UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FACULTAD DE COMERCIO INTERNACIONAL, INTEGRACIÓN, ADMINISTRACIÓN Y ECONOMÍA EMPRESARIAL

CARRERA DE LOGÍSTICA Y TRANSPORTE

Tema: “Diseño de un modelo de gestión en base tecnológica para la optimización de procesos de pedidos a proveedores de los negocios comerciales en la ciudad de Tulcán.”

Trabajo de titulación previa la obtención del

título de Ingeniero en Logística y Transporte

AUTOR: Jonnathan Xavier Aguilar Beltrán

TUTOR: Eco. Argenis Lissander Heredia Campana Msc.

Tulcán, 2021

CERTIFICADO JURADO EXAMINADOR

Certificamos que el estudiante Aguilar Beltrán Jonnathan Xavier con el número de cédula 0401622774 ha elaborado el trabajo de titulación: “Diseño de un modelo de gestión en base tecnológica para la optimización de procesos de pedidos a proveedores de los negocios comerciales en la ciudad de Tulcán.”

Este trabajo se sujeta a las normas y metodología dispuesta en el Reglamento de Titulación, Sustentación e Incorporación de la UPEC, por lo tanto, autorizamos la presentación de la sustentación para la calificación respectiva.

f.....

Econ. Heredia Campana Argenis Lissander, Msc

TUTOR

Tulcán, diciembre de 2021

AUTORÍA DE TRABAJO

El presente trabajo de titulación constituye requisito previo para la obtención del título de **Ingeniero** en la Carrera de logística y transporte de la Facultad de Comercio Internacional, Integración, Administración y Economía Empresarial

Yo, Aguilar Beltrán Jonnathan Xavier con cédula de identidad número 0401622774 declaro: que la investigación es absolutamente original, auténtica, personal y los resultados y conclusiones a los que he llegado son de mi absoluta responsabilidad.

f  **.....**

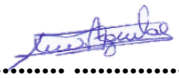
Aguilar Beltrán Jonnathan Xavier

AUTOR

Tulcán, diciembre de 2021

ACTA DE CESIÓN DE DERECHOS DEL TRABAJO DE TITULACIÓN

Yo, Aguilar Beltrán Jonnathan Xavier declaro ser autor/a de los criterios emitidos en el trabajo de investigación: “Diseño de un modelo de gestión en base tecnológica para la optimización de procesos de pedidos a proveedores de los negocios comerciales en la ciudad de Tulcán” y eximo expresamente a la Universidad Politécnica Estatal del Carchi y a sus representantes legales de posibles reclamos o acciones legales.

f  **..... .....**

Aguilar Beltrán Jonnathan Xavier

AUTOR

Tulcán, diciembre de 2021

AGRADECIMIENTO

Agradezco en primer lugar a Dios por la vida de las personas que han hecho posible que yo cumpla una meta más en mi vida.

Gracias a mis padres, en especial a mi madre que con su apoyo y sus consejos he sido capaz de superar los retos que se han presentado a lo largo de mi vida estudiantil, y con esta meta cumplida puedo confirmar que el esfuerzo es recompensado.

Gracias a mis profesores de la carrera de Logística y Transporte, por compartir sus conocimientos y experiencias que se, me serán útil para mi vida profesional. Un agradecimiento en especial al Eco. Argenis Heredia, por la paciencia y por la dedicación que me ofreció, para culminar con mi trabajo de titulación.

Por ultimo quiero agradecer a cada uno de mis compañeros, por los momentos de convivencia en el aula y fuera de ella, porque cada una de las acciones contribuyeron a que yo pueda convertirme un colega Logístico de cada uno de ellos.

DEDICATORIA

Este logro lo dedico a Dios, porque ha sido las fuerzas que faltaban para cumplir mis metas diarias.

Este esfuerzo le dedico a mis padres que demuestra el sacrificio que ellos hacen cada día para que mi hermana y yo podamos crecer como personas con valores y principios, de igual profesionalmente al regalarnos la educación.

También lo dedico a mis abuelitos, en especial a mi abuelito Luis Beltrán quien durante su vida siempre supo darme un consejo y enseñarme las cosas de la vida.

ÍNDICE

RESUMEN 13
ABSTRAC 14
INTRODUCCIÓN 14
I. PROBLEMA 16
1.1. PLANTEAMIENTO DEL PROBLEMA 16
1.3. JUSTIFICACIÓN 17
1.4. OBJETIVOS Y PREGUNTAS DE INVESTIGACIÓN 17
1.4.1. Objetivo General 17
1.4.2. Objetivos Específicos 17
1.4.3. Preguntas de Investigación 18
II. FUNDAMENTACIÓN TEÓRICA 19
2.1. ANTECEDENTES INVESTIGATIVOS 19
2.2. MARCO TEÓRICO 19
2.2.1. Modelo de gestión 20
2.2.2 Optimización de recursos 20
2.2.3 Proveedores 20
2.2.4 Almacén 20
2.2.4.1 Almacenamiento 20
2.2.4.4. Punto de re-orden 21
2.2.3 Importancia del Tiempo en un modelo de gestión de pedidos a proveedores: 21
2.2.4 Costos de inventarios 21
2.2.6 Lenguaje de programación. 21
2.2.6 Modelos de Inventarios 21
III. METODOLOGÍA 24
3.1. ENFOQUE METODOLÓGICO 24
3.1.1. Enfoque 24
3.1.2. Cualitativo 24
3.1.2. Tipo de Investigación 24
3.2. IDEA A DEFENDER 25
3.3. DEFINICIÓN Y OPERACIONALIZACIÓN DE VARIABLES 26
IV. RESULTADOS Y DISCUSIÓN 28
4.1. RESULTADOS 28

4.1.1 Describir la situación actual **d**e los procesos generales de la gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal. 28

4.1.2 Determinar los parámetros críticos de los modelos de gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal”..... 37

4.1.3 Programar el modelo de base tecnológica para la gestión de pedidos a proveedores con base en los parámetros críticos del modelo utilizando varios lenguajes de programación. 38

4.1.4 Simular los procesos de gestión de pedidos a proveedores con la implementación del modelo diseñado. 56

4.2. DISCUSIÓN 62

5. CONCLUSIONES Y RECOMENDACIONES 64

5.1. CONCLUSIONES 65

5.2. RECOMENDACIONES 65

IV. REFERENCIAS BIBLIOGRÁFICAS 67

V. ANEXOS 70

ÍNDICE DE FIGURAS

Figura 1. Diagrama de flujo de los procesos de la gestión de pedidos a proveedores 28

Figura 2. SISMANOS Sistema tecnológico 29

Figura 3. Principales proveedores 32

Figura 4. Diagrama de Pareto de la participación en los inventarios 35

Figura 5. Diagrama de flujo de los procesos de la gestión de pedidos a proveedores - puntos críticos. 37

Figura 6. Diagrama entidad relación 39

Figura 7. Proveedores 39

Figura 8. Tabla existencias 40

Figura 9. Tabla ventas 41

Figura 10. Tabla configuraciones 41

Figura 11. Método de regresión exponencial 44
Figura 12. Método de regresión logarítmica 45
Figura 13. Método de regresión lineal 45
Figura 14. Formula del Modelo de Regresión Lineal Simple 46
Figura 15. Fórmula para obtener "b" 46
Figura 16. Fórmula para obtener "a" 46
Figura 17. Fórmula para calcular la Demanda Promedio 47
Figura 18. Fórmula para el cálculo del Stock de Seguridad 47
Figura 19. Fórmula para calcular Política de inventario optimo 47
Figura 20. Formula del Modelo EOQ 47
Figura 21. Formula de la cantidad total a pedir 48
Figura 22. Formula del Costo Total del Modelo 48
Figura 23. Estructura para la programación de la Interface 49
Figura 24. Vista inicio 50
Figura 25. Vista inicio después de la petición al servidor 50
Figura 26. Vista registro producto 51
Figura 27. Vista registro proveedor 52
Figura 28.vista ventas 53
Figura 29. Vista modelo de irc-productos que se requieren 54
Figura 30. Vista modelo de irc-todos los productos 54
Figura 31. Vista estadísticas 55
Figura 32. Vista configuraciones. 56
Figura 33. Diagrama del funcionamiento del software 56
Figura 34. Parámetros para la simulación 57
Figura 35. Tabla de resultados del modelo diseñado con el 90% de nivel de confianza 58
Figura 36. Tabla de resultados del modelo diseñado con el 95% de nivel de confianza 58
Figura 37.Resultados para el Modelo de Regresión Lineal Simple 59
Figura 38. Resultados para calcular el stock de seguridad 59
Figura 39.Resultados para calcular el EOQ 59
Figura 40. Código para la creación de la base de datos 74

Figura 41. Código para importar los módulos necesarios para el servidor. 75
Figura 42. Código para la creación e inicialización del servidor 75
Figura 43. Código para la conexión a la base de datos 75
Figura 44. Código para insertar proveedores en la base de datos 75
Figura 45. Código para insertar existencias o inventarios en la base de datos..... 76
Figura 46. Código para insertar ventas en la base de datos..... 77
Figura 47. Código para insertar los parámetros del modelo en la base de datos..... 78
Figura 48. Código para obtener todos los datos de una tabla de la base de datos – proveedores 78
Figura 49. Código para obtener todos los datos de una tabla de la base de datos – existencias 78
Figura 50. Código para obtener todos los datos de una tabla de la base de datos – ventas..... 78
Figura 51. Código para obtener todos los datos de una tabla de la base de datos – configuraciones..... 79
Figura 52. Código para obtener todos los datos de dos tablas de la base de datos..... 79
Figura 53. Código para obtener un dato especifico de una tabla de la base de datos – existencias 80
Figura 54. Código para obtener un dato especifico de una tabla de la base de datos – ventas 80
Figura 55. Código para la programación de los modelos para pronostico e inventarios..... 83
Figura 56. Código del archivo App.vue de la Interface 84
Figura 57. Código de la vista inicio 85
Figura 58. Código del componente inicioc 88
Figura 59. Código de la vista registro 89
Figura 60. Código del componente registroc 94
Figura 61. Código de la vista ventas 95
Figura 62. Código del componente ventasc 97
Figura 63. Código de la vista irc 98
Figura 64. Código del componente modelo_irc 105
Figura 65. Código de la vista estadísticas 106
Figura 66. Código del componente graficas estadísticas..... 109
Figura 67. Código de la vista config 110
Figura 68. Código del componente configuraciones..... 112

Figura 69. Plantilla del método de regresión lineal 113
Figura 70. Plantilla del modelo de inventarios de revisión continua 113

ÍNDICE DE TABLAS

Tabla 1. Operalización de Variables 26
Tabla 2. Proveedores 29
Tabla 3. Principales proveedores 31
Tabla 4. Tiempos de respuesta de los proveedores 32
Tabla 5. Principales productos 33
Tabla 6. Arriendo almacén 35
Tabla 7. Servicios Básicos 36
Tabla 8. Seguridad del almacén 36
Tabla 9. Mantenimiento del Almacén 36
Tabla 10. Pronósticos de la plantilla versus el software diseñado 63
Tabla 11. Media Aritmética de la plantilla versus el software diseñado 63
Tabla 12. . Varianza de la plantilla versus el software diseñado 64
Tabla 13. Modelo del inventario de revisión continua de la plantilla versus el software diseñado 64
Tabla 14. Leguajes de programación utilizados 65

ÍNDICE DE ANEXOS

Anexo 1: Certificado o Acta del Perfil de Investigación 70
Anexo 2:Certificado del abstract por parte de idiomas 71
Anexo 3: Entrevista al propietario del negocio Tus Enkntos 73
Anexo 4: Programación del software 74

Anexo 5. Plantillas de Excel.....113
Anexo 6. Ficha del modelo de gestión automática de pedidos a proveedores 114

RESUMEN

La presente investigación fue realizada con el fin de diseñar un modelo de gestión en base tecnológica que aporte a la optimización de procesos de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal en la ciudad de Tulcán. Para el diseño del modelo de gestión, primero se realizó una encuesta semiestructurada a la Señora Paola Mejía propietaria del negocio comercial, con la cual se buscó recaudar información referente a la situación actual de los procesos de pedidos a proveedores, posteriormente se identificó los parámetros críticos de los procesos y se procedió a diseñar el modelo tomando en cuenta modelos como, Regresión Lineal Simple, Inventarios de la Política de Revisión Continua y EOQ Cantidad Óptima de Pedido, para la programación del software se procedió a utilizar leguajes de programación como SQL para la base de datos, Python para el servidor y JavaScript para la Interface. Concluyendo que los procesos de pedidos a proveedores deben de manejarse de acuerdo a los tiempos planificados por cada empresa proveedora, por lo que la revisión de inventarios debe ser eficiente con la ayuda de un software y con la simulación se comprobó que el modelo diseñado optimiza los procesos de la gestión, porque el negocio puede conocer la cantidad de pedido, la posible demanda de los productos, procurando disminuir los riegos de no tener inventario o de tener exceso de inventario

Palabras Claves: Modelo, Gestión, Procesos, Almacén, Stock, Software

ABSTRAC

The present research was developed with the objective of designing a technology-based management model that contributes to the optimization of the order processing to suppliers of the TUS ENKNTOS commercial business is dedicated to the wholesale of perfumery, cosmetic, personal hygiene products in the Tulcán city. Hence, for the design of the management model first a semi-structured survey was carried out of the owner of the commercial business, which allowed collecting information regarding the current situation of the supplier ordering processes. Subsequently, the critical parameters of the processes were identified, and the model was designed taking into account Simple linear regression, review system in inventory management and EOQ Economic Order Quantity. Likewise, for the programming of the software, programming languages such as SQL for the database, Python for the server and JavaScript for the Interface. Concluding that the supplier order processes must be handled according to the times planned by each supplier company, so the inventory review have to be efficient with the help of software. Finally, with the simulation it was found that the designed model helps to optimize the management processes, thus knowing the order quantity, the possible demand for the products, and trying to reduce the risks of not having inventory or having excess inventory.

Keywords: Model, Management, Processes, Warehouse, Stock, Software

INTRODUCCIÓN

La tecnología se creó para que los seres humanos pudieran idear, crear, rehacer, implementar e innovar los aparatos de nueva tecnología, para hacer la vida más útil y eficiente. (Mendoza, 2014). Con la presente investigación se pretendió mostrar cuan eficaz y eficiente puede ser un proceso si se lo automatiza con un software permitiendo mejorar una gestión que anteriormente era empírica y manual.

Capítulo 1: Se identifica la problemática de investigación, determinando variables principales como: modelo de gestión y procesos de gestión finalmente se estableció el objetivo general, objetivos específicos, preguntas de investigación para el desarrollo de la presente investigación.

Capítulo II: Se describió la teoría de sistemas y trabajos similares a la presente investigación realizados por otros autores, que de cierta forma sirvieron para para el desarrollo del trabajo investigativo y para la Interpretación de resultados obtenidos y discusión.

Capítulo III: Se desarrolló el enfoque metodológico cuantitativo y cualitativo, para cumplir con los objetivos propuestos y dar respuesta al problema planteado, se utilizó el tipo de investigación descriptiva y documental, métodos que se utilizó como hipotético-deductivo para la recolección de información, aplicando los instrumentos de investigación la entrevista semiestructurada tomando en cuenta la idea a defender, definición y Operacionalización de variables.

Capítulo IV: Se desarrolló los objetivos respectivos para cumplir con el objetivo general mediante la descripción de la situación actual de los procesos generales de la gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS, la determinación de los parámetros críticos del modelo de gestión, la programación del software para la gestión de pedidos a proveedores y finalmente se simulo los procesos de gestión de pedidos a proveedores con la implementación del modelo diseñado.

Capítulo V: Se plasmó las conclusiones y recomendaciones de acuerdo a los resultados obtenidos en la presente investigación.

I. PROBLEMA

1.1. PLANTEAMIENTO DEL PROBLEMA

Los negocios comerciales de los países que se encuentran en vías de desarrollo tienen dificultades para implementar tecnología que les ayude a automatizar la gestión de los diferentes procesos empresariales, en muchos de los casos, el desconocimiento hace que los negocios manejen todos los procesos de manera manual; en estos países existe el estudio o producción de tecnología es relativamente baja, causando ignorancia en sus ciudadanos sobre este tipo de temas.

Los negocios comerciales en Ecuador no demandan de tecnología que les ayude a optimizar la gestión de sus procesos es por eso que operan de manera manual situaciones que pueden ser automatizadas por medio de la misma, al ser un país en vías de desarrollo recién esta implementado una educación que incentive al estudio y a la creación de tecnología; Esta es la razón por la que muchos dueños de empresas o negocios ignoran los beneficios que tiene para sus procesos la implementación de tecnología.

En la ciudad de Tulcán los negocios comerciales que tienen como actividad económica principal, la venta al por mayor de productos de perfumería, cosméticos, productos de belleza, Artículos de aseo personal, ignoran los beneficios que tiene la implementación de tecnología para automatizar, optimizar procesos y recursos, como es el caso del negocio Tus Enkntos que por el desconocimiento, lleva una gestión manual de los procesos de pedidos a proveedores generando los problemas que a continuación se mencionan.

El principal problema es que el negocio no cuenta con un sistema tecnológico especializado en gestión de inventarios, pronóstico de demanda, cálculo del costo logístico en base a los costos de pedido y mantenimiento, por ende, el modelo de gestión de pedidos a proveedores no es el indicado, por lo que trae como consecuencia la toma de malas decisiones al realizar un pedido.

La falta de un sistema tecnológico especializado, incita al cálculo de la cantidad de pedido de forma empírica, basándose en creencias de la propietaria, lo que quiere decir que la decisión de cuanto comprar no tiene fundamento técnico, provocando una mala decisión al no pedir la cantidad de pedido óptima, siendo esto un error humano.

Al pedir la cantidad equivocada, causa que el negocio contraiga una rotura de stock antes que el proveedor pueda reabastecer el negocio, incitando a los clientes acudir a otros establecimientos que los provean de los productos que necesitan.

1.2. FORMULACIÓN DEL PROBLEMA

¿Cómo un modelo de gestión de base tecnológica aporta en la optimización de los procesos de gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal, en la ciudad de Tulcán, en el año 2021?

1.3. JUSTIFICACIÓN

La presente investigación tiene como objetivo diseñar un modelo de gestión de pedidos a proveedores que permita optimizar los procesos que lleva acabo el negocio comercial Tus Enkntos al realizar un pedido para reabastecerse de cada uno de los productos que oferta.

Para para dar respuesta al problema planteado se basó en el diagnóstico que se realizó al negocio comercial Tus Enkntos respecto a los procesos generales de pedidos a proveedores, que posteriormente permitió conocer los parámetros críticos y con los cuales se diseñó el modelo de gestión y se programó el software.

Para la programación del software se recurrió a leguajes de programación que permitan generar un software fácil de usar, pero que a su vez sea robusto, es por eso que se usó JavaScript y Python como leguajes principales.

Al implementar el software para la gestión de pedidos a proveedores optimiza los esfuerzos en recursos y tiempo, evitando los errores humanos y mejora la calidad de los procesos de la gestión de pedidos a proveedores.

. Con la realización de esta investigación se pretende ayudar al negocio comercial Tus Enkntos de la ciudad de Tulcán, incentivando al uso de tecnología para los diferentes procesos, en especial los procesos logísticos que tengan que ver con inventarios y pedidos a proveedores, permitiendo conocer los beneficios que les brinda la tecnología, como por ejemplo, tiempo para buscar estrategias que impulsen el crecimiento de su negocio, incrementando la confianza en sus clientes por el hecho de satisfacer sus requerimientos al instante.

1.4. OBJETIVOS Y PREGUNTAS DE INVESTIGACIÓN

1.4.1. Objetivo General

Diseñar un modelo de gestión en base tecnológica que aporte a la optimización de procesos de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal en la ciudad de Tulcán, en el año 2021.

1.4.2. Objetivos Específicos

- Describir la situación actual de los procesos generales de la gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal.
- Determinar los parámetros críticos del modelo de gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal".
- Programar el software para la gestión de pedidos a proveedores con base en los parámetros críticos del modelo utilizando varios lenguajes de programación.

- Simular los procesos de gestión de pedidos a proveedores con la implementación del modelo diseñado.

1.4.3. Preguntas de Investigación

- ¿Cuál es la situación actual de los procesos generales de la gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal?

- ¿Cuáles son los parámetros críticos del modelo de gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal"?

- ¿Cómo se programa el software para la gestión de pedidos a proveedores con base en los parámetros críticos del modelo utilizando varios lenguajes de programación?

- ¿Cómo se simula los procesos de gestión de pedidos a proveedores con la implementación del modelo diseñado?

II. FUNDAMENTACIÓN TEÓRICA

2.1. ANTECEDENTES INVESTIGATIVOS

Gómez y Guzmán (2016) con su trabajo de investigación buscaron la solución a la falta de un sistema de inventarios eficiente para los proyectos de construcción de la empresa Ingeniería Sólida Ltda. Los investigadores implementaron el sistema de revisión continua con demanda variable y tiempo de anticipación contante, concluyendo que el sistema desarrollado le garantizara a la empresa una disminución en las fallas, que se presentan dentro del almacén y así llevar una eficiente y exitosa administración de los recursos existentes.

Aragón (2017) realizo una investigación para el diseño de un modelo de gestión de inventarios para una empresa comercializadora. El modelo da solución al problema de estandarización de procesos de inventarios para las pymes establecidas en México, este trabajo está fundamentado en el modelo de inventarios EOQ, utilizando datos de demanda anual, costo unitario de producto, costo de mantener el inventario, costo de ordenar un pedido, tiempo de entrega del proveedor y la tasa de interés del dinero para inventario. Con esto se concluye que el modelo diseñado logra minimizar el costo total de la política actual, determinado por una decisión de balance entre el tamaño de lote y frecuencia de pedido.

Juca, Narváez, Erazo y Luna (2019) presentaron un modelo de gestión de inventarios para la determinación de los niveles óptimos en la cadena de suministros de la empresa Modesto Casajoana Cía. Ltda. en la ciudad de Quito Ecuador, en donde aplicaron el modelo probabilístico de revisión periódica con demanda dinámica para establecer el nivel óptimo de pedidos, el inventario de seguridad, los puntos de re-orden y los costos totales de inventario. Teniendo como resultado que el modelo propuesto constituye un instrumento para optimizar el control y la gestión de la cadena de suministros garantizando la conservación de productos de uso humano y que es válido para empresas distribuidoras de productos de aseo y farmacéuticos.

2.2. MARCO TEÓRICO

Teoría de sistema

La teoría de sistemas estudia los diferentes sistemas y subsistemas que tiene una un organismo que en este caso son las partes o sistemas que hacen que una empresa funciones. En relación a la cadena de suministros esta teoría pretende identificar las bases teórico - metodológicas que influyen en la formación de la cadena y los tipos de relación entre sus elementos. (Tapia, 2014)

El enfoque sistémico permite entender una organización como un conjunto de subsistemas interactúales e independientes que se relacionan formando un todo complejo. Los sistemas y subsistemas despliegan una serie de eventos que parte con una entrada y terminan en una salida;

lo que ocurre entre la entra y la salida constituye la esencia del subsistema y se conoce como proceso o caja negra.

Las entras al sistema pueden ser recursos materiales, recursos humanos o información, para esta investigación las entradas son los productos de perfumería, cosméticos y artículos de aseo personal. Una entrada puede ser la salida otro subsistema por ejemplo en la gestión de pedios a proveedores la salida es la información, que a su vez es la entra para que el producto requerido llegue al negocio comercial y este es una entrada para el proceso de venta. Moreno (2012)

Las salidas son el resultado del proceso que se les dio a las entradas que pueden traducirse en productos, servicios o información y ser la entrada de otro subsistema. Moreno (2012)

La gestión de pedios a proveedores es un subsistema que tiene como entada la información sobre los inventarios y su salida es la descripción del pedido que será emitido a los diferentes proveedores. Es por esta razón que la presente investigación toma como referencia a la teoría de sistemas.

2.2.1. Modelo de gestión

> Según Porto y Gardey (2020) un modelo de gestión es un esquema de referencia para la administración de una entidad y puede ser aplicado tanto en las empresas y negocios privados como en la administración pública.

2.2.2 Optimización de recursos

> La optimización es buscas la mejor alternativa que permita mayor eficiencia o mejor eficacia en el desempeño de algún trabajo u objetivo a lograr, en este caso del recurso de una empresa, llamándose optimización de recursos (Gonzales, 2019, p.1).

2.2.3 Proveedores

> Son las personas o entidades encargadas de suministrar las materias primas, servicios o productos terminados necesarios para que la empresa pueda seguir con sus actividades. (Montoya, 2002, p.27).

2.2.4 Almacén

> Almacén es el lugar donde se guardan o depositan mercancías o materiales, pueden ser sitios donde se venden artículos al por mayor (Múzquiz, 2013, p.7).

2.2.4.1 Almacenamiento

> El almacenamiento es el proceso o acción de guardar o archivar algo. Un ejemplo, con distintas acepciones, se produce cuando se almacenan mercancías en un depósito (Westreicher, sf).

2.2.4.2. Manejo o control del inventario

> Es el proceso por el cual una empresa administra las mercancías que mantiene en almacén. El manejo de inventarios tiene el objetivo de recopilar información de la

entrada y salida de los productos, buscando además el ahorro de costes (Westreicher, sf).

2.2.4.3. Stock mínimo o Stock de seguridad

"Ese inventario de seguridad es una protección contra la incertidumbre de la demanda, del tiempo de entrega y del suministro" (Carro & Gonzáles, 2013, p.4).

2.2.4.4. Punto de re-orden

El punto de re-orden es el nivel de inventario de un material o producto que señala la necesidad de realizar una orden de reabastecimiento. El punto de re-orden es la suma de la demanda de tiempo de entrega y las existencias de seguridad (Vermorel, 2012, p.1).

2.2.3 Importancia del Tiempo en un modelo de gestión de pedidos a proveedores:

El modelo de gestión de pedidos a proveedores tiene que tener como un punto principal el tiempo ya que de esta gestión depende la producción o la venta del producto y por ende la satisfacción del cliente. Dentro del modelo de gestión se debe establecer tiempos óptimos donde no se vea comprometida la reputación de la empresa por un descontento del cliente.

2.2.4 Costos de inventarios

2.2.4.1 Costos de mantenimiento

"son los gastos en que se incurre al mantener inventarios, p. ej. alquiler, electricidad, impuestos, pérdidas, obsolescencia, primas de seguros y costos de mano de obra." (Castillo, 2005, p.5)

2.2.4.2 Costos de pedido

"son los ocasionados por el transporte de unpedido de artículos. Abarcan actividades de compra, preparación de especificaciones y documentos, órdenes de compra, seguimiento a los proveedores e inspección de pedidos cuando llegan" (Castillo, 2005, p.4)

2.2.5 Modelos de Inventarios

2.2.5.1 Determinísticos

Betancourt (2017) afirma que:

"Son aquellos donde se toma como supuesto que tenemos certeza de la demanda. Esta puede estar dada por pronósticos de demanda o pedidos reales de los clientes" (p.1)

2.2.5.2 Probabilísticos

> “Un inventario estocástico o probabilístico presenta una demanda o tiempo de entrega desconocido (es aleatorio), por lo que esta demanda o tiempo es expresado a través de una variable aleatoria.” (Betancourt, 2018, párr.3).

2.2.6 Lenguaje de programación.

> “Es un idioma artificial diseñado para expresar computaciones que pueden ser llevadas a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión”. (Olarte, 2018, p.1)

2.2.6.1 JavaScript

> “Es un lenguaje de programación o de secuencias de comandos que te permite implementar funciones complejas en páginas web” (MDN contributors, 2021, p.1).

2.2.6.1.1 VUE JS

> Es un framework JavaScript, es decir, es un aglomerado de herramientas y funciones que permiten programar páginas web de una manera más sencilla Coding Potions (2021).

2.2.6.1.2 PWA (Progressive Web Apps)

> Es una solución basada en la web tradicional, aunque incorpora algunas particularidades que la hacen parecerse a una app nativa para dispositivos como computadores, teléfonos móviles y tabletas Vidal (2019).

2.2.6.2 Python

> Es un lenguaje de programación multiplataforma y multiplataforma, se destaca por su código legible y limpio. Robledano (2019).

2.2.6.2.1 Flask

> Es un micro Framework escrito en Python y desarrollado para simplificar y hacer más fácil la creación de Aplicaciones Web, esta herramienta es utilizada para la parte del Backend de una aplicación web Epitech (2021).

2.2.6.3 Base de datos

> Se llama base o banco de datos, a un conjunto de información perteneciente a un mismo contexto, ordenada de modo sistemático para su posterior recuperación, análisis y transmisión. Raffino (2020).

2.2.6.3.1 SQL (Structured Query Language)

El lenguaje de consultas estructuradas es un lenguaje de programación estandarizado que se utiliza para administrar bases de datos relacionales y realizar diversas operaciones con los datos que contienen. Sirkin (2021)

Tipos de datos SQL

Se indican los tipos de datos que se utilizó para el desarrollo de la investigación.

Varchar: Cadena de caractéres

Int: Número entero.

Decimal: Número decimal

Date: Fecha

2.2.7. Nivel de confianza

“Indica la proporción de veces que acertaríamos al afirmar que el parámetro θ está dentro del intervalo al seleccionar muchas muestras” (Montero, 2019, p.48)

III. METODOLOGÍA

3.1. ENFOQUE METODOLÓGICO

3.1.1. Enfoque

3.1.2. Cualitativo

"Descripciones detalladas de situaciones, eventos, personas, interacciones, conductas observadas y sus manifestaciones" (Hernandez, Fernández y Baptista, 2014, p. 9)

El desarrollo de la investigación se basará en obtener información sobre la gestión de los procesos de pedidos a proveedores del negocio comercial Tus Enkntos de la ciudad de Tulcán permitiendo describir la situación actual de los mismos que posteriormente ayudará a identificar los puntos críticos de esta gestión.

3.1.1.2 Cuantitativo

"Usa la recolección de datos para probar hipótesis con base en la medición numérica y el análisis estadístico, para establecer patrones de comportamiento y probar teorías" (Hernandez, Fernández y Baptista, 2014, p. 4)

Este tipo de investigación ayudara a obtener la información útil para medir la eficiencia de los procesos de gestión de pedido a proveedores, al implementar el modelo de gestión en base tecnología en un escenario simulado.

3.1.2. Tipo de Investigación

3.1.2.1 Investigación Descriptiva:

"La investigación descriptiva busca especificar propiedades, características y rasgos importantes de cualquier fenómeno que se analice" (Posso, 2013, p.1)

Para la realización de esta investigación se utilizó este tipo de investigación lo que resulta factible porque se describe los procesos de la gestión de pedidos

3.1.2.2 Investigación Documental

Guiza (2013) manifiesta que la investigación documental "constituye una estrategia donde se observa y reflexiona sistemáticamente sobre realidades usando diferentes tipos de documentos; indaga, interpreta, y presenta datos coherentes". (p.2)

Este tipo de investigación se la utilizo para recabar información que permita sustentar la fundamentación teórica de la investigación, así como para comparar información en la interpretación de resultados.

3.2. IDEA A DEFENDER

Un modelo de gestión de base tecnológica aporta en la optimización de los procesos de gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal en la ciudad de Tulcán, en el año 2021

3.3. DEFINICIÓN Y OPERACIONALIZACIÓN DE VARIABLES

Tabla 1. Operalización de Variables

Variables	Definición	Dimensiones	Indicadores	Técnica	Instrumento
Independiente: **Modelo de gestión**	Según Porto (2008) un modelo de gestión es un esquema o marco de referencia para la administración de una entidad.	Gestión manual	Costos de la gestión manual	Entrevista Semiestructurada	Guía de preguntas
			Tiempo		
		Gestión automatizada	Costo de gestión automática	Registros	Fichas
			Tiempo		
Dependiente: **Procesos de gestión**	Según Macia (2014) son el medio por el que una organización hace llegar a los destinatarios el producto o servicio demandado, transformando la actividad del personal y materias primas en resultados.	Manejo del Inventario	Costo total del inventario	Entrevista Semiestructurada	Guía de preguntas
			Tiempo de la gestión de pedidos a proveedores		
		Abastecimiento	Cantidad de pedido	Documentos	Fichas

3.4. MÉTODOS UTILIZADOS

3.4.1. Método Hipotético Deductivo:

Se utiliza este método de investigación ya que la idea a defender planteada posteriormente fue comprobada mediante la simulación implementando el modelo de gestión en base tecnológica, para conocer la optimización del proceso de gestión de pedidos a proveedores.

3.4.2. Descripción de la situación actual de los procesos de pedidos a proveedores

Para conocer la situación actual de los procesos del negocio comercial Tus Enkntos se utilizó la entrevista semiestructurada dirigida a la propietaria del negocio, con la información se procedió a realizar diagramas de flujos, con los datos de los proveedores con los que cuenta y de los reportes de existencias proporcionados, se procedió a elaborar tablas que muestren la participación de los proveedores durante el 2020 y la participación de las exigencias durante el mes de junio.

Diagramas que representan flujo. - Para evidenciar los procesos que se llevan a cabo al momento de realizar un pedido a proveedores y conocer cuáles son los parámetros críticos dentro de esta gestión

Diagrama de Pareto. - Para conocer los productos que tiene la mayor participación en los inventarios, y poder utilizarlos en la simulación de modelo diseñado.

Programación del software utilizo los siguientes lenguajes de programación.

SQL con la ayuda de la herramienta MySQL Workbench se programó la base de datos.

Python es el leguaje que permitió programar el servidor con su framework Flask y se utilizó el editor de código Visual Studio Code.

JavaScript es el leguaje con el que se programó la Interface con la ayuda de su framework VUE JS.

3.4.2.1 Población y Muestra

“Si la población, por el número de unidades que la integran, resulta accesible en su totalidad, no será necesario extraer una muestra” (Arias, 2012, p.83)

En la ciudad de Tulcán existe una población aproximada de 25 negocios que se dedican a la venta al por mayor de productos de perfumería, cosméticos (Productos de Belleza), Artículos de aseo personal; siguiendo con lo citado anteriormente la población es pequeña por lo que es mejor tomar todos los negocios y debido a la similitud en la gestión de sus procesos, se tomó al Negocio Tus Enkntos como piloto.

IV. RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS

4.1.1 Describir la situación actual de los procesos generales de la gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal.

El resultado de este objetivo es indagar sobre los procesos implementados en la gestión de pedidos a proveedores, y con la información proporcionada por el negocio comercial piloto conocer como calculan la cantidad de pedido, el tiempo que toma en efectuarse todas las actividades hasta abastecer el negocio. A continuación, se presenta un diagrama de flujo.

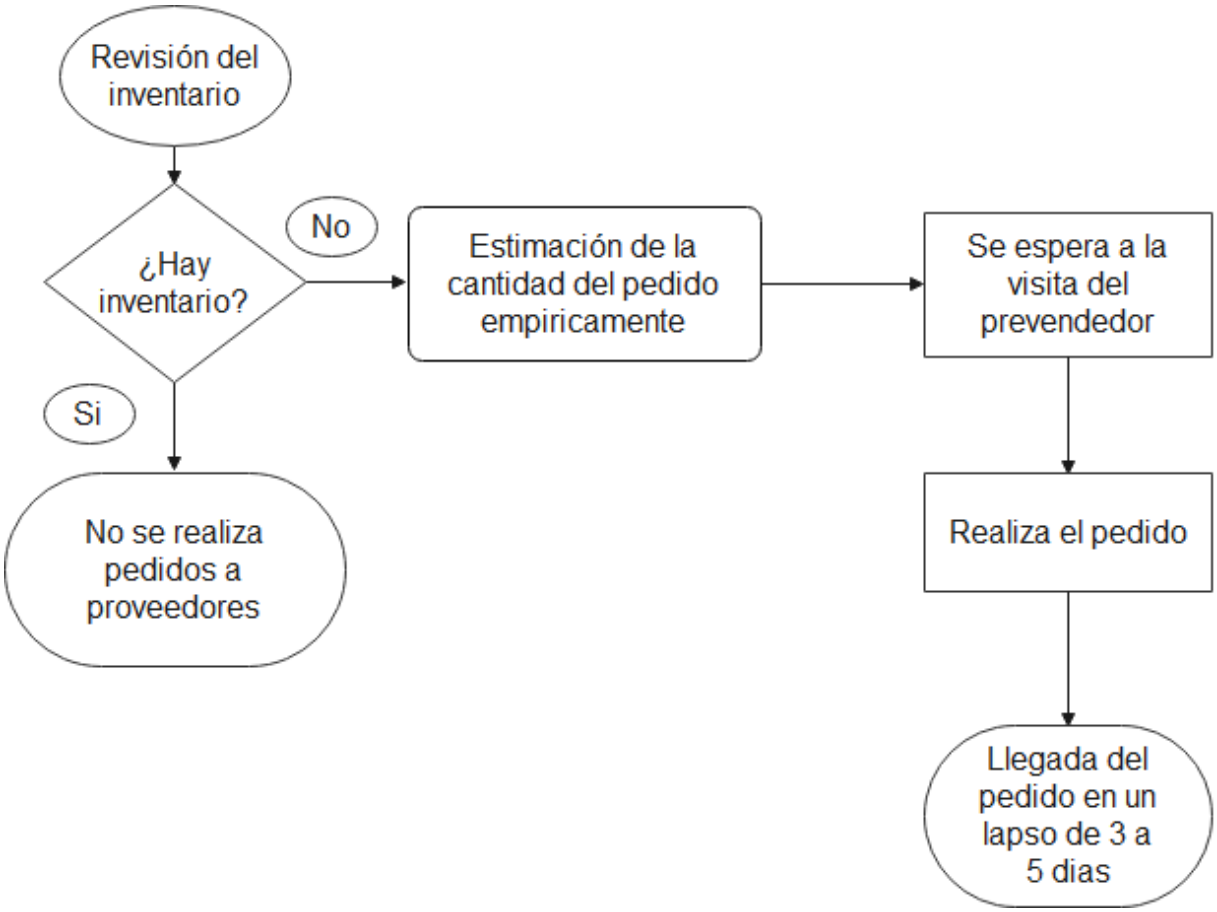


Figura 1. Diagrama de flujo de los procesos de la gestión de pedidos a proveedores

El negocio utiliza SISMANOS que es un sistema tecnológico que le ayuda a la facturación y al registro de inventarios por lo que la revisión de los inventarios no es de manera física, puesto que, el sistema les brinda un reporte de existencias, pero no existe un módulo que les ayude a estimar de manera técnica la cantidad de pedidos a proveedores debido a que el software no registra las ventas por producto. Otro punto es la revisión, que se realiza un día previo o en el instante de la visita del pre vendedor, por lo que no consideran un inventario de seguridad.

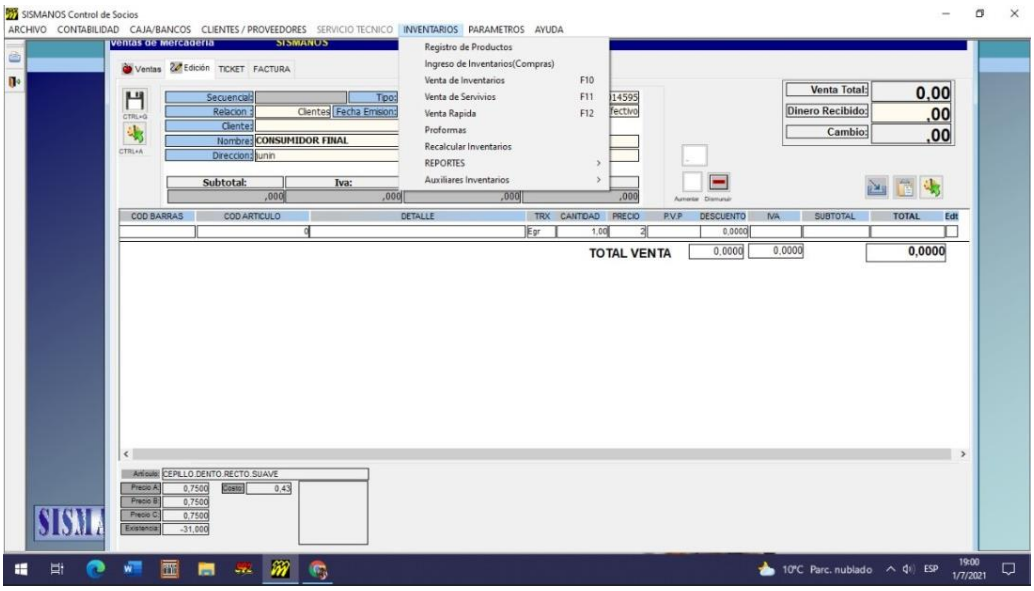


Figura 2. SISMANOS Sistema tecnológico

Fuente. Negocio comercial TUS ENKNTOS

Una vez que se hace la revisión al inventario, se toma la decisión de realizar un nuevo pedido o no, esta decisión se basa en las creencias de la propietaria, es decir, si cree que con la cantidad de existencias en el inventario puede hacer frente a la demanda, entonces no se realiza el pedido, pero si se cree que hace falta existencias, se procede a estimar la cantidad de pedido de manera empírica con respecto a la experiencia de la dueña y a la cantidad de dinero para realizar la compra del pedido.

Los proveedores tienen un sistema que se basa en pre vender sus productos, y con la ayuda de sus colaboradores realizan dos visitas, una para recoger el pedido por un pre vendedor y otra para entregarlo; el negocio cuenta con más de 40 empresas y las principales que lo proveen se encuentran en las ciudades de Quito, Ibarra, Guayaquil y Tulcán.

Tabla 2. Proveedores

CÓDIGO	APELLIDO	NOMBRES	DIRECCIÓN
3	DISTRIBUIDORA GENERAL	DISTRIGEN	GUAYAQUIL AV. RODRIGUEZ CHAVEZ SN. Y AV. PRIM.
5	PROVEEDORA	PRODISPRO	AV RODRIGO DE MIÑO VIA URCUQUI
6	PYDACO	CIA.LTDA.	LOS HUERTOS FAMILIARES-IBARRA
7	DISTRIBUIDORA	DISPAM	SANCHEZ Y CIFUENTES-IBARRA
8	VARIEDADES	DAMAEMMSAFA	BOYACA Y BOLIVAR
9	ECUAQUIMICA	ECUAQUIMICA	TULCÁN
12	CARLOS ARIAS	DISTARIAS	IBARRA HUERTOS FAMILIARES
13	VARIOS	COLOMBIA	IPIALES

14	YANBAL	YANBAL	QUITO
15	BELCORP	ESIKA,LBEL,CYZONE	QUITO
18	VICTORIA	ZOILA VICTORIA	CLEMENTE BALLEN
20	TREVO	JOYAS	GUAYAQUIL
28	DISDRIM	DISDRIM	HUERTOS FAMILIARES
4	ALBA DONOSO	DISCOM NORTE	BARRIO BELLAVISTABALTO VIA CHALTURA
11	CARLOS ARIAS	DISTARIAS	IBARRA HUERTOS FAMILIARES
17	LBEL.CORP.	ESIKA.LBEL.CYZONE	QUITO
19	ITALSTEEL	JOYERIA	BENIGNO MALO
21	RELOJES	GUAYAQUIL	GUAYAQUIL
22	VILA	VELONI	MIAMI
23	AKI	TULCAN	TULCAN
24	CUBRETE	SANDOVAL TEXTILES	QUITO
26	DISTRICOM	PRODUCTOS DE CONSUMO MASIVO	TULCAN
29	LANSEY	RECAMIER	QUITO
30	LA	FABRIL S.A.	VIA.
31	DIMA	DISTRIBUIDORA	PRINCESA PACHA
32	DISTRIBUCIONES	PyP	HERNAN GONZALES
33	QUALA	DISTRIBUISORES	IBARRA
34	CRAFIKLES	S.A.	QUITO
35	OTELO y	FABELL	TULCAN
40	AVON	DEL ECUADOR	VIA SANGOLQUI
42	DANEC	SA.	TULCAN
45	PRODUCTOS HIGIENE	PROHIGIE	ATHUALPA Y LAS PALOMAS
50	HINODE	ECUADOR	MARIANO AGUILERA
51	DISTRIBUCIONES	ALPINOR	IBARRA
55	ABARROTES	GAEL	SUCRE Y VENEZUELA
57	ACCESORIOS	ORTEGA	SANTO DOMINGO
58	BODEGA	MARK	QUITO
61	ENRIQUEZ AYALA	VERONICA ELIZABETH	SUCRE Y VENEZUELA

62	DOUS	INTERNATIONAL	AV. OSWALDO GUAYAS AMÍN
63	FARMAENLACE	COMPAÑIA LIMITADA	CAPITÁN RAFAEL RAMOS E2-210 Y CASTELLI
64	MULTIPRODUCTOS	S.A.	LORENZO DE GARAICOA
65	PROVEKOM	DISTRIBUIDORA	ESPINOSA DEL MONTEROS Y SAN JUAN
66	VARIEDADES CRIS	CRIS	TULCAN

Fuente: Negocio comercial TUS ENKNTOS

De todos los proveedores presentados anteriormente, se procedió a identificar cuáles son los principales para el negocio, al considerar que el total de las compras durante 2020 da un total de más de 1500 dólares:

Tabla 3. Principales proveedores

PROVEEDOR		TOTAL	PARTICIPACIÓN %
FARMAENLACE COMPAÑÍA LIMITADA	$	3.282,58	16%
PROVEEDORA PRODISPRO	$	2.346,01	12%
PYDACO CIA.LTDA.	$	1.936,11	10%
DISDRIM	$	1.933,75	10%
Distribuidora Distrigen	$	1.670,78	10%

Fuente: Negocio comercial TUS ENKNTOS

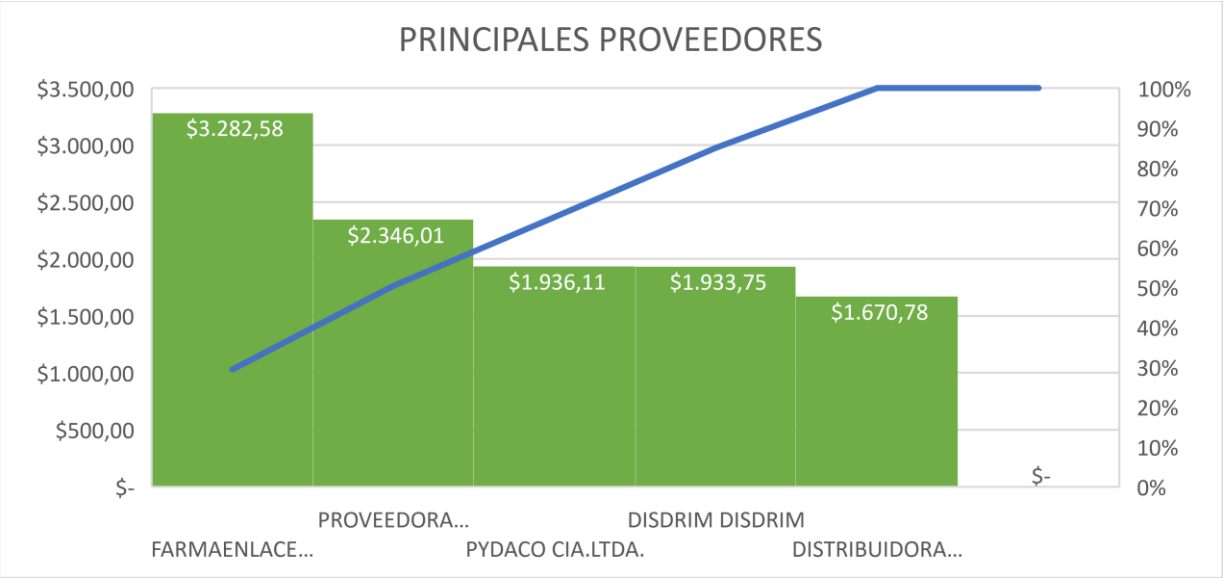


Figura 3. Principales proveedores

Estos datos son tomados de las compras a proveedores totales del año 2020, como se puede observar FARMAENLACE COMPAÑÍA LIMITADA ubicada en la ciudad de Quito, es un proveedor principal para el negocio puesto que debido a la situación actual de la pandemia el negocio se provee más de insumos médicos como es el alcohol, gel antibacterial, mascarillas entre otros; los siguientes proveedores principales localizados en las ciudades de Quito e Ibarra son denominados de esta manera, pues, suministran productos para el aseo y cuidado personal como es jabón de manos, papel higiénico, toallas sanitarias, pañales, desodorantes entro otros, siendo estos productos de uso diario por ende demandados de manera habitual.

Como se menciona en el párrafo anterior los proveedores se encuentran en otras ciudades, por lo que, las vistas de los pre vendedores se efectúan, una vez por semana, dos veces al mes y una vez cada mes, de acurdo a lo planificado por cada proveedor

Después de la visita del pre vendedor la llegada del pedido, depende del cronograma planificado por los proveedores, pero en si el tiempo es de 3 a 5 días. En si el tiempo que tarda en reabastecerse el negocio es la suma del tiempo de visita del pre vendedor y el tiempo de llega del pedido.

Tabla 4. Tiempos de respuesta de los proveedores

PROVEEDOR	VISITA PRE VENDEDOR/MES	ENTREGA PEDIDO/DÍAS
FARMAENLACE COMPAÑIA LIMITADA	2	5
PROVEEDORA PRODISPRO	4	3
PYDACO CIA.LTDA.	1	3
DISDRIM	2	5

DISTRIBUIDORA GENERAL DISTRIGEN	2	3
DIMA DISTRIBUIDORA	1	5
LANSEY RECAMIER	2	5
CRAFIKLES S.A.	1	3
DOUS INTERNATIONAL	2	5
CARLOS ARIAS DISTARIAS	2	5
LA FABRIL S.A.	2	3
VARIEDADES CRIS	2	3
MULTIPRODUCTOS S.A.	2	5
QUALA DISTRIBUISORES	1	5
PRODUCTOS HIGIENE PROHIGIE	4	3
PROVEKOM DISTRIBUIDORA	2	3
ALBA DONOSO DISCOM NORTE	4	3
DISTRIBUIDORA DISPAM	2	3
VARIEDADES DAMAEMMSAFA	2	5
BODEGA MARK	2	3
AKI TULCAN	4	3
DISTRIBUCIONES PyP	2	3
OTELO y FABELL	1	3
ABARROTES GAEL	2	5
DANEC SA.	1	3
Fuente: Negocio comercial TUS ENKNTOS		

El negocio cuenta con más de 2000 productos dentro de la cartera comercial, en la tabla que se muestra a continuación, se presentan los productos con mayor participación en los inventarios del negocio.

Tabla 5. Principales productos

Productos	**Cantidad en Stock**	**Participación**	**Acumulado**

CURITAS.CUREBAND	186	6%	6%
ESMALTE RODHER.VAR.REF.	178	6%	12%
ESMALTE.VOGUE.VAR.REF.10ML	143	5%	17%
PAPEL.HIG.FAMILIA.MEGAROLLO.ENVOL.*1	133	4%	22%
SACH.SHAM.SEDAL-KERATINA.18ML	130	4%	26%
CENTILLOS.VARIAS-REFE.	121	4%	30%
ESMALTE.MASGLO.VARI-REF	117	4%	34%
PAP.HIG.NOBLE*6	113	4%	38%
PEINILLA.BOLSILLO.HOMB.	111	4%	41%
SHAM.SACHET.HEAD-SHOULDERS.LIMP-RENOV.18ML	110	4%	45%
GUANTES.TALLA.L	100	3%	49%
MASCARILLA..QUIRUR.NIÑO	94	3%	52%
LIGAS.COLORES	92	3%	55%
PAPEL.HIG.ELITE.X4	85	3%	58%
DES.SACHET.REXONA.CLINICALMUJER.10G	84	3%	60%
DES.SACHET.REXONA.MEN.CLINICAL.10G	82	3%	63%
LAMINAS-AFEI.GILLETTE.X5	74	2%	66%
DEPILADORA.COLORES	73	2%	68%
DETER.DEJA..SUA.340G	68	2%	70%
SHAM.PANTENE.SACH.18ML	66	2%	73%
CUCHILLA.DEPILADORA	66	2%	75%
PASTA.DENT.COLGATE.TRIP-ACC.MENTA.60ML	63	2%	77%
PASTA.DENT.COLGATE.MAX-PROTEC.60ML	63	2%	79%
TRAT.KERATINA.RECAMIER.30G	60	2%	81%
PAÑITOS.HUM.PANOLINIDUO*200	60	2%	83%
CICLON.FLORAL.200G	60	2%	85%
SHAM.SEDAL.SACHET.CASPA.18ML	59	2%	87%
ESPONJA.MIX.ESTRE.2*1	59	2%	89%
PEINILLA.BEBE	58	2%	91%
DELINEADOR.LAPIZ.RAQUEL	58	2%	93%

TOALL-HIG.NOSOTRAS.BUEN-NOCH.10UND	54	2%	95%
RASURADORA.BIC.CONF.3	52	2%	97%
JAB.LAVATODO-BARR-MEGABLU	51	2%	98%
MASCARILLA.PUNTOS NEGROS6G	50	2%	100%
	2973	100%	

Fuente: Negocio comercial TUS ENKNTOS

Figura 4. Diagrama de Pareto de la participación en los inventarios

Como se puede evidenciar en la tabla, según el diagrama de Pareto, los productos que conforman en el 80% son aquellos para el aseo personal, belleza, limpieza del hogar e insumos médicos siendo estos con mayor participación, el 20% restante son productos de aseo personal, y limpieza del hogar, pero con otras presentaciones que puede ser por la cantidad que contiene estos productos o por las características de los mismos.

Costo total del inventario

Costo de pedido: El costo de pedido se basa en el costo del producto, por lo que, no cuenta con los costos de transporte, preparación. En promedio el negocio realiza compras de 80 dólares por proveedor.

Costos de mantenimiento: El costo de mantener los inventarios es aproximadamente 310 dólares cada mes compuesto entre los costos de:

Tabla 6. Arriendo almacén

Almacén	**Costo**

Arriendo	$ 100
Total	$ 100
Fuente: Negocio comercial TUS ENKNTOS	

Tabla 7. Servicios Básicos

Servicios de Básicos	Costo
Luz Eléctrica	$ 30
Agua potable	$ 20
Internet y Teléfono	$ 50
Total	$ 100
Fuente: Negocio comercial TUS ENKNTOS	

Tabla 8. Seguridad del almacén

Seguridad	Costo
Centinela del Norte Jaenplatinium Cia. Ltda	$ 40
Total	$ 40
Fuente: Negocio comercial TUS ENKNTOS	

Tabla 9. Mantenimiento del Almacén

Mantenimiento del almacén	Costo
Salario al personal de limpieza	$ 60
Insumos de limpieza	$ 10

Total	$ 70

Fuente: Negocio comercial TUS ENKNTOS

4.1.2 Determinar los parámetros críticos de los modelos de gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal".

Conociendo como son los procesos actuales de la gestión de pedidos a proveedores se procede a identificar los paramentos críticos que hacen que la gestión no se desarrolle de una manera óptima, para cumplir con este objetivo se utilizó el mismo diagrama de flujo y se redactó cada uno de los parámetros críticos explicando porque es considerado como crítico.

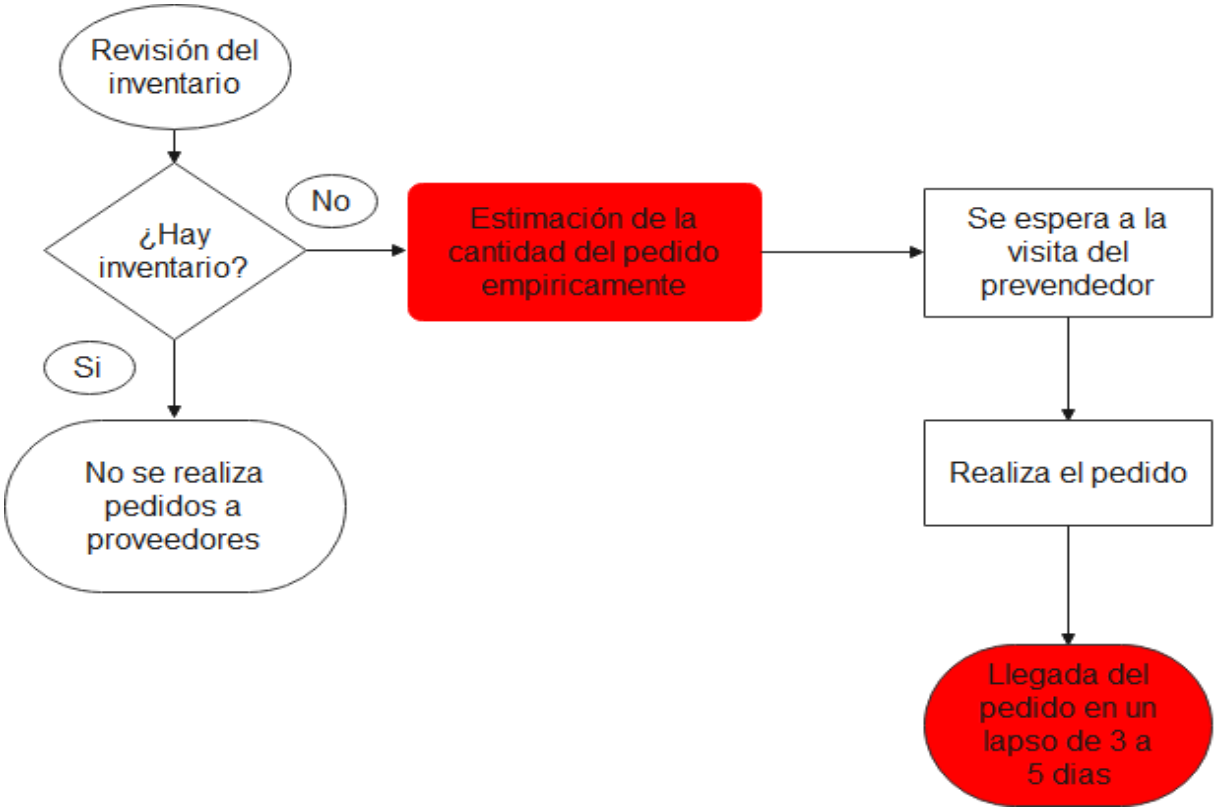


Figura 5. Diagrama de flujo de los procesos de la gestión de pedidos a proveedores - puntos críticos.

Conociendo como se hace el proceso de pedidos a proveedores por parte del negocio, se identificó los siguientes puntos críticos:

- Estimación de la cantidad de pedido, al ser un cálculo empírico, se convierte en un punto crítico porque la cantidad de pedido puede hacer frente a la demanda o no lo puede, en otro caso puede incrementar el costo de pedido y el costo de mantener el inventario o llegar a provocar gastos por deterioro del inventario si este se caduca en poco tiempo; También el negocio al hacerlo de manera empírica no toma en cuenta cual sería la

cantidad de stock de seguridad que puede ser clave para hacer frente a la posible demanda.

- Tiempo de llegada del pedido, como se menciona en el diagnóstico de los pedidos a proveedores, el tiempo que tarda en llegar un pedido es la suma del tiempo que tarde en llegar la visita del pre vendedor más el tiempo en que llega el pedido, siendo este un punto crítico porque al negocio le puede causar rotura del stock.

- La revisión del inventario también puede ser un punto crítico si se lo realiza de manera manual porque esto haría que la gestión de pedidos a proveedores se desarrolle en un periodo de tiempo más prolongado, para el negocio que se ha tomado como piloto no es un punto crítico porque si cuentan con un sistema que les ayuda a que la revisión sea más eficiente. Pero para otros negocios puede ser lo contrario es por eso que para el desarrollo del diseño del modelo de gestión de pedidos a proveedores se tomó en cuenta este punto.

4.1.3 Programar el modelo de base tecnológica para la gestión de pedidos a proveedores con base en los parámetros críticos del modelo utilizando varios lenguajes de programación.

Una vez que se identificó los parámetros críticos de los procesos actuales de la gestión de pedidos a proveedores se procedió a diseñar el modelo. Para la programación del software se utilizaron dos lenguajes de programación, Python para el Backend que hace referencia al código que debe ejecutar el servidor y la conexión con la base de datos y Javascript para el Frontend que se encarga de toda la parte de la Interface del software es decir la parte visible. Para empezar la programación del software primero se estructuro y se creó la base de datos.

4.1.3.1 Creación y Programación de la base de datos

Para la creación de la base de datos se utilizó el lenguaje de programación SQL, debido a que la estructura no demandaba un modelo multidimensional, la programación se realizó con Mysql Workbench. La base de datos fue nombrada como db_inventarios y cuenta con 4 tablas, para proveedores, existencias o inventarios, ventas y configuraciones, a continuación, se describe a cada una.

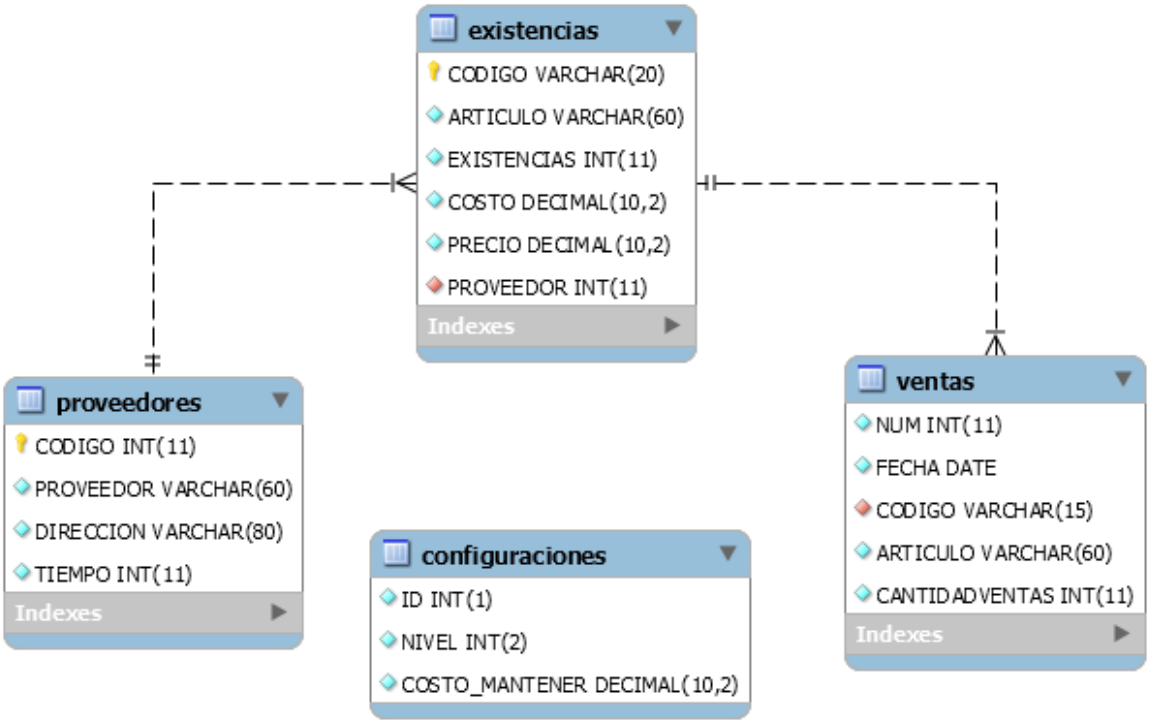


Figura 6. Diagrama entidad relación

Fuente: MySQL Workbench

Descripción de la estructura de Tabla Proveedores en la base de datos

Codigo	Proveedor	Direccion	Tiempo
3	DISTRIGEN	GUAYAQUIL AV..RODRIGUEZ CHAVEZ SN. Y AV...	18
5	PRODISPRO	AV RODRIGO DE MIÑO VIA URCUQUI	11
6	PYDACO	LOS HUERTOS FAMILIARES-IBARRA	11
28	DISDRIM DISDRIM	LOS HUERTOS FAMILIARES-IBARRA	28
63	FARMAENLACE COMPAÑÍA LIMITADA	CAPITÁN RAFAEL RAMOS E2-210 Y CASTELLI	20

Figura 7. Proveedores

Fuente: MySQL Workbench

La tabla ilustrada anteriormente contiene la información de los proveedores y los campos que se consideró en esta tabla son:

Código: Es el código del proveedor, que puede ser asignado por el encargado del negocio, este campo es numérico por lo que se utilizó el tipo de dato INT con un máximo de 11 cifras numéricas enteras, se cómo clave primaria que hace referencia al identificador único de los datos de esta fila.

Proveedor: Hace referencia al nombre de la empresa proveedora para ello se utilizó el tipo de dato VARCHAR con la posibilidad de ingresar 60 caracteres.

Dirección: Es la ubicación proveedor, al ser un texto se empleó el tipo de dato VARCHAR con la posibilidad de insertar 60 caracteres.

Tiempo: Hace mención al tiempo de respuesta por parte del proveedor para abastecer el negocio, al ser un valor numérico también se utilizó el tipo de dato INT con un máximo de 11 cifras numéricas.

Descripción de la estructura de Tabla Existencias en la base de datos

Codigo	Articulo	Existencias	Costo	PRECIO	Proveedor
780492000...	ACON. DETOX COCO ALOE 900ml*10	1	2.33	2.76	5
786100312...	NOS.INV.RAPIGx10 PREC.ESPECIAL*24	12	1.49	1.76	5
0310200408	0310200408 PRIMAVERA SERY. CLASICA C 100...	6	0.29	0.34	6
780492000...	ACON. DETOX HIERBAS S. 900ml.*10	1	2.33	2.76	5
780492000...	ACON. DETOX TE VERDE 900ml.*10	1	2.33	2.76	5
780492000...	ACON.SIN SAL AGUACATE DP 900ml*10	1	2.31	2.73	5
780492000...	ACON.SIN SAL COC.JAZM.DP 900ml*10	1	2.31	2.73	5
780492000...	ACOND.SIN SAL GRANADA DP 900ml*10	1	2.31	2.73	5
67369177	AXE DEO AER BODY SPRAY COLUSION 12X112G	10	3.19	3.78	3
67S60247	AXE DEO BS ICE CHILL 6X2X97G/150ML	5	5.44	6.44	3
0145001204	BUENDIA CAFE DESCAFEINADO 170 GR 111812	1	7.99	9.47	6
1059000301	CRUZ BLANCA ALCOHOL 49.5% SOOML 131824	6	1.46	1.73	6
1059900302	CRUZ BLANCA ALCOHOL 69.5% 1000ML 141412	6	2.65	3.14	6
1059000303	CRUZ BLARCA ALCOHOL 49.52 AOUOML 14134	4	7.19	8.52	6
1012000940	CRUZ ELANCA SANITIZANTE 20LTS 18481	1	22.5	26.6	6
67415847	DEJA DT UQ SUAVIDAD B PRIMAV BOT 6X1.9L	2	5.09	6.03	3
566502	DEJA JAB UQ FLORAL 6X1.9L	2	5.09	6.03	3
84164917	DOVE BABY JAB HIDRAT ENRIQUEC 16X3X75G	6	2.7	3.2	3
67691720	DOVE SUPER AC 60 12X2X10X12ML	1	1.87	2.21	3

Figura 8. Tabla existencias

Fuente. MySQL Workbench

La tabla existencias registra los inventarios que ingresan, se consideró los campos.

Código: Es el código de barras que viene Integrado en los productos, al ser códigos extensos y combinados con letras y números, se decidió utilizar el tipo de dato VARCHAR con la posibilidad de insertar hasta un máximo de 20 caracteres, es la clave primaria que quiere decir el identificador único de los datos de esta fila.

Artículo: En este campo va el nombre del producto, que por ser un texto se aplicó el tipo de dato VARCHAR con un máximo de 60 caracteres.

Existencias: En este campo se registra el número total de productos que ingresan al almacén.

Costo: Es el precio unitario del producto, como puede ser un valor decimal se empleó el tipo de dato Decimal con parámetros de 10 números enteros y 2 cifras decimales.

Precio: Es el pecio de venta del producto, al igual que el costo, este campo también puede ser un valor decimal, por lo que se empleó el tipo de dato Decimal con parámetros de 10 números enteros y 2 cifras decimales.

Proveedor: En este caso este campo recibe el código del proveedor permitiendo que sea una Foreign Key y que esta tabla se pueda unir con la tabla proveedores, la unión es many to one

que quiere decir que varios productos son abastecidos por un solo proveedor, la restricción que se aplico es el ON DELETE CASCADE que hace posible eliminar las filas de la tabla de existencias, cuando se eliminan las filas de la tabla de proveedores.

Descripción de la estructura de Tabla Ventas en la base de datos

Num	Fecha	Codigo	Articulo	CantidadVentas
86	2021-07-04	7702354...	NUTRIBELA TRAT.BIOKERAT. 300mI.*12	2
58	2021-07-05	84159925	SEDAL SH CASPA CONTROL 12X340ML	3
88	2021-07-05	0310100...	ECO PH 4 ROLLOS 1356R 194412	2
85	2021-07-06	7702354...	NUTRIBELA TRAT.BIOKERAT. 300mI.*12	1
84	2021-07-07	7702354...	NUTRIBELA TRAT.BIOKERAT. 300mI.*12	2
64	2021-07-08	84159925	SEDAL SH CASPA CONTROL 12X340ML	3
65	2021-07-09	84159925	SEDAL SH CASPA CONTROL 12X340ML	4
83	2021-07-09	7702354...	NUTRIBELA TRAT.BIOKERAT. 300mI.*12	2
66	2021-07-10	84159925	SEDAL SH CASPA CONTROL 12X340ML	4
71	2021-07-11	84159925	SEDAL SH CASPA CONTROL 12X340ML	2
72	2021-07-12	84159925	SEDAL SH CASPA CONTROL 12X340ML	3
74	2021-07-14	84159925	SEDAL SH CASPA CONTROL 12X340ML	3
82	2021-07-14	7702354...	NUTRIBELA TRAT.BIOKERAT. 300mI.*12	1
70	2021-07-15	84159925	SEDAL SH CASPA CONTROL 12X340ML	4

Figura 9. Tabla ventas

Fuente. MySQL Workbench

La tabla ventas como bien dice su nombre registra las ventas diarias de cada uno de los productos, para esta se tomó en cuenta los campos:

Num: Es el campo que se registra el número de venta, por lo que, es del tipo de dato INT y es de auto incremento.

Fecha: En este campo se ingresa la fecha de la venta para ello se utilizó el tipo de dato DATE que permite ingresar la fecha.

Código: Contiene el código del producto vendido por lo que lleva el mismo tipo de dato del campo código de la tabla existencias.

Artículo: Es el campo donde se registra el nombre del producto vendido, al ser un texto lleva el tipo de dato VARCHAR con un máximo de 60 caracteres.

Cantidad Ventas: Como bien dice su denominación se registra la cantidad de productos vendidos y al ser un valor numérico se utilizó el tipo de dato INT con un máximo de 11 cifras numéricas.

Descripción de la estructura de tabla configuraciones en la base de datos

ID	NIVEL	COSTO_MANTENER
1	90	310

Figura 10. Tabla configuraciones

Fuente. MySQL Workbench

Esta tabla denominada como Configuraciones fue creada para insertar los parámetros que necesita el modelo matemático que en los próximos párrafos se estará explicando de la manera más minuciosa, los campos considerados fueron:

Id: Registra un identificador de la fila y se decidió utilizar el tipo de dato INT con un máximo de 1 carácter, este campo ayudara a identificar la fila para actualizar los datos.

Nivel: Este campo hace mención al nivel de confianza con el que se va a manejar el negocio que puede ser del 80% hasta 99%, por sus características se utilizó el tipo de dato INT con un máximo de 2 cifras numéricas.

Costo_Mantener: En este campo se registra el costo mensual de mantener el inventario en el almacén de igual manera es un valor numérico, pero puede ser decimal es por eso que se utilizó el tipo de dato Decimal con la posibilidad de insertar hasta 10 cifras numéricas enteras y 2 cifras decimales.

Creación e inicialización del servidor

Para la creación del servidor se utilizó Flask, que es un framework desarrollado para el lenguaje de programación Python con el cual se puede levantar servidores de una manera sencilla y para la investigación fue levantado en el puerto local 3398, el servidor es la parte del software que se encarga de conectarse con la base de datos y de realizar operaciones para insertar y traer datos, que posteriormente son mostrados en la Interface, otra actividad es ejecutar operaciones matemáticas para pronosticar la demanda promedio del Modelo de Regresión Lineal Simple, calcular la cantidad óptima de pedido con el Modelo EOQ, y la cantidad óptima de inventarios con el Modelo de la Política de Revisión continua.

Conexión de la base de datos con el servidor

Para que el servidor se conecte con la base de datos fue necesario utilizar el módulo flask_mysql de la librería MySQL permitiendo tener las herramientas para que se realice la conexión y se pueda trabajar con la base de datos. La base de datos fue levantada en un servidor local esto quiere decir que el computador se convierte en servidor que va a almacenar la información, utilizando Apache que es un servidor web HTTP de código abierto, una vez se tiene estos requerimiento se procedió a programar la conexión mediante los parámetros de configuración que requiere el módulo flask_mysql, entre estos se encuentra el host que es localhost, el puerto donde se levanta la base de datos, el usuario que tiene acceso a la base, la contraseña y el nombre de la base de datos que es db_inventarios.

Insertar datos a la base desde el servidor

Para insertar en la base de datos se establecieron rutas específicas que identifican a la tabla donde insertar los datos; Además serán de ayuda, debido a que son las responsables de Interactuar con la Interface. Para realizar las peticiones de insertar se utilizó el método HTTP POST y sentencias SQL como es el INSERT INTO() VALUES()

Insertar Proveedores

El código que se estableció permite que el negocio pueda registrar a sus proveedores en la tabla correspondiente que se creó en la base de datos, primero se codifico una ruta especifica en el servidor que contiene una función para que pueda ejecutar la orden de insertar un proveedor.

Insertar Existencias

El código para esta sección inserta los productos en tabla de Existencias en la base de datos, para ello se implementó la misma metodología para insertar los proveedores, pero se agregó un condicional en el que verifica que, si existe el producto registrado en la tabla de Existencias creada en la base de datos, entonces se actualiza la cantidad de existencias sumando la cantidad actual con la cantidad entrante.

Insertar Ventas

Al igual que los párrafos anteriores el código sirve para insertar datos en la tabla de ventas, además de ejecutar esta sentencia, el código permite actualizar la cantidad de productos en los inventarios, por esta razón se utilizó dos funciones, la primera inserta los datos en la tabla ventas y la segunda actualiza los datos en la tabla existencias para hacerlo el código verifica que exista dicho producto vendido y resta la cantidad existente en inventarios con la cantidad vendida.

Insertar Configuraciones

Los datos que se requieren en esta tabla solo comprende una fila por lo que solo se insertaran una sola vez y las otras ocasiones solo actualizaran los datos es por eso que de utilizo un condicional que verifica que si en existen datos en la tabla entonces los insertara caso contrario los actualizara.

Obtener información de la base de datos

Una vez almacenados los datos en cada una de las tablas de la base de datos, se los debe obtener, para procesarlos y mostrarlos en una Interface más amigable, para realizar esta petición se recurrió al método de petición HTTP GET que permite obtener la información, cabe mencionar que en el código no se registra el método, debido a que el Framework Flask lo utiliza por defecto, de igual forma se usó de sentencias SQL como es el SELECT FROM.

Obtener toda la información de una tabla de la base de datos

Al igual que los códigos establecidos para insertar se estableció una ruta especifica en el servidor que permite obtener todos los datos de una tabla en particular en la base de datos.

Obtener toda la información de dos tablas de la base de datos

Se creó esta sección de código con la sentencia SQL JOIN para poder unir las informaciones de las tablas existencias y proveedores, con esto hacer más ágil el proceso de obtener los datos para calcular los modelos de inventarios.

Obtener un dato especifico de una tabla o de dos tablas de la base de datos

El modelo en ciertas ocasiones necesita obtener un solo dato para ello se estableció otras rutas las cuales contienen un identificador que permite encontrar el dato que se requiere de una tabla en específico. Por ejemplo, para obtener un producto de la tabla de existencias se utiliza como identificador el código de barras.

Programación de los modelos para pronostico e inventarios

En esta parte se estableció una ruta para obtener un dato especifico de dos tablas que son las de existencias y proveedores, de esta manera obtener los datos necesarios para cada uno de los modelos de una forma ágil.

Pronostico de demanda

Para elegir el método de pronóstico de la demanda se procedió a identificar el indicado entre los métodos, de regresión exponencial, logarítmica y lineal simple. Para ello se basó en el coeficiente de determinación R^2.

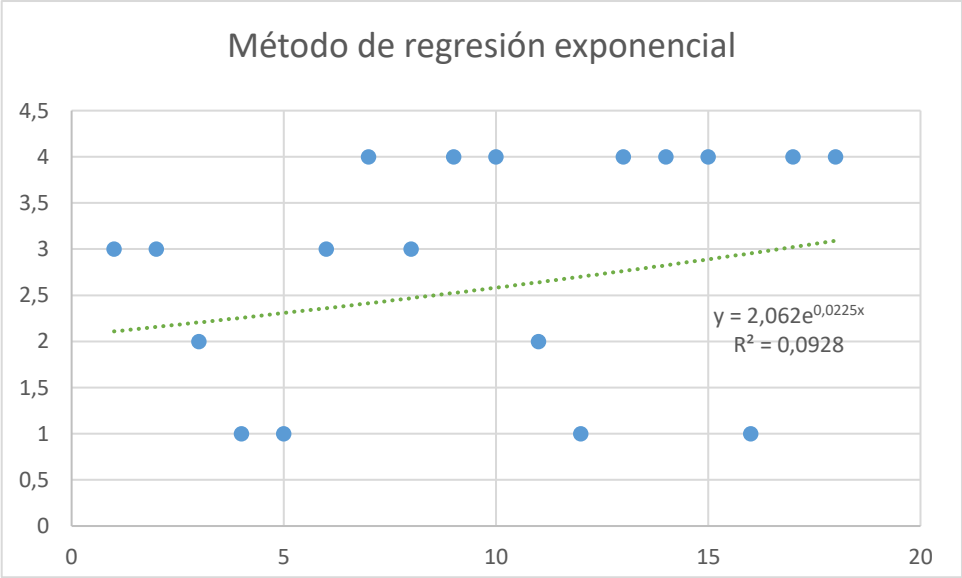


Figura 11. Método de regresión exponencial

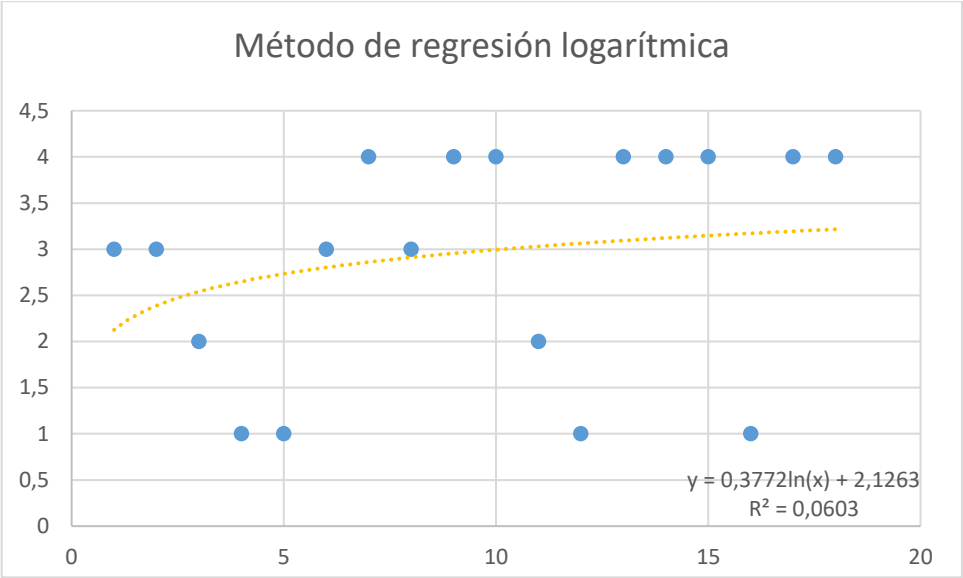


Figura 12. Método de regresión logarítmica

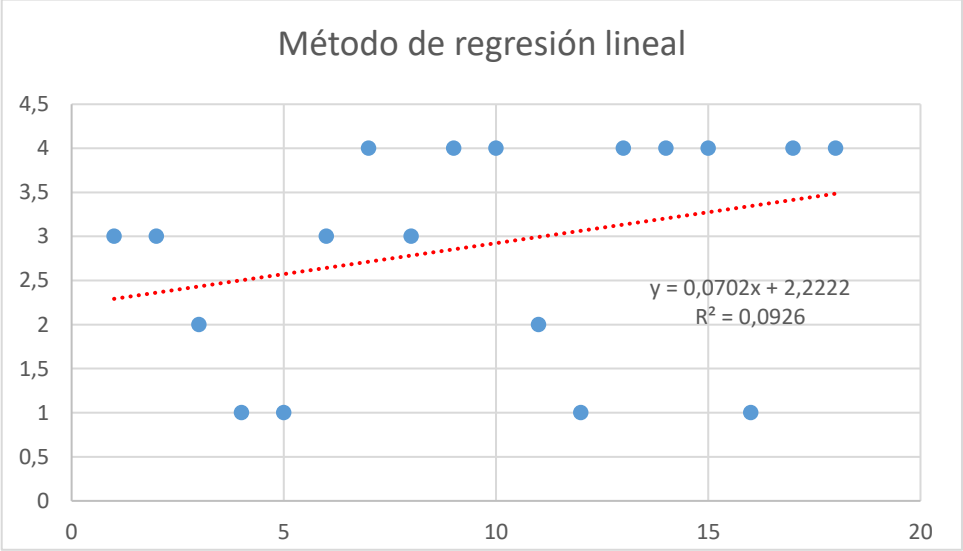


Figura 13. Método de regresión lineal

Como se observó en las figuras anteriores el método indicado es la regresión lineal siendo R^2 el valor mayor, por lo que el pronóstico es más efectivo que implementar los otros métodos.

Programación de Modelo de Regresión Lineal Simple.

Para programar el código que pronostique la demanda se necesitó importar el módulo Math con el cual permite trabajar con operaciones más complejas, para conocer la demanda futura se necesitó obtener los datos de la tabla ventas, pero como el pronóstico se lo hace por producto la ruta establecida tiene el identificador que permita encontrar un producto vendido en específico y además como otro parámetro en la ruta está el tiempo de respuesta del proveedor, con esto se obtendrá los últimos datos o últimos productos vendidos durante ese lapso de tiempo. Con los datos obtenidos se procedió a programar la fórmula del modelo, misma que se observa en la Figura 11.

$$y = a + bx$$

Figura 14. Formula del Modelo de Regresión Lineal Simple

Fuente. Heizer, J., y Render, B. (2009). Principios de Administración de Operaciones. Pearson Educación de México, S.A. de C.V.

Siguiendo el proceso para calcular el pronóstico de la demanda primero se estableció el código para obtener b y a, para ello se usó las fórmulas que se muestran en la Figura 12 y 13 respectivamente.

$$b = \frac{\sum xy - \frac{\sum x \sum y}{n}}{\sum (x)^2 - \frac{(\sum x)^2}{n}}$$

Figura 15. Fórmula para obtener "b"

Fuente. Heizer, J., y Render, B. (2009). Principios de Administración de Operaciones. Pearson Educación de México, S.A. de C.V.

$$a = \bar{y} - b\bar{x}$$

Figura 16. Fórmula para obtener "a"

Fuente. Heizer, J., y Render, B. (2009). Principios de Administración de Operaciones. Pearson Educación de México, S.A. de C.V.

Como se observa en la formulas se tiene dos variables “x” que para el software es el conjunto de los índices de la tabla ventas y “y” es el conjunto de las ventas diarias. Cada elemento de las formulas se calculó por aparte como variables diferentes. Ahora los resultados fueron remplazados en la fórmula que se muestra en la Figura 9 y en este caso la variable “x” de la formula hace referencia al periodo. Como el modelo trabaja con datos diarios de igual forma el pronóstico es el promedio diario.

Modelo de Inventarios de la Política de Revisión Continua

En este caso los datos que se requieren son los que se encuentran en la tabla de configuraciones, que son, el nivel de confianza, la desviación estándar de las ventas, el tiempo de respuesta del proveedor y por último se requiere la demanda que se pronosticó anteriormente. Para obtener el nivel de confianza se estableció una función independiente que devuelve el valor que se encuentra en la columna Nivel que es un valor en porcentaje y sirvió para buscar el valor de z dentro de un conjunto de datos que se estableció previamente donde se encuentra el nivel y el z correspondiente. Al ser la desviación estándar un valor estadístico se importó el módulo statistics que provee de herramientas para hacer cálculos estadísticos. Al tener listos los datos en seguida se estableció el código para calcular la demanda promedio, stock de seguridad y la política óptima de inventarios.

D = Demanda pronosticada

σ = Desviación estándar de la demanda diaria

Z = Número de desviaciones estándar

L = Tiempo de respuesta

$$\mu L = D \text{ x } L$$

Figura 17. Fórmula para calcular la Demanda Promedio

Fuente. Heizer, J., y Render, B. (2009). Principios de Administración de Operaciones. Pearson Educación de México, S.A. de C.V.

$$ss = Z * \sqrt{\sigma^2 \; x \; L}$$

Figura 18. Fórmula para el cálculo del Stock de Seguridad

Fuente. Heizer, J., y Render, B. (2009). Principios de Administración de Operaciones. Pearson Educación de México, S.A. de C.V.

$$Politica\ inventario\ optimo = \mu L + ss$$

Figura 19. Fórmula para calcular Política de inventario optimo

Fuente. Heizer, J., y Render, B. (2009). Principios de Administración de Operaciones. Pearson Educación de México, S.A. de C.V.

Cabe mencionar que en el código se estableció condiciones, que si la venta del producto es mayor igual a 2 entonces se procede a calcular los modelos anteriormente explicados, pero si no es el caso, se muestra un mensaje que dice "No hay suficientes ventas"

Modelo EOQ Cantidad Óptima de Pedido

El modelo EOQ permite conocer la cantidad óptima de pedio al implementar la siguiente formula

D = Demanda

S = Costo de ordenar

H = Costo de mantener

$$.Q^* = \sqrt{\frac{2DS}{H}}$$

Figura 20. Formula del Modelo EOQ

Fuente. Heizer, J., y Render, B. (2009). Principios de Administración de Operaciones. Pearson Educación de México, S.A. de C.V.

Para calcular la cantidad óptima de pedido se necesitó establecer los códigos para obtener los datos del Costo de ordenar que se encuentra en la tabla existencias y el costo de mantener que se aloja en la tabla configuraciones, la demanda implementada para este cálculo es la que se pronosticó anteriormente. Para hacer frente a la demanda esperada y cumplir con la política óptima de inventario se aplicó la siguiente formula que permite conocer la cantidad total óptima a pedir.

CSP = Candad de stock por producto

$$CTOP = \frac{Politica\ inventario\ optimo\ -\ CSP}{Q^*}\ \text{x}\ Q^*$$

Figura 21. Formula de la cantidad total a pedir

La primera parte de la formula permite calcular la cantidad de órdenes para cumplir con la demanda y con la política óptima de inventarios, y como el pedido se realiza una vez cada lapso de tiempo, se multiplica por la cantidad óptima de pedido, de esta manera se conoce la cantidad total a pedir.

Costo de imprentar el modelo

Para conocer cuál sería el costo total al implementar el modelo se utilizó la fórmula que se muestra en la Figura 19. En esta fórmula se consideró:

$\boldsymbol{Q^*}$**=** CTOP

D = Política de inventario optimo

$$\frac{Q^*}{2}H + \frac{D}{Q^*}S + Hss$$

Figura 22. Formula del Costo Total del Modelo

Fuente. Heizer, J., y Render, B. (2009). Principios de Administración de Operaciones. Pearson Educación de México, S.A. de C.V.

Antes de proceder a programar la Interface se importó un módulo en el servidor llamado flask_cors que permite conectar el servidor con la Interface y con esto generar Internación entre las dos partes.

Programación de la Interface

Para la Interface como se mencionó anteriormente el lenguaje de programación utilizado es JavaScript, pero detrás de este se encuentra otros dos leguajes como son, HTML que es un lenguaje de marcado que permite maquetar todos los elementos que van a formar la parte visible del software, y CSS que es el lenguaje para estilizar los elementos maquetados; mientras que JavaScript va a permitir tener una Interacción amigable entre el software y el usuario

Al crear la Interface también se decidió utilizar un Framework, como es VUE JS que permite trabajar de manera más eficiente ya que reúne a los tres lenguajes señalados en el párrafo anterior y además las características del software no requirió de una tecnología más avanzada.

VUE JS está estructurado principalmente por Vistas que son la parte visible, Componentes que son los elementos que componen las vistas y Rutas que son los enlaces para acceder a cada una de las vistas. Para empezar a programar la Interface VUE JS crea un archivo denominado App.vue y otro de nominado main.js que son la parte que permite inicializar el software.

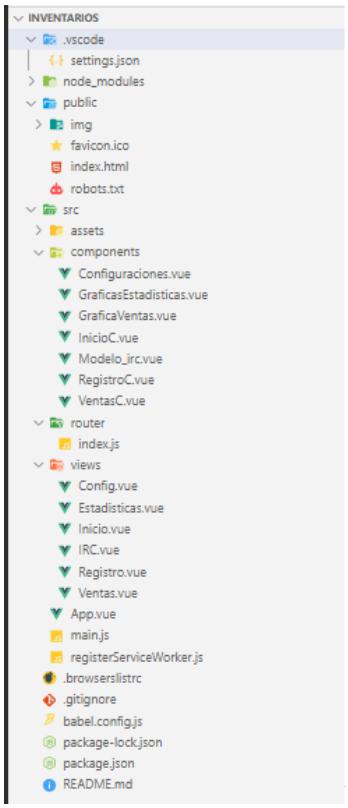


Figura 23. Estructura para la programación de la Interface

El código del archivo App.vue contiene los elementos HTML y el código de estilizado que es la parte que se encuentra entre las etiquetas <style>. Este código también aloja a las rutas con las etiquetas <router-link to="/"> que muestra una clase de botones que permiten acceder a una vista determinada que puede ser la vista de Inicio, Registro, Ventas, Modelo de IRC, Estadísticas y Configuraciones. A continuación, se explicará los procedimientos utilizados para cada una de las vistas.

Vista Inicio

Esta es la vista que se muestra en el instante que se abre el software para esta vista se importa el componente Inicio y para este caso, está estructurada por elementos HTML que permiten mostrar el componente y funcionalidades de JavaScript que importan el componente.

Componente InicioC

Este componente permite realizar una petición al servidor para traer los datos de las existencias. Para realizar esta acción se maqueto un elemento HTML que es denominado input donde se ingresa el código de barras, en la parte derecha se maqueto un botón que con programación en JavaScript realiza la acción que consiste en una petición al servidor para que se obtenga el producto solicitado de la base de datos, también se maqueto etiquetas que muestran el tipo de datos que se obtendrán, como se observa en la Figura 21. Cuando el servidor devuelve el dato se muestran los datos tal y como se ilustra en la Figura 22, esto sucede ya que, el software dinámicamente maqueta 5 Inputs donde se puede observar la fecha actual, el código del artículo, el nombre, el precio y el ultimo está en vacío debido a que es un espacio para insertar la cantidad de productos que se venden o salen del almacén.

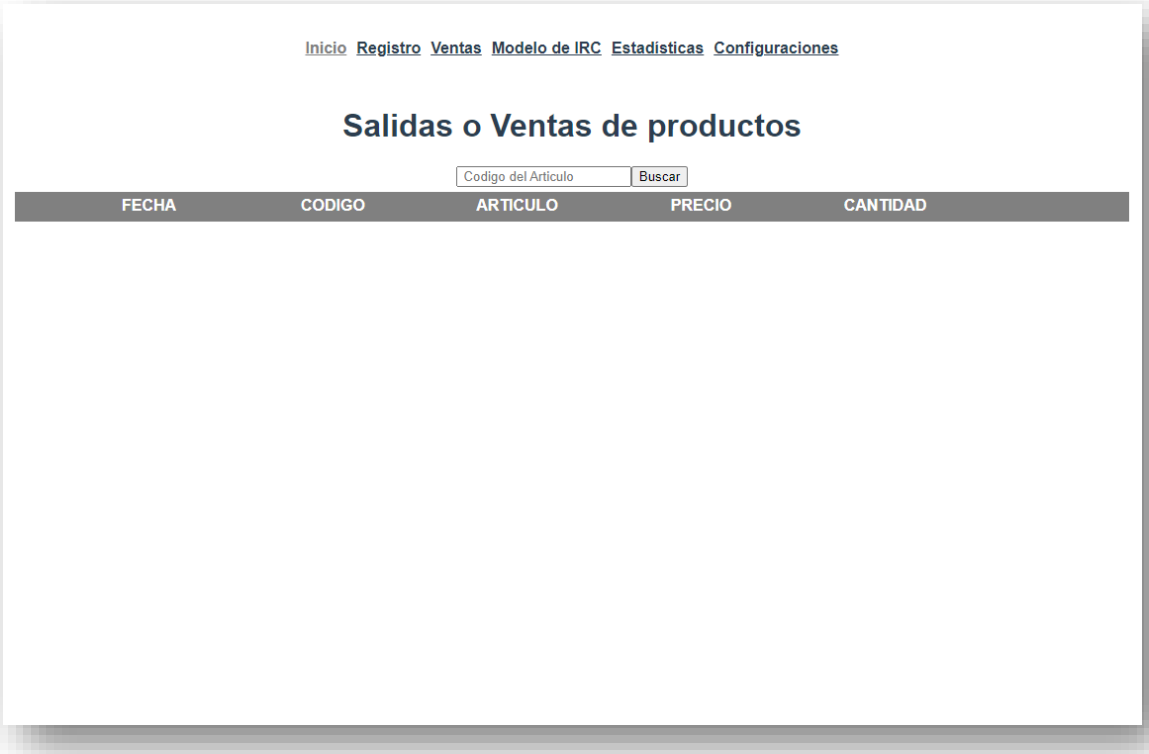


Figura 24. Vista inicio

Figura 25. Vista inicio después de la petición al servidor

Vista Registro

En esta sección del software se creó campos para registrar proveedores e inventarios que ingresan al negocio, de la misma manera está estructurada por elementos HTML que permiten mostrar el componente y funcionalidades de JavaScript que importan el componente Registro.

Componente Registro

Cuando se muestra la vista Registro, el componente permite observar al instante los campos maquetados de un formulario con elementos HTML para insertar los datos que se requieren para registrar un nuevo producto o actualizar la cantidad de existencias si ya se encuentra en stock, para esto se maqueto un botón que con programación en JavaScript permite realizar la acción que ejecuta una petición al servidor para registrar o actualizar el producto en la base de datos y en la parte inferior de estos se crearon etiquetas que muestran el nombre del dato de las que se muestra al revelar los inventarios actuales obteniéndolos de la tabla existencias, y presentándolos en una tabla . Pero en esta vista no solo se registra productos, si no, también se puede registrar a los proveedores, para ello se creó dos botones que permitan acceder al registro del producto, y otro para los proveedores, ilustrando un el formulario que contiene los campos solicitados para el registro de un nuevo proveedor y también cuenta con el botón que ejecuta la petición al servidor y registra al nuevo proveedor con la información correspondiente o actualiza la información y al igual que los productos se muestra la información de los proveedores actuales organizados en una tabla. Los mencionado se indica en siguientes Figuras.

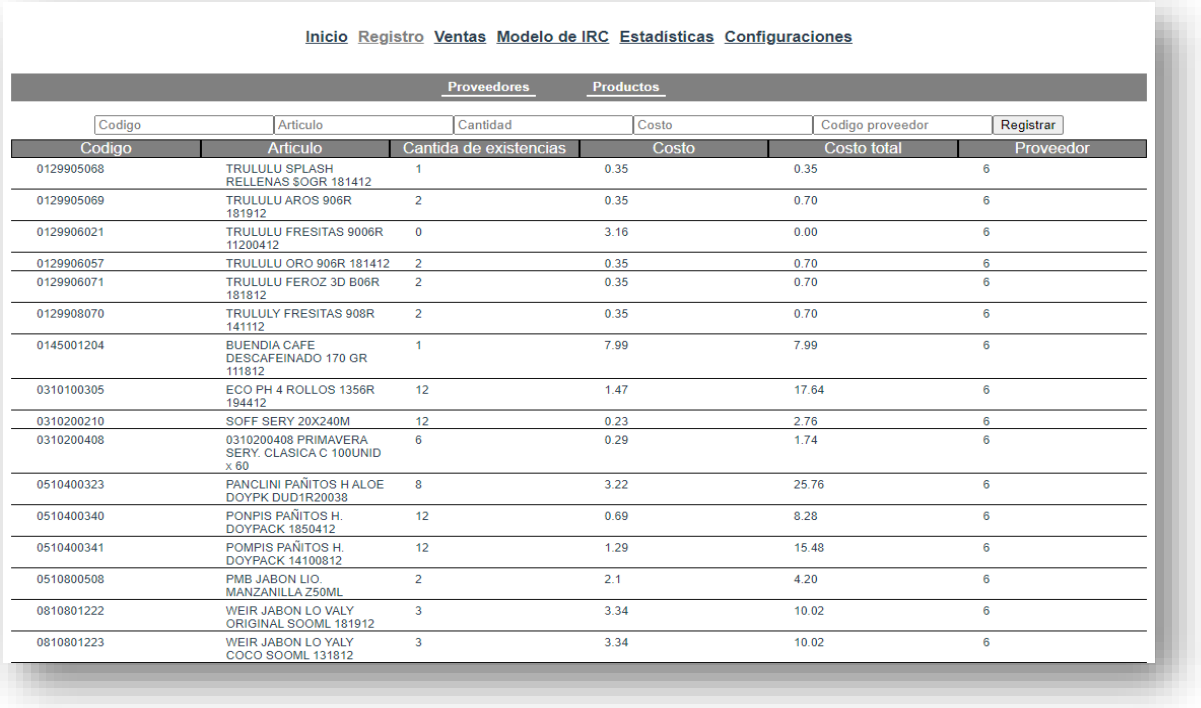


Figura 26. Vista registro producto

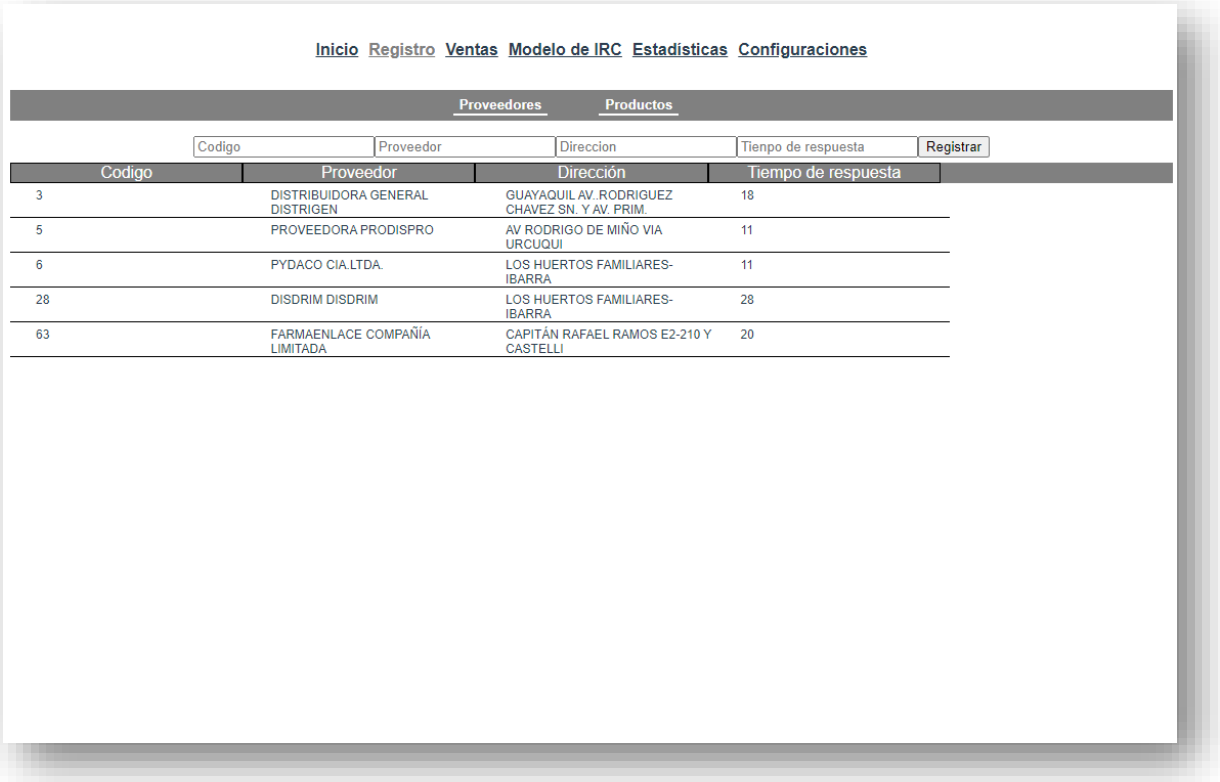


Figura 27. Vista registro proveedor

Vista Ventas

En esta vista se muestran todas las ventas de los productos realizadas por día, de la misma forma que las anteriores vistas, se estructuró con elementos HTML para maquetar y JavaScript para importar el componente denominado VistaC

Componente VistaC

Este componente tiene la principal función de realizar una petición al servidor para obtener los datos de la tabla ventas y para mostrarlos se maqueto etiquetas que indiquen el nombre del dato que se obtiene y se maqueto elementos HTML Div estilizándolos con CSS en forma de tabla y de esta manera organizar la información tal y como se muestra en la siguiente figura.

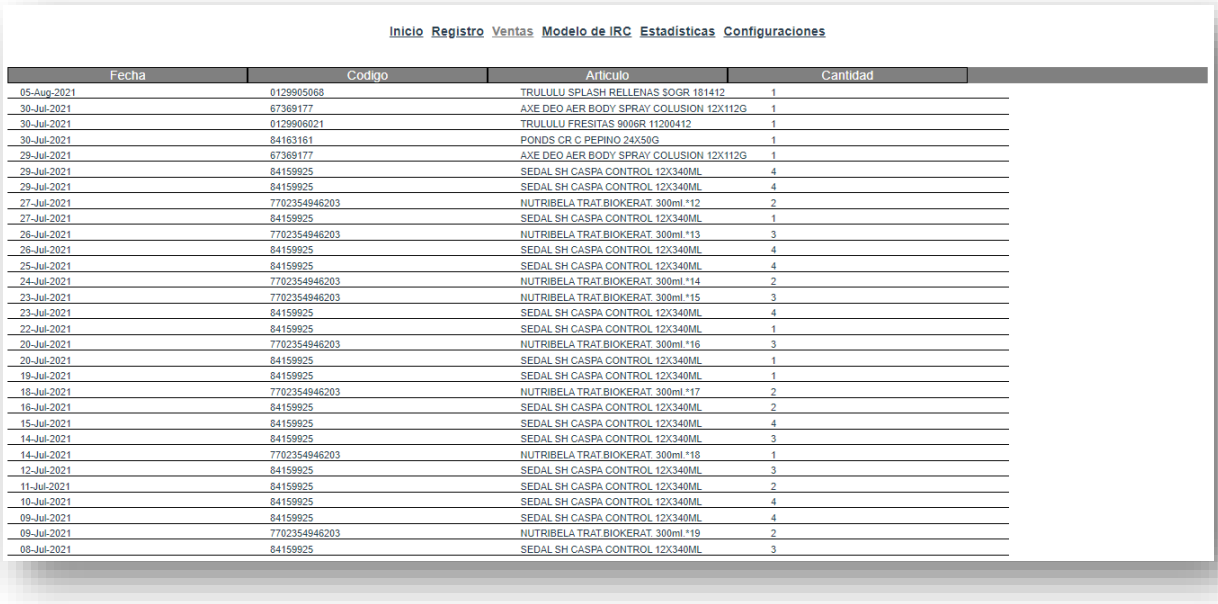


Figura 28.vista ventas

Vista Modelo de IRC

En esta vista se muestra los resultados de los modelos que ejecuta el servidor por cada uno de los productos, para ello se estructuro maquetación de elementos HTML y se codifico JavaScript para importar el componente Modelo_irc

Componente Modelo_irc

Este componente ejecuta una petición al servidor para que se muestre información de los productos junto a los resultados del modelo de Regresión Lineal Simple que se refiere el pronóstico de demanda diaria esperada, modelo de Inventarios de la Política de Revisión Continua que corresponden a la demanda total esperada, stock de seguridad y la política óptima de inventario, modelo EOQ que corresponde a la cantidad óptima promedio de pedido, la cantidad total de pedido y por ultimo muestra el costo logístico del modelo. Al momento de abrir esta sección a parecen los productos que necesitan ser reabastecidos, como se muestra en la Figura 26, ya sea porque se agotaron las existencias o porque es menor igual al stock de seguridad. Se creó un botón que cambia entre la vista donde se observa todos los productos y la vista de los productos que se requieren; Cuando se muestran los todos los productos como se evidencia en la Figura 27, el servidor ejecuta las operaciones considerando las condiciones establecidas que si las ventas son mayor igual a 2 entonces se calcula los modelos, pero si no es así se muestra el mensaje “No hay suficientes ventas”**;** Al igual que la anterior vista los datos son mostrados en una tabla con las respectivas etiquetas y fue maquetada con elementos HTML Div y estilizada con CSS en forma de tabla.

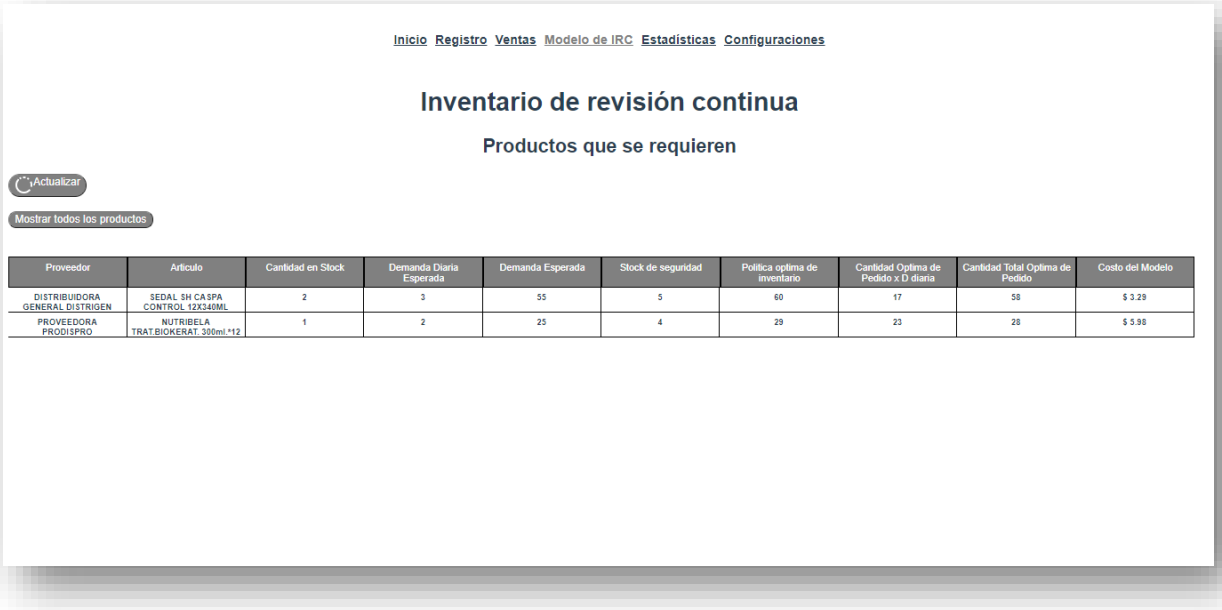


Figura 29. Vista modelo de irc-productos que se requieren

Figura 30. Vista modelo de irc-todos los productos

Vista Estadísticas

Esta vista se creó para mostrar una gráfica que evidencie el comportamiento de las ventas en el tiempo, la vista está estructurada por elementos HTML y por JavaScript, para maquetar e importar el componente GaficaEstadisticas.

Componente GaficaEstadisticas

Este es el componente responsable de obtener los datos de ventas y graficarlos, por ello se maqueto un elemento HTML Input donde se ingresa del nombre del producto y con un botón maquetado a la derecha, se realiza la petición al servidor para obtener los datos de ventas y para graficar, se utilizó el cdn de Chart.js que es un módulo que brinda herramientas necesarias para hacerlo, los parámetros que necesita son los datos para el eje X y para el eje Y, en este caso se tomó como datos las fechas de ventas y la cantidad de ventas respectivamente. También en la parte derecha de la gráfica se muestra una tabla que indica la fecha y la cantidad de ventas. Lo expuesto anteriormente se evidencia en la Figura 28.

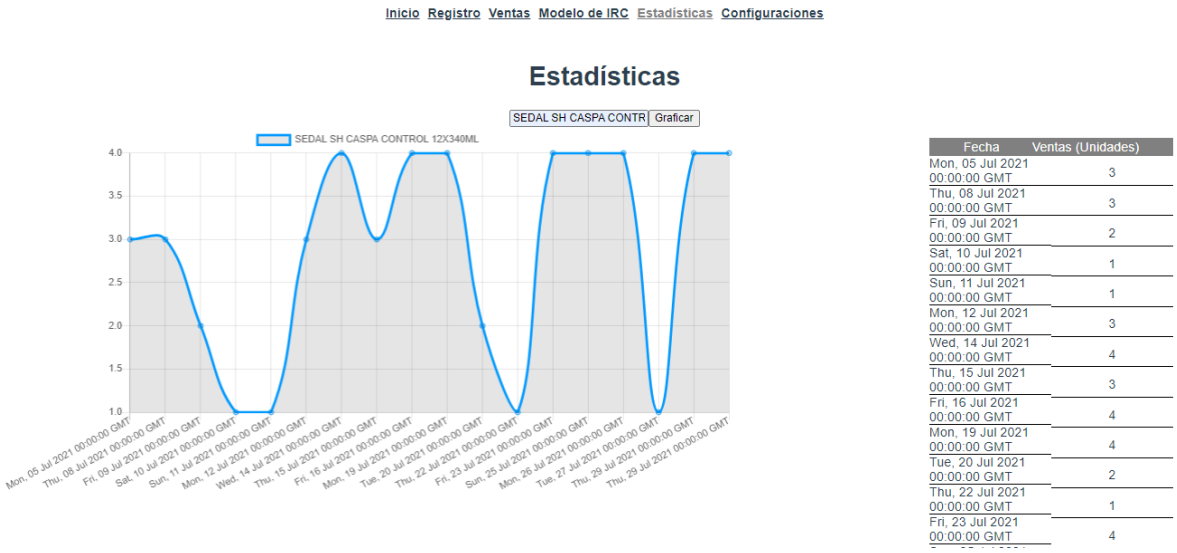


Figura 31. Vista estadísticas

Vista Configuraciones

Aquí se muestran los parámetros solicitados por el software para poder ejecutar los cálculos de los modelos previamente expuestos. Para ello se importó el componente Configuraciones mediante código JavaScript y se maqueto con elementos HTML.

Componente Configuraciones

Este componente muestra un pequeño formulario con dos campos uno para ingresar el nivel de confianza y otro para el costo total de mantener al mes, además la vista indica una información adicional que es el costo de mantener por producto; Por defecto el software coloca un nivel de confianza del 90%, como se indica en la Figura 29.

Figura 32. Vista configuraciones.

Diagrama del Funcionamiento del Software, en síntesis

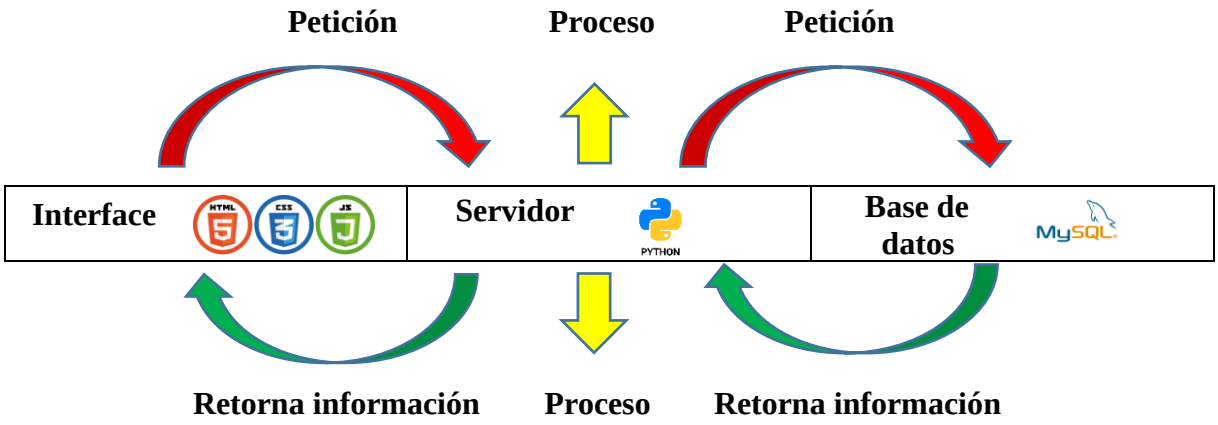


Figura 33. Diagrama del funcionamiento del software

4.1.4 Simular los procesos de gestión de pedidos a proveedores con la implementación del modelo diseñado.

Al terminar la programación del software se procedió a simular los distintos procesos que se programaron, empezando por insertar datos en la tabla proveedores y en la tabla existencias. Para cada una de las tablas se tomó en cuenta los datos de los proveedores que tuvieron más participación en el abastecimiento del negocio durante el año 2020 y sus productos, de estos datos se simulo una serie ventas durante el mes de julio. A continuación, se mostrará los resultados obtenidos para dos productos. Los parámetros para la simulación se muestran en la siguiente figura.

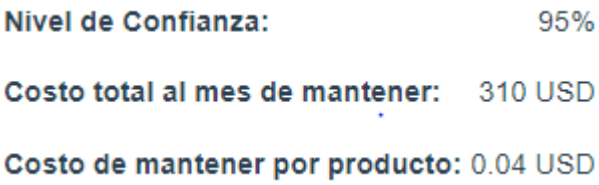


Figura 34. Parámetros para la simulación

A continuación, se construye intervalos de confianza para la media al 95% y 99% para identificar el que genera mejor la exactitud

```
#Intervalos de conafianza
#Significancia del 5%
n<-36
xmedia<-2.6
alfa<-0.05
sigma<-0.3

z<-round(abs(qnorm(alfa/2)),2)

Inferior<-xmedia-z*(sigma/sqrt(n)); Inferior
Superior<-xmedia+z*(sigma/sqrt(n)); Superior
```

```
> Inferior<-xmedia-z*(sigma/sqrt(n)); Inferior
[1] 2.502
> Superior<-xmedia+z*(sigma/sqrt(n)); Superior
[1] 2.698
```

La longitud del intervalo de confianza al 95% de confianza 0.196

Proveedor	Articulo	Cantidad en Stock	Demanda Diaria Esperada	Demanda Esperada	Stock de seguridad	Politica optima de inventario	Cantidad Optima de Pedido según D diaria	Cantidad Total Optima de Pedido	Costo del Modelo
DISTRIBUIDORA GENERAL DISTRIGEN	SEDAL SH CASPA CONTROL 12X340ML	2	3	54	7	61	17	59	$ 3.39
PROVEEDORA PRODISPRO	NUTRIBELA TRAT.BIOKERAT. 300ml.*12	1	2	22	5	27	23	26	$ 6

Figura 35. Tabla de resultados del modelo diseñado con el 95% de nivel de confianza

Proveedor	Articulo	Cantidad en Stock	Demanda Diaria Esperada	Demanda Esperada	Stock de seguridad	Politica optima de inventario	Cantidad Optima de Pedido según D diaria	Cantidad Total Optima de Pedido	Costo del Modelo
DISTRIBUIDORA GENERAL DISTRIGEN	SEDAL SH CASPA CONTROL 12X340ML	2	3	54	10	64	17	62	$ 3.57
PROVEEDORA PRODISPRO	NUTRIBELA TRAT.BIOKERAT. 300ml.*12	1	2	22	8	30	23	29	$ 6.16

Figura 36. Tabla de resultados del modelo diseñado con el 99% de nivel de confianza

Para el producto del proveedor Distribuidora General DISTRIGEN se pude observar que la cantidad en stock o de existencias es de 2 unidades; El pronóstico de demanda como se mencionó anteriormente es diario por lo que se puede observar que es de 3 unidades promedio, resultado que se obtuvo con la siguiente ecuación del Modelo de Regresión Lineal Simple que muestra en la Figura 9.

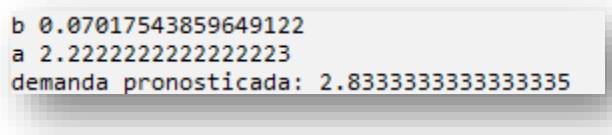


Figura 37.Resultados para el Modelo de Regresión Lineal Simple

$$2.22 + 0.070(18 + 1) = 2.83 = 3\ unidades$$

El siguiente dato que se muestra en la Figura 23 es la demanda esperada para obtenerlo se aplicó la siguiente formula del Modelo de Inventarios de la Política de Revisión Continua que se muestra en la Figura 12.

$$\mu l = 3 * 18 = 54\ unidades$$

De forma paralela se calcula el stock de seguridad al implementar la siguiente formula que se evidencia en la Figura 13, considerando el parámetro de nivel de confianza.

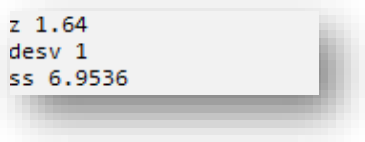


Figura 38. Resultados para calcular el stock de seguridad

$$ss = 1.64 * \sqrt{1^2 \text{x } 18} = 6.95 = 7\ unidades$$

La política óptima de inventario es el resultado de implementar la fórmula del Modelo de Inventarios de la Política de Revisión Continua que se indica en la Figura 14. Este valor es óptimo para el tiempo que tarde en responder el proveedor.

$$Politica\ inventario\ optimo = 54 + 5 = 59\ unidades$$

La cantidad óptima de pedido x demanda diaria es el resultado de implementar la fórmula del Modelo EOQ Cantidad Óptima de Pedido

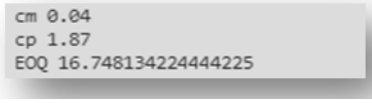


Figura 39.Resultados para calcular el EOQ

$$Q^* = \sqrt{\frac{2 \text{ x } 3 \text{ x } 1.87}{0.04}} = 16.748 = 17\ unidades$$

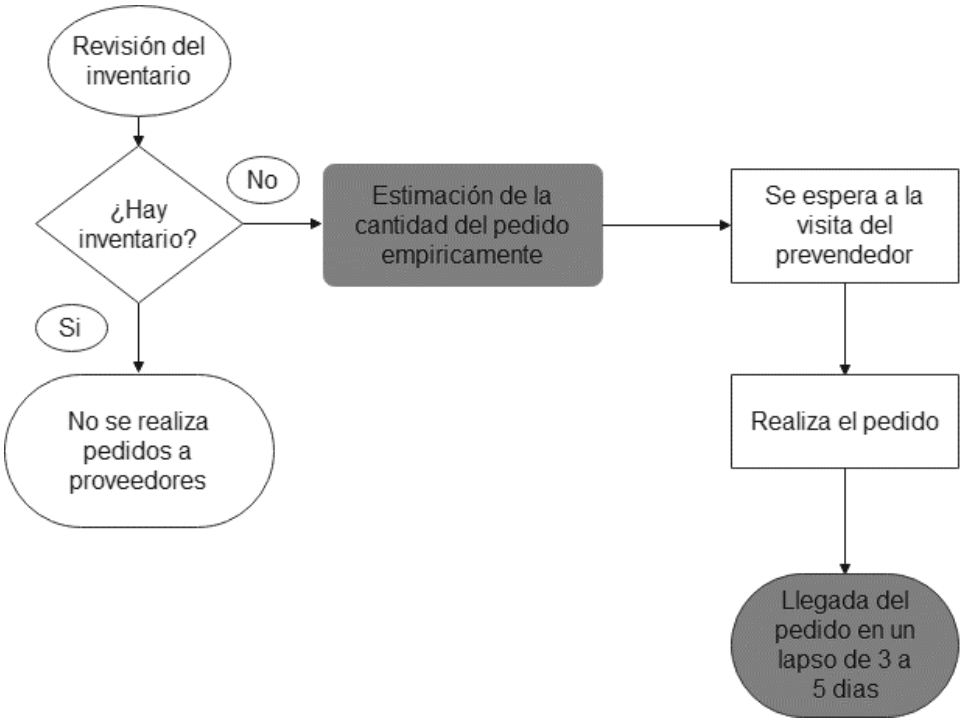
La cantidad total de pedido es el resultado de resolver la siguiente formula.

$$CTOP = \frac{(59 - 2)}{17} \text{ x } 17 = 57\ unidades$$

Por último, se muestra el costo total del modelo que se obtiene al emplear la fórmula de la Figura 17

$$C = \frac{54}{2} 0.04 + \frac{59}{54} 1.87 + 0.04 \text{ x } 5 = 3.22\ dolares$$

Análisis de los procesos identificados dentro del diagnóstico vs lo datos obtenidos con el software.



Para realizar el análisis es conveniente partir desde los parámetros que se encuentran en la gestión de pedios a proveedores. Como se observa en la anterior figura se puede observar los siguientes procesos.

Revisión del inventario. - como se mencionó en el diagnostico el negocio tiene un sistema tecnológico, que le permite registrar la cantidad de productos que ingresan al almacén, y la actualización del número de existencias cada vez que son vendidos, por lo que este proceso es eficiente. Pero en el caso de que se implemente en negocios que no cuenten con ese tipo de tecnología, el modelo diseñado, contiene un módulo para llevar un control del stock, optimizando la gestión.

Estimación de la cantidad de pedido. - El negocio realiza esta actividad de manera empírica, mientras que el modelo diseñado lo realiza de forma técnica al implementar el Modelo Regresión Lineal Simple para pronosticar la demanda diaria, el del Modelo de Inventarios de la Política de Revisión Continua para calcular la demanda esperada hasta recibir un nuevo pedido, el stock de seguridad basado en la desviación estándar y el nivel de confianza, la política óptima de inventarios dependiendo de la demanda esperada durante el tiempo de respuesta del proveedor y el Modelo EOQ Cantidad Óptima de Pedido fundamentado en la demanda pronosticada y por último la cantidad total de pedido basado en la política óptima de inventarios.

Visita del pre vendedor y Llegada del pedido. - Debido a que estos tiempos son fijados por los proveedores, no se pueden modificar, por esta razón se estableció en el modelo diseñado como tiempo de respuesta a la suma de las dos duraciones y se logró optimizarlos por medio de la cantidad óptima de pedido, ya que, el negocio puede hacer frente a la demanda hasta reabastecerse. Con esto se reduce el riesgo de una rotura de stock o generar costos innecesarios por exceso de inventarios, también permite una mayor rotación de las existencias. Además, el modelo permite conocer el costo logístico.

4.2. DISCUSIÓN

Esta investigación tuvo el propósito, generar una herramienta tecnológica que permita optimizar los procesos de gestión de pedidos a proveedores. El primer objetivo fue conocer los métodos actuales del negocio con lo que se evidencio a través una entrevista, que existen actividades manuales tales como, el cálculo de la cantidad óptima de pedido que perjudica tanto en tiempo y en costo, por lo que se deduce, que no se lleva un buen control de los inventarios. Siendo este un problema similar al trabajo de Gómez y Guzmán (2016) mencionada en los antecedentes y la solución de las dos investigaciones se basa en tecnología.

De los procesos actuales de pedidos a proveedores se identificó como parámetros críticos la estimación de la cantidad de pedido y el tiempo que tarda en reabastecerse el negocio, porque estos puntos son clave para cumplir con la demanda. Con el estudio se determinó que el tiempo de respuesta del proveedor es la suma de la visita del pre vendedor más la llegada del pedido; entonces, para que esta duración sea óptima, depende del monto de pedido, en otras palabras, la cantidad de pedido debe tener el tamaño adecuado para disponer de existencias, hasta el próximo reabastecimiento.

Para crear el software se utilizó leguajes de programación, para la base de datos SQL debido a que no demandaba un modelo multidimensional; el servidor fue creado con Python con su framework Flask por el fácil entendimiento de su sintaxis para programar y la alta capacidad al momento de trabajar con datos; en el caso den la interface su utilizo JavaScript con su framework VUE JS y con los lenguajes HTML y CSS, este conjunto de herramientas se utilizaron, puesto que, es un lenguaje es amigable al momento de desarrollar la parte visual.

Entre los modelos matemáticos que intervienen en el diseño, se encuentra, el Modelo de Regresión Lineal Simple con el que se puede pronosticar la demanda, de acuerdo, a datos diarios reales de periodos pasados, también se implementó el Modelo de Inventarios de la Política de Revisión Continua permitiendo conocer el stock de seguridad y la política óptima de inventario. Como se tiene conocimiento, la política óptima de inventarios, establece un límite para re-ordenar, esto quiere decir que cuando los inventarios lleguen a esa cantidad mínima, el negocio debe realizar un nuevo pedido al proveedor para reabastecerse, tal y como lo estableció Juca, Narváez, Erazo y Luna (2019) en su investigación, pero para el modelo diseñado en este trabajo, no se lo considera así, sino que, calcula la cantidad óptima de existencias para hacer frente a la demanda durante el tiempo de respuesta de proveedor y la visita de un pre vendedor es considerado un punto de re-orden.

De la misma forma que las investigaciones de los antecedentes se implementó el modelo EOQ cantidad óptima de pedido, que calcula en función a la demanda pronosticada como se mencionó en los resultados. Pero la cantidad total de pedido se estableció con la fórmula que se muestra en la Figura 16, en la cual se muestra dos partes, una que hace referencia al número de órdenes a pedir y la otra es el eoq calculado, dando como resultado el monto optimo que permita hacer frente a la demanda esperada y concuerde con la política óptima de inventario.

Con la simulación de los procesos de gestión de pedidos a proveedores al implementar el modelo diseñado, se evidencio, que al conocer las existencias y las ventas de cada producto se puede pronosticar la demanda y calcular el monto óptimo a pedir, por ende, el tiempo de respuesta de proveedor es óptimo, porque la cantidad total a pedir hace frente a la demanda

esperada disminuyendo el riesgo de una rotura de stock al establecer un inventario de seguridad. Al igual que la investigación de Gómez y Guzmán (2016) tener un sistema que permita llevar un control de inventarios reduce las fallas y otro beneficio que se confirmó es la optimización de los costos, ya que la frecuencia de ordenes se equilibra de acuerdo a la cantidad de pedida, de la misma manera que lo planteo Aragón (2017) en su trabajo.

Entre las herramientas que puede utilizar un negocio comercial se encuentra las hojas de cálculo (Excel) ya que es un software común en las computadoras y también es capaz de realizar cálculos de los modelos matemáticos utilizados en este trabajo con platillas que se pueden obtener de la internet. La desventaja es que el negocio tendría que insertar los datos en cada una de las plantillas. A continuación, se muestran tablas donde se observa los resultados del software diseñado versus una plantilla de Excel.

Tabla 10. Pronósticos de la plantilla versus el software diseñado

Pronósticos	
Plantilla	**Software**
2,292397660818720	2,29239766081871
2,362573099415300	2,36257309941520
2,432748538011800	2,43274853801169
2,502923976608200	2,50292397660818
2,573099415204690	2,57309941520467
2,643274853801180	2,64327485380116
2,713450292397670	2,71345029239766
2,783625730994160	2,78362573099415
2,853801169590650	2,85380116959064
2,923976608187140	2,92397660818713
2,994152046783640	2,99415204678362
3,064327485380130	3,06432748538011
3,134502923976620	3,13450292397660
3,204678362573200	3,20467836257309
3,274853801169600	3,27485380116959
3,345029239766090	3,34502923976608
3,415204678362580	3,41520467836257
3,485380116959070	3,48538011695906

Fuente: Ingenio Empresa – Software diseñado

Tabla 11. Media Aritmética de la plantilla versus el software diseñado

Media Aritmética	
Plantilla	**Software**
2,88888888888891	2,88888888888888

Fuente: Ingenio Empresa – Software diseñado

Tabla 12. . Varianza de la plantilla versus el software diseñado

Varianza	
Plantilla	**Software**
0,140350877192973	0,140350877192988

Fuente: Ingenio Empresa – Software diseñado

Tabla 13. Modelo del inventario de revisión continua de la plantilla versus el software diseñado

Modelo inventario de revisión continua		
	Plantilla	**Software**
Política óptima de inventario	61	61
Stock de seguridad	7	7

Fuente: Ingenio Empresa – Software diseñado

Como se observa en las tablas anteriores, la variabilidad de los datos es mínima y se encuentra a partir de la cifra decimal número catorce y en los dos casos tienden a uno es decir si se redondea a una cifra, prácticamente no habría varianza. deduciendo que el software es una herramienta funcional y efectiva al momento de gestionar los procesos de pedidos a proveedores.

5. CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

- La situación actual de los procesos actuales de pedidos a proveedores consta de los procesos de revisión de inventarios, Estimación de la cantidad de pedido, visita del pre vendedor, realizar el pedido y llegada del pedido.
- Los procesos de pedidos a proveedores deben de manejarse de acuerdo a los tiempos planificados por cada empresa proveedora, por lo que la revisión de inventarios debe ser eficiente con la ayuda de un software.
- Se determinó que la estimación de la cantidad de pedido y los tiempos son parámetros críticos de los modelos de pedidos a proveedores.
- El software se programó utilizando los siguientes leguajes de programación.:

Tabla 14. Leguajes de programación utilizados

Base de datos	**SQL**
Servidor	Python
Interface	JavaScript-VUE JS
	HTML
	CSS

- Los modelos utilizados en la programación fueron el Modelo de Regresión Lineal Simple, Modelo de Inventarios de la Política de Revisión Continua y el Modelo EOQ Cantidad Óptima de Pedido.
- La simulación comprobó que el modelo diseñado optimiza los procesos porque el negocio puede conocer la cantidad de pedido, la posible demanda de los productos procurando disminuir los riegos de no tener inventario o de tener exceso de inventario.

5.2. RECOMENDACIONES

- Procurar que los procesos de pedidos a proveedores se realicen de manera técnica, ya que, de forma empírica puede llegar a generar errores como costos innecesarios por el exceso de inventarios o que el negocio sea impotente frente a la demanda.
- Utilizar un software que le permita conocer las ventas por producto, puesto que, así se podrá establecer una posible demanda para los próximos periodos. Si es posible emplear un software que se encargue de analizar las ventas y calcular la cantidad óptima de pedido.
- Es recomendable usar un nivel de confianza del 95%, debido a que genera resultados más acordes a la realidad de la demanda del negociocomercial.

- Permitir que la tecnología se encargue de gestionar procesos que se pueden automatizar, con esto el encargado del negocio tiene tiempo de buscar estrategias de crecimiento.

IV. REFERENCIAS BIBLIOGRÁFICAS

Aragón, J. (2017). *Diseño de un Modelo de Gestión de Inventarios para una Empresa Comercializadora. Monterrey, Nuevo León:* (Tesis de pregrado) Instituto Tecnológico y de Estudios Superiores de Monterrey, Escuela de Ingeniería y Ciencias. Recuperado el 17 de junio de 2021, de https://repositorio.tec.mx/bitstream/handle/11285/630016/TESIS%20Jesus%20Aragon.pdf?sequence=1&isAllowed=y

Arias, F. (2012). *El Proyecto de investigación Introducción a la investigación científica*. Caracas: Editorial Episteme.

Betancourt, D. (2017). *Ingenio Empresa*. Recuperado el 15 de noviembre de 2020, de www.ingenioempresa.com/modelos-deterministicos-de-inventario.

Betancourt, D. (2018). *Ingenio Empresa*. Recuperado el 15 de noviembre de 2020, de www.ingenioempresa.com/modelos-probabilisticos-inventario.

Carro, R., & Gonzáles, D. (2013). *Gestión de stocks. Mar de la Plata*: Universidad Nacional de Mar del Plata. Recuperado el 17 de noviembre de 2021, de http://nulan.mdp.edu.ar/1830/1/gestion_stock.pdf

Castillo, K. (2005). *Propuesta de política de inventarios para productos "A" de la empresa REFA Mexicana S.A. de C.V. Puebla*: Universidad de las Américas Puebla. Recuperado el 24 de noviembre de 2020, de http://catarina.udlap.mx/u_dl_a/tales/documentos/lmnf/castillo_g_ka/capitulo1.pdf

Coding Potions. (2021). *Coding Potions*. Recuperado el 13 de julio de 2021, de https://codingpotions.com/que-es-vue

Epitech. (2021). *Epitech*. Recuperado el 13 de julio de 2021, de https://www.epitech-it.es/flask-python/

Gómez, R., & Guzmán, O. (2016). *Desarrollo de un sistema de inventarios para el control de materiales, equipos y herramientas dentro de la empresa de construcción Ingeniería Sólida Ltda.* (Tesis de pregrado) Universidad Libre Facultad De Ingeniería Programa de Ingeniería Industrial, Bogotá, D.C. Recuperado el 17 de julio de 2021, de https://repository.unilibre.edu.co/bitstream/handle/10901/9170/proyecto.pdf

Gonzales, P. (2019). *¿Qué es Optimización de Recursos?* Recuperado el 15 de febrero de 2020, de https://www.billin.net/glosario/definicion-optimizacion-de-recursos/

Guiza, F. (2013). *slideshare*. Recuperado el 11 de agosto de 2020, de https://es.slideshare.net/FerGiza/investigacin-documental-16405056

Heizer, J., & Render, B. (2009). *Principios de Administración de Operaciones. Mexico: Pearson Educación de México, S.A. de C.V.* Recuperado el 15 de noviembre de 2020, de https://clea.edu.mx/biblioteca/files/original/47cb70cab6ec78aa65b34e6c70ce8822.pdf

Hernández, R., Fernández, C., & Baptista, P. (2014). *Metodología de la investigación. En Metodología de la investigación. Ciudad de Mexico: Interamericana Editores, S.A. de C.V.* Recuperado el 15 de noviembre de 2020, de https://drive.google.com/file/d/0B7fKI4RAT39QeHNzTGh0N19SME0/view?resourcekey=0-Tg3V3qROROH0Aw4maw5dDQ

Juca, C., Narváez, C., Erazo, J., & Luna, K. (2019). *Modelo de gestión y control de inventarios para la determinación de los niveles óptimos en la cadena de suministros de la Empresa Modesto Casajoana Cía. Ltda.* Dialnet. Recuperado el 17 de junio de 2021, de https://dialnet.unirioja.es/servlet/articulo?codigo=7144054

Macia, L. (2014). *Gestión clínica*. Mexico: Elesevier. Recuperado el 17 de febrero de 2021, de https://rua.ua.es/dspace/bitstream/10045/36416/1/Tema_5Gestion_por_procesos.pdf

Mc, F. (1996). *Psicología Experimental. Métodos de Investigación. Sexta Edición.* Mexico: Prentice Hall Hispanoamericana

.

MDN contributors. (2021). *MDN Web Docs*. Recuperado el 11 de febrero de 2021, de https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript

Montoya, A. (2002). *Administración de compras. Mexico: Ecoe Ediciones*. Recuperado el 12 de agosto de 2020, de https://es.scribd.com/document/380829071/Cap-3-Los-Proveedores-

Administración-de-Compras-Alberto-Montoya-Palacio

Múzquiz, D. (2013). *Administración de inventarios y almacenes.* Mazatlán: Centro de Educación Continua Unidad Mazatlán. Recuperado el 2021 de noviembre de 17, de

https://repositoriodigital.ipn.mx/bitstream/123456789/17612/1/manual%20admon%20de%20inventarios%20y%20almacenes%202013.pdf

Olarte, G. (2018). *Conogasi*. Recuperado el 13 de julio de 2020, de http://conogasi.org/articulos/lenguaje-de-programacion/

Porto, J. (2008). *Definicion De*. Recuperado el 17 de febrero de 2021, de https://definicion.de/modelo-de-gestion/

Raffino, M. E. (2020). *Concepto.de*. Recuperado el 11 de febrero de 2021, de Concepto.de: https://concepto.de/estadistica-inferencial/

Robledano, A. (2019). *Open Webinars*. Recuperado el 11 de febrero de 2021, de https://openwebinars.net/blog/que-es-python/

Sirkin, J. (2021). *TechTarget*. Recuperado el 13 de julio de 2021, de https://www.computerweekly.com/es/definicion/SQL-Structured-Query-Language-o-Lenguaje-de-consultas-estructuradas

Vermorel, J. (2012). *Lokad Quantitative Supply Chain*. Recuperado el 12 de agosto de 2020, de https://www.lokad.com/es/definicion-punto-de-reorden

Vidal, M. (2019). *IEBS*. Recuperado el 13 de julio de 2021, de https://www.iebschool.com/blog/progressive-web-apps-analitica-usabilidad/

Westreicher, G. (s.f.). *Economipedia.* Recuperado el 15 de noviembre de 2019, de https://economipedia.com/definiciones/optimizacion.html

Westreicher, G. (s.f.). *Economipedia.* Recuperado el 17 de noviembre de 2021, de https://economipedia.com/definiciones/almacenamiento.html

Westreicher, G. (s.f.). *Economipedia*. Recuperado el 17 de noviembre de 2021, de https://economipedia.com/definiciones/control-de-inventario.html

V. ANEXOS

Anexo 1: Certificado o Acta del Perfil de Investigación

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI
FACULTAD DE COMERCIO INTERNACIONAL, INTEGRACION, ADMINISTRACION Y ECONOMIA EMP
CARRERA DE LOGISTICA Y TRANSPORTE

Educamos para transformar el mundo

ACTA

DE LA SUSTENTACIÓN DE PREDEFENSA DEL DEL TRABAJO DE INTEGRACIÓN CURRICULAR:

NOMBRE: AGUILAR BELTRÁN JONNATHAN XAVIER **CÉDULA DE IDENTIDAD:** 0401622774
NIVEL/PARALELO: EGRESADO **PERIODO ACADÉMICO:** 2021 A

TEMA DEL TIC: Diseño de un modelo de gestión en base tecnológica para la optimización de procesos de pedidos a proveedores de los negocios comerciales en la ciudad de Tulcán

Tribunal designado por la dirección de esta Carrera, conformado por:
PRESIDENTE: MSC. LÓPEZ RUANO JUAN CARLOS
DOCENTE TUTOR: MSC. HEREDIA CAMPAÑA ARGENIS LISSANDER
DOCENTE: MSC. MAFLA BOLAÑOS IVÁN GABRIEL

De acuerdo al artículo 32: Una vez entregados los documentos; y, cumplidos los requisitos para la realización de la pre-defensa el Director/a de Carrera designará el Tribunal, fijando lugar, fecha y hora para la realización de este acto:

EDIFICIO DE AULAS VIRTUAL **AULA:** 0
FECHA: jueves, 18 de noviembre de 2021
HORA: 10H00
Obteniendo las siguientes notas:
1) Sustentación de la predefensa: 6.67
2) Trabajo escrito 2.90
Nota final de PRE DEFENSA **9.57**
Por lo tanto: **APRUEBA CON OBSERVACIONES** ; debiendo acatar el siguiente artículo:

Art. 36.- De los estudiantes que aprueban el informe final del TIC con observaciones.- Los estudiantes tendrán el plazo de 10 días para proceder a corregir su informe final del TIC de conformidad a las observaciones y recomendaciones realizadas por los miembros del Tribunal de sustentación de la pre-defensa.

Para constancia del presente, firman en la ciudad de Tulcán el jueves, 18 de noviembre de 2021

MSC. LÓPEZ RUANO JUAN CARLOS
PRESIDENTE

MSC. HEREDIA CAMPAÑA ARGENIS LISSANDER
DOCENTE TUTOR

MSC. MAFLA BOLAÑOS IVÁN GABRIEL
DOCENTE

Adj.: Observaciones y recomendaciones

Anexo 2:Certificado del abstract por parte de idiomas

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI
FOREIGN AND NATIVE LANGUAGE CENTER

ABSTRACT- EVALUATION SHEET

NAME: Jonnathan Xavier Aguilar Beltrán
DATE: 7 de diciembre de 2021

TOPIC: "Diseño de un modelo de gestión en base tecnológica para la optimización de procesos de pedidos a proveedores de los negocios comerciales en la ciudad de Tulcán."

MARKS AWARDED **Q U A N T I T A T I V E AND Q U A L I T A T I V E**

VOCABULARY AND WORD USE	Use new learnt vocabulary and precise words related to the topic	Use a little new vocabulary and some appropriate words related to the topic	Use basic vocabulary and simplistic words related to the topic	Limited vocabulary and inadequate words related to the topic
	EXCELLENT: 2	GOOD: 1,5	AVERAGE: 1	LIMITED: 0,5
WRITING COHESION	Clear and logical progression of ideas and supporting paragraphs.	Adequate progression of ideas and supporting paragraphs.	Some progression of ideas and supporting paragraphs.	Inadequate ideas and supporting paragraphs.
	EXCELLENT: 2	GOOD: 1,5	AVERAGE: 1	LIMITED: 0,5
ARGUMENT	The message has been communicated very well and identify the type of text	The message has been communicated appropriately and identify the type of text	Some of the message has been communicated and the type of text is little confusing	The message hasn't been communicated and the type of text is inadequate
	EXCELLENT: 2	GOOD: 1,5	AVERAGE: 1	LIMITED: 0,5
CREATIVITY	Outstanding flow of ideas and events	Good flow of ideas and events	Average flow of ideas and events	Poor flow of ideas and events
	EXCELLENT: 2	GOOD: 1,5	AVERAGE: 1	LIMITED: 0,5
SCIENTIFIC SUSTAINABILITY	Reasonable, specific and supportable opinion or thesis statement	Minor errors when supporting the thesis statement	Some errors when supporting the thesis statement	Lots of errors when supporting the thesis statement
	EXCELLENT: 2	GOOD: 1,5	AVERAGE: 1	LIMITED: 0,5
TOTAL/AVERAGE	9 - 10: EXCELLENT 7 - 8,9: GOOD 5 - 6,9: AVERAGE 0 - 4,9: LIMITED	**TOTAL 9**		

1

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI FOREIGN AND NATIVE LANGUAGE CENTER

Informe sobre el Abstract de Artículo Científico o Investigación.

Autor: Jonnathan Xavier Aguilar Beltrán
Fecha de recepción del abstract: 7 de diciembre de 2021
Fecha de entrega del informe: 7 de diciembre de 2021

El presente informe validará la traducción del idioma español al inglés si alcanza un porcentaje de: 9 – 10 Excelente.

Si la traducción no está dentro de los parámetros de 9 – 10, el autor deberá realizar las observaciones presentadas en el ABSTRACT, para su posterior presentación y aprobación.

Observaciones:
Después de realizar la revisión del presente abstract, éste presenta una apropiada traducción sobre el tema planteado en el idioma Inglés. Según los rubrics de evaluación de la traducción en Inglés, ésta alcanza un valor de 9, por lo cual se validad dicho trabajo.

Atentamente

Firmado electrónicamente por:
EDISON BOANERGES PENAFIEL ARCOS

Ing. Edison Peñafiel Arcos MSc
Coordinador del CIDEN

2

Anexo 3: Entrevista al propietario del negocio Tus Enkntos

Entrevista Semiestructurada

OBJETIVO**:** Describir la situación actual de los procesos generales de la gestión de pedidos a proveedores del negocio comercial TUS ENKNTOS que se dedica a la venta al por mayor de productos de perfumería, cosméticos, artículos de aseo personal.

1. **¿Cómo es la gestión de pedidos a proveedores en el negocio?**
   Esta gestión se basa en realizar una revisión de los inventarios mediante el Software SISMANOS, si hay inventario no se realiza un pedido, pero si no hay o se cree que falta inventario se establece la cantidad de pedido basada en las creencias de la cantidad que se puede vender, posteriormente se realiza el pedido con el pre vendedor y el último punto es la llegada del pedido.

2. **¿Cuáles son los costos que conlleva de la gestión de pedidos a proveedores?**
   Los costos que conlleva la gestión de pedios al mes se basa en el mantenimiento del almacén que son alrededor de $ 70, arriendo del almacén $ 100, servicios básicos $ 100, y seguridad $ 40. Otro costo depende la cantidad por producto y por productos. Pero en si no se conoce el costo total de la gestión.

3. **¿Qué tiempo se tarda en realizar la gestión de pedidos a proveedores?**

   El tiempo de la gestión depende de la planificación de los proveedores que se está estructurado por la visita de los pre vendedores que puede ser una, dos o cuatro veces al mes y el otro tiempo es la llega del pedido que puede tardar entre 3 a 5 días.

4. **¿Cómo le ayuda el software SISMANOS a su negocio?**

   El software solo lo utiliza para la facturación y registrar los inventarios, pero también genera reportes de las existencias, y de las compras realizadas a los proveedores, pero solo en dinero, mas no en cantidad de producto comprado. Las ventas las registra por factura.

Anexo 4: Programación del software

```
CREATE DATABASE DB_INVENTARIOS;
USE DB_INVENTARIOS;
CREATE TABLE PROVEEDORES (
CODIGO INT(11) NOT NULL PRIMARY KEY,
PROVEEDOR VARCHAR(60) NOT NULL,
DIRECCION VARCHAR(80) NOT NULL,
TIEMPO  INT(11) NOT NULL
) ;
CREATE TABLE EXISTENCIAS (
CODIGO VARCHAR(20) NOT NULL PRIMARY KEY,
ARTICULO VARCHAR(60) NOT NULL,
EXISTENCIAS INT(11) NOT NULL,
COSTO DECIMAL(10,2) NOT NULL,
PRECIO DECIMAL(10,2) NOT NULL,
PROVEEDOR INT(11) NOT NULL,
FOREIGN KEY (PROVEEDOR) REFERENCES PROVEEDORES(CODIGO) ON DELETE CASCADE
) ;
CREATE TABLE VENTAS (
NUM INT(11) NOT NULL,
FECHA DATE NOT NULL,
CODIGO VARCHAR(15) NOT NULL,
ARTICULO VARCHAR(60) NOT NULL,
CANTIDADVENTAS INT(11) NOT NULL,
FOREIGN KEY (CODIGO) REFERENCES EXISTENCIAS(CODIGO) ON DELETE CASCADE
);
CREATE TABLE CONFIGURACIONES (
ID INT(1) NOT NULL,
NIVEL INT(2) NOT NULL,
COSTO_MANTENER DECIMAL(10,2) NOT NULL
 );
```

Figura 40. Código para la creación de la base de datos

```
from flask import Flask, jsonify, request, redirect
from flask_cors import CORS
from flask_mysqldb import MySQL
from nivelDeconfianza import nivelC
import math
import statistics
```

Figura 41. Código para importar los módulos necesarios para el servidor.

```
app = Flask(__name__)
if __name__ == '__main__':
    app.run(port=3398)
```

Figura 42. Código para la creación e inicialización del servidor

```
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_PORT'] = 3307
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = ''
app.config['MYSQL_DB'] = 'db_inventarios'
mysql = MySQL(app)
```

Figura 43. Código para la conexión a la base de datos

```
@app.route('/insert-proveedor', methods=['POST'])
def insetarProveedor():
    if request.method == 'POST':
        _codigo = request.form['codigo']
        _proveedor = request.form['proveedor']
        _direccion = request.form['direccion']
        _tiempo = request.form['tiempo']
        cur = mysql.connection.cursor()
        cur.execute(
            'INSERT INTO proveedores (CODIGO, PROVEEDOR, DIRECCION, TIEMPO) VALUES (%s, %s, %s, %s)',
            (_codigo, _proveedor, _direccion, _tiempo))
        mysql.connection.commit()
        return redirect(rutaInterfase + 'registro')
```

Figura 44. Código para insertar proveedores en la base de datos

```python
@app.route('/registro-inventario',  methods=['POST', 'PUT'])
def add_product():
    if request.method == 'POST':
        _codigo = request.form['codigo']
        _articulo = request.form['articulo']
        _existencias = request.form['cantidad']
        _costo_unit = request.form['costo']
        _proveedor = request.form['proveedor']
        cur = mysql.connection.cursor()
        verificar = cur.execute(
            'SELECT * FROM EXISTENCIAS WHERE CODIGO = %s',[str(_codigo)]
        )
        if verificar is 0:
            cur.execute(
                'INSERT INTO existencias (CODIGO, ARTICULO, EXISTENCIAS, COSTO, PROVEEDOR) VALUES (%s, %s, %s, %s, %s)',
                (_codigo, _articulo, _existencias, _costo_unit, _proveedor))
            mysql.connection.commit()
        elif verificar is 1:
            cur.execute(
                """UPDATE existencias
                    SET EXISTENCIAS = %s
                    WHERE CODIGO = %s""", (_existencias, _codigo)
                )
            mysql.connection.commit()
        return redirect(rutaInterfase +'registro')
```

Figura 45. Código para insertar existencias o inventarios en la base de datos

```python
@app.route('/insert-ventas', methods=['POST', 'PUT'])
def insertarVentas():
    if request.method == 'POST':
        _num = request.form['codigo']
        _fecha = request.form["fecha"]
        _codigo = request.form['codigo']
        _articulo = request.form['articulo']
        _cantidad = request.form['cantidad']
        cur = mysql.connection.cursor()
        verificar = cur.execute(
            'SELECT FECHA, CODIGO FROM VENTAS WHERE FECHA = %s AND  CODIGO = %s', (_fecha, _codigo)
        )
        if verificar is 0:
            cur.execute(
                'INSERT INTO ventas (NUM, FECHA, CODIGO, ARTICULO, CANTIDADVENTAS) VALUES (%s, %s, %s, %s, %s)',
                ('', _fecha, _codigo, _articulo, _cantidad))

            mysql.connection.commit()
            ## Llamado de la funcion
            updateInventarios(_codigo, _cantidad)

            return redirect(rutaInterfase)
        elif verificar is 1:

            cur.execute(
            'SELECT CANTIDADVENTAS FROM VENTAS WHERE FECHA = %s AND  CODIGO = %s LIMIT 1', (_fecha, _codigo)
            )
            dato = cur.fetchall()
            ## OBTENER CANTIDAD ACTUAL

            _cantidadActual = 0
            for e in dato:
                _cantidadActual = e[0]

            ## ACTUALIZACION DE LA VENTA
            _newDato = _cantidadActual + int(_cantidad)                                                    ##

            cur.execute(
                """
                UPDATE ventas
                SET CANTIDADVENTAS = %s
                WHERE FECHA = %s AND CODIGO = %s
                """, (_newDato, _fecha, _codigo)
                )
            mysql.connection.commit()
              ##Llamado de la funcion
              updateInventarios(_codigo, _cantidad)
              return redirect(rutaInterfase)

## Funcion para la actualizacion de la Tabla de existencias
def updateInventarios(e, cantidad):

        cur= mysql.connection.cursor()
        cur.execute(
            'SELECT EXISTENCIAS FROM EXISTENCIAS WHERE CODIGO = %s LIMIT 1', [e]
        )
        datoE = cur.fetchall()
        _cantidadActualE = 0
        for i in datoE:
            _cantidadActualE = i[0]

        _newDatoE = _cantidadActualE - int(cantidad)
        cur.execute(
                    """
                        UPDATE existencias
                        SET EXISTENCIAS = %s
                        WHERE CODIGO = %s
                    """, ((_newDatoE), e))

        mysql.connection.commit()
```

Figura 46. Código para insertar ventas en la base de datos

```python
@app.route('/datos-config-update', methods=['POST'])
def datosConfigUpadate():
    if request.method == 'POST':
        nivel = request.form['nivel']
        costoM = request.form['costoM']
        idConf = 1
        cur = mysql.connection.cursor()
        verificar = cur.execute(
            'SELECT * FROM CONFIGURACIONES WHERE ID = %s', [idConf]
        )
        if verificar is 0:
            cur.execute(
                'INSERT INTO CONFIGURACIONES (ID, NIVEL, COSTO_MANTENER) VALUES (%s, %s, %s)', (id, nivel, costoM)
            )
            mysql.connection.commit()
            return redirect(rutaInterfase, 'Config')
        elif verificar is 1:
            cur.execute(
            """UPDATE CONFIGURACIONES
            SET NIVEL = %s,  COSTO_MANTENER = %s WHERE ID = %s""", (id, nivel, costoM))
            mysql.connection.commit()
            return redirect(rutaInterfase, 'Config')
```

Figura 47. Código para insertar los parámetros del modelo en la base de datos

```python
@app.route('/get-proveedores')
def getProveedores():
        cur = mysql.connection.cursor()
        cur.execute(
            'SELECT * FROM PROVEEDORES'
        )
        proveedores = cur.fetchall()
        mysql.connection.commit()
        return jsonify(proveedores)
```

***Figura 48**. Código para obtener todos los datos de una tabla de la base de datos – proveedores*

```python
@app.route('/get-inventarios/')
def get_inventario():
    cur = mysql.connection.cursor()
    cur.execute('SELECT * FROM EXISTENCIAS')
    datas = cur.fetchall()
    print(datas)
    return jsonify(datas)
```

Figura 49. Código para obtener todos los datos de una tabla de la base de datos – existencias

```python
## TARER VENTAS PARA MOSTRAR EN LA INTERFASE
@app.route('/get-ventas')
def getventas():
    cur = mysql.connection.cursor()
    cur.execute(
        'SELECT FECHA, CODIGO, ARTICULO, CANTIDADVENTAS FROM VENTAS ORDER BY FECHA DESC'
    )
    ventas = cur.fetchall()
    return jsonify(ventas)
```

Figura 50. Código para obtener todos los datos de una tabla de la base de datos – ventas

```python
@app.route('/datos-config')
def datosConfig():
    cur = mysql.connection.cursor()
    cur.execute(
        'SELECT * FROM CONFIGURACIONES'
    )
    datos = cur.fetchall()
    return jsonify(datos, configuraciones_costoM())


def configuraciones_costoM():
    cur = mysql.connection.cursor()
    cur.execute('SELECT COSTO_MANTENER FROM CONFIGURACIONES')
    nivel = cur.fetchall()
    print(nivel)
    cm = 0
    for c in nivel:
        cm = round((float(c[0])/30)/float(totalExistencias()), 2)
    print(cm)
    print(totalExistencias())
    return cm

def totalExistencias():
    cur = mysql.connection.cursor()
    cur.execute(
        'SELECT SUM(EXISTENCIAS) FROM EXISTENCIAS'
    )
    getDato = cur.fetchall()
    for i in getDato:
        dato = i[0]
    print('totalE', dato)
    return(dato)
```

Figura 51. Código para obtener todos los datos de una tabla de la base de datos – configuraciones

```python
@app.route('/all-datos')
def allDatos():
    cur = mysql.connection.cursor()
    cur.execute(
        """SELECT  * FROM PROVEEDORES
        JOIN EXISTENCIAS ON PROVEEDORES.CODIGO=EXISTENCIAS.PROVEEDOR"""
    )
    datos = cur.fetchall()
    return jsonify(datos)
```

Figura 52. Código para obtener todos los datos de dos tablas de la base de datos

```python
@app.route('/get-datos/<string:id>')
def get_datos(id):
    cur = mysql.connection.cursor()
    cur.execute('SELECT * FROM EXISTENCIAS WHERE CODIGO = %s', [id])
    data = cur.fetchall()
    print(data)
    return jsonify(data)
```

Figura 53. Código para obtener un dato especifico de una tabla de la base de datos – existencias

```python
@app.route('/get-ventas/<string:articulo>')
def getventasArticulo(articulo):
    cur = mysql.connection.cursor()
    cur.execute(
        'SELECT FECHA, CODIGO, ARTICULO, CANTIDADVENTAS FROM VENTAS WHERE ARTICULO = %s', [articulo]
    )
    ventas = cur.fetchall()
    return jsonify(ventas)
```

Figura 54. Código para obtener un dato especifico de una tabla de la base de datos – ventas

```python
@app.route('/<string:_codigo>/<int:_tiempo>')
def get_Ventas_algoritmo(_codigo, _tiempo):
    cur = mysql.connection.cursor()
    cur.execute(
            """SELECT VENTAS.CODIGO, CANTIDADVENTAS, COSTO, EXISTENCIAS, VENTAS.FECHA FROM VENTAS
            JOIN EXISTENCIAS ON VENTAS.CODIGO = EXISTENCIAS.CODIGO
            WHERE VENTAS.CODIGO = %s ORDER BY FECHA ASC LIMIT %s""",
            (_codigo, _tiempo))
    cVentas = cur.fetchall()
    print(cVentas)
    ## DATOS PARA EL PRONOSTICO CON REGRESION LINEAL
    x = []
    y = []

    _e = 0
    _datos = cVentas
    print('Fecha',_datos)
    for i in _datos:
        _e += 1
        x.append(_e)
        y.append(i[1])
        cp = i[2]
        cantidadStock = i[3]

    if(len(x)>2):
        i =0
        suma_x = sum(x)
        xPromedio = suma_x/len(x)
        print(x, y)

        suma_y = sum(y)
        yPromedio = suma_y/len(y)
        print("y =", yPromedio, suma_y)


        xy = [x*y for x, y in zip(x, y)]
        suma_xy = sum(xy)
        print("xy =", xy)

        x_2 = [x*y for x, y in zip(x, x)]
        sumar_x_2 = sum(x_2)
        print("x^2 =", sumar_x_2)
```

```
    y_2 = [x*y for x, y in zip(y, y)]
    sumar_y_2 = sum(y_2)
    print("y^2 =", sumar_y_2)

    ##PRONOSTICO DE LA DEMANDA

    b = round((suma_xy-(len(x)*xPromedio * yPromedio)/len(x))/(pow(suma_x,2) - (pow(suma_x,2)/len(x))),2)
    print('b', b)

    a = yPromedio - b*xPromedio
    print('a', a)

    pronostico = (b*_tiempo) + a
    d = pronostico
    print('demanda pronosticada:', d, _tiempo)

    ## Modelo del Inventario de revision continua o politica de revision continua
    ##DATOS PARA EL MODELO DE INVENTARIO
    ## NIVEL DE CONFIANZA
    z = configuracionesNc()
    cm = configuraciones_costoM()
    print('z', z)
    desv = statistics.stdev(y)
    o = round(desv)
    lt = _tiempo
    dlt = round(math.sqrt(lt),2)
    print('dlt',dlt)
        ## inventario de seguridad
    ss = round(z*o*dlt)
    ss2 = z*o*dlt
    print("z", z)
    print("desv", o)
    print("ss", ss2)

    dEspera = round(d * lt)
    eReserva = z*o*math.sqrt(lt)
    politicaOptima = round(dEspera + eReserva)


    ## CANTDIDAD OPTIMA DE PEDIDO
    eoq = round(math.sqrt((2*round(d)*float(cp))/float(cm)))
    print('cm', cm)
    print('cp', cp)
    print('EOQ',round(eoq))
    ## CANTIDAD TOTAL DE PEDIDO
    cantidadOptima = ((politicaOptima-cantidadStock)/round(eoq)) * round(eoq)

    ## COSTO TOTAL DEL MODELO
    s = (cantidadOptima/2)*float(cm)
    h = (politicaOptima/cantidadOptima)*float(cp)
    c = ((cantidadOptima/2)*float(cm))+((politicaOptima/cantidadOptima)*float(cp))+(float(cm)*ss)
    print('CL', c, s, h)
    print('resultados',ss, eReserva, dEspera, politicaOptima, round(eoq), cantidadOptima, d)
    print(cantidadStock)

    return jsonify(ss, dEspera, politicaOptima, round(d), eoq, cantidadOptima, round(c,2), cp, cm)


elif(len(x)<= 2):
    mensaje = []

    for i in range(5):
        mensaje.append('No hay suficientes ventas')
    print(mensaje)

    ## mensaje
    return jsonify(mensaje)
```

```
## FUNCION DE NIVEL DE CONFIANZA
def configuracionesNc():
    cur = mysql.connection.cursor()
    cur.execute('SELECT NIVEL FROM CONFIGURACIONES')
    nivel = cur.fetchall()
    print(nivel)
    nv = 0
    for c in nivel:
        nv = c[0]


    zfound = [n for n in nivelC if n['nivel'] == float(nv)]

    s = json.dumps(zfound)
    _z = json.loads(s)[0]['z']
    print('nivel de confianza', _z)
    return _z
```

Figura 55. Código para la programación de los modelos para pronostico e inventarios

```vue
<template>
  <div id="nav">
    <router-link to="/">Inicio</router-link>
    <router-link to="registro">Registro</router-link>
    <router-link to="ventas">Ventas</router-link>
    <router-link to="IRC">Modelo de IRC</router-link>
    <router-link to="Estadisticas">Estadísticas</router-link>
    <router-link to="Config">Configuraciones</router-link>
  </div>
  <router-view />
</template>

<style>
#app {
  font-family: Avenir, Helvetica, Arial, sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-align: center;
  color: #2c3e50;
}

#nav {
  padding: 30px;
}

#nav a {
  font-weight: bold;
  color: #2c3e50;
  margin: 5px;
}

#nav a.router-link-exact-active {
  color: gray;
}
</style>
```

Figura 56. Código del archivo App.vue de la Interface

```vue
<template>
  <div class="home">
    <InicioC />
  </div>
</template>

<script>
// @ is an alias to /src
import InicioC from "@/components/InicioC.vue";

export default {
  name: "InicioC",
  components: {
    InicioC,
  },
};
</script>
```

Figura 57. Código de la vista inicio

```
<template>
  <div class="hello">
    <h1>Salidas o Ventas de productos</h1>
    <input
      type="number"
      name="search"
      id="search"
      placeholder=" Codigo del Articulo"
      v-model="codigo"
      @keyup.enter="getDato()"
    />
    <button @click="getDato()">Buscar</button>

    <div class="etiquetas">
      <div>FECHA</div>
      <div>CODIGO</div>
      <div>ARTICULO</div>
      <div>PRECIO</div>
      <div>CANTIDAD</div>
      <div></div>
    </div>
    <div class="productos-content">
      <form
        class="existencias"
        :action="accion"
        method="POST"
        namae="form"
        enctype="multipart/form-data"
        v-for="(e, index) in productos"
        :key="index"
      >
        <input type="text" name="fecha" id="fecha" :value="fecha" />
        <input type="text" name="codigo" :value="e[0]" />
        <input type="text" name="articulo" :value="e[1]" />
        <input type="text" name="articulo" :value="e[4]" />
        <input
          type="number"
          name="cantidad"
          placeholder="Cantidad"
          v-model="cantidad"
          @keyup.enter="submit"
        />
        <input type="submit" value="Vender" />
      </form>
    </div>
```

```
  </div>
</template>

<script>
export default {
  name: "InicioC",
  props: {
    msg: String,
  },
  data() {
    return {
      productos: null,
      accion: "http://127.0.0.1:3398/insert-ventas",
      fecha: null,
      codigo: null,
    };
  },
  mounted() {
    var getFecha = new Date();
    this.fecha =
      getFecha.getFullYear() + "-" + (getFecha.getMonth() + 1) + "-" + getFecha.getDate();

    console.log(getFecha);
  },

  methods: {
    async getDato() {
      let data = await fetch(`http://127.0.0.1:3398/get-datos/${this.codigo}`);
      this.productos = await data.json();
      console.log(data);
      return JSON.parse(this.products);
    },
  },
};
</script>

<!-- Add "scoped" attribute to limit CSS to this component only -->
<style scoped>
.etiquetas {
  display: flex;
  justify-content: space-evenly;
  background-color: gray;
  margin: 5px;
  padding: 5px;
```

```
.etiquetas div {
  height: 20px;
  width: 150px;
  color: white;
  font-weight: bold;
  margin-left: 75px;
}

.existencias {
  display: flex;
  justify-content: space-evenly;
}

.h-datos {
  border-bottom: 1px solid black;
  padding: 10px 0px 5px 35px;
  font-size: 15px;
}
</style>
```

Figura 58. Código del componente inicioc

```
<template>
  <div>
    <RegistroC />
  </div>
</template>

<script>
import RegistroC from "../components/RegistroC.vue";

export default {
  components: { RegistroC },
};
</script>
Registro
```

Figura 59. Código de la vista registro

```
<template>
  <div>
    <div class="btns-registro" v-if="estado === true">
      <button @click="viewRegistroProveedores">Proveedores</button>
      <button @click="viewRegistroProductos">Productos</button>
    </div>
    <div class="registro-proveedores" v-show="showRps === true">
      <form
        action="http://127.0.0.1:3398/insert-proveedor"
        method="POST"
        name="form"
        enctype="multipart/form-data"
      >
        <input type="text" name="codigo" placeholder="Codigo" v-model="codigoProveedor" />
        <input type="text" name="proveedor" placeholder="Proveedor" v-model="proveedor" />
        <input type="text" name="direccion" placeholder="Direccion" v-model="direccion" />
        <input
          type="number"
          name="tiempo"
          id="tiempo"
          placeholder="Tienpo de respuesta"
        />

        <input type="submit" value="Registrar" @keyup.enter="submit" />
      </form>
    </div>
    <div class="registro-productos" v-show="showRp === true">
      <form
        action="http://127.0.0.1:3398/registro-inventario"
        method="POST"
        name="form"
        enctype="multipart/form-data"
      >
        <input type="text" name="codigo" placeholder="Codigo" v-model="codigo" />
        <input type="text" name="articulo" placeholder="Articulo" v-model="articulo" />
        <input type="number" name="cantidad" placeholder="Cantidad" v-model="cantidad" />
        <input type="number" name="costo" placeholder="Costo" v-model="costo" />
        <input
          type="number"
          name="proveedor"
          id="proveedor"
          v-model="proveedor"
          placeholder=" Codigo proveedor"
        />
```

```
      <input type="submit" value="Registrar" @keyup.enter="submit" />
    </form>
  </div>
  <div class="etiquetas-productos" v-show="showRp === true">
    <div>Codigo</div>
    <div>Articulo</div>
    <div>Cantida de existencias</div>
    <div>Costo</div>
    <div>Costo total</div>
    <div>Proveedor</div>
  </div>
  <div class="productos-content" v-show="showRp === true">
    <div class="existencias-productos" v-for="(e, index) in products" :key="index">
      <div>
        {{ e[0] }}
      </div>
      <div>
        {{ e[1] }}
      </div>
      <div>
        {{ e[2] }}
      </div>
      <div>
        {{ e[3] }}
      </div>
      <div>
        {{ parseFloat(e[2] * e[3]).toFixed(2) }}
      </div>
      <div>
        {{ e[5] }}
      </div>
    </div>
  </div>
  <div class="etiquetas" v-show="showRps === true">
    <div>Codigo</div>
    <div>Proveedor</div>
    <div>Dirección</div>
    <div>Tiempo de respuesta</div>
  </div>
  <div class="proveedores-content" v-show="showRps === true">
    <div class="proveedores" v-for="(e, index) in proveedores" :key="index">
      <div>
        {{ e[0] }}
```

```
            </div>
            <div>
              {{ e[1] }}
            </div>
            <div>
              {{ e[2] }}
            </div>
            <div>
              {{ e[3] }}
            </div>
          </div>
        </div>
      </div>
    </template>

    <script>
    export default {
      name: "RegistroC",
      data() {
        return {
          estado: false,
          accionRp: "http://127.0.0.1:3398/registro-inventario",
          codigo: null,
          articulo: null,
          cantidad: null,
          costo: null,
          productos: [],
          proveedores: null,
          products: null,
          showRp: false,
          showRps: false,
        };
      },
      created() {
        this.viewRegistroProductos();
        setTimeout(() => {
          this.estado = true;
          this.showRp = true;
        }, 2000);
      },
      methods: {
        async viewRegistroProductos() {
          this.showRp = true;
          this.showRps = false;
```

```
    }, 2000);
  },
  methods: {
    async viewRegistroProductos() {
      this.showRp = true;
      this.showRps = false;
      let data = await fetch(`http://127.0.0.1:3398/get-inventarios`);
      this.products = await data.json();
      return JSON.parse(this.products);
    },
    async viewRegistroProveedores() {
      this.showRps = true;
      this.showRp = false;
      let proveedor = await fetch(`http://127.0.0.1:3398/get-proveedores`);
      this.proveedores = await proveedor.json();
      return JSON.parse(this.proveedores);
    },
  },
};
</script>

<style scoped>
.registro-content input {
  margin: 50px;
}
.btns-registro {
  background-color: gray;
  margin: 0 0 15px 0;
  padding: 5px;
  box-sizing: border-box;
}
.btns-registro button {
  margin: 0 50px 0 0;
  color: white;
  font-weight: bold;
  background: none;
  border-bottom: 2px solid white;
  border-top: none;
  border-left: none;
  border-right: none;
}
.etiquetas {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr 1fr 1fr;
  justify-content: center;
```

```
  margin: 5px 0 5px 0;
  background-color: gray;
  color: white;
}
.etiquetas div {
  border: 1px solid black;
}
.proveedores {
  display: flex;
  font-size: 12px;
  margin: 0 0 5px 0;
}
.proveedores div {
  width: 18%;
  text-align: left;
  padding-left: 25px;
  border-bottom: 1px solid black;
}

.etiquetas-productos {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr 1fr 1fr 1fr;
  justify-content: center;
  margin: 5px 0 5px 0;
  background-color: gray;
  color: white;
}
.etiquetas-productos div {
  border: 1px solid black;
}
.existencias-productos {
  display: flex;
  font-size: 12px;
  margin: 0 0 5px 0;
}
.existencias-productos div {
  width: 19.85%;
  text-align: left;
  padding-left: 25px;
  border-bottom: 1px solid black;
}
</style>
```

Figura 60. Código del componente registroc

```vue
<template>
  <div>
    <VentasC />
  </div>
</template>

<script>
import VentasC from "../components/VentasC.vue";
export default {
  components: { VentasC },
};
</script>

<style></style>
```

Figura 61. Código de la vista ventas

```vue
<template>
  <div>
    <div class="etiquetas">
      <div>Fecha</div>
      <div>Codigo</div>
      <div>Articulo</div>
      <div>Cantidad</div>
    </div>
    <div class="ventas-content" v-for="(v, index) in datosVentas" :key="index">
      <div>
        {{ v[0][5] }}{{ v[0][6] }}-{{ v[0][8] }}{{ v[0][9] }}{{ v[0][10] }}-{{ v[0][12]
        }}{{ v[0][13] }}{{ v[0][14] }}{{ v[0][15] }}
      </div>
      <div>
        {{ v[1] }}
      </div>
      <div>
        {{ v[2] }}
      </div>
      <div>
        {{ v[3] }}
      </div>
    </div>
  </div>
</template>

<script>
export default {
  name: "VentasC",
  data() {
    return {
      datosVentas: null,
    };
  },
  created() {
    setTimeout(() => {
      this.getVentas();
    }, 3000);
  },
  methods: {
    async getVentas() {
      let ventas = await fetch(`http://127.0.0.1:3398/get-ventas`);
      this.datosVentas = await ventas.json();
      console.log(this.datosVentas);
      return this.datosVentas;
```

```
    },
  },
};
</script>

<style scoped>
.etiquetas {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr 1fr 1fr;
  justify-content: center;
  margin: 5px 0 5px 0;
  background-color: gray;
  color: white;
}
.etiquetas div {
  border: 1px solid black;
}
.ventas-content {
  display: flex;
  font-size: 12px;
  margin: 0 0 5px 0;
}
.ventas-content div {
  width: 19.85%;
  text-align: left;
  padding-left: 15px;
  border-bottom: 1px solid black;
}
</style>
```

Figura 62. Código del componente ventasc

```vue
<template>
  <div>
    <Modelo_IRC />
  </div>
</template>

<script>
import Modelo_IRC from "../components/Modelo_irc.vue";
export default {
  components: { Modelo_IRC },
};
</script>

<style></style>
```

Figura 63. Código de la vista irc

```
<template>
  <div class="content">
    <h1>Inventario de revisión continua</h1>
    <h2 v-if="todos === true">Productos que se requieren</h2>
    <h2 v-if="todos === false">Todos los producatos</h2>
    <button @click="examinar()">
      <div></div>
      Actualizar</button
    ><br />
    <button v-if="todos === true" @click="todos = false">
      Mostrar todos los productos
    </button>
    <button v-if="todos === false" @click="todos = true">
      Mostrar productos que se requieren
    </button>
    <br /><br />
    <div class="inventarios-content">
      <div class="datos-content">
        <div class="etiquetas">
          <div>Proveedor</div>
          <div>Articulo</div>
          <div>Cantidad en Stock</div>
          <div>Demanda Diaria Esperada</div>
          <div>Demanda Esperada</div>
          <div>Stock de seguridad</div>
          <div>Politica optima de inventario</div>
          <div>Cantidad Optima de Pedido según D diaria</div>
          <div>Cantidad Total Optima de Pedido</div>
          <div>Costo del Modelo</div>
        </div>
        <div v-if="todos === true">
          <div v-for="(item, index) in productoSeleccionado" :key="index" class="datosR">
            <div
              class="productos-requeridos"
              v-if="item[0][6] <= respuestas[index][0] || item[0][6] === 0"
            >
              {{ item[0][1] }}
            </div>

            <div
              class="productos-requeridos"
              v-if="item[0][6] <= respuestas[index][0] || item[0][6] === 0"
            >
              {{ item[0][5] }}
            </div>
```

```
<div
  class="productos-requeridos"
  v-if="item[0][6] <= respuestas[index][0] || item[0][6] === 0"
>
  {{ item[0][6] }}
</div>

<div
  class="productos-requeridos"
  v-if="item[0][6] <= respuestas[index][0] || item[0][6] === 0"
>
  {{ respuestas[index][3] }}
</div>

<div
  class="productos-requeridos"
  v-if="item[0][6] <= respuestas[index][0] || item[0][6] === 0"
>
  {{ respuestas[index][1] }}
</div>

<div
  class="productos-requeridos"
  v-if="item[0][6] <= respuestas[index][0] || item[0][6] === 0"
>
  {{ respuestas[index][0] }}
</div>

<div
  class="productos-requeridos"
  v-if="item[0][6] <= respuestas[index][0] || item[0][6] === 0"
>
  {{ respuestas[index][2] }}
</div>

<div
  class="productos-requeridos"
  v-if="item[0][6] <= respuestas[index][0] || item[0][6] === 0"
>
  {{ respuestas[index][4] }}
</div>
<div
  class="productos-requeridos"
  v-if="item[0][6] <= respuestas[index][0] || item[0][6] === 0"
>
```

```
            $ {{ respuestas[index][6] }}
          </div>
        </div>
      </div>
      <div v-if="todos === false">
        <div v-for="(item, index) in productoSeleccionado" :key="index" class="datos">
          <div>
            {{ item[0][1] }}
          </div>

          <div>
            {{ item[0][5] }}
          </div>

          <div>
            {{ item[0][6] }}
          </div>

          <div>
            {{ respuestas[index][3] }}
          </div>

          <div>
            {{ respuestas[index][1] }}
          </div>

          <div>
            {{ respuestas[index][0] }}
          </div>

          <div>
            {{ respuestas[index][2] }}
          </div>

          <div>
            {{ respuestas[index][4] }}
          </div>
          <div>
            {{ respuestas[index][5] }}
          </div>
          <div>$ {{ respuestas[index][6] }}</div>
        </div>
      </div>
    </div>
```

```
  </div>
</template>

<script>
export default {
  name: "Modelo_IRC",
  data() {
    return {
      todos: true,
      productos: null,
      articulo: null,
      productoSeleccionado: [],
      calculos: null,
      estado: false,
      lista1: [],
      respuestas: [],
      getRespuestas: [],
      productosRequeridos: 0,
    };
  },
  created() {
    this.datoIn();
    //this.datosCarga();
    setTimeout(() => {
      this.examinar();
    }, 3000);
  },
  methods: {
    async datoIn() {
      let response = await fetch(`http://127.0.0.1:3398/all-datos`);
      this.productos = await response.json();
      for (let index = 0; index < this.productos.length; index++) {
        const tiempoR = this.productos[index][3];
        const codigoR = this.productos[index][4];
        this.lista1.push([codigoR, tiempoR]);
        console.log(this.lista1);
      }

      return JSON.parse(this.products);
    },
    async examinar() {
      this.productoSeleccionado = [];
      for (let index = 0; index < this.lista1.length; index++) {
        const codigoR = this.productos[index][4];
        let response = await fetch(`http://127.0.0.1:3398/datos-proveedor/${codigoR}`);
```

```
        let dato = await response.json();
        this.productoSeleccionado.push(dato);
        console.log(this.productoSeleccionado);
      }
      this.datosCarga();
      return JSON.parse(this.dato);
    },

    async datosCarga() {
      for (let index = 0; index < this.lista1.length; index++) {
        const element = this.lista1[index];
        let response = await fetch(`http://127.0.0.1:3398/${element[0]}/${element[1]}`);
        this.getRespuestas = await response.json();
        this.respuestas.push(this.getRespuestas);
      }
      console.log(this.respuestas);

      //this.datosCompletos();
      this.estado = true;
      this.alerta();

      return JSON.parse(this.getRespuestas);
    },

    alerta() {
      var a = document.getElementsByClassName("productos-requeridos").length;
      console.log("divs", a);
      return this.productosRequeridos;
    },
  },
  computed: {},
};
</script>

<style scoped>
.inventarios-content {
  display: flex;
  justify-content: space-between;
}
.inventarios input {
  margin-bottom: 5px;
}
```

```css
.datos-content {
  font-size: 12px;
}
.etiquetas {
  display: flex;
  background-color: gray;
  color: white;
}
.etiquetas div {
  width: 136px;
  border: 1px solid black;
  padding: 5px;
}

.datos {
  display: flex;
  justify-content: space-evenly;
  flex-wrap: wrap;
  margin-bottom: 5px;
  font-size: 10.5px;
  font-weight: bold;
}
.datos div {
  width: 136px;
  height: 20px;
  padding: 5px;
  border-bottom: 1px solid black;
  border-right: 1px solid black;
}
.datosR {
  display: flex;
  justify-content: space-evenly;
  flex-wrap: wrap;
  font-size: 10.5px;
  font-weight: bold;
}
.datosR div {
  width: 137.1px;
  height: 20px;
  padding: 5px;
  border-bottom: 1px solid black;
  border-right: 1px solid black;
}
```

```
}
button {
  display: flex;
  background-color: gray;
  border-radius: 15px;
  color: white;
}
button div {
  width: 15px;
  height: 15px;
  border-radius: 50%;
  border: 3px white;
  border-style: dotted dashed solid double;
  margin: 1px;
}
</style>
```

Figura 64. Código del componente modelo_irc

```vue
<template>
  <div>
    <GraficasEstadisticas />
  </div>
</template>

<script>
import GraficasEstadisticas from "../components/GraficasEstadisticas";

export default {
  components: { GraficasEstadisticas },
};
</script>

<style></style>
```

Figura 65. Código de la vista estadísticas

```
<template>
  <div><h1>Estadísticas</h1></div>
  <input
    type="text"
    name="articulo"
    id="articulo"
    v-model="articulo"
    @keyup.enter="datos()"
  />
  <button @click="datos()">Graficar</button>
  <div class="content">
    <canvas id="grafica"></canvas>
    <div class="tabla-datos">
      <div class="etiquetas" v-if="estado === true">
        <div>Fecha</div>
        <div>Ventas (Unidades)</div>
      </div>
      <div class="datos">
        <div>
          <div v-for="(item, index) in datosX" :key="index" class="datosX">
            <div>{{ item }}</div>
          </div>
        </div>
        <div>
          <div v-for="(item, index) in datosY" :key="index" class="datosY">
            <div>{{ item }}</div>
          </div>
        </div>
      </div>
    </div>
  </div>
</template>

<script>
export default {
  name: "GraficasEstadisticas",
  data() {
    return {
      estado: false,
      datosVentas: null,
      articulo: null,
      datosX: [],
      datosY: [],
    };
  },
```

```
methods: {
  async datos() {
    let ventas = await fetch(`http://127.0.0.1:3398/get-ventas/${this.articulo}`);
    this.datosVentas = await ventas.json();
    console.log(this.datosVentas);
    this.datosXY();
    return this.datosVentas;
  },
  datosXY() {
    this.datosX = [];
    this.datosY = [];
    for (let index = 0; index < this.datosVentas.length; index++) {
      const x = this.datosVentas[index][0];
      const y = this.datosVentas[index][3];
      this.datosX.push(x);
      this.datosY.push(y);
    }

    this.graficar();
  },
  graficar() {
    const data = {
      labels: this.datosX,
      datasets: [
        {
          label: this.articulo,
          //backgroundColor: "rgb(255, 99, 132)",
          borderColor: "rgb(0, 153, 255)",
          data: this.datosY,
        },
      ],
    };
    const config = {
      type: "line",
      data,
      options: {},
    };

    var grafica = new Chart(document.getElementById("grafica"), config);
    grafica.canvas.parentNode.style.height = "500px";
    grafica.canvas.parentNode.style.width = "900px";
    this.estado = true;
  },
},
};
```

```
<style scoped>
.content {
  display: flex;
  justify-content: space-between;
}
.tabla-datos {
  position: absolute;
  top: 200px;
  right: 50px;
}
.etiquetas {
  display: flex;
  justify-content: space-evenly;
  background-color: gray;
  color: white;
  padding: 2px;
}
.etiquetas div {
  text-align: center;
}
.datos {
  display: grid;
  grid-template-columns: 1fr 1fr;
}
.datosX {
  width: 150px;
  border-bottom: 1px solid black;
  text-align: start;
}
.datosY {
  height: 24px;
  width: 150px;
  border-bottom: 1px solid black;
  padding-top: 12px;
  justify-content: center;
}
</style>
```

Figura 66. Código del componente graficas estadísticas

```vue
<template>
  <div>
    <Configuraciones />
  </div>
</template>

<script>
import Configuraciones from "../components/Configuraciones.vue";
export default {
  components: { Configuraciones },
};
</script>

<style></style>
```

Figura 67. Código de la vista config

```
<template>
  <div><h1>Configuraciones</h1></div>
  <div class="content">
    <div class="etiquetas" v-if="estado === true">
      <div class="etiqueta">
        <b>Nivel de Confianza: </b>
        <div v-if="estado === true">{{ datos[0][0][1] }}%</div>
      </div>
      <div class="etiqueta">
        <b>Costo total al mes de mantener: </b>
        <div v-if="estado === true">{{ datos[0][0][2] }} USD</div>
      </div>
      <div class="etiqueta">
        <b>Costo de mantener por producto: </b>
        <div v-if="estado === true">{{ datos[1] }} USD</div>
      </div>
    </div>
    <form
      action="http://127.0.0.1:3398/datos-config-update"
      method="Post"
      name="form"
      enctype="multipart/form-data"
    >
      <input
        type="text"
        name="nivel"
        id="nivel"
        placeholder="Actualizar Nivel de confianza"
      /><br /><br />
      <input
        type="text"
        name="costoM"
        id="costoM"
        placeholder="Actualizar Costo de mantener"
      /><br /><br />
      <input type="submit" value="Actualizar" />
    </form>
  </div>
</template>
```

```
<script>
export default {
  name: "Configuraciones",
  data() {
    return {
      datos: null,
      estado: false,
    };
  },
  created() {
    this.getDatos();
  },
  methods: {
    async getDatos() {
      let response = await fetch("http://127.0.0.1:3398/datos-config");
      this.datos = await response.json();
      this.estado = true;
      return JSON.parse(this.datos);
    },
  },
};
</script>

<style scoped>
form {
  padding-top: 10px;
  width: 300px;
}

.content {
  display: flex;
  text-align: start;
  padding: 50px 0 0 200px;
}
.etiquetas {
  text-align: start;
  padding-top: 12px;
}
.etiquetas div {
  display: flex;
  justify-content: space-between;
  /*border: 1px solid black;*/
}

}
.etiqueta {
  margin: 0 15px 22px 0;
}
.etiqueta div {
  margin-left: 5px;
}
</style>
```

Figura 68. Código del componente configuraciones

Anexo 5. Plantillas de Excel

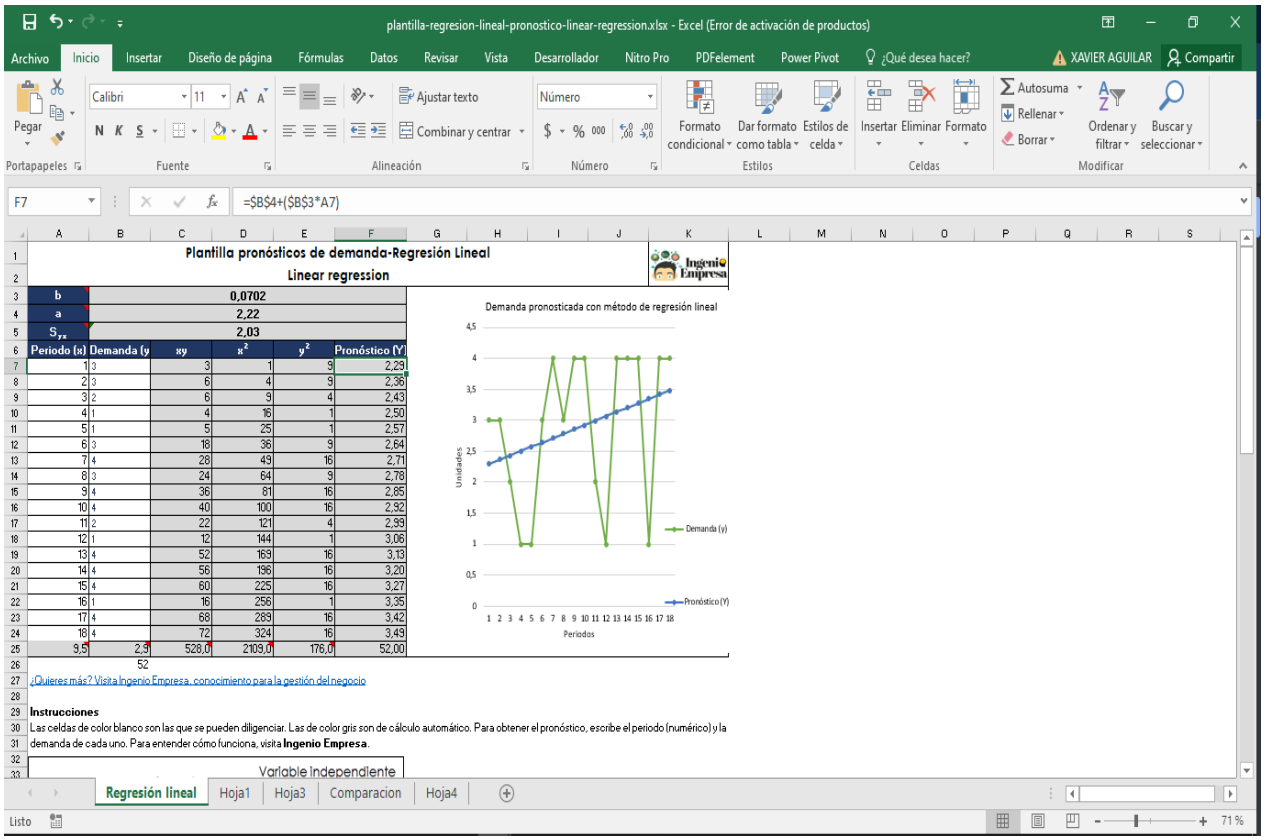
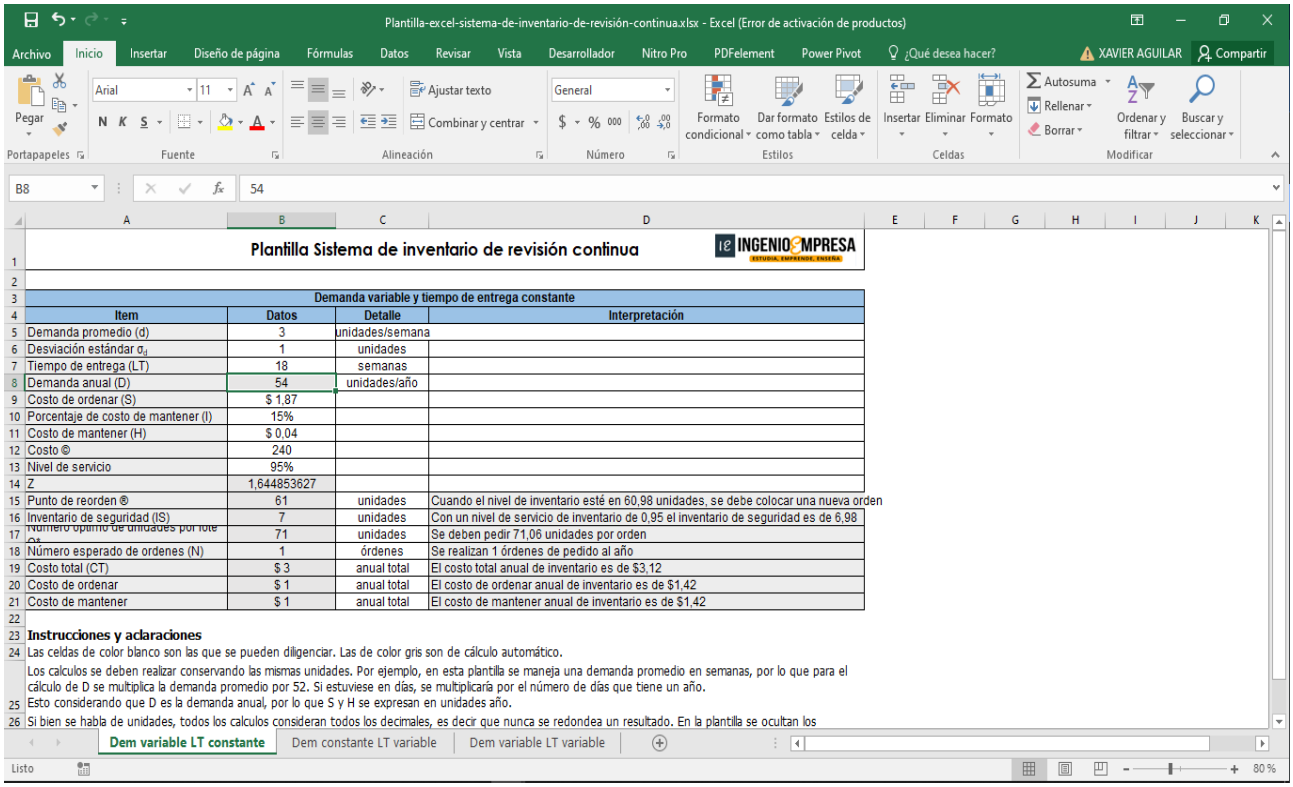


Figura 69. Plantilla del método de regresión lineal



Item	Datos	Detalle	Interpretación
Demanda promedio (d)	3	unidades/semana	
Desviación estándar σ_d	1	unidades	
Tiempo de entrega (LT)	18	semanas	
Demanda anual (D)	54	unidades/año	
Costo de ordenar (S)	$ 1,87		
Porcentaje de costo de mantener (I)	15%		
Costo de mantener (H)	$ 0,04		
Costo ©	240		
Nivel de servicio	95%		
Z	1,644853627		
Punto de reorden ®	61	unidades	Cuando el nivel de inventario esté en 60,98 unidades, se debe colocar una nueva orden
Inventario de seguridad (IS)	7	unidades	Con un nivel de servicio de inventario de 0,95 el inventario de seguridad es de 6,98
Número óptimo de unidades por lote	71	unidades	Se deben pedir 71,06 unidades por orden
Número esperado de ordenes (N)	1	órdenes	Se realizan 1 órdenes de pedido al año
Costo total (CT)	$ 3	anual total	El costo total anual de inventario es de $3,12
Costo de ordenar	$ 1	anual total	El costo de ordenar anual de inventario es de $1,42
Costo de mantener	$ 1	anual total	El costo de mantener anual de inventario es de $1,42

Figura 70. Plantilla del modelo de inventarios de revisión continua

Anexo 6. Ficha del modelo de gestión automática de pedidos a proveedores

Ficha

1	**Producto:** SEDAL SH CASPA CONTROL 12X340ML		**Ficha No:** 001
2	Cantidad de pedido	54 unidades	**OBSERVACIONES:**
3	Costo de gestión automática	$ 3.29	**OBSERVACIONES:**
4	Tiempo	18 días	**OBSERVACIONES:**