In [66]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [68]:
```python
df = pd.read_csv("data-export (1).csv")
```

In [70]:
```python
df.head()
```

Out[70]:

# --		Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	
0	Session primary channel group (Default channel…	Date + hour (YYYYMMDDHH)	Users	Sessions	Engaged sessions	Average engagement time per session	
1	Direct	2024041623	237	300	144	47.526666666666700	0.(
2	Organic Social	2024041719	208	267	132	32.09737827715360	0.(
3	Direct	2024041723	188	233	115	39.93991416309010	0.(
4	Organic Social	2024041718	187	256	125	32.16015625	0.(

In [72]:
```python
df.columns = df.iloc[0]
df = df.drop(index = 0).reset_index(drop = True)
df.columns = ["channel group", "DateHour", "Users", "Sessions", "Engaged Session
```

In [74]:
```python
df.head()
```

Out[74]:

	channel group	DateHour	Users	Sessions	Engaged Sessions	Average engagement time per session	Engaged sess per
0	Direct	2024041623	237	300	144	47.526666666666700	0.607594936708
1	Organic Social	2024041719	208	267	132	32.09737827715360	0.634615384615
2	Direct	2024041723	188	233	115	39.93991416309010	0.611702127659
3	Organic Social	2024041718	187	256	125	32.16015625	0.668449197860
4	Organic Social	2024041720	175	221	112	46.918552036199100	

In [76]: `df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3182 entries, 0 to 3181
Data columns (total 10 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   channel group                        3182 non-null   object
 1   DateHour                             3182 non-null   object
 2   Users                                3182 non-null   object
 3   Sessions                             3182 non-null   object
 4   Engaged Sessions                     3182 non-null   object
 5   Average engagement time per session  3182 non-null   object
 6   Engaged sessions per user            3182 non-null   object
 7   Events per session                   3182 non-null   object
 8   Engagement rate                      3182 non-null   object
 9   Event count                          3182 non-null   object
dtypes: object(10)
memory usage: 248.7+ KB
```

cleaning data and data validation for the above data set

In [79]: `df.head()`

Out[79]:

	channel group	DateHour	Users	Sessions	Engaged Sessions	Average engagement time per session	Engaged sess per
0	Direct	2024041623	237	300	144	47.526666666666700	0.607594936708
1	Organic Social	2024041719	208	267	132	32.09737827715360	0.634615384615
2	Direct	2024041723	188	233	115	39.93991416309010	0.611702127659
3	Organic Social	2024041718	187	256	125	32.16015625	0.668449197860
4	Organic Social	2024041720	175	221	112	46.918552036199100	

In [81]: `df["DateHour"] = pd.to_datetime(df["DateHour"], format="%Y%m%d%H", errors='coerc`

In [83]:
```python
numeric_cols = df.columns.drop(["channel group", "DateHour"])
df[numeric_cols] = df[numeric_cols].apply(pd.to_numeric, errors='coerce')
df["Hour"] = df["DateHour"].dt.hour
```

In [85]: `df.head()`

Out[85]:

	channel group	DateHour	Users	Sessions	Engaged Sessions	Average engagement time per session	Engaged sessions per user	Events per session	Eng
0	Direct	2024-04-16 23:00:00	237	300	144	47.526667	0.607595	4.673333	
1	Organic Social	2024-04-17 19:00:00	208	267	132	32.097378	0.634615	4.295880	
2	Direct	2024-04-17 23:00:00	188	233	115	39.939914	0.611702	4.587983	
3	Organic Social	2024-04-17 18:00:00	187	256	125	32.160156	0.668449	4.078125	
4	Organic Social	2024-04-17 20:00:00	175	221	112	46.918552	0.640000	4.529412	

In [87]: `df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3182 entries, 0 to 3181
Data columns (total 11 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   channel group                        3182 non-null   object
 1   DateHour                             3182 non-null   datetime64[ns]
 2   Users                                3182 non-null   int64
 3   Sessions                             3182 non-null   int64
 4   Engaged Sessions                     3182 non-null   int64
 5   Average engagement time per session  3182 non-null   float64
 6   Engaged sessions per user            3182 non-null   float64
 7   Events per session                   3182 non-null   float64
 8   Engagement rate                      3182 non-null   float64
 9   Event count                          3182 non-null   int64
 10  Hour                                 3182 non-null   int32
dtypes: datetime64[ns](1), float64(4), int32(1), int64(4), object(1)
memory usage: 261.2+ KB
```

In [89]: `df.describe()`

Out[89]:

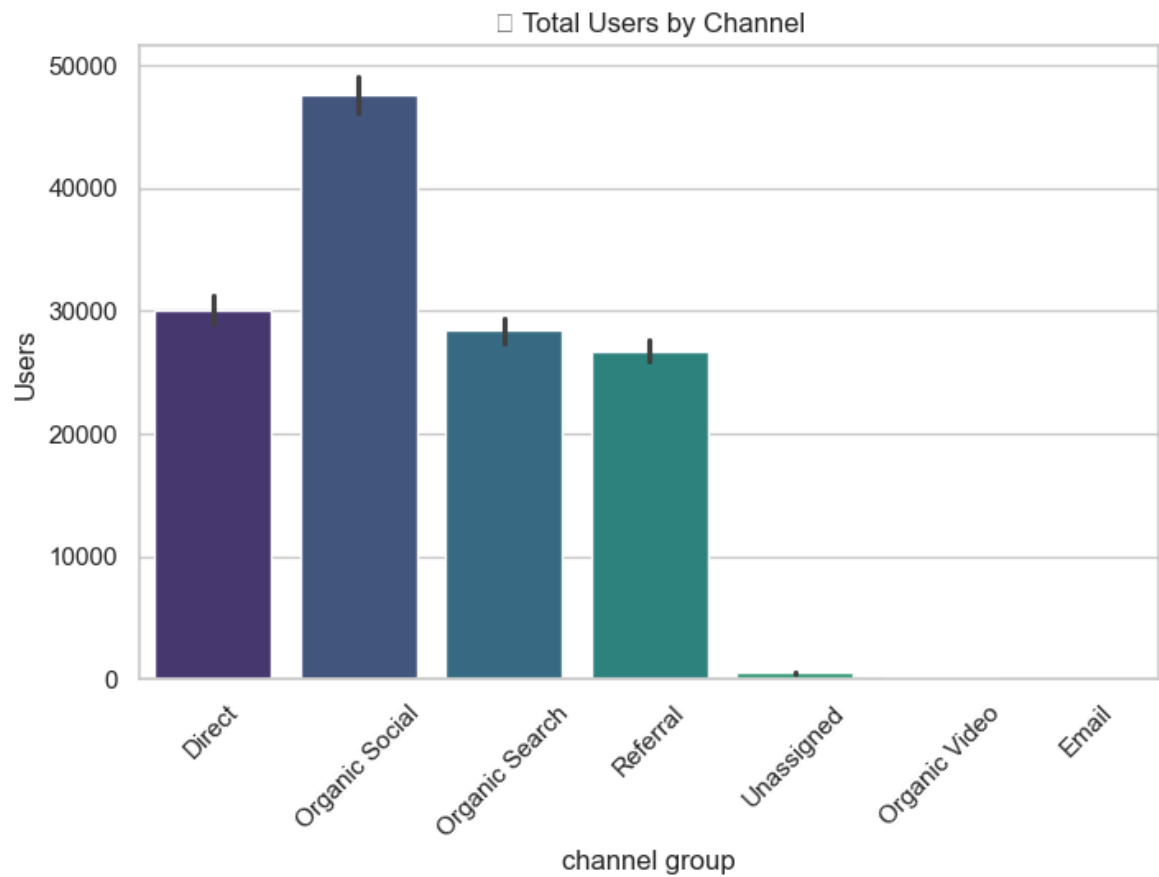|	DateHour	Users	Sessions	Engaged Sessions	Average engagement time per session	Enga sess per
count	3182	3182.000000	3182.000000	3182.000000	3182.000000	3182.00(
mean	2024-04-20 01:17:07.278441216	41.935889	51.192646	28.325581	66.644581	0.60(
min	2024-04-06 00:00:00	0.000000	1.000000	0.000000	0.000000	0.00(
25%	2024-04-13 02:15:00	20.000000	24.000000	13.000000	32.103034	0.56
50%	2024-04-20 02:00:00	42.000000	51.000000	27.000000	49.020202	0.66(
75%	2024-04-26 22:00:00	60.000000	71.000000	41.000000	71.487069	0.75(
max	2024-05-03 23:00:00	237.000000	300.000000	144.000000	4525.000000	2.00(
std	NaN	29.582258	36.919962	20.650569	127.200659	0.26<

sessions and user over time

In [92]:
```python
sns.set(style="whitegrid")
```

In [94]:
```python
plt.figure(figsize=(10,5))
df.groupby("DateHour")[["Sessions","Users"]].sum().plot(ax=plt.gca())
plt.title("Sessions and users over time")
plt.xlabel("DateHour")
plt.ylabel("count")
plt.show()
```

Sessions and users over time
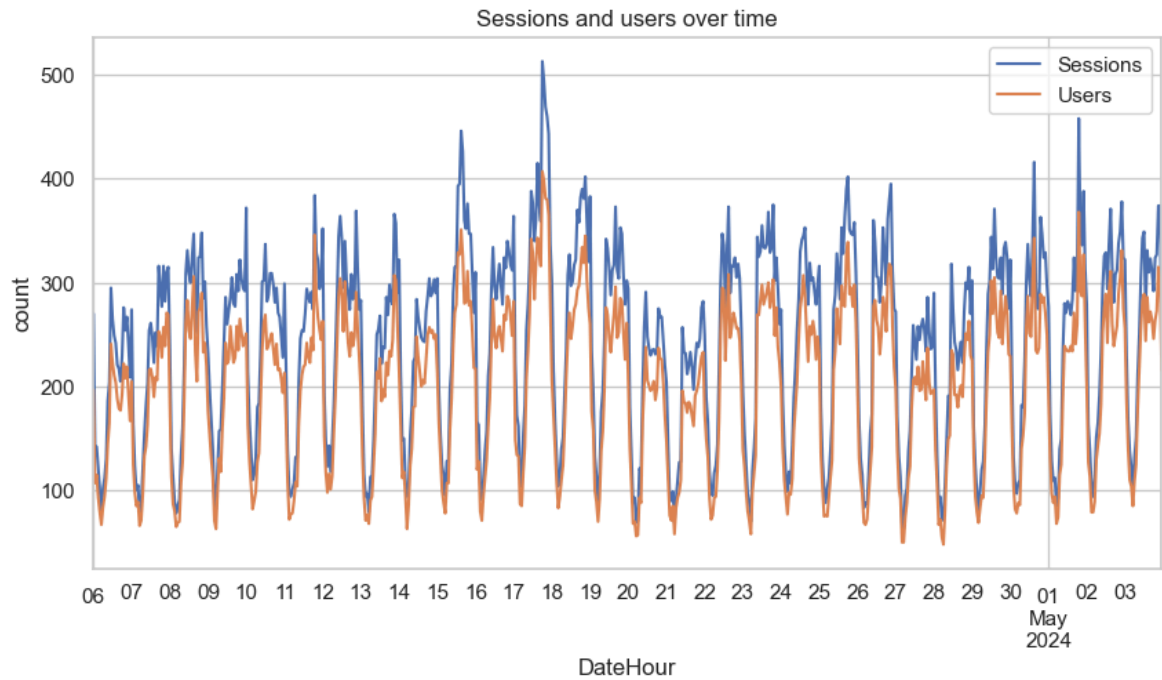
total users by channel

```
In [96]: plt.figure(figsize=(8, 5))
         sns.barplot(data=df, x="channel group", y="Users", estimator=np.sum, palette="vi
         plt.title(" Total Users by Channel"  )
         plt.xticks(rotation=45)
         plt.show()
```

☐ Total Users by Channel


```
In [97]:  df.head()
```

Out[97]:

	channel group	DateHour	Users	Sessions	Engaged Sessions	Average engagement time per session	Engaged sessions per user	Events per session	Eng
0	Direct	2024-04-16 23:00:00	237	300	144	47.526667	0.607595	4.673333	
1	Organic Social	2024-04-17 19:00:00	208	267	132	32.097378	0.634615	4.295880	
2	Direct	2024-04-17 23:00:00	188	233	115	39.939914	0.611702	4.587983	
3	Organic Social	2024-04-17 18:00:00	187	256	125	32.160156	0.668449	4.078125	
4	Organic Social	2024-04-17 20:00:00	175	221	112	46.918552	0.640000	4.529412	

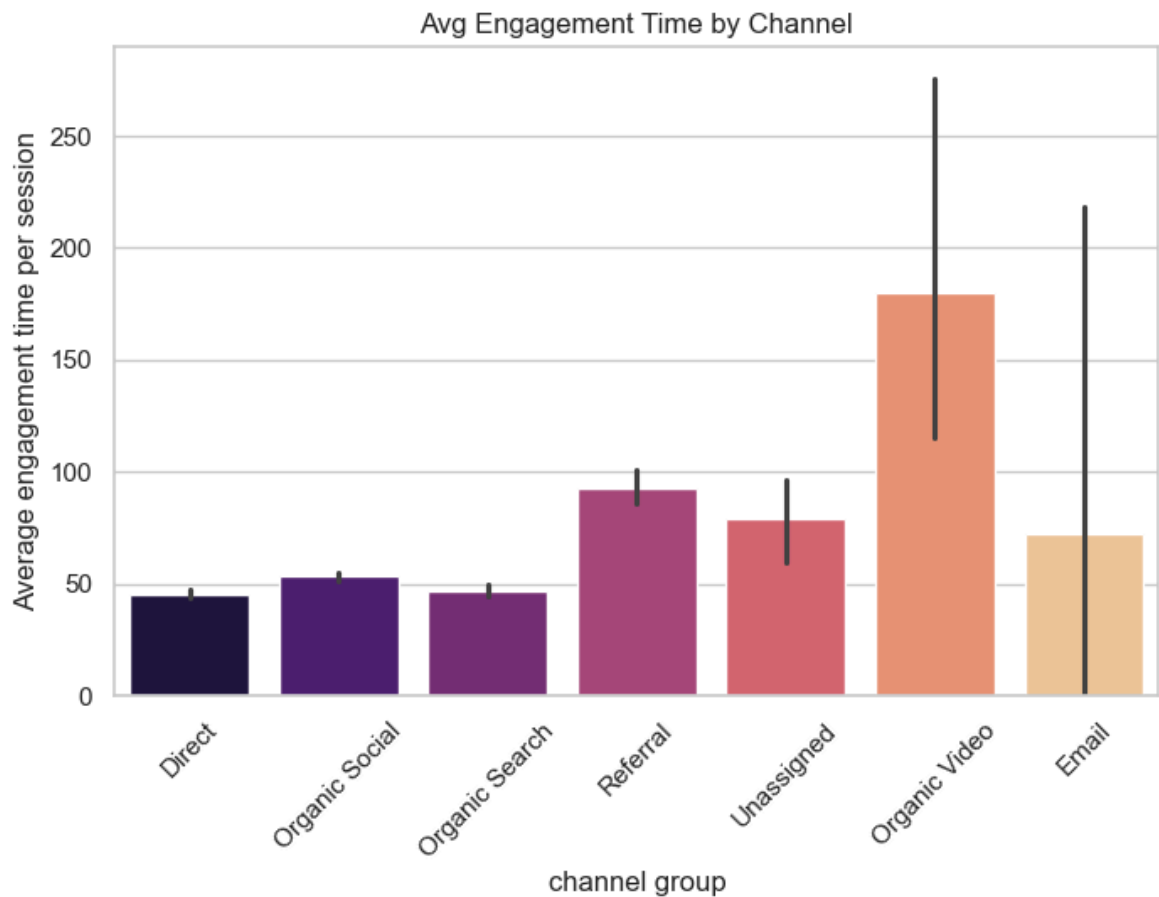# Average engagement time by channel

```
In [100…   plt.figure(figsize=(8, 5))
           sns.barplot(data=df, x="channel group", y="Average engagement time per session",
           plt.title("Avg Engagement Time by Channel")
           plt.xticks(rotation=45)
           plt.show()
```

```
C:\Users\dherya\AppData\Local\Temp\ipykernel_21616\523820305.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.barplot(data=df, x="channel group", y="Average engagement time per sessio
n", estimator=np.mean, palette="magma")
```


Avg Engagement Time by Channel

```
In [107…   df.head()
```

Out[107…]

	channel group	DateHour	Users	Sessions	Engaged Sessions	Average engagement time per session	Engaged sessions per user	Events per session	Eng
0	Direct	2024-04-16 23:00:00	237	300	144	47.526667	0.607595	4.673333	
1	Organic Social	2024-04-17 19:00:00	208	267	132	32.097378	0.634615	4.295880	
2	Direct	2024-04-17 23:00:00	188	233	115	39.939914	0.611702	4.587983	
3	Organic Social	2024-04-17 18:00:00	187	256	125	32.160156	0.668449	4.078125	
4	Organic Social	2024-04-17 20:00:00	175	221	112	46.918552	0.640000	4.529412	

# Engagement Rate Distribution by channel

In [112…]

```python
plt.figure(figsize=(8, 5))
sns.boxplot(data=df, x="channel group", y="Engagement rate", palette="coolwarm")
plt.title("📦 Engagement Rate Distribution by Channel")
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\dherya\AppData\Local\Temp\ipykernel_21616\2490929186.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.boxplot(data=df, x="channel group", y="Engagement rate", palette="coolwar
m")
C:\Users\dherya\anaconda3\Lib\site-packages\IPython\core\pylabtools.py:170: UserWa
rning: Glyph 128230 (\N{PACKAGE}) missing from font(s) Arial.
  fig.canvas.print_figure(bytes_io, **kw)
```

☐ Engagement Rate Distribution by Channel
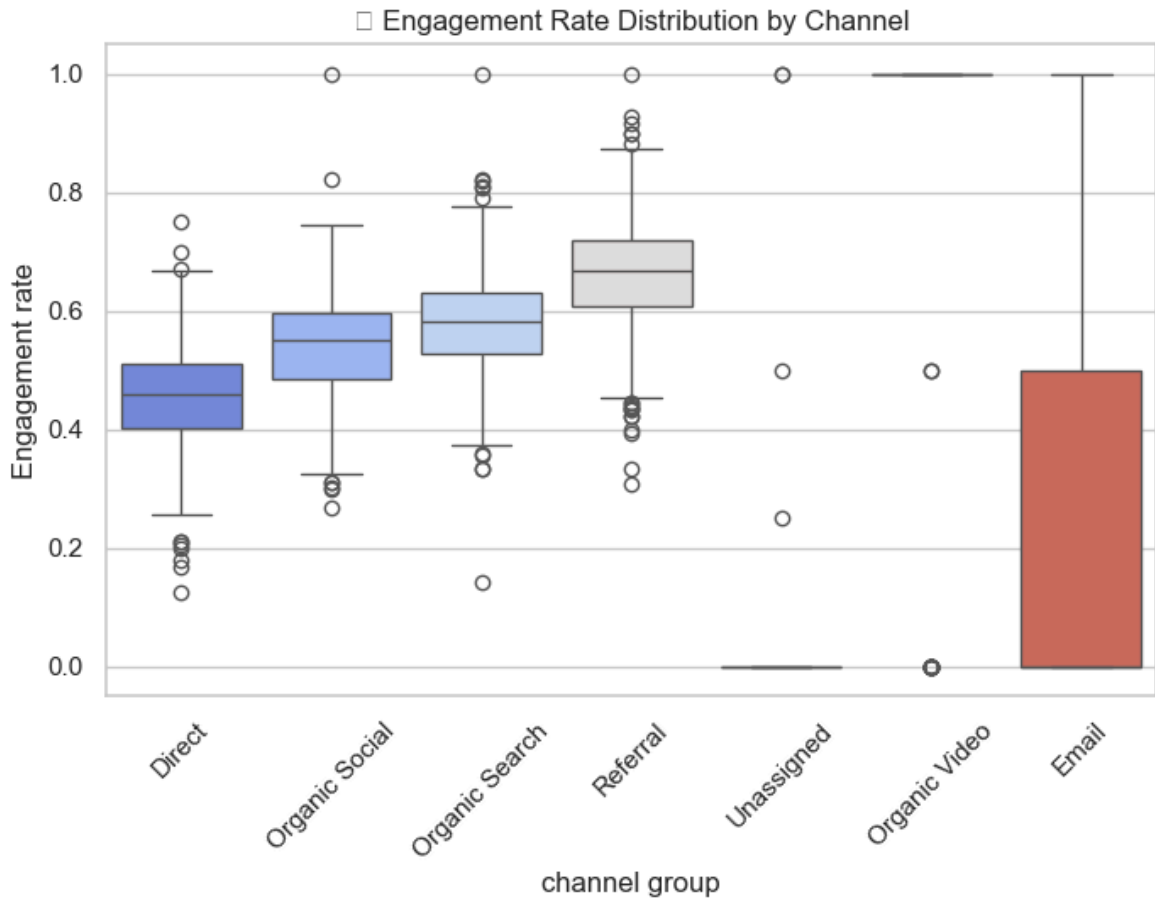
Engaged vs non engaged sessions

In [116...   `df.head()`

Out[116...

|	channel group	DateHour	Users	Sessions	Engaged Sessions	Average engagement time per session	Engaged sessions per user	Events per session	Eng
0	Direct	2024-04-16 23:00:00	237	300	144	47.526667	0.607595	4.673333	
1	Organic Social	2024-04-17 19:00:00	208	267	132	32.097378	0.634615	4.295880	
2	Direct	2024-04-17 23:00:00	188	233	115	39.939914	0.611702	4.587983	
3	Organic Social	2024-04-17 18:00:00	187	256	125	32.160156	0.668449	4.078125	
4	Organic Social	2024-04-17 20:00:00	175	221	112	46.918552	0.640000	4.529412	

In [124...
```python
session_df = df.groupby("channel group")[["Sessions", "Engaged Sessions"]].sum()
session_df["Non-Engaged"] = session_df["Sessions"] - session_df["Engaged Session
session_df_melted = session_df.melt(id_vars="channel group", value_vars=["Engage

plt.figure(figsize=(8, 5))
sns.barplot(data=session_df_melted, x="channel group", y="value", hue="variable"
plt.title(" 🧍  Engaged vs Non-Engaged Sessions")
plt.xticks(rotation=45)
plt.show()
```

C:\Users\dherya\anaconda3\Lib\site-packages\IPython\core\pylabtools.py:170: UserWa
rning: Glyph 129485 (\N{STANDING PERSON}) missing from font(s) Arial.
  fig.canvas.print_figure(bytes_io, **kw)
C:\Users\dherya\anaconda3\Lib\site-packages\IPython\core\pylabtools.py:170: UserWa
rning: Glyph 65039 (\N{VARIATION SELECTOR-16}) missing from font(s) Arial.
  fig.canvas.print_figure(bytes_io, **kw)
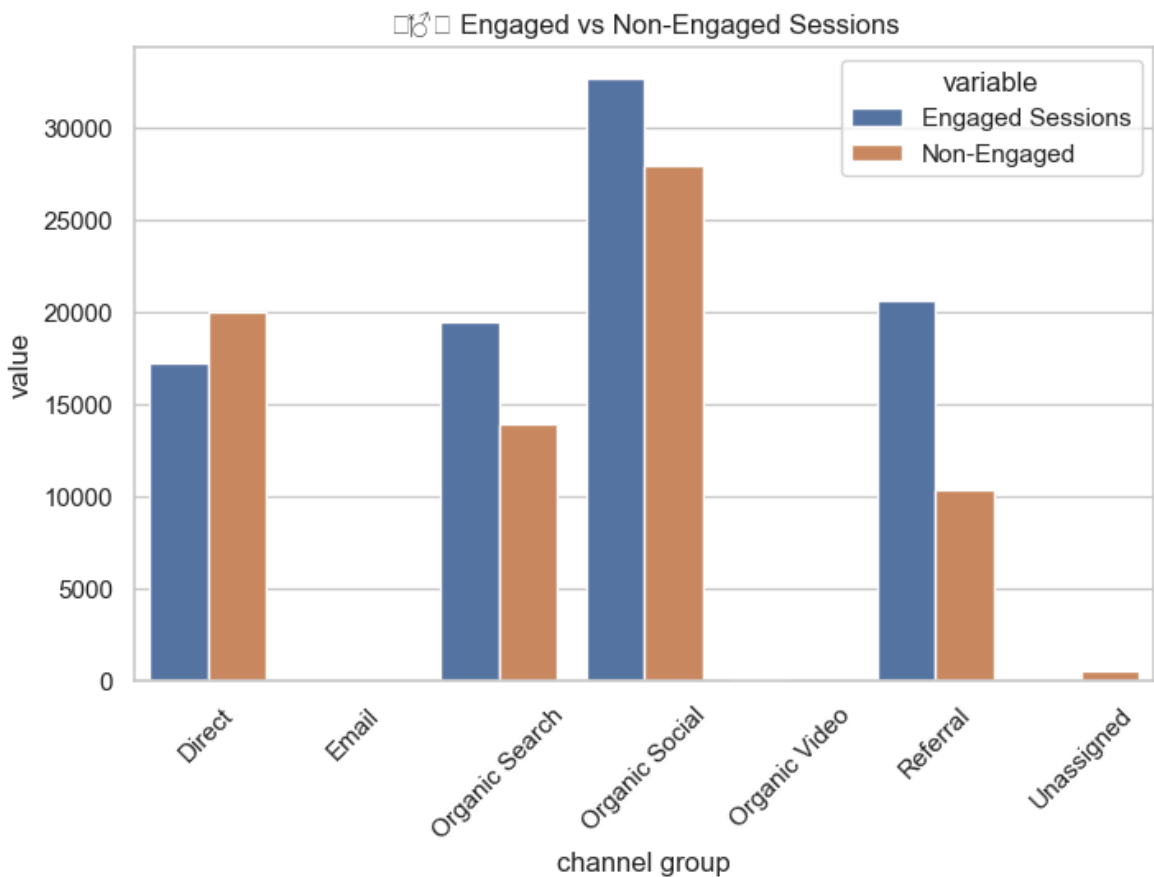
traffic by hour and channel

In [130...
```python
heatmap_data = df.groupby(["Hour", "channel group"])["Sessions"].sum().unstack()

plt.figure(figsize=(12, 6))
sns.heatmap(heatmap_data, cmap="YlGnBu", linewidths=.5, annot=True, fmt='.0f')
plt.title(" ⏰  Traffic by Hour and Channel")
plt.xlabel("Channel Group")
plt.ylabel("Hour of Day")
plt.show()
```

C:\Users\dherya\anaconda3\Lib\site-packages\IPython\core\pylabtools.py:170: UserWa
rning: Glyph 9200 (\N{ALARM CLOCK}) missing from font(s) Arial.
  fig.canvas.print_figure(bytes_io, **kw)

Traffic by Hour and Channel

Hour of Day	Direct	Email	Organic Search	Organic Social	Organic Video	Referral	Unassigned
0	1684	0	1311	3917	6	1204	26
1	1196	0	984	2108	5	923	12
2	887	1	804	1537	2	755	13
3	771	0	606	1249	2	560	11
4	666	1	535	1081	2	495	6
5	679	0	506	951	1	453	8
6	768	0	639	1171	1	565	17
7	889	0	778	1524	2	743	10
8	1078	0	938	1886	4	862	13
9	1347	0	1269	2390	4	1192	19
10	1621	0	1649	2834	9	1648	30
11	1892	0	1839	3069	8	1790	31
12	1881	0	1871	2842	7	1763	34
13	1806	0	1758	2691	5	1623	22
14	1803	0	1964	2866	7	1723	36
15	1809	0	1898	3250	8	1644	38
16	1802	0	1709	3325	9	1589	33
17	1774	0	1598	3188	6	1575	24
18	1937	0	1844	3157	6	1620	29
19	2062	1	1887	3469	12	1660	38
20	2062	0	1924	3206	10	1762	32
21	2059	0	1838	3323	11	1799	26
22	2149	0	1814	3027	12	1744	31
23	2581	0	1409	2566	2	1298	20

Channel Group

engagement rate vs sessions over time

In [133…

```
df.head()
```

Out[133…

	channel group	DateHour	Users	Sessions	Engaged Sessions	Average engagement time per session	Engaged sessions per user	Events per session	Eng
0	Direct	2024-04-16 23:00:00	237	300	144	47.526667	0.607595	4.673333	
1	Organic Social	2024-04-17 19:00:00	208	267	132	32.097378	0.634615	4.295880	
2	Direct	2024-04-17 23:00:00	188	233	115	39.939914	0.611702	4.587983	
3	Organic Social	2024-04-17 18:00:00	187	256	125	32.160156	0.668449	4.078125	
4	Organic Social	2024-04-17 20:00:00	175	221	112	46.918552	0.640000	4.529412	

In [137…

```
df_plot = df.groupby("DateHour")[["Engagement rate", "Sessions"]].mean().reset_i

plt.figure(figsize=(10, 5))
plt.plot(df_plot["DateHour"], df_plot["Engagement rate"], label="Engagement rate
plt.plot(df_plot["DateHour"], df_plot["Sessions"], label="Sessions", color="blue
plt.title("📊 Engagement Rate vs Sessions Over Time")
```
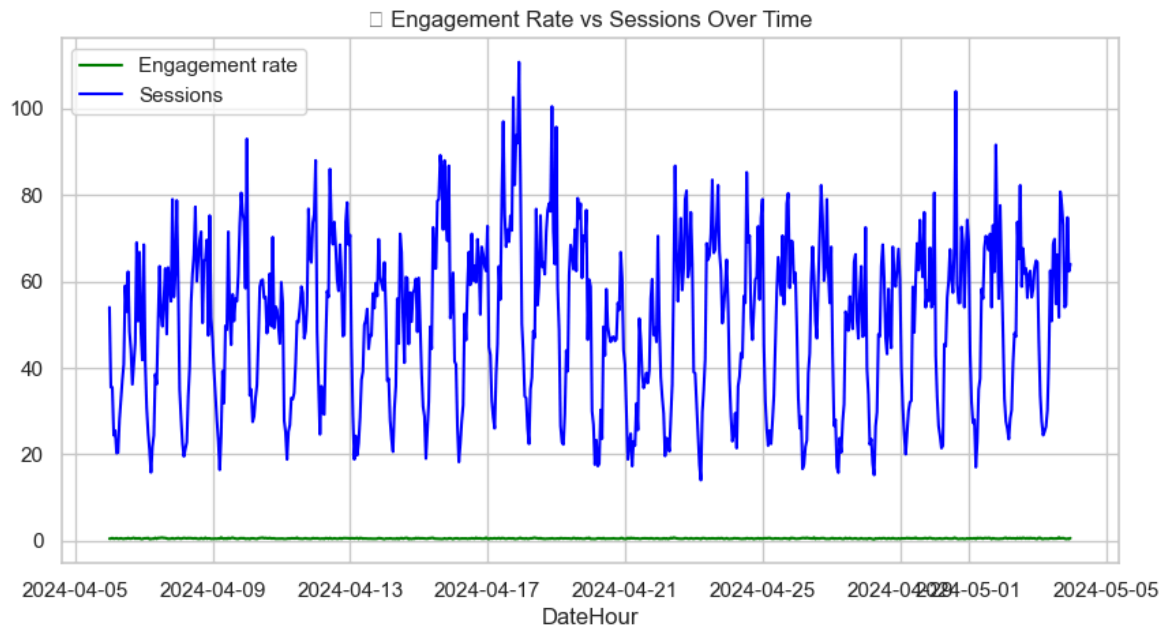
```
plt.xlabel("DateHour")
plt.legend()
plt.grid(True)
plt.show()
```

☐ Engagement Rate vs Sessions Over Time