```
import pandas as pd
import numpy as np
import json
import matplotlib.pyplot as plt
import seaborn as sns
import re # For Regular expression


# Load datasets
customer_file_path = "/content/drive/MyDrive/PEI DataSets/Customer.xlsx"
order_file_path = "/content/drive/MyDrive/PEI DataSets/Order.csv"
shipping_file_path = "/content/drive/MyDrive/PEI DataSets/Shipping.json"


# customer_df = pd.read_excel(customer_file_path, engine="xlrd")
# Engine= xlrd as the file is in xls format which is old one
# customer_file_path = "/content/drive/MyDrive/PEI DataSets/Customer.xls"
#customer_df = pd.read_excel(customer_file_path, engine="xlrd")


# Load Customer Data
customer_df = pd.read_excel(customer_file_path)

# Load Order Data
order_df = pd.read_csv(order_file_path)

# Load Shipping Data
shipping_df = pd.read_json(shipping_file_path)


# Function to Perform EDA + Data Cleaning
# https://emojidb.org/stats-emojis emojis or icons are taken from this website for better look and feel

def perform_eda_and_clean(df, name):
    print(f"\n📊 EDA + Data Cleaning for {name} Dataset:")

    # 📝 1. Columns and Data Types
    print("\n📝 Columns and Data Types:")
    print(df.info())

    # 🔍 2. Printing First 5 Rows
    print("\n🔍 First 5 Rows:")
    print(df.head())

    # 📌 3. Check for Missing values
    print("\n Missing Values Count:")
    print(df.isnull().sum())

    # 📈 4. Summary Statistics for Numerical Data
    print("\n📈 Summary Statistics (Numerical Data):")
    print(df.describe())

    # ✅ 5. Unique Values Per Column
    print("\n✅ Unique Values Per Column:")
    print(df.nunique())

    # 🔍 6. Check for Special Characters in String Columns
    print("\n🔍 Special Character Check:")

    # Define regex pattern for special characters (excluding space, a-z, A-Z, 0-9, and basic punctuation)
    special_char_pattern = re.compile(r'[^A-Za-z0-9\s.,]')

    for col in df.select_dtypes(include=["object"]).columns:
        # Find all special characters in the column
        special_chars = df[col].astype(str).apply(lambda x: set(re.findall(special_char_pattern, x)))

        # Get unique special characters found in the column
        unique_special_chars = set().union(*special_chars)

        if unique_special_chars:
            print(f"⚠️ Column `{col}` contains {len(unique_special_chars)} unique special characters: {unique_special_chars}")
        else:
            print(f"✅ Column `{col}` has no special characters.")

        # 🔥 7. Data Cleaning - Remove Special Characters
        df[col] = df[col].apply(lambda x: re.sub(special_char_pattern, '', str(x)))
    # 🔥 8. Handle Missing Values
    for col in df.columns:
        if df[col].isnull().sum() > 0:  # If missing values exist
            if df[col].dtype == "object":
                df[col].fillna("Unknown", inplace=True)  # Fill text columns with "Unknown"
            else:
                df[col].fillna(df[col].median(), inplace=True)  # Fill numeric columns with median

    # 🔥 9. Remove Duplicate Rows
    before = len(df)
    df.drop_duplicates(inplace=True)
    after = len(df)
    print(f"\n✅ Removed {before - after} duplicate rows.")

    # 🔥 10. Ensure Correct Data Types
    if "Age" in df.columns:
        df["Age"] = df["Age"].astype(int)  # Convert Age to integer

    if "Amount" in df.columns:
        df["Amount"] = df["Amount"].astype(float)  # Convert Amount to float

    print("\n✅ Data Cleaning Completed! Dataset is Ready for Analysis 🚀")
    return df  # Return cleaned DataFrame



# Perform EDA on each dataset
customer_df = perform_eda_and_clean(customer_df, "Customer")


# Perform EDA on Order dataset
order_df = perform_eda_and_clean(order_df, "Order")


shipping_df = perform_eda_and_clean(shipping_df, "Shipping")
```

```
📊 EDA + Data Cleaning for Shipping Dataset:

📝 Columns and Data Types:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250 entries, 0 to 249
Data columns (total 3 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Shipping_ID  250 non-null    int64 
 1   Status       250 non-null    object
 2   Customer_ID  250 non-null    int64 
dtypes: int64(2), object(1)
memory usage: 6.0+ KB
None

🔍 First 5 Rows:
   Shipping_ID     Status  Customer_ID
0            1    Pending          173
1            2    Pending          155
2            3  Delivered          242
3            4    Pending          223
4            5  Delivered           72

 Missing Values Count:
Shipping_ID    0
Status         0
Customer_ID    0
dtype: int64

📈 Summary Statistics (Numerical Data):
       Shipping_ID  Customer_ID
count   250.000000   250.000000
mean    125.500000   120.620000
std      72.312977    73.893848
min       1.000000     1.000000
25%      63.250000    53.250000
50%     125.500000   118.000000
75%     187.750000   187.500000
max     250.000000   248.000000

✅ Unique Values Per Column:
Shipping_ID    250
Status           2
Customer_ID    154
dtype: int64
```

```
🔍 Special Character Check:
✅ Column `Status` has no special characters.

✅ Removed 0 duplicate rows.

✅ Data Cleaning Completed! Dataset is Ready for Analysis 🚀
```

```
'''
#Check for Duplicates in each dataset
df = order_df
duplicates = df[df.duplicated(keep=False)]  # Get all duplicate rows
total_duplicates = df.duplicated().sum()  # Count duplicate rows
print(f"\n📊 Checking Duplicates in {df} Dataset:")
print(f"🔄 Total Duplicate Rows: {total_duplicates}") '''
```