Facultad 4

Centro de Informática Industrial

Tema: Sistema de gestión móvil para terminal de punto de venta

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Jorge Luis Guerra Diez

Tutor: Ing. Frank Geiler Vega Duverger

Tutor: Ing. Yordan Carlos Pérez Attanasov

Co-Tutor: Ing. Lázaro Moreno Ramos

Curso docente: 2016-2017.

Declaración de Autoría

Declaro ser el autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

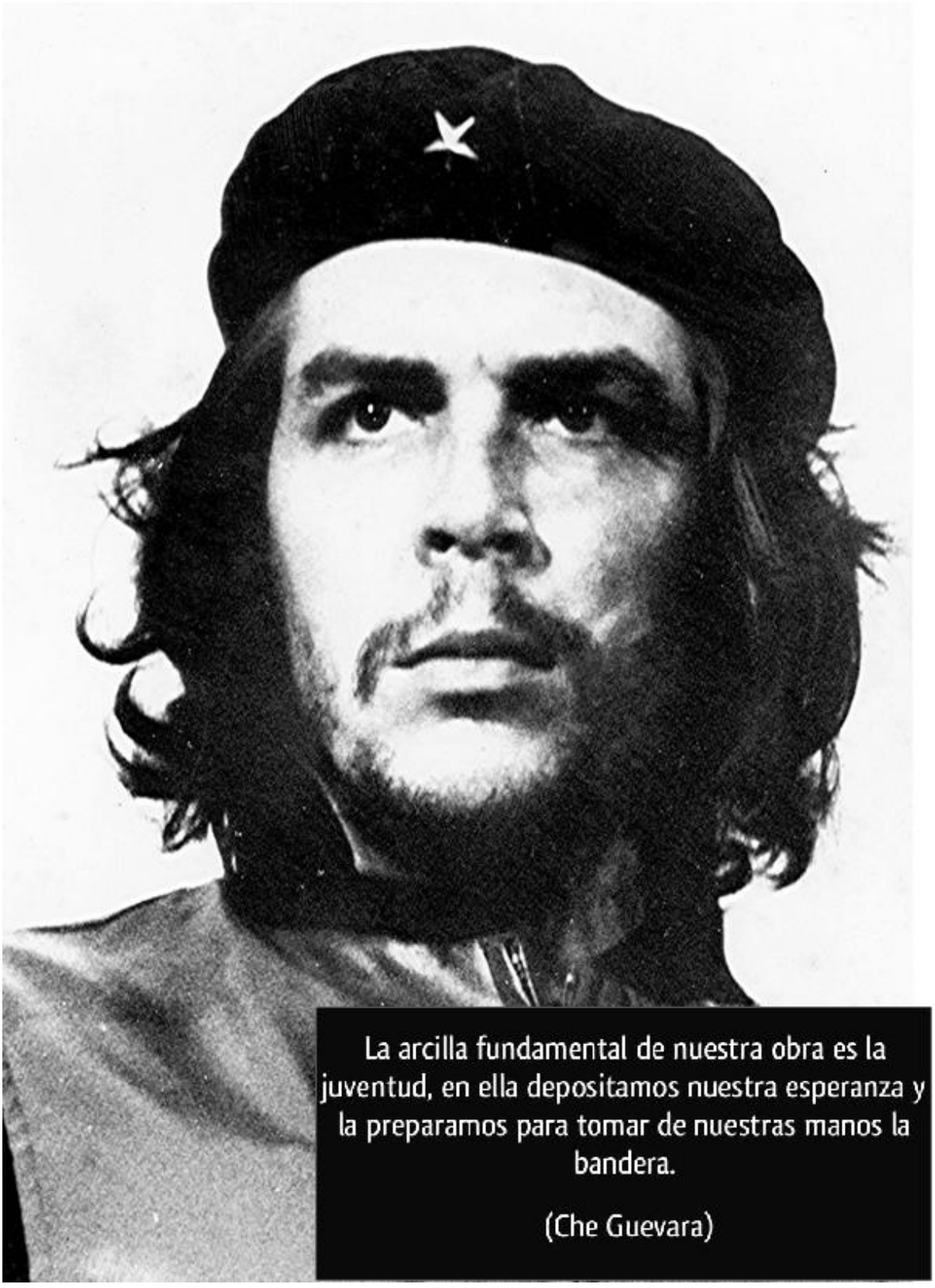
Para que así conste se firma la presente a los _____ días del mes de _____ del año _____.

Firma del Autor: Jorge Luis Guerra Diez

Firma del Tutor: Ing. Frank Geiler Vega Duverger

Firma del Tutor: Ing. Yordan Carlos Pérez Attanasov

Firma del Cotutor: Ing. Lázaro Moreno Ramos


La arcilla fundamental de nuestra obra es la juventud, en ella depositamos nuestra esperanza y la preparamos para tomar de nuestras manos la bandera.
(Che Guevara)

Dedicatoria

A mis padres y a mis abuelos por creer siempre en mí.

Agradecimientos

A todas las personas que de una forma u otra me ayudaron en el desarrollo de la tesis; en especial a mis tutores; a Ever, Claudia y David.

A las profesoras Olga, Yadira, Andrea y Zobeida que hicieran cosas que el momento no entendíamos o no nos gustaban, todo lo hacían para nuestro beneficio, muchas gracias por pensar en nosotros como más que meros estudiantes.

A los que ya no están, pero siempre recordamos con cariño; Abelito, a Fernando, al chino, a Hanssel, a Lima y Luisito.

A mi gente de la facultad 2 por darme resguardo sin apenas conocerme, con especial cariño a Frank, Pedro, Esnery y Dariel.

A mis amigos Jose, Oscar, Miguel, Hansel, Yosvani, Richard, Keigel, Lamillo, Almirola y Jander por hacer de esta universidad durante estos 5 años mi segunda casa; y ustedes mi segunda familia.

A mi novia por apoyarme siempre y darme esa seguridad de que todo es posible, gracias por ser el pilar que soporta mi vida.

A mis abuelos que sé que me miran desde el cielo, espero que estén orgullosos de mí.

A mis abuelas que gracias a dios pudieron estar ambas aquí en este día tan importante.

A Tito por quererme, apoyarme y convertirme en una más de tu familia.

A mis padres por ayudarme y creer en mi a pesar de todo, los quiero mucho.

A todos ustedes, gracias.

Resumen

La presente investigación se basó en el desarrollo de un Sistema POS (*Point of Sale* por sus siglas en inglés, Terminal de Punto de Venta en castellano), para las empresas estatales, negocios particulares y cooperativas, ya que la mayoría de estas empresas no cuentan con puntos de ventas automatizados (POS).

Para lograr un mejor desempeño y acelerar los procesos desarrollados en las empresas estatales, negocios particulares y cooperativas que no cuentan con sistemas POS, se desarrolló una aplicación que gestiona todo el proceso de venta adaptándose a cualquier negocio. Garantizando una interfaz fácil de usar por el usuario, la impresión de un recibo como constancia de pago, el pago por tarjeta y en efectivo, la detección de productos por escáner, la posibilidad de gestión de un inventario sencillo y un sistema de reportes con estadísticas y gráficas.

Para la elaboración del sistema y darles cumplimiento a los objetivos planteados se utilizó la metodología de desarrollo de *software* AUP-UCI, proporcionando esta las bases para el desarrollo del sistema. Se utilizaron diversas tecnologías como el lenguaje de programación JavaScript y HTML5, los frameworks Angularjs, Bootstrap 3 para las interfaces, Json para el envío de mensajes y el editor de texto Sublime Text 3.

Palabras Claves: *Point of sale* (POS), impresión de recibos, terminal de punto de venta, sistema móvil.

Contenido

Índice de Ilustraciones

Índice de Tablas

Introducción

Con el avance de la ciencia y la tecnología se ha incrementado el desarrollo de la sociedad, esto ha traído como consecuencia un valioso aumento del uso de las tecnologías de la información y las comunicaciones (TIC), esparciéndose las mismas en todas las esferas de la sociedad. Dada su incremental utilización y su avance desmesurado las TIC se han convertido en uno de los pilares del desarrollo de la humanidad, las cuales han permitido el perfeccionamiento de las actividades realizadas en instituciones, empresas y organismos.

En disímiles tipos de negocios se han tenido dificultades para administrar y automatizar inventarios, transacciones, ofertas (descuentos, promociones especiales) y tipos de pago. Se han desarrollado aplicaciones para este tipo de negocio que en la actualidad han logrado que los problemas antes mencionados se mitiguen, además se han mejorado las interacciones de los recursos humanos del negocio y sus clientes, evitando así pérdida de tiempo en la gestión de los servicios ofrecidos.

Cuba no está exenta de estos problemas, tanto en empresas estatales, cooperativas y particulares; la falta de un sistema que gestione y mejore sus servicios es evidente, aunque se han hecho avances en algunas ramas de las empresas pertenecientes al estado y al sector particular. La mayoría de los sistemas existentes en el país que cumplen con este objetivo están en las tiendas recaudadoras de divisas, mientras que la mayoría de las empresas particulares no cuentan con estos sistemas.

En la Universidad de las Ciencias Informáticas (UCI) se encuentra el Centro de Informática Industrial (CEDIN), perteneciente a la Facultad 4 el cual tiene como objetivo desarrollar productos que satisfagan las necesidades de informatización del país, estos productos son más enfatizados a las áreas industriales y mercantiles. En entrevistas realizadas a los diferentes recursos humanos de los negocios existentes en la Universidad y fuera de la misma, se detectó que se les hace necesario gestionar todo el proceso de la venta de los artículos o servicios que ofertan y hacerlo de manera manual es un proceso poco factible, que puede generar retrasos, extravío de recursos y en consecuencia pérdida de capital.

Partiendo de lo analizado anteriormente surge la siguiente **situación problemática**:

- La mayoría de los sistemas POS existentes en el país se encuentran en las tiendas recaudadoras de divisas, desplegados en cajas registradoras de tipo Quorion QMP Q-2044, que viene instalada con el *software* QMP POS. Este terminal de punto de venta tiene un alto costo y no permite la actualización del *software* POS incluido por defecto. En Cuba, la mayoría de las empresas estales y particulares no cuentan con un software POS que permita automatizar su proceso de venta.

Como resultado de lo anteriormente planteado, se define como **problema de investigación** la siguiente interrogante: ¿Cómo gestionar los recursos y servicios de los puntos de ventas en empresas estatales, empresas particulares y cooperativas, para facilitar el servicio de compra venta?

Podemos definir como **objeto de estudio** los sistemas de gestión de punto de venta, teniendo como **campo de acción**, los sistemas de gestión para terminales de punto de venta en dispositivos móviles.

Para dar respuesta al problema planteado se formula el siguiente **objetivo general**:

- Desarrollar un sistema multiplataforma para dispositivos móviles que permita gestionar el proceso de venta en terminales de punto de venta.

Para dar cumplimiento al objetivo general se trazaron varias **tareas de investigación**:

- Estudio del estado del arte del tema en cuestión valorando las funcionalidades de sistemas de puntos de ventas existentes.
- Entrevista a los recursos humanos de diferentes establecimientos para conocer el funcionamiento de su proceso de negocios.
- Identificación de los requisitos funcionales para el desarrollo de la solución propuesta a través de las entrevistas realizadas.
- Generación de los artefactos relacionados con el análisis y diseño de la solución propuesta.
- Propuesta de la arquitectura del sistema de gestión cumpliendo con las políticas de *software* libre.

- Propuesta de un mecanismo de perfiles de usuario para la obtención de la información generada en un punto de venta de acuerdo a los permisos del cliente en el sistema.
- Diseño e implementación de la solución propuesta.

Para dar cumplimiento a las tareas de investigación descritas anteriormente, se utilizarán los siguientes métodos de investigación:

Métodos Teóricos:

- Modelación: Empleado para representar los requisitos funcionales y los no funcionales del sistema mediante diagramas y gráficas.
- Histórico-Lógico: Empleado para la investigación con mayor profundidad las metodologías y lenguajes que permitan el funcionamiento y avance del sistema a realizar.
- Analítico-Sintético: Empleado para el análisis y conceptualización para sintetizar los aspectos teóricos necesarios para el desarrollo de la implementación del sistema.

Métodos Empíricos:

- Entrevista: Empleado para conocer las dificultades y problemas del funcionamiento del negocio; y una mayor comprensión del mismo.

Resultado esperado: Obtener un sistema multiplataforma para dispositivos móviles que permita gestionar el proceso de venta en terminales de punto de venta.

Por lo cual podemos definir para el presente trabajo de diploma la siguiente estructura por capítulos:

Capítulo 1: Fundamentación Teórica. En este capítulo se exponen los principales conceptos y objetivos referentes a la investigación, se presentan las bases teóricas relacionadas con estos conceptos y se define las tecnologías, metodologías y herramientas utilizadas para el desarrollo del *software*.

Capítulo 2: Propuesta de Solución.

Se enuncian las características del sistema definiendo sus actividades, roles y artefactos para así modelar el mismo utilizando la metodología de desarrollo de *software*.

Capítulo 3. Implementación y pruebas de la solución propuesta

En este capítulo queda registrado el diseño de la solución propuesta, junto con las tareas de ingeniería de *software* que complementan el sistema hasta finalmente evaluar los resultados obtenidos a partir de las pruebas realizadas al mismo.

Capítulo 1. Fundamentación teórica:

Introducción

En este capítulo se muestran los conceptos relacionados con la temática de investigación dando a conocer la base teórica en la cual se basa este tema investigativo; también se explica la elección de las diferentes herramientas, la metodología y tecnologías utilizadas para el desarrollo de la solución propuesta.

1.1 Definiciones y conceptos asociados

1.1.1 Venta

El Diccionario de la Real Academia Española, define a la venta como "la acción y efecto de vender. Cantidad de cosas que se venden. Contrato en virtud del cual se transfiere a dominio ajeno una cosa propia por el precio pactado" (1).

1.1.2 Punto

La palabra punto está asociada a una amplia gama de temas por lo que su concepto resulta versátil a la hora de usarlo. El punto, no es más que la más simple de las representaciones de un objeto en un determinado espacio. El punto hace referencia siempre a la especificación de la localización de un cuerpo (2).

1.1.3 Punto de Venta

Punto de Venta o POS (*Point of Sale* por sus siglas en inglés) es el punto de contacto del consumidor con las marcas o productos para su compra.

El punto de venta es el lugar que gestiona el proceso de salida y cobro de la mercancía en los distintos tipos de negocio, en las que se organiza el proceso de venta de artículos en el mostrador de manera fácil, ágil y cómoda para vendedores y compradores; contando con una caja registradora, un cobrador o cajero y emitiéndose un comprobante de venta como evidencia de la actividad realizada (3).

1.1.4 Sistema

En principio, sistema es un conjunto de entidades mutuamente relacionadas que pretenden un objetivo común (4).

1.1.5 Entrevista

La entrevista consiste en un “intento sistemático de recoger información de otra persona” a través de una comunicación interpersonal que se lleva a cabo por medio de una conversación estructurada donde es muy importante la forma en que se plantea la conversación y la relación que se establece en la entrevista (5).

Entrevista estructurada

En la entrevista estructurada todas las preguntas son respondidas por la misma serie de preguntas preestablecidas con un límite de categorías por respuestas. Así, en este tipo de entrevista las preguntas se elaboran con anticipación y se plantean a las personas participantes con cierta rigidez o sistematización (5).

1.2 Sistemas de Punto de Venta (POS)

Un sistema de punto de venta o terminal de punto de venta puede definirse como un sistema que está compuesto por dos partes fundamentales, la primera es el *hardware* (dispositivos físicos), como un escáner de código de barra, una impresora de recibos, lectores de banda magnética y monitores. La parte de *software* es un programa de gestión de venta de productos o servicios creado específicamente para agilizar los procesos de compra-venta (3).

1.2.1 Características

El *software*[1] POS es uno de los programas o de los sistemas que más popularidad están alcanzando debido a su utilidad en el control; este tipo de programa está hecho para servir como control de ventas e inventario de gran número de tiendas u otros sectores de negocio. Estas son algunas de las características más importantes que se pueden mencionar:

[1] Conjunto de programas e instrucciones que permiten ejecutar tareas en una computadora.

- Es un *software* de tipo contable: si se buscan soluciones informáticas que incluyan la temática de la contabilidad, este programa o sistema puede ser la solución necesaria para un negocio pequeño o en expansión (6).
- Fácil de utilizar: otra de las características que hacen a este sistema o *software* ideal para ser utilizado en supermercados, es muy sencillo de utilizar, lo que permite que se adapte a las necesidades de cada usuario (6).
- Facturación: es también un *software* de facturación (resultado de entregar, registrar y gestionar cualquier proceso que incluya productos o mercancía y extender una constancia del mismo dígase factura o recibo) (6).

1.2.2 Tipos de Terminales de Punto de Venta

Cuando se habla de un tipo de punto de venta se hace referencia a las clases de terminales según el medio mediante el que se produce la operación, es decir, físicamente en el negocio, mediante una página web o en cualquier parte utilizando un teléfono móvil como terminal. Se explicará a continuación los diferentes tipos de Terminales de Punto de Venta o por sus siglas en inglés (*Terminal Point of Sale*):

- **TPV tradicional:** Se compone de un *software* y un *hardware* para que los clientes hagan su compra por medio de un canal físico en la misma tienda. El dependiente pasa el código de barras por el lector, sale el producto, imprime un recibo y la compra queda registrada (7).
- **TPV virtual:** es el que se utiliza con frecuencia en las entidades financieras para hacer operaciones comerciales utilizando internet. Lo ideal sería que la base de datos del negocio en Internet esté sincronizada con el *software* TPV, para que cuando la base de datos quede sin existencias de algún producto, automáticamente la página web se lo indique al usuario (7).
- **TPV móvil:** El ejemplo más acertado de un TPV móvil es cuando un vendedor autónomo se desplaza al domicilio de los clientes y debe cobrar allí mismo, hace falta un datáfono[2], o un lector de tarjetas en el móvil. Esto permite disponer del *hardware* móvil necesario para hacer operaciones de compra-venta con pago con tarjeta, o simplemente registrar la compra en el *software* TPV (7).

[2] Dispositivo que permite el cobro por tarjeta bancaria.

1.2.3 Ventajas

Procesamiento de pago: Un sistema TPV lleva registros de los métodos con los que los clientes pagan por sus productos. La mayoría actuará como una máquina para tarjetas de crédito, equipadas también con lectores de cheques, tarjetas de débito y tarjetas inteligentes. Todos los pagos se procesan a través del sistema TPV; por lo tanto, no se requieren máquinas separadas para los pagos. Los Terminales de Puntos de Venta también organizan los impuestos de ventas y otros sistemas de contabilidad (8).

Comunicación: Un sistema TPV incrementa la comunicación dentro de un negocio. El personal de servicio puede introducir las órdenes en el sistema y enviarlas directamente a la administración. Aumentar la comunicación en un negocio puede ayudar a reducir las posibilidades de error cuando se trata de la orden de un cliente (8).

Registro y organización de los negocios: Un sistema TPV puede configurar y registrar todas las transacciones financieras de una empresa. La nómina, el inventario, los costos de alimentos y los artículos más populares del menú pueden organizarse en el sistema de la computadora, permitiendo que los negocios funcionen de forma más eficiente (8).

Precisión: Los sistemas de punto de venta ofrecen la precisión en cuanto a operaciones para la obtención y adición de productos. Cada elemento se introduce mediante el escaneo de un código de barras o manualmente al seleccionar una opción de un menú en la pantalla. El precio para cada artículo se gestiona en el sistema de modo que el precio correcto se obtenga al efectuar el pago de determinado producto (9).

Velocidad: Los sistemas POS aceleran el proceso de registro de salida, ya que los cajeros no tienen que recordar y escribir los precios de cada artículo. El número de transacciones procesadas por hora puede ser el doble o triple en un sistema manual, con la consiguiente reducción en el número de empleados necesarios y salvar la estructura superior (9).

Control de Inventario: Los sistemas de punto de venta permiten la gestión para mantener una estrecha vigilancia sobre los niveles de inventario en la tienda, hacer pedidos a tiempo y averiguar qué artículos se están vendiendo y lo rápido que están vendiendo. Cuando se realiza una venta, el artículo vendido se reducirá del inventario de forma automática (9).

Sistema de reporte: Los sistemas POS tienen una robusta generación de informes. La gestión tiene la capacidad de generar informes sobre los niveles de inventario y la disponibilidad de cada producto en tiempo real, dando a conocer estadísticas fiables de estos productos y así tener un mayor control sobre lo que se tiene en inventario. Proporcionan informes de series de tiempo que logran averiguar qué momentos del día es el más ocupado que los empleados, para así poder restructurar el trabajo de manera que los empleados cumplan con sus funciones de forma más eficiente sin sentirse explotados (9).

1.2.4 Desventajas

***Hardware* podría causar problemas:** Los Terminales de Puntos de Venta, al igual que cualquier sistema de computadora, pueden fallar provocando que las compañías pierdan información guardada. Aunque hay programas de respaldo disponibles, cuando las TPV fallan durante un periodo de mucha actividad, las órdenes y la comunicación con el personal pueden retrasarse o perderse. Si se tiene un problema con el *hardware*[3], no es tan fácil de resolver, ya que es un sistema basado en la *web*[4] y el fabricante es el más capacitado para resolver el problema; es decir, se depende completamente del fabricante para arreglar cualquier rotura o problema, esto puede costar bastante tiempo y dinero (8).

Costo: La instalación de un Terminal de Punto de Venta puede ser bastante costosa, pagando un poco más se obtiene un servicio de garantía donde los técnicos está disponibles las 24 horas del día en caso de que se necesiten reparaciones. Aunque la mayoría de los Terminales de Puntos de Venta vienen con un programa de garantía limitada, las compañías deberían considerar que los problemas podrían continuar ocurriendo después de que la garantía inicial haya terminado (8).

1.2.5 Sistemas similares POS

La mayoría de los sistemas POS, ya sea en su página oficial o en una ayuda en el propio *software* brinda información y soporte a los usuarios para una mejor utilización del mismo, además informan al usuario de cualquier actualización creada para este sistema. Son sistemas desarrollados no solo para computadoras de escritorio, sino que el mismo

[3] Conjunto de componentes que conforma la parte física de una computadora
[4] Documento electrónico que contiene información digital.

sistema es desplegado tanto en la *web* como en una aplicación para dispositivos móviles. A continuación, se muestran diferentes sistemas POS:

Cartagena: Es un sistema móvil de administración desarrollado para facturación, el mismo permite que varias computadoras trabajen simultáneamente en red, almacenando la información en una única computadora o servidor, en tiempo real en una base de datos de una forma totalmente organizada, clasificada y centralizada. Posee facturas en recibos tamaño carta, sistemas de ventas sencillos por código de barras, nombre o táctil con imágenes. Tiene como funcionalidades: un inventario inteligente y moderno; una identificación de productos con imágenes y especificaciones como el precio, exportación e importación de inventario a Excel, posee un sistema de descuento y precios fijos y reportes avanzados acerca de los recursos y el estado del inventario.

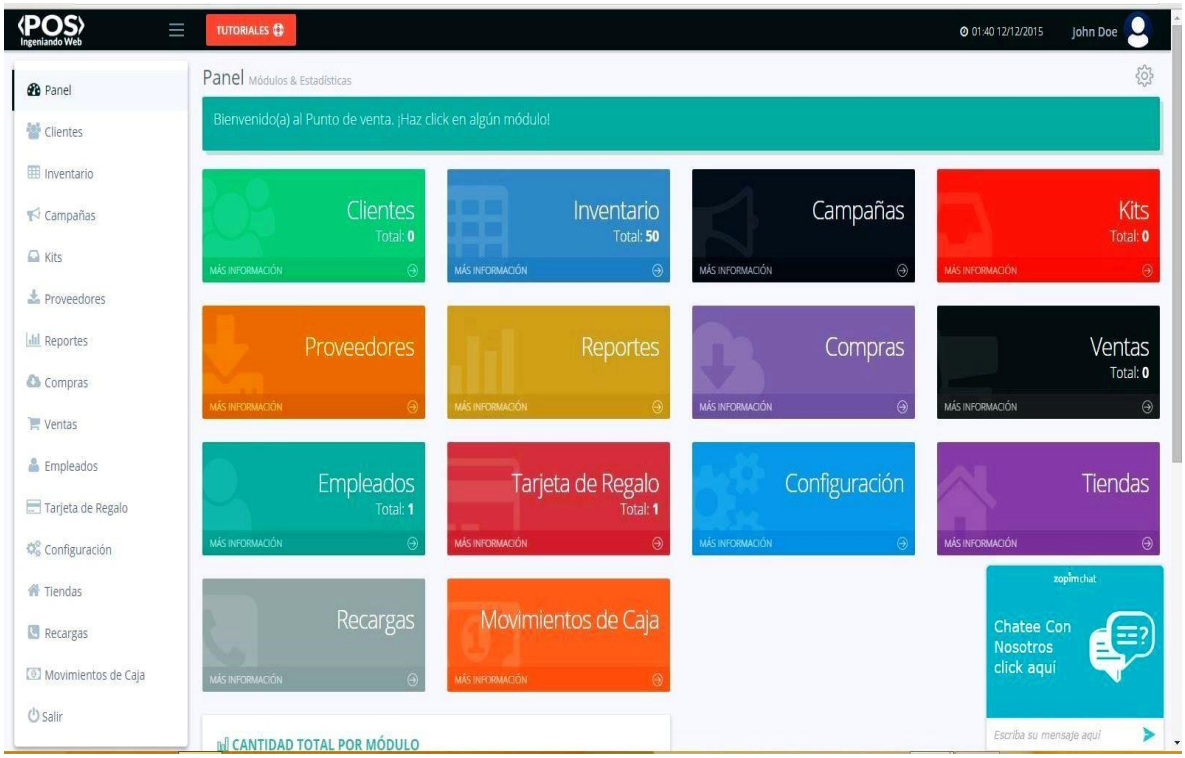


Ilustración 1 Interfaz principal del POS Cartagena

TabletSP

TableSP permite disponer de un sistema de punto de venta móvil innovador y de fácil uso diseñado para Android. Tiene multitud de funciones, como reportes avanzados de

inventario, encriptación de la base de datos para mayor seguridad, soporta gran variedad de periféricos y administración desde la nube. Incluye pagos por tarjeta bancaria, no requiere conexión permanente a Internet, cuenta con una asistencia técnica básica y actualizaciones. Posee un amplio una licencia de por vida y es adecuado para negocios como: tiendas de ropa, cafeterías, farmacias, heladerías, panaderías y restaurantes. Posee un sistema de descuento y posee una interfaz tan sencilla que no requiere capacitación previa.

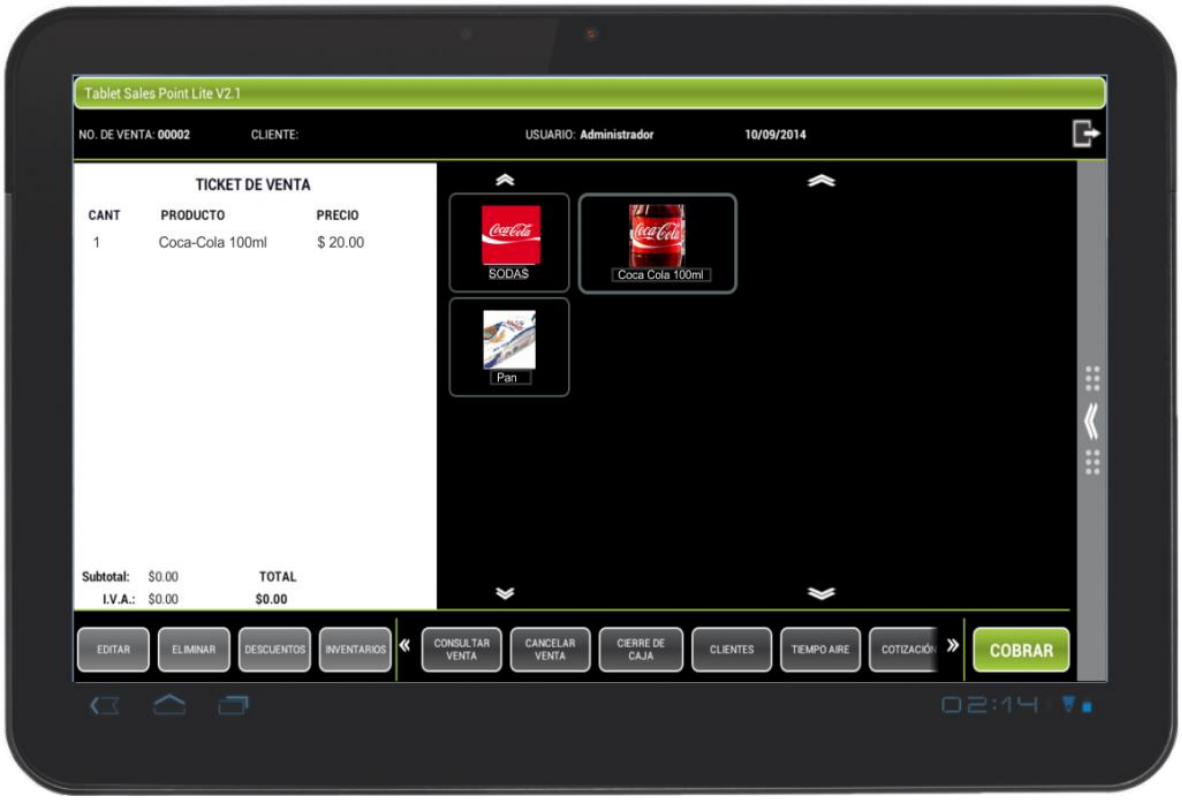


Ilustración 2 Interfaz principal de TableSP

SICAR Punto de Venta: Posee un excelente control de inventarios, ayuda a crear clientes fieles al negocio mediante una funcionalidad dedicada a descuentos y promociones. Consta con facturación. Tiene servicios de recarga y pago por servicios, un buscador que asegura productos compatibles y similares en caso de no tenerlo. Cuenta con un importador de artículos desde Excel y un etiquetador de mercancía, graficas que muestran el crecimiento y comparativas de los productos: lo cual hace este POS sumamente completo y fácil de usar.

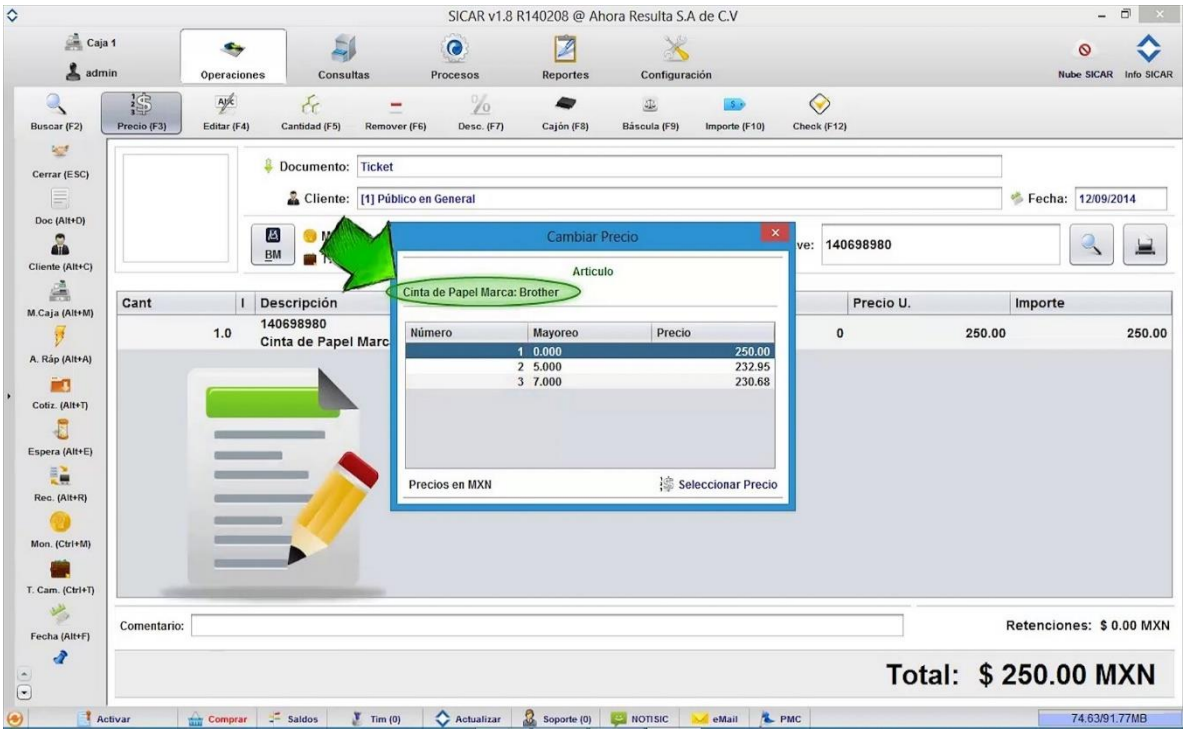


Ilustración 3 Interfaz principal de SICAR POS

Caja Registradora Quorion QMP Q-2044: Registradora muy versátil y de fácil manejo. Uso intuitivo, informes personalizables, teclado programable, cinta de control e impresora de gráficos para logotipos. La Quorion QMP Q-2044 es una registradora adecuada para comercios o restaurantes, conexión a computadoras, balanzas, lector código de barras. El QMP *software* que gestiona la caja registradora QUORION se basa en casi 20 años de experiencia en programación de cajas registradoras, por lo que incluye toda la evolución de *software* de cajas registradoras. Acceso para los dispositivos periféricos, tales como computadoras, escáner, conexión USB y LAN. Almacenamiento de gran cantidad de artículos y productos en sus bases de datos. Incluye asignación dinámica de memoria, que mejora la distribución de los recursos de memoria las áreas diferentes.

Ilustración 4 Caja Registradora Quorion QMP Q-2044

A continuación, se ha llevado a cabo una investigación acerca de las principales desventajas de estos sistemas, la cual se muestra a continuación:

Cartagena al ser un POS desarrollado en la nube no funciona sin internet, aunque puede ser instalado localmente sin internet, pero no se podría acceder desde otros computadores o dispositivos, la garantía, soporte y asesoría solamente es para aquellos usuarios que la instalen en la nube. La única forma de adquirir los equipos para el POS es mediante una tienda en línea. La licencia de por vida es extremadamente cara.

SICAR solo soporta monedas locales, tiene diferentes tipos de paquetes de instalación siendo la aplicación en la nube la única que es gratis, además la facturación electrónica conlleva a un importe extra. No posee venta de los equipos para la utilización del POS como Lector de código de barras, Pantalla o visor electrónico del TPV, Lector de banda magnética.

TableSP solo se encuentra disponible para el sistema operativo Android, dentro de la gestión de inventario no posee cortes de caja. Permite el control de productos de forma limitada y el soporte es solo para los clientes suscritos a la nube.

La caja registradora Quorion QMP Q-2044 tiene como principal desventaja que ya viene con el *software* QMP POS incluido y esta no soporta actualizaciones de *software*. Además, no tiene una funcionalidad que permita la gestión sencilla de inventarios,

tampoco ostenta de una interfaz moderna la cual tenga la identificación de productos por imágenes, gráficos que muestren mediante un reporte el estado de los productos y el negocio.

Se convierte esta investigación en pilar y soporte para el desarrollo de un sistema POS para la gestión de ventas, basándose en este estudio se propone como solución una aplicación que cumpla con todos los requerimientos propios de un Sistema de Punto de Venta, teniendo en cuenta la creación de funcionalidades importantes como la gestión de inventario, la impresión de recibos, la gestión de gráficos y reportes.

1.3 Metodologías de desarrollo de *software*

Las metodologías de desarrollo de *software* son un marco de trabajo implementado para controlar el proceso de desarrollo de *software* mediante enfoques de desarrollo, modelos de sistemas, prototipos de diseño, técnicas y procedimientos. Las metodologías han evolucionado de manera significativa en las últimas décadas permitiendo así el éxito de muchos de los sistemas desarrollados para distintas áreas. Todo desarrollo de *software* es difícil de controlar, pero si se utiliza una metodología para llevar a cabo el proyecto en cuestión se obtiene un mejor resultado y una buena aceptación por parte del cliente. Dichas metodologías tienen como objetivo guiar a los desarrolladores en la evolución e implementación del *software*, pero al ser tan variados los requisitos y las funcionalidades entre los *softwares*, esto ha tenido como consecuencia la creación de varias metodologías de desarrollo (10). Las cuales se pueden clasificar en 2 grupos:

Tabla 1 Comparación entre metodologías ágiles y tradicionales (10)

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo)	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Más artefactos
Pocos roles	Más roles
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos

AUP

El Proceso Unificado Ágil de Scott Ambler *o Agile Unified Process* (AUP) en inglés es una versión simplificada del Proceso Unificado de Rational (RUP). Este describe de una manera simple la forma de desarrollar aplicaciones de *software* de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. AUP aplica técnicas ágiles incluyendo Desarrollo Dirigido por Pruebas o *Test Driven Development* (TDD), modelado ágil, gestión de cambios ágil, y refactorización de base de datos para mejorar la productividad (11).

El proceso unificado (*Unified Process* o UP) es un marco de desarrollo de *software* iterativo e incremental. A menudo es considerado como un proceso altamente ceremonioso porque especifica muchas actividades y artefactos involucrados en el desarrollo de un proyecto de *software*. Dado que es un marco de procesos, puede ser adaptado y la más conocida es RUP (*Rational Unified Process*) de IBM (11).

AUP se preocupa especialmente de la gestión de riesgos. Propone que aquellos elementos con alto riesgo obtengan prioridad en el proceso de desarrollo y sean abordados en etapas tempranas del mismo. Para ello, se crean y mantienen listas

identificando los riesgos desde etapas iniciales del proyecto. Especialmente relevante en este sentido es el desarrollo de prototipos ejecutables durante la base de elaboración del producto, donde se demuestre la validez de la arquitectura para los requisitos clave del producto y que determinan los riesgos técnicos. El proceso AUP establece un Modelo más simple que el que aparece en RUP por lo que reúne en una única pauta las disciplinas de Modelado de Negocio, Requisitos y Análisis y Diseño. El resto de disciplinas (Implementación, Pruebas, Despliegue, Gestión de Configuración, Gestión y Entorno) coinciden con las restantes de RUP (11).

Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva y que acaban con hitos claros alcanzados:

- ***Inception* (Concepción):** El objetivo de esta fase es obtener una comprensión común cliente-equipo de desarrollo del alcance del nuevo sistema y definir una o varias arquitecturas candidatas para el mismo.
- **Elaboración:** El objetivo es que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema y en validar la arquitectura.
- **Construcción:** Durante la fase de construcción el sistema es desarrollado y probado al completo en el ambiente de desarrollo.
- **Transición:** el sistema se lleva a los entornos de preproducción donde se somete a pruebas de validación y aceptación y finalmente se despliega en los sistemas de producción.

Las disciplinas se llevan a cabo de manera sistemática, a la definición de las actividades que realizan los miembros del equipo de desarrollo a fin de desarrollar, validar, y entregar el *software* de trabajo que responda a las necesidades de sus interlocutores (11).

AUP-UCI

Al no existir una metodología de *software* universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable, se decide hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI. De las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes fases de AUP en una sola, la que lleva el nombre de Ejecución y se agrega la fase de Cierre (12).

Tabla 2 Comparación entre las fases de AUP y AUP-UCI (12)

Fases AUP	Fases Variación AUP-UCI	Objetivos de las fases (Variación AUP-UCI)
Inicio	Inicio	Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
Elaboración	Ejecución	En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.
Construcción		
Transición		
	Cierre	En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

1.3.1 Selección de las metodologías

Al haber estudiado estos dos grupos de metodologías se acometen el uso de una metodología ágil ya que estas soportan cambios durante el desarrollo del proyecto, su utilización es adecuada para pequeños grupos de trabajo y proyectos de poca duración. Se considera AUP-UCI como metodología de desarrollo de *software* a utilizar después del análisis de las principales y más utilizadas metodologías vistas anteriormente, ya que cumple con los rasgos principales que se pretende llevar a cabo; además de estandarizar el proceso de desarrollo de *software* en la UCI.

1.4 Herramientas

1.4.1 *Sublime Text 3*

Sublime Text es un editor de código multiplataforma, ligero y con pocas concesiones a los adornos. Es una herramienta concebida para programar sin distracciones. Su interfaz de color oscuro y la riqueza de coloreado de la sintaxis, centra la atención completamente. El programa dispone de auto-guardado, muchas opciones de personalización, cuenta con un buen número de herramientas para la edición del código y automatización de tareas. Soporta *macros*[5], y auto completar, como principales funcionalidades. Algunas de sus características son ampliables mediante *plugins* (13).

1.4.2 *Bootstrap* 3

Bootstrap es un conjunto de conceptos, prácticas y criterios (*framework*) desarrollado por Mark Otto y Jacoob Thornton dentro de *Twitter* con la intención de estandarizar el conjunto de herramientas que utilizaban los desarrolladores de *front-end*[6]. Bootstrap es el *framework* más popular para desarrollar proyectos móviles de primera respuesta en la Web utilizando *HTML*, *CSS* y JavaScript. Bootstrap hace que el desarrollo web *front-end* sea más rápido y más fácil. Está hecho para desarrolladores de todos los niveles de habilidad y proyectos de todos los tamaños. Posee una extensa documentación para elementos HTML comunes, docenas de componentes HTML y CSS personalizados, y complementos jQuery (14).

Se integra con las principales librerías de *Javascript* como *AngularJS* y *Json* y marcos de desarrollo como Apache Cordova.

1.4.3 *Visual Paradigm*

Visual Paradigm para *UML* (Unified Modeling Language o Lenguaje Unificado de Modelado) es una herramienta para desarrollo de aplicaciones utilizando modelado *UML* ideal para Ingenieros de *Software*, Analistas de Sistemas y Arquitectos de sistemas que están interesados en construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos. Visual Paradigm también ofrece: navegación intuitiva entre la escritura del código y su visualización; potente generador de informes en formato PDF/HTML; ambiente visualmente superior de modelado; sofisticado

[5] Programa que ejecuta una serie de instrucciones.
[6] Programación de cara al cliente, conjunto de tecnologías que corren del lado del navegador.

diagramador automáticamente de *layout*[7]; sincronización de código fuente en tiempo real (15).

UML

UML se trata de un estándar que se ha adoptado a nivel internacional por numerosos organismos y empresas para crear esquemas, diagramas y documentación relativa a los desarrollos de *software* (programas informáticos). No es más que una serie de normas y estándares gráficos respecto a cómo se deben representar los esquemas relativos al *software*. Es el lenguaje utilizado para trabajar con la herramienta Visual Paradigm.

1.4.4 *Apache Cordova*

Cordova es un marco de desarrollo móvil de código abierto. Permite utilizar las tecnologías estándar web como HTML5, CSS3 y JavaScript para desarrollo multiplataforma, evitando el lenguaje de desarrollo nativo para cada plataforma móvil. Las aplicaciones se ejecutan dentro de envolturas para cada plataforma y dependen de enlaces estándares API (Interfaz de Programación de Aplicaciones o Aplication Programming Interface en inglés) para acceder a de cada dispositivo sensores, datos y estado de la red. Cordova soporta diversas plataformas como *Android, Windows Phone*, *Blackberry* 10, *Amazon Fire OS*, *Firefox OS*, *Ubuntu, Windows 8*. Posee como funciones principales: establecer un ambiente de desarrollo basado en tecnologías *web* capaz de ser desplegado para diferentes sistemas, permitiendo extender una aplicación en varias plataformas, sin tener que volver a implementarlo con el lenguaje y herramienta de cada plataforma. Se mezclan los componentes de la aplicación nativa con una *vista Web* (navegador) que puede tener acceso a las API de nivel de dispositivo. Las aplicaciones se basan en un archivo común que proporciona información acerca de la aplicación y especifica los parámetros que afectan a cómo funciona; este archivo se adhiere a la especificación de empaquetado de la aplicación. La misma aplicación se implementa como una página web, que hace referencia a cualquier CSS, JavaScript, imágenes, archivos multimedia, u otros recursos necesarios para que esta se ejecute. La aplicación se ejecuta como una vista *web* dentro de la envoltura de la aplicación nativa, esta es capaz de interactuar con el dispositivo móvil como si esta fuera una aplicación nativa de este sistema (16).

[7] Esquema de distribución de los elementos dentro de un diseño.

1.4.5 *WebSockets*

Los navegadores modernos se les ha dotado de nuevas tecnologías de comunicación entre cliente y servidor sin la necesidad de un tercero (complemento); la tecnología se denomina *WebSockets*. Esta tecnología es una buena alternativa porque no depende de *plugins* y permite el ahorro de recursos tecnológicos (17).
La especificación *WebSocket* define una API que establece una conexión tipo socket entre un navegador web y un servidor. Es decir, se crea una conexión cliente/servidor persistente *full duplex*[8], donde cualquier parte puede comenzar una comunicación (18).

1.4.6 *RestFul*

El estilo *Representational State Transfer* (REST) es una abstracción de los elementos arquitectónicos dentro de un sistema distribuido. REST ignora los detalles de la implementación de componentes y la sintaxis del protocolo para centrarse en los roles de los componentes, las restricciones sobre su interacción con otros componentes y su interpretación de los elementos de datos significativos. Abarca las restricciones fundamentales sobre componentes, conectores y datos que definen la base de la arquitectura Web y, por tanto, la esencia de su comportamiento como una aplicación basada en red (19).

Cabe aclarar que cuando hablamos de *REST* y *RestFul* no hablamos de conceptos diferentes, sino que es lo mismo, salvo ciertos matices. *REST* es un concepto de arquitectura que sigue los principios anteriormente mencionados, en cambio *RestFul* son los servicios web que siguen esos principios.

1.4.7 Tarjeta SBC o Raspberry Pi

Raspberry Pi o Tarjeta SBC (*Single Board Computer,* Computadora de placa única) no es más que una diminuta placa base de 85 x 54 milímetros en la que se aloja un chip Broadcom BCM2837 con procesador ARMv8 hasta a 1.2 GHz de velocidad, GPU VideoCore IV y hasta 1 Gbytes de memoria RAM. Para que funcione, basta con que se añada un medio de almacenamiento (como por ejemplo una tarjeta de memoria SD), conectarlo a la corriente gracias a cualquier cargador de tipo micro USB (el mismo que sirve para recargar la mayoría de los teléfonos móviles). La fundación de Raspberry Pi

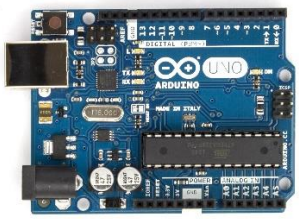
[8] Transmite datos en ambas direcciones, pero no simultáneamente.

pone a disposición desde su página web Raspbian, una distribución de Linux basada en Debian. Raspberry Pi cuenta con un puerto de salida de video HDMI y otro de tipo mini Jack compuesto que además es salida de audio, y cuatro puertos USB 2.0 a los que se puede conectar teclado, ratón y otros periféricos. Dispone de Ethernet RJ45, Wifi 802.11n y Bluetooth 4.1. Raspberry Pi es una microcomputadora completamente funcional (20).

Comparación entre Arduino y Raspberry Pi

Las principales plataformas empleadas en la actualidad para el desarrollo de sistemas embebidos se identifican las Raspberry Pi y la placa Arduino. Aunque están destinadas a resolver problemas similares, puedan lucir muy parecidas e incluso es posible que hayamos asumido que este par de plataformas de *hardware* compiten para resolver problemas similares, son muy diferentes, debido a que una Raspberry Pi es identificada como una computadora completamente funcional, mientras que Arduino es identificado como un microcontrolador.

Tabla 3 Comparación entre Arduino y Raspberry Pi (21)

	 Ilustración 5 Arduino.	 Ilustración 6 Raspberry Pi.
Precio en Dólares	$30	$37
Tamaño	7.6 x 1.9 x 6.4 cm	8.6 x 5.4 x 1.7 cm
Memoria	0.002MB	1GB
Velocidad del Reloj	16MHz	1.2 GHz
Network	Ninguna	10/100 wired Ethernet RJ45, Wi-Fi 802.11n, Bluetooth 4.1
Multitarea	No	Si

Voltaje de entrada	7 a 12 V	5 V
Memoria Flash	32 KB	Tarjeta SD (2 a 16GB)
Puertos USB	Uno	Cuatro
Sistema Operativo	Ninguno	Distribuciones de Linux y Windows
Entorno de desarrollo integrado(IDE)	Arduino	Scratch, IDLE, cualquiera con soporte Linux.

Teniendo en cuenta la comparación antes realizada se decidió utilizar Raspberry Pi 3 debido a que esta plataforma está disponible para las condiciones de desarrollo actual en el centro CEDIN de la UCI. Esta posee Bluetooth, Wifi, es de *hardware* y código abierto y además cumple con las políticas de *software* libre de la soberanía tecnológica del país.

1.4.8 HTML5

HTML, que significa Lenguaje de Marcado para Hipertextos (*HyperText Markup Language*) es el elemento de construcción más básico de una página web y se usa para crear y representar visualmente una página web. Determina el contenido de la página web, pero no su funcionalidad. HTML le da "valor añadido" a un texto estándar en español. Hiper Texto se refiere a enlaces que conectan una página Web con otra, ya sea dentro de una página web o entre diferentes sitios web (22).

HTML5 es la última versión de HTML, con nuevos elementos, atributos y comportamientos. Contiene un conjunto más amplio de tecnologías que permite a los sitios Web y a las aplicaciones ser más diversas y de gran alcance (23).

1.4.9 CCS 3

Hojas de Estilo en Cascada (*Cascading Style Sheets*) es el lenguaje utilizado para describir la presentación de documentos HTML o XML, esto incluye varios lenguajes basados en XML como son XHTML o SVG. CSS describe como debe ser renderizado el elemento estructurado en pantalla, en papel, hablado o en otros medios (24).

CSS3 es la última evolución del lenguaje de las Hojas de Estilo en Cascada, y pretende ampliar la versión CSS2.1. Trae consigo muchas novedades altamente esperadas, como las esquinas redondeadas, sombras, gradientes, transiciones o animaciones, y nuevos *layouts*, cajas flexibles o maquetas de diseño en cuadrícula (25).

1.5 Lenguaje de Programación

En esencia un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar. Por lo tanto, un lenguaje de programación es un modo práctico para que los seres humanos puedan dar instrucciones a un equipo (26).

1.5.1 *JavaScript*

JavaScript es el lenguaje interpretado orientado a objetos desarrollado por Netscape que se utiliza en millones de páginas web y aplicaciones de servidor en todo el mundo. Es un lenguaje de programación dinámico que soporta construcción de objetos basado en prototipos. La sintaxis básica es similar a Java y C++ con la intención de reducir el número de nuevos conceptos necesarios para aprender el lenguaje. Las capacidades dinámicas de *JavaScript* incluyen construcción de objetos en tiempo de ejecución, listas variables de parámetros, variables que pueden contener funciones, creación de scripts dinámicos (27).

1.6 Frameworks

1.6.1 JSON

JSON (*JavaScript Object Notation* - Notación de Objetos de *JavaScript*) es un formato ligero de intercambio de datos. Está basado en un subconjunto del Lenguaje de Programación *JavaScript*, Standard ECMA-262 3rd Edition - Diciembre 1999. JSON es un formato de texto que es completamente independiente del lenguaje, pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, *JavaScript*, *Perl*, *Python*. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos. JSON está constituido por dos estructuras: una colección de pares de nombre/valor (en varios lenguajes esto es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo) y una lista ordenada de valores (en la mayoría de

los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias). Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan (28).

1.6.2 *AngularJS*

AngularJS es un *framework* cliente desarrollado por Google, al ser un *framework* con un gran auge en el mundo *web* cuenta con un buen set de herramientas y *plugins*[9] ya desarrollados, así como también cuenta con manuales y recomendaciones de buenas prácticas, los cuales pueden ayudar a crear un proyecto sostenible. *AngularJS* permite extender el vocabulario HTML con directivas y atributos, manteniendo la semántica y sin necesidad de emplear librerías externas como *jQuery* o *Underscore.js* para que funcione. Cuenta con un set de utilidades, opciones y configuraciones medianamente amplio. *Angular* está desarrollado en base a módulos y cuenta con su propio sistema de inyección de dependencias (29).

Angular promueve y usa patrones de diseño de *software* como Modelo-Vista-Controlador. Básicamente estos patrones nos marcan la separación del código de los programas dependiendo de su responsabilidad. Posee mejoras del *HTML*, viene cargado con todas las herramientas que los creadores ofrecen para que los desarrolladores sean capaces de crear ese *HTML* enriquecido. La palabra clave que permite ese *HTML* declarativo en *AngularJS* es "directiva", que no es otra cosa que código Javascript que mejora el HTML (30).

Conclusiones parciales

- Concluido este capítulo se determina que el estudio realizado acerca de los principales conceptos asociados al dominio del problema, permitió un mayor entendimiento por parte del autor y sentó las bases para iniciar el proceso de desarrollo del sistema.
- Se llevó a cabo una investigación acerca de herramientas similares al sistema a desarrollar que permitió tomar estas herramientas como punto de partida para el desarrollo de la solución.

[9] Aplicación informática que añade una nueva funcionalidad o característica a *software* ya creado.

- La metodología AUP-UCI será la encargada de guiar el ciclo de vida de la propuesta de solución durante su desarrollo y contará para ello con las herramientas y tecnologías escogidas.

Capítulo 2: Propuesta de Solución

Introducción

En el presente capítulo se demuestra de manera práctica los procesos de análisis y el diseño de *software*. Se lleva a cabo la descripción y generación de los requisitos funcionales y no funcionales por los cuales se medirá y guiará el desarrollo de la solución propuesta. A partir de estos requisitos, se determinan las historias de usuarios asociadas, se describirá el funcionamiento del sistema mediante diagramas, se presentan los patrones de diseño de *software* que serán utilizados en la propuesta y se diseña la arquitectura a utilizar.

2.1 Modelo de Dominio

La siguiente imagen se utiliza como fuente de comprensión a la solución propuesta, mostrando el flujo de trabajo del sistema, sus relaciones entre si y funcionamiento. En la vista principal se muestra un listado de productos donde el Cliente es el encargado de la compra de los mismos y realizar la compra de productos de manera correcta, luego dicha compra genera una venta con los costos por producto, el costo total de la venta y el importe en efectivo a devolver, la cual tiene como garantía el recibo de pago. También se muestran gráficos para medir parámetros de cantidad para las Ventas y Productos. Por último, las ventas tienen dos vías como forma de pago: en efectivo y por tarjeta.

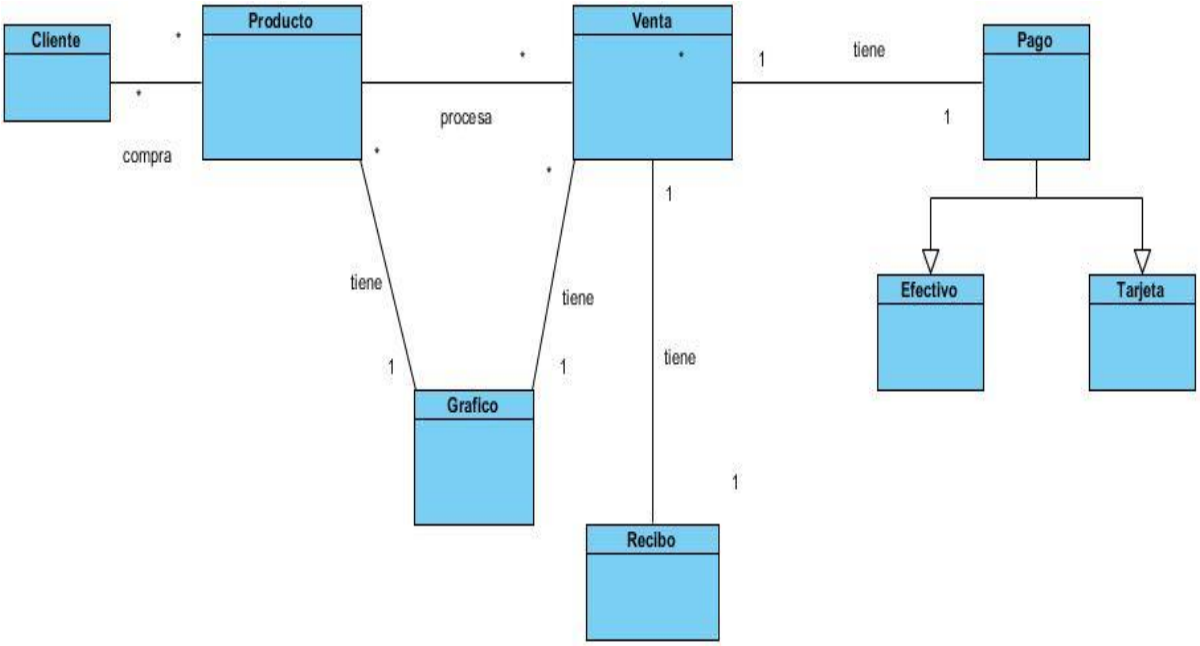


Ilustración 5 Modelo de dominio.

Para el modelo de negocio se tuvieron en cuenta la descripción de las siguientes entidades:

- **Cliente:** es el encargado de acceder a los productos y realizar las acciones de compra y pago de las ofertas que brinda el sistema.
- **Producto:** entidad que se encarga de mostrar la variedad de ofertas, precios y cantidad de activos.
- **Venta:** entidad que se encarga de la gestión de la venta de productos.
- **Pago:** entidad que se encarga de recibir el pago que realiza el cliente ya sea en efectivo o por tarjeta.
- **Recibo:** entidad que se encarga de realizar el comprobante de compra.
- **Gráfico:** entidad que muestra a través de datos cómo se comporta de manera específica las ventas y los productos.

2.2 Especificación de requisitos

Un requisito es forma o funcionalidad de un producto o servicio. Se usa ingeniería de *software* e ingeniería de requisitos. Es una “condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado”. Un requisito también

puede verse como una declaración abstracta de alto nivel de un servicio que el sistema debe proporcionar, una definición matemática detallada y formal de una función del sistema.(31)

Se utilizó como técnica de obtención de los requisitos la entrevista (ver Anexo 4) este formato de entrevista estructurada fue obtenida del repositorio institucional de la Universidad Nacional de Colombia. Esta entrevista se les realizó a diferentes puntos de venta fuera y dentro de la universidad, para que manifieste las necesidades que se pretenden solucionar con las tecnologías móviles, o para obtener las características que debe tener la aplicación.

2.2.1 Requisitos funcionales

RF1.El sistema debe permitir mostrar producto disponible para la venta.

RF2. El sistema debe permitir elaborar listado de pedidos.

RF3.El sistema debe permitir eliminar productos del listado de pedidos.

RF4. El sistema debe permitir crear e imprimir recibo.

RF5. El sistema debe permitir realizar pago en efectivo.

RF6. El sistema debe permitir notificar detección de tarjeta.

RF7. El sistema debe permitir mostrar notificación de descuento por tarjeta.

RF8. El sistema debe permitir comprar producto.

RF9. El sistema debe permitir adicionar categoría.

RF10. El sistema debe permitir eliminar categoría.

RF11. El sistema debe permitir mostrar estadísticas.

RF12. El sistema debe permitir adicionar producto.

RF13. El sistema debe permitir mostrar producto.

RF14. El sistema debe permitir modificar producto.

RF15. El sistema debe permitir eliminar producto.

RF16. El sistema debe permitir adicionar imagen del producto.

RF17. El sistema debe permitir ordenar listado de productos.

RF18. El sistema debe permitir mostrar modo sin conexión.

RF19. El sistema debe permitir cambiar usuario y contraseña.

2.2.2 Requisitos no funcionales

Usabilidad:

RNF1. El sistema debe presentar una vista sencilla, descriptiva y fácil de usar, con el objetivo de propiciarles a los usuarios un mejor entendimiento de la aplicación.

***Software*:**

RNF2. Requiere el sistema operativo Android en su versión 4.2 o superior.

***Hardware*:**

RNF3. Requiere una capacidad disponible de 30 Mb en la memoria del *smartphone* (teléfono inteligente) o tableta electrónica para la instalación y ejecución de la aplicación.

2.3 Historias de usuario

Las historias de usuarios son la propuesta de las metodologías ágiles para la especificación de los requisitos de cliente. Se escriben desde el punto de vista del usuario del sistema y usando su vocabulario. (32)

Las historias de usuario son formularios o tablas en dónde el interesado describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. Cada historia de usuario debe ser lo suficientemente comprensible y delimitada para que se pueda implementar en un corto tiempo. Se pondrá de ejemplo a continuación las 2 historias de usuario principales las demás se encuentran en el anexo 1.

Tabla 4 HU1-Mostrar producto disponible para la venta.

Historia de Usuario	
Código: HU1	**Usuario:** Administrador
Nombre historia: Mostrar producto.	
Referencia:	**Prioridad:** Alta

Puntos estimados: 0.6	**Iteración asignada:** 1
Descripción: El sistema debe conectarse a un servidor *RestFul* y permitir que se muestren los productos disponibles, su imagen, precios y categoría a la que pertenece, ordenada alfabéticamente para que el cliente pueda realizar la compra de forma eficiente y factible. En caso de no tener conexión con el servidor mostrar una notificación.	
Observaciones:	

Tabla 5 HU2-Mostrar listado de pedidos.

Historia de Usuario	
Código: HU2	**Usuario:** Cliente
Nombre historia: Elaborar listado de pedidos.	
Referencia: 1	**Prioridad:** Alta
Puntos estimados: 0.4	**Iteración asignada:** 1
Descripción: El sistema debe permitir que el cliente llene en una tabla de venta o carro de compra el listado de los productos que desea comprar, con los precios individuales y totales de estos productos.	
Observaciones:	

Las historias de usuarios son estimadas en puntos de estimación en semanas por los programadores para calcular un aproximado del tiempo de demora en implementar las mismas.

Tabla 6 Estimación por esfuerzo por HU.

Historias de usuario	Puntos de estimación (semanas)
El sistema debe permitir mostrar producto disponible para la venta.	0.6
El sistema debe permitir mostrar listado de pedidos.	0.4

El sistema debe permitir eliminar productos del listado de pedidos.	0.8
El sistema debe permitir crear e imprimir recibo.	0.4
El sistema debe permitir realizar pago en efectivo.	0.4
El sistema debe permitir notificar detección de tarjeta.	0.4
El sistema debe permitir mostrar notificación de descuento por tarjeta.	1.2
El sistema debe permitir comprar producto.	0.4
El sistema debe permitir adicionar categoría.	0.4
El sistema debe permitir eliminar categoría.	0.4
El sistema debe permitir adicionar imagen de la categoría.	0.6
El sistema debe permitir mostrar estadísticas.	0.4
El sistema debe permitir adicionar producto.	0.4
El sistema debe permitir mostrar producto.	0.4
El sistema debe permitir modificar producto.	0.4
El sistema debe permitir eliminar producto.	0.4
El sistema debe permitir adicionar imagen del producto.	0.2
El sistema debe permitir ordenar listado de productos.	0.8

Después de realizar un análisis se pudo estimar que el costo de desarrollo para el proyecto es de 9 semanas. Los valores de las HU están comprendidos entre 0.2 y 1.2 dependiendo de la complejidad correspondiente a cada una.

2.4 Propuesta de solución

Para una mejor comprensión del sistema y su funcionamiento, el siguiente esquema muestra cómo estarán conectados los componentes físicos y su comunicación entre sí. Se accederá a través de una conexión Wi-Fi a una tarjeta SBC o Raspberry Pi, desde cualquier dispositivo móvil que se encuentre en el alcance de la red wifi de dicha tarjeta y tengo instalado el sistema desarrollado; la cual poseerá un Servidor *RestFul* que contendrá los datos de los productos, categorías y servicios que se ofertan en el sistema,

este es el encargado de manejar la información resultante de las ventas y la adición de nuevos. También contará con un servidor *Websockets* que posee los *drivers*[10] para el manejo de los periféricos de entrada: Lector de tarjetas, Escáner de Barras e Impresora de recibos; los cuales estarán conectados a la Raspberry Pi vía USB.

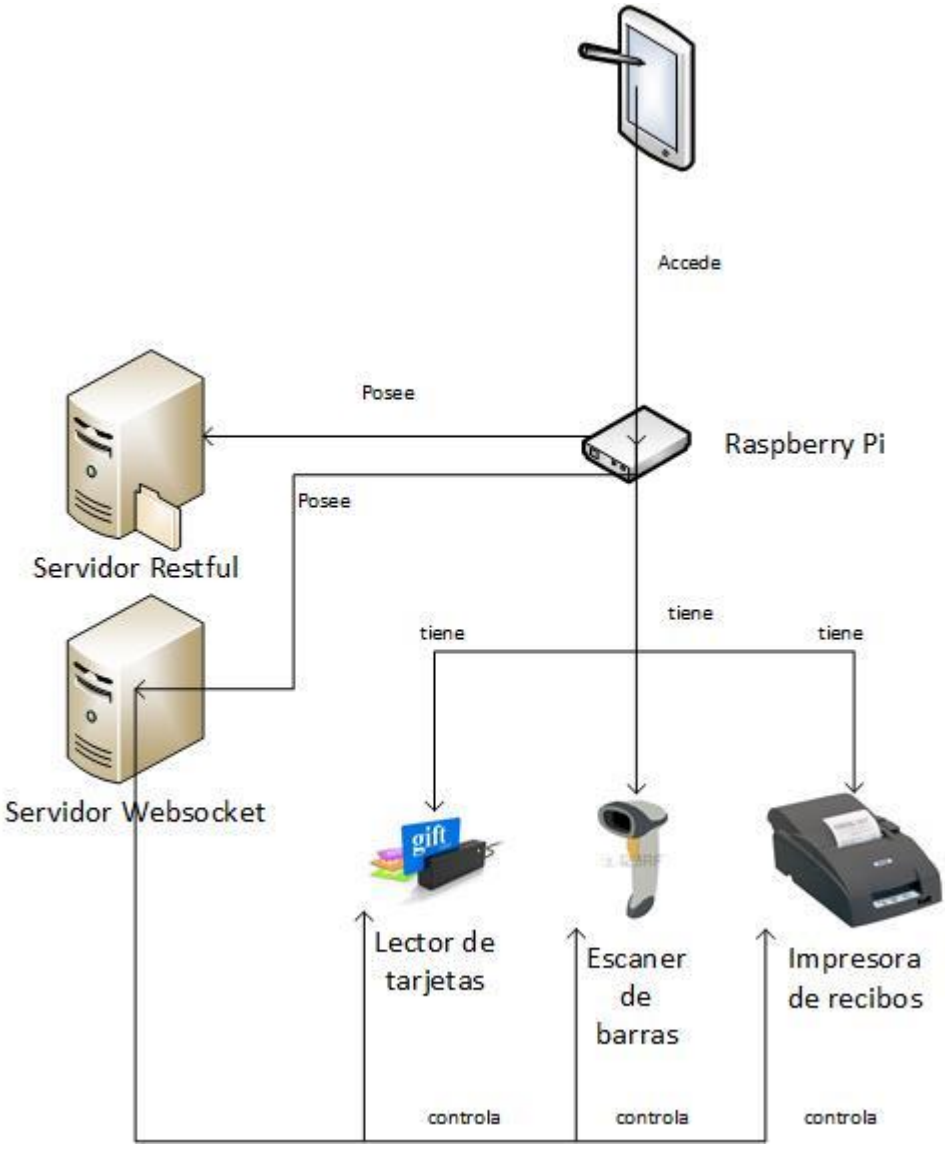


Ilustración 6 Propuesta de solución

La propuesta de solución se encontrará desplegada en el dispositivo móvil la cual poseerá identificación de productos con imágenes y un carrito de compra para una mejor interacción del usuario con el proceso de la venta. Tendrá un sistema de logueo para una

[10] Manejador o controlador de dispositivos. Permite al sistema operativo interactuar con un periférico.

mayor seguridad en la administración, impresión de recibo de compra, pago por tarjeta y en efectivo. Además, tendrá un sistema de gráficos el cual muestra el estado del sistema en tiempo real.

2.5 Arquitectura del sistema

El sistema está diseñado usando una arquitectura Modelo Vista Controlador (MVC), la cual permite separar los datos y la lógica de negocio de la interfaz de la aplicación a realizar y el módulo que gestiona los sucesos y las comunicaciones. El MVC busca mantener buenas prácticas de programación como la reutilización de código y la separación de conceptos. Esta arquitectura propone la creación de tres módulos o componentes:

- **Modelo:** En esta capa se encuentran 2 servidores de los cuales se van a consumir servicios para diferentes funcionalidades del sistema. Un servidor *RestFul* para la obtención de los datos como los productos, las categorías, la cantidad de ventas realizadas en un determinado tiempo, y todos los atributos asociados a estos. Por otro parte se encuentra un servidor que utiliza tecnología *WebSockets* para el intercambio de información bidireccional entre el cliente o administrador del negocio y los *drivers*, gestionado requisitos como el pago por tarjeta, el escáner de código y la impresión de recibos.
- **Vista:** En esta capa se encuentran las páginas *home.html* y *admin.html* que no son más que las interfaces o vistas que muestra el negocio. La página *home* es la encargada de mostrar al cliente su interfaz en la cual se puede interactuar con las funcionalidades realizadas; el cliente puede llevar a cabo la compra de cualquier producto ofertado en el negocio, mandar a imprimir su recibo de compra y llevar a cabo el pago por esta compra. La página *admin* se encarga de la administración del negocio en su totalidad; añadiendo, modificando y eliminando tanto productos como categorías; también permite la verificación del estado del negocio mediante estadísticas y la personalización de la interfaz del cliente.

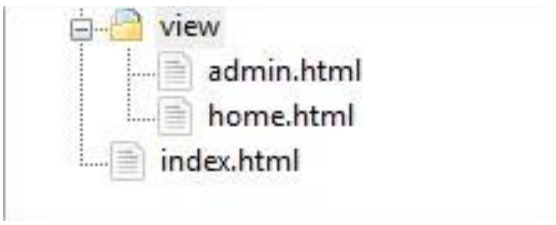


Ilustración 7 Vista

- **Controlador:** En esta se encuentran los módulos controladores *home.js* y *admin.js* los cuales contienen el código para manejar y procesar la información pasada por las vistas. El modulo *home.js* se encarga de generar el carrusel o *banner*, gestionar el pago en efectivo o por tarjeta, añadir al carro de compra los productos seleccionados por el usuario, así como eliminarlos y aumentar su cantidad a comprar, realizar la compra o cancelarla. El módulo *admin.js* es el encargado de gestionar los productos y categorías, enviarle la información de las gráficas al *html* asociado para la visualización de las estadísticas, gestionar el cambio en las interfaces del cliente. El módulo paginación.js se encarga del cambio de página interactivo, cuando hay una cantidad de 50 productos en la página, el siguiente producto es añadido en una nueva página pero que pertenece a esa misma categoría. Cada clase implementadas incluye el tratamiento de errores para cada funcionalidad haciendo un sistema más entendible a la hora de interactuar con el usuario.

Ilustración 8 Controlador

2.6 Patrones de diseño

Los patrones de diseño son soluciones predefinidas para problemas que se han dado anteriormente en otros proyectos, es decir, aunque cada aplicación es única, posee partes comunes con otras aplicaciones ya sea el acceso a datos, las operaciones entre sistemas o la creación de objetos. Se puede solucionar problemas utilizando algún patrón de diseño ya que son soluciones probadas y documentadas por una gran cantidad de programadores.

Inyección de dependencias: La Inyección de Dependencias (o la Inversión del control) se basa en que, en vez de que una clase vaya a buscar los objetos que necesita llamando a métodos que se los proporcionan, se hace al revés, siendo estos últimos los que se asignan automáticamente al objeto. En el sistema implementado para admin.js como para

otras clases controladoras, se les pasa los scripts o clases necesarias para su funcionamiento, en la creación del método constructor como se observa en la ilustración 9.

```
(function () {
    'use strict';
    angular
            .module('app')
            .controller('Admin', Admin);
    Admin.$inject = ['$rootScope', '$scope', 'Product', '$location',
                'Category', '$http', 'API_URL', '$cookieStore', 'Estadisticas'];
    function Admin($rootScope, $scope, Product, $location,
                Category, $http, API URL, $cookieStore, Estadisticas) {
        var ac = this;

        ac.Estadisticas = {};
        ac.logout = function () {};
        ac.changeUserOpc = function () {};
        ac.Product = {};
        ac.Category = {};
        ac.ActualizarGraf = function () {};
        ac.Config = {};
        $scope.isOpenRight = function () {};
    }

})();
```

Ilustración 9 Patrón Inyección de dependencias

2.6.1 Patrones de diseño GOF

***Singleton*:** Es un patrón especifico que utiliza *angularjs*, se pone de manifiesto en la página principal *index.html*, esta encapsula las vistas *home.html* y *home.html* haciendo que solo se pueda instanciar *index* y acceder a las vistas home y admin a través de ella, por lo que se puede describir como una instancia única y que posee un acceso global a ella.

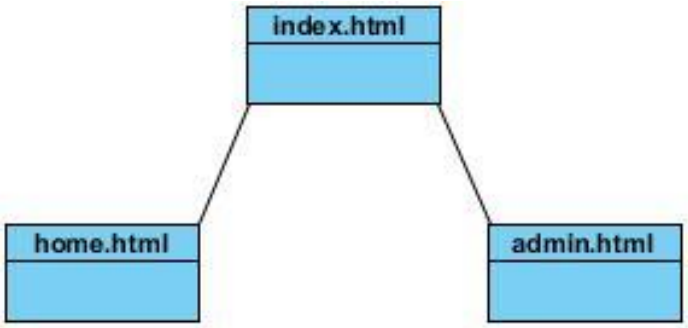


Ilustración 10 Patrón Singleton

***Observer*:** El controlador *javascript* está observando los cambios realizados en la base de datos para actualizarlos luego en la interfaz. En funcionalidades como adicionar un nuevo producto, este se guarda en el servidor, a su vez en la interfaz home.html, se debe mostrar el nuevo producto a vender por lo que el script home.js se encarga de observar si hay algún cambio en este servidor y pedir los nuevos datos para actualizarlos en la interfaz asignada.

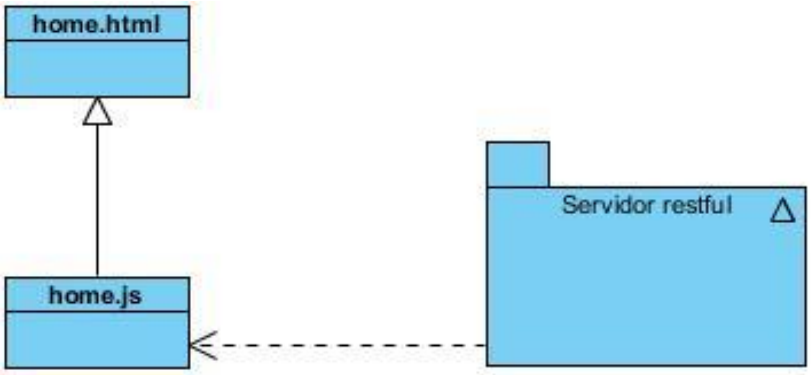


Ilustración 11 Patrón Observer

2.6.2 Patrones de diseño GRASP

Controlador: Este patrón se pone de manifiesto en el módulo *admin.js* siendo este el encargado de manejar los eventos de varias entidades, se evidencia este patrón de manera más clara en la arquitectura MVC utilizada para el desarrollo de *software*.

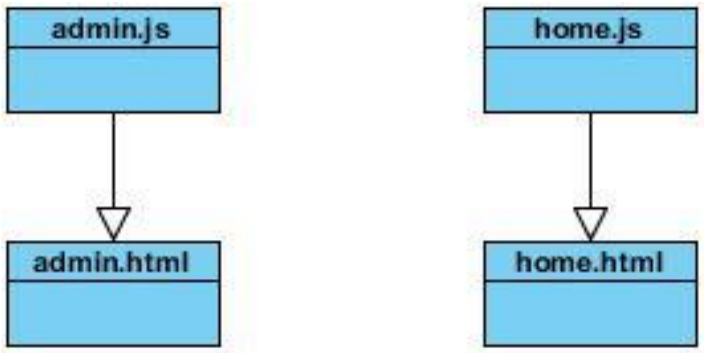


Ilustración 12 Patrón Controlador

Experto: El módulo *admin.js* tiene la información necesaria para llevar a cabo las tareas requeridas, en funcionalidades del sistema tales como la creación y modificación de productos.

```
add: function () {
    ac.dataLoading = true;
    Product.Create(this.newProduct).then(function (res) {
        ac.Product.list.push(res);
        ac.dataLoading = false;
        alert("Producto Agregado correctamente");
    });
```

Ilustración 13 Patrón Experto

Alta Cohesión: Los módulos *admin.js*, *home.js*; así como los *scripts* que implementan los servicios: Producto.js, Carrusel.js, Category.js, Estadisticas.js, Loguin.js poseen una alta cohesión ya que cada módulo se encarga de generar las operaciones de su responsabilidad. En el módulo Producto.js solamente se realiza la obtención de datos del servidor mediante el consumo de un servicio *RestFul* que involucra a los productos o sus funciones.

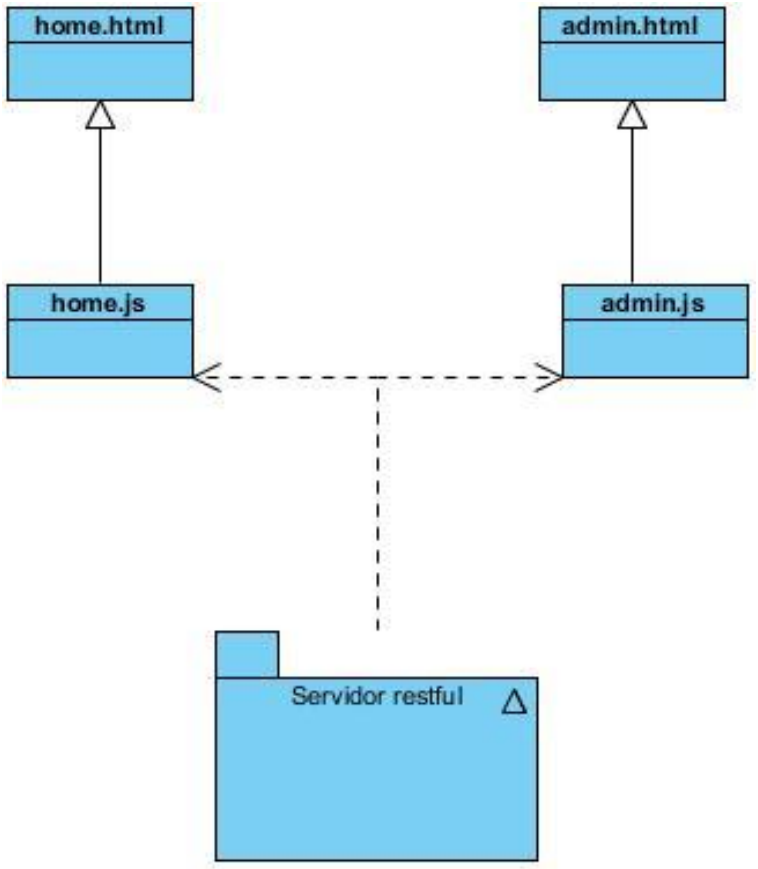


Ilustración 14 Patrón Alta Cohesión

Conclusiones parciales

- El levantamiento de los requisitos funcionales de la aplicación, permitió dar respuesta a las necesidades del problema, describiendo lo que el sistema debe hacer en cada momento.
- La definición de la propuesta de solución del problema, detallándola con la ayuda de los artefactos propuestos por la metodología AUP-UCI, posibilitó estructurar todo el proceso de desarrollo.
- La selección del patrón arquitectónico y los patrones de diseño, facilitó garantizar el correcto funcionamiento y organización del sistema.

Capítulo 3: Implementación y Pruebas

Introducción:

El objetivo de este capítulo luego de haber realizado el diseño es desarrollar la solución y luego someter la herramienta a pruebas internas pues podría presentar fallas, mediante casos de pruebas donde se validarán las soluciones y finalmente poder realizar el despliegue en los sistemas de producción.

3.1 Tareas de ingeniería o programación

Mediante la metodología de desarrollo de *software* escogida se crean diferentes etapas o planes, una de ellas es el plan de iteraciones en el cual las historias de usuarios seleccionadas para cada entrega son desarrolladas y probadas en un ciclo de iteración, de acuerdo al orden preestablecido. La planificación de las tareas de ingeniería está relacionada con dichas iteraciones por lo que se documentó para este sistema la sucesión de tres iteraciones, se escogieron de cada iteración las 2 HU principales a las cuales se les realiza el desglose por tareas. La siguiente tabla muestra estas relaciones:

Tabla 7 Tareas por Historias de Usuario

Historias de usuario	Tareas por historia de usuario
Elaborar listado de pedidos	1. Desarrollar una funcionalidad que permita seleccionar un tipo de producto. 2. Desarrollar una funcionalidad que permita vincular cada producto al servidor *RestFul*. 3. Desarrollar una funcionalidad que permita mostrar el producto seleccionado en la tabla de venta o carrito de compra.
Realizar pago en efectivo	4. Desarrollar una funcionalidad que permita calcular el precio total de la venta. 5. Desarrollar una funcionalidad que permita insertar el monto a pagar. 6. Desarrollar una funcionalidad que permita calcular el valor del vuelto resultante.

Mostrar estadísticas	7. Desarrollar una funcionalidad que permita consumir un servicio del servidor *RestFul* el cual tenga la información para la gestión de las estadísticas. 8. Desarrollar una funcionalidad que permita mostrar las estadísticas antes planteadas.
Adicionar producto	9. Desarrollar una funcionalidad que permita adicionar un nuevo producto con sus respectivos atributos. 10. Desarrollar una funcionalidad que permita actualizar la lista de los productos después de la adición.
Modificar producto	11. Desarrollar una funcionalidad que permita modificar cualquiera de los campos o atributos de un producto dado, de la lista de productos. 12. Desarrollar una funcionalidad que permita actualizar la lista de los productos después de la modificación.
Eliminar producto	13. Desarrollar una funcionalidad que permita eliminar un producto dado, de la lista de productos. 14. Desarrollar una funcionalidad que permita actualizar la lista de los productos después de la eliminación.

Para la implementación del sistema cada HU es separada en diferentes tareas de ingeniería; las cuales poseen varias especificaciones o campos a llenar para un mejor entendimiento de cada tarea. Para confeccionar las tablas para las tareas de ingeniería se generaron los siguientes campos:

- **No. de tarea:** Numeración continua que identifica a la tarea.
- **No. de HU:** Número de la HU a la cual pertenece.
- **Nombre de la tarea:** Identificación literal de la tarea.

- **Tipo de tarea:** Tipo de tarea, dígase diseño, desarrollo, prueba.
- **Puntos estimados:** Representación en porciento de la cantidad de tiempo estimada de una semana, que se utilizará para su realización.
- **Fecha inicio:** Fecha estimada de inicio de realización.
- **Fecha fin:** Fecha estimada de fin de realización.
- **Descripción:** Se describe en que consiste la tarea y que elementos deben cumplirse para declarar la tarea terminada.

Se pondrá de ejemplo a continuación las 3 tareas de ingeniería referente a la HU1, las demás se encuentran en el anexo 2.

Tabla 8 TI1-Seleccionar un tipo de producto

Tarea de Ingeniería	
No. de tarea: 1	**No. de HU:** 1
Nombre de la tarea: Desarrollar una funcionalidad que permita seleccionar un tipo de producto.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.6
Fecha inicio:	**Fecha fin:**
Descripción: Se implementa en la vista principal de la aplicación donde se exponen todos los productos existentes en el servidor *RestFul*, y el sistema debe permitir la selección del producto deseado por el cliente.	

Tabla 9 TI2-Vincular cada producto al servidor RestFul

Tarea de Ingeniería	
No. de tarea: 2	**No. de HU:** 1
Nombre de la tarea: Desarrollar una funcionalidad que permita vincular cada producto al servidor *RestFul*.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.6
Fecha inicio:	**Fecha fin:**

Descripción: Se implementa la comunicación con el servidor *RestFul*, mediante la cual va a consumir los servicios que brinda este servidor y contiene los datos de los productos. El sistema debe permitir que cuando el cliente escoja un producto en la interfaz este sea comparado con los productos descritos en el servidor y ser encontrado el que el cliente desea.

Tabla 10 TI3-Mostrar producto

Tarea de Ingeniería	
No. de tarea: 3	**No. de HU:** 1
Nombre de la tarea: Desarrollar una funcionalidad que permita mostrar el producto seleccionado en la tabla de venta o carrito de compra.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.6
Fecha inicio:	**Fecha fin:**
Descripción: Una vez encontrado el producto seleccionado, el sistema muestra los datos necesarios del producto en la tabla de venta o carrito de compra.	

3.2 Estándar de codificación.

Al comenzar un proyecto de *software*, se debe establecer un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Un estándar de codificación completo comprende todos los aspectos de la generación de código; desde la creación de las variables hasta la definición de clases. Aunque la legibilidad en el código es el resultado de muchos factores, las técnicas de codificación son las principales responsables de lograr un código legible. Estas técnicas de codificación se le deben efectuar revisiones de rutina para probar si se están cumpliendo y si deben modificarse por alguna razón en específico.
A continuación, se muestra las técnicas utilizadas en este sistema:

- Ninguna función debe tener más de 200 líneas.
- Los valores de los numerativos deben ser con letras mayúsculas.

- Los nombres de las variables son cortos y significativos, así cualquiera que no haya desarrollado el código pueda entender que guarda esa variable.

Ejemplo: var wSocket = new WebSocket(API_URL);

- Las variables utilizan la nomenclatura *camelcase* (encamellado) comienzan con letra minúscula y cada palabra consecutiva en el nombre comienza con letra mayúscula.

- Las carpetas y archivos del sistema están escritos en inglés.

3.3 Estructura y descripción de la aplicación

- La estructura principal del sistema se encuentra organizada por capetas. El *index.htlm,* archivo principal de la aplicación donde se cargan las vistas gestionadas en el sistema.

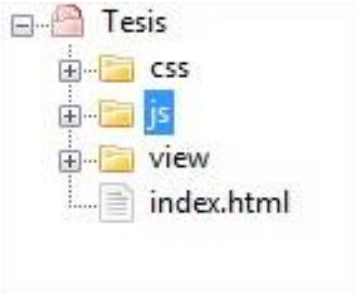


Ilustración 15 Estructura principal

- La carpeta css posee los materiales y estilos que se utilizan para la creación de las vistas.

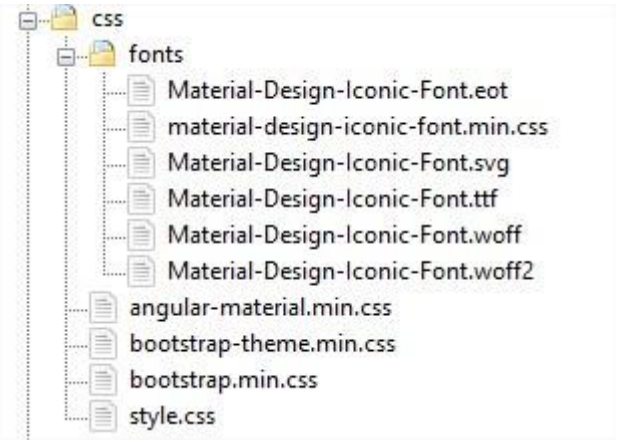


Ilustración 16 Carpeta css

- La carpeta *js* contiene en una primera instancia la carpeta *controller* donde se encuentran las clases controladoras del sistema; luego en la carpeta *lib* se observa las librerías utilizadas para la gestión de diversas funcionalidades y los *frameworks* necesario para la creación del sistema. A continuación, se muestra en una carpeta llamada *script* el archivo encargado de hacer funcionar el sistema. En una última carpeta llamada *services* se tienen los archivos encargados de procesar y adquirir la información del servidor *RestFul*.

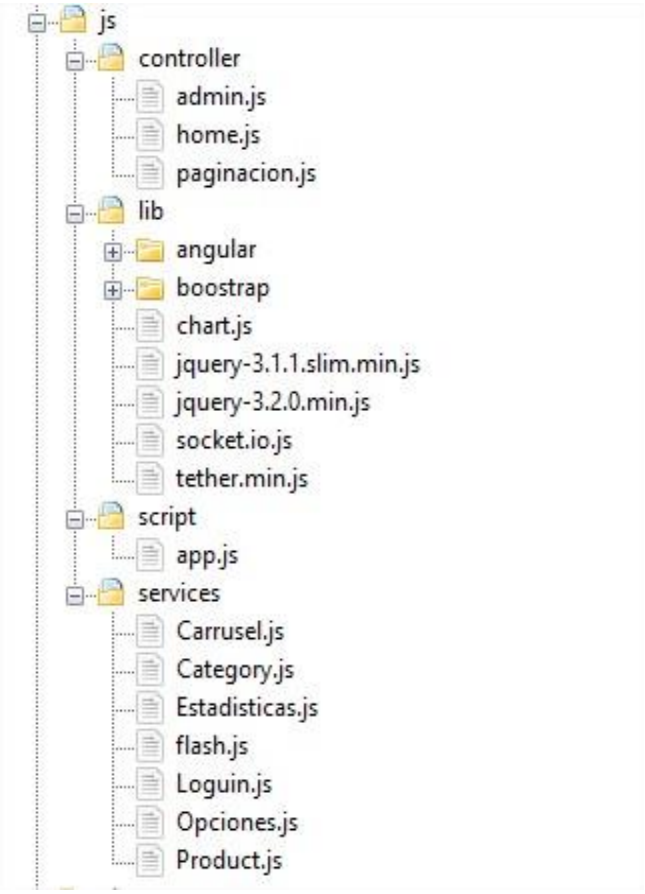


Ilustración 17 Carpeta js

- La carpeta *view* contiene 2 archivos con un fragmento de código, en dependencia de la interfaz a la que se quiera acceder este fragmento de código es ejecutado en el archivo principal *index.htlm*.

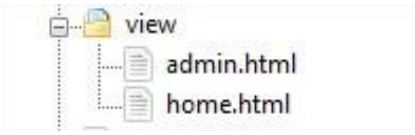


Ilustración 18 Carpeta view

3.4 Descripción del sistema

El sistema cuenta con 2 interfaces; en la interfaz principal se encuentra una ventana donde se muestra la tabla de venta o carrito de compra y una ventana en la cual se tiene la interfaz de logueo. En la interfaz de administración se encuentra la gestión de

productos, la gestión de características, una ventana que muestra las gráficas y estadísticas y una ventana que muestra las opciones de administración.

3.4.1 Descripción de la interfaz principal

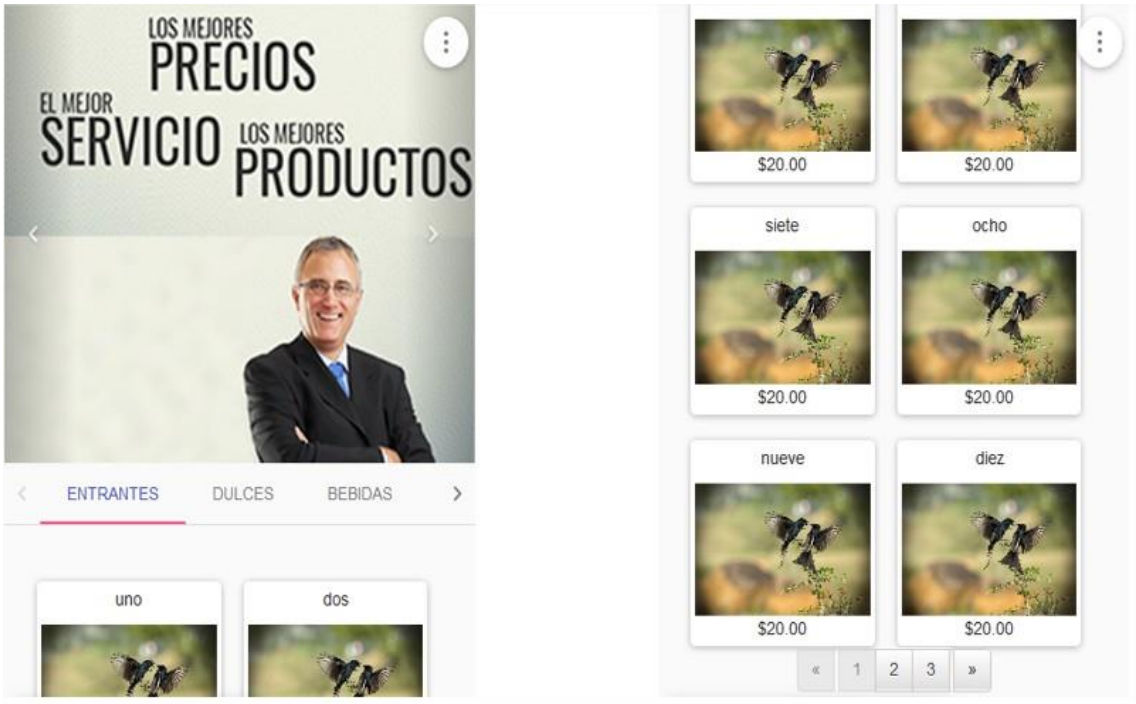


Ilustración 19 Interfaz principal

En esta interfaz se muestra en un primer plano un *banner* interactivo el cual puede ser modificado en la administración, a la izquierda de este *banner* existe un botón que despliega el acceso al carrito de compra y a la interfaz de *logueo*. Después se exponen los productos ofertados al cliente divididos por categorías; donde el vendedor puede seleccionar el o los productos deseados por el cliente haciendo *touch*[11], en el icono correspondiente al producto el cual posee una foto que lo representa. Luego el producto en consecuencia es añadido junto con sus datos en la tabla de venta o carrito de compra. Al final de la interfaz se tiene un paginado el cual mediante que se vallan añadiendo nuevo productos ira aumentando su cantidad para mantener 50 productos por cada página, y así brindar una interfaz más ordenada y eficaz.

[11] Equivalente al clic en dispositivos móviles.

3.4.2 Descripción de la tabla de venta o carrito de compra

Ilustración 20 Interfaz carrito de compra

En esta interfaz o *modal*[12] se muestra el nombre de la tienda o negocio, este puede ser cambiado en la administración, a continuación, se observa un listado de todos los productos requeridos por el cliente y su respectivo precio, de estos productos se pueden cambiar la cantidad que desea compra haciendo *touch* nuevamente en el mismo producto. Debajo se aprecia un campo o *input* en el cual se ingresa el monto en efectivo de dinero a pagar por el cliente y siguiendo a este el vuelto resultante de la compra; si se ingresa un valor no aceptable (no numérico) la venta no podrá llevarse a cabo y poniendo en rojo el campo incorrecto. El precio por producto y el precio total de las ventas se actualizan automáticamente con cada cambio en la tabla de venta. Luego se tiene los botones “Aceptar” para completar la venta, “Cancelar” para descartar la venta e “Imprimir” para imprimir esta venta en un recibo de pago; para los botones “Aceptar” e “Imprimir” muestra una confirmación después de realizada la acción asignada a ellos. En el caso del botón “Aceptar” muestra “Compra realizada con éxito”, e “Imprimir” muestra el siguiente mensaje: “la venta se ha mandado a imprimir, espere su recibo”

[12] Es un formulario que se superpone a las demás quedando en primer plano de la web.

3.4.3 Descripción de la interfaz de logueo

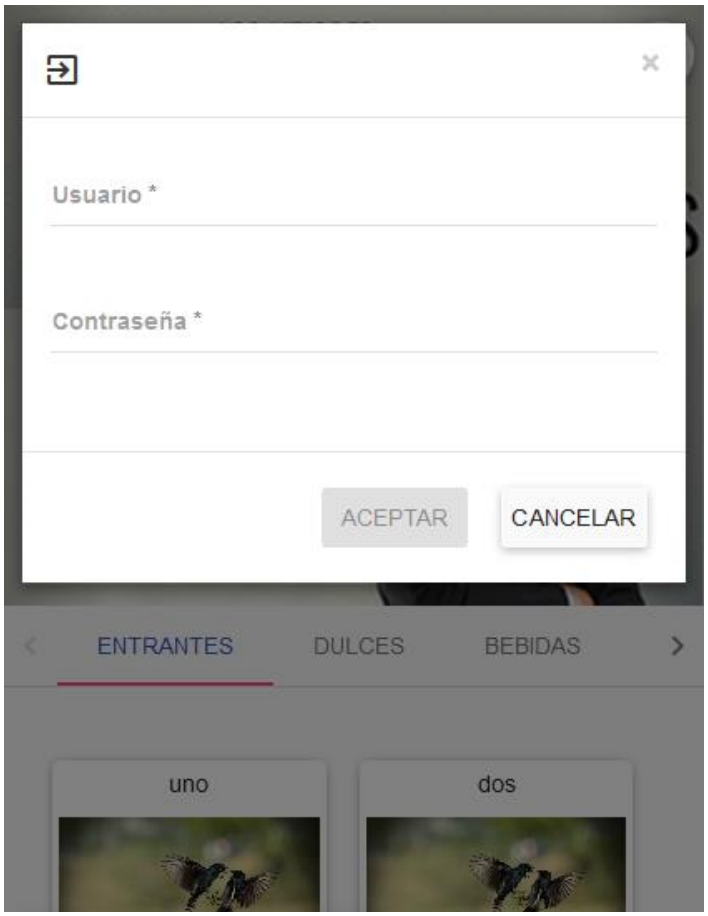


Ilustración 21 Interfaz de logueo

En esta interfaz o *modal* se muestra un campo o *input* en el cual se ingresa el usuario de administrador y un campo donde se ingresa la contraseña asociada a ese usuario; si en cualquiera de los campos se ingresa un valor no aceptable el logueo no podrá llevarse a cabo poniendo en rojo los campos incorrectos. Luego se tiene los botones “Aceptar” para acceder a la interfaz de administración y “Cancelar” para descartar el *logueo* y salir de esta interfaz.

3.4.4 Descripción de la gestión de productos

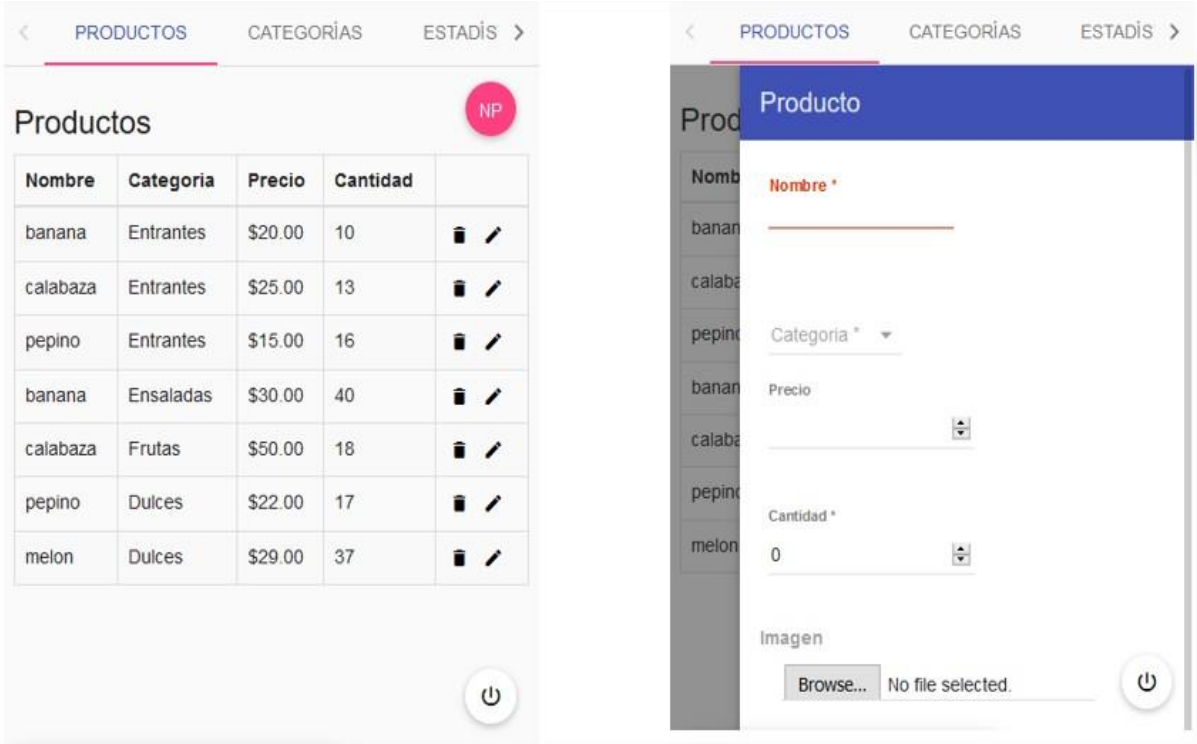


Ilustración 22 Interfaz para la gestión de productos

En esta interfaz se muestra un listado de los productos existentes en el servidor *RestFul*; de cada producto se conoce su nombre, categoría, precio y la cantidad existente en inventario. Por cada producto se tienen 2 botones uno para eliminarlos y otro para editar sus campos. En la parte derecha de esta interfaz se observa un botón el cual despliega una interfaz o *modal* para la adición de un nuevo producto; donde se muestran campos como: Nombre, Categoría, Precio, Cantidad e Imagen. Si se insertan valores no válidos en estos campos no se podrá adicionar el nuevo producto y el campo incorrecto se pondrá en rojo. Luego se tienen 2 botones “Cancelar” el cual descarta la adición de un nuevo producto y sale de este desplegable; y “Adicionar” el cual adiciona un nuevo producto al listado de productos y muestra el siguiente mensaje: “Producto agregado correctamente”. Por ultimo al final y a la derecha posee un botón salir de la interfaz de administración y volver a la interfaz principal.

3.4.5 Descripción de la gestión de categorías

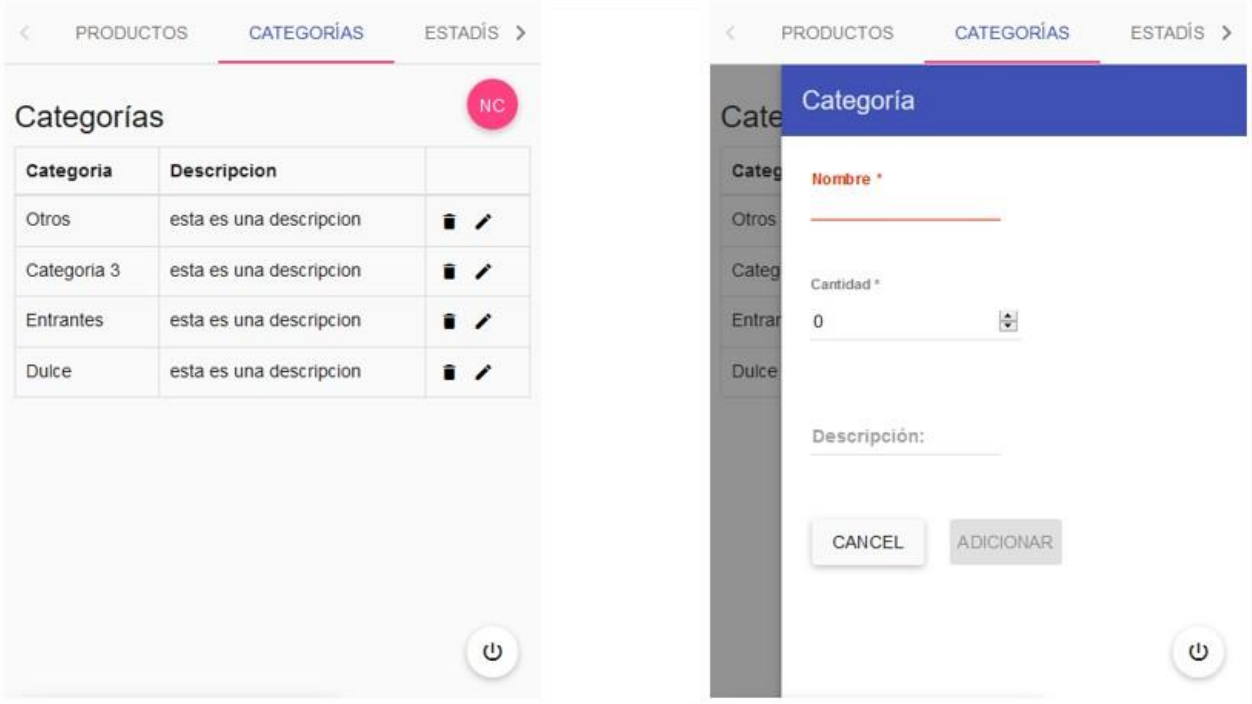


Ilustración 23 Interfaz para la gestión de categorías

Cuando se despliega el sistema, en un principio, tanto las categorías como los productos no tienen existencias en el servidor, luego el administrador se encarga de ingresar las categorías que va a tener el negocio.

En esta interfaz se muestra el listado de las categorías existentes en el servidor *RestFul*, donde se puede conocer sus nombres y las descripciones de las mismas. Por cada categoría se tienen 2 botones uno para eliminarlas y otro para editar sus campos. En la parte derecha de esta interfaz se observa un botón el cual despliega una interfaz o *modal* para la adición de una nueva categoría donde se muestran campos como: Nombre, y Descripción. Si se insertan valores no válidos en estos campos no se podrá adicionar la nueva categoría y el campo incorrecto se pondrá en rojo. Luego se tienen 2 botones “Cancelar” el cual descarta la adición de una nueva categoría y sale de este desplegable; y “Adicionar” el cual adiciona una nueva categoría al listado de categoría y muestra el siguiente mensaje: “Categoría agregada correctamente”. Por ultimo al final y a la derecha posee un botón salir de la interfaz de administración y volver a la interfaz principal.

3.4.6 Descripción de los gráficos y estadísticas

Ilustración 24 Gráficos y estadísticas

En esta interfaz se muestra un gráfico de barras donde se muestra la cantidad de productos existentes por categorías. Luego muestra con su foto correspondiente el producto más y menos vendido del día, la semana y el mes. Por ultimo al final y a la derecha posee un botón salir de la interfaz de administración y volver a la interfaz principal.

Un control amplio sistema de inventario y contabilidad cuenta con funcionalidades como; manejar un sistema de descuento por compra al por mayor, la exportación e importación de la contabilidad de inventario a Excel, manejar la venta teniendo en cuenta el IVA (Impuesto sobre el valor agregado), control de lotes y caducidad de producto. También

cuentan con un historial de modificaciones ordenadas por fecha (ventas, traspasos, entradas y salidas de productos) e informes automáticos sobre la existencia de productos en inventario. Estas funcionalidades pudieran ser agregadas en una segunda versión del sistema para aumentar el nivel de inventario y contabilidad.

3.4.7 Descripción de la gestión de las opciones de administración

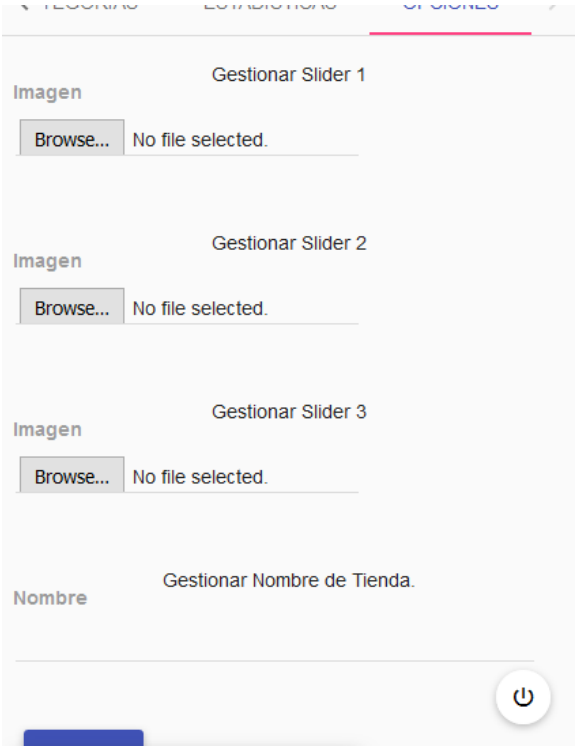


Ilustración 25 Interfaz para el Admin

El banner interactivo que se encuentra en la interfaz principal se le puede añadir 3 imágenes de información. En esta interfaz de administración se muestra un botón “Seleccionar archivo”, el cual añade una imagen de información al banner interactivo. Luego se tiene un campo al cual se le añade el nombre que se le quiera poner a la tienda o negocio, el cual se muestra en el carrito de compra o tabla de venta y un botón aceptar para realizar la acción. Por ultimo al final y a la derecha posee un botón salir de la interfaz de administración y volver a la interfaz principal.

3.4.8 Descripción de la interfaz de usuario.

Ilustración 26 Interfaz de usuario

En esta interfaz se muestra un campo en el cual se puede cambiar el usuario y la contraseña del administrador; si en cualquiera de los campos no se ingresa ningún valor el cambio de usuario y contraseña no podrá llevarse a cabo poniendo en rojo los campos incorrectos. Luego se tiene el botone “Aceptar” para guardar estos cambios en el servidor.

3.4.9 Descripción de la interfaz modo sin conexión.

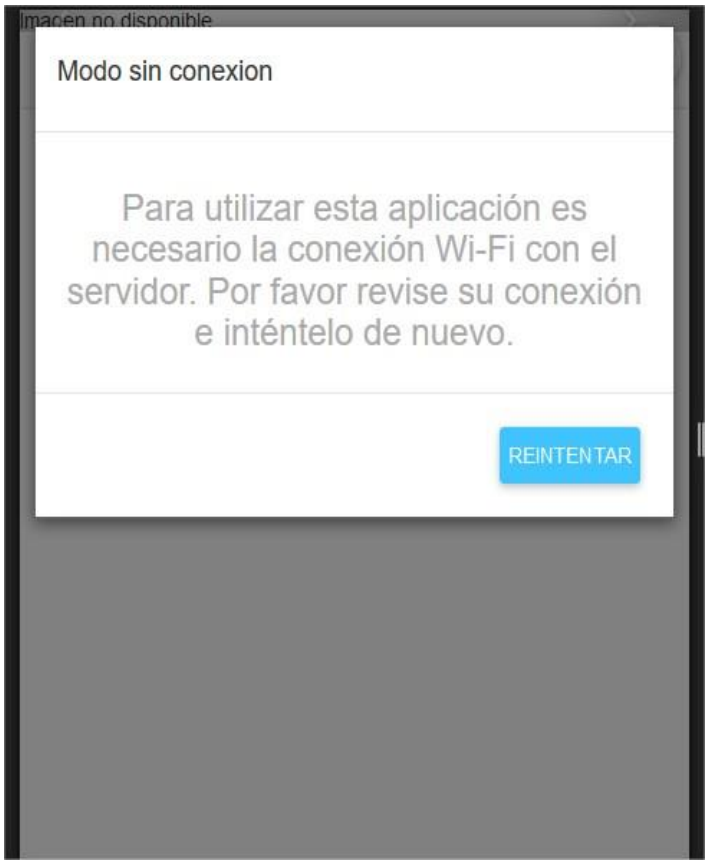


Ilustración 27 Interfaz modo sin conexión

Si el sistema no detecta el servidor *RestFul* donde se encuentra los datos de los productos este muestra este la anterior interfaz, la cual ofrece la notificación de desconexión y la posibilidad de volver a intentar conectarse.

3.5 Diagrama de despliegue

Un Diagrama de Despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria. (33)

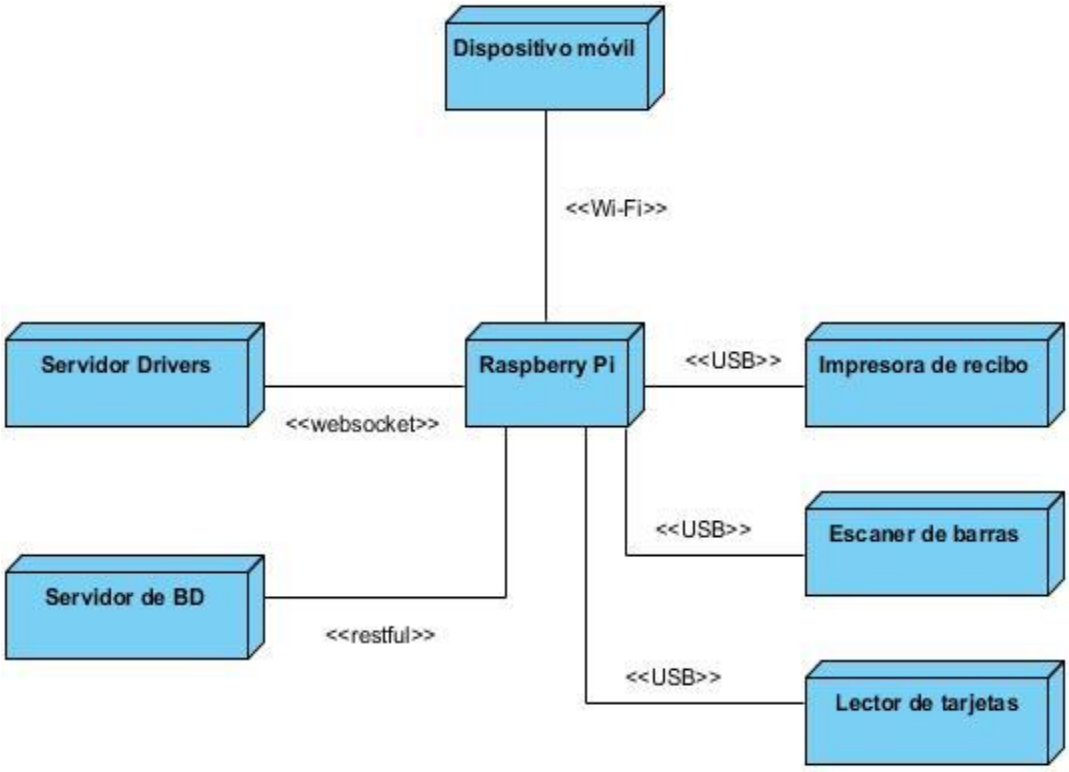


Ilustración 28 Diagrama de despliegue

Dispositivo móvil: Es la estación de trabajo que posee la aplicación móvil la cual el usuario utilizará para acceder a la información que brindan los servidores y donde se encontraran los *hardware* del terminal de punto de venta.

Servidor de *Drivers*: Es el servidor encargado de las funcionalidades y manejo del *hardware* que posee el terminal de punto de venta.

Servidor de BD: Se hace reseña al gestor de datos en los que se encuentra la información necesaria para la correcta utilización del sistema.

Conexión *Websocket*: Posibilita el establecimiento de una conexión *full-duplex,* entre el cliente y el Servidor de *Drivers*.

Conexión *RestFul*: Posibilita mediante un protocolo HTTP que el cliente consuma servicios del servidor de base de datos.

3.6 Pruebas de Aceptación

Las pruebas de aceptación es un tipo de ensayo que se realiza con el fin de verificar si el producto ha sido desarrollado de acuerdo con las normas y criterios establecidos y que el

mismo cumple con todos los requisitos especificados por el cliente. Las pruebas de aceptación pertenecen a la metodología de pruebas de caja negra, donde el usuario no está muy interesado en el trabajo interno o la codificación del sistema, sino que evalúa el funcionamiento global del sistema y los compara con los requisitos establecidos por ellos (34).

Las pruebas de aceptación se describen mediante una tabla llamada Caso de Prueba de Aceptación cuyo objetivo es probar si se desarrolló de manera correcta la funcionalidad acarreada a una HU específica. Se elaborarán las pruebas de aceptación para las principales HU para asegurar un correcto funcionamiento del sistema.
Para la elaboración de la tabla Caso de Prueba de Aceptación se describen los siguientes aspectos:

- **Código:** Identifica la prueba en cuestión e incluye el número de HU.
- **HU:** Número de la HU a la cual pertenece.
- **Nombre:** Nombre del caso de prueba.
- **Descripción**: Acciones que debe realizar el sistema.
- **Condiciones de ejecución:** Describe las características y elementos en general que debe tener el sistema para realizar el caso de prueba.
- **Entrada/Pasos de Ejecución:** Incluye las entradas necesarias del sistema y los pasos necesarios para realizar el caso de prueba.
- **Resultados Esperados:** Respuesta que debe dar el sistema luego de aplicar el caso de prueba.
- **Resultado Obtenido:** Respuesta visual del sistema después de realizar el caso de prueba.
- **Evaluación de la prueba:** Clasificación de la prueba en satisfactoria o insatisfactoria.

  Se pondrá de ejemplo a continuación 3 de los casos de pruebas aplicados al sistema:

Tabla 11 HU2_P1-Elaborar listado

Caso de Prueba de Aceptación	
Código: HU2_P1	**Historia de Usuario:** Elaborar listado de pedidos.
Nombre: Elaborar listado de pedidos.	

Descripción: El sistema debe permitir que el cliente llene en una tabla de venta o carro de compra el listado de los productos que desea comprar, con los precios individuales y totales de estos productos.
Condiciones de Ejecución: En la base de datos tienen que existir los productos que desea el cliente para satisfacer el pedido.
Entradas/Pasos de Ejecución: 1. Seleccionar los productos deseados por el cliente.
Resultado Esperado: El sistema debe mostrar los productos seleccionados, en la tabla de venta o carro de compra.
Resultado Obtenido: El producto fue mostrado correctamente.
Evaluación de la Prueba: Satisfactoria.

Tabla 12 HU5_P2-Realizar pago en efectivo

Caso de Prueba de Aceptación	
Código: HU5_P2	**Historia de Usuario:** Realizar pago en efectivo.
Nombre: Realizar pago en efectivo.	
Descripción: El sistema debe permitir que el cliente pueda pagar en efectivo para realizar la compra de cualquiera de los productos ofertados.	
Condiciones de Ejecución: En el sistema se tiene que haber elaborado un listado de pedidos.	
Entradas/Pasos de Ejecución: 1. Acceder a la tabla de venta o carrito de compra. 2. Entra el valor del monto de efectivo en el campo correspondiente	
Resultado Esperado: El sistema debe permitir entrar el valor del monto de dinero en efectivo con el cual es cliente va a pagar.	
Resultado Obtenido: El valor fue añadido correctamente.	
Evaluación de la Prueba: Satisfactoria.	

Tabla 13 HU11_P3-Mostrar estadísticas

Caso de Prueba de Aceptación	
Código: HU11_P3	**Historia de Usuario:** Mostrar estadísticas.
Nombre: Mostrar estadísticas.	
Descripción: El sistema debe permitir que se muestren diferentes estadísticas de los productos como: productos más y menos vendidos diariamente, semanalmente y mensualmente; y porciento de productos por categorías.	
Condiciones de Ejecución: En el sistema tiene que tener los datos de las ventas diarias, semanales y mensuales. El usuario tiene que estar logueado ya que solo se puede realizar esta acción desde la interfaz de administración.	
Entradas/Pasos de Ejecución: 1. Loguearse para acceder a la interfaz de administración. 2. Acceder a la ventana de gráficas.	
Resultado Esperado: El sistema debe permitir mostrar los datos en forma de gráficas y estadísticas.	
Resultado Obtenido: Los valores fueron mostrados correctamente.	
Evaluación de la Prueba: Satisfactoria.	

Las pruebas de aceptación fueron realizadas en una serie iteraciones; cuando las pruebas dan resultados no satisfactorios se analizan las no conformidades que aportaron estas pruebas y se implementa de manera correcta estas no conformidades obtenidas. Después de corregidas se vuelve a realizar la prueba de aceptación en una segunda iteración de pruebas a estas funcionalidades que presentaron no conformidades. El resultado de estas pruebas por iteración se muestra en el siguiente gráfico:

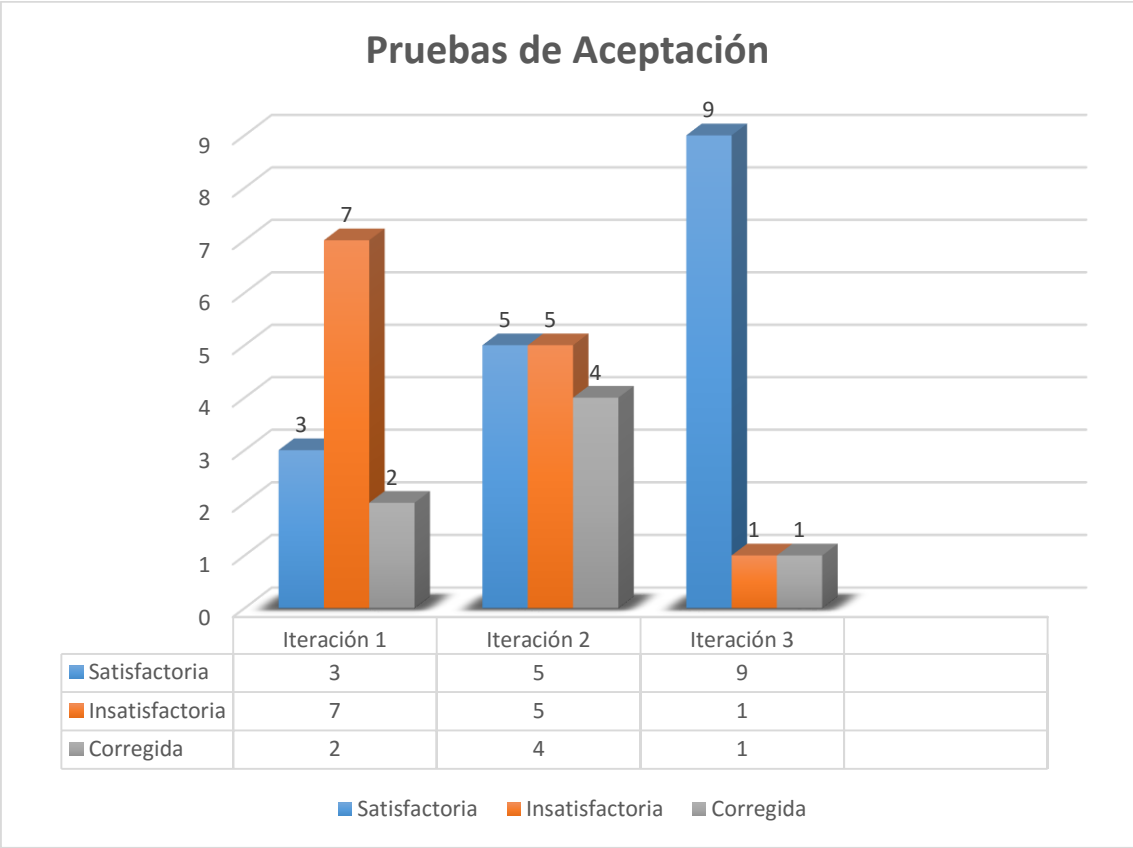


Ilustración 29 Gráfica de pruebas de aceptación

Conclusiones Parciales

- Mediante elaboración de las tareas de ingeniería para las historias permitió para una mejor compresión de los requisitos funcionales para su implementación.
- A través de la definición del estándar de codificación, se logró un mejor entendimiento del código resultante.
- A partir de las pruebas de aceptación a diferentes funcionalidades del sistema permitió documentar la evolución de la corrección de los errores obtenidos de estas pruebas.

Conclusiones generales

Con la realización de este trabajo se desarrolló un sistema multiplataforma que permite gestionar el proceso de venta para terminales de punto de venta, dándole cumplimiento al objetivo propuesto al inicio de la investigación. Conjuntamente se evidenció que:

- El estudio de los conceptos asociados a la investigación realizada y el análisis de las herramientas, tecnologías y aplicaciones similares permitió llevar a cabo el desarrollo del sistema.
- La utilización de la metodología de desarrollo AUP-UCI fue base para el desarrollo de la propuesta de solución y generó la documentación asociada a esta.
- Con la implementación del sistema se solucionan los problemas descritos en la situación problemática.
- Se realizaron pruebas al sistema, mejorando las funcionalidades del mismo mediante la corrección de los errores obtenidos.

Recomendaciones

Después de culminado el proceso de desarrollo de este sistema se recomienda:

- La creación de un módulo que permita; una vez tomada una foto para la creación de un producto o para la personalización del *banner*, este convierta la imagen al formato SVG.
- Aumentar el nivel de las funcionalidades definidas para control de inventario y contabilidad del sistema.

Bibliografía

1. DLE: venta - Diccionario de la lengua española - Edición del Tricentenario. [online]. [Accessed 25 January 2017]. Available from: http://dle.rae.es/?id=bXt7EYJ

2. ¿Qué es Punto? - Su Definición, Concepto y Significado. [online]. [Accessed 25 January 2017]. Available from: http://conceptodefinicion.de/punto/

3. NUÑEZ HORTA, Eddy, GARCÍA PÉREZ, Idris, NIÑO BENÍTEZ, Yisel, OLIVA MARTÍNEZ, Saily, MEJÍAS, Reyes and ROBERTO, Leandro. Subsistema Punto de venta del Sistema de Gestión Integral de Organizaciones Xedro-ERP. [online]. 27 April 2016. [Accessed 22 June 2017]. Available from: http://repositorio.uci.cu/jspui/handle/ident/9256

4. CONTRERAS, Andrés Velásquez. LA ORGANIZACIÓN, EL SISTEMA Y SU DINÁMICA: UNA VERSIÓN DESDE NIKLAS LUHMANN. *Revista EAN*. 31 July 2013. Vol. 0, no. 61, p. 129–156.

5. VARGAS-JIMÉNEZ, Ileana. LA ENTREVISTA EN LA INVESTIGACIÓN CUALITATIVA: NUEVAS TENDENCIAS Y RETOS. THE INTERVIEW IN THE QUALITATIVE RESEARCH: TRENDS AND CHALLENGERS. *Revista Electrónica Calidad en la Educación Superior*. 4 May 2012. Vol. 3, no. 1, p. 119–139.

6. Características del software POS - CSI SA Soluciones. [online]. [Accessed 25 January 2017]. Available from: http://www.csisasoluciones.com/noticias/noticias/noticias-generales/caracteristicas-del-software-pos

7. Los 3 tipos de TPV: sistemas tradicionales, web y móvil. [online]. [Accessed 25 January 2017]. Available from: http://www.gadae.com/blog/tipos-tpv-tradicional-virtual-movil/

8. Ventajas y desventajas de los sistemas de puntos de venta | eHow en Español. [online]. [Accessed 25 January 2017]. Available from: http://www.ehowenespanol.com/ventajas-desventajas-sistemas-puntos-venta-lista_528507/

9. Las ventajas del sistema de POS; Precisión; velocidad; Control de Inventario; Reporting. [online]. [Accessed 25 January 2017]. Available from: http://es.alpha-nouvelles.com/article/las-ventajas-del-sistema-de-pos

10. PATRICIO LETELIER TORRES and EMILIO A. SÁNCHEZLÓPEZ. *Metodologías Ágiles en el Desarrollo de Software*. [no date].

11. AUP Ingenieria de Software. [online]. [Accessed 25 January 2017]. Available from: http://ingenieriadesoftware.mex.tl/63758_AUP.html

12. TAMARA RODRÍGUEZ SÁNCHEZ. *Metodología de desarrollo para la Actividad productiva de la UCI*. [no date].

13. Sublime Text, un sofisticado editor de código multiplataforma - Genbeta. [online]. [Accessed 25 January 2017]. Available from: http://www.genbeta.com/herramientas/sublime-text-un-sofisticado-editor-de-codigo-multiplataforma

14. Bootstrap · The world's most popular mobile-first and responsive front-end framework. [online]. [Accessed 27 June 2017]. Available from: http://getbootstrap.com/

15. Visual Paradigm para UML. [online]. [Accessed 25 January 2017]. Available from: http://www.software.com.ar/p/visual-paradigm-para-uml#product-description

16. Perspectiva general - Apache Cordova. [online]. [Accessed 25 January 2017]. Available from: https://cordova.apache.org/docs/es/3.1.0/guide/overview/

17. Encuentro de Investigadores y Docentes de Ingeniería: - 33-10-3-2014-PaperWebRTC_TICAL.pdf. [online]. [Accessed 22 June 2017]. Available from: http://dspace.redclara.net/bitstream/10786/623/1/33-10-3-2014-PaperWebRTC_TICAL.pdf

18. FERNÁNDEZ, Burón, JAVIER, Francisco, ROMÁN PEÑA, Raúl, GARCÍA SALCINES, Enrique, UCEDA, Ramírez, MIGUEL, José and DE CASTRO LOZANO, Carlos. Mando a distancia virtual usable para la interacción con la televisión ubicua. *Revista Cubana de Ciencias Informáticas*. September 2013. Vol. 7, no. 3, p. 74–84.

19. Fielding Dissertation: CHAPTER 5: Representational State Transfer (REST). [online]. [Accessed 22 June 2017]. Available from: http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm#sec_5_2

20. ¿Qué es Raspberry Pi, dónde comprarla y cómo usarla? *ComputerHoy* [online]. [Accessed 27 June 2017]. Available from: http://computerhoy.com/noticias/hardware/que-es-raspberry-pi-donde-comprarla-como-usarla-8614

21. Arduino o Raspberry Pi, ¿cuál es la mejor herramienta para ti? « Hacedores.com | Maker Community. [online]. [Accessed 27 June 2017]. Available from: http://hacedores.com/arduino-o-raspberry-pi-cual-es-la-mejor-herramienta-para-ti/

22. HTML. *Mozilla Developer Network* [online]. [Accessed 27 June 2017]. Available from: https://developer.mozilla.org/es/docs/Web/HTML

23. HTML5. *Mozilla Developer Network* [online]. [Accessed 27 June 2017]. Available from: https://developer.mozilla.org/es/docs/HTML/HTML5

24. CSS. *Mozilla Developer Network* [online]. [Accessed 27 June 2017]. Available from: https://developer.mozilla.org/es/docs/Web/CSS

25. CSS3. *Mozilla Developer Network* [online]. [Accessed 27 June 2017]. Available from: https://developer.mozilla.org/es/docs/Web/CSS/CSS3

26. Lenguajes de programación. [online]. [Accessed 25 January 2017]. Available from: http://es.ccm.net/contents/304-lenguajes-de-programacion

27. Acerca de JavaScript. *Mozilla Developer Network* [online]. [Accessed 27 June 2017]. Available from: https://developer.mozilla.org/es/docs/Web/JavaScript/Acerca_de_JavaScript

28. JSON. [online]. [Accessed 27 June 2017]. Available from: http://www.json.org/json-es.html

29. ¿Qué es AngularJS? Introducción a este framework MVW. [online]. [Accessed 25 January 2017]. Available from: http://blog.escuelaweb.net/que-es-angularjs-introduccion-a-este-framework-mvw/

30. Qué es AngularJS. [online]. [Accessed 25 January 2017]. Available from: http://www.desarrolloweb.com/articulos/que-es-angularjs-descripcion-framework-javascript-conceptos.html

31. Microsoft PowerPoint - 2-requisitos.ppt [Modo de compatibilidad] - 2-requisitos.pdf. [online]. [Accessed 2 June 2017]. Available from: https://www.infor.uva.es/~mlaguna/is1/apuntes/2-requisitos.pdf

32. Requisitos para Sistemas de Información - get.php. [online]. [Accessed 2 June 2017]. Available from: http://www.lsi.us.es/docencia/get.php?id=6890

33. MARCA HUALLPARA HUGO MICHAEL and QUISBERT LIMACHI NANCY SUSANA. *Diagrama de Despliegue* [online]. Available from: virtual.usalesiana.edu.bo/web/practica/archiv/despliegue.doc

34. Software Testing – Pruebas de Aceptación | Que Computadora Comprar. [online]. [Accessed 2 June 2017]. Available from: http://quecomputadoracomprar.com/software-testing-pruebas-de-aceptacion-2/

Anexo

Anexo 1: Historias de usuario

Historia de Usuario	
Código: HU3	**Usuario:** Cliente
Nombre historia: Eliminar productos del listado de pedidos.	
Referencia: 2	**Prioridad:** Alta
Puntos estimados: 0.8	**Iteración asignada:** 3
Descripción: El sistema debe permitir que se eliminen los productos que el cliente no desea comprar de la lista de compra o carrito.	
Observaciones:	

Historia de Usuario	
Código: HU4	**Usuario:** Cliente
Nombre historia: Crear e imprimir un recibo de pago.	
Referencia: 1,2	**Prioridad:** Alta
Puntos estimados: 0,8	**Iteración asignada:** 1
Descripción: El sistema debe permitir imprimir un recibo como constancia de pago con la información de todos los productos añadidos al carito de compra como el precio y el nombre, también debe poseer el importe a pagar y el vuelto resultante.	
Observaciones:	

Historia de Usuario

Código: HU5	**Usuario:** Cliente
Nombre historia: Realizar pago en efectivo.	
Referencia: 1,2	**Prioridad:** Alta
Puntos estimados: 0.4	**Iteración asignada:** 1
Descripción: El sistema debe permitir que el cliente pueda pagar en efectivo para realizar la compra de cualquiera de los productos ofertados.	
Observaciones:	

Historia de Usuario	
Código: HU6	**Usuario:** Administrador
Nombre historia: Notificar detección de tarjeta.	
Referencia:	**Prioridad:** Alta
Puntos estimados: 0.4	**Iteración asignada:** 1
Descripción: El sistema debe permitir que se notifique con una alerta cuando ha sido detectada una tarjeta y mostrar la información de la misma, como el monto actual de dinero que esta posee.	
Observaciones:	

Historia de Usuario	
Código: HU7	**Usuario:** Administrador
Nombre historia: Mostrar notificación de descuento por tarjeta.	
Referencia: 1,2,6	**Prioridad:** Alta

Puntos estimados: 0.4	**Iteración asignada:** 1
Descripción: El sistema debe permitir que al realizar la compra de algún producto mediante una tarjeta este notifique el descuento realizado y el monto actual de la tarjeta después de realizada la compra.	
Observaciones:	

Historia de Usuario	
Código: HU8	**Usuario:** Cliente
Nombre historia: Comprar producto.	
Referencia: 1,2,5	**Prioridad:** Alta
Puntos estimados: 1.2	**Iteración asignada:** 2
Descripción: El sistema debe permitir que el cliente realice la compra de los productos añadidos a la tabla o carrito de compra, a su vez debe permitir cancelar la compra que no desee o que haya añadido por equivocación.	
Observaciones:	

Historia de Usuario	
Código: HU9	**Usuario:** Administrador
Nombre historia: Adicionar categoría.	
Referencia:	**Prioridad:** Alta
Puntos estimados: 0.4	**Iteración asignada:** 2
Descripción:	

El sistema debe permitir que se adicionen nuevas categorías de productos para una mejor organización y estructura del negocio.
Observaciones:

Historia de Usuario	
Código: HU10	**Usuario:** Administrador
Nombre historia: Eliminar categoría	
Referencia: 9	**Prioridad:** Alta
Puntos estimados: 0.4	**Iteración asignada:** 2
Descripción: El sistema debe permitir que se eliminen cualquiera de las categorías de productos no deseadas creadas anteriormente.	
Observaciones:	

Historia de Usuario	
Código: HU11	**Usuario:** Administrador
Nombre historia: Mostrar estadísticas.	
Referencia: 8,12,14,15	**Prioridad:** Alta
Puntos estimados: 0.6	**Iteración asignada:** 2
Descripción: El sistema debe permitir que se muestren diferentes estadísticas de los productos como: productos más y menos vendidos diariamente, semanalmente y	

mensualmente; y cantidad de productos por categorías.
Observaciones:

Historia de Usuario	
Código: HU12	**Usuario:** Administrador
Nombre historia: Adicionar producto.	
Referencia:	**Prioridad:** Alta
Puntos estimados: 0.4	**Iteración asignada:** 3
Descripción: El sistema debe permitir que se adicione un nuevo producto a ofertar con toda su información relacionada.	
Observaciones:	

Historia de Usuario	
Código: HU13	**Usuario:** Administrador
Nombre historia: Mostrar producto.	
Referencia: 12	**Prioridad:** Alta
Puntos estimados: 0.4	**Iteración asignada:** 2
Descripción: El sistema debe permitir que se muestre en un listado los productos con sus respectivos datos como: Nombre, Categoría, Precio y Cantidad para llevar un registro de los productos que se tienen en inventario.	
Observaciones:	

Historia de Usuario	
Código: HU14	**Usuario:** Administrador
Nombre historia: Modificar producto.	
Referencia: 12	**Prioridad:** Alta
Puntos estimados: 0.4	**Iteración asignada:** 3
Descripción: El sistema debe permitir que se modifiquen los parámetros de los productos como: Nombre, Categoría, Precio y Cantidad para una mejor gestión del sistema.	
Observaciones:	

Historia de Usuario	
Código: HU15	**Usuario:** Administrador
Nombre historia: Eliminar producto.	
Referencia: 12	**Prioridad:** Alta
Puntos estimados: 0.4	**Iteración asignada:** 3
Descripción: El sistema debe permitir que se eliminen los productos deseados por el administrador.	
Observaciones:	

Historia de Usuario	
Código: HU16	**Usuario:** Administrador
Nombre historia: Adicionar imagen del producto.	
Referencia: 12	**Prioridad:** Alta

Puntos estimados: 0.4	**Iteración asignada:** 3
Descripción: El sistema debe permitir que se adicione una imagen que represente esos productos para un mejor acoplamiento del cliente con el sistema.	
Observaciones:	

Historia de Usuario	
Código: HU17	**Usuario:** Administrador
Nombre historia: Ordenar listado de productos.	
Referencia: 12	**Prioridad:** Alta
Puntos estimados: 0.2	**Iteración asignada:** 3
Descripción: El sistema debe permitir que se ordenen los productos de acuerdo a: si son números; ascendentemente o viceversa y si son nombres; alfabéticamente.	
Observaciones:	

Anexo 2: Tareas de Ingeniería

Tarea de Ingeniería	
No. de tarea: 4	**No. de HU: 5**
Nombre de la tarea: Desarrollar una funcionalidad que permita calcular el precio total de la venta.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio:	**Fecha fin:**
Descripción: Se implementa en la vista principal de la aplicación donde al obtener los precios por productos, se quiere lograr que muestre el valor del precio total de la venta.	

Tarea de Ingeniería	
No. de tarea: 5	**No. de HU: 5**
Nombre de la tarea: Desarrollar una funcionalidad que permita insertar el monto a pagar.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio:	**Fecha fin:**
Descripción: Se implementa en la vista principal de la aplicación donde se creara un campo permita entrar valores numéricos, donde este se convertirá en el efectivo con el cual va a pagar el cliente.	

Tarea de Ingeniería	
No. de tarea: 6	**No. de HU: 5**
Nombre de la tarea: Desarrollar una funcionalidad que permita calcular el valor del vuelto resultante.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.2
Fecha inicio:	**Fecha fin:**

Descripción: Se calcula este valor restando el valor del precio total y el valor entrado como monto a pagar. Luego se implementa en la vista principal para mostrar dicho valor al cliente.

Tarea de Ingeniería	
No. de tarea: 7	**No. de HU: 11**
Nombre de la tarea: Desarrollar una funcionalidad que permita consumir un servicio del servidor *RestFul* el cual tenga la información para la gestión de las estadísticas.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio:	**Fecha fin:**
Descripción: Se implementa la comunicación con el servidor *RestFul*, mediante la cual va a consumir los servicios que registra la información de cuáles son los productos que más se venden y menos se venden en el día, la semana y el mes. También la información de las 3 categorías que más venden y una comparación entre ellas por semana.	

Tarea de Ingeniería	
No. de tarea: 8	**No. de HU: 11**
Nombre de la tarea: Desarrollar una funcionalidad que permita mostrar las estadísticas antes planteadas.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.6
Fecha inicio:	**Fecha fin:**
Descripción: Se muestran en una lista y en una gráfica de barras los valores obtenidos para las estadísticas.	

Tarea de Ingeniería	
No. de tarea: 9	**No. de HU: 12**

Nombre de la tarea: Desarrollar una funcionalidad que permita adicionar un nuevo producto con sus respectivos atributos.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.6
Fecha inicio:	**Fecha fin:**
Descripción: Se implementa en la vista de administración de la aplicación donde se expone en la sección de productos un listado de los productos y sus atributos existentes en el servidor. El sistema debe permitir la adición de un nuevo producto y agregarle al mismo cada uno de sus atributos (Nombre, Categoría, Precio, Cantidad e Imagen).	

Tarea de Ingeniería	
No. de tarea: 10	**No. de HU: 12**
Nombre de la tarea: Desarrollar una funcionalidad que permita actualizar la lista de los productos después de la adición.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio:	**Fecha fin:**
Descripción: Se implementa en la vista de administración de la aplicación donde se muestra la lista de productos actualizada, después de añadido el nuevo producto.	

Tarea de Ingeniería	
No. de tarea: 11	**No. de HU: 14**
Nombre de la tarea: Desarrollar una funcionalidad que permita modificar cualquiera de los campos o atributos de un producto dado, de la lista de productos.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio:	**Fecha fin:**

Descripción: Se implementa en la vista de administración de la aplicación donde se expone en la sección de productos un listado de los productos y sus atributos existentes en el servidor. El sistema debe permitir la modificación de cualquier campo del producto elegido.

Tarea de Ingeniería	
No. de tarea: 12	**No. de HU: 14**
Nombre de la tarea: Desarrollar una funcionalidad que permita actualizar la lista de los productos después de la modificación.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio:	**Fecha fin:**
Descripción: Se implementa en la vista de administración de la aplicación donde se muestra la lista de productos actualizada, después de modificado el producto requerido por el administrador.	

Tarea de Ingeniería	
No. de tarea: 13	**No. de HU: 15**
Nombre de la tarea: Desarrollar una funcionalidad que permita eliminar un producto dado, de la lista de productos.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio:	**Fecha fin:**
Descripción: Se implementa en la vista de administración de la aplicación donde se expone en la sección de productos un listado de los productos y sus atributos existentes en el servidor. El sistema debe permitir la eliminación de cualquier producto requerido por el administrador.	

Tarea de Ingeniería	
No. de tarea: 14	**No. de HU: 15**

Nombre de la tarea: Desarrollar una funcionalidad que permita actualizar la lista de los productos después de la eliminación.	
Tipo de tarea: Desarrollo.	**Puntos estimados:** 0.4
Fecha inicio:	**Fecha fin:**
Descripción: Se implementa en la vista de administración de la aplicación donde se muestra la lista de productos actualizada, después de eliminado el producto requerido por el administrador.	

Anexo 3: Pruebas de Aceptación

Caso de Prueba de Aceptación	
Código: HU11_P4	**Historia de Usuario:** Adicionar producto.
Nombre: Adicionar producto.	
Descripción: El sistema debe permitir que se adicione un nuevo producto a ofertar con toda su información relacionada.	
Condiciones de Ejecución: El usuario tiene que estar logueado ya que solo se puede realizar esta acción desde la interfaz de administración.	
Entradas/Pasos de Ejecución: 3. Loguearse para acceder a la interfaz de administración. 4. Acceder a la ventana de productos.	
Resultado Esperado: El sistema debe permitir añadir un nuevo producto con sus respectivos datos.	
Resultado Obtenido: El producto fue añadido correctamente.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU14_P5	**Historia de Usuario:** Modificar producto.
Nombre: Modificar producto.	
Descripción: El sistema debe permitir que se modifiquen los parámetros de los productos como: Nombre, Categoría, Precio y Cantidad para una mejor gestión del sistema.	
Condiciones de Ejecución: El usuario tiene que estar logueado ya que solo se puede realizar esta acción desde la interfaz de administración.	
Entradas/Pasos de Ejecución: 1. Loguearse para acceder a la interfaz de administración.	

2. Acceder a la ventana de productos.
Resultado Esperado: El sistema debe permitir modifica cualquiera de los productos existentes.
Resultado Obtenido: El producto fue modificado correctamente.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: HU14_P6	**Historia de Usuario:** Eliminar producto.
Nombre: Eliminar producto.	
Descripción: El sistema debe permitir que se eliminen los productos deseados por el administrador.	
Condiciones de Ejecución: El usuario tiene que estar logueado ya que solo se puede realizar esta acción desde la interfaz de administración.	
Entradas/Pasos de Ejecución: 1. Loguearse para acceder a la interfaz de administración. 2. Acceder a la ventana de productos.	
Resultado Esperado: El sistema debe permitir eliminar cualquiera de los productos existentes.	
Resultado Obtenido: El producto fue eliminado correctamente.	
Evaluación de la Prueba: Satisfactoria.	

Anexo 4: Entrevista

Entrevista 1: Punto de venta - El Garaje

PRESENTACIÓN

Buenos días, mi nombre es Jorge Luis Guerra Diez y como parte de mi trabajo de diploma

para optar por el título de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas estoy realizando una investigación acerca de los puntos de venta desplegados en el país. La información brindada en esta entrevista es de carácter confidencial, solo será utilizada para los propósitos de la investigación. Agradezco su colaboración.

INFORMACIÓN DEL PUNTO DE VENTA

Empresa: El Garaje

Persona entrevistada: María de José Martínez

Experiencia (Años): 2 años

Dirección de la empresa: Calle independencia, entre calle enamorado y A, reparto Pomo de Oro, Guanabacoa, La Habana.

PREGUNTAS-RESPUESTAS

P1 - ¿Cómo se realiza el proceso de venta desde la entrada del cliente al establecimiento?

R/ Todos los productos que se ofertan aquí tienen en una etiqueta el precio de los mismos. El cliente entra al establecimiento, si le gusta algo lo pide y yo se lo doy y le cobro.

P2 - ¿Cómo hace la dependiente para saber cuánto es el monto total de la venta?

R/ En este caso la dependiente soy yo, simplemente miro la etiqueta del producto y cálculo mentalmente o con calculadora el total si son más de un producto lo que el cliente desea.

P3 - ¿Cómo se realiza el cierre de caja al final del día?

R/ Voy apuntando en una hoja todo lo que voy vendiendo con el precio al lado del producto, al final del día el dinero que tenga debe ser igual a la suma de los precios de estos productos.

P4 - ¿Cómo se lleva el inventario en el establecimiento?

R/ El dueño tiene en varias hojas todos los productos y la cantidad que hay. Las hojas que yo escribo con las ventas del día, el dueño la rebaja de su inventario en hojas una vez por semana y cuenta los productos existentes para ver si coincide con los que él tiene en su hoja de inventario una vez al mes.

P5 - Si se fuera a crear un sistema informático para hacer estos procesos más fáciles, ¿qué funcionalidades le gustaría que tuviera?

R/ Me gustaría tener un sistema donde el dueño se le hiciera más fácil el trabajo con el inventario. También que pudiera imprimir un recibo de compra para que los clientes sepan de donde sale el dinero a pagar y para que puedan venir a cambiar su producto o a devolverlo si tienen algún problema. Y estoy segura que al dueño le gustaría que tuviera algo que le muestre cuales son los productos más vendidos para así priorizar la compra de estos.

Entrevista 2: Punto de venta – La piscina

PRESENTACIÓN

Buenos días, mi nombre es Jorge Luis Guerra Diez y como parte de mi trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas estoy realizando una investigación acerca de los puntos de venta desplegados en el país. La información brindada en esta entrevista es de carácter confidencial, solo será utilizada para los propósitos de la investigación. Agradezco su colaboración.

INFORMACIÓN DEL PUNTO DE VENTA

Empresa: La Piscina

Persona entrevistada: Rosendo Araujo Romero

Experiencia (Años): 20 años

Dirección de la empresa: Se encuentra en los interiores de la Universidad de las Ciencias Informáticas.

PREGUNTAS-RESPUESTAS

P1 - ¿Cómo se realiza el proceso de venta desde la entrada del cliente al establecimiento?

R/ El cliente hace la cola hasta que le toque su turno, en ese tiempo puede observar los productos que oferta el establecimiento en un menú, el cual se actualiza diariamente. Luego cuando es su turno hace el pedido a la dependiente, y la dependiente pasa la orden a los elaboradores para la realización del pedido. La dependiente sigue tomando pedidos y cuando está la orden del cliente, esta le da su pedido y le dice cuanto es lo que tiene que pagar; la dependiente cobra y le da el vuelto al cliente.

P2 - ¿Cómo hace la dependiente para saber cuánto es el monto total de la venta?

R/ Al principio se le da un papel con los precios de todos los productos, la dependiente consulta ese papel hasta que se aprenda todos los precios de memoria.

P3 - ¿Cómo se calcula el vuelto resultante de la venta de un producto?

R/ El vuelto se calcula con una calculadora o con un papel y una hoja.

P4 - ¿Cómo se realiza el cierre de caja al final del día?

R/ La dependiente tiene una hoja con la mercancía que se sacó del almacén, al final del día los productos sobrantes tienen que coincidir con los productos que no se han vendido, el total de dinero coincidir con el costo de los productos que si se vendieron.

P5 - ¿Cómo se lleva el inventario en el establecimiento?

R/ El administrador tiene en unas hojas los productos en el almacén y al final de la semana con las hojas que se sacan diariamente con los productos a vender, este realiza la rebaja de los productos en inventario o la entrada de algún producto que haya llegado algún día de esa semana, ya que este se apunta en esa hoja diaria también.

P6 - Si se fuera a crear un sistema informático para hacer estos procesos más fáciles, ¿qué funcionalidades le gustaría que tuviera?

R/ Me gustaría tener un sistema donde el administrador tuviera en el celular o en la computadora todos los productos y poder realizar de manera sencilla la rebaja de estos productos del almacén. También me gustaría que tuviera una lista que te muestre con nombre, precio y cantidad los productos que existen en el almacén.

Entrevista 3: Tienda recaudadora de divisa – Peletería Claudia

PRESENTACIÓN

Buenos días, mi nombre es Jorge Luis Guerra Diez y como parte de mi trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas estoy realizando una investigación acerca de los puntos de venta desplegados en el país. La información brindada en esta entrevista es de carácter confidencial, solo será utilizada para los propósitos de la investigación. Agradezco su colaboración.

INFORMACIÓN DEL PUNTO DE VENTA

Empresa: Peletería Claudia

Persona entrevistada: Reinaldo Ortega

Experiencia (Años): 16 años

Dirección de la empresa: calle 1ra entre 2 y Paseo, Hotel Cohíba, Vedado, La Habana.

PREGUNTAS-RESPUESTAS

P1 - ¿Cómo se realiza el proceso de venta desde la entrada del cliente al establecimiento?

R/ El cliente llega a la tienda, pregunta por algún producto por el que esté interesado y como todos los productos están en vitrinas se le busca uno igual en el almacén, al tener el producto en su mano el cliente se lo prueba en caso de ser un zapato o un cinto o lo revisa en caso de ser un bolso o una mochila y si le gusta pasa a la caja. En la caja, la cajera agrega el pedido a la caja registradora, le dice el precio total del pedio y el cliente paga. Después de realizado el pago; la cajera le da el vuelto resultante de la compra, el producto envuelto y el comprobante o recibo de compra.

P2 - ¿Cómo son realizado los cálculos referentes al vuelto y el total a pagar por el cliente?

R/ La cajera toma el código de barras de cada producto a comprar y lo pasa por el escáner de barras, este se muestra en la caja registradora con el precio a pagar, el nombre y la cantidad que se va a comprar. Se le añade también la cantidad de dinero con la que el cliente pagó, y la caja registradora calcula de forma automática el vuelto.

P3 - ¿Cómo se controla el inventario en esta caja registradora?

R/ Tanto para el control de inventario como para los cortes de caja la información de la caja registradora es enviado a una computadora por la red donde el administrador informático es el encargado de revisar y supervisar esa información.

P3 - ¿Cuál es el nombre de esta caja registradora y del sistema que tiene instalado?

R/ Caja Registradora Quorion QMP Q-2044 y el sistema es QMP POS.

P4- ¿Podría instalarse otro sistema más actual en esta caja registradora?

R/ No, estos sistemas no admiten actualizaciones.

P6 - Si se fuera a crear un sistema informático para hacer estos procesos más fáciles, ¿qué funcionalidades le gustaría que tuviera?

R/ Me gustaría que el sistema tuviera algún tipo de alerta para saber cuándo se está acabando un producto determinado.