

Juan Díaz de la Vega Pérez  
A00575311

- Integrante D → `win_table.py`

Reportes de Contaminación				
ID	Usuario	Ubicación	Tipo de desecho	Fecha
001	Juan	Mazatlán	Vidrio	01/01/25
002	Pato	Veracruz	metal	01/01/25
003	Enrique	Puerto morelos	Plástico	01/01/25

Agregar

Editar

Eliminar

Tabla Interactiva:

```
# win_table.py
import tkinter as tk
from tkinter import ttk, messagebox, simpledialog

class TableWindow(tk.Toplevel):
    def __init__(self, master=None):
        super().__init__(master)
        self.title("Reportes de Contaminación")
        self.geometry("700x400")

        # Definir columnas de la tabla
        self.columns = ("ID", "Usuario", "Ubicación", "Tipo de desecho", "Fecha")
        self.tree = ttk.Treeview(self, columns=self.columns, show="headings")
        self.tree.pack(fill="both", expand=True, pady=10)

        # Encabezados
        for col in self.columns:
            self.tree.heading(col, text=col)
            self.tree.column(col, width=120, anchor="center")

        # Scrollbar
        scrollbar = ttk.Scrollbar(self, orient="vertical", command=self.tree.yview)
        self.tree.configure(yscrollcommand=scrollbar.set)
        scrollbar.pack(side="right", fill="y")

        # Datos iniciales de ejemplo
        data = [
            (1, "Juan", "Río Bravo", "Plástico", "2025-09-30"),
            (2, "Ana", "Playa Norte", "Vidrio", "2025-09-29"),
            (3, "Luis", "Laguna Verde", "Metal", "2025-09-28"),
        ]
        for row in data:
            self.tree.insert("", tk.END, values=row)

        # Botones de acción
        btn_frame = tk.Frame(self)
        btn_frame.pack(pady=10)

        tk.Button(btn_frame, text="Agregar", command=self.add_row).pack(side="left", padx=5)
        tk.Button(btn_frame, text="Editar", command=self.edit_row).pack(side="left", padx=5)
        tk.Button(btn_frame, text="Eliminar", command=self.delete_row).pack(side="left",
padx=5)

        # --- Funciones interactivas ---
        def add_row(self):
            try:
                new_id = len(self.tree.get_children()) + 1
```

```

usuario = simpledialog.askstring("Nuevo registro", "Usuario:")
ubicacion = simpledialog.askstring("Nuevo registro", "Ubicación:")
tipo = simpledialog.askstring("Nuevo registro", "Tipo de desecho:")
fecha = simpledialog.askstring("Nuevo registro", "Fecha (YYYY-MM-DD):")

if usuario and ubicacion and tipo and fecha:
    self.tree.insert("", tk.END, values=(new_id, usuario, ubicacion, tipo, fecha))
else:
    messagebox.showwarning("Datos incompletos", "Debes llenar todos los campos.")
except Exception as e:
    messagebox.showerror("Error", str(e))

def edit_row(self):
    selected = self.tree.selection()
    if not selected:
        messagebox.showwarning("Selecciona un registro", "Debes elegir un registro para
editar.")
    return

    item = selected[0]
    values = self.tree.item(item, "values")

    usuario = simpledialog.askstring("Editar registro", "Usuario:", initialvalue=values[1])
    ubicacion = simpledialog.askstring("Editar registro", "Ubicación:", initialvalue=values[2])
    tipo = simpledialog.askstring("Editar registro", "Tipo de desecho:",
initialvalue=values[3])
    fecha = simpledialog.askstring("Editar registro", "Fecha (YYYY-MM-DD):",
initialvalue=values[4])

    if usuario and ubicacion and tipo and fecha:
        self.tree.item(item, values=(values[0], usuario, ubicacion, tipo, fecha))

def delete_row(self):
    selected = self.tree.selection()
    if not selected:
        messagebox.showwarning("Selecciona un registro", "Debes elegir un registro para
eliminar.")
    return

    confirm = messagebox.askyesno("Eliminar", "¿Seguro que quieres eliminar este
registro?")
    if confirm:
        for item in selected:
            self.tree.delete(item)

# Para pruebas individuales
if __name__ == "__main__":

```

```
root = tk.Tk()
root.withdraw()
app = TableWindow(root)
app.mainloop()
```