

Anexo C

Documentación del Sistema de Gestión de Inventario en Python (mejoras en la base de datos)

1. Introducción

El sistema de gestión de inventarios desarrollado en Python utiliza una interfaz gráfica moderna creada con Tkinter. El sistema permite a los usuarios realizar operaciones de gestión del inventario (crear, leer, actualizar y eliminar) y, además, proporciona acceso directo a documentos importantes y funcionalidades visuales mejoradas como botones con íconos, logotipos y tablas interactivas.

2. Importación de librerías

Primero, se importan las librerías necesarias para el funcionamiento del programa como se muestra a continuación:

Figura 1

Importación de librerías

```
1 import json
2 import os
3 from tkinter import *
4 from tkinter import ttk, messagebox, filedialog
5 from datetime import datetime
6 import openpyxl
7 import mysql.connector
8 from PIL import Image, ImageTk
```

- json y os son módulos utilizados para manejar archivos y datos de configuración.
- tkinter, ttk, messagebox y filedialog permiten crear la interfaz gráfica.



- datetime es usado para obtener la fecha actual.
- openpyxl permite trabajar con archivos de Excel.
- mysql.connector se utiliza para conectar a la base de datos MySQL.

3. Interfaz Gráfica Mejorada

3.1 Logotipos en el menú Superior

Se añadieron tres logotipos al menú superior para representar las instituciones asociadas. Estos se implementan en un **Frame** y se cargan dinámicamente desde la carpeta de **assets**.

Figura 2

Logotipos del menú superior

```
1 # Crear el marco para los logotipos
2 logo_frame = Frame(window, bg="#f0f0f0", height=100)
3 logo_frame.pack(fill=X, pady=5)
4
5 # Cargar imágenes
6 logo_izquierda = cargar_imagen(iconos['logo_izquierda'], tamaño
7                               =(200, 100))
8 logo_central = cargar_imagen(iconos['logo_central'], tamaño
9                              =(200, 100))
```



```

8 logo_derecha = cargar_imagen(iconos['logo_derecha'], tamaño
    =(200, 100))
9
10 # Agregar logotipos al marco
11 Label(logo_frame, image=logo_izquierda, bg="#f0f0f0").pack(side=
    LEFT, padx=10)
12 Label(logo_frame, image=logo_central, bg="#f0f0f0").pack(side=
    LEFT, padx=10, expand=True)
13 Label(logo_frame, image=logo_derecha, bg="#f0f0f0").pack(side=
    RIGHT, padx=10)

```

Descripción:

- El logotipo izquierdo representa al laboratorio o departamento.
- El logotipo central simboliza la institución principal.
- El logotipo derecho pertenece a un socio estratégico.
- Los logotipos son imágenes redimensionadas para mantener la coherencia visual.

3.2 Botones de Documentos con Funcionalidad

Se añadieron tres botones en el menú principal para abrir documentos clave: plan de mantenimiento, normas generales y el manual de usuario. Estos botones permiten abrir los archivos específicos directamente desde la interfaz gráfica.

Figura 3

Botones para acceso a documentos



```

1 # Funci n para abrir archivos
2 def abrir_archivo(ruta_archivo):
3     if os.path.exists(ruta_archivo):
4         os.startfile(ruta_archivo)
5     else:
6         messagebox.showerror("Error", f"El archivo no se
7             encuentra: {ruta_archivo}")
8
9 # Botones del men
10 crear_boton_icono(menu_frame, iconos['plan'], "Plan de
11     Mantenimiento", lambda: abrir_archivo(os.path.join(DATA_DIR, "
12         Plan.xlsx")))
13 crear_boton_icono(menu_frame, iconos['norma'], "Normas Generales"
14     , lambda: abrir_archivo(os.path.join(DATA_DIR, "Normas.xlsx")))
15 crear_boton_icono(menu_frame, iconos['manual'], "Manual de
16     Usuario", lambda: abrir_archivo(os.path.join(DATA_DIR, "Manual.
17         pdf")))
```

Funcionalidad de los Botones:

- Plan de Mantenimiento: Abre el archivo Excel Plan.xlsx, que contiene el plan detallado de mantenimiento.
- Normas Generales: Abre el archivo Excel Normas.xlsx, donde se describen las normas y políticas aplicables.
- Manual de Usuario: Abre el archivo PDF Manual.pdf, que proporciona una guía detallada sobre el uso del sistema.

3.3 Tooltips para Botones

Para mejorar la experiencia del usuario, cada botón incluye un tooltip que se muestra al pasar el cursor sobre él.



Figura 4

Tooltips para Botones

```

1 def crear_boton_icono(parent, ruta_imagen, texto_tooltip, comando
2 ):
3     icono = cargar_imagen(ruta_imagen)
4     btn = Button(parent, image=icono, relief=FLAT, command=
5         comando)
6     btn.image = icono
7     btn.pack(side=LEFT, padx=10)
8
9     def mostrar_tooltip(event):
10         tooltip = Toplevel(parent)
11         tooltip.wm_overrideredirect(True)
12         tooltip.wm_geometry(f"+{event.x_root + 10}+{event.y_root
13             + 10}")
14         label = Label(tooltip, text=texto_tooltip, background="
15             lightyellow", relief=SOLID, padx=5)
16         label.pack()
17         btn.bind("<Leave>", lambda e: tooltip.destroy())
18         btn.bind("<Enter>", mostrar_tooltip)

```

Descripción:

- Los tooltips brindan información adicional sobre la funcionalidad del botón.
- El texto se muestra en un cuadro flotante cuando el usuario pasa el cursor sobre el botón.

3.4 Tabla de Resultados

La interfaz incluye una tabla para mostrar los datos del inventario con columnas que incluyen información como hoja de vida, proveedor, y valor de compra.

Figura 5

Tabla para mostrar datos



```
1 my_tree = ttk.Treeview(table_frame, show='headings', height=20)
2 my_tree['columns'] = ("ID", "Cantidad", "Descripcion", "Marca", "
    Fecha", "Observaciones", "Lugar", "Condicion", "Proveedor", "
    A os", "Valor", "Hoja_de_vida")
3 for col in my_tree['columns']:
4     my_tree.heading(col, text=col)
5     my_tree.column(col, anchor=W)
```

Mejoras Implementadas:

- Se añadieron columnas adicionales para una gestión más completa del inventario.
- Las columnas incluyen Hoja de vida para almacenar la ruta de documentos específicos relacionados con los artículos.

4. Resumen de las Mejoras en la Interfaz Grafica

- Inclusión de logotipos representativos en el menú superior.
- Botones para acceso rápido a documentos importantes.
- Tooltips en los botones para mejorar la usabilidad.
- Tabla interactiva con columnas personalizadas para mostrar datos del inventario.

5. Conclusión

El sistema no solo permite la gestión de inventarios, sino que también mejora significativamente la experiencia del usuario con una interfaz gráfica atractiva, accesos rápidos a documentos y funcionalidades visuales mejoradas. Este diseño lo convierte en una herramienta eficiente y profesional para laboratorios o instituciones.