```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
import warnings
warnings.filterwarnings('ignore')
```

```
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
pd.set_option('display.max_rows', 100)
```

```
plt.style.use('seaborn-v0_8-darkgrid')
sns.set_palette("husl")
```

```
df = pd.read_csv("/content/marketing_dataset.csv")
```

```


print("Load the dataset using Pandas and display:")

print("\nFirst 10 rows:")
print(df.head(10))
```

```
Load the dataset using Pandas and display:

First 10 rows:
         Date     Adset      URL  Platform  Spend  Visitors  Leads  \
0  2025-09-15   Adset-3  URL-378  WhatsApp   7999     67934   2299
1  2025-09-15   Adset-9  URL-301  Facebook   8988      9551   1882
2  2025-09-15   Adset-6  URL-246  LinkedIn   2760     27887   1966
3  2025-09-15   Adset-4  URL-400  Facebook  16961     11326   2042
4  2025-09-15  Adset-10  URL-116  Facebook  16905     96596   1369
5  2025-09-15  Adset-11  URL-327      Meta   5101     19381   1872
6  2025-09-15  Adset-13  URL-437  LinkedIn   2054      9277   1054
7  2025-09-15   Adset-9  URL-457  LinkedIn  11727      5335   1295
8  2025-09-15   Adset-7  URL-457  Facebook  13032     66438   2761
9  2025-09-15   Adset-8  URL-339    Google   2625     91416   2686
```

```
   SiteVisits  Closure            Project
0          54        4    Sunshine Meadows
1          77       10    Sunshine Meadows
2          23       24           ASBL Loft
3          37       13           ASBL Palm
4          59        0    Riverfront Homes
5          61        4           ASBL Loft
6          52        6  GreenNest Residency
7          57        9       Skyline Towers
8          42       21  GreenNest Residency
9          11        8       Skyline Towers
```

```
print(f"\nShape (rows & columns): {df.shape}")
print(f"Rows: {df.shape[0]}, Columns: {df.shape[1]}")
```

```

Shape (rows & columns): (2000, 10)
Rows: 2000, Columns: 10
```

```
print("\nSummary statistics:")
print(df.describe())
```

```

Summary statistics:
             Spend      Visitors        Leads   SiteVisits      Closure
count   2000.000000   2000.000000  2000.000000  2000.000000  2000.000000
mean   10450.328000  51888.399500  2015.453500    48.977500    14.152000
std     5541.002137  27559.982671   573.566141    28.357275     8.519224
min     1003.000000   5002.000000  1000.000000     0.000000     0.000000
25%     5730.750000  27921.750000  1509.500000    26.000000     7.000000
50%    10288.500000  50926.000000  2024.000000    49.000000    14.000000
75%    15273.250000  76250.750000  2499.000000    73.000000    21.000000
max    19993.000000  99951.000000  2999.000000    99.000000    29.000000
```

```python
print("Convert the Date column to datetime and sort the dataset by date")

df['Date'] = pd.to_datetime(df['Date'])
df = df.sort_values('Date').reset_index(drop=True)
print("\nDate column converted to datetime and sorted:")
print(df[['Date']].head(10))
print(f"Date range: {df['Date'].min()} to {df['Date'].max()}")
```

```
Convert the Date column to datetime and sort the dataset by date

Date column converted to datetime and sorted:
        Date
0 2025-09-15
1 2025-09-15
2 2025-09-15
3 2025-09-15
4 2025-09-15
5 2025-09-15
6 2025-09-15
7 2025-09-15
8 2025-09-15
9 2025-09-15
Date range: 2025-09-15 00:00:00 to 2025-11-14 00:00:00
```

```python
print("List all unique values")



print(f"\nUnique Platforms ({df['Platform'].nunique()}):")
print(df['Platform'].unique())

print(f"\nUnique Projects ({df['Project'].nunique()}):")
print(df['Project'].unique())

print(f"\nUnique URLs ({df['URL'].nunique()}):")
print(f"Total unique URLs: {df['URL'].nunique()}")
print("First 10 URLs:", df['URL'].unique()[:10])

print(f"\nUnique Adsets ({df['Adset'].nunique()}):")
print(df['Adset'].unique())
```

```
List all unique values

Unique Platforms (7):
['WhatsApp' 'Instagram' 'Meta' 'YouTube' 'Facebook' 'Google' 'LinkedIn']

Unique Projects (10):
['Sunshine Meadows' 'GreenNest Residency' 'ASBL Loft' 'ASBL Lakeside'
 'Elite Enclave' 'Skyline Towers' 'Riverfront Homes' 'ASBL Palm'
 'ASBL Spire' 'ASBL Spectra']

Unique URLs (20):
Total unique URLs: 20
First 10 URLs: ['URL-378' 'URL-412' 'URL-366' 'URL-327' 'URL-437' 'URL-164' 'URL-116'
 'URL-339' 'URL-432' 'URL-172']

Unique Adsets (14):
['Adset-3' 'Adset-5' 'Adset-2' 'Adset-9' 'Adset-14' 'Adset-12' 'Adset-1'
 'Adset-13' 'Adset-10' 'Adset-7' 'Adset-4' 'Adset-6' 'Adset-8' 'Adset-11']
```

```python
print("Filter all rows where Platform = 'Google' and Visitors > 10,000")



filtered_df = df[(df['Platform'] == 'Google') & (df['Visitors'] > 10000)]
print(f"\nFiltered rows count: {len(filtered_df)}")
print("\nFirst 10 filtered rows:")
print(filtered_df.head(10))
```

```
Filter all rows where Platform = 'Google' and Visitors > 10,000

Filtered rows count: 264

First 10 filtered rows:
          Date     Adset      URL Platform  Spend  Visitors  Leads  \
5   2025-09-15  Adset-12  URL-164   Google   8987     78438   1218
8   2025-09-15  Adset-12  URL-116   Google   4454     97531   2367
10  2025-09-15  Adset-13  URL-339   Google   4890     76165   2832
12  2025-09-15   Adset-3  URL-366   Google   3652     79564   1597
32  2025-09-15   Adset-8  URL-339   Google   2625     91416   2686
37  2025-09-15  Adset-14  URL-377   Google   5302     62543   2967
57  2025-09-16   Adset-4  URL-356   Google   5890     11208   2411
102 2025-09-17  Adset-12  URL-432   Google  10616     14051   1604
106 2025-09-18   Adset-5  URL-432   Google   9910     65031   2694
```

```
110 2025-09-18  Adset-14  URL-378   Google  11063     43067   1361

     SiteVisits  Closure             Project
5            91       17       ASBL Lakeside
8            56        8  GreenNest Residency
10           22       26  GreenNest Residency
12           36       27       ASBL Lakeside
32           11        8       Skyline Towers
37           46        2          ASBL Spire
57           31       16       Elite Enclave
102          46       19       Skyline Towers
106          11       18     Riverfront Homes
110          20        2  GreenNest Residency
```

```python
print("Create CPL = SPEND/LEADS column and show top 5 expensive adsets")



df['CPL'] = df['Spend'] / df['Leads']
df['CPL'] = df['CPL'].replace([np.inf, -np.inf], np.nan)

top_5_expensive = df.nlargest(5, 'CPL')[['Adset', 'Platform', 'Spend', 'Leads', 'CPL']]
print("\nTop 5 expensive adsets based on CPL:")
print(top_5_expensive)
```

```
Create CPL = SPEND/LEADS column and show top 5 expensive adsets

Top 5 expensive adsets based on CPL:
         Adset  Platform  Spend  Leads        CPL
1500  Adset-12   YouTube  19889   1007  19.750745
1298   Adset-4    Google  19696   1020  19.309804
274   Adset-13    Google  19463   1019  19.100098
1236   Adset-5  LinkedIn  19078   1004  19.001992
93    Adset-13  WhatsApp  18736   1033  18.137464
```

```python
print("SECTION 2: INTERMEDIATE DATA ANALYSIS")



print("Group data by Platform and calculate metrics")
```

```
platform_summary = df.groupby('Platform').agg({
    'Spend': 'sum',
    'Visitors': 'sum',
    'Leads': 'sum',
    'Closure': 'sum',
    'CPL': 'mean'
}).round(2)

platform_summary.columns = ['Total Spend', 'Total Visitors', 'Total Leads', 'Total Closure', 'Avg CPL']
print("\nPlatform-wise Summary:")
print(platform_summary)
```

```
SECTION 2: INTERMEDIATE DATA ANALYSIS
Group data by Platform and calculate metrics

Platform-wise Summary:
           Total Spend  Total Visitors  Total Leads  Total Closure  Avg CPL
Platform
Facebook       2936520        14445121       590784           4037     5.43
Google         2883009        13954264       564143           4165     5.60
Instagram      2862269        15198685       568086           3852     5.47
LinkedIn       3131366        15869864       601915           4112     5.65
Meta           2931312        14556931       567164           4194     5.56
WhatsApp       3016225        15251032       573911           3719     5.91
YouTube        3139955        14500902       564904           4225     6.10
```

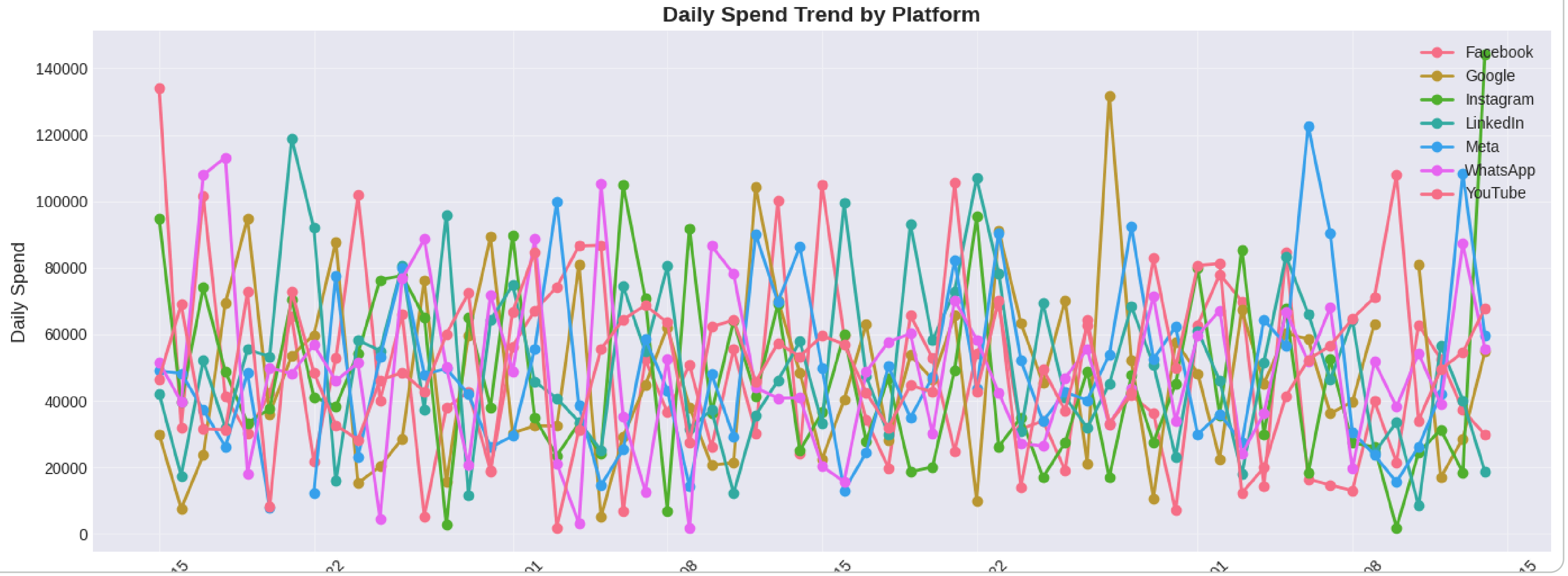
```
print("Find the Project with the highest total SiteVisits")

project_sitevisits = df.groupby('Project')['SiteVisits'].sum().sort_values(ascending=False)
print(f"\nProject with highest SiteVisits: {project_sitevisits.index[0]}")
print(f"Total SiteVisits: {project_sitevisits.iloc[0]}")
print("\nTop 5 Projects by SiteVisits:")

print(project_sitevisits.head())
```

```
Find the Project with the highest total SiteVisits

Project with highest SiteVisits: Skyline Towers
Total SiteVisits: 11266

Top 5 Projects by SiteVisits:
```

```
Project
Skyline Towers         11266
ASBL Lakeside          10908
ASBL Spire             10172
GreenNest Residency     9774
Sunshine Meadows        9612
Name: SiteVisits, dtype: int64
```

```
print(" Calculate daily Spend trend and plot")


daily_spend = df.groupby(['Date', 'Platform'])['Spend'].sum().reset_index()
pivot_spend = daily_spend.pivot(index='Date', columns='Platform', values='Spend')

plt.figure(figsize=(14, 6))
for platform in pivot_spend.columns:
    plt.plot(pivot_spend.index, pivot_spend[platform], marker='o', label=platform, linewidth=2)
plt.xlabel('Date', fontsize=12)
plt.ylabel('Daily Spend', fontsize=12)
plt.title('Daily Spend Trend by Platform', fontsize=14, fontweight='bold')
plt.legend()
plt.grid(True, alpha=0.3)
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig('Q8_daily_spend_trend.png', dpi=300, bbox_inches='tight')
plt.show()
plt.close()
```

```
Calculate daily Spend trend and plot
```



```python
print("Calculate conversion funnel metrics")


df['Lead_Conversion_Rate'] = (df['Leads'] / df['Visitors']).replace([np.inf, -np.inf], 0) * 100
df['SiteVisit_Conversion_Rate'] = (df['SiteVisits'] / df['Leads']).replace([np.inf, -np.inf], 0) * 100
df['Closure_Conversion_Rate'] = (df['Closure'] / df['SiteVisits']).replace([np.inf, -np.inf], 0) * 100

print("\nConversion funnel metrics added. Sample data:")
print(df[['Date', 'Platform', 'Visitors', 'Leads', 'SiteVisits', 'Closure',
          'Lead_Conversion_Rate', 'SiteVisit_Conversion_Rate', 'Closure_Conversion_Rate']].head(10))

print("\nAverage Conversion Rates:")
print(f"Lead Conversion Rate: {df['Lead_Conversion_Rate'].mean():.2f}%")
print(f"SiteVisit Conversion Rate: {df['SiteVisit_Conversion_Rate'].mean():.2f}%")
print(f"Closure Conversion Rate: {df['Closure_Conversion_Rate'].mean():.2f}%")
```

```
Calculate conversion funnel metrics

Conversion funnel metrics added. Sample data:
```

```
         Date   Platform  Visitors  Leads  SiteVisits  Closure  \
0 2025-09-15   WhatsApp     67934   2299          54        4   
1 2025-09-15  Instagram     94132   2844          94       17   
2 2025-09-15       Meta     99584   1296          69        0   
3 2025-09-15    YouTube      5914   2405          42       10   
4 2025-09-15   Facebook     16166   2135          29       10   
5 2025-09-15     Google     78438   1218          91       17   
6 2025-09-15  Instagram     17332   2961          20        7   
7 2025-09-15    YouTube     87335   1295          84       26   
8 2025-09-15     Google     97531   2367          56        8   
9 2025-09-15   WhatsApp     10184   1919          86        2   

   Lead_Conversion_Rate  SiteVisit_Conversion_Rate  Closure_Conversion_Rate  
0              3.384167                   2.348847                 7.407407  
1              3.021289                   3.305204                18.085106  
2              1.301414                   5.324074                 0.000000  
3             40.666216                   1.746362                23.809524  
4             13.206730                   1.358314                34.482759  
5              1.552819                   7.471264                18.681319  
6             17.084006                   0.675447                35.000000  
7              1.482796                   6.486486                30.952381  
8              2.426921                   2.365864                14.285714  
9             18.843284                   4.481501                 2.325581  

Average Conversion Rates:
Lead Conversion Rate: 6.47%
SiteVisit Conversion Rate: 2.66%
Closure Conversion Rate: 77.77%
```

```
print("Identify anomalies (>2 standard deviations)")

# z-scores for Visitors and Leads
visitors_mean = df['Visitors'].mean()
visitors_std = df['Visitors'].std()
leads_mean = df['Leads'].mean()
leads_std = df['Leads'].std()

df['Visitors_Zscore'] = (df['Visitors'] - visitors_mean) / visitors_std
df['Leads_Zscore'] = (df['Leads'] - leads_mean) / leads_std

anomalies = df[(abs(df['Visitors_Zscore']) > 2) | (abs(df['Leads_Zscore']) > 2)]
print(f"\nTotal anomalies detected: {len(anomalies)}")
print("\nFirst 10 anomalies:")
print(anomalies[['Date', 'Platform', 'Adset', 'Visitors', 'Leads', 'Visitors_Zscore', 'Leads_Zscore']].head(10))
```

```
Identify anomalies (>2 standard deviations)

Total anomalies detected: 0

First 10 anomalies:
Empty DataFrame
Columns: [Date, Platform, Adset, Visitors, Leads, Visitors_Zscore, Leads_Zscore]
Index: []
```

```
print("SECTION 3: ADVANCED PYTHON ANALYSIS")

print("Best-performing Adset within each Platform (highest Closure)")


idx = df.groupby('Platform')['Closure'].idxmax()
best_adsets = df.loc[idx, ['Platform', 'Adset', 'Closure', 'Leads', 'Spend']]
print("\nBest-performing Adset per Platform:")
print(best_adsets)
```

```
SECTION 3: ADVANCED PYTHON ANALYSIS
Best-performing Adset within each Platform (highest Closure)

Best-performing Adset per Platform:
      Platform     Adset  Closure  Leads  Spend
119   Facebook  Adset-13       29   1829  12818
179     Google   Adset-3       29   1673   6436
122  Instagram   Adset-8       29   2151   8570
462   LinkedIn   Adset-2       29   1946  15298
420       Meta   Adset-5       29   2013   1725
242   WhatsApp   Adset-9       29   2666  15198
135    YouTube   Adset-5       29   1804  17877
```

```
print("Correlation analysis and heatmap")

correlation_cols = ['Spend', 'Visitors', 'Leads', 'SiteVisits', 'Closure']
correlation_matrix = df[correlation_cols].corr()
```

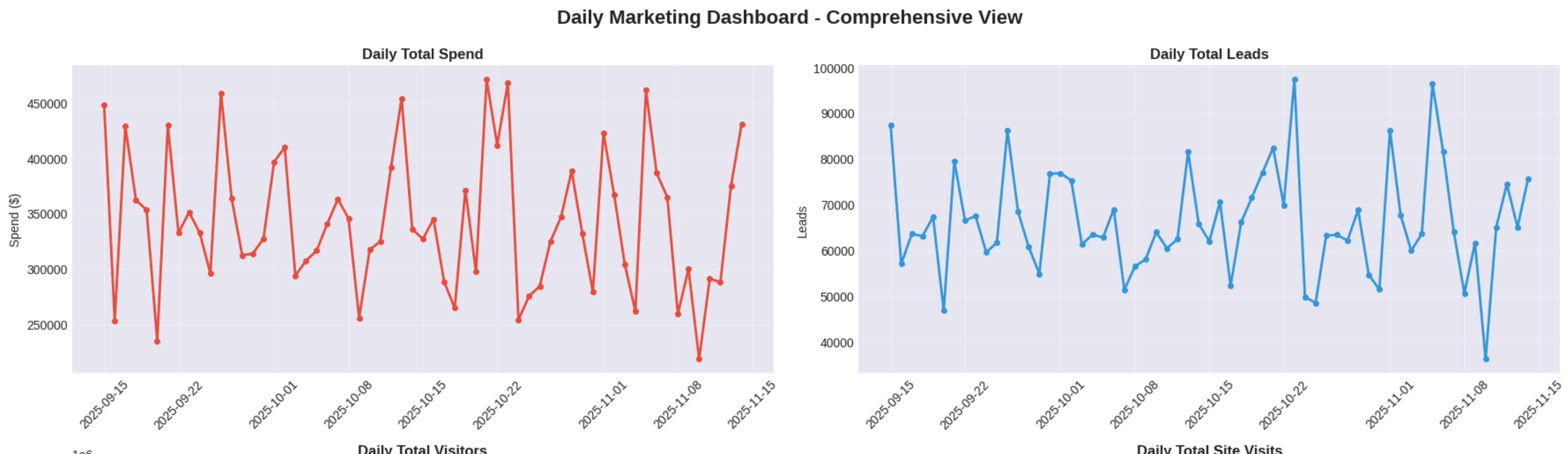
```
print("\nCorrelation Matrix:")
print(correlation_matrix)

# Find strongest correlations
corr_pairs = []
for i in range(len(correlation_cols)):
    for j in range(i+1, len(correlation_cols)):
        corr_pairs.append({
            'Variable 1': correlation_cols[i],
            'Variable 2': correlation_cols[j],
            'Correlation': correlation_matrix.iloc[i, j]
        })

corr_df = pd.DataFrame(corr_pairs).sort_values('Correlation', ascending=False)
print("\nStrongest Correlations:")
print(corr_df.head(10))

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', center=0,
            square=True, linewidths=1, cbar_kws={"shrink": 0.8})
plt.title('Correlation Heatmap: Spend, Visitors, Leads, SiteVisits, Closure',
          fontsize=14, fontweight='bold')
plt.tight_layout()
plt.savefig('Q12_correlation_heatmap.png', dpi=300, bbox_inches='tight')
plt.show()
plt.close()
```

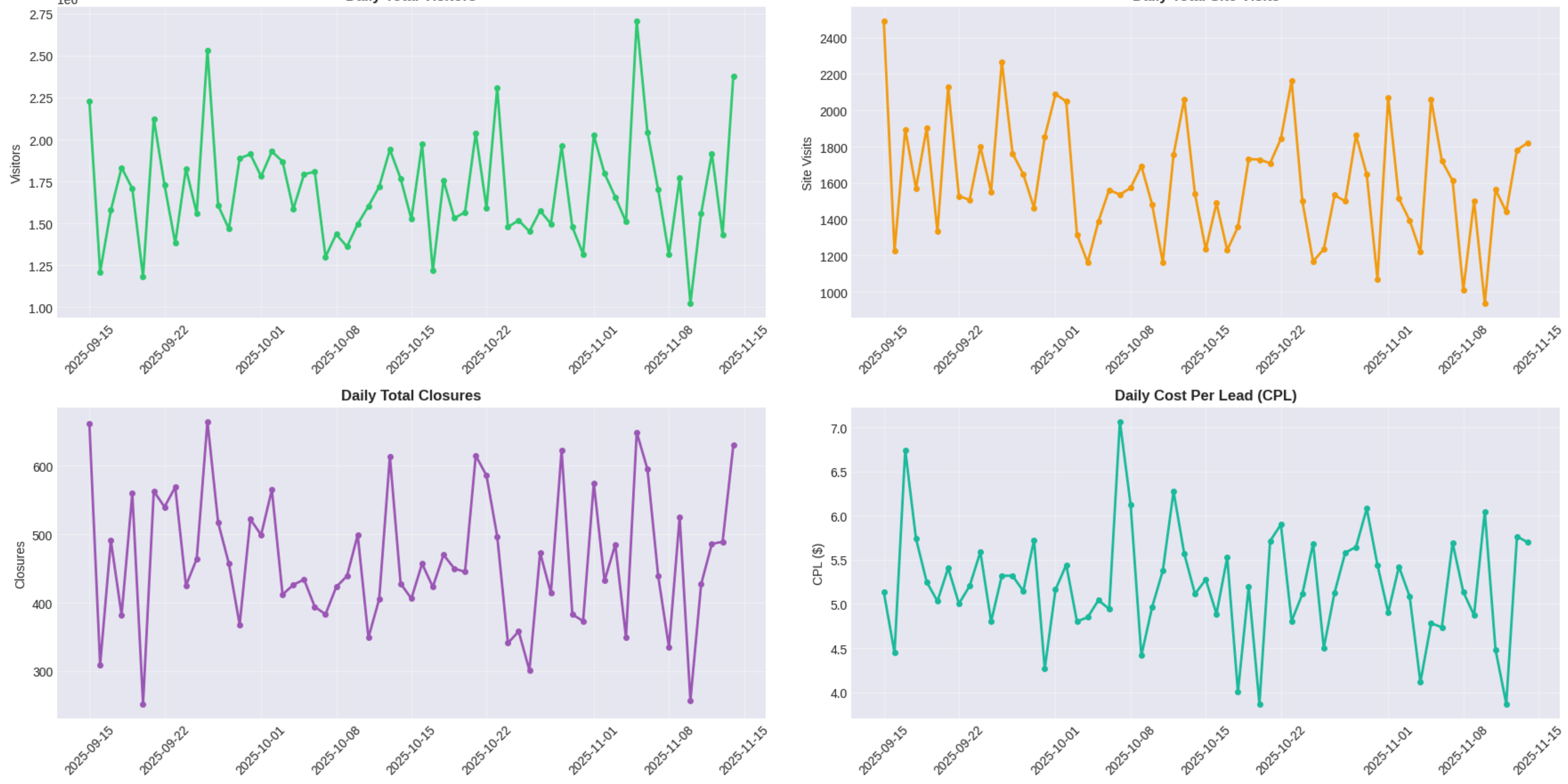
```
Correlation analysis and heatmap

Correlation Matrix:
                Spend  Visitors     Leads  SiteVisits   Closure
Spend        1.000000  0.008267 -0.002095    0.007707 -0.001522
Visitors     0.008267  1.000000 -0.022227    0.016743 -0.005333
```

```
print("Create daily dashboard dataframe")


daily_dashboard = df.groupby('Date').agg({
    'Spend': 'sum',
    'Leads': 'sum',
    'Visitors': 'sum',
    'SiteVisits': 'sum',
    'Closure': 'sum'
}).reset_index()

daily_dashboard.columns = ['Date', 'Total_Spend', 'Total_Leads', 'Total_Visitors',
                           'Total_SiteVisits', 'Total_Closure']

print("\nDaily Dashboard (first 10 rows):")
print(daily_dashboard.head(10))

# Saving to CSV
daily_dashboard.to_csv('daily_dashboard.csv', index=False)
print("\nDaily dashboard saved as 'daily_dashboard.csv'")

print("DAILY DASHBOARD - VISUAL REPRESENTATION")


# Converting Date to datetime
daily_dashboard['Date'] = pd.to_datetime(daily_dashboard['Date'])

# comprehensive dashboard
fig, axes = plt.subplots(3, 2, figsize=(18, 14))
fig.suptitle('Daily Marketing Dashboard - Comprehensive View',
             fontsize=16, fontweight='bold', y=0.995)

# Total Spend Over Time
axes[0, 0].plot(daily_dashboard['Date'], daily_dashboard['Total_Spend'],
                marker='o', linewidth=2, color='#e74c3c', markersize=4)
axes[0, 0].set_title('Daily Total Spend', fontweight='bold')
axes[0, 0].set_ylabel('Spend ($)')
axes[0, 0].grid(True, alpha=0.3)
axes[0, 0].tick_params(axis='x', rotation=45)

# Total Leads Over Time
axes[0, 1].plot(daily_dashboard['Date'], daily_dashboard['Total_Leads'],
                marker='o', linewidth=2, color='#3498db', markersize=4)
axes[0, 1].set_title('Daily Total Leads', fontweight='bold')
axes[0, 1].set_ylabel('Leads')
axes[0, 1].grid(True, alpha=0.3)
axes[0, 1].tick_params(axis='x', rotation=45)

# Total Visitors Over Time
axes[1, 0].plot(daily_dashboard['Date'], daily_dashboard['Total_Visitors'],
                marker='o', linewidth=2, color='#2ecc71', markersize=4)
axes[1, 0].set_title('Daily Total Visitors', fontweight='bold')
axes[1, 0].set_ylabel('Visitors')
axes[1, 0].grid(True, alpha=0.3)
axes[1, 0].tick_params(axis='x', rotation=45)

# Total Site Visits Over Time
axes[1, 1].plot(daily_dashboard['Date'], daily_dashboard['Total_SiteVisits'],
                marker='o', linewidth=2, color='#f39c12', markersize=4)
axes[1, 1].set_title('Daily Total Site Visits', fontweight='bold')
axes[1, 1].set_ylabel('Site Visits')
axes[1, 1].grid(True, alpha=0.3)
axes[1, 1].tick_params(axis='x', rotation=45)

# Total Closures Over Time
axes[2, 0].plot(daily_dashboard['Date'], daily_dashboard['Total_Closure'],
                marker='o', linewidth=2, color='#9b59b6', markersize=4)
axes[2, 0].set_title('Daily Total Closures', fontweight='bold')
axes[2, 0].set_ylabel('Closures')
axes[2, 0].grid(True, alpha=0.3)
axes[2, 0].tick_params(axis='x', rotation=45)

# Cost Per Lead Over Time
daily_dashboard['CPL'] = daily_dashboard['Total_Spend'] / daily_dashboard['Total_Leads'].replace(0, 1)
axes[2, 1].plot(daily_dashboard['Date'], daily_dashboard['CPL'],
                marker='o', linewidth=2, color='#1abc9c', markersize=4)
axes[2, 1].set_title('Daily Cost Per Lead (CPL)', fontweight='bold')
axes[2, 1].set_ylabel('CPL ($)')
axes[2, 1].grid(True, alpha=0.3)
axes[2, 1].tick_params(axis='x', rotation=45)

plt.tight_layout()
plt.show()

# SUMMARY STATISTICS
print("\n" + "="*80)
print("DASHBOARD SUMMARY STATISTICS")
print("="*80)

summary_stats = pd.DataFrame({
    'Metric': ['Total Spend', 'Total Leads', 'Total Visitors',
               'Total Site Visits', 'Total Closures', 'Avg Cost Per Lead'],
    'Total': [
        f"${daily_dashboard['Total_Spend'].sum():,.2f}",
        f"{daily_dashboard['Total_Leads'].sum():,.0f}",
        f"{daily_dashboard['Total_Visitors'].sum():,.0f}",
        f"{daily_dashboard['Total_SiteVisits'].sum():,.0f}",
        f"{daily_dashboard['Total_Closure'].sum():,.0f}",
        f"${daily_dashboard['Total_Spend'].sum() / daily_dashboard['Total_Leads'].sum():,.2f}"
    ],
    'Daily Average': [
        f"${daily_dashboard['Total_Spend'].mean():,.2f}",
        f"{daily_dashboard['Total_Leads'].mean():,.2f}",
        f"{daily_dashboard['Total_Visitors'].mean():,.2f}",
        f"{daily_dashboard['Total_SiteVisits'].mean():,.2f}",
        f"{daily_dashboard['Total_Closure'].mean():,.2f}",
        f"${daily_dashboard['CPL'].mean():,.2f}"
    ]
})

print(summary_stats.to_string(index=False))

print("CORRELATION ANALYSIS")

plt.figure(figsize=(10, 8))
correlation_matrix = daily_dashboard[['Total_Spend', 'Total_Leads', 'Total_Visitors',
                                       'Total_SiteVisits', 'Total_Closure']].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', center=0,
            fmt='.2f', square=True, linewidths=1)
plt.title('Correlation Matrix - Daily Metrics', fontweight='bold', fontsize=14)
plt.tight_layout()
plt.show()


print("COMBINED METRICS VIEW")


fig, ax1 = plt.subplots(figsize=(16, 6))

# spend on primary y-axis
color = 'tab:red'
ax1.set_xlabel('Date', fontweight='bold')
ax1.set_ylabel('Total Spend ($)', color=color, fontweight='bold')
ax1.plot(daily_dashboard['Date'], daily_dashboard['Total_Spend'],
         color=color, marker='o', linewidth=2, label='Spend', markersize=4)
ax1.tick_params(axis='y', labelcolor=color)
ax1.tick_params(axis='x', rotation=45)
ax1.grid(True, alpha=0.3)

# secondary y-axis for leads and closures
ax2 = ax1.twinx()
color = 'tab:blue'
ax2.set_ylabel('Leads & Closures', color=color, fontweight='bold')
ax2.plot(daily_dashboard['Date'], daily_dashboard['Total_Leads'],
         color='tab:blue', marker='s', linewidth=2, label='Leads', markersize=4)
ax2.plot(daily_dashboard['Date'], daily_dashboard['Total_Closure'],
         color='tab:green', marker='^', linewidth=2, label='Closures', markersize=4)
ax2.tick_params(axis='y', labelcolor=color)

# legends
lines1, labels1 = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()
ax1.legend(lines1 + lines2, labels1 + labels2, loc='upper left')

plt.title('Daily Spend vs Leads & Closures', fontweight='bold', fontsize=14)
fig.tight_layout()
plt.show()
```

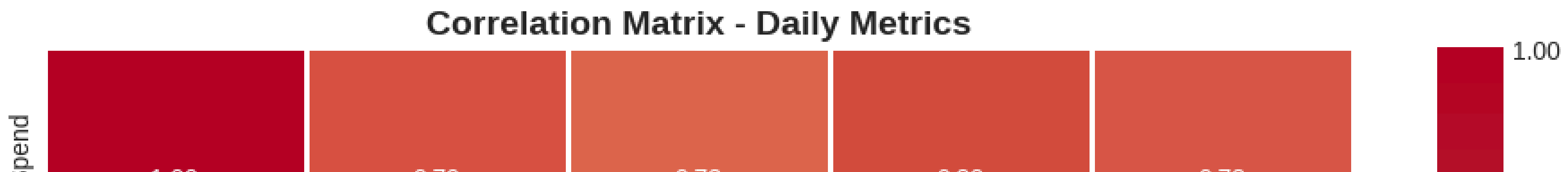
```
    print("\n" + "="*80)
    print("Dashboard visualization complete!")
    print("="*80)
```

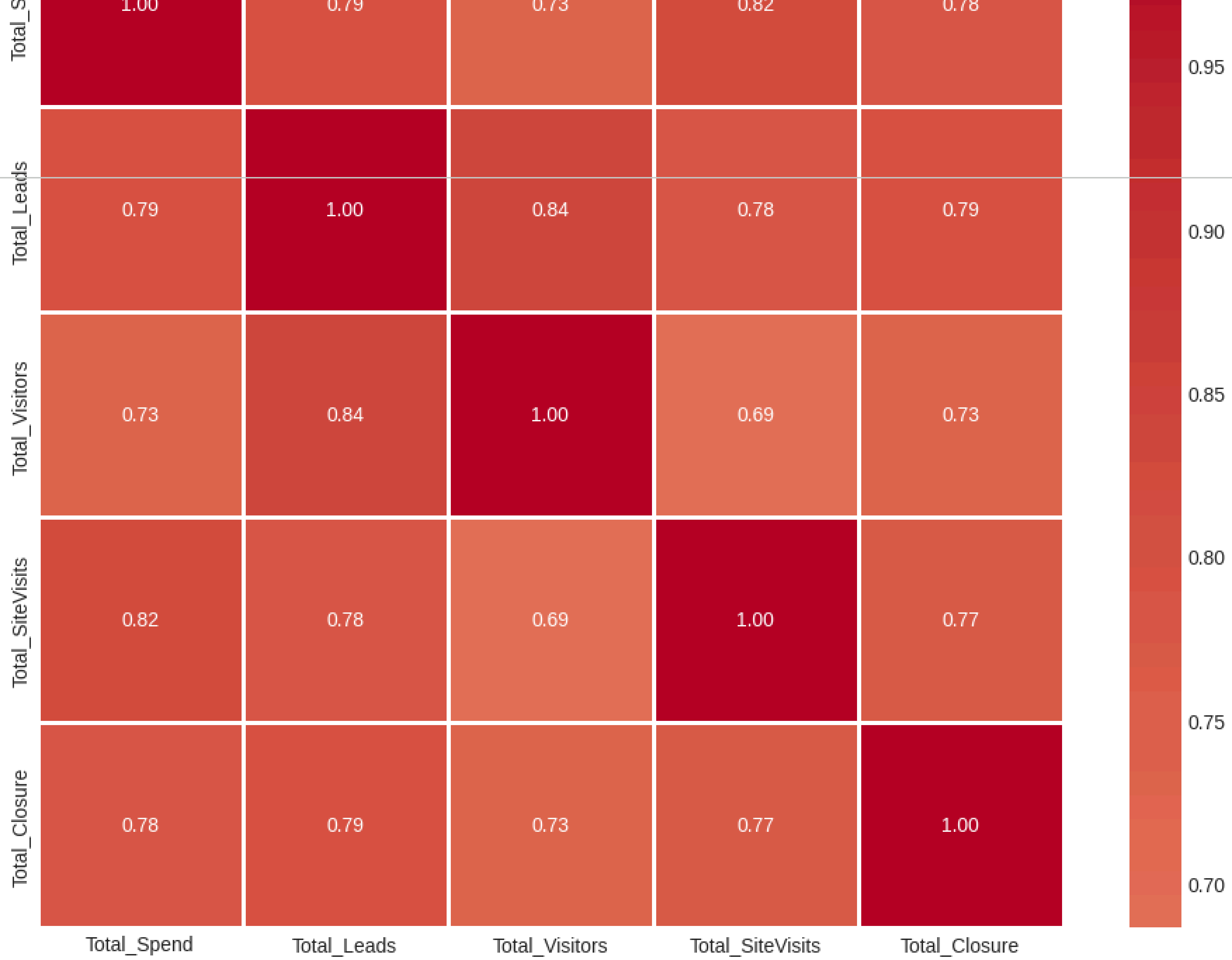
```
9 2025-09-24       332442        59487         1826923              1799

   Total_Closure
0            662
1            309
2            491
3            382
4            560
5            252
6            563
7            540
8            569
9            425

Daily dashboard saved as 'daily_dashboard.csv'
DAILY DASHBOARD - VISUAL REPRESENTATION
```



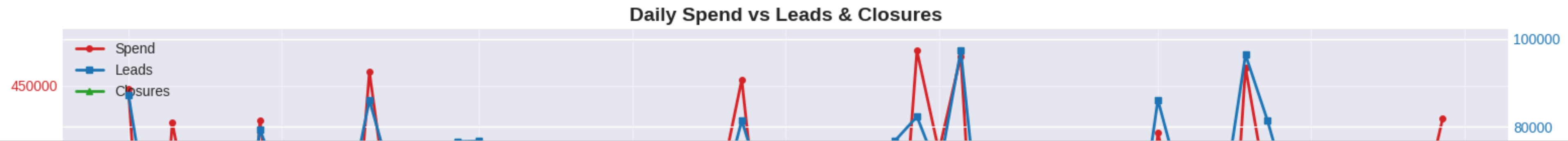


```
================================================================================
DASHBOARD SUMMARY STATISTICS
================================================================================
          Metric           Total  Daily Average
     Total Spend  $20,900,656.00    $342,633.70
     Total Leads       4,030,907      66,080.44
  Total Visitors     103,776,799   1,701,259.00
Total Site Visits         97,955       1,605.82
  Total Closures          28,304         464.00
Avg Cost Per Lead          $5.19          $5.22
CORRELATION ANALYSIS
```





COMBINED METRICS VIEW

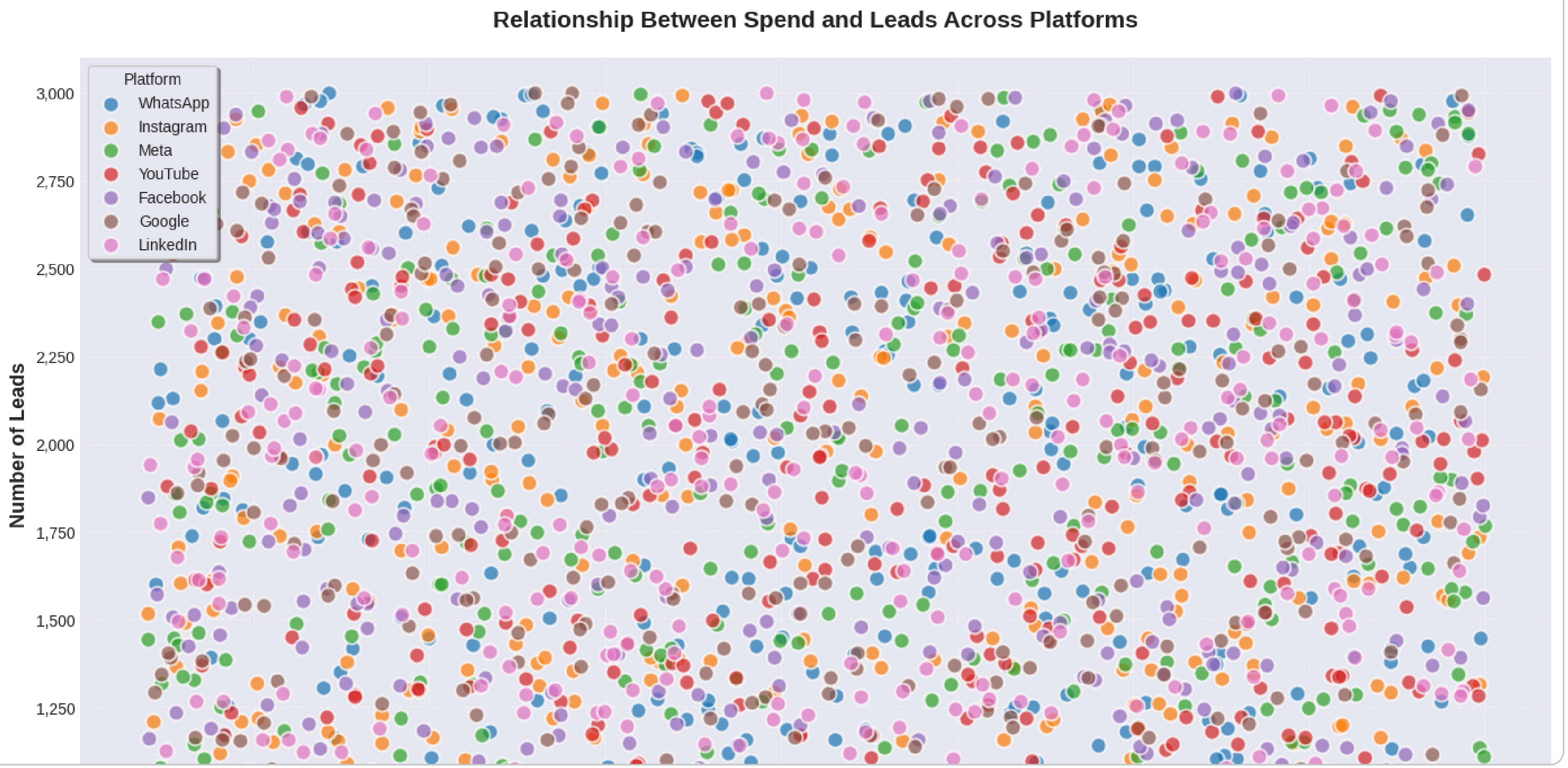


```
print("Funnel summary for each Project")

project_funnel = df.groupby('Project').agg({
    'Visitors': 'sum',
    'Leads': 'sum',
    'SiteVisits': 'sum',
    'Closure': 'sum'
}).reset_index()

project_funnel['Visitor_to_Lead_%'] = (project_funnel['Leads'] / project_funnel['Visitors'] * 100).round(2)
project_funnel['Lead_to_SiteVisit_%'] = (project_funnel['SiteVisits'] / project_funnel['Leads'] * 100).round(2)
project_funnel['SiteVisit_to_Closure_%'] = (project_funnel['Closure'] / project_funnel['SiteVisits'] * 100).round(2)
project_funnel['Overall_Conversion_%'] = (project_funnel['Closure'] / project_funnel['Visitors'] * 100).round(2)

print("\nProject Funnel Summary:")
print(project_funnel.to_string())
```

```
Funnel summary for each Project

Project Funnel Summary:
              Project  Visitors   Leads  SiteVisits  Closure  Visitor_to_Lead_%  Lead_to_SiteVisit_%  SiteVisit_to_Closure_%  Ove
0       ASBL Lakeside  11508590  434182       10908     3066               3.77                 2.51                   28.11
1           ASBL Loft  10449957  361803        9461     2501               3.46                 2.61                   26.43
2           ASBL Palm   9243689  392933        9052     2840               4.25                 2.30                   31.37
3        ASBL Spectra   9533411  380669        9480     2722               3.99                 2.49                   28.71
4          ASBL Spire  11533066  435083       10172     3068               3.77                 2.34                   30.16
5       Elite Enclave  10564029  387880        9153     2839               3.67                 2.36                   31.02
6  GreenNest Residency   9975586  393216        9774     3044               3.94                 2.49                   31.14
7     Riverfront Homes   8700737  363999        9077     2402               4.18                 2.49                   26.46
8      Skyline Towers  11815964  473821       11266     2982               4.01                 2.38                   26.47
9    Sunshine Meadows  10451770  407321        9612     2840               3.90                 2.36                   29.55
```

```
print("Scatter plot: Spend vs Leads for all platforms")
plt.figure(figsize=(14, 8))

platforms = df['Platform'].unique()
colors = plt.cm.tab10.colors

for i, platform in enumerate(platforms):
    platform_data = df[df['Platform'] == platform]
    plt.scatter(platform_data['Spend'],
                platform_data['Leads'],
                label=platform,
                alpha=0.7,
                s=100,
                color=colors[i],
                edgecolors='white',
                linewidth=1.5)

plt.xlabel('Spend (₹)', fontsize=13, fontweight='bold')
plt.ylabel('Number of Leads', fontsize=13, fontweight='bold')
plt.title('Relationship Between Spend and Leads Across Platforms',
          fontsize=15, fontweight='bold', pad=20)

plt.legend(title='Platform', fontsize=10, loc='upper left', frameon=True, shadow=True)
plt.grid(True, alpha=0.3, linestyle='--')

from matplotlib.ticker import FuncFormatter, StrMethodFormatter

ax = plt.gca()
ax.xaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))

plt.tight_layout()
plt.savefig('spend_vs_leads_scatter.png', dpi=300, bbox_inches='tight')
plt.show()
plt.close()
```

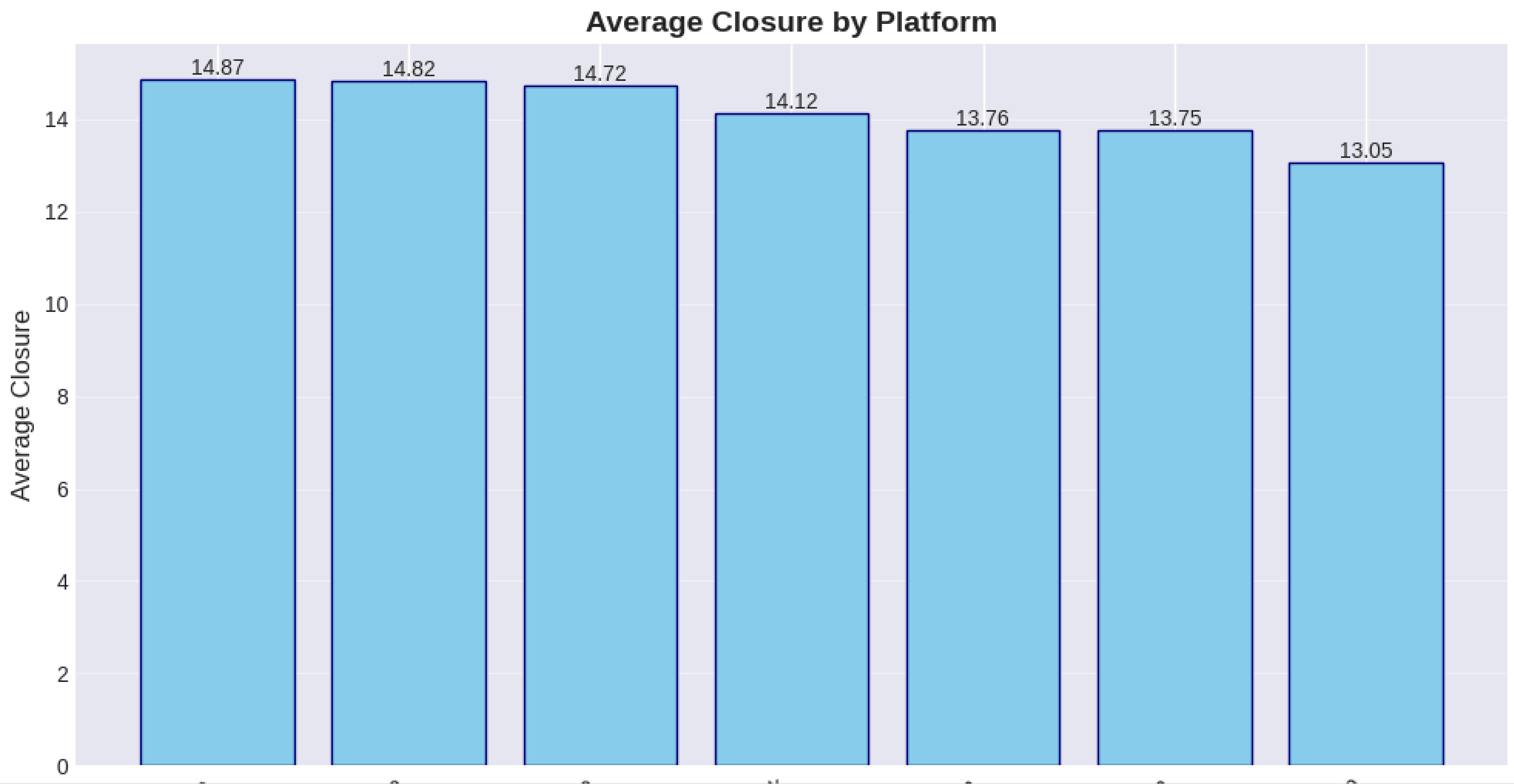
```
Scatter plot: Spend vs Leads for all platforms
```



```
print("Bar chart: Average Closure for each platform")


avg_closure = df.groupby('Platform')['Closure'].mean().sort_values(ascending=False)

plt.figure(figsize=(10, 6))
bars = plt.bar(avg_closure.index, avg_closure.values, color='skyblue', edgecolor='navy')
plt.xlabel('Platform', fontsize=12)
plt.ylabel('Average Closure', fontsize=12)
plt.title('Average Closure by Platform', fontsize=14, fontweight='bold')
plt.xticks(rotation=45, ha='right')
```

```
for bar in bars:
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2., height,
             f'{height:.2f}',
             ha='center', va='bottom', fontsize=10)

plt.grid(True, alpha=0.3, axis='y')
plt.tight_layout()


plt.show()
plt.close()
```

```
Bar chart: Average Closure for each platform
```



```python
print("Line chart: Daily Visitors for 3 platforms")

# top 3 platforms by total visitors
top_3_platforms = df.groupby('Platform')['Visitors'].sum().nlargest(3).index.tolist()

daily_visitors = df[df['Platform'].isin(top_3_platforms)].groupby(['Date', 'Platform'])['Visitors'].sum().reset_index()
pivot_visitors = daily_visitors.pivot(index='Date', columns='Platform', values='Visitors')

plt.figure(figsize=(14, 6))
for platform in pivot_visitors.columns:
    plt.plot(pivot_visitors.index, pivot_visitors[platform],
             marker='o', label=platform, linewidth=2.5, markersize=4)

plt.xlabel('Date', fontsize=12)
plt.ylabel('Daily Visitors', fontsize=12)
plt.title(f'Daily Visitors Comparison: Top 3 Platforms', fontsize=14, fontweight='bold')
plt.legend(fontsize=11)
plt.grid(True, alpha=0.3)
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig('daily_visitors_3_platforms.png', dpi=300, bbox_inches='tight')

plt.show()
plt.close()
```