

```

import io
import pandas as pd
import streamlit as st

# --- FUNCIÓN PARA CORRECCIÓN NUMÉRICA ---
@st.cache_data
def standardize_numeric_column(series):
    """
    Convierte una serie de Pandas a valores numéricos de manera robusta,
    reemplazando comas por puntos como separador decimal.
    """
    series_clean = series.astype(str).str.replace(",", ".", regex=False)
    return pd.to_numeric(series_clean, errors="coerce")

def display_plotly_download_buttons(fig, file_prefix):
    """
    Muestra botones de descarga para un gráfico Plotly (HTML y PNG).
    """
    st.markdown("---")
    col1, col2 = st.columns(2)
    with col1:
        html_buffer = io.StringIO()
        fig.write_html(html_buffer, include_plotlyjs="cdn")
        st.download_button(
            label="Descargar Gráfico (HTML)",
            data=html_buffer.getvalue(),
            file_name=f"{file_prefix}.html",
            mime="text/html",
            key=f"dl_html_{file_prefix}",
        )
    with col2:
        try:
            img_bytes = fig.to_image(format="png", width=1200, height=700, scale=2)
            st.download_button(
                label="Descargar Gráfico (PNG)",
                data=img_bytes,
                file_name=f"{file_prefix}.png",
                mime="image/png",
                key=f"dl_png_{file_prefix}",
            )
        except Exception:
            st.warning(
                "No se pudo generar la imagen PNG. Asegúrate de tener 'kaleido' instalado."
            )
    )

def add_folium_download_button(map_object, file_name):

```

```
"""Muestra un botón de descarga para un mapa de Folium (HTML)."""
st.markdown("---")
map_buffer = io.BytesIO()
map_object.save(map_buffer, close_file=False)
st.download_button(
    label="Descargar Mapa (HTML)",
    data=map_buffer.getvalue(),
    file_name=file_name,
    mime="text/html",
    key=f"dl_map_{file_name.replace('.', '_')}",
)

```