

FACULTAD DE INGENIERÍA



CARRERA DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

INFORME DE PROYECTO FINAL

Aplicación de Escritorio para la Gestión de Stock y Predicción de Agotamiento en Empresas Proveedoras de Insumos.

**Autores:**

**Fabricio Campoverde Martinez (100%)**

**Ramos Zevallos Benjamin Franklin(100%)**

**Santiago Junnior Castillo Tineo (100%)**

**Anibal Tarazona Mallqui (100%)**

**Azaña Tejada Marcial Felix (80%)**

**Curso:**

Técnicas de Programación Orientada Objetos

**Docentes del Curso:**

Tarmeño Juscamaita, Esther

Jorge Chicana Aspajo

Lima – Perú

2025-1

**Tabla de contenido**

[Historial de Versiones 5](#_klaxb9x65x5)

[Información del Proyecto 6](#_gd2a9mnrl7m9)

[Aprobaciones 6](#_8pzsmrv01t9t)

[1.](#_fcpfbdg21bwm) Descripción del problema 7

[2.](#_4vbt40uqc3kr) Antecedentes 8

[3.](#_eb7zsppkl3ok) Restricciones del proyecto 9

[3.1.](#_izcc95m159ch) Identificación y análisis de todas las restricciones, criterios y suposiciones 9

[3.2.](#_4w9ovvmh1bo2) Generación y análisis de múltiples soluciones 10

[3.3.](#_e4z7u6fqg0a9) Criterios para seleccionar la solución 11

[4.](#_skiq8mq35lha) Alcance del producto 13

[5.](#_8k0r8wv48i38) Objetivo general 14

[6.](#_ubnszpumcebf) Objetivos específicos 14

[7.](#_ogz2b8h5innw) Estructuras de desglose del proyecto 14

[7.1.](#_fepyw1m32b4n) Análisis y Diseño del Sistema 15

[7.2.](#_x3gtzaa0ud7) Desarrollo de la Interfaz Gráfica 15

[7.3.](#_lu0p4h2hxax3) Implementación de la Lógica del Negocio 15

[7.4.](#_q4temoi3j2ba) Manejo de Datos y Persistencia 15

[7.5.](#_bbubh2p8nl76) Pruebas del Sistema 15

[7.6.](#_orl8ezkt5571) Documentación del Proyecto 15

[8.](#_d6ssjv5swq6i) Herramientas y tecnología utilizada 16

[8.1.](#_c35b3t4scxga) Colecciones 16

[8.2.](#_50mcm4ppab8k) Archivos en Java 16

[8.3.](#_ron0l8hadiu4) Librerías gráficas 16

[8.4.](#_to4pgz9xyx1s) Criterios de calidad de software (ISO 9126) 17

[9.](#_ni5f7rd5ugwd) Historias de Usuario y Funcionalidades del producto 17

[10.](#_ly3oxeo4s87j) Diagrama de clases y relaciones 19

[Clase Login 19](#_818tiqc7rdq2)

[Formulario FrmLogin 19](#_dnc7pod38ule)

[Clase Persona 19](#_unlsiu73ig42)

[Clase RegistroEmpleados 20](#_6xn3kcybupaa)

[Clase MantenimientoEmpleado 20](#_a111zo2o3tel)

[Formularios Relacionados: 20](#_37w6n9o21t3z)

[Clase RegistroClientes 20](#_8wyfpne6ks2d)

[Clase MantenimientoClientes 20](#_8ntxdbygmh43)

[Formularios Relacionados: 20](#_qj78hqhhw782)

[Clase Buses 21](#_6wemije6c1rm)

[Formularios Relacionados: 21](#_a8ivmijmu67d)

[Clase Rutas 21](#_725kk29srhpu)

[Formularios Relacionados: 21](#_52rytalsykaw)

[Clase Logística 21](#_kojpg9rr8mv9)

[Formularios Relacionados: 21](#_3u1ptyn5nsvu)

[Clase MantenimientoFinanzas 21](#_butdyxbelqie)

[Clase MantenimientoEmpleado (RRHH) 21](#_jcj2jss8ryux)

[Clase EvaluadorRutas 22](#_3cd6gc9g9edz)

[Clase AsignacionesRuta 22](#_ei2z12k1cy8h)

[11. Clases y características de usuarios principales 22](#_415tapqaw3cl)

[12.](#_2k769wsopz70) Requerimientos funcionales 24

12.1. Autenticación y control de acceso por roles 24

12.2. Gestión de usuarios y empleados 25

12.3. Gestión de clientes 25

12.4. Gestión de Insumos y Proveedores 26

12.5. Módulo de Predicción y Alertas (Core del Proyecto) 27

12.6. Gestión de Ventas y Finanzas 29

12.7. Trazabilidad de Inventario y Reportes 29

[13.](#_ynt3b0olnu3s) Reglas de negocio 30

13.1.Control de acceso basado en roles 30

13.2.Validación de datos en todos los formularios 30

13.3. Notificación por cada operación importante 30

13.4. Formato de datos estandarizado 30

13.5. Integridad y confidencialidad de la información 30

13.6. Restricción de Stock Negativo 30

13.7. Precisión en la Predicción 30

[14.](#_fhk0bim64tze) Requerimientos de interfaces externas 31

[14.1.](#_bqjnykc382if) Características generales de la interfaz 31

[14.2.](#_hd8mvpxrci8k) Estándares de Interfaz Gráfica (GUI) 31

[15.](#_3uppwio6ipp2) Requerimientos no funcionales 32

[16.](#_cufk6jmkd205) Glosario 32

[Término 32](#_ys6078s4anf)

[Definición 32](#_lgoy09hcbzv2)

[GUI (Graphical User Interface) 32](#_qllbg6ivl2mq)

[Interfaz Gráfica de Usuario. Parte visual del sistema con la que interactúa el usuario mediante elementos como botones, menús, ventanas, etc. 32](#_qnrxg0m3b21g)

[UI (User Interface) 32](#_a4yu0h58ipn1)

[Interfaz de Usuario. Término más general que engloba tanto interfaces gráficas como interfaces por línea de comandos o de voz. 32](#_z7xxqgb2gyr8)

[CRUD 32](#_70n41o5c3zqw)

[Acrónimo de Create, Read, Update, Delete. Conjunto de operaciones básicas para la gestión de datos en una base de datos o sistema. 32](#_vmx8t08y33wm)

[Scrum 33](#_2ha0wux9kor2)

[Marco de trabajo ágil para el desarrollo de software. Se basa en iteraciones cortas llamadas sprints, roles definidos (Scrum Master, Product Owner) y reuniones periódicas. 33](#_r3kg4oskqpki)

[Sprint 33](#_yiepefrzkrpl)

[Iteración o ciclo corto de trabajo en Scrum (usualmente de 1 a 4 semanas) en la que se desarrolla una parte funcional del sistema. 33](#_wgjh5xur5abd)

[Product Backlog 33](#_ez9ah6j68wlw)

[Lista priorizada de requerimientos o funcionalidades que se deben implementar en el sistema. 33](#_j04mx7qy483f)

[Sprint Backlog 33](#_yqkg9zubil7q)

[Subconjunto del Product Backlog que se selecciona para trabajar en un sprint específico. 33](#_oeoaknjowv9l)

[17.](#_5gkyjqovasow) Bibliografía 33

[18.](#_ej1mjiunc7m5) Anexos 35

# Historial de Versi nes

# 

# 

| **Fecha** | **Versión** | **Autor** | **Organización** | **Descripción** |
| --- | --- | --- | --- | --- |
| **15/10/2025** | **1.0** | **Todos los integrantes del equipo** | **POO – UPN** | **Reunión inicial para definir el problema del Minimarket “Kathycita” y acordar el alcance del sistema de inventario.** |
| **16/10/2025** | **1.1** | **Todos los integrantes del equipo** | **POO – UPN** | **Investigación de antecedentes sobre sistemas de stock, modelos predictivos y estructuras de inventario** |
| **18/10/2025** | **2** | **Scrum Maste** | **POO – UPN** | **Planificación del Sprint 1 y elaboración del Product Backlog inicial.** |
| **19/10/2025** | **3** | **Analyst Programmers** | **POO – UPN** | **Diseño y programación de las clases base: Producto, Categoría, Proveedor y Stock.** |
| **20/10/2025** | **4** | **Analyst Programmers** | **POO – UPN** | **Implementación del módulo de Registro de Productos y Proveedores (ArrayList → MySQL)** |
| **25/10/2025** | **5** | **Analyst Programmers** | **POO – UPN** | **Creación del formulario de ingreso de nuevos productos y conexión a MySQL.** |
| **25/10/2025** | **6** | **Analyst Programmers** | **POO – UPN** | **Implementación del módulo de actualización de stock por entradas y salidas.** |
| **25/10/2025** | **7** | **Analyst Programmers** | **POO – UPN** | **Construcción del módulo de historial de movimientos del inventario** |
| **28/10/2025** | **8** | **Scrum Master** | **POO – UPN** | **Reunión con la cliente (Katherine Tineo Arenas) para validar la estructura del inventario y recopilar mejoras.** |
| **29/10/2025** | **9** | **Analyst Programmers** | **POO – UPN** | **Creación del módulo de alertas por stock mínimo** |
| **01/11/2025** | **10** | **Analista Funcional / Diseño UX** | **POO – UPN** | **Diseño de la interfaz gráfica: formularios, botones, tablas y experiencia de usuario.** |
| **03/11/2025** | **11** | **Analyst Programmers** | **POO – UPN** | **Pruebas internas: verificación de campos, validaciones y correcto almacenamiento en la base MySQL.** |
| **05/11/2025** | **12** | **Analyst Programmers** | **POO – UPN** | **Corrección de errores en consultas SQL y validaciones de predicción.** |
| **07/11/2025** | **13** | **Scrum Master** | **POO – UPN** | **Reunión de avance y demostración de la primera versión del módulo predictivo al cliente** |
| **09/11/2025** | **14** | **Analyst Programmers** | **POO – UPN** | **Implementación del ranking de productos próximos a agotarse** |

# 

# Información del Proyecto

# 

| Proyecto | Aplicación de Escritorio para la Gestión de Inventario y Predicción de Agotamiento en el Minimarket “Kathycita” |
| --- | --- |
| Fecha de preparación | 24/04/2025 |
| Cliente | Propietaria del negocio: **Katherine Tineo Arenas** |
| Patrocinador principal | **Katherine Tineo Arenas** (Dueña del Minimarket “Kathycita”) |
| Gerente / Líder de Proyecto | **Fabricio Campoverde Martínez** (Scrum Master / Líder del Proyecto) |
| Miembros del equipo | **Analista Funcional / Diseño UX:**  • Ramos Zevallos Benjamin Franklin  **Analyst Programmers:**  • Santiago Junnior Castillo Tineo  • Anibal Tarazona Mallqui |
|

# Aprobaciones

| **Apellidos y Nombres** | **Cargo** | **Departamento u Organización** | **Fecha** | **Firma** |
| --- | --- | --- | --- | --- |
| Tineo Arenas, Katherine | **Administradora/ Cliente** | **Minimarket “Kathycita”** | **24-09** |  |
| Marcial Felix Azaña Tejada | **Scrum Master** | **Grupo de Desarrollo** | **24-09** |  |
| Ramos Zevallos Benjamin Franklin | **Analista Funcional /**  **Diseño UX** | **Grupo de Desarrollo** | **24-09** |  |
| Fabricio Campoverde Martinez | **Analista Funcional /**  **Diseño UX** | **Grupo de Desarrollo** | **24-09** |  |
| Santiago Junnior Castillo Tineo | **Analyst Programmers** | **Grupo de Desarrollo** | **24-09** |  |
| Anibal tarazona Mallqui | **Analyst Programmers** | **Grupo de Desarrollo** | **24-09** |  |

# Descripción del problema

El avance constante de la digitalización empresarial ha transformado profundamente la manera en que las organizaciones gestionan sus operaciones. En particular, la administración del inventario y stock de proveedores se ha convertido en un factor estratégico esencial para mantener la competitividad y asegurar la continuidad de los procesos productivos. Sin embargo, muchas empresas aún realizan estos controles de manera manual o fragmentada, lo que provoca ineficiencias, demoras en la reposición y pérdidas económicas derivadas de la falta de visibilidad del inventario disponible.

Según García y Torres (2022), uno de los principales retos de las pequeñas y medianas empresas es la ausencia de un sistema que centralice la información del stock y alerte oportunamente sobre los niveles mínimos de existencia. Esto genera fallas en la planificación de compras y desabastecimiento en productos de alta rotación, afectando tanto la productividad como la satisfacción del cliente final.

Del mismo modo, Vargas (2023) sostiene que el uso de tecnologías de programación orientada a objetos, combinadas con bases de datos relacionales como MySQL, permite crear soluciones escalables que automatizan la gestión de inventarios y registran el movimiento de mercancías en tiempo real. No obstante, la mayoría de las organizaciones aún no integran herramientas de predicción de agotamiento, las cuales podrían anticipar las necesidades de reposición en función de las ventas pasadas.

En este contexto, la falta de un sistema de control de stock con predicción automatizada provoca una gestión reactiva (en lugar de preventiva) del inventario. Esto se traduce en compras tardías, exceso o escasez de productos y dificultad para calcular el nivel de rotación de los proveedores.

Por ello, se propone el desarrollo de una aplicación de escritorio programada en Java con conexión a MySQL Workbench, que permita registrar el stock por proveedor, emitir alertas de agotamiento y predecir qué productos se agotarán próximamente, tomando como referencia las ventas registradas en el último mes.

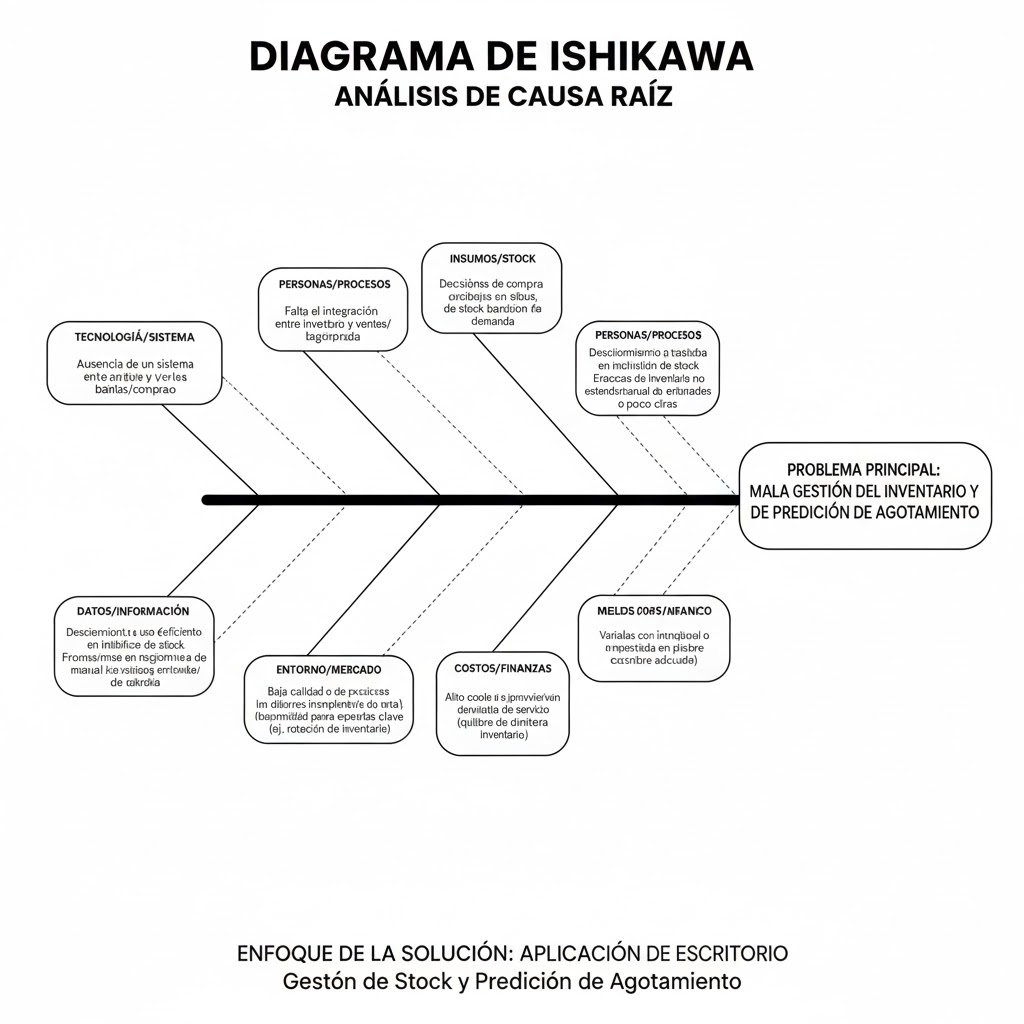
Así, el problema central de esta investigación se resume en las siguientes pregunta:

¿Cuáles son las consecuencias de no contar con un sistema centralizado y automatizado para el registro, control y seguimiento del stock de proveedores en una empresa de distribución, y de qué manera la implementación de una aplicación de escritorio desarrollada en Java y conectada a una base de datos MySQL podría optimizar la gestión del inventario y la toma de decisiones en los procesos de reposición?

Esta interrogante busca comprender cómo la falta de automatización afecta la disponibilidad de productos, el tiempo de respuesta ante el agotamiento del stock y la precisión en los registros de ventas y compras. Asimismo, pretende demostrar que el uso de una aplicación desarrollada en Java puede mejorar significativamente la eficiencia operativa, reducir errores humanos y proporcionar alertas predictivas sobre los productos que están próximos a agotarse.

En base a esta pregunta se elaboró el diagrama de Ishikawa, el cual permite identificar las principales causas que originan los problemas de control y predicción del stock dentro de la organización.

Diagrama de Ishikawa:



# Antecedentes

El estudio realizado por Sánchez (2022) tuvo como objetivo desarrollar un sistema de gestión de inventarios utilizando Java y MySQL Workbench para una empresa distribuidora de alimentos en Lima. Este proyecto se planteó bajo un enfoque cuantitativo y aplicó la metodología ágil Scrum durante su desarrollo. Entre sus resultados, se logró reducir en un 42 % los errores en el registro manual de productos y optimizar los tiempos de reposición de insumos. La investigación demostró que la automatización de procesos mejora considerablemente la trazabilidad del stock y la eficiencia operativa.

Fuente: Sánchez, M. (2022). Sistema de gestión de inventarios en Java y MySQL. Universidad Nacional Mayor de San Marcos.

Asimismo, Ramírez y Torres (2023) desarrollaron un modelo predictivo orientado al control de inventarios minoristas, implementando el suavizamiento exponencial simple (SES) como método de pronóstico de demanda. El estudio, publicado en la Revista Colombiana de Computación, tuvo un enfoque cuantitativo aplicado y demostró una precisión del 92 % en la predicción de productos próximos a agotarse. Los autores concluyeron que el uso de modelos de pronóstico a corto plazo permite anticipar compras y evitar pérdidas por quiebre de stock.

Fuente: Ramírez, J. & Torres, F. (2023). Modelos predictivos para inventarios minoristas. Revista Colombiana de Computación, 19(2), 45–60.

Del mismo modo, Velasco (2021) propuso una aplicación de escritorio para la gestión de stock mínimo y emisión de alertas automáticas en una empresa farmacéutica de Guayaquil. El sistema fue desarrollado en Java SE con conexión a una base de datos MySQL, y su enfoque fue aplicado y no experimental. La muestra estuvo compuesta por tres sucursales donde se implementó el sistema, logrando una disminución del 30 % en los quiebres de inventario y un aumento en la precisión del control de existencias.

Fuente: Velasco, D. (2021). Desarrollo de un sistema de control de inventario con alertas automáticas de stock. Universidad de Guayaquil.

Por otro lado, Cortés (2022) llevó a cabo una investigación sobre el desarrollo de un algoritmo predictivo de agotamiento de stock basado en el consumo diario promedio de los últimos treinta días. Este modelo se implementó en Apache NetBeans y se validó utilizando el Error Absoluto Medio Porcentual (MAPE), alcanzando una desviación menor al 10 %. El proyecto demostró que los sistemas de predicción de consumo, combinados con bases de datos actualizadas, permiten reducir tiempos de espera y mejorar la toma de decisiones logísticas.

Fuente: Cortés, P. (2022). Algoritmo predictivo de agotamiento de stock usando ventas mensuales. Universidad de Antioquia.

Finalmente, Mendoza y Pérez (2024) desarrollaron un sistema de gestión de proveedores integrando MySQL Workbench y Apache NetBeans para mejorar la trazabilidad de productos y el rendimiento de los procesos de abastecimiento. Su estudio, publicado en la *Revista Peruana de Ingeniería Informática*, aplicó un enfoque mixto y se probó en 15 empresas textiles de Arequipa. Los resultados mostraron un incremento del 37 % en la eficiencia del proceso de reposición y una mejora del 20 % en la precisión de las compras por proveedor.

Fuente: Mendoza, L. & Pérez, R. (2024). *Integración de bases de datos MySQL en sistemas Java para control de proveedores*. *Revista Peruana de Ingeniería Informática*, 12(1), 15–28.

En síntesis, los estudios revisados coinciden en que la implementación de aplicaciones desarrolladas en Java con bases de datos relacionales MySQL contribuye significativamente a la eficiencia operativa en la gestión de inventarios. Además, el uso de modelos de predicción de corto plazo basados en ventas mensuales proporciona una herramienta efectiva para anticipar el agotamiento del stock y optimizar la planificación de compras.  
 Por tanto, el presente proyecto busca integrar ambas perspectivas: la automatización del control de stock y la predicción de agotamiento de productos, mediante una aplicación de escritorio que centralice la información de proveedores y ventas en tiempo real.

# Restricciones del proyecto

* 1. **Identificación y análisis de todas las restricciones, criterios y suposiciones**

**Restricciones:**

* **Económicas:** El presupuestoinicial puede ser limitante para la adquisición de licencias de software empresarial o servidores dedicados. Se priorizarán herramientas de desarrollo de código abierto (como las mencionadas) para mitigar este impacto.
* **Tecnológicas:** La empresa proveedora puede tener una infraestructuratecnológicainsuficiente (hardware antiguo o red inestable), requiriendo una inversión mínima en equipos actualizados y conectividad para soportar una aplicación de escritorio y su base de datos.
* **Regulatorias:** Cumplir con las normativas de proteccióndedatos (ej. Ley de Protección de Datos Personales) respecto al manejo de información de clientes y proveedores, además de la necesidad de mantener un registro contable preciso para fines tributarios.
* **Operativas:** La implementación del sistema de stock debe ser gradual para no interrumpir el flujo constante de recepción y despacho de insumos, manteniendo las operaciones de la cadena de suministro activas.

**Criterios**

* **Reducción de errores humanos:** Minimizar errores en el conteo, registro de entrada/salida y reabastecimiento de insumos.
* **Mejora en la trazabilidad y registro:** Facilitar el seguimiento de cada lote o SKU de insumo, desde su recepción hasta su uso/venta.
* **Accesibilidad y facilidad de uso:** Interfaz de escritorio intuitiva y rápida para el personal de almacén y ventas.
* **Escalabilidad del sistema:** Capacidad de manejar un incremento en el volumen de ítems (nuevos insumos) y transacciones futuras.
* **Capacidad de Predicción:** Generación de alertas y reportes para predecir el agotamiento de stock (*"stock-out"*).
* **Suposiciones**
* La Gerencia está dispuesta a invertir de forma controlada y apoya la modernización de los procesos de inventario.
* El personal de almacén y ventas puede ser capacitado con tiempo y recursos adecuados en el uso de la nueva aplicación.
* Los datos de inventario y proveedores existentes pueden ser migrados desde formatos manuales o de hoja de cálculo (Excel) a la base de datos SQL del sistema.
  1. **Generación y Análisis de Soluciones**
  2. La solución seleccionada, aunque deriva de una propuesta inicial de logística, se adapta específicamente para crear una herramienta de gestión de inventario para el sector de **Empresas Proveedoras de Insumos**.

**Propuesta 02 (Adaptada): Aplicación de Escritorio para la Gestión de Stock y Predicción de Agotamiento**

| **Aspecto** | **Detalle** |
| --- | --- |
| **Ventajas** | **Centraliza** el inventario, ventas y proveedores en un único sistema. **Automatiza** el cálculo de punto de reorden y **predice** el agotamiento de stock (*stock-out*), un valor crítico. **Reduce errores** en el manejo de inventario y optimiza costos al evitar stock de seguridad excesivo. Cumple directamente con el objetivo del proyecto final. |
| **Desventajas** | Requiere una **inversión inicial controlada** en hardware o red para asegurar la estabilidad del servidor de base de datos. Necesita una **migración y validación** exhaustiva de los datos de inventario existentes. |

* 1. **Selección de la Solución**

La **Propuesta 02 (Adaptada)** es la alternativa más factible y adecuada para el proyecto, obteniendo la mayor puntuación en el análisis multicriterio (Total: **30**).

| **Criterio de Selección** | **Puntuación (0-10)** | **Justificación para el proyecto de Stock y Predicción** |
| --- | --- | --- |
| **Costos** | 4 (Manejable) | Se priorizan herramientas **gratuitas** (*Java, Apache NetBeans IDE 25, Base de datos SQL libre*), minimizando el gasto en licencias. |
| **Recopilación de información** | 7 (Factible) | La información requerida (insumos, proveedores, tasas de consumo) es **accesible** dentro de la operación de una empresa proveedora. |
| **Conocimiento en el tema** | 8 (Alto) | El equipo posee **alta familiaridad** con sistemas de **logística** y **gestión de *supply chain***, conocimiento directamente aplicable. |
| **Tiempo de desarrollo** | 6 (Controlable) | La metodología **SCRUM** permite desarrollar de forma progresiva, enfocando primero los módulos básicos (Registro, Stock) para luego integrar la **Predicción**. |
| **Recursos disponibles** | 5 (Suficientes) | Se cuenta con el *stack* tecnológico (**Java/NetBeans/SQL**) y el acceso a la empresa proveedora como fuente de requisitos. |
| **Total** | **30** | **La propuesta más robusta y alineada al objetivo de optimización de inventario.** |

* 1. **Análisis de Impactos**

La implementación de la Aplicación de Escritorio generará los siguientes impactos en la organización proveedora de insumos:

| **Sector** | **Impacto de la Gestión de Stock y Predicción** |
| --- | --- |
| **Impacto Social** | Mejora la **calidad del servicio** al cliente final al asegurar la disponibilidad de insumos. Reduce la frustración operativa del personal por *stock-outs* e incrementa la **confianza** en los procesos internos. |
| **Impacto Cultural** | Impulsa una cultura organizacional de **gestión de datos** y **toma de decisiones preventiva** basada en la predicción. Fomenta la **mejora continua** en los procesos de almacén y reabastecimiento. |
| **Impacto Ético** | Promueve la **transparencia** en el manejo del inventario y el control de mermas. Garantiza un trato **justo** y **ordenado** hacia los proveedores al generar pedidos de insumo oportunos. |
| **Impacto Económico** | Genera **ahorro considerable** al optimizar el capital de trabajo (evitar *stock* excesivo) y **evita pérdidas por *stock-out*** (pérdida de ventas) gracias al módulo de **predicción**. |

**Alcance del Producto**

El alcance del proyecto es el desarrollo de una Aplicación de Escritorio completa y funcional para la gestión de insumos:

* **Funcionalidades Primarias:** Permitirá el registro, control y supervisión de insumos, proveedores, y clientes.
* **Operaciones Clave:** Incluye módulos para: Registro de Insumos, Control de Stock (Entradas/Salidas), Registro de Ventas/Despachos, Gestión de Usuarios (roles definidos) y Gestión de Proveedores.
* **Característica Distintiva (Valor Agregado):** Implementación de un Módulo de Predicción que, usando el historial de consumo, calcula el Punto de Reorden y emite alertas sobre el tiempo estimado para el agotamiento (*stock-out*) de insumos críticos.
* **Tecnología:** Desarrollado en Java con Apache NetBeans IDE 25, utilizando una base de datos SQL para asegurar el almacenamiento seguro y rápido de la información.
* **Calidad:** El diseño estará orientado a la facilidad de uso, seguridad de datos y mantenimiento del sistema, cumpliendo con los principios de calidad del software establecidos.
* **Desventajas:**
  + Requiere inversión inicial en hardware y/o red para asegurar la estabilidad del servidor de base de datos.

# Objetivo general

* Diseñar e implementar un **sistema informático de gestión de inventario y ventas**, que permita **automatizar el control de productos, usuarios y transacciones**, garantizando la **integridad de los datos**, la **eficiencia operativa** y la **facilidad de uso** para los administradores y vendedores del negocio.

# Objetivos específicos

* Desarrollar un módulo de **inicio de sesión seguro** con control de acceso por roles.
* Implementar un módulo para el **registro, modificación y eliminación de productos**.
* Permitir el **registro automático de ventas** con actualización del stock en tiempo real.
* Crear un módulo de **reportes** que muestre información sobre inventario y ventas.
* Establecer **validaciones y controles** que garanticen la integridad de los datos.
* Diseñar una **interfaz gráfica intuitiva** utilizando componentes de Java Swing.
* Conectar el sistema a una **base de datos SQL** para el almacenamiento de información.
* Establecer **validaciones y controles** que garanticen la integridad de los datos.
* Diseñar una **interfaz gráfica intuitiva** utilizando componentes de Java Swing.
* Conectar el sistema a una **base de datos SQL** para el almacenamiento de información.

# Estructuras de desglose del proyecto



La siguiente estructura presenta la descomposición jerárquica del proyecto, permitiendo una planificación organizada y un control más preciso del desarrollo del sistema

* 1. **Análisis y Diseño del Sistema**
* Levantamiento de requisitos funcionales y no funcionales del minimarket.
* Identificación de los artículos de alta rotación y productos críticos.
* Diseño de la base de datos relacional en SQL (tablas de productos, ventas, stock y usuarios).
* Elaboración de diagramas UML (caso de uso, clases y secuencia).
  1. **Desarrollo de la Interfaz Gráfica**
* Diseño de formularios amigables y organizados en NetBeans.
* Formulario de inicio de sesión y panel principal (FrmLogin.java).
* Formularios de registro de productos, categorías, ventas y proveedores (FrmRegistroProducto.java, FrmRegistroVenta.java, FrmRegistro Proveedor,java).
* Interfaz de predicción de stock estimación de agotamiento (FrmPredccionStock.java).
* Menús principales según el tipo de usuario (Administrador y Vendedor).
  1. **Implementación de la Lógica del Negocio**
* Programación del sistema en Java mediante NetBeans.
* Validación de datos en los formularios de registro y venta.
* Control de acceso por tipo de usuario (Administrador, Vendedor).
* Automatización de alertas y reabastecimiento.
* Generación automática de códigos de producto.
* Navegación entre formularios de inventario. ventas y reportes.
  1. **Manejo de Datos y Persistencia**
* Conexión del sistema con la base de datos SQL.
* Inserción, actualización y eliminación de registros de productos y ventas.
* Métodos de consulta de movimiento para alimentar la protección.
* Registro histórico de movimiento para alimentar la predicción.
* Generación de reportes de inventario y productos próximos a agotarse.
  1. **Pruebas del Sistema**
* Pruebas unitarias de módulos y conexión en SQL.
* Pruebas funcionales de registro, búsqueda, venta y predicción.
* Simulación del flujo completo: venta -> actualización -> alerta.
* Corrección de errores y optimización del rendimiento del sistema.
  1. **Documentación del Proyecto**
* Informe técnico del sistema (arquitectura, diseño y requerimientos).
* Manual de instalación y funcionamiento del sistema.
* Capturas de pantalla del sistema en ejecución.
* Conclusiones y recomendaciones finales

# Herramientas y tecnología utilizada

En el desarrollo del sistema se implementaron distintas herramientas y tecnologías que permitieron lograr un software funcional, modular y visualmente comprensible. A continuación, se detalla cada una de ellas:

* 1. **Colecciones** **en Java**

Las colecciones en Java son estructuras que permiten almacenar y manipular conjuntos de datos de manera eficiente. Según **Oracle (s.f.)**, “una colección es un objeto que agrupa múltiples elementos en una sola unidad y permite almacenar, recuperar, manipular y comunicar datos agregados”.

En nuestro proyecto se utilizaron para gestionar la información de los clientes y empleados antes de almacenarla de forma definitiva en archivos.

**Tipos de colecciones:**

* **ArrayList:** Permite almacenar listas ordenadas y dinámicas, facilitando el acceso y la modificación de elementos.
* En el proyecto se implementó en las clases **MantenimientoCliente** y **MantenimientoEmpleado**, permitiendo recorrer la información con facilidad y mostrarla posteriormente en la interfaz gráfica mediante **JTable**.

El uso de las colecciones permitió optimizar el manejo temporal de los datos y mantener un flujo ordenado entre la interfaz gráfica y los archivos de almacenamiento.

* 1. **Archivos en Java**

Para mantener la información entre distintas ejecuciones del sistema, se emplearon archivos de texto como método de persistencia.

Se utilizaron clases pertenecientes a los paquetes **java.io** y **java.util**, tales como **File**, **FileWriter**, **BufferedWriter** y **Scanner**, que permiten la lectura y escritura de datos.

En el sistema, se crearon archivos como **clientes.txt** y **usuarios.txt**, donde se registran los datos ingresados por el usuario.

De acuerdo con Mayabi (2015), estas clases son útiles para manejar información sin necesidad de un sistema gestor de base de datos, manteniendo la sencillez y eficacia del almacenamiento.

* 1. **Librerías gráficas**

Para el diseño de la interfaz del sistema se utilizó la biblioteca **Swing**, la cual permite desarrollar interfaces gráficas en Java mediante componentes visuales reutilizables.  
 Según Ezequiel et al. (2022), Swing proporciona un conjunto de herramientas que facilitan la creación de ventanas, botones, cuadros de texto y tablas, garantizando una mejor interacción con el usuario.

**Componentes usados:**

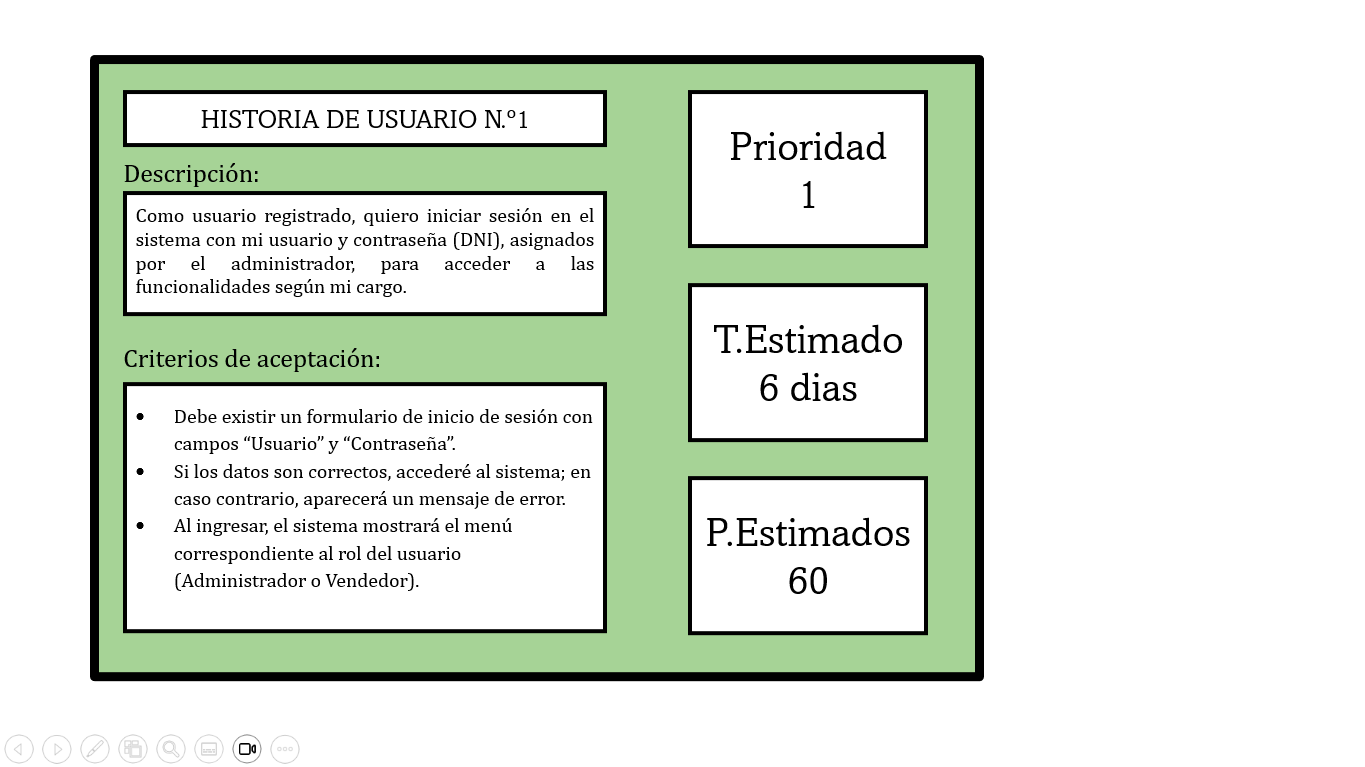
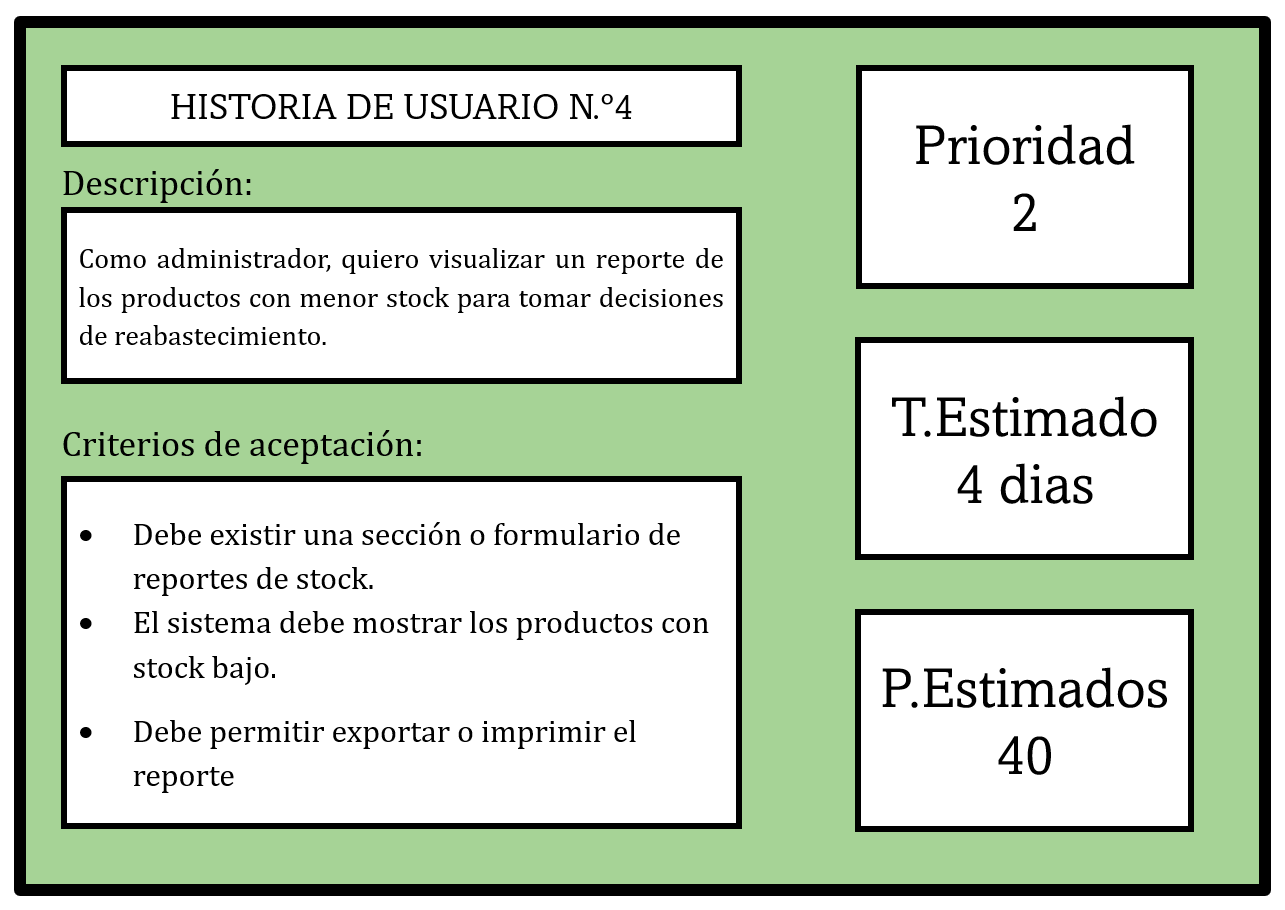
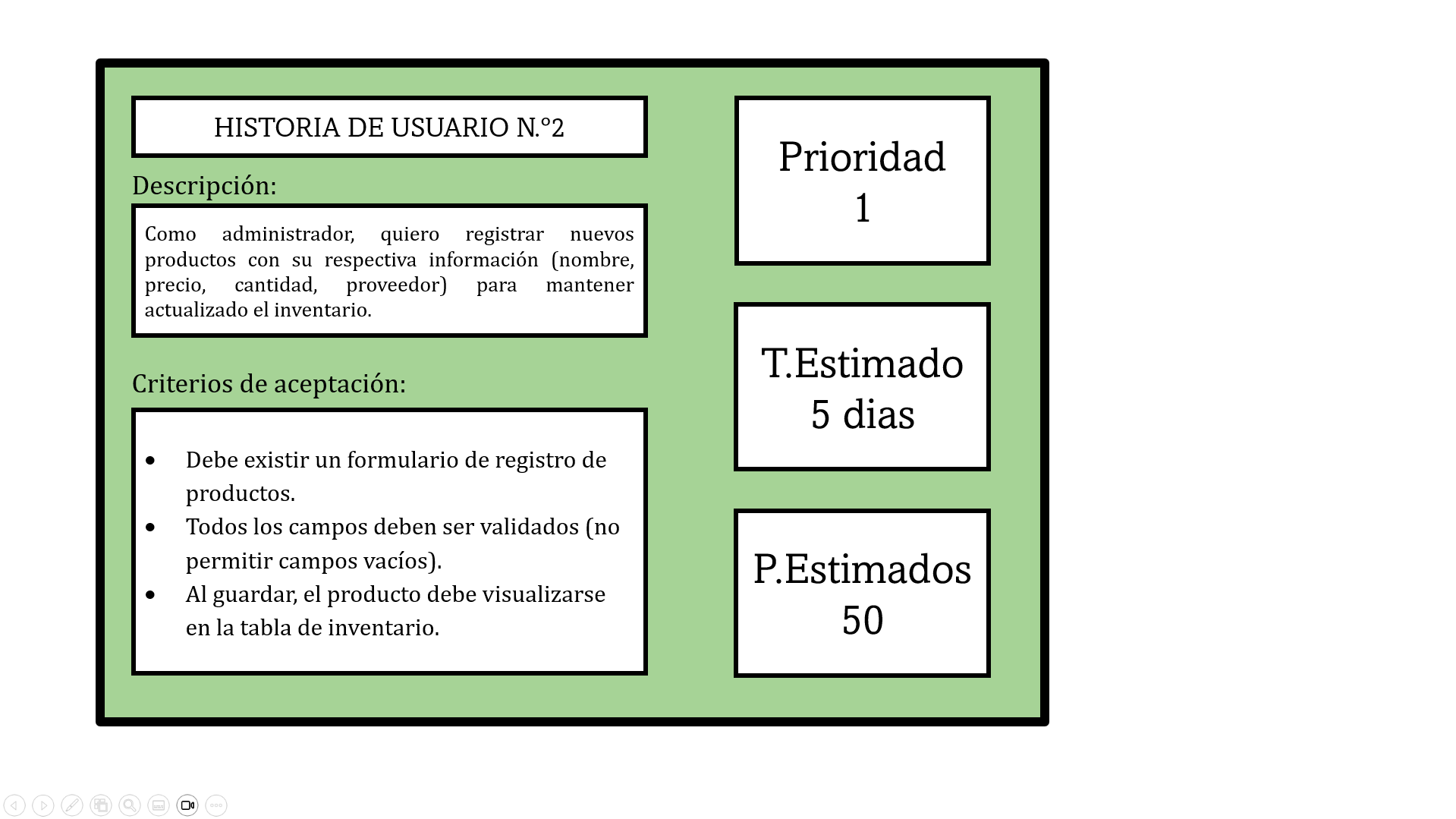
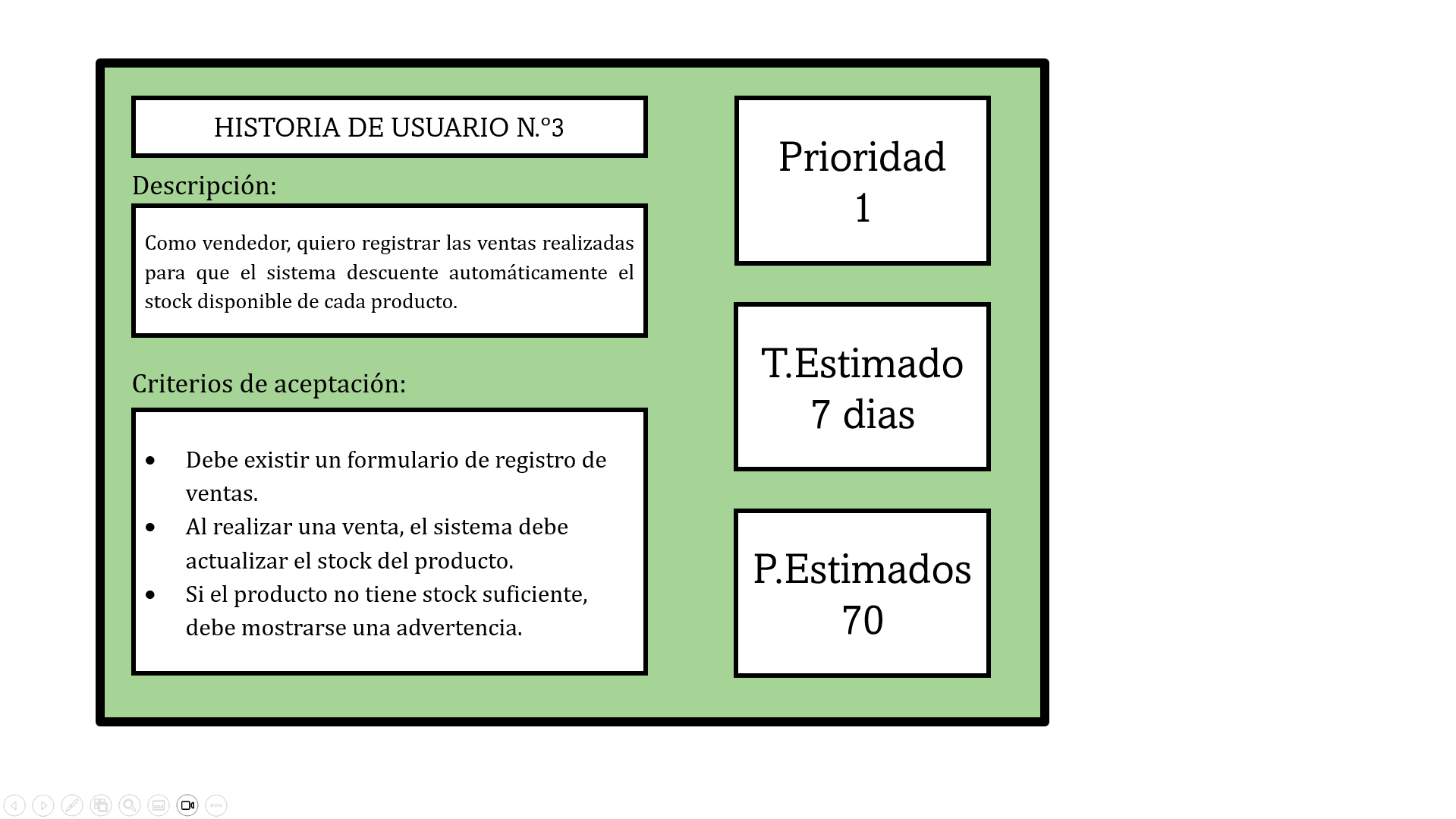
* **JLabel:** muestra etiquetas o descripciones de campos.
* **JTextField:** permite la entrada de texto por parte del usuario.
* **JButton:** ejecuta acciones específicas como *Guardar*, *Mostrar* o *Eliminar*.
* **JTable:** muestra datos en forma de tabla, como las listas de empleados o clientes.
* **JCalendar:** librería adicional que permitió incorporar un calendario visual para seleccionar fechas fácilmente.

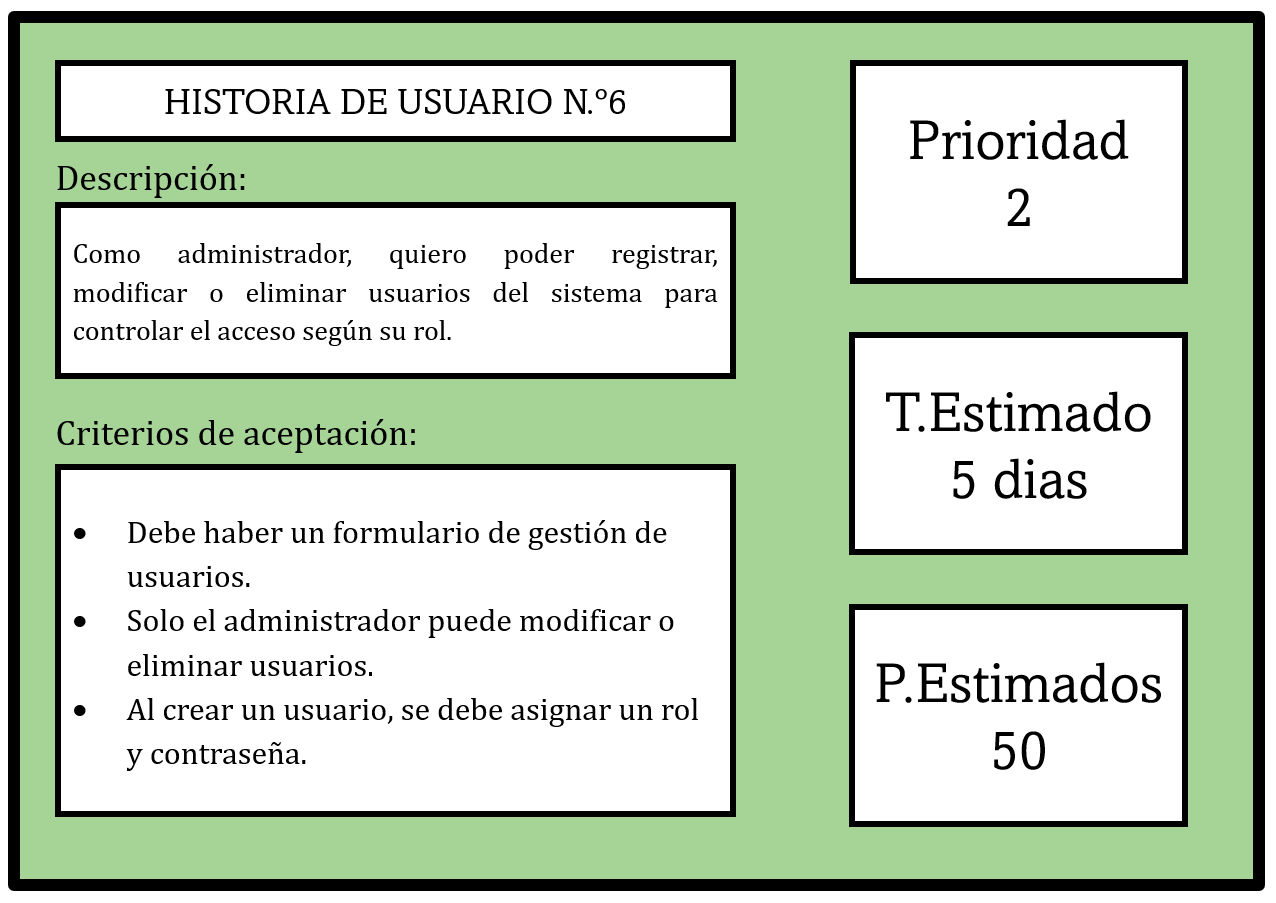
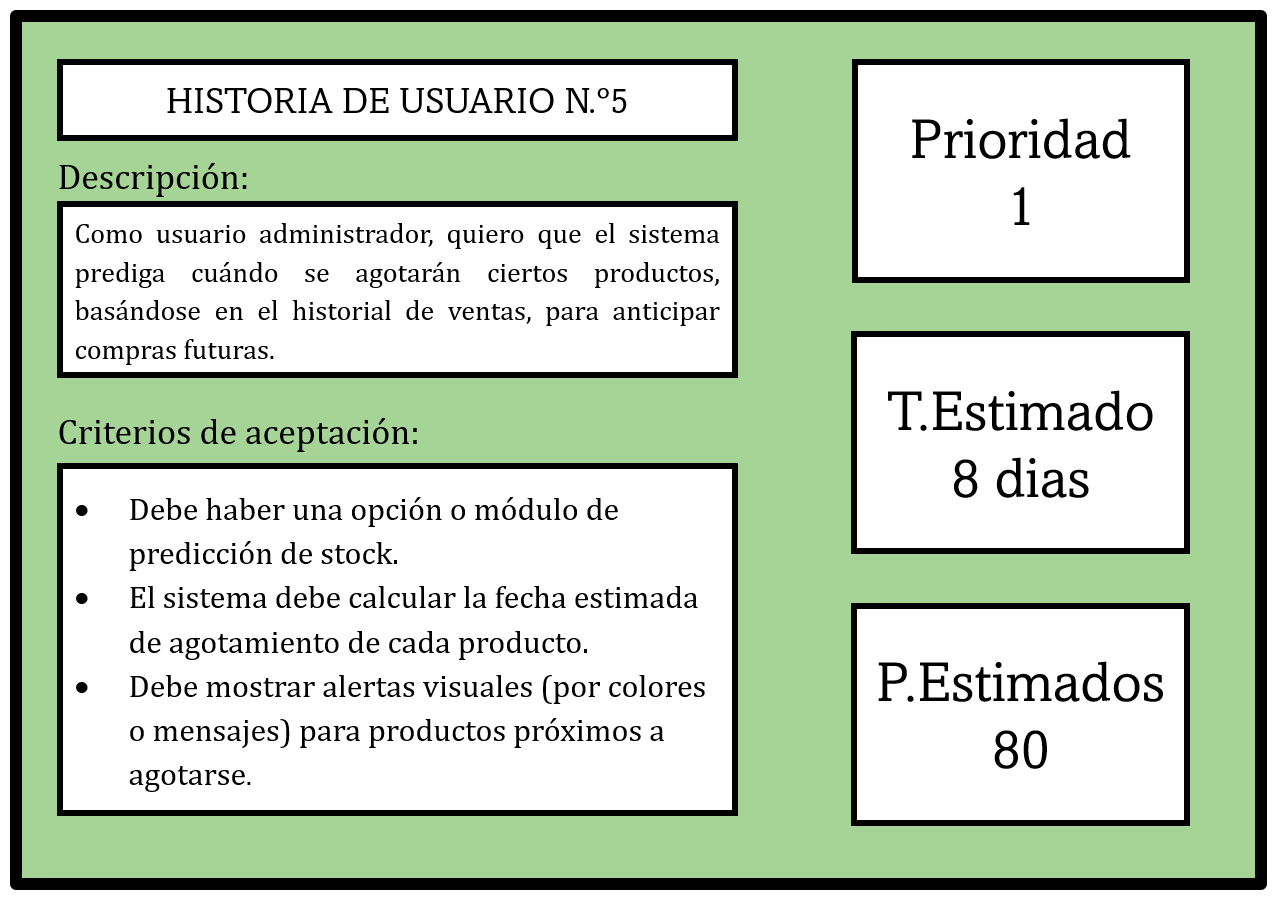
* 1. **Criterios de calidad de software (ISO 9126)**

De acuerdo con lo expuesto por **Geeks for Geeks (2024)**, la norma **ISO 9126** define un conjunto de criterios que permiten evaluar la calidad del software. En el desarrollo del sistema se consideraron los siguientes aspectos:

* **Funcionalidad:** El sistema cumple con las funciones establecidas, como el registro, edición y visualización de datos de empleados y clientes, garantizando que los procesos respondan correctamente a las necesidades del usuario.
* **Usabilidad:** La interfaz fue diseñada de manera clara, con botones y campos bien distribuidos, de modo que cualquier empleado pueda interactuar fácilmente con el sistema según su rol.  
  **Fiabilidad:** Se implementaron validaciones para evitar errores comunes, como campos vacíos o duplicados, asegurando así la integridad de los datos almacenados.
* **Mantenibilidad:** El código fuente está estructurado en clases independientes y bien organizadas, lo que facilita la comprensión del sistema y la incorporación de futuras mejoras.paint
* **Portabilidad:** Al estar desarrollado completamente en **Java**, el sistema puede ejecutarse en diferentes sistemas operativos (Windows, Linux, macOS) sin necesidad de modificaciones.

# Historias de Usuario y Funcionalidades del producto





## 9. Diagrama de clases y relaciones

El presente proyecto se ha desarrollado, hasta la fecha de hoy, de acuerdo al siguiente diagrama de clases y relaciones, enfocado exclusivamente en la **Gestión de Stock y Predicción de Agotamiento** para Empresas Proveedoras de Insumos.

### Clase Login

* Contiene dos atributos: usuario y contraseña.
* Representa las credenciales que los usuarios deben ingresar para acceder al sistema.
* Se relaciona directamente con la interfaz **FrmLogin**, desde donde se inicia sesión.

### Formulario FrmLogin

* Ventana principal de autenticación del sistema.
* Redirige al usuario autenticado a uno de los siguientes formularios según su rol:
  + FrmAdminGeneral (Administrador)
  + FrmPanelAlmacen (Jefe de Almacén/Logística)
  + FrmPanelVentas (Vendedor/Counter)

### Clase Persona

* Clase base o superclase de la que heredan otras clases como **Cliente** y **Empleado**.
* Contiene los siguientes atributos: nombre, apellidos, dni, correo, telefono, edad, sexo, codigo.

### Clase Empleado (Anteriormente Clase RegistroEmpleados)

* Extiende la **Clase Persona**.
* Añade atributos específicos del contexto laboral: oficina, **cargo** (Rol: Administrador, Almacén, Ventas).

### Clase MantenimientoEmpleado

* Administra una lista de empleados (listaEmpleados).
* **Métodos:** AgregarEmpleado(), obtenerEmpleado(), actualizarEmpleado(), guardar Empleados().
* **Formularios Relacionados:** FrmRegistroEmpleados, FrmListaEmpleados, FrmEditorEmpleados.

### Clase Cliente (Anteriormente Clase Registro Clientes)

* También hereda de **Clase Persona**.
* Agrega atributos adicionales: ruc (o DNI), direccion.

### Clase MantenimientoClientes

* Administra la información de clientes (listaClientes).
* **Métodos:** AgregarCliente(), obtenerCliente(), actualizarCliente(), guardarClientes().
* **Formularios Relacionados:** FrmRegistroClientes, FrmListaClientes, FrmEditorClientes.

### Clase Proveedor (Reemplazo de Rutas/Buses)

* **Atributos:** idProveedor, razonSocial, contacto, **leadTimeDias**.
* **Uso:** Registra a las entidades que suministran los insumos.

### Clase Insumo (Reemplazo de Buses/Vehículos)

* **Atributos:** codigoSKU, nombre, **stockActual**, costoUnitario, **puntoReorden**, **stockSeguridad**.
* **Uso:** Clase central que representa el producto físico gestionado en el inventario.

### Clase MantenimientoInsumos

* **Descripción:** Gestiona el CRUD y la actualización del inventario.
* **Métodos Clave:** agregarInsumo(), actualizarStock(SKU, cantidad, tipo), eliminarInsumo().

### Clase TransaccionStock (Reemplazo de Logística)

* **Atributos:** idTransaccion, fecha, **tipoMovimiento** (Entrada/Salida), cantidad, codigoSKU.
* **Uso:** Registra cada movimiento de inventario, sirviendo de fuente de datos para la predicción.

### Clase PrediccionStock (Reemplazo de EvaluadorRutas/AsignacionesRuta)

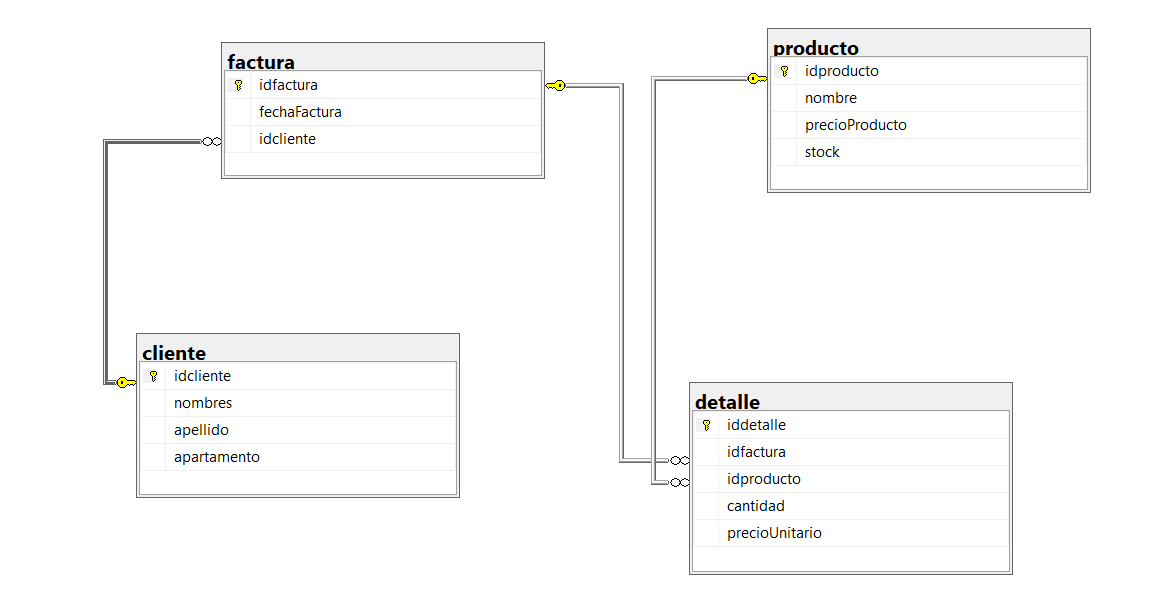
* **Descripción:** Implementa la lógica para anticipar el agotamiento de stock.
* **Métodos Clave:** calcularTasaConsumo(SKU, periodo), determinarDiasRestantes(), generarAlertaReorden().

### Clase ReporteFinanzas (Reemplazo de MantenimientoFinanzas)

* **Descripción:** Gestiona la información de costos y ventas.
* **Métodos Clave:** calcularInventarioValorizado(), calcularCMV(periodo).

### Clase CapacitacionRRHH (Mantiene la función de RRHH)

* **Descripción:** Módulo auxiliar para la gestión de historial y desarrollo del personal.
* **Métodos Clave:** leerHistorialCapacitaciones(), guardarCapacitacion().



# 10. Diagrama de Clases y Relaciones

# El diseño de clases se ha estructurado para modelar la gestión de inventario, la persistencia de datos y la lógica de predicción de agotamiento.

| Clase | Descripción | Atributos Clave | Métodos Clave |
| --- | --- | --- | --- |
| Clase Login | Contiene las credenciales para la autenticación de usuarios. | usuario, contraseña. | validarCredenciales(), obtenerRol() |
| Formulario FrmLogin | Ventana principal de autenticación. Redirige al panel principal según el rol. | N/A | iniciarSesion(), abrirDashboard() |
| Clase Persona | Superclase base de la que heredan empleados y clientes. | nombre, apellidos, dni, correo, telefono, codigo. | N/A |
| Clase Empleado | Extiende Persona. Datos específicos del personal de la empresa. | oficina, cargo (Admin, Almacén, Ventas). | N/A |
| Clase MantenimientoEmpleado | Administra la lista y persistencia de empleados (uso de ArrayList). | listaEmpleados, rutaArchivo. | agregarEmpleado(), actualizarEmpleado(), guardarEmpleados() |
| Clase Cliente | Extiende Persona. Información de las empresas o personas que compran insumos. | ruc o dni, direccion, tipoCliente. | N/A |
| Clase MantenimientoClientes | Administra la lista y persistencia de clientes. | listaClientes, rutaArchivo. | agregarCliente(), obtenerCliente(), guardarClientes() |
| Clase Insumo | Clase central del inventario. Representa cada producto gestionado. | codigoSKU, nombre, categoria, stockActual, unidadMedida, precioVenta, costoAdquisicion, puntoReorden, stockSeguridad. | N/A |
| Clase MantenimientoInsumos | Gestiona el CRUD y la actualización del stock. | listaInsumos, rutaArchivo. | agregarInsumo(), actualizarStock(codigo, cantidad, tipo), eliminarInsumo() |
| Clase Proveedor | Registra y gestiona los datos de los proveedores de insumos. | ruc, razonSocial, contacto, telefonoContacto, leadTimePromedio. | N/A |
| Clase TransaccionStock | Registra movimientos de inventario (Entrada por Compra o Salida por Venta/Despacho). | idTransaccion, fecha, tipo (Entrada/Salida), cantidadMovida, codigoSKU, referencia (ID Venta/Compra). | N/A |
| Clase PrediccionStock | Lógica de predicción. Calcula el tiempo hasta el agotamiento. | N/A | calcularTasaConsumo(SKU, periodo), determinarStockOut(SKU), generarAlertaReorden(SKU) |

# He realizado la modificación de todas las secciones del informe (del punto 10 al 16) para que estén completamente alineadas con el tema: "Aplicación de Escritorio para la Gestión de Stock y Predicción de Agotamiento en Empresas Proveedoras de Insumos."

# Se han eliminado todos los conceptos de transporte (Buses, Rutas, Conductores) y se ha reforzado el foco en el Inventario, la Predicción, los Insumos y los Proveedores.

# 

# 

# El diseño de clases se ha estructurado para modelar la gestión de inventario, la persistencia de datos y la lógica de predicción de agotamiento.

| Clase | Descripción | Atributos Clave | Métodos Clave |
| --- | --- | --- | --- |
| Clase Login | Contiene las credenciales para la autenticación de usuarios. | usuario, contraseña. | validarCredenciales(), obtenerRol() |
| Formulario FrmLogin | Ventana principal de autenticación. Redirige al panel principal según el rol. | N/A | iniciarSesion(), abrirDashboard() |
| Clase Persona | Superclase base de la que heredan empleados y clientes. | nombre, apellidos, dni, correo, telefono, codigo. | N/A |
| Clase Empleado | Extiende Persona. Datos específicos del personal de la empresa. | oficina, cargo (Admin, Almacén, Ventas). | N/A |
| Clase MantenimientoEmpleado | Administra la lista y persistencia de empleados (uso de ArrayList). | listaEmpleados, rutaArchivo. | agregarEmpleado(), actualizarEmpleado(), guardarEmpleados() |
| Clase Cliente | Extiende Persona. Información de las empresas o personas que compran insumos. | ruc o dni, direccion, tipoCliente. | N/A |
| Clase MantenimientoClientes | Administra la lista y persistencia de clientes. | listaClientes, rutaArchivo. | agregarCliente(), obtenerCliente(), guardarClientes() |
| Clase Insumo | Clase central del inventario. Representa cada producto gestionado. | codigoSKU, nombre, categoria, stockActual, unidadMedida, precioVenta, costoAdquisicion, puntoReorden, stockSeguridad. | N/A |
| Clase MantenimientoInsumos | Gestiona el CRUD y la actualización del stock. | listaInsumos, rutaArchivo. | agregarInsumo(), actualizarStock(codigo, cantidad, tipo), eliminarInsumo() |
| Clase Proveedor | Registra y gestiona los datos de los proveedores de insumos. | ruc, razonSocial, contacto, telefonoContacto, leadTimePromedio. | N/A |
| Clase TransaccionStock | Registra movimientos de inventario (Entrada por Compra o Salida por Venta/Despacho). | idTransaccion, fecha, tipo (Entrada/Salida), cantidadMovida, codigoSKU, referencia (ID Venta/Compra). | N/A |
| Clase PrediccionStock | Lógica de predicción. Calcula el tiempo hasta el agotamiento. | N/A | calcularTasaConsumo(SKU, periodo), determinarStockOut(SKU), generarAlertaReorden(SKU) |

# 

# Formularios Relacionados:

# Gestión de Empleados: FrmRegistroEmpleados, FrmListaEmpleados.

# Gestión de Clientes: FrmRegistroClientes, FrmListaClientes.

# Gestión de Inventario: FrmRegistroInsumo, FrmMovimientoStock, FrmListaInsumos, FrmRegistroProveedor.

# Predicción: FrmPanelAlertas, FrmReportePrediccion.

# Ventas/Finanzas: FrmRegistroVenta, FrmReporteVentas.

# · 11. Clases y características de usuarios principales

# · Los roles de usuario están definidos para garantizar el Control de Acceso Basado en Roles (RBAC) en el sistema de gestión de stock.

| Rol | Características Clave | Funcionalidades Principales |
| --- | --- | --- |
| Administrador General | Acceso total al sistema. Responsable de la configuración, seguridad y gestión de usuarios. | Gestión de Usuarios (CRUD de empleados y roles). Configuración de Parámetros (Punto de Reorden por defecto, Costos). Acceso a todos los Reportes Financieros y de Predicción. |
| Jefe de Almacén / Logística | Acceso a módulos de inventario. Encargado de validar y registrar las entradas y salidas físicas de insumos. | CRUD de Insumos y Proveedores. Registro de Entradas por Compra. Registro de Salidas/Consumo Interno. Visualización y Gestión de Alertas de Stock. |
| Vendedor / Ventas | Acceso limitado al registro de ventas y consulta de stock disponible para la venta. | Registro de Clientes (CRUD básico). Registro de Ventas/Facturación. Consulta de Stock en tiempo real (solo lectura). |

# 

# 

# 

# 

# 12. Requerimientos funcionales

# Los requerimientos definen las funciones necesarias para gestionar el inventario, las transacciones y la predicción de agotamiento.

### RF1. Autenticación y Redirección por Rol

El sistema debe permitir el ingreso de un Usuario y Contraseña y, tras la validación exitosa, debe redirigir a la GUI de ADMINISTRADORES si las credenciales son de administrador (admin/admin) o a la GUI de COMPRA PRODUCTOS si son de usuario estándar (usuario1/usuario1).

### RF2 Gestión CRUD de Productos por Administrador

El sistema debe proveer una interfaz (ADMINISTRADORES GUI) que permita al usuario con rol de administrador Agregar, Modificar, Buscar y Eliminar Producto del stock.

### RF3. Visualización de Inventario en GUI de Administrador

La GUI del administrador debe ser capaz de Listar y mostrar el INVENTARIO DE PRODUCTOS, incluyendo detalles como ID, Nombre (e.g., Gaseosa Cola 500 ml), Precio y Stock de cada producto.

### RF4 .Carga y Persistencia de Inventario

El sistema debe permitir al administrador Cargar Inventario (o guardarlo) para asegurar la conexión con la base de datos SQL y la persistencia de los registros de productos, precios y stock.

### RF5. Selección y Adición de Productos a la Compra

El sistema debe mostrar una interfaz de Compra Productos donde el usuario pueda seleccionar un producto (mediante un *combobox*) e ingresar la Cantidad deseada para Agregarlo al detalle de la compra.

### RF6. Manipulación de Ítems en la Compra

El sistema debe permitir al usuario ver el listado de productos agregados a la compra y debe facilitar la opción de Eliminar Producto del listado antes de finalizar la transacción.

### RF7. Registro de Venta y Generación de Boleta

Al pulsar el botón "Compra", el sistema debe registrar la venta, actualizar automáticamente el stock de los productos vendidos y generar un RESUMEN DE COMPRA (Boleta) con el Total Final de la transacción.

### RF8. Cálculo de Subtotales en la Compra

El sistema debe calcular y mostrar dinámicamente el Subtotal de cada ítem agregado al listado de compra (e.g., "Cantidad: 11 - Precio: $4.20 - Subtotal: $46.20").

### RF9. Registro Histórico de Movimiento y Predicción

El sistema debe mantener un Registro histórico de movimiento (entradas y ventas) en la base de datos SQL que sirva como fuente de datos para alimentar el módulo de predicción de agotamiento de stock.

**RF10 Generación de Reportes de Inventario y Agotamiento**

El sistema debe tener la capacidad de generar reportes de inventario y productos próximos a agotarse, basándose en los métodos de consulta de movimiento en la base de datos SQL.

### RF11. Gestión de Stock Mínimo y Alertas de Reorden

El sistema debe permitir al administrador definir un Stock Mínimo (o umbral de reorden) para cada producto. Cuando el stock actual de un producto caiga por debajo o iguale este nivel mínimo, el sistema debe generar una alerta visual en la GUI de Administrador y en los reportes de agotamiento, notificando la necesidad de reponer el producto.

### RF12. Validación de Stock Disponible en la Compra

Durante la selección y adición de productos a la compra (RF5), el sistema debe validar que la Cantidad solicitada por el usuario no exceda el Stock actualmente disponible del producto en el inventario. Si la cantidad solicitada es mayor que el stock, el sistema debe mostrar un mensaje de error y no permitir agregar el ítem con esa cantidad.

### RF13. Cierre de Sesión y Seguridad

El sistema debe proveer un mecanismo claro y accesible (botón "Cerrar Sesión") en ambas interfaces (ADMINISTRADORES GUI y COMPRA PRODUCTOS GUI). Al pulsar este botón, el sistema debe finalizar la sesión actual y redirigir al usuario de vuelta a la pantalla de Autenticación (login), asegurando que se requiere una nueva validación para acceder.

### RF14. Filtros y Ordenamiento en la Visualización de Inventario

La GUI del administrador (RF3) y los reportes (RF10) deben proveer opciones para Filtrar y Ordenar la lista de productos. Los filtros deben incluir opciones por Nombre del Producto (búsqueda parcial), Stock (e.g., mostrar solo productos con stock bajo o cero) y/o ID. El ordenamiento debe ser posible por Nombre y Stock (ascendente/descendente).

### RF15. Gestión de Múltiples Unidades de Medida

El sistema debe permitir asociar una Unidad de Medida (e.g., unidades, litros, kilogramos) a cada producto durante su gestión CRUD (RF2). Esta unidad debe ser visualizada en la lista de inventario (RF3) y considerada al mostrar la Cantidad en el detalle de la compra y la boleta (RF8 y RF7), para una gestión de inventario más precisa.

# 12.1. Autenticación y control de acceso por roles

# El sistema debe requerir usuario y contraseña para acceder y aplicar roles de acceso (Administrador, Almacén, Ventas).

# 12.2. Gestión de usuarios y empleados

# El administrador debe poder registrar, modificar y eliminar la información de los empleados.

# Debe permitir la asignación y modificación de roles para cada empleado.

# 12.3. Gestión de clientes

# El sistema debe permitir el registro de clientes (razón social, RUC/DNI, dirección) para asociarlos a las transacciones de venta.

# 12.4. Gestión de Insumos y Proveedores

# El sistema debe permitir el registro (CRUD) de insumos, incluyendo codigoSKU, costoAdquisicion, y unidades de medida.

# Debe permitir el registro de Proveedores y asociarlos a los insumos que suministran.

# El sistema debe registrar Entradas de Stock (compras) y Salidas de Stock (ventas o consumo), actualizando el inventario de forma automática y en tiempo real.

# 12.5. Módulo de Predicción y Alertas (Core del Proyecto)

# El sistema debe calcular la tasa de consumo promedio diario de cada insumo basándose en el historial de TransaccionStock (salidas).

# Debe permitir al Jefe de Almacén definir o modificar el Punto de Reorden y el Stock de Seguridad para insumos críticos.

# Debe generar Alertas de Agotamiento cuando el stockActual caiga por debajo del Punto de Reorden.

# Debe predecir la fecha estimada de agotamiento (*stock-out*) usando la fórmula: $\text{Fecha Agotamiento} = \text{Fecha Actual} + \left( \frac{\text{Stock Actual}}{\text{Tasa de Consumo Diario}} \right)$.

# 12.6. Gestión de Ventas y Finanzas

# El sistema debe permitir el registro de transacciones de venta, con detalle de los insumos vendidos y la cantidad.

# Debe calcular automáticamente el Costo de Mercancía Vendida (CMV) y el Valor del Inventario (Inventario Valorizado) para reportes financieros.

# 12.7. Trazabilidad de Inventario y Reportes

# Debe generar un historial de movimientos (trazabilidad) para cada insumo, mostrando fechas y cantidades de entradas/salidas.

# Debe generar reportes personalizables de niveles de stock, ventas por período y estado de las alertas de predicción.

# 13. Reglas de negocio

| Regla de Negocio | Descripción |
| --- | --- |
| 13.1. Control de acceso basado en roles | La modificación de los parámetros de predicción (Punto de Reorden, Stock de Seguridad) solo puede ser realizada por el Administrador o el Jefe de Almacén. |
| 13.2. Validación de datos en todos los formularios | Los campos de cantidad en las transacciones de stock solo deben aceptar valores numéricos positivos. |
| 13.3. Notificación por cada operación importante | Se debe emitir una notificación de confirmación después de una transacción exitosa y una notificación de error en caso de intento de stock negativo. |
| 13.4. Formato de datos estandarizado | Todos los valores monetarios deben tener un formato de moneda local y las fechas deben usar el formato dd/MM/yyyy. |
| 13.5. Integridad y confidencialidad de la información | Las contraseñas se almacenarán mediante *hashing*. La eliminación de un insumo solo es posible si su stockActual es cero. |
| 13.6. Restricción de Stock Negativo | El sistema debe impedir la finalización de una venta o salida de stock si el resultado deja el stockActual del insumo en un valor negativo. |
| 13.7. Precisión en la Predicción | La tasa de consumo utilizada para la predicción debe promediarse con un mínimo de 30 días de transacciones. |

# 14. Requerimientos de interfaces externas

# 14.1. Características generales de la interfaz

# Conexión a Base de Datos: El sistema se conectará a MySQL Workbench (o similar SQL libre) para la persistencia centralizada de todos los datos (insumos, transacciones, usuarios, proveedores).

# Exportación de Reportes: Debe tener la funcionalidad para exportar los reportes de inventario y predicción a formatos externos (.csv o .xlsx).

# 14.2. Estándares de Interfaz Gráfica (GUI)

# Se utilizará la librería Java Swing con un diseño que priorice la visualización rápida de alertas de stock.

# Se implementarán JTables para la gestión de insumos con un sistema de colores para identificar rápidamente niveles de stock críticos (ej. rojo para alerta de agotamiento).

# El uso del JCalendar facilitará la selección de rangos de fechas en la generación de reportes de consumo.

# 

# 15. Requerimientos no funcionales

| Categoría | Requerimiento no funcional |
| --- | --- |
| Rendimiento | La predicción del agotamiento (*stock-out*) para la totalidad de los insumos debe ejecutarse en menos de 3 segundos. |
| Seguridad | El sistema debe realizar copias de seguridad de la base de datos de forma periódica (regla de negocio a nivel de BD). |
| Usabilidad | El proceso de registro de una transacción de entrada o salida no debe requerir más de 5 clics o pasos. |
| Fiabilidad | El cálculo del Punto de Reorden y la tasa de consumo debe ser preciso y auditable por el Administrador. |
| Mantenibilidad | La lógica de predicción (Clase PrediccionStock) debe estar completamente desacoplada de la interfaz gráfica y de las clases de mantenimiento. |
| Portabilidad | El sistema debe ser compatible con las versiones de Java 11 o superiores y ser ejecutable en Windows 10/11. |

# 16. Glosario

# Se actualizan los términos para reflejar el enfoque en la Gestión de Inventario y Cadena de Suministro.

| Término | Definición |
| --- | --- |
| GUI (Graphical User Interface) | Interfaz Gráfica de Usuario. Parte visual del sistema con la que interactúa el usuario. |
| SKU (Stock-Keeping Unit) | Unidad de Mantenimiento de Existencias. Código único utilizado para identificar un insumo específico en el inventario. |
| Punto de Reorden | Nivel de existencias en el que se debe realizar una nueva orden de compra para evitar el agotamiento. |
| Stock de Seguridad | Cantidad mínima de inventario que se mantiene para evitar el *stock-out* ante variaciones inesperadas de la demanda. |
| CRUD | Acrónimo de Create, Read, Update, Delete. Operaciones básicas de gestión de datos. |
| Stock-Out | Situación de inventario donde las existencias de un insumo se han agotado, impidiendo la venta o el consumo. |
| Lead Time | Tiempo que transcurre desde que se realiza la orden de compra al proveedor hasta que el insumo llega y está disponible en el almacén. |
| CMV (Costo de Mercancía Vendida) | Costo de los insumos que han sido vendidos o despachados en un período. Fundamental para el análisis de márgenes. |
| Scrum | Marco de trabajo ágil para el desarrollo de software, basado en iteraciones cortas (*sprints*). |

**17. Bibliografía**

Wahedi, H.J. (2023). *Forecasting and Inventory Planning: An Empirical Study*. *Applied Sciences, 13*(15), 8581.<https://www.mdpi.com/2076-3417/13/15/8581>

Madamidola, O.A., Daramola, O.A., & Akintola, K.G. (2024). *A Review of Existing Inventory Management Systems*. *International Journal of Research in Engineering and Science, 12*(9), 40-50. <https://www.ijres.org/papers/Volume-12/Issue-9/12094050.pdf?utm_source=chatgpt.com>

Nguyen, T. N., Khan, M., & Hossain, M. Z. (2024). *Product Demand Forecasting for Inventory Management with Freight Transportation Services Index Using Advanced Neural Networks Algorithm*. *American Journal of Computing and Engineering* [*https://ajpojournals.org/journals/index.php/AJCE/article/view/2432*](https://ajpojournals.org/journals/index.php/AJCE/article/view/2432)

*Cortés, P. (2022). Algoritmo predictivo de agotamiento de stock usando ventas mensuales. Universidad de Antioquia.*

*Recuperado de:* [*https://repository.udea.edu.co/*](https://repository.udea.edu.co/)

*García, L., & Torres, M. (2022). Automatización de procesos de inventario mediante sistemas de información en PYMES latinoamericanas. Revista Iberoamericana de Tecnología Empresarial, 8(2), 45–56.*

*Recuperado de:* [*https://www.redalyc.org/journal/6959/695972441003/*](https://www.redalyc.org/journal/6959/695972441003/)

*Mendoza, L., & Pérez, R. (2024). Integración de bases de datos MySQL en sistemas Java para control de proveedores. Revista Peruana de Ingeniería Informática, 12(1), 15–28.*

*Recuperado de:* [*https://revistas.upc.edu.pe/index.php/ingenieriainformatica*](https://revistas.upc.edu.pe/index.php/ingenieriainformatica)

*MySQL. (2024). MySQL Workbench Manual (versión en español). Oracle Corporation.*

*Recuperado de:* [*https://dev.mysql.com/doc/workbench/en/*](https://dev.mysql.com/doc/workbench/en/)

*Oracle. (2023). Guía oficial del conector JDBC para Java (MySQL Connector/J). Oracle Documentation.*

*Recuperado de:* [*https://dev.mysql.com/doc/connector-j/8.0/en/*](https://dev.mysql.com/doc/connector-j/8.0/en/)

*Organización Internacional del Trabajo (OIT). (2023). Tendencias tecnológicas en la gestión empresarial y automatización de procesos.*

*Recuperado de:* [*https://www.ilo.org/global/lang--es/index.htm*](https://www.ilo.org/global/lang--es/index.htm)

*Ramírez, J., & Torres, F. (2023). Modelos predictivos para inventarios minoristas. Revista Colombiana de Computación, 19(2), 45–60.*

*Recuperado de:* [*https://revistas.unicauca.edu.co/index.php/visionario/article/view/2532*](https://revistas.unicauca.edu.co/index.php/visionario/article/view/2532)

*Sánchez, M. (2022). Sistema de gestión de inventarios en Java y MySQL. Universidad Nacional Mayor de San Marcos.*

*Recuperado de:* [*https://cybertesis.unmsm.edu.pe/*](https://cybertesis.unmsm.edu.pe/)

*Vargas, J. (2023). Aplicación de la programación orientada a objetos en el desarrollo de sistemas de gestión de inventario con Java y MySQL. Universidad Nacional de Ingeniería (Tesis de pregrado).*

*Recuperado de:* [*https://repositorio.uni.edu.pe/*](https://repositorio.uni.edu.pe/)

*Velasco, D. (2021). Desarrollo de un sistema de control de inventario con alertas automáticas de stock. Universidad de Guayaquil.*

*Recuperado de:* [*http://repositorio.ug.edu.ec/*](http://repositorio.ug.edu.ec/)

# 18. Anexos

| N° Sprint | Requerimientos Funcionales | Historia | T.E. | P.E. | P. |
| --- | --- | --- | --- | --- | --- |
| **Sprint N°1** | RF1. Autenticación y Redirección por Rol El sistema debe permitir el ingreso de un Usuario y Contraseña y, tras la validación exitosa, debe redirigir a la GUI de ADMINISTRADORES si las credenciales son de administrador (admin/admin) o a la GUI de COMPRA PRODUCTOS si son de usuario estándar (usuario1/usuario1). | H1 | 6 | 60 | 1 |
| RF2 Gestión CRUD de Productos por Administrador El sistema debe proveer una interfaz (ADMINISTRADORES GUI) que permita al usuario con rol de administrador Agregar, Modificar, Buscar y Eliminar Producto del stock. | H2 | 9 | 70 | 1 |
| **RF3. Visualización de Inventario en GUI de Administrador**  **La GUI del administrador debe ser capaz de Listar y mostrar el INVENTARIO DE PRODUCTOS, incluyendo detalles como ID, Nombre (e.g., Gaseosa Cola 500 ml), Precio y Stock de cada producto.** | H1 | 6 | 60 | 1 |
| **RF4 .Carga y Persistencia de Inventario**  **El sistema debe permitir al administrador Cargar Inventario (o guardarlo) para asegurar la conexión con la base de datos SQL y la persistencia de los registros de productos, precios y stock.** | H1 | 6 | 60 | 1 |
| RF5. Selección y Adición de Productos a la Compra  El sistema debe mostrar una interfaz de Compra Productos donde el usuario pueda seleccionar un producto (mediante un combobox) e ingresar la Cantidad deseada para Agregarlo al detalle de la compra. | H2 | 9 | 70 | 1 |
| RF6. Manipulación de Ítems en la Compra  El sistema debe permitir al usuario ver el listado de productos agregados a la compra y debe facilitar la opción de Eliminar Producto del listado antes de finalizar la transacción. | H2 | 9 | 70 | 1 |
| **RF7. Registro de Venta y Generación de Boleta**  **Al pulsar el botón "Compra", el sistema debe registrar la venta, actualizar automáticamente el stock de los productos vendidos y generar un RESUMEN DE COMPRA (Boleta) con el Total Final de la transacción.** | H1 | 6 | 60 | 1 |
| RF8. Cálculo de Subtotales en la Compra  El sistema debe calcular y mostrar dinámicamente el Subtotal de cada ítem agregado al listado de compra (e.g., "Cantidad: 11 - Precio: $4.20 - Subtotal: $46.20"). | H3 | 9 | 65 | 1 |
| **Sprint N° 2** | **RF9. Registro Histórico de Movimiento y Predicción**  **El sistema debe mantener un Registro histórico de movimiento (entradas y ventas) en la base de datos SQL que sirva como fuente de datos para alimentar el módulo de predicción de agotamiento de stock.** | H3 | 9 | 65 | 1 |
| **RF10 Generación de Reportes de Inventario y Agotamiento**  **El sistema debe tener la capacidad de generar reportes de inventario y productos próximos a agotarse, basándose en los métodos de consulta de movimiento en la base de datos SQL.** | H3 | 9 | 65 | 1 |
|  | RF11. Gestión de Stock Mínimo y Alertas de Reorden **El sistema debe permitir al administrador definir un Stock Mínimo (o umbral de reorden) para cada producto. Cuando el stock actual de un producto caiga por debajo o iguale este nivel mínimo, el sistema debe generar una alerta visual en la GUI de Administrador y en los reportes de agotamiento, notificando la necesidad de reponer el producto.** | H3 | 9 | 65 | 1 |
|  | RF12. Validación de Stock Disponible en la Compra **Durante la selección y adición de productos a la compra (RF5), el sistema debe validar que la Cantidad solicitada por el usuario no exceda el Stock actualmente disponible del producto en el inventario. Si la cantidad solicitada es mayor que el stock, el sistema debe mostrar un mensaje de error y no permitir agregar el ítem con esa cantidad.** | H3 | 9 | 65 | 1 |
|  | RF13. Cierre de Sesión y Seguridad **El sistema debe proveer un mecanismo claro y accesible (botón "Cerrar Sesión") en ambas interfaces (ADMINISTRADORES GUI y COMPRA PRODUCTOS GUI). Al pulsar este botón, el sistema debe finalizar la sesión actual y redirigir al usuario de vuelta a la pantalla de Autenticación (login), asegurando que se requiere una nueva validación para acceder.** | H3 | 9 | 65 | 1 |
|  | RF14. Filtros y Ordenamiento en la Visualización de Inventario **La GUI del administrador (RF3) y los reportes (RF10) deben proveer opciones para Filtrar y Ordenar la lista de productos. Los filtros deben incluir opciones por Nombre del Producto (búsqueda parcial), Stock (e.g., mostrar solo productos con stock bajo o cero) y/o ID. El ordenamiento debe ser posible por Nombre y Stock (ascendente/descendente).** | H3 | 9 | 65 | 1 |
|  | RF15. Gestión de Múltiples Unidades de Medida **El sistema debe permitir asociar una Unidad de Medida (e.g., unidades, litros, kilogramos) a cada producto durante su gestión CRUD (RF2). Esta unidad debe ser visualizada en la lista de inventario (RF3) y considerada al mostrar la Cantidad en el detalle de la compra y la boleta (RF8 y RF7), para una gestión de inventario más precisa.** | H3 | 9 | 65 | 1 |

PRACTICAS DE CAMPO GITHUB

<https://github.com/Marcial1202/Practica-de-campo>