# Project Development Phase – Sprint 3

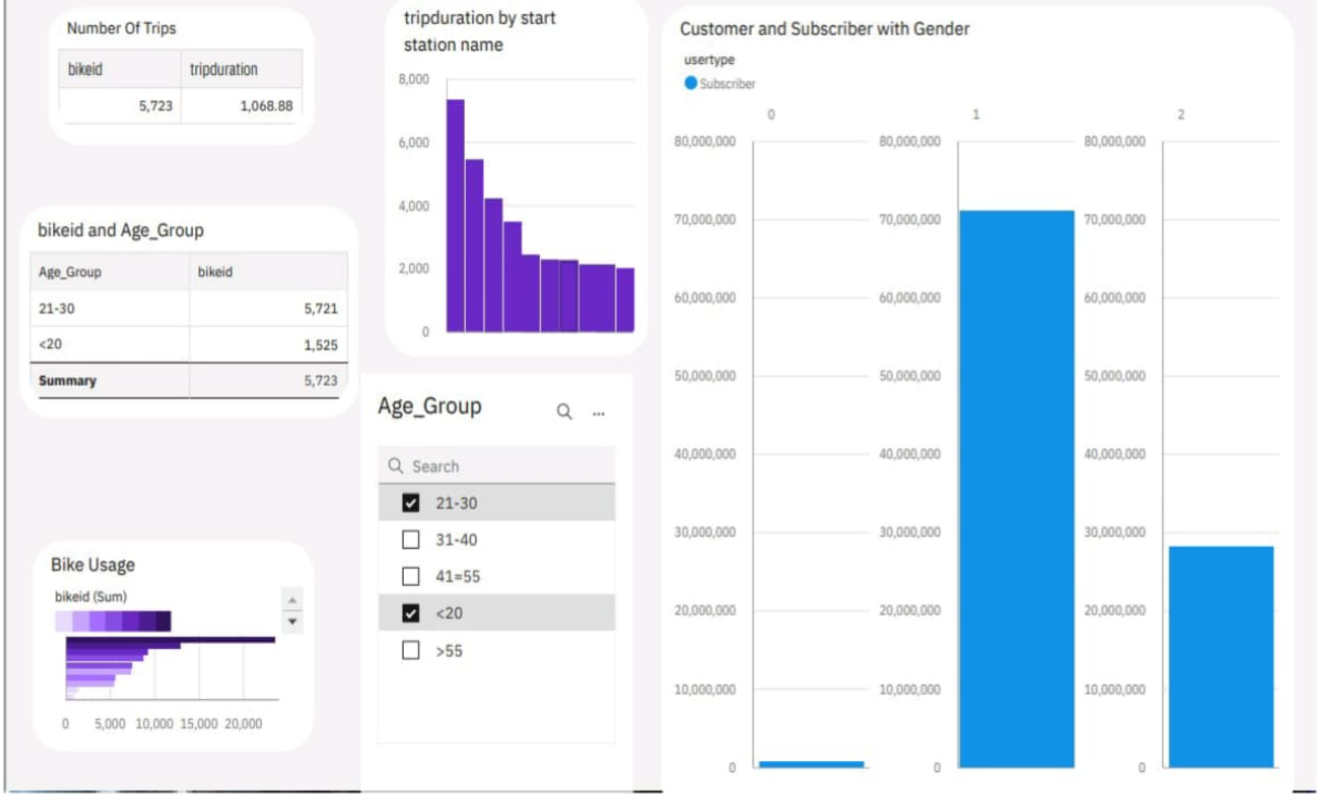
| | |
|---|---|
| Date | 07 November 2022 |
| Team ID | PNT2022TMID45599 |
| Project Name | A new hint to transportation – Analysis of the NYC bike share system. |
| Maximum Marks | 20 Marks |

Creating a dashboard including all the visualizations created in the cognos platform:

This dashboard has the charts including

i) Number of trips ii) Customer and
Subscriber percentage with gender iii) Bike
Usage iv) BikeId and Age Group

v) Trip duration by start station name



Number Of Trips

| bikeid | tripduration |
|---|---|
| 5,723 | 1,068.88 |

bikeid and Age_Group

| Age_Group | bikeid |
|---|---|
| 21-30 | 5,721 |
| <20 | 1,525 |
| **Summary** | 5,723 |

Age_Group

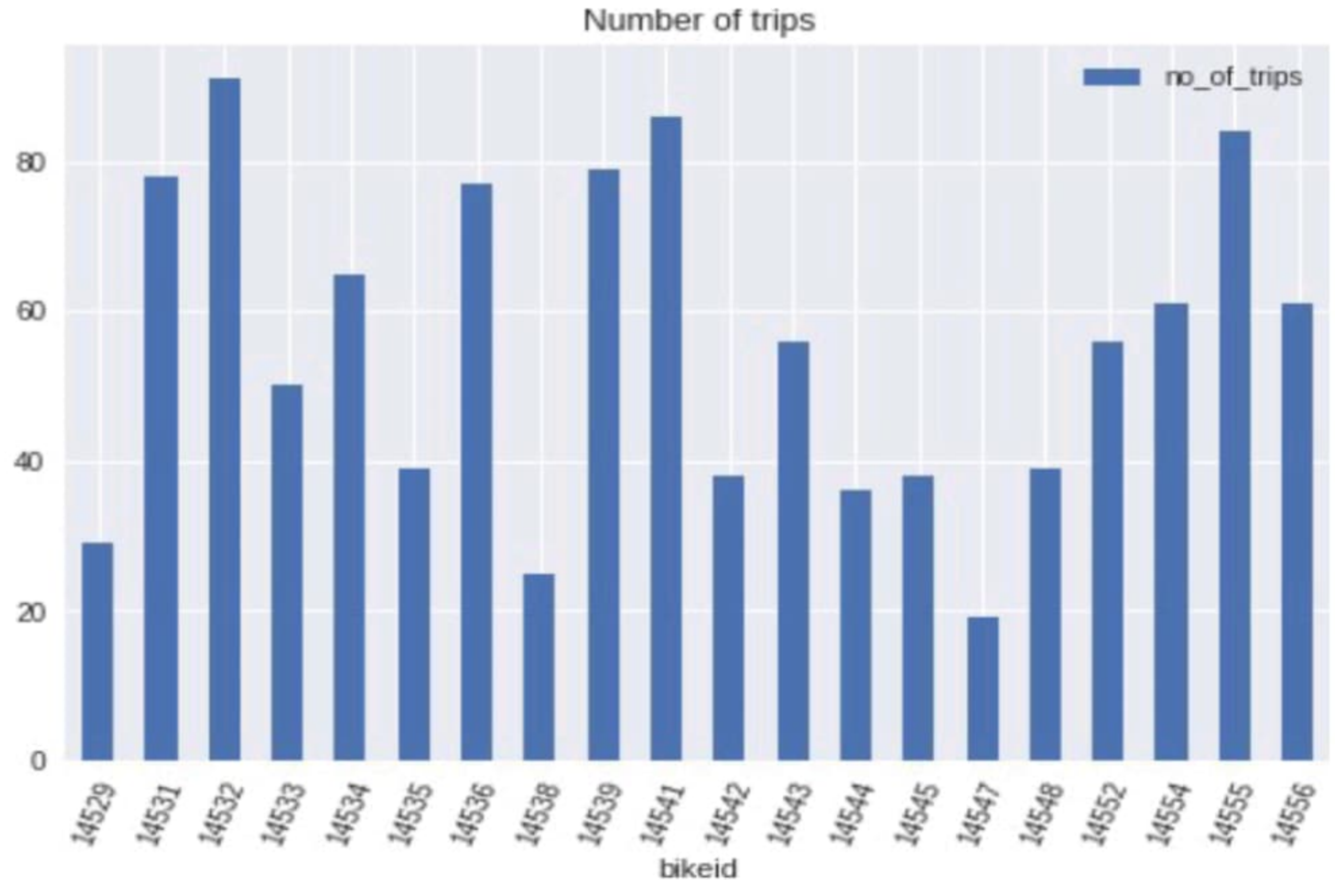
Search

☑ 21-30
□ 31-40
□ 41=55
☑ <20
□ >55

## Visualization Charts using Python:

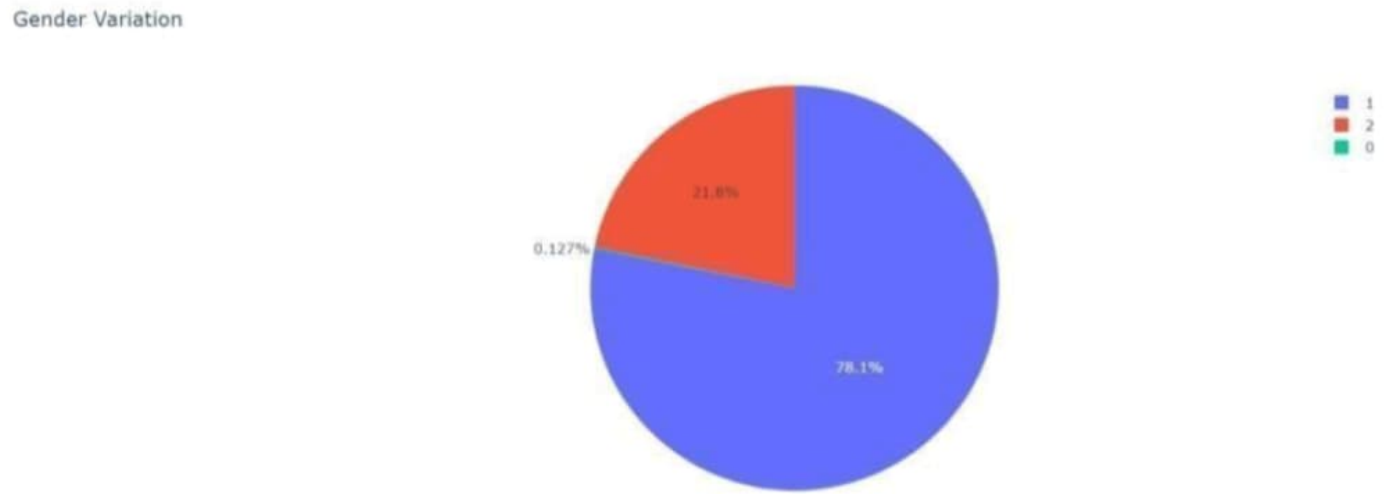
**Finding the number of trips per bike:**

```
trips = pd.DataFrame() #creating a

dataframe

trips['no_of_trips'] = df.groupby("bikeid")["bikeid"].count() #finding the number of trips by

each bike trips['avg_duration'] = df.groupby("bikeid")["tripduration"].mean() #avg duration of

the trips trips_graph=trips.head(20)


trips_graph.plot.bar(x="bikeid", y="no_of_trips", rot=70, title="Number of trips")
```

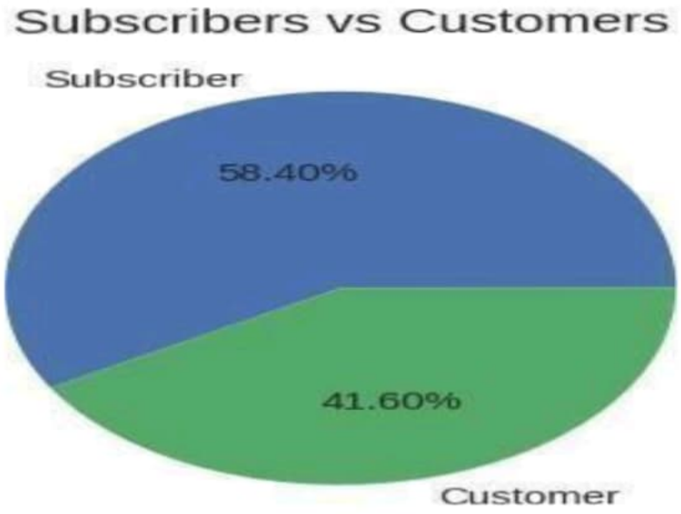

Number of trips
no_of_trips
0
20
40
60
80
14529
14531
14532
14533
14534
14535
14536
14538
14539
14541
14542
14543
14544
14545
14547
14548
14552
14554
14555
14556
bikeid

**Gender Variation:**

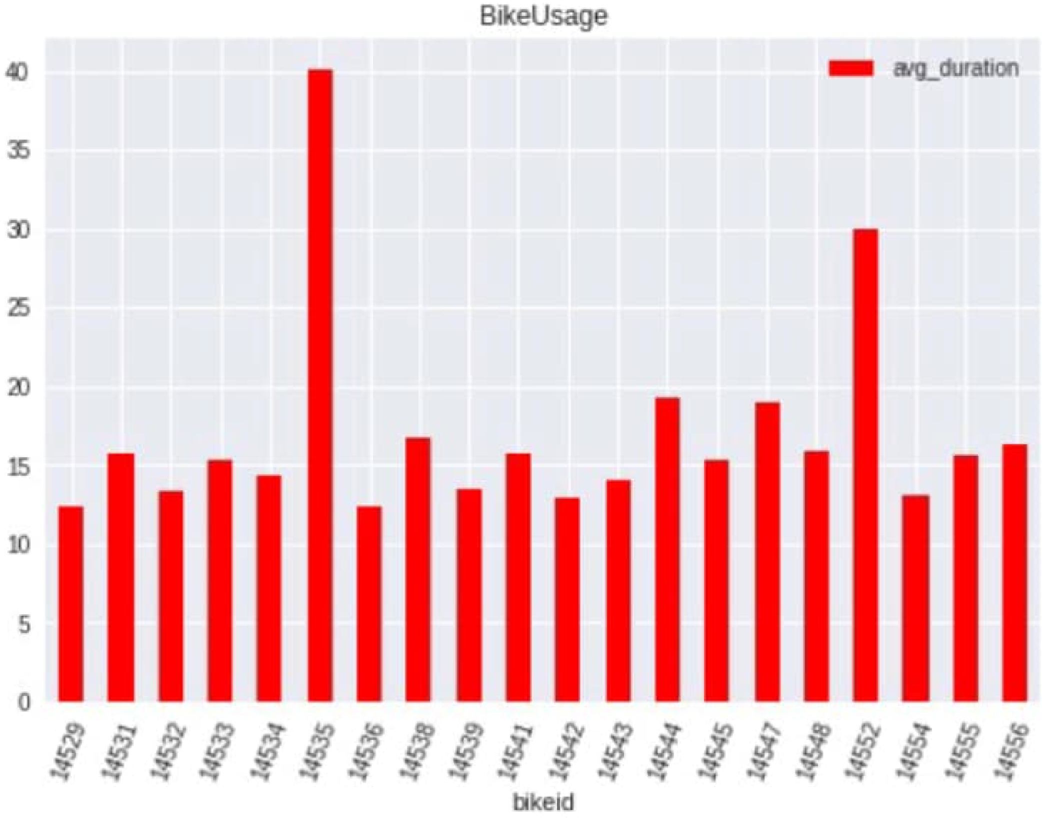
```
plt.pie(values =
df_bike['Gender'].value_counts(), names
=df_bike['Gender'].value_counts().index, title
="Gender Variation")
```


Gender Variation
21.8%
0.127%
78.1%
1
2
0

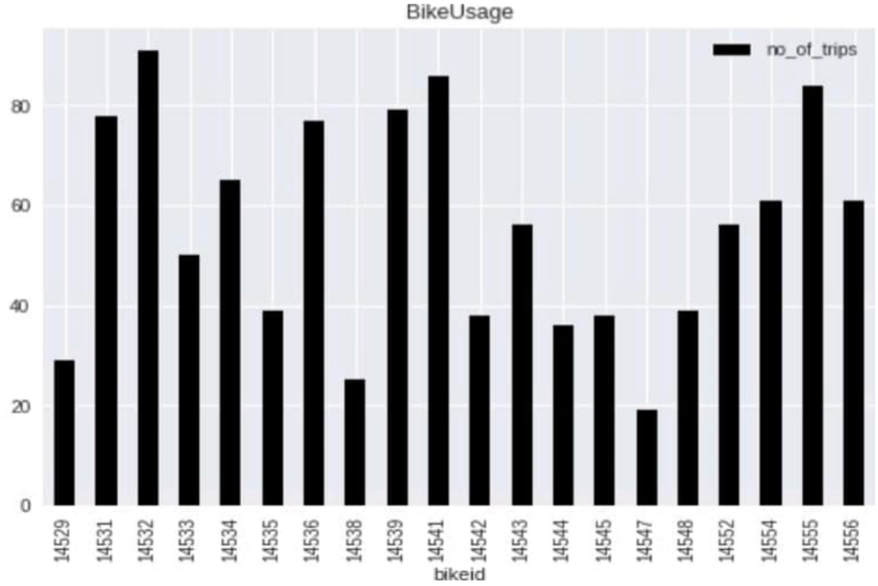
**Percentage of Subscribers and Customers:**



**Bike Usage Based on Average Duration:** trips_graph.plot.bar(x="bikeid", y="avg_duration", rot=70, title="BikeUsage",color="red")

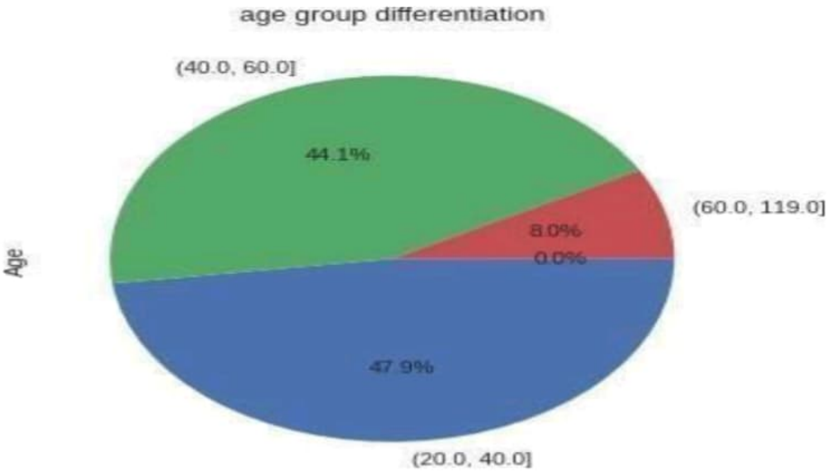

BikeUsage
avg_duration
0
5
10
15
20
25
30
35
40
14529
14531
14532
14533
14534
14535
14536
14538
14539
14541
14542
14543
14544
14545
14547
14548
14552
14554
14555
14556
bikeid

**Bike Usage Based on No of Trips:** trips_graph.plot.bar(x="bikeid", y="no_of_trips", rot=90, title="BikeUsage",color="black")



## Age Group Differentiation:

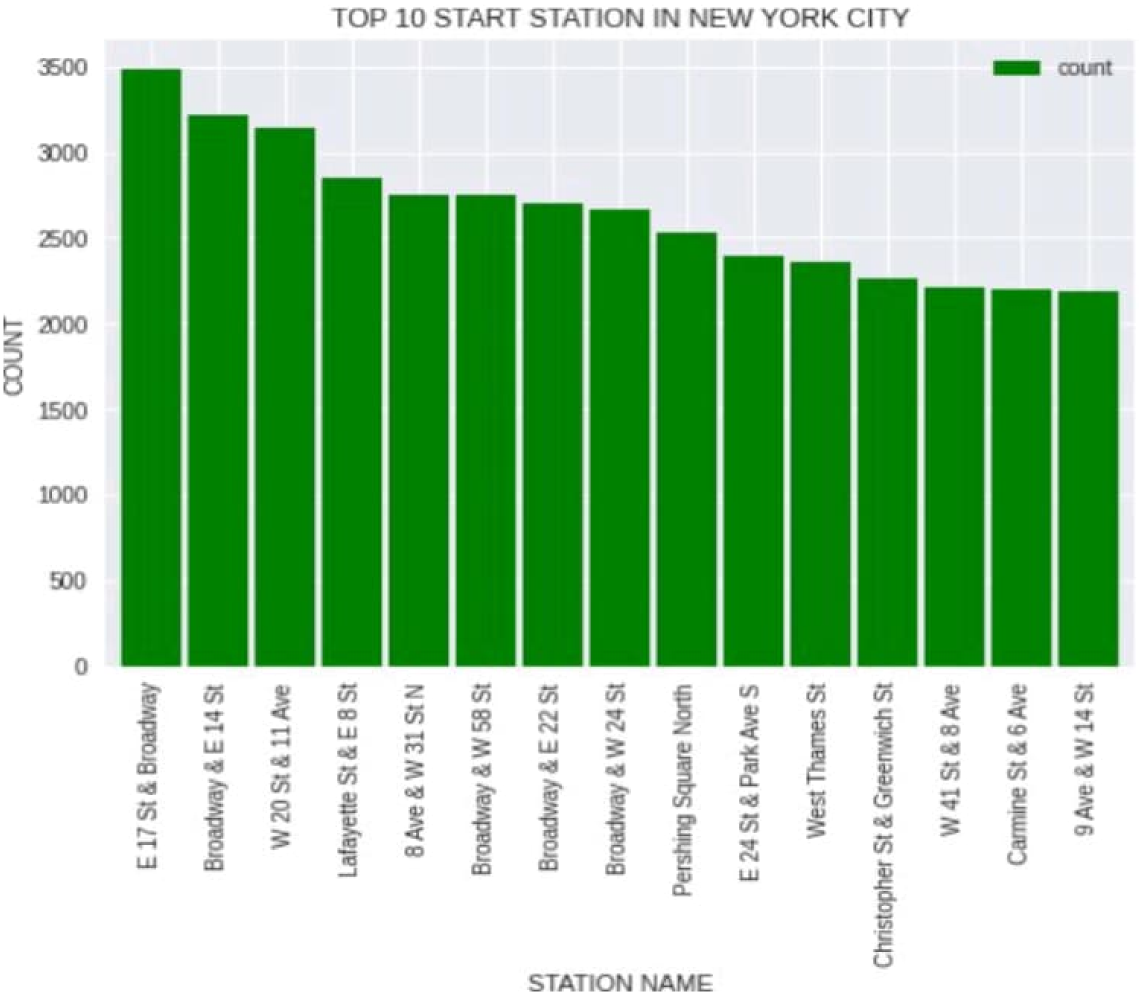
```
agegroup = pd.cut(df['Age'], bins=bins).value_counts()

agegroup.plot.pie(autopct="%.1f%%",title='age group differentiation',counterclock=False);
```



**Top 10 Start Station:**

```
most=pd.DataFrame()

most_graph=pd.DataFrame()

most['name']=df["start station name"].value_counts().index

most['count']=df["start station name"].value_counts().values

most_graph=most.head(15)
```

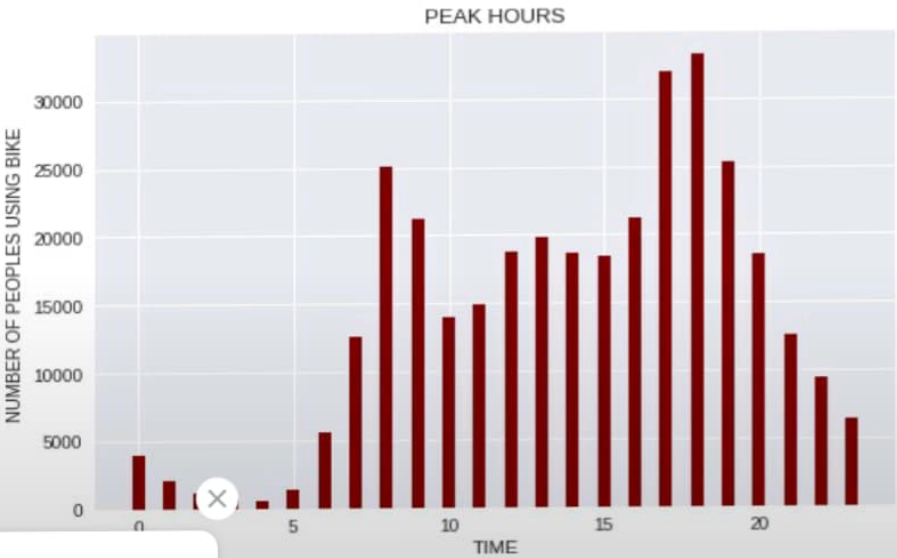
```
most_graph.plot.bar(x="name", y="count", width=0.9,rot=90,
title="BikeUsage",color="green") plt.xlabel("STATION NAME") plt.ylabel("COUNT")
plt.title("TOP 10 START STATION IN NEW YORK CITY") plt.show()
```



TOP 10 START STATION IN NEW YORK CITY
count
COUNT
3500
3000
2500
2000
1500
1000
500
0
E 17 St & Broadway
Broadway & E 14 St
W 20 St & 11 Ave
Lafayette St & E 8 St
8 Ave & W 31 St N
Broadway & W 58 St
Broadway & E 22 St
Broadway & W 24 St
Pershing Square North
E 24 St & Park Ave S
West Thames St
Christopher St & Greenwich St
W 41 St & 8 Ave
Carmine St & 6 Ave
9 Ave & W 14 St
STATION NAME

**Finding the Peak Hours of Travel:**

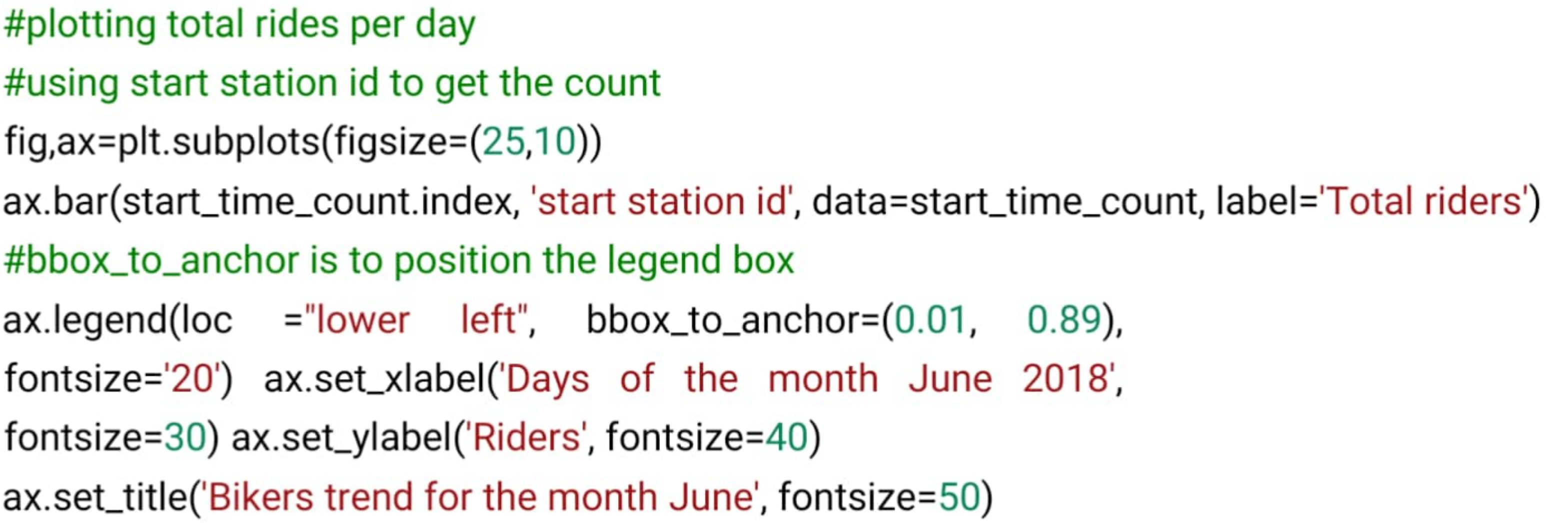
```
ind=peak_hour["Hour"].value_counts().index

y=peak_hour["Hour"].value_counts().values

plt.bar(ind, y, color ='maroon', width = 0.4)

plt.xlabel("TIME")

plt.ylabel("NUMBER OF PEOPLES USING BIKE")

plt.title("PEAK HOURS") plt.show()
```


PEAK HOURS
NUMBER OF PEOPLES USING BIKE
30000
25000
20000
15000
10000
5000
0
0
5
10
15
20
TIME

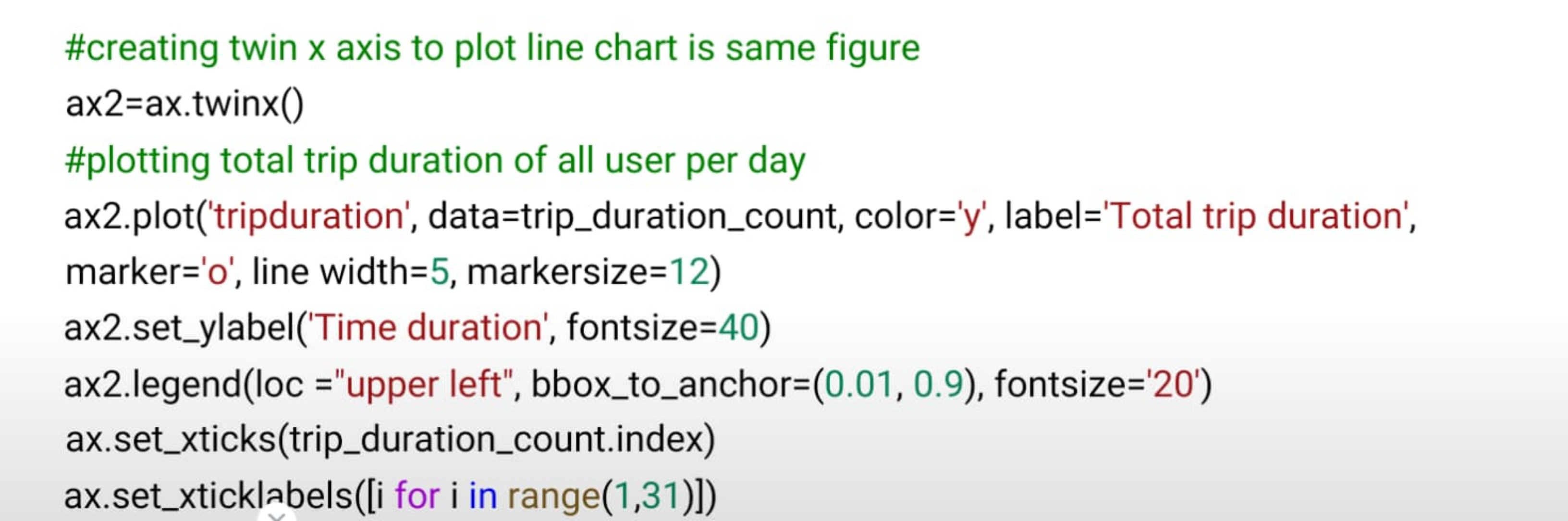
**Bike Trend for the month June:**

```
#converting string to datetime object
df['starttime']= pd.to_datetime(df['starttime'])

#since we are dealing with single month, we grouping by
days
#using count aggregation to get number of occurances i.e, total trips per day
start_time_count = df.set_index('starttime').groupby(pd.Grouper(freq='D')).count()

#we have data from July month for only one day which is at last row, lets drop it
start_time_count.drop(start_time_count.tail(1).index, axis=0, inplace=True)
```

```
#again grouping by day and aggregating with sum to get total trip duration per day
#which will used while plotting
trip_duration_count = df.set_index('starttime').groupby(pd.Grouper(freq='D')).sum()

#again dropping the last row for same reason
trip_duration_count.drop(trip_duration_count.tail(1).index, axis=0, inplace=True)
```

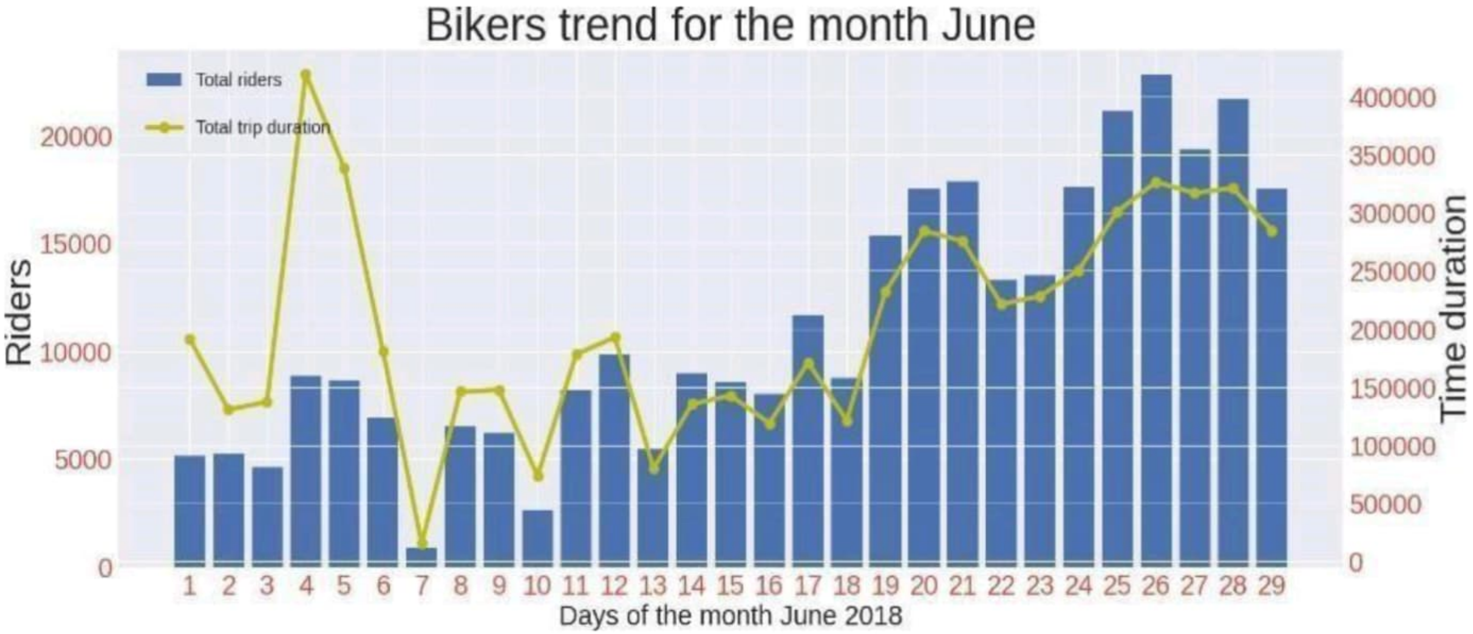


```
#plotting total rides per day
#using start station id to get the count
fig,ax=plt.subplots(figsize=(25,10))
ax.bar(start_time_count.index, 'start station id', data=start_time_count, label='Total riders')
#bbox_to_anchor is to position the legend box
ax.legend(loc    ="lower    left",    bbox_to_anchor=(0.01,    0.89),
fontsize='20')   ax.set_xlabel('Days   of   the   month   June   2018',
fontsize=30) ax.set_ylabel('Riders', fontsize=40)
ax.set_title('Bikers trend for the month June', fontsize=50)
```

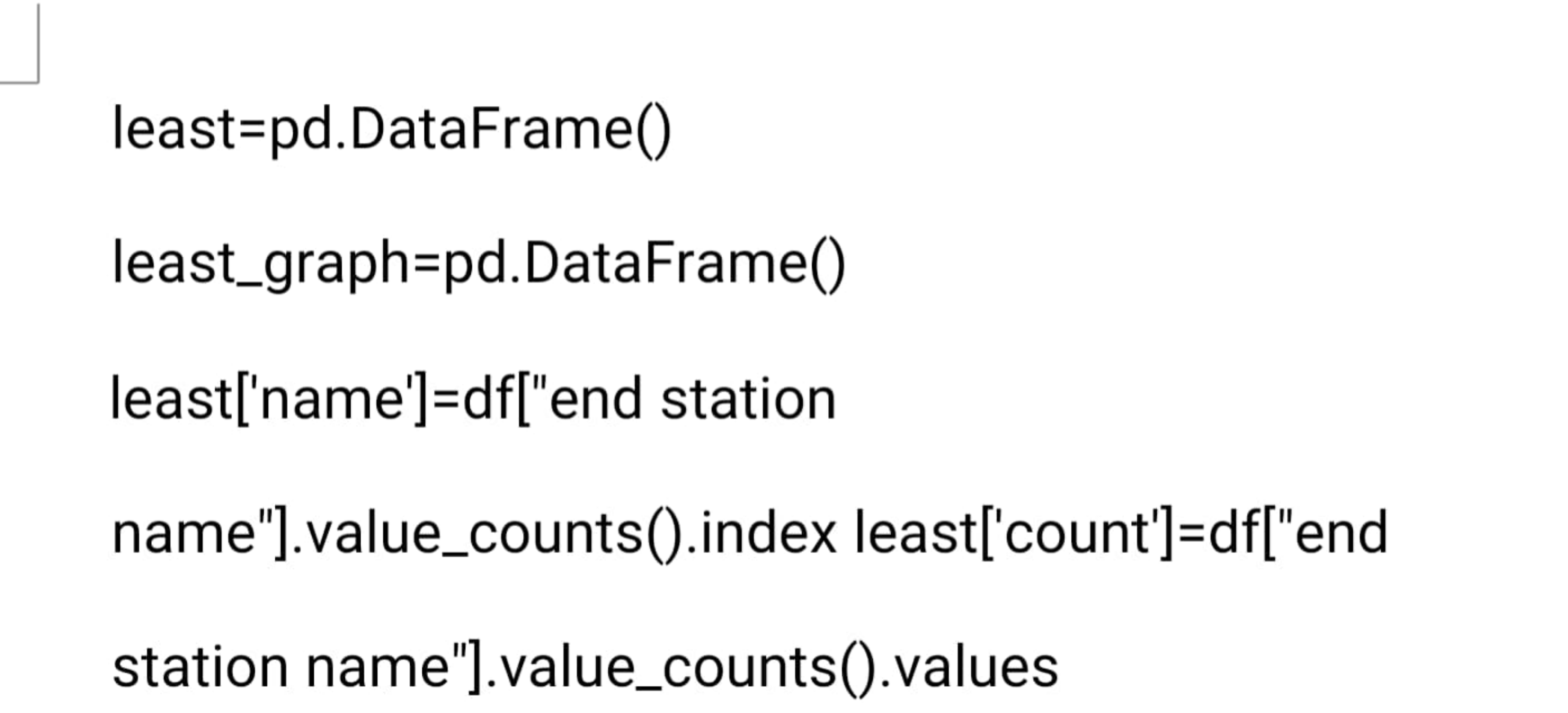


```
#creating twin x axis to plot line chart is same figure
ax2=ax.twinx()
#plotting total trip duration of all user per day
ax2.plot('tripduration', data=trip_duration_count, color='y', label='Total trip duration',
marker='o', line width=5, markersize=12)
ax2.set_ylabel('Time duration', fontsize=40)
ax2.legend(loc ="upper left", bbox_to_anchor=(0.01, 0.9), fontsize='20')
ax.set_xticks(trip_duration_count.index)
ax.set_xticklabels([i for i in range(1,31)])
```

```
#tweeking x and y ticks labels of axes1
ax.tick_params(labelsize=30, labelcolor='#eb4034')
#tweeking x and y ticks labels of axes2
ax2.tick_params(labelsize=30, labelcolor='#eb4034')


plt.show()
```

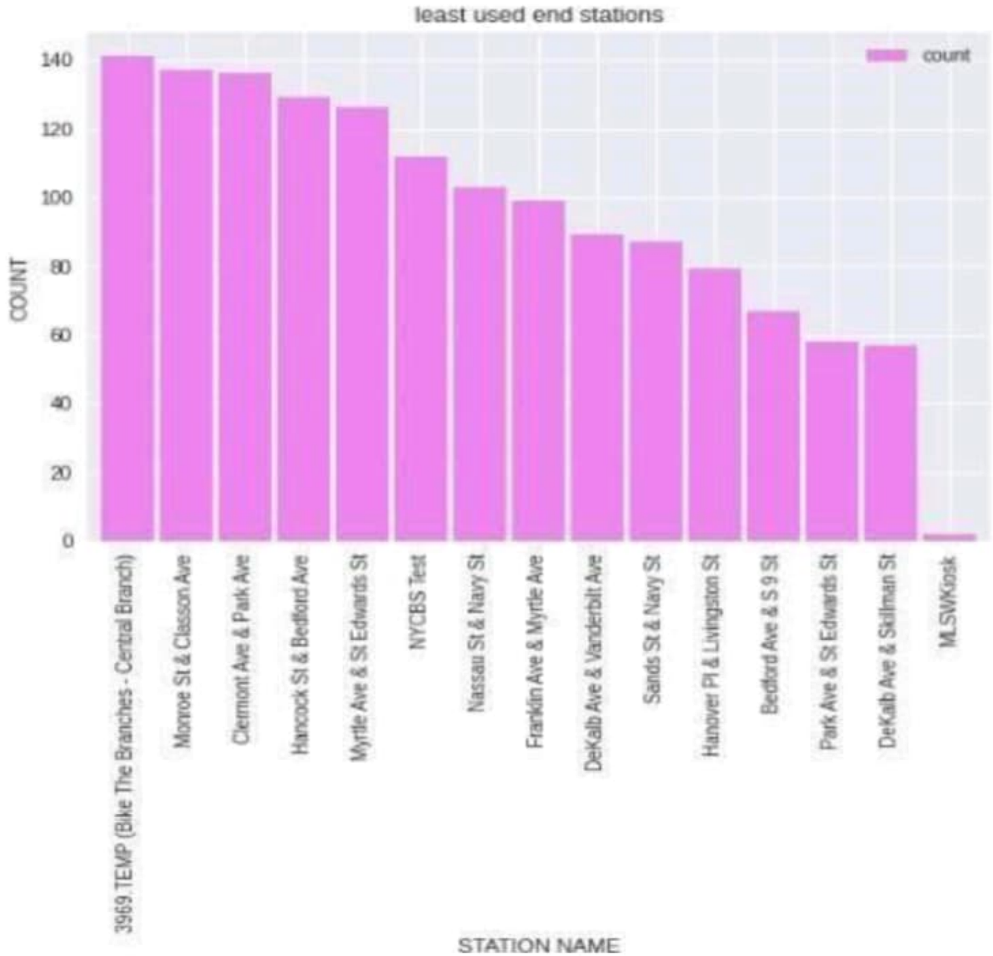

Bikers trend for the month June
Total riders
Total trip duration
Riders
0
5000
10000
15000
20000
Time duration
0
50000
100000
150000
200000
250000
300000
350000
400000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
Days of the month June 2018

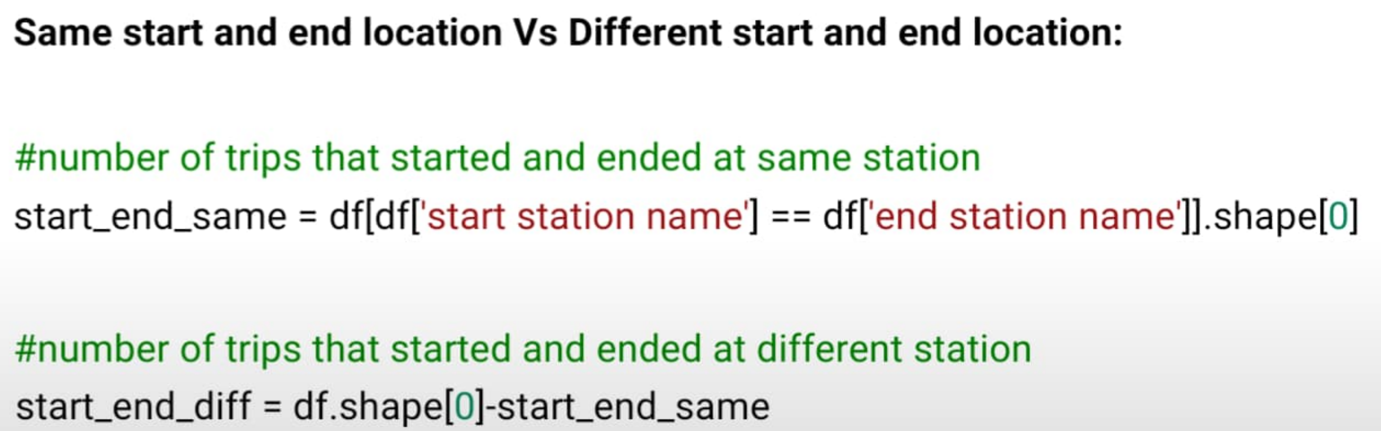
## Least Used End Stations:



```
least=pd.DataFrame()

least_graph=pd.DataFrame()

least['name']=df["end station

name"].value_counts().index least['count']=df["end

station name"].value_counts().values
```

```
least_graph=most.tail(15) least_graph

least_graph.plot.bar(x="name",                    y="count",

title="BikeUsage",color="violet") plt.xlabel("STATION NAME")

plt.title("least used end stations")

plt.show()
```


least used end stations
count
COUNT
0
20
40
60
80
100
120
140
3969 TEMP (Bike The Branches - Central Branch)
Monroe St & Classon Ave
Clermont Ave & Park Ave
Hancock St & Bedford Ave
Myrtle Ave & St Edwards St
NYCBS Test
Nassau St & Navy St
Franklin Ave & Myrtle Ave
DeKalb Ave & Vanderbilt Ave
Sands St & Navy St
Hanover Pl & Livingston St
Bedford Ave & S 9 St
Park Ave & St Edwards St
DeKalb Ave & Skillman St
MLSWKiosk
STATION NAME

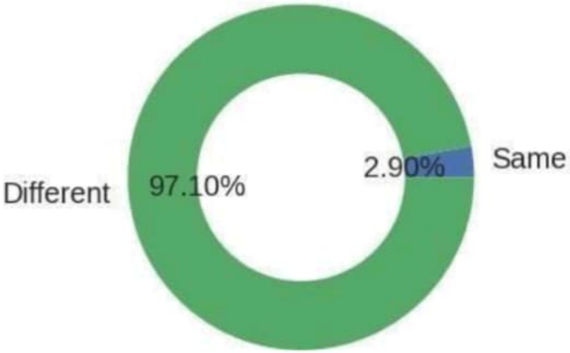


**Same start and end location Vs Different start and end location:**

```
#number of trips that started and ended at same station
start_end_same = df[df['start station name'] == df['end station name']].shape[0]

#number of trips that started and ended at different station
start_end_diff = df.shape[0]-start_end_same
```

```
fig,ax=plt.subplots()
ax.pie([start_end_same,start_end_diff], labels=['Same', 'Different'], autopct='%1.2f%%',
textprops={'f ontsize': 20})
ax.set_title('Same start and end location vs Different start and end location', fontsize=20)


circle = Circle((0,0), 0.6, facecolor='white')
ax.add_artist(circle)

plt.show()
```

Same start and end location vs Different start and end location


Same
Different