



Universidad de Cienfuegos
“Carlos Rafael Rodríguez”
Facultad de Ingeniería

Título: Tienda online de Cementos Cienfuegos S.A.

Trabajo de diploma para optar por el título de
Ingeniero Informático

Autor: Edgar Leonardo Llerena Martínez.

Tutores: Ing. Ángel Echevarría Zamora.

MsC. Rafaela Cristina Leal León.

Cienfuegos, Cuba

Curso: 2023

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconocer a la Universidad Cienfuegos los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

(Autor)

(Tutor)

(Tutor)

Dedicatoria

A mi familia por todo el apoyo brindado en todos estos años.

A mi hijo por ser mi inspiración durante todo el proceso.

A mi novia gracias por ayudarme tanto, gracias por la paciencia y la fuerza que me has dado en esta etapa final, te amo.

A mis amigos Leo, Ismael, Yinoby gracias por estar a mi lado en los buenos y malos momentos.

A mis tutores gracias por todo el tiempo y esfuerzo dedicado.

A mis profesores que fueron mi guía en la carrera, gracias por guiarme al mundo profesional.

A mis compañeros de estudio, a los viejos y a los nuevos, quienes me ayudaron y apoyaron en algunos momentos, con los que compartí una magnífica etapa de mi vida.

Agradecimientos

A mis padres por apoyarme todos estos años.

A mi hermana por ayudarme en todos los años.

A mi novia Melissa, por su comprensión, dedicación y paciencia en todo el proceso de tesis.

A mi tío Osmel Morales y todo el colectivo de la Fábrica de Cementos, Cementos Cienfuegos S.A. que me ofrecieron el tema a desarrollar.

A mis amigos Leo, Ismael, Yinoby, Gabriela, Lianna, etc. Un millón de gracias por sus consejos y acompañamiento.

A el Ing. Ángel Echevarría Zamora por haberme impulsado al mundo investigativo y por aportarme las herramientas en el desarrollo de la aplicación.

A todos los profesores que de una forma u otra formaron parte de mi formación como profesional.

A la MsC. Rafaela Cristina León por su tiempo y recomendaciones que me sirvieron de mucha ayuda.

Resumen

Las tiendas online, o tiendas en línea, son plataformas en internet que permiten a los usuarios realizar compras de productos y servicios. Las ventajas de estas, hacen que cada vez más personas prefieran realizar sus compras a través de internet. El desarrollo de una tienda online implica el uso de tecnologías de la información y comunicación (TIC) para la creación de un sistema de informatización que incluye componentes como servidores, redes de comunicación, software de comercio electrónico, herramientas de análisis de datos, sistemas de pago y envío, entre otros. Estas tecnologías permiten el procesamiento, almacenamiento, transmisión y gestión de la información relacionada con la tienda virtual. El objetivo del presente estudio es Desarrollar una aplicación web para la gestión de ventas de los productos realizados en Cementos Cienfuegos S.A., considerando que la implementación de esta aplicación web permitirá a Cementos Cienfuegos S.A. ampliar su alcance de mercado y llegar a más clientes potenciales, lo que podrá aumentar sus ventas y con ello su entrada de divisas extranjeras. Para ello, se abordaron las principales concepciones teóricas asociados al dominio del problema, la cual contiene una descripción detallada de la metodología aplicada y se presentaron además un estudio sobre las principales tecnologías, lenguajes, y herramientas actuales.

Se planteó y se describió la propuesta de solución y los mecanismos generados por el escenario número 4 en la metodología Agile Iniciad Procesos (AUP) en las etapas de análisis, diseño e implementación. Y, por último, se valora la viabilidad de la propuesta de solución mediante la validación de los requisitos y el diseño propuesto, además de la realización de pruebas funcionales, unitarias y de aceptación a la aplicación web. Lo que lleva a concluir que con el desarrollo de la aplicación web ofrece a los clientes la comodidad de realizar sus compras desde cualquier lugar o dispositivo con acceso a internet. Además, que permite acceder a una amplia variedad de productos, comparar precios, y aprovechar ofertas y descuentos. Todo lo anterior, facilita la gestión de ventas de Cementos Cienfuegos S.A.

Palabras Claves: tienda online, aplicación web, laspailas.com, Cementos Cienfuegos S.A.

Abstract

Online stores, or online stores, are Internet platforms that allow users to make purchases of products and services. The advantages of these make more and more people prefer to make their purchases online. The development of an online store involves the use of information and communication technologies (ICT) for the creation of a computerization system that includes components such as servers, communication networks, electronic commerce software, data analysis tools, information systems. payment and shipping, among others. These technologies allow the processing, storage, transmission and management of information related to the virtual store. The objective of this study is to develop a web application for sales management of products made at Cementos Cienfuegos S.A., considering that the implementation of this web application will allow Cementos Cienfuegos S.A. expand your market reach and reach more potential customers, which can increase your sales and thus your income of foreign currencies. To this end, the main theoretical concepts associated with the problem domain are addressed, which contains a detailed description of the applied methodology and a study on the main current technologies, languages and tools is also presented.

The solution proposal and the mechanisms generated by scenario number 4 in the Agile Initiate Processes (AUP) methodology were proposed and described in the analysis, design and implementation stages. And, finally, the viability of the solution proposal is assessed by validating the requirements and the proposed design, in addition to performing functional, unit and acceptance tests on the web application. Which leads to the conclusion that with the development of the web application it offers customers the convenience of making purchases from any place or device with internet access. In addition, it allows you to access a wide variety of products, compare prices, and take advantage of offers and discounts. All of the above facilitates the sales management of Cementos Cienfuegos S.A.

Keywords: online store, web application, laspailas.com, Cementos Cienfuegos S.A.

Índice

Introducción	8
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.....	16
1.1 Sistema de Informatización	16
1.2 Soporte para sistemas informáticos	16
1.3 Metodología de desarrollo de software.....	17
1.3.1 Metodología AUP	18
1.4 Arquitectura de software	20
1.4.1 Estilos arquitectónicos.....	21
1.4.2 Framework de desarrollo.....	23
1.4.2.1 Django.....	24
1.4.3 Bootstrap.....	25
1.5 Paradigmas de programación	26
1.5.1 Programación orientada a Objetos (POO).....	27
1.6 Herramientas CASE.....	28
1.6.1 Visual Paradigm for UML	28
1.7 Lenguajes de Programación	29
1.7.1 Python.....	30
1.7.2 JavaScript	31
1.8 Entorno de desarrollo integrado	31
1.8.1 Visual Studio Code IDE.....	33
1.9 Sistema Gestor de Base de Datos	34
1.9.1 MySQL.....	35
1.10 Servidor web	35
1.10.1 Apache 2.4.48.....	36
1.11 Conclusiones parciales	37
CAPÍTULO 2: DESCRIPCIÓN DE LA PROPUESTA DE SOLUCIÓN:	38
2.1 Análisis del proceso de la tienda online	38
2.2 Requisitos de software	39
2.2.1 Requisitos Funcionales (RF).....	39
2.2.2 Requisitos no funcionales (RNF).....	41
2.3 Historias de usuarios.....	43
2.4 Descripción de la arquitectura.....	45

2.5 Diagrama de clases diseño	47
2.6 Patrones de diseño utilizados.....	48
2.7 Modelo de datos	50
2.8 Estándar de codificación empleado.....	51
2.9 Conclusiones parciales.	51
CAPÍTULO 3: VALIDACIÓN DE LA VIABILIDAD DE LA PROPUESTA DE SOLUCIÓN.....	52
3.1 Técnicas de validación de requisitos.....	52
3.2 Métricas aplicadas a los requisitos.....	52
3.3 Validación del diseño	53
3.4 Pruebas de software	61
3.4.1 Pruebas de caja blanca.....	63
3.4.2 Pruebas de caja negra	66
3.5 Validación de la solución propuesta	68
3.6 Conclusiones Parciales	71
Conclusiones Generales	72
Recomendaciones	74
Referencias Bibliográficas.....	75

Introducción

El comercio electrónico, también conocido como eCommerce, se refiere a la compra y venta de bienes y servicios a través de Internet. La historia del comercio electrónico se remonta a la década de 1960, cuando se establecieron las bases para una red global de computadoras. A medida que la tecnología avanzaba, las empresas comenzaron a utilizar métodos de comercio electrónico para realizar transacciones en línea. ⁽¹⁾

En la década de 1990, el comercio electrónico comenzó a ganar popularidad con el lanzamiento de sitios web de comercio electrónico. Estos sitios permitían a los consumidores comprar productos en línea y recibirlos por correo. El pago se realizaba en línea mediante tarjetas de crédito y sistemas de pago en línea. ⁽²⁾

En los años siguientes, el comercio electrónico evolucionó a medida que se desarrollaban nuevas tecnologías y plataformas de comercio electrónico. Las tiendas en línea se volvieron más sofisticadas y personalizadas, permitiendo a los usuarios crear perfiles y recibir recomendaciones de productos basadas en su historial de compras. ⁽²⁾

Con la llegada de smartphones y tabletas, el comercio electrónico se ha vuelto aún más accesible. Los consumidores pueden realizar compras en línea desde cualquier lugar y en cualquier momento utilizando sus dispositivos móviles. Además, las redes sociales se han convertido en una plataforma importante para el comercio electrónico, ya que las empresas pueden vender productos directamente a través de sus perfiles de redes sociales.

Podemos decir que la precursora del comercio electrónico y la primera actividad comercial que abrió el juego a la diversificación de canales fue la venta por catálogo. A fines del siglo XX, en los Estados Unidos, esta práctica cobra popularidad, y muchos vendedores comienzan a enviar a los domicilios de potenciales clientes catálogos de productos con imágenes, precios y descripciones atractivas. ⁽⁶¹⁾

Cabe destacar que en la década de 1970 aparece el intercambio de datos electrónicos (EDI), lo que permitió a muchas empresas realizar facturas y transacciones financieras de manera electrónica, mejorando la gestión del negocio y su competitividad en el mercado. ⁽⁶¹⁾

Si bien la venta por catálogo funcionaba bien, hay otro antecedente que potenció aún más la compra de forma remota: el Teleshopping. Fue en los años '80 cuando la televisión se volvió una herramienta fundamental para la venta de productos y/o servicios. Este canal les permitía a los consumidores obtener una imagen más realista sobre lo que se estaba vendiendo, podían comprarlo en el momento con su tarjeta de crédito llamando al número que aparecía en la pantalla, y recibirlo en sus hogares a los pocos días.

Con la creación de la World Wide Web en los '90, la historia del e-commerce abrió un nuevo capítulo. Internet estandarizó el comercio electrónico y convirtió la venta digital en algo cotidiano tanto para empresas como consumidores.

La creación de las primeras páginas web les permitió a los comerciantes dar a conocer sus marcas a nuevos usuarios con recursos nunca antes imaginados. Las empresas podían mostrar sus productos y/o servicios con diferentes diseños y jerarquías de texto, contenidos audiovisuales y otras ventajas que ayudaban a aumentar sus ventas a nivel global.

Además, en esa época la tecnología detrás de la “vidriera” que era la tienda online también había evolucionado. El correo electrónico fue fundamental para la comunicación entre empresas y consumidores, además de herramientas de facturación electrónica para la automatización del proceso de compra y la gestión del cobro.

En 1995 nacen las dos compañías más relevantes en el mundo del comercio electrónico: eBay y Amazon. Y tres años después entra en el juego PayPal, fundamental para el desarrollo del pago electrónico en dichas plataformas.

Para el cambio de siglo, el comercio electrónico se estaba expandiendo a niveles inimaginables. En 2001 Amazon lanza su e-commerce Mobile, y con la llegada de la Web 2.0 en 2005, las plataformas se convirtieron más interactivas y “amigables” para el consumidor.

Hoy, con la innovación de la Web 3.0, las constantes actualizaciones de software y optimizaciones UX, el e-commerce es una de las plataformas más importantes del comercio electrónico. La posibilidad de poder acceder desde cualquier dispositivo

móvil a una tienda online y la sencillez del proceso de compra significa un gran porcentaje de las ventas anuales de muchas empresas.

Esto no quita que los comerciantes tengan todo resuelto. Cada cambio en las plataformas y en el comportamiento del consumidor significan nuevos desafíos que deben enfrentar con tecnología. Por suerte para muchos, en Gtd contamos con herramientas que pueden ayudar a superar esos retos. Descubre nuestras soluciones digitales para la gestión del negocio aquí. ⁽⁶¹⁾

Como antecedentes a la creación de tiendas online cubanas de ventas mayoristas y minoristas, están: TuEnvio. Ante el surgimiento y auge del comercio electrónico en Cuba, en un contexto de pandemia de la COVID-19, muchos fueron los que comenzaron a utilizar la propuesta estatal TuEnvio ⁽³⁾. Esta plataforma digital permitió en su inicio que miles de personas pudieran acceder a productos en tiendas online cubanas con su correspondiente envío a domicilio.

En la actualidad el comercio electrónico ha evolucionado hasta convertirse en una industria masiva y global. Se espera que siga creciendo y evolucionando a medida que se desarrollen nuevas tecnologías y plataformas de comercio electrónico.

Una tienda online es un sitio web para vender que cuenta con una especie de catálogo virtual de productos e integraciones para recibir pagos. A diferencia de otras plataformas de venta, en una tienda virtual con carrito de compras el vendedor tiene contacto directo con el comprador y también tiene el control total sobre la gestión de sus ventas ⁽⁴⁾.

Una tienda online debe reunir las siguientes características ⁽⁵⁾:

- **Productos:** con sus fotografías, características y precios. Lo ideal es que estén posicionados y separados por categorías o secciones.
- **Formas de pago:** lo deseable es que se ofrezcan todas las formas de pago que resulten posibles e incluso se dé la posibilidad de vender en otras monedas para aproximarse a un mercado global.
- **Carro de la compra:** contar con un buen carro de la compra y vistoso es esencial para hacer que las ventas se produzcan de forma más rápida y segura.

- **Registro de usuarios:** es muy recomendable de cara a la fidelización crear sistemas de usuarios. Sin embargo, muchas empresas que usan software de creación de tiendas online no los utilizan.
- **Valoraciones y comentarios:** Se está en la web 2.0 y cada producto debería ofrecer la posibilidad de comentarse, compartirse en las redes sociales y valorarse con estrellas de 0 a 5.
- **Diseño atractivo:** una buena página principal y un diseño atractivo es crucial para vender por Internet. Disponer de herramientas como un «slider» o diapositivas ayudan mucho a las ventas. También una adecuada disposición de los productos.

La tienda online ofrece numerosas ventajas para los consumidores y los propietarios de negocios. Algunas de las ventajas más destacadas son ⁽⁶⁾:

1. **Comodidad:** los clientes pueden comprar desde cualquier lugar con acceso a internet, sin tener que desplazarse físicamente a la tienda.
2. **Disponibilidad 24/7:** la tienda online está abierta todo el tiempo, lo que significa que los clientes pueden hacer compras en cualquier momento, incluso en días festivos o fuera del horario comercial.
3. **Mayor alcance:** una tienda online puede llegar a clientes de todo el mundo, lo que aumenta significativamente el potencial de ventas.
4. **Costo reducido:** el costo de operar una tienda online suele ser menor que el de una tienda física, ya que no hay gastos asociados con el alquiler de un local, la luz y el agua, entre otros.
5. **Personalización:** las tiendas online pueden personalizar la experiencia de compra de los clientes, ofreciendo recomendaciones y promociones basadas en su historial de compras y preferencias.
6. **Análisis de datos:** las empresas pueden recopilar y analizar datos sobre el comportamiento de los clientes en la tienda online, lo que les permite mejorar la experiencia del usuario y tomar decisiones comerciales informadas.

7. Flexibilidad: una tienda online es fácil de actualizar y modificar según las necesidades del negocio, lo que permite una mayor flexibilidad en comparación con una tienda física.

La Empresa de Cementos Cienfuegos S.A. (en lo adelante CCSA) se encarga de la producción de cemento y su comercialización. Fundada en 1980 ha sido sometida, de entonces a la fecha, a diversos procesos de mantenimiento y mejoras, siendo hoy la mayor productora de cemento del país y una de las mayores de América Latina ⁽⁷⁾.

En los últimos años la Empresa CCSA, como el resto de país, ha tenido limitaciones materiales y financieras debido a diversos factores a nivel mundial como propios del país, conocidos ampliamente. Esta situación ha llevado a la Empresa a plantearse la necesidad de adquirir divisas para poder continuar de la mejor forma el proceso productivo, por lo que urge buscar mecanismos y vías para lograr ese objetivo.

Actualmente el comercio electrónico puede ser una forma más viable para los clientes, y potenciales clientes, de obtener productos, específicamente cemento en este caso, pues, entre otros factores, prácticamente se elimina la distancia geográfica entre proveedores y clientes, ya que de lo contrario tendrían que ir directamente a la Empresa a realizar los trámites correspondientes para su compra, lo que supone también gasto de combustible o pasaje. Además, da la posibilidad de que puede ser utilizado por posibles clientes internacionales que estén interesados en importar a su país el cemento cubano, de una forma fácil y sencilla.

Por todo lo anteriormente expuesto se plantea como problema del presente trabajo:

Problema: ¿Cómo contribuir al logro de una mayor adquisición de divisas extranjeras en Cementos Cienfuegos S.A.?

Objeto de estudio: La gestión de ventas de productos online.

Objetivo general: Desarrollar una aplicación web para mejorar la gestión de ventas de los productos realizados en Cementos Cienfuegos S.A.

Objetivos específicos:

- Diseñar una arquitectura que responda a las características propias del trabajo, así como la interfaz gráfica de la aplicación.
- Implementar el diseño propuesto para la aplicación y los requerimientos funcionales.
- Validar resultados del desarrollo de la aplicación web.

Campo de acción: Ventas online de productos en Cementos Cienfuegos S.A.

Idea a defender: La implementación de una aplicación web permitirá a Cementos Cienfuegos S.A. ampliar su alcance de mercado y llegar a más clientes potenciales, lo que podrá aumentar sus ventas y con ello su entrada de divisas extranjeras.

Tareas a cumplir para el desarrollo de la investigación:

- Conceptualización de los aspectos teóricos.
- Caracterización de los sistemas de gestión de ventas online.
- Definición de las herramientas y tecnologías utilizadas.
- Caracterización de la propuesta para la creación de una aplicación web que permita mejorar la gestión de ventas online.
- Descripción del Modelo de Negocio.
- Definición de los requisitos funcionales y no funcionales.
- Representar la Arquitectura de la aplicación web.
- Implementación de la aplicación web que permitan la gestión de los productos, el proceso de compra y venta de los mismos y la gestión de usuarios.
- Implementación de la interfaz visual y el diseño de la aplicación.
- Realización de las pruebas correspondientes.
- Evaluación del resultado del proceso de pruebas de software.

Métodos lógicos:

- Métodos lógicos formales:

Entre los métodos lógicos formales, se podrían considerar el uso de técnicas de modelado como UML (Lenguaje de Modelado Unificado) ⁽⁸⁾.

Las técnicas de modelado son un conjunto de herramientas y métodos que se utilizan para representar, visualizar, especificar y documentar los diferentes aspectos de un sistema. UML es una técnica de modelado ampliamente utilizada en la industria del software para crear modelos visuales de sistemas de software.

UML proporciona un conjunto de diagramas que se pueden utilizar para representar diferentes aspectos de un sistema, como la estructura estática, la interacción dinámica entre los objetos y los procesos, la arquitectura del sistema y su despliegue en diferentes entornos.

Cada tipo de diagrama de UML se utiliza para representar diferentes aspectos del sistema y permite a los desarrolladores y otros interesados en el proyecto comprender mejor cómo funciona el sistema y cómo se relacionan sus diferentes componentes ⁽⁹⁾.

Algunas de las técnicas de modelado que se utilizarán con UML son ⁽⁹⁾:

1. Diagramas de casos de uso: se utilizan para identificar las funciones principales del sistema y cómo interactúan con los usuarios.
2. Diagramas de clase: se utilizan para representar la estructura estática del sistema en términos de clases, atributos y relaciones entre ellas.
3. Diagramas de secuencia: se utilizan para modelar la interacción dinámica entre los objetos y sus procesos en el sistema a lo largo del tiempo.
4. Diagramas de actividad: se utilizan para modelar el flujo de trabajo o proceso de negocio dentro del sistema.
5. Diagramas de componentes: se utilizan para modelar la arquitectura del sistema y sus componentes principales.
6. Diagramas de despliegue: se utilizan para modelar la distribución física del sistema en diferentes entornos de hardware.

- **Métodos empíricos:**

Entre los métodos empíricos a utilizar están la encuesta que es el instrumento más utilizado para recolectar datos, consiste en un conjunto de preguntas respecto a una o más variables a medir. Se utilizará para obtener información sobre los intereses, necesidades y expectativas de los clientes y además de los directivos de la empresa ⁽¹⁰⁾.

Las entrevistas en profundidad buscan la libre manifestación, por parte de los actores sociales, de sus intereses informativos, creencias y deseos. En este caso se emplearán para obtener información sobre la gestión de las ventas de los productos que brinda ⁽¹¹⁾.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo se abordan los principales conceptos teóricos asociados al dominio del problema, contiene una descripción detallada de la metodología aplicada y se presenta además un estudio sobre las principales tecnologías, lenguajes, y herramientas actuales.

1.1 Sistema de Informatización

Las Tecnologías de la Información y las Comunicaciones (TIC) pueden ser utilizadas como un sistema de informatización para el diseño y gestión de una tienda virtual.

El sistema de informatización basado en TIC para una tienda virtual puede incluir componentes como servidores, redes de comunicación, software de comercio electrónico, herramientas de análisis de datos, sistemas de pago y envío, entre otros. Estos componentes se combinan para crear un sistema que permite el procesamiento, almacenamiento, transmisión y gestión de información relacionada con la tienda virtual ⁽¹²⁾.

La implementación de un sistema de informatización basado en TIC puede tener muchos beneficios para una empresa, como una mayor eficiencia y productividad, una mejor toma de decisiones empresariales, una reducción de los errores humanos, una mayor transparencia en los procesos y una mejor interacción con agentes externos como proveedores y clientes, lo cual también posibilita nuevos negocios en la red y mejora los existentes ⁽¹³⁾.

Las TIC son fundamentales en el diseño y gestión de una tienda virtual, permitiendo una mejor gestión y aprovechamiento de la información, así como una mayor eficiencia en la venta y distribución de los productos o servicios.

1.2 Soporte para sistemas informáticos

El soporte para sistemas informáticos de una tienda virtual es crucial para garantizar que el sitio web funcione correctamente, se mantenga actualizado y sea seguro para los usuarios. Es importante contar con un proveedor confiable y experimentado en

el campo del comercio electrónico para brindar un servicio de calidad y satisfacer las necesidades específicas de la tienda virtual ⁽¹⁴⁾.

1.3 Metodología de desarrollo de software

Las metodologías de desarrollo de software son planes estructurados que utilizan herramientas, técnicas y modelos para llevar a cabo un proyecto. Se documentan para que los programadores comprendan la metodología y ciclo de vida del software. Las metodologías ofrecen una garantía de calidad y permiten estimar y controlar los costos de desarrollo del proyecto, mediante la organización del proceso y la revisión de resultados. Existen dos tipos principales: las ágiles, más flexibles y adaptativas, y las tradicionales, más controladas y orientadas al análisis y diseño previo ⁽¹⁵⁾.

Metodologías ágiles:

Las metodologías ágiles son especialmente adecuadas para proyectos que están preparados para cambios y tienen un proceso menos controlado. Algunas características comunes incluyen tener un cliente parte del equipo de desarrollo, grupos pequeños trabajando juntos y menos énfasis en la arquitectura del software. Las metodologías ágiles suelen obviar gran parte de la documentación y se adaptan a proyectos con requisitos imprecisos y cambiantes ⁽¹⁶⁾.

Ejemplos de metodologías ágiles incluyen:

1. Extreme Programming (XP): Se enfoca en la calidad del software y la satisfacción del cliente mediante ciclos cortos de desarrollo, pruebas continuas, integración continua y una comunicación constante entre el cliente y el equipo de desarrollo.
2. SCRUM: Se enfoca en la entrega temprana y frecuente de software funcional, con reuniones diarias de seguimiento y planificación, iteraciones cortas de trabajo y una colaboración estrecha entre el equipo de desarrollo y el cliente.
3. Crystal Methodologies: Son una familia de metodologías ligeras que se adaptan a las necesidades específicas del proyecto y utilizan principios como la comunicación constante, la simplicidad y la retroalimentación continua para lograr una entrega rápida y eficiente del software.

4. Metodología AUP:

Metodologías tradicionales

Las metodologías tradicionales son recomendadas para proyectos con cierta resistencia al cambio y un proceso más controlado, con numerosas políticas y normas. Suelen tener un contrato prefijado con el cliente y la interacción del cliente es mediante reuniones. También se adaptan mejor a grupos grandes y posiblemente distribuidos, y dan mayor importancia a la arquitectura del software, expresada mediante modelos ⁽¹⁷⁾.

En las metodologías tradicionales se genera mucha documentación antes de implementar el proyecto, y se enfatiza en el análisis y diseño previo.

Ejemplos de metodologías tradicionales incluyen:

1. Rational Unified Process (RUP): Es una metodología orientada al proceso que se enfoca en la planificación detallada, el análisis y diseño previo, la documentación exhaustiva y la gestión rigurosa del riesgo.
2. Métrica 3: Es una metodología de desarrollo de software basada en el análisis y el diseño previo, la gestión rigurosa del riesgo y la generación de documentación detallada en cada etapa de desarrollo. Se enfoca en la entrega de software de alta calidad y bien estructurado.

1.3.1 Metodología AUP

El Proceso Unificado Ágil (AUP por Agile Unified Process en Inglés) es un enfoque de desarrollo de software derivado de RUP que fue desarrollado por Scott Ambler desde 2002 hasta 2006 y combina los principales artefactos de RUP. AUP combina el Modelo de Negocio, Modelo de Casos de Uso, Modelo de Análisis y Diseño de RUP en un solo Modelo ⁽¹⁸⁾.

AUP es una metodología de desarrollo de software que se caracteriza por ser interactiva e incremental. Esto significa que los proyectos importantes se dividen en pequeños proyectos, lo que permite tener un mayor control sobre el proceso y solucionar problemas más rápidamente. Cada pequeña parte del proyecto se considera una interacción y se enfoca en asegurar que el sistema cumpla con los requerimientos del usuario. Los casos de uso son la guía para todas las actividades

del desarrollo de software. En general, AUP toma un enfoque iterativo y centrado en el usuario para desarrollar productos de software de alta calidad ⁽¹⁹⁾.

Principios

AUP se basa en principios de simplicidad, agilidad y centrarse en actividades de alto valor. Se ajusta a la Alianza Ágil y sus principios para satisfacer al cliente entregando software con valor y aceptando cambios en los requisitos. La entrega frecuente de software funcional es prioritaria y el equipo de negocio y desarrolladores trabajan juntos durante todo el proyecto. Los procesos son iterativos y promueven la excelencia técnica y un diseño eficiente ⁽¹⁸⁾.

AUP tiene cuatro fases ⁽¹⁸⁾⁽¹⁹⁾: inicio, elaboración, construcción y transición:

- La fase de inicio se enfoca en establecer el alcance del proyecto y obtener la aceptación de las partes interesadas.
- En la fase de elaboración se prueba la arquitectura del sistema.
- La fase de construcción se enfoca en producir entregables del sistema priorizando las necesidades de los interesados.
- La fase de transición valida y despliega el sistema en el ambiente de producción.

Disciplinas

La metodología AUP (Proceso Unificado Ágil) incluye varias disciplinas que guían el desarrollo de software. Estas disciplinas son ⁽¹⁸⁾⁽²⁰⁾⁽²¹⁾:

Modelado: Comprender el negocio de la organización y transformar este entendimiento en un modelo que sirva como base para el desarrollo del software.

Implementación: Transformar el modelo en código ejecutable y realizar pruebas a nivel básico, incluyendo pruebas unitarias.

Pruebas: Realizar una evaluación objetiva para asegurar la calidad del software, incluyendo la detección de defectos.

Despliegue: Ejecutar el sistema y ponerlo a disposición de los usuarios finales.

Gestión de Configuración: Gestionar los cambios en el software y en la configuración del entorno de desarrollo.

Gestión de Proyecto: Gestionar el proyecto de desarrollo de software, incluyendo la planificación, seguimiento y control del mismo.

Entorno: Proporcionar el entorno adecuado (herramientas, estándares, guías, etc.) para el desarrollo del software.

Roles

Los roles no deben ser tomados como puestos de trabajos, ya que una persona puede realizar varios roles y un rol puede ser asumido por varias personas.

1.4 Arquitectura de software

La arquitectura de software se refiere a la estructura y organización del código y componentes de una aplicación. Una buena arquitectura de software debe promover modularidad, reutilización, escalabilidad y facilidad de mantenimiento. Existen varios patrones de diseño comunes en la arquitectura de software, como el Modelo-Vista-Controlador (MVC), que separa las responsabilidades de la aplicación en tres partes principales: el modelo, la vista y el controlador. En general, una arquitectura de software bien diseñada puede hacer que una aplicación sea más eficiente, efectiva y fácil de desarrollar y mantener ⁽²²⁾.

La arquitectura de software se refiere a la estructura o diseño del software, que define cómo los diferentes componentes del software interactúan entre sí para lograr los objetivos del sistema. Se debe considerar toda la estructura del software, incluyendo las decisiones de diseño, la organización del código, la selección de tecnologías y plataformas, los patrones de interacción de los usuarios y los procesos de negocio subyacentes ⁽²²⁾.

La arquitectura de software se divide en varias capas, cada una con su propia función. La capa de presentación, por ejemplo, es responsable de la interfaz de usuario, mientras que la capa de lógica de negocios maneja los datos y las reglas empresariales detrás del software. La capa de persistencia maneja la forma en que los datos se almacenan y recuperan del sistema ⁽²²⁾.

Una buena arquitectura de software debe ser escalable, flexible y fácil de mantener. Debe ser capaz de adaptarse a las necesidades cambiantes del negocio y permitir

actualizaciones y mejoras sin tener que reescribir todo el código. Además, la arquitectura del software debe ser segura y resistente a los ataques y vulnerabilidades, protegiendo los datos y la privacidad de los usuarios.

1.4.1 Estilos arquitectónicos

Los estilos arquitectónicos describen diferentes categorías de sistemas de software que contienen componentes y conectores que permiten la comunicación y la cooperación entre ellos. Además, tienen restricciones que definen cómo se pueden integrar los componentes y modelos semánticos que ayudan al diseñador a entender las propiedades globales del sistema ⁽²³⁾.

Estos estilos sirven para sintetizar soluciones estructurales y abstraer una gran variedad de configuraciones concretas en patrones posibles de aplicaciones. Asimismo, son útiles para evaluar diferentes arquitecturas alternativas con ventajas y desventajas conocidas ante distintos conjuntos de requerimientos no funcionales. Entre los estilos arquitectónicos más comunes se encuentran ⁽²³⁾:

Estilo de Flujo de Datos

El estilo de arquitectura de flujo de datos se enfoca en la forma en que los datos se mueven a través del sistema y cómo se transforman por medio de diferentes procesos. Este estilo se basa principalmente en un modelo de procesamiento en el cual los componentes del sistema son responsables de recibir y transformar los datos que fluyen a través de ellos.

En este estilo arquitectónico, los componentes se dividen en tres tipos principales: fuentes de datos, transformaciones de datos y sumideros de datos. Las fuentes de datos son responsables de proporcionar datos al sistema, mientras que las transformaciones de datos procesan y manipulan los datos. Los sumideros de datos reciben los datos transformados y los envían fuera del sistema.

Estilos Centrados en Datos

El estilo arquitectónico centrado en datos se enfoca en cómo los datos son almacenados, accedidos y manipulados por el sistema de software. Este estilo se basa en la idea de que los datos son el elemento principal del sistema y todos los demás componentes y procesos están diseñados para trabajar con ellos.

En este estilo arquitectónico, los componentes se organizan en capas, donde cada capa representa una forma diferente de acceso a los datos. Por ejemplo, la capa de presentación maneja la interfaz de usuario y las solicitudes de entrada de datos, mientras que la capa lógica de negocios maneja la lógica de procesamiento y los cálculos relacionados con los datos.

Estilos de Llamada y Retorno

El estilo arquitectónico de llamada y retorno se enfoca en cómo los diferentes componentes del sistema interactúan a través de la comunicación de mensajes. Este estilo se basa en la idea de que los componentes se dividen en dos categorías distintas: aquellos que llaman a un componente para realizar una tarea y aquellos que proporcionan una respuesta o un valor de retorno para esa tarea.

En este estilo arquitectónico, los componentes se organizan en una jerarquía de llamadas, donde los componentes superiores llaman a los inferiores para hacer tareas específicas y esperan una respuesta antes de continuar. Los conectores utilizados en este estilo arquitectónico se centran en la transferencia de mensajes entre los componentes, y pueden ser de diferentes tipos, como métodos, protocolos de comunicación o servicios web.

Es muy útil para sistemas distribuidos o para sistemas con múltiples procesos, ya que permite la comunicación y coordinación entre los componentes de manera eficiente y escalable. Además, este estilo promueve la modularidad y el acoplamiento débil, lo que facilita el mantenimiento y la evolución del sistema.

Estilos de Código Móvil

El estilo arquitectónico de código móvil se enfoca en cómo los diferentes componentes del sistema pueden intercambiar y ejecutar código de manera dinámica. Este estilo se basa en la idea de que el código del programa puede ser transferido a través de una red desde un servidor centralizado a dispositivos clientes, donde se ejecutará localmente para realizar tareas específicas.

En este los componentes se organizan en dos categorías: aquellos que proporcionan el código y aquellos que lo reciben y lo ejecutan. Los conectores utilizados en este estilo arquitectónico se centran en la transferencia de código entre los componentes, y pueden ser de diferentes tipos, como descarga de código a

través de internet, sistemas de distribución de software, o aplicaciones con módulos adicionales que se pueden descargar.

Este estilo es muy útil para sistemas que necesitan actualizar su funcionalidad con regularidad, ya que permite la actualización en tiempo real y sin requerir la intervención manual de usuarios. Además, este estilo promueve la flexibilidad y la escalabilidad del sistema, ya que el código puede ser compartido y utilizado por múltiples dispositivos.

Estilos Peer-to-Peer

El estilo arquitectónico peer-to-peer (P2P) se enfoca en cómo los diferentes componentes de un sistema colaboran entre sí de manera igualitaria y dinámica, sin depender de un servidor centralizado. Este estilo se basa en la idea de que todos los nodos del sistema son iguales y pueden actuar como clientes y/o servidores.

En este estilo arquitectónico, los nodos del sistema se organizan en una red distribuida, donde cada nodo puede conectarse con cualquier otro nodo y compartir recursos como datos, procesamiento y almacenamiento. Los conectores utilizados en este estilo arquitectónicos se centran en la transferencia de recursos entre los nodos, y pueden ser de diferentes tipos, como protocolos de comunicación específicos, sockets o canales directos.

Es muy útil para sistemas que requieren alta escalabilidad y disponibilidad, ya que no dependen de un servidor centralizado y los nodos pueden unirse o salir de la red sin afectar el funcionamiento general. Además, este estilo promueve la robustez y la resistencia a fallos, ya que los nodos pueden trabajar juntos para compensar cualquier falla individual.

1.4.2 Framework de desarrollo

Un framework de desarrollo es una herramienta que proporciona una estructura para desarrollar software, aplicaciones o sistemas. Básicamente, se trata de un conjunto de bibliotecas, componentes y herramientas que simplifican el proceso de desarrollo al proporcionar una estructura predefinida.

Los frameworks son un conjunto de reglas, convenciones y mejores prácticas que permiten a los desarrolladores construir soluciones de software más rápido y eficientemente. Esto se logra al ofrecer una base de código común o una

arquitectura de aplicación común en la cual se pueden construir aplicaciones específicas ⁽²⁵⁾.

Los frameworks de desarrollo suelen incluir múltiples funcionalidades, como componentes de interfaz de usuario, controladores, módulos de seguridad, acceso a bases de datos, entre otros. Al utilizar un framework, los desarrolladores pueden acelerar el proceso de desarrollo ya que no tienen que empezar desde cero, sino que pueden aprovechar la estructura y las funcionalidades disponibles. Existen frameworks para diferentes lenguajes de programación y para diferentes tipos de aplicaciones, desde aplicaciones web hasta aplicaciones móviles y escritorio.

Algunos ejemplos de frameworks de desarrollo son ⁽²⁴⁾:

1. Angular: es un framework de código abierto desarrollado por Google para construir aplicaciones web de una sola página (SPA). Utiliza HTML, CSS y TypeScript para crear aplicaciones altamente interactivas.
2. React: es otro framework de código abierto utilizado para crear interfaces de usuario en aplicaciones web. Fue desarrollado por Facebook y se basa en la idea de crear pequeños componentes reutilizables que se combinan para formar una aplicación completa.
3. Django: es un framework de Python pensado para el desarrollo rápido de aplicaciones web. Proporciona herramientas y módulos para manejar la autenticación, los formularios, el acceso a bases de datos y otras funcionalidades comunes.
4. Laravel: es un framework de PHP utilizado para crear aplicaciones web robustas y escalables. Proporciona una sintaxis clara y concisa para trabajar con bases de datos, rutas, controladores y vistas.
5. Ruby on Rails: es un framework de Ruby para el desarrollo rápido de aplicaciones web. Se centra en la simplicidad, la convención sobre la configuración y el principio DRY.

1.4.2.1 Django

Django es un framework de código abierto para el desarrollo de aplicaciones web en Python. Fue creado en 2003 y se ha convertido en uno de los frameworks más

populares para el desarrollo web debido a su simplicidad, eficiencia y escalabilidad. Django es conocido por su capacidad para desarrollar aplicaciones web complejas de forma rápida y fácil, gracias a sus características preconstruidas y su arquitectura MVC (Modelo-Vista-Controlador) ⁽²⁶⁾.

Django ofrece una gran cantidad de herramientas y características preconstruidas que facilitan el desarrollo de aplicaciones web, como la creación automática de formularios y administradores de bases de datos, autenticación de usuarios y control de acceso, y soporte para múltiples idiomas y zonas horarias.

Además, Django se integra bien con otras tecnologías web, como HTML, CSS, JavaScript y RESTful APIs, lo que lo convierte en una opción popular para proyectos modernos de desarrollo web.

Django es un framework potente y flexible para el desarrollo de aplicaciones web en Python, que ofrece características preconstruidas y una arquitectura MVC que permite un desarrollo rápido y eficiente.

1.4.3 Bootstrap

Bootstrap es un framework de diseño web front-end de código abierto que se utiliza para desarrollar sitios web y aplicaciones móviles. Fue creado por Twitter y lanzado por primera vez en 2011. Bootstrap está basado en HTML, CSS y JavaScript y tiene como objetivo simplificar el proceso de desarrollo web al proporcionar una serie de herramientas y componentes preconstruidos ⁽²⁷⁾.

Bootstrap ofrece una amplia gama de características y elementos de diseño, como botones, formularios, tablas, alertas, carruseles, iconos y mucho más. También incluye soporte para diseño responsivo, lo que significa que los sitios web y aplicaciones creados con Bootstrap se ajustan automáticamente a diferentes tamaños de pantalla, desde dispositivos móviles hasta pantallas de escritorio.

Además, Bootstrap cuenta con una gran comunidad de desarrolladores que contribuyen continuamente al proyecto, lo que significa que siempre hay nuevas características y mejoras disponibles.

Bootstrap es un framework front-end muy popular para el diseño web, que ofrece una amplia variedad de herramientas preconstruidas y componentes de diseño, así

como soporte para diseño responsivo. Es fácil de usar y se actualiza regularmente gracias a su gran comunidad de desarrolladores.

1.5 Paradigmas de programación

Los paradigmas de programación se refieren a los diferentes enfoques o estilos que existen para programar un software. Cada paradigma tiene sus propias reglas y convenciones que definen cómo se estructura el código, cómo se manipulan los datos y cómo se resuelven los problemas ⁽²⁸⁾.

Los paradigmas de programación pueden influir en la forma en que se aborda un problema y en la facilidad con que se puede resolver. Algunos de los paradigmas más comunes son la programación imperativa, la programación orientada a objetos, la programación funcional, la programación declarativa y la programación basada en eventos.

Es importante tener en cuenta que muchos lenguajes de programación admiten varios paradigmas, lo que significa que un programador puede utilizar diferentes enfoques para resolver un mismo problema. La elección del paradigma adecuado dependerá de varios factores, como el tipo de proyecto, el tamaño del equipo y las preferencias personales del programador.

Existen varios tipos de paradigmas de programación, entre los cuales se destacan ⁽²⁸⁾:

1. Programación imperativa: este paradigma se centra en la definición de una serie de instrucciones que le indican al ordenador qué hacer paso a paso. La programación imperativa utiliza variables para almacenar datos y utiliza estructuras de control de flujo como bucles y condicionales.
2. Programación orientada a objetos: este paradigma se basa en la creación de objetos que contienen tanto datos como funciones. Los objetos interactúan entre sí mediante la llamada a métodos y el intercambio de mensajes. El encapsulamiento, la herencia y el polimorfismo son conceptos centrales en la programación orientada a objetos.
3. Programación funcional: en este paradigma, el énfasis está en las funciones y en cómo se utilizan para transformar datos de entrada en salida. Las funciones

son tratadas como ciudadanos de primera clase, lo que significa que pueden ser pasadas como argumentos a otras funciones y devueltas como resultado de una función. La programación funcional se basa en conceptos matemáticos como la lambda-cálculo.

4. Programación declarativa: en este paradigma, el programador se centra en describir el resultado deseado en lugar de la lógica subyacente necesaria para obtener ese resultado. La programación declarativa se divide en dos sub-paradigmas: la programación lógica y la programación funcional declarativa.
5. Programación basada en eventos: este paradigma se centra en la gestión de eventos y su respuesta asociada. Los eventos son interacciones que ocurren en el sistema, como una acción del usuario o una notificación de sistema. Las aplicaciones basadas en eventos se componen de piezas independientes que interactúan entre sí mediante la emisión y recepción de eventos.

1.5.1 Programación orientada a Objetos (POO)

La programación orientada a objetos (POO) es un paradigma de programación que se basa en la definición y manipulación de objetos. Un objeto es una instancia de una clase, que define su estructura y comportamiento ⁽²⁹⁾.

En POO, los programas se organizan en conjuntos de objetos que interactúan entre sí para realizar tareas específicas. Cada objeto tiene sus propias propiedades, como variables y métodos, que definen su estado y comportamiento respectivamente.

Las clases son la base de la POO y permiten definir nuevas estructuras de datos con propiedades y métodos específicos. Una vez definida una clase, se pueden crear múltiples instancias de esa clase, o sea, objetos, cada uno con sus propias propiedades y métodos ⁽²⁹⁾.

La POO ofrece una serie de ventajas, incluyendo el modularidad, la reutilización de código, la encapsulación de datos y la abstracción de conceptos complejos. Por estas razones, se utiliza ampliamente en el desarrollo de software moderno y en muchos lenguajes de programación populares, como Java, Python y C++.

1.6 Herramientas CASE

Las herramientas CASE (Computer-Aided Software Engineering) son programas de software que ayudan en el proceso de desarrollo de software, mediante la automatización de diversas tareas relacionadas con el ciclo de vida del software.

Algunas de las tareas que las herramientas CASE pueden automatizar incluyen (30)(31):

- **Análisis y diseño de sistemas:** Estas herramientas permiten a los desarrolladores crear diagramas de flujo, diagramas de clases y otros modelos visuales para representar la estructura y el comportamiento del sistema.
- **Generación automática de código:** Algunas herramientas CASE permiten generar automáticamente código fuente a partir de los modelos creados durante el análisis y diseño del sistema.
- **Pruebas de software:** Las herramientas CASE también pueden incluir funcionalidades para la generación de casos de prueba y la automatización de pruebas de software.
- **Gestión de proyectos:** Muchas herramientas CASE ofrecen funcionalidades para la planificación y seguimiento de proyectos de software, así como para la gestión de versiones y control de cambios.

Algunas de las herramientas CASE más populares incluyen Rational Rose, Enterprise Architect, Visual Paradigm, entre otras.

1.6.1 Visual Paradigm for UML

Visual Paradigm for UML es una herramienta de modelado de software que permite a los desarrolladores crear y mantener modelos UML (Lenguaje Unificado de Modelado) para el diseño de sistemas de software. La herramienta se utiliza para diagramar, diseñar y documentar sistemas de software complejos desde la fase de análisis hasta la implementación ⁽³²⁾.

Visual Paradigm for UML es compatible con UML 2.x y ofrece una amplia gama de características para el modelado de software. Las principales características

incluyen la creación de diagramas UML, la generación de código, la ingeniería inversa, la documentación y la gestión de requisitos.

La herramienta permite crear varios tipos de diagramas UML, como diagramas de casos de uso, diagramas de clases, diagramas de secuencia, diagramas de estado, diagramas de actividad y diagramas de componentes. También ofrece características avanzadas para la creación de modelos de datos y la integración con bases de datos ⁽³²⁾.

Además, Visual Paradigm for UML integra herramientas de colaboración en equipo, lo que permite a los desarrolladores trabajar juntos en proyectos de software de manera eficiente. Ofrece soporte para control de versiones y herramientas de seguimiento de problemas, así como características de integración con otras herramientas de desarrollo de software.

Visual Paradigm for UML es una herramienta de modelado de software completa y poderosa para el diseño y documentación de sistemas de software. Proporciona una amplia gama de características y herramientas avanzadas para el modelado UML y la colaboración en equipo.

1.7 Lenguajes de Programación

Un lenguaje de programación es un conjunto de reglas, símbolos y convenciones que se utilizan para escribir programas de ordenador. Hay muchos lenguajes de programación diferentes que se utilizan para una variedad de propósitos, desde el desarrollo de software hasta la creación de sitios web e incluso aplicaciones móviles ⁽³³⁾.

Algunos ejemplos de lenguajes de programación incluyen ⁽³⁴⁾:

- Java: Java es un lenguaje de programación orientado a objetos que se utiliza en una variedad de aplicaciones, desde desarrollo de software hasta aplicaciones móviles y juegos.
- Python: Python es un lenguaje de programación interpretado y de alto nivel que se utiliza comúnmente en ciencia de datos, inteligencia artificial, desarrollo web y otras áreas.

- JavaScript: JavaScript es un lenguaje de programación de scripting que se utiliza principalmente en desarrollo web, para crear efectos interactivos y dinámicos en los sitios web.
- C#: C# es un lenguaje de programación orientado a objetos desarrollado por Microsoft que se utiliza para crear aplicaciones de escritorio, juegos y aplicaciones web.
- PHP: PHP es un lenguaje de programación utilizado para el desarrollo web del lado del servidor. Se utiliza comúnmente para crear aplicaciones web dinámicas y sitios web interactivos.

Además de estos, hay muchos otros lenguajes de programación disponibles, cada uno con sus propias características, ventajas y desventajas. La elección del lenguaje de programación adecuado dependerá de la tarea específica que se esté realizando y de las preferencias personales del programador.

1.7.1 Python

Python es un lenguaje de programación de alto nivel, interpretado y orientado a objetos, diseñado para ser fácil de usar y leer. Es utilizado principalmente en aplicaciones de ciencia de datos, inteligencia artificial, desarrollo web, scripting y automatización.

Entre las principales características de Python se encuentran ⁽³⁵⁾:

- Sintaxis legible: Python utiliza una sintaxis clara y concisa, lo que hace que sea fácil de leer y entender.
- Amplia gama de bibliotecas y módulos: Python cuenta con una gran comunidad de desarrolladores que han creado una gran cantidad de bibliotecas y módulos para realizar diferentes tareas, lo que facilita la creación de aplicaciones complejas.
- Multiplataforma: Python se puede ejecutar en una amplia variedad de sistemas operativos, incluyendo Windows, Linux y MacOS.
- Fácil de aprender: Python es uno de los lenguajes de programación más fáciles de aprender, lo que lo hace ideal para principiantes.

- Interactivo: Python tiene un intérprete interactivo que permite a los desarrolladores experimentar con el lenguaje y probar su código rápidamente.

Además, Python es conocido por su eficiencia en el procesamiento de grandes volúmenes de datos y su capacidad para trabajar bien con otros lenguajes de programación.

Python es uno de los lenguajes de programación más populares y versátiles disponibles actualmente, gracias a su facilidad de uso, su amplia gama de bibliotecas y módulos, y su capacidad para manejar grandes volúmenes de datos y automatizar tareas.

1.7.2 JavaScript

JavaScript es un lenguaje de programación utilizado principalmente para el desarrollo web del lado del cliente. Permite a los desarrolladores crear interfaces de usuario interactivas, validar la entrada del usuario, manipular el DOM (Modelo de Objeto de Documento) y comunicarse con servidores para actualizar dinámicamente el contenido en las páginas web sin necesidad de actualizar la página ⁽³⁶⁾.

JavaScript también se puede utilizar en el lado del servidor a través de plataformas como Node.js, lo que permite la creación de aplicaciones web escalables y scripting del lado del servidor.

Algunas características clave de JavaScript incluyen su naturaleza dinámica, tipado débil, soporte para closures y funciones de orden superior, y su capacidad para manejar la programación asincrónica a través de devoluciones de llamada, promesas y async/await.

En general, JavaScript se ha convertido en una herramienta esencial para los desarrolladores web, impulsando muchos sitios web y aplicaciones web populares.

1.8 Entorno de desarrollo integrado

Un entorno de desarrollo integrado (IDE, por sus siglas en inglés) es una herramienta que facilita el proceso de desarrollar software. Un IDE típicamente

incluye un editor de código, un depurador, un compilador/interpretador, y otros elementos para ayudar a los programadores a escribir, probar, y depurar código.

Los IDEs también suelen incluir características como resaltado de sintaxis, autocompletado, refactorización de código, integración con herramientas de control de versiones, y la posibilidad de ejecutar y probar el código dentro del mismo entorno ⁽³⁷⁾.

Hay muchas opciones de IDE disponibles, algunos ejemplos populares son Visual Studio Code, IntelliJ IDEA, Eclipse, NetBeans, y PyCharm. Cada uno tiene sus propias fortalezas y debilidades, y la elección de IDE dependerá del lenguaje de programación utilizado y las necesidades específicas del desarrollador.

Algunos entornos de desarrollo integrado (IDE) populares son ⁽³⁷⁾:

1. Visual Studio Code: es un IDE ligero y gratuito desarrollado por Microsoft que admite una amplia variedad de lenguajes de programación y multiplataforma, incluidos JavaScript, TypeScript, Python y C #.
2. IntelliJ IDEA: es un IDE Java de alta calidad desarrollado por JetBrains. Ofrece una gran cantidad de funciones avanzadas, desde refactorización de código hasta pruebas unitarias integradas y depuración remota.
3. Eclipse: es un IDE gratuito y de código abierto para Java que también admite otros lenguajes de programación a través de complementos. Se utiliza comúnmente en la industria para el desarrollo de aplicaciones empresariales.
4. Xcode: es el IDE oficial de Apple para el desarrollo de aplicaciones iOS y MacOS. Incluye herramientas para diseño de interfaz de usuario, edición de código y depuración, y admite varios lenguajes de programación, como Swift y Objective-C.
5. PyCharm: es un IDE Python desarrollado por JetBrains. Ofrece características avanzadas para el desarrollo web con Django y Flask, así como análisis de código, refactorización, seguimiento de pruebas y depuración.
6. Android Studio: es el IDE oficial de Google para el desarrollo de aplicaciones Android. Está basado en IntelliJ IDEA y viene con herramientas para diseño de interfaz, edición de código y depuración.

Estos son solo algunos ejemplos de IDE populares, hay muchos más disponibles según las necesidades específicas del desarrollador y el lenguaje de programación utilizado.

1.8.1 Visual Studio Code IDE

Visual Studio Code (VSC) es un IDE. VSC es un editor de código fuente gratuito y de código abierto desarrollado por Microsoft con el marco de trabajo Electron, para Windows, Linux y macOS. VSC incluye soporte para depuración, resaltado de sintaxis, finalización de código inteligente, fragmentos de código, refactorización de código y Git integrado. Los usuarios pueden cambiar el tema, los atajos de teclado, las preferencias e instalar extensiones que agregan funcionalidad. VSC es un editor de código fuente que se puede utilizar con una variedad de lenguajes de programación, incluyendo C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python, Rust y Julia. VSC se basa en el marco de trabajo Electron, que se utiliza para desarrollar aplicaciones web Node.js que se ejecutan en el motor de diseño Blink. VSC emplea el mismo componente de editor (codenamed “Monaco”) utilizado en Azure DevOps.⁽³⁸⁾

Algunas características clave de Visual Studio Code incluyen:

- Finalización inteligente de código y sugerencias
- Herramientas de depuración para el desarrollo local y remoto
- Compatibilidad integrada con el control de versiones de Git
- Mercado de extensiones para agregar funcionalidad y personalizar el editor
- Terminal integrado para ejecutar comandos y scripts dentro del editor
- Edición de varios cursores para realizar cambios en varias líneas simultáneamente

En general, Visual Studio Code es una herramienta potente y versátil para el desarrollo de software que se ha convertido rápidamente en uno de los editores de código más populares disponibles en la actualidad.

1.9 Sistema Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD) es una aplicación que permite crear, modificar y gestionar bases de datos. Los SGBD se utilizan para almacenar grandes cantidades de datos y permiten a los usuarios acceder a ellos de manera eficiente y segura.

Algunos ejemplos de SGBD son MySQL, Oracle, Microsoft SQL Server, PostgreSQL y MongoDB ⁽³⁹⁾.

- MySQL es un SGBD muy popular y ampliamente utilizado en la web. Es un sistema de base de datos relacional de código abierto que se utiliza principalmente en aplicaciones web y es compatible con muchas plataformas diferentes.
- Oracle es otro SGBD popular que se utiliza en empresas y organizaciones de gran tamaño. Este sistema de base de datos relacional ofrece capacidades de alta disponibilidad y escalabilidad, lo que lo hace ideal para aplicaciones empresariales críticas. Ofrece una variedad de opciones, incluyendo servicios gratuitos y de pago.
- Microsoft SQL Server es un SGBD que se ejecuta en el sistema operativo Windows y es utilizado por empresas y organizaciones de todos los tamaños. Es compatible con una variedad de lenguajes de programación y puede integrarse fácilmente con otras herramientas y tecnologías de Microsoft.
- PostgreSQL es un SGBD de código abierto que se utiliza para aplicaciones web y empresariales. Ofrece características avanzadas como transacciones ACID y soporte para objetos JSON y XML.
- MongoDB es un SGBD NoSQL que se utiliza para aplicaciones modernas de gran escala. Utiliza un modelo de datos basado en documentos y se utiliza en la nube y en entornos de contenedores.

Un SGBD es una herramienta esencial para la gestión eficiente de grandes cantidades de datos, y hay una variedad de opciones disponibles para adaptarse a las necesidades de diferentes aplicaciones.

1.9.1 MySQL

MySQL es un sistema de gestión de bases de datos relacionales (RDBMS) de código abierto. Fue desarrollado originalmente por MySQL AB, que fue adquirido posteriormente por Oracle Corporation. MySQL utiliza el lenguaje SQL para la manipulación y recuperación de datos en una base de datos.

Entre las características principales de MySQL se encuentran ⁽⁴⁰⁾:

- Soporte para diferentes sistemas operativos como Windows, Linux, macOS, etc.
- Soporte para múltiples lenguajes de programación, incluyendo PHP, Java, Python, entre otros.
- Alta escalabilidad, lo que permite manejar grandes volúmenes de datos.
- Seguridad mejorada con la posibilidad de asignar permisos a usuarios específicos y cifrado de datos.
- Facilidad de uso y administración mediante herramientas como MySQL Workbench.

MySQL se ha convertido en una opción popular para aplicaciones web, especialmente aquellas que requieren una gran cantidad de datos, debido a su rendimiento, velocidad y confiabilidad. Además, al ser de código abierto, los desarrolladores pueden personalizarlo y adaptarlo a sus necesidades específicas.

1.10 Servidor web

Un servidor web es un programa que se ejecuta en un ordenador y permite a los usuarios acceder a páginas web y otros contenidos a través de Internet. Cuando un usuario solicita una página web, el servidor web procesa la solicitud y envía la información solicitada al navegador del usuario.

Entre los servidores web se incluyen ⁽⁴¹⁾:

- Apache HTTP Server: Apache es uno de los servidores web más populares y ampliamente utilizados en la web. Es un software libre y de código abierto que se ejecuta en una variedad de sistemas operativos.

- Nginx: es un servidor web de alta velocidad que también se utiliza como proxy inverso y balanceador de carga. Es conocido por su capacidad para manejar grandes volúmenes de tráfico y ser eficiente en el uso de recursos.
- Microsoft IIS: Internet Information Services (IIS) es el servidor web de Microsoft que se ejecuta en el sistema operativo Windows. Se utiliza principalmente para alojar sitios web y aplicaciones .NET.
- Lighttpd: es un servidor web ligero y rápido que se utiliza comúnmente para alojar sitios web y aplicaciones web de alto rendimiento.
- Caddy: Caddy es un servidor web moderno que se ejecuta en múltiples plataformas y está diseñado para ser fácil de configurar y utilizar.

Un servidor web es una parte esencial de la infraestructura de la web, ya que permite a los usuarios acceder a contenido en línea. Hay una variedad de opciones disponibles para adaptarse a las necesidades de diferentes aplicaciones, desde servidores web de propósito general hasta opciones de alto rendimiento y baja latencia.

1.10.1 Apache 2.4.48

Es un servidor web de código abierto multiplataforma que implementa el protocolo HTTP/1.1 muy popular por su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza. Además, permite la personalización de respuestas ante errores que puedan ocurrir en el servidor, inclusive programarlo para que ejecute scripts cuando esto suceda ⁽⁴¹⁾⁽⁴²⁾.

Apache es utilizado principalmente para enviar páginas web estáticas y dinámicas en la World Wide Web. Muchas aplicaciones web son diseñadas asumiendo como ambiente de implantación a Apache, o que utilizarán características propias de este servidor web.

Apache es altamente configurable, permite bases de datos de autenticación y negociado de contenido.

Apache es usado cuando el contenido necesita ser puesto a disposición en una forma segura y confiable. Ejemplos: Cuando se comparten archivos desde una

computadora personal hacia Internet, cuando se comparten documentos importantes dentro de determinadas las áreas de una empresa.

Los programadores de aplicaciones web a veces utilizan una versión local de Apache con el fin de previsualizar y probar código mientras éste es desarrollado.

Sus principales ventajas son:

- Modular
- Código Abierto
- Multiplataforma
- Popular (fácil de conseguir ayuda/soporte)

La última versión estable de Apache HTTP Server es la 2.4.48. (APACHE, 2021)

1.11 Conclusiones parciales

A partir de la revisión teórica realizada, se puede afirmar que existen diversas estrategias y técnicas para lograr un posicionamiento adecuado en Internet. Algunas de las principales consideraciones para tener en cuenta son la optimización del contenido y la estructura de la aplicación web, la creación de enlaces de calidad, la presencia en redes sociales y la generación de contenido fresco y original. Además, es necesario combinar estrategias y técnicas efectivas con el uso de tecnologías y herramientas apropiadas, como lo son: el VSC, Django, MySQL y Apache. Es importante mantenerse actualizado en este ámbito en constante evolución para poder adaptarse a las demandas del mercado y maximizar la visibilidad y el impacto de Cementos Cienfuegos S.A. en línea.

CAPÍTULO 2: DESCRIPCIÓN DE LA PROPUESTA DE SOLUCIÓN:

Introducción:

En el presente capítulo se plantea y describe la propuesta de solución y los mecanismos generados por el escenario número 4 en la metodología Agile Iniciad Procesos (AUP) en las etapas de análisis, diseño e implementación.

2.1 Análisis del proceso de la tienda online

La propuesta de solución consiste en crear una aplicación web para gestionar las ventas de productos a través de Internet. Una tienda online es una aplicación web que cuenta con un catálogo virtual de productos e integraciones para recibir pagos, lo que permite al vendedor tener contacto directo con el comprador y control sobre la gestión de sus ventas. Tiene una serie de actividades que son una larga jornada de trabajo consultando documentación. Estos son procesos que se derivan de los requisitos que se definan.

Dicho proceso tiene varios subprocesos que se describen a continuación:

Autenticarse:

Crear Nuevo Usuario: Crear un usuario en un sitio web implica introducir credenciales de usuario, como un nombre de usuario, una contraseña, y una dirección de correo Gmail para llevar a cabo un proceso de verificación de identidad que permita al usuario acceder a los recursos protegidos del sitio.

Iniciar Sesión: Implica la introducción de las credenciales de usuario, como un nombre de usuario y una contraseña ya creadas anteriormente en dicho sitio web.

Olvido su contraseña: Mediante su correo Gmail puede recuperar su contraseña olvidada.

Productos:

Ventas en Cuba: Se muestran los productos que oferta la entidad para la venta mayorista y minorista en el país.

Ventas en el Extranjero: Muestra un catálogo de los diversos productos que se ofertan en la entidad para su venta en el extranjero.

Acerca de: Muestra una breve descripción de la entidad y de sus distintos organismos para que el público ajeno a esta se informe y conozca dicha entidad.

Contáctenos: Brinda las distintas formas de comunicarse con la entidad por si presenta algunas dificultades o le interesa ponerse en contacto con los administrativos de la organización.

Tienda:

Esta se muestra una vez se ingrese al sitio web con un nombre de usuario y una contraseña ya creados anteriormente. En este menú se muestra el submenú siguiente:

Seleccionar productos: Selecciona el producto a comprar y lo agrega al carro de compras.

Carro: Muestra todas las compras que ha agregado al carrito de compras.

Nombre del usuario autenticado:

Inbox: Muestra los pedidos pendientes a pagar.

Perfil: Muestra los datos personales del usuario que se registran en el sitio web.

Salir: Cierra sesión.

2.2 Requisitos de software

2.2.1 Requisitos Funcionales (RF)

A partir de entrevistas y reuniones con el cliente, se obtuvieron 14 requisitos funcionales que permiten determinar de manera clara lo que debe hacer el sistema propuesto, lo que se logra mediante una especificación detallada de las responsabilidades del sistema, basada en modelos, patrones y abstracciones teóricas.

No	Nombre	Descripción	Prioridad	Complejidad
RF1	Autenticarse	El usuario introduce sus datos, el sistema lo comprueba y si son válidos, el usuario queda autenticado con	Alta	Baja

		el nivel de privilegios asignados; si los datos no son válidos el sistema muestra un mensaje de error		
RF2	Registrarse (Crear Nuevo Usuario)	Pide los datos necesarios para crear una cuenta de usuario con la cual registrarse en el sitio web	Alta	Alta
RF3	Reiniciar contraseña	Ingresa el correo Gmail con el que se creó su cuenta para enviar el código de la nueva contraseña.	Alta	Media
RF4	Listar todos los Productos	Muestra todos los productos ofertados por la entidad.	Media	Baja
RF5	Listado de productos ofertados en Cuba	Muestra un catálogo con los productos que se ofertan en Cuba	Alta	Baja
RF6	Listado de productos para el Exterior	Muestra un catálogo con los productos que se ofertan en el exterior	Alta	Baja
RF7	Información General de la Empresa	Brinda una breve reseña de la entidad.	Media	Baja

RF8	Contáctenos	Muestra los distintos métodos para ponerse en contacto con los administrativos de la entidad y de atención al cliente.	Media	Media
RF9	Listado de pedidos pendientes	Muestra los pedidos pendientes de pago	Alta	Media
RF10	Perfil	Muestra los datos personales del Usuario	Alta	Baja
RF11	Salir	Cierra sesión	Alta	Baja
RF12	Listado de Compras reservadas	Muestra los pedidos reservados del usuario	Media	Baja
RF13	Seleccionar productos	Selecciona el producto a comprar y lo agrega al carro de compras.	Alta	Media
RF14	Eliminar producto del carro de compras	Elimina el producto del carro de compras.	Media	Media

Tabla 3. Descripción de los requisitos funcionales del sistema.

2.2.2 Requisitos no funcionales (RNF)

Los requerimientos no funcionales son aquellos que especifican las propiedades o cualidades que el producto debe tener, como restricciones del entorno o de implementación, rendimiento, dependencias de la plataforma, facilidad de mantenimiento, extensibilidad, entre otros. En el desarrollo de la aplicación web, se

definieron 7 RNF, los cuales se obtuvieron a partir de entrevistas y reuniones con el cliente.

No.	Descripción	Tipo de requisito
RNF1	La herramienta propuesta debe presentar una interfaz legible, fácil de usar, con colores armónicos, discreta, lo que facilita la operabilidad del usuario.	Requerimientos de apariencia o interfaz externa y de Usabilidad
RNF2	Diseño atractivo, intuitivo y responsive. Carga y búsquedas rápidas.	Requerimientos de Rendimiento
RNF3	Una formación inicial y soporte continuo de entrenamiento inicial en áreas como alojamiento web, una plataforma adecuada, y un equipo de soporte técnico y mantenimiento para garantizar el correcto funcionamiento de la aplicación web.	Requerimientos de Soporte
RNF4	Se deben cumplir con los requisitos legales, fiscales y técnicos necesarios. Los requisitos legales incluyen la forma jurídica, el alta en Hacienda y Seguridad Social, y la Ley General para la Defensa	Requerimientos Legales

	de Consumidores.	
RNF5	- Utilizar un certificado SSL para encriptar la información. - Habilitar la autenticación de dos pasos para el inicio de sesión. - Utilizar una plataforma de procesamiento de pagos seguros. - Realizar copias de seguridad de los datos. - Cumplir con la norma PCI. - Elegir una plataforma de eCommerce segura. - Realizar escaneos de seguridad periódicos. - Contar con reconocidas opciones para realizar pagos.	Requerimientos de Seguridad
RNF6	- Integración con herramientas de gestión de información de productos. - Soporte técnico y mantenimiento a largo plazo.	Requerimiento de Software
RNF7	La interfaz del sistema debe ser a través de una página Web.	Restricciones en el diseño y la implementación

Tabla 4. Descripción de los requisitos no funcionales del sistema.

2.3 Historias de usuarios

Una historia de usuario es una técnica utilizada para describir una funcionalidad o requisito del software desde la perspectiva del usuario final. Se trata de una breve

descripción escrita en lenguaje natural que captura los objetivos, necesidades y acciones que un usuario específico desea realizar dentro de la aplicación ⁽⁴³⁾. En total se definieron 14 historias de usuarios, a continuación, se muestra la historia de usuario de mayor criticidad, “Crear Nuevo Usuario”:

Historia de usuario	
Número: 2	Requisito: Crear Nuevo Usuario
Programador: Edgar Leonardo Llerena Martínez. Ing. Ángel Echevarría Zamora.	Iteración Asignada: 3
Prioridad: Alta	Tiempo Estimado: 5 minutos
Riesgo en Desarrollo: Alta	Tiempo Real:
Usuario: Nombre que desea ponerle a la cuenta a crear. (Obligatorio) Correo electrónico: Correo electrónico a vincular con la cuenta. (Obligatorio) Nombre: Nombre del Usuario. (Obligatorio) Apellidos: Apellidos de Usuario. (Obligatorio) Compañía: Se puede pedir el nombre de una compañía para los siguientes propósitos: -Para identificar a la empresa que realiza la compra. - Para facturación y cumplimiento de requisitos fiscales. - Para verificar la autenticidad de la empresa y evitar fraudes. - Para personalizar la experiencia del cliente y ofrecer promociones especiales a empresas. (Opcional si no es una compañía la que desea comprar) Número de identificación tributaria: código único utilizado para identificar a los contribuyentes en cuestiones tributarias, cambiarias y aduaneras. (Obligatorio) Contraseña: Contraseña de la cuanta. (Obligatorio)	

Repetir Contraseña: Ratifica la contraseña (Obligatorio)

Prototipo elemental de interfaz gráfica de usuario:

The screenshot shows a web interface for 'LAS PAILAS DE CEMENTO'. At the top, there is a navigation bar with links: 'Productos' (with a dropdown arrow), 'Acerca de', and 'Contáctenos'. The main content area is divided into two sections. On the left, there is a large logo consisting of two interlocking semi-circles, one grey and one teal, with the text 'LAS PAILAS DE CEMENTO' below them. On the right, there is a form titled 'Crear nuevo usuario'. The form contains the following fields: 'Usuario', 'Correo electrónico', 'Nombre', 'Apellidos', 'Compañía', 'Número de identificación tributaria', 'Contraseña', and 'Repetir contraseña'. Below the 'Repetir contraseña' field, there is a checkbox labeled 'Mostrar contraseña' which is currently unchecked. At the bottom of the form, there is a button labeled 'Registrar' with a small icon of a person.

2.4 Descripción de la arquitectura

La arquitectura de software es un proceso crítico en el desarrollo de cualquier pieza de software, ya que proporciona una guía teórica detallada que nos permite entender cómo van a encajar cada una de las piezas del producto o servicio. Es una fase de planificación crítica a la hora de comenzar cualquier proyecto para cualquier pieza de software, ya que proporciona una hoja de ruta clara sobre el camino a seguir en el desarrollo, sobre qué componentes formarán parte del producto ⁽²³⁾.

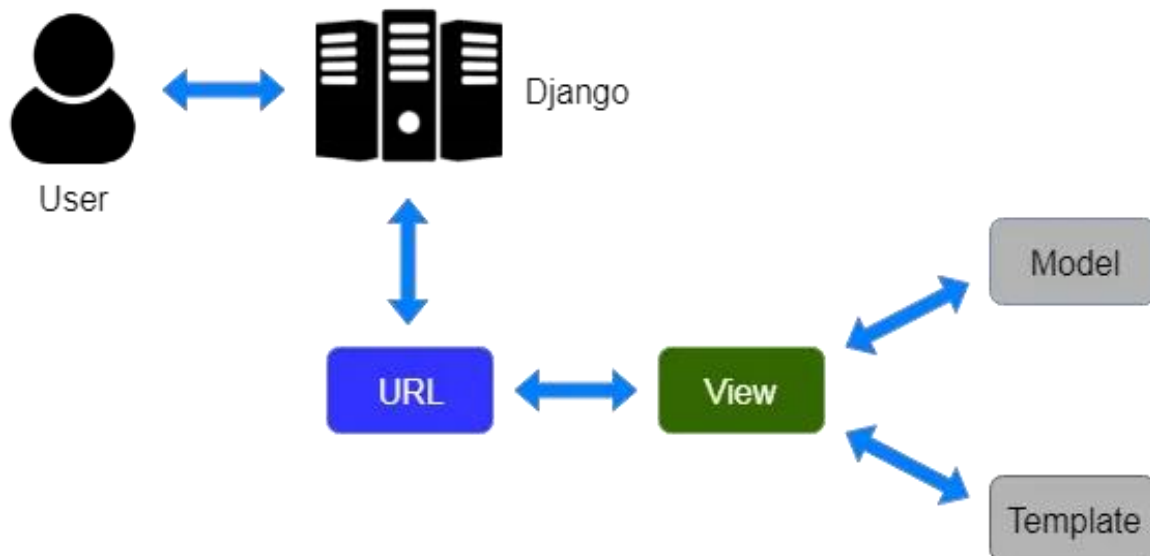
El desarrollo de software es un proceso complejo que requiere un enfoque estructurado y eficiente para lograr resultados óptimos. En este contexto, los patrones de diseño de software se han convertido en una herramienta invaluable para los desarrolladores, brindando soluciones probadas y comprobadas para problemas comunes de diseño. Los patrones de diseño son soluciones generales, reutilizables y aplicables a diferentes problemas de diseño de software. Al comprender y aplicar estos patrones, los profesionales del desarrollo pueden

mejorar significativamente la eficiencia y mantenibilidad del código, evitando errores costosos y optimizando el tiempo de desarrollo. Existen diferentes tipos de patrones de diseño, como los patrones creacionales, estructurales y de comportamiento, cada uno con su propia función y aplicación.

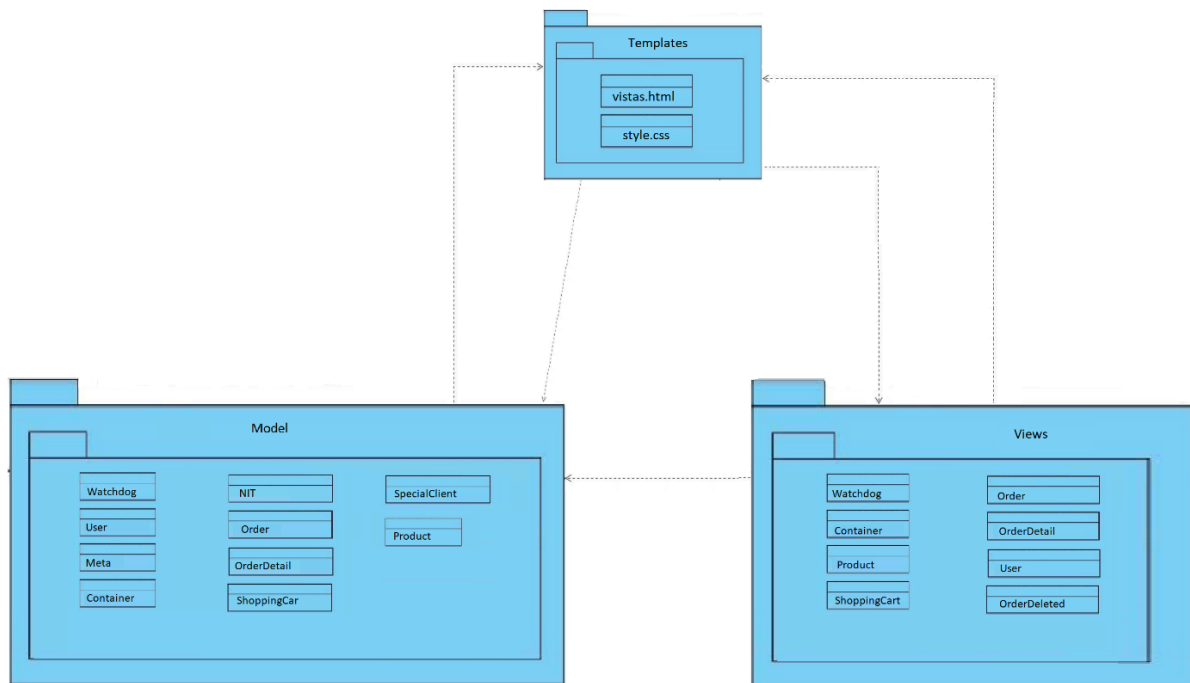
Patrón de arquitectura Model Views Templates:

El término “Model Views Template” (MVT) se refiere a un patrón de diseño de software utilizado en el desarrollo de aplicaciones web. El patrón MVT se utiliza en el framework de desarrollo web Django de Python y consta de tres componentes principales: Modelo (Model), Vista (View) y Plantilla (Template). El modelo se encarga de manejar los datos y se representa mediante una base de datos. La vista es una función de Python que se utiliza para manejar las solicitudes web. La plantilla contiene contenido estático como HTML, CSS y JavaScript. El patrón MVT se utiliza para separar la lógica de la aplicación, la presentación y el manejo de datos en componentes separados y bien definidos. Además, el término “Modelo Vista Templates” también se puede encontrar en el contexto de plantillas de currículum o CV, donde se refiere a plantillas de currículum personalizables y editables que se pueden descargar en línea. ⁽⁴⁴⁾

Se muestra a continuación en la siguiente figura como interactúa estos tres componentes en la arquitectura definida:



A continuación, se muestra la aplicación de la arquitectura a la propuesta de solución:

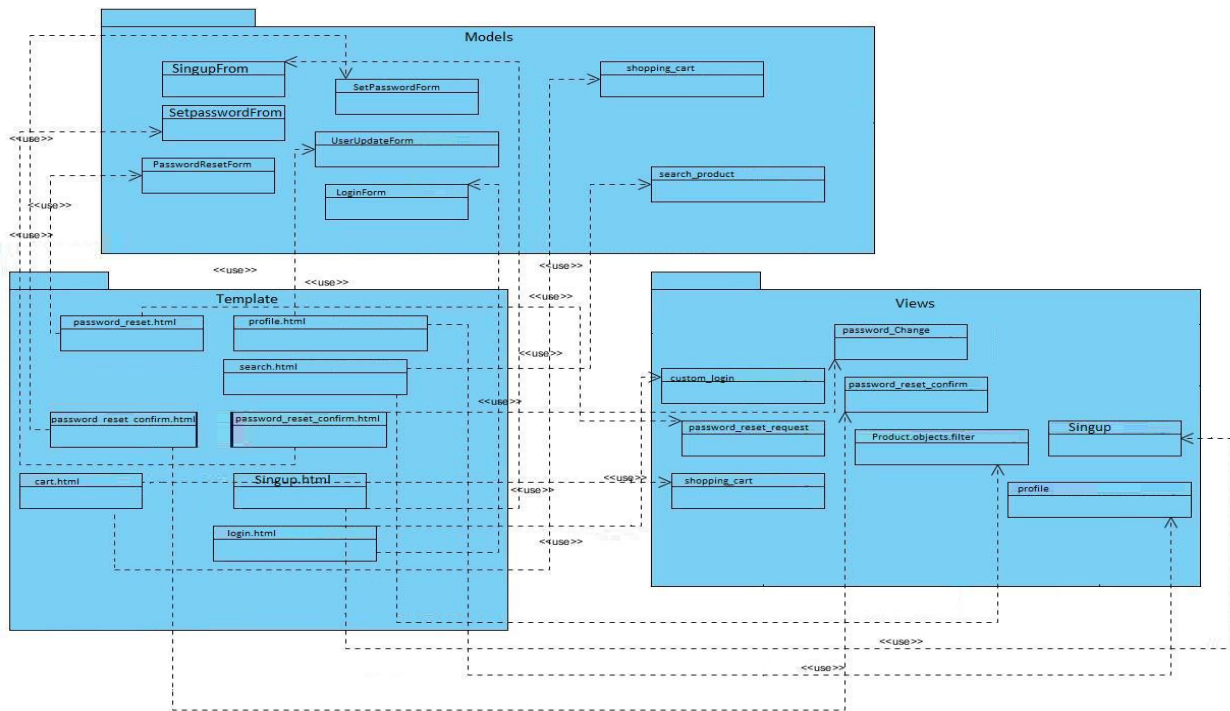


2.5 Diagrama de clases diseño

El propósito de un diagrama de clases es describir las clases que conforman el modelo de un determinado sistema. Dado el carácter de refinamiento iterativo que caracteriza un desarrollo orientado a objeto, el diagrama de clases va a ser creado y refinado durante las fases de análisis y diseño, estando presente como guía en la implementación del sistema.

Los diagramas de clases son uno de los tipos de diagramas más útiles en UML, ya que trazan claramente la estructura de un sistema concreto al modelar sus clases, atributos, operaciones y relaciones entre objetos ⁽⁴⁵⁾.

A continuación, se muestra el diagrama de clases de la propuesta de solución:



2.6 Patrones de diseño utilizados

El desarrollo de software es un proceso complejo que requiere un enfoque estructurado y eficiente para lograr resultados óptimos. En este contexto, los patrones de diseño de software se han convertido en una herramienta invaluable para los desarrolladores, brindando soluciones probadas y comprobadas para problemas comunes de diseño.

Patrón GRASP (General Responsibility Assignment Software Patterns)

GRASP es un conjunto de patrones o heurísticas para definir el reparto de responsabilidades de un sistema orientado a objetos. Los principios GRASP son el fundamento de otros muchos principios y nos ayudan a encontrar respuesta a preguntas bastante básicas. Algunos de los patrones GRASP que se pueden aplicar en una tienda en línea son Creator, Information Expert, Controller y Polymorphism. Estos patrones ayudan a definir quién debe crear un objeto, a qué clase pertenece una responsabilidad, quién recibe y coordina una operación del sistema, y cómo manejar distintas cosas en función del tipo de objeto o información que se recibe.

- **Creator:** Este patrón se enfoca en la creación de objetos y en determinar quién debe ser responsable de crearlos. Se busca asignar la responsabilidad de creación de objetos a la clase que tenga la información necesaria para crearlos. ⁽⁴⁷⁾
- **Information Expert:** Este patrón se enfoca en determinar quién debe ser responsable de una tarea específica. Se busca asignar la responsabilidad a la clase que tenga la información necesaria para cumplir con la tarea. ⁽⁴⁷⁾
- **Controller:** Este patrón se enfoca en la coordinación de las operaciones del sistema. Se busca asignar la responsabilidad de coordinación a una clase que tenga una visión general del sistema y pueda coordinar las operaciones de las demás clases. ⁽⁴⁷⁾
- **Polymorphism:** Este patrón se enfoca en la capacidad de los objetos de una clase para tomar diferentes formas. Se busca asignar la responsabilidad de la operación a la clase que tenga la capacidad de tomar diferentes formas y pueda realizar la operación de manera adecuada. ⁽⁴⁷⁾

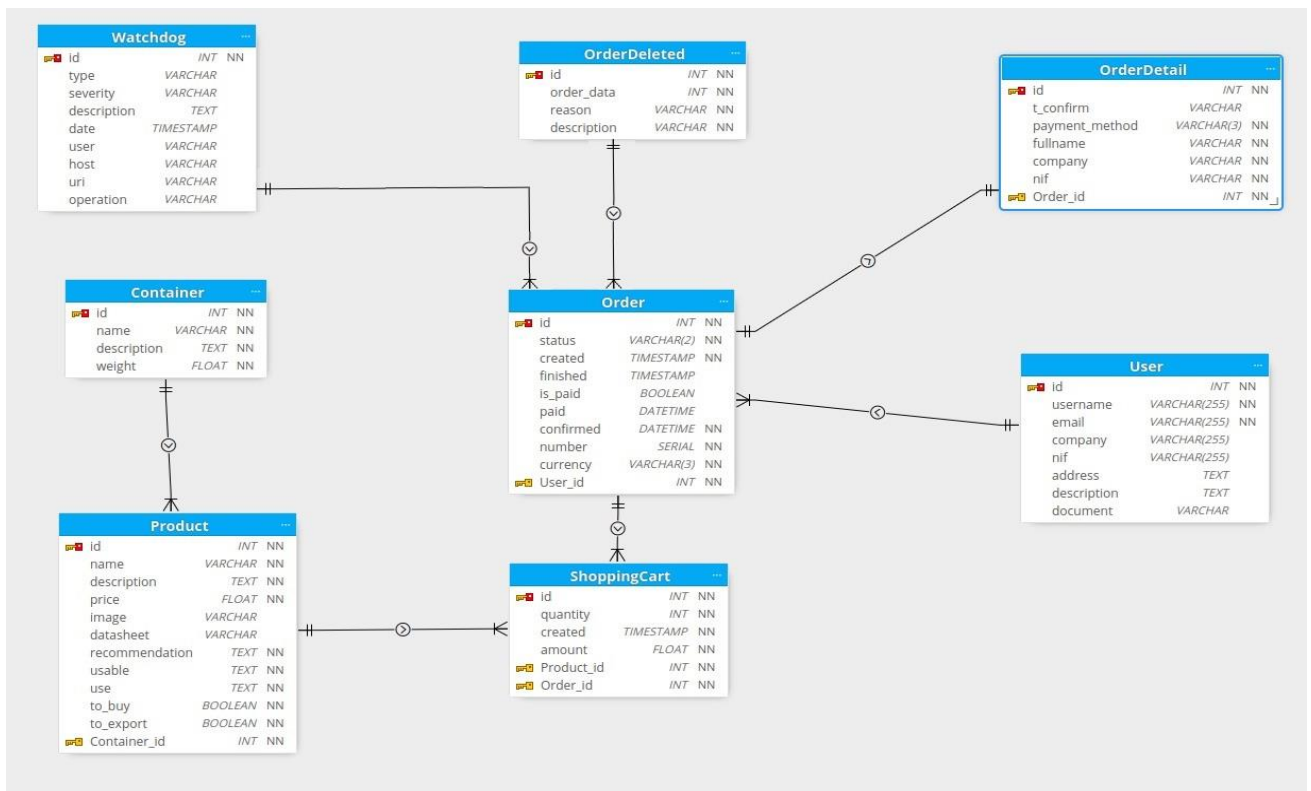
Patrones GoF (Gang of four): Los patrones GoF (Gang of Four) son un conjunto de patrones de diseño que buscan facilitar la reutilización de diseños y arquitecturas software exitosas. Los patrones GoF se dividen en tres tipos: patrones de creación, patrones estructurales y patrones de comportamiento. Algunos patrones GoF que se pueden aplicar en una tienda en línea son Abstract Factory, Builder, Facade y Singleton.

- **Abstract Factory:** Es un patrón de diseño creacional que permite producir familias de objetos relacionados sin especificar sus clases concretas. Este patrón ofrece una interfaz para crear objetos a partir de cada clase de la familia de productos, lo que permite crear la variante correcta de un producto que combine con los productos que ya ha creado la aplicación. ⁽⁴⁸⁾
- **Builder:** Se enfoca en construir objetos complejos, paso a paso. Este patrón permite ejecutar algunos pasos adicionales de construcción antes de extraer el producto. Las clases del Abstract Factory a menudo se basan en un grupo de métodos de fábrica, pero también puedes utilizar Prototype para escribir los métodos de estas clases. ⁽⁴⁹⁾

- **Facade:** Es un patrón estructural que proporciona una interfaz unificada para un conjunto de interfaces en un subsistema. Este patrón oculta la complejidad del subsistema y proporciona una interfaz simplificada para el cliente. El patrón Facade puede servir como alternativa a Abstract Factory cuando se desea esconder la forma en que se crean los objetos del subsistema a partir del código cliente. ⁽⁵⁰⁾
- **Singleton:** Es un patrón creacional que garantiza que una clase tenga una sola instancia y proporciona un punto de acceso global a ella. Este patrón se utiliza para garantizar que solo haya una instancia de una clase y para proporcionar un punto de acceso global a ella.

2.7 Modelo de datos

Un modelo de BD muestra la estructura lógica de la base, incluidas las relaciones y limitaciones que determinan cómo se almacenan los datos y cómo se accede a ellos. Los modelos de bases de datos individuales se diseñan en base a las reglas y los conceptos de cualquier modelo de datos más amplio que los diseñadores adopten. A continuación, el modelo de BD ⁽⁵¹⁾.



2.8 Estándar de codificación empleado

Para la codificación en una tienda online, se pueden utilizar diferentes estándares y sistemas de codificación, como los siguientes: ⁽⁵²⁾

- Códigos de barras: Permiten identificar los productos de manera única y se pueden utilizar tanto a nivel interno como externo. Los códigos de barras más comunes son los códigos UPC y EAN.
- SKU: Son códigos internos que se utilizan para hacer seguimiento del inventario y generar informes de ventas.
- UTF-8: Es un estándar de codificación que permite una comunicación digital global y eficiente, diseñado para que diferentes sistemas de escritura puedan coexistir sin problemas.
- Estándares de clasificación: Son normas que identifican a los productos o servicios, siendo los codificadores aquellos instrumentos que solo asignan un código a elementos ya definidos por otras vías.

Es importante elegir un formato de codificación que tenga sentido para el negocio y que permita una gestión eficiente del inventario y las ventas. Por tanto, el que más se ajusta a mis necesidades es el Estándar de clasificación.

2.9 Conclusiones parciales.

Para desarrollar una aplicación web de tienda online, es esencial considerar requisitos funcionales y no funcionales clave, así como las características de los usuarios, la arquitectura de la aplicación y el modelo de datos. Los requisitos funcionales incluyen la gestión de productos, diseño de tienda, sistema de carrito de compras, y soporte multiplataforma. Los requisitos no funcionales abarcan una interfaz atractiva, eficiencia, integración con sistemas de pago y seguridad. Además, se consideran las historias de usuarios, la arquitectura adaptable, patrones de diseño, modelo de datos y estándares de codificación. Así en este capítulo describe la propuesta del sistema basado en los requisitos.

CAPÍTULO 3: VALIDACIÓN DE LA VIABILIDAD DE LA PROPUESTA DE SOLUCIÓN.

En el presente capítulo se valora la viabilidad de la propuesta de solución mediante la validación de los requisitos y el diseño propuesto, además de la realización de pruebas funcionales, unitarias y de aceptación a la aplicación web.

3.1 Técnicas de validación de requisitos

Con el objetivo de ratificar que los requisitos del software obtenidos definen el sistema que el cliente desea se llevó a cabo un proceso de validación de los mismos, para el cual se emplearon las siguientes técnicas: ⁽⁵³⁾

Revisiones de los requisitos: El equipo de desarrollo y el cliente realizaron revisiones exhaustivas de todos los requisitos. Durante las revisiones internas, se encontraron problemas técnicos y errores ortográficos, los cuales fueron corregidos a tiempo. En las reuniones con el cliente, se llevaron a cabo dos encuentros. En la primera reunión, el cliente expresó su satisfacción con el trabajo realizado por el equipo de desarrollo, aunque se realizaron ajustes en algunos requisitos funcionales para mejorarlo. En la segunda reunión, se realizaron más ajustes en los requisitos funcionales y finalmente se aprobaron todos los requisitos.

Diseño de prototipos web: Se presentaron prototipos al cliente, los cuales mostraban una aplicación web ejecutable. Esto le permitió al cliente tener una idea preliminar de cómo sería la aplicación y, a través de la interacción con los prototipos, se verificó si cumplían con sus necesidades. Los prototipos fueron aprobados por el cliente.

Generación de casos de prueba: Como parte del proceso de validación de los requisitos funcionales del sistema fueron diseñados pruebas de caja blanca y caja negra.

3.2 Métricas aplicadas a los requisitos

Se evaluó la calidad de la especificación de los requisitos utilizando la métrica de Calidad de la especificación (CE). Para determinar la comprensibilidad y precisión de los requisitos, se calculó el número total de requisitos en la especificación como se muestra a continuación:

Nr: *El total de requisitos de especificación.*

Nf: Cantidad de requisitos funcionales.

Nnf: Cantidad de requisitos no funcionales.

Como resultado de la sustitución de los valores, para el sistema se obtiene:

$$Nr = Nf + Nnf$$

$$Nr = 14 + 7$$

$$Nr = 21$$

Para determinar, finalmente, la Especificidad de los Requisitos (ER) o ausencia de ambigüedad en los mismos se realiza la siguiente operación:

$$ER = Nui / Nr$$

Donde ***Nui*** es el número de requisitos para los cuales todos los revisores tuvieron interpretaciones idénticas. Mientras más cerca de 1 esté el valor de ***ER***, menor será la ambigüedad. Para el caso de los requisitos obtenidos para la aplicación solo dos produjeron contradicción en las interpretaciones. Sustituyendo las variables se obtiene:

$$ER = 19 / 21$$

$$ER = 0.90$$

Arrojando un resultado final satisfactorio, indicando que el grado de ambigüedad de los requisitos es sumamente bajo (10%) ya que el 90% es entendible. Los requisitos ambiguos fueron modificados y validados para garantizar su correcta comprensión.

3.3 Validación del diseño

Las métricas de desarrollo de software son medidas utilizadas para evaluar el progreso, la calidad, la productividad y el estado del proceso de desarrollo de software. Estas métricas pueden ser cualitativas o cuantitativas, y se utilizan para medir la eficiencia del equipo de desarrollo, la calidad del software y la satisfacción del usuario, entre otros aspectos. Además, estas métricas también se utilizan para identificar áreas problemáticas, lo que permite desarrollar soluciones y mejorar el proceso de desarrollo del software. ⁽⁵⁴⁾

Las métricas para evaluar la calidad del diseño del sistema y su relación con los atributos de calidad son las siguientes:

- **Tamaño operacional de clases (TOC)**

El tamaño operacional de las clases se determina por el número de métodos asignados a una clase. Esto permite calcular el nivel de responsabilidad de los métodos, la

complejidad de su implementación y su reutilización, lo que nos permite evaluar la efectividad del diseño. Existe una relación directa con los dos primeros factores mencionados y una relación inversa con el último. ⁽⁵⁵⁾

Atributo de calidad	Modo en que lo afecta
Responsabilidad	Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.
Complejidad de implementación	Un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
Reutilización	Un incremento del TOC implica una disminución del grado de reutilización de la clase.

Tabla 5. Atributos de calidad evaluados por la métrica TOC.

Atributo de calidad	Categoría	Criterio
Responsabilidad	Baja	< =Prom.
	Media	Entre Prom y 2* Prom.
	Alta	> 2* Prom.
Complejidad de implementación	Baja	< =Prom.
	Media	Entre Prom y 2* Prom.
	Alta	> 2* Prom.
Reutilización	Baja	> 2*Prom.
	Media	Entre Prom. y 2* Prom
	Alta	<= Prom.

Tabla 6. Criterios de evaluación de la métrica TOC.

Resultados obtenidos de la aplicación de la métrica TOC

El siguiente gráfico refleja que la mayoría de las clases tienen de 1 a 6 procedimientos. Este resultado demuestra que el funcionamiento general del sistema está distribuido equitativamente entre la mayoría de las clases, de esta forma se garantiza que, en caso

de ocurrir algún cambio en el sistema de clases del componente, estaría menos comprometido el funcionamiento correcto del mismo.

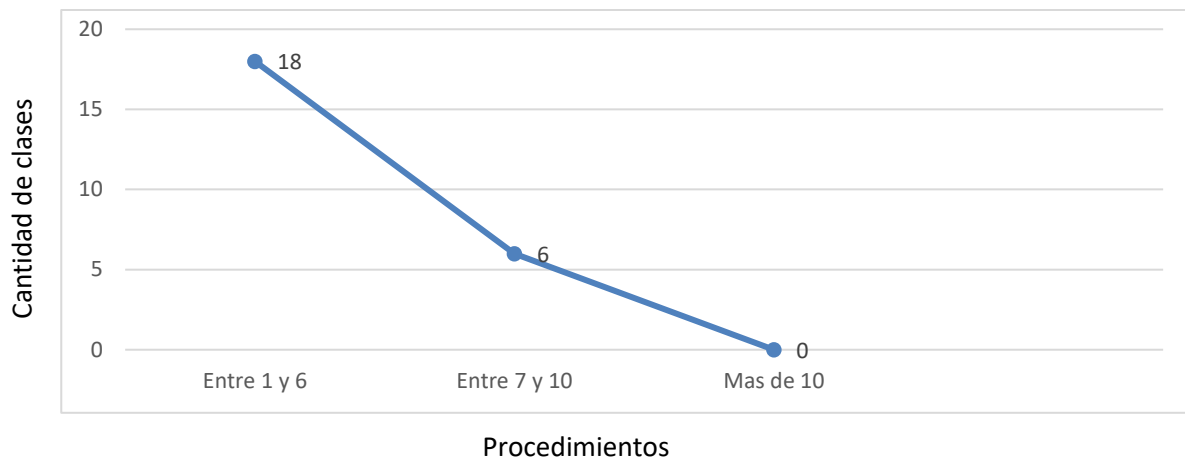


Figura 6. Resultados gráficos obtenidos de la aplicación de la métrica TOC.

La siguiente figura muestra la representación en % de los resultados obtenidos.

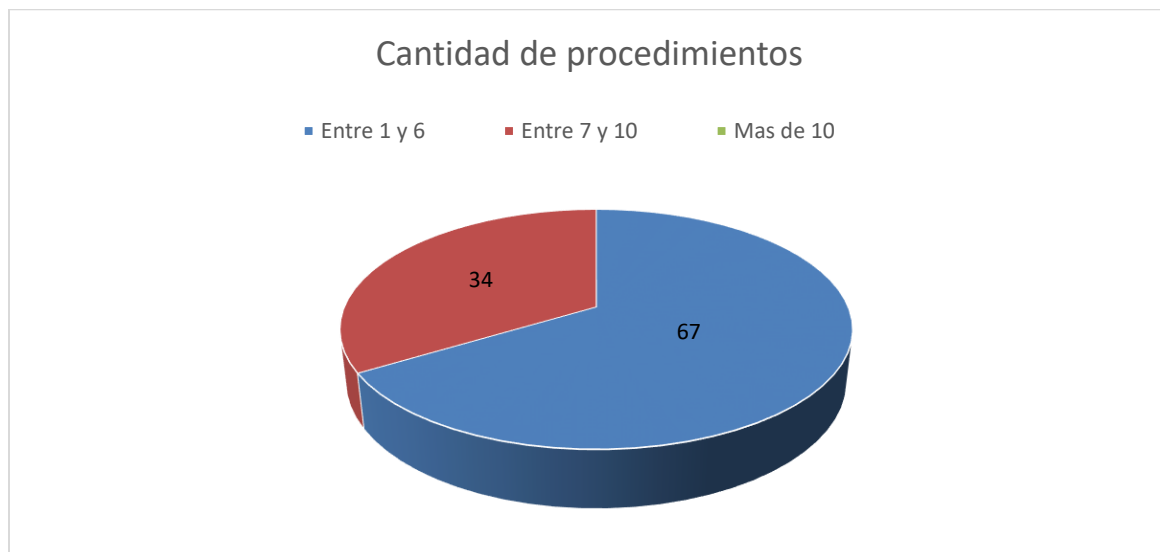


Figura 7. Resultados obtenidos de aplicar la métrica TOC en %.

A continuación, se muestra la representación de la incidencia de los resultados de la evaluación de la métrica TOC en los diferentes atributos.



Figura 8. Atributo responsabilidad con la métrica TOC.

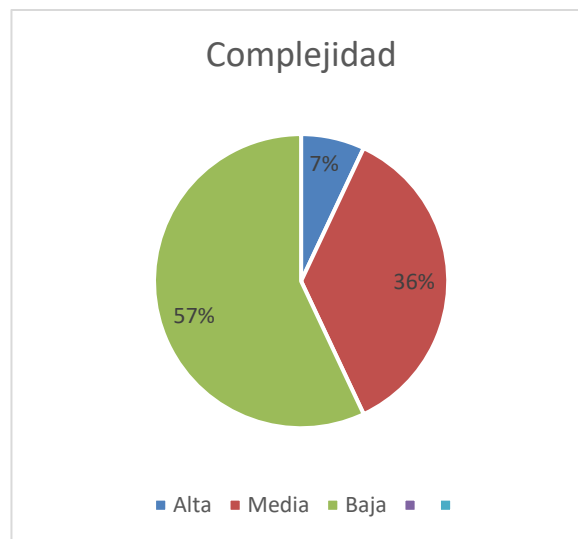


Figura 9.

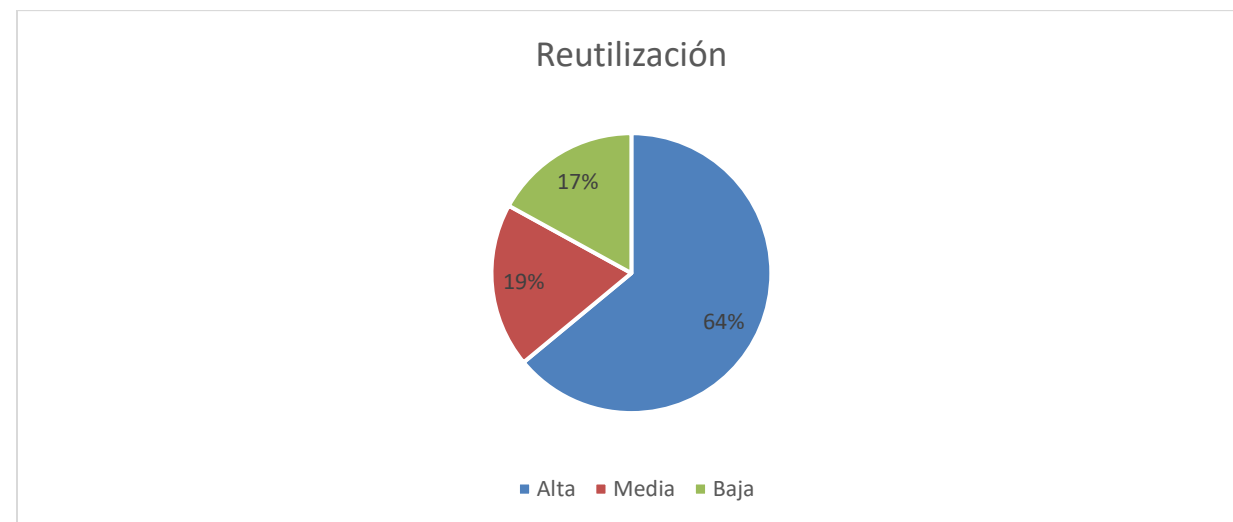


Figura 10. Atributo Reutilización con la métrica TOC.

Los resultados de la evaluación de la herramienta de medición TOC son positivos. El sistema que se analizó tiene una calidad aceptable porque el 64 % de las clases que forman parte de él se utilizan considerablemente. Además, estas mismas clases también obtuvieron buenas calificaciones en los atributos de calidad, lo que significa que no tienen tanta responsabilidad, no son tan complicadas y se reutilizan.

- **Relaciones entre clases**

Las relaciones entre las clases se pueden describir mediante el número de relaciones de uso que una clase tiene con otras, es decir, el número de dependencias que una clase tiene con otra. Estas relaciones permiten calcular métricas importantes como el acoplamiento, la complejidad de mantenimiento, la reutilización y la cantidad de pruebas necesarias para evaluar la calidad del diseño. En este caso, se puede afirmar que existen relaciones directas entre el acoplamiento, la complejidad de mantenimiento, la reutilización y la cantidad de pruebas, y una relación inversa con el número de relaciones de uso.

Atributo de calidad	Modo en que lo afecta
Acoplamiento	Un aumento del RC implica un aumento de la responsabilidad asignada a la clase.
Complejidad de mantenimiento	Un aumento del RC implica un aumento de la complejidad de implementación de la clase.
Reutilización	Un incremento del RC implica una disminución del grado de reutilización de la clase.
Cantidad de pruebas	Un aumento del RC implica un aumento de la Cantidad de pruebas de unidad necesarias para probar una clase.

Tabla 7. Atributos de calidad evaluados por la métrica RC.

Atributo de calidad	Modo en que lo afecta	
Acoplamiento	Ninguno	0
	Bajo	1
	Medio	2
	Alto	> 2
Complejidad de mantenimiento	Baja	< =Prom.
	Media	Entre Prom y 2* Prom.
	Alta	> 2* Prom.
Reutilización	Baja	> 2*Prom.
	Media	Entre Prom y 2* Prom
	Alta	<= Prom.
Cantidad de pruebas	Baja	< =Prom.
	Media	Entre Prom y 2* Prom.
	Alta	> 2* Prom.

Tabla 8. Criterios de evaluación para la métrica RC.

Resultados obtenidos de la aplicación de la métrica RC

La representación gráfica muestra los resultados obtenidos agrupados en intervalos definidos. Después de un análisis, se determinó que no es factible eliminar estas dependencias debido a la responsabilidad arquitectónica que las clases deben mantener. Esto se debe a la complejidad que conlleva rastrear todas las dependencias, lo que resulta en un aumento de la complejidad y la imposibilidad de eliminarlas sin introducir errores. ⁽⁵⁵⁾

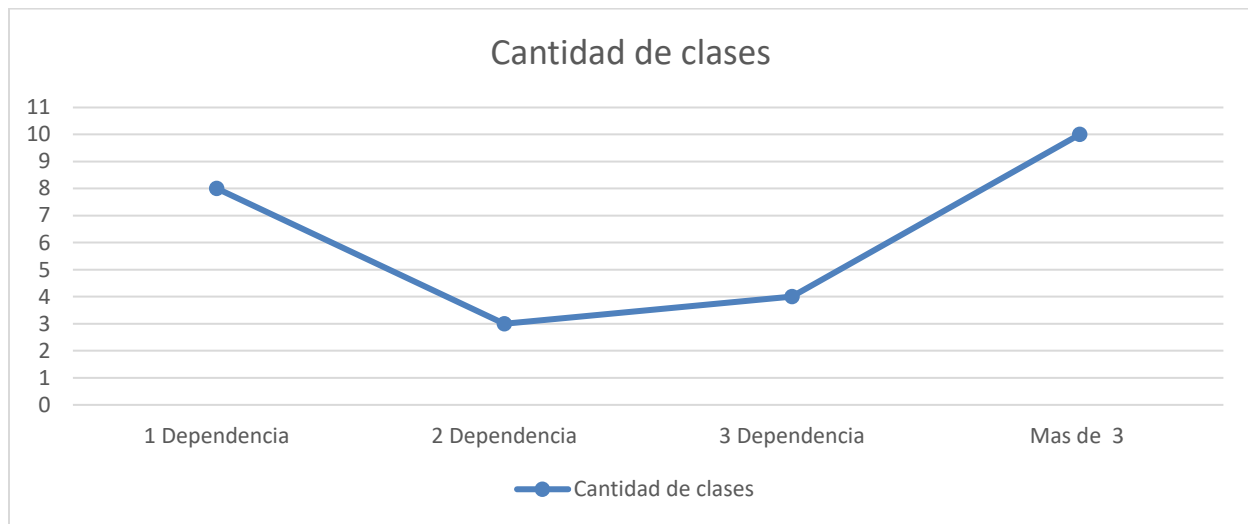


Figura 11. Representación de la evaluación de la métrica RC.

A continuación, se muestra la representación en % de los resultados obtenidos, agrupados en los intervalos definidos.

Cantidad de Dependencias

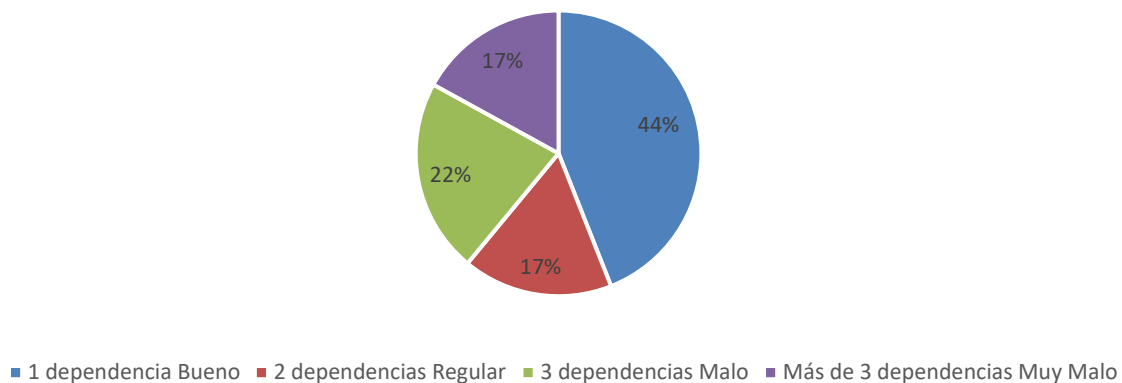


Figura 12. Representación en % de la evaluación de la métrica RC.

Acoplamiento

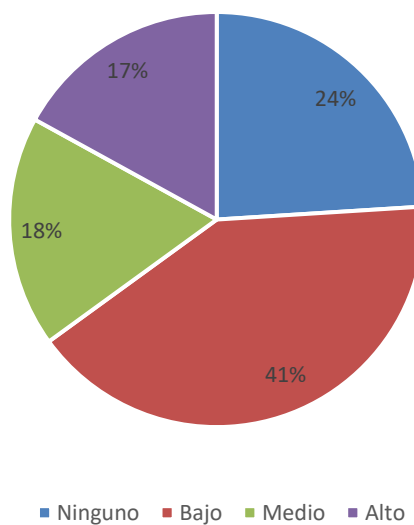


Figura 13. Atributo acoplamiento con la métrica RC.

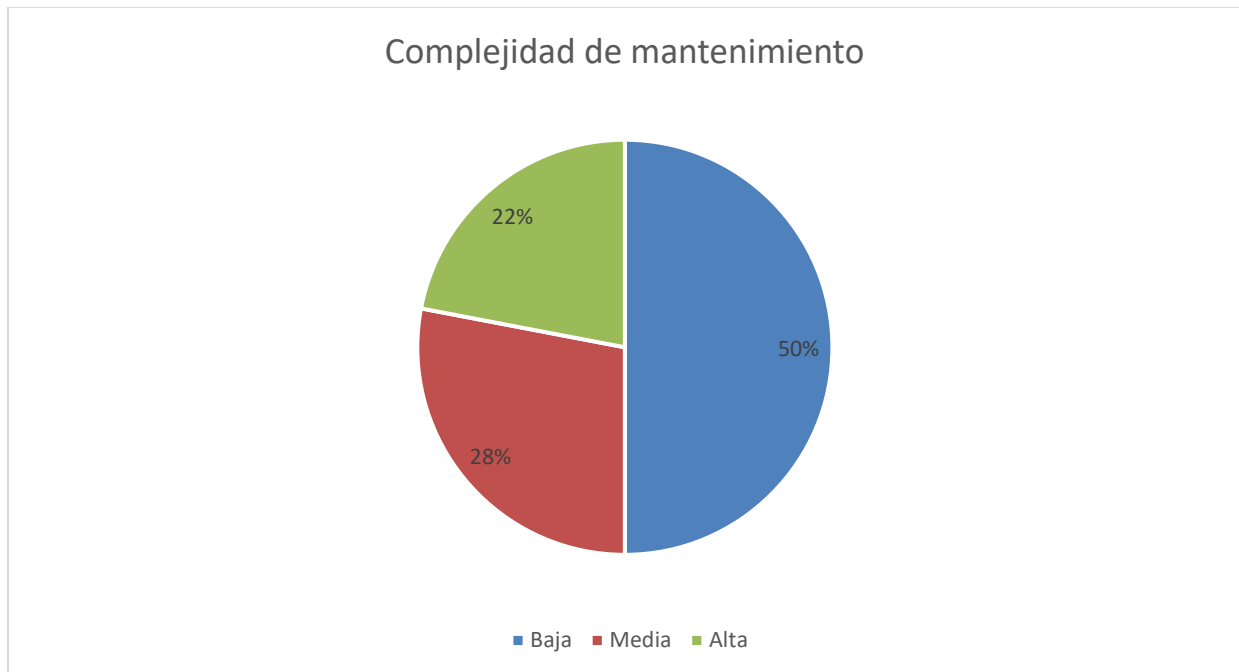


Figura 14. Atributo complejidad de mantenimiento con la métrica RC

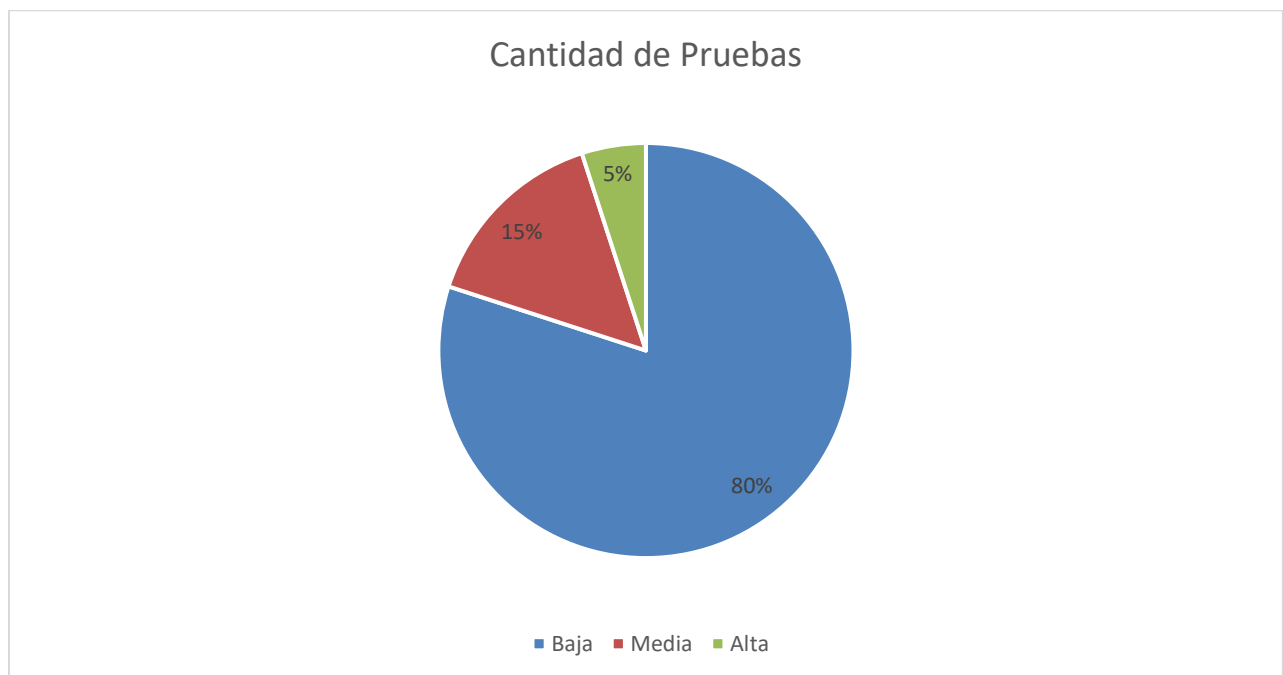


Figura 15. Representación de la evaluación de la métrica RC en el atributo Cantidad de pruebas.



Figura 16. Representación de la evaluación de la métrica RC en el atributo Reutilización.

Después de analizar los resultados obtenidos en la evaluación de la métrica RC, se puede concluir que el diseño del sistema cumple con los niveles de calidad requeridos. En particular, se observa que el 61% de las clases tiene entre 0 y 3 dependencias, lo que indica una elevada reutilización y baja complejidad. Además, el 41% de las clases tiene índices aceptables en cuanto a acoplamiento, lo que sugiere un bajo acoplamiento. Los atributos de complejidad de mantenimiento y cantidad de pruebas también se comportan satisfactoriamente, con un 50% y 80% de valor en las clases, respectivamente. En general, estos resultados confirman la elevada reutilización, el bajo acoplamiento, la baja complejidad y la baja cantidad de pruebas necesarias para evaluar la efectividad del diseño. ⁽⁵⁶⁾

3.4 Pruebas de software

Niveles de pruebas

Los niveles de pruebas permiten evaluar diferentes aspectos del software en distintas etapas de su desarrollo, contribuyendo a garantizar su calidad y correcto funcionamiento en el contexto del desarrollo de software. Estos se dividen en cuatro categorías

principales, que se enfocan en diferentes aspectos del diseño del sistema. Estos niveles son: ⁽⁵⁷⁾

1. Pruebas de componente o unitarias: Estas pruebas se realizan en los componentes, unidades o módulos del software, y se enfocan en verificar que los comportamientos funcionales y no funcionales del componente sean diseñados y especificados correctamente.

2. Pruebas de integración: Este nivel de prueba se centra en la interacción entre distintos componentes o sistemas, y busca encontrar defectos en las interfaces o en los componentes o sistemas.

3. Pruebas de sistema: En este nivel, se prueba el sistema como un todo, y se validan aspectos como la funcionalidad, el rendimiento y la seguridad.

4. Pruebas de aceptación: Estas pruebas se realizan en el sistema completo y en condiciones reales de uso, con el objetivo de validar que el sistema cumpla con los requisitos y expectativas del usuario.

En diferentes libros se han propuesto varias estrategias de prueba de software, todas ellas proporcionan una plantilla o brindan una guía al profesional y un conjunto de hitos al jefe de proyecto para la correcta realización de las pruebas. Estas estrategias deben incluir pruebas de bajo nivel que verifiquen que se ha implementado correctamente todos los pequeños segmentos de código y pruebas de alto nivel que validen que las funcionalidades del sistema tengan correspondencia con lo especificado por el cliente. Para los efectos de la presente investigación, se propone la siguiente estrategia de pruebas:

Tipo de Prueba	Método de prueba	Técnica de prueba
Unitarias	Caja blanca	Camino básico
Funcionalidad	Caja negra	Partición de equivalencia

Tabla 9. Estrategia de prueba aplicada a la solución.

3.4.1 Pruebas de caja blanca

Las pruebas de caja blanca, también conocidas como pruebas de caja transparente o pruebas de caja de cristal, son una técnica de prueba de software que se enfoca en la estructura interna y el diseño del sistema, analizando el código y la arquitectura para identificar posibles defectos y garantizar la funcionalidad correcta. Estas pruebas se realizan cuando el código es visible para los testers, y su objetivo principal es cubrir la estructura interna del sistema, incluyendo código, arquitectura y flujos de trabajo. Para ello se empleó la técnica de la **prueba de caminos básicos**, la cual consiste en un conjunto de pruebas que cubren los flujos de ejecución más simples y comunes del programa, garantiza que se pueda probar al menos una vez cada una de las sentencias del programa, partiendo de la obtención de la medida de la complejidad de un procedimiento o algoritmo y la obtención de un conjunto básico de caminos de ejecución de este, los cuales son utilizados para obtener los casos de prueba. ⁽⁵⁸⁾

Las pruebas de caja blanca son una herramienta esencial en el proceso de prueba de software que permite analizar el funcionamiento interno del sistema e identificar defectos en el código y la arquitectura. ⁽⁵⁸⁾

Seguidamente se muestra el proceso de pruebas realizado a la funcionalidad **signup**, específicamente a la controladora **users** por ser de mayor criticidad en el sistema.

Primeramente, se comienza por analizar el código y enumerar las instrucciones:

```
33 @user_not_authenticated
34 def signup(request): /1
35     if request.method == 'POST': /2
36         form = SignupForm(request.POST)
37         if form.is_valid(): /3
38             user = form.save(commit=False)
39             user.is_active = False
40             user.save()
41             activate_email(request, user)
42             watchdog(request,description=f'Nuevo usuario creado "{user.username}"',operation='usuario')
43             return redirect('core:index') /4
44         else: /5
45             return render(request, 'users/signup.html', {
46                 'form': form
47             }) /6
48     else: /7
49         form = SignupForm()
50         return render(request, 'users/signup.html', {
51             'form': form
52         }) /8
```

Para obtener los casos de prueba a partir de la técnica seleccionada se debe construir seguidamente el grafo de flujo correspondiente al código de la función como se muestra en la Figura

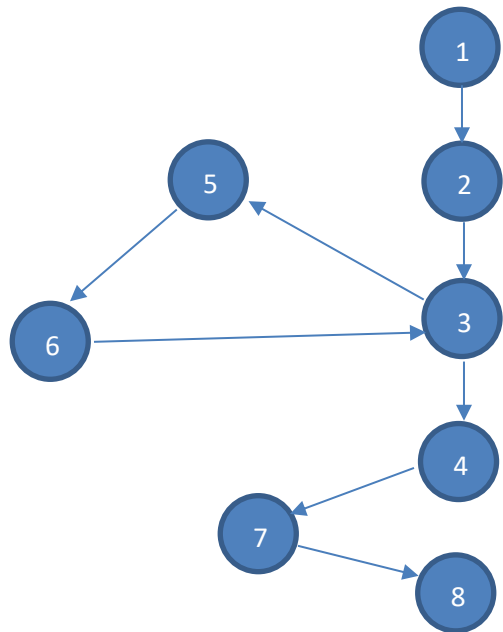


Figura 17. Representación de la evaluación de la métrica RC en el atributo Reutilización.

Luego se determina la complejidad ciclomática $V(G)$ del grafo resultante, la cual es un indicador del número de caminos independientes que existen en un grafo, es decir, es cualquier camino dentro del código que introduce por lo menos un nuevo conjunto de sentencias de proceso o una nueva condición. La complejidad ciclomática puede ser calculada de 3 formas:

1. $V(G) = a - n + 2$, siendo a el número de arcos o aristas del grafo y n el número de nodos.
2. $V(G) = r$, siendo r el número de regiones cerradas del grafo.
3. $V(G) = c + 1$, siendo c el número de nodos de condición.

Realizando los cálculos correspondientes se obtiene por cualquiera de las variantes el siguiente resultado:

$$1. V(G) = a - n + 2$$

$$V(G) = 8 - 8 + 2$$

$$V(G) = 2$$

$$2. V(G) = 2$$

$$3. V(G) = c + 1$$

$$V(G) = 1 + 1$$

$$V(G) = 2$$

Por lo que el conjunto de caminos básico sería:

- Camino básico 1: 1-2-3-4-8
- Camino básico 2: 1-2-3-5-6-7-3-4-8

Luego se definen los casos de prueba para cada uno de los caminos básicos obtenidos. A continuación, se muestra el resultado de las pruebas aplicadas a los caminos básicos 1 y 2.

Descripción: Se verificará que se agregue la información del usuario al sistema.
Condición de ejecución: Para ello debe haber seleccionado la opción autenticarse.
Entrada: Datos de usuario correctos
Resultados esperados: Que se inserte correctamente el nombre de usuario y la contraseña al sistema.
Resultado obtenido: Satisfactorio.

Tabla 10. Caso de prueba del camino 1

Descripción: Se verificará que se agregue la información del usuario al sistema.
Condición de ejecución: Para ello debe haber seleccionado la opción autenticarse.
Entrada: Datos de usuario incorrectos.

Resultados esperados: Que no se inserte correctamente el nombre de usuario y la contraseña al sistema y muestre un mensaje de identificación incorrecta.
Resultado obtenido: Satisfactorio.

Tabla 11. Caso de prueba del camino 2

Los resultados obtenidos están relacionados con la cantidad de rutas independientes que existen en una sección de código, lo cual es fundamental para comprender la complejidad y el diseño de casos de prueba correspondientes.

3.4.2 Pruebas de caja negra

Las pruebas de caja negra, también conocidas como "pruebas de comportamiento", se enfocan en validar los requisitos funcionales del software. Estas pruebas se realizan sin conocer el funcionamiento interno del sistema, centrándose en las entradas y salidas del software. Permiten descubrir distintos tipos de errores en áreas como funciones ausentes o incorrectas, problemas de interfaz, errores en la manipulación de datos o en el acceso a bases de datos externas, cuestiones de rendimiento, así como errores en la inicialización y terminación del sistema ⁽⁵⁸⁾.

Para desarrollar las pruebas de caja negra existen varias técnicas, entre ellas se encuentran:

- Técnica de la Partición de Equivalencia: divide el campo de entrada en clases de datos que ejercitan determinadas funciones del software.
- Técnica del Análisis de Valores Límites: prueba la habilidad del programa para manejar datos en los límites aceptables.
- Técnica de Grafos de Causa-Efecto: permite validar complejos conjuntos de acciones y condiciones.

La técnica de la Partición de Equivalencia es una de las más efectivas, ya que permite examinar los valores válidos e inválidos de las entradas existentes en el software, descubriendo de forma inmediata una clase de errores que, de otro modo, requerirían la

ejecución de muchos casos antes de detectar el error genérico. Esta técnica se dirige a la definición de casos de pruebas que descubran clases de errores. ⁽²⁰⁾

Con el objetivo de comprobar las funcionalidades del sistema se utilizó la técnica de partición de equivalencia cuya idea es identificar clases de datos que sean "equivalentes" y asegurarse de que probamos al menos un caso de cada clase.

En el caso del requisito funcional **Autenticarse** se aplica la técnica de partición de equivalencia de la siguiente manera:

Usuario:

Datos Válidos:

-Nombre de Usuario incorrecto

Contraseña:

Datos Válidos:

-Contraseña incorrecta.

A continuación, a través de la siguiente tabla se le hacen varias pruebas con juegos de datos válidos e inválidos al requisito funcional Autenticarse donde se conoce la respuesta del sistema con dichos datos, el cual el resultado es satisfactorio.

Pruebas con juegos de datos	Respuesta del sistema
Nombre de usuario incorrecto.	Autenticarse Usuario o correo electrónico Contraseña Mostrar contraseña ☐ Usuario o contraseña incorrectos. Por favor verifique. Entrar ¿Olvidó su contraseña? ¿Aún no tiene usuario? Regístrese aquí

Contraseña incorrecta.	Autenticarse edllerena777@gmail.com Contraseña Mostrar contraseña ☐ Usuario o contraseña incorrectos. Por favor verifique. Entrar ¿Olvidó su contraseña? ¿Aún no tiene usuario? Regístrese aquí
-------------------------------	--

Tabla 12: Prueba de caja negra al requisito funcional Iniciar sesión

3.5 Validación de la solución propuesta

La validación de la solución propuesta es esencial para garantizar que el software cumple con las expectativas del cliente y resuelve eficazmente el problema planteado. En el caso de la aplicación web, se deben establecer escenarios de validación que abarquen diferentes situaciones de uso para asegurar su correcto funcionamiento y cumplimiento de los requisitos. Este proceso es fundamental para garantizar la calidad y fiabilidad del software desarrollado. ⁽⁵⁹⁾⁽⁶⁰⁾

Como estrategia para la validación del software se establecieron un conjunto de escenarios poniendo a prueba el sistema.

Escenario de prueba #1:

Un usuario desea entrar en la aplicación web para visualizar las ofertas del menú de productos y realizar compras.

El usuario podrá visualizar el menú de productos ofertados, pero no podrá comprar las ofertas ya que se establecieron roles y permisos, que requiere estar registrado en el

sistema. El sistema permite que cuando un usuario inicie sesión, sean interfaces distintas, como se muestra a continuación.



Figura 18: Interfaz del usuario sin estar autenticado en el sistema.



Figura 19: Interfaz del usuario autenticado en el sistema.

Escenario de prueba #2:

Un usuario desea entrar en la aplicación web para visualizar los pedidos que ha realizado.

El usuario podrá visualizar los pedidos que ha realizado, pero tendrá que estar registrado en el sistema y desplazarse a la pestaña tienda y en ella seleccionar la opción carro donde podrá ver todos los pedidos que ha realizado.

The screenshot shows the login page of the 'Tienda Las Pailas de Cemento' website. The header includes the company logo and name, along with navigation links for 'Productos', 'Acerca de', and 'Contáctenos'. Below the header is a navigation bar with a home icon and the text 'Autenticarse'. The main heading is 'Autenticarse'. A red-bordered box contains a 'NOTA NECESARIA' regarding the use of provisional bags. The login form includes fields for 'Usuario o correo electrónico' and 'Contraseña', a 'Mostrar contraseña' checkbox, and an 'Entrar' button. Links for '¿Olvidó su contraseña?' and '¿Aún no tiene usuario? Regístrese aquí' are at the bottom of the form.

Figura 19: Interfaz de la autenticación.

The screenshot shows the shopping cart page of the 'Tienda Las Pailas de Cemento' website. The header is identical to the login page. The navigation bar now includes 'Productos', 'Acerca de', 'Tienda', 'Contáctenos', and 'edgar'. Below the navigation bar, a breadcrumb trail shows 'Carro de compra' and 'Carro'. The main heading is 'Carro de compras'. A red-bordered box contains the same 'NOTA NECESARIA' as the login page. Below this, a box indicates 'Carro vacío.'.

Figura 19: Interfaz del Carro de compras donde se guardan los pedidos realizados.

3.6 Conclusiones Parciales

La validación de la viabilidad de la propuesta de solución es un proceso crucial en el desarrollo de una aplicación web de tienda online, en este se utilizan diversas técnicas de validación de requisitos, métricas y pruebas de software para garantizar que la solución propuesta sea eficiente y efectiva. A partir de la aplicación de estas técnicas y métricas, se obtuvo que la viabilidad y eficacia de la aplicación web son satisfactorias, y que la validación de requisitos y la arquitectura, junto con las pruebas de software y la retroalimentación de los clientes, permiten ajustar y refinar la solución propuesta, garantizando que se cumplan los objetivos y expectativas del proyecto.

Conclusiones Generales

La gestión de ventas para Cementos Cienfuegos S.A. representa la capacidad de la empresa para administrar eficientemente el proceso de venta de sus productos. Esto implica la utilización de herramientas tecnológicas que permitan agilizar y mejorar este proceso, como lo es el desarrollo de una aplicación web. Esta herramienta no solo beneficiará a la empresa en términos de control y eficiencia en sus operaciones, sino que también brindará a los clientes la comodidad de realizar sus compras en línea y acceder a una variedad de productos.

Cienfuegos Cementos S.A. ha llegado a plantearse la necesidad de adquirir divisas para poder continuar de la mejor forma el proceso productivo, lo que ha llevado al desarrollo de una aplicación web para la gestión de las ventas de los productos ofertados por esta entidad.

A partir de la revisión teórica, se afirma que existen diversas estrategias y técnicas para lograr un posicionamiento adecuado en Internet, además de combinar estrategias y técnicas efectivas con el uso de tecnologías y herramientas apropiadas. Como también mantenerse actualizado en este ámbito para poder adaptarse a las demandas del mercado y maximizar la visibilidad y el impacto de Cementos Cienfuegos S.A. en línea.

En el desarrollo se detectan 14 requisitos funcionales y 7 no funcionales claves. Los requisitos funcionales incluyen la gestión de productos, diseño de tienda, sistema de carrito de compras, y soporte multiplataforma. Los requisitos no funcionales abarcan una interfaz atractiva, eficiencia, integración con sistemas de pago y seguridad. Además, se consideran las características de los usuarios, la arquitectura de la aplicación y el modelo de datos, así como, historias de usuarios, la arquitectura adaptable, patrones de diseño, y estándares de codificación.

A partir de la aplicación de estas técnicas y métricas, se obtiene que la viabilidad y eficacia de la aplicación web son satisfactorias, y que la validación de requisitos y la arquitectura, junto con las pruebas de software y la retroalimentación de los clientes, permiten ajustar y refinar la solución propuesta, garantizando que se cumplan los objetivos y expectativas del proyecto.

Por lo tanto, se concluye que con el desarrollo de la aplicación web se ofrece a los clientes la comodidad de realizar sus compras desde cualquier lugar o dispositivo con acceso a internet. Además, que permite acceder a una amplia variedad de productos, comparar precios, y aprovechar ofertas y descuentos. Todo esto facilita la gestión de ventas de Cementos Cienfuegos S.A.

Recomendaciones

Para el correcto funcionamiento de la aplicación web de Cementos Cienfuegos S.A. se recomiendan las siguientes acciones:

- Ofrecer atención al cliente a través de la aplicación web, para resolver dudas y problemas de manera rápida y eficiente.
- Mantener la aplicación web actualizada con información relevante sobre productos, promociones, eventos, etc.
- Realizar campañas de marketing digital para promocionar la aplicación web y atraer nuevos usuarios.
- Implementar un sistema de gestión de clientes (CRM) para poder gestionar de manera eficiente las relaciones con los clientes y potenciales clientes.

Referencias Bibliográficas

- [1] M. M. Canelo, «Qué son los Patrones de Diseño de software / Design Patterns», *Profile Software Services*, 24 de junio de 2020. <https://profile.es/blog/patrones-de-diseno-de-software/> (accedido 30 de septiembre de 2023).
- [2] «Qué es el comercio electrónico: características, ventajas y ejemplos». <https://blog.hubspot.es/sales/comercio-electronico> (accedido 7 de diciembre de 2023).
- [3] «Comercio electrónico | Acelera pyme». <https://www.acelerapyme.gob.es/kit-digital/comercio-electronico> (accedido 7 de diciembre de 2023).
- [4] M. Sánchez, «Tienda online cubana TuEnvío 2.0 prueba proceso de registro», *Directorio Cubano*, 19 de marzo de 2022. <https://www.directoriocubano.info/panorama/tienda-online-cubana-tuenvio-2-0-prueba-proceso-de-registro/> (accedido 7 de diciembre de 2023).
- [5] «Tienda virtual: ¿qué es y cómo crear una que sea exitosa?», *Rock Content - ES*, 28 de octubre de 2019. <https://rockcontent.com/es/blog/tienda-virtual/> (accedido 7 de diciembre de 2023).
- [6] «¿Qué es una Tienda Virtual?: Características - La Mina Digital». <https://www.laminadigital.es/que-es-una-tienda-virtual/> (accedido 7 de diciembre de 2023).
- [7] «10 ventajas de tener una tienda online para tu negocio - Digival.es». <https://www.digival.es/blog/10-ventajas-tienda-online/> (accedido 7 de diciembre de 2023).
- [8] «8 Ventajas de tener una tienda online o e-Commerce | Baética». <https://baetica.com/ventajas-tienda-online/> (accedido 7 de diciembre de 2023).
- [9] «Construyamos juntos | Cementos Cienguegos S.A.». <https://www.cementoscienguegos.com/es> (accedido 7 de diciembre de 2023).
- [10] «La guía sencilla para la representación de UML mediante diagramas y la creación de modelos de base de datos». <https://www.microsoft.com/es-es/microsoft-365/business-insights-ideas/resources/guide-to-uml-diagramming-and-database-modeling> (accedido 7 de diciembre de 2023).
- [11] «▷ Métodos de investigación». <https://www.sdelsol.com/blog/tendencias/metodos-de-investigacion/> (accedido 7 de diciembre de 2023).
- [12] Cimec, «Entrevista en Profundidad | Qué es y cómo realizarla», *CIMEC*, 21 de julio de 2023. <https://www.cimec.es/entrevista-en-profundidad/> (accedido 7 de diciembre de 2023).
- [13] «Servicios informáticos para empresas en Barcelona | Tic-Systems». <https://tic-systems.com/es/> (accedido 7 de diciembre de 2023).
- [14] «▷ ¿Cómo obtener el mejor soporte informático para una tienda online? - Empresas Mantenimiento Informático». <https://www.empresasmantenimientoinformatico.com/soporte-informatico-tienda-online/> (accedido 7 de diciembre de 2023).
- [15] «Las mejores metodologías para un correcto desarrollo de software», 27 de abril de 2022. <https://www.occamagenciadigital.com/blog/las-mejores-metodologias-para-un-correcto-desarrollo-de-software> (accedido 7 de diciembre de 2023).
- [16] «<http://6327868.hs-sites.com/hs-web-interactive-6327868-131587635123>». <http://6327868.hs-sites.com/hs-web-interactive-6327868-131587635123> (accedido 7 de diciembre de 2023).
- [17] «Metodologías Ágiles vs Tradicionales», *POSITIVO*. <https://www.positivo.pro/blog/metodologias-agiles-vs-tradicionales/> (accedido 7 de diciembre de 2023).

- [18] «Servicios informáticos para empresas en Barcelona | Tic-Systems». <https://tic-systems.com/es/> (accedido 7 de diciembre de 2023).
- [19] «AUP Ingenieria de Software». https://ingenieriadesoftware.mex.tl/63758_aup.html (accedido 7 de diciembre de 2023).
- [20] «METODOLOGIA AUP - Coggle Diagram». <https://coggle.it/diagram/YIx3ohnhhyOhuCAg/t/metodologia-aup> (accedido 7 de diciembre de 2023).
- [21] «METODOLOGÍAS ÁGILES. Proceso Unificado Ágil (AUP) Ingeniería del Software II Análisis de Sistemas - PDF Descargar libre». <https://docplayer.es/27314349-Metodologias-agiles-proceso-unificado-agil-aup-ingenieria-del-software-ii-analisis-de-sistemas.html> (accedido 8 de diciembre de 2023).
- [22] «Ingenieria de Soporte Logico - Agile Unified Process (AUP)». <https://sites.google.com/site/ingsoportelogico/desarrollo-%C3%A1gil-de-software/agile-unified-process-aup> (accedido 8 de diciembre de 2023).
- [23] www.dreams.es, «Patrones de arquitectura de software Desarrolladores Páginas Web», *dreams*, 7 de octubre de 2019. <http://www.dreams.es/transformacion-digital/desarrolladores-paginas-web/patrones-de-arquitectura-de-software> (accedido 8 de diciembre de 2023).
- [24] «03_apit_estilos_arquitectonicos_nueva(1).pdf».
- [25] «Frameworks de desarrollo de software: tendencias para 2023 - Apiumhub». <https://apiumhub.com/es/tech-blog-barcelona/frameworks-de-desarrollo-de-software-tendencias-para-2023/> (accedido 8 de diciembre de 2023).
- [26] «Frameworks en el desarrollo web | WAM». <https://www.wearemarketing.com/es/blog/frameworks-en-el-desarrollo-web-las-mejores-practic-as-para-tu-negocio-online.html> (accedido 8 de diciembre de 2023).
- [27] «¿Qué es Django? - Explicación del software Django - AWS». <https://aws.amazon.com/es/what-is/django/> (accedido 8 de diciembre de 2023).
- [28] D. A., «¿Qué es Bootstrap? - Una guía para principiantes», *Tutoriales Hostinger*, 28 de agosto de 2020. <https://www.hostinger.es/tutoriales/que-es-bootstrap> (accedido 8 de diciembre de 2023).
- [29] «Paradigma de programación», *Wikipedia, la enciclopedia libre*. 29 de octubre de 2023. Accedido: 8 de diciembre de 2023. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=Paradigma_de_programaci%C3%B3n&oldid=154946258
- [30] M. M. Canelo, «¿Qué son los paradigmas de programación?», *Profile Software Services*, 9 de junio de 2020. <https://profile.es/blog/que-son-los-paradigmas-de-programacion/> (accedido 8 de diciembre de 2023).
- [31] «Paradigma de la programación orientada a objetos». https://ferestrepoca.github.io/paradigmas-de-programacion/poo/poo_teoría/index.html (accedido 8 de diciembre de 2023).
- [32] «Software - CASE Herramientas». https://www.tutorialspoint.com/es/software_engineering/case_tools_overview.htm (accedido 8 de diciembre de 2023).
- [33] E. M. Alonso, «Herramientas CASE para el proceso de desarrollo de Software», *Monografias.com*, 10 de agosto de 2009.

- <https://www.monografias.com/trabajos73/herramientas-case-proceso-desarrollo-software/herramientas-case-proceso-desarrollo-software> (accedido 8 de diciembre de 2023).
- [34] «Free UML Tool». <https://www.visual-paradigm.com/solution/freeumltool/> (accedido 8 de diciembre de 2023).
- [35] «¿Cuáles son los 10 lenguajes de programación más usados en la actualidad?» <https://kodigo.org/cuales-son-los-10-lenguajes-de-programacion-mas-usados-en-la-actualidad/> (accedido 8 de diciembre de 2023).
- [36] «Los 10 lenguajes de programación más utilizados en la actualidad», *Talently Blog*, 29 de agosto de 2022. <https://talently.tech/blog/los-10-lenguajes-de-programacion-mas-utilizados/> (accedido 8 de diciembre de 2023).
- [37] «Estas son las características de Python: ¡domina el código! | Tokio». <https://www.tokioschool.com/noticias/caracteristicas-principales-de-python/> (accedido 8 de diciembre de 2023).
- [38] «¿Qué es JavaScript? - Explicación de JavaScript (JS) - AWS», *Amazon Web Services, Inc.* <https://aws.amazon.com/es/what-is/javascript/> (accedido 8 de diciembre de 2023).
- [39] «Fundamentos de JavaScript - Aprende desarrollo web | MDN», 18 de julio de 2023. https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/JavaScript_basics (accedido 8 de diciembre de 2023).
- [40] D. Team, «IDE : ¿Qué es un Entorno de Desarrollo Integrado?», *Formation Data Science | DataScientest.com*, 2 de septiembre de 2022. <https://datascientest.com/es/ide-que-es> (accedido 8 de diciembre de 2023).
- [41] «Visual Studio: IDE y Editor de código para desarrolladores de software y Teams», *Visual Studio*. <https://visualstudio.microsoft.com/es/> (accedido 8 de diciembre de 2023).
- [42] «SGBD: introducción al sistema gestor de base de datos - IONOS». <https://www.ionos.es/digitalguide/hosting/cuestiones-tecnicas/sistema-gestor-de-base-de-datos-sgbd/> (accedido 8 de diciembre de 2023).
- [43] «MySQL: el gran tutorial para principiantes», *IONOS Digital Guide*, 8 de enero de 2020. <https://www.ionos.es/digitalguide/servidores/know-how/guia-para-aprender-a-utilizar-mysql/> (accedido 8 de diciembre de 2023).
- [44] B. V, «¿Qué es un servidor web y cómo funciona?», *Tutoriales Hostinger*, 9 de febrero de 2022. <https://www.hostinger.es/tutoriales/que-es-un-servidor-web> (accedido 8 de diciembre de 2023).
- [45] G. B, «¿Qué es Apache? Descripción completa», *Tutoriales Hostinger*, 31 de agosto de 2018. <https://www.hostinger.es/tutoriales/que-es-apache/> (accedido 8 de diciembre de 2023).
- [46] Asana, «Historias de usuario: 3 ejemplos para generar valor para el usuario [2022] • Asana», *Asana*. <https://asana.com/es/resources/user-stories> (accedido 8 de diciembre de 2023).
- [47] «MVT Architecture Of Django. The data guys needed a common interface... | by Atufa Shireen | Medium». <https://atufashireen.medium.com/mvt-architecture-of-django-fcb93cb49dbe> (accedido 8 de diciembre de 2023).
- [48] «GRASP - General Responsibility Assignment Software Patterns Explained — Kamil Grzybek». <https://www.kamilgrzybek.com/blog/posts/grasp-explained> (accedido 8 de diciembre de 2023).

- [49] «GRASP Design Patterns in Object-Oriented Design - Video & Lesson Transcript», *study.com*. <https://study.com/academy/lesson/grasp-design-patterns-in-object-oriented-design.html> (accedido 8 de diciembre de 2023).
- [50] «Patrones de diseño - Abstract Factory - 🎮 The power ups 🍌». <https://thepowerups-learning.com/patrones-de-diseno-abstract-factory/> (accedido 8 de diciembre de 2023).
- [51] «Abstract Factory». <https://refactoring.guru/es/design-patterns/abstract-factory> (accedido 8 de diciembre de 2023).
- [52] «Facade pattern: definición, ejemplos y representación en diagrama UML - IONOS». <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/facade-pattern/> (accedido 8 de diciembre de 2023).
- [53] «Qué es un modelo de base de datos», *Lucidchart*. <https://www.lucidchart.com/pages/es/que-es-un-modelo-de-base-de-datos> (accedido 8 de diciembre de 2023).
- [54] «Cómo Proteger la Información Personal: Una Guía para Negocios | Comisión Federal de Comercio». <https://www.ftc.gov/es/guia-para-negocios/como-proteger-la-informacion-personal-una-guia-para-negocios> (accedido 8 de diciembre de 2023).
- [55] «IS__Libro_Sommerville_9.pdf».
- [56] «IS__Libro_Pressman_7.pdf».
- [57] S. Siriwardhana, «El Mejor Tutorial de Diagramas de Clase Para Ayudar a Modelar sus Sistemas Fácilmente», *Blog de Creately*, 25 de octubre de 2020. <https://creately.com/blog/es/diagramas/tutorial-de-diagrama-de-clases/> (accedido 8 de diciembre de 2023).
- [58] «Pruebas de integración de software: qué son, niveles y tipos». <https://www.testingit.com.mx/blog/pruebas-de-integracion-de-software> (accedido 8 de diciembre de 2023).
- [59] «Pruebas de caja blanca profundizando en el funcionamiento interno de las pruebas de aceptación», *FasterCapital*. <https://fastercapital.com/es/contenido/Pruebas-de-caja-blanca--profundizando-en-el-funcionamiento-interno-de-las-pruebas-de-aceptacion.html> (accedido 8 de diciembre de 2023).
- [60] «Qué es la validación de un proceso o método - TCM Consultoría y Formación». <https://www.tcmetrologia.com/blog/validacion/> (accedido 8 de diciembre de 2023).
- [61] «Historia del e-commerce: cómo nació el comercio electrónico». <https://es.linkedin.com/pulse/historia-del-e-commerce-c%C3%B3mo-naci%C3%B3-el-comercio-electr%C3%B3nico-gtdchile> (accedido 9 de diciembre de 2023)