In [24]:
```
#Write a Pandas Library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [25]:
```
#Data Upload
df = pd.read_csv('Websites.csv')
```

In [26]:
```
df.head()
```

Out[26]:

# ---------- ---------- ---------- ----	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	Enga
1	Direct	2024041623	237	300	144	47.526666666666700	0.60759
2	Organic Social	2024041719	208	267	132	32.09737827715360	0.63461
3	Direct	2024041723	188	233	115	39.93991416309010	0.61170
4	Organic Social	2024041718	187	256	125	32.16015625	0.66844

In [27]:
```
#Remove the Header
df.columns = df.iloc[0]
df.head()
```

Out[27]:

Session primary channel group (Default channel group)	Date + hour (YYYYMMDDHH)	Users	Sessions	Engaged sessions	Average engagement time per session	Engaged ses pe
0	Session primary channel group (Default channel...	Date + hour (YYYYMMDDHH)	Users	Sessions	Engaged sessions	Average engagement time per session	Engaged se p
1	Direct	2024041623	237	300	144	47.526666666666700	0.6075949367(
2	Organic Social	2024041719	208	267	132	32.09737827715360	0.6346153846
3	Direct	2024041723	188	233	115	39.93991416309010	0.6117021276
4	Organic Social	2024041718	187	256	125	32.16015625	0.6684491978(

In [28]:
```
#Drop the header and Create newone Rename Column Name
df = df.drop(index = 0).reset_index(drop=True)
df.columns = ["Channel group","Datehour","Users","Sessions","Engaged Sessio
              "Average engagement","Engaged sessions per user","Events per
```

In [29]:
```
df.head()
```

Out[29]:

	Channel group	Datehour	Users	Sessions	Engaged Sessions	Average engagement	Engaged sessions per use
0	Direct	2024041623	237	300	144	47.526666666666700	0.607594936708861(
1	Organic Social	2024041719	208	267	132	32.09737827715360	0.634615384615385(
2	Direct	2024041723	188	233	115	39.93991416309010	0.611702127659574(
3	Organic Social	2024041718	187	256	125	32.16015625	0.668449197860963(
4	Organic Social	2024041720	175	221	112	46.918552036199100	0.64

In [30]:
```
#Data type in Datebase
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         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
```

In [31]:
```
#Change Data Type
df["Datehour"] = pd.to_datetime(df["Datehour"], format="%Y%m%d%H", errors='
```

In [34]:
```
df.head()
```

Out[34]:

	Channel group	Datehour	Users	Sessions	Engaged Sessions	Average engagement	Engaged sessions per user	Events per session	Engager
0	Direct	2024-04-16 23:00:00	237	300	144	47.526667	0.607595	4.673333	0.48(
1	Organic Social	2024-04-17 19:00:00	208	267	132	32.097378	0.634615	4.295880	0.49
2	Direct	2024-04-17 23:00:00	188	233	115	39.939914	0.611702	4.587983	0.49
3	Organic Social	2024-04-17 18:00:00	187	256	125	32.160156	0.668449	4.078125	0.48
4	Organic Social	2024-04-17 20:00:00	175	221	112	46.918552	0.640000	4.529412	0.50(

In [33]:
```
#Separate hour from Datehour and other dataType in Numeric
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 [35]:
```
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        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   int64
dtypes: datetime64[ns](1), float64(4), int64(5), object(1)
memory usage: 273.6+ KB
```

In [36]:
```
df.describe()
```

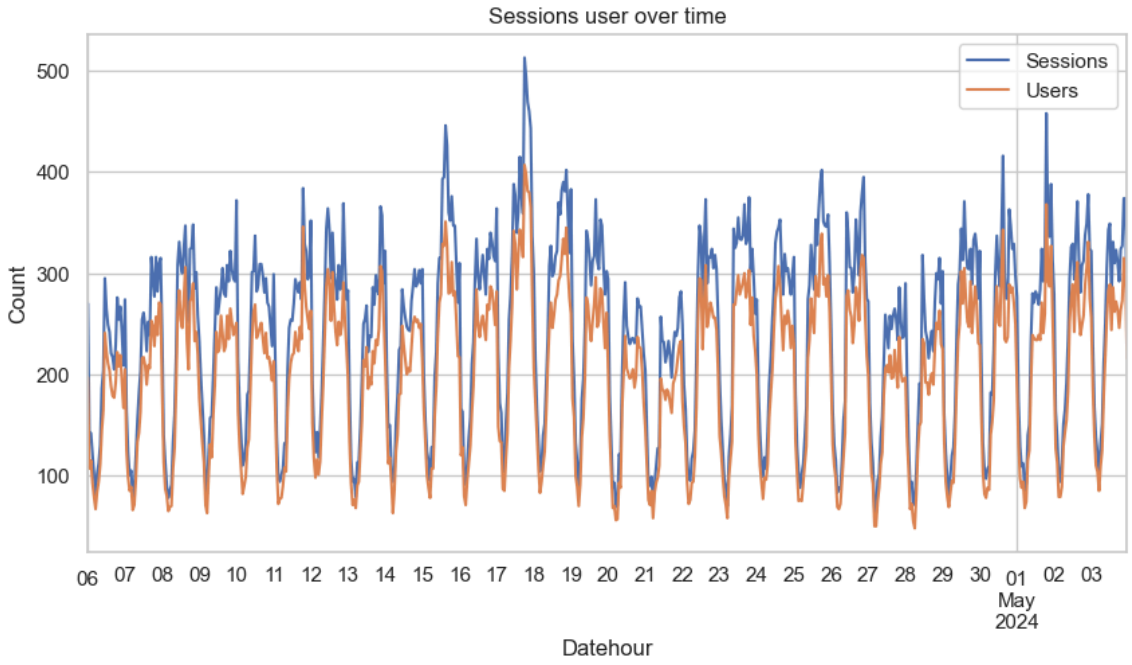
Out[36]:

	Users	Sessions	Engaged Sessions	Average engagement	Engaged sessions per user	Events per session	Engag
count	3182.000000	3182.000000	3182.000000	3182.000000	3182.000000	3182.000000	3182.(
mean	41.935889	51.192646	28.325581	66.644581	0.606450	4.675969	0.5
std	29.582258	36.919962	20.650569	127.200659	0.264023	2.795228	0.2
min	0.000000	1.000000	0.000000	0.000000	0.000000	1.000000	0.(
25%	20.000000	24.000000	13.000000	32.103034	0.561404	3.750000	0.4
50%	42.000000	51.000000	27.000000	49.020202	0.666667	4.410256	0.5
75%	60.000000	71.000000	41.000000	71.487069	0.750000	5.217690	0.6
max	237.000000	300.000000	144.000000	4525.000000	2.000000	56.000000	1.(

Sessions and User Over Time

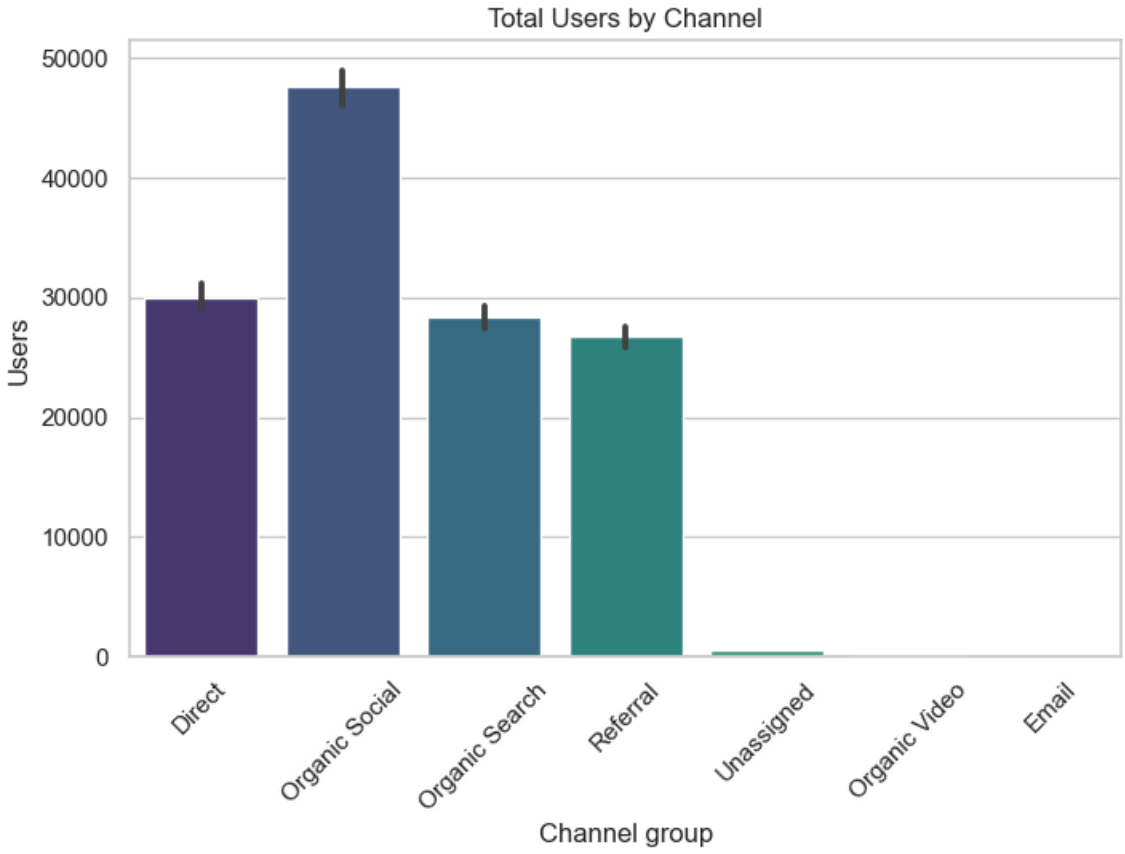
In [37]:
```
sns.set(style="whitegrid")
```

In [41]:
```
plt.figure(figsize=(10,5))
df.groupby("Datehour")[["Sessions","Users"]].sum().plot(ax=plt.gca())
plt.title("Sessions user over time")
plt.xlabel("Datehour")
plt.ylabel("Count")
plt.show()
```



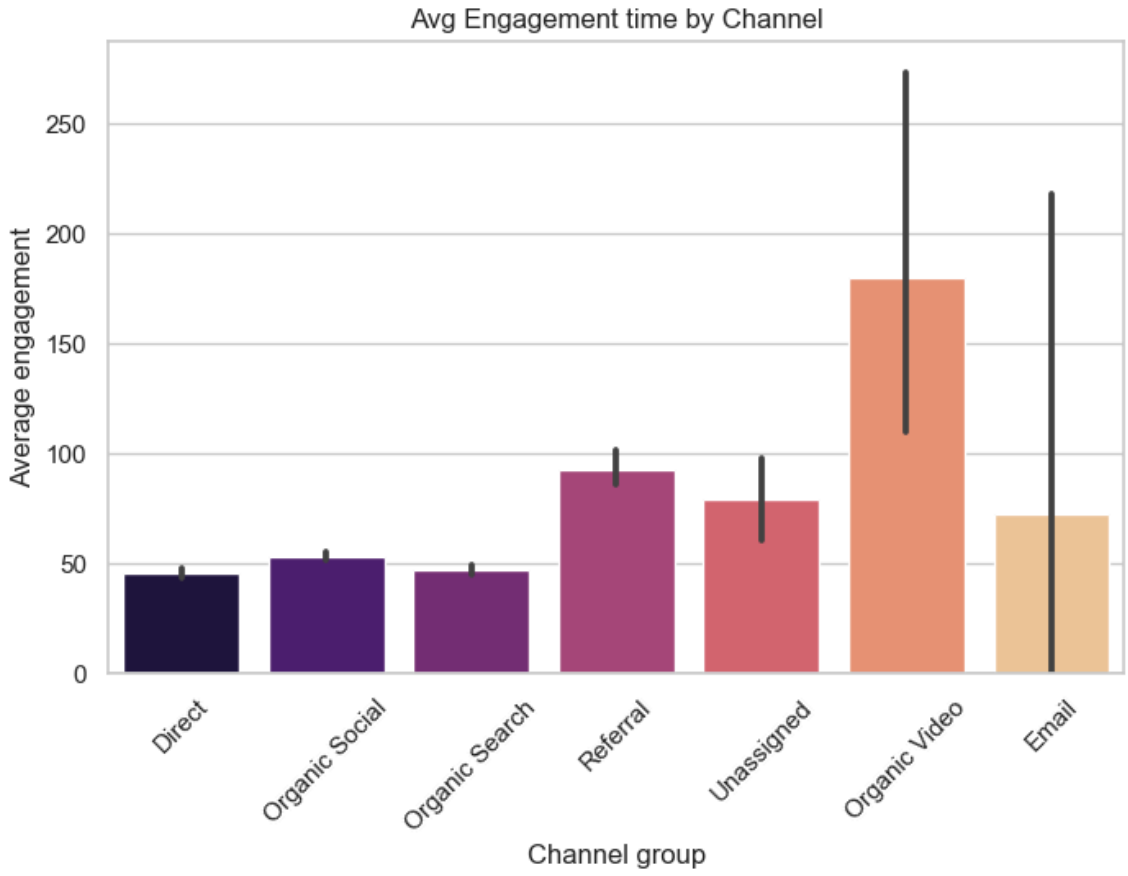
Total Users by Channel

In [42]:
```
plt.figure(figsize=(8,5))
sns.barplot(data=df, x='Channel group', y='Users', estimator=np.sum, palett
plt.title("Total Users by Channel")
plt.xticks(rotation=45)
plt.show()
```



Average Engagement time by Channel

In [45]:
```
plt.figure(figsize=(8,5))
sns.barplot(data=df, x='Channel group', y= 'Average engagement',estimator=n
plt.title("Avg Engagement time by Channel")
plt.xticks(rotation=45)
plt.show()
```



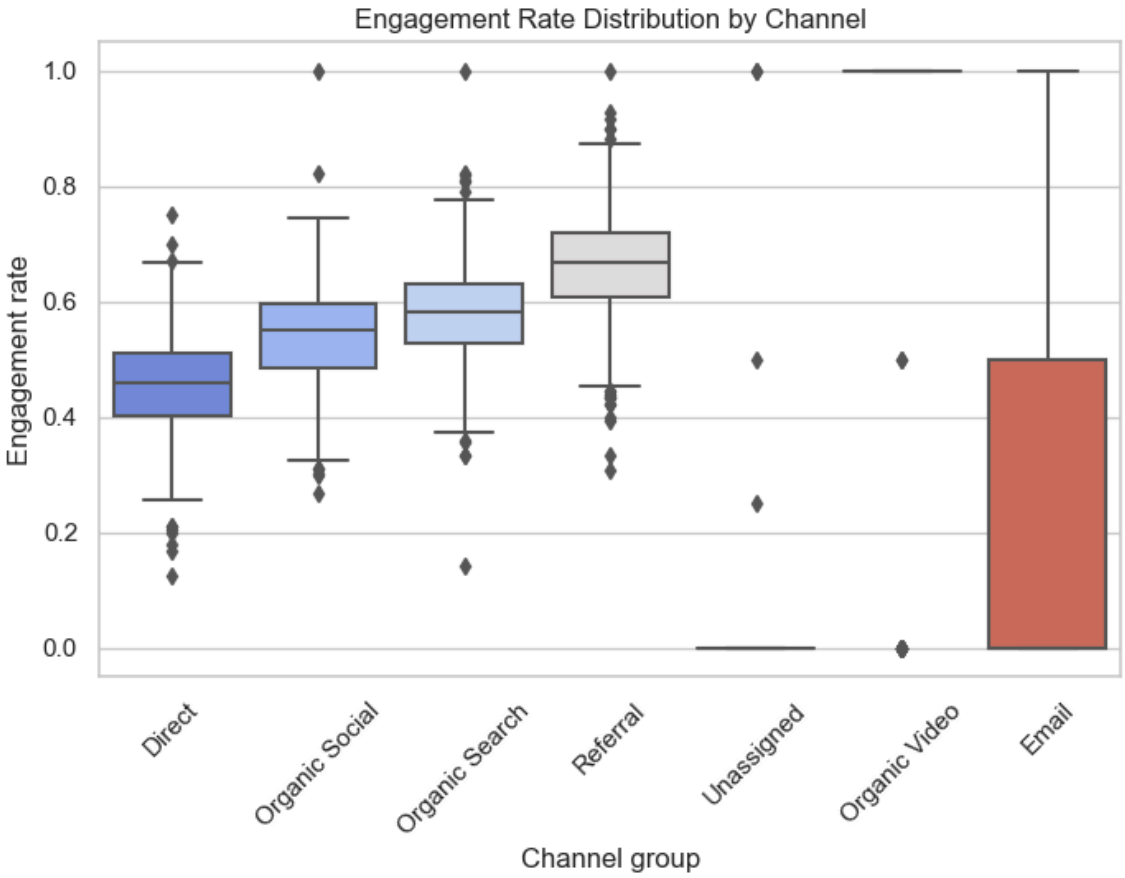
In [46]:
```
df.head()
```

Out[46]:

	Channel group	Datehour	Users	Sessions	Engaged Sessions	Average engagement	Engaged sessions per user	Events per session	Engager
0	Direct	2024-04-16 23:00:00	237	300	144	47.526667	0.607595	4.673333	0.48(
1	Organic Social	2024-04-17 19:00:00	208	267	132	32.097378	0.634615	4.295880	0.49
2	Direct	2024-04-17 23:00:00	188	233	115	39.939914	0.611702	4.587983	0.49
3	Organic Social	2024-04-17 18:00:00	187	256	125	32.160156	0.668449	4.078125	0.48
4	Organic Social	2024-04-17 20:00:00	175	221	112	46.918552	0.640000	4.529412	0.50(

Engagement rate distribution by Channel

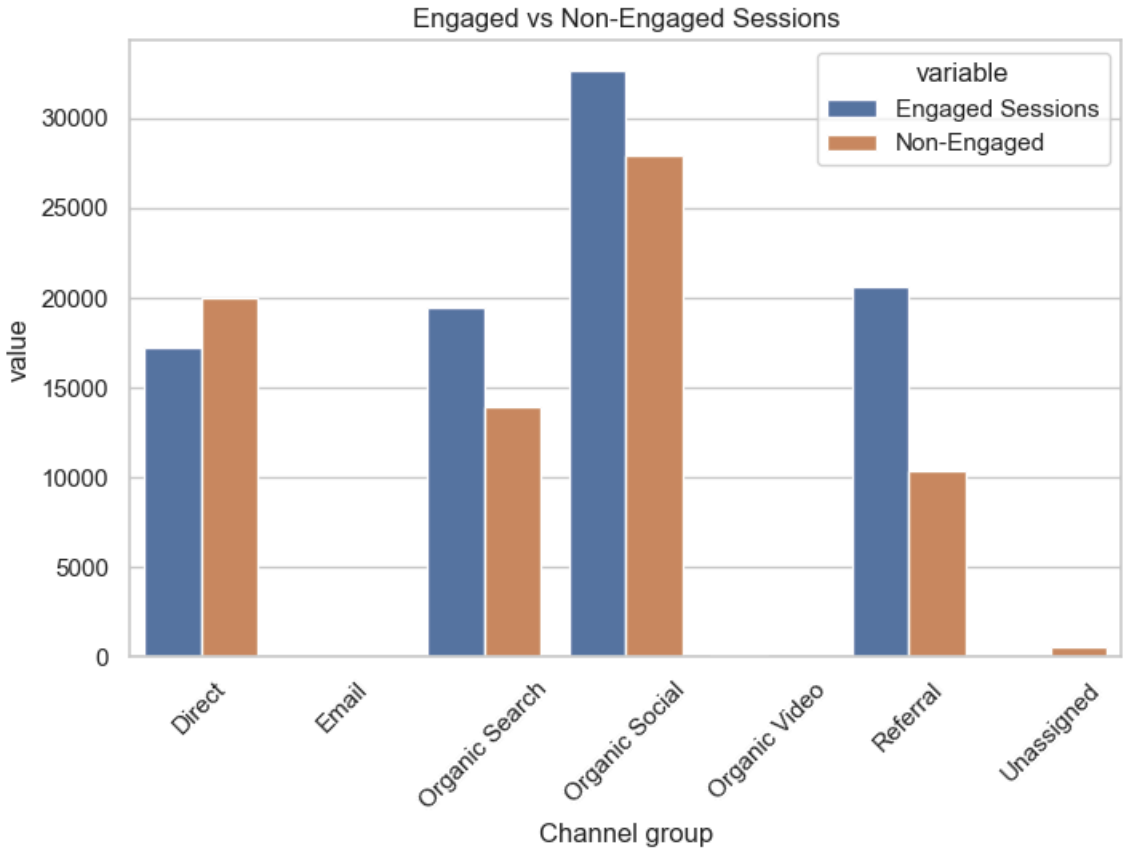
In [47]:
```
plt.figure(figsize=(8,5))
sns.boxplot(data=df, x='Channel group', y='Engagement rate', palette= 'cool
plt.title('Engagement Rate Distribution by Channel')
plt.xticks(rotation=45)
plt.show()
```



Engaged and non engaged sessions

In [49]:
```
#Group Engaged Sessions (Engaged&non-engaged)
session_df = df.groupby('Channel group')[['Sessions','Engaged Sessions']].s
session_df ['Non-Engaged'] = session_df['Sessions']-session_df['Engaged Ses
session_df_melted = session_df.melt(id_vars='Channel group', value_vars = [
```

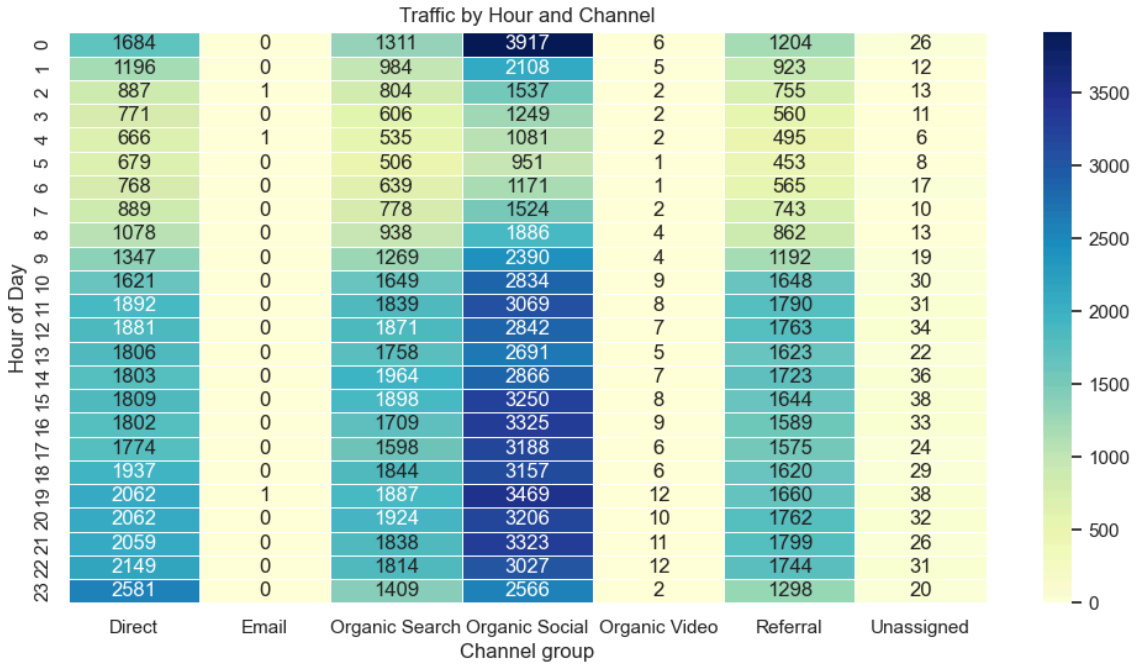
In [51]:
```
plt.figure(figsize=(8,5))
sns.barplot(data=session_df_melted, x='Channel group', y = 'value', hue='va
plt.title('Engaged vs Non-Engaged Sessions')
plt.xticks(rotation = 45)
plt.show()
```



Traffic by Hour and Channel

In [54]:
```
heatmap_data = df.groupby(['Hour', 'Channel group'])['Sessions'].sum().unst

plt.figure(figsize=(12,6))
sns.heatmap(heatmap_data, cmap='YlGnBu', linewidths=0.5, annot=True, fmt='.
plt.title('Traffic by Hour and Channel')
plt.xlabel('Channel group')
plt.ylabel('Hour of Day')
plt.show()
```



In [58]:
```
df_plot = df.groupby('Datehour')[['Engagement rate', 'Sessions']].mean().re

plt.figure(figsize=(10,5))
plt.plot(df_plot['Datehour'],df_plot['Engagement rate'], label='Engagement
plt.plot(df_plot['Datehour'],df_plot['Sessions'], label='Sessions', color='
plt.title('Engagement Rate vs Sessions over Time')
plt.xlabel('Datehour')
plt.legend()
plt.grid(True)
plt.show()
```