Interfaz de usuario gráfica, Gráfico, Aplicación

El contenido generado por IA puede ser incorrecto.

**INGENIERÍA DE SOFTWARE**

**Total Drinks Pro**

**Nombre del Sistema:** Total Drinks Pro

**Apellido y Nombre:** Fernández Matías Nahuel

**Documento:** 37.152.184  
**Localización:** Castelar 095  
**Comisión:** A  
**Turno:** N  
**Año de Cursada**: 2025

**Apellido y Nombre del Docente:** Rodriguez Escobedo Julian Martin

Propuesta de Proyecto

Índice

[Historial de Revisión 3](#_Toc203156249)

[CARPETA DE PROYECTO 4](#_Toc203156250)

[G00. Descripción Global del Producto 4](#_Toc203156251)

[G01. Propósito 4](#_Toc203156252)

[G02. Descripción funcional del producto y Alcance 4](#_Toc203156253)

[G03. Planificación del proyecto 4](#_Toc203156254)

[G04. Definiciones, Acrónimos y Abreviaturas 4](#_Toc203156255)

[G05. Descripción de las personas participantes en el desarrollo del sistema de información y los usuarios (Roles) 5](#_Toc203156256)

[G06. Otros Requisitos 5](#_Toc203156257)

[G07. Diagrama de clases parcial de todos los módulos implementado 6](#_Toc203156258)

[G08. Modelo de datos parcial de todos los módulos implementados 7](#_Toc203156259)

[Información 8](#_Toc203156260)

[N00. Procesos de negocio 9](#_Toc203156261)

[N01. Especificación funcional por proceso de negocio 9](#_Toc203156262)

[N02. Especificaciones de Casos de Uso 10](#_Toc203156263)

[T00. Documentos de aspectos técnicos que provee el sistema de información. 17](#_Toc203156264)

[T01. Arquitectura Base 17](#_Toc203156265)

[T02. Gestión de Log In / Log Out del Sistema 21](#_Toc203156266)

[T03. Gestión de Encriptado 23](#_Toc203156267)

[T04. Gestión de Perfiles de Usuario 23](#_Toc203156268)

[T05. Gestión de Múltiples Idiomas 24](#_Toc203156269)

[T06. Gestión de Bitácora y Control de cambios 25](#_Toc203156270)

[Cambios realizados: 25](#_Toc203156271)

[4. Mejoras de interfaz: 25](#_Toc203156272)

# Historial de Revisión

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Autor | Descripción |
| 09/05/2025 | V1 | Matias N Fernandez | 1er entrega: Singleton, armado general de la solución |
| 13/06/2025 | V2 | Matias N Fernandez | 2da entrega:  Composite, Observer |
| 11/07/2025 | V3 | Matias N Fernandez | Transacciones, Arreglo de documento en Gral. |
| 17/10/2025 | V4 | Matias N Fernandez |  |

## G00. Descripción Global del Producto

Total Drinks Pro es un sistema integral de gestión para la administración de bebidas, diseñado para optimizar el control de inventarios, ventas, stock y reportes financieros en empresas dedicadas al comercio de bebidas.

## G01. Propósito

El propósito de Total Drinks Pro es resolver los desafíos de gestión de inventarios, ventas y control de stock que enfrentan las empresas de bebidas, minimizando pérdidas económicas y mejorando la eficiencia operativa.

## G02. Descripción funcional del producto y Alcance

Total Drinks Pro permite:

1. **Gestión de Usuarios:**
   * Alta, baja y modificación de usuarios (administradores y vendedores).
   * Control de permisos y autenticación segura mediante hash y salt en las contraseñas.
2. **Gestión de Productos (Bebidas):**
   * Alta, baja y modificación de productos.
   * Consulta de stock en tiempo real.
   * Actualización automática al registrar ventas.
3. **Gestión de Ventas:**
   * Registro de ventas con actualización de stock.
   * Control de ventas por cliente y producto.
   * Identificación de ventas no realizadas por falta de stock.
4. **Consultas de Stock:**
   * Visualización de productos y su disponibilidad.
   * Búsqueda de productos específicos por categoría o nombre.
5. **Reportes de Ventas:**
   * Generación de informes detallados por fecha, zona o producto.
   * Exportación de reportes para análisis externo.

## G03. Planificación del proyecto

## G04. Definiciones, Acrónimos y Abreviaturas

CLIENTE: Persona física o jurídica que adquiere bebidas.

PRODUCTO TERMINADO: Bebida lista para su venta.

ADMIN: Administrador del sistema.

VEND: Vendedor del sistema.

API: Interfaz de Programación de Aplicaciones.

UI: Interfaz de Usuario.

## G05. Descripción de las personas participantes en el desarrollo del sistema de información y los usuarios (Roles)

Roles:  
  
Vendedor: Registra ventas y consulta stock:

* **Venta y Descuento de Stock:**
  + Registro de ventas en tiempo real, reducción automática del stock y actualización en la base de datos.

Administrador: Gestiona usuarios, productos y reportes.

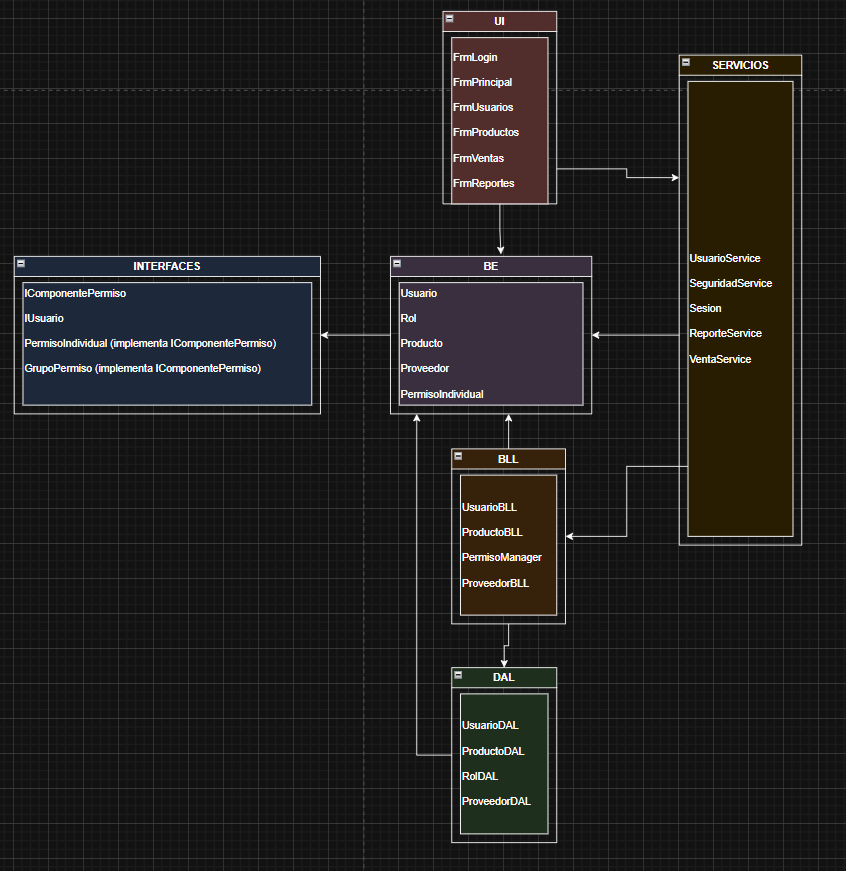
* **Compra de Inventario:**
  + Ingreso de nuevos productos al sistema, actualización de stock y categorización por tipo de bebida.
* **Gestión de Usuarios y Roles:**
  + Control de permisos para cada usuario, diferenciando entre Administrador y Vendedor.
* **Generación de Reportes:**
  + Creación de reportes mensuales, semanales o por pedido especial para evaluar desempeño de ventas.
* **Consulta de Stock y Control de Inventario:**
  + Visualización del inventario disponible para evitar sobreventa y manejar reabastecimiento.

Interesados:  
- Dueños de negocios: Requieren reportes financieros.  
- Operarios de almacén: Necesitan control de stock.

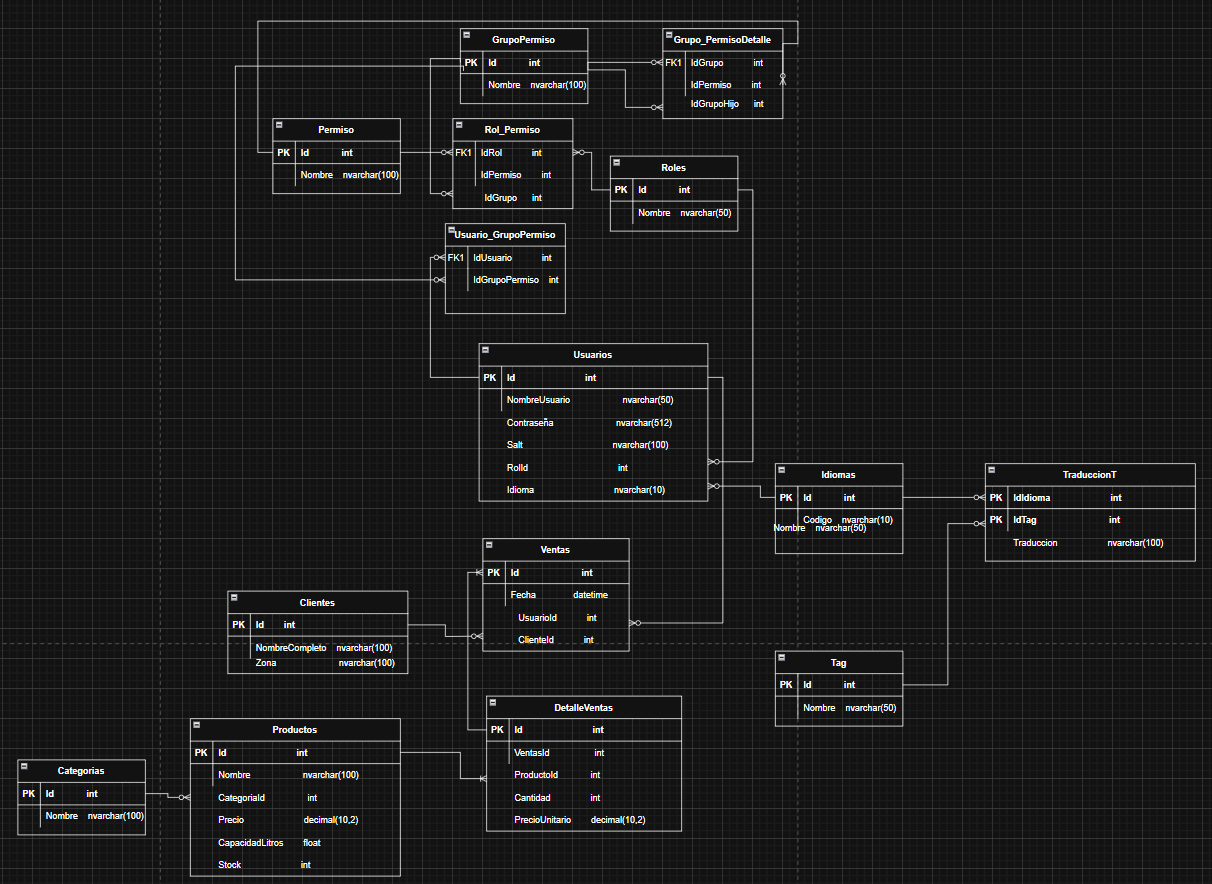
## G06. Otros Requisitos

* Requisitos de sistema: Windows 10+, .NET Framework 4.8, SQL Server 2019.
* Requisitos de desempeño: Respuesta en menos de 2s para consultas de stock.
* Requisitos de seguridad: Autenticación con hash y salt; conexiones cifradas.
* Requisitos de documentación: Manual de usuario y ayuda en línea.
* Requisitos de instalación: Instalador MSI que incluya base de datos.

## G07. Diagrama de clases parcial de todos los módulos implementado



## G08. Modelo de datos parcial de todos los módulos implementados



## Información

**Lenguaje de Programación:** C# en Visual Studio 2022.

**Arquitectura:** 5 capas (BE, BLL, DAL, Servicios, UI).

**Base de Datos:** SQL Server Management Studio.

**Seguridad:** Autenticación mediante hash y salt para contraseñas.

**Usuarios del Sistema:** Administradores y Vendedores.

**Interfaz Gráfica:** Windows Forms para el manejo de interfaz amigable y funcional.

## N00. Procesos de negocio **Descripción general**

El sistema de gestión de bebidas contempla los siguientes procesos de negocio principales:

| **Proceso** | **Descripción** |
| --- | --- |
| Gestión de Usuarios | Administración de usuarios, roles y permisos. |
| Gestión de Productos | Control de stock, fechas de vencimiento y alertas. |
| Registro de Ventas | Procesamiento de ventas con impacto en inventario. |
| Control de Stock | Consulta y validación de disponibilidad. |
| Reportes de Ventas | Generación de reportes por zona, fecha o producto. |
| Alertas de Vencimiento | Detección automática de productos próximos a vencer. |

### N01. Especificación funcional por proceso de negocio

**Diagrama general de casos de uso de negocio**

El siguiente diagrama representa las principales funcionalidades del sistema y los roles que intervienen en cada una.  
Muestra la interacción del Administrador y del Vendedor con los distintos casos de uso que integran el proceso de negocio general del sistema.

Diagrama

El contenido generado por IA puede ser incorrecto.

**Administrador:** tiene acceso a todas las gestiones del sistema (usuarios, productos, reportes y stock).

**Vendedor:** puede registrar ventas y consultar el stock disponible.

**Casos de uso de negocio incluidos:**

* **Gestionar Usuarios:** alta, baja y modificación de usuarios.
* **Gestionar Productos:** administración del inventario y control de vencimientos.
* **Registrar Ventas:** registro de ventas con control automático de stock.
* **Consultar Stock:** visualización del inventario disponible.
* **Generar Reportes de Ventas:** análisis por fecha, producto o zona.

**N01-A: Proceso de Venta con control de stock**

**1) Descripción funcional (negocio)**

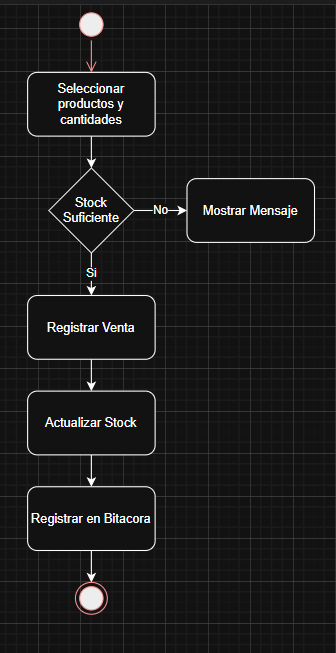
El proceso permite registrar una venta, afectando stock de productos, validando disponibilidad, registrando el detalle por ítem, calculando totales e inscribiendo la operación en Bitácora. Si falta stock, la venta no se confirma. La venta se asocia a un Cliente y a un Vendedor (usuario logueado).

**Objetivo:** asegurar una venta consistente, trazable y con impacto inmediato en inventario y auditoría.  
**Alcance:** Venta mostrador / simple (no incluye cuentas corrientes ni devoluciones).  
**Entradas:** selección de productos, cantidades, cliente, forma de pago (opcional).  
**Salidas:** venta registrada, stock actualizado, asiento en bitácora, posibilidad de emitir comprobante.  
**Reglas de negocio:**

* RB-V1: No se permite confirmar venta con alguna línea sin stock suficiente.
* RB-V2: El precio unitario proviene del producto vigente al momento de la venta.
* RB-V3: Toda venta genera un asiento en Bitácora (Usuario, Fecha, Entidad=Venta, Acciones=Alta).

Roles intervinientes: Vendedor, Administrador, Sistema  
Entradas: selección de productos, cantidades, cliente  
Comportamiento: validación de stock, registro de venta, actualización de inventario, registro en bitácora  
Salidas: confirmación de venta, comprobante impreso

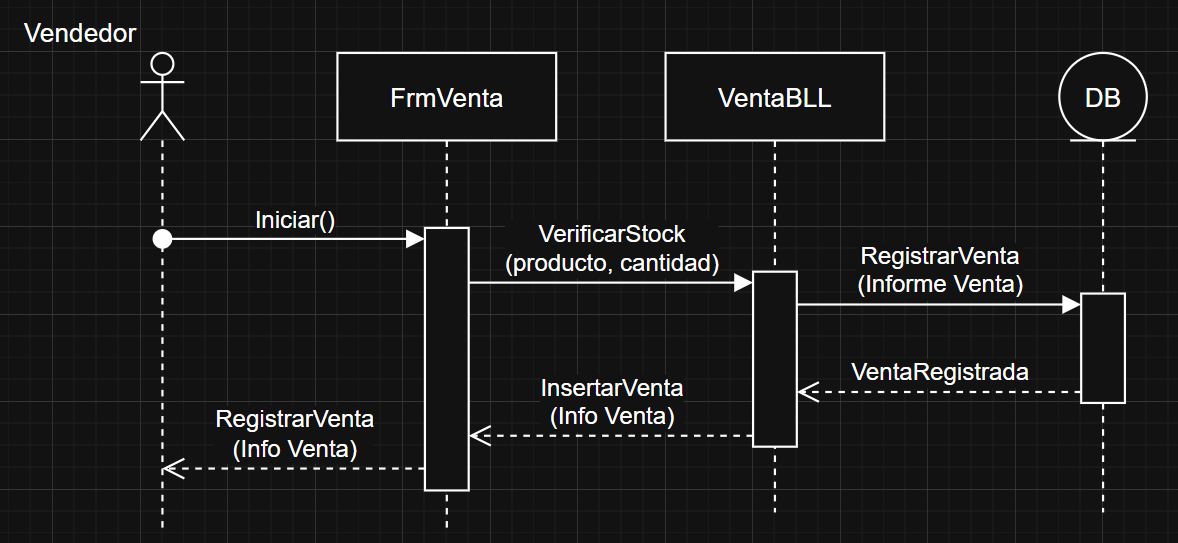
**Diagrama de Actividad**



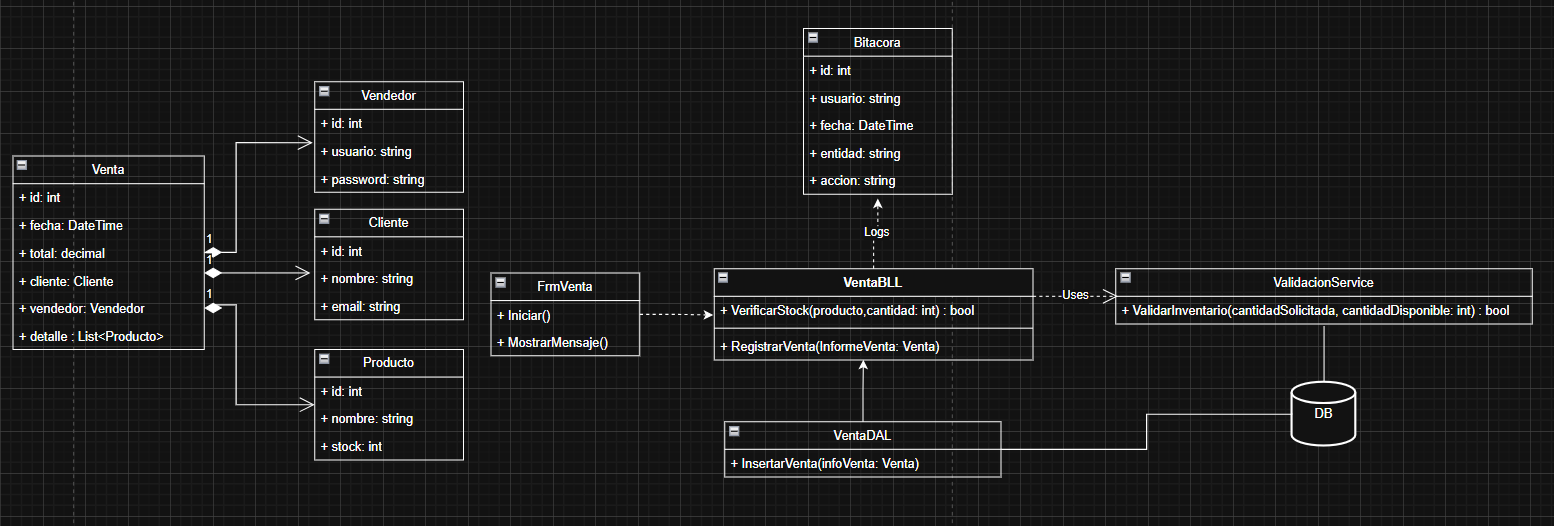
**Modelo conceptual (Clases involucradas):**

* Venta
* VentaDetalle
* Producto
* Usuario
* Bitacora

**Diagrama de Secuencia**



**Diagrama de Clases**



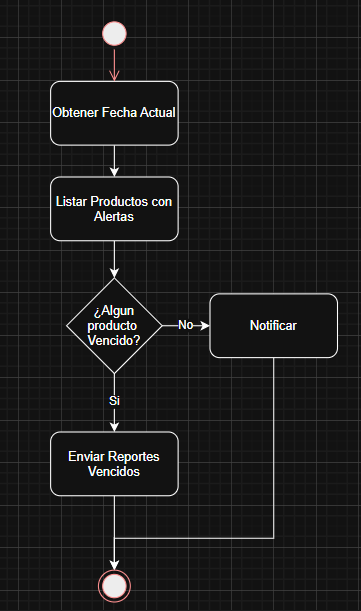
**N01-B: Proceso de Alertas de Vencimiento**

**Roles intervinientes:** Sistema, Administrador  
**Entradas:** fecha actual, umbral configurado (por ejemplo, 30 días antes del vencimiento)  
**Comportamiento:** consulta productos próximos a vencer, genera alertas automáticas, registra en bitácora  
**Salidas:** notificación en pantalla, reporte de productos próximos a caducar

**Reglas de negocio:**

* RB-A1: Se generan alertas automáticamente 30 días antes del vencimiento.
* RB-A2: Las alertas se marcan como “resueltas” si el producto se descarta o se vende.
* RB-A3: Toda alerta se registra en la bitácora del sistema.

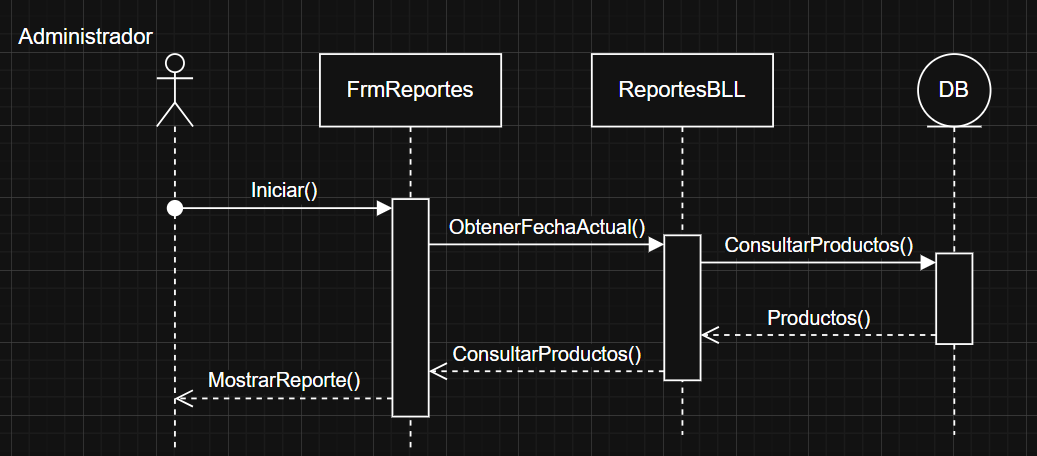
**Diagrama de Actividad**



**Modelo conceptual (Clases involucradas):**

* Producto
* Alerta
* Usuario
* Bitacora

**Diagrama de Secuencia**



### N02. Especificaciones de Casos de Uso

#### CU01: Iniciar sesión

|  |  |
| --- | --- |
| **CU:** | CU01 |
| **Nombre:** | Iniciar sesión |
| **Descripción:** | El sistema valida las credenciales del usuario y permite el acceso si son correctas. |
| **Actores:** | Usuario (Administrador o Vendedor) |
| **Precondiciones:** | El usuario debe estar registrado en el sistema con su rol asignado. |
| **Camino principal:** | 1. El usuario abre el sistema. 2. El sistema solicita nombre de usuario y contraseña. 3. El usuario introduce sus credenciales. 4. El sistema valida contra la base de datos. 5. El sistema permite el acceso al menú principal. |
| **Postcondiciones:** | El usuario accede al sistema con su rol correspondiente. |
| **Camino Alternativo:** | 3a. Si los datos son incorrectos, se muestra un mensaje de error y se solicita reintento. |

Diagrama

El contenido generado por IA puede ser incorrecto.

Diagrama

El contenido generado por IA puede ser incorrecto.

#### CU02: Gestionar usuarios (ABM)

|  |  |
| --- | --- |
| **CU:** | CU02 |
| **Nombre:** | Gestionar usuarios |
| **Descripción:** | Permite al administrador crear, modificar, eliminar o listar usuarios. |
| **Actores:** | Administrador |
| **Precondiciones:** | El administrador debe haber iniciado sesión. |
| **Camino principal:** | 1. El administrador accede a la sección de usuarios. 2. El administrador selecciona “Agregar” para registrar un nuevo usuario. 3. El administrador completa el formulario con los datos requeridos. 4. El administrador también puede editar o eliminar usuarios existentes. 5. El sistema valida y guarda el usuario. |
| **Postcondiciones:** | Se actualiza la lista de usuarios en la base de datos. |
| **Camino Alternativo:** | 3a. Si los datos están incompletos o son inválidos, se muestra un error. |

Gráfico

El contenido generado por IA puede ser incorrecto.

#### CU03: Gestionar productos (ABM)

|  |  |
| --- | --- |
| **CU:** | CU03 |
| **Nombre:** | Gestionar productos |
| **Descripción:** | Permite registrar, editar o eliminar productos del inventario. |
| **Actores:** | Administrador |
| **Precondiciones:** | El administrador debe haber iniciado sesión. |
| **Camino principal:** | 1. El administrador accede a la sección de productos. 2. El administrador puede agregar, editar o eliminar productos. 3. El sistema valida los campos ingresados. 4. El sistema actualiza la base de datos. |
| **Postcondiciones:** | Se actualiza el inventario. |
| **Camino Alternativo:** | 2a. Si otro rol accede, el sistema deshabilita o oculta las opciones de edición/eliminación. |

Imagen que contiene interior, tabla, hombre, parado

El contenido generado por IA puede ser incorrecto.

#### CU04: Registrar ventas

|  |  |
| --- | --- |
| **CU:** | CU04 |
| **Nombre:** | Registrar ventas |
| **Descripción:** | Permite ingresar una venta realizada, afectando el stock. |
| **Actores:** | Vendedor |
| **Precondiciones:** | El usuario debe haber iniciado sesión. |
| **Camino principal:** | 1. El vendedor selecciona los productos vendidos. 2. El vendedor ingresa cantidad, cliente y detalles. 3. El sistema descuenta del stock y registra la venta. |
| **Postcondiciones:** | La venta queda registrada en el historial. |
| **Camino Alternativo:** | 2a. Si no hay stock suficiente, se notifica al usuario. |

Calendario

El contenido generado por IA puede ser incorrecto.

#### CU05: Consultar stock

|  |  |
| --- | --- |
| **CU:** | CU05 |
| **Nombre:** | Consultar stock |
| **Descripción:** | Permite visualizar los productos disponibles y sus cantidades. |
| **Actores:** | Administrador, Vendedor |
| **Precondiciones:** | El usuario debe haber iniciado sesión. |
| **Camino principal:** | 1. El usuario abre la opción “Consultar stock”. 2. El sistema lista los productos y cantidades disponibles. |
| **Postcondiciones:** | Se muestra la información en pantalla. |
| **Camino Alternativo:** | - |

Gráfico

El contenido generado por IA puede ser incorrecto.

#### CU06: Generar reportes de ventas

|  |  |
| --- | --- |
| **CU:** | CU06 |
| **Nombre:** | Generar reportes de ventas |
| **Descripción:** | El administrador puede generar reportes de ventas por zona, fecha o producto. |
| **Actores:** | Administrador |
| **Precondiciones:** | Existencia de ventas registradas. |
| **Camino principal:** | 1. El administrador accede a la sección de reportes. 2. Elige el tipo de reporte a generar. 3. El sistema consulta los datos y genera el informe. |
| **Postcondiciones:** | Se muestra el reporte o se exporta. |
| **Camino Alternativo:** | 2a. Si no hay ventas, se notifica al usuario. |

Diagrama

El contenido generado por IA puede ser incorrecto.

## T00. Documentos de aspectos técnicos que provee el sistema de información.

### T01. Arquitectura Base

**Capa INTERFACES**

Clases e interfaces puramente abstractas, sin lógica de negocio ni acceso a datos.

* **IComponentePermiso**
  + **Atributos**: –
  + **Métodos**:
    - void Agregar(IComponentePermiso)
    - void Quitar(IComponentePermiso)
    - List<IComponentePermiso> ObtenerHijos()
    - bool Contiene(string)
    - bool EsGrupo { get; }
  + **Responsabilidades**
    - **Conocer**:
      * Estructura jerárquica de permisos.
    - **Hacer**:
      * Definir operaciones de composición y búsqueda.
* **IUsuario**
  + **Métodos**:
    - bool ValidarDatos()
  + **Responsabilidades**
    - **Conocer**:
      * Propiedades básicas del usuario.
* **GrupoPermiso** (implementa IComponentePermiso)
* **PermisoIndividual** (implementa IComponentePermiso)

**2. Capa BE (Entidades)**

Entidades simples con propiedades y constructor.

* **Rol**
  + **Atributos**: Id, Nombre, colección de permisos
  + **Métodos**: Agregar(), Quitar(), ObtenerHijos(), Contiene()
  + **Responsabilidades**
    - **Conocer**:
      * Valores de Id, Nombre, lista de hijos.
    - **Hacer**:
      * Componer árboles de permisos.
* **Usuario**
  + **Atributos**: Id, NombreUsuario, ContraseñaHasheada, Rol
  + **Métodos**: –
  + **Responsabilidades**
    - **Conocer**:
      * Datos de autenticación y el rol asignado.
* **Producto**, **Proveedor**, **PermisoIndividual**
  + Cada uno define sus atributos primarios y getters/setters.
  + **Responsabilidades**:
    - **Conocer**: sus datos encapsulados (p.ej. precio, stock para Producto).

**3. Capa DAL (Acceso a datos)**

Cada clase gestiona transacciones y mapea resultados de SQL a entidades BE.

* **UsuarioDAL**
  + **Atributos**: \_connectionString
  + **Métodos**:
    - ObtenerPorNombreUsuario(), ObtenerSaltPorUsuario(), Insertar(), ObtenerTodos(), Modificar(), Eliminar()
  + **Responsabilidades**
    - **Conocer**:
      * Consultas SQL y esquemas de tablas.
    - **Hacer**:
      * Ejecutar transacciones A/B/C (Begin/Commit/Rollback) para operaciones críticas.
* **ProductoDAL**, **RolDAL**
  + Mismos patrones:
    - **Conocer**: conexión y comandos.
    - **Hacer**: gestionar transacciones en Agregar/Modificar/Eliminar.

**4. Capa BLL (Lógica de negocio)**

Orquesta llamadas a DAL, aplica validaciones sencillas.

* **UsuarioBLL**
  + **Atributos**: instancia de UsuarioDAL
  + **Métodos**:
    - ObtenerPorNombre(), ObtenerSalt(), ObtenerTodos(), InsertarUsuario(), ModificarUsuario(), EliminarUsuario()
  + **Responsabilidades**
    - **Conocer**:
      * Reglas de negocio básicas (p.ej. que el nombre no esté vacío).
    - **Hacer**:
      * Transformar entidades de DAL antes de enviar a UI/Servicios.
* **ProductoBLL**, **PermisoManager**
  + Validaciones de dominio en la creación de productos y permisos.

**5. Capa SERVICIOS**

Encapsula Seguridad, Sesión y casos de uso más elevados.

* **SeguridadService**
  + **Métodos**: GenerarHashConSalt(), VerificarPassword()
  + **Responsabilidades**
    - **Conocer**:
      * Algoritmos de hash y salt.
    - **Hacer**:
      * Encriptar/validar credenciales.
* **UsuarioService**
  + **Atributos**: instancia de UsuarioBLL
  + **Métodos**: VerificarLogin(), ObtenerUsuario(), ObtenerTodos(), AgregarUsuario(), ModificarUsuario(), EliminarUsuario(), SeedAdmin()
  + **Responsabilidades**
    - **Hacer**:
      * Coordinar BLL y SeguridadService para autenticar y gestionar usuarios.
* **Sesion** (Singleton)
  + **Atributos**: Usuario actual
  + **Métodos**: IniciarSesion(), CerrarSesion()
  + **Responsabilidades**
    - **Conocer**:
      * Quién está logueado.
    - **Hacer**:
      * Controlar el acceso a funciones protegidas.

**6. Capa UI (Presentación)**

Forms WinForms que consumen Servicios y muestran datos.

* **FrmLogin**
  + **Responsabilidades**
    - **Conocer**: controles de texto y botones.
    - **Hacer**:
      * Solicitar credenciales y llamar a UsuarioService.VerificarLogin().
* **FrmUsuarios**, **FrmProductos**, **FrmVentas**, **FrmReportes**, **FrmPrincipal**
  + **Responsabilidades**
    - **Conocer**: componentes UI (DataGridView, ComboBox, TextBox).
    - **Hacer**:
      * Llamar a UsuarioService, ProductoService o VentaService para CRUD y refrescar vistas

### T02. Gestión de Log In / Log Out del Sistema

**T02a. Políticas de autenticación**

* Cada usuario se autentica con su **Nombre de Usuario** y **Contraseña** (almacenada como hash+salt).
* Tras 3 intentos fallidos consecutivos, la cuenta se bloquea temporalmente durante 5 minutos.
* Las contraseñas expiran cada 90 días, forzando un cambio al ingresar.
* El sistema obliga siempre a usar HTTPS en el canal de autenticación.

**T02b. Proceso de arranque del sistema**

* Se inicializa el contexto de la aplicación y la conexión a la base de datos.
* Se carga la configuración de seguridad (salt rounds, políticas de bloqueo).
* Se invoca UsuarioService.SeedAdmin() para asegurar existencia de al menos un administrador.
* Se muestra la ventana de login (**FrmLogin**) en modo modal.

**T02c. Proceso de Log In**

* El usuario ingresa nombre y contraseña en **FrmLogin**.
* **FrmLogin** llama a UsuarioService.VerificarLogin().
* UsuarioService a su vez:
  + Obtiene la entidad **Usuario** y su salt desde BLL/DAL.
  + Invoca a SeguridadService.VerificarPassword().
* Si la validación falla:
  + Se incrementa el contador de intentos en un log de auditoría.
  + Se muestra mensaje de error y se bloquea o suspende según política.
* Si la validación es exitosa:
  + Se instancia (o reutiliza) el singleton **Sesion** y se almacena el objeto **Usuario** con su **Rol** y **Permisos**.
  + Se registra en el log de auditoría el evento LOGIN con timestamp y userId.
  + Se abre **FrmPrincipal** y se oculta el **FrmLogin**.

**T02d. Proceso de Log Out**

* El usuario selecciona la opción “Cerrar sesión” desde el menú de la aplicación.
* El formulario invoca Sesion.Instancia.CerrarSesion().
* **Sesion**:
  + Limpia todas las referencias al usuario actual y su árbol de permisos.
  + Cierra las transacciones pendientes (commit/rollback).
* Se registra en el log de auditoría el evento LOGOUT con timestamp y userId.
* Se vuelve a mostrar **FrmLogin** en modo modal; la aplicación permanece abierta.

**T02e. Proceso de apagado del sistema**

* El usuario selecciona “Salir” en **FrmPrincipal**.
* Si hay sesión activa, se ejecuta internamente el proceso de Log Out (T02d).
* Se cerran todas las conexiones a la base de datos de forma ordenada.
* Se liberan recursos, se guardan configuraciones de usuario y se registra el evento SHUTDOWN en el log.
* La aplicación cierra su ventana principal y finaliza el proceso.

**T02f. Auditoría y control de integridad**

* Todos los eventos relevantes (LOGIN, LOGOUT, SHUTDOWN, errores críticos) se insertan en una tabla de auditoría con:
  1. IdAudit, UserId, TipoEvento, FechaHora, Descripción.
* Antes de cada Log Out o Shutdown, se verifica que no queden transacciones abiertas; si existen, se hace Rollback.
* Se implementa validación de integridad referencial a nivel de base de datos (FK, triggers de integridad).

**Uso del patrón Singleton**

* La clase **Sesion** es un singleton que expone Sesion.Instancia y métodos:
  + IniciarSesion(Usuario u) → almacena el usuario y su árbol de permisos.
  + CerrarSesion() → limpia la sesión y libera recursos.
* Garantiza un único punto de acceso al contexto de usuario en toda la aplicación.

### T03. Gestión de Encriptado

**Algoritmo**: PBKDF2 (RFC 2898) con HMAC–SHA256

**Salting**:

Se genera un **salt aleatorio** de 16 bytes por usuario.

Se almacena junto al hash en la base de datos.

**Parámetros**:

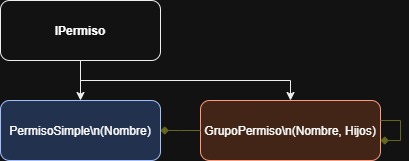
Iteraciones: 100 000

Longitud de salida: 32 bytes (256 bits)

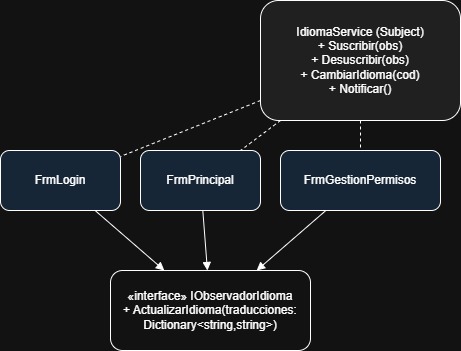
**Flujo**:

* Al registrar o cambiar contraseña, GenerarHashConSalt(plain, out salt) usa Rfc2898DeriveBytes.
* El hash y el salt resultantes se guardan en las columnas Contraseña y Salt de la tabla Usuarios.
* Al verificar, VerificarPassword(plain, storedHash, salt) recomputa el PBKDF2 con los mismos parámetros

### T04. Gestión de Perfiles de Usuario



### T05. Gestión de Múltiples Idiomas



### T06. Gestión de Bitácora y Control de cambios

Cambios realizados:

1. Internacionalización del sistema (Multi-idioma):

* Implementación del patrón Observer para traducciones dinámicas.
* Incorporación de combo de idiomas en el FrmLogin y FrmPrincipal.
* Creación de las tablas: Idiomas, Tag, TraduccionT.
* Traducciones agregadas para: Español, Inglés, Francés, Alemán, Italiano, Portugués.

1. Sistema de Permisos con Composite:

* Implementación del patrón Composite para representar grupos de permisos.
* Formularios permiten crear grupos de permisos (roles complejos) y asignarlos a roles.
* Validación para evitar grupos duplicados o vacíos.
* Evita permisos duplicados visualmente en el TreeView.

1. Correcciones en la base de datos:

* Eliminación de permisos obsoletos.
* Limpieza de registros huérfanos en Grupo\_PermisoDetalle.
* Inserción estructurada de nuevos permisos y relaciones padre/hijo.

### Mejoras de interfaz:

* Formularios hijos ahora se abren maximizados por defecto.
* Traducción visual del TreeView de gestión de permisos según idioma seleccionado.
* Mejora en validación visual al asignar o desasignar permisos.