UNIVERSIDAD PÚBLICA DE EL ALTO

CARRERA INGENIERÍA DE SISTEMAS

PROYECTO DE GRADO

SISTEMA INFORMÁTICO WEB DE COMPRAS, VENTAS Y CONTROL DE STOCK DE AUTOPARTES CASO: EMPRESA BARRA

Para optar al título de Licenciatura en Ingeniería de Sistemas

Mención: INFORMÁTICA Y COMUNICACIONES

Postulante:	Lidia Mamani Villca
Tutor Metodológico:	Ing. Marisol Arguedas Balladares
Tutor Especialista:	Lic. Freddy Salgueiro Trujillo
Tutor Revisor:	Ing. Ramiro Kantuta Limachi

El Alto – Bolivia

2020

DEDICATORIA

A Diosito que me ha dado la vida y fortaleza para terminar el presente Proyecto de Grado.

A mis señores padres Jose Mamani Canaza y Teodocia Villca y hermanos por estar ahí cuando más los necesite por su ayuda y constante cooperación en los momentos difíciles y por sus sabias enseñanzas y concejos.

AGRADECIMIENTOS

A Dios, por darme la inspiración, la vida y el conocimiento necesario en mi diario vivir.

Agradecer a todos los que tuvieron que ver en el desarrollo y conclusión de este proyecto de grado.

A mi tutora metodológico Ing. Marisol Arguedas Balladares por su conocimiento, apoyo, confianza, tiempo, persistencia, paciencia y motivación que brindo hacia el desarrollo del presente.

A mi tutor especialista Lic. Freddy Salgueiro Trujillo por compartir sus conocimientos, brindarme sus orientaciones, sugerencias con paciencia motivación durante el desarrollo del presente proyecto.

A mi tutora revisora Ing. Ramiro Kantuta Limachi por su disponibilidad de tiempo, su acertada orientada y observaciones brindadas en la realización del proyecto.

A la Universidad Pública de El Alto, por acogerme en sus aulas durante todos los años de estudio, así también a la carrera Ingeniería de Sistemas y a mis compañeros (as)de estudio por su apoyo incondicional.

"El éxito es fruto de una

persistente labor en torno

a los objetivos trazados."

(Osorio, 2008)

RESUMEN

En la actualidad los sistemas se han convertido en una herramienta fundamental y precisa para el crecimiento de toda empresa. El presente proyecto fue desarrollado para agilizar el proceso compra, venta y control de stock de autopartes para la empresa barra.

La web ha evolucionado desde su creación de forma rápida en diferentes aspectos, como sabemos la tecnología es parte del desarrollo cabe mencionar que los sistemas de información transforman las instituciones y cambia su estructura por lo que permite administrar, procesar datos en cualquier parte del mundo sin importar su plataforma para el procesamiento.

Por todo lo mencionado anteriormente es el caso de la empresa BARRA genera mucha información ya que comercializa gran cantidad de autopartes para diferentes marcas de vehículos.

Para el análisis de la aplicación web se usó la metodología UWE UML (UML-Based Web Engineering) para la construcción y el diseño. UWE es el proceso de desarrollo para la aplicación web, basadas de UML. Para evaluar la calidad del software se utilizó ISO 9126 y finalmente para la estimación del costo de producto se usó COCOMO II basado en kilo líneas de código.

La arquitectura MVC (Movimiento Vista Controlador) ayuda al desarrollo y a mantener separado los aspectos visuales de la lógica de negocios, PHP es el lenguaje de programación utilizando el framework Codeigniter y MariaDB como gestor de base de datos.

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

INDICE DE TABLAS

CAPITULO I

1. MARCO PRELIMINAR

1.1. INTRODUCCIÓN.

Hoy en día las aplicaciones tecnológicas e inteligentes juegan un papel importante en las empresas e instituciones. No es un lujo sino una necesidad apoyarse en herramientas tecnológicas con el software adecuado, para la toma de decisiones oportuna; procesos de planificación y administración de las empresas. Prácticamente todas las empresas buscan garantizar su éxito, limitar los riesgos, reducir costos y aumentar las ganancias. En este contexto, el vertiginoso avance de las telecomunicaciones y el progreso que han experimentado las ciencias informáticas obligan a las empresas a entrar al mundo de la tecnología, a ser competitivos y no quedarse relegados.

Actualmente la empresa de autopartes “Barra” de carácter unipersonal dedicada a la comercialización de autopartes en la ciudad de El Alto del departamento de La Paz, entre sus procesos más relevantes se encuentran, la adquisición y venta de sus productos. La empresa confronta con problemas en el control de estos debido a que el proceso de registro y consulta de los productos se realiza de forma semiautomática en archivos de Excel, provocando un esfuerzo por parte de los encargados de esta área porque cada vez se tarda más en la ubicación de un producto además de sus características básicas, también son susceptibles a ser alimentadas con datos erróneos y por ende pérdidas económicas para la empresa.

Uno de los medios más útiles y requeridos para poder acceder a la información son los sistemas web porque no requieren de un sistema operativo o plataforma específica, ya que pueden ser alojados en un servidor de internet o intranet (red local) y ser utilizados desde cualquier navegador web, sin importar el dispositivo. Por lo cual, se aprovechará esta tecnología para beneficiar tanto a los funcionarios que trabajan en ella y a la gerencia de la misma, que servirá de apoyo a la empresa brindándole una información precisa y oportuna.

El presente proyecto tiene por objeto desarrollar un sistema informático web de compras y ventas de autopartes que brinde la información oportuna para la toma de decisiones y evite la pérdida económica de la empresa BARRA. Para el desarrollo del sistema se utilizará: como lenguaje de programación PHP, como gestor de base de datos MariaDB, se aplicará la metodología UWE para el desarrollo del sistema y se aplicará la metodología de inventario PEP para el manejo de inventario.

1.2. ANTECEDENTES

1.2.1. Antecedentes Internacionales

[Fran Kevin, 2017], Propuesta tecnológica de un sistema de facturación de una microempresa de compra y venta de artículos de ferretería con enfoque electrónico es un sistema de facturación que está basado en un lenguaje de programación C# y SQL Server, realizado por PatsyBetzabeth Jiménez Maquilón y Fran Kevin SolisPeñaherrera en la universidad estatal de Guayaquil facultad de ciencias administrativas.

[Reyes, 2010], Diseño, Desarrollo e Implementación de un Sistema Eficiente de facturación Enfocado a los Requerimientos y Características de la PEUSB (Proveeduría Estudiantil de la Universidad Simón Bolívar), realizado por Mauricio Reyes de la Universidad Simón Bolívar Departamento de Ciencias de la Computación y Tecnología de la Información en marzo de 2010, el cual integra los procesos involucrados referentes al área de facturación, inventario y reportes además de ajustar las características particulares de PEUSB.

[Arguelles, 2014], Desarrollo de un sistema de información automatizado para el control del proceso de facturación de la empresa Ferre-Impercon C.A, realizado por Arguelles, Lissett Cristina en Maracaibo de la República Bolivariana de Venezuela en la Universidad Rafael Urdaneta Facultad de Ingeniería de Computación en septiembre 2014, cuyo sistema pretende automatizar el proceso de facturación de Ferre-Impercon, C:A aplicando la metodología xp para el cual se hace uso de aplicaciones Xammpp versión 1.7.7, NetBeans 8.0, Apache 2.2.21, MySQL 5.5.16 y JasperReport.

[Amaro, 2017], Sistema de Emisión de Comprobantes de Pago Electrónicos en el Procesos de Facturación de Contasis SAC, presentada por Amaro Quispe KennidyJordy de la Universidad Nacional del Centro del Perú, Huancayo-2017, de la Facultad de Ingeniería de Sistemas, para la construcción del sistema se aplicó la metodología Proceso Unificado Agil(AUP) combinada con la RUP y fue desarrollada con el lenguaje C# y SQL.

1.2.2. Antecedentes Nacionales

[Claudia Chiri Honorio, 2009], "Sistema de Entrada y Salidas e Inventarios en Bolital S.R.L" El objetivo principal es de desarrollar e implementar un sistema de entrada y salida de inventarios, de tal forma que se mejore los procesos operativos y administrativos. Realizado en la Universidad Mayor de San Andrés (U.M.S.A) La Paz.

[Alison Wendy Mendoza Rufino, 2009], el "Sistema de Información de Control de Inventario caso 'Emcotevis S.R.L.'" El objetivo principal es de desarrollar e implementar un sistema de información que brindara la atención de los clientes y dando a conocer información útil y confiable acerca de la atención y administración de la empresa. Realizada Universidad Técnica de Oruro.

1.3. PLANTEAMIENTO DEL PROBLEMA

1.3.1. Problema Principal

Actualmente la empresa de autopartes BARRA no tiene información actualizada de los productos que comercializa y por ello se suscitan problemas cotidianos como: Información respecto a entradas, salidas, actualización del stock, y características de los productos que comercializa no es confiable debido a que este es manejado de forma semi manual en archivos de Excel, afectando a la óptima toma de decisiones y como consecuencia pérdidas económicas para la empresa.

1.3.2. Problemas Secundarios

A continuación, se menciona los problemas que ocurren en la empresa de autopartes BARRA.

- Información desactualizada del registro de compras de los productos debido al proceso semi manual.
- Tardanza en concretar las ventas, generando molestias al cliente debido a que este es realizado de manera semi manual.
- No existe información confiable y oportuna respecto a los productos existentes y el stock de estos en almacén.
- Desconocimiento de registro de los clientes que adquieren productos de la empresa lo cual dificulta el lanzamiento de promociones a clientes fidedignos.
- Falta de la información completa de los proveedores causando tardanza en la reposición de productos.

1.3.3. Planteamiento del Problema

¿De qué manera el “Sistema Informático Web De Compras, Ventas Y Control De Stock De Autopartes puede ayudar a controlar las entradas, salidas y actualización del stock existente de productos que comercializa la empresa BARRA?

1.4. OBJETIVOS

1.4.1. Objetivo General

Desarrollar un sistema informático Web de compras, ventas y control de stock que permita controlar las entradas, salidas y actualización del stock existente de productos que comercializa la empresa de autopartes BARRA, para obtener información confiable y oportuna.

1.4.2. Objetivos Específicos

- Realizar un de formulario prediseñado para el registro de compra de productos.
- Diseñar un módulo de venta de productos para agilizar el proceso de venta.
- Efectuar un módulo de los productos existentes y el stock de estos en almacén de forma automática.
- Desplegar el registro de los clientes que adquieren productos de la empresa.
- Generar reportes de los proveedores con la información completa y de forma automática.

1.5. JUSTIFICACIÓN

Las justificaciones son desarrolladas de acuerdo a tres aspectos técnicos, económicos y sociales

1.5.1. Justificación Técnica

El proyecto se justifica técnicamente porque la empresa “Autopartes Barra” cuenta con los equipos computacionales necesarios capaces de soportar el sistema de gestión de información a desarrollar. Se emplearán herramientas actuales para el desarrollo del sistema como ser PHP y MariaDB herramientas que permitirán el acceso y manipulación de la información de forma fácil y segura.

Por otra parte, este sistema de gestión de información será multiplataforma y podrá ser accedido desde cualquier dispositivo electrónico este podría ser computadora de escritorio, computadora personal y dispositivos móviles.

1.5.2. Justificación Económica

El proyecto se justifica económicamente porque el sistema a implementar en la empresa tendrá información actualizada y oportuna del inventario de sus productos para una mejor toma de decisiones evitando así pérdidas económicas, además que la atención al cliente será más eficiente lo cual conlleva al incremento económico para la empresa.

1.5.3. Justificación Social

El sistema se justifica socialmente porque beneficiara al personal encargada del área de almacenes de la empresa facilitando las condiciones de trabajo, optimizando las tareas, reduciendo la probabilidad de errores que pueden presentarse, beneficiara también al personal de área de gerencia de la empresa brindando información rápida y precisa para una buena toma de decisiones.

1.6. METODOLOGÍAS Y TÉCNICAS

1.6.1. Método de Ingeniería

A. Metodología UWE

UWE (Uml-based Web Engineering) es una metodología basada en el Proceso Unificado y UML para el desarrollo de aplicaciones Web, cubre todo el siglo de vida de las aplicaciones Web. Su proceso de desarrollo se basa en tres fases principales: la fase de captura de requisitos, la fase de análisis y diseño y la fase de la implementación.

Características de la metodología UWE

La metodología UWE define vistas especiales representadas gráficamente por diagramas es UML, tales como el modelo de navegación y el modelo de presentación. (Nieves, Ucán y Menéndez, 2014).

Un perfil de UML consiste en una jerarquía de estereotipos y un conjunto restricciones, Los estereotipos son utilizados para representar instancias de las clases. Las ventajas de utilizar los perfiles de UML, es que casi todas las herramientas CASE de UML los reconocen. Los modelos deben ser fácilmente adaptables al cambio en cualquier etapa de desarrollo.

Otras características relevantes del proceso y método UWE son el uso del paradigma orientado a objetos, su orientación al usuario, la definición de un meta-modelo (modelo de referencia) que da soporte al método y el grado de formalismo que alcanza debido al soporte que proporciona para la definición de restricciones sobre los modelos.

Principales aspectos de la metodología UWE

Los principales de aspectos en los que se fundamenta UWE son los siguientes:

Uso de una notación estándar, para todos los modelos (UML: Lenguaje de modelado unificado).

Definición de métodos: Definición de los pasos para la construcción de los diferentes modelos.

Especificación de Restricciones: Se recomienda el uso de restricciones escritas (OCL: Lenguaje de restricciones de objetos) para aumentar la exactitud de los modelos.

Fases del Desarrollo Web

UWE hace un uso exclusivo de estándares reconocidos como UML y el lenguaje de especificación de restricciones asociado OCL. Para simplificar la captura de las necesidades de las aplicaciones web, UWE propone una extensión que se utiliza a lo largo del proceso de autoría. Este proceso de autoría está dividido en cuatro pasos o actividades:

Análisis de Requisitos: Fija los requisitos funcionales de la aplicación Web para reflejarlos en un modelo de casos de uso.

Diseño Conceptual: Materializado en un modelo de dominio, considerando los requisitos reflejados en los casos de uso.

Diseño Navegacional: se puede sub dividir en:

Diseño de Presentación: Representa las vistas del interfaz del usuario mediante Modelo del Espacio de Navegacional.

Modelo de la Estructura de navegación: Muestra la forma de navegar ante el espacio de navegación.

Diseño de Presentación: Representa las vistas del interfaz del usuario mediante modelos estándares de interacción UML.

B. Métricas de calidad

Para la verificación de la calidad del software que es muy importante dentro de la metodología utilizada UWE, se llegara a verificar los siguientes puntos con los parámetros de medición de la ISO-IEC9126.

Funcionalidad: Grado en el que el software satisface las necesidades planteadas según las establece los atributos siguientes: adaptabilidad, exactitud, interoperabilidad, cumplimiento y seguridad.

Confiabilidad: cantidad de tiempo que el software se encuentra disponible para su uso, según lo indican los siguientes atributos, madurez, tolerancia a fallas y recuperación.

Usabilidad: grado en el que el software es fácil de usar, según lo indican los siguientes sub atributos entendible y operable.

Eficiencia: grado en que el software emplea óptimamente los recursos del sistema, según lo indicado.

Facilidad de recibir mantenimiento: facilidad con la que pueden efectuarse reparaciones al software según lo indican los atributos analizables, cambiable, estable, susceptible de someterse a pruebas.

Portabilidad: debido a la dinamicidad del entorno tecnológico, a menudo es necesario implementar una misma aplicación en distintas plataformas, distintas arquitecturas, distintas tecnologías y/o atendiendo a distintos dispositivos de acceso, lo que obliga a desarrollar técnicas, modelos y herramientas que faciliten la reutilización e independiza hasta donde es posible el desarrollo de la aplicación y facilidad con la que el software puede llevarse de un ambiente a otro.

C. Método de estimación de costos

El modelo de construcción de Costes (COCOMO) es un modelo matemático de base empírica. Incluye tres sub modelos, cada uno frece un nivel de detalle y aproximación, cada vez mayor, a medida que avanza el proceso de desarrollo del software: básico, intermedio y detallado.

Para la estimación de costos se utiliza el método COCOMO II por ser un modelo constructivo de costos, está compuesto por tres modelos denominados composición de aplicación, diseño temprano, y post-arquitectura, los tres modelos de COCOMO II se adapta tanto a las necesidades de los diferentes sectores descritos, como al tipo y cantidad de información.

D. Métodos de Prueba de Software

El método de pruebas caja blanca y de caja negra para realizar las pruebas del software. Cuando una aplicación es probada desde dentro, usando su lógica aplicativa y cuando una aplicación es probada usando su interfaz externa, generalmente la GUI (Katz-Lichtenstein, 2003).

1.6.2. Técnicas

1.6.2.1. Técnicas de recopilación de datos

- Entrevistas; La entrevista es una conversación dirigida, con un propósito específico y que usa un formato de preguntas y respuestas.

- Cuestionarios; Una encuesta es un conjunto de preguntas normalizadas dirigidas a una muestra representativa de la población o instituciones, con el fin de conocer estados de opinión o hechos específicos.

- Observación directa; La observación es otra técnica útil para el analista en su proceso de investigación, consiste en observar a las personas cuando efectúan su trabajo.

1.6.2.2. Técnicas de seguridad

El uso de Password; Permiten crear contraseñas y se debe usar un usuario y contraseña distinta para cada acceso y cámbialas cada 3 meses.

Copias de Seguridad; Respaldar los archivos en programas como Dropbox o Google Drive cuando sea necesario así tendrás una copia de archivos en caso de existir fallos irreparables en tu sistema operativo o disco duro.

Encriptación de datos con el método SHA256; función que transforma un mensaje en una serie ilegible aparentemente aleatoria, usando una clave de encriptación que puede ser revertida (es decir, obtener el mensaje original) sólo por quienes conocen dicha clave. Para garantizar la seguridad del sistema.

1.7. HERRAMIENTAS

PHP (Hypertext Pre-processor): Lenguaje de código abierto muy popular especialmente adecuado para el desarrollo Web y que puede ser incrustado en HTML.

MariaDB: Sistema de gestión de base de datos derivado de MySQL con licencia GPL. Es desarrollado por Michael Widenios fundador de MySQL, la fundación MariaDB y la comunidad de desarrolladores de software libre.

Servidor Web Apache: Un servidor Web es un servidor que se encarga de aceptar las solicitudes HTTP desde clientes web y que les presten servicio HTTP respuestas, generalmente en forma de páginas web contiene estático (texto imágenes, etc) y dinámico (guiones) de contenido.

Framework Codeigniter: El diseño orientado al rendimiento de este framework de desarrollo web se revela en su parca arquitectura, pues se basa en el patrón Modelo-Vista-Controlador (MVC). El principio fundamental que sustenta a la arquitectura de desarrollo MVC es la estricta separación entre el código y la presentación, gracias a una estructura modular de software y a la externalización del código PHP.

Bootstrap. Es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como extensiones de JavaScript adicionales..

Html (HiperText Markup Language): Es un lenguaje de marcado que se utiliza para el desarrollo de páginas de internet. Se trata de la sigla que corresponde a, que podría ser traducido como Lenguaje de Formato de Documentos para Hipertexto.

Css: Lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado de escrito en un lenguaje de marcado css, corresponde a la expresión inglesa Cascading Style Sheets que se traduce en “Hojas de estilo en cascada”.

JavaScript: Es un lenguaje de programación interpretado, dialecto del estándar ECMASScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

1.8. LIMITES Y ALCANCES

El presente proyecto pretende analizar la situación actual de las operaciones dentro de la empresa, la relación con otras áreas dentro la empresa BARRA.
De ahí que se identifican los siguientes límites y alcances:

1.8.1. Limites

El presente proyecto se limita a realizar:

- El sistema no realizara el vínculo de facturación con impuestos nacionales.
- El sistema no realizara operaciones de pagos al personal.
- El sistema será de uso exclusivo de los funcionarios de la empresa.

1.8.2. Alcances

El presente proyecto tiene los siguientes alcances:

- Módulo de registro nuevos productos, modificación, eliminación y actualización.
- Módulo de registro de nuevos compras, modificación y eliminación.
- Módulo de registro de nuevas ventas, modificación y eliminación.
- Módulo de registro de nuevos clientes, modificación y eliminación.

- El listado detallado información del proveedor y los productos que provee.

- Módulo de presentación de reportes semanales, mensuales y anuales.

1.9. APORTES

Dado el gran avance tecnológico que esta dado hoy en día es necesario que las instituciones, empresas y la sociedad en común puedan conocer y beneficiarse de este. Con el presente proyecto se pretende dar los siguientes aportes.

- Se creará un entorno de trabajo sistematizado y actualizado de los productos que, comercializa la empresa.

- Utilización de las nuevas tecnologías que están surgiendo día a día.

- Obtención de información confiable y segura de cada producto, sus características, y sus precios.

- Permitirá tener información centralizada en una sola base de datos.

- Facilitar reportes que ayuden a una mejor toma de decisiones.

- Agilizar los procesos manuales de actualización de productos dentro de la empresa, mejorando así el tiempo de atención al cliente.

CAPÍTULO II

2. MARCO TEÓRICO

2.1. ANTECEDENTES INSTITUCIONALES

La empresa Barra se dedica a la a la comercialización de partes, piezas y accesorios de vehículos livianos. Tiene una trayectoria en el mercado desde el 2008 en la ciudad de El Alto del departamento de La Paz. Actualmente está constituida como empresa unipersonal, y sigue siendo de tipo familiar, pues las personas que ahí laboran son sus hijos, sobrinos, etc. Y la gerencia sigue siendo ejecutada por el Sr. Mario Barra. La empresa comercializa piezas y accesorios de vehículos livianos en las marcas hilux, hiace, Isuzu, Suzuki, Toyota, etc. Cabe mencionar también que las adquisiciones de estos son a nivel local.

El mercado al que se dirigen los productos de la empresa son clientes particulares, cooperativas de transporte, mecánicas en general.

2.1.1. Misión

Comercializar autopartes de alta calidad para vehículos livianos, así como la prestación de servicio al cliente, satisfaciendo sus principales necesidades y expectativas de manera eficiente.

2.1.2. Visión

Ser una empresa líder en la comercialización de autopartes para vehículos livianos, ofreciendo un amplio portafolio de productos con precios competitivo y productos de alta calidad para cumplir con los estándares exigidos por el mercado.

Las metodologías, métodos modelos, técnicas y herramientas que se utilizan en el presente proyecto se detallan en este capítulo.

2.2. SISTEMA

Un sistema es un conjunto de elementos organizados que se encuentran organizados en iteración, que buscan alguna meta o metas comunes operando para ello sobre datos o información sobre energía o materia u organismo sobre una referencia temporal para producir como salida de información, materia, energía u organismos, esta definición es aplicable a todos los tipos de sistemas de información, incluyendo a los sistemas de información gerencial. (Beekman G. , 1999)

Un sistema es módulo ordenado de elementos que se encuentran interrelacionados y que interactúan entre sí. El concepto se utiliza tanto para definir a un conjunto de conceptos como a objetos reales dotados de organización también puede mencionarse la noción del informático, muy común en las sociedades modernas. Este tipo de sistemas denominan al conjunto de hardware, software y soporte humano que forman parte de una empresa u organización. Incluyen ordenadores con los programas necesarios para procesar datos y las personas encargadas de su manejo. (Rojas Daniela, 2018).

Sistema; es un conjunto de elementos (software, hardware y recurso humano) relacionados entre sí que persiguen un objetivo en común, cada elemento desempeña funciones específicas.

2.3. SISTEMA DE INFORMACIÓN

“Sistema de información es aquel conjunto de componentes interrelacionados que capturan, almacenan, procesan y distribuyen la información para apoyar la toma de decisiones, el control, análisis y visión de una organización”. pues establece cuáles son las necesidades de información de las empresas, cómo las va a solucionar y qué medios (tecnologías de información) va a emplear. (K y J Laudon, 1996),

Un sistema de información es conjunto de elementos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio. Teniendo muy en cuenta el equipo computacional necesario para que el sistema de información pueda operar y el recurso humano que interactúa con el Sistema de Información, el cual está formado por las personas que utilizan el sistema. (Peralta, 2008).

Un sistema de información es un conjunto de elementos (software, hardware y recursos humanos) que interactúan entre sí con la finalidad de apoyar las actividades de una organización.

2.4. SISTEMA DE INFORMACIÓN WEB

Un sistema de información que va más allá de un simple servidor informático, que va más allá de un simple servidor informático, ya que interpretan a los sistemas web como un depósito documental que funciona a modo de memoria documental, como tareas que superan la mera publicación para convertirse en un sistema de organización de la información. (García, 1997).

Conjunto formal de procesos que, operando sobre una colección de datos estructurada de acuerdo a las necesidades de la empresa, recopila, elabora y distribuyen selectivamente la información necesaria para la operación de dicha empresa y para las actividades de dirección y control correspondientes, apoyando, al menos en parte, los procesos de toma de decisiones necesarios para desempeñar funciones de negocio de la empresa de acuerdo con su estrategia. (Andreu, Ricart , 1991).

Un sistema informático se puntualiza como un sistema de información fundamentado en la utilización de la computación. En este sentido, el hardware, software y recursos humanos se interrelacionan entre sí. Su finalidad es brindar soporte al procesamiento, almacenamiento, y entrada-salida de datos (Moreno y Santos, 2012)

2.5. COMPRA

Una compra es adquirir bienes y servicios de la calidad adecuada, en el momento y al precio adecuado y del proveedor más apropiado.
Dentro del concepto de la empresa moderna las compras se deben manejar por un departamento especializado que debe formar parte de la propia organización de la compañía. (Limusa, 2006).

Adquirir productos y servicios en la cantidad, calidad, precio, momento, sitio y proveedor justo o adecuado. Buscando la máxima rentabilidad de la empresa. (Alberto Montoya Palacio, Conceptos modernos de compras).

Es la actividad que incluye el conocimiento de la necesidad, localización y selección del suministrador, negociación con el establecimiento de precio y términos, seguimiento para el aseguramiento de la entrega.

2.6. VENTA

Es un contrato en el que el vendedor se obliga a transmitir una cosa o un derecho al comprador, a cambio de una determinada cantidad de dinero". También incluye en su definición, que "la venta puede considerarse como un proceso personal o impersonal mediante el cual, el vendedor pretende influir en el comprador. (Diccionario de Marketing de Cultural S.A.).

Una venta promueve un intercambio de productos y/o servicios. La venta puede ser:

1) al contado, cuando se paga la mercancía en el momento de tomarla, 2) a crédito, cuando el precio se paga con posterioridad a la adquisición y 3) a plazos, cuando el pago se fracciona en varias entregas sucesivas". (Ricardo Romero, autor del libro "Marketing").

Venta es la cesión de una mercancía mediante un precio convenido. La venta puede ser: 1) Al contado, cuando se paga la mercancía en el momento de tomarla 2) A crédito, cuando el precio se paga con posterioridad a la adquisición 3) A plazos, cuando el pago se fracciona en varias entregas sucesivas (Romero, 2006).

2.7. CONTROL

Control es el proceso de verificar para determinar si se están cumpliendo los planes o no, si existe un progreso hacia los objetivos y metas. El control es necesario para corregir cualquier desviación. (Theo Haimann, 2014).

El control es el proceso de verificar el desempeño de distintas áreas o funciones de una organización. Usualmente implica una comparación entre un rendimiento observado, para verificar si están cumpliendo los objetivos de forma eficiente y eficaz y tomar acciones correctivas cuando sea necesario. La función de control se relaciona con la función de planificación, porque el control busca que el desempeño se ajuste a los planes.

El proceso administrativo, desde el punto de vista tradicional, es un proceso circular que se retroalimenta. Es por ello que durante la gestión el control permite tomar medidas correctivas. (Concepto de Control, 2018).

Control; herramienta de la administración que nos permite verificación de que las acciones que se están ejecutando nos lleve a lo planeado y si es necesario implementar correcciones para asegurar el cumplimiento de los objetivos de la organización.

2.8. STOCK

Se considera stock aquella cantidad de un producto que se encuentra acumulada en un lugar determinado, fija o bien en movimiento hacia sus centros de distribución. Su función es la de servir de instrumento de regulación de toda la cadena logística, con el fin de conseguir un flujo de materiales continuo. (Portal Carlos, 2013, p. 27)

Stock es todo aquel bien que se almacena para ser posteriormente vendido o usado en el proceso productivo. Normalmente, el stock es asociado a un almacén (de ahí que se importante saber gestionarlo) aunque no siempre sucede: puede que existan existencias en el tiempo en que se desplaza a un cliente. (Acevedo Suarez, 2006).

Stock; Conocida también como existencias, es la cantidad de mercaderías, herramientas o materia prima que permanece almacenada como repuesto, para compensar a las que están en uso o destinadas a la venta, si es necesario reponerlas. Por ejemplo, pueden tenerse en stock, máquinas por si alguna se rompe y no parar la producción hasta que sean reparadas, cajas para empaquetar, o mercaderías destinadas a la venta si crece la demanda y tarda la reposición, y no perder así la posibilidad de comercializar el producto por no contar con la cantidad suficiente.

2.9. CONTROL DE STOCK

El control de los stocks o las existencias de una organización es el control de aquellos activos poseídos por la empresa para ser vendidos en el curso normal de su actividad ordinaria, o para ser incorporados o transformados en el proceso productivo que la misma desarrolla. (Bosch, 2013).

El control de stock es una herramienta fundamental en la administración moderna, ya que esta permite a las empresas y organizaciones conocer las cantidades existente de productos disponibles para la venta, en un lugar y tiempo determinado, así como las condiciones de almacenamiento aplicables en las industrias. (Orlando Espinoza, 2011).

El control de stock es una herramienta fundamental para la gestión logística, permite conocer con exactitud las cantidades existentes por artículo, lotes, vencimientos y cualquier otra observación que su juicio personal determine que debe ser notificada.
sobre todo, si queremos evitar un exceso de costos, tener el cálculo del stock (movimientos de entrada y salida de mercadería) para saber cuál es el mínimo que tenemos que tener almacenado para no sufrir desabastecimiento, o lo que es lo mismo, evitar una rotura de stock.

2.10. AUTOPARTES

Autoparte es cualquier pieza o componente que se monte en un automóvil. El sector de autopartes abarca diferentes líneas de producción como: carrocerías y remolques, motores y sus partes, sistemas de dirección, sistemas de suspensión, sistemas de frenado y componentes. (Manuel Markel Guevara, 2018).

Se les llama autopartes a todas aquellas piezas que en su conjunto intervienen para el armado de un automóvil, las cuales son visibles tanto para el usuario como para las personas. Estas piezas vienen con el auto o pueden añadirse cuando sea necesario (o quieran implementarse en el diseño original) y se pueden adquirir por separado sin ningún problema. En otras palabras, son los elementos que conforman la carrocería o acabados de un vehículo que involucran su aspecto, entre las piezas que entran en esta categoría se encuentran:

- ✓ Faros
- ✓ Espejos
- ✓ Puertas
- ✓ Toldo
- ✓ Asientos
- ✓ Acabados (tablero, seguros, manijas)

2.11. METODOLOGÍA UWE

UWE (UML Web Engineering, en español ingeniería Web basada en UML) es una metodología que permite especificar de mejor manera una aplicación Web, para el proceso de creación de aplicaciones, con una gran cantidad de especificaciones, en el proceso de diseño lista que debe utilizarse. Procede de manera iterativa e incremental, coincidiendo con UML, incluyendo flujos de trabajo y puntos de control.

UWE es una metodología basada en Proceso Unificado y UML para el desarrollo de aplicaciones Web, cubre todo el ciclo de vida de las aplicaciones Web. Su proceso de desarrollo se divide en tres fases principales: la fase de capturas de requisitos la fase de análisis y diseño y la fase de implementación.

2.11.1. Características

Las principales características de la metodología UWE son las siguientes:

- ✓ Uso de una notación estándar: el uso del lenguaje de modelado UML.
- ✓ Definición de métodos: presenta definición de los pasos para la construcción de los diferentes modelos.
- ✓ Especificación de restricciones: uso de restricciones estrictas (OCL: Lenguaje de restricciones de objetos) para aumentar la exactitud de los modelos.

UWE es una metodología dirigida o enfocada al modelado de aplicaciones Web, ya que está basada estrictamente en UML, esta metodología no garantiza que sus modelos sean fáciles de entender. En la siguiente figura podemos ver una vista general de UWE.

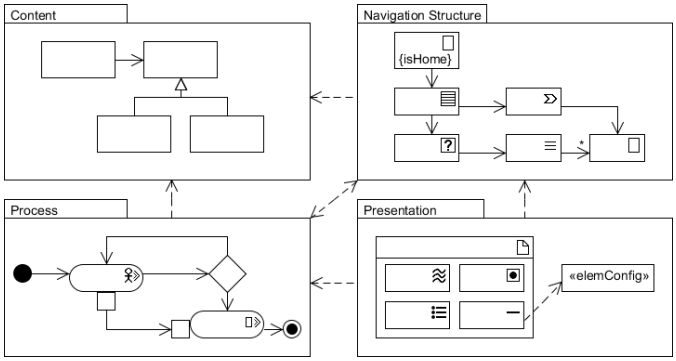


Figura 2. 1. Vista General de Modelos UWE
Fuente: Nolivos y Coronel, (T-ESPE, 2013)

2.11.2. Modelo de Análisis de Requisitos

Se adquiere adquieren, reúnen y especifican las características funcionales y no funcionales que deberá cumplir la aplicación web.
Trata de diferente forma las necesidades de información, las necesidades de navegación, las necesidades de adaptación y las de interfaz de usuario, así como algunos requisitos adicionales

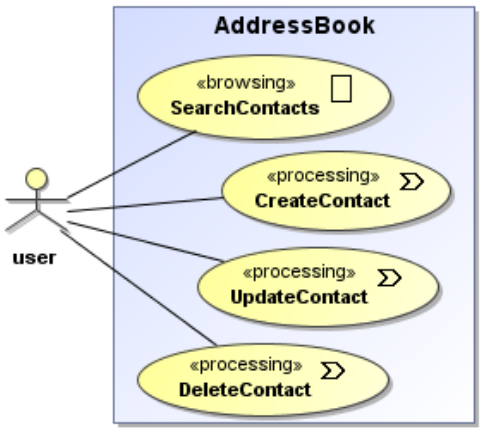


Figura 2. 2. Modelo de Caso de Uso
Fuente: Ludwig-Maximilians (UWE, 2014)

2.11.3. Modelo conceptual

Caracterizado por un modelo de dominio, que utiliza los requisitos que se detallan en los casos de uso. En esta etapa se representa el dominio del problema con un diagrama de clases de UML, que permite determinar, métodos y atributos. El propósito de este diagrama es construir un modelo del dominio que intenta no considerar el paseo de la navegación, la presentación y los aspectos de interacción. Aspectos que se analizaran en los pasos respectivos de navegación y presentación de la planificación.

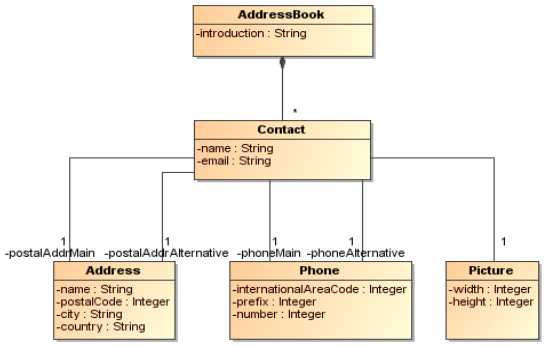


Figura 2. 3. Diagrama de Contenido
Fuente: Ludwig-Maximilians (UWE, 2014).

2.11.4. Modelo de navegación

Durante esta etapa se realizan las tareas que comúnmente se conocen como programación; que consiste, especialmente en llevar a código fuente, en el lenguaje de programación elegido, todo lo diseñado en la fase anterior.
Basado en el diagrama de la fase anterior, donde se especifica los objetivos que serán visitados dentro de la aplicación web y la relación entre los mimos. Su objetivo principal es representar el diseño y estructura de las rutas de navegación al usuario para evitar la desorientación en el proceso de navegación.

Este modelo se destaca en el marco de UWE como el más importante, ya que representa elementos estáticos, a la vez que se pueden incorporar lineamiento semántico de referencia para las funcionalidades dinámicas de una aplicación Web.

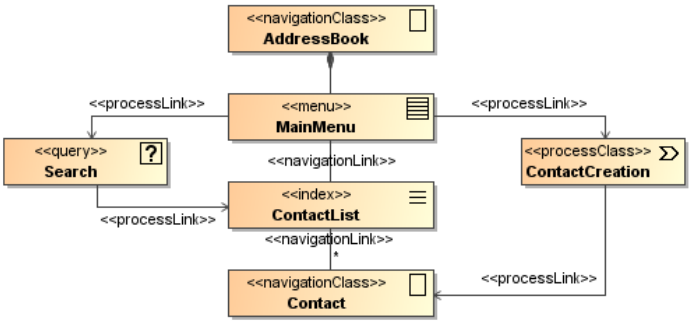


Figura 2. 4. Diagrama de navegación UML
Fuente: Ludwig-Maximilians (UWE, 2014)

El modelo de navegación a su vez podemos dividirlo en dos áreas:

- **Modelo del espacio de navegación:** basada en lo estructurado en la fase de conceptualización, es decir en los diagramas de clases.

- **Modelo de la estructura de navegación:** Muestra la forma de navegar entre el espacio de navegación. Están constituidas por menús, índices, vistas guiadas, y formularios.

  - ✓ **Los índices** es la colección de objetos permitiendo una navegación directa.
  - ✓ **Las visitas guiadas** compuesta por grupo de referencias, permitiendo una navegación secuencial.
  - ✓ **Un menú** es un elemento parte de la navegación con un numero especifico de conexiones a otros objetos.
  - ✓ **Un formulario** facilita al usuario ingresar información para completar las condiciones de selección de objetos pertenecientes a las colecciones de índices y visitas guiadas.

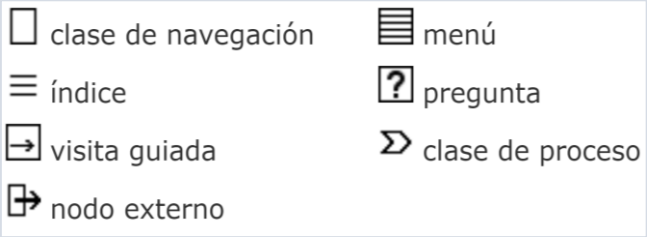


Figura 2. 5. Símbolo de estereotipos-Modelo de Navegación
Fuente: Nolivos y Coronel, (T-ESPE, 2013)

2.11.5. Diseño de Presentación

La presentación del diseño de modelo tiene como objetivo la representación de las vistas del interfaz del usuario final, la representación gráfica de esta fase se encuentra basada en los diagramas realizados en las fases anteriores.
Las clases del modelo de presentación representan páginas Web o parte de ellas, organizando la composición de los elementos de la interfaz de usuario y las jerarquías del modelo de presentación.

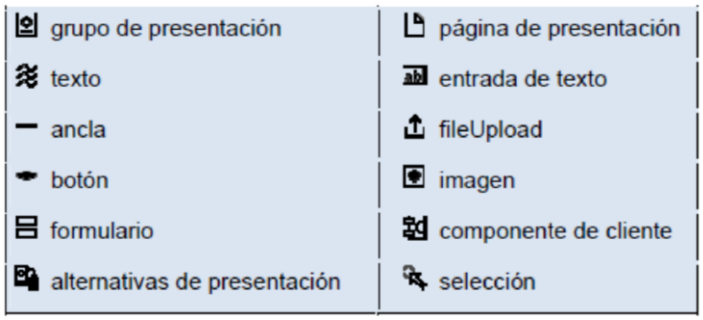


Figura 2. 6. Nombre y símbolo de estereotipos-Modelo de Presentación
Fuente: Nolivos y Coronel, (T-ESPE, 2013)

El diagrama de esta fase representa los objetos de navegación y elementos de acceso, por ejemplo, en que marco o ventana se encuentra el contenido y que será remplazado cuando se accione en enlace. En la siguiente imagen podemos observar un ejemplo de diagrama de presentación mediante UWE.

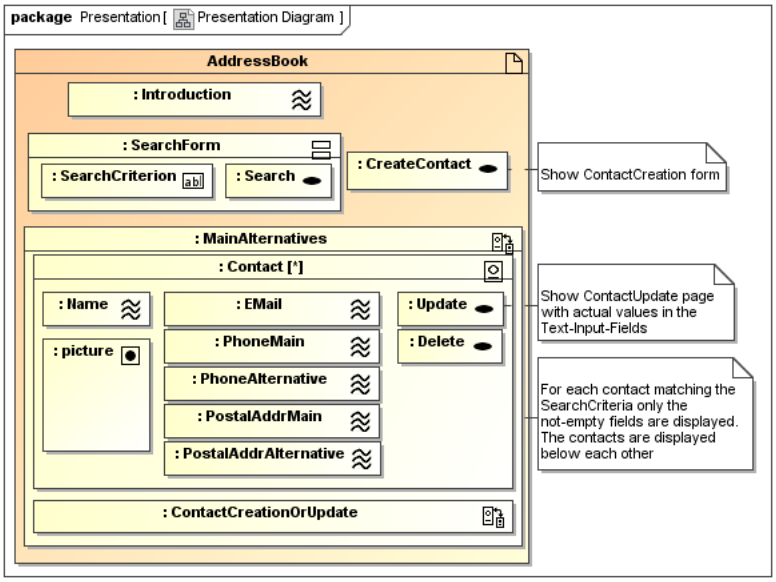


Figura 2. 7. Diseño de Presentación UWE
Fuente: Ludwig-Maximilians (UWE, 2014)

2.11.6. Modelo de procesos

Los modelos de procesos o tareas integran los procesos de negocios al modelo UWE, especificando los comportamientos de cada proceso de las interfaces que permiten manejar a cada uno de ellos. Representa la parte dinámica de la aplicación Web, especificando la funcionalidad de las transacciones y de los flujos de trabajos complejos de las actividades, contrario al modelo de navegación, que representa la parte estática de la información (T-ESPE,2013).

2.11.7. Fases de UWE

UWE cubre todo el ciclo de vida del software centrando su atención en aplicaciones personalizadas o adaptativas. Siendo las fases o etapas a utilizar las siguientes:

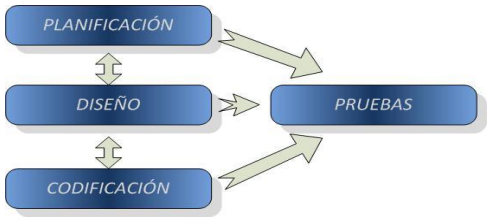


Figura 2. 8. Fase de la metodología UWE
Fuente: (koch,2015)

1) Análisis de Requerimientos
2) Diseño del Sistema
3) Codificación del software
4) Pruebas
5) Instalación o fase de la implementación
6) Mantenimiento.

2.11.7.1. Fase Análisis de Requerimientos

La fase de Análisis de Requerimientos realiza la captura de los mismos mediante diagramas de casos de uso acompañado de documentación que detalla.

2.11.7.2. Fase Diseño del Sistema

Se basa en la especificación de requisitos producido por el análisis de los requerimientos (fase de análisis), el diseño define como estos requisitos se cumplirán, la estructura que debe darse a la aplicación web.

2.11.7.3. Fase Codificación del Software

Durante esta etapa se realizan las tareas que comúnmente se conocen como programación; que consiste, especialmente, en llevar a código fuente, en el lenguaje de programación elegido en la fase anterior.

2.11.7.4. Fase Pruebas

Las pruebas se utilizan para asegurar el correcto funcionamiento de secciones de código.

2.11.7.5. Fase Instalación o fase de la Implementación

Es el proceso por el cual los programas desarrollados son transferidos apropiadamente al computador destino, inicializados, y eventualmente configurados; todo ello con el propósito de ser ya utilizados por el usuario final.

2.11.7.6. Fase Mantenimiento

Es el proceso de control, mejora y optimización del software ya desarrollado e instalado, que también incluye depuración de errores y defectos que puedan haberse filtrado de la fase de pruebas de control.

2.11.7.7. Ventajas y Desventajas de la metodología UWE

Las principales razones para el uso de los mecanismos de extensión de UML en lugar de una técnica de modelado de propiedad es la aceptación del UML en el desarrollo de sistemas de software, de la flexibilidad para la definición de un lenguaje de modelado especifico de dominio Web: el llamado perfil UML, y amplio apoyo de modelado visual por herramientas CASE UML existentes.

UWE utiliza “puro” notación UML y tipos de diagramas UML siempre que sea posible para el análisis y diseño de aplicaciones Web, es decir, sin las extensiones de cualquier tipo. Por las características Web, como nodos y enlaces de la estructura de hipertexto, el perfil UWE incluye estereotipos, valores etiquetados y restricciones definidas para los elementos de modelado.

UWE permite un modelado de aplicaciones Web basado en las demasiadas de cada usuario en particular, separando requerimientos, enfoques, interfaces, adaptabilidad, aspectos y componentes para mayor flexibilidad, definiendo un conjunto de procesos adecuados durante todas las etapas del desarrollo, cual permite mantener la integridad del diseño y la funcionalidad del sistema.

2.12. INGENIERÍA WEB

Ingeniería web es una disciplina que hace uso de principios científicos, de ingeniería y de gestión con un enfoque sistemático con el objetivo de desarrollar, desplegar con éxito el mantenimiento de alta calidad de los sistemas basados en la Web y aplicaciones. (Roger S. Pressman, 2010).

La ingeniería web se debe al crecimiento desenfrenado que está teniendo la Web está ocasionando un impacto en la sociedad y el nuevo manejo que se le está dando a la información en las diferentes áreas en que se presenta ha hecho que las personas tiendan a realizar todas sus actividades por esta vía.

2.12.1. Características

La principal característica que los distingue (aplicaciones de sitio web) es que los sitios web son sitios en la web en donde se publica contenido generalmente estático o un muy bajo nivel de interactividad con el usuario, mientras que las aplicaciones son lugares con alto contenido de interactividad y funcionalidades que bien podrían ser un software comercial.

Por lo tanto, para lograr un mayor éxito en el desarrollo de aplicaciones Web complejas y gran escala es necesario enfocarse en la ingeniería Web como disciplina y las siguientes actividades forman parte del proceso de la ingeniería Web y que son aplicables a cualquier aplicación Web independiente del tamaño de complejidad. (Roger S. Pressman, 2010).

1. La Formulación identifica objetivos y establece el alcance de la primera entrega.

2. La **Planificación** genera estimación de costo, la evaluación de riesgos y el calendario del desarrollo y fechas de entrega.

3. El **Análisis** especifica los requerimientos e identifica el contenido.

4. La **Modelización** consta de dos partes:

    - Diseño y producción del contenido.

    - Diseño de la arquitectura, navegación e interfaz del usuario.

5. En la **Generación de Páginas** se integran arquitectura, navegación e interfaz para la creación más visible del proyecto, que son las páginas.

6. El **Test** son pruebas en busca de errores en todos los niveles: contenido, funcional, navegación, etc.

7. El resultado final es sometido a **Evaluación del Cliente**.

2.12.2. Etapas

El desarrollo web exige adaptación, estrategias y cambios continuos. Los usuarios se preocupan por una aplicación Web sea entregada cuando lo necesitan y no sobre el trabajo que se lleva a cabo para crear una aplicación Web sea entregada cuando lo necesitan y no sobre el trabajo que se lleva a cabo para crear una aplicación, por lo tanto, el equipo de desarrollo de un proyecto Web debe enfatizar la agilidad.

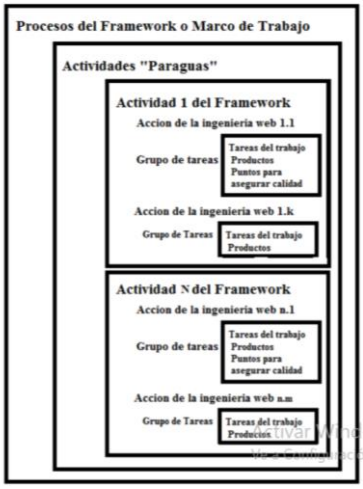


Figura 2. 9. Procesos de la Ingeniería de Web
Fuente: Roger S. Pressman, 2010).

Cada acción de Ingeniería Web está representada por un conjunto de tareas, cada una de una colección de tareas de trabajo de Ingeniería Web, productos de trabajo relacionado, relacionados, puntos de garantía de calidad e hitos del proyecto.

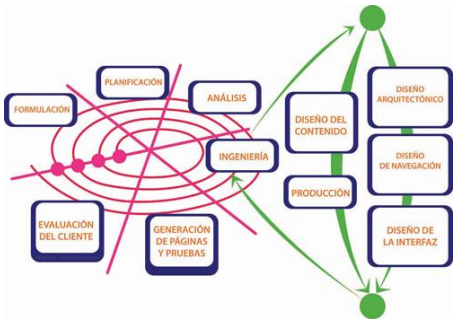


Figura 2. 10. Proceso de Ingeniería Web
Fuente: (Pressman, 2012)

Las siguientes actividades de la Ingeniería Web son partes de un Framework genérico y son aplicables a la gran mayoría de proyectos de aplicación web:

- **Comunicación:** la interacción y colaboración con el cliente

- **Planeamiento:** un plan incremental para que la ingeniería web produzca resultados.

- **Modelado:** Abarca la creación de modelos que asisten a los desarrolladores y clientes a entender los requerimientos de la aplicación web y como se van a lograr esos requerimientos.

- **Construcción:** combina el uso de las tecnologías web y las pruebas que serán usadas para descubrir errores en el código.

- **Implementación:** entrega una App web incremental para que el cliente lo evalué y proporcione retroalimentación del mismo.

2.12.3. Ventajas y desventajas

Ventajas:

- **Soporte:** una aplicación web reside en una red y debe de dar servicio a una comunidad diversa de clientes.

- **Inmediatez:** se refiere al corto tiempo que normalmente tienen los proyectos web para terminar, o por lo menos, lanzar una versión oficial.

- **Evolución:** las aplicaciones web y sitios web están en constante evolución, y se actualizan gradualmente.

- **Medible:** se puede conocer la cantidad de usuarios que visitan el sitio web, así como los patrones de comportamiento.

Desventajas:

- **Seguridad:** Dado que no controlamos quien pueda acceder a nuestro sitio, la seguridad y confidencialidad de la información requiere de un énfasis especial.

- **Estética:** Nuestro sitio debe ser ergonómico, atractivo y usable para que sea más agradable para el usuario.

La web es un campo que crese día con día y a un paso acelerado es por eso que se decidió hacer una metodología especial para esta clase de proyectos, esta metodología brinda soporte ya que se presenta en corto tiempo, que normalmente tiene los proyectos web para terminar, o por lo menos, lanzar una versión oficial.

2.13. INGENIERÍA DE SOFTWARE

La ingeniería de software es el establecimiento y uso de sólidos principios de ingeniería y buenas prácticas de gestión, así como la evolución de herramientas y métodos aplicables y su uso cuando sea apropiado para obtener, dentro de las limitaciones de recursos existentes, software que sea de alta calidad en un sentido explícitamente definido. (Bauer, 1972).

Se debe señalar, que el desarrollo del software va unido a lo que se conoce en el campo del software “ciclo de vida del software” que consiste en cuatro etapas que se conocen como:

- **La concepción** determina la repercusión del proyecto y diseña el modelo de negocio.

- **La elaboración** precisa la planificación del proyecto, especificando las características y apoya la arquitectura.

- **La construcción** es la elaboración del producto.

- **La transición** es la entrega del producto terminado a los usuarios.

2.13.1. Etapas de la ingeniería de software

La ingeniería de Software consta de siete etapas. A continuación, cada una de ellas brevemente definidas:

- **Etapa de análisis:** Es el procedimiento de investigación de un problema al que se desea encontrar la solución. Se define con claridad el Problema que hay que resolver o el programa que se desea inventar, identificando los elementos principales que conformarán el producto.

- **Etapa de Diseño:** Es el procedimiento que emplea la información acumulada en la etapa de análisis al diseño del producto. La labor principal de la etapa de diseño es crear un modelo o las características precisas para el producto o Componentes del Sistema.

- **Etapa de Desarrollo:** Consiste en el empleo de los diseños creados durante la etapa de diseño para elaborar los elementos a utilizarse en el sistema.

- **Etapa de Pruebas o Verificación Prueba:** Consiste en garantizar que los elementos individuales que componen el sistema o producto, presentan las características requeridas en la especificación creada durante la etapa de diseño.

- **Etapa de Implementación o Entrega Implantación:** Consiste en la distribución del producto y hacerlo llegar a manos del cliente.

- **Etapa de Mantenimiento:** Consiste en aplicar las soluciones apropiadas a cualquier problema del producto y re- liberar el producto mejorado, dándole una nueva versión.

- **Etapa final EOL (End-of-Life)**: Consiste en ejecutar todas las labores que garanticen que tanto los clientes como los empleados tiene la certeza de que el producto ya no estará más a la disposición, por lo que no se venderá más.

2.13.2. Modelos de Proceso de Software

Los Modelos de Proceso de Software se los puede definir como una descripción simplificada de un proceso del software que presenta una visión de ese proceso. Los modelos de proceso de software, pueden incluir actividades que son parte de los procesos, productos de software y el papel de las personas involucradas en la ingeniería de del software. (Pressman, 2010).

La mayor parte de los modelos de procesos de software se basa en uno de los tres modelos generales o paradigmas de desarrollo de software:

- **Enfoque en cascada:** Considera las actividades anteriores y las representa con fases de procesos separados, tales como la especificación de requerimientos, el diseño de software, la implementación, las pruebas etc.

- **Desarrollo iterativo:** Este enfoque entrelaza las actividades de especificación, desarrollo y validación. Un sistema inicial se desarrolla rápidamente a partir de especificaciones muy abstractas.

- **Modelo Prototipado:** Un modelo sirve de prototipo para la construcción del sistema final.

- **Transformación Formal:** Un modelo matemático del sistema se transforma formalmente en la implementación.

- **Desarrollo basado en Reutilización:** El sistema es ensamblado a partir de componentes existentes.

2.13.3. Modelos de Desarrollo de Software

Una parte importante de la ingeniería de software es el desarrollo de metodologías. Donde se mencionan: Metodologías Estructurada, Metodologías Orientada a Objetos, Metodologías Tradicionales y Metodología Agiles.

- **Metodología Estructurada:** Las metodologías estructuradas se basan en la estructuración y descomposición funcional de problemas en unidades más pequeñas interrelacionadas entre sí.

- **Metodología Orientada a Objetos:** Las metodologías orientadas a objetos modelan el sistema examinando el dominio del problema como un conjunto de objetos que interactúan entre sí.

- **Metodología Tradicional:** La metodología tradicional imponen una disciplina de trabajo sobre el proceso de desarrollo del software. Se centran especialmente en el control del proceso, con definición de roles, actividades, artefactos, herramientas y 15 notaciones para el modelado y documentación detallada.

- **Metodología Ágil:** Los procesos ágiles trabajan con requisitos desconocidos o variables. Si no existen requisitos estables, no existe una gran posibilidad de tener un diseño estable y de seguir un proceso totalmente planificado, que no vaya a variar ni en tiempo ni en dinero.

2.14. MÉTODO DE PRUEBA DEL SOFTWARE

Las pruebas son básicamente un conjunto de actividades dentro del desarrollo de **software**. Dependiendo del tipo de pruebas. Estas actividades podrán ser implementadas en cualquier momento de dicho proceso de desarrollo. Existen distintos modelos de desarrollo de software, así como modelos de pruebas. (Pressman, Mc Graw Hill, 2006). A cada uno corresponde un nivel distinto de involucramiento en las actividades de desarrollo.

2.14.1. Características y Fases de la Prueba

A la aplicación de técnicas de evaluación dinámicas se le denomina también pruebas de softwares. El contexto en el que se realiza la prueba de software. Concretamente la prueba de software se puede definir como una actividad en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas (configuración de la prueba), registrándose los resultados obtenidos.

El objetivo de las pruebas no es asegurar la ausencia de defectos en un software, únicamente puede mostrar que existen defectos en el software.

A continuación, se presentan algunas características de una buena prueba:

- Una buena prueba de tener una alta probabilidad de encontrar un fallo. Para alcanzar este objetivo el responsable de la prueba debe entender el software e intentar desarrollar una imagen mental de cómo podría fallar.

- Una buena prueba debe centrarse en dos objetivos:
    1. Probar si el software no hace lo que debe hacer.
    2. Probar si el software hace lo que no debe hacer.

- Una buena prueba no debe ser redundante. El tiempo y lo recursos son limitados, así que todas las pruebas deberían tener un propósito diferente.

- Una buena prueba debería ser la “mejor de la cosecha”, se debe emplear la prueba que tenga la más alta probabilidad de descubrir una clase entera de errores.

- Una buena prueba no debería ser ni demasiado sencilla ni demasiado compleja, pero si se quiere combinar varias pruebas a la vez se pueden enmascarar errores, por lo que en general, cada prueba debería realizarse separadamente.

Tareas a realizar para probar un software:

1. **Diseño de las pruebas.** Esto es, identificación de la técnica o técnicas de pruebas que se utilizaran para probar el software. Distintas técnicas de prueba ejercitan diferentes criterios como guía para realizar las pruebas. Seguidamente veremos algunas de estas técnicas.

2. **Generación de los casos de prueba.** Los casos de prueba representan los datos que se utilizar como entrada para ejecutar el software a probar. Más concretamente los casos de prueba determinan un conjunto de entradas, condiciones de ejecución y resultados esperados para un objetivo particular.

3. **Definición de los procedimientos de la prueba.** Esto es, especificación de cómo se va a llevar a cabo el proceso, quien lo va a realizar, cuando.

4. **Ejecución de la prueba**, aplicando los casos de prueba generados previamente e identificando los posibles fallos producidos al comparar los resultados esperados con los resultados obtenidos.

5. **Realización de un informe de la prueba**, con el resultado de la ejecución de las pruebas, que casos de prueba pasaron satisfactoriamente, cuálesno, y que fallos se detectaron. Tras estas tareas es necesario realizar un proceso de depuración de las faltas asociadas a las fallas identificados.

2.14.2. Definición Caso de Prueba

Un caso de prueba o test case es, en ingeniería de software, un conjunto de condiciones o variables bajo las cuales un analista determinara si una aplicación, un sistema software (software system), o una característica de estos es parcial o completamente satisfactoria. (Pressman, McGraw-Hill,1993)

2.14.3. Estructura de los Casos de Prueba

Los casos de pruebas tienen siguientes partes y subdivisiones.

- **Introducción/visión general:** contiene información general acerca de los casos de prueba.

  - ✓ **Identificador:** Es un identificador único para futuras referencias, por ejemplo, mientras se describe un defecto encontrado.

  - ✓ **Caso de prueba dueño/creador:** Es el nombre del analista o diseñador de pruebas quien ha desarrollado pruebas o es responsable de su desarrollo.

  - ✓ **Versión:** la actual definición del caso de prueba.

- ✓ **Nombre**: El caso de prueba debe ser un título entendible por personas, para la fácil comprensión del propósito del caso de prueba y su campo de aplicación.

- ✓ **Identificador de requerimientos:** el cual está incluido por el caso de prueba. También aquí puede ser un identificador de casos de uso o de especificación funcional.

- ✓ **Propósito:** contiene una breve descripción del propósito de la prueba, y la funcionalidad que chequea.

- ✓ **Dependencias:** indica que otros subsistemas están involucrados y en qué grado.

- ➢ **Actividades** de los casos de prueba

  - ✓ **Ambiente de prueba/ configuración:** Contiene información acerca de la configuración del hardware o software en el cual se ejecutará el caso de prueba.

  - ✓ **Inicialización:** Describe acciones, que deben ser ejecutadas antes de que los casos de prueba se hayan inicializado. Por ejemplo, debemos abrir algún archivo.

  - ✓ **Finalización:** describe acciones, que deben ser ejecutadas después de realizar el caso de prueba. Por ejemplo, si el caso de prueba estropea la base de datos, el analista debe restaurarla antes de que otro caso de prueba sea ejecutado.

- ✓ **Acciones:** pasos a realizar para completar la prueba.
- ✓ Descripción de los **datos de entrada.**

- Resultados
  - ✓ **Salida esperada:** contiene una descripción de lo que el analista debería ver tras haber completado todos los pasos de la prueba.

  - ✓ **Salida obtenida:** contiene una breve descripción de lo que el analista debería ver tras haber completado todos los pasos de la prueba.

  - ✓ **Resultado:** indica el resultado cualitativo de la ejecución del caso de prueba, a menudo con un **correcto/fallido.**

  - ✓ **Severidad:** indica el impacto del defecto en el sistema: Grave, Mayor, Normal, Menor.

  - ✓ **Evidencia:** en los casos que aplica, contiene informacion, bien un link al print de pantalla(screenshot), bien una consuta a una base de datos, bien el contenido de un fichero de trazas, donde se evidencia la salida obtenida.

  - ✓ **Seguimiento:** si un caso de prueba falla, frecuentemente la referencia al defecto implicado se debe enumerar en esta columna. Contiene el código correlativo del defecto, a menudo corresponde al código del sistema de tracking de bugs que se esté usando.

  - ✓ **Estado:** indica si el caso de prueba esta: no indica En curso, o terminado.

2.14.4. Prueba de caja blanca

Las pruebas de caja blanca (también conocidas como pruebas de caja de cristal o pruebas estructurales) se centran en los detalles procedimentales del software, por lo que su diseño está fuertemente ligado al código fuente. El ingeniero de pruebas escoge distintos valores de entrada para examinar cada uno de los posibles flujos de ejecución del programa y cerciorarse de que se devuelven los valores de salida adecuados. (Freddy M.M, 2008).

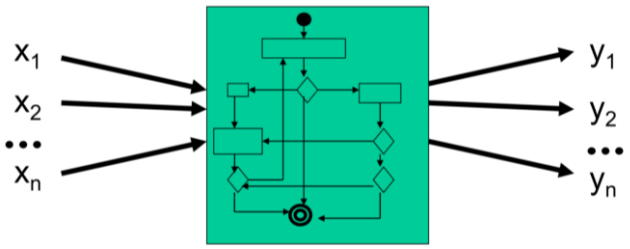


Figura 2. 11. Prueba de caja blanca
Fuente: (Carlos Blanco –IS2)

Las principales técnicas de diseño de pruebas de caja blanca son:

- **Pruebas de flujo de control;** En estas pruebas se quiere encontrar errores en la parte lógica del programa, utilizando las condiciones simples operador-relación o con condiciones complejas.

- **Pruebas de flujo de datos;** por medio de esta herramienta se hace las selecciones más adecuadas del flujo de datos, para llegar a una resolución correcta, esto para probar las variables y definiciones en el programa.

- **Pruebas de bifurcación (branch testing);** Como lo dice el título, es ligado a una bifurcación o para bucles. Con ella podemos definir si algún bucle esta concretamente implementado y si la línea de código donde exista una condición es la mejor opción o si debería ser cambiada.

- **Pruebas de caminos básicos;** Esta prueba lo que demuestra es el conjunto de pasos base del programa, lo que quiere lograr es que cada sentencia de código se ejecute mínimo una vez.

2.14.5. Prueba de caja negra

La prueba de caja negra, es una técnica de prueba de software en la cual la funcionalidad se verifica sin tomar en cuenta la estructura interna de código, detalles de implementación o escenarios de ejecución internos en el software. (Freddy M.M, 2008).

En las pruebas de caja negra, nos enfocamos solamente en la entradas y salidas del sistema. Sin preocuparnos en tener conocimiento de la estructura interna del programa del programa de software. Para obtener el detalle de cuáles deben ser esas entradas y salidas, nos basamos en los requerimientos de software y especificaciones funcionales.

Figura 2. 12. Prueba de caja negra
Fuente: (testing funcional-2019)

2.14.5.1. Pasos para realizar las pruebas de caja negra

Cada empresa o tester. Tiene su estrategia a la hora de aplicar este tipo de prueba, dependiendo del tipo de aplicación o el tiempo asignado a pruebas, entre otros factores, se realizan las pruebas de caja negra de una forma más intensiva o más exploratorias, aun así, y los pasos son:

- Lo primero será un previo análisis de los requisitos y especificaciones del software.

- El tester diseñara una batería de entrada validas, también llamado escenario de prueba positiva, para verificar si el software las procesa correctamente.

- También se diseñan entradas no validas (llamado escenario de prueba negativa) para comprobar si el software que se está probando es capaz de detectarlas y reaccionar antes entradas.

- Basándose en las entradas, el tester determina para cada una de estas las salidas esperadas correspondientes.

- Una vez que se tienen las entradas y sus correspondientes salidas, se diseñan los casos de prueba.

- Se ejecutan esos casos de pruebas.

- El tester comprueba la salida que ha emitido el software con la salida esperada de los casos de prueba.

- Si la salida del software coincide con la salida esperada, el software hace lo que tiene que hacer para esa entrada.

2.14.5.2. Tipos de pruebas de caja negra

Hay muchos tipos de prueba de caja negra, en la siguiente lista vamos a ver las más conocidas con una pequeña descripción:

- **Clase de equivalencias:**

  - ✓ Consisten en diseñar y clasificar entradas de datos para una funcionalidad similar del software, esperando que sean procesados de la misma manera.

  - ✓ Estas clases se definir tanto para datos validos como inválidos.

  - ✓ Estas clases se pueden definir tanto para datos validos como inválidos.

  - ✓ Las formas de clasificación se definen según criterios en función de las salidas de datos, valores internos, funcionalidades, eventos, etc.

  - ✓ Según estos datos, se diseñan casos de prueba para cubrir todos o parte de los datos validos o inválidos.

- **Análisis de valores límites:**

  - ✓ Se vale de los análisis de las pruebas de clase de equivalencias para obtener datos límites que acoten las posibles causisticas de los catos. Basándose en criterios como la probabilidad que se presentan más errores.

  - ✓ Los valores máximos y mínimos son los valores límites.

- ✓ Se pueden aplicar para datos validos e inválidos.

- ✓ Se diseña un caso de prueba por cada valor límite.

- ✓ Es una forma de identificar defectos de forma óptima y eficiente.

- **Tabla de decisiones:**

  - ✓ Las tablas de decisiones son una herramienta fundamental para la documentación de las reglas de negocio que conllevan una alta complejidad.

  - ✓ Se crean a partir de las especificaciones funcionales y de las reglas de negocio.

  - ✓ Las entradas y salidas se representan en muchas ocasiones con valores booleanos (verdadero o falso).

  - ✓ Las tablas de decisiones contienen secuencias de condiciones con valor.

  - ✓ Las tablas de decisiones contienen secuencias de condiciones encadenadas con las combinaciones de los valores booleanos para cada entrada de datos, así como su resultado esperado de cada combinación.

  - ✓ La columna de las tablas de decisiones se corresponde con una de las reglas de negocios que se representan con la combinación de las condiciones y de sus resultados.

- **Transición entre estados:**
  - ✓ Se basan en los diferentes estados de un software.
  - ✓ Se representan en un diagrama de transición entre estados.
  - ✓ Este diagrama de estados permite al tester visualizar información importante sobre el software, como pueden ser las transiciones, entradas, salidas o eventos encadenados.
  - ✓ Una tabla de estado nos permite ver las relaciones entre los estados del software y las entradas de datos. Nos ayuda a ver posibles transiciones inválidas.

- **Pruebas de caso de uso**
  - ✓ Estas pruebas de caja negra se basan en la representación de las interacciones entre actores (usuarios o sistemas que interactúan con nuestro software). Basándose en estas interacciones se pueden diseñar casos de prueba.

    Suelen ir acompañadas de precondiciones que deben cumplirse para que los actores funcionen de forma adecuada.
  - ✓ Cada caso de uso termina con pos condiciones que serán los resultados analizados después de la ejecución.
  - ✓ Se suelen usar para definir las pruebas de aceptación en la que participan usuarios o clientes.

- **Pruebas de historias de usuario:**

- ✓ Prueba muy utilizada en las metodologías Ágiles como puede ser el Scrum. Los requerimientos de usuario son preparados como historias de usuario.

- ✓ Estas historias de usuario describen una funcionalidad que puede ser desarrollada o probada en una misma secuencia.

- ✓ En estas descripciones en las historias de usuarios suelen aparecer funcionalidades a implementar, requerimientos no funcionales y los criterios de aceptación.

- ✓ Los criterios de aceptación abarcan la cobertura mínima requerida para la historia de usuario.

- ✓ Los casos de pruebas se basan en estos criterios de aceptación.

2.15. MÉTRICAS DE CALIDAD DE SOFTWARE

En general, la medición persigue tres objetivos fundamentales: ayudarnos a entender qué ocurre durante el desarrollo y el mantenimiento, permitirnos controlar qué es lo que ocurre en nuestros proyectos y poder mejorar nuestros procesos y nuestros productos. (Fenton y Pfleeger, 1997).

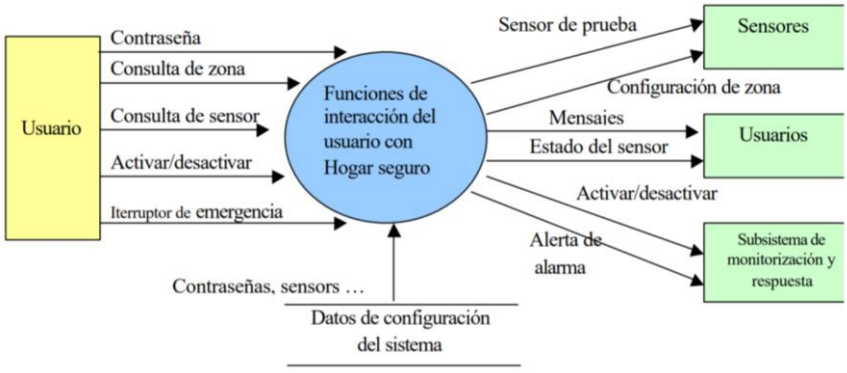


Figura 2. 13. Diagrama de Flujo de Hogar Seguro
Fuente: (presmman, 98)

2.15.1. Definición de calidad

Los desarrolladores de software más experimentados estarán de acuerdo en que obtener software de alta calidad es una meta importante. Pero, ¿cómo se define la calidad del software?

En el sentido más general se define como: Proceso eficaz de software que se aplica de manera que crea un producto útil que proporciona valor medible a quienes lo producen y a quienes lo utilizan. (Roger S.Pressman, 2013)

Visión General de los Factores que Afectan a la Calidad

Los primeros pasos hacia el desarrollo de métricas de la calidad del software. Estos factores evalúan el software desde tres puntos de vista distintos:

1. operación del producto (utilizándolo)
2. revisión del producto (cambiándolo)
3. transición del producto (modificándolo para que funcione en un entorno diferente, por ejemplo: “portándolo”)

Medida de la calidad

Aunque hay muchas medidas de la calidad de software, la corrección, facilidad de mantenimiento, integridad y facilidad de uso suministran indicadores útiles para el equipo del proyecto. Se sugiere definiciones y medidas para cada uno de ellos, tales como:

- **Corrección:** A un programa le corresponde operar correctamente o suministrará poco valor a sus usuarios. La corrección es el grado en el que el software lleva a cabo una función requerida. La medida más común de corrección son los defectos por KLDC,

- **Facilidad de mantenimiento.** El mantenimiento del software cuenta con más esfuerzo que cualquier otra actividad de ingeniería del software. La facilidad de mantenimiento es la habilidad con la que se puede corregir un programa si se encuentra un error, se puede adaptar si su entorno cambia u optimizar si el cliente desea un cambio de requisitos.

- **Integridad.** En esta época de intrusos informáticos y de virus, la integridad del software ha llegado a tener mucha importancia. Este atributo mide la habilidad de un sistema para soportar ataques (tanto accidentales como intencionados) contra su seguridad. El ataque se puede ejecutar en cualquiera de los tres componentes del software, ya sea en los programas, datos o documentos. Para medir la integridad, se tienen que definir dos atributos adicionales: amenaza y seguridad.

Facilidad de uso. El calificativo “amigable con el usuario” se ha transformado universalmente en disputas sobre productos de software. Si un programa no es “amigable con el usuario”, prácticamente está próximo al fracaso, incluso aunque las funciones que realice sean valiosas. La facilidad de uso es un intento de cuantificar “lo amigable que pude ser con el usuario” y se consigue medir en función de cuatro características:

- **Medidas de fiabilidad y de disponibilidad.** Los trabajos iniciales sobre fiabilidad buscaron extrapolar las matemáticas de la teoría de fiabilidad del hardware a la predicción de la fiabilidad del software. La mayoría de los modelos de fiabilidad relativos al hardware van más orientados a los fallos debidos al desajuste, que, a los fallos debidos a defectos de diseño, ya que son más probables debido al desgaste físico de los fallos relativos al diseño.

- **Eficacia de la Eliminación de Defectos** Una métrica de la calidad que proporciona beneficios tanto a nivel del proyecto como del proceso, es la eficacia de la eliminación de defectos (EED) En particular el EED es una medida de la habilidad de filtrar las actividades de la 68 garantía de calidad y de control al aplicarse a todas las actividades del marco de trabajo del proceso.

2.15.2. Estándar Internacional para Evaluar el Software ISO 9126

La ISO 9126 es un estándar internacional para evaluar la calidad del software en base a un conjunto de características y sub-características de la calidad. Cada sub-característica consta de un conjunto de atributos que son medidos por una serie de métricas.

Estas métricas miden artefactos obtenidos en etapas tardías del desarrollo de software, aumentando el costo de detección y corrección de errores. Por esta razón, en la literatura ha surgido un mayor interés por la definición de métricas que pretenden evaluar una o varias de las características de calidad definidas en el estándar ISO 9126, en etapas tempranas del desarrollo de software. (García, 2009).

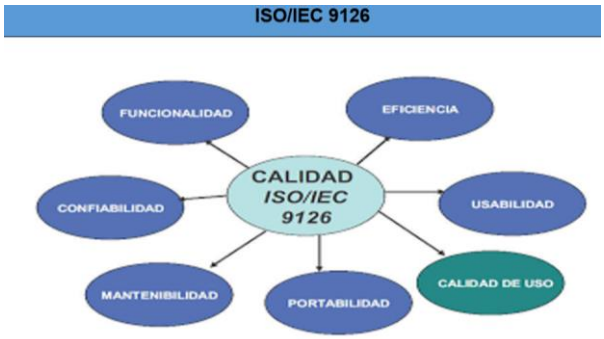


Figura 2. 14. Normas de Evaluación ISO/IEC 9126
Fuente: (isoiec-9126, 2013)

- **Funcionabilidad:** conjunto de atributos que se relacionan con la existencia de un conjunto de funciones y sus propiedades específicas. Las funciones son aquellas que satisfacen lo indicado o implica necesidades.

  - ✓ **Idoneidad:** Se enfoca a evaluar si el SW cuenta con un conjunto de funciones apropiadas para efectuar las tareas que fueron especificadas en su definición.

- ✓ **Exactitud:** Permite evaluar si el SW presenta resultados o efectos acordes a las necesidades para las cuales fue creado.

- ✓ **Interoperabilidad:** Permite evaluar la habilidad del SW de interactuar con otros sistemas previamente especificados.

- ✓ **Seguridad:** Se refiere a la habilidad de prevenir el acceso no autorizado, ya sea accidental o promediado, a los programas y datos.

- ✓ **Conformidad:** Evalúa si el SW se adhiere a estándares, convenciones o regulaciones en leyes y prescripciones similares.

- ➢ **Confiabilidad:** conjunto de atributos relacionados con la capacidad de mantener un nivel de presentación bajo condiciones establecidas durante un periodo de tiempo establecido.

  - ✓ **Madurez:** Permite medir la frecuencia de falla por errores en el SW.

  - ✓ **Recuperación:** Se refiere a la capacidad de restablecer el nivel de operación y recobrar los datos que hayan sido afectados directamente por una falla, así como al tiempo y el esfuerzo necesario para lograrlo.

  - ✓ **Tolerancia de fallos:** Se refiere a la habilidad de mantener un nivel específico de funcionamiento en caso de fallas del SW o de cometer infracciones de su interfaz específica.

- ➢ **Conformidad de la Fiabilidad:** La capacidad del software de cumplir a los estándares o normas relacionadas a la fiabilidad.

Figura 2. 15. Característica de Confiabilidad ISO/IEC 9126
Fuente: (Isoiec-9126, 2013)

- **Usabilidad:** La usabilidad es la capacidad del software de ser entendido, aprendido, y usado en forma fácil y atractiva. Algunos criterios de funcionalidad, fiabilidad y eficiencia afectan la usabilidad, pero para los propósitos de la ISO/IEC 9126 ellos no clasifican como usabilidad.

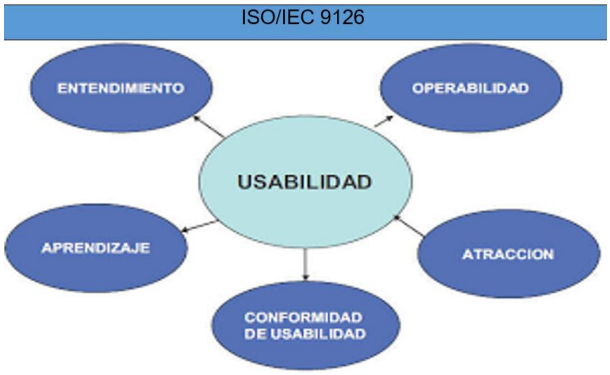


Figura 2. 16. Características de Usabilidad ISO/IEC 9126
Fuente: [isoiec-9126, 2013]

- ✓ **Entendimiento:** La capacidad que tiene el software para permitir al usuario entender si es adecuado, y de una manera fácil como ser utilizado para las tareas y las condiciones particulares de la aplicación. En este criterio se debe tener en cuenta la documentación y de las ayudas que el software entrega.

- ✓ **Aprendizaje:** La forma como el software permite al usuario aprender su uso. También es importante considerar la documentación.

- ✓ **Operatividad:** La manera como el software permite al usuario operarlo.

- ✓ **Atracción:** La presentación del software debe ser atractiva al usuario. Esto se refiere a las cualidades del software para hacer más agradable al usuario, ejemplo, el diseño gráfico.

- ✓ **Conformidad de Uso:** La capacidad del software de cumplir los estándares o normas relacionadas a su usabilidad.

- ✓ **Eficiencia:** La eficiencia del software es la forma del desempeño adecuado, de acuerdo a al número recursos utilizados según las condiciones planteadas. Se debe tener en cuenta otros aspectos como la configuración de hardware, el sistema operativo, entre otros.

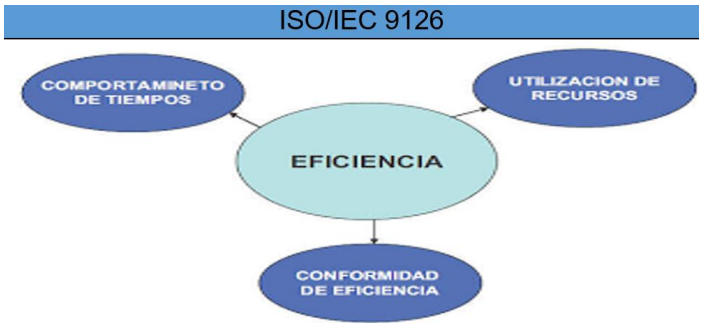


Figura 2. 17. Características de Eficiencia ISO/IEC 9126
Fuente: (isoiec-9126, 2013)

- ✓ **Comportamiento de Tiempos:** Los tiempos adecuados de respuesta y procesamiento, el rendimiento cuando realiza su función en condiciones específicas. Ejemplo, ejecutar el procedimiento más complejo del software y esperar su tiempo de respuesta, realizar la misma función, pero con más cantidad de registros.

- ✓ **Utilización de Recursos:** La capacidad del software para utilizar cantidades y tipos adecuados de recursos cuando este funciona bajo requerimientos o condiciones establecidas. Ejemplo, los recursos humanos, el hardware, dispositivos externos.

- ✓ **Conformidad de Eficiencia:** La capacidad que tiene el software para cumplir con los estándares o convenciones relacionados a la eficiencia.

- ➢ **Capacidad de Mantenimiento:** La capacidad de mantenimiento es la cualidad que tiene el software para ser modificado. Incluyendo correcciones o mejoras del software, a cambios en el entorno, y especificaciones de requerimientos funcionales.

Figura 2. 18. Capacidad de Mantenimiento ISO/IEC 9126
Fuente: (isoiec-9126, 2013)

El mantenimiento se divide en 5 criterios:

- ✓ **Capacidad de ser Analizado:** La forma como el software permite diagnósticos de deficiencias o causas de fallas, o la identificación de partes modificadas.

- ✓ **Confiabilidad:** La capacidad del software para que la implementación de una modificación se pueda realizar, incluye también codificación, diseño y documentación de cambios.

- ✓ **Estabilidad:** La forma como el software evita efectos inesperados para modificaciones del mismo.

- ✓ **Facilidad de Prueba:** La forma como el software permite realizar pruebas a las modificaciones sin poner el riesgo los datos.

- ✓ **Conformidad de Facilidad de Mantenimiento:** La capacidad que tiene el software para cumplir con los estándares de facilidad de mantenimiento.

- ✓ **Portabilidad:** La capacidad que tiene el software para ser trasladado de un entorno a otro.

Figura 2. 19. Características de portabilidad ISO/IEC 9126
Fuente: (isoiec-9126,2013)

La portabilidad se divide en cinco criterios:

- ✓ **Adaptabilidad:** Es como el software se adapta a diferentes entornos especificados (hardware o sistemas operativos) sin que implique reacciones negativas ante el cambio. Incluye la escalabilidad de capacidad interna.

- ✓ **Coexistencia:** La capacidad que tiene el software para coexistir con otro o varios softwares, la forma de compartir recursos comunes con otro software o dispositivo.

- ✓ **Conformidad de portabilidad:** La capacidad que tiene el software para cumplir con los estándares relacionados a la portabilidad.

- ✓ **Reemplazabilidad:** La capacidad que tiene el software para ser remplazado por otro software del mismo tipo, y para el mismo objetivo.

- ✓ **Facilidad de Instalación:** La facilidad del software para ser instalado en un entorno específico o por el usuario final.

- **Calidad de Uso:** Calidad en uso es la calidad del software que el usuario final refleja, la forma como el usuario final logra realizar los procesos con satisfacción, eficiencia y exactitud. La calidad en uso debe asegurar la prueba o revisión de todas las opciones que el usuario trabaja diariamente y los procesos que realiza esporádicamente relacionados con el mismo software.

Figura 2.20. Calidad de Uso de Software ISO/IEC 9126
Fuente: [isoiec-9126,2013]

La calidad de uso se divide en 4 criterios:

- ✓ **Eficacia:** La capacidad del software para permitir a los usuarios finales realizar los procesos con exactitud e integridad.

- ✓ **Productividad:** La forma como el software permite a los usuarios emplear cantidades apropiadas de recursos, en relación a la eficacia lograda en un contexto específico de uso.

- ✓ **Seguridad:** Se refiere al que el Software no tenga niveles de riesgo para causar daño a las personas, instituciones, software, propiedad intelectual o entorno. Los riesgos son normalmente el resultado de deficiencias en la funcionalidad (Incluyendo seguridad), fiabilidad, usabilidad o facilidad de mantenimiento.

- ✓ **Satisfacción:** La satisfacción es la respuesta del usuario a la interacción con el software, e incluye las actitudes hacia el uso del mismo.

2.15.3. Formula de evaluación de calidad

En la búsqueda de calidad de software se enfatizan tres puntos importantes:

1. Los requisitos del software son la base de las medidas de la calidad. La falta de concordancia con los requisitos es una falta de calidad.

2. Unos estándares específicos definen un conjunto de criterios de desarrollo que guían la manera en que se hace la ingeniería de software. Si no se siguen los criterios, habrá seguramente poca calidad.

3. Existe un conjunto de requisitos implícitos que a menudo no se nombran (por ejemplo, facilidad de mantenimiento). Si el software cumple con sus requisitos explícitos, pero falla en los implícitos, la calidad del software no será fiable.

2.16. MÉTODO DE ESTIMACIÓN DE COSTOS COCOMO

“Una parte importante de la toma de decisiones al comenzar un nuevo proyecto de desarrollo de software está dada por el costo que éste tendrá. La estimación de estos costos ha preocupado a analistas de sistema, gerentes de proyecto e ingenieros de software durante décadas. El primer obstáculo es clarificar el alcance del proyecto. Disponible en: (IBM, Estimación de Costos del Software,2009).

Una estimación que proporciona una vista suficientemente clara de la realidad del proyecto como para permitir al gestor del proyecto tomar buenas decisiones sobre cómo controlar el proyecto para lograr sus objetivos”. (Fernando Bersal, 2018).

COCOMO, Estimar el tamaño

Para hacer una **estimación efectiva** es necesario estimar primero el tamaño del software y después la planeación. Existen varias formas para estimar el tamaño de un proyecto, entre ellas:

- Utilizar un enfoque algorítmico, como los puntos de función, para **estimar el tamaño del programa** a partir de los requerimientos.

- Utilizar un software para estimar el tamaño del programa, a partir de la descripción de los requerimientos del programa (interfaces interactivas, funcionalidad lógica, persistencia de datos, etc.)

- Una vez que se tiene la estimación del tamaño, se puede pasar al segundo paso, que es **la estimación del esfuerzo**. Ésta es necesaria para poder saber a cuántas personas hay que incorporar en el proyecto; además, con una ésta se facilita la estimación de la planeación.

- El dato clave para hacer las estimaciones es el **número de líneas de código fuente**: LDC. La cantidad de Líneas de Código se debe estimar por experiencia, por analogía con otros proyectos semejantes, o por otros datos que se posean.

2.16.1. Características

Se tiene las siguientes características:

- Es una herramienta basada en las líneas de código la cual la hace muy poderosa para la estimación de costos y no como otros que solamente miden el esfuerzo en base al tamaño.
- Representa el más extenso modelo empírico para la estimación de software.
- Existen herramientas automáticas que estiman costos basados en Cocomo como ser: Costar, Cocomo8.

Objetivos para la Construcción de COCOMO II

- Desarrollar un modelo de estimación de costo y cronograma de proyectos de software que se adaptara tanto a las prácticas de desarrollo de la década del 90 como a las futuras.

- Construir una base de datos de proyectos de software que permitiera la calibración continua del modelo, y así incrementar la precisión en la estimación.

- Implementar una herramienta de software que soportara el modelo.

- Proveer un marco analítico cuantitativo y un conjunto de herramientas y técnicas que evaluaran el impacto de las mejoras tecnológicas de software sobre los costos y tiempos en las diferentes etapas del ciclo de vida de desarrollo.

- **Modelo de composición de aplicación:** Utilizado durante las primeras etapas de la Ingeniería del software, donde el prototipo de las interfaces de usuario, la interacción del sistema y del software, la evaluación del rendimiento, y la evaluación de la madurez de la tecnología son de suma importancia.

- **Modelo de fase de diseño previo:** Utilizado una vez que se han estabilizado los requisitos y que se ha establecido la arquitectura básica del software.

- **Modelo de fase posterior a la arquitectura:** Utilizado durante la construcción del software. (Aparicio, 2012).

2.16.2. Modelos de estimación de costo con COCOMO II

En la estimación del tamaño de software Cocomo II utiliza tres técnicas:

- Puntos Objeto
- Puntos Función No Ajustados y
- Líneas de Código Fuente.

Puntos objeto

El procedimiento para determinar Puntos Objeto en un proyecto software se resume en:

- **Determinar Cantidad de Objetos:** Estimar la cantidad de pantallas, reportes, componentes que contendrá la aplicación.

- **Clasificar cada instancia de un objeto** según sus niveles de complejidad (simple, media o difícil).

Determinar la cantidad de Puntos Objeto sumando todos los pesos de las instancias de los tipos de objetos especificados.

PUNTOS OBJETO

Para Pantallas			
Cantidad de Vistas Contenidas	Cantidad y fuente de las tablas de datos		
	Total < 4 (< 2 servidor < 3 cliente)	Total < 8 (< 2 - 3 servidor < 3 - 5 cliente)	Total 8 + (> 3 servidor < 5 cliente)
< 3	Simple	Simple	Media
3 - 7	Simple	Media	Difícil
> 8	Media	Difícil	Difícil
Para Reportes			
Cantidad de Vistas Contenidas	Cantidad y fuente de las tablas de datos		
	Total < 4 (< 2 servidor < 3 cliente)	Total < 8 (< 2- 3 servidor < 3-5 cliente)	Total 8 + (> 3 servidor < 5 cliente)
0 o 1	Simple	Simple	Media
2 o 3	Simple	Media	Difícil
4 +	Media	Difícil	Difícil

Figura 2. 141. Puntos Objeto
Fuente: (modelos de estimación,2012)

Puntos Función No Ajustados

El modelo Cocomo II usa Puntos Función y/o Líneas de Código Fuente (SLOC) como base para medir tamaño en los modelos de estimación de Diseño Temprano y Post-Arquitectura.

Los puntos función están basados en información disponible en las etapas tempranas del ciclo de vida del desarrollo de software.

- **Entradas Externas (Inputs):** Entrada de datos del usuario o de control que ingresan desde el exterior del sistema para agregar y/o cambiar datos a un archivo lógico interno.

- **Salidas Externas (Outputs):** Salida de datos de usuario o de control que deja el límite del sistema de software.

- **Archivo Lógicos Internos (Archivos):** Incluye cada archivo lógico, es decir cada grupo lógico de datos que es generado, usado, o mantenido por el sistema de software.

- **Archivos Externos de Interface (Interfaces):** Archivos transferidos o compartidos entre sistemas de software.

- **Líneas de código Fuente (SLOC):** El objetivo es medir la cantidad de trabajo intelectual puesto en el desarrollo de un programa. Definir una línea de código.

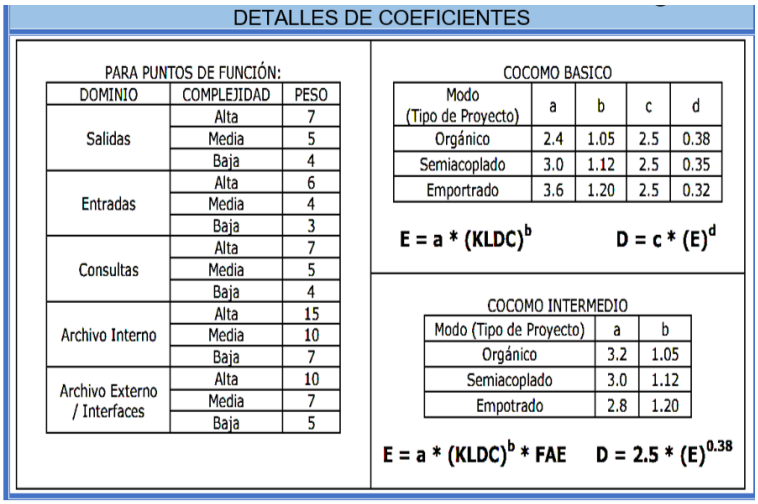
Figura 2. 22. Puntos Función no Ajustados
Fuente: (Modelos de estimación, 2012)

Por otro lado, el modelo Cocomo Calcula esfuerzo y coste en función del tamaño del programa (LDC). Cocomo está definido para tres tipos de proyectos de Software:

- **Modo orgánico:** Proyectos pequeños y sencillos, con equipos de experiencia en la aplicación y requisitos poco rígidos.

- **Modo semi-acoplado:** Proyectos intermedios (más complejos), con equipos que poseen variados niveles de experiencia y requisitos más rígidos.

- **Modo empotrado:** Proyectos que deben ser desarrollados en un conjunto de Hardware, Software y restricciones muy grandes.



DETALLES DE COEFICIENTES

PARA PUNTOS DE FUNCIÓN:

DOMINIO	COMPLEJIDAD	PESO
Salidas	Alta	7
	Media	5
	Baja	4
Entradas	Alta	6
	Media	4
	Baja	3
Consultas	Alta	7
	Media	5
	Baja	4
Archivo Interno	Alta	15
	Media	10
	Baja	7
Archivo Externo / Interfaces	Alta	10
	Media	7
	Baja	5

COCOMO BASICO

Modo (Tipo de Proyecto)	a	b	c	d
Orgánico	2.4	1.05	2.5	0.38
Semiacoplado	3.0	1.12	2.5	0.35
Emportrado	3.6	1.20	2.5	0.32

$E = a * (KLDC)^b$ $D = c * (E)^d$

COCOMO INTERMEDIO

Modo (Tipo de Proyecto)	a	b
Orgánico	3.2	1.05
Semiacoplado	3.0	1.12
Empotrado	2.8	1.20

$E = a * (KLDC)^b * FAE$ $D = 2.5 * (E)^{0.38}$

Figura 2. 23. Detalle de coeficientes de COCOMO II
Fuente: (Estimaciones Cocomo).

2.16.3. Ventajas y desventajas

Ventajas

- Es fácil de realizar y de interpretar.

- Tiene pocas variables.

- Se acerca a la realidad en la mayoría de los casos.

- COCOMO es transparente, se puede ver cómo trabaja con otros modelos como SLIM (Software Life Cycle Management).

- Manejadores de costo que ayudan particularmente a el estimador a comprender el impacto de diferentes factores que afectan en el costo del proyecto.

Desventajas

- No saca resultados fiables en proyectos demasiado pequeños.
- La elección de las variables es muy subjetiva y depende de persona que realiza el estudio.
- El triunfo depende ampliamente de la adaptación del modelo a las necesidades de la organización, usando datos históricos; los cuales no siempre están disponibles.

2.17. TÉCNICAS DE SEGURIDAD

La Seguridad Informática es un proceso complejo que conlleva tres aspectos: Gente, procesos y tecnología. Si estas variables no se evalúan y resuelven como partes de un todo, se obtiene como producto final un potencial o real desastre. (Blanco, 2005).

- **Seguridad de Hardware:** La seguridad de hardware se puede relacionar con un dispositivo que se utiliza para escanear un sistema o controlar el tráfico de red. Los ejemplos más comunes incluyen cortafuegos o firewalls de hardware y servidores proxy.

- **Seguridad de Software:** La seguridad de software se utiliza para proteger el software contra ataques maliciosos de hackers y otros riesgos, de forma que nuestro software siga funcionando correctamente con este tipo de riesgos potenciales. Esta seguridad de software es necesaria para proporcionar integridad, autenticación y disponibilidad.

- **Seguridad de red:** La seguridad de red se refiere a cualesquiera actividades diseñadas para proteger la red. En concreto, estas actividades protegen la facilidad de uso, fiabilidad, integridad y seguridad de su red y datos. La seguridad de red efectiva se dirige a una variedad de amenazas y la forma de impedir que entren o se difundan en una red de dispositivos

2.18. HERRAMIENTAS DE DESARROLLO

2.18.1. Gestor de Base de Datos MariaDB

Un SGBD debe permitir especificar tipos y estructuras, permite la manipulación de los datos mediante consultas y la actualización de la base de datos de manera sencilla, existen diferentes gestores de bases de datos, pero lo más utilizado son: MariaDB: es uno de los gestores de base de datos más usados, tanto por la comunidad estudiantil como por las empresas, está desarrollada bajo las licencias de GPL y la licencia comercial de Oracle. (Gázquez, 2018)

- ✓ MariaDB es multiplataforma, funciona en una amplia lista de sistemas operativos, al contrario de otros sistemas como por ejemplo MS SQL Server que solo es compatible con Windows.

- ✓ Trae soporte para aproximadamente diez motores de almacenamiento, cada uno con sus características especiales; entre ellas se deben destacar:

- ✓ Bajos costos: la versión Community es libre bajo la licencia GNU y la versión comercial es relativamente más económica en relación con otros sistemas. (Gázquez, 2018)

2.18.1.1. Clasificación de los SGBD

Esta clasificación está basada en el modelo de datos en que está basado el SGBD. Los modelos de datos más habituales son:

- ➢ Relacional (SGBDR): representa a la base de datos como una colección de tablas. Estas bases de datos suelen utilizar SQL como lenguaje de consultas de alto nivel.

- ➢ Orientado a objetos: define a la base de datos en términos de objetos, sus propiedades y sus operaciones. Todos los objetos que tienen la misma estructura y comportamiento pertenecen a una clase y las clases de organizan en jerarquías.

- ➢ Objeto-relacional o relacional extendido: son los sistemas relacionales con características de los orientados a objetos.

- ➢ Jerárquico: representa los datos como estructuras jerárquicas de árbol. Un SGBD también puede clasificarse por el número de usuario a los que da servicio:

    a) Monousuario
    b) Multiusuario

También puede clasificarse según el número de sitios de la base de datos:

c) Centralizado: la base de datos y el software SGBD están almacenados en un solo sitio.

d) Distribuido (SGBDD): la base de datos y el software SGBD pueden estar distribuidos en múltiples sitios conectados por una red.

2.18.2. Lenguaje de Programación “php”

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. PHP fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos.

2.18.3. Javascript

JavaScript es un lenguaje interpretado que permite incluir macros en páginas Web. Estas macros se ejecutan en el ordenador del visitante de nuestras páginas, y no en el servidor. JavaScript proporciona los medios para:

CAPITULO III

3. MARCO APLICATIVO

En el presente capitulo se desarrollará las faces correspondientes a las conceptualizaciones, análisis y diseño del sistema siguiendo el proceso de desarrollo de la metodología UWE detallado en el capítulo II, las mismas que nos servirá para el desarrollo del sistema.

3.1. FASE I: FORMULACIÓN

Para el diagnostico de los objetivos y alcances del sistema para el uso de la empresa se realizó:

Entrevista.

Se realizaron dos entrevistas, la primera al Gerente General de la Empresa, y la segunda a los encargados de venta de la empresa, quiénes expresan la necesidad de poder cambiar la manera en que se maneja la información de las compras, ventas y actualización del stock de los productos que comercializa.

3.2. FASE II: PLANIFICACIÓN

La Planificación genera estimación de costo, la evaluación de riesgo y el calendario del desarrollo y fechas de entrega.

Consiste en el cálculo del costo integral del proyecto y se determinan las amenazas que se relacionan con el impulso del desarrollo además se determina un plan muy detallado para el desarrollo y progresos de la aplicación.

3.3. FASE III: ANÁLISIS

Esta fase es la fundamental en el desarrollo del sistema, se muestra los requerimientos de los interesados.

3.3.1. Análisis de Requerimientos

3.3.1.1. Requerimiento de Hardware

Características mínimas necesarias en cuanto a hardware para montar la aplicación, se describen a continuación.

- Memoria RAM 512 MB o superior.
- Disco Duro 20 GB.
- Servidor Web.
- Motor de Base de datos MariaDB.
- Conexión a internet 512Kb.
- Tarjeta gráfica de 512 Mb o superior.

3.3.1.2. Requerimiento de Software

La interfaz de usuario será creada utilizando como base PHP5, HTML5, java script y como gestor de base de datos MariaDB.

3.3.2. Funciones del Sistema

La descripción del producto muestra las características del sistema en una perspectiva general, señalando las necesidades de la empresa y las propuestas de solución planteadas, por el presente proyecto.

NECESIDADES	CARACTERISTICAS	
Registro automatizado de ventas de productos.	Registrar las ventas de manera automatizada, y luego tendrá reportes de manera instantánea y precisa.	Alta
Gestionar Información del cliente.	Registra datos de clientes, para su posterior tratamiento.	Alta
Gestionar información de Productos.	Registra datos de productos, para su posterior tratamiento.	Alta
Control del stock de d productos.	La aplicación deberá ofrecer información del stock existente de productos.	Alta
Administración de interfaz segura y confiable.	Posibilidad de editar los contenidos del sistema, de manera segura con el uso de niveles de usuario que cada usuario registrado tiene, también debe registrar las actividades realizadas en el sistema de cada usuario.	Alta

Tabla 3. 1. Descripción del sistema
Fuente: (Elaboración Propia)

3.3.2.1. Requerimientos Funcionales

En base al pedido de la empresa en las entrevistas se formula la siguiente tabla en la que muestra los requisitos funcionales que el sistema debe satisfacer detalladas a continuación.

REF.	FUNCION	CATEGORIA
R.1.1.	El sistema debe tener seguridad en el acceso a la información del sistema	Evidente
R.1.2.	Acceder al sistema por tipos de usuario (Administrador, Encargado).	Evidente
R.1.3.	Registro de productos y proveedores	Evidente
R.1.4.	Validación de entradas de texto	Oculto
R.1.5.	Generar reportes de salida de productos	Evidente
R.1.6.	Generación de reporte existencias de productos	Evidente
R.1.7	Registros de usuarios con sus respectivos roles	Evidente
R.1.8	Inicio y cierre del sistema	Oculto

Tabla 3. 2. Requerimientos Funcionales "Sistema Informático Web"
Fuente: (Elaboración Propia)

3.3.2.2. Requerimientos No Funcionales

En las siguientes tablas se muestra los requerimientos no funcionales.

ROL	FUNCION	CATEGORIA
R.1.1.	El sistema debe visualizarse y funcionar correctamente en cualquier navegador como ser internet Explore, Mozilla, Chrome, Opera.	Evidente
R.1.2.	Mantenimiento adecuado de la red local	Evidente
R.1.3.	Respaldo energético del servidor, para asegurar la disponibilidad del sistema.	Evidente
R.1.4.	Soporte y mantenimiento periódico para asegurar el buen rendimiento del sistema	Evidente

Tabla 3. 3. Requerimientos No Funcionales Sistema
Fuente: (Elaboración Propia)

Las funciones que debe realizar se clasifican en tres categorías como se detallan en la siguiente tabla.

No se encuentran elementos de tabla de ilustraciones.	SIGNIFICADO
Evidente	Debe realizarse y que los usuarios estén conscientes que se ha realizado.
Oculto	Debe realizarse, aunque no es visible para los usuarios. Estos se aplican a muchos servicios técnicos subyacentes, por ejemplo, guardar información en un mecanismo persistente de almacenamiento.
Superflua	Opcionales, su conclusión no repercute de forma significativa en costo ni en otras funciones.

Tabla 3. 4. Categoría de Funciones
Fuente: (Arias, 2008)

3.4. FASE IV: INGENIERÍA

3.4.1. Modelo de Caso de Uso

En este punto se plasma el análisis de requerimiento del sistema mediante el diseño de casos de uso, que describe el comportamiento del sistema frente a las acciones de los actores del mismo, funcionamiento del sistema y además elementos que permiten la abstracción del problema. A continuación, se realiza el modelamiento donde se puede apreciar cómo interactúan los actores sobre los casos de uso.

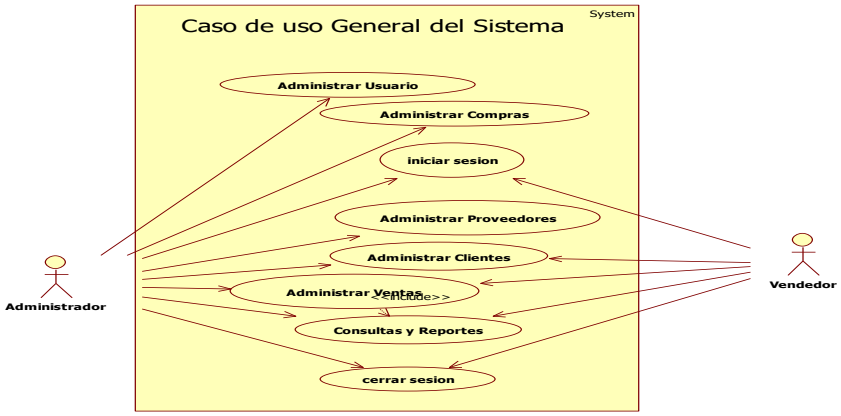


Figura 3. 1. Diagrama de Caso de uso General del Sistema
Fuente: (Elaboración Propia)

ACTORES	DESCRIPCIÓN
Administrador.	Encargado de administrar los usuarios, las compras y ventas de autopartes, así también registrar los clientes y proveedores de la empresa. Realiza altas, bajas y/o modificaciones de los clientes y proveedores.
Vendedor.	Encargado de realizar las ventas de los productos que vende la empresa además de registrar el cliente que lo está comprando.

Tabla 3. 5. Descripción de Actores de Casos de Uso General del Sistema
Fuente: Elaboración propia

3.4.2. Detallado de Casos de Uso.

Caso de Uso: Iniciar Sesión

EL Administrador y los Vendedores pueden iniciar sesión, introduciendo sus datos personales como usuario y contraseña, en el caso de que los datos no coincidirán con los datos de usuarios de la base de datos del sistema no podrán iniciar sesión.

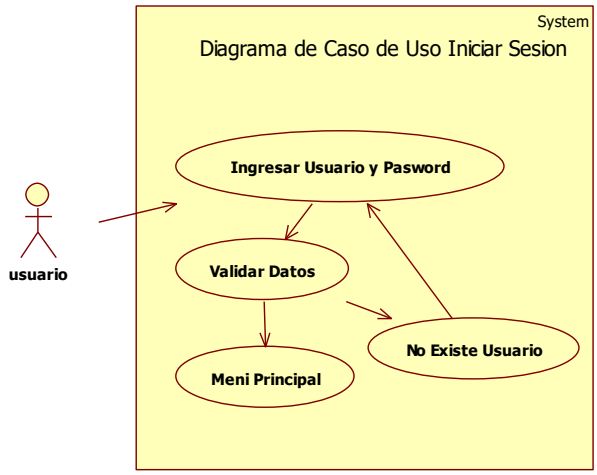


Figura 3. 2. Diagrama de Caso de Uso Iniciar Sesión
Fuente: (Elaboración propia)

CASO DE USO	
Nombre	Iniciar sesión
Descripción.	Permite identificar al usuario para poder acceder a las funcionalidades del sistema, dependiendo del tipo de usuario se le asignan sus funciones o permisos.
Actores	Administrador y Vendedor.
Precondiciones	Para que un Administrador y Vendedor inicie una sesión debe estar registrado en el sistema.

Tabla 3. 6. Caso de Uso Iniciar Sesión
Fuente: (Elaboración propia)

FLUJO NORMAL	FLUJO ALTERNATIVO
1. Introduce datos personales (usuario y contraseña). 2. El sistema verifica los datos. 3. Sesión iniciada.	Si el usuario ingresa datos incorrectos el sistema denegará su acceso.

Tabla 3. 7. Flujo de Eventos: Caso de Uso Iniciar Sesión
Fuente: (Elaboración propia)

Caso de Uso: Administrar Usuario.

El usuario administrador será el encargado de administrar a los vendedores. Podrá crear nuevos, modificarlos, eliminarlos, habilitarlos y/o inhabilitarlos. Estas dos últimas opciones sirven para que, aunque un vendedor tenga registrada una cuenta en el sistema no pueda acceder al sistema si no está activo.

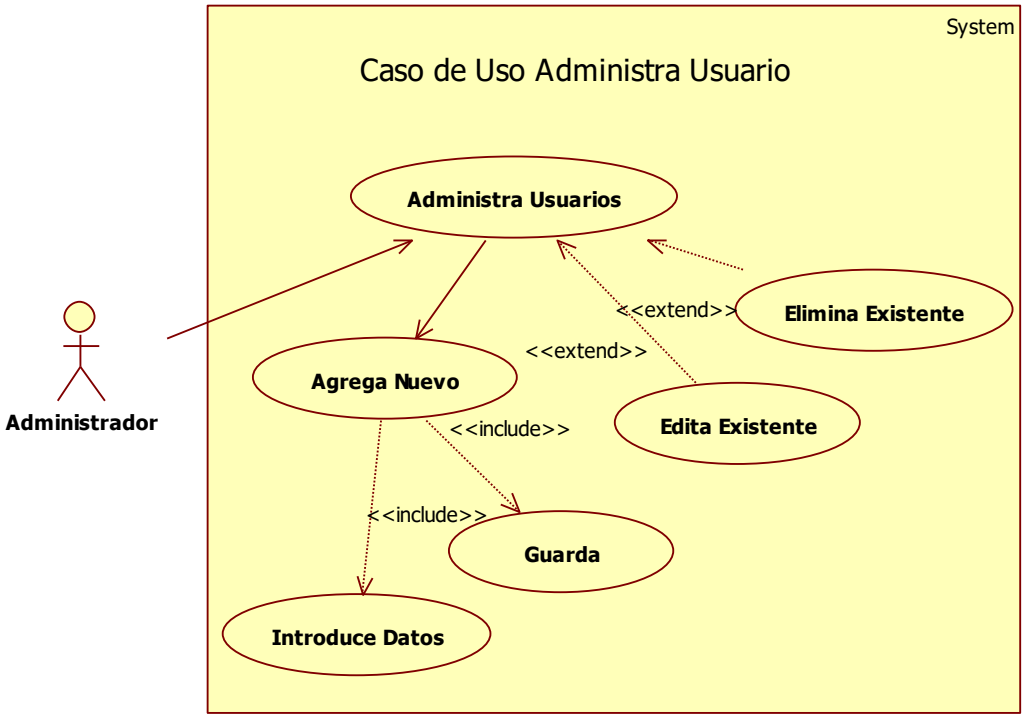


Figura 3. 3. Caso de Uso Administrar Usuario
Fuente: (Elaboración propia)

CASO DE USO	
Nombre	Administrar Usuario.
Descripción.	El administrador agrega, modifica y/o elimina usuarios para que puedan acceder al sistema y realizar sus respectivas labores.
Actores	Administrador.
Precondiciones	Debe tener una cuenta con el suficiente permiso, además de tener una sesión activa.

Tabla 3. 8. Caso de Uso Administrar Usuarios

Fuente: (Elaboración Propia)

FLUJO NORMAL	FLUJO ALTERNATIVO
1. Elige la opción Administrar Usuario. 2. Escoge adicionar, editar o eliminar un usuario. 3. Introduce datos personales en caso de editar y adicionar. 4. Pulsa en Guardar	Si el usuario no tuviera los permisos necesarios no se habilitará la administración.

Tabla 3. 9. Flujo de Eventos: Caso de Uso Administrar Usuario

Fuente: (Elaboración propia)

Caso de Uso: Administrar Clientes.

El usuario administrador y/o vendedor serán los encargados de administrar los clientes. Podrán crear nuevos, modificarlos, eliminarlos.

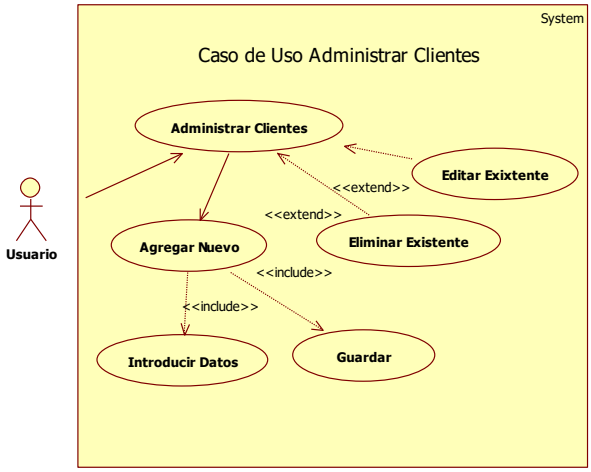


Figura 3. 4. Caso de Uso Administrar Clientes

Fuente: (Elaboración propia)

CASO DE USO	
Nombre	Administrar Clientes.
Descripción.	El usuario agrega, modifica y/o elimina Clientes para que se pueda realizar la venta.
Actores	Administrador, Vendedor.
Precondiciones	Debe tener una cuenta con el suficiente permiso, además de tener una sesión activa.

Tabla 3. 10. Caso de Uso Administrar Clientes

Fuente: (Elaboración Propia)

FLUJO NORMAL	FLUJO ALTERNATIVO
1. Elige la opción Administrar Clientes. 2. Escoge adicionar, editar o eliminar un Clientes. 3. Introduce datos personales en caso de editar y adicionar. 4. Pulsa en Guardar	Si el usuario no tuviera los permisos necesarios no se habilitará la administración.

Tabla 3. 11. Flujo de Eventos: Caso de Uso Administrar Clientes

Fuente: (Elaboración propia)

Caso de Uso: Administrar Proveedores.

El usuario administrador será el encargado de administrar los proveedores.

Podrán crear nuevos, modificarlos, eliminarlos.

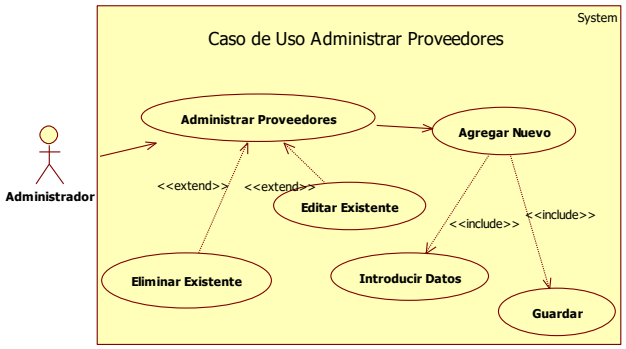


Figura 3. 5. Caso de Uso Administrar Proveedores

Fuente: (Elaboración propia)

CASO DE USO	
Nombre	Administrar Proveedores.
Descripción.	El administrador agrega, modifica y/o elimina Clientes para que se pueda realizar la venta.
Actores	Administrador.
Precondiciones	Debe tener una cuenta con el suficiente permiso, además de tener una sesión activa.

Tabla 3. 12. Caso de Uso Administrar Proveedores

Fuente: (Elaboración Propia)

FLUJO NORMAL	FLUJO ALTERNATIVO
1. Elige la opción Administrar Proveedores. 2. Escoge adicionar, editar o eliminar un Proveedores. 3. Introduce datos personales en caso de editar y adicionar. 4. Pulsa en Guardar	Si el usuario no tuviera los permisos necesarios no se habilitará la administración.

Tabla 3. 13. Flujo de Eventos: Caso de Uso Administrar Proveedores

Fuente: [Elaboración propia]

Caso de Uso: Administrar Compras.

El usuario administrador será el encargado de administrar las Compras. Podrán crear nuevos, modificarlos.

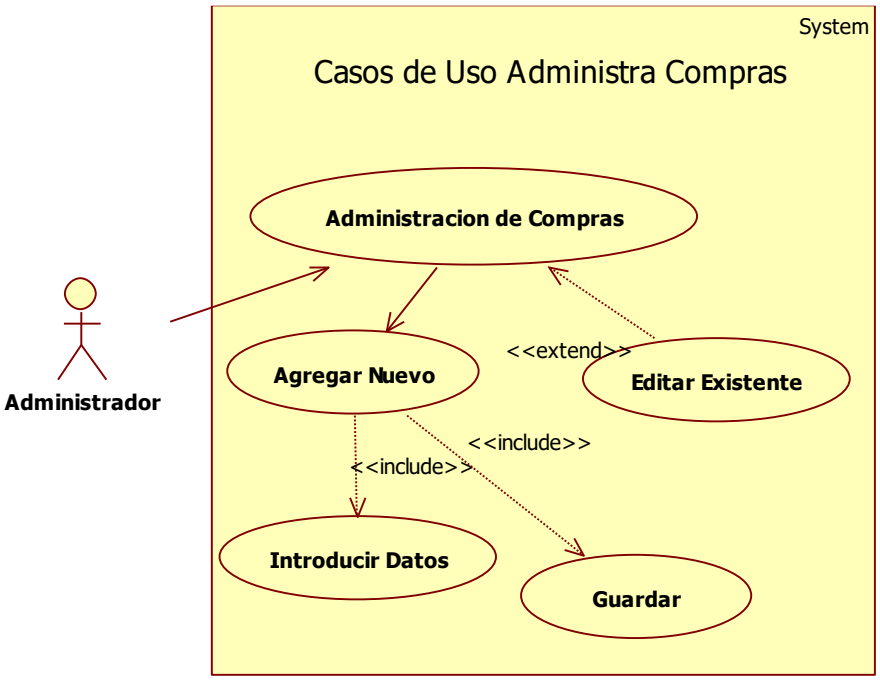


Figura 3. 6. Caso de Uso Administrar Compras
Fuente: (Elaboración propia)

CASO DE USO	
Nombre	Administrar Compras.
Descripción.	El administrador agrega, modifica compras.
Actores	Administrador.
Precondiciones	Debe tener una cuenta con el suficiente permiso, además de tener una sesión activa.

Tabla 3. 14. Caso de Uso Administrar Compras
Fuente: (Elaboración Propia)

FLUJO NORMAL	FLUJO ALTERNATIVO
1. Elige la opción Administrar Compras. 2. Escoge adicionar, editar una Compra. 3. Introduce datos personales en caso de editar y adicionar. 4. Pulsa en Guardar	Si el usuario no tuviera los permisos necesarios no se habilitará la administración.

Tabla 3. 15. Flujo de Eventos: Caso de Uso Administrar Compras
Fuente: (Elaboración propia)

Caso de Uso: Administrar Ventas.

El usuario administrador y/o vendedor será el encargado de administrar las Ventas. Podrán crear nuevos, modificarlos.

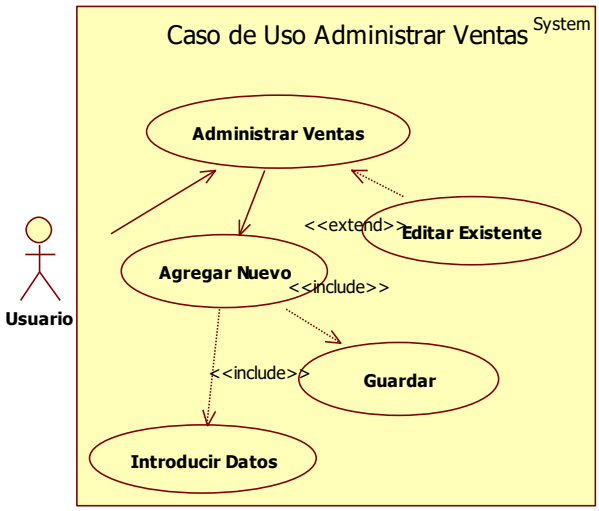


Figura 3. 7. Caso de Uso Administrar Ventas
Fuente: (Elaboración propia)

CASO DE USO	
Nombre	Administrar Ventas.
Descripción.	El usuario agrega, modifica Ventas.
Actores	Administrad, Vendedor.
Precondiciones	Debe tener una cuenta con el suficiente permiso, además de tener una sesión activa.

Tabla 3. 16. Caso de Uso Administrar Ventas

Fuente: (Elaboración Propia)

FLUJO NORMAL	FLUJO ALTERNATIVO
1. Elige la opción Administrar Ventas. 2. Escoge adicionar, editar una Venta. 3. Introduce datos personales en caso de editar y adicionar. 4. Pulsa en Guardar	Si el usuario no tuviera los permisos necesarios no se habilitará la administración.

Tabla 3. 17. Flujo de Eventos: Caso de Uso Administrar Ventas

Fuente: (Elaboración propia)

Caso de Uso: Cerrar sesión.

Es la acción de terminar una sesión de un usuario específico, finalizando así el uso de nuestro sistema. Es la acción inversa de iniciar sesión.

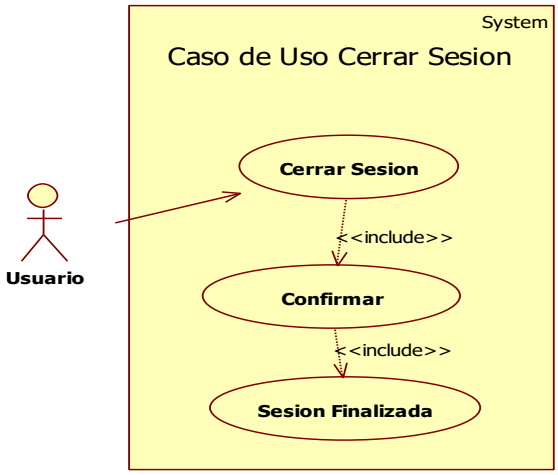


Figura 3. 8. Caso de Uso Cerrar Sesión

Fuente: (Elaboración propia)

CASO DE USO	
Nombre	Cerrar sesión.
Descripción.	Permite a los usuarios a desconectar los permisos del sistema.
Actores	Administrador y vendedor.
Precondiciones	Haber iniciado sesión

Tabla 3. 18. Caso de Uso Cerrar Sesión

Fuente: Elaboración propia

FLUJO NORMAL	FLUJO ALTERNATIVO
1. Clic en Cerrar sesión. 2. Confirmar en la alerta que muestra el sistema. 3. Sesión finalizada.	Si ocurre algún error el sistema mostrará una notificación.

Tabla 3. 19. flujo de Eventos: Caso de Uso Cerrar Sesión

Fuente: (Elaboración propia)

3.4.3. Modelo de Navegación

Estos tienen por contenido ilustrar los vínculos lógicos y de navegación entre clases. El modelo de clase navegación y las asociaciones de navegación general del proyecto, que expresan la navegación directa.
Por tratarse de una aplicación orientado a la web es necesario modelar la navegación de la aplicación.

3.4.4. Modelo de Estructura de Navegación

Este modelo de estructura de navegación permite ilustrar los vínculos lógicos y de navegación entre clases, menús, índices y clases de proceso para tener una mejor comprensión del sistema se muestra en la siguiente figura.

3.4.3.1. Modelo de navegación: Administrador

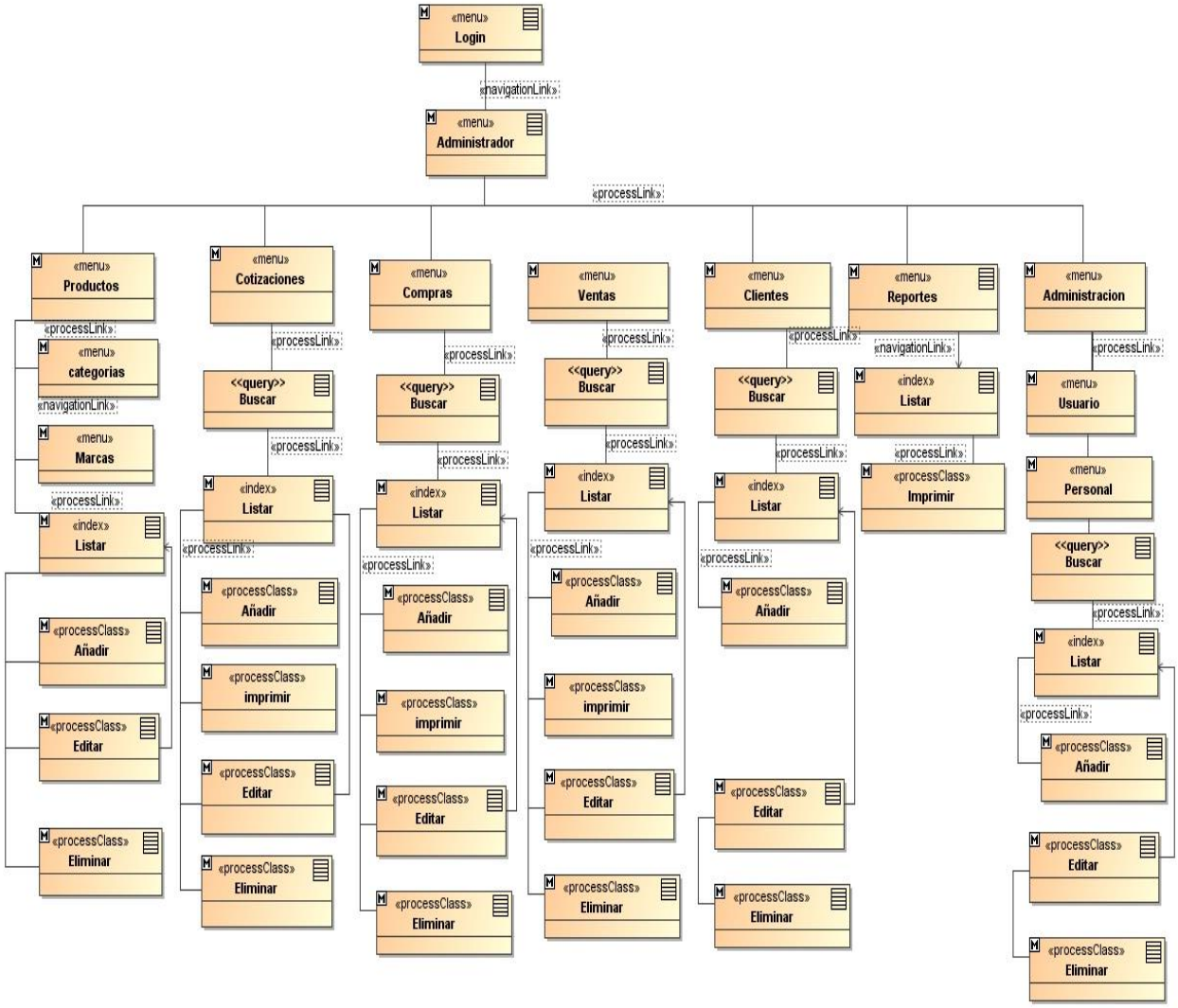


Figura 3. 9. Modelo de Navegación: Administrador
Fuente: Elaboración Propia

3.4.3.2. Modelo de Navegación: Vendedor

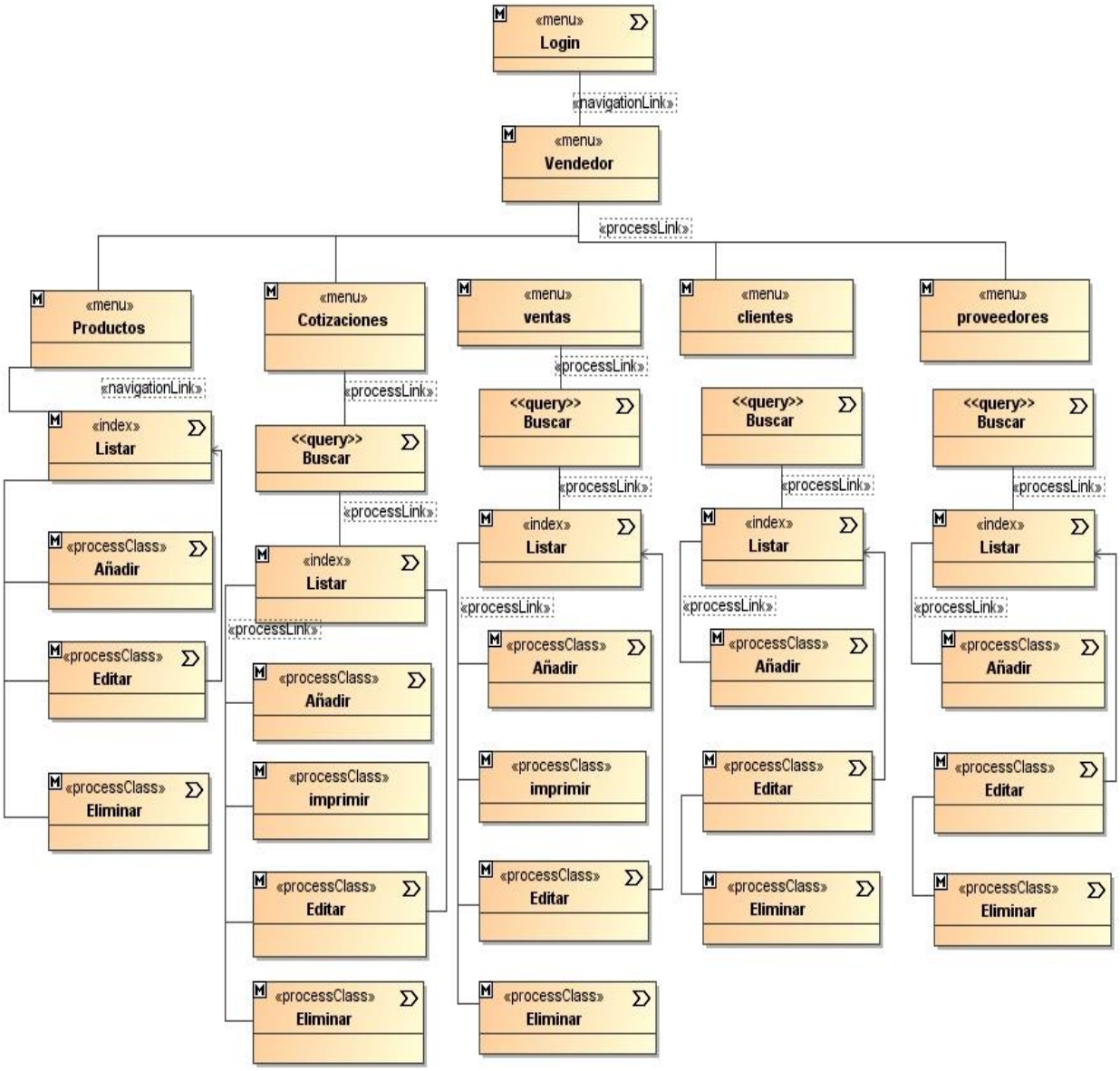


Figura 3. 10. Modelo de Navegación: Vendedor
Fuente: Elaboración Propia

3.4.4. Modelo de Presentación

A continuación, se muestra los modelos de presentación, se muestra como los usuarios podrán acceder al sistema mostrando los menús correspondientes según el tipo de usuario.

3.4.4.1 Modelo de Presentación: Página de Inicio

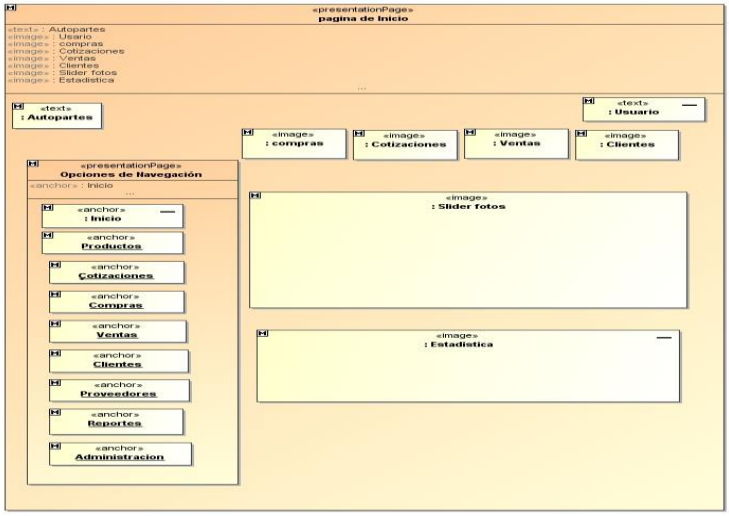


Figura 3. 11. Modelo de Presentación de Página de Inicio
Fuente: Elaboración Propia

3.4.4.2 Modelo de Presentación: Control de Productos

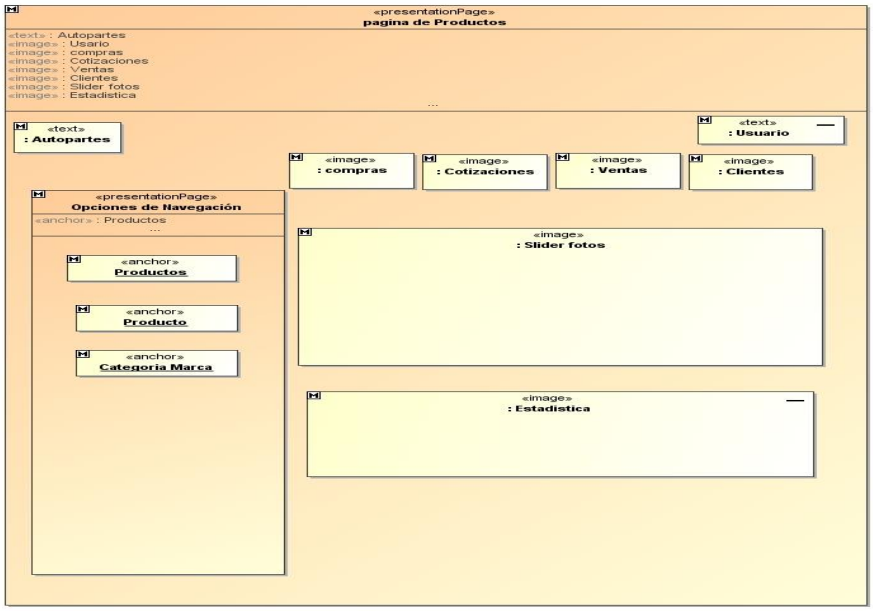


Figura 3. 12. Modelo de Presentación de Control de Productos
Fuente: Elaboración Propia

3.4.4.3 Modelo de Presentación: Control de Compra

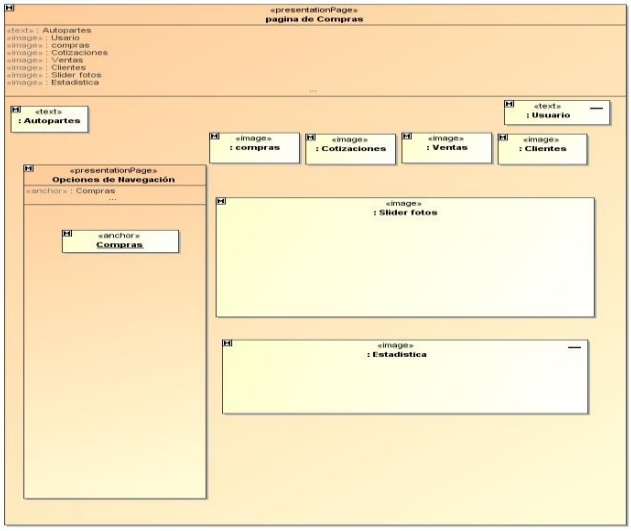


Figura 3. 13. Modelo de Presentación de Control de Compras
Fuente: Elaboración Propia

3.4.4.4. Modelo de Presentación: Control de Ventas

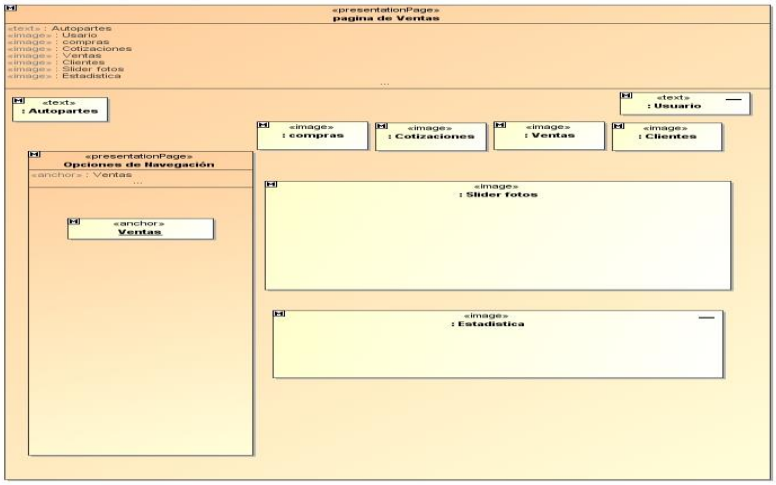


Figura 3. 14. Modelo de Presentación de Control de Ventas
Fuente: Elaboración Propia

3.4.4.5. Modelo de Presentación: Control de Clientes

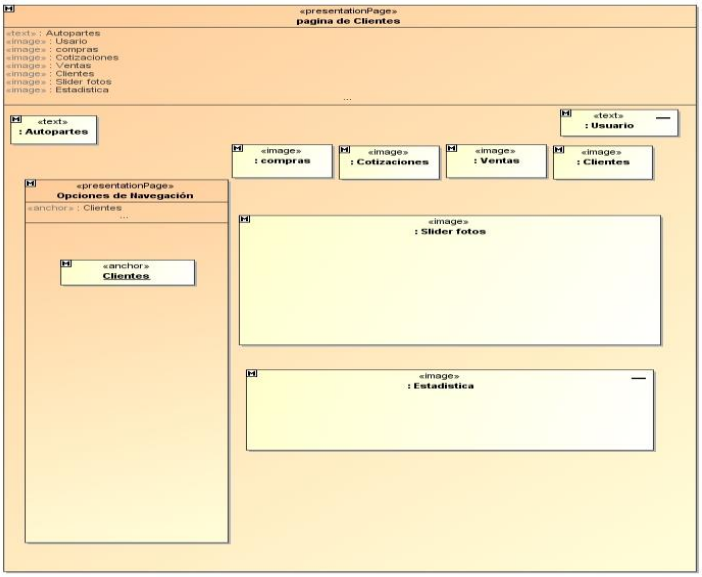


Figura 3. 15. Modelo de Presentación de Control de Clientes
Fuente: Elaboración Propia

3.4.4.6. Modelo de Presentación: Control de Proveedores

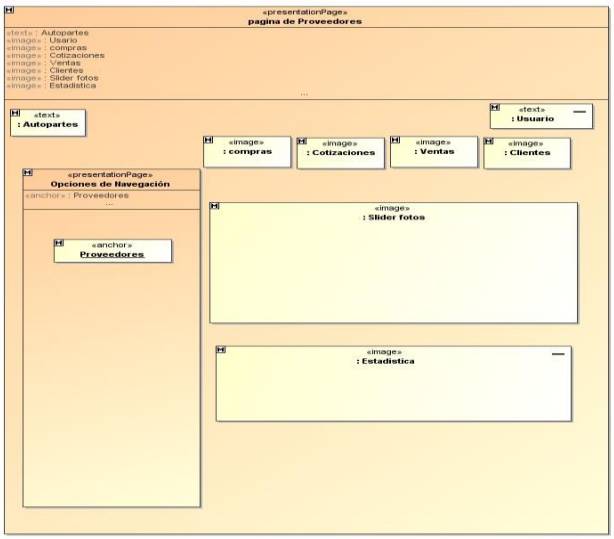


Figura 3. 16. Modelo de Presentación de Control de Clientes
Fuente: Elaboración Propia

3.4.4.7. Modelo de Presentación: Control de Reportes

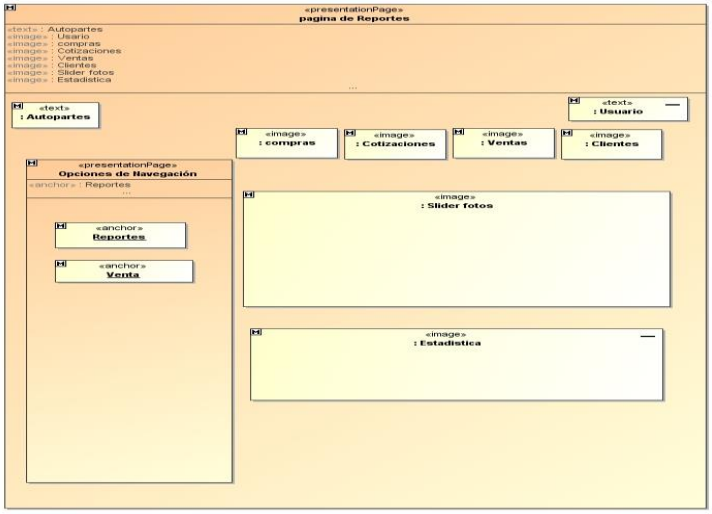


Figura 3. 17. Modelo de Presentación de Reportes
Fuente: Elaboración Propia

3.4.4.8. Modelo de Presentación: Control de Reportes

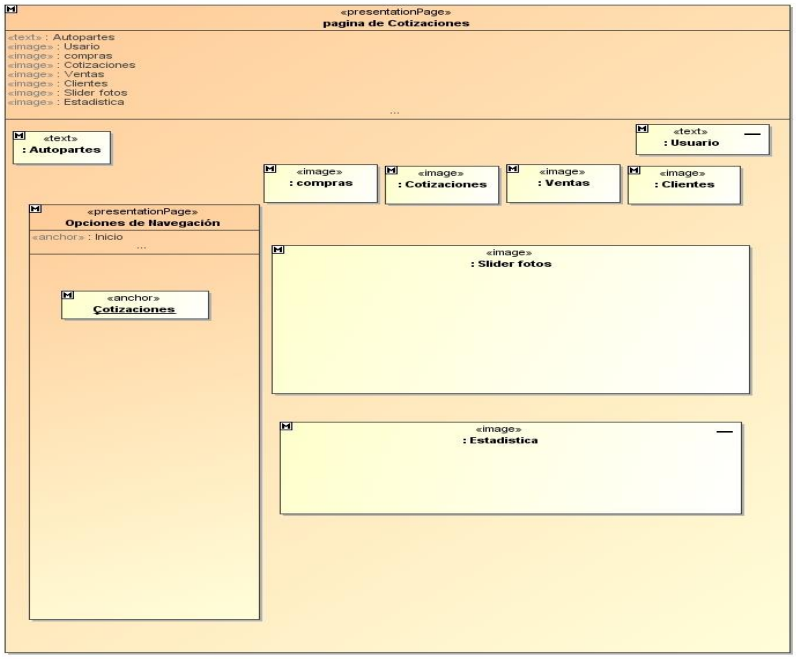


Figura 3. 18. Modelo de Presentación de Reportes
Fuente: Elaboración Propia

3.4.5. Diseño Conceptual

Se basa en la especificación de requerimientos producido por el análisis de los requerimientos (fase de análisis), el diseño define como estos requerimientos se cumplirán, la estructura que debe darse a la aplicación web

3.4.6. Modelo de Contenidos

El diagrama de contenido tiene por propósito mostrar las relaciones entre las entidades y la estructura de los datos que se encuentran alojadas en el sistema. Esto se representa mediante un diagrama de clases UML como se muestra en la siguiente figura:

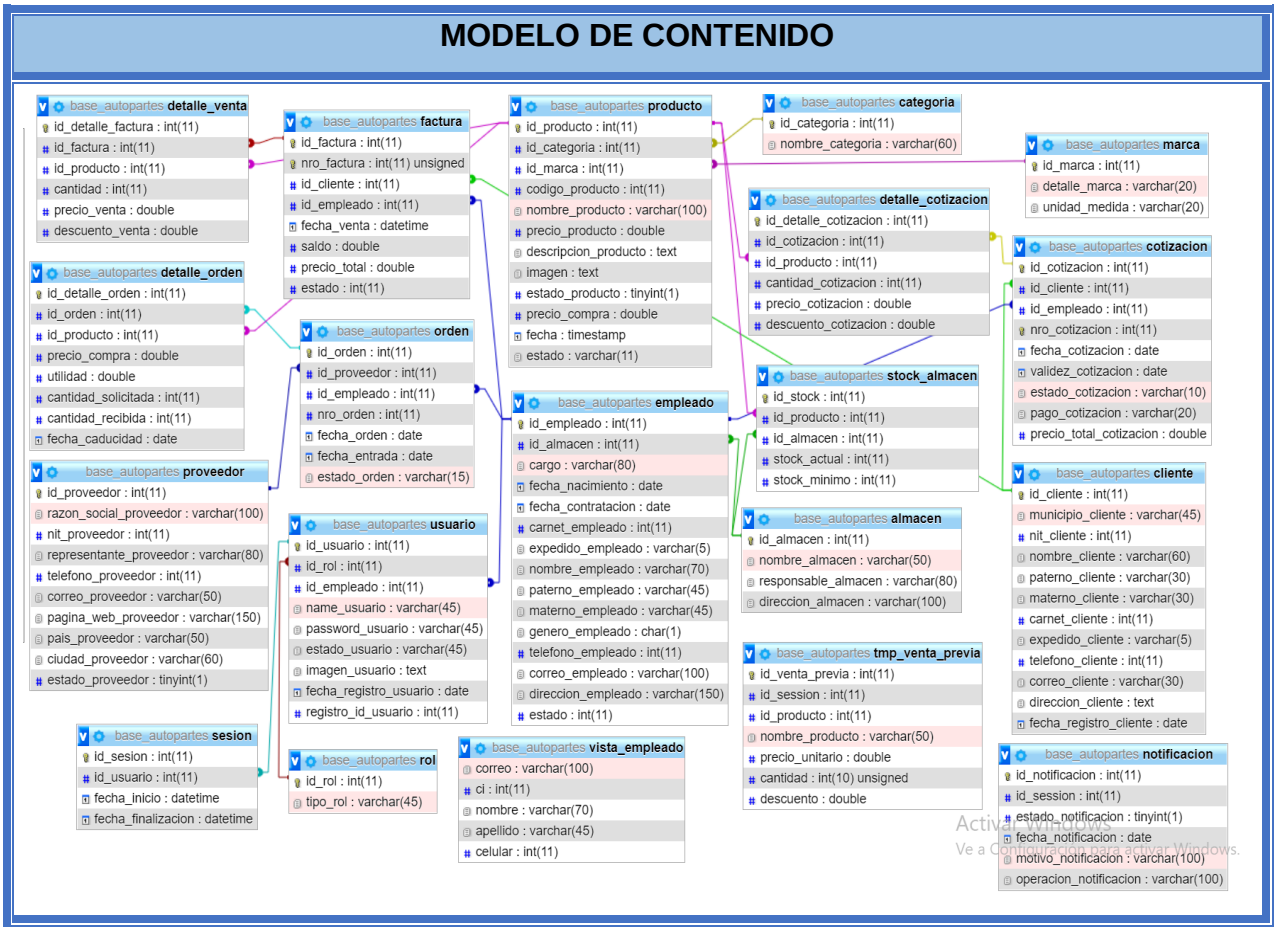


Figura 3. 19. Modelo de Contenido del Sistema
Fuente:(Elaboración Propia)

3.5. FASE V: GENERACIÓN DE PÁGINAS Y PRUEBAS

3.5.1. Diseño de Implementación

La implementación consiste en mostrar el desarrollo de presentación de las interfaces del sistema y sus elementos constructivos para el sistema tenemos la siguiente secuencia de pantalla básica.

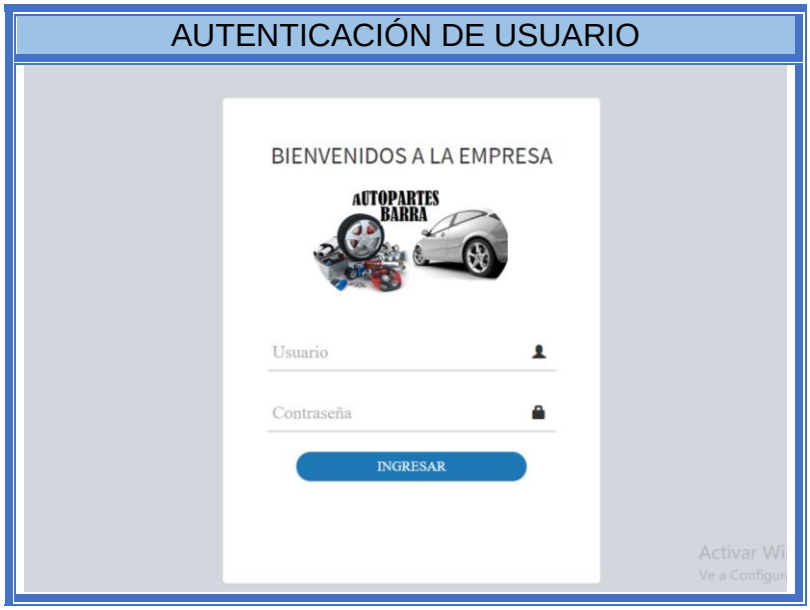


Figura 3. 20. Pantalla de Autenticación de Usuario
Fuente: (Elaboración Propia)

Una vez autenticado el usuario, ingresa a la pantalla principal, donde se puede encontrar el menú en pestañas según el tipo de usuario. Tal como se observa en la figura 3.12.

Figura 3. 21. Pantalla de Menú del Usuario Administrador
Fuente:(Elaboración Propia)

En la figura 3.13 Al seleccionar la opción ver Usuarios del menú que se encuentra en la parte superior de la ventana, el usuario puede ver una lista desplegable con las opciones de ver se puede ver una lista de todos los usuarios registrados.

Figura 3. 22. Pantalla de Menú del Usuario Administrador
Fuente:(Elaboración Propia)

En la figura 3.14 Se muestra en la pantalla donde se realiza el registro de nuevos usuarios, mismo que accederá al sistema.

Figura 3. 23. Pantalla de Registro de Usuarios
Fuente:(Elaboración Propia)

En la figura 3.15 Se muestra los productos registrados en el sistema donde el administrador puede ver, agregar un nuevo producto, editar o eliminar a un producto.

Figura 3. 24. Pantalla Administración de Productos
Fuente:(Elaboración Propia)

En la figura 3.16 Se muestra en la pantalla donde se realiza el registro de nuevo producto, donde se introduce el número de carnet del proveedor.

Figura 3. 25. Pantalla Nuevo Productos
Fuente:(Elaboración Propia)

En la figura 3.17 se muestra el listado de compras de productos.

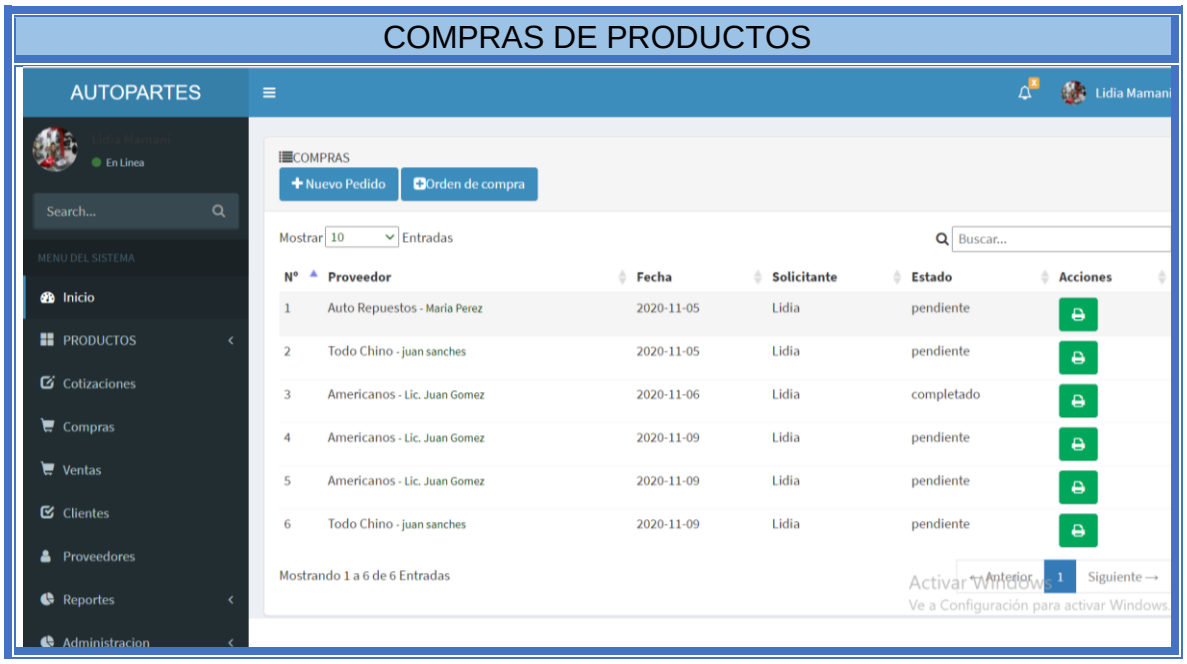


Figura 3. 26. Pantalla Compra de Productos
Fuente:(Elaboración Propia)

En la figura 3.18. Se muestra en la pantalla donde se realiza el registro de nuevos usuarios, mismo que accederá al sistema.

Figura 3. 27. Pantalla de Registro de Usuarios
Fuente:(Elaboración Propia)

En la figura 3.19 Se muestra el formulario de ventas de productos, donde agregar el producto, fecha y el cliente.

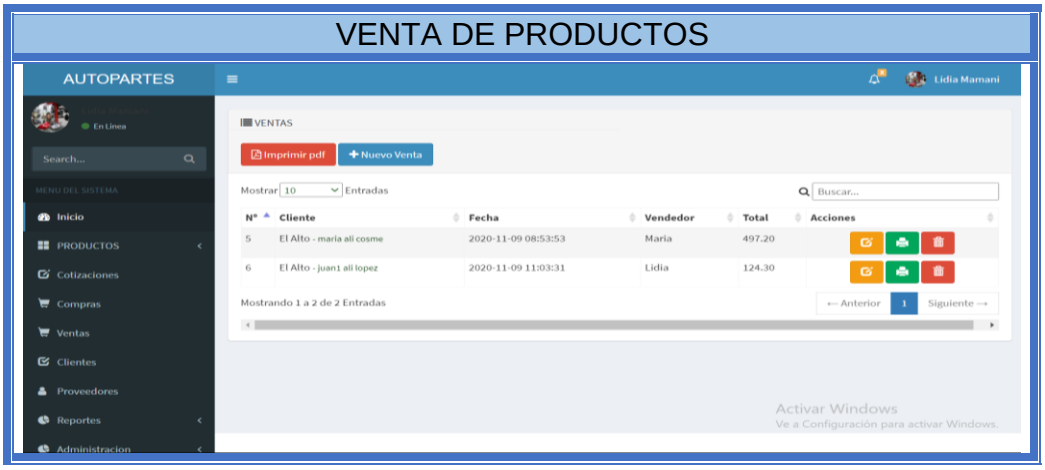


Figura 3. 28. Pantalla Salida de Productos
Fuente:(Elaboración Propia)

En la figura 3.20 Se muestra el formulario para la venta de productos, donde se puede seleccionar productos para la venta.

Figura 3. 29. Pantalla Productos a seleccionar para la venta
Fuente:(Elaboración Propia)

En la figura 3.21 Se encuentra el reporte de la venta de producto en formato PDF, donde se detallan los productos que se están vendiendo y su precio respectivo.

Figura 3. 30. Reporte de venta de productos
Fuente:(Elaboración Propia)

En la figura 3.22 Se encuentra el Reporte semanales, mensuales y anuales de venta de productos.

Figura 3. 31. Pantalla Reporte de venta de Productos
Fuente:(Elaboración Propia)

3.6. FASE VI: EVALUACIÓN DEL CLIENTE

En este punto, se efectúan todas las modificaciones y variaciones que se encontraron en la etapa de pruebas y se incorporan al sistema para el siguiente incremento, de tal modo que se asegure la satisfacción por parte del cliente, según los requerimientos solicitados.

3.6.1. Pruebas de Software

Para las pruebas de software se utiliza el método de pruebas de caja negra el cual evalúa las entradas introducidas por los usuarios y analiza el resultado devuelto por el sistema además de la prueba de funcionalidad.

3.6.2. Pruebas de Caja Blanca

Esta prueba se orienta al cálculo de las regiones que deben ser consideradas como partes independientes del sistema, este se basa en el diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivarlos, de forma general, se debe seguir:

- Emplear el diseño del sistema para elaborar el grafo del programa.

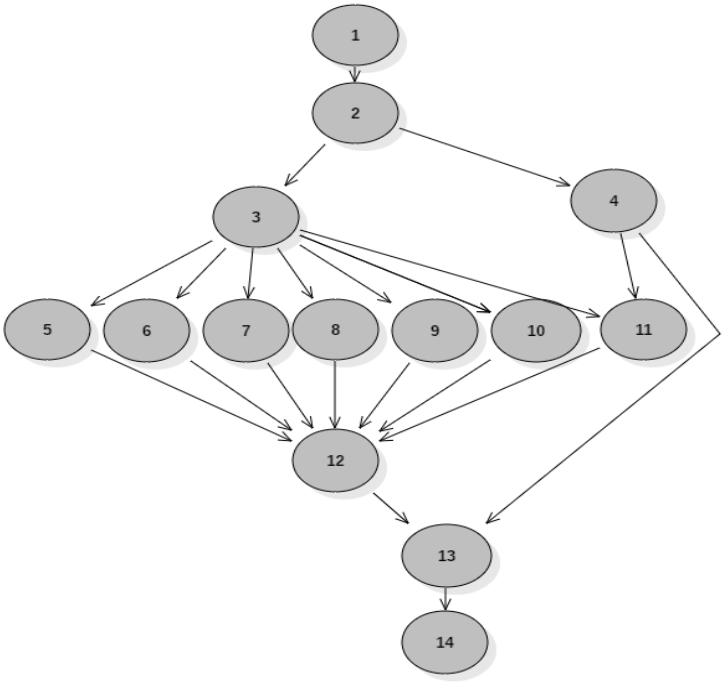


Figura 3. 32. Caja Blanca
Fuente: Elaboración Propia

Dónde:

- Inicio del sistema (1)
- Menú principal (2)
- Módulo Productos (3)
- Información de los Productos (4)
- Formulario de cotización (5)
- Compra (6)
- Mostrar información de las compras (7)
- Ventas (8)
- Mostrar información de las ventas (9)
- Clientes (10)
- Mostrar información de los clientes (11)
- Fin de ciclo usuario (12)
- Fin ciclo Sistema (3)
- Fin del sistema (14)

Analizado el grafo generado a partir de las características del sistema, ahora se procede a determinar la complejidad ciclomática del grafo mediante:

$$\mathbf{V\ (G) = A – N + 2}$$

Dónde:

A = 21 (Aristas)

N = 14 (Nodos)

Por tanto, **V (G)** = 21 – 14 +2 = 9

Determinar el conjunto básico de caminos linealmente independientes. Los caminos que deben ser probados dadas ciertas variables son 9. Estos caminos son los siguientes:

Camino 1: 1-2-3-4-12-13-14

Camino 2: 1-2-3-6-12-13-14

Camino 3: 1-2-11-12-13-14

Camino 4: 1-2-3-12-13-14

Camino 5: 1-2-5-7-12-13-14

Camino 6: 1-2-5-8-12-13-14

Camino 7: 1-2-11-12-13-14

Camino 8: 1-2-5-11-12-13-14

Camino 9: 1-2-4-13-14

Preparar los casos de prueba para forzar la ejecución de cada camino. Esta última condición establece que, para la ejecución de ciertos caminos, se deben establecer las condiciones en las que al menos se ejecuta los nodos establecidos en el camino.

Camino 1: Formulario de Productos se ejecuta a partir del listado de productos.

Camino 2: Este módulo se ejecuta en el instante de la compra de productos.

Camino 3: Se muestra la información de los diferentes clientes.

Camino 4: se muestra la lista de control de los

productos y control mínimo

Camino 5: Muestra el formulario para hacer una venta, la cantidad existente de productos.

Camino 6: Muestra el formulario para hacer una compra de productos.

Camino 7: Muestra información de los clientes.

Camino 8: La cotización se registra en él sistema.

Camino 9: El encargado del almacén ingresa a Administrar y concluye.

3.6.3. Pruebas de Caja Negra

3.6.3.1 Prueba de Caja Negra -Inicio de Sesión

se realiza las pruebas a la interfaz mostrada a continuación:

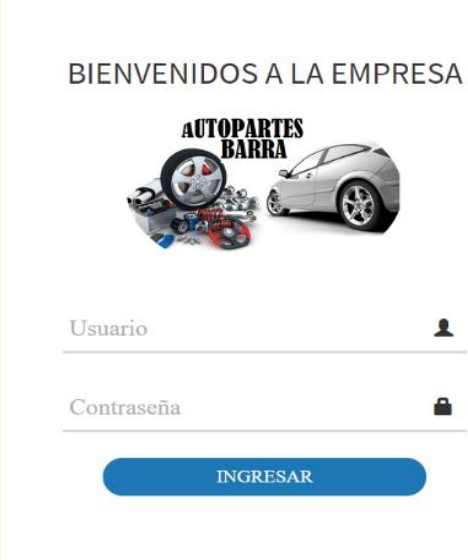


Figura 3. 33. Prueba de Caja Negra Iniciar sesión

Fuente: Elaboración Propia

Campo	Entrada Valida	Entrada Invalida
Usuario	Cadena de texto	Caracteres Especiales, espacios en blanco
Contraseña	Cadena de texto	Caracteres especiales, espacio en blanco

Tabla 3. 20. Valores Limite – Inicio de Sesión

Fuente: Elaboración Propia

Entradas		Salida	Resultados
Usuario	**Contraseña**		
		“ingrese el usuario y contraseña”	El sistema valida que no se ingresen datos en blanco
Administrador	12345	“Bienvenido al sistema de Inventarios”	Al introducir datos validos el sistema concede al acceso al mismo

Tabla 3. 21. Prueba de Caja Negra – Inicio de Sesión

Fuente: Elaboración Propia

Como se observó la interfaz de inicio se sesión cumple con la función programada para que el usuario se identifique al empezar el sistema

3.6.3.2. Prueba de Caja Negra -Registro de Productos

En el proceso de registrar productos descrita en la figura 3.25, el mismo cumple con la función de ingresar los datos del producto al sistema, de esta forma podrá ser utilizado para las Ventas.

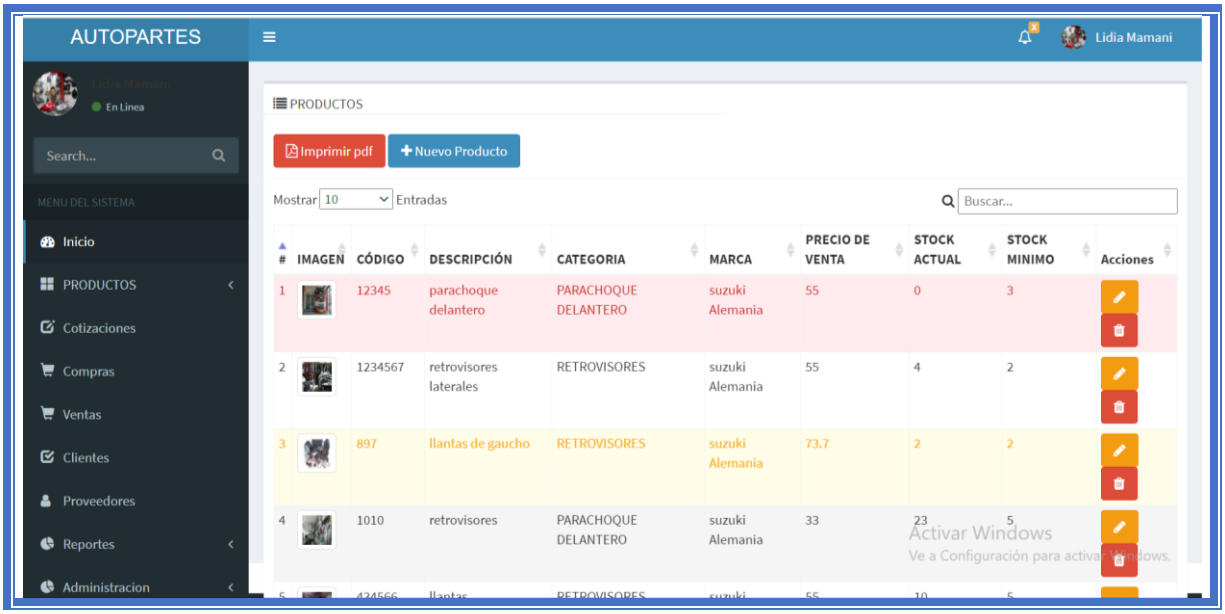


Figura 3. 34. Prueba de Caja negra – Registro Productos

Fuente: Elaboración Propia

Campo	Entrada Valida	Entrada Invalida
Código Primario Código Secundario	Cadena de texto	Caracteres Especiales, espacios en blanco
Nombre del Producto	Cadena de texto	Caracteres especiales, espacio en blanco
Código del Producto	Cadena de texto	Caracteres especiales, espacio en blanco
Descripción	Cadena de texto	Caracteres especiales, espacio en blanco
Categoría	Selección	Caracteres especiales, espacio de selección
Stock Actual	Cadena Numérico	Caracteres especiales, iniciado en 0
Stock Mínimo	Cadena Numérico	Caracteres especiales, iniciado en 0

Tabla 3. 22. Valores Limite – Registrar Producto

Fuente: Elaboración Propia

Entradas	**Código de Producto**	PVT-0003456
	Nombre del producto	**Retrovisor Lateral**
	Cantidad	**Unidades**
	Stock Actual	**20**
	Categoría	**Retrovisores**
	Marca	**Volvo**
Salida	"Ingrese los datos del producto"	"Se registró correctamente los datos del nuevo Producto"
Resultado	El sistema valida que no se deje en blanco los campos del Producto, el campo de descripción es opcional puede ser llenado o no.	Cuando el usuario introduce datos validos el sistema registra la información en la base de datos.

Tabla 3. 23. Prueba de Caja Negra Registrar Productos
Fuente: Elaboración Propia

Una vez realizado la prueba de caja negra a la interfaz de registro de producto se evidencia que la misma cumple con la función programada del registro de los datos del producto, obligando al usuario a registrar los campos obligatorios.

3.6.4. Pruebas de Funcionalidad

Esta prueba es necesaria para garantizar el funcionamiento del sistema, tomando en cuenta los casos de uso representativos del mismo. El uso de las pruebas funcionales es para asegurar correcto trabajo de entrada de datos, la navegación en el sistema, procedimientos y obtención de resultados.

PROCEDIMIENTO	DESCRIPCIÓN	VALOR
Prueba previa requerida	Registro de usuario	Si
Usuario	Administrador, vendedor.	
SECUENCIA DE PRUEBA		
PROCEDIMIENTOS	RESULTADOS ESPERADOS	CALIFICACIÓN DE FUNCIONALIDAD
Ingresa al sistema con el nombre de usuario y contraseña	Valida el sistema el ingreso	SI
FALLAS ENCONTRADAS	DESCRIPCIÓN	GRAVEDAD
Ninguna		-

	Pasos de Prueba	Resultados esperados	Pos.	Neg.
1	Desde la pantalla de login se ingresa al sistema con un usuario y contraseña.	El usuario ingresara al sistema si los datos son correctos, y según el grado de privilegios que tenga.	X	
2	Una vez que se ingresa de forma autentificada se comprueba que tenga acceso a todas las áreas que puede realizar según sus privilegios.	El usuario debe tener acceso a cada uno de las áreas según su privilegio.	X	
3	Los usuarios ingresan a la gestión de cuenta.	En la gestión de cuenta pueden cambiar su contraseña y usuario.	X	
4	El administrador puede registrar a un nuevo usuario.	El administrador debe tener acceso a la modificación de datos de empleado y de usuario del sistema.	X	

COMENTARIO DE LA PRUEBA REALIZADA
Las pruebas de ingreso al sistema y de gestión de usuario se efectuaron con absoluta normalidad. Se obtuvo el resultado esperado en cuanto a validación de usuario y clave, se mostraron alertas respuestas al ingresar con usuarios no registrados.

PROCEDIMIENTO	**DESCRIPCIÓN**	**VALOR**
Prueba previa requerida	Autentificado y con privilegios para el área	Si
Usuario	Administrador, Encargado de Ventas	

Tabla 3. 24. Caso de Prueba: Interfaz de Inicio de Sesión

Fuente: Elaboración Propia

SECUENCIA DE PRUEBA				
PROCEDIMIENTOS		**RESULTADOS ESPERADOS**	**CALIFICACIÓN**	
Registrar datos de nuevo producto y/o modificar datos de producto.		El sistema registre los datos añadidos o modificados	SI	
FALLAS ENCONTRADAS		**DESCRIPCIÓN**	**GRAVEDAD**	
Ninguna		Ninguna	-	
	Pasos de Prueba	**Resultados esperados**	**Pos.**	**Neg.**
1	Se prueba el registro de un nuevo Producto	Se inserta correctamente se actualiza en el panel de listado.	**X**	
2	Se elige un producto existente y se procede a editar los valores de su registro.	Posteriormente al cambio los valores son cambiado, y un mensaje de confirmación.	**X**	
3	Eliminación de un producto	Posteriormente a la eliminación los valores son cambiados y un mensaje de confirmación.	**X**	
COMENTARIO DE LA PRUEBA REALIZADA				
Las pruebas de registros de productos se efectuaron con absoluta normalidad. Se obtuvo el resultado esperado en cuanto al registro y modificación de usuario, se mostraron alertas de respuestas al modificar o agregar un muevo producto.				

Tabla 3. 25. Caso de Prueba: Registro de Productos
Fuente: Elaboración Propia

PROCEDIMIENTO	DESCRIPCIÓN	VALOR
Prueba previa requerida	Autentificado y con privilegios para el área	Si
Usuario		
SECUENCIA DE PRUEBA		
PROCEDIMIENTOS	**RESULTADOS ESPERADOS**	**CALIFICACIÓN DE FUNCIONALIDAD**
Proceso de compra y emisión de reportes de compra.	Sistema debe registrar los datos de nuevas compras y poder realizar emisión de reportes.	SI
FALLAS ENCONTRADAS	**DESCRIPCIÓN**	**GRAVEDAD**
Ninguna	**Ninguna**	**---**

	Pasos de Prueba	Resultados esperados	Pos.	Neg.
1	Se prueba el registro de un nueva compra.	Se debe registrar una nueva compra y el informe de que la operación se realizó con éxito.	**X**	
2	Reporte de compra.	Se muestra el reporte de las compras en formato pdf.	**X**	

COMENTARIO DE LA PRUEVA REALIZADA
Las pruebas de registro de nueva compra que realiza el encargado se va directamente hacia a los proveedores se imprime un formulario en donde se acredita el día de compra del producto con absoluta normalidad obteniendo el resultado esperado mostraron alertas de las acciones registradas.

PROCEDIMIENTO	DESCRIPCIÓN	VALOR
Prueba previa requerida	Autentificado y con privilegios para el área	Si
Usuario	Personal Encargado	

Tabla 3. 26. Caso de Prueba: Nueva Compra

Fuente: Elaboración Propia

SECUENCIA DE PRUEBA				
PROCEDIMIENTOS		**RESULTADOS ESPERADOS**	**CALIFICACIÓN DE FUNCIONALIDAD**	
Gestión de ventas y emisión de reportes de cotizaciones.			SI	
FALLAS ENCONTRADAS		**DESCRIPCIÓN**	**GRAVEDAD**	
Ninguna		Ninguna	-	
	Pasos de Prueba	**Resultados esperados**	**Pos.**	**Neg.**
1	Se prueba el registro de una nueva venta.		**X**	
2	Se elige la lista para obtener su reporte	Se muestra el reporte de ventas en formato pdf.	**X**	
COMENTARIO DE LA PRUEBA REALIZADA				
Las pruebas de registro de nuevas ventas y la gestión de cotización en cuando a la emisión de reportes por el encargado de autorizado se efectuaron con absoluta normalidad obteniendo el resultado esperado mostraron alertas de las acciones registradas.				

Tabla 3. 27. Caso de Prueba Ventas

Fuente: Elaboración Propia

PROCEDIMIENTO		DESCRIPCIÓN	VALOR	
Prueba previa requerida		Autentificado y con privilegios para el área	Si	
Usuario		Administrador, Personal encargado		
SECUENCIA DE PRUEBA				
PROCEDIMIENTOS		**RESULTADOS ESPERADOS**	**CALIFICACIÓN**	
Registrar datos de nuevo cliente, proveedor y/o modificar datos de producto e Registra nuevo producto.			SI	
FALLAS ENCONTRADAS		**DESCRIPCIÓN**	**GRAVEDAD**	
Ninguna		Ninguna	-	
	Pasos de Prueba	**Resultados esperados**	**Pos.**	**Neg.**
1	Se prueba el registro de un nuevo cliente, proveedor y venta y compra	Se debe registrar un nuevo cliente, proveedor, compra, venta y el informe de que la operación se realizó con éxito.	**X**	
2	Se elige un registro de cliente, proveedor y compra y , existente y se procede a editar los valores de su registro.	Posteriormente al cambio los valores son cambiado, y un mensaje de confirmación.	**X**	
3	Reporte de datos de clientes de proveedores y almacenes.	Lista de clientes, proveedores, y almacenes completa y oportuna.	**X**	
COMENTARIO DE LA PRUEBA REALIZADA				
Las pruebas de clientes, proveedores y almacenes se efectuaron con absoluta normalidad. Se obtuvo el resultado esperado en cuanto al registro y modificación, se mostraron alertas de respuestas al modificar o agregar un muevo registro.				

Tabla 3. 28. Caso de Pruebas: Cliente, Proveedores, Control de Compras y Ventas
Fuente: Elaboración Propia

CAPITULO IV

4. CALIDAD Y SEGURIDAD

4.1 CALIDAD

se hará la medición de calidad del software mediante la métrica de ISO 9126, que establece cualquier componente de la calidad de software puede ser descrito en términos de una de seis características básicas las cuales son: funcionalidad, confiabilidad, usabilidad, eficiencia, mantenimiento y portabilidad.

4.1.1. Funcionabilidad

Este atributo valora las características y capacidades del programa, la generalidad de las funciones entregadas y la seguridad del sistema global. La funcionalidad es el grado en que el sistema satisface las necesidades que indican los siguientes sub atributos: estabilidad, exactitud, interoperabilidad, cumplimientos de seguridad.

Punto función Para el cálculo de punto función se toma en cuenta cinco características, el dominio de información, como son números de entrada, salida, condiciones, archivos e interfaz externa. Luego se realiza el cálculo de punto de función hallando la suma de estas características, parámetros de medición y el factor de ponderación también llamado punto medio de ponderación.

Numero de entrada de usuarios: Se cuenta cada entrada de usuario que proporciona datos al sistema.

Número de salidas de usuario: Se refiere cada salida que proporciona el sistema al usuario. Entre estos pueden ser informes, reportes y mensajes advertencia, notificaciones y errores.

Numero de archivos: Se toma en cuenta cada archivo, estos pueden ser grupos lógicos de datos (tablas de base de datos).

Numero de interfaces externas: Se cuentan todas las interfaces legibles por la máquina. Aplicando lo anterior al proyecto se tiene los siguientes datos:

Parámetro de medición	Cuenta
Número de entradas de usuario	22
Número de salidas de usuario	14
Número de peticiones de usuario	18
Numero de archivos	48
Numero de interfaces externas	2

Tabla 4. 1. Parámetros de Medición

Fuente: Elaboración Propia

Para calcular el punto de función se tiene que realizar el cálculo de la cuenta total con los factores de ponderación especificados en la siguiente tabla:

Parámetro de medición	Cuenta	Factor	total
Número de entradas de usuario	24	4	96
Número de salidas de usuario	22	5	110
Número de peticiones de usuario	18	4	72
Numero de archivos	48	10	480
Numero de interfaces externas	2	7	14
Cuenta Total			772

Tabla 4. 2. Tabla de Putos de función no ajustados

Fuente: Elaboración Propia

En la tabla anterior se muestra la cuenta total que se obtiene de la sumatoria de los factores de ponderación a los parámetros de medición. Para determinar los valores de ajustes de complejidad se indica según se corresponda a las preguntas de la siguiente tabla:

Importancia	**0%**	**20%**	**40%**	**60 %**	**80%**	**100%**	**Fi**
Escala	No Influencia	Incidencial	Moderado	Medio	Significativo	Esencial	
Factor	0	1	2	3	4	5	
1. ¿Requiere el sistema copias de seguridad y de recuperación fiables?						X	5
2. ¿Se requiere comunicación de datos?					X		4
3. ¿Existen funciones de procesamiento distribuido?				X			3
4. ¿Es critico el rendimiento?				X			3
5. ¿Se ejecuta el sistema en un entorno operativo existente y fuertemente utilizado?					X		4
6. ¿Requiere el sistema entrada de datos interactiva?					X		4
7. ¿Requiere la entrada de datos interactiva que las transacciones de entrada se lleven a cabo sobre múltiples pantallas u operaciones?			X				2
8. ¿Se actualizan los archivos maestros de forma interactiva?			X				2
9. ¿Son complejos las entradas, las salidas, los archivos o peticiones?				X			3
10. ¿Es complejo el procesamiento interno?				X			3
11. ¿Se ha utilizado el código para ser reutilizable?				X			3
12. ¿Están incluidas en el diseño la conversión y la instalación?					X		4
13. ¿Se ha diseñado el sistema para soportar múltiples instalaciones en diferentes organizaciones?					X		4
14. ¿Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente utilizada por el usuario?					X		4
TOTAL							48

Tabla 4. 3. Parámetros de Medición
Fuente: Elaboración Propia

Fuente: Elaboración Propia

Calculando el punto de función mediante la siguiente ecuación:

$$\boldsymbol{PF = CuentaTotal * (0,65 + 0.1 * \sum Fi)}$$

Donde:

Cuenta total: es la suma del producto del factor de ponderación y valores de los parámetros.

$\sum \boldsymbol{Fi}$**:** es la sumatoria de los valores de ajuste de la complejidad.

Calculando:

$$\boldsymbol{PF} = 772 * (0{,}65 + 0.01 * 48)$$

$$\boldsymbol{PF} = 772 * 1.13$$

$$\boldsymbol{PF} = 872.36$$

Consideramos el máximo valor de complejidad $\sum \boldsymbol{Fi}$**=70** calculamos al 100% el nivel de confianza de la siguiente manera:

$$\boldsymbol{PF_{max} = CuentaTotal * (0,65 + 0.1 * \sum Fi)}$$

$$\boldsymbol{PF_{max}} = 772 * (0{,}65 + 0.01 * 70)$$

$$\boldsymbol{PF_{max}} = 772 * 1.35$$

$$\boldsymbol{PF_{max}} = 1042.2$$

La relación obtenida entre ambos es la funcionalidad:

$$\boldsymbol{Funcionalidad} = \frac{PF}{PF_{max}}$$

$$\boldsymbol{Funcionalidad} = \frac{872.36}{1042.2} = 0.837$$

$$\boldsymbol{Funcionalidad} = 0.837 * 100 = 83{,}7\%$$

Por lo que se concluye que la funcionalidad del sistema es un 83,7%, esto requiere decir que el sistema tiene un 84% de funcionar sin riesgos a fallar con operatividad constante y un 16% aproximadamente de colapso del sistema.

4.1.2. Confiabilidad

La confiabilidad del sistema se define como la probabilidad de operación libre de fallos de un programa o computadora.
Donde se encuentra:

$\boldsymbol{P(T \leq t)}$ Probabilidad de fallas (el termino en el cual sistema trabaja sin fallas)

$\boldsymbol{P(T \leq t)}$ **=1-** $\boldsymbol{F(t)}$ Probabilidad de trabajo sin fallas (Tiempo en el cual no falla el sistema)

Para calcular la confiabilidad del sistema se toma en cuenta el periodo de tiempo en el que se ejecuta y se obtiene muestras.

$$\boldsymbol{F(t) = f * e^{(-\mu * t)}}$$

Donde:

$\boldsymbol{f}$**:** Funcionalidad del sistema.
$\boldsymbol{\mu}$**:** Es la probabilidad de error que puede tener el sistema.
$\boldsymbol{t}$**:** Tiempo de duración de gestión en el sistema.

Para lo que consideramos un periodo de 20 días como tiempo de prueba donde se define que cada 10 ejecuciones se presenta una falla.

Calculando:

$$\boldsymbol{F(t) = f * e^{(-\frac{\mu}{10} * 20)}}$$

$$\boldsymbol{F(t) = 0,837 * e^{(-\frac{1}{10} * 20)}}$$

$$F(t) = 0,113 * 100 = 11,3\%$$

Reemplazando en las fórmulas de probabilidades:

$$P(T \leq t) = F(t) \quad \rightarrow \quad P(T \leq t) = 0,113 = 11,3\%$$

$$P(T \leq t) = 1 - F(t) \quad \rightarrow \quad P(T \leq t) = 1 - 0,113$$

$$P(T \leq t) = 0,887 = 88,7\%$$

Por lo tanto, la confiabilidad del sistema es el 89% en un periodo de 20 días como tiempo de prueba.

4.1.3. Usabilidad

Para conocer si el sistema satisface los requerimientos establecidos por el usuario, se realiza una evaluación del mismo en base a encuestas planteadas a los usuarios del sistema, los cuales califican en una ponderación al 100% los usuarios tienen conocimiento de los procesos que realizan y los resultados se refleja en la **Tabla 4.5**.

Para determinar la usabilidad del sistema se utiliza la siguiente ecuación:

$$FU = [\left(\sum \frac{Xi}{n}\right) * 100]$$

Donde:

Xi: Es la sumatoria de valores

n: Es el número de preguntas

Para responder a las preguntas se debe considerar la siguiente tabla:

Escala	Valor
Muy Bueno	5

Bueno	4
Regular	3
Malo	2
Pésimo	1

Tabla 4. 4. Parámetros de Medición

Fuente: Elaboración Propia

Nro	Preguntas	SI	NO	Evaluación
1	¿Puedo utilizar con facilidad él sistema?	5	0	1
2	¿Puedo controlar operaciones que el sistema solicite?	4	1	0,8
3	¿El sistema permitió la retroalimentación de información?	4	1	0,8
4	¿El sistema cuenta con interfaz amigable a la vista?	4	1	0,8
5	¿La respuesta del sistema es satisfactoria?	4	1	0,8
6	¿Le parecen complicadas las funciones del sistema?	3	2	0,6
7	¿Los resultados que proporciona el sistema facilitan el trabajo?	5	0	1
8	¿Durante el uso del sistema se produjo errores?	2	3	0,4
Total				6,2

Tabla 4. 5. Parámetros de Medición

Fuente: Elaboración Propia

Calculamos la usabilidad con la ecuación anterior:

$$\boldsymbol{FU = [(\frac{6,2}{8}) * 100]}$$

$$\boldsymbol{FU = [0.775 * 100]}$$

$$\boldsymbol{FU = 78\%}$$

Por lo tanto, existe un 78% de comprensión o entendimiento de los usuarios con respecto a la capacidad del sistema.

4.1.4. Mantenibilidad

El mantenimiento se da las modificaciones del sistema a los nuevos requerimientos según los usuarios de la empresa Barra.
Por lo que el índice de madurez del software (IMS)se determina con la siguiente ecuación:

$$IMS = \frac{[Mt-(Fa+Fc+Fd)]}{Mt}$$

Donde:

Descripción	Valor
Mt= Número de módulos de la visión actual	4
Fc= Número de módulos en la versión actual que se han modificado	1
Fa= Número de módulos en la versión actual que se han añadido	0
Fd= Número de módulos de la anterior versión que se han borrado en la versión actual.	0

Tabla 4. 6. Valores para Determinar la Mantenibilidad
Fuente: Elaboración Propia

Calculando:

$$IMS = \frac{[4-(1+0+0)]}{4}$$

$$IMS = 0.75 * 100 = 75\%$$

Por lo tanto, se puede decir que el sistema tiene un índice de mantenibilidad de 75% que es la facilidad de mantenimiento, el 25% restante es el margen de error corresponde a los cambios y modificaciones que se realizan al sistema.

4.1.5. Portabilidad

En este caso, se refiere a la habilidad del software de ser transferido de un ambiente a otro, y considera los siguientes aspectos:

- Adaptabilidad: Evalúa la oportunidad para adaptar el software a diferentes ambientes sin necesidad de aplicarle modificaciones.
- Facilidad de Instalación: Es el esfuerzo necesario para instalar el software en un ambiente determinado.
- Conformidad: Permite evaluar si el software se adhiere a estándares o convenciones relativas a portabilidad.
- Capacidad de reemplazo: Se refiere a la oportunidad y el esfuerzo usado en sustituir el software por otro producto con funciones similares.

El sistema fue desarrollado en PHP y sistemas operativos, y de la base de datos MariaDB, se ejecuta en todos los servidores web. También se comprobó que, en los distintos navegadores más usados en nuestra área, se le da una calificación del 95% de portabilidad. El resultado del 90% indica que el desenvolvimiento del sistema es correcto en los distintos navegadores.

4.1.6. Resultados

El factor de calidad total está directamente relacionado con el grado de satisfacción con el usuario que ingresa al sistema.

Características	Resultado
Funcionabilidad	84%
Confiabilidad	89%
Usabilidad	78%
Mantenibilidad	75%
Portabilidad	90%
Evaluación de calidad total	83 %

Tabla 4. 7. Resultados de Calidad del Software

Fuente: Elaboración Propia

El nivel de aceptación satisfactorio, indica que los valores de preferencia se encuentran en el rango de 60-100. La calidad del sistema corresponde al 83%, lo que se interpreta como la satisfacción que tiene un usuario al interactuar con el sistema.

4.2. Seguridad del Software

Para la seguridad del sistema se aplica la iso-27002 que evalúa y rectifica la implementación mediante el cumplimiento de las normas, así como la mejora continua de un conjunto de controles que permiten reducir el riesgo de sufrir incidentes de seguridad en el funcionamiento de la institución en cuanto a la seguridad de la información, para lo cual se tomó los siguientes tipos de seguridad:

4.2.1 Seguridad Lógica

- **Copias de seguridad:**

  Los respaldos (back-up) de la base datos, el sistema se deberá realizar de acuerdo a la siguiente tabla:

Descripción	Duración
En periodo de registro de productos	1 vez por semana
En periodo de registro de proveedores	1 vez por semana
El periodo de registro de clientes	1 vez por semana
En periodo de registros de compras	1 vez al día
En periodo de registro de ventas	1 vez al día
En periodo de registros de Entradas	1 vez al día
En periodo de registro de Salidas	1 vez al día

Tabla 4. 8. Copias de Seguridad
Fuente: Elaboración Propia

El personal que interviene en el proceso de ventas deberá cambiar el password del sistema periódicamente 1 vez cada 20 días o 1 vez al mes. En caso de ser usuario se recomienda cambiar el password periódicamente.

- **Identificación y autentificación:**
  Permite prevenir el ingreso de personas que no son usuarios, para ello el sistema cuenta con un control estricto en el ingreso con un Usuario y una contraseña estrictamente controlada.

- **Encriptación:**
  Se aplica la encriptación de seguridad para la contraseña, un dato de suma importancia para el ingreso al sistema. de este modo se está utilizando lo que es el algoritmo de SHA256 una encriptación de alta seguridad.

4.2.2. Seguridad Física

- **Seguridad física y del entorno**

  Se prevé recomendación de los back-up 3 o hacer copias que sean almacenadas en distintos lugares.

  Los back-up de la base de datos deberán ser protegidos en áreas seguras, además será permitido el acceso al personal autorizado.

- **Equipamiento**

  Una adecuada protección física y mantenimiento permanente de los equipos e instalaciones que conforman los activos de la empresa.

- **Control de acceso físico al área de Sistemas**

  Se restringe el acceso físico a las áreas críticas a toda persona no autorizada, para reducir el riesgo de accidentes fraudulentos.

4.2.3. Seguridad Organizativa

La información referente al sistema debe recibir un nivel de protección apropiada como ser:

- **Gestión de archivos**

  Etiquetar y manejar los Back-ups de acuerdo a la fecha en que se realizaron los mismos.

- **Recursos Humanos**

  Una vez que el personal técnico que interviene en el proceso concluya con el contrato de prestación de servicio en la empresa o por alguna razón fue despedido se deberá dar de baja el acceso al sistema.

CAPITULO V

5.1. ANÁLISIS COSTO Y BENEFICIO

La estimación de costo y beneficio son requeridos para el desarrollo de software. Se han producido varios modelos algorítmicos como base para estimar el esfuerzo, agenda y costes de un proyecto software, permite definir la factibilidad de las alternativas planteadas de un proyecto a ser desarrollado.

5.2. MÉTODO DE ESTIMACIÓN DE COSTO COCOMO II

La estimación de costos del sistema ha sido desarrollada bajo KLDC (Kilo-Líneas de código) como de detalle a continuación:

Como: **KLDC=LDC/1000**

KLDC=8732/1000

KLDC=8,732 KLDC

Por lo que la evaluación del sistema ha sido considerada bajo las 8,732 KLDC. Los coeficientes que se usaran los valores que se detallan en la siguiente tabla:

Proyecto de Software	a	b	c	d
Orgánico	2,4	1,05	2,5	0,38
Semi-acoplado	3,0	1,12	2,5	0,35
Empotrado	3,6	1,20	2,5	0,32

Tabla 5. 1. Coeficiente del COCOMO II
Fuente (S. Pressman,2010)

Ecuaciones para calcular el costo de Software:

Variable	Ecuación	Tipo/Unidad

Esfuerzo requerido por el proyecto	$E = a * (KLDC)^b * FAE$	Personas/Mes
Tiempo requerido por el proyecto	$T = c * (E)^d$	Meses
Número de personas requeridos para el proyecto	$NP = \frac{E}{T}$	Personas
Costo Total	$CT = SueldoMes * NP * T$	$us.

Tabla 5. 2. Ecuaciones del COCOMO II

Fuente: (Prentice- Haa, 1981)

Para hallar los valores de FAE se utiliza la tabla 5.3.

Atributos que afectan al Coste	Valor					
	Muy bajo	Bajo	Nominal	Alto	Muy alto	Extra alto
Atributos del Software						
Fiabilidad del software	0,75	0,88	1,00	1,15	1,40	
Tamaño base de datos		0,94	1,00	1,08	1,16	
Complejidad del producto	0,70	0,85	1,00	1,15	1,30	1,65
Atributos del Hardware						
Restricciones de tiempo de ejecución			1,00	1,11	1,30	1,66
Restricciones de memoria			1,00	1,06	1,21	1,56
Volatilidad de máquina virtual		0,87	1,00	1,15	1,30	
Tiempo de respuesta		0,87	1,00	1,07	1,15	
Atributos de Personal						
Capacidad de análisis	1,46	1,19	1,00	0,86	0,71	
Experiencia en la aplicación	1,29	1,13	1,00	0,91	0,82	
Capacidad de programadores	1,42	1,17	1,00	0,86	0,70	
Experiencia de S.O. usado	1,21	1,10	1,00	0,90		
Experiencia en el lenguaje de programación	1,14	1,07	1,00	0,95		
Atributos del proyecto						
Uso de técnicas actuales de programación	1,24	1,10	1,00	0,91	0,82	
Uso de herramientas de software	1,24	1,10	1,00	0,91	0,83	
Restricciones de tiempo de desarrollo	1,23	1,08	1,00	1,04	1,10	
TOTAL, FAE=	**0,431**					

Tabla 5. 3. Calculo de los Atributos FAE

Fuente (Elaboración Propia)

Aplicando las ecuaciones (descriptas en la **tabla 5.2**) así como los coeficientes a y c y los exponentes b y d que en nuestro caso el tipo orgánico será el más

apropiado ya que el número de líneas de código no supera los 50 KLDC, (descritos en la **Tabla 5.1**) y el cálculo de los atributos FAE (descrito en la **Tabla N.º 5.3**)

Se tiene:

Calculando el Esfuerzo:

$$E = a * (KLDC)^b * FAE$$
$$E = 2{,}4 * (8{,}732)^{1{,}05} * 0{,}431$$
$$E = 10{,}06\ Personas/Mes$$

Calculando el Tiempo:

$$T = c * (E)^d$$
$$T = 2{,}5 * (10{,}06)^{0{,}38}$$
$$T = 6{,}01\ Equivale\ a\ 6\ Meses$$

Calculando el Personal Promedio:

$$NP = \frac{E}{T}$$
$$NP = \frac{10{,}06}{6{,}01}$$
$$NP = 1{,}67\ \ Equivale\ a\ 2\ Personas$$

Calculando el Costo Total:

$$CT = SueldoMes * NP * T$$
$$CT = 450 * 2 * 6$$
$$CT = 5400\ \$us$$

Entonces se requiere estimando 2 personas un trabajo de 6 meses para el desarrollo del sistema con un costo total de 5400 $ dólares.

CAPITULO VI

6. CONCLUSIONES Y RECOMENDACIÓN

6.1. CONCLUSIONES

Finalizando el proyecto de grado Sistema Informático Web de Compras, Ventas y Control de Stock de Autopartes, se ha logrado alcanzar el objetivo principal planteado que indicaba el desarrollar un sistema web que mejore el proceso de compra y ventas de autopartes, cumpliendo con las necesidades de la empresa Barra. Tomando en cuenta los objetivos planteados se llega a las siguientes conclusiones:

Se logra mejorar el tiempo empleado en la atención de ventas de productos a los clientes, ya que se realiza este proceso de forma más automático y con este proceso poder evitar errores.

- ✓ Se logró mejorar el registro de los productos, se tiene las características detalladas de los mismos y así con estos registros mejorar la atención al cliente.

- ✓ Se logra disminuir los tiempos en la generación de reportes tanto de productos existentes y ventas, para así poder tener un mejor control del movimiento de la empresa.

- ✓ Se logró tener registro de los clientes y de esta manera poder tratar a cada cliente de manera personalizada, obteniendo información oportuna y en tiempo real de los mismos de los pedidos y compras que realizaron en la empresa.

- ✓ Se logró que el usuario pueda acceder al sistema con un determinado privilegio por medio de una autenticación donde debe registrar su nombre de usuario y contraseña.

- ✓ Los reportes pueden ser exportados, impresos y guardados en formato PDF.

6.2. RECOMENDACIONES

Para el mejor funcionamiento del presente proyecto se propone las siguientes recomendaciones.

- ✓ Se recomiendo a la empresa, implementar, utilizar y administrar el sistema de acuerdo las instrucciones brindadas.

- ✓ Capacitar a los nuevos usuarios para poder operar el sistema de forma correcta.

- ✓ Para resguardar la información, el administrador del sistema debe realizar copias de seguridad de la base de datos.

- ✓ Se recomienda mucha discreción en el manejo de sus usuarios y contraseñas ya que el sistema contiene información de mucha importancia.

- ✓ Mantener un control acerca del equipo que hace de servidor físico.

- ✓ La revisión periódica por cierto periodo de tiempo es recomendables para la eficiencia y un funcionamiento adecuado del sistema.

- ✓ Se debe realizar backups mensualmente de la base de datos.

- ✓ Se debe hacer revisión a los archivos **log** tanto del servidor web, la base de datos y del servidor de audio streaming, todo esto para que el disco duro del servidor no se colapse.

BIBLIOGRAFÍA

REFERENCIAS BIBLIOGRAFÍAS

Laudon, K.C. Y Laudon, J.P. (1996): Administración de los Sistemas de Información, Prentice Hall, México LA ADMINISTRACION EFICIENTE

Orlando Espinoza. (2011): LOS INVENTARIOS, EDITORIAL: LA ENSENADA, 1RA EDICION MADRID,

AUTOR Manuel Markel Guevara Oficina Económica y Comercial de España en Ciudad de México Buzón oficial de la Oficina: mexico@comercio.mineco.es Fecha: 29/05/2018Pressman, R.: Ingeniería de Software. Un Enfoque práctico. McGraw-Hill; 1993

ANDREU, R., RICART J. E. Y VALOR, J. (1991): Estrategia y Sistemas de Información. Mc Graw-Hill, Madrid

Acevedo Suarez (2006): La logística moderna en la empresa Editorial Logicuba.

FREDDY, M. M. (2008). *PRUEBAS DEL SOFTWARE CAJA BLANCA Y CAJA NEGRA*

McDermid, J. y P. Rook, (1993) "*Software Development Process Modelw"*, en *Softilwe Engineers Reference Book***,** CRC Press, pp. 15/26-15/28.

Garcia Marco, Francisco Javier. Servicios de informacion en la World Wide Web: relevancia, planificación y diseño. En Baro Queralt, j.; Cild Leal,P. (ed). Anuario SOCADI de Informacio y Documentacion:1997. Barcelona: SOCADI, 1997, 83-93.

R. Pressman, (2000) *"Software Engineering: A Practitioner´s Approach". 5th edition,* Mc Graw-Hill 2000. Chapter 29, "Web Engineering," pág. 813 – 843.

PRESSMAN, Roger S. “Ingeniería del software: Un enfoque práctico”, 6a edición, Mc Graw Hill. 2006

R. Pressman, (2003) “Ingeniería de Software un Enfoque Práctico” 6ta Edición MC Graw Hill, pp. 501-638

Roger S. Pressman*, (2005)* “*Ingeniería del Software*” 5o Edicion, Editorial Concepción Fernández Madrid, 2005.

Roger S. Pressman*, (2006)* “*Ingeniería de Software*” 6º Edición, Editorial McGraw-Hill, 2007.

Murugesan et al. (2001), *“Web Engineering: A New Discipline for Development of Web- Based Systems.”* Lecture Notes in Computer Science 2016 Springer 2001, pág. 3 – 13.

Olsina, L. et al.(1999) “Specifying Quality Characteristics and Attributes for Web Sites”, Proc. Ist ICSE Workshop on Web Engineering, ACM, Los Angeles, Mayo de 1999.

REFERENCIAS DE INTERNET

www4 (Daniela Rojas, 2018) Definición de Sistemas, http://sisteminfogth.blogspot.com/

www7 (GRUPOASTREA, 2015). Sistema de Inventarios, http://grupoastrea.mx/implementacion-de-sistemas-de-inventarios/

www5 (**Geraldyn A. Ramírez,2011**) definición de Sistemas, http://sisteminfogth.blogspot.com/

www15 (Wikipedia, 2018) Normas ISO 9000, https://es.m.wikipedia.org/wiki/ISO/IEC_9126

www16 (LuisMi Gracia, 2012) Modelo de Estimacion de costos COCOMO II**,** https://unpocodejava.com/2012/02/07/modelos-de- estimacion-un-poco-sobre-cocomo-ll/

www17 (William Pandini, 2018) Normas ISO 27002, https://ostec.blog/es//generico/iso-27002-buenas-practicas-gsi

Quiroga, A. (2015), Metodología UWE-UML (UML-Basad Web Engineering), http://proyectogradoingenieriasistemas. blogspot.com/2015/03/metodologia-uwe-uml-uml-based-web.html?m=1, (15/04/2017 20:30).

Anónimo (2016), Ventajas y desventajas de MYSQL, http://abcarticulos.info/article/ventajas-y-desventajas-de-mysql, (01/05/2017 21:39).