```
import numpy as np
import pandas as pd
# import os
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

```
# Load the dataset
df = pd.read_excel("/content/mahakumbh_detailed_data.xlsx")

# Display basic info
print("Dataset Info:")
(df.info())
```

```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22 entries, 0 to 21
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Date                    22 non-null     datetime64[ns]
 1   No of Visitors          22 non-null     object
 2   Total Male              22 non-null     object
 3   Total Female            22 non-null     object
 4   Total Children          22 non-null     object
 5   Total Saint             22 non-null     object
 6   Mode of Transport       22 non-null     object
 7   Duration of Stay        22 non-null     object
 8   Accommodation Type      22 non-null     object
 9   Water Quality           22 non-null     object
 10  Event Type              22 non-null     object
 11  Crowd Density           22 non-null     object
 12  Accident Report         22 non-null     int64
 13  Social Media Sentiment  22 non-null     object
 14  Revenue Collected       22 non-null     int64
 15  Waste Management Data   22 non-null     int64
 16  Medical Emergencies     22 non-null     int64
 17  Security Incidents      22 non-null     int64
 18  Pilgrimage Type         22 non-null     object
 19  Popular Spots           22 non-null     object
dtypes: datetime64[ns](1), int64(5), object(14)
memory usage: 3.6+ KB
```

```
import pandas as pd

# Load the Excel file
file_path ="/content/mahakumbh_detailed_data.xlsx"
xls = pd.ExcelFile(file_path)

# Define the sheet names (based on the given list)
sheet_name = ["mahakumbh_detailed_data" , "Public sentiment" , "Environmetal_impact"]

# Dictionary to store DataFrames for each sheet
dfs = {sheet: pd.read_excel(xls, sheet_name=sheet) for sheet in sheet_name}

# Display each sheet as a table
for sheet, df in dfs.items():
    print(f"\n📊 Table for Sheet: {sheet}")
```

```
print(df.head())  # Display the first 5 rows of each sheet as a sample
print(f"\nColumns in {sheet}: {df.columns}\n")
```

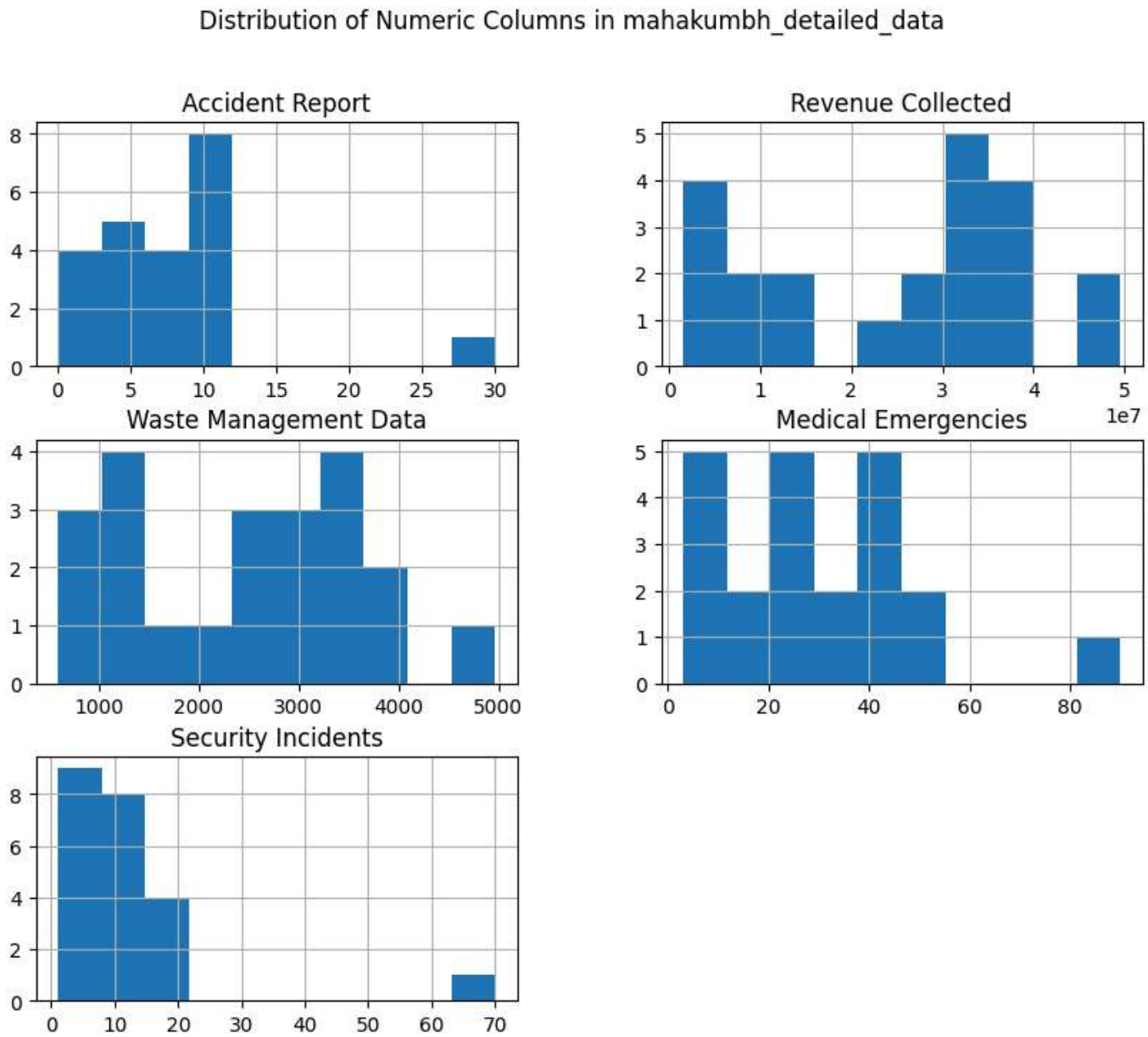
```
2                Neutral           3306721                   1662
3               Positive           3558662                   3847
4                Neutral           1510563                   2528

   Medical Emergencies  Security Incidents         Pilgrimage Type  \
0                   36                  12                   Group
1                   28                  11                  Family
2                   22                   4  Religious Organization
3                   47                  20              Individual
4                   23                   4                   Group

        Popular Spots
0      Triveni Sangam
1  Kumbh Mela Grounds
2      Hanuman Mandir
3         Akshaya Vat
4      Triveni Sangam

Columns in mahakumbh_detailed_data: Index(['Date', 'No of Visitors', 'Total Male', 'Total Female',
       'Total Children', 'Total Saint', 'Mode of Transport',
       'Duration of Stay', 'Accommodation Type', 'Water Quality', 'Event Type',
       'Crowd Density', 'Accident Report', 'Social Media Sentiment',
       'Revenue Collected', 'Waste Management Data', 'Medical Emergencies',
       'Security Incidents', 'Pilgrimage Type', 'Popular Spots'],
      dtype='object')


📊 Table for Sheet: Public sentiment
        Date        Popular spot  Positive_Sentiment_Percentage  \
0 2025-01-13      Triveni Sangam                      52.124145
1 2025-01-14  Kumbh Mela Grounds                      75.046930
2 2025-01-15      Hanuman Mandir                      50.477605
3 2025-01-16         Akshaya Vat                      84.337735
4 2025-01-17      Triveni Sangam                      53.973196

   Negative_Sentiment_Percentage
0                      43.723081
1                      39.579832
2                      43.739096
3                      20.493400
4                      43.790738

Columns in Public sentiment: Index(['Date', 'Popular spot', 'Positive_Sentiment_Percentage',
       'Negative_Sentiment_Percentage'],
      dtype='object')


📊 Table for Sheet: Environmetal_impact
        Date water quality  Waste management data
0 2025-01-13      Moderate                   3152
1 2025-01-14      Moderate                   3602
2 2025-01-15          Poor                   1662
3 2025-01-16          Poor                   3847
4 2025-01-17          Good                   2528

Columns in Environmetal_impact: Index(['Date', 'water quality', 'Waste management data'], dtype='obj
```

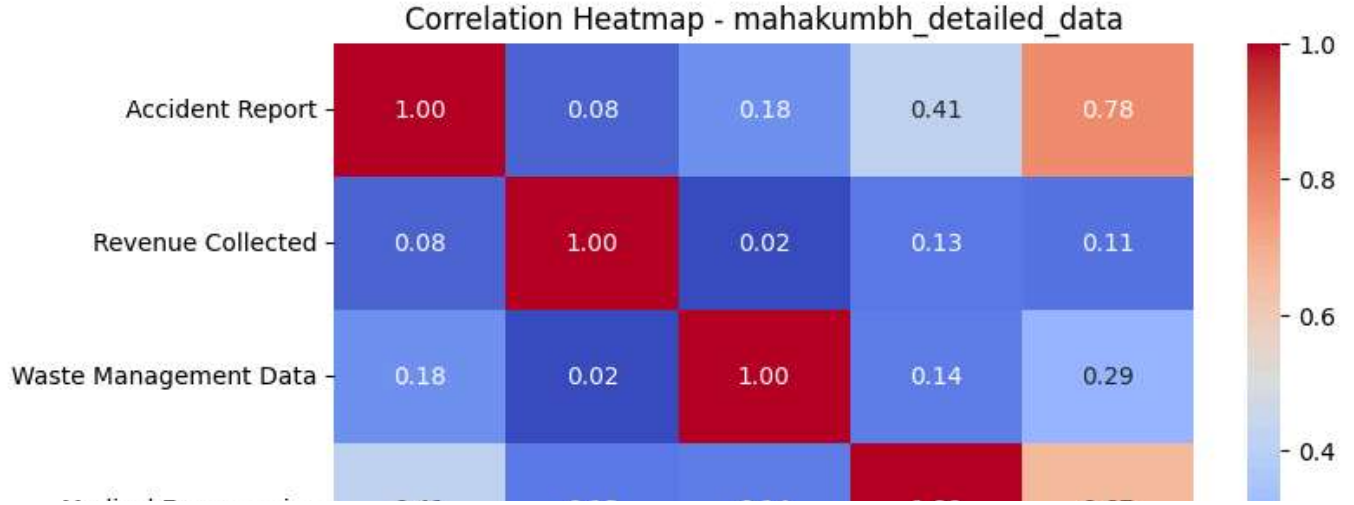
```
# Dictionary to store DataFrames for each sheet
dfs = {sheet: pd.read_excel(xls, sheet_name=sheet) for sheet in sheet_name}
```

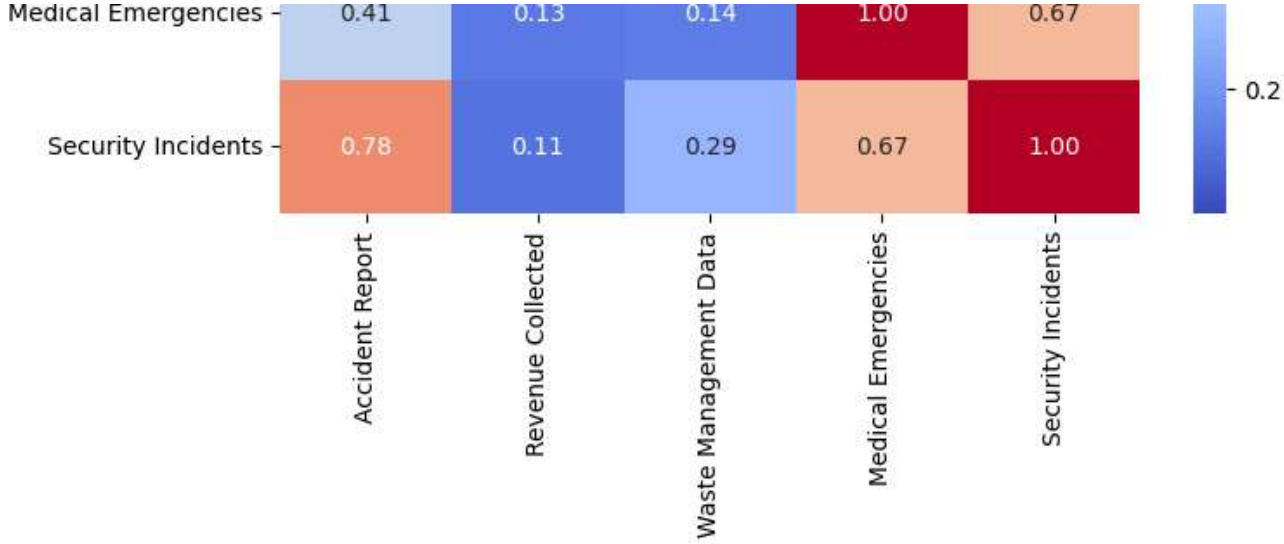
```
# EDA on all sheets
for sheet, df in dfs.items():
    print(f"\n📌 Sheet: {sheet}")
    print(df.info())  # Column types, missing values
    print(df.describe())  # Summary statistics
    print(f"Missing Values:\n{df.isnull().sum()}\n")

    # Visualization (modify based on available columns)
    numeric_cols = df.select_dtypes(include=['number']).columns
    if not numeric_cols.empty:
        df[numeric_cols].hist(figsize=(10, 8))
        plt.suptitle(f"Distribution of Numeric Columns in {sheet}")
        plt.show()

    # Correlation Heatmap
    if len(numeric_cols) > 1:
        plt.figure(figsize=(8, 5))
        sns.heatmap(df[numeric_cols].corr(), annot=True, cmap="coolwarm", fmt=".2f")
        plt.title(f"Correlation Heatmap - {sheet}")
        plt.show()
```

```
📌 Sheet: mahakumbh_detailed_data
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22 entries, 0 to 21
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Date                    22 non-null     datetime64[ns]
 1   No of Visitors          22 non-null     object
 2   Total Male              22 non-null     object
 3   Total Female            22 non-null     object
 4   Total Children          22 non-null     object
 5   Total Saint             22 non-null     object
 6   Mode of Transport       22 non-null     object
 7   Duration of Stay        22 non-null     object
 8   Accommodation Type      22 non-null     object
 9   Water Quality           22 non-null     object
 10  Event Type              22 non-null     object
 11  Crowd Density           22 non-null     object
 12  Accident Report         22 non-null     int64
 13  Social Media Sentiment  22 non-null     object
 14  Revenue Collected       22 non-null     int64
 15  Waste Management Data   22 non-null     int64
 16  Medical Emergencies     22 non-null     int64
 17  Security Incidents      22 non-null     int64
 18  Pilgrimage Type         22 non-null     object
 19  Popular Spots           22 non-null     object
dtypes: datetime64[ns](1), int64(5), object(14)
memory usage: 3.6+ KB
None
                      Date  Accident Report  Revenue Collected  \
count                   22        22.000000       2.200000e+01
mean   2025-01-23 12:00:00         7.090909       2.490617e+07
min    2025-01-13 00:00:00         0.000000       1.510563e+06
25%    2025-01-18 06:00:00         3.000000       8.510798e+06
50%    2025-01-23 12:00:00         7.500000       3.070604e+07
75%    2025-01-28 18:00:00         9.000000       3.574725e+07
max    2025-02-03 00:00:00        30.000000       4.954264e+07
std                    NaN         6.030945       1.528259e+07

       Waste Management Data  Medical Emergencies  Security Incidents
count              22.000000            22.000000           22.000000
mean             2488.545455            30.136364           12.090909
min               588.000000             3.000000            1.000000
25%              1355.250000            13.500000            6.000000
50%              2582.000000            27.500000            8.500000
75%              3521.250000            44.750000           13.500000
max              4964.000000            90.000000           70.000000
std              1184.146618            20.326608           13.856094
Missing Values:
Date                      0
No of Visitors            0
Total Male                0
Total Female              0
Total Children            0
Total Saint               0
Mode of Transport         0
Duration of Stay          0
Accommodation Type        0
Water Quality             0
Event Type                0
Crowd Density             0
Accident Report           0
Social Media Sentiment    0
Revenue Collected         0
```

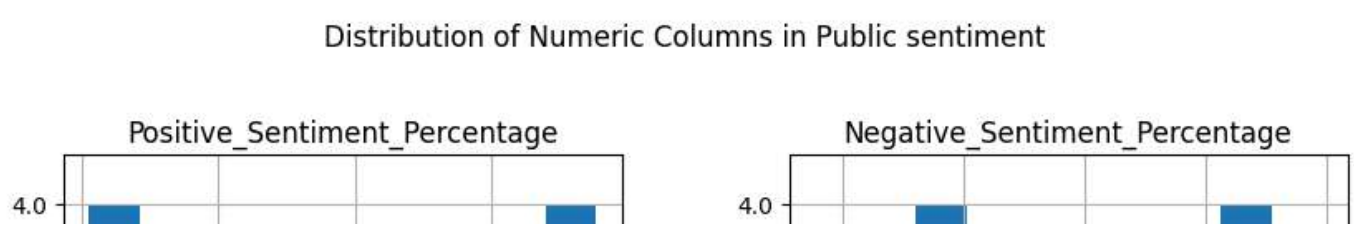
```
Waste Management Data     0
Medical Emergencies       0
Security Incidents        0
Pilgrimage Type           0
Popular Spots             0
dtype: int64
```

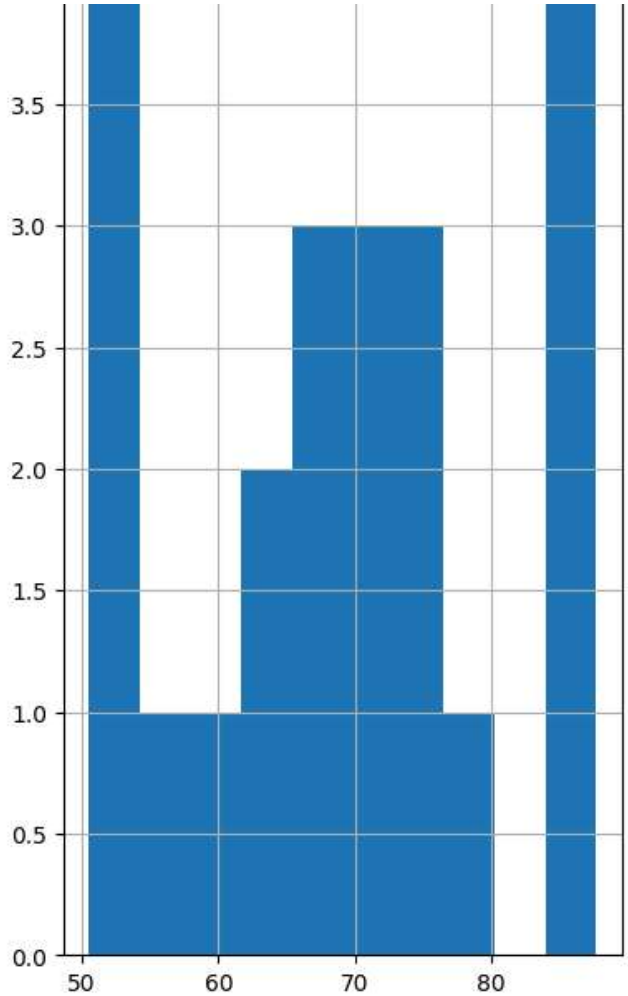


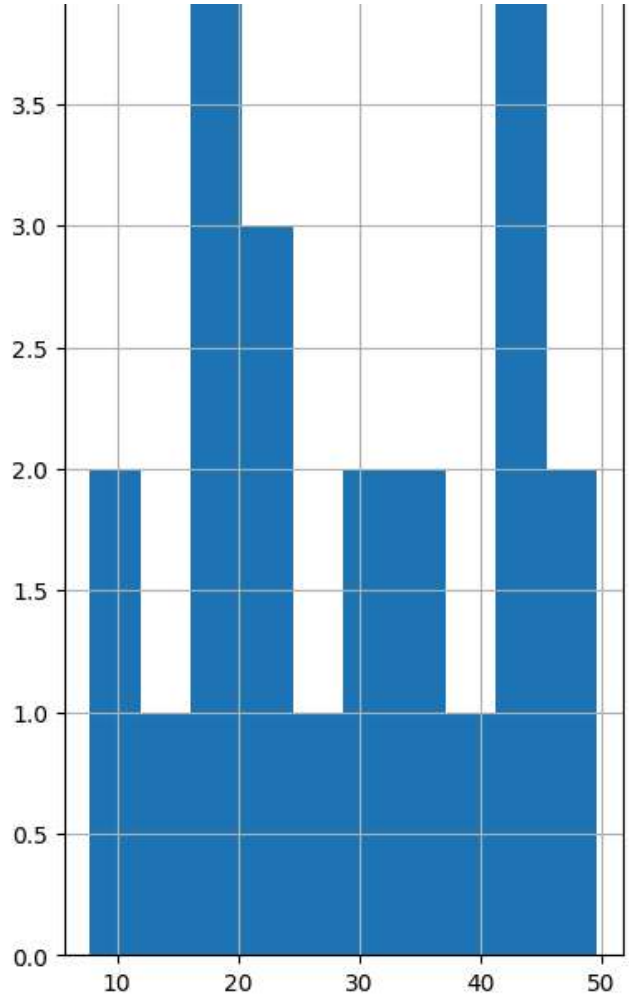


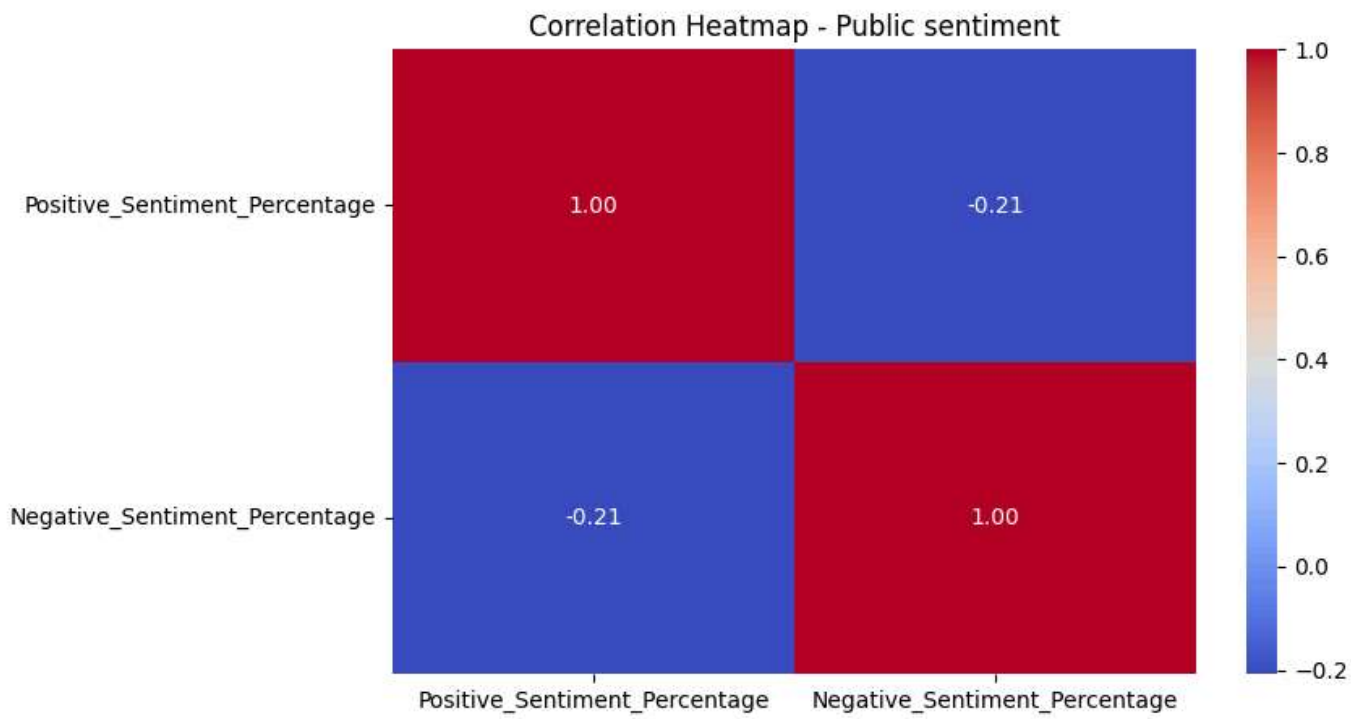



```
📌 Sheet: Public sentiment
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22 entries, 0 to 21
Data columns (total 4 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Date                          22 non-null     datetime64[ns]
 1   Popular spot                  22 non-null     object
 2   Positive_Sentiment_Percentage  22 non-null     float64
 3   Negative_Sentiment_Percentage  22 non-null     float64
dtypes: datetime64[ns](1), float64(2), object(1)
memory usage: 836.0+ bytes
None
                     Date  Positive_Sentiment_Percentage  \
count                  22                      22.000000
mean   2025-01-23 12:00:00                      68.767544
min    2025-01-13 00:00:00                      50.477605
25%    2025-01-18 06:00:00                      61.606212
50%    2025-01-23 12:00:00                      69.392595
75%    2025-01-28 18:00:00                      75.004470
max    2025-02-03 00:00:00                      87.667619
std                    NaN                      11.480091

       Negative_Sentiment_Percentage
count                      22.000000
mean                       29.066050
min                         7.741177
25%                        20.153200
50%                        28.259915
75%                        42.102098
max                        49.699176
std                        12.681239
Missing Values:
Date                            0
Popular spot                    0
Positive_Sentiment_Percentage   0
Negative_Sentiment_Percentage   0
dtype: int64
```



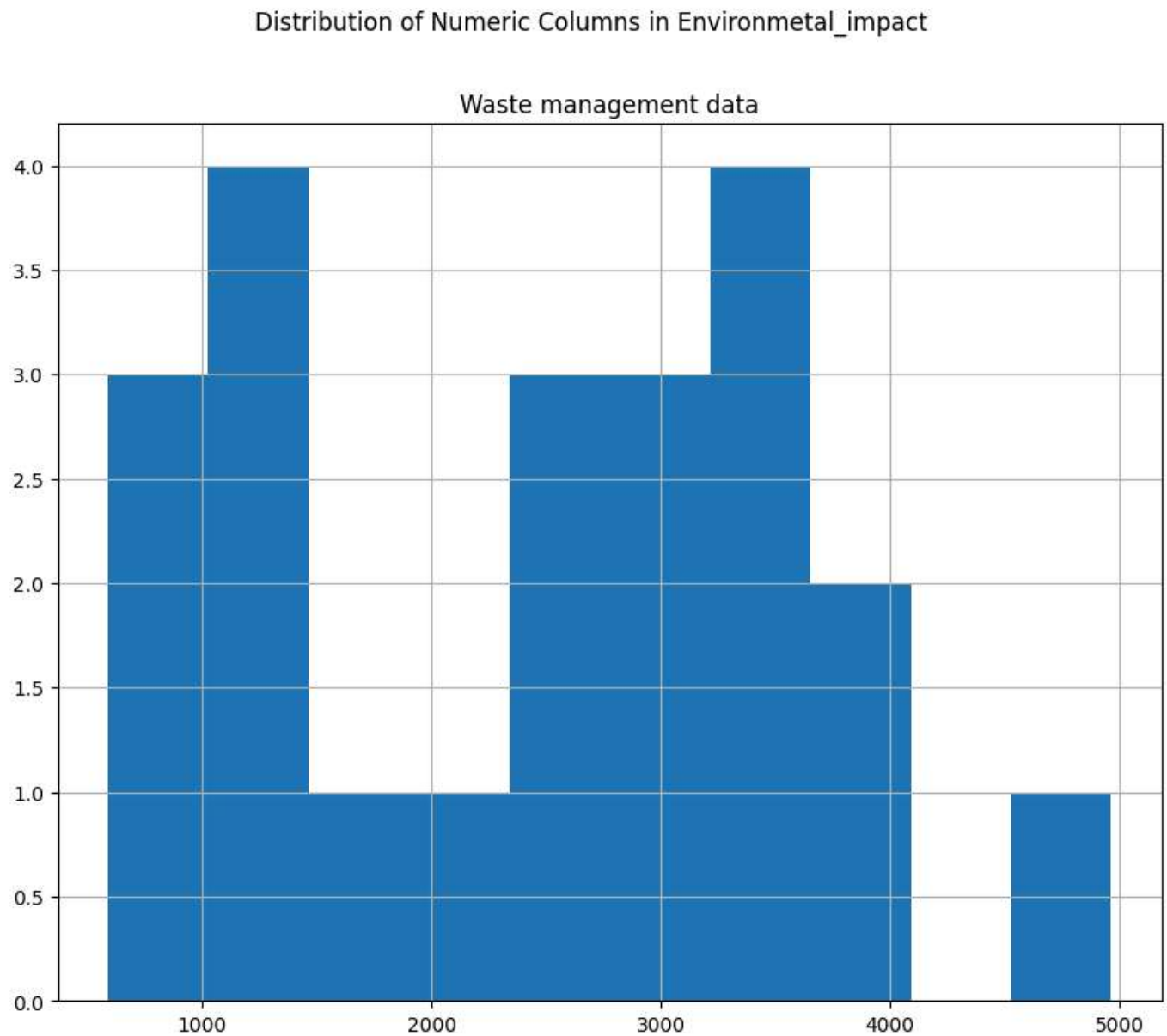







```
📌 Sheet: Environmetal_impact
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22 entries, 0 to 21
Data columns (total 3 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Date                   22 non-null     datetime64[ns]
```

```
 0   Date                    22 non-null     datetime64[ns]
 1   water quality           22 non-null     object
 2   Waste management data   22 non-null     int64
dtypes: datetime64[ns](1), int64(1), object(1)
memory usage: 660.0+ bytes
None
                      Date  Waste management data
count                   22              22.000000
mean   2025-01-23 12:00:00            2488.545455
min    2025-01-13 00:00:00             588.000000
25%    2025-01-18 06:00:00            1355.250000
50%    2025-01-23 12:00:00            2582.000000
75%    2025-01-28 18:00:00            3521.250000
max    2025-02-03 00:00:00            4964.000000
std                    NaN            1184.146618
Missing Values:
Date                     0
water quality            0
Waste management data    0
dtype: int64
```



```
import pandas as pd

# Load the Excel file
file_path = "/content/mahakumbh_detailed_data.xlsx"
xls = pd.ExcelFile(file_path)

# Extract all sheet names
sheet_names = xls.sheet_names

# Create a DataFrame to display sheet names
sheet_df = pd.DataFrame({
    "Sheet Number": range(1, len(sheet_names) + 1),
    "Sheet Name": sheet_names
})

# Display the table
print(sheet_df)
```

```
   Sheet Number               Sheet Name
0             1  mahakumbh_detailed_data
1             2         Public sentiment
2             3      Environmetal_impact
```

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the Excel file
file_path = "/content/mahakumbh_detailed_data.xlsx"
df = pd.read_excel(file_path)

# Convert 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'])

# Extract Year (or use directly if year is separate)
df['Year'] = df['Date'].dt.year

# Apply styling
sns.set_style("darkgrid")

# Create subplots
fig, axes = plt.subplots(3, 2, figsize=(18, 15))
fig.suptitle("Mahakumbh 2025 - Prayagraj Insights", fontsize=16, fontweight="bold")

# --- 1. Total Visitors Breakdown ---
# Ensure visitor columns are numeric
visitor_cols = ['Total Male', 'Total Female', 'Total Children', 'Total Saint']
for col in visitor_cols:
    df[col] = pd.to_numeric(df[col], errors='coerce')  # convert to numeric, set errors to NaN

# Calculate total visitors
df['Total Visitors'] = df['Total Male'] + df['Total Female'] + df['Total Children'] + df['Total Saint']

# Group by Year for plotting
grouped = df.groupby('Year')[visitor_cols].sum().reset_index()

# Plotting
ax = axes[0, 0]
```

```
grouped.plot(x='Year', kind='bar', stacked=True, ax=ax, colormap='tab20')
ax.set_title("Visitor Demographics per Year")
ax.set_ylabel("Count")


# --- 2. Revenue Collected Over Years ---
ax = axes[0, 1]
rev_df = df.groupby('Date')['Revenue Collected'].sum().reset_index()
sns.lineplot(data=rev_df, x='Date', y='Revenue Collected', ax=ax, color="black")
ax.set_title("Total Revenue Collected Over Date")
ax.set_ylabel("Revenue (in ₹)")

# --- 3. Waste vs Water Quality ---
ax = axes[1, 0]
sns.scatterplot(data=df, x='Waste Management Data', y='Water Quality', ax=ax, color='brown')
ax.set_title("Waste Management vs Water Quality")
ax.set_xlabel("Waste (Tons)")
ax.set_ylabel("Water Quality Index")

# --- 4. Sentiment vs Security Incidents ---
ax = axes[1, 1]
sns.scatterplot(data=df, x='Security Incidents', y='Social Media Sentiment', ax=ax, color='purple')
ax.set_title("Security Incidents vs Social Media Sentiment")
ax.set_xlabel("Security Incidents")
ax.set_ylabel("Sentiment Score (%)")

# --- 5. Medical Emergencies vs Crowd Density ---
ax = axes[2, 0]
sns.scatterplot(data=df, x='Crowd Density', y='Medical Emergencies', ax=ax, color='red')
ax.set_title("Crowd Density vs Medical Emergencies")
ax.set_xlabel("Crowd Density Index")
ax.set_ylabel("Medical Cases")

# --- 6. Popular Spot Mentions Over Time (if possible) ---
if 'Popular Spots' in df.columns:
    ax = axes[2, 1]
    df['Popular Spots'] = df['Popular Spots'].astype(str)  # ensure it's string
    top_spots = df['Popular Spots'].value_counts().nlargest(5).index
    df_filtered = df[df['Popular Spots'].isin(top_spots)]
    sns.countplot(data=df_filtered, x='Popular Spots', order=top_spots, ax=ax, palette='Set2')
    ax.set_title("Top 5 Popular Spots Mentions")
    ax.set_ylabel("Mentions")
else:
    axes[2, 1].axis('off')
    axes[2, 1].text(0.5, 0.5, 'No Popular Spots Data Available', horizontalalignment='center', verticalalign

# Final layout
plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.show()
```