Importing Relevent Libraries

In [2]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
from scipy.stats import chi2_contingency
from statsmodels.stats.proportion import proportions_ztest
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading Data

In [3]:
```
try:
    df = pd.read_csv(r'C:\Users\Lenovo\Downloads\AdSmartABdata - AdSmartABc
    data_imported = True
except File_Not_Found_Error:
    data_imported = False

print('#------------------')
print('#Data Imported Status')

if data_imported:
    print('Data imported Successfully!')
else:
    print('Failed to fetch data. Check the file path or format')

print('#------------------')
print('\n#Dimensions-------------')
num_observations, num_columns = df.shape
print(f'observation:{num_observations} column:{num_columns}\n')
print('#Data_Types-------------')
object_vars = df.select_dtypes(include='object').columns.tolist()
int_vars = df.select_dtypes(include='int64').columns.tolist()
bool_vars = df.select_dtypes(include='bool').columns.tolist()

def print_variable_info(var_type, variables):
    num_variables = len(variables)
    print(f'{var_type}: variables')
    print(f'#of variables: {num_variables}')
    print(f' {variables}\n')

print_variable_info('object', object_vars)
print_variable_info('integer', int_vars)
print_variable_info('boolean', bool_vars)

print('#Missing Values--------')
if df.isnull().sum().sum() == 0:
    print('Are there missing values? \n No missing value')
else:
    print('Are there missing values ? \n yes there are missing values')
```

```
#-----------------
#Data Imported Status
Data imported Successfully!
#-----------------

#Dimensions-------------
observation:8077 column:9

#Data_Types-------------
object: variables
#of variables: 5
 ['auction_id', 'experiment', 'date', 'device_make', 'browser']

integer: variables
#of variables: 4
 ['hour', 'platform_os', 'yes', 'no']

boolean: variables
#of variables: 0
 []

#Missing Values--------
Are there missing values?
 No missing value
```

Summary Statistics

In [4]:
```
df.head()
```

Out[4]:

	auction_id	experiment	date	hour	device_make	platform_os	browser	yes	no
0	0008ef63-77a7-448b-bd1e-075f42c55e39	exposed	2020-07-10	8	Generic Smartphone	6	Chrome Mobile	0	0
1	000eabc5-17ce-4137-8efe-44734d914446	exposed	2020-07-07	10	Generic Smartphone	6	Chrome Mobile	0	0
2	0016d14a-ae18-4a02-a204-6ba53b52f2ed	exposed	2020-07-05	2	E5823	6	Chrome Mobile WebView	0	1
3	00187412-2932-4542-a8ef-3633901c98d9	control	2020-07-03	15	Samsung SM-A705FN	6	Facebook	0	0
4	001a7785-d3fe-4e11-a344-c8735acacc2c	control	2020-07-03	15	Generic Smartphone	6	Chrome Mobile	0	0

In [5]:
```
responses = ['yes', 'no']
summary_stats = df['hour'].describe([0.01, 0.05, 0.10, 0.20, 0.50, 0.80, 0.
summary_stats
```

Out[5]:
```
count    8077.000000
mean       11.615080
std         5.734879
min         0.000000
1%          0.000000
5%          1.000000
10%         3.000000
20%         6.000000
50%        13.000000
80%        16.000000
90%        19.000000
95%        20.000000
99%        22.000000
max        23.000000
Name: hour, dtype: float64
```

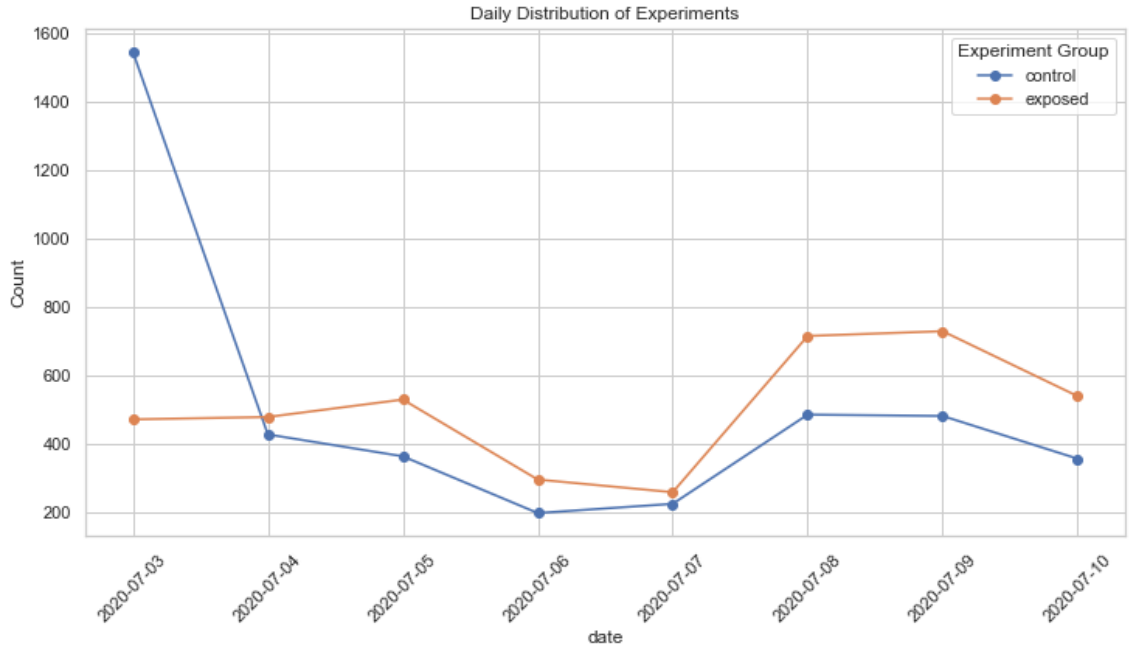
In [6]:
```
control_count = df[df['experiment']=='control'].shape[0]
exposed_count = df[df['experiment']=='exposed'].shape[0]
print(f'Control count : {control_count}')
print(f'Exposed count : {exposed_count}')
```

```
Control count : 4071
Exposed count : 4006
```

EDA

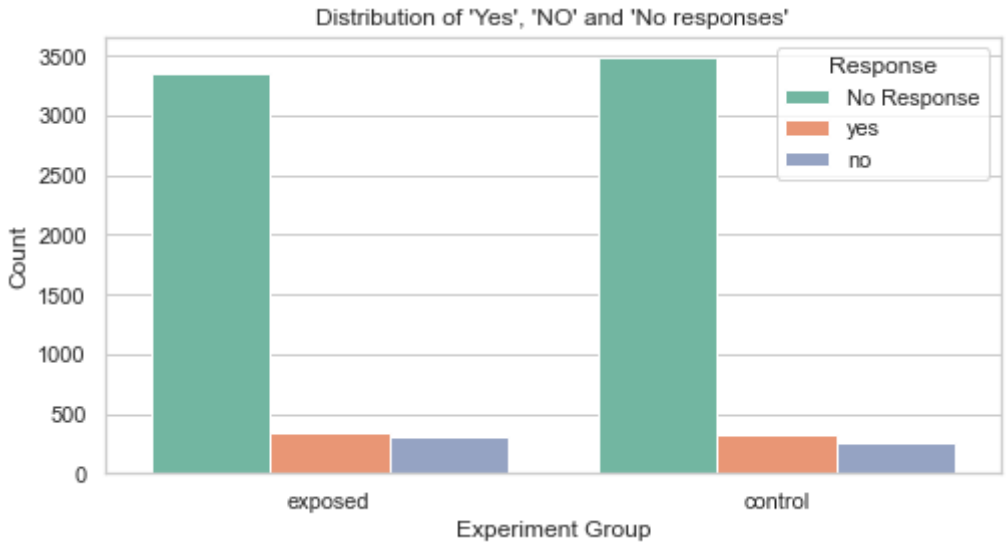
In [7]:
```
min_date = df['date'].min()
max_date = df['date'].max()
print(f'Minimum Date: {min_date}')
print(f'Maximum Date: {max_date}')

sns.set(style="whitegrid")
grouped_df = df.groupby(['date', 'experiment']).size().unstack()
plt.figure(figsize=(12,6))
for column in grouped_df.columns:
    plt.plot(grouped_df.index, grouped_df[column], marker='o', label=column
plt.title('Daily Distribution of Experiments')
plt.xlabel('date')
plt.ylabel('Count')
plt.legend(title='Experiment Group')
plt.xticks(rotation=45)
plt.show()
grouped_df.reset_index
```

```
Minimum Date: 2020-07-03
Maximum Date: 2020-07-10
```



Out[7]:
```
<bound method DataFrame.reset_index of experiment  control  exposed
date
2020-07-03     1545      470
2020-07-04      426      477
2020-07-05      362      528
2020-07-06      196      294
2020-07-07      223      257
2020-07-08      484      714
2020-07-09      480      728
2020-07-10      355      538>
```

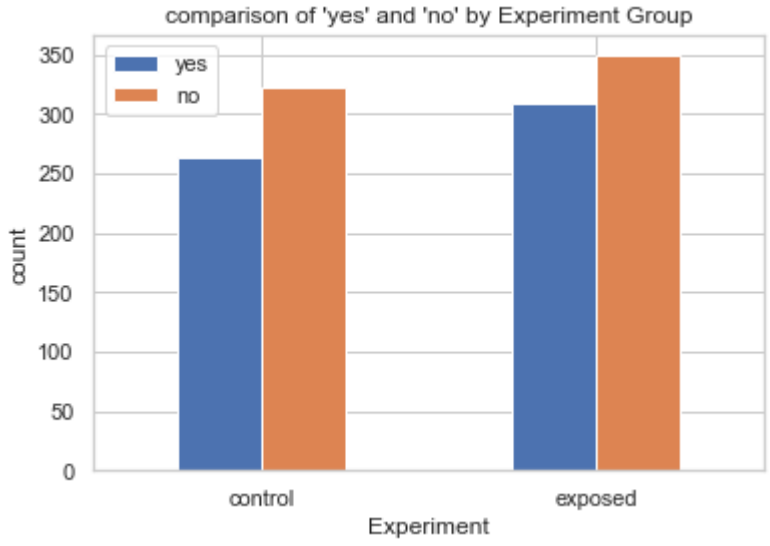
In [8]:
```
df['No_Response']=df.apply(lambda row: 'No Response' if row['yes']==0 and r
        else 'yes' if row['yes']==0 else 'no', axis=1)
sns.set(style='whitegrid')
plt.figure(figsize= (8,4))
sns.countplot(data=df,x='experiment', hue='No_Response', palette='Set2')
plt.title("Distribution of 'Yes', 'NO' and 'No responses'")
plt.xlabel('Experiment Group')
plt.ylabel('Count')
plt.legend(title='Response')
plt.show()
```



Data Preprocessing

In [9]:
```
#filtering : selecting experiment with only yes and no responses.
experiment_counts = df.groupby('experiment').agg({'yes': 'sum', 'no': 'sum'
print(experiment_counts)

#Calculating total 'yes' and 'no' counts
exposed_yes = df[df['experiment'] == 'exposed']['yes'].sum()
exposed_no = df[df['experiment'] == 'exposed']['no'].sum()
control_yes = df[df['experiment'] == 'control']['yes'].sum()
control_no = df[df['experiment'] == 'exposed']['no'].sum()
exposed_conversion_rate = (exposed_yes/(exposed_no+exposed_yes))*100
control_conversion_rate = (control_yes/(control_yes+control_no))*100
print(f"\nconversion rate in exposed group: {exposed_conversion_rate:.2f}")
print(f"conversion rate in control group: {control_conversion_rate:.2f}")

experiment_counts.plot(kind = 'bar', stacked = False)
plt.title("comparison of 'yes' and 'no' by Experiment Group")
plt.xlabel('Experiment')
plt.ylabel('count')
plt.xticks(rotation=0)
plt.show()

#creating new dataframe
filtered_df = df[(df['yes'] > 0) | (df['no'] > 0)]
filtered_df.info()
```

```
            yes   no
experiment
control     264  322
exposed     308  349

conversion rate in exposed group: 46.88
conversion rate in control group: 43.07
```



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1243 entries, 2 to 8071
Data columns (total 10 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   auction_id   1243 non-null   object
 1   experiment   1243 non-null   object
 2   date         1243 non-null   object
 3   hour         1243 non-null   int64 
 4   device_make  1243 non-null   object
 5   platform_os  1243 non-null   int64 
 6   browser      1243 non-null   object
 7   yes          1243 non-null   int64 
 8   no           1243 non-null   int64 
 9   No_Response  1243 non-null   object
dtypes: int64(4), object(6)
memory usage: 106.8+ KB
```

In [15]:

```
exposed_total_counts = df[df['experiment']== 'exposed'].shape[0]
control_total_counts = df[df['experiment']== 'control'].shape[0]
stat, pval = proportions_ztest([exposed_yes, control_yes], [exposed_total_c
print(pval)
if pval < 0.05:
    print("There's significant difference in 'yes' responses between the 'c
else:
    print("theree's no significant difference in 'yes' responses between th
```

```
0.035005825968324515
There's significant difference in 'yes' responses between the 'control' a
nd 'exposed' groups.
```