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_a_histogram_sales_short_long_delays.py

```
import os
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')

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

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

# Filtrar las órdenes con retrasos moderados y prolongados
short_delay_orders = data[data['delay_status'] == 'Corto']
long_delay_orders = data[data['delay_status'] == 'Largo']

# Extraer las ventas de las órdenes completas para cada categoría de retraso
short_sales = short_delay_orders['total_sales']
long_sales = long_delay_orders['total_sales']

# Crear la figura y los ejes para los histogramas
plt.figure(figsize=(10, 6))
```

```
# Histogramas para retrasos cortos y largos
plt.hist(short_sales, bins=30, color='blue', alpha=0.6, label='Retraso Corto')
plt.hist(long_sales, bins=30, color='red', alpha=0.6, label='Retraso Largo')

# Añadir etiquetas y título
plt.xlabel('Total Sales')
plt.ylabel('Frequency')
plt.title('Histogram of Total Sales for Short and Long Delays')
plt.legend()

# Guardar la imagen como un archivo PNG
plt.savefig(os.path.join(RESULT_PATH, FILE_RESULT))

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

3_b_correlation_matrix_complete_orders.py

```
import os
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')

# 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_b_correlation_matrix_complete_orders.png'

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

# Filtrar las órdenes completadas
complete_orders = data[data['order_status'] == 'delivered']

# Seleccionar las variables numéricas junto con delta_days
numeric_columns = ['total_sales', 'total_products', 'delta_days', 'distance_distribution_center']

# Filtrar las columnas relevantes para las órdenes completadas
correlation_data = complete_orders[numeric_columns]

# Calcular la matriz de correlación
correlation_matrix = correlation_data.corr()

# Crear la figura y el mapa de calor
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1, linewidths=0.5, fmt='.2f')
```

```python
# Añadir título
plt.title('Correlation Matrix for Complete Orders')

# Guardar la figura en un archivo PNG
plt.savefig(os.path.join(RESULT_PATH, FILE_RESULT))

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

3_c_delta_day_by_state_and_delay_type.py

```
import os
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')

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

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

# Crear la figura para la visualización
plt.figure(figsize=(14, 8))

# Crear un gráfico de cajas (boxplot) para observar la distribución de delta_days
por estado y delay_status
sns.boxplot(x='customer_state', y='delta_days', hue='delay_status', data=data)

# Añadir título y etiquetas
plt.title('Distribución de Delta Days por Estado y Tipo de Retraso')
plt.xlabel('Estado')
plt.ylabel('Delta Days')

# Rotar las etiquetas del eje x para una mejor legibilidad
plt.xticks(rotation=45)

# Guardar la figura en un archivo PNG
plt.savefig(os.path.join(RESULT_PATH, FILE_RESULT))
```

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