

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 DE FACTURACION ELECTRONICA PARA LA TIENDA DE ROPA “MODA MENS PLUS” UBICADO EN EL CANTON SANTA ROSA.

**AUTOR:**

RONALDO JOSE PRIETO GONZALEZ

**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 DE FACTURACION ELECTRONICA PARA LA TIENDA DE “ROPA MODA MENS PLUS” UBICADO EN EL CANTON SANTA ROSA,** realizado por Sr. **RONALDO JOSE PRIETO GONZALEZ** 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**

RONALDO JOSE PRIETO GONZÁLEZ

DECLARO QUE:

EL proyecto de grado denominado: **ANALISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE FACTURACION ELECTRONICA PARA LA TIENDA DE ROPA “MODA MENS PLUS” UBICADO EN EL CANTON SANTA ROSA**, 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

***Ronaldo Jose Prieto González***

**CERTIFICADO**

ANDREA CARTUCHE JARA

Que el trabajo titulado **ANALISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE FACTURACION ELECTRONICA PARA LA TIENDA DE ROPA “MODA MENS PLUS” UBICADO EN EL CANTON SANTA ROSA*,*** 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**

Esta tesis está dedicada a mi señora madre Patricia Guadalupe González Ullo y a mi señor padre José Lorenzo Prieto, quienes me han enseñado elegir el mejor camino en la vida, lo cual me ha dejado un gran legado para poder ser un gran ejemplo en el camino, quienes se han convertido en el motor de mi vida y han estado en los mejores como también en las peores etapas de la vida.

***Ronaldo José Prieto González***

**AGRADECIMIENTO**

A mis compañeros de estudio quienes siempre supieron demostrar una gran amistad desde el inicio de nuestras metas.

A mi tutora, que no solo fue una profesional guiándome durante la obtención de mi título, sino también un gran apoyo al momento de poderme aconsejar en el proceso de aprendizaje para una buena formación profesional.

A mi querido Instituto Superior Tecnológico Particular Bolívar Madero Vargas, por haberme permitido tener la oportunidad de adquirir conocimientos y formarme profesionalmente.

***Ronaldo José Prieto González***

**ÍNDICE GENERAL**

DECLARACIÓN DE RESPONSABILIDAD

CERTIFICACIÓN

DEDICATORIA

AGRADECIMIENTO

ÍNDICE GENERAL

INTRODUCCIÓN

[**1.** **MARCO TEÓRICO** - 3 -](#_Toc193185304)

[**1.1.** **Antecedentes** - 3 -](#_Toc193185305)

[**1.2.** **Justificación** - 6 -](#_Toc193185306)

[**1.3.** **Objetivos** - 7 -](#_Toc193185307)

[**1.3.1.** **Objetivo General** - 7 -](#_Toc193185308)

[**1.3.2.** **Objetivos Específicos** - 7 -](#_Toc193185309)

[**1.4.** **Descripción de la Empresa** - 8 -](#_Toc193185310)

[ **Nombre:** MODA Mens Plus - 8 -](#_Toc193185311)

[ **Propietaria:** Tatiana Lisseth Fernández González - 8 -](#_Toc193185312)

[ **Cantón:** SantaRosa - 8 -](#_Toc193185313)

[ **Celular:** 0939050689 - 8 -](#_Toc193185314)

[**1.4.1.** **Ubicación** - 9 -](#_Toc193185315)

[**1.4.2.** **Misión** - 9 -](#_Toc193185316)

[**1.4.3.** **Visión** - 10 -](#_Toc193185317)

[**1.4.4.** **Orgánico Funcional** - 10 -](#_Toc193185318)

[**1.4.5.** **Especificación de Procesos** - 11 -](#_Toc193185319)

[**1.5.** **Fuente de la Información** - 12 -](#_Toc193185320)

[**1.5.1.** **Entrevista** - 12 -](#_Toc193185321)

[**1.5.2.** **Resultado de la Entrevista** - 16 -](#_Toc193185322)

[**1.4.3.** **Encuesta** - 16 -](#_Toc193185323)

[**1.4.4.** **Análisis de la Encuesta** - 23 -](#_Toc193185324)

[**1.4.5.** **Problemas Encontrados (Causas y Efectos)** - 24 -](#_Toc193185325)

[**1.4.6.** **Alternativas de la Solución con el Sistema Informático a desarrollar.** - 24 -](#_Toc193185326)

[2. **ANÁLISIS DEL SISTEMA** - 25 -](#_Toc193185327)

[**2.1.** **Especificaciones de Requisitos** - 25 -](#_Toc193185328)

[**2.1.1.** **Requisitos Funcionales** - 26 -](#_Toc193185329)

[**2.1.3 Requerimientos de Interfaces Externas** - 29 -](#_Toc193185330)

[**2.1.4 Definiciones, Siglas y Abreviaturas** - 30 -](#_Toc193185331)

[**2.2 Usuarios** - 32 -](#_Toc193185332)

[**2.2.1 Características del Usuario** - 32 -](#_Toc193185333)

[**2.2.2 Roles del Usuario** - 32 -](#_Toc193185334)

[**2.3 Estudio de Factibilidad** - 33 -](#_Toc193185335)

[**2.3.1 Factibilidad Técnica** - 33 -](#_Toc193185336)

[**2.3.2 Factibilidad Económica** - 34 -](#_Toc193185337)

[**2.3.3 Factibilidad Operativa** - 36 -](#_Toc193185338)

[**2.3.4 Factibilidad Legal** - 36 -](#_Toc193185339)

[**3.** **FASES DE DISEÑO** - 40 -](#_Toc193185340)

[**3.1.** **Diseño** - 40 -](#_Toc193185341)

[**3.1.1.** **Diseño arquitectónico UML** - 40 -](#_Toc193185342)

[**3.1.1.1.** **Diagramas de Casos de Uso** - 41 -](#_Toc193185343)

[**3.1.2.** **Diseño de Base de Datos** - 48 -](#_Toc193185344)

[**3.1.2.1.** **Modelo lógico de la Base de Datos** - 48 -](#_Toc193185345)

[**3.1.2.2.** **Modelo físico de la Base de Datos** - 49 -](#_Toc193185346)

[**3.1.2.3.** **Diccionario de datos** - 49 -](#_Toc193185347)

[**3.1.3.** **Interfaz de Usuario** - 52 -](#_Toc193185348)

**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 para “**Moda Mens Plus”**, una tienda de ropa ubicada en el Cantón Santa Rosa, especializada en moda masculina. La tienda enfrenta dificultades para gestionar de manera eficiente sus ventas, inventarios y la emisión de facturas, lo que impacta en la productividad y el control de sus operaciones.

El objetivo principal de este proyecto es desarrollar una solución automatizada que optimice el proceso de facturación, 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 y el control financiero de la tienda, lo que contribuirá al crecimiento y la sostenibilidad del negocio.

**CAPÍTULO I**

**MARCO TEÓRICO**

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

La facturación electrónica en Ecuador ha tenido un proceso de implementación gradual que comenzó hace más de una década y que ha evolucionado a lo largo del tiempo, con el objetivo 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.

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). Esto representaba una gran desventaja 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 era más común.

En este contexto, el SRI comenzó a buscar formas de modernizar la administración tributaria y mejorar el control fiscal, tomando como modelo otras experiencias internacionales de implementación de sistemas de facturación electrónica.

Uno de los primeros pasos importantes para la implementación de la facturación electrónica en Ecuador fue la reforma de la **Ley de Régimen Tributario Interno**, en el año 2011. Esta reforma incluía disposiciones que permitían el uso de la facturación electrónica como una alternativa legal para las empresas, especialmente para aquellas que realizaban transacciones de mayor volumen.

A partir de esta reforma, el SRI empezó a crear el marco legal necesario para facilitar la transición hacia la facturación electrónica, estableciendo una serie de normativas y requisitos técnicos para su implementación.

El proceso de implementación de la facturación electrónica en Ecuador comenzó de manera formal con pruebas piloto en 2014. Durante este periodo, el SRI seleccionó un grupo de empresas grandes para probar el sistema y evaluar su viabilidad. Esto permitió al SRI ajustar los aspectos técnicos y operativos del sistema antes de su implementación a gran escala.

En 2015, el SRI lanzó de manera oficial la obligación de usar facturación electrónica para todas las empresas de ciertos sectores, principalmente aquellas de mayor tamaño o que ya estaban dentro del sistema tributario. A partir de ese año, se estableció un cronograma progresivo de obligatoriedad que abarcaba a más empresas de diferentes sectores.

A medida que la facturación electrónica se fue consolidando en grandes empresas, el SRI comenzó a extender la obligación a las pequeñas y medianas empresas (PYMES). En 2016 y 2017, las autoridades fiscales implementaron un proceso gradual que permitía a las PYMES adaptarse a la nueva normativa.

Este proceso gradual fue importante para mitigar los posibles impactos negativos de la transición y para asegurar que las pequeñas empresas tuvieran tiempo para adaptarse al nuevo sistema. El SRI ofreció capacitaciones y herramientas de apoyo para facilitar la implementación de la facturación electrónica en estos negocios.

Para 2018, el SRI decretó 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 significó que, tanto grandes empresas como pequeños comercios, tuvieron que integrar el sistema de facturación electrónica en sus operaciones diarias.

A partir de este momento, el uso de la facturación electrónica pasó a ser obligatorio para todos los contribuyentes del país, y el SRI continuó implementando mejoras y ajustes técnicos en el sistema, tales como la integración con otros servicios electrónicos y la optimización de la plataforma para facilitar su uso.

A lo largo de los años, el SRI ha continuado perfeccionando la plataforma de facturación electrónica, añadiendo nuevas funcionalidades y mejorando la experiencia de los usuarios. Una de las mejoras clave fue la creación de un sistema de **validación automática de facturas** que permite que las transacciones sean verificadas en tiempo real. Esto no solo mejora la transparencia y reduce la posibilidad de fraude fiscal, sino que también asegura que las facturas emitidas estén alineadas con la normativa tributaria.

El sistema de facturación electrónica del SRI está diseñado para ser accesible y sencillo de usar, incluso para las pequeñas empresas, que pueden optar por un software gratuito provisto por el SRI o adquirir software homologado que se ajuste a sus necesidades. 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.

* 1. **Justificación**

La creciente demanda de soluciones tecnológicas en el ámbito empresarial ha hecho evidente la necesidad de mejorar los procesos administrativos y comerciales de pequeñas y medianas empresas (PYMEs), en especial aquellas ubicadas en zonas rurales o menos urbanizadas. El sector de la moda, específicamente en tiendas de ropa como Moda Mens Plus, se enfrenta a retos importantes relacionados con la gestión de ventas, inventarios, control de cobros y emisión de facturas.

En este contexto, la implementación de un sistema de facturación 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, la tienda podría enfrentar dificultades como errores humanos en los cálculos de ventas, confusión en el inventario, retrasos en los procesos de facturación, incumplimiento de normativas fiscales locales y, en general, una disminución en la competitividad del negocio.

* 1. **Objetivos**

* + 1. **Objetivo General**

Diseñar e implementar un sistema de facturación automatizado para la tienda de ropa Moda Mens Plus ubicada en el Cantón Santa Rosa, con el fin de optimizar los procesos administrativos, mejorar el control de ventas e inventarios, garantizar el cumplimiento de la normativa fiscal vigente y fortalecer la competitividad del negocio en el mercado local.

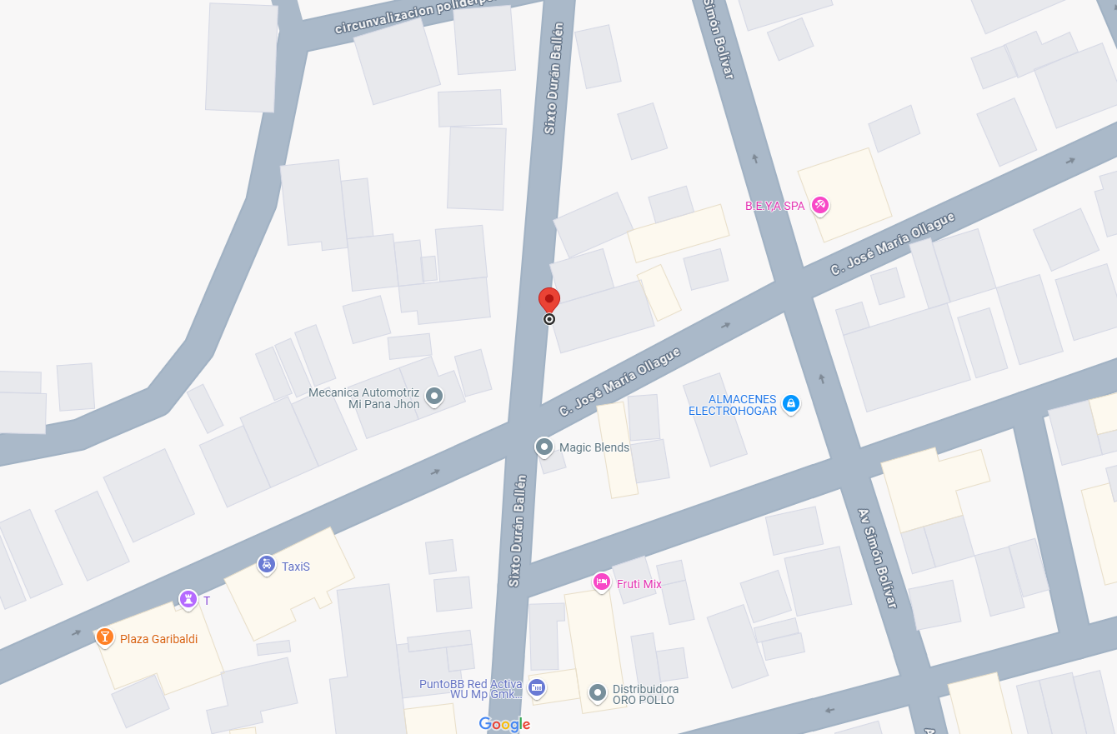
* + 1. **Objetivos Específicos**
* Analizar los procesos actuales de facturación y gestión administrativa de la tienda Moda Mens Plus para identificar áreas de mejora.
* Diseñar un sistema de facturación automatizado adaptado a las necesidades de la tienda.
* Implementar el sistema de facturación, integrándolo con los recursos existentes en la tienda.
* Capacitar al personal en el uso del sistema para garantizar su correcto funcionamiento.
* Evaluar el impacto del sistema en la eficiencia operativa y el cumplimiento de la normativa fiscal.
  1. **Descripción de la Empresa**

Moda Mens Plus es una empresa dedicada a la venta de ropa para hombres, ofreciendo una amplia variedad de prendas diseñadas para satisfacer diferentes estilos y necesidades. Su catálogo incluye desde pantalones cortos, gorras y camisetas, hasta conjuntos casuales y accesorios, asegurando opciones para toda ocasión. Moda Mens Plus se destaca por combinar comodidad, calidad y tendencias, brindando a sus clientes una experiencia de compra única.

* **Nombre:** MODA Mens Plus
* **Propietaria:** Tatiana Lisseth Fernández González
* **Cantón:** SantaRosa
* **Celular:** 0939050689

**Razón Social:** Venta de Ropa para hombre al por menor.

* + 1. **Ubicación**

****

* + 1. **Misión**

Desarrollar e implementar un sistema de facturación integral, eficiente y automatizado para la tienda de ropa "Moda Mens Plus", ubicado en el cantón Santa Rosa, que permita optimizar los procesos de ventas, control de inventarios y generación de documentos fiscales. Este sistema tiene como objetivo mejorar la experiencia de compra de los clientes, incrementar la productividad del personal y garantizar una gestión clara y transparente de los recursos, contribuyendo al crecimiento y competitividad de la tienda en el mercado local.

* + 1. **Visión**

Ser un referente en la implementación de soluciones tecnológicas en el sector comercial de la moda, 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 "Moda Mens Plus" como una tienda líder en eficiencia y servicio al cliente en el cantón Santa Rosa, asegurando una gestión empresarial más sostenible y rentable en el futuro.

Estas declaraciones reflejan el enfoque en mejorar la operatividad de la tienda mediante un sistema moderno y eficiente, al mismo tiempo que posicionan la tienda como un actor destacado en la localidad.

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

El sistema de facturación debe incluir un módulo que permita registrar productos y clientes de manera automatizada. También debe gestionar los inventarios, actualizándolos automáticamente al realizar una venta y generando órdenes de compra para reponer artículos cuando sea necesario. Además, el sistema debe emitir facturas electrónicas con toda la información necesaria, registrar todas las transacciones y enviar automáticamente las facturas a los clientes. Por otro lado, debe incluir reportes para analizar ventas, identificar necesidades de reposición y consolidar toda la información, ayudando así en la toma de decisiones. El sistema debe ser intuitivo, accesible desde diversos dispositivos y contar con medidas de seguridad, como copias de seguridad automáticas.

Pedido de Productos: En este proceso, las consultoras pueden hacer pedidos de productos. Cada pedido incluirá los detalles del producto y el descuento aplicable. Los pedidos se ingresarán en el sistema a través de una interfaz de usuario y se almacenarán en la base de datos MongoDB.

Ingreso de códigos: Este proceso implica la generación de códigos de cada producto con su respectivo descuento y cantidad el cual entregara la suma de cada uno de estos productos.

Cobranzas: En las fechas programadas, el sistema generará automáticamente las facturas correspondientes para cada consultora y las enviará para su cobro.

El sistema será desarrollado utilizando Python con Flask para el backend, y Visual Studio Code será utilizado como el entorno de desarrollo. La estructura de carpetas en Visual Studio Code permitirá un manejo y navegación eficientes del código.

Es importante destacar que cada uno de estos procesos debe ser diseñado y desarrollado con cuidado para garantizar que el sistema sea fácil de usar para la directora y que los datos se manejen de manera segura y eficiente.

* 1. **Fuente de la Información**

Desde el punto de vista del desarrollo de esta investigación, es de suma importancia revisar textos relacionados con este tema, incluyendo revistas, manuales, folletos y sitios web, todas estas investigaciones son muy importantes para obtener información suficiente, la misma que apoye el proceso a lo largo del proyecto.

La observación directa es un método a utilizar en el presente trabajo ya que permite la recolección de datos sobre un individuo, fenómeno o situación particular. Se caracteriza porque el investigador se encuentra en el lugar en el que se desarrolla el hecho sin intervenir ni alterar el ambiente, ya que de lo contrario los datos obtenidos no serían válidos.

Este método de recolección de datos, denominado también investigación primaria, se emplea en ocasiones en las que otros sistemas (como encuestas, cuestionarios, entre otros) no son efectivos.

* + 1. **Entrevista**

Para el desarrollo de la tesis sobre el sistema de facturación para la tienda "Moda Mens Plus", se realizaron entrevistas con el propietario de la tienda. El objetivo fue entender los procesos actuales de facturación, identificar problemas y conocer las expectativas de mejora.

El propietario proporcionó información sobre las necesidades del negocio, mientras que los empleados compartieron sus experiencias sobre los inconvenientes en la gestión de ventas e inventarios. Los clientes fueron consultados para conocer su nivel de satisfacción con el proceso de compra. Además, se entrevistaron proveedores y expertos en software para evaluar opciones tecnológicas que se adapten a las necesidades de la tienda.

Esta información fue clave para el análisis y diseño del sistema de facturación propuesto.

**Desarrollo de la Entrevista:**

**Nombre:** Sr. Ronaldo Prieto

**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 la tienda.

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

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

Por el momento usamos un proceso manual.

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

Hay errores frecuentes.

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

Hay dificultades para mantener un control preciso del inventario durante el proceso de ventas.

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

Creo que un sistema automatizado podría mejorar la eficiencia y la satisfacción del cliente.

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 el nuevo sistema mejore la eficiencia operativa, reduzca los errores en la facturación y facilite el control de inventario. También espero que se traduzca en una mejor experiencia para nuestros clientes, ya que podríamos ser más rápidos en el proceso de compra.

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, siempre y cuando se justifique con los beneficios a largo plazo. La idea es que el sistema no solo sea eficiente, sino que también nos ayude a ahorrar tiempo y dinero, por lo que un sistema más completo podría ser una buena opción.

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 sin duda mejoraría la relación con los clientes, ya que las compras serían más rápidas y precisas, lo que llevaría a una mayor satisfacción. Además, nos permitiría ofrecer un mejor servicio, lo que nos haría más competitivos frente a otras tiendas de la zona

1. **¿Qué tan importante es para usted la capacitación del personal en el uso de este nuevo sistema?**Es fundamental. Aunque quiero un sistema fácil de usar, sé que la capacitación adecuada es clave para aprovechar todas las funcionalidades. Si los empleados no se sienten cómodos con el nuevo sistema, podría afectar la eficiencia. Por lo tanto, me gustaría que se proporcionara una formación adecuada antes de implementar el sistema.

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

La entrevista con el propietario de "Moda Mens Plus" permitió identificar varios problemas en el proceso de facturación actual, que se realiza manualmente y genera errores y retrasos. El propietario destacó la necesidad de automatizar la facturación y vincularla con el control de inventarios para mejorar la precisión y eficiencia.

Se expresó que el sistema ideal debe ser fácil de usar, rápido y generar reportes automáticos de ventas e inventarios. También se mencionó la importancia de capacitar al personal en el uso del nuevo sistema. El propietario está dispuesto a invertir en una solución avanzada si los beneficios, como la reducción de errores y el ahorro de tiempo, son evidentes.

* + 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 "Moda Mens Plus"?**

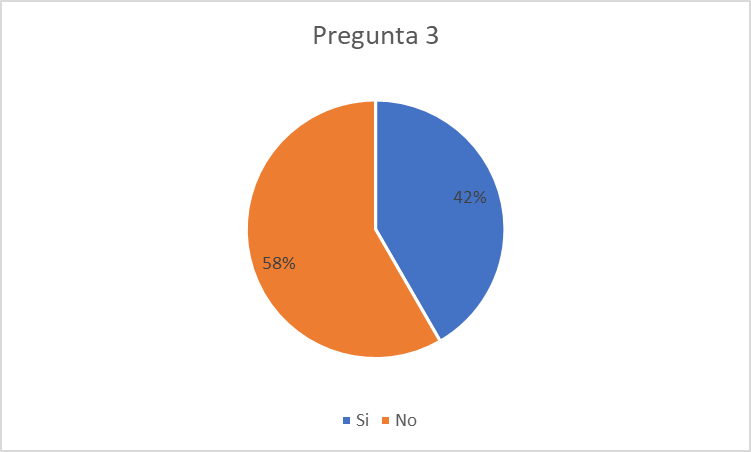
|  |  |  |
| --- | --- | --- |
| Numero de muestra: 12 | | |
| RESPUESTA | **FRECUENCIA** | **PORCENTAJE** |
| PROPIETARIO | 1 | 8.33% |
| EMPLEADO ADMINISTRATIVO | 2 | 16.67% |
| EMPLEADO DE VENTAS | 5 | 41.67% |
| TOTAL | **12** | **100%** |

**Pregunta 2. ¿ Actualmente existe un control automatizado de ventas e inventarios en la tienda "Moda Mens Plus"?**

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

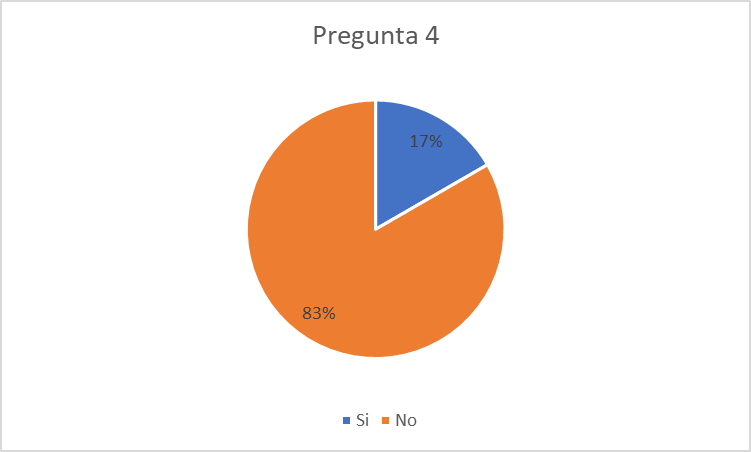
**Pregunta 3. ¿Considera que el sistema de facturación actual de la tienda "Moda Mens Plus" es ágil y eficiente?**

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

****

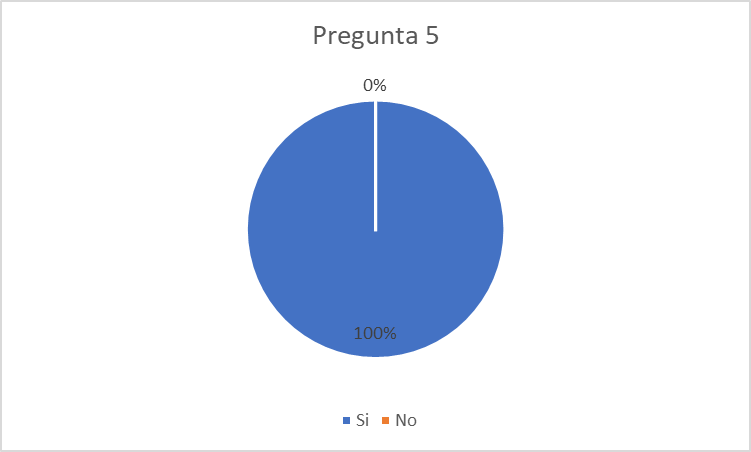
**Pregunta 4. ¿En el sistema de facturación actual de la tienda "Moda Mens Plus" es posible consultar y verificar los registros de ventas e inventarios?**

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

****

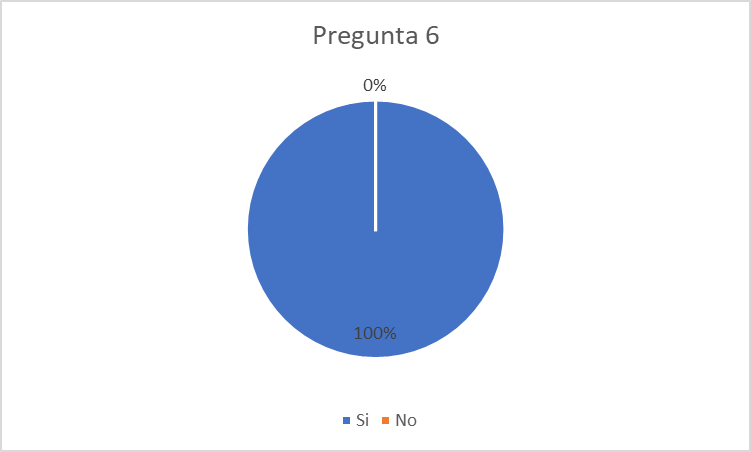
**Pregunta 5. ¿Cree usted que es importante utilizar nuevas herramientas para automatizar el proceso de facturación y control de inventarios en la tienda "Moda Mens Plus"?**

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

****

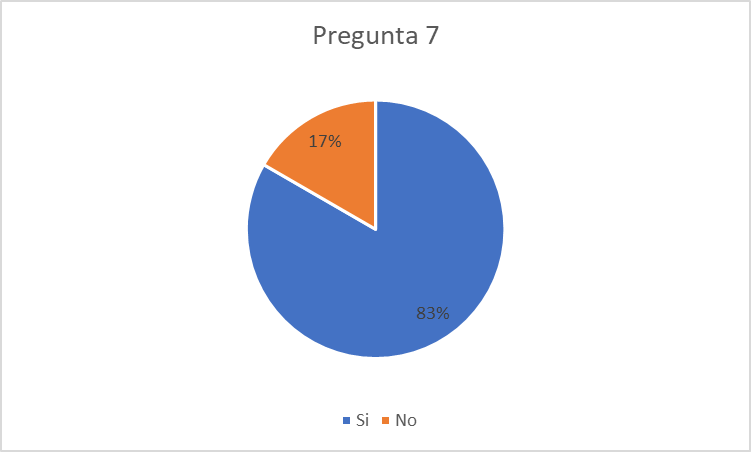
**Pregunta 6. ¿Considera usted que la implementación de un sistema de facturación automatizado que controle los procesos de ventas e inventarios ayudará a que la gestión de la información sea más eficiente en la tienda "Moda Mens Plus"?**

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

****

**Pregunta 7. ¿Considera usted que la implementación de un sistema automatizado de facturación y control de inventarios le permitirá agilizar los procesos operativos y administrativos en la tienda "Moda Mens Plus"?**

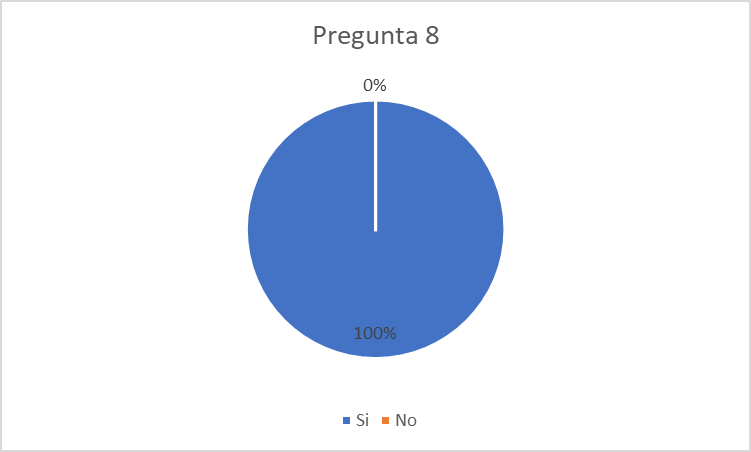
|  |  |  |
| --- | --- | --- |
| Numero de muestra: 12 | | |
| RESPUESTA | **FRECUENCIA** | **PORCENTAJE** |
| SI | 10 | 83.33% |
| NO | 2 | 16.67% |
| TOTAL | **12** | **100%** |

****

**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**

Gran parte de los encuestados piensan que es necesario la implementación de un sistema de facturación que automatice el registro de las facturas y control de los productos. Debido a que esto permitirá mantener la información segura y disponible.

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

|  |  |  |
| --- | --- | --- |
| **PROBLEMA** | **CAUSA** | **EFECTO** |
| El registro de forma manual es muy complicado y tiende a errores. | Al ser un proceso manual tiende que se puedan colocar de manera equivocada valores. | Malestar e incomodidad por parte de los clientes. |
| No se puede medir los valores que salen de la tienda. | Al ser algo manual si uno quiere visualizar ciertas cosas o utilizar estadística se vuelve muy complicado e ineficiente. | Inseguridad y posible pérdida de información. |

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

|  |  |  |
| --- | --- | --- |
| **ALTERNATIVA I** | | |
| **DETALLE** | **VENTAJAS** | **DESVENTAJAS** |
| Implementar un sistema que mantenga la información actualizada y que pueda entregar facturas de forma inmediata al cliente. | Los clientes obtendrán de forma rápida sus facturas y sin necesidad de procesos repetitivos. | Se necesitará tiempo para entender los recursos necesarios y que se dé la capacitación al personal indicado |

**Tabla 2: Alternativa de Solución**

|  |  |  |
| --- | --- | --- |
| **ALTERNATIVA II** | | |
| **DETALLE** | **VENTAJAS** | **DESVENTAJAS** |
| Implementar un sistema que mantenga la información segura en casos de desastres o por daños en el equipo. | La información registrada estará más segura ya que al implementarla se podrá tener respaldos y conservar información importante. | Los archivos enviados pueden estar guardados con un tipo de extensión diferente y se debe de contener una forma eficiente de como guardar esas extensiones. |

Con la implementación de este sistema de facturación brindará a la compañía la facilidad de registro y acceso a la información ingresada al sistema y contribuirá en que el cliente pueda obtener los datos de una forma más rápida y eficiente con todo esto la empresa MODA MENS PLUS podrá a contribuir a la ciudad de Santa Rosa con avances en el campo de la tecnología.

**CAPÍTULO II**

**ANÁLISIS DEL SISTEMA**

1. **ANÁLISIS DEL SISTEMA**

Es el componente esencial en el desarrollo del sistema de facturación, permitiendo la recopilación de diversos datos que ayudarán a la correcta construcción del software de acuerdo a las necesidades de la organización, proporcionando una solución a largo plazo. Este proceso permite recopilar toda la información necesaria para garantizar la construcción adecuada del software.

* 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 “**Moda Mens Plus**. 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:** Se refiere a los roles de los usuarios para el ingresado de la plataforma:

* Administrador
* Usuarios

**Gestión de productos y servicios:**

• Crear, editar, eliminar y listar productos y servicios ofrecidos por “**Moda Mens Plus**”.

**Gestión de facturación:**

• Emisión, anulación y consulta de facturas electrónicas.

• Integración con el sistema del SRI para validar los documentos emitidos.

**Reportes:**

• Reporte de ventas mensuales.

**2.1.2. Requisitos No Funcionales**

**• Rendimiento:** El sistema deberá procesar la información rápidamente para no interrumpir las operaciones del negocio.

**• Seguridad:** Uso de protocolos de seguridad para proteger la información de clientes y transacciones.

**• Usabilidad:** Interfaces amigables e intuitivas para facilitar su uso por personal no técnico.

**• Disponibilidad:** Acceso al sistema en horarios comerciales y soporte técnico en caso de inconvenientes.

**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 para garantizar su funcionamiento:

• 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**

Se requiere una inversión inicial para el desarrollo del sistema, que incluye:

• Hardware: Computadora y periféricos existentes en “**Moda Mens Plus”**.

• Software: Licencias necesarias para herramientas de documentación.

**Factibilidad Operativa**

El personal de “**Moda Mens Plus”** será capacitado para usar el sistema de manera eficiente. Se implementarán manuales de usuario y soporte técnico para garantizar la operatividad continua

**MÓDULOS DEL SISTEMA**

**MÓDULO DE SEGURIDAD**

**Objetivos:**

* **Gestionar Clientes**

**MÓDULO DE PRODUCTOS**

**Objetivos:**

**• Gestionar el catálogo de productos.**

**Beneficios:**

**• Permite registrar, actualizar y eliminar productos del inventario.**

**• Mantiene actualizado el stock de productos en tiempo real.**

**MÓDULO DE FACTURACIÓN ELECTRÓNICA**

**Objetivos:**

**• Gestionar la emisión de facturas electrónicas.**

**Beneficios:**

**• Cumple con los requerimientos del SRI para validación y emisión de documentos tributarios.**

**• Automatiza el proceso de facturación, reduciendo errores.**

**MÓDULO DE REPORTES**

**Objetivos:**

**• Generar reportes financieros y de inventario.**

**Beneficios:**

**• Proporciona análisis de ventas y tendencias del mercado.**

**• Mejora la toma de decisiones mediante informes detallados y visualizaciones.**

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

Los requerimientos de interfaces externas son aquellos que afectan el desarrollo del software en cuanto a cómo interactúan los usuarios con el sistema, los dispositivos físicos necesarios y el entorno operativo adecuado.

**• Interfaces de Usuario:** Se desarrollarán interfaces amigables e intuitivas adaptadas a las necesidades de “**Moda Mens Plus**” para garantizar una experiencia sencilla y eficiente. Las interfaces se diseñarán con tecnologías web modernas como HTML, CSS y JavaScript, asegurando un diseño responsivo y funcional.

**• Interfaces de Software:** El sistema se desarrollará para ser compatible con Windows 10 en adelante, utilizando una base de datos como MongoDB, y el lenguaje de programación Python con el framework Flask.

**• 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** | Máxima autoridad del local **Moda Mens Plus**, responsable de supervisar la gestión financiera y operativa del negocio. |
| **Asistente** | Persona encargada de registrar ventas, emitir facturas electrónicas, realizar reportes financieros y mantener actualizada la documentación del sistema |
| **Cajero** | Empleado encargado de procesar las ventas, emitir comprobantes de pago, registrar los métodos de pago y asegurar la correcta atención al cliente |
| **Cliente** | Persona que adquiere productos o servicios del local “**Moda Mens Plus”**, generándose un registro de sus compras en el sistema de facturación |

#### **2.1.4.2 Del Sistema**

|  |  |
| --- | --- |
| **Definiciones** | **Significado** |
| **CRUD** | Acción de crear, listar, modificar y eliminar la información del sistema. |
| **Búsqueda** | Proceso por el cual buscamos información en la base de datos y se presenta en el sistema. |
| **Usuario** | Persona autorizada para ingresar al sistema de facturación mediante su usuario y contraseña. Puede incluir roles como administradores o cajeros. |
| **Permiso** | Las acciones específicas que cada usuario puede realizar en el sistema, como gestionar productos, emitir facturas o generar reportes. Los permisos están vinculados al rol asignado. |
| **Rol** | Conjunto de permisos y responsabilidades asignados a un usuario, como administrador o cajero. |
| **Administrador** | Es la máxima autoridad dentro del sistema de facturación. Tiene acceso completo a todas las funcionalidades, incluyendo la configuración del sistema, gestión de usuarios y generación de reportes globales. |

#### **2.1.4.3 De la Tecnología**

|  |  |
| --- | --- |
| **Abreviaturas** | **Significado** |
| **Motor de Base de Datos** | Es un software que sirve para diseñar nuestra base de datos y establecer conexión con nuestro sistema para que se vaya almacenando la información. |
| **Base de Datos** | Es un conjunto de diferentes tipos de datos el cual almacenará información. |
| **Software** | Sistema web diseñado específicamente para optimizar los procesos de facturación e inventario en el local “**Moda Mens Plus”**, agilizando la emisión de facturas y la gestión administrativa. |
| **MONGODB** | MongoDB es un sistema de gestión de base de datos No SQL, lo que significa que se diferencia significativamente de un sistema de gestión de base de datos relacional (RDBMS) en varios aspectos clave. |
| **Visual Studio Code** | Visual Studio Code, a menudo conocido como VS Code, es un entorno de desarrollo integrado (IDE) ligero pero potente, desarrollado por Microsoft. Aunque es más ligero que un IDE tradicional, VS Code ofrece una serie de características que facilitan la creación de aplicaciones de escritorio y web. |

## 

## **2.2 Usuarios**

### **2.2.1 Características del Usuario**

El usuario del sistema es la persona encargada de interactuar con el software de facturación, llevando a cabo acciones específicas según su rol y obteniendo resultados que faciliten su trabajo. Cada usuario puede realizar tareas según su nivel de acceso, lo que le permite satisfacer sus necesidades dentro del sistema.

### **2.2.2 Roles del Usuario**

Cada usuario tendrá un rol específico dentro del sistema, lo que le permitirá realizar sus tareas sin interferir con el trabajo de otros usuarios. El Administrador: Es el único rol con acceso total al sistema, incluyendo la capacidad de recuperar datos eliminados, por lo que también brindará soporte en caso de que se requiera asistencia técnica.

## **2.3 Estudio de Factibilidad**

El análisis de viabilidad examina las razones y los propósitos para los cuales se contempla la implementación del sistema, al mismo tiempo que determina si el proyecto puede llevarse a cabo a través de diversas evaluaciones. Esto permite llegar a una conclusión sobre si la propuesta que tenemos es viable o no. Por lo tanto, para la ejecución de este proyecto, se han tenido en cuenta los análisis de viabilidad técnica, operacional, financiera y jurídica, para señalar la conveniencia de realizar este proyecto.

### **2.3.1 Factibilidad Técnica**

Se analiza si los equipos, software y recursos técnicos necesarios están disponibles para el desarrollo del sistema de facturación web. Además, se evalúa si el hardware y el software tienen la capacidad suficiente para ejecutar las funciones y alternativas de diseño que se consideren dentro del sistema, asegurando que el proyecto pueda ser implementado correctamente.

**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**

Los siguientes datos son obtenidos gracias a la investigación de la situación actual de los recursos humanos en el desarrollo de software, lo que ha permitido sacar conclusiones de cuánto gana un programador, considerando la situación de estudiante, tiempo de autoaprendizaje, entre otros. A su vez los componentes hardware y software podemos tener como referencia los precios en internet o algún proveedor.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 tecnólogo del mañana que lidera el desarrollo de este sistema posee la pericia requerida para la generación de innovadoras aplicaciones de escritorio, manteniéndose en un aprendizaje constante de tecnologías emergentes e identificación de herramientas. Existen múltiples campos de trabajo y una variedad de lenguajes de programación que facilitan la creación de software, lo que también implica el costo del desarrollador al investigar sobre las herramientas esenciales para el avance del proyecto. Esto finalmente resulta en la superación de la curva de aprendizaje y un manejo eficaz de las tecnologías identificadas.

En conclusión, esto también permite afirmar que el proyecto es viable operativamente. El usuario final colaborará estrechamente con el desarrollador, asegurando que el producto final esté alineado con los requerimientos establecidos y satisfaga sus necesidades. Al mismo tiempo, implementaremos los estándares de accesibilidad y calidad para garantizar una excelente experiencia de usuario.

Esto es especialmente relevante para un sistema de cobranzas, donde la eficiencia y la facilidad de uso son cruciales para el éxito del proyecto.

### **2.3.4 Factibilidad Legal**

Evaluar la viabilidad jurídica tiene como meta evidenciar que el proyecto no infringe ninguna disposición del Código Orgánico Integral Penal en vigor en Ecuador. Por lo tanto, los individuos involucrados en el desarrollo, asesoramiento y usuarios finales del proyecto han consentido las condiciones de preservar la confidencialidad y el manejo ético de la información, garantizando en todo momento la seguridad de la integridad de cada uno de los individuos y los datos comprometidos, por lo que el proyecto se encuentra completamente dentro del marco legal.

**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:

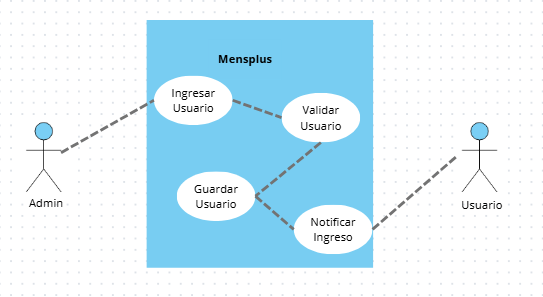
* Actor: Representa a un usuario, dispositivo, sistema o cualquier entidad que interactúe con el sistema.
* Caso de uso: Describe una funcionalidad específica del sistema que se lleva a cabo en respuesta a una solicitud de un actor.
* Relaciones de inclusión: Describen cómo un caso de uso puede incluir 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.

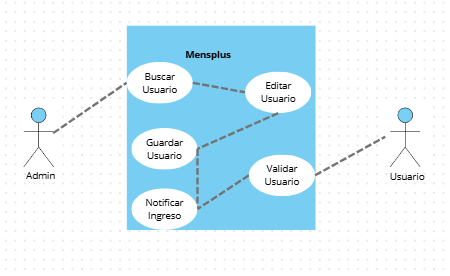
**Descripción de Actores**

|  |  |  |
| --- | --- | --- |
| **ACTOR** | **DESCRIPCIÓN** | **PERFIL** |
| Administrador | El administrador es el único que tiene el control total del sistema puede crear asistentes y visualizar los reportes de las asistencias. | Es una persona con conocimiento en el uso total del sistema. |
| Usuario | Tiene ciertos privilegios como el administrador, pero diferentes ya que tienes sus limitaciones. | Podrá ingresar clientes los productos y visualizar clientes. |

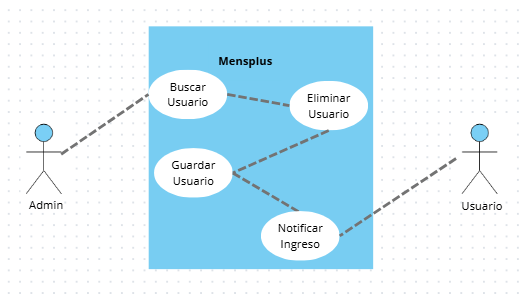
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO I – Gestionar Usuario** | | |
| **Descripción** | Permite al administrador crear usuarios del sistema. | |
| **Actores** | Administrador, Usuarios | |
| **Precondiciones** | El administrador previamente debe estar registrado en el sistema para el ingreso, el usuario debe estar previamente ingresado para la eliminación y edición. | |
| **Flujo Principal** | Pasos | Acción |
| ESCENARIO 1: Crear Usuario | |
| 1 | Ingresar a usuario. |
| 2 | Validar usuario existente |
| 3 | Guardar usuario |
| 4 | Notificar ingreso |
| **Excepciones** |  |  |
|  | 1 | Cedula incorrecta |
|  | 2 | Campos en blanco. |



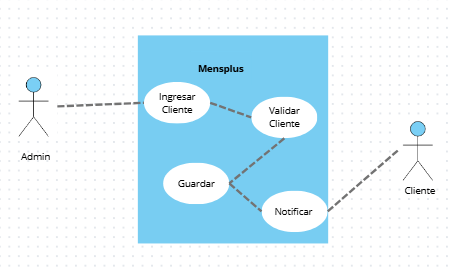
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO II – Gestionar Usuario** | | |
| **Descripción** | Permite al administrador modificar usuarios del sistema. | |
| **Actores** | Administrador, Usuarios | |
| **Precondiciones** | El administrador previamente debe estar registrado en el sistema para el ingreso,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



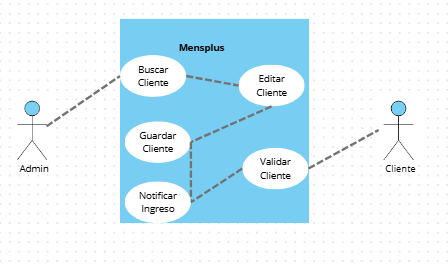
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO III – Gestionar Usuario** | | |
| **Descripción** | Permite al administrador eliminar usuarios del sistema. | |
| **Actores** | Administrador, Usuarios | |
| **Precondiciones** | El administrador previamente debe estar registrado en el sistema para el ingreso, 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. |



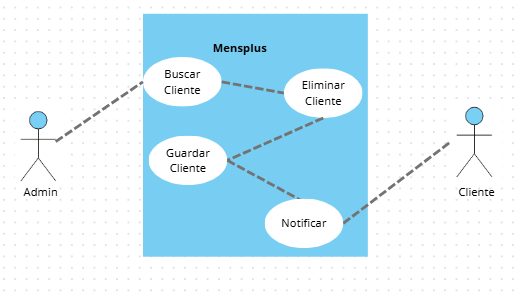
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO IV – Gestionar Cliente** | | |
| **Descripción** | Permite al administrador crear clientes del sistema. | |
| **Actores** | Administrador, cliente | |
| **Precondiciones** | El administrador previamente debe estar registrado en el sistema. | |
| **Flujo 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. |



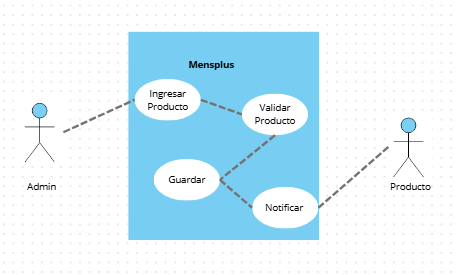
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO V – Gestionar Cliente** | | |
| **Descripción** | Permite al administrador modificar clientes del sistema. | |
| **Actores** | Administrador y cliente | |
| **Precondiciones** | El administrador y cliente previamente debe estar registrado en el sistema. | |
| **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. |



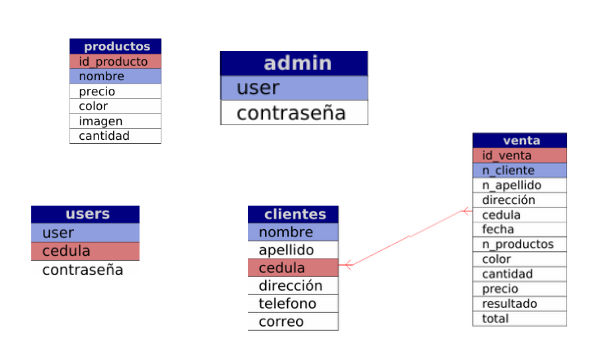
|  |  |  |
| --- | --- | --- |
| **CASOS DE USO IX – Gestionar Ventas** | | |
| **Descripción** | Permite al administrador guardar venta. | |
| **Actores** | Administrador y usuario | |
| **Precondiciones** | El administrador y usuario previamente debe estar registrado en el sistema. | |
| **Flujo Principal**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

* + 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.

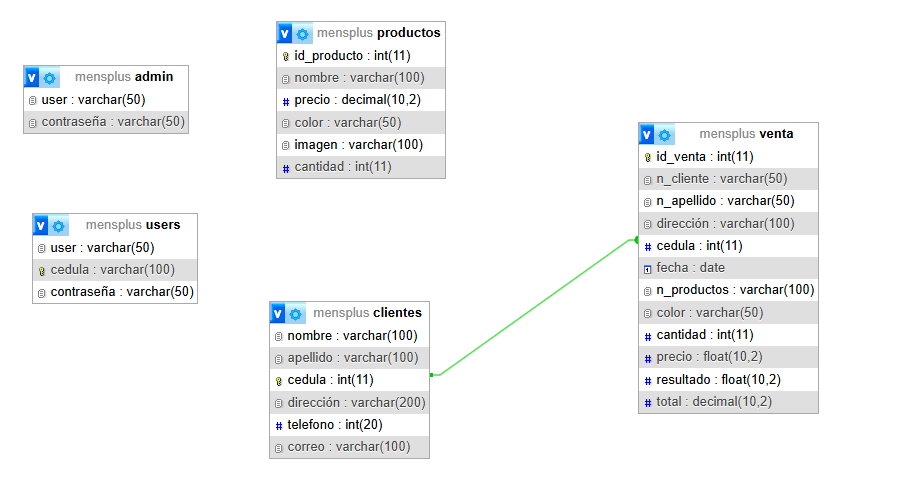
Este modelo se enfoca en la estructura lógica de la base de datos, es decir, en cómo se organiza la información y cómo se relacionan los diferentes elementos que la componen. El objetivo del modelo lógico es definir la estructura de la base de datos de una manera que permita a los usuarios interactuar 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 Modamesplus | | | | | 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 Modamesplus | | | | | 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 Modamesplus | | | | | Esta entidad hace referencia a la tabla productos | | |
| **Entidad:** productos | | **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** | Color | | Varchar(50) | Color | | Si |  |
| **05** | Imagen | | Varchar(100) | Imagen | | Si |  |
| **06** | Cantidad | | Int(11) | Cantidad | | Si |  |

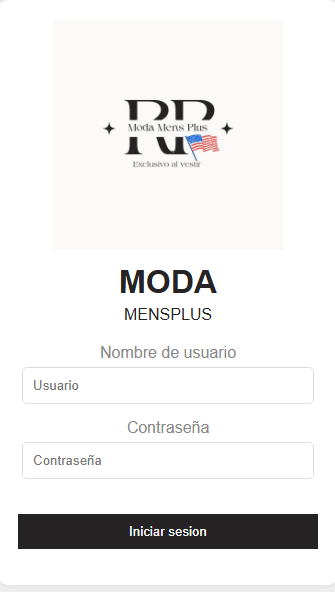
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SISTEMA:** Desarrollo de un sistema de facturación para Modamesplus | | | | | Esta entidad hace referencia a user | | |
| **Entidad:** User | | **LLAVE:** | | |
| **#** | **Campo** | | **Tipo** | **Descripción** | | **Restricción** | |
| **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 Modamesplus | | | | | 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** | Color | Varchar(50) | Color | 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.

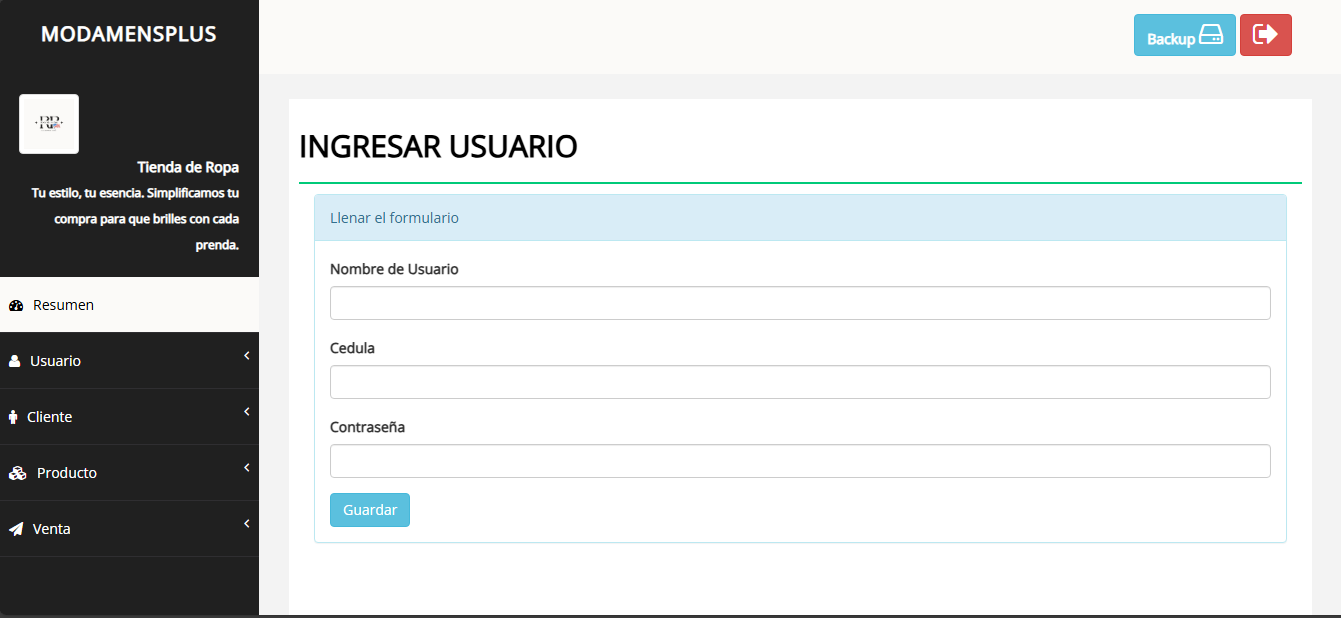


**Figura 1: Login del sistema MENSPLUS para ingreso a facturar**

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**



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

1. **PROGRAMACIÓN**
   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.
  + 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.

* + 1. **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.

* + 1. **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
  1. **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**

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'))

    if request.method == 'POST':

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

        producto = db["producto"]

        nombre = request.form['nombre']

        precio = request.form['precio']

        color = request.form['color']

        cantidad = request.form['cantidad']

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

        if exist\_nombre\_color:

            flash("El nombre del producto y el color 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, precio, color, 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')

# 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"]

        color = request.form["color"]

        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, color, cantidad]

        try:

            if all(campos):

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

                    "id\_producto": id\_producto,

                    "nombre": nombre,

                    "precio": precio,

                    "color": color,

                    "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)

# Este es para agregar el resumen de los productos

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

def resumen():

    if 'username' not in session:

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

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

    producto = db["producto"].find()

    venta = db ["venta"].find()

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

**User**

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

def aduser():

    # Verifica si el usuario está en la sesión

    if 'username' not in session:

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

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

    if request.method == 'POST':

        usere = db["user"]

        use = request.form['user']

        cedula = request.form['cedula']

        contraseña = request.form['contraseña']

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

        exist\_use = usere.find\_one ({"user":use})

        exist\_contraseña = usere.find\_one ({"contraseña":contraseña})

        if exist\_cedula:

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

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

        elif exist\_use:

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

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

        elif exist\_contraseña:

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

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

        else:

            useri = User(use,cedula,contraseña)

            usere.insert\_one(useri.UserDBCollection())

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

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

    else:

        return render\_template('admin/in\_user.html')

# Editar usuario

@user.route('/edit\_us/<string:edaduser>', methods=['GET', 'POST'])#

def edit\_user(edaduser):

    use = db["user"]

    user = request.form["user"]

    cedula = request.form["cedula"]

    contraseña = request.form["contraseña"]

    if user and cedula and contraseña:

        use.update\_one ({'cedula' : edaduser},{"$set" :{"user": user , "cedula":cedula, "contraseña":contraseña}})

        flash("Usuario " + user + " con cedula " + cedula + " Actualizado correctamente")

        return redirect(url\_for('user.v\_user'))

    else:

        return render\_template("admin/user.html")

# Eliminar usuario

@user.route('/delete\_user/<string:eliaduser>')

def delete\_user(eliaduser):

    user = db["user"]

    user.delete\_one({"cedula":eliaduser})

    flash("Usuario eliminado correctamente " )

    return redirect(url\_for('user.v\_user'))

# Visualizar usuario

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

def v\_user():

    if "username" not in session:

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

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

    user = db["user"].find()

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

**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')

@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()

    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"]

        # Recoger los productos

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

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

        colores = request.form.getlist("color")

        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("Colores:", colores)

        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 '',

                "color": colores[i] if i < len(colores) 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,

            "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"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)

# 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, "MODAMENSPLUS")

    # 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(200, height - 240, "Color")

    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(200, y, producto['color'])

        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, "MODAMENSPLUS")

    # 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(200, height - 240, "Color")

    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(200, y, producto['color'])

        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,\nMODAMENSPLUS"

        )

        # 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))

* 1. **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,

        }

**PRODUCTO**

class Producto:

    def \_\_init\_\_(self,id\_producto,nombre,precio,color,imagen,cantidad):

        self.id\_producto=id\_producto

        self.nombre=nombre

        self.precio=precio

        self.color=color

        self.imagen=imagen

        self.cantidad=cantidad

    def ProductoDBCollection(self):

        return{

            "id\_producto":self.id\_producto,

            "nombre":self.nombre,

            "precio":self.precio,

            "color":self.color,

            "imagen":self.imagen,

            "cantidad":self.cantidad,

        }

**USER**

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

* 1. **Conclusiones**

Después de llevar a cabo los análisis pertinentes de los procesos que maneja la microempresa “MENSPLUS”, 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 “MENSPLUS” 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 de la propietaria de la tienda de ropa “MENSPLUS”

* 1. **Recomendaciones**

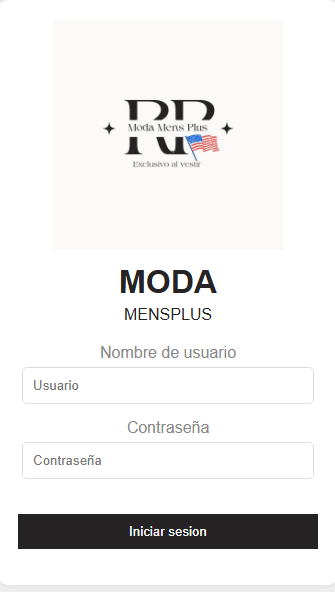
Se sugiere a la Institución, asignar proyectos de menor envergadura a los estudiantes en formación para que vayan ganando experiencia en su realización y estén más preparados al momento de llevar a cabo su tesis de grado.

Para los futuros egresados, recomiendo equiparse con un computador de alta calidad, con características sobresalientes para poder trabajar de manera adecuada en nuestro proyecto, y así evitar la pérdida de tiempo o de información causada por problemas de congelamiento de nuestro computador o gestor de base de datos. Como es de conocimiento general, estos programas requieren muchos recursos para funcionar correctamente.

Al concluir el proyecto, es crucial que evalúes todo lo que se ha logrado, considerando tanto los errores como los aciertos. A través de este análisis, aprenderás qué hacer y qué evitar en proyectos futuros, adquirirás conocimientos sobre los pasos de ejecución y experiencia en el manejo de situaciones futuras.

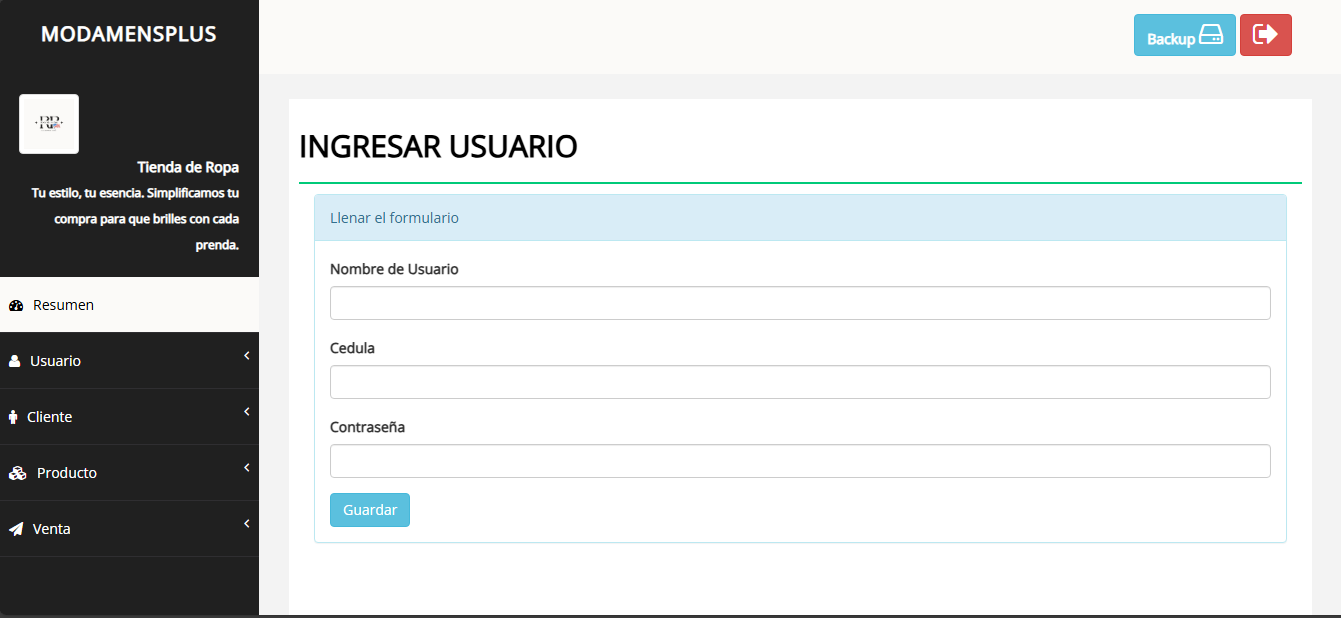
* 1. **Anexos**
     1. **Manual de Usuario**

ACCESO AL SISTEMA



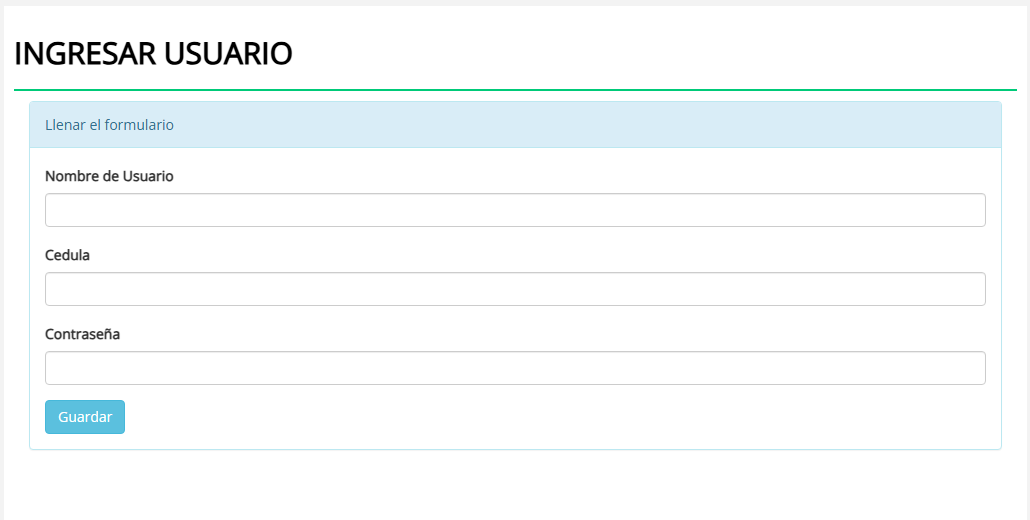
Aquí el usuario debe ingresar el usuario y contraseña otorgadas por el administrador del sistema. Si el usuario o contraseña son incorrectas no se habilita el boton Ingresar, caso contrario nos dara acceso al menú principal.

**MENÚ PRINCIPAL**

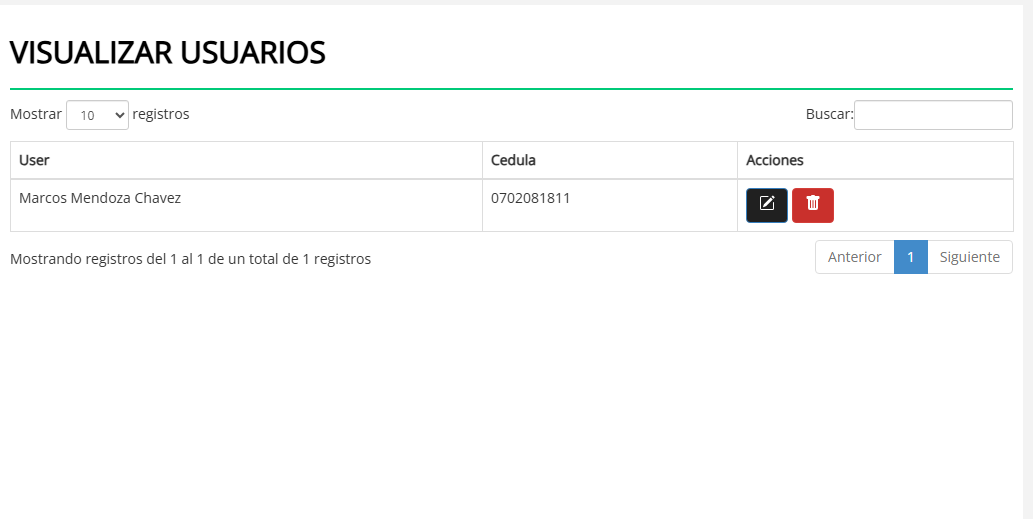
****

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**



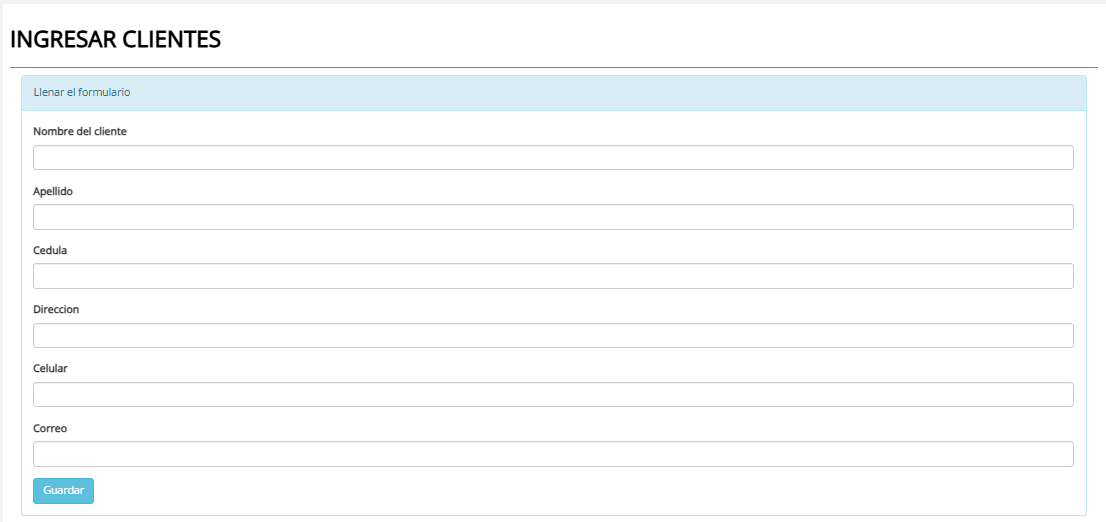
En esta primera ventana es para ingresar los usuarios que podran usar el sistema de acuerdo al rol que tengan.



En esta segunda ventana podra visualizar a los usuarios registrados.

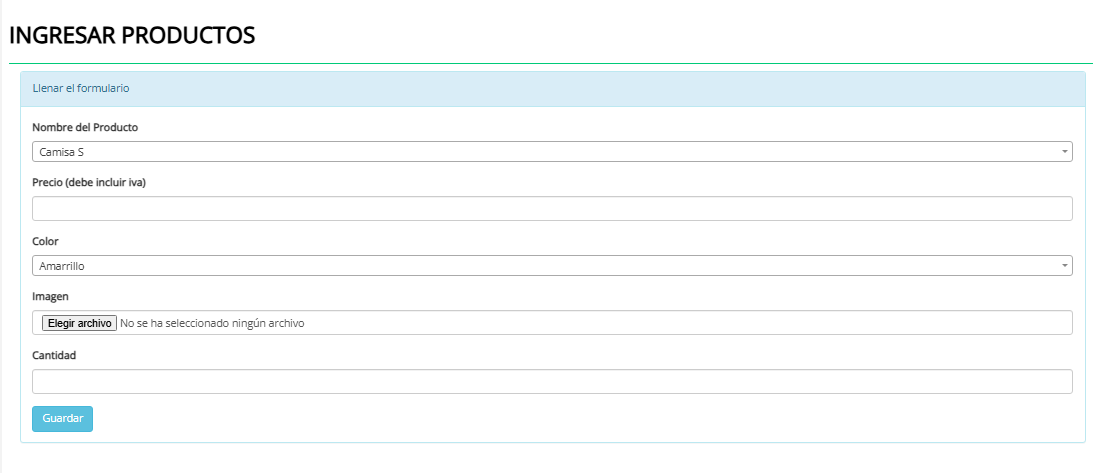
**CLIENTES**

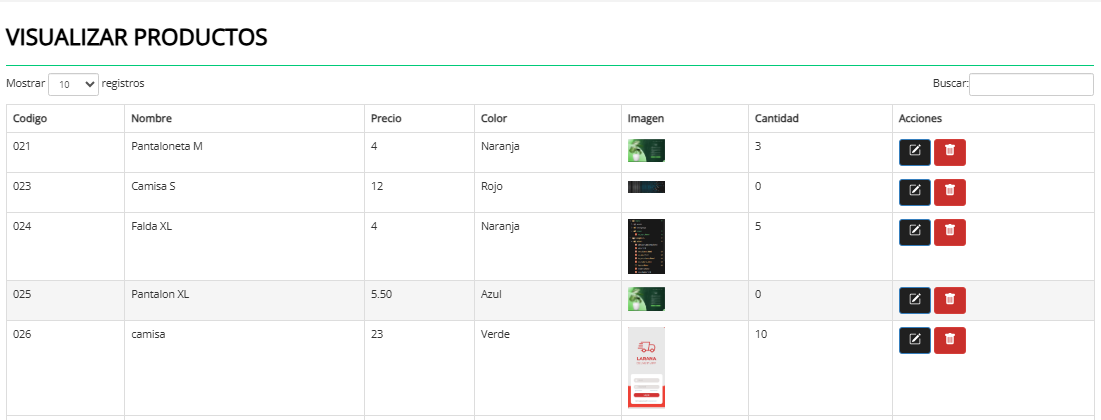
Este formulario podra ingresar a los clientes para poder hacer las ventas es necesario que se pueda ingresar cada uno de los datos ya que en este caso el sistema necesita todos los campos ya que no se puede dejar nada en blanco si lo hace el sistema lo que produce un error.



**INGRESO DE PRODUCTOS**

En este formulario se ingresa los productos que ingresaran en el sistema como es el nombre del producto y para facilitar el proceso se coloco un selector que puede elegir que es lo que esta ingresando como es el caso la tienda hay cosas que se estan repitiendo como es el caso de varios produtos el cual el selector lo facilita ademas de esto es necesatrio el precio que ya debe incluir el IVA y el color de la camisa que va a tener en el otro apartado tiene que ser ingresado lo que es la imagen y mas abajo la cantidad lo cual ayudara para el control de inventario el sistema no permitira las ventas si no hay productos .

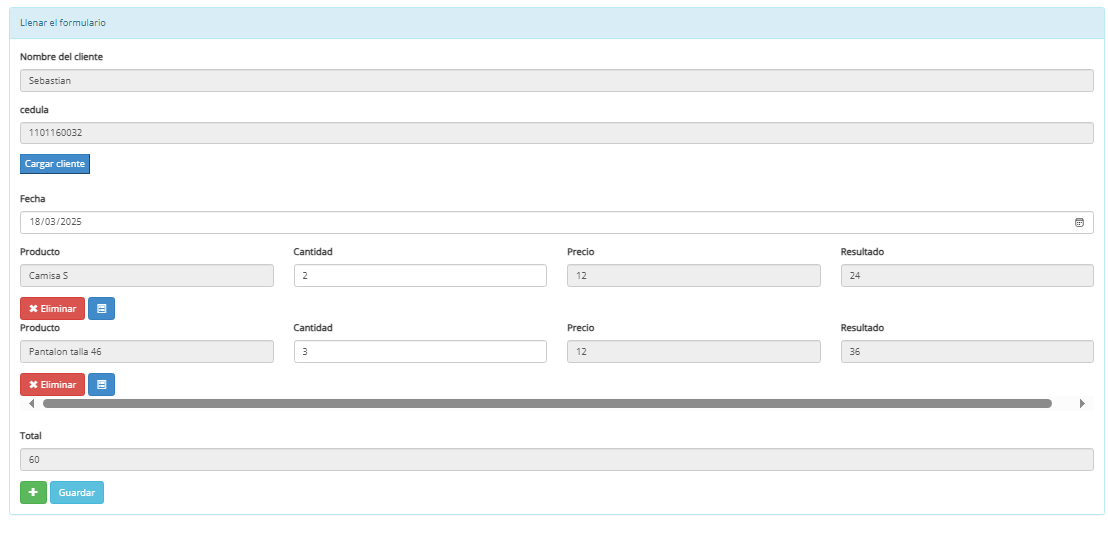


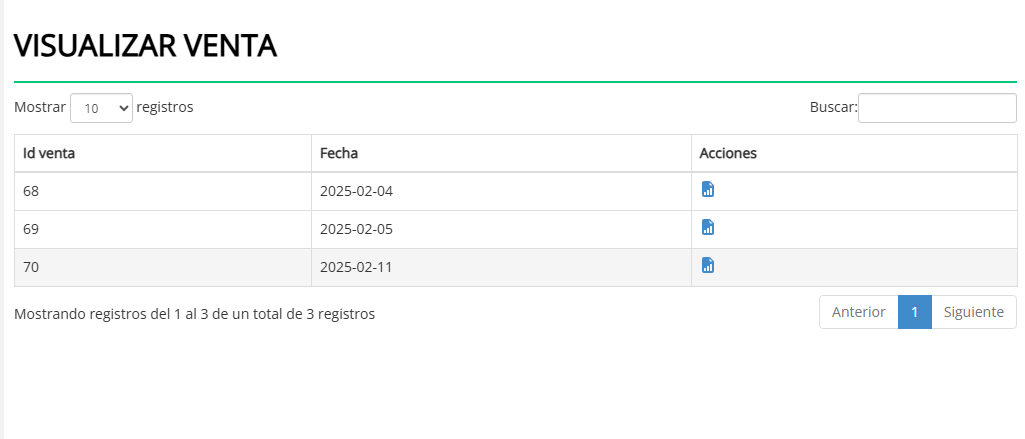


En este apartado se podra ver todos los productos ingresados y que tambien en este lado se podra actualizar todo producto que se ha hecho una venta y se tendra que ingresar las cantidades que faltan.

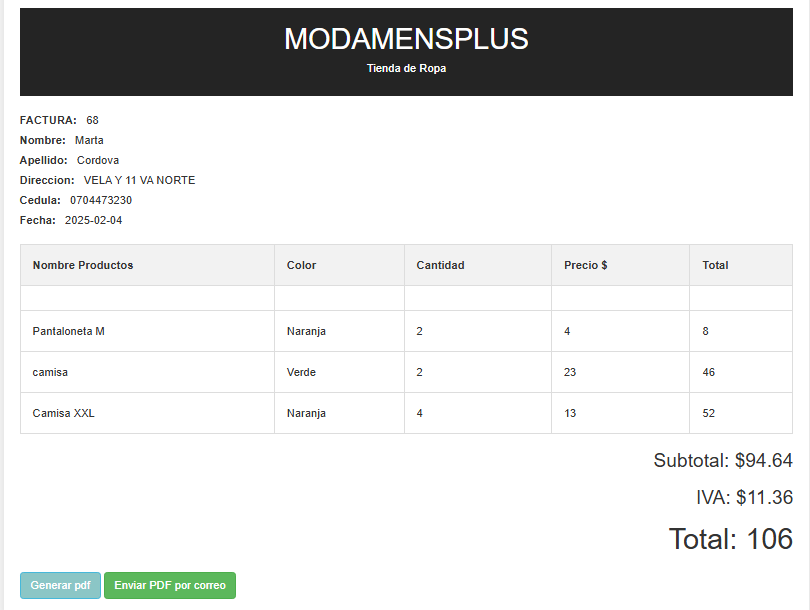
**VENTAS**

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.



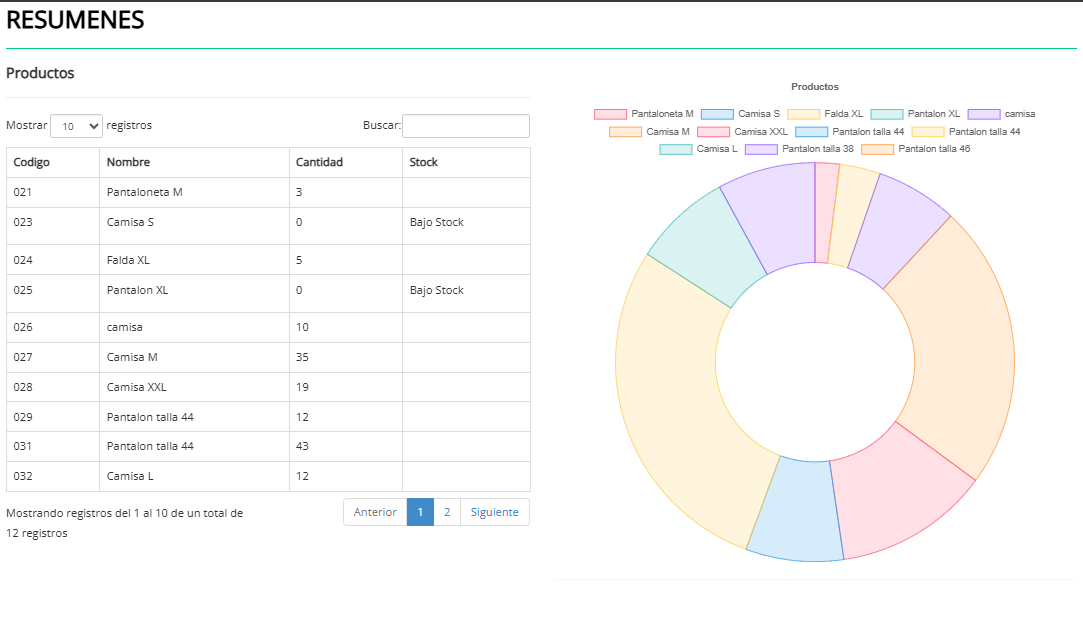
****

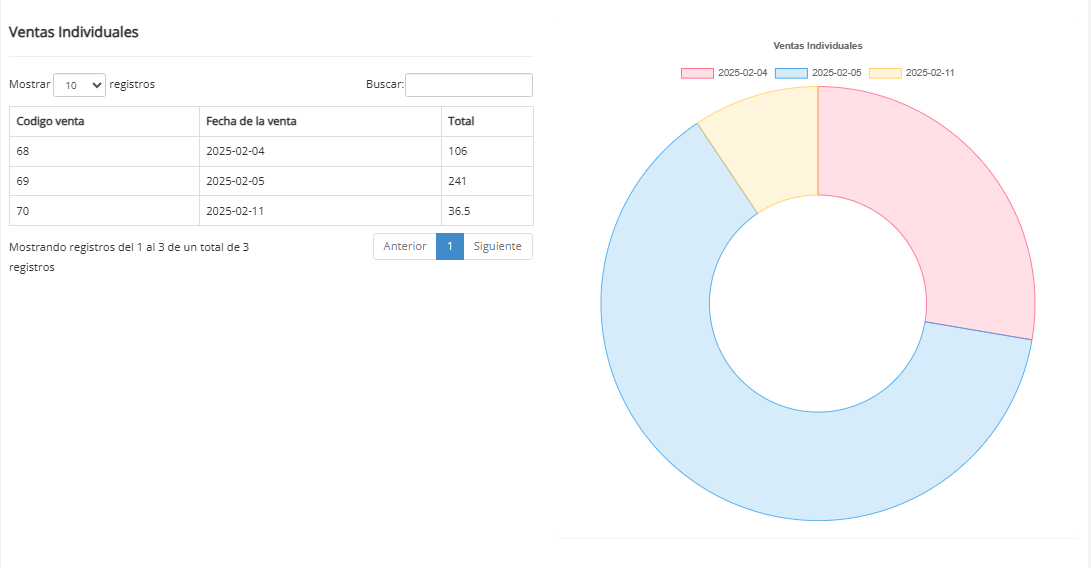
En este apartado se podra visualizar lo que es las ventas de cada cliente tambien se podra dar clic en el archivo para poder visualizar lo que se logro vender al cliente y al dar clic aparece en el siguiente apartado.



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>