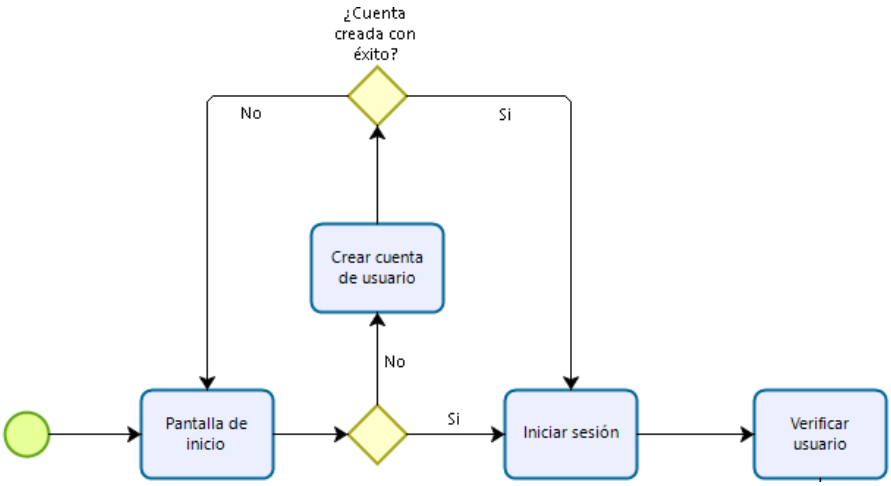


Figura 15. Modelo de negocios procesos de sesión de usuario

Posteriormente (ver Figura 16), si el usuario es un cliente entonces, se mostrará una interfaz con una lista de productos donde podrá buscar los que le interese adquirir, el cliente podrá agregar a una lista de compra los productos que este por comprar, cuando termine de seleccionar los productos se verificará el stock de los que eligió, cuando se apruebe la disponibilidad de todos los productos seleccionados podrá realizar la compra o seguir viendo más.

Cuando se procede a realizar la compra entonces se accede al subproceso “Proceso de compra” en este proceso se calcula el total de la compra, se verifica la información de envió y se procede a generar el método de pago para el usuario.

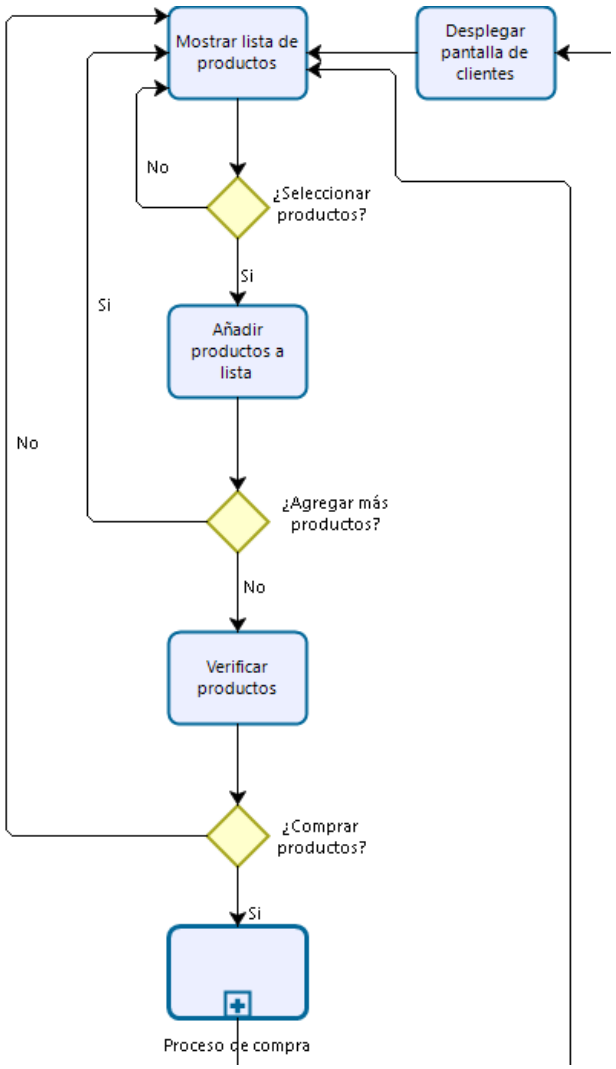


Figura 16. Modelo de negocios procesos del cliente

Cuando el pago se verifique y se haya realizado con éxito entonces, se entra en el subproceso de “Proceso de venta” donde se aparta el stock de la compra y se le avisa al vendedor la información de los productos seleccionados y la información del cliente para realizar él envió de los productos, cuando se notifique que el cliente tiene los productos que solicitó sin contratiempos o desperfectos el costo de la venta se verá reflejado en la cuenta del vendedor para que lo pueda retirar por uno de los medios que estén disponibles (ver Figura 17).

En caso de que no se haya podido realizar el pago de los productos se enviará una notificación al cliente con el error del pago, entonces el cliente decidirá si verificar el pago de nuevo o finalizar ahí el proceso.

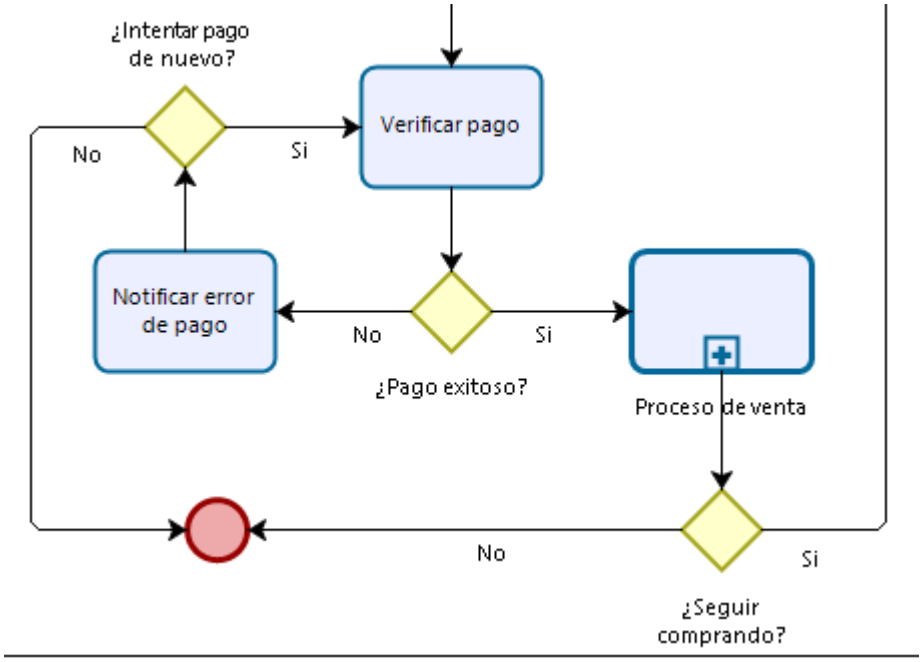


Figura 17. Modelo de negocios procesos de venta

En caso de que el usuario sea un vendedor (Ver Figura 18) puede realizar 3 acciones en su interfaz de vendedor que serán el alta de productos, la modificación o eliminación de stock e informes de sus ventas realizadas.

En la parte de informes puede ver la actividad de sus ventas semanales o mensuales, posteriormente se mostrará una gráfica con el historial de las ventas realizadas y por último un informe detallado de las ventas diarias.

En el alta de productos el vendedor puede agregar productos a su stock para que estén disponibles dentro de la aplicación y los clientes puedan encontrar estos productos.

Por último, está la parte del inventario donde el vendedor modificará o eliminará el stock de sus productos según sea necesario, una vez el vendedor haga las modificaciones necesarias y acepte los cambios el stock se modificará.

Si se requieren más acciones entonces vuelve de nuevo a la interfaz del vendedor en caso de que no entonces finaliza el proceso.

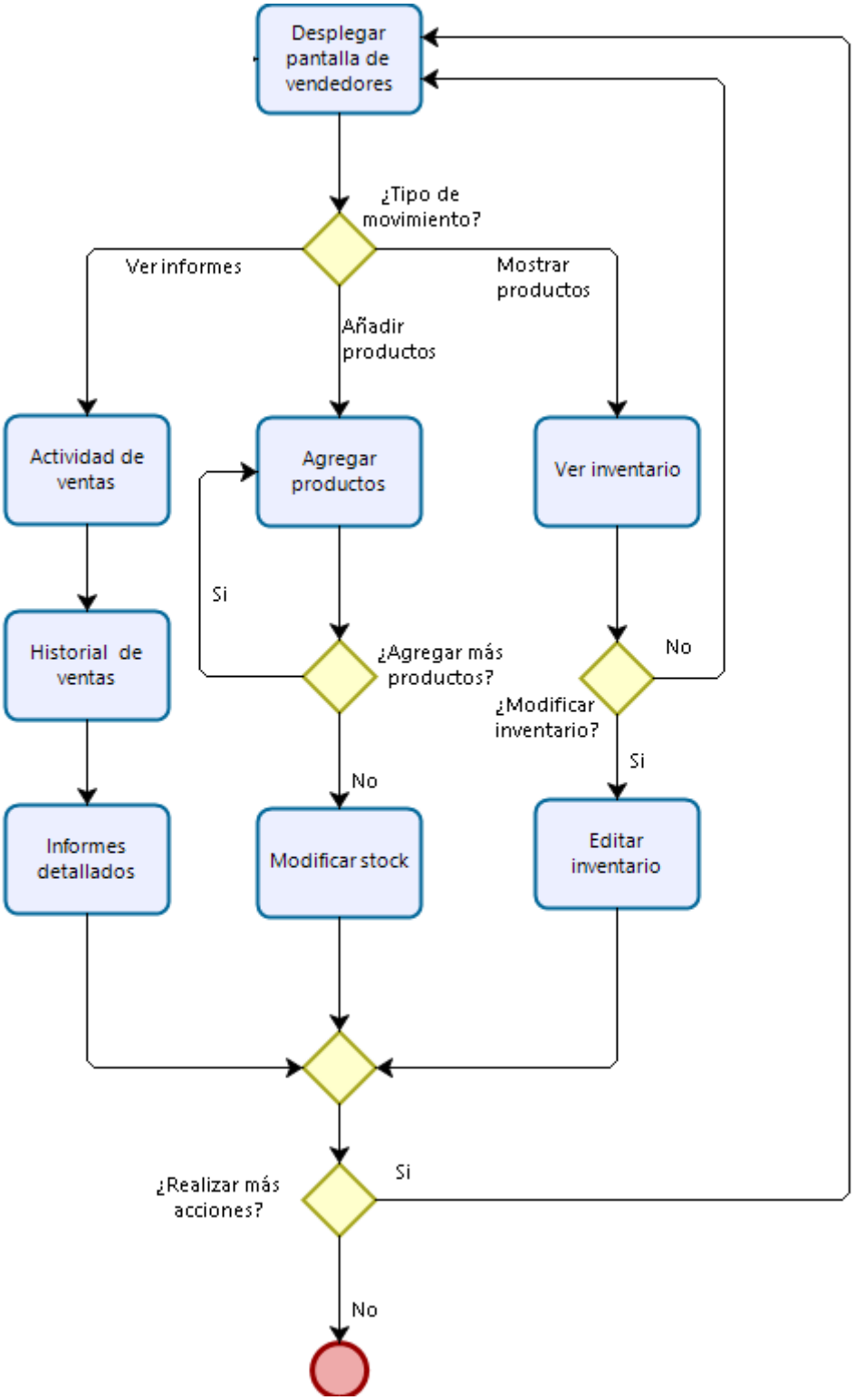


Figura 18. Modelo de negocios procesos del vendedor.

4.1.3 Codificación.

Se crearon un total de 11 Web Services en el lenguaje PHP para la gestión de la información entre la base de datos y la aplicación.

Consultar usuarios

El propósito de este código es consultar en la base de datos si el usuario ingresado en la aplicación se encuentra registrado o no, esto se usó tanto para el manejo del inicio de sesión y el registro de usuarios (Ver Figura 19).

```php
<?php
$hostname_localhost="localhost";
$database_localhost="systock";
$username_localhost="root";
$password_localhost="";
$json=array();
  if(isset($_GET["correo"])){
    $correo=$_GET['correo'];
    $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhc
    $consulta="select nombre_usuario, apellidos_usuario, correo, contrasenia from usua
    $resultado=mysqli_query($conexion,$consulta);
    if($registro=mysqli_fetch_array($resultado)){
      $json['usuario'][]=$registro;
    }else{
      $resultar["nombre"]='no registra';
      $resultar["apellidos"]='no registra';
      $resultar["contrasenia"]='no registra';
      $json['usuario'][]=$resultar;
    }
    mysqli_close($conexion);
    echo json_encode($json);
  }
?>
```

Figura 19. Web ServicesconsultarUsuarios.php.

Registro de usuarios

Esta parte se programó para realizar el registro de un nuevo usuario en la base de datos, validando que el correo ingresado no se encuentra ya registrado en el sistema, si el correo está ya registrado retornará un error, de lo contrario la información se envía a la base de datos (Ver Figura 20).

```php
if(isset($_GET["nombre"]) && isset($_GET["apellidos"]) && isset($_GET["correo"]) && isset($_GET["contrasenia"])){
  $nombre=$_GET['nombre'];
  $apellidos=$_GET['apellidos'];
  $correo=$_GET['correo'];
  $contrasenia=$_GET['contrasenia'];

  $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
  $consulta="select correo from usuario where correo = '{$correo}'";
  $resultado=mysqli_query($conexion,$consulta);
  if(!$registro=mysqli_fetch_array($resultado)){
    $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
    $insert="INSERT INTO usuario(nombre_usuario, apellidos_usuario, correo, contrasenia) VALUES ('{$nombre}','{$apellidos}','{$corre
    $resultado_insert=mysqli_query($conexion,$insert);

    if($resultado_insert){
      $consulta="SELECT * FROM usuario WHERE correo = '{$correo}'";
      $resultado=mysqli_query($conexion,$consulta);

      if($registro=mysqli_fetch_array($resultado)){
        $json['usuario'][]=$registro;
      }
      mysqli_close($conexion);
      echo json_encode($json);
    }
    else{
      $resulta["nombre"]='No Registra';
      $resulta["apellidos"]='No Registra';
      $resulta["correo"]='No Registra';
```

Figura 20. Web ServicesregistrarUsuarios.php.

Registro de productos

Parte de código que se encarga del registro de productos captando la información enviada desde la aplicación para almacenarla en la base de datos (ver Figura 21).

```php
if(isset($_GET["nombre"]) && isset($_GET["descripcion"]) && isset($_GET["cantidad"]) && isset($_GET["precio"])){
  $nombre=$_GET['nombre'];
  $descripcion=$_GET['descripcion'];
  $cantidad=$_GET['cantidad'];
  $precio=$_GET['precio'];

    $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
    $insert="INSERT INTO productos(nombre_producto, descripcion, stock, precio) VALUES ('{$nombre}','{$descripcion}','{$cantic
    $resultado_insert=mysqli_query($conexion,$insert);

    if($resultado_insert){
      $consulta="SELECT * FROM productos WHERE nombre_producto = '{$nombre}'";
      $resultado=mysqli_query($conexion,$consulta);

      if($registro=mysqli_fetch_array($resultado)){
        $json['producto'][]=$registro;
      }
      mysqli_close($conexion);
      echo json_encode($json);
    }
    else{
      $resulta["nombre"]='No Registra';
      $resulta["descripcion"]='No Registra';
      $resulta["cantidad"]='No Registra';
      $resulta["precio"]='No Registra';
      $json['producto'][]=$resulta;
      echo json_encode($json);
```

Figura 21. Web ServicesregistrarProductos.php.

Modificar producto

Actualiza la información de los productos almacenada en la BD con los datos nuevos provenientes de la aplicación (ver Figura 22).

```php
if(isset($_GET["id"]) && isset($_GET["nombre"]) && isset($_GET["descripcion"]) && isset($_GET["cantidad"]) && isset($_GET["pre
  $id=$_GET['id'];
  $nombre=$_GET['nombre'];
  $descripcion=$_GET['descripcion'];
  $cantidad=$_GET['cantidad'];
  $precio=$_GET['precio'];
    $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
    $insert="UPDATE productos SET nombre_producto='{$nombre}', descripcion='{$descripcion}', stock='{$cantidad}', precio='{$pr
    $resultado_insert=mysqli_query($conexion,$insert);
    if($resultado_insert){
      $consulta="SELECT * FROM productos WHERE id_producto = '{$id}'";
      $resultado=mysqli_query($conexion,$consulta);

      if($registro=mysqli_fetch_array($resultado)){
        $json['producto'][]=$registro;
      }
      mysqli_close($conexion);
      echo json_encode($json);
    }
    else{
      $resulta["nombre"]='No Registra';
      $resulta["descripcion"]='No Registra';
      $resulta["cantidad"]='No Registra';
      $resulta["precio"]='No Registra';
      $json['producto'][]=$resulta;
      echo json_encode($json);
    }
```

Figura 22. Web ServicesmodificarProducto.php.

Limpiar carrito de compras

Elimina toda la información del carrito de compras que se crea temporalmente en la base de datos mientras la sesión del usuario está activa (ver Figura 23).

```php
<?php
$hostname_localhost="localhost";
$database_localhost="systock";
$username_localhost="root";
$password_localhost="";
$json=array();
    $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
    $consulta="DELETE FROM carrito";
    $resultado=mysqli_query($conexion,$consulta);
    mysqli_close($conexion);
    echo json_encode($json);
?>
```

Figura 23. Web ServiceslimpiarCarro.php.

Eliminar ítems del carrito de compras

Cuando el usuario quiere eliminar productos del carrito de compras este fragmento de código se encarga de eso, se localiza el id del producto a eliminar y se procede a quitarlo de la lista (ver Figura 24).

```php
<?php
$hostname_localhost="localhost";
$database_localhost="systock";
$username_localhost="root";
$password_localhost="";
$json=array();
    if(isset($_GET["id"])){
    $id=$_GET['id'];
    $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
    $consulta="DELETE FROM carrito WHERE idProducto = '{$id}'";
    $resultado=mysqli_query($conexion,$consulta);
    $resulta["id"]=$id;
    $json['delete'][]=$resulta;
    echo json_encode($json);
  }
?>
```

Figura 24. Web ServiceseliminarProductoCarrito.php.

Eliminar producto

Clase encargada de eliminar los productos que el vendedor ya no necesite en su inventario, al eliminar el producto ya no será posible verlo dentro de la aplicación (ver Figura 25).

```php
$hostname_localhost="localhost";
$database_localhost="systock";
$username_localhost="root";
$password_localhost="";

$json=array();

  if(isset($_GET["nombre"])){
    $nombre=$_GET['nombre'];

    $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
    $consulta="select * from productos where nombre_producto = '{$nombre}'";
    $resultado=mysqli_query($conexion,$consulta);

    while($registro=mysqli_fetch_array($resultado)){
      $json['productosE'][]=$registro;
      //echo $registro['id'].' - '.$registro['nombre'].'<br/>';
    }
    mysqli_close($conexion);
    echo json_encode($json);

    }else{
      $resultar["nombre"]='no registra';
      $resultar["apellidos"]='no registra';
      $resultar["contrasenia"]='no registra';
      $json['usuario'][]=$resultar;
    }
```

Figura 25. Web ServiceseliminarProducto.php.

Consultar lista de productos

Se obtiene una lista de todos los productos disponibles almacenados en la base de datos para mostrarlos en la pantalla que se solicite esa información (ver Figura 26).

```php
<?php
$hostname_localhost="localhost";
$database_localhost="systock";
$username_localhost="root";
$password_localhost="";
$json=array();
    $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
    $consulta="select * from productos";
    $resultado=mysqli_query($conexion,$consulta);
    while($registro=mysqli_fetch_array($resultado)){
      $json['productosC'][]=$registro;
      //echo $registro['id'].' - '.$registro['nombre'].'<br/>';
    }
    mysqli_close($conexion);
    echo json_encode($json);
?>
```

Figura 26. Web ServicesconsultarListaProductos.php.

Consultar carrito de compras

Devuelve un objeto JSON con todos los productos que se encuentran en el carrito de compras para mostrarlos al usuario (ver Figura 27).

```php
<?php
$hostname_localhost="localhost";
$database_localhost="systock";
$username_localhost="root";
$password_localhost="";
$json=array();
    $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
    $consulta="select * from carrito";
    $resultado=mysqli_query($conexion,$consulta);
    while($registro=mysqli_fetch_array($resultado)){
      $json['carritoC'][]=$registro;
      //echo $registro['id'].' - '.$registro['nombre'].'<br/>';
    }
    mysqli_close($conexion);
    echo json_encode($json);
?>
```

Figura 27. Web ServicesconsultarCarrito.php.

Manejo del carrito de compras

Se encarga de almacenar todos los productos que agregue el cliente al carrito de compras mientras la sesión de ese usuario este activa (ver Figura 28).

```php
<?php
    $hostname_localhost="localhost";
    $database_localhost="systock";
    $username_localhost="root";
    $password_localhost="";

    $json=array();

      if(isset($_GET["id"]) && isset($_GET["nombre"]) && isset($_GET["cantidad"]) && isset($_GET["descripcion"]) && isset($_GET["pr
        $id=$_GET['id'];
        $nombre=$_GET['nombre'];
        $cantidad=$_GET['cantidad'];
        $descripcion=$_GET['descripcion'];
        $precio=$_GET['precio'];
        $cantidadTotal=$_GET['cantidadTotal'];

          $conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);
          $insert="INSERT INTO carrito(idProducto, nombre, cantidad, descripcion, precio, cantidadTotal) VALUES ('{$id}','{$nombre}
          $resultado_insert=mysqli_query($conexion,$insert);
          if($resultado_insert){
            $consulta="SELECT * FROM carrito WHERE idProducto = '{$id}'";
            $resultado=mysqli_query($conexion,$consulta);

            if($registro=mysqli_fetch_array($resultado)){
              $json['carrito'][]=$registro;
            }
            mysqli_close($conexion);
            echo json_encode($json);
          }
```

Figura 28. Web Servicescarrito.php.

Clases Android

Estas clases son las que componen todo el funcionamiento de la aplicación, estas están divididas en capas de acuerdo al MVVM.

Modelo

En el modelo se crearon 5 clases que contienen toda la lógica de negocios. En la Figura 29 se muestran las clases creadas en esta capa.

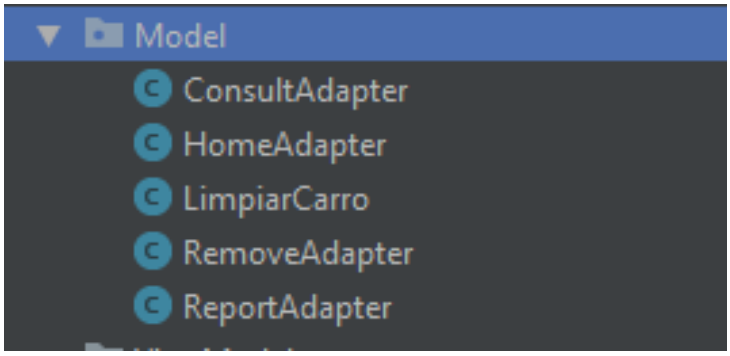


Figura 29: Clases de la capa Model.

En la siguiente parte del código, su función es, consultar la información de la base de datos para retornar y mostrar en pantalla los datos solicitados por el usuario (Ver Figura 30).

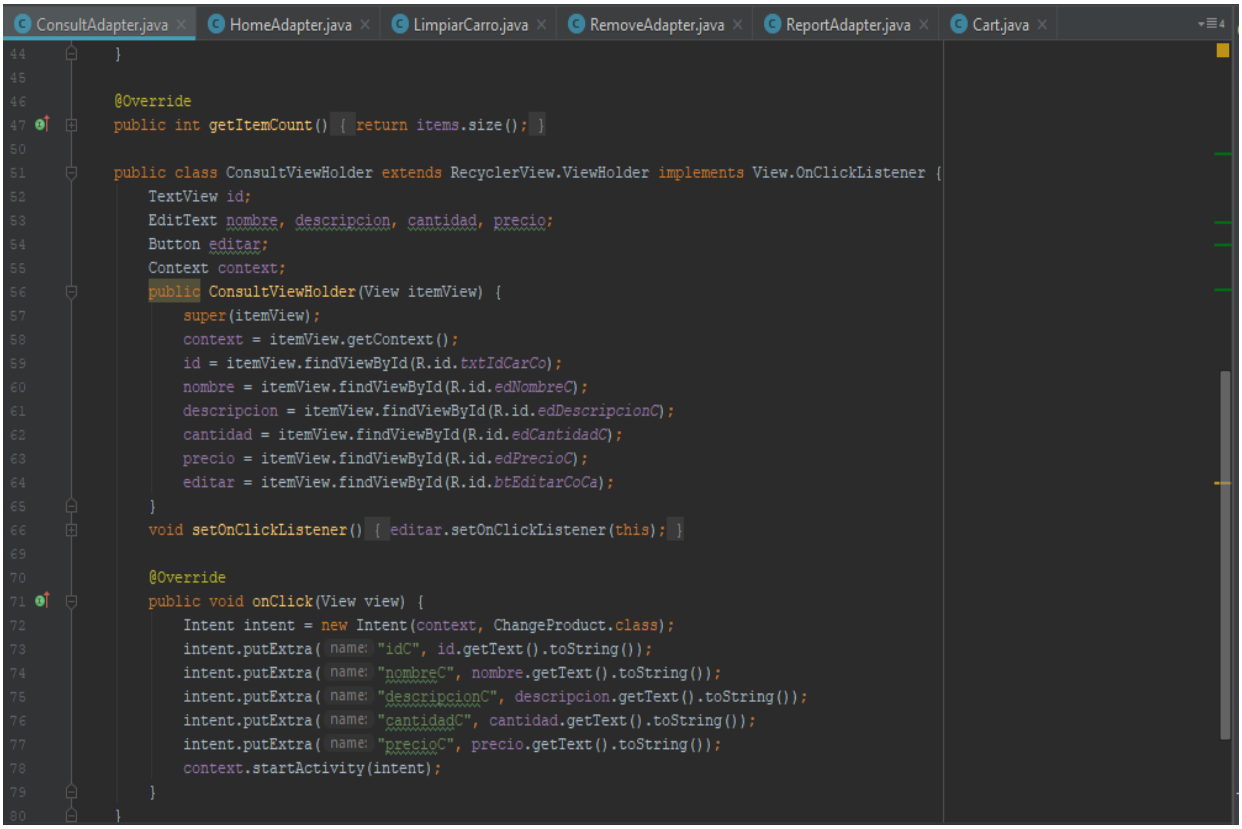


Figura 30: Clase ConsultAdapter.

A continuación, el siguiente código se encarga de mostrar los datos de la aplicación una vez que se inicia, así mismo cargar la información de un producto en específico o la gestión de eventos por parte del usuario (Ver Figura 31).

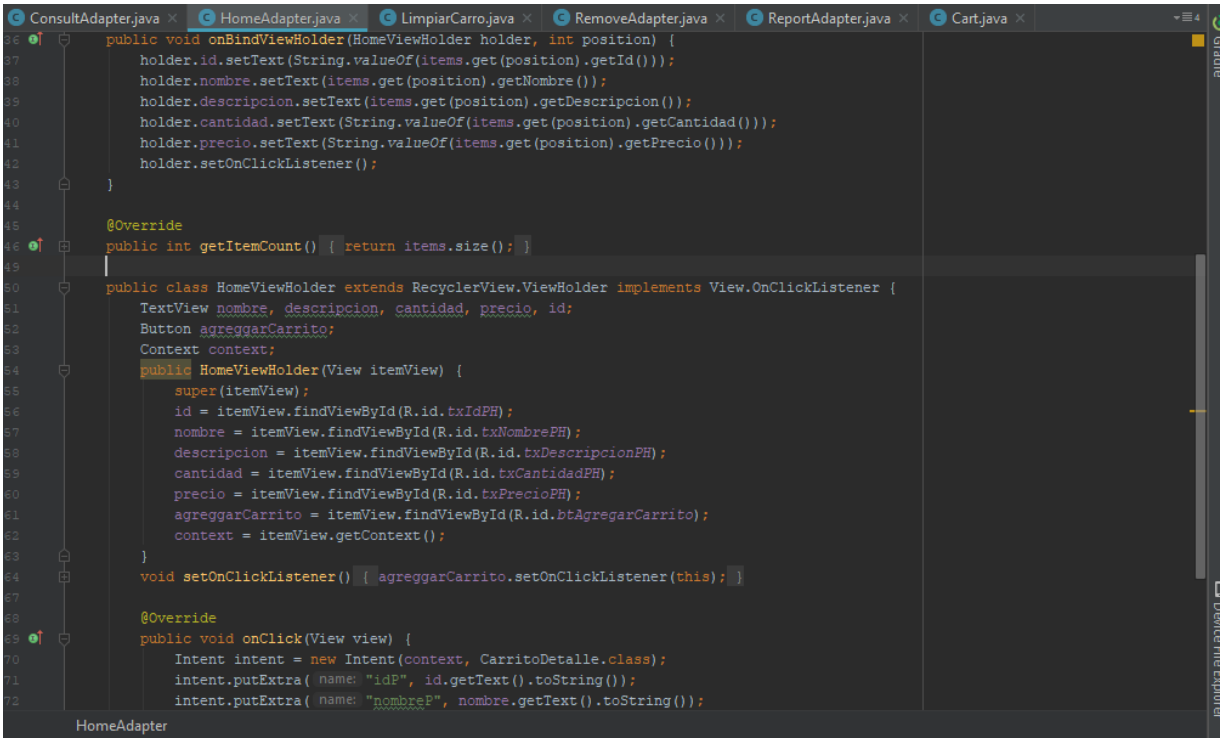
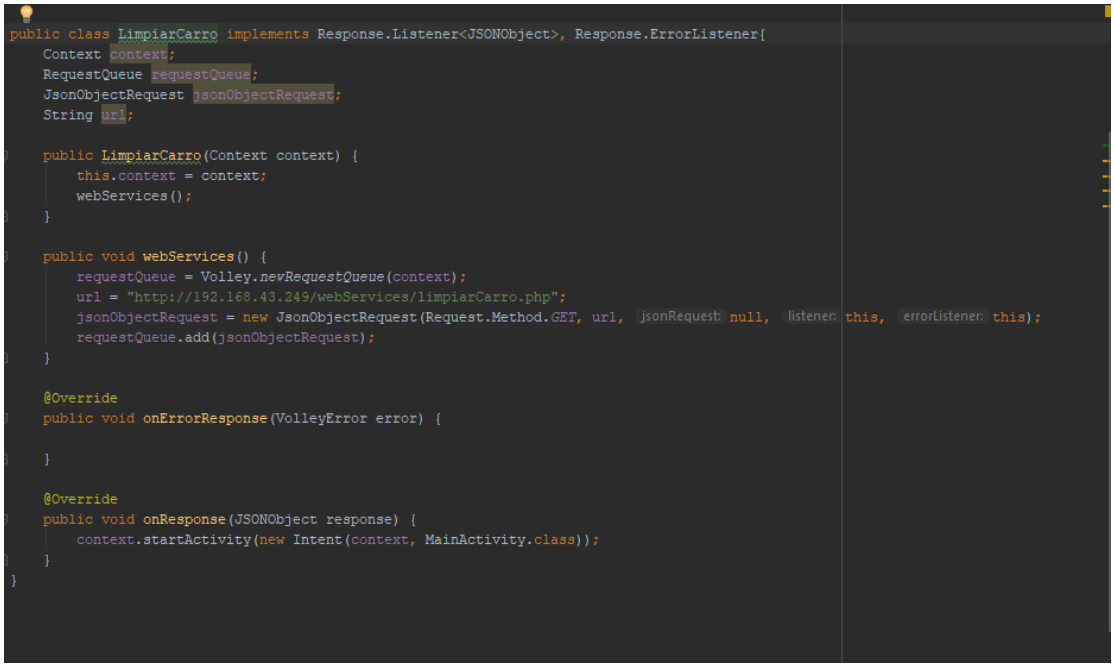


Figura 31: Clase HomeAdapter.

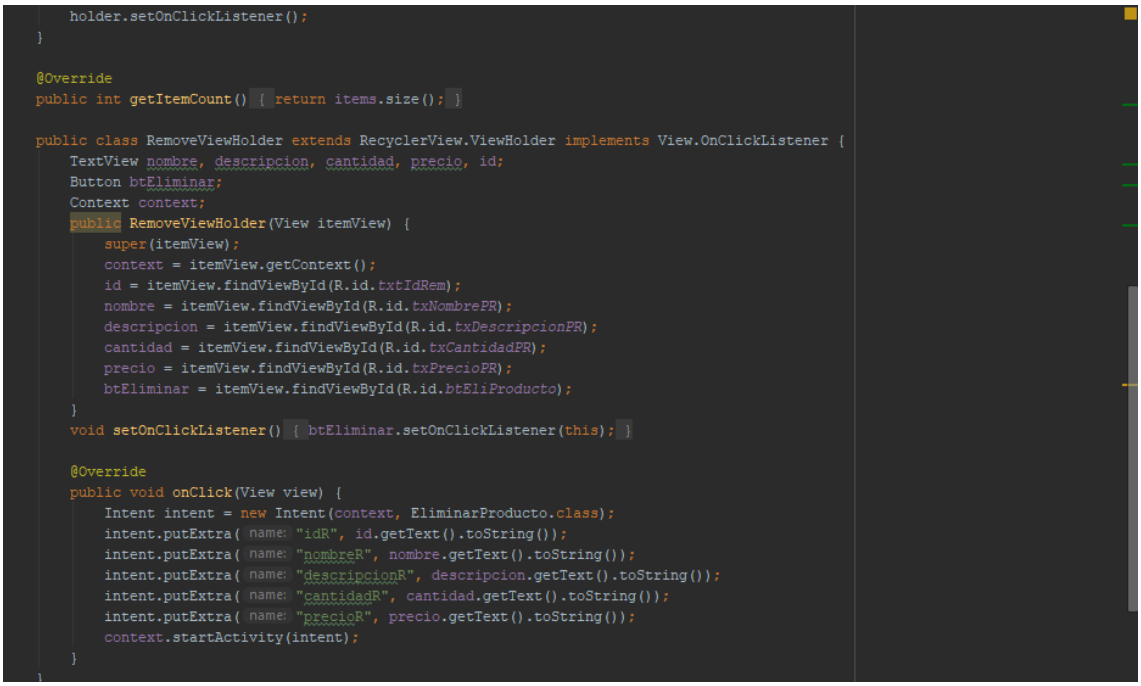
La aplicación al crear un carrito de compras lo que hace es cargar los datos en una tabla llamada “Carrito” de la base de datos con la información de los productos agregados, este código lo que hace es llamar al Web ServicesLimpiarCarrito.PHP para eliminar los datos almacenados en esa tabla (Ver Figura 32).



```
public class LimpiarCarro implements Response.Listener<JSONObject>, Response.ErrorListener{
    Context context;
    RequestQueue requestQueue;
    JsonObjectRequest jsonObjectRequest;
    String url;

    public LimpiarCarro(Context context) {
        this.context = context;
        webServices();
    }

    public void webServices() {
        requestQueue = Volley.newRequestQueue(context);
        url = "http://192.168.43.249/webServices/limpiarCarro.php";
        jsonObjectRequest = new JsonObjectRequest(Request.Method.GET, url, jsonRequest: null, listener: this, errorListener: this);
        requestQueue.add(jsonObjectRequest);
    }

    @Override
    public void onErrorResponse(VolleyError error) {

    }

    @Override
    public void onResponse(JSONObject response) {
        context.startActivity(new Intent(context, MainActivity.class));
    }
}
```

Figura 32: Clase LimpiarCarrito.

La clase RemoveAdapter se encarga de eliminar los productos seleccionados por el vendedor en la aplicación y en la base de datos (Ver Figura 33).



```
    holder.setOnClickListener();
}

@Override
public int getItemCount() { return items.size(); }

public class RemoveViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener {
    TextView nombre, descripcion, cantidad, precio, id;
    Button btEliminar;
    Context context;
    public RemoveViewHolder(View itemView) {
        super(itemView);
        context = itemView.getContext();
        id = itemView.findViewById(R.id.txtIdRem);
        nombre = itemView.findViewById(R.id.txNombrePR);
        descripcion = itemView.findViewById(R.id.txDescripcionPR);
        cantidad = itemView.findViewById(R.id.txCantidadPR);
        precio = itemView.findViewById(R.id.txPrecioPR);
        btEliminar = itemView.findViewById(R.id.btEliProducto);
    }
    void setOnClickListener() { btEliminar.setOnClickListener(this); }

    @Override
    public void onClick(View view) {
        Intent intent = new Intent(context, EliminarProducto.class);
        intent.putExtra( name: "idR", id.getText().toString());
        intent.putExtra( name: "nombreR", nombre.getText().toString());
        intent.putExtra( name: "descripcionR", descripcion.getText().toString());
        intent.putExtra( name: "cantidadR", cantidad.getText().toString());
        intent.putExtra( name: "precioR", precio.getText().toString());
        context.startActivity(intent);
    }
}
```

Figura 33: Clase RemoveAdapter.

Este código elimina los productos del carrito de compras que el cliente ya no quiere, para esto llama al Web ServiceseliminarProductoCarrito.php enviando la información necesaria para realizar el proceso (Ver Figura 34).

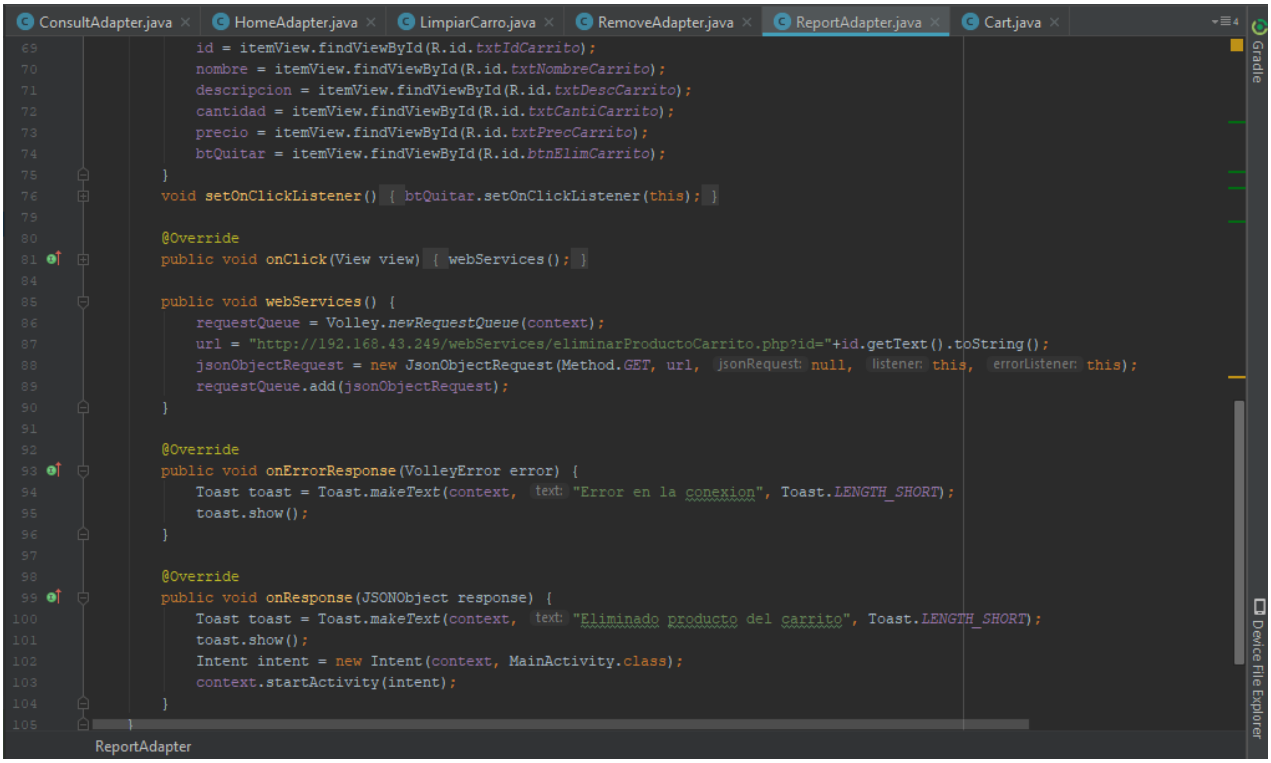


Figura 34: Clase ReportAdapter.

ViewModel

En el ViewModel se encuentran 4 clases encargadas de conectar la capa de las vistas con la capa de la lógica de negocios (ver Figura 35).

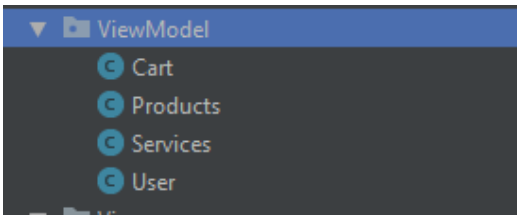
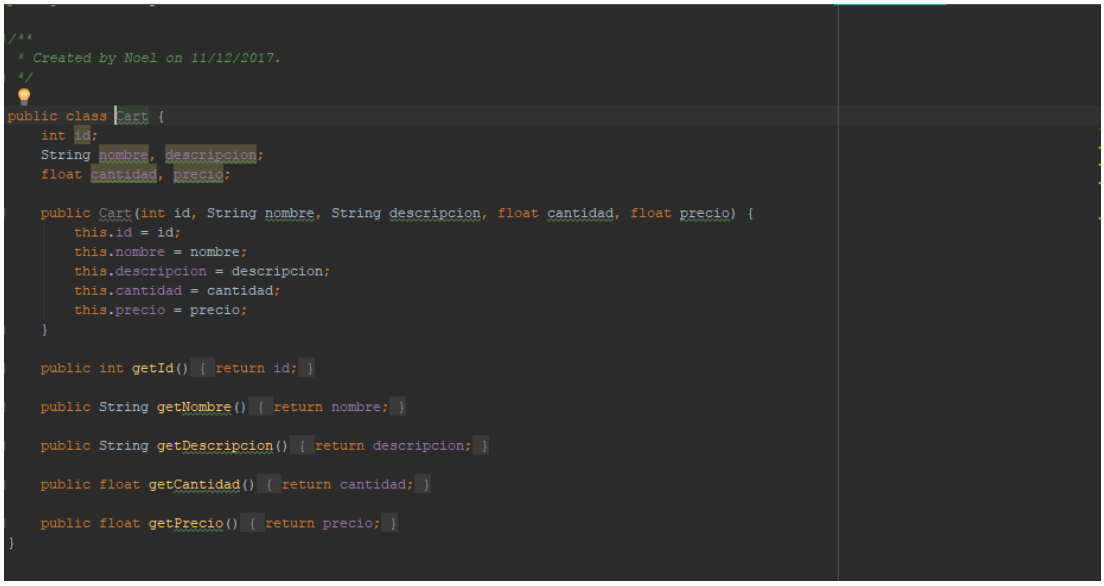


Figura 35: Clases de la capa ViewModel.

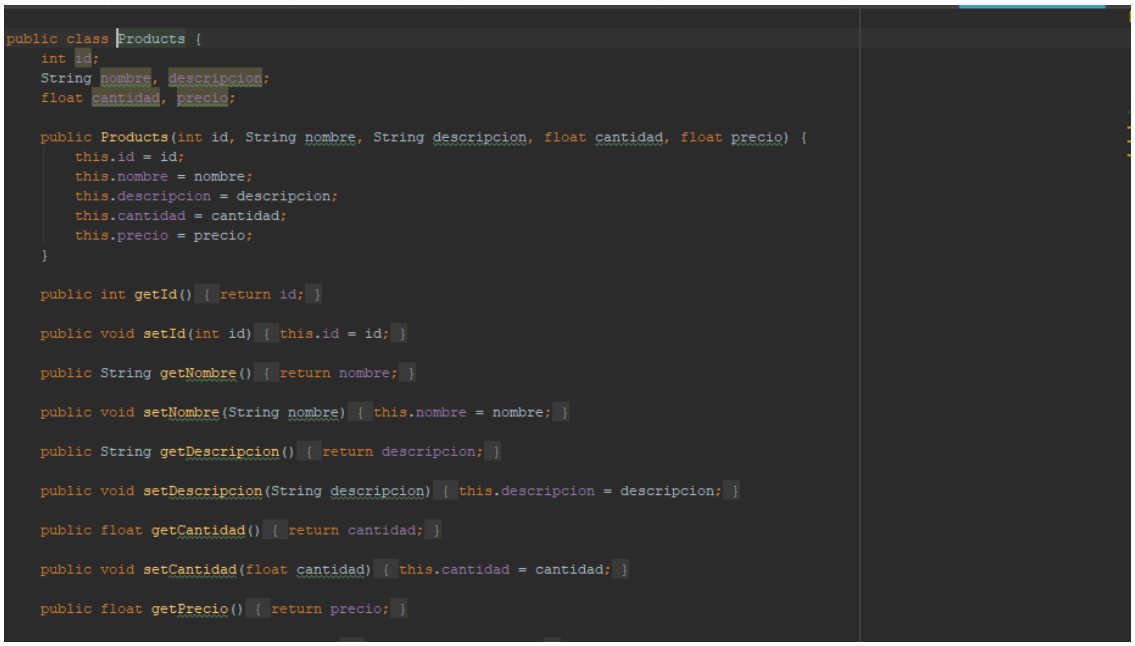
La clase Cart se encarga de la comunicación de la vista con el modelo al momento que se agregan productos al carrito de compras (ver Figura 36).



```
/**
 * Created by Noel on 11/12/2017.
 */
public class Cart {
    int id;
    String nombre, descripcion;
    float cantidad, precio;

    public Cart(int id, String nombre, String descripcion, float cantidad, float precio) {
        this.id = id;
        this.nombre = nombre;
        this.descripcion = descripcion;
        this.cantidad = cantidad;
        this.precio = precio;
    }

    public int getId() { return id; }

    public String getNombre() { return nombre; }

    public String getDescripcion() { return descripcion; }

    public float getCantidad() { return cantidad; }

    public float getPrecio() { return precio; }
}
```

Figura 36: Clase Cart.

Comunica toda la información referente a los productos de la vista con el modelo para poder realizar las acciones solicitadas por el usuario (ver Figura 37).



```
public class Products {
    int id;
    String nombre, descripcion;
    float cantidad, precio;

    public Products(int id, String nombre, String descripcion, float cantidad, float precio) {
        this.id = id;
        this.nombre = nombre;
        this.descripcion = descripcion;
        this.cantidad = cantidad;
        this.precio = precio;
    }

    public int getId() { return id; }

    public void setId(int id) { this.id = id; }

    public String getNombre() { return nombre; }

    public void setNombre(String nombre) { this.nombre = nombre; }

    public String getDescripcion() { return descripcion; }

    public void setDescripcion(String descripcion) { this.descripcion = descripcion; }

    public float getCantidad() { return cantidad; }

    public void setCantidad(float cantidad) { this.cantidad = cantidad; }

    public float getPrecio() { return precio; }
```

Figura 37: Clase Product.

A continuación, se muestra el código que se encarga de comunicar toda la información del usuario entre la vista y el modelo (ver Figura 38).

```
/**
 * Created by Noel on 10/12/2017.
 */
public class User {
    private String nombre;
    private String apellidos;
    private String correo;
    private String contraseña;


    public String getNombre() { return nombre; }

    public void setNombre(String nombre) { this.nombre = nombre; }

    public String getApellidos() { return apellidos; }

    public void setApellidos(String apellidos) { this.apellidos = apellidos; }

    public String getCorreo() { return correo; }

    public void setCorreo(String correo) { this.correo = correo; }

    public String getContraseña() { return contraseña; }

    public void setContraseña(String contraseña) { this.contraseña = contraseña; }
}
```

Figura 38: Clase User.

View

En esta capa se encuentran las clases que se encargan de actualizar la información de las vistas, toda la información recibida del ViewModel la muestra en pantalla, actualizando todos los elementos de la GUI necesarios (ver Figura 39).

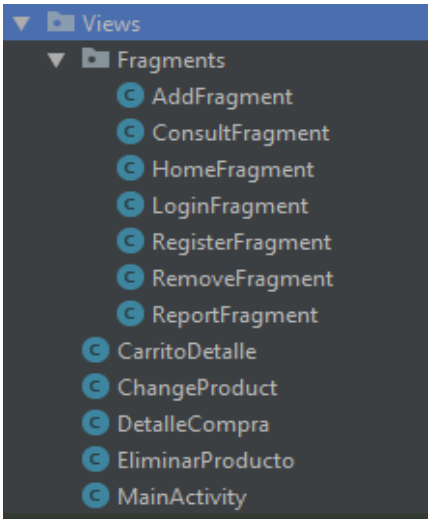


Figura 39: Clases de la capa View.

4.1 Evaluación del producto.

Con la aplicación desarrollada durante este trabajo se pretendió lograr alcanzar los objetivos planteados en el punto 1.3, se trató de lograr los requisitos necesarios para solventar la problemática descrita en el punto 1.2 y cumpliendo con los estándares de calidad de la ingeniería de software.

El sistema desarrollado no cumple todos los objetivos que se plantearon en este trabajo debido a la falta de tiempo, de personal y presupuesto para el desarrollo de la aplicación, sin embargo, se diseñó con la particularidad de ser una aplicación escalable y mantenible de forma sencilla, lo que permitirá en un futuro realizar mejoras, modificaciones, correcciones o nuevas implementaciones que sea necesario incluir, con el fin de lograr un sistema robusto y completo.

Cada uno de los módulos que conforman el producto final se desarrollaron de acuerdo a las necesidades y requerimientos especificados por los dueños del negocio al que va destinada la aplicación, sin limitar la facilidad de operación del usuario final.

Entre las funciones implementadas, se tiene una correcta gestión de inventario, dicha función representa un menor tiempo en la realización de esta tarea por parte de los negocios, así como una correcta implementación del módulo de ventas, que permite modificar automáticamente la cantidad disponible de productos conforme se generen ventas.

Por último, el correcto rendimiento en la gestión de la información almacenada dentro del sistema se debe a la estructura de la base de datos, la cual mediante el modelo entidad relación permitió un correcto flujo de información, evitando así la redundancia de datos y la inconsistencia de información, logrando proteger la integridad del sistema.

4.2 Trabajos futuros

Para seguir complementando el trabajo aquí realizado se proponen realizar las siguientes actividades como trabajos futuros para implementar nuevas funciones al sistema, esto para obtener un mejor producto cubriendo mayores necesidades de los usuarios a los que va dirigida la aplicación.

- Implementación de geolocalización de productos.
- Mejorar el manejo y almacenamiento de información en Firebase.
- Implementar métodos de pagos dentro de la aplicación.
- Agregar funciones de seguridad al momento de realizar compras en la aplicación.
- Expandir el alcance de la aplicación a otras regiones cercanas.
- Fomentar nuevos métodos de envió de productos en la región que se puedan contratar por los vendedores generando nuevos empleos.
- Implementación de IA para recomendar productos más precisos a los clientes.
- Mejorar la interfaz gráfica para que se más agradable y se ajuste a las tendencias actuales de diseño.
- Agregar más opciones de seguridad en las cuentas de los usuarios.
- Agregar roles en los negocios para que los comerciantes puedan agregar empleados y estos puedan atender funciones más limitadas pero necesarias para liberar carga de trabajo.
- Agregar lista de contacto de proveedores.

4.3 Conclusiones.

De acuerdo a los resultados obtenidos se llegó a la conclusión de que las tecnologías móviles son una herramienta importante para los negocios ya que permite tener un mayor acercamiento con los cliente, en este caso la aplicación desarrollada en este proyecto sólo está adaptada para 2 tipos de negocios que son la venta de calzado y de ropa que se encuentran dentro de la categoría de "Moda" cubriendo así las necesidades principales para estos tipos de negocios, el desarrollar una aplicación que se adapte a cualquier negocio es complejo debido a que existen necesidades específicas para cada uno de ellos.

Sin embargo, el desarrollo de aplicaciones móviles en Android es algo que va cambiando ofreciendo más opciones y facilidades para poder crear aplicaciones más potentes y mejor diseñadas, con un mejor manejo de código, con una buena base de paradigma de programación y un patrón de arquitectura robusto que permita extender las funcionalidades de la aplicación sin tanto esfuerzo como el patrón MVVM es la clave para un correcto desarrollo de aplicaciones para que puedan ser mantenibles durante el tiempo de soporte que reciba la aplicación.

Por lo tanto, para poder crear una aplicación adaptable para la mayoría de los negocios es necesario un mayor esfuerzo de ingeniería y de recursos económicos para poder implementar tecnologías más recientes en el desarrollo de la aplicación que le brinden un mayor rendimiento en su funcionamiento.

Bibliografía

1. Clarke, I. (2001). *"Emerging value proposition for m-commerce", Journal of Business Strategies, Vol. 18.*
2. *Encuesta sobre Disponibilidad y Uso de las Tecnologías de la Información en los Hogares 2017*. (15 de 02 de 2018). Obtenido de INGEGI: http://www.beta.inegi.org.mx/programas/vabcoel/2018/
3. *Estudio de Comercio Electrónico en México 2018*. (1 de Enero de 2019). Obtenido de Asociación de Internet: https://www.asociaciondeinternet.mx/es/component/remository/Comercio-Electronico/Estudio-de-Comercio-Electronico-en-Mexico-2018/lang,es-es/?Itemid=
4. Fernandez, M. I., & Medina, J. K. (2018). *El Comercio Electrónico.*
5. Gallardo López, D., & Promares Puig, C. (2008). *"Programación orientada a objetos," Lenguajes y Paradigmas de Programación.*
6. Lee, C. C. (2007). *An emprical study of mobile commerce in insurance industry. Task-technology fit and individual differences, Decision Support Systems, Vol. 43.*
7. Pressman, R. S. (2005). *Ingeniería de software un enfoque práctico.* Madrid: McGraw-Hill.
8. Silberschatz, A., Korth, H. F., & Sudarshan, S. (2002). *Fundamentos de base de datos.* Madrid: McGraw-Hill.
9. Sommerville, I. F. (2005). *Ingeniería del Software.* Pearson Educación.
10. Trejos Buriticá, O. I. (2014). *Relaciones de aprendizaje significativo entre dos paradigmas de programación a partir de dos lenguajes de programación.* Tecnura.
11. *Valor agregado bruto del comercio electrónico en México, a precios corrientes*. (15 de 02 de 2018). Obtenido de INEGI: http://www.beta.inegi.org.mx/contenidos/programas/vabcoel/2018/metodologias/Info_coel.pdf