```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import missingno as msno
import seaborn as sns


june1_df = pd.read_csv("/content/202306-citibike-tripdata_1.csv", parse_dates=Tru
june2_df = pd.read_csv("/content/202306-citibike-tripdata_2.csv", parse_dates=Tru
june3_df = pd.read_csv("/content/202306-citibike-tripdata_3.csv", parse_dates=Tru
june4_df = pd.read_csv("/content/202306-citibike-tripdata_4.csv", parse_dates=Tru
```

```
<ipython-input-2-05de994420b8>:1: DtypeWarning: Columns (5,7) have mixed types
  june1_df = pd.read_csv("/content/202306-citibike-tripdata_1.csv", parse_date
<ipython-input-2-05de994420b8>:2: DtypeWarning: Columns (5) have mixed types.
  june2_df = pd.read_csv("/content/202306-citibike-tripdata_2.csv", parse_date
<ipython-input-2-05de994420b8>:3: DtypeWarning: Columns (5,7) have mixed types
  june3_df = pd.read_csv("/content/202306-citibike-tripdata_3.csv", parse_date
<ipython-input-2-05de994420b8>:4: DtypeWarning: Columns (7) have mixed types.
  june4_df = pd.read_csv("/content/202306-citibike-tripdata_4.csv", parse_date
```

```
dataset = pd.concat([june1_df, june2_df, june3_df, june4_df], axis = 0)


citibike_df = dataset.copy()
```

EDA

```
citibike_df.head()
```

	ride_id	rideable_type	started_at	ended_at	start_station_name	s
0	984F50BCBC76DD9A	classic_bike	2023-06-11 06:54:21	2023-06-11 07:12:28	W 84 St & Columbus Ave	
1	03E3D62E7FB76B05	classic_bike	2023-06-19 15:23:11	2023-06-19 16:00:05	E 89 St & York Ave	
2	8E7EE421A0B8BBF3	classic_bike	2023-06-06 16:07:05	2023-06-06 16:15:14	E 51 St & 2 Ave	
3	24D66A0C46493CB1	classic_bike	2023-06-26 19:52:23	2023-06-26 19:55:47	India St Pier	
4	E944882A074B8F61	classic_bike	2023-06-05 08:57:57	2023-06-05 09:13:36	E 47 St & 2 Ave	

```
citibike_df.columns
```

```
Index(['ride_id', 'rideable_type', 'started_at', 'ended_at',
       'start_station_name', 'start_station_id', 'end_station_name',
       'end_station_id', 'start_lat', 'start_lng', 'end_lat', 'end_lng',
       'member_casual'],
      dtype='object')
```

```
citibike_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 699200 entries, 0 to 187664
Data columns (total 13 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   ride_id             699200 non-null  object 
 1   rideable_type       699200 non-null  object 
 2   started_at          699200 non-null  object 
 3   ended_at            699200 non-null  object 
 4   start_station_name  697227 non-null  object 
 5   start_station_id    697226 non-null  object 
 6   end_station_name    692098 non-null  object 
 7   end_station_id      692098 non-null  object 
 8   start_lat           699197 non-null  float64
 9   start_lng           699197 non-null  float64
 10  end_lat             696841 non-null  float64
 11  end_lng             696841 non-null  float64
 12  member_casual       699196 non-null  object 
dtypes: float64(4), object(9)
memory usage: 74.7+ MB
```

```
citibike_df.isnull().sum()
```

```
ride_id                 0
rideable_type           0
started_at              0
ended_at                0
start_station_name    1972
start_station_id      1972
end_station_name      7808
end_station_id        7808
start_lat                0
start_lng                0
end_lat               2726
end_lng               2726
member_casual            0
dtype: int64
```

```
#bc theres over 3,000,000 rows of data even dropping 7808 rows won't effect the a
data_loss = round((citibike_df['end_station_name'].isna().sum()/citibike_df.shape
data_loss
```

```
0
```

```
citibike_df.dropna(axis=0, inplace=True)
citibike_df.isna().sum()
```

```
ride_id               0
rideable_type         0
started_at            0
ended_at              0
start_station_name    0
start_station_id      0
end_station_name      0
end_station_id        0
start_lat             0
start_lng             0
end_lat               0
end_lng               0
member_casual         0
dtype: int64
```

```
citibike_df.nunique()
```

```
ride_id               3552400
rideable_type               3
started_at            1626980
ended_at              1633381
start_station_name       1870
start_station_id         3787
end_station_name         1897
end_station_id           2731
start_lat              710352
start_lng              607021
end_lat                  2557
end_lng                  2553
member_casual               2
dtype: int64
```

```
citibike_df.member_casual.value_counts()
```

```
member_casual
member    2846031
casual     706369
Name: count, dtype: int64
```

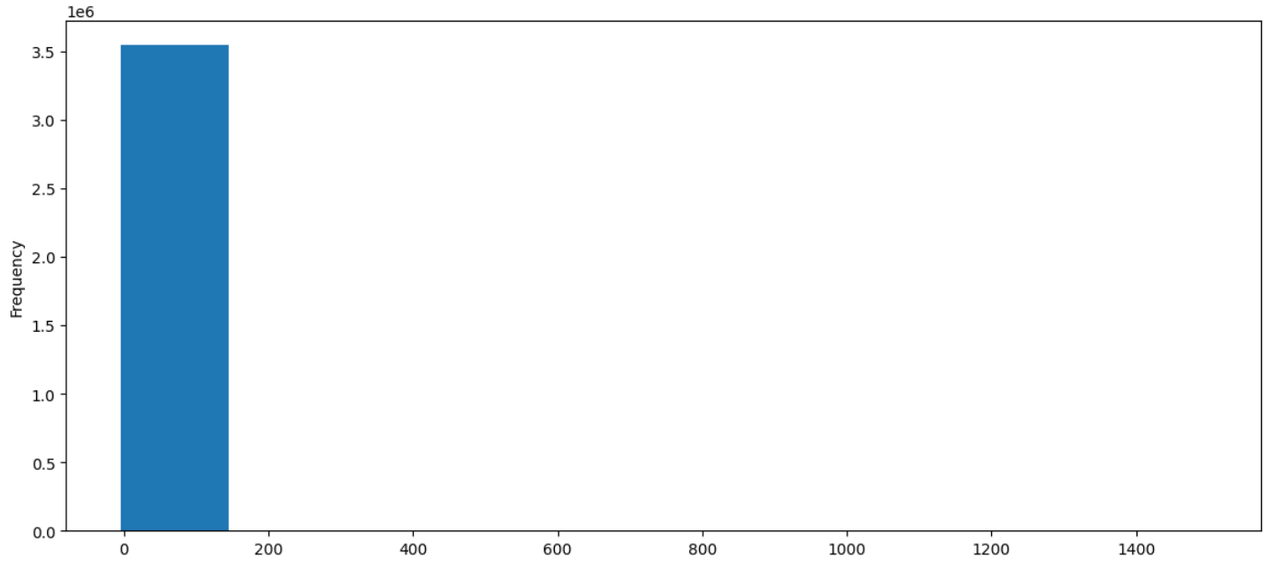
Majority of users are members

```
#calculate duration
citibike_df['duration'] = citibike_df['ended_at'] - citibike_df['started_at']
```

```
citibike_df['durationInSeconds'] = citibike_df['duration'].dt.total_seconds()
citibike_df['durationInMinutes'] = round((citibike_df['durationInSeconds']/60), 0
```

```
citibike_df['durationInMinutes'].plot(kind='hist', figsize = (14,6))
```

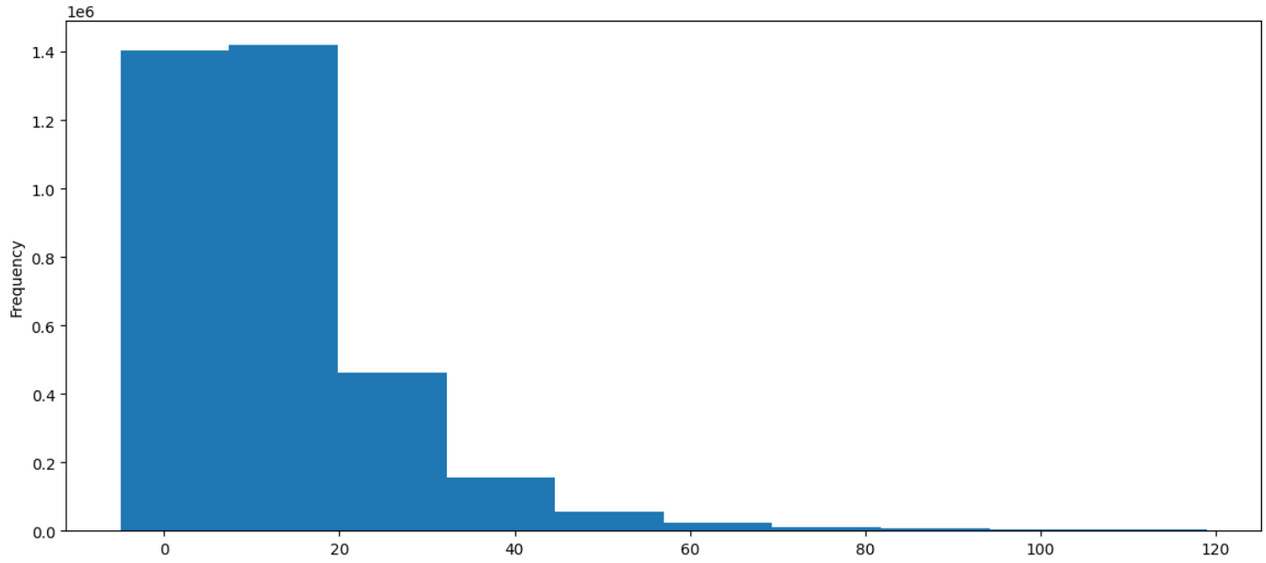
```
<Axes: ylabel='Frequency'>
```



Majority of duration is under 200 minutes

```
citibike_df[citibike_df.durationInMinutes < 120].durationInMinutes.plot(kind='his
```

```
<Axes: ylabel='Frequency'>
```



Looking closer majority of the duration is under 45 minutes with more bikerides being less than 20 minutes long

```
duration_per_hour = pd.value_counts(citibike_df['durationInMinutes'])
duration_per_hour.head(n=20)
```

```
durationInMinutes
6.0     229671
4.0     227044
5.0     226900
7.0     209234
8.0     199529
3.0     192414
9.0     175583
10.0    164387
2.0     154514
11.0    143097
12.0    131615
13.0    114339
14.0    105710
1.0      97379
15.0     92460
16.0     86107
17.0     74812
18.0     69839
0.0      66167
19.0     61264
Name: count, dtype: int64
```

There are 66,167 bikes that were used for 0 minutes and 97,379 bikes that were used for 1 minute, which isn't enough time to bike anywhere. This is most likely due to users taking out a bike and putting it back due to a mistake or getting a broken bike so we'll remove these values if the start and end station are the same.

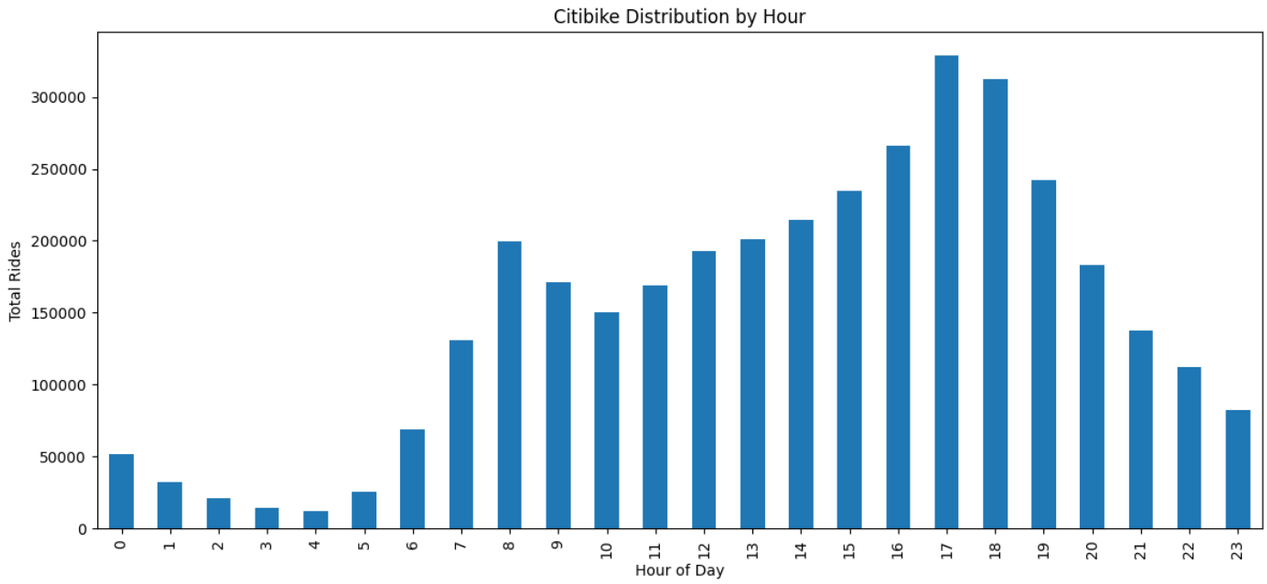
```
#drop duration in minutes that are is only 0 or 1 minute long if start and end st
citibike_df = citibike_df.loc[~(((citibike_df['durationInMinutes'] == 0) & (citib
```

```
#looking at what time of day users bike at
citibike_df['hour'] = citibike_df['started_at'].dt.hour
citibike_df['hour'].value_counts()
```

```
hour
17    328898
18    312354
16    266106
19    242383
15    235036
14    214192
13    200869
8     199299
12    192632
20    183122
9     171248
11    169199
10    150274
21    137418
7     130560
22    112281
23     82009
6      68677
0      51456
1      32183
5      25546
2      21179
3      13889
4      11590
Name: count, dtype: int64
```

```
citibike_df['hour'].value_counts().sort_index().plot(kind = 'bar', figsize=(14,6)

plt.ylabel('Total Rides')
plt.xlabel('Hour of Day')
plt.title("Citibike Distribution by Hour")
```

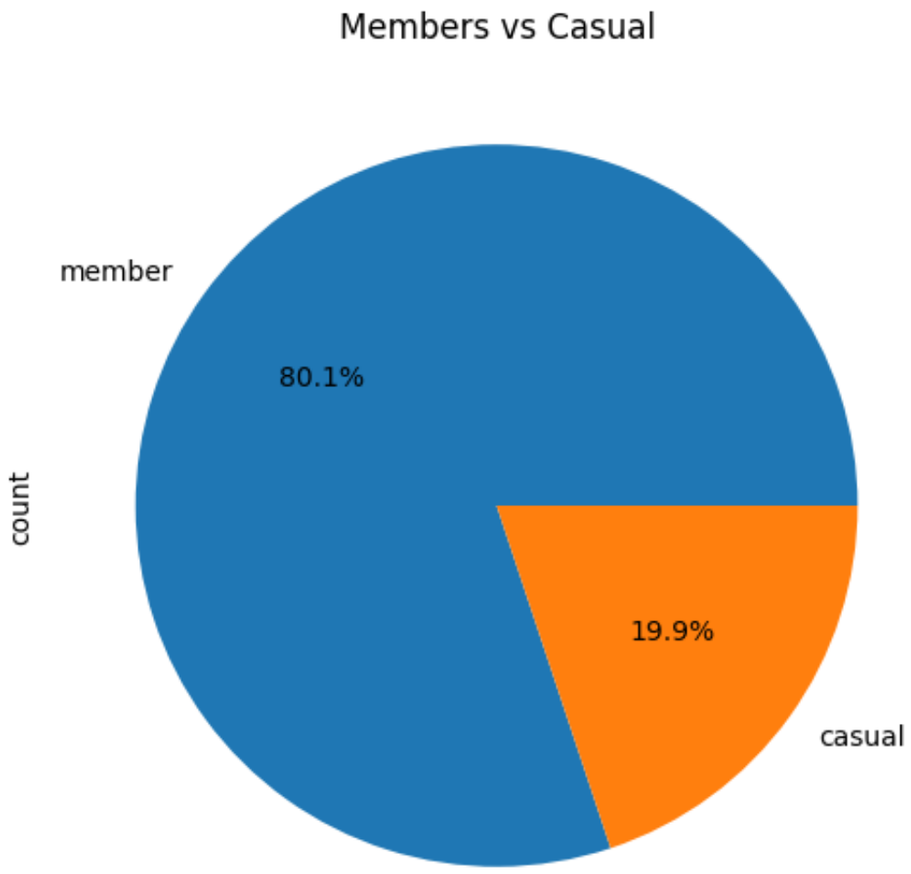
```
Text(0.5, 1.0, 'Citibike Distribution by Hour')
```



Members vs casual

```
citibike_df['member_casual'].value_counts().plot(kind = 'pie', figsize=(14,6), au
plt.title('Members vs Casual')
```

```
Text(0.5, 1.0, 'Members vs Casual')
```



Majority of users are members

Diving deeper into when users ride

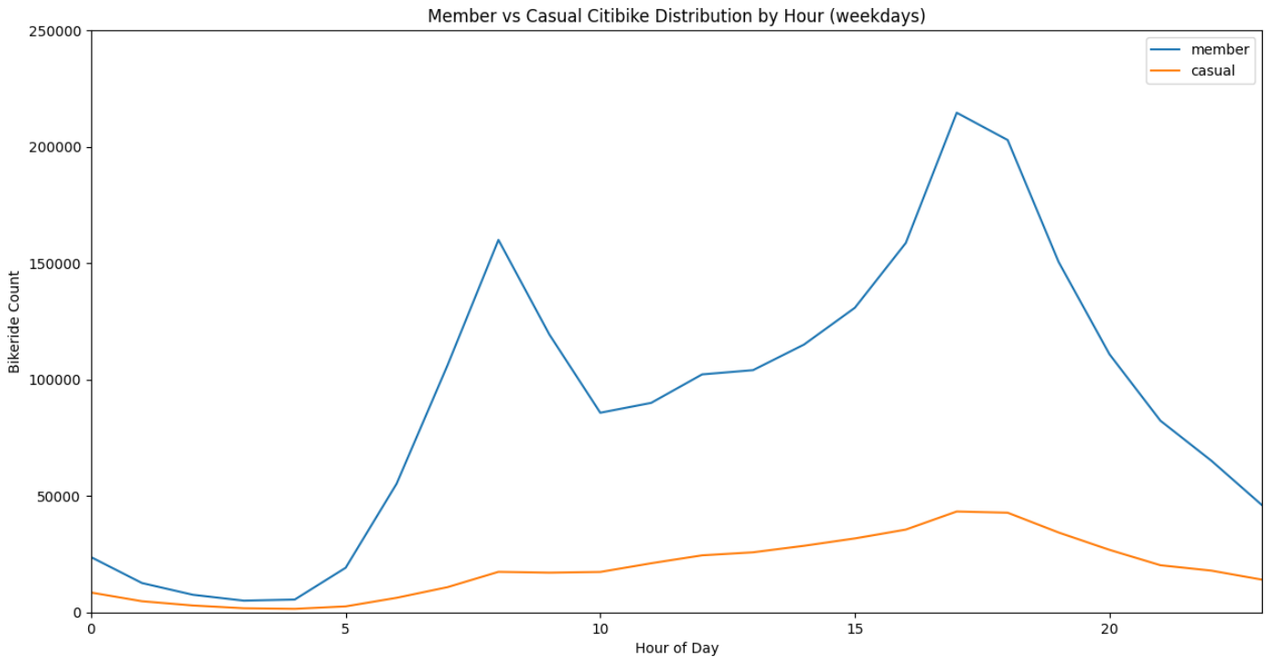
Visualizing when users bike the most can help us understand why they're using them. For example, whether they're using Citibikes to commute to work or to explore NYC.

What times do users ride their bike during weekdays?

```
#fitering for only data on weekdays
citibike_df['dayOfWeek'] = citibike_df['started_at'].dt.dayofweek

#creating a new column that displays whether a date is a weekend or weekday
bins = [-1, 4, 7, np.inf]
weekday_weekend = ['Weekday','Weekend', '']
citibike_df['weekday_weekend'] = pd.cut(citibike_df['dayOfWeek'], bins, labels=we

df2_member = citibike_df[(citibike_df["weekday_weekend"] == 'Weekday') & (citibik
time_member = pd.value_counts(df2_member['hour'])
time_member_df = time_member.to_frame()
time_member_df.reset_index(inplace = True)
```

```
df2_casual = citibike_df[(citibike_df["weekday_weekend"] == 'Weekday') & (citibik
time_casual = pd.value_counts(df2_casual['hour'])
time_casual_df = time_casual.to_frame()
time_casual_df.reset_index(inplace = True)

plt.figure(figsize=(15,7.5))

ax = sns.lineplot(x = time_member.index, y = time_member, data = time_member_df,
ax = sns.lineplot(x = time_casual.index, y = time_casual, data = time_casual_df,
ax.set(xlim=(0, 23))
ax.set(ylim=(0, 250000))

plt.title("Member vs Casual Citibike Distribution by Hour (weekdays)")
plt.xlabel("Hour of Day")
plt.ylabel("Bikeride Count")
```

```
Text(0, 0.5, 'Bikeride Count')
```

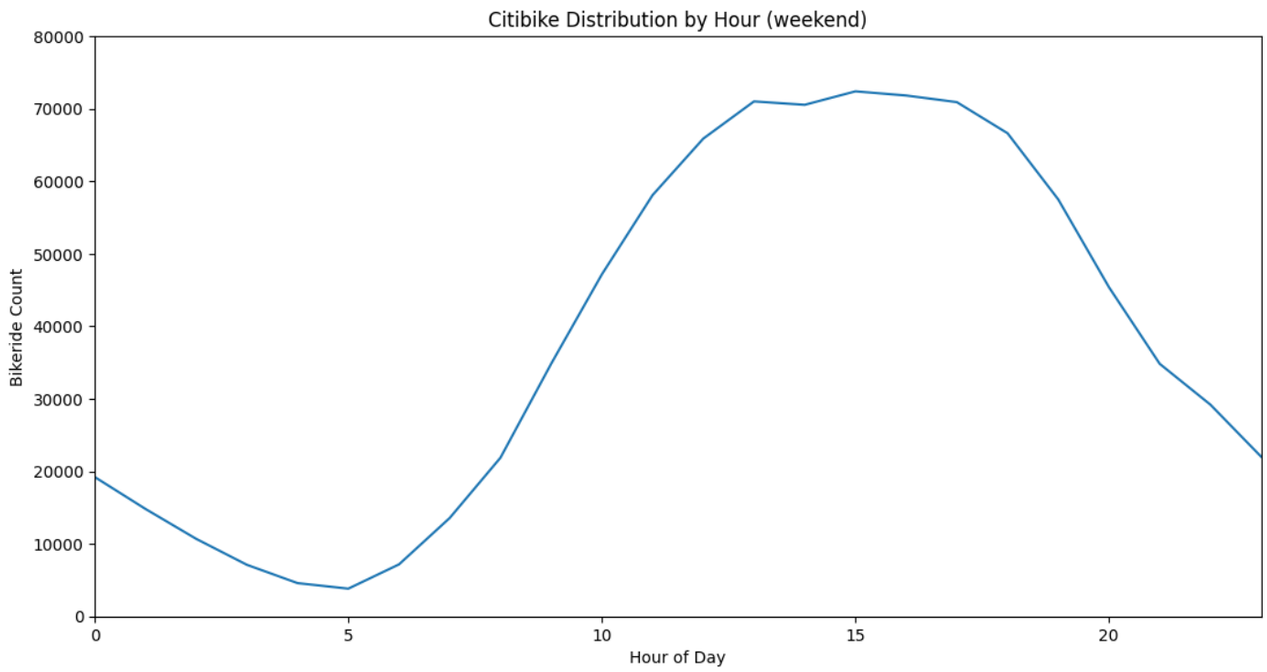


On weekdays, members increased between 6-10 am and 4-8 pm due to work hours and commuting, while casual users remained consistent without as much of a significant increase. People who are commuting are members because when they use Citibikes almost everyday it's more cost effective to pay 219.99 a year than paying $ 4.79 for 30 minutes each trip. This shows that members who are commuting get more exercise through Citibikes as they use them nearly everyday to get to work.

What times do users ride their bike during weekends?

```
weekend_df = citibike_df[(citibike_df["weekday_weekend"] == 'Weekend')]

timeWeekends = pd.value_counts(weekend_df['hour'])
timeWeekends_df = timeWeekends.to_frame()

plt.figure(figsize=(13,6.5))

ax = sns.lineplot(x = timeWeekends.index, y = timeWeekends, data=timeWeekends_df)
ax.set(xlim=(0, 23))
ax.set(ylim=(0, 80000))

plt.title("Citibike Distribution by Hour (weekend)")
plt.xlabel("Hour of Day")
plt.ylabel("Bikeride Count")
```

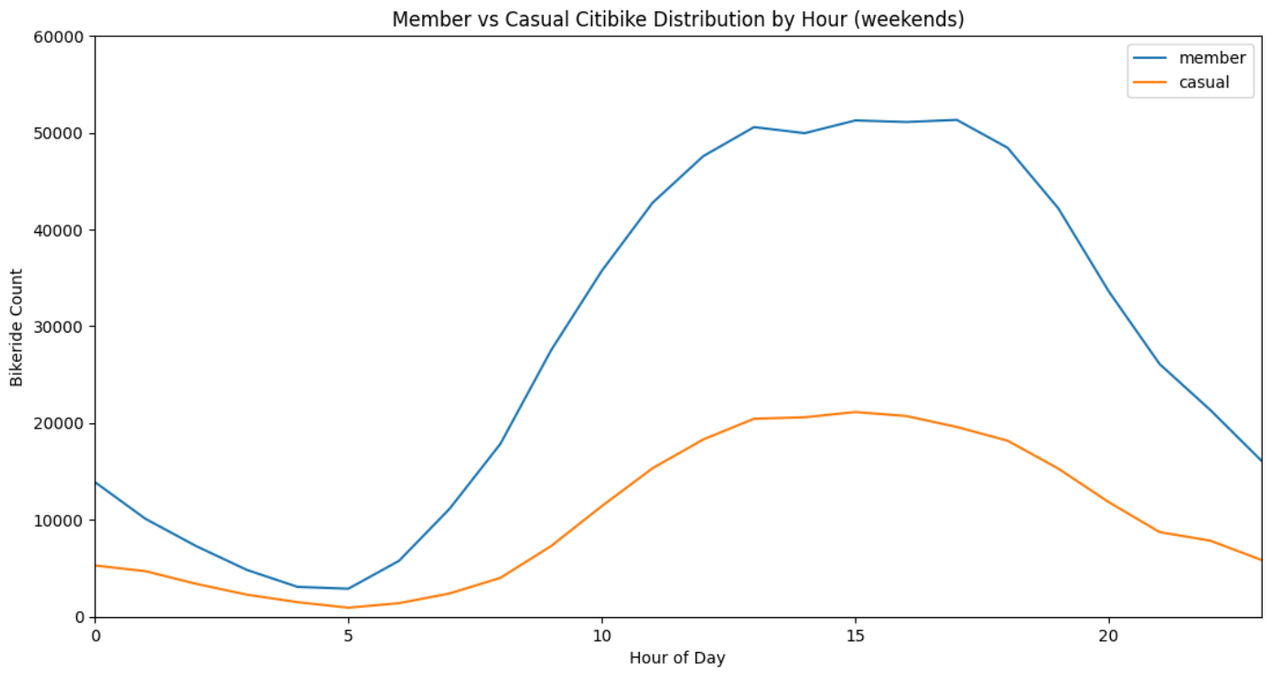
```
Text(0, 0.5, 'Bikeride Count')
```



During the weekends, there's a major decrease in the overall number of citibikes used from an overall max of 328,898 during the weekdays to only a max of 72,419 during weekends.

```
weekend_member = citibike_df[(citibike_df['weekday_weekend'] == 'Weekend') & (cit

timeWeekends_member = pd.value_counts(weekend_member['hour'])
timeWeekends_member_df = timeWeekends_member.to_frame()

weekend_casual = citibike_df[(citibike_df['weekday_weekend'] == 'Weekend') & (cit
timeWeekends_casual = pd.value_counts(weekend_casual['hour'])
timeWeekends_casual_df = timeWeekends_casual.to_frame()

plt.figure(figsize=(13,6.5))

ax = sns.lineplot(x = timeWeekends_member.index, y = timeWeekends_member, data=ti
ax = sns.lineplot(x = timeWeekends_casual.index, y = timeWeekends_casual, data=ti

ax.set(xlim=(0, 23))
ax.set(ylim=(0, 60000))

plt.title("Member vs Casual Citibike Distribution by Hour (weekends)")
plt.xlabel("Hour of Day")
plt.ylabel("Bikeride Count")
```

```
Text(0, 0.5, 'Bikeride Count')
```



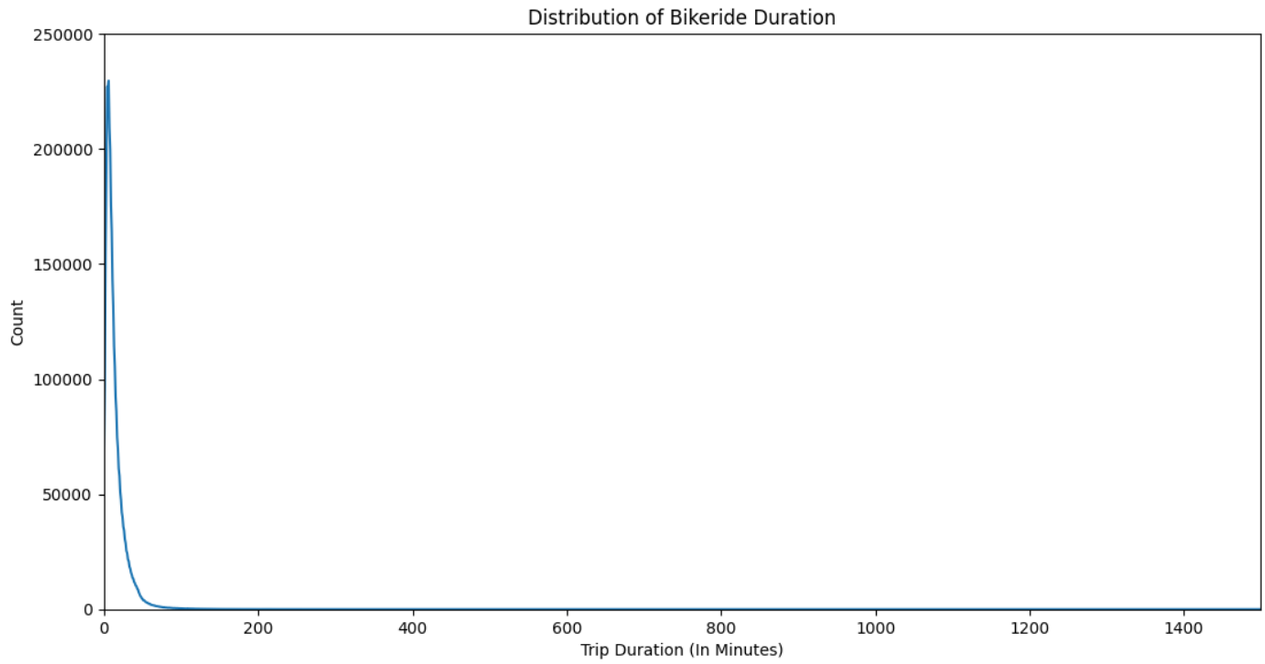
Unlike weekdays, during weekends customers and users have a similar distribution where most biked between 8am and 10pm. And similar to weekdays, there are many more members than casual users riding citibikes. As users have more free time during the weekend, Citibike could provide more of an incentive by offering a lower price or allowing them to use bikes for a longer time on weekends to encourage being more active.

Bike Ride Duration

Analyzing the duration helps understand how long users are riding bikes and how much exercise they're getting. The higher the duration, the more exercise the users get.

```
duration = pd.value_counts(citibike_df['durationInMinutes'])
dur_df = duration.to_frame()

plt.figure(figsize=(13,6.5))

ax = sns.lineplot(x = duration.index, y = duration, data=dur_df)
ax.set(xlim=(0, 1500))
ax.set(ylim=(0, 250000))

plt.title("Distribution of Bikeride Duration")
plt.xlabel("Trip Duration (In Minutes)")
plt.ylabel("Count")
```

```
Text(0, 0.5, 'Count')
```

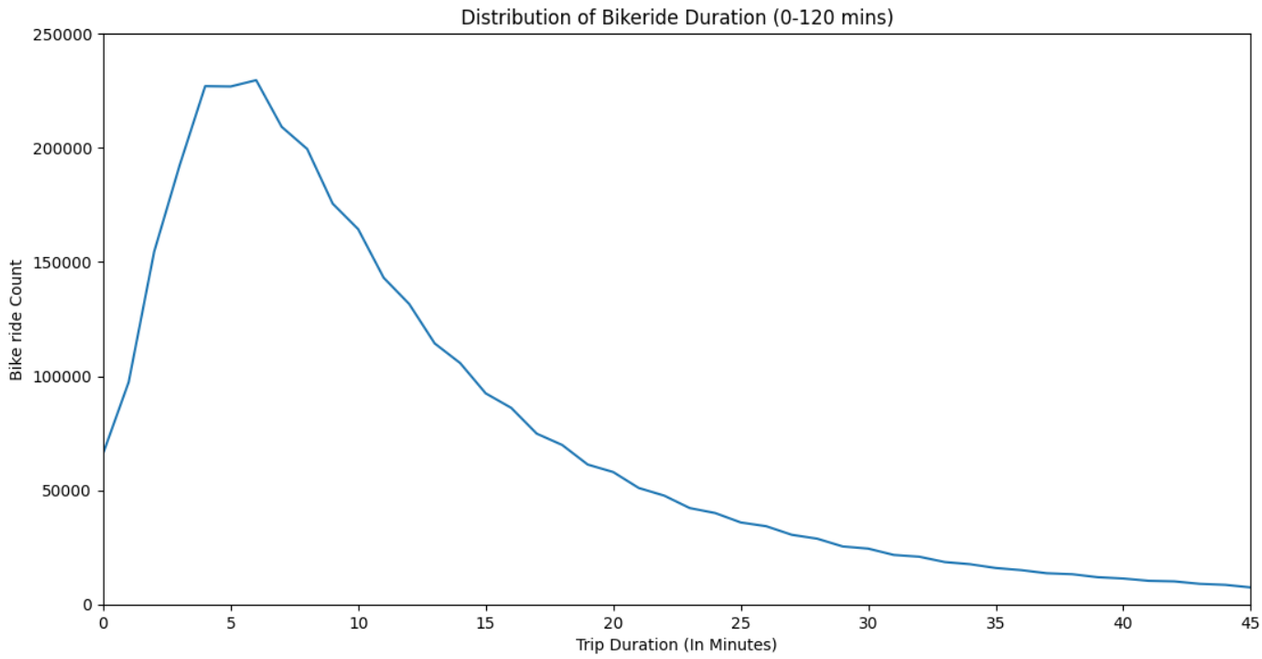


As we saw during exploratory data analysis, majority of bikes were ridden for less than 45 minutes. The bikes taken for 120+ minutes were most likely stolen, lost, or mistakes of ppl forgetting to return bikes as after 30 minutes for casual users and 45 minutes for members it costs $4 for every extra 15 minutes which is a high price to pay. So we'll drop these values.

```
#dropping values greater than 120
citibike_df = citibike_df.loc[~(citibike_df['durationInMinutes'] > 120)]
```

```
duration_per_hour = pd.value_counts(citibike_df['durationInMinutes'])
dph_df = duration_per_hour.to_frame()
dph_df.reset_index(inplace=True)

plt.figure(figsize=(13,6.5))

ax = sns.lineplot(x = duration_per_hour.index, y = duration_per_hour, data=dph_df

ax.set(xlim=(0, 45))
ax.set(ylim=(0, 250000))

plt.title("Distribution of Bikeride Duration (0-120 mins)")
plt.xlabel("Trip Duration (In Minutes)")
plt.ylabel("Bike ride Count")
```

```
Text(0, 0.5, 'Bike ride Count')
```



Most bike rides are between 0 and 20 minutes with most being around only 5 minutes.

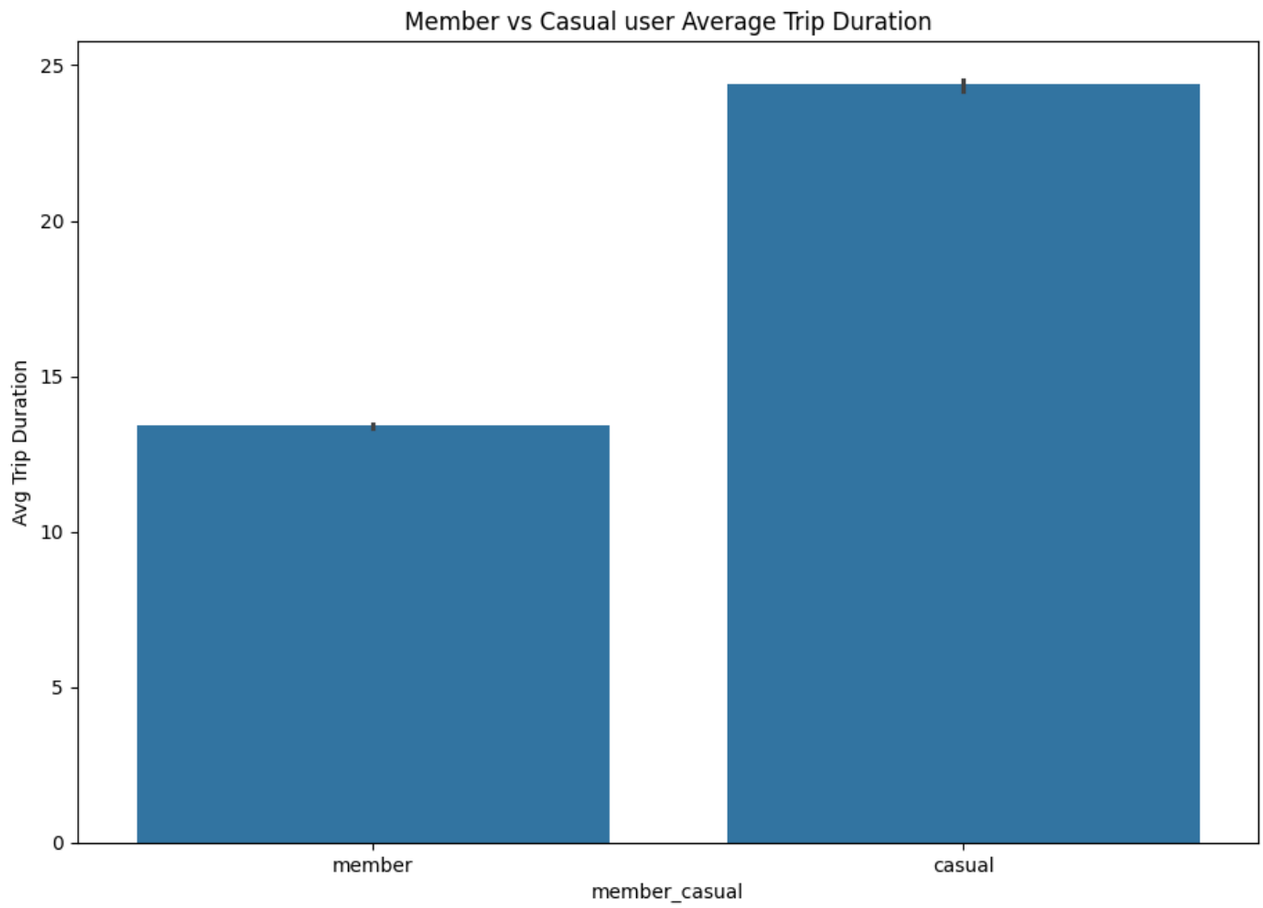
Member vs Casual user Trip Duration

```
weekend_df.groupby('member_casual').durationInMinutes.mean()
```

```
member_casual
casual    24.386421
member    13.402710
Name: durationInMinutes, dtype: float64
```

```
plt.figure(figsize=(11,7.5))

sns.barplot(x = 'member_casual', y = 'durationInMinutes', data=weekend_df)
plt.title("Member vs Casual user Average Trip Duration")
plt.ylabel("Avg Trip Duration")
```

```
Text(0, 0.5, 'Avg Trip Duration')
```

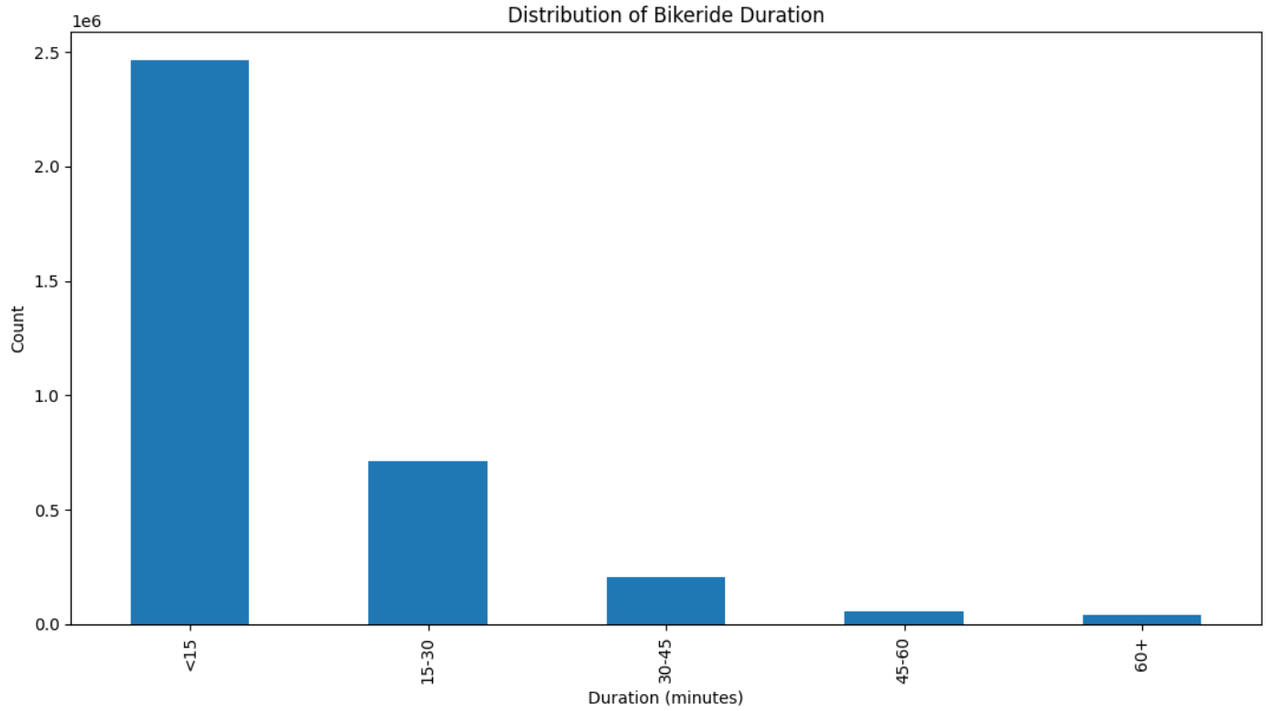


Casual users biked for a longer duration than members which was surprising but it does make sense as casual users use citibikes less often and pay more for each trip so want to ride for a longer time to get the most benefit. As can see above the average for casual users is 24 minutes when the time limit is 30 minutes. So unless there's a problem with not having enough bikes for everyone, Citibike could consider having a higher time limit which further encourages being active so users could ride for a longer time.

```
#Dividing duration into categories
bins = [0, 15, 30, 45, 60, np.inf]
minutesCategory = ['<15', '15-30', '30-45', '45-60', '60+']

citibike_df['minutesCat'] = pd.cut(citibike_df['durationInMinutes'], bins, labels=
```

```
citibike_df['minutesCat'].value_counts().plot(kind='bar', figsize=(13,6.5))
plt.title("Distribution of Bikeride Duration")
plt.xlabel("Duration (minutes)")
plt.ylabel("Count")
```

```
Text(0, 0.5, 'Count')
```

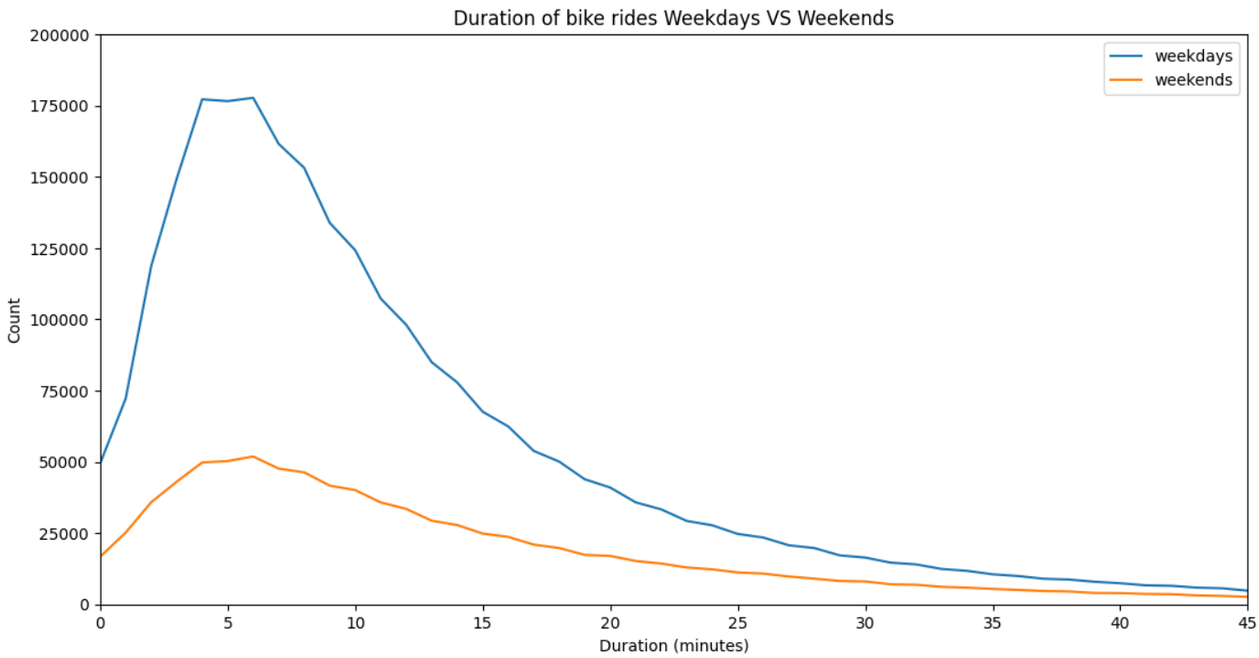


As expected, majority of bikers ride for less than 15 minutes, followed by 15-30 and it gets lower as we go further after 45.

Weekdays VS Weekends Trip Duration

```
weekend_df = citibike_df[(citibike_df["weekday_weekend"] == 'Weekend')]

durationWeekends = pd.value_counts(weekend_df['durationInMinutes'])
timeWeekends_df = durationWeekends.to_frame()

durationWeekday = citibike_df[citibike_df['weekday_weekend'] == 'Weekday']
durationWeekday = pd.value_counts(durationWeekday['durationInMinutes'])
timeWeekday_df = durationWeekday.to_frame()

plt.figure(figsize=(13,6.5))

ax = sns.lineplot(x = durationWeekday.index, y = durationWeekday, data=timeWeekda
ax = sns.lineplot(x = durationWeekends.index, y = durationWeekends, data=timeWeek

ax.set(xlim=(0, 45))
ax.set(ylim=(0, 200000))

plt.title("Duration of bike rides Weekdays VS Weekends")
plt.xlabel("Duration (minutes)")
plt.ylabel("Count")
```

```
Text(0, 0.5, 'Count')
```

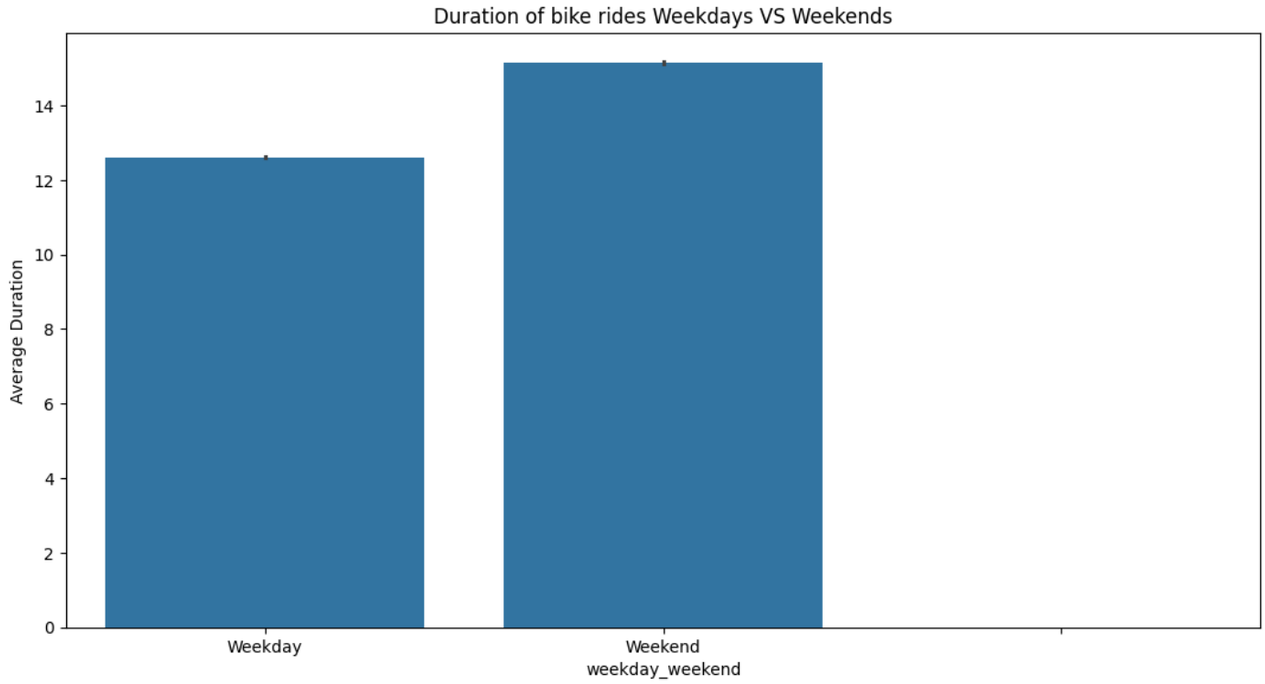



```
citibike_df.groupby('weekday_weekend').durationInMinutes.mean()
```

```
weekday_weekend
Weekday    12.599312
Weekend    15.149218
                 NaN
Name: durationInMinutes, dtype: float64
```

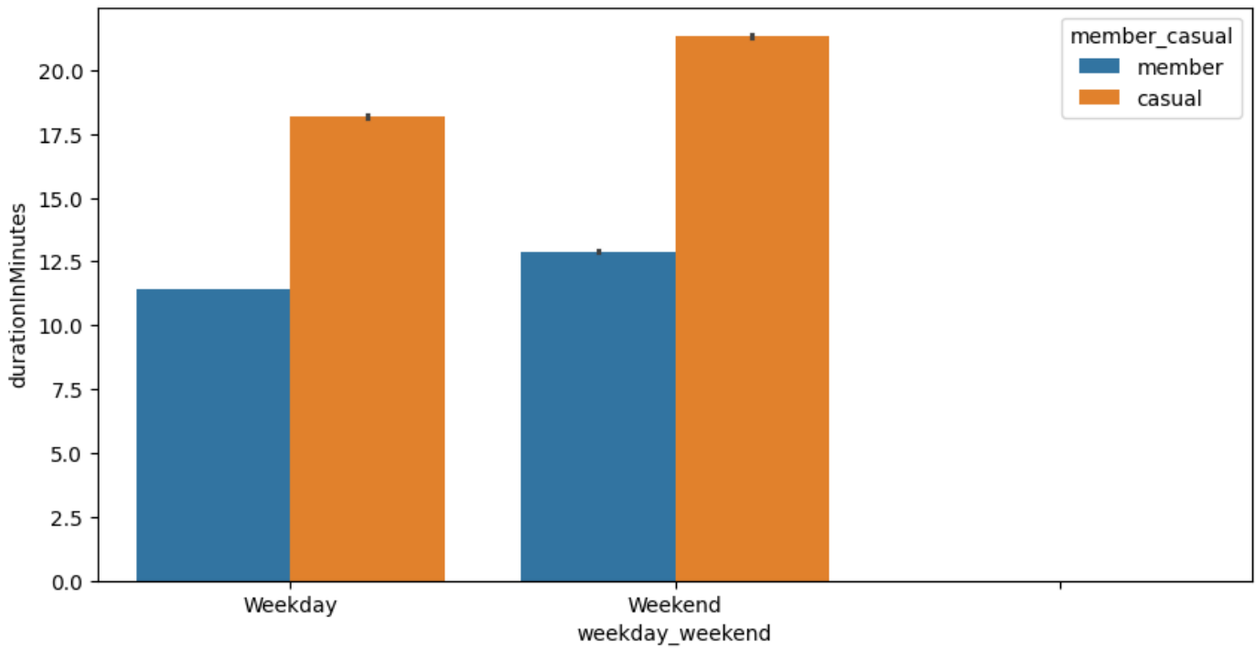
```
plt.figure(figsize=(13,6.5))

sns.barplot(x = 'weekday_weekend', y = 'durationInMinutes', data = citibike_df)

plt.title("Duration of bike rides Weekdays VS Weekends")
plt.ylabel("Average Duration")
```

```
Text(0, 0.5, 'Average Duration')
```



The average duration biked during weekends is slightly higher than weekdays which makes sense as people have more free time to bike longer and farther. To further encourage users to be active Citibike could gamify the process by providing badges when they bike for a certain amount of time or adding the ability to compare ride duration with friends.

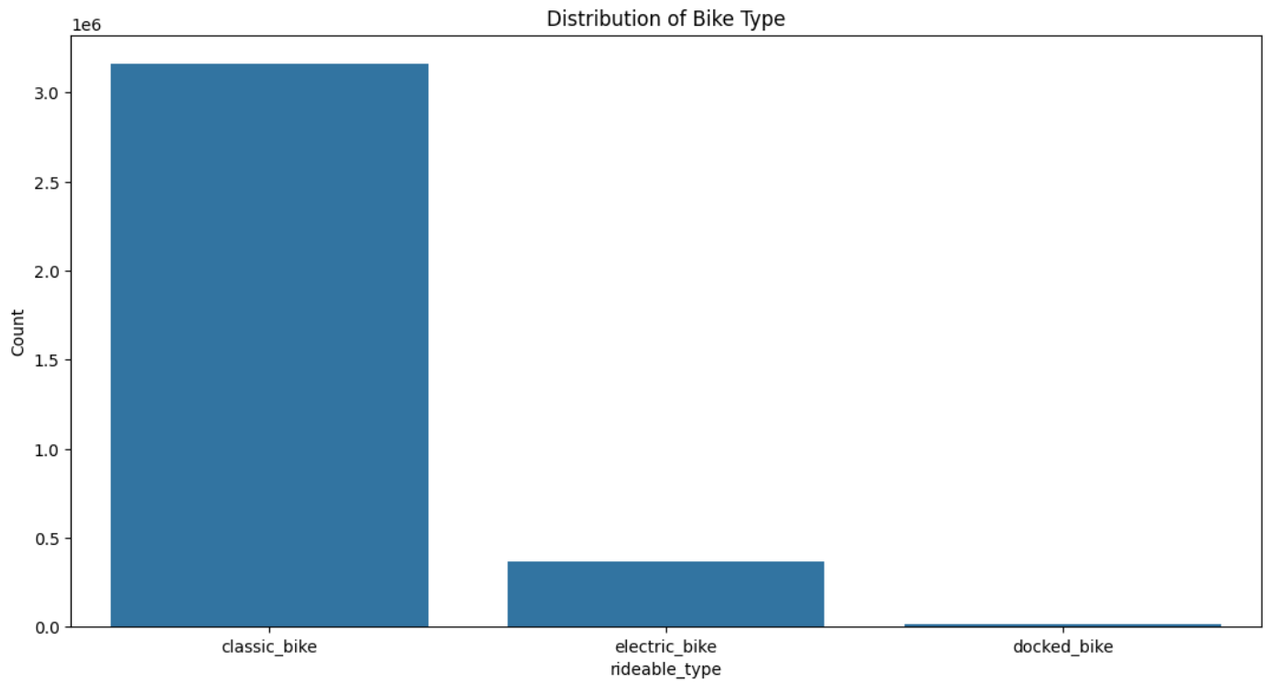
```
plt.figure(figsize=(13,6.5))
sns.barplot(x = 'weekday_weekend', y = 'durationInMinutes', data=citibike_df, hue
```



What type of Bikes are users using?

```
rideable_type = citibike_df['rideable_type'].value_counts()

plt.figure(figsize = (13,6.5))
sns.barplot(x = rideable_type.index, y = rideable_type)

plt.title("Distribution of Bike Type")
plt.ylabel("Count")
```

```
Text(0, 0.5, 'Count')
```



Majority of users use classic bikes compared to ebikes. While ebikes allow the rider to travel faster and get to their destination quicker they provide less exercise than classic bikes so its good that more users ride classic bikes.

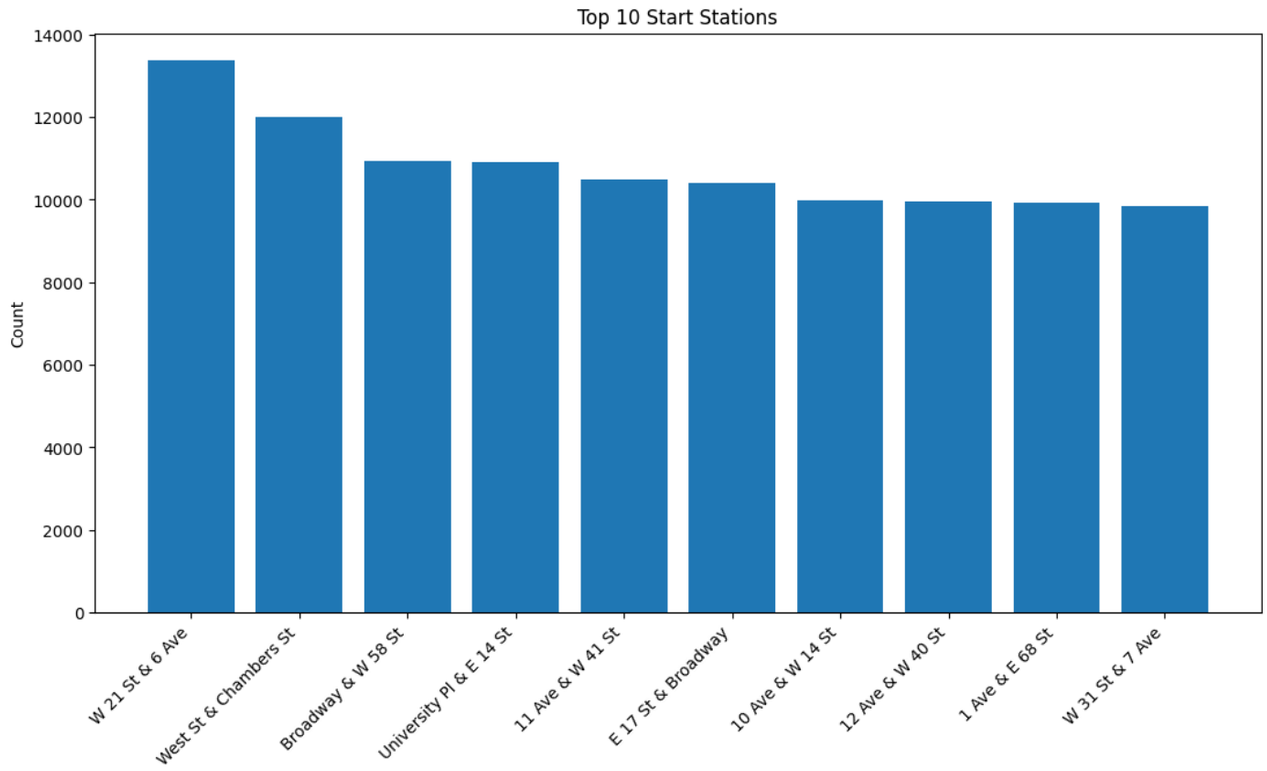
Top 10 Start Stations

```
top_start_station = citibike_df['start_station_name'].value_counts()[:15]
top_start_station
```

```
start_station_name
W 21 St & 6 Ave                13367
West St & Chambers St          12012
Broadway & W 58 St             10936
University Pl & E 14 St        10921
11 Ave & W 41 St               10499
E 17 St & Broadway             10408
10 Ave & W 14 St                9973
12 Ave & W 40 St                9963
1 Ave & E 68 St                 9922
W 31 St & 7 Ave                 9836
E 40 St & Park Ave              9734
West St & Liberty St            9421
W 22 St & 10 Ave                9247
W 30 St & 10 Ave                9232
7 Ave & Central Park South      9220
Name: count, dtype: int64
```

```
top_start_station = citibike_df['start_station_name'].value_counts()[:10]
top_start_station = top_start_station.nlargest(10)

fig,ax = plt.subplots(figsize=(13,6.5))
ax.bar(x=top_start_station.index, height=top_start_station.values)

ax.set_xticklabels(top_start_station.index, rotation = 45, ha="right")

plt.title("Top 10 Start Stations")
plt.ylabel("Count")
```

```
<ipython-input-49-cee50e16fba3>:7: UserWarning: FixedFormatter should only be
  ax.set_xticklabels(top_start_station.index, rotation = 45, ha="right")
Text(0, 0.5, 'Count')
```

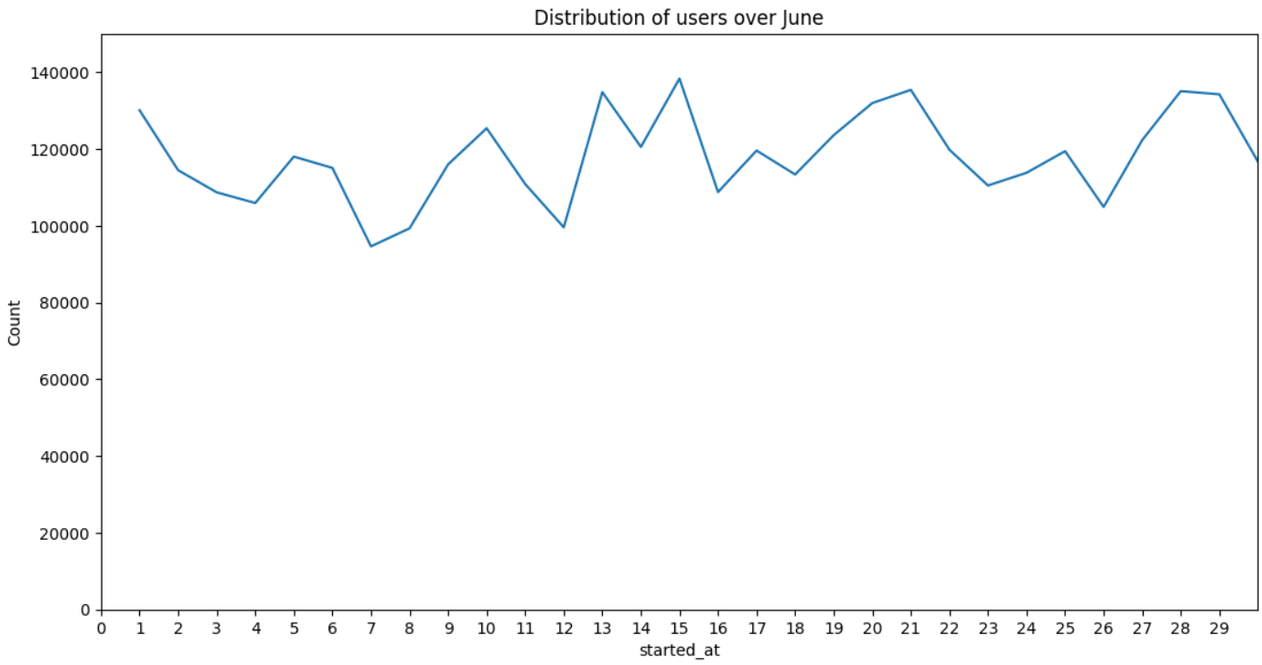


Knowing which start stations are used the most helps Citibike understand which stations need the most bikes to be stocked so everyone has access to them.

Count distribution of users over June

```
dayOfMonth = citibike_df['started_at'].dt.day
dayOfMonth_value = dayOfMonth.value_counts()
dof_df = dayOfMonth_value.to_frame()

plt.figure(figsize=(13,6.5))

ax = sns.lineplot(x = dayOfMonth_value.index, y = dayOfMonth_value, data=dof_df)

ax.set_xticks(range(30))
ax.set_xticklabels([i for i in range(0,30)])

plt.title("Distribution of users over June")
plt.ylabel("Count")

ax.set(xlim=(0, 30))
ax.set(ylim=(0, 150000))
```

```
[(0.0, 150000.0)]
```



Conclusion & Final thoughts

Compared to using the subway or a taxi, Citibikes allow users to get in a quick workout while also getting to their destination. So it's not only efficient, affordable, and better for the environment but also helps New Yorkers be more active.

However, I believe there is a chance to encourage users to be even more active. Currently there aren't as many people riding bikes during the weekends and majority of bike rides are under 15 minutes. While any amount of biking is good this could be higher.

So while Citibikes encourage exercise, to encourage New Yorkers to ride Citibikes even more they could,

```
- Provide information on bike trails in their app and information such as distance and tir
- Gamification: Provide badges when users ride a certain distance or certain time and abi
- Offer slightly cheaper prices on weekends because from the data can see that less people
eventhough they have more time
- Ensure users have proper access to bikes at stations and enough stations to return the b
- On weekdays the ride duration for casual users is very close to the 30 minute time limit
increased if not having enough bikes at the station isnt a problem so users ride for longe
the bike on time
```

This would not only encourage being active but also increase Citibike's revenue if people ride bikes more often