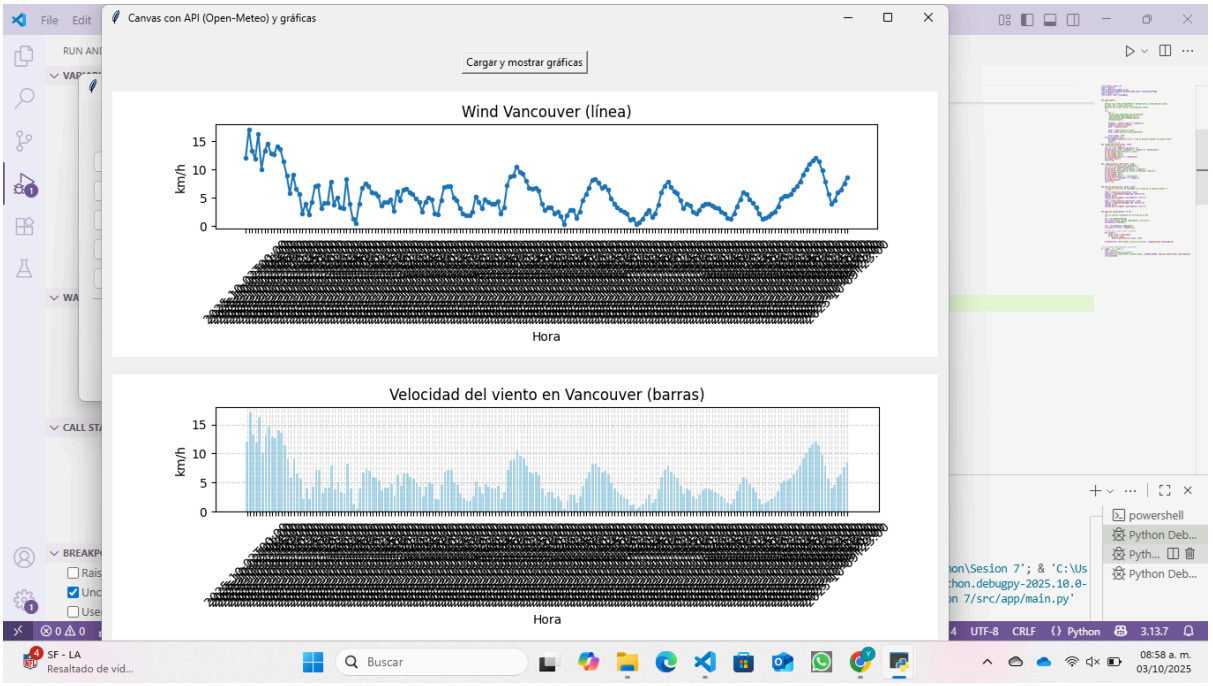
Yenyfer González Campos

Sesión 7

Cambios ventana win_canvas.py
-Ciudad de Vancouver en vez de León
-Windspeed en vez de Temperatura



Código:

```
from tkinter import ttk
import requests
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
import tkinter as tk
from tkinter import messagebox


def fetch_data():
    """
    Conecta con la API de Open-Meteo y obtiene horas y velocidad del
viento
    de León, Gto (últimas 24 horas).
    Devuelve dos listas: horas y velocidad del viento.
    """
    try:
        url = (
        "https://api.open-meteo.com/v1/forecast"
```

```
        "?latitude=49.28&longitude=-123.12"
        "&hourly=wind_speed_10m&past_days=1"
        "&timezone=auto"
        )
        response = requests.get(url, timeout=15)
        response.raise_for_status()
        data = response.json()

        horas = data["hourly"]["time"]
        wind = data["hourly"]["wind_speed_10m"]

        return horas, wind
    except Exception as e:
        messagebox.showerror("Error", f"No se pudieron obtener los
datos:\n{e}")
        print(e)
        return [], []
def create_line_chart(horas, wind):
    """Gráfica de línea."""
    fig, ax = plt.subplots(figsize=(6, 3))
    ax.plot(horas, wind, linestyle="-", marker="o", markersize=3)
    ax.set_title("Wind Vancouver (línea)")
    ax.set_xlabel("Hora")
    ax.set_ylabel("km/h")
    ax.tick_params(axis="x", rotation=45)
    fig.tight_layout()
    return fig

def create_wind_bar_chart(horas, wind):
    """Gráfica de barras para windspeed."""
    fig, ax = plt.subplots(figsize=(6, 3))
    ax.bar(horas, wind, color='skyblue', alpha=0.7)
    ax.set_title("Velocidad del viento en Vancouver (barras)")
    ax.set_xlabel("Hora")
    ax.set_ylabel("km/h")
    ax.tick_params(axis="x", rotation=45)
    ax.grid(True, linestyle="--", alpha=.5)
    fig.tight_layout()
    return fig


def mostrar_graficas(frm, horas, wind):
```

```
    """Inserta la gráfica de windspeed en el frame de la ventana
tkinter."""
    # Línea
    fig1 = create_line_chart(horas, wind)
    canvas1 = FigureCanvasTkAgg(fig1, master=frm)
    canvas1.draw()
    canvas1.get_tk_widget().pack(pady=10, fill="x")
    # Barras de viento
    fig3 = create_wind_bar_chart(horas, wind)
    canvas3 = FigureCanvasTkAgg(fig3, master=frm)
    canvas3.draw()
    canvas3.get_tk_widget().pack(pady=10, fill="x")


def open_win_canvas(parent: tk.Tk):
    """
    Crea la ventana secundaria con gráficas de la API.
    """
    win = tk.Toplevel(parent)
    win.title("Canvas con API (Open-Meteo) y gráficas")
    win.geometry("960x1000")

    frm = ttk.Frame(win, padding=12)
    frm.pack(fill="both", expand=True)

    # Botón para cargar datos y graficar
    def cargar():
        horas, wind = fetch_data()
        if horas and wind:
            mostrar_graficas(frm, horas, wind)

    tk.Button(frm, text="Cargar y mostrar gráficas",
command=cargar).pack(pady=10)


# Para pruebas independientes (opcional)
if __name__ == "__main__":
    root = tk.Tk()
    root.title("Prueba win_canvas")
    tk.Button(root, text="Abrir ventana Canvas", command=lambda:
open_win_canvas(root)).pack(pady=20)
    root.mainloop()
```