```
import pandas as pd

try:
    df_business = pd.read_excel("Gies_Business_Jan15-Mar15-Cus_Data.xlsx")
    display(df_business.head())
except FileNotFoundError:
    print("Error: 'Gies_Business_Jan15-Mar15-Cus_Data.xlsx' not found.")
except Exception as e:
    print(f"An error occurred: {e}")
```

	Date	Event name	Page title	Page location	Sessions	Views	New users	Returning users
0	20250115	page_view	MSA Admissions | Gies Master's Degree in Accou...	https://giesbusiness.illinois.edu/msa/admissio...	542	543	0	5
1	20250115	session_start	MSA Admissions | Gies Master's Degree in Accou...	https://giesbusiness.illinois.edu/msa/admissio...	541	0	0	6
2	20250115	first_visit	MSA Admissions | Gies Master's Degree in Accou...	https://giesbusiness.illinois.edu/msa/admissio...	531	0	531	0
3	20250115	page_view	Gies College of Business | University of Illin...	https://giesbusiness.illinois.edu/	407	535	0	122
4	20250115	session_start	Gies College of Business | University of Illin...	https://giesbusiness.illinois.edu/	337	0	0	90

```
try:
    df_business.to_csv("Gies_Business_Jan15-Mar15-Cus_Data.csv", index=False)
    import os
    if os.path.exists("Gies_Business_Jan15-Mar15-Cus_Data.csv"):
        file_size = os.path.getsize("Gies_Business_Jan15-Mar15-Cus_Data.csv")
        if file_size > 0:
            print("Successfully converted to CSV.")
        else:
            print("Error: CSV file created but is empty.")
    else:
        print("Error: CSV file not found.")
except Exception as e:
    print(f"An error occurred: {e}")
```

```
Successfully converted to CSV.
```

```
try:
    df_online = pd.read_excel("Gies_Online_Jan15-Mar15-Cus_Data.xlsx")
    display(df_online.head())
except FileNotFoundError:
    print("Error: 'Gies_Online_Jan15-Mar15-Cus_Data.xlsx' not found.")
except Exception as e:
    print(f"An error occurred: {e}")
```

	Date	Event name	Page title	Page location	Sessions	Views	New users	Returning users
0	20250115	page_view	Gies Online MBA Program Overview | iMBA at Ill...	https://giesonline.illinois.edu/explore-progra...	486	620	0	142
1	20250115	user_engagement	Gies Online MBA Program Overview | iMBA at Ill...	https://giesonline.illinois.edu/explore-progra...	399	0	0	132
2	20250115	session_start	Gies Online MBA Program Overview | iMBA at Ill...	https://giesonline.illinois.edu/explore-progra...	396	0	0	115
3	20250115	session_start	Mock Live Session: Innovating Business Models ...	https://giesonline.illinois.edu/event/2025/01/...	250	0	0	69
4	20250115	page_view	Mock Live Session: Innovating Business Models ...	https://giesonline.illinois.edu/event/2025/01/...	246	269	0	63

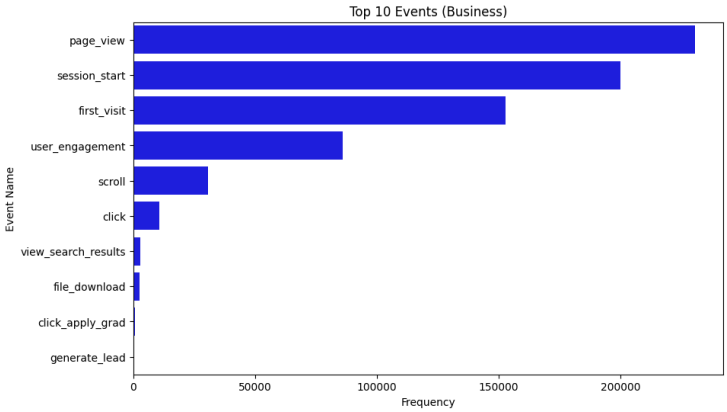
```
try:
    df_online.to_csv("Gies_Online_Jan15-Mar15-Cus_Data.csv", index=False)
    import os
    if os.path.exists("Gies_Online_Jan15-Mar15-Cus_Data.csv"):
        file_size = os.path.getsize("Gies_Online_Jan15-Mar15-Cus_Data.csv")
        if file_size > 0:
            print("Successfully converted to CSV.")
        else:
            print("Error: CSV file created but is empty.")
    else:
        print("Error: CSV file not found.")
except Exception as e:
    print(f"An error occurred: {e}")
```

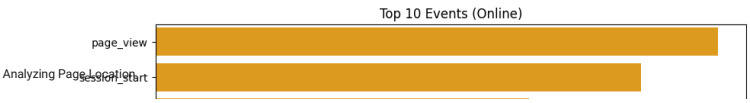
```
Successfully converted to CSV.
```

Top 10 Events in Business vs. Online Journeys

```
# For the business dataset
business_event_counts = df_business['Event name'].value_counts().reset_index()
business_event_counts.columns = ['Event name', 'Frequency']
print(business_event_counts.head(10))  # Top 10 events

# For the online dataset
online_event_counts = df_online['Event name'].value_counts().reset_index()
online_event_counts.columns = ['Event name', 'Frequency']
print(online_event_counts.head(10))   # Top 10 events

import matplotlib.pyplot as plt
import seaborn as sns

# Plot top 10 events for business
top_10_business = business_event_counts.head(10)
plt.figure(figsize=(10,6))
sns.barplot(data=top_10_business, x='Frequency', y='Event name', color='blue')
plt.title('Top 10 Events (Business)')
plt.xlabel('Frequency')
plt.ylabel('Event Name')
plt.show()

# Plot top 10 events for online
top_10_online = online_event_counts.head(10)
plt.figure(figsize=(10,6))
sns.barplot(data=top_10_online, x='Frequency', y='Event name', color='orange')
plt.title('Top 10 Events (Online)')
plt.xlabel('Frequency')
plt.ylabel('Event Name')
plt.show()
```

```
            Event name  Frequency
0            page_view     230491
1        session_start     199955
2          first_visit     152911
3      user_engagement      86033
4               scroll      30610
5                click      10693
6  view_search_results       2875
7        file_download       2513
8     click_apply_grad        729
9        generate_lead        497
        Event name  Frequency
0        page_view     328602
1    session_start     283823
2      first_visit     218048
3  user_engagement      64671
4           scroll      39142
5            click       6628
6       form_start       1161
7      video-start        993
8      click_apply        919
9    generate_lead        914
```

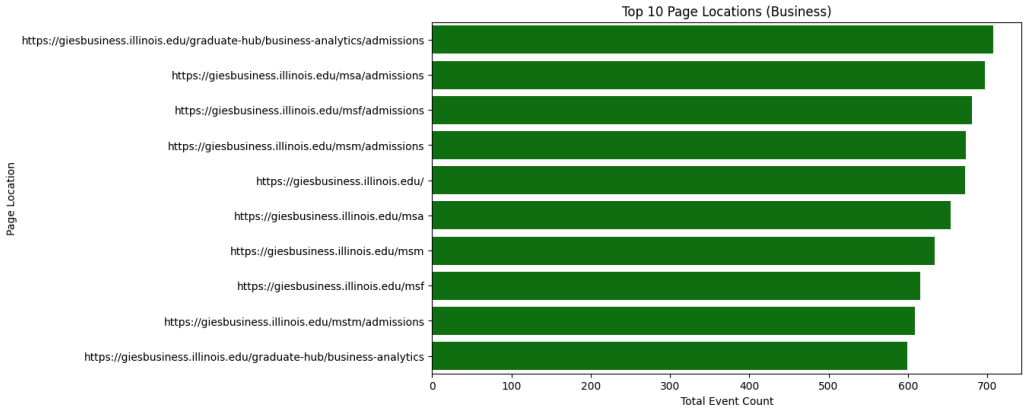




Analyzing Page Location

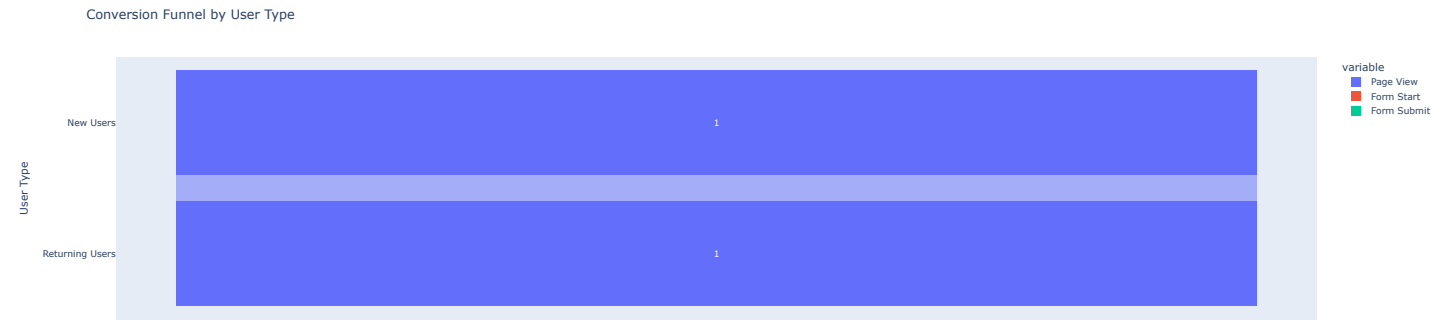
```
# For the business dataset
business_page_counts = (
    df_business.groupby('Page location', as_index=False)['Event name'].count() # Changed 'Event count' to 'Event name' and sum() to count()
    .sort_values(by='Event name', ascending=False) # Changed 'Event count' to 'Event name'
    .rename(columns={'Event name': 'Event count'}) # Rename the column to 'Event count'
)

# For the online dataset
online_page_counts = (
    df_online.groupby('Page location', as_index=False)['Event name'].count() # Changed 'Event count' to 'Event name' and sum() to count()
    .sort_values(by='Event name', ascending=False) # Changed 'Event count' to 'Event name'
    .rename(columns={'Event name': 'Event count'}) # Rename the column to 'Event count'
)

# Display the top rows
print("Top Business Page Locations:")
print(business_page_counts.head(10))

print("\nTop Online Page Locations:")
print(online_page_counts.head(10))

import matplotlib.pyplot as plt
import seaborn as sns

# Take the top 10 for Business
top_10_business_pages = business_page_counts.head(10)
plt.figure(figsize=(10,6))
sns.barplot(data=top_10_business_pages, x='Event count', y='Page location', color='green')
plt.title('Top 10 Page Locations (Business)')
plt.xlabel('Total Event Count')
plt.ylabel('Page Location')
plt.show()

# Take the top 10 for Online
top_10_online_pages = online_page_counts.head(10)
plt.figure(figsize=(10,6))
sns.barplot(data=top_10_online_pages, x='Event count', y='Page location', color='red')
plt.title('Top 10 Page Locations (Online)')
plt.xlabel('Total Event Count')
plt.ylabel('Page Location')
plt.show()

```

```
Top Business Page Locations:
                                        Page location  Event count
31674   https://giesbusiness.illinois.edu/graduate-hub...          708
110701   https://giesbusiness.illinois.edu/msa/admissions          697
123031   https://giesbusiness.illinois.edu/msf/admissions          681
132469   https://giesbusiness.illinois.edu/msm/admissions          673
151                 https://giesbusiness.illinois.edu/          672
110699              https://giesbusiness.illinois.edu/msa          654
132467              https://giesbusiness.illinois.edu/msm          634
123028              https://giesbusiness.illinois.edu/msf          616
152115  https://giesbusiness.illinois.edu/mstm/admissions          609
31673   https://giesbusiness.illinois.edu/graduate-hub...          599

Top Online Page Locations:
                                        Page location  Event count
197642  https://giesonline.illinois.edu/explore-progra...          817
163437  https://giesonline.illinois.edu/explore-progra...          773
134807  https://giesonline.illinois.edu/explore-progra...          679
110212  https://giesonline.illinois.edu/credential/str...          673
40                  https://giesonline.illinois.edu/          643
290647       https://giesonline.illinois.edu/request-info          629
111141  https://giesonline.illinois.edu/explore-progra...          617
287899  https://giesonline.illinois.edu/explore-progra...          603
134808  https://giesonline.illinois.edu/explore-progra...          595
197991  https://giesonline.illinois.edu/explore-progra...          563
```



Funnel Analysis


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

# Define the steps in the funnel
funnel_steps = ['page_view', 'form_start', 'form_submit']

# Filter the data for relevant events
funnel_data = df_business[df_business['Event name'].isin(funnel_steps)]

# Create separate funnels for new and returning users
def create_funnel(data):
    user_funnel = data.groupby('New users')['Event name'].apply(list).reset_index()  # Group by 'new users'
    step_counts = [user_funnel['Event name'].apply(lambda x: funnel_steps[0] in x).sum()]
    for i in range(1, len(funnel_steps)):
        step_counts.append(user_funnel['Event name'].apply(lambda x: funnel_steps[i] in x and funnel_steps[i-1] in x).sum())
    return step_counts

new_user_counts = create_funnel(funnel_data[funnel_data['Returning users'] == 0])  # Filter for new users
returning_user_counts = create_funnel(funnel_data[funnel_data['Returning users'] == 1])  # Filter for returning users

# Create a DataFrame for visualization
funnel_df = pd.DataFrame({
    'User Type': ['New Users', 'Returning Users'],
    'Page View': [new_user_counts[0], returning_user_counts[0]],
    'Form Start': [new_user_counts[1], returning_user_counts[1]],
    'Form Submit': [new_user_counts[2], returning_user_counts[2]]
})

# Visualize the funnels using Plotly
fig = px.funnel(funnel_df, x=['Page View', 'Form Start', 'Form Submit'], y='User Type', title='Conversion Funnel by User Type')
fig.show()
```



Time Series Analysis

```
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose

# 1. Create separate DataFrames for new and returning users
new_users_df = df_business[df_business['New users'] == 1]  # Filter for new users
returning_users_df = df_business[df_business['Returning users'] == 1]  # Filter for returning users

# 2. Calculate daily traffic for new users
daily_new_users = new_users_df.groupby('Date')['New users'].count().reset_index()
daily_new_users.rename(columns={'New users': 'New User Count'}, inplace=True)  # Rename column for clarity

# 3. Calculate daily traffic for returning users
daily_returning_users = returning_users_df.groupby('Date')['Returning users'].count().reset_index()
daily_returning_users.rename(columns={'Returning users': 'Returning User Count'}, inplace=True)  # Rename column
```

```
16
17 # 4. Merge the two DataFrames on 'Date'
18 daily_traffic = pd.merge(daily_new_users, daily_returning_users, on='Date', how='outer')
19 daily_traffic.fillna(0, inplace=True)  # Fill any missing values with 0
20
21 # 5. Perform time series analysis for new users
22 result_new_users = seasonal_decompose(daily_traffic['New User Count'], model='additive', period=7)
23 result_new_users.plot()
24 plt.title('Time Series Decomposition for New Users')
25 plt.show()
26
27 # 6. Perform time series analysis for returning users
28 result_returning_users = seasonal_decompose(daily_traffic['Returning User Count'], model='additive', period=7)
29 result_returning_users.plot()
30 plt.title('Time Series Decomposition for Returning Users')
31 plt.show()
32
```