Gobierno de Reconciliación y Unidad Nacional

El Pueblo, Presidente!

INATEC
Tecnológico Nacional

Curso: logica de programacion

Docente: Ing. Arnoldo Jose Contreras Mercado

Tema: Sistema de Inventario

Documentación del Proyecto:

El presente documento describe un Sistema de Gestión de Inventario desarrollado en Python. El sistema sigue una arquitectura modular y utiliza Tkinter para la Interfaz Gráfica de Usuario (GUI) y MySQL/MariaDB (según configuración de bdd.py) como motor de persistencia de datos. El sistema implementa un control de acceso basado en roles (Admin y Usuario) para gestionar permisos sobre operaciones críticas del inventario.

1. Arquitectura y Componentes del Proyecto

El proyecto está organizado en módulos de Python que separan claramente las responsabilidades.

1.1 Estructura de Archivos

Archivo	Rol Principal	Tecnologías	Descripción
main.py	**Punto de Entrada**	Python, Tkinter	Inicializa la aplicación principal y el bucle de eventos de Tkinter.
bdd.py	**Capa de Persistencia**	Python, MySQL	Contiene la clase BaseDatos para manejar la conexión y las consultas CRUD con el servidor MySQL.
producto.py	**Modelo de Datos**	Python	Define la clase Producto, incluyendo atributos con *getters* y *setters* (@property).
inventario.py	**Lógica de Negocio (Productos)**	Python	Define la clase Inventario que gestiona la colección de objetos Producto.
interfazinventario.py	**Capa de**	Python, Tkinter	Implementa la GUI

	Presentación		principal, el login de roles, y el manejo de todas las interacciones de usuario.
inventario_mariadb.sql	Esquema DB	SQL	Archivo que contiene el Data Definition Language (DDL) para crear las tablas en MySQL/MariaDB.
requirements.txt	Dependencias	N/A	Lista las dependencias del proyecto (tkinter, mysql- *nota: bdd.py usa mysql.connector*).

2. Modelos de Datos

2.1 Producto (producto.py)

La clase Producto encapsula los datos de un artículo del inventario.

Atributo	**Tipo**	**Descripción**
producto_id	int	Identificador único del producto (PK en la DB).
nombre	str	Nombre comercial del producto.
descripcion	str	Descripción detallada.
precio	float	Precio de venta.
cantidad_stock	int	Unidades disponibles en el inventario.
proveedor	str	Nombre o ID del proveedor.

- **Implementación:** Todos los atributos principales son implementados con *properties* (@property y @setter) para permitir la validación o el manejo controlado de cambios.

2. Capa de Persistencia (bdd.py)

La clase BaseDatos es la capa de abstracción de datos.

2.1 Motor y Conexión

- **Motor:** El código utiliza mysql.connector y asume una base de datos **MySQL/MariaDB** con la configuración:

```
MYSQL_CONFIG = {
    'host': 'localhost',
    'user': 'root',
```

```
    'password': '',
    'database': 'sistema_inventario'
}
```

- **Manejo de Errores:** Incluye manejo de excepciones específicas (mysql.connector.Error) y muestra advertencias (messagebox.showerror) si la conexión falla o una consulta da error.

2.2 Esquema de Tablas (Según inventario_mariadb.sql)

El diseño de la base de datos utiliza un esquema relacional con las siguientes tablas clave:

Tabla	Propósito	Relaciones (FK)	Notas
usuarios	Almacena información de acceso y el rol (Admin/Usuario).	N/A	Útil para el control de permisos en la GUI.
productos	Almacena los artículos del inventario.	FK a proveedores	Contiene el campo cantidad_stock crítico.

3. Lógica de Negocio y Flujo de la Aplicación

3.1 Inventario y Validación

- **Clase Inventario:** Actúa como el manejador de la colección de productos en memoria.
  - Métodos clave: agregar_producto, eliminar_producto, buscar_producto, generar_informe.
  - **Validación:** La lógica de negocio (inventario.py) es la responsable de manejar la lista de objetos Producto y asegurar la integridad de los datos antes de persistir en la DB.

3.2 Interfaz de Usuario (interfazinventario.py)

La GUI es la capa de presentación que orquesta las llamadas a la Lógica de Negocio y a la Base de Datos.

Control de Acceso (Permisos)

- **Rol de Usuario:** Se establece en el inicio de la aplicación (self.user_role).
- **Método aplicar_permisos_por_rol():** Restringe funcionalidades en la interfaz:
  - Si self.user_role es **"Usuario"**, los botones de Eliminar y Modificar producto son configurados a state=tk.DISABLED.
  - Si self.user_role es **"Admin"**, todas las funciones están disponibles.

Funcionalidades Principales (Pestañas Tkinter)

El uso de ttk.Notebook organiza las siguientes vistas:

1. **Agregar Producto:** Formulario para la creación de nuevos ítems.
2. **Modificar Producto:** Búsqueda, carga y edición de datos de un producto existente.
3. **Estadísticas:** Muestra métricas agregadas del inventario (ej. valor total, cantidad total).
4. **Ventas:** Interfaz para el registro de nuevas transacciones y visualización de informes de ventas.

3.3 Flujo de Inicio (main.py)

1. Se inicializa la ventana principal de Tkinter (root = tk.Tk()).
2. Se crea la instancia de la aplicación app = InterfazInventario(root).
    - Dentro del constructor de InterfazInventario, se inicializan los objetos BaseDatos, Inventario y Ventas, y se cargan los productos desde la DB.
3. Se inicia el bucle principal de la aplicación (root.mainloop()).