

INSTITUTO SUPERIOR TECNOLOGICO

PARTICULAR

**“BOLÍVAR MADERO VARGAS”**

**TRABAJO DE GRADUACIÓN**

PREVIO A LA OBTENCIÓN DEL TITULO DE TECNÓLOGO SUPERIOR EN DESARROLLO DE SOFTWARE

**TEMA:**

ANALISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA INFORMATICO WEB DESARROLLO DE UN SISTEMA DE FACTURACION ELECTRONICA PARA EL LOCAL “LUCKY COMPUTERS” UBICADO EN LA CIUDAD DE MACHALA.

**AUTOR:**

CRISTHIAN ANDRES ROJAS CABRERA

**DIRECTORA:**

ING. ANDREA CARTUCHE JARA

**AÑO LECTIVO**

**2024-2025**

**MACHALA EL ORO ECUADOR**



**INSTITUTO SUPERIOR TECNOLOGICO**

**PARTICULAR**

**“BOLÍVAR MADERO VARGAS”**

**TRIBUNAL DE GRADO**

**CERTIFICAMOS:**

Que, el presente **TRABAJO DE GRADUACIÓN, ANALISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA INFORMATICO WEB DESARROLLO DE UN SISTEMA DE FACTURACION ELECTRONICA PARA EL LOCAL “LUCKY COMPUTERS” UBICADO EN LA CIUDAD DE MACHALA,** realizado por Sr. **CRISTHIAN ANDRES ROJAS CABRERA** del **SEXTO NIVEL** y bajo la dirección de la Srta. **Ing. Andrea Cartuche Jara**, previó a la obtención del Título de **TECNÓLOGO SUPERIOR EN DESARROLLO DE SOFTWARE**, una vez **REVISADO**, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_con lo dispuesto en el Art.164 del Reglamento Interno del Plantel, por lo que procedemos a\_\_\_\_\_\_\_\_\_\_\_ y \_\_\_\_\_\_\_\_\_\_\_\_\_con \_\_\_\_\_\_\_\_\_\_\_ **PUNTOS** sobre **DIEZ PUNTOS** Equivalentes a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Machala, a \_\_\_\_ de\_\_\_\_\_\_\_del 2023.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Miembro del Tribunal Miembro del Tribunal

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Miembro Tribunal

**DECLARACIÓN DE RESPONSABILIDAD**

CRISTHIAN ANDRES ROJAS CABRERA.

DECLARO QUE:

EL proyecto de grado denominado: **ANALISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE FACTURACION ELECTRONICA PARA EL LOCAL “LUCKY COMPUTERS” UBICADO EN LA CIUDAD DE MACHALA**, ha sido desarrollado en base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas que constan el pie de páginas correspondiente, cuyas fuentes se incorporan en la bibliografía. Consecuentemente este trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del proyecto de grado en mención.

Machala, 26 de abril 2025

***Cristhian Andrés Rojas Cabrera***

**CERTIFICADO**

ANDREA CARTUCHE JARA

Que el trabajo titulado **ANALISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE FACTURACION ELECTRONICA PARA EL LOCAL “LUCKY COMPUTERS” UBICADO EN LA CIUDAD DE MACHALA*,*** realizado por el Sr. Juan José Flores Pérezha sido guiado y revisado periódicamente y cumple normas estatuarias establecidas por la institución.

Machala, 28 de abril de 2025

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ing. Andrea Cartuche Jara

**DIRECTORA DE TESIS**

**DEDICATORIA**

**DEDICATORIA**

Dedico este trabajo con todo mi amor y gratitud a mis padres, **Gemny Patricia Cabrera Romero** y **William Hernán Rojas Mayo**, por ser mi mayor inspiración y apoyo incondicional en cada etapa de mi vida. Gracias por sus sacrificios, enseñanzas y por creer siempre en mí. Este logro es tan mío como suyo.

***Cristhian Andrés Rojas Cabrera***

**AGRADECIMIENTO**

A mis compañeros de estudio, quienes desde el inicio de esta etapa académica supieron brindarme su amistad, compañerismo y apoyo incondicional, convirtiéndose en parte fundamental de este camino.

A mi tutora, por ser más que una guía académica. Su profesionalismo, dedicación y valiosos consejos fueron clave en mi proceso de formación, brindándome el impulso necesario para alcanzar mis metas.

Al Instituto Superior Tecnológico Particular Bolívar Madero Vargas, por abrirme las puertas al conocimiento y brindarme las herramientas necesarias para desarrollarme como profesional. Gracias por ser el pilar de mi crecimiento académico y personal.

***Cristhian Andrés Rojas Cabrera***

**ÍNDICE GENERAL**

DECLARACIÓN DE RESPONSABILIDAD

CERTIFICACIÓN

DEDICATORIA

AGRADECIMIENTO

ÍNDICE GENERAL

INTRODUCCIÓN

[**1.** **MARCO TEÓRICO** - 3 -](#_Toc195901470)

[**1.1.** **Antecedentes** - 3 -](#_Toc195901471)

[**1.2.** **Justificación** - 6 -](#_Toc195901472)

[**1.3.** **Objetivos** - 7 -](#_Toc195901473)

[**1.3.1.** **Objetivo General** - 7 -](#_Toc195901474)

[**1.3.2.** **Objetivos Específicos** - 7 -](#_Toc195901475)

[**1.4.** **Descripción de la Empresa** - 8 -](#_Toc195901476)

[**1.4.1.** **Ubicación** - 9 -](#_Toc195901477)

[**1.4.2.** **Misión** - 9 -](#_Toc195901478)

[**1.4.3.** **Visión** - 9 -](#_Toc195901479)

[**1.4.4.** **Orgánico Funcional** - 10 -](#_Toc195901480)

[**1.4.5.** **Especificación de Procesos** - 10 -](#_Toc195901481)

[**1.5.** **Fuente de la Información** - 11 -](#_Toc195901482)

[**1.5.1.** **Entrevista** - 11 -](#_Toc195901483)

[**1.5.2.** **Resultado de la Entrevista** - 14 -](#_Toc195901484)

[**1.4.3.** **Encuesta** - 15 -](#_Toc195901485)

[**1.4.4.** **Análisis de la Encuesta** - 21 -](#_Toc195901486)

[**1.4.5.** **Problemas Encontrados (Causas y Efectos)** - 22 -](#_Toc195901487)

[**1.4.6.** **Alternativas de la Solución con el Sistema Informático a desarrollar.** - 23 -](#_Toc195901488)

[2. **ANÁLISIS DEL SISTEMA** - 26 -](#_Toc195901489)

[**2.1.** **Especificaciones de Requisitos** - 26 -](#_Toc195901490)

[**2.1.1.** **Requisitos Funcionales** - 27 -](#_Toc195901491)

[**2.1.3 Requerimientos de Interfaces Externas** - 30 -](#_Toc195901492)

[**2.1.4 Definiciones, Siglas y Abreviaturas** - 31 -](#_Toc195901493)

[**2.2 Usuarios** - 33 -](#_Toc195901494)

[**2.2.1 Características del Usuario** - 33 -](#_Toc195901495)

[**2.2.2 Roles del Usuario** - 33 -](#_Toc195901496)

[**2.3 Estudio de Factibilidad** - 33 -](#_Toc195901497)

[**2.3.1 Factibilidad Técnica** - 34 -](#_Toc195901498)

[**2.3.2 Factibilidad Económica** - 35 -](#_Toc195901499)

[**2.3.3 Factibilidad Operativa** - 36 -](#_Toc195901500)

[**2.3.4 Factibilidad Legal** - 36 -](#_Toc195901501)

[**3.** **FASES DE DISEÑO** - 39 -](#_Toc195901502)

[**3.1.** **Diseño** - 39 -](#_Toc195901503)

[**3.1.1.** **Diseño arquitectónico UML** - 39 -](#_Toc195901504)

[**3.1.1.1.** **Diagramas de Casos de Uso** - 40 -](#_Toc195901505)

[**3.1.2.** **Diseño de Base de Datos** - 51 -](#_Toc195901506)

[**3.1.2.1.** **Modelo lógico de la Base de Datos** - 51 -](#_Toc195901507)

[**3.1.2.2.** **Modelo físico de la Base de Datos** - 52 -](#_Toc195901508)

[**3.1.2.3.** **Diccionario de datos** - 52 -](#_Toc195901509)

[**3.1.3.** **Interfaz de Usuario** - 55 -](#_Toc195901510)

[**4.** **PROGRAMACION** 58](#_Toc195901511)

[**4.1. Ventajas y desventajas del lenguaje de programación.** 58](#_Toc195901512)

[**4.1.1. Herramientas de Programación** 61](#_Toc195901513)

[**4.1.2. Herramientas de Diseño** 62](#_Toc195901514)

[**4.1.3. Herramienta de Administración** 64](#_Toc195901515)

[**4.2. CÓDIGO FUENTE CAPA ENTIDADES** 65](#_Toc195901516)

[**4.3. CODIGO FUENTE LOGICA DE NEGOCIOS** 79](#_Toc195901517)

[**5. CONCLUSIONES Y RECOMENDACIONES** 83](#_Toc195901518)

[**5.1. Conclusiones** 83](#_Toc195901519)

[**5.2. Recomendaciones** 84](#_Toc195901520)

[**5.3. Anexos** 85](#_Toc195901521)

[**5.3.1. Manual de Usuario** 85](#_Toc195901522)

**INTRODUCCIÓN GENERAL**

En la actualidad, la tecnología se ha convertido en un factor clave para el desarrollo y la competitividad de las empresas, especialmente en el ámbito comercial. Uno de los procesos más importantes en la gestión de cualquier negocio es la facturación, ya que no solo permite llevar un control adecuado de las ventas e inventarios, sino que también asegura el cumplimiento de las normativas fiscales. Sin embargo, muchas pequeñas empresas aún dependen de sistemas manuales o tradicionales para realizar estos procesos, lo que puede generar errores, demoras y dificultades para gestionar adecuadamente la información financiera.

Este trabajo de tesis se enfoca en el análisis, diseño e implementación de un sistema de facturación electrónica para *Lucky Computers*, un local ubicado en la ciudad de Machala, especializado en la venta de repuestos para computadoras y prestación de servicios técnicos. Actualmente, el local enfrenta dificultades para gestionar de manera eficiente sus ventas, el control de inventarios y la emisión de facturas, lo que impacta en la productividad y la organización de sus operaciones.

El objetivo principal de este proyecto es desarrollar una solución automatizada que optimice el proceso de facturación electrónica, mejorando la eficiencia operativa, reduciendo los errores humanos y proporcionando información clara y precisa para la toma de decisiones. A través de este sistema, se pretende no solo agilizar la emisión de facturas, sino también facilitar la gestión del inventario, la administración de servicios, el control financiero y la generación de reportes estadísticos, contribuyendo así al crecimiento y sostenibilidad del negocio.

CAPÍTULO I

MARCO TEÓRICO

1. **MARCO TEÓRICO**
   1. **Antecedentes**

La facturación electrónica en Ecuador ha experimentado una implementación gradual que comenzó hace más de una década, con el fin de mejorar la eficiencia administrativa, reducir la evasión fiscal y facilitar el cumplimiento de las normativas tributarias. A continuación, se presentan los principales antecedentes de este sistema en el contexto ecuatoriano, con especial énfasis en la aplicabilidad para negocios como el de un local de computadoras y servicios relacionados.

Antes de la implementación de la facturación electrónica, las empresas en Ecuador utilizaban el sistema tradicional de facturación manual o mediante software no homologado por el Servicio de Rentas Internas (SRI). Este enfoque generaba desventajas tanto para las empresas como para el Estado, ya que las facturas físicas podían extraviarse, era difícil hacer un control efectivo sobre las transacciones y la evasión fiscal se volvía más común. En este contexto, el SRI comenzó a explorar alternativas para modernizar la administración tributaria y mejorar el control fiscal, tomando como referencia experiencias internacionales exitosas en la implementación de sistemas de facturación electrónica.

Uno de los primeros pasos clave para la implementación de la facturación electrónica en Ecuador fue la reforma de la Ley de Régimen Tributario Interno en 2011. Esta reforma introdujo disposiciones que permitían el uso de la facturación electrónica como una alternativa legal para las empresas, especialmente aquellas con volúmenes de transacciones más elevados, como es el caso de un local de computadoras que gestiona tanto ventas de equipos como servicios de mantenimiento.

A partir de esta reforma, el SRI empezó a desarrollar el marco legal necesario para facilitar la transición hacia la facturación electrónica, estableciendo una serie de normativas y requisitos técnicos que permitieran a las empresas, incluidas aquellas del sector tecnológico y servicios, adaptarse al nuevo sistema.

El proceso formal de implementación de la facturación electrónica comenzó con pruebas piloto en 2014. Durante este periodo, el SRI seleccionó un grupo de empresas grandes para evaluar la viabilidad del sistema. Este proceso permitió hacer ajustes técnicos y operativos antes de la implementación generalizada del sistema. En 2015, el SRI hizo oficial la obligatoriedad de la facturación electrónica para todas las empresas de ciertos sectores, especialmente aquellas de mayor tamaño o que ya formaban parte del sistema tributario. A partir de ese momento, se estableció un cronograma progresivo que abarcaba a más empresas de diferentes sectores, incluidas las que operan en la venta de productos tecnológicos y servicios relacionados.

A medida que el sistema de facturación electrónica se consolidaba en las grandes empresas, el SRI comenzó a extender la obligatoriedad a las pequeñas y medianas empresas (PYMES). En 2016 y 2017, las autoridades fiscales implementaron un proceso gradual para que las PYMES, incluidas aquellas dedicadas a la venta de computadoras y servicios de reparación, pudieran adaptarse a la nueva normativa. Este proceso fue importante para asegurar que las pequeñas empresas tuvieran tiempo suficiente para ajustarse al sistema de facturación electrónica y para mitigar posibles impactos negativos.

Para 2018, el SRI estableció que todos los contribuyentes, sin importar su tamaño, debían adoptar la facturación electrónica como parte de la estrategia para reducir la evasión fiscal y mejorar el control tributario en el país. Esto implicó que, tanto grandes empresas como pequeños comercios, incluidos los locales de computadoras, tuvieron que integrar el sistema de facturación electrónica en sus operaciones diarias.

A partir de ese momento, el uso de la facturación electrónica se convirtió en una obligación para todos los contribuyentes, y el SRI siguió implementando mejoras en el sistema, como la integración con otros servicios electrónicos y la optimización de la plataforma para facilitar su uso. Esta evolución ha sido clave para negocios que ofrecen productos tecnológicos y servicios como el mantenimiento de computadoras, ya que la automatización y la facilidad de uso del sistema permiten un manejo más eficiente de las facturas y los registros tributarios.

El SRI ha seguido perfeccionando su plataforma de facturación electrónica, agregando nuevas funcionalidades que permiten una validación automática de las facturas en tiempo real. Este avance no solo mejora la transparencia y reduce el fraude fiscal, sino que también asegura que las facturas emitidas estén alineadas con la normativa tributaria vigente. El sistema está diseñado para ser accesible y fácil de usar, incluso para las pequeñas empresas del sector tecnológico. Además, las empresas pueden optar por un software gratuito proporcionado por el SRI o adquirir software homologado que se ajuste mejor a sus necesidades, como es el caso de los locales de computadoras, que podrían necesitar herramientas adicionales para la gestión de inventarios y servicios.

A su vez, el SRI ha creado una plataforma en línea que permite a las empresas realizar consultas, gestionar sus documentos electrónicos y presentar sus declaraciones fiscales de manera automatizada. Este tipo de plataformas es esencial para negocios como el de un local de computadoras, donde las transacciones incluyen tanto productos como servicios, y donde la precisión en la facturación es crucial.

* 1. **Justificación**

La creciente demanda de soluciones tecnológicas en el ámbito empresarial ha resaltado la necesidad de mejorar los procesos administrativos y comerciales, especialmente en pequeñas y medianas empresas (PYMEs) dedicadas a la venta de productos tecnológicos y la prestación de servicios de mantenimiento. Los locales de computadoras, como el caso de **Lucky Computers**, enfrentan retos específicos relacionados con la gestión de ventas de equipos informáticos, accesorios, mantenimiento y la emisión de facturas.

En este contexto, la implementación de un sistema de facturación electrónica eficiente no solo contribuye a la mejora de la operatividad interna, sino que también optimiza la atención al cliente, mejora la transparencia fiscal y permite una mejor toma de decisiones a partir de la información generada. Sin un sistema adecuado, el local podría enfrentar dificultades como errores humanos en los cálculos de ventas, confusión en la gestión del inventario de productos y servicios, retrasos en los procesos de facturación, incumplimiento de normativas fiscales locales, y una disminución en la competitividad del negocio. Además, la falta de un control adecuado en los servicios de mantenimiento puede generar un seguimiento deficiente de las reparaciones y la pérdida de confianza de los clientes.

La facturación electrónica no solo agiliza el proceso de emisión de facturas, sino que también permite mantener un registro digital eficiente de las transacciones, asegurando que el local cumpla con las normativas fiscales del Servicio de Rentas Internas (SRI). Este tipo de sistema es fundamental para mantener la transparencia en las operaciones, reducir la evasión fiscal y facilitar la integración con otros servicios electrónicos que optimicen la gestión administrativa.

Un sistema de facturación adecuado también ayuda en la gestión del inventario de computadoras y accesorios, lo que es crucial para un negocio como **Lucky Computers**, que maneja una amplia gama de productos. Además, la capacidad de generar reportes y estadísticas facilita la toma de decisiones estratégicas, permitiendo a los propietarios y administradores del local tomar decisiones informadas sobre la compra de nuevos productos, la planificación de servicios de mantenimiento, y la optimización de la experiencia del cliente.

En resumen, la implementación de un sistema de facturación electrónica contribuirá a mejorar la eficiencia, la competitividad y la rentabilidad de **Lucky Computers**, optimizando tanto las operaciones comerciales como los procesos administrativos. Esto permitirá al local ofrecer un mejor servicio al cliente, cumplir con las obligaciones fiscales y mejorar su desempeño en el mercado de computadoras y servicios de mantenimiento.

* 1. **Objetivos**

* + 1. **Objetivo General**

Diseñar e implementar un sistema de facturación automatizado para el local de computadoras **Lucky Computers**, ubicado en la ciudad de Machala, con el fin de optimizar los procesos administrativos, mejorar el control de ventas de productos tecnológicos y servicios de mantenimiento, garantizar el cumplimiento de la normativa fiscal vigente y fortalecer la competitividad del negocio en el mercado local.

* + 1. **Objetivos Específicos**
* Analizar los procesos actuales de facturación y gestión administrativa de “Lucky Computers” para identificar áreas de mejora en ventas y servicios.
* Diseñar un sistema de facturación automatizado adaptado a las necesidades “Lucky Computers” optimizando ventas y servicios.
* Capacitar al personal de Lucky Computers en el uso del sistema para asegurar su correcto funcionamiento.
* Evaluar el impacto del sistema en la eficiencia operativa y el cumplimiento fiscal de Lucky Computers.
  1. **Descripción de la Empresa**

Lucky Computers es un negocio dedicado a la venta de productos tecnológicos y la prestación de servicios de mantenimiento de computadoras. Su catálogo incluye equipos de cómputo, accesorios, periféricos y software, además de servicios como reparación, optimización y asesoría técnica. Lucky Computers se distingue por ofrecer soluciones tecnológicas de calidad, combinando atención personalizada con productos y servicios que satisfacen las necesidades de sus clientes.

* **Nombre:** “LUCKY COMPUTERS”
* **Propietario:** William Andrés Rojas Cabrera
* **Cantón:** Machala
* **Celular:** 0998297867

**Razón Social:** Venta de computadoras, accesorios y servicios de mantenimiento al por menor.

* + 1. **Ubicación**

****

* + 1. **Misión**

Desarrollar e implementar un sistema de facturación integral, eficiente y automatizado para Lucky Computers, ubicado en la ciudad de Machala, que optimice los procesos de ventas, control de inventarios y generación de documentos fiscales. Este sistema busca mejorar la experiencia del cliente, incrementar la productividad del personal y garantizar una gestión clara y transparente de los recursos, contribuyendo al crecimiento y competitividad del negocio en el sector tecnológico y de mantenimiento informátco.

* + 1. **Visión**

Ser un referente en la implementación de soluciones tecnológicas en el sector comercial de productos tecnológicos y servicios de mantenimiento, destacándonos por ofrecer un sistema de facturación innovador, fácil de usar y adaptable a las necesidades cambiantes del mercado. A través de la digitalización y automatización de procesos, buscamos posicionar a **Lucky Computers** como un negocio líder en eficiencia y servicio al cliente en la ciudad de Machala, asegurando una gestión empresarial más sostenible y rentable en el futuro.

* + 1. **Orgánico Funcional**
    2. **Especificación de Procesos**

El sistema de facturación debe incluir un módulo que permita registrar productos, servicios y clientes de manera automatizada. También debe gestionar los inventarios, actualizándolos en tiempo real al realizar una venta y generando alertas para la reposición de productos cuando sea necesario. Además, el sistema debe emitir facturas electrónicas con la información exigida por el **Servicio de Rentas Internas (SRI)**, registrar todas las transacciones y enviar automáticamente las facturas a los clientes.

Por otro lado, debe incluir reportes para analizar ventas, servicios prestados y necesidades de reposición, consolidando toda la información para facilitar la toma de decisiones. El sistema debe ser intuitivo, accesible desde diversos dispositivos y contar con medidas de seguridad como copias de seguridad automáticas y control de accesos por roles.

* 1. **Fuente de la Información**

Para el desarrollo de esta investigación, es fundamental revisar documentos relacionados con el tema, incluyendo libros, artículos científicos, manuales, normativas del Servicio de Rentas Internas (SRI) y sitios web especializados. Estos recursos proporcionan información clave para el análisis, diseño e implementación del Sistema de Facturación Electrónica para Lucky Computers.

Además, se empleará la observación directa como método de recolección de datos, permitiendo obtener información sobre los procesos actuales de facturación, gestión de inventarios y atención al cliente en el negocio. Este método consiste en la recopilación de datos sin intervenir en el ambiente, asegurando la validez de la información obtenida.

La investigación también incluirá fuentes primarias, como entrevistas con el personal de Lucky Computers, para identificar necesidades y validar la funcionalidad del sistema. Estas técnicas complementarán el estudio, garantizando un diseño que responda a los requerimientos específicos del negocio.

* + 1. **Entrevista**

Para el desarrollo de la tesis sobre el **Sistema de Facturación Electrónica para Lucky Computers**, se realizaron entrevistas con el propietario del negocio y su equipo de trabajo. El objetivo fue comprender los procesos actuales de facturación, identificar problemas en la gestión de ventas e inventarios y conocer las expectativas de mejora.

El propietario brindó información clave sobre las necesidades del negocio, mientras que los empleados compartieron sus experiencias con la gestión de productos, servicios y facturación. También se consultó a los clientes para evaluar su nivel de satisfacción con el proceso de compra y emisión de facturas.

Además, se entrevistaron proveedores y expertos en software para evaluar tecnologías adecuadas que cumplan con las regulaciones del Servicio de Rentas Internas (SRI) y mejoren la eficiencia del negocio.

Esta información resultó fundamental para el análisis y diseño del sistema de facturación, asegurando que responda a los requerimientos específicos de Lucky Computers.

**Desarrollo de la Entrevista:**

**Nombre:** Sr. William Rojas.

**Asunto:** Análisis de los procesos de facturación y la implementación de un sistema automatizado para mejorar la eficiencia y gestión de ventas e inventarios en el local Lucky Computers

**Fecha:** 16 de Diciembre del 2024.

1. **¿Podría describir brevemente cómo funcionan actualmente los procesos de facturación en su tienda?**

Actualmente, utilizamos un proceso manual para la facturación de ventas de computadoras y servicios de mantenimiento. Las transacciones se registran manualmente en papel, lo que puede generar errores y retrasa el proceso. Además, no tenemos un sistema automatizado para actualizar los inventarios o generar facturas electrónicas, lo que complica el cumplimiento con las normativas fiscales del SRI y la gestión eficiente del stock de productos y repuestos.

1. **¿Cuáles son los principales problemas o desafíos que enfrenta su tienda con el sistema de facturación actual?**

Hay errores frecuentes en el registro de ventas y servicios, lo que afecta el control de inventarios y dificulta el cumplimiento de las normativas fiscales.

1. **¿Cómo gestionan el inventario actualmente y cómo está vinculado con la facturación?**

Tenemos dificultades para mantener un control preciso del inventario, ya que no está automatizado y no se actualiza en tiempo real con cada venta.

1. **¿Qué tan importante considera que es la automatización de la facturación en su negocio?**

Es muy importante, ya que mejoraría la eficiencia, reduciría errores y agilizaría el proceso de ventas.

1. **En su opinión, ¿qué características debería tener un sistema de facturación ideal para su tienda?**

Pues debería ser un sistema completo que maneje facturación e inventario de la tienda para un mejor control de la mercadería.

1. **¿Qué expectativas tiene respecto a la implementación de un nuevo sistema de facturación?**

Espero que mejore la eficiencia, reduzca errores y facilite el control de inventario, lo que hará el proceso de compra más rápido y mejorará la experiencia del cliente.

1. **¿Está dispuesto a invertir en un sistema de facturación que sea más avanzado, pero que implique costos adicionales, o prefiere una opción más económica?**

Estoy dispuesto a invertir en un sistema más avanzado si los beneficios a largo plazo justifican el costo. Un sistema eficiente puede ahorrarnos tiempo y dinero.

1. **¿Cómo percibe el impacto de un sistema de facturación eficiente en la relación con los clientes y la competitividad de su tienda en el mercado local?**

Un sistema eficiente haría las compras más rápidas y precisas, mejorando la satisfacción del cliente y aumentando nuestra competitividad frente a otras tiendas.

1. **¿Qué tan importante es para usted la capacitación del personal en el uso de este nuevo sistema?**

Es crucial. Aunque el sistema debe ser fácil de usar, una buena capacitación es esencial para asegurar que los empleados aprovechen todas sus funciones y mantengan la eficiencia.

* + 1. **Resultado de la Entrevista**

La entrevista con el propietario de la tienda reveló varios desafíos clave en el proceso de facturación actual, que se realiza de forma manual, lo que genera errores y ralentiza el flujo de trabajo. El propietario destacó la urgencia de automatizar el sistema de facturación y vincularlo al control de inventarios para asegurar una gestión más precisa y eficiente.

El propietario también mencionó que el sistema ideal debería ser fácil de usar, agilizar el proceso de ventas y generar reportes automáticos sobre ventas e inventarios. Además, se destacó la importancia de la capacitación del personal para asegurar el buen uso del sistema y maximizar su eficacia. Finalmente, el propietario expresó su disposición a invertir en un sistema más avanzado si los beneficios a largo plazo, como la reducción de errores y el ahorro de tiempo, justifican la inversión.

* + 1. **Encuesta**

El objetivo de la siguiente encuesta es determinar que piensan diferentes actores sobre el sistema de cobranzas, y si estarían de acuerdo a la automatización del mismo

**Pregunta 1. ¿ Cuál es su rol en la tienda "Lucky Computers"?**

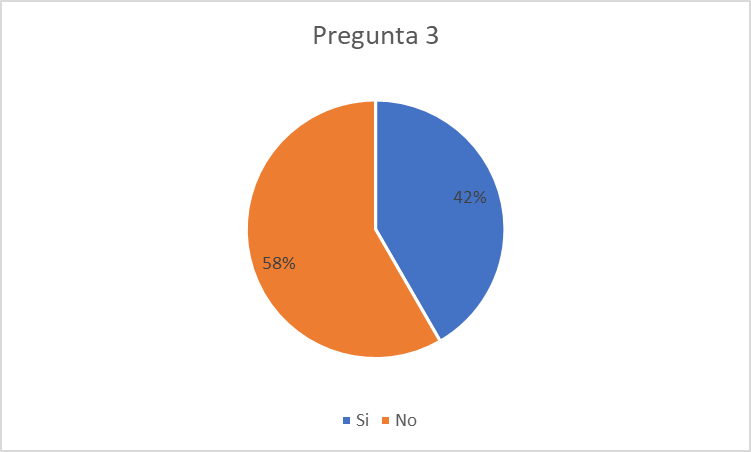
|  |  |  |
| --- | --- | --- |
| Numero de muestra: 12 | | |
| RESPUESTA | **FRECUENCIA** | **PORCENTAJE** |
| PROPIETARIO | 1 | 8.33% |
| EMPLEADO ADMINISTRATIVO | 2 | 16.67% |
| EMPLEADO DE VENTAS | 2 | 41.67% |
| TOTAL | **12** | **100%** |

**Pregunta 2. ¿Actualmente existe un control automatizado de ventas e inventarios en la tienda "Lucky Computers"?**

|  |  |  |
| --- | --- | --- |
| Numero de muestra: 12 | | |
| RESPUESTA | **FRECUENCIA** | **PORCENTAJE** |
| SI | 2 | 16.67% |
| NO | 10 | 83.33% |
| TOTAL | **12** | **93,3%** |

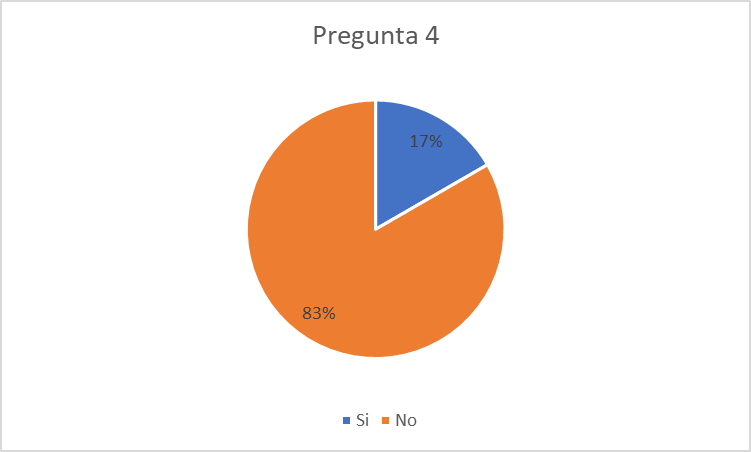
**Pregunta 3. ¿Considera que el sistema de facturación actual de la tienda "Lucky Computers" es ágil y eficiente?**

|  |  |  |
| --- | --- | --- |
| Numero de muestra: 12 | | |
| RESPUESTA | **FRECUENCIA** | **PORCENTAJE** |
| SI | 5 | 41.67% |
| NO | 7 | 58.33% |
| TOTAL | **12** | **100%** |

****

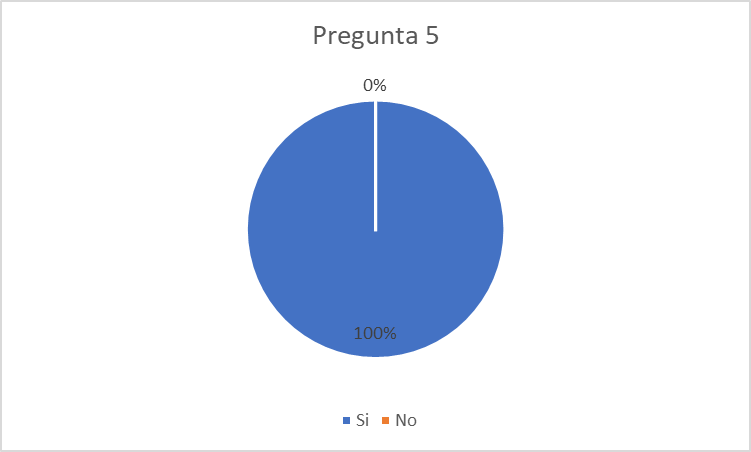
**Pregunta 4. ¿En el sistema de facturación actual de la tienda "Lucky Computers" se pueden consultar y verificar fácilmente los registros de ventas e inventarios?**

|  |  |  |
| --- | --- | --- |
| Numero de muestra: 12 | | |
| RESPUESTA | **FRECUENCIA** | **PORCENTAJE** |
| SI | 2 | 16.67% |
| NO | 10 | 83.33% |
| TOTAL | **12** | **100%** |

****

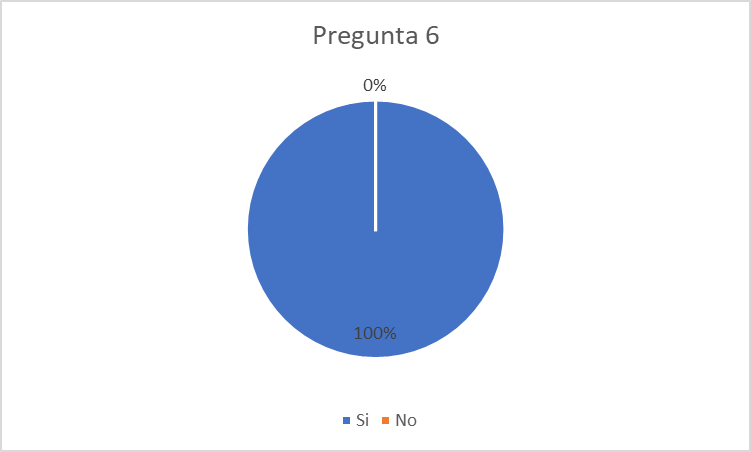
**Pregunta 5. ¿Cree usted que es importante utilizar nuevas herramientas para automatizar el proceso de facturación y control de inventarios en la tienda "Lucky Computers"?**

|  |  |  |
| --- | --- | --- |
| Numero de muestra: 12 | | |
| RESPUESTA | **FRECUENCIA** | **PORCENTAJE** |
| SI | 12 | 100% |
| NO | 0 | 0% |
| TOTAL | **12** | **100%** |

****

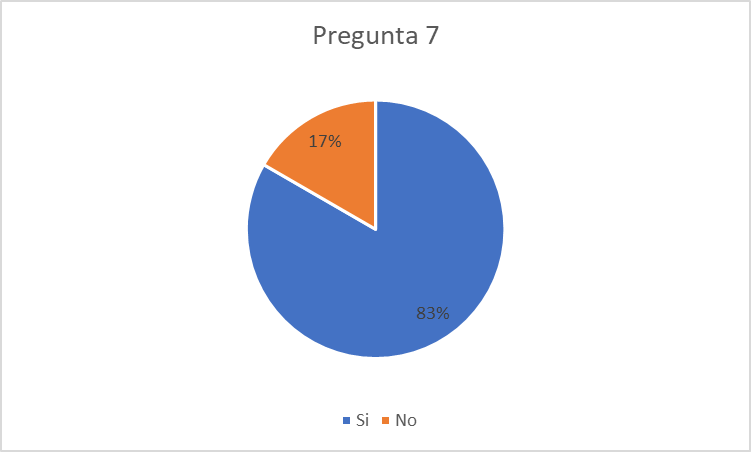
**Pregunta 6. ¿Considera usted que la implementación de un sistema de facturación automatizado que controle los procesos de ventas e inventarios ayudará a que la gestión de la información sea más eficiente en la tienda "Lucky Computers"?**

|  |  |  |
| --- | --- | --- |
| Numero de muestra: 12 | | |
| RESPUESTA | **FRECUENCIA** | **PORCENTAJE** |
| SI | 12 | 100% |
| NO | 0 | 0% |
| TOTAL | **12** | **100%** |

****

**Pregunta 7. ¿Cree usted que implementar un sistema automatizado de facturación y control de inventarios agilizará los procesos operativos y administrativos en la tienda "Lucky Computers"?**

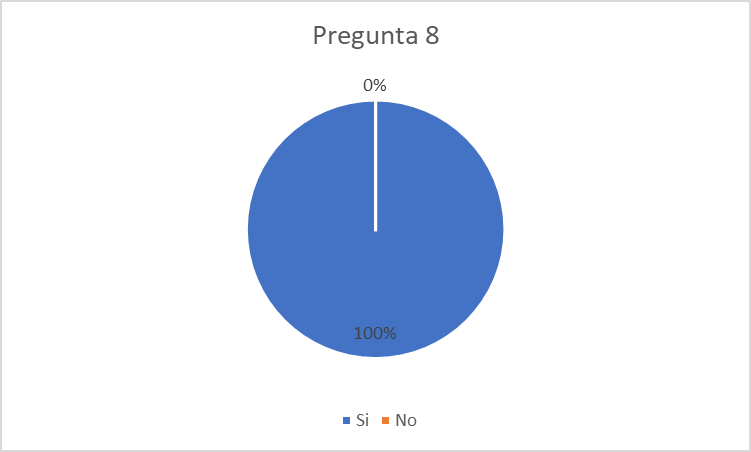
|  |  |  |
| --- | --- | --- |
| Numero de muestra: 12 | | |
| RESPUESTA | **FRECUENCIA** | **PORCENTAJE** |
| SI | 10 | 83.33% |
| NO | 2 | 16.67% |
| TOTAL | **12** | **100%** |

****

**Pregunta 8. ¿Considera Ud. que el sistema automatizado ayudara a mejorar la toma de decisiones de docentes y autoridades?**

**Numero de muestra:12**

|  |  |  |
| --- | --- | --- |
| RESPUESTA | FRECUENCIA | PORCENTAJE |
| SI | 12 | 100% |
| NO | 0 | 0% |
| TOTAL | **12** | **100%** |

****

* + 1. **Análisis de la Encuesta**

La mayoría de los encuestados considera necesaria la implementación de un sistema de facturación automatizado, ya que este permitiría registrar las facturas y controlar los productos de manera más eficiente, garantizando así la seguridad y disponibilidad de la información.

* + 1. **Problemas Encontrados (Causas y Efectos)**

|  |  |  |
| --- | --- | --- |
| **PROBLEMA** | **CAUSA** | **EFECTO** |
| El proceso de facturación y control de inventario se realiza de forma manual, lo que lo vuelve complejo y propenso a errores. | La falta de un sistema automatizado facilita el ingreso incorrecto de datos y dificulta la verificación y análisis de la información. | Esto genera errores, pérdida de información y afecta la atención al cliente. |
| No se puede medir con precisión los productos que salen de la tienda. | El control manual dificulta el acceso a datos y estadísticas confiables | Genera inseguridad, posible pérdida de información y limita la toma de decisiones. |

* + 1. **Alternativas de la Solución con el Sistema Informático a desarrollar.**

|  |  |  |
| --- | --- | --- |
| **ALTERNATIVA I** | | |
| **DETALLE** | **VENTAJAS** | **DESVENTAJAS** |
| Implementar un sistema automatizado que mantenga actualizada la información de ventas e inventario, y que genere facturas electrónicas de forma inmediata. | Elimina procesos manuales repetitivos.  Mejora la eficiencia operativa. | Inversión inicial en software y adaptación del equipo.  Se requiere tiempo para capacitación del personal. |

**Tabla 2: Alternativa de Solución**

|  |  |  |
| --- | --- | --- |
| **ALTERNATIVA II** | | |
| **DETALLE** | **VENTAJAS** | **DESVENTAJAS** |
| Implementar un sistema que garantice copias de seguridad automáticas para proteger la información ante desastres o fallos del equipo. | La información estará segura y respaldada.  Se evita la pérdida de datos importantes.  Facilita la recuperación en caso de fallos. | Puede requerir configuración adicional para los formatos de respaldo.  Se necesita un método eficiente para manejar distintos tipos de archivo. |

Con la implementación de este sistema de facturación, “Lucky Computers” podrá registrar y acceder rápidamente a la información ingresada al sistema, lo que facilitará la entrega inmediata de las facturas a los clientes. Además, el sistema ayudará a mejorar la eficiencia operativa y garantizará la seguridad de los datos. Esto no solo optimizará los procesos internos, sino que también contribuirá al crecimiento de la empresa, posicionándola como un referente en el ámbito de la tecnología y los servicios de mantenimiento de computadoras en la ciudad de Machala.

**CAPÍTULO II**

**ANÁLISIS DEL SISTEMA**

1. **ANÁLISIS DEL SISTEMA**

El análisis del sistema es un componente esencial en el desarrollo del sistema de facturación para Lucky Computers. Permite recopilar datos clave sobre los procesos actuales de ventas, inventario y atención al cliente, asegurando que el software se construya de acuerdo con las necesidades reales del negocio. Este proceso es fundamental para diseñar una solución eficiente y sostenible a largo plazo, optimizando la gestión y mejorando la toma de decisiones en la organización.

* 1. **Especificaciones de Requisitos**

El análisis del sistema inicia con la obtención de requisitos en función de las necesidades de la tienda y los usuarios finales. En este caso, se propone el desarrollo de un Sistema de Facturación Electrónica para la tienda de ropa “**Lucky Computers”**. Este sistema debe cumplir con las siguientes especificaciones:

* Gestión de productos y servicios
* Gestión de clientes y proveedores
* Emisión de facturas electrónicas conforme a los estándares del SRI
* Registro y control de ventas
  + 1. **Requisitos Funcionales**

**Autenticación de Usuarios:** El sistema permitirá el ingreso a través de credenciales según los roles definidos:

* Administrador
* Usuarios

**Gestión de productos y servicios:**

• Crear, editar, eliminar y listar repuestos de computadoras y servicios de mantenimiento ofrecidos por **Lucky Computers**.

**Gestión de facturación:**

• Emisión, anulación y consulta de facturas electrónicas por ventas de repuestos y servicios de mantenimiento.

• Integración con el sistema del **SRI** para validar los documentos emitidos de forma automática.

**Reportes:**

• Reporte de ventas mensuales.

**2.1.2. Requisitos No Funcionales**

**• Rendimiento:** El sistema deberá procesar la información de ventas, inventario y servicios de manera rápida para no afectar la atención al cliente ni las operaciones diarias del local.

**• Seguridad:** Se implementarán protocolos de seguridad para proteger los datos de los clientes, las facturas electrónicas y el acceso al sistema, garantizando la confidencialidad y la integridad de la información.

**• Usabilidad:** El sistema contará con interfaces amigables e intuitivas, diseñadas para facilitar su uso por parte del personal administrativo y técnico, sin necesidad de conocimientos avanzados en informática.

**• Disponibilidad:** El sistema estará disponible durante el horario comercial del local, con soporte técnico en caso de fallos o inconvenientes que puedan interrumpir la facturación o gestión del inventario.

**Usuarios y Roles**

**Características del Usuario**

El sistema estará diseñado para:

**• Administrador:** Responsable de la configuración general y gestión avanzada del sistema.

**• Usuario:** Personal encargado de realizar facturaciones y consultas de reportes.

**Roles específicos:**

**Administrador:**

• Gestión completa de usuarios, productos y ventas.

• Supervisión y validación de las operaciones diarias.

**Usuario**:

• Gestión básica de facturas y consultas.

• Generación de reportes simples para control diario.

**Estudio de Factibilidad**

**Factibilidad Técnica**

El sistema será desarrollado utilizando tecnologías modernas y accesibles, asegurando un funcionamiento eficiente y adaptable a las necesidades de *Lucky Computers*:

• Lenguaje de programación: Python.

• Base de datos: MongoDB.

• Framework: Flask para la creación de una aplicación web escalable.

• Editor de código: Visual Studio Code.

**Factibilidad Económica**

La implementación del sistema requiere una inversión inicial accesible, considerando:

• Hardware: Se utilizarán las computadoras y periféricos ya existentes en “*Lucky Computers”*.

• Software: Se contemplan licencias básicas para herramientas de documentación y desarrollo, en su mayoría de código abierto, reduciendo costos.

**Factibilidad Operativa**

El personal de *Lucky Computers* será capacitado para el uso eficiente del sistema. Se elaborarán manuales de usuario y se contará con soporte técnico básico para garantizar su correcta utilización y mantenimiento, permitiendo una transición sin afectar las operaciones del negocio.

**MÓDULOS DEL SISTEMA**

MÓDULO DE SEGURIDAD

**Objetivos:**

* Gestionar el acceso al sistema mediante usuarios y roles.

MÓDULO DE PRODUCTOS

Objetivos:

• Administrar el catálogo de repuestos y componentes.

Beneficios:

• Permite registrar, actualizar y eliminar productos del inventario.

• Mantiene actualizado el stock en tiempo real, facilitando el control de mercadería.

**MÓDULO DE FACTURACIÓN ELECTRÓNICA**

**Objetivos:**

• Gestionar de forma eficiente la emisión, anulación y consulta de facturas electrónicas generadas por la venta de repuestos y servicios de mantenimiento.

**Beneficios:**

• Cumple con los requerimientos establecidos por el SRI para la validación y emisión de comprobantes electrónicos.

• Agiliza la atención al cliente al emitir facturas de forma rápida y precisa

**MÓDULO DE REPORTES**

**Objetivos:**

**• Generar reportes financieros, de inventario y servicios prestados relacionados con la venta de repuestos y mantenimiento de computadoras.**

**Beneficios:**

**• Permite analizar el comportamiento de las ventas, servicios y niveles de stock.**

**• Contribuye al control y planificación de inventario, detectando productos más vendidos o con baja rotación.**

### **2.1.3 Requerimientos de Interfaces Externas**

Los requerimientos de interfaces externas definen cómo los usuarios interactuarán con el sistema, los dispositivos necesarios para su funcionamiento y el entorno operativo donde será implementado.

**• Interfaces de Usuario:** El sistema contará con interfaces web intuitivas y responsivas, diseñadas con HTML, CSS y JavaScript, adaptadas a las necesidades de **Lucky Computers** para un uso eficiente.

**• Interfaces de Software:** Será compatible con **Windows 10 o superior**, desarrollado en **Python** con **Flask** y utilizando **MongoDB**, garantizando escalabilidad y seguridad.

**• Interfaces de Hardware**

El sistema soportará las siguientes actividades.

|  |  |
| --- | --- |
| Navegación entre Interfaces | Mouse y Teclado |
| *Ingreso de datos* | *Teclado, Mouse* |
| *Salida de datos* | *Monitor* |

### **2.1.4 Definiciones, Siglas y Abreviaturas**

#### **2.1.4.1 Del Negocio**

|  |  |
| --- | --- |
| **Definiciones** | **Significado** |
| **Propietario** | Responsable de supervisar la gestión financiera y operativa del local **Lucky Computers**. |
| **Asistente** | Registra ventas, emite facturas, genera reportes y mantiene la documentación del sistema. |
| **Cajero** | Procesa ventas, emite comprobantes y gestiona los métodos de pago. |
| **Cliente** | Persona que adquiere productos o servicios; sus compras se registran en el sistema. |

#### **2.1.4.2 Del Sistema**

|  |  |
| --- | --- |
| **Definiciones** | **Significado** |
| **CRUD** | Funciones básicas del sistema: crear, leer, modificar y eliminar información. |
| **Búsqueda** | Consulta de datos almacenados en la base para mostrarlos en el sistema. |
| **Usuario** | Persona con acceso al sistema mediante credenciales. Puede tener distintos roles. |
| **Permiso** | Acciones que un usuario puede realizar según su rol, como facturar o editar productos. |
| **Rol** | Conjunto de permisos asignados a un usuario (ej. administrador, cajero). |
| **Administrador** | Usuario con acceso total: configura el sistema, gestiona usuarios y genera reportes. |

#### **2.1.4.3 De la Tecnología**

|  |  |
| --- | --- |
| **Abreviaturas** | **Significado** |
| **Motor de Base de Datos** | |  | | --- | |  |  |  | | --- | | Software que permite crear y gestionar bases de datos, conectándolas al sistema para almacenar información. | |
| **Base de Datos** | Conjunto estructurado de datos donde se guarda la información del sistema. |
| **Software** | Aplicación web creada para optimizar la facturación e inventario en el local "Lucky Computers". |
| **MONGODB** | Sistema de base de datos NoSQL, orientado a documentos, ideal para manejar datos no estructurados. |
| **Visual Studio Code** | Editor de código fuente ligero y potente, usado para programar aplicaciones web y de escritorio. |

## 

## **2.2 Usuarios**

### **2.2.1 Características del Usuario**

El usuario interactúa con el sistema de facturación según su rol, realizando tareas específicas que le permiten cumplir con sus responsabilidades. Los usuarios tienen acceso a funciones basadas en su nivel de autorización, optimizando sus tareas dentro del sistema.

### **2.2.2 Roles del Usuario**

Cada usuario tiene un rol definido, lo que le permite acceder a funcionalidades específicas sin interferir con las tareas de otros. El **Administrador** tiene acceso total al sistema, incluyendo la capacidad de recuperar datos eliminados, y será el encargado de brindar soporte técnico en caso de requerirse.

## **2.3 Estudio de Factibilidad**

### 

El análisis de viabilidad evalúa la justificación y los objetivos del proyecto, determinando si su implementación es posible a través de diversas evaluaciones. Este proceso ayuda a concluir si el proyecto es viable o no. Para la ejecución de este sistema de facturación, se han considerado los análisis de viabilidad técnica, operativa, financiera y jurídica, para asegurar que la propuesta sea adecuada y conveniente para su desarrollo.

### **2.3.1 Factibilidad Técnica**

Evalúa la disponibilidad de equipos, software y recursos técnicos necesarios para desarrollar e implementar el sistema de facturación web. También considera si el hardware y software existentes son adecuados para soportar las funciones y características del sistema, garantizando un funcionamiento eficiente y estable.

**RECURSOS TÉCNICOS PARA EL DESARROLLO DEL PROYECTO**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipo** | **Nombre del Recurso** | **Descripción** | **Cant.** |
| **HARDWARE** | Computador | Laptop:  Procesador Intel i7  Disco duro de 1TB  Disco Solido 445GB  Memoria (RAM) 12GB  Tipo de sistema 64 Bits | 1 |
| **SOFTWARE** | Python Flask | Programa para la ejecución del sistema web | 1 |
| **SOFTWARE** | Mongodb | Manejador de Base de Datos No SQL | 1 |
| **SOFTWARE** | Visual Studio Code | Herramienta para el desarrollo de la aplicación | 1 |
| **SOFTWARE** | Windows 11 | Sistemas Operativo | 1 |
| **SOFTWARE** | Microsoft Office | Herramienta para la aplicación de ofimática | 1 |
| **SOFTWARE** | Microsoft Project | Gestión del Proyecto | 1 |
| **SOFTWARE** | Bootstrap | Herramienta para Diseño | 1 |

**Conclusión:** según los recursos técnicos establecidos para el desarrollo del proyecto, no tienen ninguna restricción en el país, por ende, el proyecto es factible técnicamente.

### 

### **2.3.2 Factibilidad Económica**

Este análisis considera los costos asociados al desarrollo del sistema, incluyendo el tiempo invertido por el desarrollador, tomando en cuenta su condición de estudiante y su proceso de autoaprendizaje. También se contempla el uso de hardware y software disponible en el local Lucky Computers, así como los precios referenciales obtenidos a través de proveedores o fuentes en línea, asegurando que el proyecto sea viable económicamente.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tipo** | **Descripción** | **Cant.** | **P/UNIT.** | **Costo** |
| **RR – HH** | Programador Web: (salario anual / 12 meses) | 1 | 650.00 | 350.00 |
| **HARDWARE** | Computadora de escritorio: (Intel Core i3, 250 GB de disco duro, 8 GB de RAM, teclado y mouse) | 1 | $309.99 | $309.99 |
|  |  |  |  |  |
| **SOFTWARE** | Python, Flask | 1 | 0,00 | 0,00 |
|  | MongoDB | 1 | 0,00 | 0,00 |
|  | Visual Studio Code | 1 | 0,00 | 0,00 |
|  | Bootstrap | 1 | 0,00 | 0,00 |
| **TOTAL, EGRESOS PARCIALES** | | | | 659.99 |
| **10% POR GASTOS EN IMPREVISTOS** | | | | 65,99 |
| **TOTAL, EGRESOS** | | | | 725.98 |
| **Conclusión:** según los recursos establecidos para el desarrollo del proyecto, no tienen ninguna restricción en el país y es de bajo costo, por ende, el proyecto es factible económicamente y financiado por la institución educativa. | | | | |

### 

### **2.3.3 Factibilidad Operativa**

El desarrollador del sistema cuenta con los conocimientos y habilidades necesarias para implementar soluciones tecnológicas efectivas, manteniéndose en constante aprendizaje sobre nuevas herramientas y lenguajes de programación. Esto permite superar la curva de aprendizaje y aplicar tecnologías adecuadas para el proyecto.

Además, el usuario final participará activamente durante el proceso de desarrollo, garantizando que el sistema cumpla con los requerimientos y se adapte a las necesidades del local Lucky Computers. Se implementarán buenas prácticas de accesibilidad y usabilidad, fundamentales para un sistema de facturación eficiente y fácil de utilizar.

### **2.3.4 Factibilidad Legal**

La viabilidad legal del proyecto garantiza que el desarrollo e implementación del sistema de facturación web para “Lucky Computers” cumple con la normativa vigente en Ecuador, en especial con lo dispuesto en el Código Orgánico Integral Penal (COIP) en lo relacionado con el tratamiento de datos y la confidencialidad de la información.

Todos los participantes en el proyecto —desarrollador, asesores y usuarios— han aceptado mantener la confidencialidad y hacer un uso ético de la información manejada en el sistema. Se promueve la protección de datos y la integridad de las personas involucradas, lo que asegura que el proyecto se ejecuta dentro del marco legal correspondiente.

**CAPÍTULO III**

**FASES DEL DISEÑO**

1. **FASES DE DISEÑO**
   1. **Diseño**
      1. **Diseño arquitectónico UML**

UML significa Lenguaje de Modelado Unificado (en inglés, Unified Modeling Language) y es un lenguaje estándar para visualizar, diseñar, especificar y documentar sistemas orientados a objetos.

Los tipos de UML son:

* **Diagramas estructurales**: Estos diagramas representan la estructura estática del sistema, es decir, las partes que lo componen y cómo se relacionan entre sí. [Los subtipos de diagramas estructurales incluyen](https://ideascale.com/es/blogs/diferentes-tipos-de-diagramas-uml/):

Diagrama de clases

Diagrama de objetos

Diagrama de implantación

Diagrama de estructura compuesta

Diagrama de paquetes

Diagrama de perfil

Diagrama de componentes

* [**Diagramas de comportamiento**: Estos diagramas representan el comportamiento dinámico del sistema, es decir, cómo interactúan las diferentes partes del sistema a lo largo del tiempo](https://diagramasuml.com/).
  + - 1. **Diagramas de Casos de Uso**

Un diagrama de casos de uso es una herramienta de modelado visual utilizada en UML (Lenguaje de Modelado Unificado) para representar las interacciones entre un sistema y sus actores (usuarios, dispositivos, otros sistemas, etc.).

El objetivo de un diagrama de casos de uso es capturar los requisitos funcionales de un sistema en términos de las interacciones que tienen lugar entre los actores y el sistema. Cada caso de uso describe una funcionalidad específica del sistema que puede ser llevada a cabo por uno o varios actores.

Los elementos principales de un diagrama de casos de uso son:

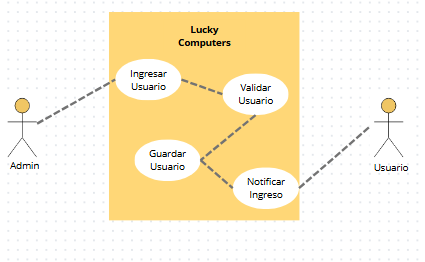
* Actor: Representa a un usuario, dispositivo, sistema o cualquier entidad que interactúe con el sistema.
* Caso de uso: Describe una funcionalidad específica del sistema que se lleva a cabo en respuesta a una solicitud de un actor.
* Relaciones de inclusión: Describen cómo un caso de uso puede incluir a otro caso de uso como parte de su flujo de ejecución.
* Relaciones de extensión: Describen cómo un caso de uso puede extender a otro caso de uso para proporcionar funcionalidades adicionales.

Los diagramas de casos de uso son útiles para visualizar las funcionalidades de un sistema y para comunicar los requisitos del sistema a los desarrolladores y otros stakeholders del proyecto.

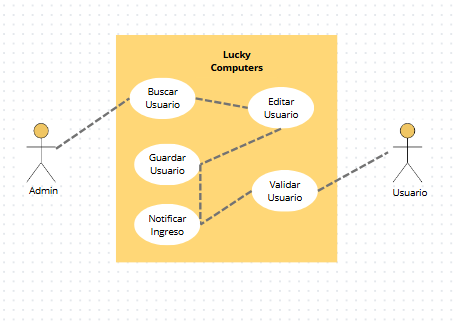
**Descripción de Actores**

|  |  |  |
| --- | --- | --- |
| **ACTOR** | **DESCRIPCIÓN** | **PERFIL** |
| Administrador | Usuario con control total del sistema. Puede crear usuarios (asistentes o cajeros), gestionar productos, emitir facturas y visualizar reportes completos del sistema. | Persona con conocimientos avanzados en el uso y configuración del sistema. |
| Usuario | Tiene permisos limitados respecto al administrador. Puede ingresar clientes, registrar productos, emitir facturas y consultar información básica. | Personal encargado de operaciones diarias, como ventas y atención al cliente. |

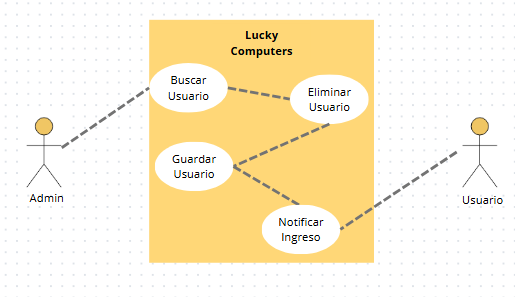
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO I – Gestionar Usuario** | | |
| **Descripción** | Permite al administrador crear usuarios en el sistema. | |
| **Actores** | Administrador, Usuarios | |
| **Precondiciones** | * El administrador debe haber iniciado sesión correctamente. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Crear Usuario | |
| 1 | Ingresar usuario. |
| 2 | |  | | --- | |  |  |  | | --- | | Validar usuario. | |
| 3 | Guardar usuario. |
| 4 | Notificar ingreso. |
| **Excepciones** |  |  |
|  | 1 | Cedula incorrecta |
|  | 2 | Campos en blanco. |



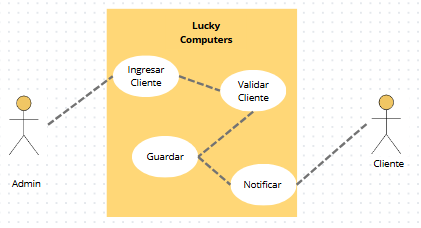
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO II – Gestionar Usuario** | | |
| **Descripción** | Permite al administrador modificar la información de los usuarios existentes en el sistema. | |
| **Actores** | Administrador, Usuarios | |
| **Precondiciones** | El administrador debe haber iniciado sesión correctamente.  El usuario a modificar debe estar previamente registrado en el sistema para la eliminación y edición. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Editar Usuario | |
| 1 | |  | | --- | | Buscar usuario. | |
| 2 | Editar usuario. |
| 3 | Guardar. |
| 4 | |  | | --- | |  |  |  | | --- | | Notificar. | |
| 5 | Validar usuario existente |
| **Excepciones** |  |  |
|  | 1 | Cedula incorrecta |
|  | 2 | Campos en blanco. |



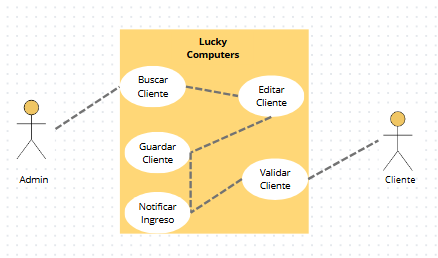
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO III – Gestionar Usuario** | | |
| **Descripción** | Permite al administrador eliminar usuarios del sistema. | |
| **Actores** | Administrador, Usuarios | |
| **Precondiciones** | El administrador previamente debe estar registrado en el sistema para el ingreso, el usuario debe estar previamente ingresado para la eliminación y edición. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Eliminar Usuario | |
| 1 | Buscar usuarios. |
| 2 | Eliminar usuario. |
| 3 | |  | | --- | |  |  |  | | --- | | Guardar. | |
| 4 | Notificar. |
| **Excepciones** |  |  |
|  | 1 | Información errónea. |
|  | 2 | Información duplicada. |



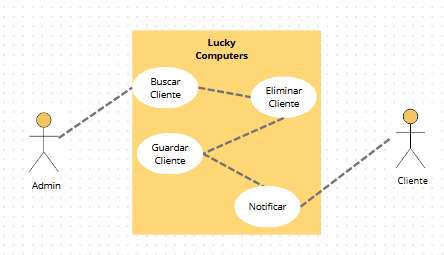
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO IV – Gestionar Cliente** | | |
| **Descripción** | Permite al administrador crear clientes del sistema. | |
| **Actores** | Administrador, cliente | |
| **Precondiciones** | El administrador previamente debe estar registrado en el sistema. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Ingresar Cliente | |
| 1 | Ingresar a cliente. |
| 2 | Validar cliente existente |
| 3 | Guardar cliente |
| 4 | Notificar ingreso |
| **Excepciones** |  |  |
|  | 1 | Cedula incorrecta |
|  | 2 | Campos en blanco. |



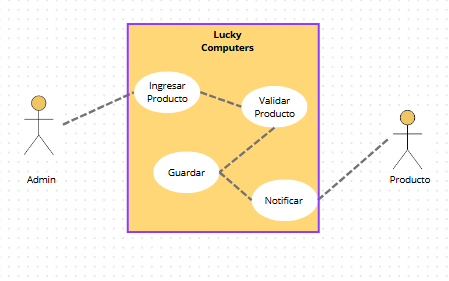
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO V – Gestionar Cliente** | | |
| **Descripción** | Permite al administrador modificar clientes del sistema. | |
| **Actores** | Administrador y cliente | |
| **Precondiciones** | El administrador y cliente previamente debe estar registrado en el sistema. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Editar Cliente | |
| 1 | Buscar Cliente. |
| 2 | Editar cliente. |
| 3 | Guardar. |
| 4 | Notificar |
| 5 | Validar cliente existente |
| **Excepciones** |  |  |
|  | 1 | Información errónea. |
|  | 2 | Campos en blanco. |



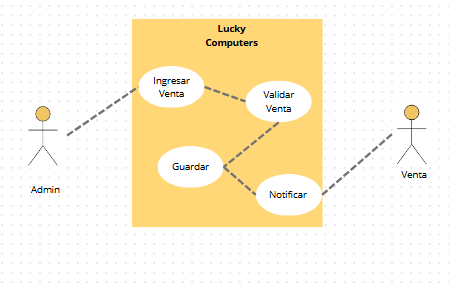
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO VI – Gestionar Cliente** | | |
| **Descripción** | Permite al administrador eliminar clientes del sistema. | |
| **Actores** | Administrador y cliente | |
| **Precondiciones** | El administrador y cliente previamente debe estar registrado en el sistema. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Eliminar Cliente | |
| 1 | Buscar usuarios. |
| 2 | Eliminar usuario. |
| 3 | Guardar. |
| 4 | Notificar |
| **Excepciones** |  |  |
|  | 1 | Información errónea. |
|  | 2 | Información duplicada. |



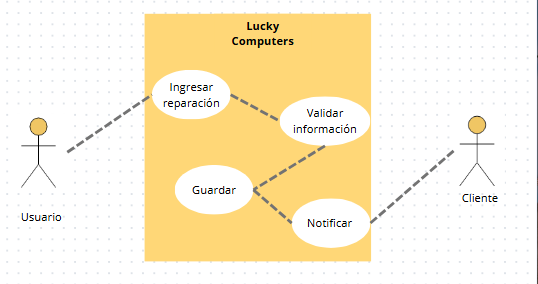
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO VII – Gestionar Productos** | | |
| **Descripción** | Permite al administrador agregar los productos. | |
| **Actores** | Administrador | |
| **Precondiciones** | El administrador previamente debe estar registrado en el sistema. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Ingresar Productos | |
| 1 | Ingresar producto. |
| 2 | Validar producto |
| 3 | Guardar producto |
| 4 | Notificar ingreso |
| **Excepciones** |  |  |
|  | 1 | Información errónea. |
|  | 2 | Información duplicada. |



|  |  |  |
| --- | --- | --- |
| **CASOS DE USO VIII – Gestionar Ventas** | | |
| **Descripción** | Permite al administrador ingresar venta. | |
| **Actores** | Administrador | |
| **Precondiciones** | El administrador previamente debe estar registrado en el sistema. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Ingresar venta | |
| 1 | Validar |
| 2 | Guardar |
| 3 | Notificar |
| **Excepciones** |  |  |
|  | 1 | Información errónea. |
|  | 2 | Información duplicada. |



|  |  |  |
| --- | --- | --- |
| **CASOS DE USO IX – Gestionar Reparaciones** | | |
| **Descripción** | Permite al usuario ingresar reparaciones. | |
| **Actores** | Usuario, Cliente | |
| **Precondiciones** | El usuario y cliente previamente debe estar registrado en el sistema. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Ingresar reparación | |
| 1 | Validamos información |
| 2 | Guardar |
| 3 | Notificar |
| **Excepciones** |  |  |
|  | 1 | Información errónea. |
|  | 2 | Campos en blanco. |

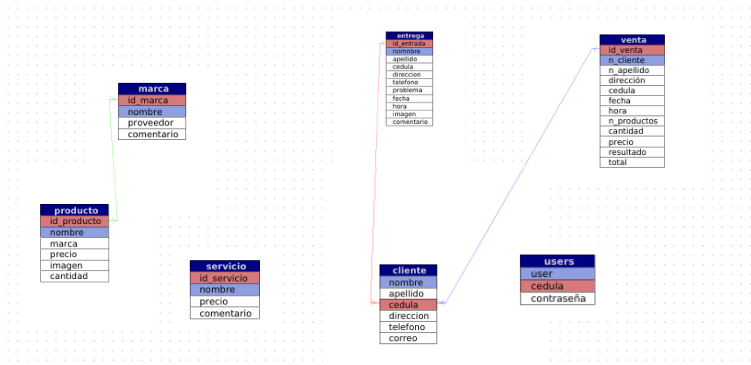


* + 1. **Diseño de Base de Datos**
       1. **Modelo lógico de la Base de Datos**

El Modelo Lógico de una Base de Datos es una representación detallada

y estructurada de la organización de los datos en una base de datos, que utiliza una notación gráfica y/o un lenguaje formal de definición de datos para describir los objetos y relaciones que existen entre ellos.

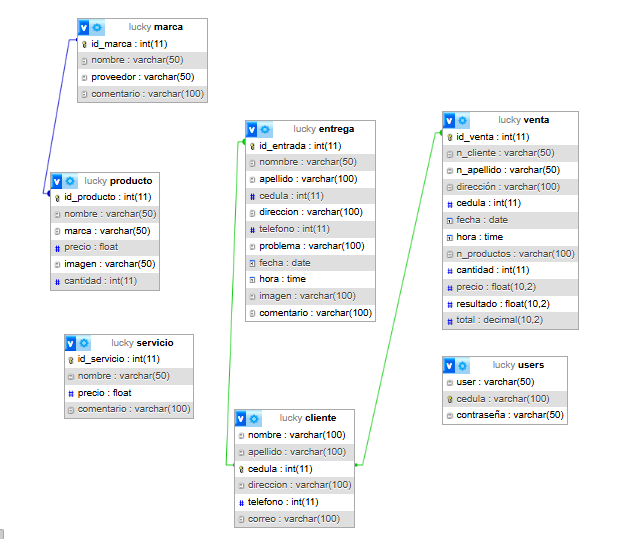
Este modelo se enfoca en la estructura lógica de la base de datos, es decir, en cómo se organiza la información y cómo se relacionan los diferentes elementos que la componen. El objetivo del modelo lógico es definir la estructura de la base de datos de una manera que permita a los usuarios interactuar con ella de manera eficiente y efectiva.



El modelo lógico describe la base de datos en términos de tablas, columnas y relaciones entre ellas, y se utiliza para guiar el diseño de la base de datos y para garantizar que cumpla con los requisitos del usuario. Además, proporciona una base para la implementación física de la base de datos en un sistema de gestión de bases de datos específico.

* + - 1. **Modelo físico de la Base de Datos**

El Modelo Físico de una Base de Datos es una representación detallada de la estructura física de la base de datos, que describe cómo se almacenan los datos en el disco duro o en cualquier otro medio de almacenamiento.



* + - 1. **Diccionario de datos**

Un Diccionario de Datos de una Base de Datos es una colección de metadatos que describe los datos, estructuras y relaciones que existen en una base de datos. Es una herramienta que permite a los usuarios entender la estructura y contenido de la base de datos, lo que facilita el desarrollo de aplicaciones, el mantenimiento y la administración de la base de datos.

El diccionario de datos contiene información como el nombre de la tabla, las columnas y sus tipos de datos, las relaciones entre las tablas, los índices, las restricciones y otras características importantes de la base de datos.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SISTEMA:** Desarrollo de un sistema de facturación para Lucky Computers | | | | | Esta entidad hace referencia a administrador | | |
| **Entidad:** admin | | **LLAVE:** | | |
| **#** | **Campo** | | **Tipo** | **Descripción** | | **Restricción** | |
| **Nulo** | **Clave** |
| **01** | User | | Varchar(50) | Administrador User | | No |  |
| **02** | Password | | Varchar(50) | Contraseña | | No |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SISTEMA:** Desarrollo de un sistema de facturación para Lucky Computers | | | | | Esta entidad hace referencia a la tabla cliente | | |
| **Entidad:** cliente | | **LLAVE:** | | |
| **#** | **Campo** | | **Tipo** | **Descripción** | | **Restricción** | |
| **Nulo** | **Clave** |
| **01** | Nombre | | Varchar(100) | Nombre cliente | | Si |  |
| **02** | Apellido | | Varchar(100) | Apellido | | Si |  |
| **03** | Cedula | | Int(11) | Cedula | | No |  |
| **04** | Dirección | | Varchar(200) | Dirección del cliente | | Si |  |
| **05** | Telefono | | Int(20) | Celular | | Si |  |
| **06** | Correo | | Varchar(100) | Correo | | Si |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SISTEMA:** Desarrollo de un sistema de facturación para Lucky Computers | | | | | Esta entidad hace referencia a la tabla productos | | |
| **Entidad:** servicio | | **LLAVE:** | | |
| **#** | **Campo** | | **Tipo** | **Descripción** | | **Restricción** | |
| **Nulo** | **Clave** |
| **01** | Id\_producto | | Int | Código | | No |  |
| **02** | Nombre | | Varchar(100) | Nombre | | Si |  |
| **03** | Precio | | decimal(10,2) | Precio | | Si |  |
| **04** | Servicio | | Varchar(50) | Servicio | | Si |  |
| **05** | Imagen | | Varchar(100) | Imagen | | Si |  |
| **06** | Cantidad | | Int(11) | Cantidad | | Si |  |

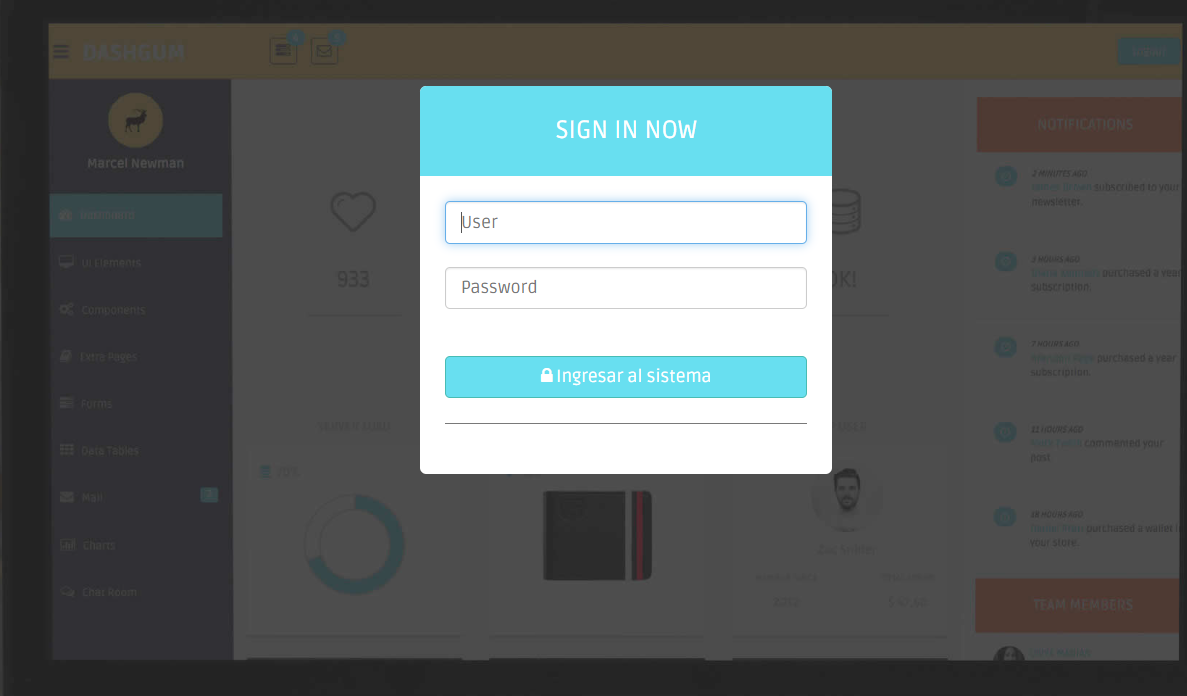
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SISTEMA:** Desarrollo de un sistema de facturación para Lucky Computers | | | | | Esta entidad hace referencia a user | | |
| **Entidad:** User | | **LLAVE:** | | |
| **#** | **Campo** | | **Tipo** | **Descripción** | | **Restricción** | |
| **Nulo** | **Clave** |
| **01** | User | | Varchar(50) | Nombre de usuario | | Si |  |
| **02** | Cedula | | Varchar(100) | Cedula | | No |  |
| **03** | Contraseña | | Varchar(50) | Contraseña | | Si |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SISTEMA:** Desarrollo de un sistema de facturación para Lucky Computers | | | | | Esta entidad hace referencia a ventas | | |
| **Entidad:** Venta | | **LLAVE:** | | |
| **#** | **Campo** | | **Tipo** | **Descripción** | | **Restricción** | |
| **Nulo** | **Clave** |
| **01** | Id\_venta | | int(11) | Codigo de venta | | No |  |
| **02** | N\_cliente | | Varchar(50) | Nombre del cliente | | Si |  |
| **03** | N\_apellido | | Varchar(50) | Apellido | | Si |  |
| **04** | Dirección | | Varchar(100) | Direccion | | Si |  |
| **05** | Cedula | | int(11) | Cedula | | Si |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **06** | Fecha | Date | Fecha | Si |  |
| **07** | N\_productos | Varchar(100) | Nombre del producto | Si |  |
| **08** | Servicio | Varchar(50) | Servicio | Si |  |
| **09** | Cantidad | Int(11) | Cantidad | Si |  |
| **10** | Precio | Float(10,2) | Precio | Si |  |
| **11** | Resultado | Float(10,2) | Resultado | Si |  |
| **12** | Total | Decimal(10,2) | Total | Si |  |

* + 1. **Interfaz de Usuario**

La interfaz de usuario (UI, por sus siglas en inglés) se refiere a la forma en que una persona interactúa con un dispositivo o sistema, ya sea una aplicación, un sitio web, un software o cualquier otra cosa que tenga una pantalla y controles interactivos. En otras palabras, la interfaz de usuario es la forma en que los usuarios interactúan con un sistema para realizar una tarea o cumplir un objetivo.



**Figura 1: Login del sistema “Lucky Computers” para ingreso a facturar**

La interfaz de usuario se compone de varios elementos, como iconos, botones, menús, ventanas, formularios, campos de texto y gráficos, entre otros. Estos elementos se organizan y presentan de manera que los usuarios puedan entender fácilmente cómo utilizar el sistema.

El objetivo principal de la interfaz de usuario es facilitar la interacción entre los usuarios y el sistema, y hacer que la experiencia del usuario sea lo más intuitiva y sencilla posible. Una buena interfaz de usuario debe ser fácil de usar, fácil de entender y fácil de navegar, lo que ayuda a los usuarios a realizar sus tareas de manera efectiva y eficiente.

**Figura 2: Modulo Administrador**

Tabla

AI-generated content may be incorrect.

El esquema de la **figura 2** se establecen los demás módulos del sistema.

**Sección 1:** Se encuentra todos los botones para entrar a cada uno de los módulos del sistema.

**Sección 2:** Menciona en que modulo se encuentra.

**Sección 3:** Rellenado de formulario

CAPITULO IV

PROGRAMACION

# **PROGRAMACION**

## **4.1. Ventajas y desventajas del lenguaje de programación.**

“Flask” es un marco de trabajo web WSGI sencillo y ligero. Su diseño permite un arranque rápido y sencillo, con la posibilidad de escalar a aplicaciones más complejas. Inicialmente, era un simple envoltorio alrededor de Werkzeug y Jinja, pero ahora se ha convertido en uno de los marcos de trabajo web más populares en Python.

Flask ofrece configuraciones y convenciones con valores predeterminados razonables para empezar. Este marco de trabajo en Python es compacto y liviano, proporcionando herramientas y funciones útiles que facilitan la creación de aplicaciones web en Python. Proporciona a los desarrolladores flexibilidad y un marco más accesible para los nuevos desarrolladores, ya que permite crear una aplicación web rápidamente usando solo un archivo Python3.

Su sintaxis básica se deriva de Python y utiliza el modelo de objetos de Flask, similar al de Django, pero con mejoras inspiradas en otros lenguajes.

Flask es también extensible, no impone una estructura de directorio específica ni requiere código estándar complicado antes de empezar. Flask utiliza el motor de plantillas Jinja para generar dinámicamente páginas HTML utilizando conceptos familiares de Python como variables, bucles, listas, etc

Ventajas  
  
Las ventajas que Python Flask ofrece en comparación con otros lenguajes de programación son:

* Marco de trabajo Micro: Flask es ideal para desarrollar una aplicación básica o para crear una aplicación de manera rápida y eficiente. • Incluye un servidor web de desarrollo: No es necesario tener una infraestructura con un servidor web para probar las aplicaciones web, simplemente se puede ejecutar un servidor web de manera sencilla para observar los resultados obtenidos.
* Depurador y soporte para pruebas unitarias integrado: Si hay algún error en el código que se está desarrollando, se puede depurar ese error y también se pueden visualizar los valores de las variables.
* Compatibilidad con WSGI: Para servir las páginas web creadas en Python, se pueden utilizar servidores web que soporten este protocolo, el WSGI.
* Excelente manejo de rutas: Al trabajar con aplicaciones web creadas en Flask Python, se tiene un controlador que recibe todas las peticiones que hacen los clientes y se puede determinar a qué ruta está accediendo el cliente para ejecutar el código correspondiente.
* Extensible: Flask es extensible y no impone una estructura de directorio específica ni requiere código estándar complicado antes de comenzar. Existe una gran cantidad de extensiones (plugins) que se pueden instalar con Flask para añadir funcionalidad.
* Fomenta la colaboración: Todas las aplicaciones construidas con Flask tendrán los mismos elementos y los mismos archivos.

**Desventajas**

* Microframework: Siendo un microframework, cada programador tiene la libertad de decidir cómo expandir la aplicación que está desarrollando, lo que podría resultar en la ruptura de las buenas prácticas y los estándares.
* Ausencia de librerías integradas: Flask no incluye librerías integradas, por lo que es necesario descargarlas para poder utilizar diferentes funciones. Esto puede generar problemas al realizar migraciones o pruebas unitarias.
* Requiere un ORM externo: Para conectar con bases de datos, es necesario utilizar un mapeo de objetos relacionales (ORM) externo.
* No incluye programación asíncrona: Flask no incluye programación asíncrona, lo que puede ser una limitación para algunas aplicaciones.
* Configuración compleja: La configuración predeterminada para el diseño de aplicaciones en Flask puede ser bastante complicada en términos de reutilización y limpieza del código.
* Mayor costo en términos de tiempo de desarrollo y seguridad: Desde una perspectiva empresarial, Flask puede ser más costoso de usar que Django porque tiene menos funciones incorporadas y requiere más mantenimiento.

### **4.1.1. Herramientas de Programación**

**VISUAL STUDIO 2012**

****

[“**Visual Studio Code**” es un editor de código fuente ligero, potente y disponible en Windows, macOS, Linux e incluso en tu navegador1](https://code.visualstudio.com/). [Viene con soporte incorporado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes y tiempos de ejecución](https://code.visualstudio.com/).

Debido a los elementos que forman parte de Visual Studio Code, cualquier programador tendrá la posibilidad de completar un proyecto por completo sin ninguna necesidad de recurrir a otros programas ni elementos externos.

Algunas de las características de Visual Studio Code son:

[• **Edición de código redefinida**: Visual Studio Code es gratuito y está construido sobre código abierto. • **Integración con Git**: VS Code tiene un control de versiones integrado y soporte para Git. • **Depuración integrada**: Visual Studio Code viene con un depurador incorporado, lo que facilita la depuración y ejecución de aplicaciones desde el editor. • **Extensiones**: Puedes elegir entre cientos de extensiones para potenciar tu experiencia con VS Code. • **Personalizable**: Puedes personalizar la interfaz de usuario de VS Code y el diseño para que se ajuste a tu estilo de codificación1](https://code.visualstudio.com/)[. • **GitHub Copilot**: Escribe código más rápido y de manera más inteligente con GitHub Copilot, tu programador de pares de IA](https://code.visualstudio.com/)[. • **Control de código fuente integrado**: VS Code tiene un control de código fuente integrado y soporte para Git1](https://code.visualstudio.com/)[. • **GitHub Codespaces**: Codespaces proporciona entornos de desarrollo impulsados por la nube para cualquier actividad](https://code.visualstudio.com/)[. • **vscode.dev**: VS Code para la Web (también conocido como vscode.dev) proporciona una experiencia de Microsoft Visual Studio Code gratuita y completamente en tu navegador](https://code.visualstudio.com/).

Estas características hacen de Visual Studio Code una opción muy atractiva para el desarrollo de aplicaciones web y en la nube.

### **4.1.2. Herramientas de Diseño**

**MICROSOFT OFFICE 2016**



Esta edición de 2016 trae varias aplicaciones bien diferenciadas, cada una de la cuales tiene un cometido muy específico:

**Word:** el procesador y editor de textos.

**Excel:** el programa de hojas de cálculo.

**PowerPoint:** la aplicación de presentaciones y pases de diapositivas.

**Access:** la herramienta para trabajar con bases de datos.

Esta nueva edición del paquete ofimático de Microsoft se nos presenta con interesantes novedades centradas principalmente en cinco campos:

**Eficiencia:** el nuevo Office es más rápido y eficiente. Con la nueva función Tell Me podrás decirle a Word, Excel o PowerPoint exactamente lo que quieres hacer para que estas apps te guíen a lo largo del proceso. Además, con la nueva búsqueda inteligente de Bing podrás realizar consultas en Internet sin ni siquiera salir del documento sobre el que estás trabajando.

**Accesibilidad:** gracias a las opciones de almacenamiento en la nube de OneDrive y que las nuevas aplicaciones de Office están adaptadas tanto a Windows tradicionales como a variantes del sistema operativo concebidas para pantallas táctiles, podrás acceder y trabajar sobre tus documentos desde donde quieras y cuando quieras, incluso desde un móvil o un Mac.

**Protección de datos:** otra característica destacada de este nuevo paquete son sus políticas de Prevención de Pérdida de Datos o Data Loss Prevention (DLP) para proteger documentos de uso compartido de infracciones en sus políticas de uso, lo que implica la identificación de material sensible y evitar su compartición accidental a través de las herramientas de Office.

**Integración en Windows 10:** aunque funcione en versiones anteriores del sistema operativo de Microsoft, este Office ha sido diseñado pensando en Windows 10. Por ello se integra perfectamente con Cortana, para que el asistente virtual te ayude en todas las tareas que tengas que realizar con sus herramientas, sobre todo Outlook, y se sincroniza perfectamente con las aplicaciones móviles diseñadas para Windows 10 en teléfonos y tabletas.

### **4.1.3. Herramienta de Administración**

**MongoDB**



MongoDB es un sistema de administración de datos NoSQL, gratuito y de código abierto, que proporciona un almacén de datos flexible y rico en contenido para aplicaciones de escritorio y sitios web ligeros.

**Requisitos:**

* Sistema operativo compatible Linux: Ubuntu 16.04, RHEL/CentOS 7.2; Windows: Server 2012 R2, Server 2016, 7/8/10; macOS: 10.12 y posteriores.
* Procesador Intel: procesador compatible con una velocidad mínima de 1 GHz o un procesador más rápido
* RAM Mínimo de 512 MB para MongoDB con herramientas y MongoDB con servicios avanzados. Sin embargo, para un rendimiento óptimo, se recomienda al menos 4 GB.
* Espacio en disco duro Dependiendo de la cantidad de datos que planea almacenar, pero para empezar, se recomienda al menos 5 GB de espacio en disco

## **4.2. CÓDIGO FUENTE CAPA ENTIDADES**

**Cliente**

from flask import Blueprint, render\_template, request, flash, session, redirect, url\_for

from controllers.database import Conexion as dbase

from modules.cliente import Cliente

from pymongo import MongoClient

db = dbase()

cliente = Blueprint('cliente', \_\_name\_\_)

# Ingreso con un Id unico

@cliente.route('/admin/in\_cliente',methods=['GET','POST'])

def adcli():

    if 'username' not in session:

        flash("Inicia sesion con tu usuario y contraseña")

        return redirect(url\_for('cliente.index'))  # Redirige al usuario al inicio si no está en la sesión

    if request.method == 'POST':

        cliente = db["cliente"]

        nombre = request.form['nombre']

        apellido = request.form['apellido']

        cedula = request.form['cedula']

        direccion = request.form['direccion']

        telefono = request.form['telefono']

        correo = request.form['correo']

        exist\_cedula = cliente.find\_one({"cedula": cedula})

        exist\_telefono = cliente.find\_one({"telefono": telefono})

        exist\_correo = cliente.find\_one({"correo": correo})

        if exist\_cedula:

            flash("La cedula ya existe" , "danger")

            return redirect(url\_for('cliente.adcli'))

        elif exist\_telefono:

            flash("El celular ya existe" ,"danger")

            return redirect(url\_for('cliente.adcli'))

        elif exist\_correo:

            flash("El correo ya existe" , "danger")

            return redirect(url\_for('cliente.adcli'))

        else:

            client = Cliente( nombre, apellido, cedula, direccion, telefono, correo)

            cliente.insert\_one(client.ClienteDBCollection())

            flash("Enviado a la base de datos" , "success")

            return redirect(url\_for('cliente.adcli'))

    else:

        return render\_template('admin/in\_cliente.html',message=request.args.get('message'))

@cliente.route('/edit\_cli/<string:edacli>', methods=['GET', 'POST'])#

def edit\_cli(edacli):

    cliente = db['cliente']

    nombre = request.form["nombre"]

    apellido = request.form["apellido"]

    cedula = request.form['cedula']

    direccion = request.form["direccion"]

    telefono = request.form["telefono"]

    correo = request.form['correo']

    if nombre and apellido  and cedula and direccion and telefono and correo:

        cliente.update\_one({'cedula' : edacli}, {'$set' : {'nombre' : nombre, 'apellido' : apellido, 'cedula' : cedula ,"direccion" :direccion , "telefono" : telefono , "correo" : correo}})

        flash("Cliente  "+ nombre + " con  cedula " + cedula + " editado correctamente " , "success")

        return redirect(url\_for('cliente.v\_cli'))

    else:

        return render\_template('admin/cliente.html')

# \* Eliminar cliente

@cliente.route('/delete\_cli/<string:eliacli>')

def delete\_cli(eliacli):

    cliente = db["cliente"]

    documento =  cliente.find\_one({"cedula":eliacli})

    nombre = documento["nombre"]

    apellido = documento["apellido"]

    cedula = documento["cedula"]

    cliente.delete\_one({"cedula":eliacli})

    flash("Cliente  "+ nombre +" "+ apellido +" con cedula " + cedula  + " eliminado correctamente" , "success")

    return redirect(url\_for('cliente.v\_cli'))

# Visualizar cliente

@cliente.route("/admin/cliente")

def v\_cli():

    if 'username' not in session:

        flash("Inicia sesion con tu usuario y contraseña")

        return redirect(url\_for('cliente.index'))

    cliente = db['cliente'].find()

    return render\_template("admin/cliente.html", cliente=cliente)

**Producto**

from flask import Blueprint, render\_template, request, flash, session, redirect, url\_for, current\_app

from controllers.database import Conexion as dbase

from werkzeug.utils import secure\_filename

import os

from modules.producto import Producto

from pymongo import MongoClient

db = dbase()

producto = Blueprint('producto', \_\_name\_\_)

# Este es para lo que es las imagenes

@producto.route('/alguna\_ruta')

def alguna\_funcion():

    UPLOAD\_FOLDER = current\_app.config['UPLOAD\_FOLDER']

# codigo de verificacion de productos con las imagenes

ALLOWED\_EXTENSIONS = {'png', 'jpg', 'jpeg', 'gif'}

# Este codigo es para las  imagenes

def allowed\_file(filename):

    return '.' in filename and \

        filename.rsplit('.', 1)[1].lower() in ALLOWED\_EXTENSIONS

# Este codigo es para lo que es el ID

def get\_next\_sequence(name):

    seq = db.seqs.find\_one({'\_id': name})

    if seq is None:

        # Inicializa 'productoId' en 220 si no existe

        db.seqs.insert\_one({'\_id': 'productoId', 'seq': 0})

        seq = db.seqs.find\_one({'\_id': name})

    result = db.seqs.find\_one\_and\_update(

        {'\_id': name},

        {'$inc': {'seq': 1}},

        return\_document=True

    )

    return result.get('seq')

@producto.route('/admin/in\_producto', methods=['GET', 'POST'])

def adpro():

    if 'username' not in session:

        flash("Inicia sesion con tu usuario y contraseña")

        return redirect(url\_for('user.index'))

    marcas = db['marca'].find()

    if request.method == 'POST':

        id\_producto = str(get\_next\_sequence('productoId')).zfill(3)

        producto = db["producto"]

        nombre = request.form['nombre']

        marca = request.form['marca']

        precio = request.form['precio']

        cantidad = request.form['cantidad']

        exist\_nombre\_color = producto.find\_one({"nombre": nombre})

        if exist\_nombre\_color:

            flash("El nombre del producto  ya existen", "danger")

            return redirect(url\_for('producto.adpro'))

        else:

            if 'imagen' not in request.files:

                flash('No file part')

                return redirect(request.url)

            file = request.files['imagen']

            if file.filename == '':

                flash('Selecciona una imagen')

                return redirect(request.url)

            if file and allowed\_file(file.filename):

                filename = secure\_filename(file.filename)

                file\_path = os.path.join(current\_app.config['UPLOAD\_FOLDER'], filename)

                file.save(file\_path)

                imagen\_filename = filename

            produc = Producto(id\_producto, nombre,marca, precio, imagen\_filename, cantidad)

            producto.insert\_one(produc.ProductoDBCollection())

            flash("Producto agregado correctamente" , "success")

            return redirect(url\_for('producto.adpro'))

    else:

        return render\_template('admin/in\_producto.html',marcas = marcas)

# Editar Producto

@producto.route('/edit\_pro/<string:edipro>', methods=['GET', 'POST'])

def edit\_pro(edipro):

    if 'username' not in session:

        flash("Inicia sesion con tu usuario y contraseña")

        return redirect(url\_for('producto.index'))

    producto = db['producto']

    producto\_existente = producto.find\_one({"id\_producto": edipro})

    if request.method == 'POST':

        id\_producto = request.form["id\_producto"]

        nombre = request.form["nombre"]

        precio = request.form["precio"]

        cantidad = request.form["cantidad"]

        if "imagen" in request.files and request.files['imagen'].filename != '':

            file = request.files['imagen']

            if file and allowed\_file(file.filename):

                filename = secure\_filename(file.filename)

                file\_path = os.path.join(current\_app.config['UPLOAD\_FOLDER'], filename)

                file.save(file\_path)

                imagen\_filename = os.path.join('img', filename)

        else:

            imagen\_filename = producto\_existente['imagen']

        campos = [id\_producto, nombre, precio, cantidad]

        try:

            if all(campos):

                producto.update\_one({"id\_producto": edipro}, {"$set": {

                    "id\_producto": id\_producto,

                    "nombre": nombre,

                    "precio": precio,

                    "cantidad": cantidad,

                    "imagen": imagen\_filename

                }})

                flash("Producto " + nombre + " actualizado correctamente" ,"success")

                return redirect(url\_for('producto.v\_product'))

            else:

                flash("Todos los campos son obligatorios")

                return redirect(url\_for('producto.edit\_pro', edipro=edipro))

        except Exception as e:

            print(e)

            flash("Ha ocurrido un error" , "danger")

            return redirect(url\_for('producto.edit\_pro', edipro=edipro))

    return render\_template('admin/edit\_pro.html', producto=producto\_existente)

# Eliminar Producto

@producto.route('/delete\_pr/<string:eliadpro>')

def delete\_pr(eliadpro):

    producto = db["producto"]

    documento = producto.find\_one({"id\_producto": eliadpro})

    if documento:

        nombre = documento["nombre"]

        producto.delete\_one({"id\_producto": eliadpro})

        flash("Producto " + nombre + " eliminado correctamente" , "success")

    else:

        flash("Producto no encontrado" , "danger")

    return redirect(url\_for('producto.v\_product'))

# Visualizar producto

@producto.route("/admin/producto")

def v\_product():

    if 'username' not in session:

        flash("Inicia sesion con tu usuario y contraseña")

        return redirect(url\_for('producto.index'))

    producto = db["producto"].find()

    return render\_template('admin/producto.html', producto=producto)

**Venta**

from flask import Blueprint, make\_response,send\_file, render\_template, request, flash, session, redirect, url\_for

from controllers.database import Conexion as dbase

from modules.venta import Venta

from pymongo import MongoClient

from flask import jsonify

from bson import ObjectId

from reportlab.pdfgen import canvas # \*pip install reportlab este es para imprimir reportes

from reportlab.lib.pagesizes import letter #\* pip install reportlab

from reportlab.lib.units import inch

from reportlab.pdfbase import pdfmetrics

from reportlab.pdfbase.ttfonts import TTFont

from reportlab.platypus import SimpleDocTemplate, Table, Paragraph, TableStyle, Spacer ,Image

from reportlab.lib import colors

from reportlab.lib.styles import getSampleStyleSheet ,ParagraphStyle

import io

from flask import Flask

db = dbase()

from mail\_config import mail

from flask\_mail import Message

# Importar mail después de definir el Blueprint

venta = Blueprint('venta', \_\_name\_\_)

mail = None

# Función para inicializar mail

def init\_mail(mail\_instance):

    global mail

    mail = mail\_instance

# Este codigo es para lo que es el ID

def get\_next\_sequence(name):

    seq = db.seqs.find\_one({'\_id': name})

    if seq is None:

        # Inicializa 'productoId' en 220 si no existe

        db.seqs.insert\_one({'\_id': 'ventaId', 'seq': 0})

        seq = db.seqs.find\_one({'\_id': name})

    result = db.seqs.find\_one\_and\_update(

        {'\_id': name},

        {'$inc': {'seq': 1}},

        return\_document=True

    )

    return result.get('seq')

# Registro de ventas

@venta.route("/admin/in\_venta", methods=['GET', 'POST'])

def adventa():

    if 'username' not in session:

        flash("Inicia sesion con tu usuario y contraseña")

        return redirect(url\_for('cliente.index'))

    cliente = db["cliente"].find()

    producto = db["producto"].find()

    servicio = db["servicio"].find()

    if request.method == 'POST':

        id\_venta = str(get\_next\_sequence('ventaId')).zfill(1)

        venta = db["venta"]

        n\_cliente = request.form["n\_cliente"]

        n\_apellido = request.form["n\_apellido"]

        direccion = request.form["direccion"]

        cedula = request.form["cedula"]

        fecha = request.form["fecha"]

        hora = request.form["hora"]

        # Recoger los productos

        id\_productos = request.form.getlist("id\_producto")

        n\_productos = request.form.getlist("n\_productos")

        cantidades = request.form.getlist("cantidad")

        precios = request.form.getlist("precio")

        resultados = request.form.getlist("resultado")

        totales = request.form.getlist("total")

        # Debugging

        print("ID Productos:", id\_productos)

        print("Productos:", n\_productos)

        print("Cantidades:", cantidades)

        print("Precios:", precios)

        print("Resultados:", resultados)

        print("Totales:", totales)

        productos = []

        for i in range(len(n\_productos)):

            producto = {

                "id\_producto": id\_productos[i] if i < len(id\_productos) else '',

                "n\_producto": n\_productos[i] if i < len(n\_productos) else '',

                "cantidad": cantidades[i] if i < len(cantidades) else '',

                "precio": precios[i] if i < len(precios) else '',

                "resultado": resultados[i] if i < len(resultados) else '',

                "total": totales[i] if i < len(totales) else ''

            }

            productos.append(producto)

        # Crear el documento de venta

        venta\_documento = {

            "id\_venta": id\_venta,

            "n\_cliente": n\_cliente,

            "n\_apellido": n\_apellido,

            "direccion": direccion,

            "cedula": cedula,

            "fecha": fecha,

            "hora": hora,

            "productos": productos

        }

        # Insertar el documento en la colección de ventas

        venta.insert\_one(venta\_documento)

        print("Documento de Venta:", venta\_documento)

        # Actualizar las cantidades de los productos

        for i in range(len(id\_productos)):

            id\_producto = id\_productos[i]

            cantidad\_vendida = cantidades[i]

            if cantidad\_vendida:

                cantidad\_vendida = int(cantidad\_vendida)

                # Obtener el producto de la base de datos

                producto\_db = db["producto"].find\_one({"id\_producto": id\_producto})

                if producto\_db:

                    nueva\_cantidad = int(producto\_db["cantidad"]) - cantidad\_vendida

                    # Actualizar la cantidad del producto en la base de datos

                    db["producto"].update\_one({"id\_producto": id\_producto}, {"$set": {"cantidad": str(nueva\_cantidad)}})

                    print(f"Producto {id\_producto} actualizado. Nueva cantidad: {nueva\_cantidad}")

                else:

                    print(f"Producto con id {id\_producto} no encontrado, la operación continúa")

            else:

                print(f"Cantidad no válida para el producto {id\_producto}")

        flash("Venta registrada con éxito y cantidades actualizadas","success")

        return redirect(url\_for('venta.adventa'))

    else:

        return render\_template("admin/in\_venta.html", cliente=cliente, producto=producto, servicio=servicio)

# Visualizar venta

@venta.route("/admin/venta")

def v\_cli():

    if 'username' not in session:

        flash("Inicia sesion con tu usuario y contraseña")

        return redirect(url\_for('venta.index'))

    venta = db['venta'].find()

    return render\_template("admin/venta.html", venta=venta)

# Visualizar detalles del cliente por ID y que se pueda revisar

@venta.route("/admin/venta/<id>")

def v\_cliente(id):

    if 'username' not in session:

        flash("Inicia sesión con tu usuario y contraseña")

        return redirect(url\_for('venta.index'))

    cliente = db['venta'].find\_one({"\_id": ObjectId(id)})

    return render\_template("admin/v\_cliente.html", cliente=cliente)

# Nueva ruta para generar el PDF

@venta.route("/admin/venta/<id>/pdf")

def generar\_pdf(id):

    if 'username' not in session:

        flash("Inicia sesión con tu usuario y contraseña")

        return redirect(url\_for('venta.index'))

    cliente = db['venta'].find\_one({"\_id": ObjectId(id)})

    if not cliente:

        flash("Cliente no encontrado")

        return redirect(url\_for('venta.index'))

    # Crear el PDF en memoria

    buffer = io.BytesIO()

    c = canvas.Canvas(buffer, pagesize=letter)

    width, height = letter

    # Título

    c.setFont("Helvetica-Bold", 20)

    c.drawCentredString(width / 2.0, height - 50, "LUCKYCOMPUTERS")

    # Detalles del cliente

    c.setFont("Helvetica", 12)

    c.drawString(50, height - 100, f"FACTURA: {cliente['id\_venta']}")

    c.drawString(50, height - 120, f"Nombre: {cliente['n\_cliente']}")

    c.drawString(50, height - 140, f"Apellido: {cliente['n\_apellido']}")

    c.drawString(50, height - 160, f"Direccion: {cliente['direccion']}")

    c.drawString(50, height - 180, f"Cedula: {cliente['cedula']}")

    c.drawString(50, height - 200, f"Fecha: {cliente['fecha']}")

    # Tabla de productos

    c.drawString(50, height - 240, "Nombre de los productos")

    c.drawString(300, height - 240, "Cantidad")

    c.drawString(400, height - 240, "Precio $")

    c.drawString(500, height - 240, "Total")

    y = height - 260

    for producto in cliente['productos']:

        c.drawString(50, y, producto['n\_producto'])

        c.drawString(300, y, str(producto['cantidad']))

        c.drawString(400, y, str(producto['precio']))

        c.drawString(500, y, str(producto['resultado']))

        y -= 20

    # Total

    c.setFont("Helvetica-Bold", 14)

    c.drawString(50, y - 40, f"Total: {cliente['productos'][0]['total']}")

    c.showPage()

    c.save()

    buffer.seek(0)

    return send\_file(buffer, as\_attachment=True, download\_name=f'factura\_{cliente["id\_venta"]}.pdf', mimetype='application/pdf')

# Función para obtener el correo del cliente

def obtener\_correo\_cliente(cedula):

    cliente = db['cliente'].find\_one({"cedula": cedula})

    if cliente:

        return cliente.get('correo')

    return None

# Función para generar el PDF

def generar\_pdf\_cliente(cliente):

    buffer = io.BytesIO()

    c = canvas.Canvas(buffer, pagesize=letter)

    width, height = letter

    # Título

    c.setFont("Helvetica-Bold", 20)

    c.drawCentredString(width / 2.0, height - 50, "LUCKYCOMPUTERS")

    # Detalles del cliente

    c.setFont("Helvetica", 12)

    c.drawString(50, height - 100, f"FACTURA: {cliente['id\_venta']}")

    c.drawString(50, height - 120, f"Nombre: {cliente['n\_cliente']}")

    c.drawString(50, height - 140, f"Apellido: {cliente['n\_apellido']}")

    c.drawString(50, height - 160, f"Direccion: {cliente['direccion']}")

    c.drawString(50, height - 180, f"Cedula: {cliente['cedula']}")

    c.drawString(50, height - 200, f"Fecha: {cliente['fecha']}")

    # Tabla de productos

    c.drawString(50, height - 240, "Nombre de los productos")

    c.drawString(300, height - 240, "Cantidad")

    c.drawString(400, height - 240, "Precio $")

    c.drawString(500, height - 240, "Total")

    y = height - 260

    for producto in cliente['productos']:

        c.drawString(50, y, producto['n\_producto'])

        c.drawString(300, y, str(producto['cantidad']))

        c.drawString(400, y, str(producto['precio']))

        c.drawString(500, y, str(producto['resultado']))

        y -= 20

    # Total

    c.setFont("Helvetica-Bold", 14)

    c.drawString(50, y - 40, f"Total: {cliente['productos'][0]['total']}")

    c.showPage()

    c.save()

    buffer.seek(0)

    return buffer

# Función para enviar el correo con el PDF adjunto

def enviar\_correo\_cliente(correo, pdf\_buffer, nombre\_cliente):

    try:

        msg = Message(

            subject="Gracias por tu compra",

            sender="tu\_correo@example.com",  # Remitente

            recipients=[correo],  # Destinatario

            body=f"Hola {nombre\_cliente},\n\nGracias por tu compra. Adjuntamos tu factura en PDF.\n\nSaludos,\nLUCKYCOMPUTERS"

        )

        # Adjuntar el PDF

        msg.attach(

            filename=f"factura\_{nombre\_cliente}.pdf",

            content\_type="application/pdf",

            data=pdf\_buffer.getvalue()

        )

        # Enviar el correo

        mail.send(msg)

        return True

    except Exception as e:

        print(f"Error al enviar el correo: {str(e)}")

        return False

# Ruta para enviar el PDF por correo

@venta.route("/admin/venta/<id>/enviar\_correo")

def enviar\_factura\_correo(id):

    if 'username' not in session:

        flash("Inicia sesión con tu usuario y contraseña")

        return redirect(url\_for('venta.index'))

    # Obtener los datos del cliente desde la colección venta

    cliente = db['venta'].find\_one({"\_id": ObjectId(id)})

    if not cliente:

        flash("Cliente no encontrado")

        return redirect(url\_for('venta.index'))

    # Obtener el correo del cliente desde la colección cliente

    correo\_cliente = obtener\_correo\_cliente(cliente['cedula'])

    if not correo\_cliente:

        flash("No se encontró el correo del cliente")

        return redirect(url\_for('venta.v\_cliente', id=id))

    # Generar el PDF

    pdf\_buffer = generar\_pdf\_cliente(cliente)

    # Enviar el correo con el PDF adjunto

    if enviar\_correo\_cliente(correo\_cliente, pdf\_buffer, cliente['n\_cliente']):

        flash("Correo enviado correctamente","success")

    else:

        flash("Error al enviar el correo","alert")

    return redirect(url\_for('venta.v\_cliente', id=id))

## **4.3. CODIGO FUENTE LOGICA DE NEGOCIOS**

**CLIENTE**

class Cliente:

def \_\_init\_\_(self, nombre, apellido, cedula, direccion, telefono, correo):

self.nombre = nombre

self.apellido = apellido

self.cedula = cedula

self.direccion = direccion

self.telefono = telefono

self.correo = correo

def ClienteDBCollection(self):

return {

"nombre": self.nombre,

"apellido": self.apellido,

"cedula": self.cedula,

"direccion": self.direccion,

"telefono": self.telefono,

"correo": self.correo,

}

**SERVICIO**

class Servicio:

def \_\_init\_\_(self, id\_servicio, nombre, precio, marca, imagen, cantidad):

self.id\_servicio = id\_servicio

self.nombre = nombre

self.precio = precio

self.marca = marca

self.imagen = imagen

self.cantidad = cantidad

def ServicioDBCollection(self):

return {

"id\_servicio": self.id\_servicio,

"nombre": self.nombre,

"precio": self.precio,

"marca": self.marca,

"imagen": self.imagen,

"cantidad": self.cantidad,

}

**USUARIO**

class User:

def \_\_init\_\_(self, user, cedula, contraseña):

self.user = user

self.cedula = cedula

self.contraseña = contraseña

def UserDBCollection(self):

return {

'user': self.user,

'cedula': self.cedula,

'contraseña': self.contraseña,

}

**VENTA**

class Venta:

def \_\_init\_\_(self, id\_venta, n\_cliente, n\_apellido, direccion, cedula, fecha,

n\_productos, color, cantidad, precio, resultado, total):

self.id\_venta = id\_venta

self.n\_cliente = n\_cliente

self.n\_apellido = n\_apellido

self.direccion = direccion

self.cedula = cedula

self.fecha = fecha

self.n\_productos = n\_productos

self.color = color

self.cantidad = cantidad

self.precio = precio

self.resultado = resultado

self.total = total

def VentaDBCollection(self):

return {

'id\_venta': self.id\_venta,

'n\_cliente': self.n\_cliente,

'n\_apellido': self.n\_apellido,

'direccion': self.direccion,

'cedula': self.cedula,

'fecha': self.fecha,

'n\_productos': self.n\_productos,

'color': self.color,

'cantidad': self.cantidad,

'precio': self.precio,

'resultado': self.resultado,

'total': self.total

}

CAPITULO V

CONCLUSIONES

Y RECOMENDACIONES

# **5. CONCLUSIONES Y RECOMENDACIONES**

## **5.1. Conclusiones**

Después de llevar a cabo los análisis pertinentes de los procesos que maneja la microempresa **“Lucky Computers”**, hemos identificado los problemas y necesidades que enfrentan diariamente los usuarios al gestionar la información y realizar sus tareas.

Como resultado, hemos diseñado e implementado un sistema de facturación que permite a **“Lucky Computers”** y a su equipo de trabajo optimizar las tareas relacionadas con la gestión de información de una forma ordenada, rápida y eficiente, mejorando significativamente el servicio ofrecido a sus clientes y propietarios.

Gracias al diseño e implementación de este sistema, hemos puesto en práctica los conocimientos adquiridos a lo largo de nuestros módulos de aprendizaje, ganando a su vez una valiosa experiencia en el desarrollo de software.

El sistema de facturación está operando de manera satisfactoria en un servidor local de la microempresa, cumpliendo con los resultados esperados por parte del propietario de **“Lucky Computers”**, dedicada a la venta de equipos, accesorios y servicios tecnológicos.

## **5.2. Recomendaciones**

Se sugiere a la Institución asignar proyectos de menor envergadura a los estudiantes en formación, para que vayan ganando experiencia en su desarrollo y estén mejor preparados al momento de llevar a cabo su tesis de grado.

Para los futuros egresados, se recomienda contar con un equipo de cómputo de buena calidad, con características técnicas sobresalientes que permitan trabajar de manera adecuada en proyectos como el sistema desarrollado para **Lucky Computers**. Esto evitará pérdidas de tiempo o información provocadas por congelamientos del equipo o fallas en el gestor de base de datos, considerando que estas herramientas suelen requerir una cantidad considerable de recursos para funcionar correctamente.

Al concluir el proyecto, es fundamental realizar una evaluación de todo lo que se ha alcanzado, considerando tanto los errores como los aciertos. Este análisis permitirá aprender de la experiencia, conocer mejor los pasos de ejecución y estar mejor preparados para afrontar situaciones similares en futuros desarrollos.

## **5.3. Anexos**

## **5.3.1. Manual de Usuario**

ACCESO AL SISTEMA

Interfaz de usuario gráfica, Aplicación

AI-generated content may be incorrect.

Aquí el usuario debe ingresar el usuario y contraseña otorgadas por el administrador del sistema. Si el usuario o contraseña son incorrectas no se habilita el boton Ingresar, caso contrario nos dara acceso al menú principal.

**MENÚ PRINCIPAL**

**Interfaz de usuario gráfica, Aplicación, Teams

AI-generated content may be incorrect.**

Mediante el Menú principal se puede acceder a todas las funciones, como registro de usurios ,ingreso de clientes ,ingreso de productos ,ventas.

**INGRESO DE USUARIOS**

Interfaz de usuario gráfica, Aplicación

AI-generated content may be incorrect.

En esta primera ventana es para ingresar los usuarios que podran usar el sistema de acuerdo al rol que tengan.

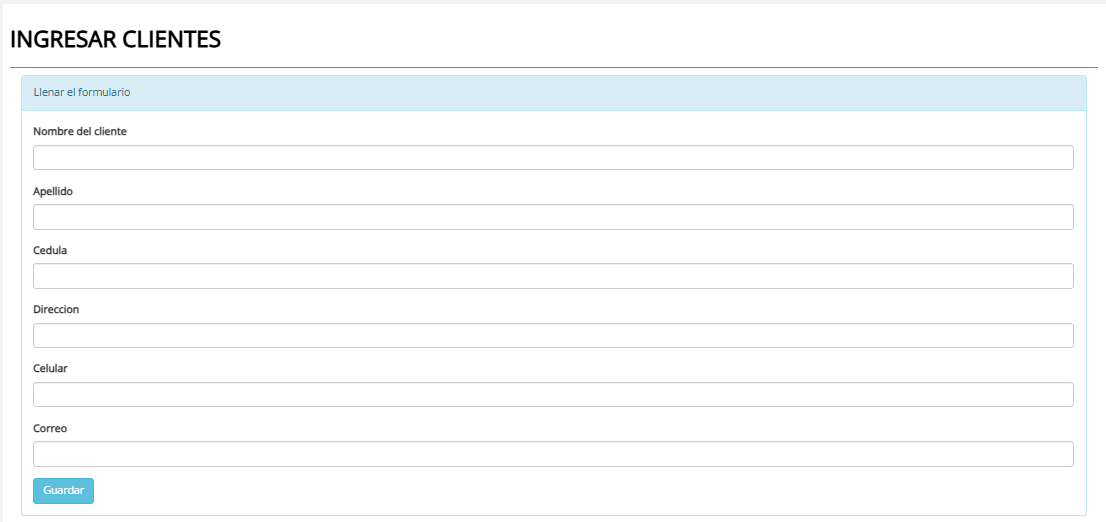
Tabla

AI-generated content may be incorrect.

En esta segunda ventana podra visualizar a los usuarios registrados.

**CLIENTES**

Este formulario podra ingresar a los clientes para poder hacer las ventas es necesario que se pueda ingresar cada uno de los datos ya que en este caso el sistema necesita todos los campos ya que no se puede dejar nada en blanco si lo hace el sistema lo que produce un error.



**INGRESO DE SERVICIOS**

En este formulario se ingresan los servicios que serán registrados en el sistema, como por ejemplo el nombre del servicio técnico a ofrecer. Para facilitar el proceso, se ha incorporado un selector que permite elegir el tipo de servicio que se está ingresando, lo cual es especialmente útil considerando que en el local existen servicios que se repiten con frecuencia.

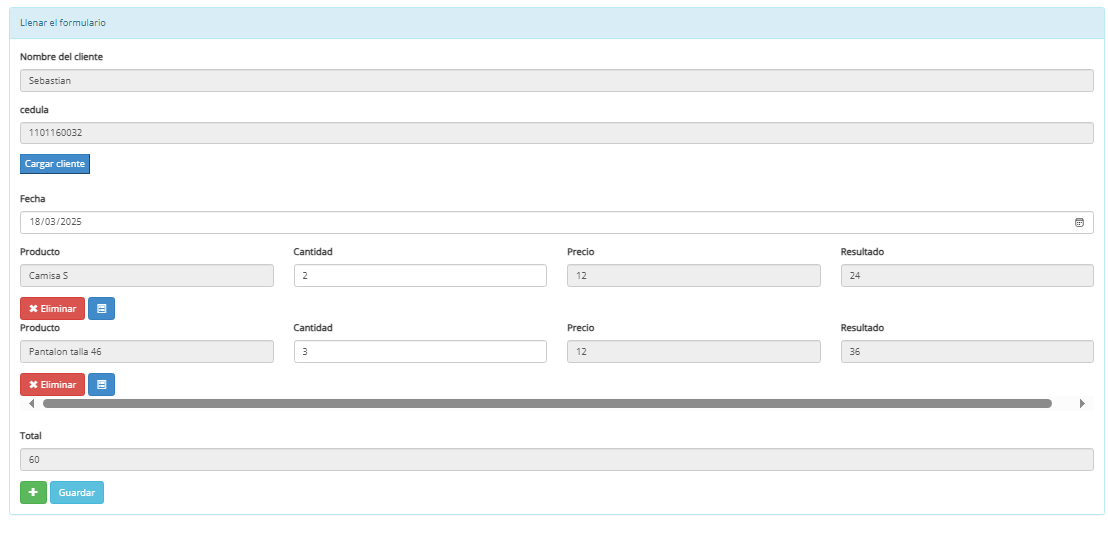
Además, es necesario especificar el costo del servicio, el cual ya debe incluir el IVA, y en caso de que el servicio incluya algún componente adicional, se puede detallar en un campo correspondiente. En otro apartado se debe ingresar una imagen ilustrativa que identifique el servicio, y más abajo, se indica la disponibilidad o número de solicitudes que se pueden gestionar, lo cual ayudará al control y organización interna.

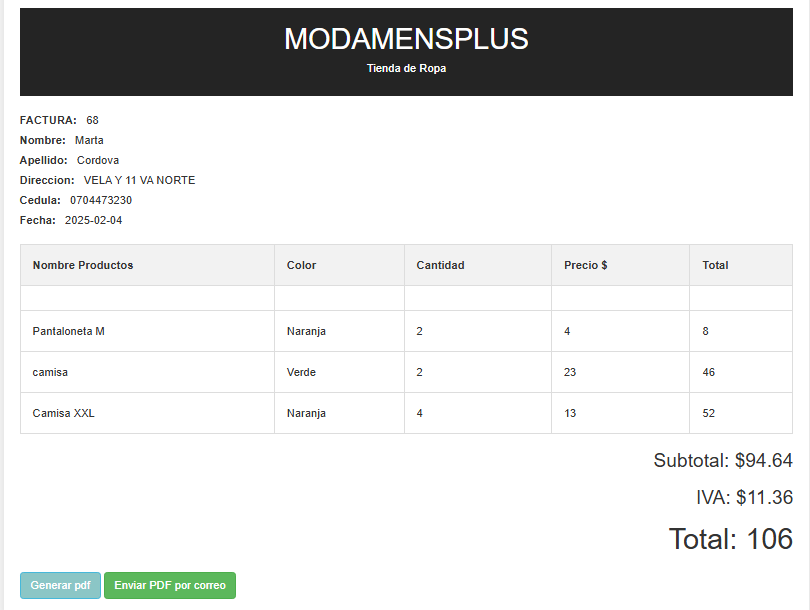
El sistema no permitirá generar facturas o registrar ventas si no hay disponibilidad del servicio, asegurando así una correcta gestión del inventario de servicios ofrecidos.



**VENTAS**

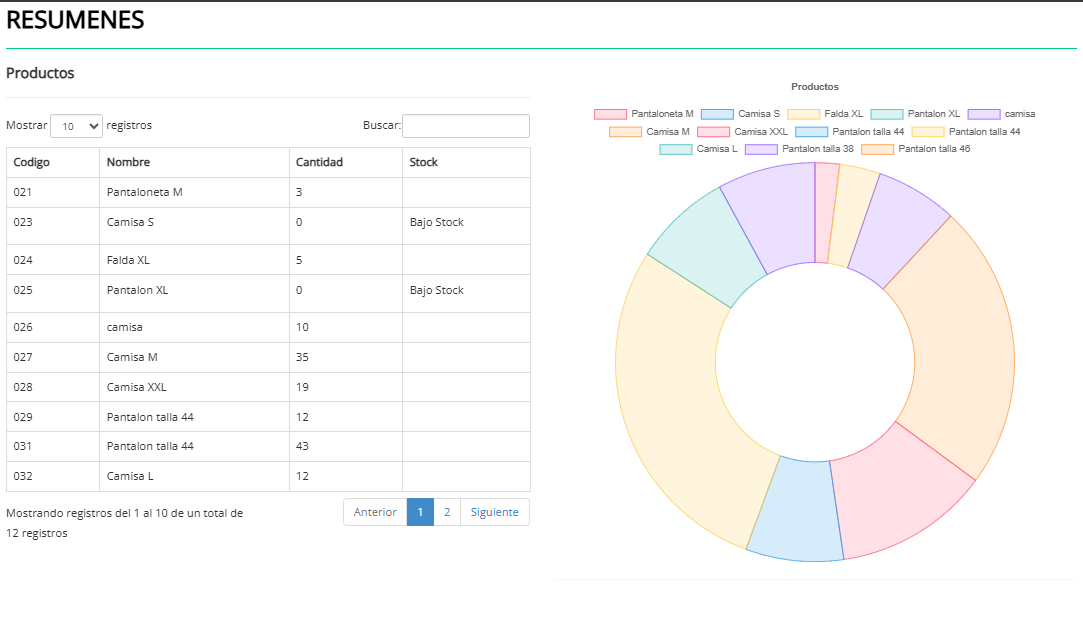
En este formulario se tendra que ingresar lo que es ventas de todos los productos de la tienda se tendra que colocar lo que es el cliente la fecha que se esta ingresando la venta y se podra el producto y la cantidad que esta saliendo de los productos ingresados el sistema en este lado no podra hacer una venta si la cantidad de productos es cero si todos los datos se encuentran bien el sistema lo valida y registra la venta realizada.

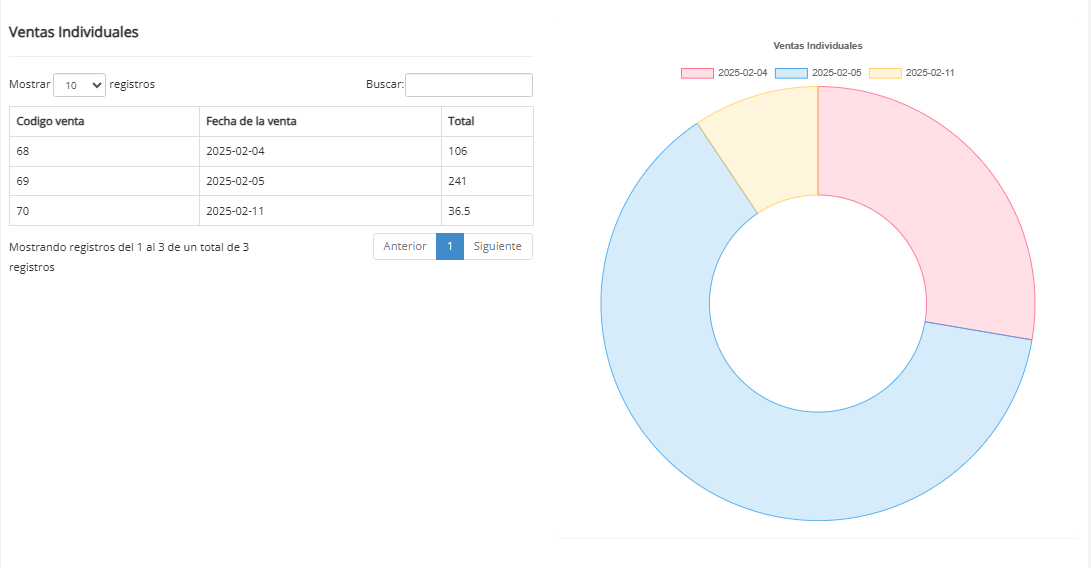




Se podra visualizar lo que compro el cliente la fecha de la compra y el detalle de los productos ademas se puede ver la generacion de pdf y enviar ese pdf al correo del cliente.

**RESUMEN**

****



En este apartado se podra visualizar lo que son resumenes de los productos y las ventas individuales de cada uno de lo que se sale de la tienda ademas de mostrar una tabla de cada venta y producto se puede mostrar en el lado derecho lo que son graficos estadisticos que reflejaran de una manera visual lo que sale de las ventas y productos .

**BIBLIOGRAFIA**

**Taty.** (2023, 10 de abril). Sitio web oficial. https://taty.com.ec/

**MongoDB.** (2023, 10 de abril). Base de datos NoSQL de código abierto. <https://www.mongodb.com/>

**Bootstrap.** (2023, 10 de abril). Framework de código abierto para crear sitios web y aplicaciones web responsive. <https://getbootstrap.com/>

**Microsoft Corporation.** (2023, 10 de abril). Microsoft 365 - Microsoft Office. <https://www.microsoft.com/en-us/microsoft-365/microsoft-office>

**Pallets Projects.** (2023, 10 de abril). Flask: Microframework web de Python ligero y flexible. <https://flask.palletsprojects.com/>

**Microsoft Corporation.** (2023, 13 de noviembre). Visual Studio Code Copilot: Un compañero de IA más inteligente. Blogs de Visual Studio Code. <https://code.visualstudio.com/blogs/2023/11/13/vscode-copilot-smarter>