Nombre	Hidequel Puga
NAO ID	3049
Fecha	24-ago-24
Trayectoria	Data Analyst Core
Reto	Bibliotecas de Python y herramientas de visualización de datos

3_d_evolution_delayed_orders_by_region.py

```
import os
import pandas as pd
import plotly.express as px

# Cargar los datos procesados
DATA_PATH="C:/Users/pugah/Documents/CFS/GitHub/DigitalNAO-Challenges/PythonVisualizacionDatos/data/"
FILE_CONSOLIDATED_DATA = 'oilst_processed.csv'
RESULT_PATH="C:/Users/pugah/Documents/CFS/GitHub/DigitalNAO-Challenges/PythonVisualizacionDatos/Sprint_2/Desarrolla/resultado/"
FILE_RESULT='3_d_evolution_delayed_orders_by_region.html'

data = pd.read_csv(os.path.join(DATA_PATH, FILE_CONSOLIDATED_DATA))

# Filtrar las órdenes con retrasos prolongados
long_delay_orders = data[data['delay_status'] == 'Largo']

# Crear una columna con el año y el mes de la fecha de compra
long_delay_orders['year_month'] = pd.to_datetime(long_delay_orders['order_purchase_timestamp']).dt.to_period('M')

# Agrupar las órdenes con retrasos prolongados por año_mes y estado
delayed_orders_by_region = long_delay_orders.groupby(['year_month', 'customer_state']).size().reset_index(name='order_count')

# Crear una figura de barras apiladas interactiva
fig = px.bar(delayed_orders_by_region,
             x='year_month',
             y='order_count',
             color='customer_state',
```

```
            title="Evolución de Órdenes con Retrasos Prolongados por Mes y
Región",
            labels={'year_month': 'Mes y Año', 'order_count': 'Cantidad de
Órdenes'},
            barmode='stack')

# Ajustar la presentación del gráfico
fig.update_layout(xaxis_title="Mes y Año",
                  yaxis_title="Cantidad de Órdenes",
                  legend_title="Estado",
                  xaxis_tickangle=-45)

# Guardar la figura interactiva en un archivo HTML
fig.write_html(os.path.join(RESULT_PATH, FILE_RESULT))

print("El archivo " + FILE_RESULT + " ha sido generado con éxito.")
```

3_e_map_long_delays_by_state.py

```
import os
import pandas as pd
import folium
import geopandas as gpd
from folium.plugins import HeatMap

# Cargar los datos procesados
DATA_PATH="C:/Users/pugah/Documents/CFS/GitHub/DigitalNAO-Challenges/PythonVisual
izacionDatos/data/"
FILE_CONSOLIDATED_DATA = 'oilst_processed.csv'
FILE_GEODATA = 'brasil_geodata.json'
RESULT_PATH="C:/Users/pugah/Documents/CFS/GitHub/DigitalNAO-Challenges/PythonVisu
alizacionDatos/Sprint_2/Desarrolla/resultado/"
FILE_RESULT='3_e_map_long_delays_by_state.html'

data = pd.read_csv(os.path.join(DATA_PATH, FILE_CONSOLIDATED_DATA))

# Filtrar las órdenes con retrasos prolongados
long_delay_orders = data[data['delay_status'] == 'Largo']

# Agrupar los retrasos por estado
delays_by_state =
long_delay_orders.groupby('customer_state').size().reset_index(name='count')

# Cargar el GeoJSON con las fronteras de los estados
geodata = gpd.read_file(os.path.join(DATA_PATH, FILE_GEODATA))

# Crear un mapa centrado en Brasil
m = folium.Map(location=[-15.7801, -47.9292], zoom_start=4)

# Añadir la capa de colores por estado
folium.Choropleth(
    geo_data=geodata,
    name="choropleth",
    data=delays_by_state,
    columns=["customer_state", "count"],
    key_on="feature.properties.ESTADO",
```

```
    fill_color="YlOrRd",
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name="Número de retrasos prolongados por estado",
).add_to(m)

# Guardar el mapa en un archivo HTML
m.save(os.path.join(RESULT_PATH, FILE_RESULT))

print("El archivo " + FILE_RESULT + " ha sido generado con éxito.")
```

3_f_map_monetary_value_long_delays_by_state.py

```
import os
import pandas as pd
import folium
import geopandas as gpd

# Cargar los datos procesados
DATA_PATH="C:/Users/pugah/Documents/CFS/GitHub/DigitalNAO-Challenges/PythonVisualizacionDatos/data/"
FILE_GEODATA = 'brasil_geodata.json'
FILE_CONSOLIDATED_DATA = 'oilst_processed.csv'
RESULT_PATH="C:/Users/pugah/Documents/CFS/GitHub/DigitalNAO-Challenges/PythonVisualizacionDatos/Sprint_2/Desarrolla/resultado/"
FILE_RESULT='3_f_map_monetary_value_long_delays_by_state.html'

data = pd.read_csv(os.path.join(DATA_PATH, FILE_CONSOLIDATED_DATA))

# Filtrar las órdenes con retrasos prolongados
long_delay_orders = data[data['delay_status'] == 'Largo']

# Agrupar el valor monetario total de las órdenes con retrasos prolongados por estado
monetary_value_by_state = long_delay_orders.groupby('customer_state')['total_sales'].sum().reset_index()

# Cargar el GeoJSON con las fronteras de los estados (ajusta la ruta al archivo)
geodata = gpd.read_file(os.path.join(DATA_PATH, FILE_GEODATA))

# Crear un mapa centrado en Brasil
m = folium.Map(location=[-15.7801, -47.9292], zoom_start=4)

# Añadir la capa de colores por estado basada en el valor monetario
folium.Choropleth(
    geo_data=geodata,
    name="choropleth",
    data=monetary_value_by_state,
    columns=["customer_state", "total_sales"],
    key_on="feature.properties.ESTADO",
```

```
    fill_color="YlGnBu",
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name="Valor Monetario de Retrasos Prolongados por Estado
(total_sales)",
).add_to(m)

# Guardar el mapa en un archivo HTML
m.save(os.path.join(RESULT_PATH, FILE_RESULT))

print("El archivo " + FILE_RESULT + " ha sido generado con éxito.")
```