Insight for Cab Investment firm

Business problem:

XYZ is a private firm in US. Due to remarkable growth in the Cab Industry in last few years and multiple key players in the market, it is planning for an investment in Cab industry and as per their Go-to-Market(G2M) strategy they want to understand the market before taking final decision.

Properties of the data provided (data intake report):

After merging 4 csv files, the final dataset contains 3,59,392 rows and 22 columns containing information of 2 cab services from 19 cities.

Steps taken in order to create an applicable data set:

1. Merged Cab and City data on 'City' column.
2. Merged Customer and Transaction data on 'Transaction_ID'.
3. Finally merged the above two data on 'Transaction_ID'.

Steps taken perform analysis:

1. Convert 'Date of Travel' column into pandas datetime column and set it as the index
2. Created new columns to better analyze the trend.
3. EDA
4. Hypothesis Testing

Type of analysis performed:

1. Univariate Analysis
2. Bivariate Analysis
3. Time series Analysis

Assumptions made:

1. Outliers are present in "Price Charged" feature but due to unavailability of trip duration details, we are not treating this as outlier.
2. Profit of rides are calculated keeping other factors constant and only "Price Charged" and "Cost of Trip" features used to calculate profit.
3. Users feature of city dataset is treated as number of cab users in the city.

Data Collection

Import Libraries & set default style

In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib

sns.set_style('darkgrid')
matplotlib.rcParams['font.size'] = 14
matplotlib.rcParams['figure.figsize'] = (15, 9)
matplotlib.rcParams['figure.facecolor'] = '#00000000'
import matplotlib.pyplot as plt
%matplotlib inline

import warnings
warnings.filterwarnings("ignore")
```

Read csv files

In [2]:

```
cab = pd.read_csv("Cab_Data.csv")
city = pd.read_csv("City.csv")
customer = pd.read_csv("Customer_ID.csv")
transaction = pd.read_csv("Transaction_ID.csv")
```

Merge into one dataframe

In [3]:

```
df_1 = pd.merge(cab, city, on="City")
df_2 = pd.merge(customer, transaction, on="Customer ID")
df = pd.merge(df_1, df_2, on="Transaction ID")
```

Create new columns

In [4]:

```
df['Date of Travel'] = pd.to_datetime(df['Date of Travel'])
df['Day'] = df['Date of Travel'].dt.day
df['Weekday'] = df['Date of Travel'].dt.weekday
df['Month'] = df['Date of Travel'].dt.month
df['Year'] = df['Date of Travel'].dt.year
df['Profit'] = df['Price Charged'] - df['Cost of Trip']
df['Profit Percentage per Trip'] = ((df['Profit'] / df['Cost of Trip'])*100).round(2)
df['Profit per KM'] = ((df['Profit'] / df['KM Travelled']))

df['Population'] = df['Population'].str.replace(',', '').astype(float)
df['Users'] = df['Users'].str.replace(',', '').astype(float)
df['Users Density'] = df['Users'] / df['Population']
```

In [5]:

```
df.sort_values(by='Date of Travel', inplace=True)
df.set_index('Date of Travel', inplace=True)
```

Data Exploration

In [6]:

```
pd.set_option("display.max_columns", 25)
df
```

Out[6]:

	Transaction ID	Company	City	KM Travelled	Price Charged	Cost of Trip	Population	Users
Date of Travel								
2016-01-02	10004899	Yellow Cab	LOS ANGELES CA	25.53	402.89	327.8052	1595037.0	144132
2016-01-02	10005402	Yellow Cab	WASHINGTON DC	44.08	694.53	587.1456	418859.0	127001
2016-01-02	10004271	Pink Cab	BOSTON MA	38.61	358.05	405.4050	248968.0	80021
2016-01-02	10004399	Pink Cab	SAN DIEGO CA	4.72	50.88	51.9200	959307.0	69995
2016-01-02	10005419	Yellow Cab	WASHINGTON DC	46.00	765.04	552.0000	418859.0	127001
...	...	...	...	...	...	...	...	
2018-12-31	10435303	Yellow Cab	NEW YORK NY	39.20	1000.88	508.0320	8405837.0	302149
2018-12-31	10435591	Yellow Cab	NEW YORK NY	37.74	918.58	511.7544	8405837.0	302149
2018-12-31	10434932	Yellow Cab	LOS ANGELES CA	22.88	396.35	315.7440	1595037.0	144132
2018-12-31	10437814	Yellow Cab	BOSTON MA	17.10	238.07	240.0840	248968.0	80021
2018-12-31	10438259	Yellow Cab	DALLAS TX	34.00	635.45	428.4000	942908.0	22157

359392 rows × 21 columns

In [7]:

```
df.shape
```

Out[7]:

```
(359392, 21)
```

Final dataset contains 3,59,392 rows & 21 columns

In [8]:

```
df.drop(['Transaction ID', 'Customer ID'], axis=1, inplace=True)
```

Get some statistical values of each Numerical colums

In [9]:

```
df.describe()
```

Out[9]:

	KM Travelled	Price Charged	Cost of Trip	Population	Users	A
count	359392.000000	359392.000000	359392.000000	3.593920e+05	359392.000000	359392.0000
mean	22.567254	423.443311	286.190113	3.132198e+06	158365.582267	35.3367
std	12.233526	274.378911	157.993661	3.315194e+06	100850.051020	12.5942
min	1.900000	15.600000	19.000000	2.489680e+05	3643.000000	18.0000
25%	12.000000	206.437500	151.200000	6.712380e+05	80021.000000	25.0000
50%	22.440000	386.360000	282.480000	1.595037e+06	144132.000000	33.0000
75%	32.960000	583.660000	413.683200	8.405837e+06	302149.000000	42.0000
max	48.000000	2048.030000	691.200000	8.405837e+06	302149.000000	65.0000

Since there is no null value and also we can see that the minimum and maximum km travelled, price and cost are all valid values so no need to drop any rows from the dataset

Get type, null-value count

In [10]:

```
df.info();
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 359392 entries, 2016-01-02 to 2018-12-31
Data columns (total 19 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0   Company                     359392 non-null  object
 1   City                        359392 non-null  object
 2   KM Travelled                359392 non-null  float64
 3   Price Charged               359392 non-null  float64
 4   Cost of Trip                359392 non-null  float64
 5   Population                  359392 non-null  float64
 6   Users                       359392 non-null  float64
 7   Gender                      359392 non-null  object
 8   Age                         359392 non-null  int64
 9   Income (USD/Month)          359392 non-null  int64
 10  Payment_Mode                359392 non-null  object
 11  Day                         359392 non-null  int64
 12  Weekday                     359392 non-null  int64
 13  Month                       359392 non-null  int64
 14  Year                        359392 non-null  int64
 15  Profit                      359392 non-null  float64
 16  Profit Percentage per Trip  359392 non-null  float64
 17  Profit per KM               359392 non-null  float64
 18  Users Density               359392 non-null  float64
dtypes: float64(9), int64(6), object(4)
memory usage: 54.8+ MB
```

There are no missing values

Check Duplicate Rows if any

In [11]:

```
duplicate = df[df.duplicated()]
duplicate
```

Out[11]:

	Company	City	KM Travelled	Price Charged	Cost of Trip	Population	Users	Gender	Age	Income (USD/Month
Date of Travel										

There are no duplicate rows!

Find unique values of each column

In [12]:

```
df.nunique()
```

Out[12]:

```
Company                          2
City                            19
KM Travelled                   874
Price Charged                99176
Cost of Trip                 16291
Population                      19
Users                           19
Gender                           2
Age                             48
Income (USD/Month)           22725
Payment_Mode                     2
Day                             31
Weekday                          7
Month                           12
Year                             3
Profit                      303907
Profit Percentage per Trip   21939
Profit per KM               356133
Users Density                   19
dtype: int64
```

There are 2 cab service provider in 19 different cities

City with highest no. of running cabs

In [13]:

```
df['City'].value_counts()
```

Out[13]:

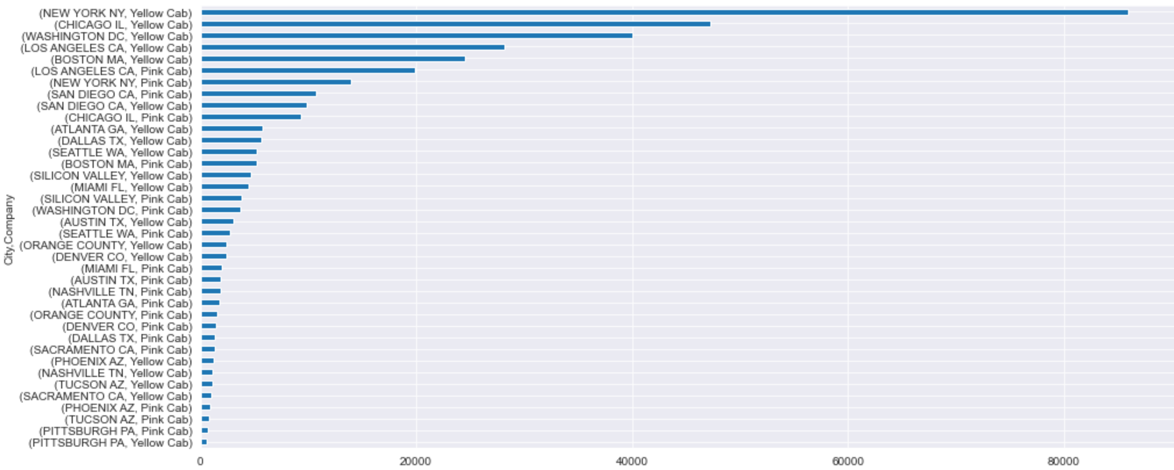
```
NEW YORK NY       99885
CHICAGO IL        56625
LOS ANGELES CA    48033
WASHINGTON DC     43737
BOSTON MA         29692
SAN DIEGO CA      20488
SILICON VALLEY     8519
SEATTLE WA         7997
ATLANTA GA         7557
DALLAS TX          7017
MIAMI FL           6454
AUSTIN TX          4896
ORANGE COUNTY      3982
DENVER CO          3825
NASHVILLE TN       3010
SACRAMENTO CA      2367
PHOENIX AZ         2064
TUCSON AZ          1931
PITTSBURGH PA      1313
Name: City, dtype: int64
```

New York City count in the dataset is the highest which may imply more no. of cabs are running in this city. This may be due to high population also.

Demand of the 2 cab service providers in each city

In [14]:

```
plt.figure(figsize=(15, 7))
df.groupby('City').Company.value_counts().sort_values(ascending=True).plot(kind='barh');
```



Yellow Cabs are dominating in most of the cities

In [15]:

```
city_grp = df.groupby('City')
city_grp['Company'].value_counts().unstack()
```

Out[15]:

Company	Pink Cab	Yellow Cab
City		
ATLANTA GA	1762	5795
AUSTIN TX	1868	3028
BOSTON MA	5186	24506
CHICAGO IL	9361	47264
DALLAS TX	1380	5637
DENVER CO	1394	2431
LOS ANGELES CA	19865	28168
MIAMI FL	2002	4452
NASHVILLE TN	1841	1169
NEW YORK NY	13967	85918
ORANGE COUNTY	1513	2469
PHOENIX AZ	864	1200
PITTSBURGH PA	682	631
SACRAMENTO CA	1334	1033
SAN DIEGO CA	10672	9816
SEATTLE WA	2732	5265
SILICON VALLEY	3797	4722
TUCSON AZ	799	1132
WASHINGTON DC	3692	40045

People prefer Yellow Cabs over Pink Cabs in every city except these 4:

1. Nashville

2. Pittsburgh

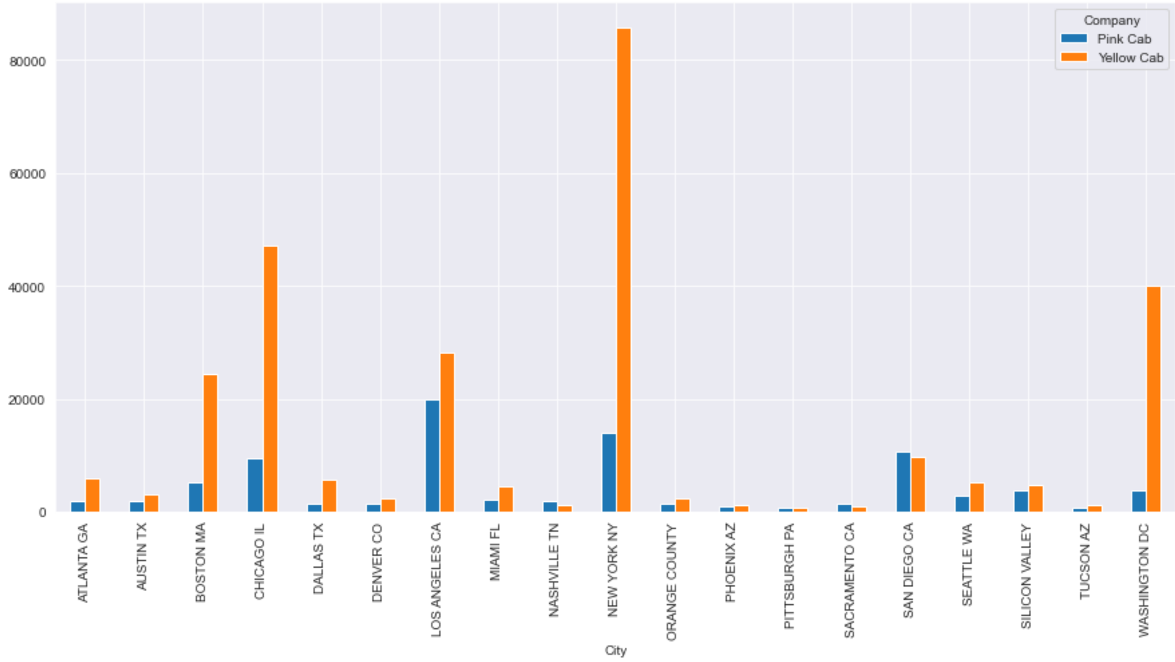
3. Sacramento

4. San Diego

Visual Comparison:

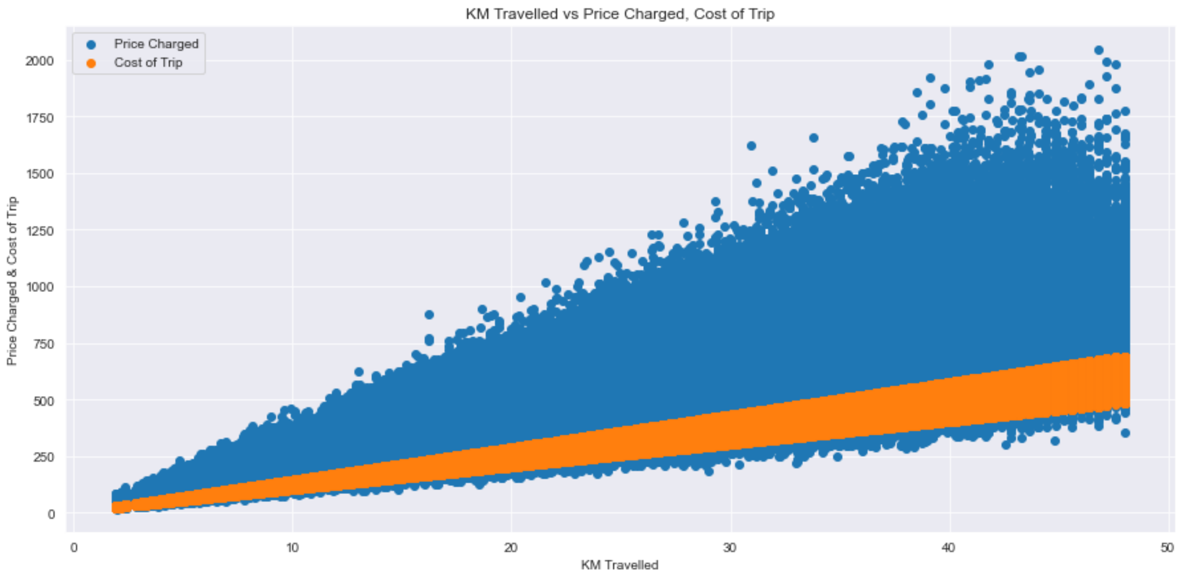
In [16]:

```
city_grp['Company'].value_counts().unstack().plot(kind='bar', figsize=(15, 7));
```



In [17]:

```
fig, ax = plt.subplots(figsize=(15,7))
ax.scatter(x = df['KM Travelled'], y = df['Price Charged']);
ax.scatter(x = df['KM Travelled'], y = df['Cost of Trip']);
plt.xlabel("KM Travelled")
plt.ylabel("Price Charged & Cost of Trip")
plt.title("KM Travelled vs Price Charged, Cost of Trip")
ax.legend(['Price Charged', 'Cost of Trip'])
plt.show()
```



As the distance increases, both cost and price increases linearly but the difference becomes more pronounced

Profit per KM City wise

In [18]:

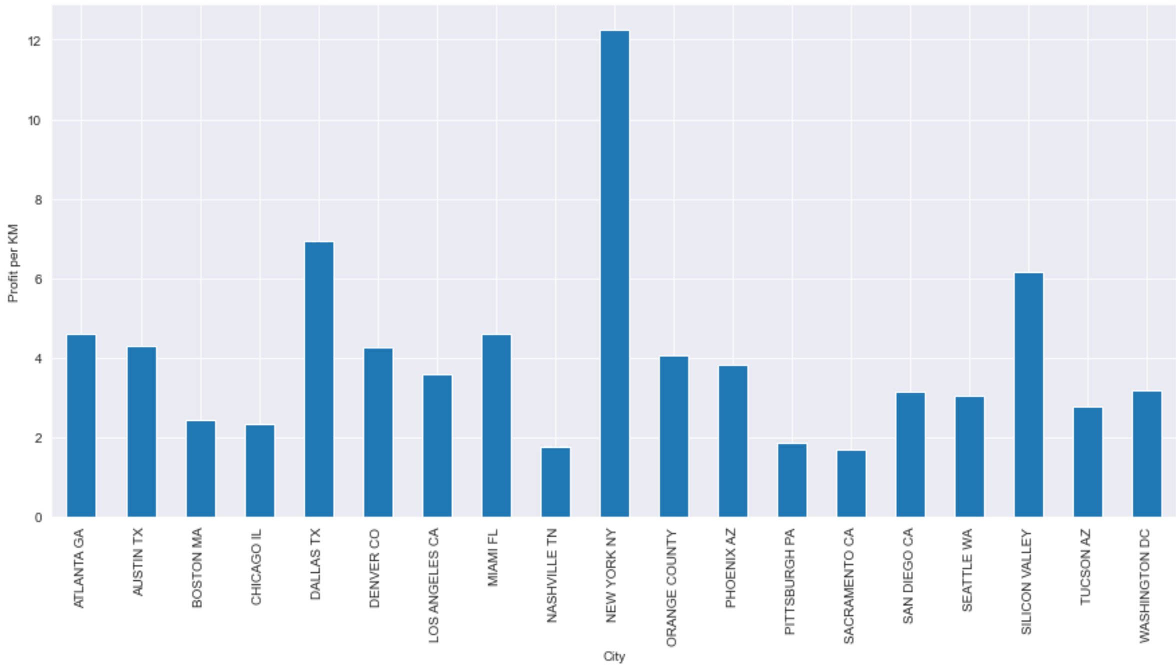
```
df.groupby('City')['Profit per KM'].median()
```

Out[18]:

```
City
ATLANTA GA         4.591498
AUSTIN TX          4.296468
BOSTON MA          2.448953
CHICAGO IL         2.329217
DALLAS TX          6.936991
DENVER CO          4.262593
LOS ANGELES CA     3.570160
MIAMI FL           4.599710
NASHVILLE TN       1.769476
NEW YORK NY       12.268408
ORANGE COUNTY      4.051526
PHOENIX AZ         3.815931
PITTSBURGH PA      1.863913
SACRAMENTO CA      1.696495
SAN DIEGO CA       3.131335
SEATTLE WA         3.052507
SILICON VALLEY     6.169811
TUCSON AZ          2.770540
WASHINGTON DC      3.171498
Name: Profit per KM, dtype: float64
```

In [19]:

```
df.groupby('City')['Profit per KM'].median().plot(kind='bar', figsize=(15,7), ylabel='Profi
```



New York City has the highest Profit per KM while Sacramenyo has the lowest Profit per KM

Overall profit analysis over 3 years

In [20]:

```
month_year_group = df.groupby(['Month', 'Year'])
(month_year_group[['KM Travelled', 'Profit', 'Profit Percentage per Trip', 'Profit per KM']
```

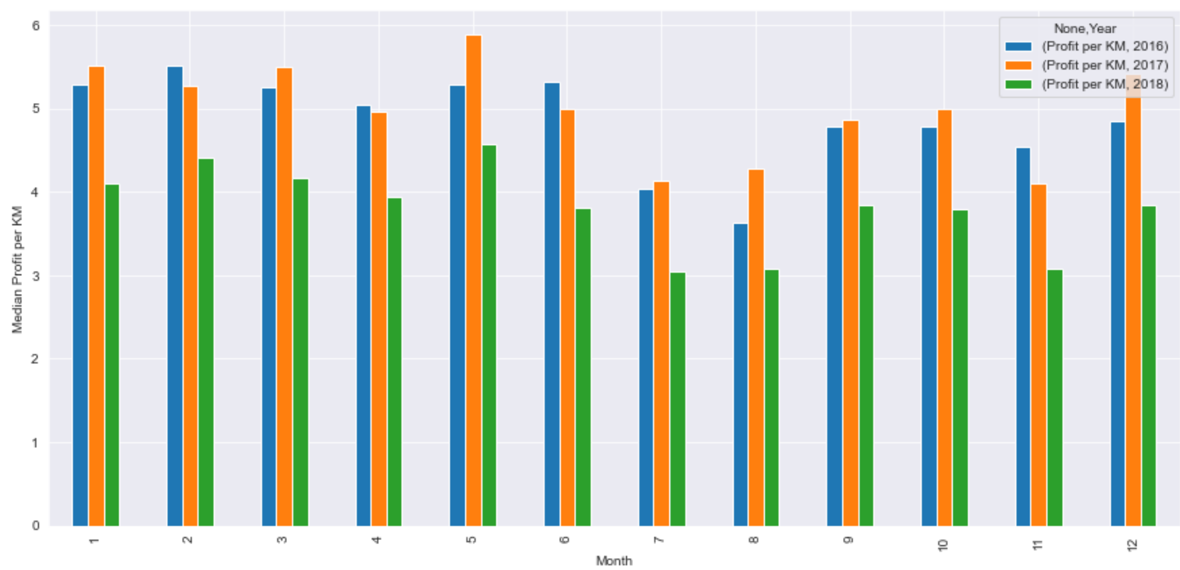
Out[20]:

	KM Travelled			Profit			Profit Percentage per Trip			Profit per KM	
Year	2016	2017	2018	2016	2017	2018	2016	2017	2018	2016	201
Month											
1	22.68	22.310	22.47	101.3955	102.5436	78.1024	42.165	43.920	32.780	5.284826	5.5
2	22.23	22.800	22.42	105.1704	98.8360	83.3908	44.110	41.760	35.100	5.508758	5.2
3	22.66	22.310	22.40	99.0996	100.4620	80.1816	42.190	43.920	33.610	5.263040	5.4
4	22.47	22.200	22.77	94.0260	84.4970	75.0600	40.620	39.725	31.440	5.038333	4.9
5	22.14	22.000	22.44	97.5768	106.3420	83.5194	41.825	46.110	36.135	5.280400	5.8
6	22.54	22.680	22.04	101.4520	92.0100	71.1640	42.320	39.860	30.220	5.326667	4.9
7	22.88	22.420	22.47	75.8940	75.5400	56.7342	32.635	33.380	24.590	4.035815	4.1
8	22.44	22.420	22.04	66.4040	79.0450	56.9450	29.500	34.605	25.060	3.624775	4.2
9	22.31	22.610	22.31	86.2068	89.8596	72.2480	37.830	38.960	30.840	4.783265	4.8
10	22.31	22.455	22.61	84.6760	86.0140	70.9784	38.220	39.445	30.600	4.787408	4.9
11	22.54	22.540	22.67	85.1220	71.8090	58.1984	37.030	33.280	25.165	4.537607	4.1
12	22.60	22.610	22.54	91.5096	95.3624	73.1400	39.300	43.640	31.170	4.850633	5.4

On comparison, we see that there is slight decrement in the profit margin for the year 2018.

In [21]:

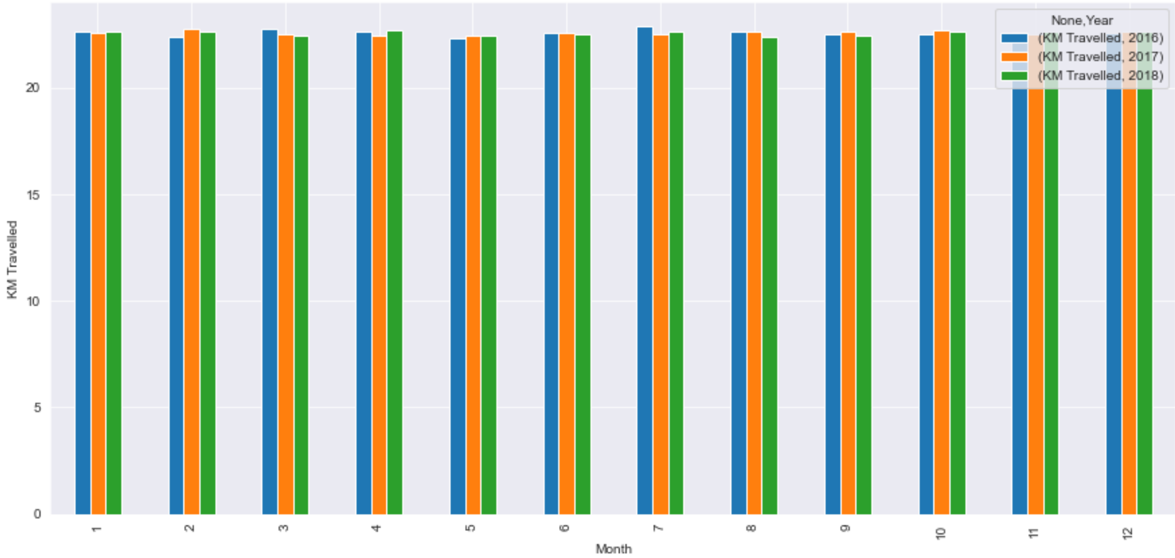
```
((month_year_group[['Profit per KM']].median()).unstack()).plot(kind='bar', figsize=(15,7),
```



There is a dip in profit per KM each year during July and August which implies there is some seasonality.

In [22]:

```
((month_year_group[['KM Travelled']].mean()).unstack()).plot(kind='bar', figsize=(15,7), yl
```



Avg distance travelled is 22.5 KM. Later we will prove it using null hypothesis.

Weekly Analysis:

In [23]:

```
weekday_group = df.groupby(['Weekday'])
(weekday_group[['KM Travelled', 'Profit', 'Profit Percentage per Trip', 'Profit per KM']].m
```

Out[23]:

Weekday	KM Travelled	Profit	Profit Percentage per Trip	Profit per KM
0	22.31	73.5640	32.180	3.996667
1	22.44	75.9660	33.750	4.181606
2	22.42	73.8000	32.670	4.056515
3	22.31	71.8356	32.045	3.970767
4	22.54	86.0720	37.370	4.637247
5	22.54	89.6004	38.360	4.776257
6	22.54	91.4540	39.830	4.973045

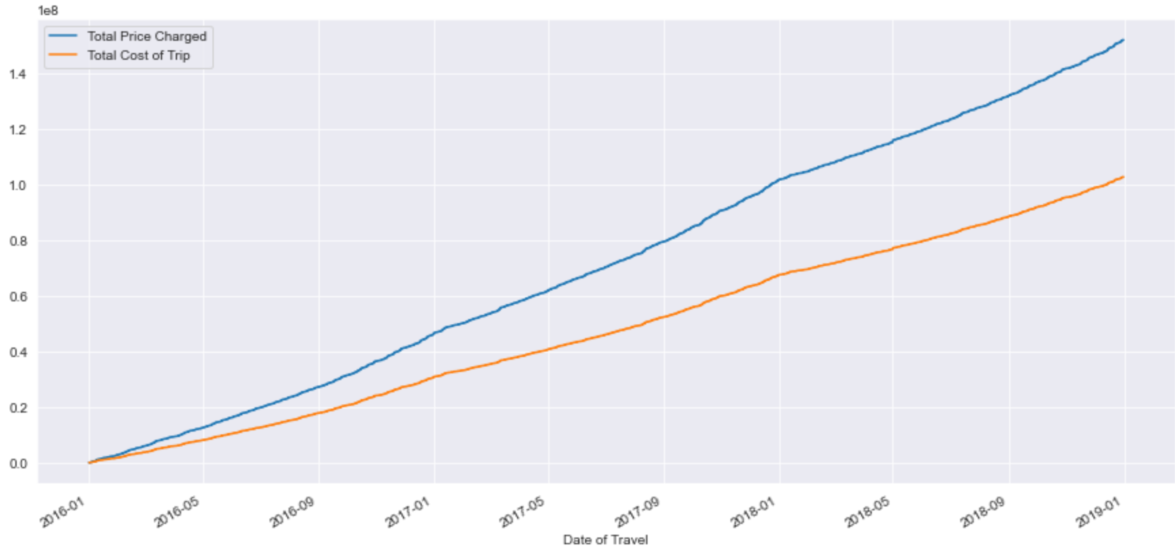
Over the weekends: distance travelled increases slightly => Profit increases

Cummulative Profit Vs Cost over the years b/w 2 cab service providers

In [24]:

```
df['Total Price Charged'] = df['Price Charged'].cumsum()
df['Total Cost of Trip'] = df['Cost of Trip'].cumsum()

plt.figure(figsize=(15, 7))
df['Total Price Charged'].plot();
df['Total Cost of Trip'].plot();
plt.legend();
```

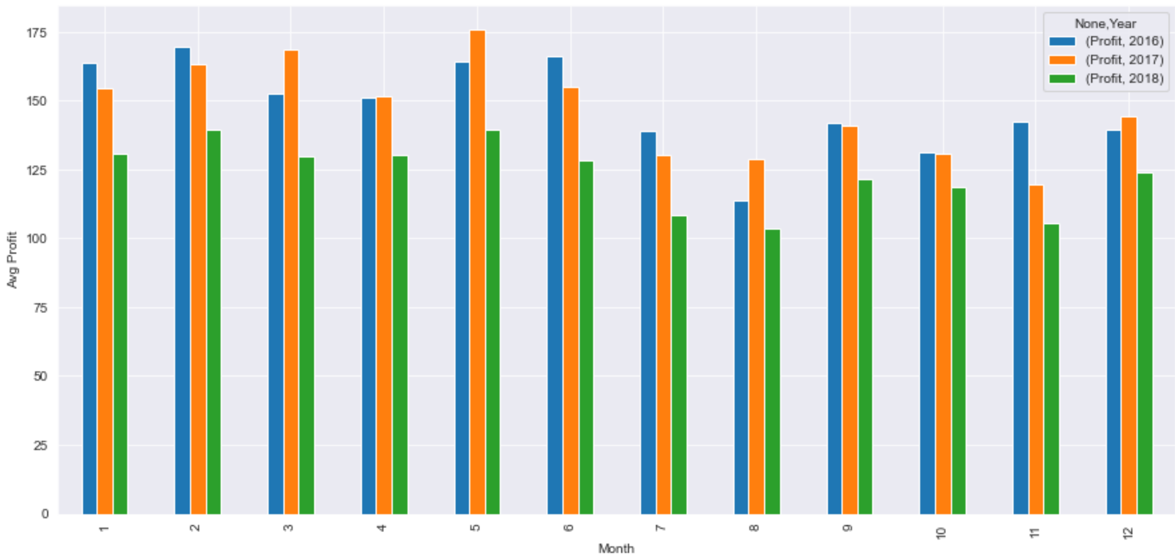


Above graph shows the power of compounding effect.

To maximize the profit, XYZ should invest for a long term

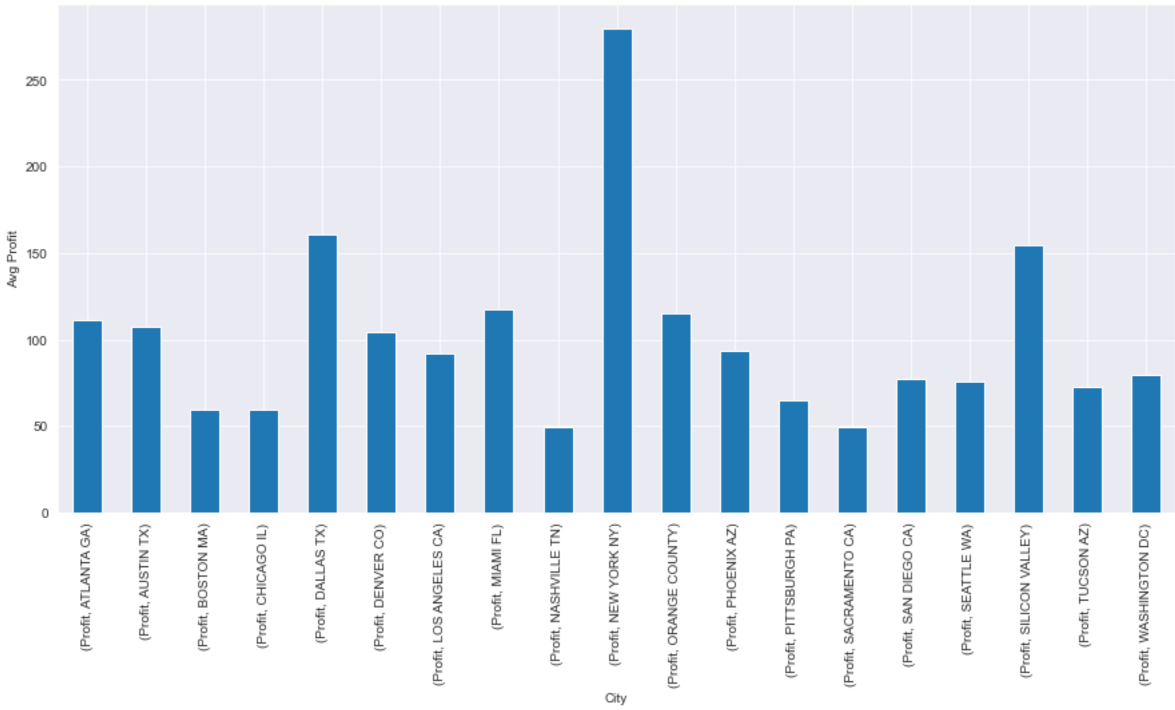
In [25]:

```
((month_year_group[['Profit']].mean()).unstack()).plot(kind='bar', figsize=(15, 7), ylabel=
```



In [26]:

```
((city_grp[['Profit']].mean()).unstack()).plot(kind='bar', figsize=(15, 7), xlabel='City',
```



Top 5 cities with highest avg profit (in descending order):

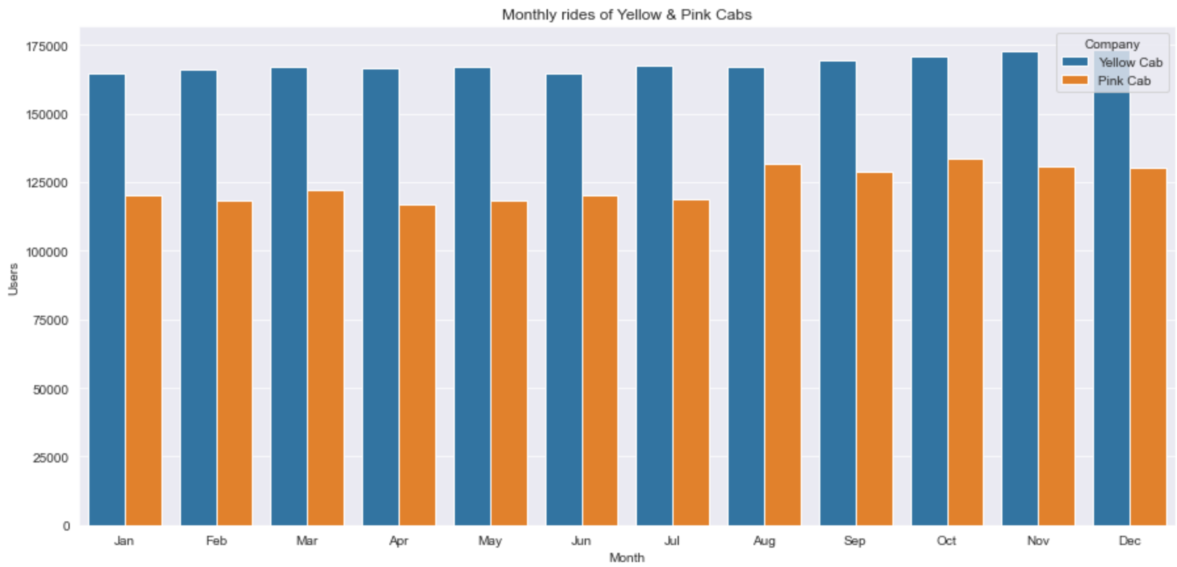
1. New York
2. Dallas
3. Silicon Valley
4. Miami
5. Orange County

In [27]:

```
month = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'
```

In [28]:

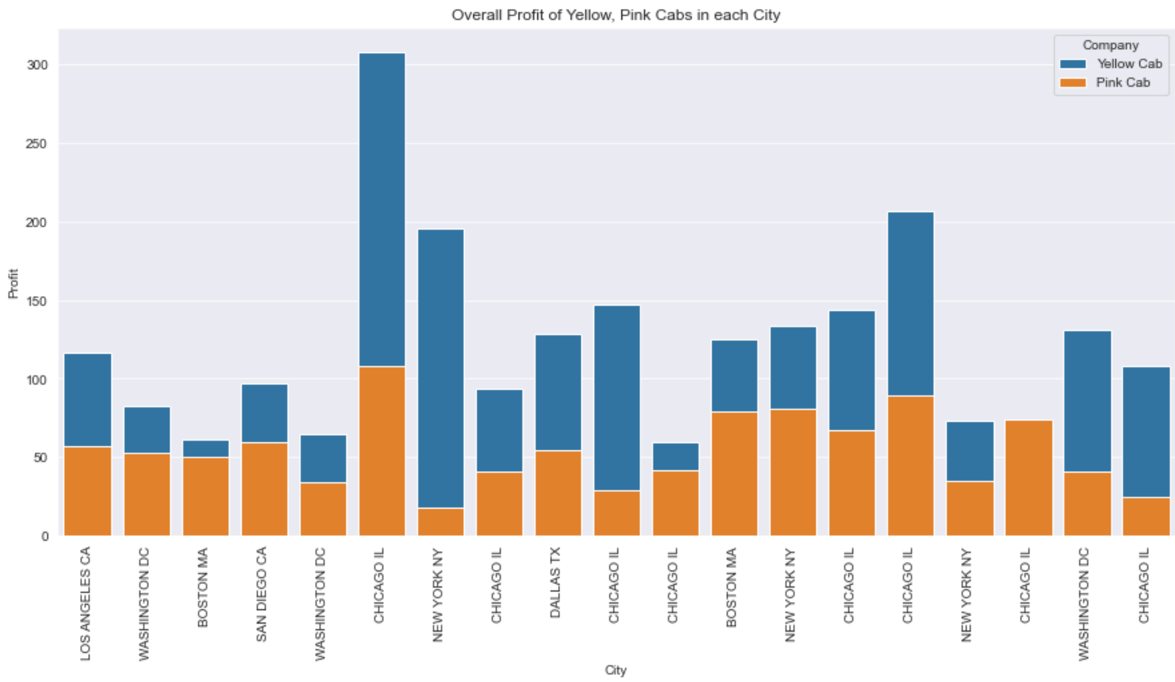
```
plt.figure(figsize=(15,7))
g = sns.barplot('Month', 'Users', data=df, hue='Company', ci=None);
g.set_xticklabels(labels=month, rotation=0)
g.set_title('Monthly rides of Yellow & Pink Cabs')
plt.show()
```



Yellow cab has more users each month over the years

In [29]:

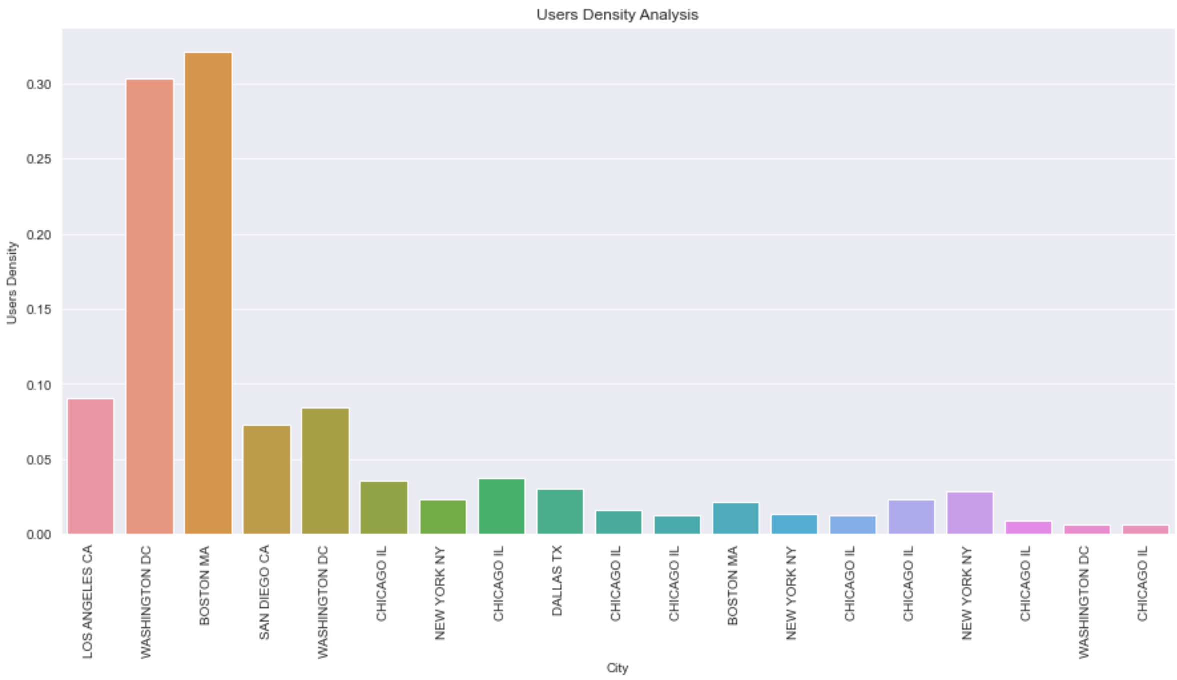
```
plt.figure(figsize=(15,7))
g = sns.barplot('City', 'Profit', data=df, hue='Company', ci=None, dodge=0);
g.set_xticklabels(labels=df['City'], rotation=90)
g.set_title('Overall Profit of Yellow, Pink Cabs in each City')
plt.show()
```



Except Chicago, Yellow Cab has more profit margin in each city.

In [30]:

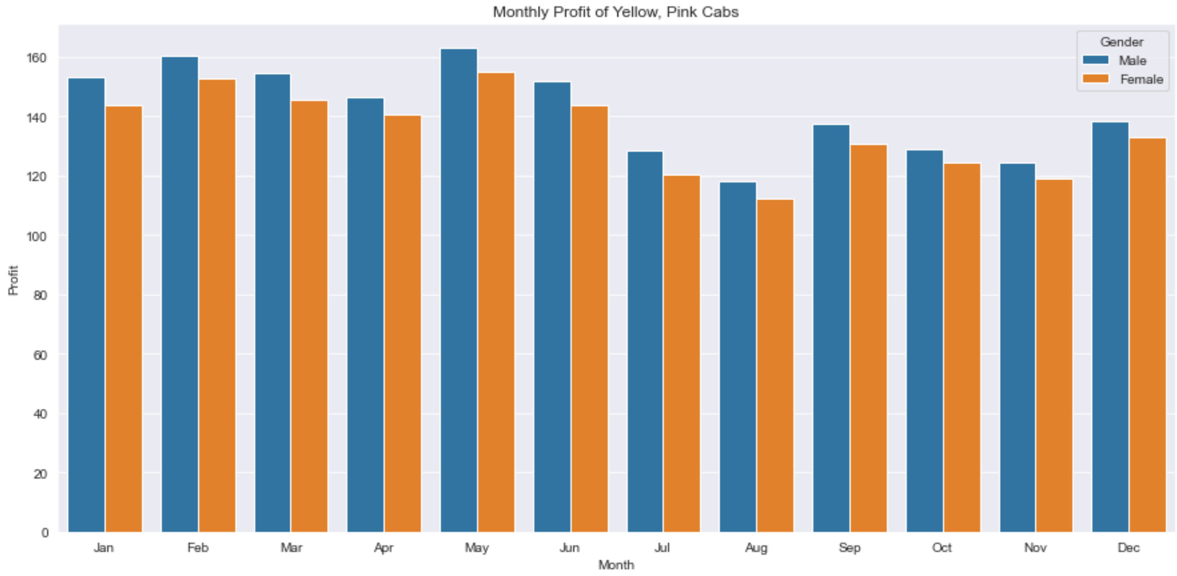
```
plt.figure(figsize=(15,7))
g = sns.barplot('City', 'Users Density', data=df, ci=None, dodge=1);
g.set_xticklabels(labels=df['City'], rotation=90)
g.set_title('Users Density Analysis')
plt.show()
```



Around 30% of the population in Washington DC and Boston use cab services whereas for all other cities it's less than 10%

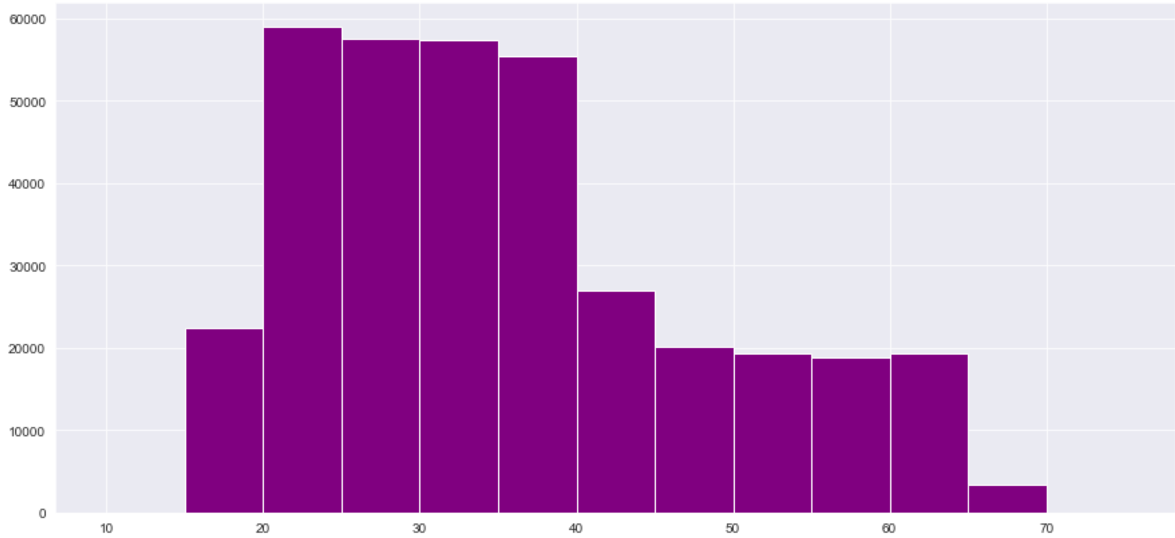
In [31]:

```
plt.figure(figsize=(15,7))
g = sns.barplot('Month', 'Profit', data=df, hue='Gender', ci=None);
g.set_xticklabels(labels=month, rotation=0)
g.set_title('Monthly Profit of Yellow, Pink Cabs')
plt.show()
```



In [32]:

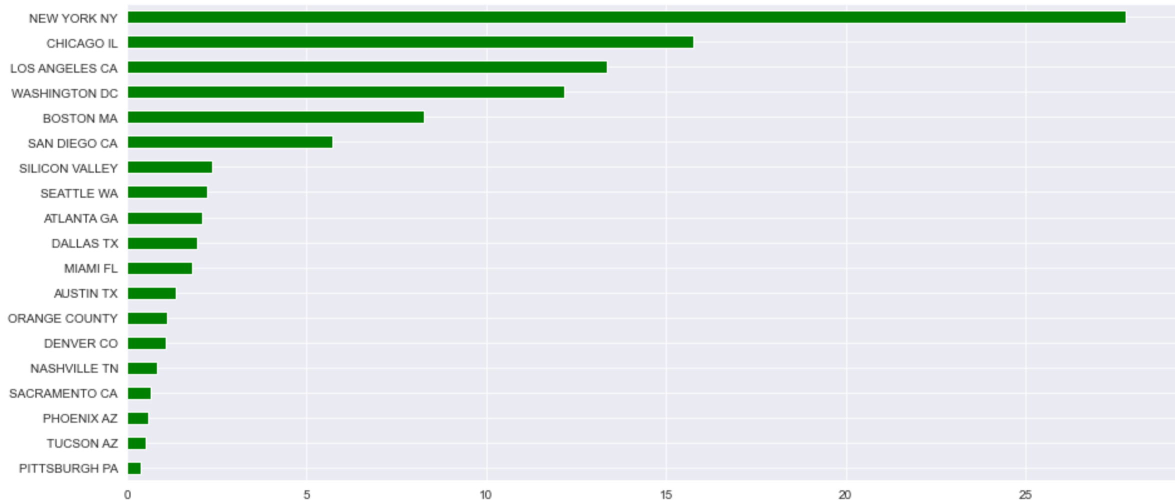
```
plt.figure(figsize=(15,7))
plt.hist(df.Age, bins=np.arange(10, 80, 5), color='purple');
plt.show()
```



Most of the users are aged between 20 to 40 years

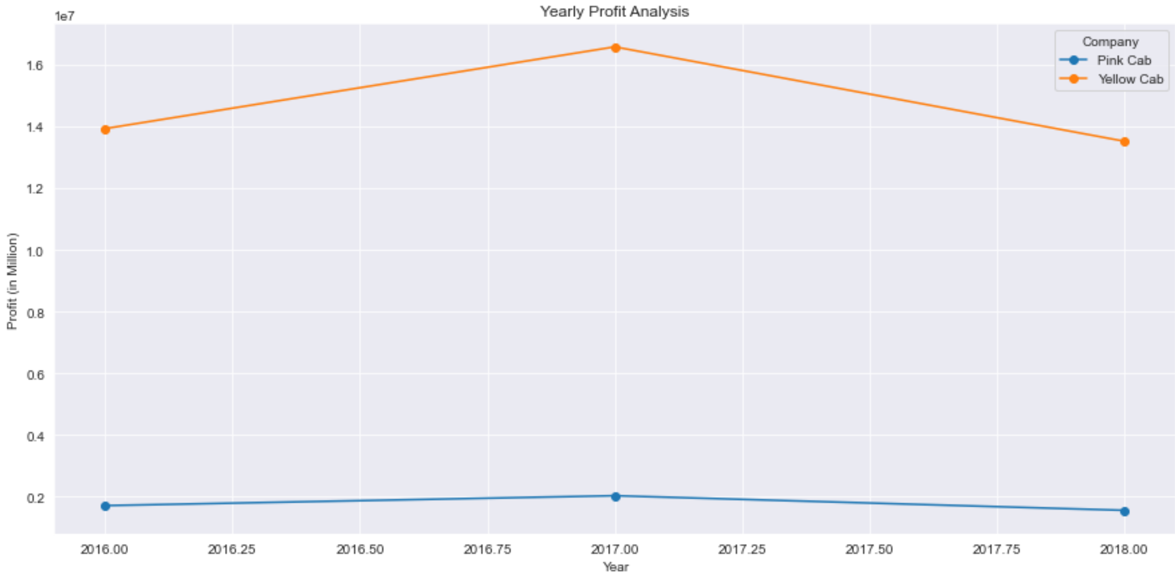
In [33]:

```
plt.figure(figsize=(15,7))
(df.City.value_counts(normalize=True, ascending=True)*100).plot(kind='barh', color='g');
plt.show()
```



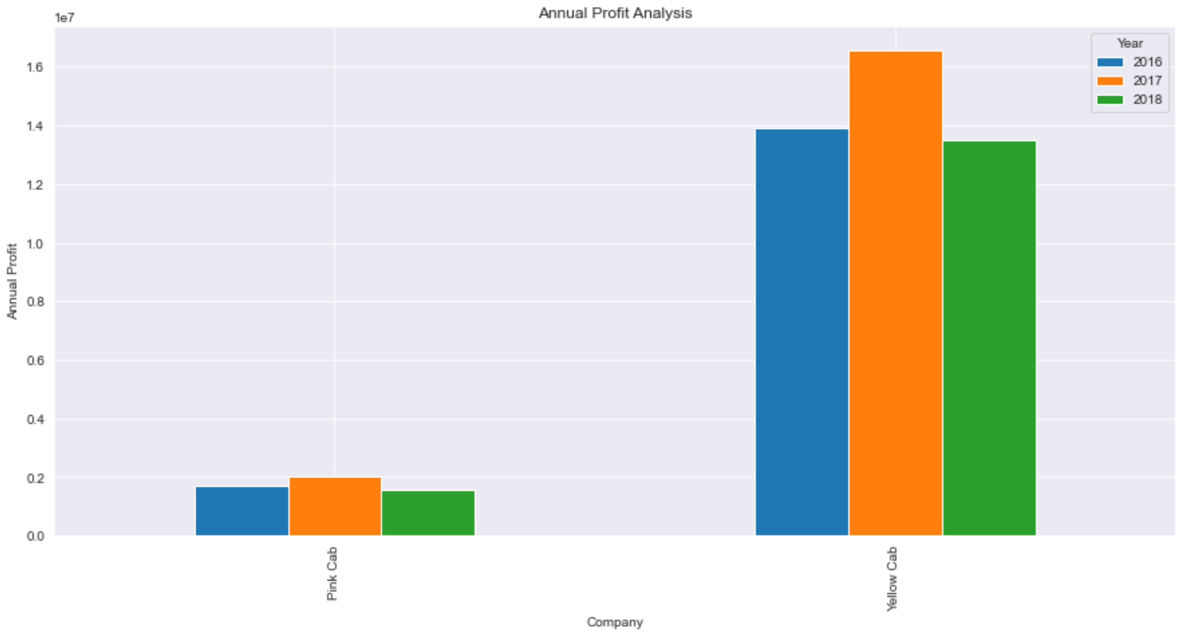
In [34]:

```
fig, ax = plt.subplots(figsize=(15, 7))
df.groupby(['Year', 'Company']).sum()['Profit'].unstack().plot(ax=ax, title='Yearly Profit
```



In [35]:

```
(df.pivot_table(index='Company', columns='Year', values='Profit', aggfunc='sum')).plot(kind
```



In [36]:

```
df.pivot_table(index='Company', columns='Year', values='Profit', aggfunc='sum', margins=Tru
```

Out[36]:

Year	2016	2017	2018	All
Company				
Pink Cab	1.713511e+06	2.033655e+06	1.560162e+06	5.307328e+06
Yellow Cab	1.392700e+07	1.657598e+07	1.351740e+07	4.402037e+07
All	1.564051e+07	1.860963e+07	1.507756e+07	4.932770e+07

In [37]:

```
fig, ax = plt.subplots(figsize=(15, 7))
df.groupby(['Month', 'Company']).sum()['Profit'].unstack().plot(ax=ax, title='Monthly Profi
```

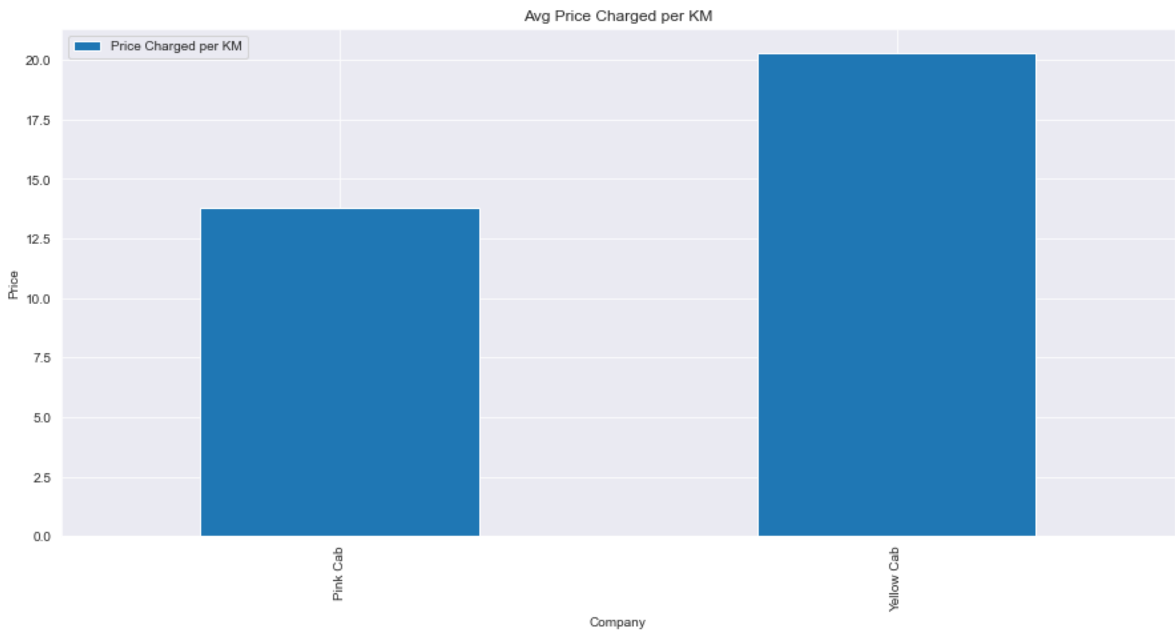
There is a decrease in Monthly Profit of Yellow Cab during June to August whereas Pink Cab has an increase

In [38]:

```
df['Price Charged per KM'] = df['Price Charged'] / df['KM Travelled']
```

In [39]:

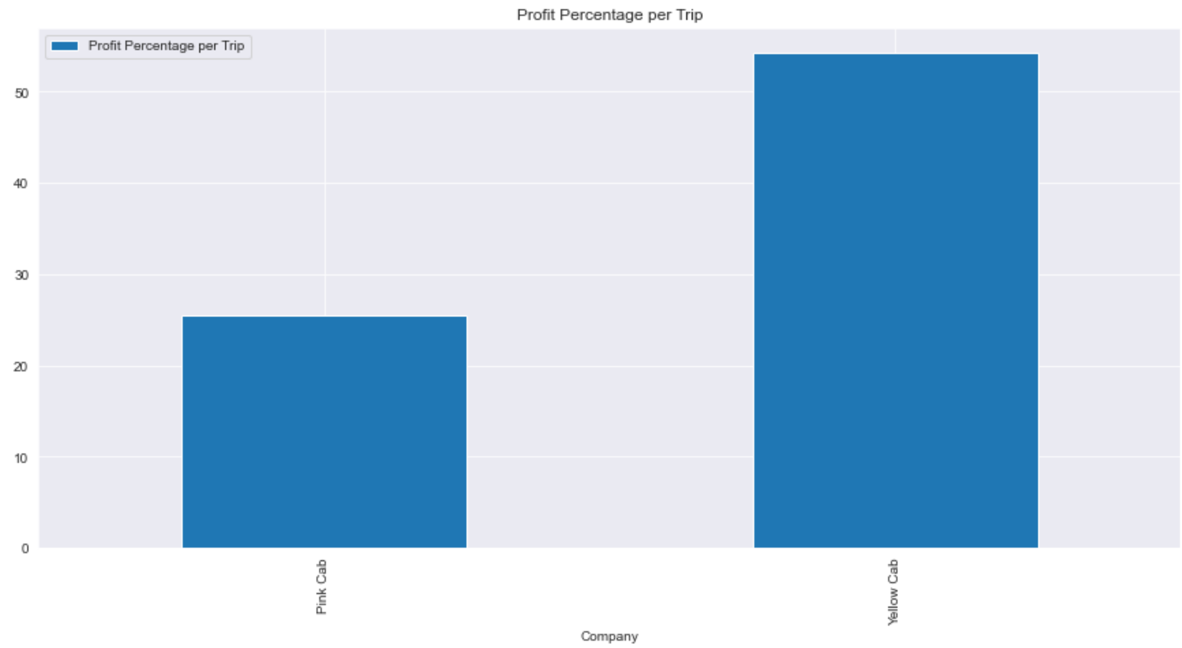
```
((df[['Price Charged per KM', 'Company']].groupby('Company')).mean()).plot(kind='bar', figs
plt.show()
```



Avg Price Charged per KM for Yellow Cab is 20.3 USD & for Pink Cab is 13.76 USD

In [40]:

```
df[['Profit Percentage per Trip', 'Company']].groupby('Company').mean().plot(kind='bar', fi
plt.show()
```



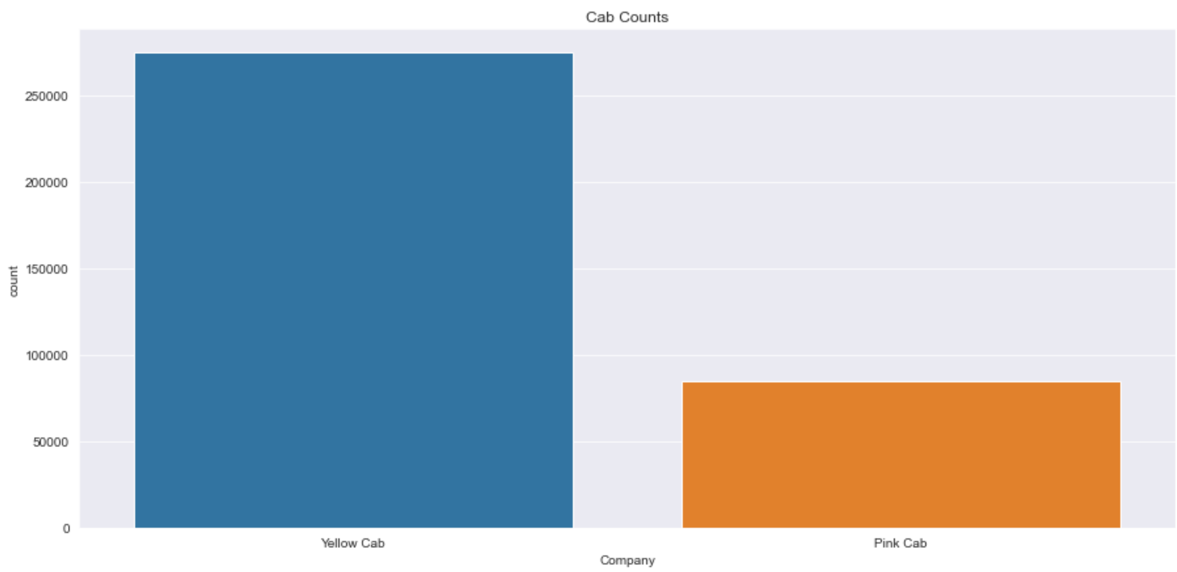
Profit Percentage per Trip for

Pink Cab is 25.559567

Yellow Cab is 54.296631

In [41]:

```
plt.figure(figsize=(15,7))
g=sns.countplot(x='Company', data=df);
g.set_title('Cab Counts')
plt.show()
```



In [42]:

```
df['Company'].value_counts(normalize=True)
```

Out[42]:

```
Yellow Cab    0.764294
Pink Cab      0.235706
Name: Company, dtype: float64
```

In [43]:

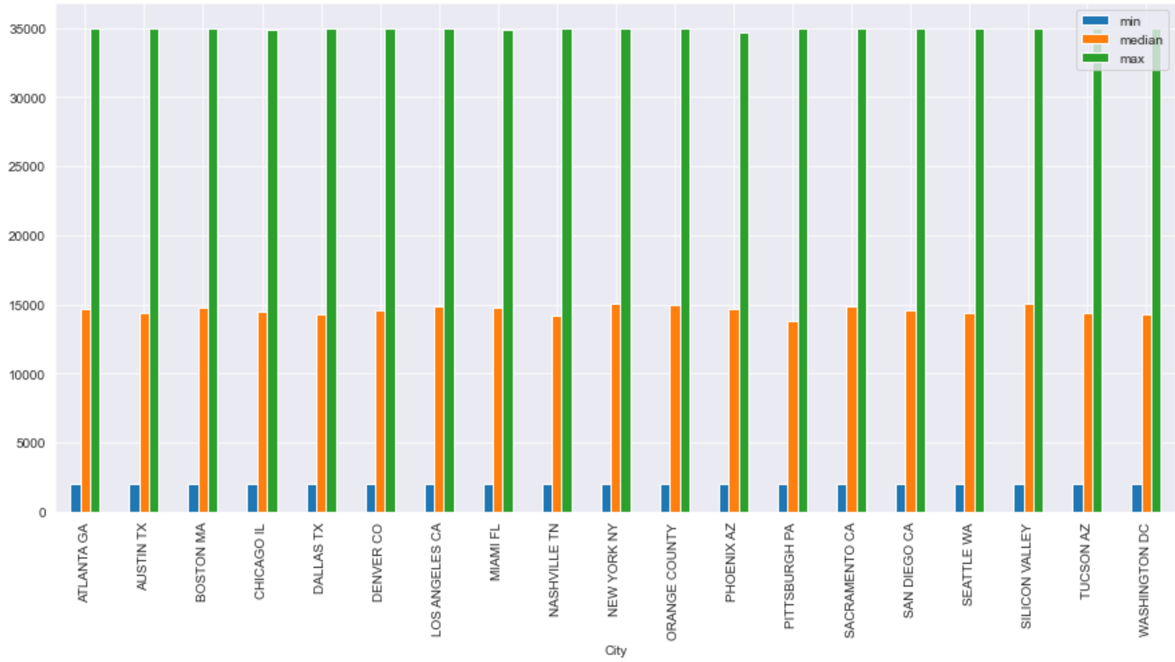
```
city_grp['Income (USD/Month)'].agg(['median', 'mean', 'min', 'max'])
```

Out[43]:

City	median	mean	min	max
ATLANTA GA	14655	14933.150986	2029	34972
AUSTIN TX	14374	14696.495711	2027	34921
BOSTON MA	14743	15128.563317	2019	34985
CHICAGO IL	14527	15101.718269	2007	34901
DALLAS TX	14242	14846.508194	2007	34996
DENVER CO	14580	14975.655163	2022	35000
LOS ANGELES CA	14889	15064.550455	2007	34984
MIAMI FL	14759	14984.887202	2013	34862
NASHVILLE TN	14195	14734.359801	2002	34960
NEW YORK NY	15024	15184.765801	2012	34989
ORANGE COUNTY	14963	15188.944500	2030	34979
PHOENIX AZ	14646	15012.038275	2011	34681
PITTSBURGH PA	13833	14410.332064	2010	34984
SACRAMENTO CA	14829	15268.225180	2001	34995
SAN DIEGO CA	14612	15049.874854	2016	34936
SEATTLE WA	14358	14840.748281	2000	34967
SILICON VALLEY	15107	15248.547717	2000	34977
TUCSON AZ	14422	14942.952356	2012	34928
WASHINGTON DC	14268	14727.430162	2003	34996

In [44]:

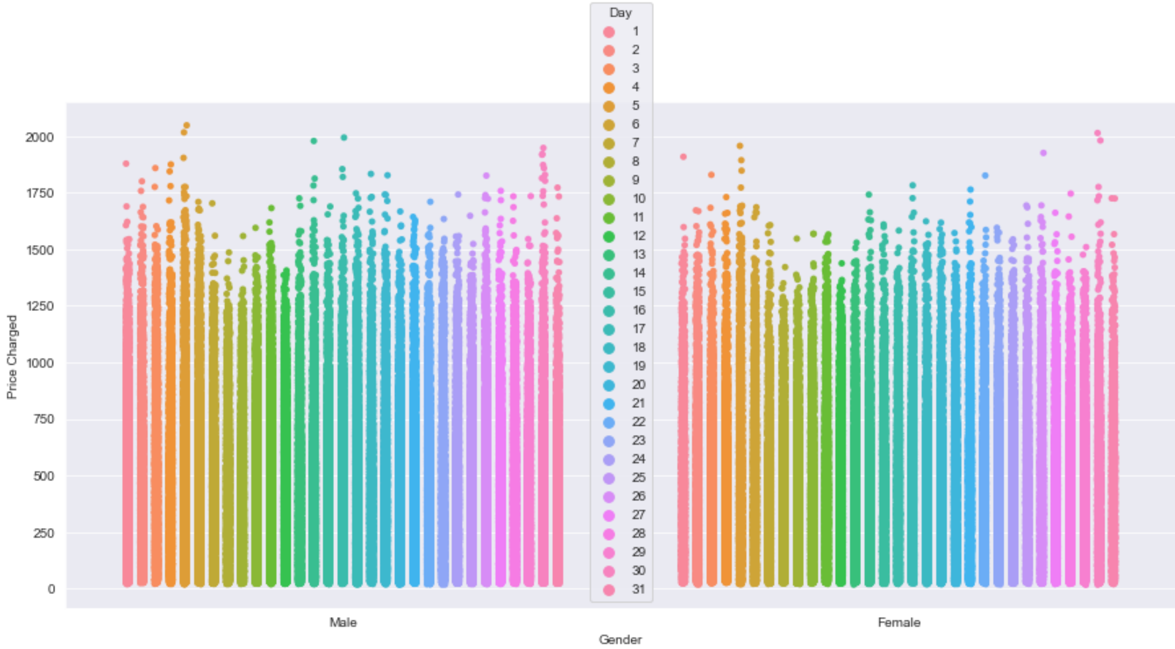
```
(city_grp['Income (USD/Month)'].agg(['min', 'median', 'max'])).plot(kind='bar', figsize=(15
```



There is equal range of incomes for each city

In [45]:

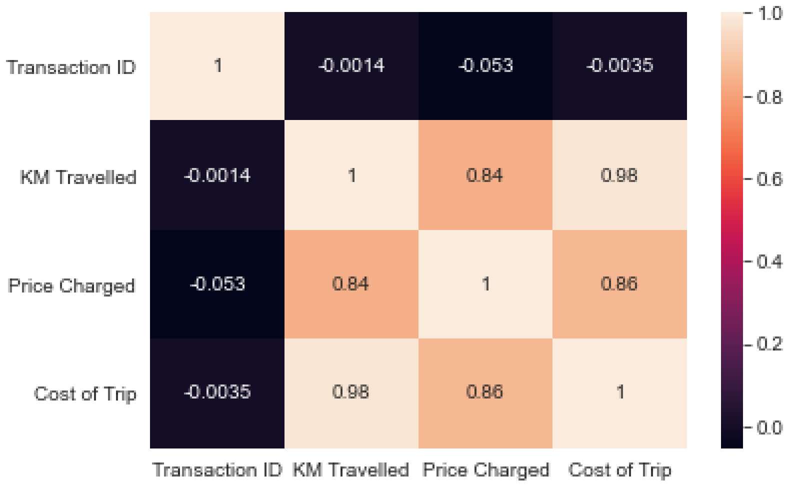
```
plt.figure(figsize=(15,7))
sns.stripplot(x="Gender", y="Price Charged", hue="Day", data = df, dodge=True)
plt.show()
```



There is no discount for Female customers

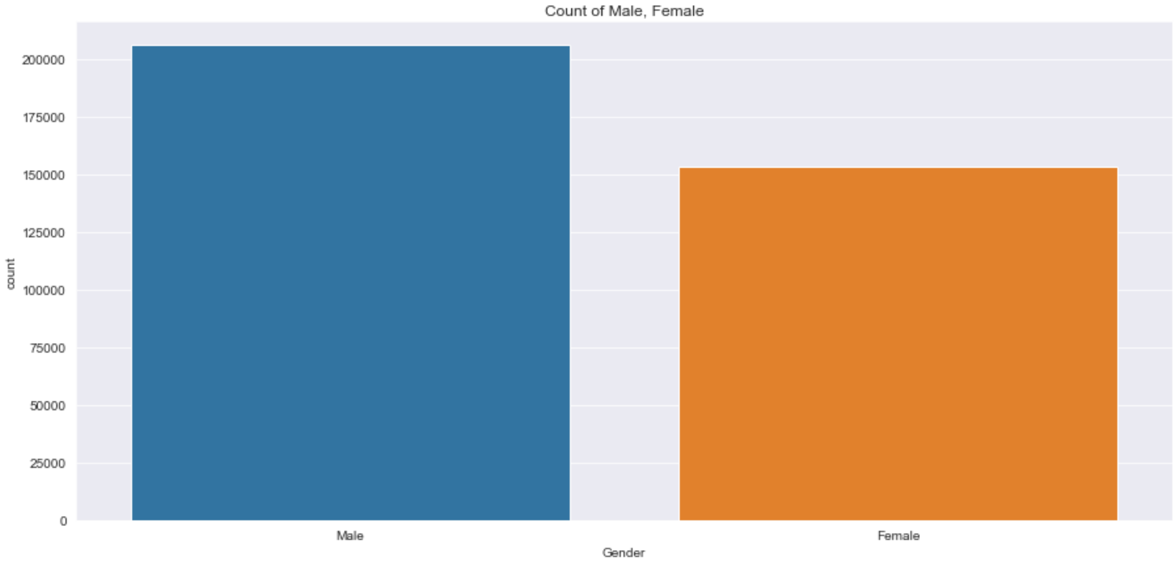
In [46]:

```
sns.heatmap(cab.corr(), color='b', annot=True);
```



In [47]:

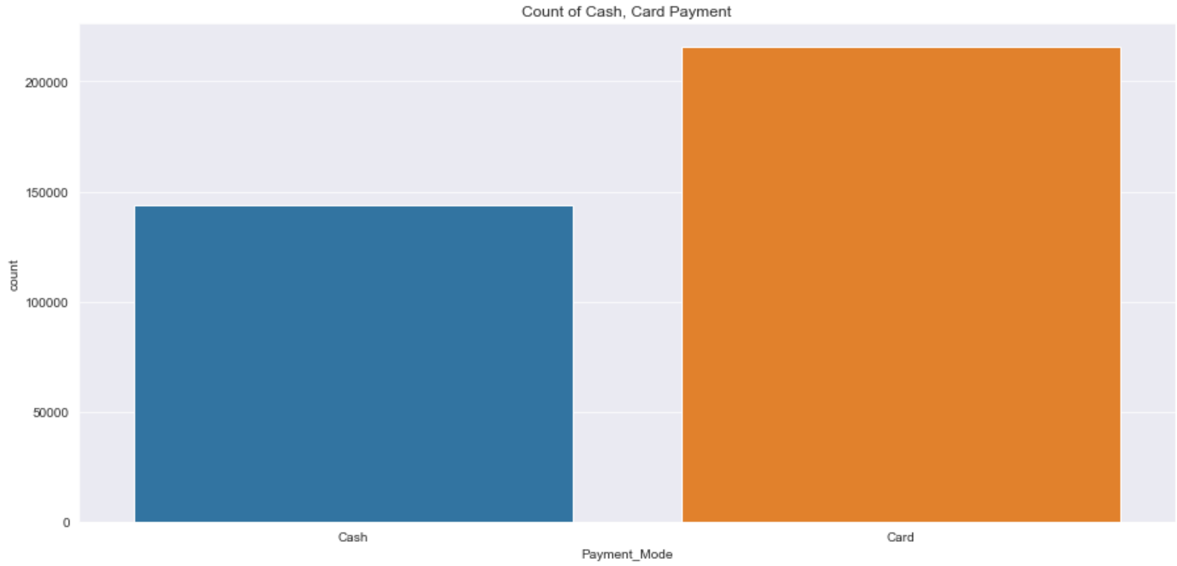
```
plt.figure(figsize=(15,7))
g=sns.countplot(x = 'Gender', data = df);
g.set_title('Count of Male, Female')
plt.show()
```



More no. of Male users

In [48]:

```
plt.figure(figsize=(15,7))
g=sns.countplot(df['Payment_Mode']);
g.set_title('Count of Cash, Card Payment')
plt.show()
```



No. of Card payment is more than no. of cash payment

In [49]:

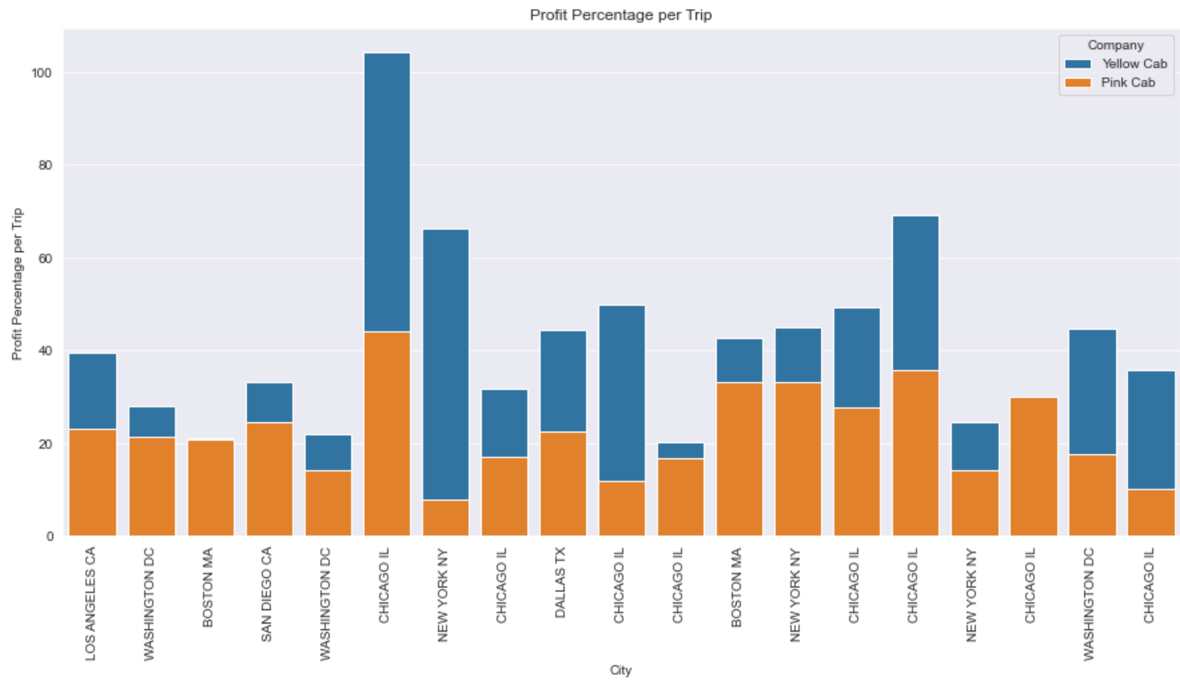
```
df['Payment_Mode'].value_counts()
```

Out[49]:

```
Card    215504
Cash    143888
Name: Payment_Mode, dtype: int64
```

In [50]:

```
plt.figure(figsize=(15,7))
g = sns.barplot(x='City', y='Profit Percentage per Trip', hue='Company', data=df, dodge=0,
g.set_title('Profit Percentage per Trip')
g.set_xticklabels(labels=df['City'], rotation=90);
plt.show()
```



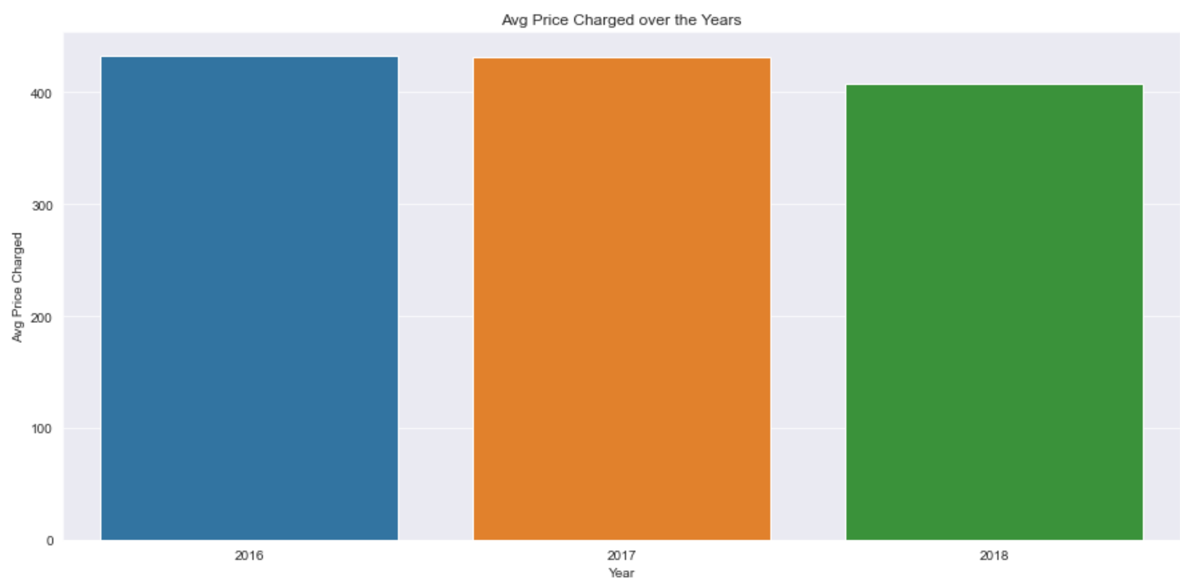
Except Chicago and Boston Yellow Cab makes more profit per trip in every city

In [51]:

```
yearly_price = df.groupby(['Year'])['Price Charged'].mean().reset_index().rename(columns =
```

In [52]:

```
plt.figure(figsize=(15,7))
sns.barplot(x = 'Year', y = 'Avg Price Charged', data = yearly_price).set_title("Avg Price
plt.show()
```



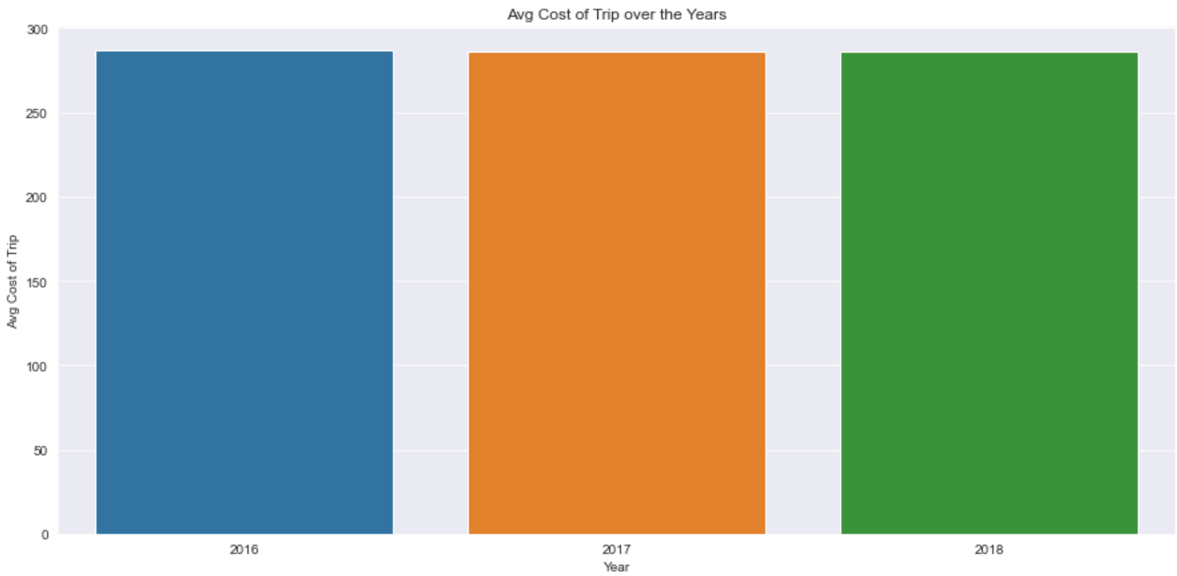
The avg price charged for the year 2018 is comparitively less.

In [53]:

```
yearly_cost = df.groupby(['Year'])['Cost of Trip'].mean().reset_index().rename(columns = {'
```

In [54]:

```
plt.figure(figsize=(15,7))
sns.barplot(x = 'Year', y = 'Avg Cost of Trip', data = yearly_cost).set_title("Avg Cost of
plt.show()
```



The avg cost of trip over the years remains constant

Join US Holidays Dataset for further Analysis

In [55]:

```
holidays = pd.read_csv('us_bank_holidays.csv')
holidays.head()
```

Out[55]:

	date	holiday_name	holiday	year	month	wday	weekend	long_holiday	school_break
0	2012-01-01	New Year Day	True	2012	1	7	True	True	christmas_break
1	2012-01-02	FALSE	False	2012	1	1	False	False	no_break
2	2012-01-03	FALSE	False	2012	1	2	False	False	no_break
3	2012-01-04	FALSE	False	2012	1	3	False	False	no_break
4	2012-01-05	FALSE	False	2012	1	4	False	False	no_break

In [56]:

```
holidays.drop(['year', 'month', 'wday', 'dayno'], axis=1, inplace=True)
holidays['date'] = pd.to_datetime(holidays['date'])
```

In [57]:

```
holidays.sort_values(by='date', inplace=True)
holidays.set_index('date', inplace=True)

df = pd.merge(df, holidays, left_index=True, right_index=True)
df.head()
```

Out[57]:

	Company	City	KM Travelled	Price Charged	Cost of Trip	Population	Users	Gender	A
2016-01-02	Yellow Cab	LOS ANGELES CA	25.53	402.89	327.8052	1595037.0	144132.0	Male	
2016-01-02	Yellow Cab	WASHINGTON DC	44.08	694.53	587.1456	418859.0	127001.0	Female	
2016-01-02	Pink Cab	BOSTON MA	38.61	358.05	405.4050	248968.0	80021.0	Male	
2016-01-02	Pink Cab	SAN DIEGO CA	4.72	50.88	51.9200	959307.0	69995.0	Male	
2016-01-02	Yellow Cab	WASHINGTON DC	46.00	765.04	552.0000	418859.0	127001.0	Male	

5 rows × 27 columns

In [58]:

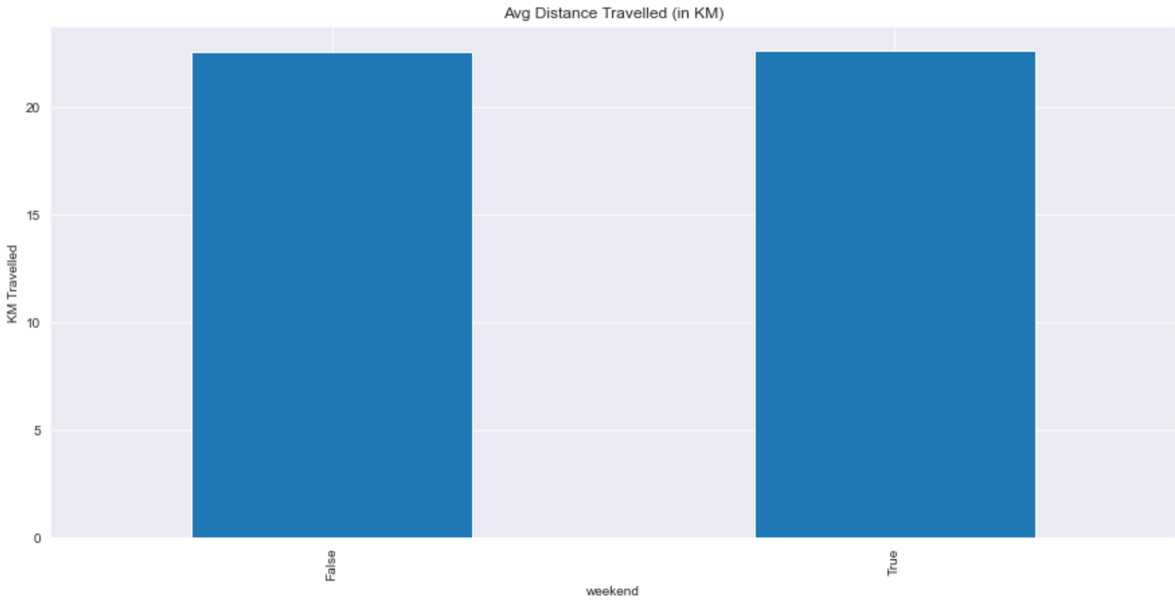
```
df.groupby('weekend')['KM Travelled'].mean()
```

Out[58]:

```
weekend
False    22.544356
True     22.606887
Name: KM Travelled, dtype: float64
```

In [59]:

```
df.groupby('weekend')['KM Travelled'].mean().plot(kind='bar', figsize=(15,7), ylabel='KM Tr
```



Distance travelled is more or less the same for both weekdays and weekends

In [60]:

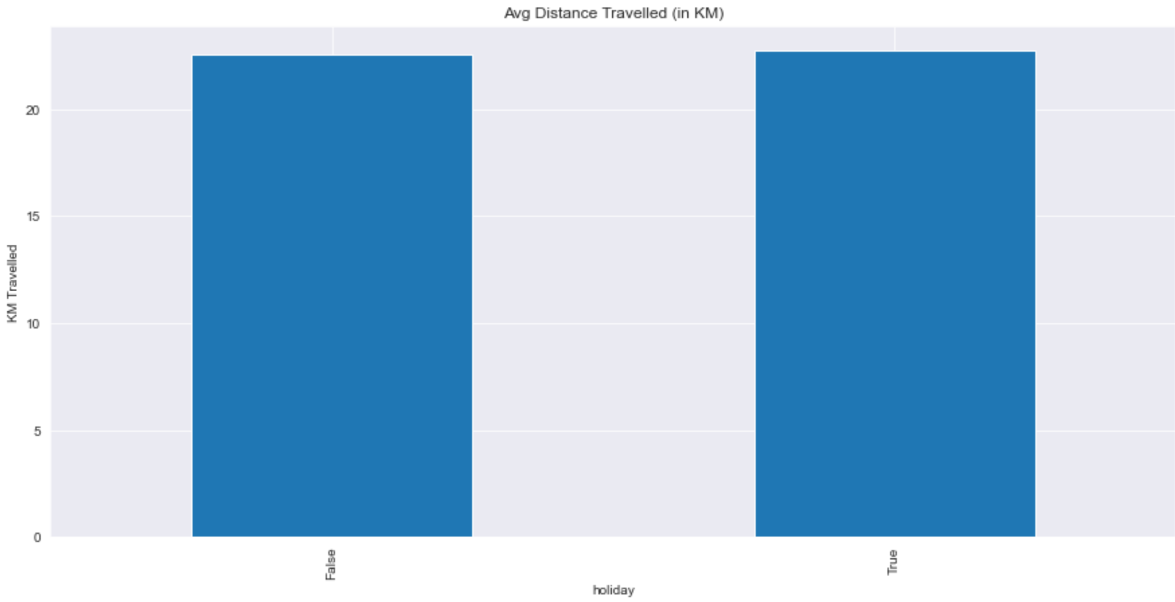
```
df.groupby('holiday')['KM Travelled'].mean()
```

Out[60]:

```
holiday
False    22.560705
True     22.756876
Name: KM Travelled, dtype: float64
```

In [61]:

```
ıoliday')['KM Travelled'].mean().plot(kind='bar', figsize=(15,7), ylabel='KM Travelled', tit
```



Again the distance travelled does not depend on holidays

In [62]:

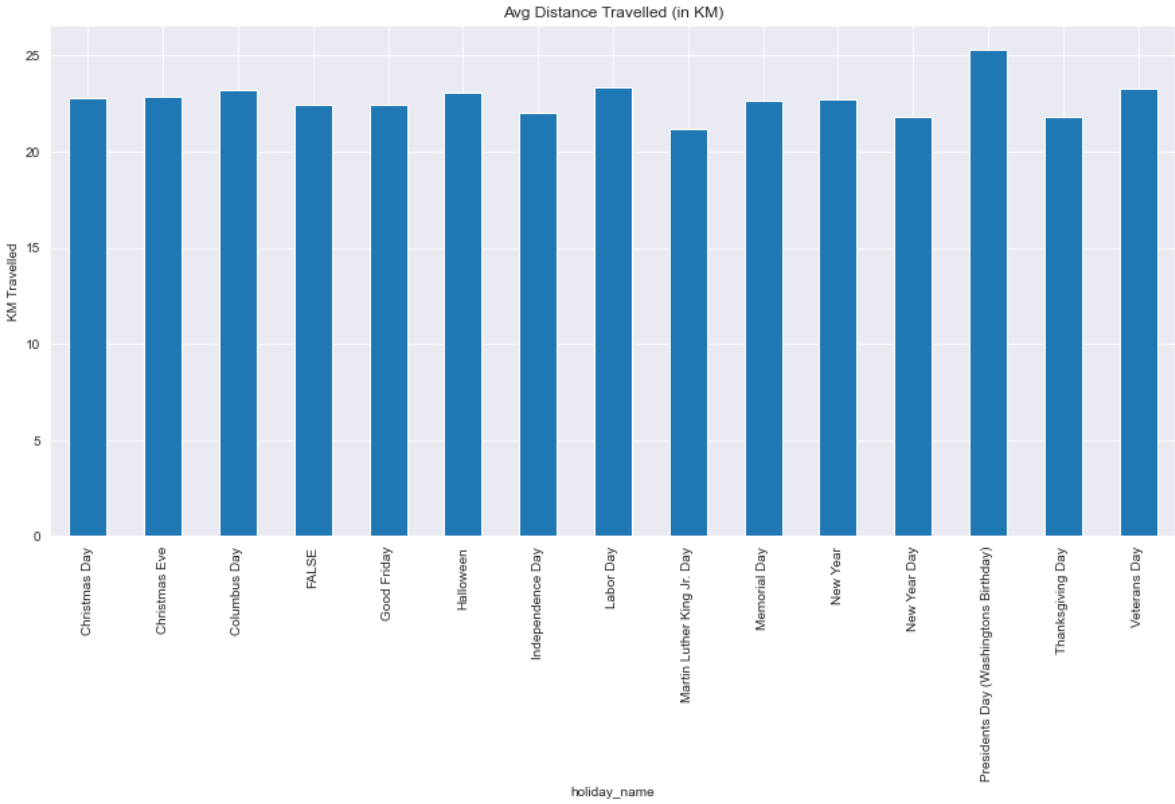
```
df.groupby('holiday_name')['KM Travelled'].median()
```

Out[62]:

```
holiday_name
Christmas Day                         22.770
Christmas Eve                         22.880
Columbus Day                          23.230
FALSE                                 22.440
Good Friday                           22.440
Halloween                             23.100
Independence Day                      22.040
Labor Day                             23.340
Martin Luther King Jr. Day            21.200
Memorial Day                          22.680
New Year                              22.725
New Year Day                          21.800
Presidents Day (Washingtons Birthday) 25.300
Thanksgiving Day                      21.780
Veterans Day                          23.255
Name: KM Travelled, dtype: float64
```

In [63]:

```
df.groupby('holiday_name')['KM Travelled'].median().plot(kind='bar', figsize=(15,7), ylabel
```



Avg Distance travelled is max on Presidents Day (Washington's Birthday)

In [64]:

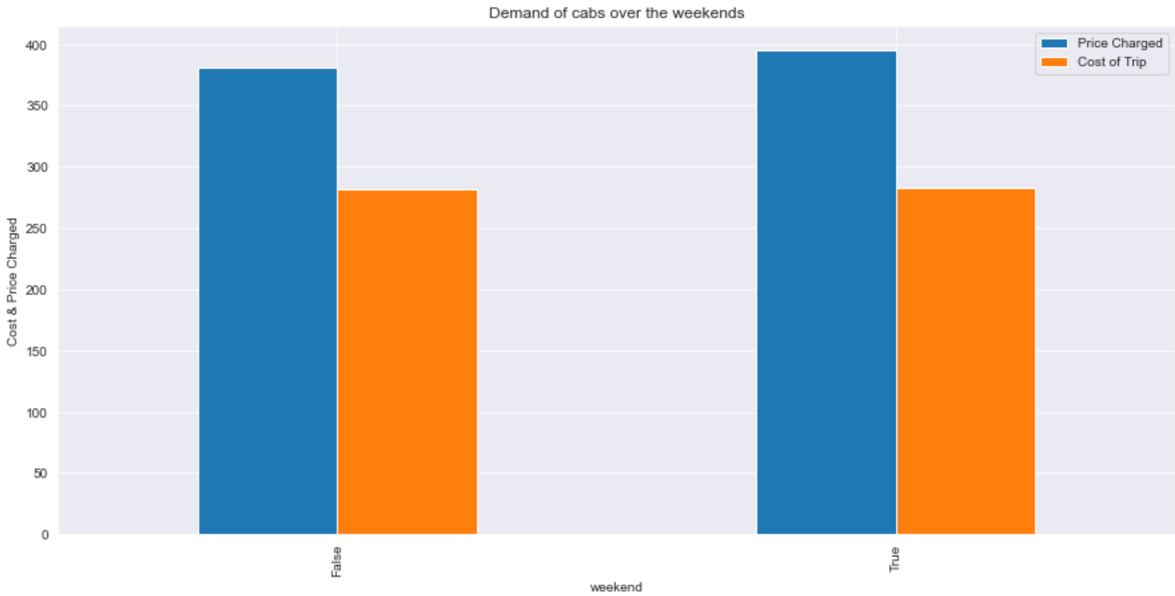
```
df.groupby('weekend')['Price Charged', 'Cost of Trip'].agg(['min', 'max', 'median', 'std'])
```

Out[64]:

	Price Charged				Cost of Trip			
	min	max	median	std	min	max	median	std
weekend								
False	15.60	2048.03	381.025	270.672870	19.0	691.2	281.808	158.10698
True	17.11	2013.95	395.455	280.406014	19.0	691.2	283.140	157.79643

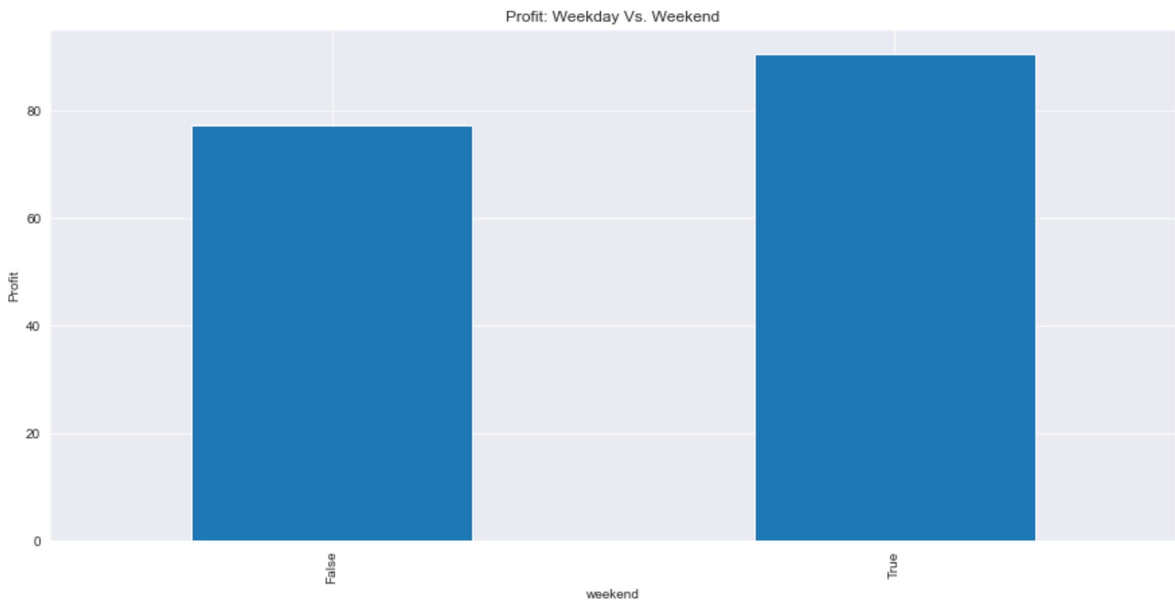
In [65]:

```
'Cost of Trip'].median().plot(kind='bar', figsize=(15,7), ylabel='Cost & Price Charged', tit
```



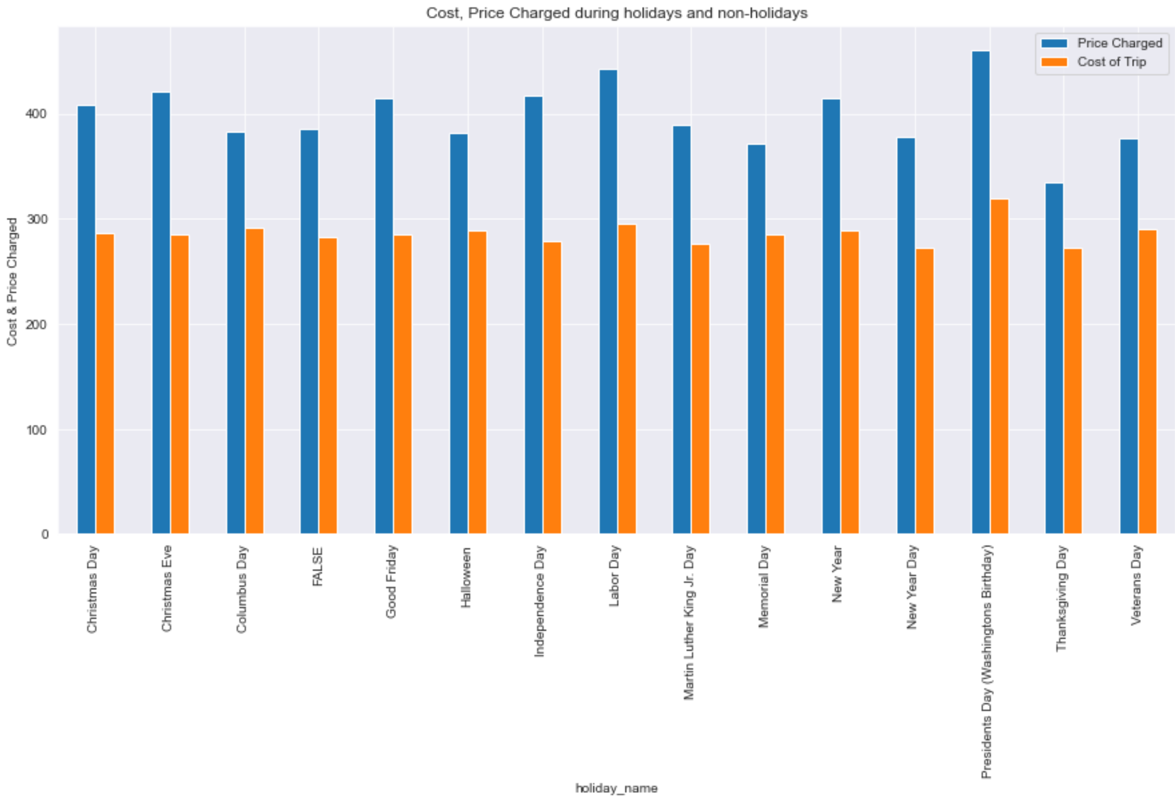
In [66]:

```
df.groupby('weekend')['Profit'].median().plot(kind='bar', figsize=(15,7), ylabel='Profit',
```



In [67]:

```
median().plot(kind='bar', figsize=(15,7), ylabel='Cost & Price Charged', title='Cost, Price
```



T Test

A t-test is a type of inferential statistic which is used to determine if there is a significant difference between the means of two groups which may be related in certain features

T-test has 2 types :

1. One sampled t-test
2. Two-sampled t-test.

In [68]:

```
import scipy.stats as stat
from scipy.stats import ttest_1samp
from scipy.stats import ttest_ind
```

In [69]:

```
df.describe()
```

Out[69]:

	KM Travelled	Price Charged	Cost of Trip	Population	Users	A
count	359392.000000	359392.000000	359392.000000	3.593920e+05	359392.000000	359392.0000
mean	22.567254	423.443311	286.190113	3.132198e+06	158365.582267	35.3367
std	12.233526	274.378911	157.993661	3.315194e+06	100850.051020	12.5942
min	1.900000	15.600000	19.000000	2.489680e+05	3643.000000	18.0000
25%	12.000000	206.437500	151.200000	6.712380e+05	80021.000000	25.0000
50%	22.440000	386.360000	282.480000	1.595037e+06	144132.000000	33.0000
75%	32.960000	583.660000	413.683200	8.405837e+06	302149.000000	42.0000
max	48.000000	2048.030000	691.200000	8.405837e+06	302149.000000	65.0000

In [70]:

```
sample_size = int((10/100)*359392) # Considering 10% values as sample data

def T_Test(a, b):
    sample_a = np.random.choice(a, sample_size)
    sample_b = np.random.choice(b, sample_size)
    ttest, p_value = ttest_ind(sample_a, sample_b, equal_var = False)
    print(f'p-value: {p_value}')
    if p_value < 0.05:    # alpha value is 0.05 or 5%
        print("We are rejecting null hypothesis (H0)")
    else:
        print("We are accepting null hypothesis (H0)")
```

H0 = Price charged by Pink, Yellow Cabs are same

H1 = Price charged by Pink, Yellow Cabs are not same

In [71]:

```
df['Price Charged per KM'].groupby(df['Company']).mean()
```

Out[71]:

```
Company
Pink Cab      13.768510
Yellow Cab    20.306073
Name: Price Charged per KM, dtype: float64
```

In [72]:

```
T_Test(df[df['Company'] == 'Yellow Cab']['Price Charged per KM'], df[df['Company'] == 'Pink
```

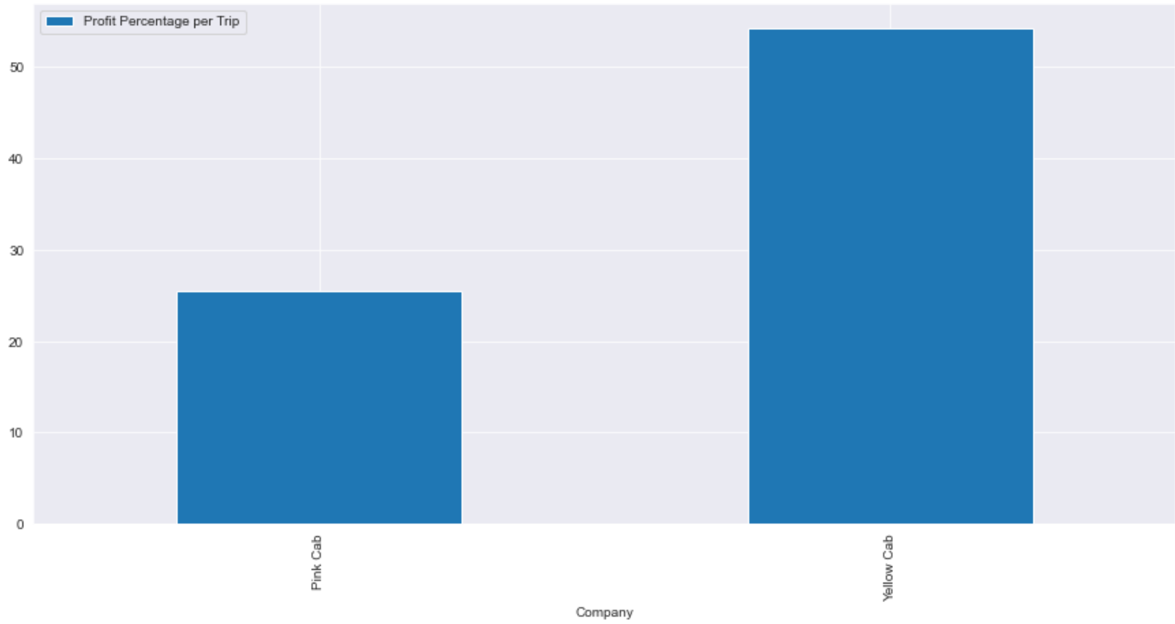
```
p-value: 0.0
We are rejecting null hypothesis (H0)
```

H0 = Profit Percentage per Trip is same for both cab service providers

H1 = Profit Percentage per Trip is not same for both cab service providers

In [73]:

```
df[['Profit Percentage per Trip', 'Company']].groupby('Company').mean().plot(kind='bar', fi
plt.show()
```



In [74]:

```
df['Profit Percentage per Trip'].groupby(df['Company']).mean()
```

Out[74]:

```
Company
Pink Cab      25.559567
Yellow Cab    54.296631
Name: Profit Percentage per Trip, dtype: float64
```

In [75]:

```
T_Test(df[df['Company'] == 'Yellow Cab']['Profit Percentage per Trip'], df[df['Company'] ==
```

```
p-value: 0.0
We are rejecting null hypothesis (H0)
```

H0 = Profit is same for both cab service providers

H1 = Profit is not same for both cab service providers

In [76]:

```
df['Profit'].groupby(df['Company']).mean()
```

Out[76]:

```
Company
Pink Cab       62.652174
Yellow Cab    160.259986
Name: Profit, dtype: float64
```

In [77]:

```
T_Test(df[df['Company'] == 'Yellow Cab']['Profit'], df[df['Company'] == 'Pink Cab']['Profit
```

```
p-value: 0.0
We are rejecting null hypothesis (H0)
```

H0 = Cost is same for both cab service providers

H1 = Cost is not same for both cab service providers

In [78]:

```
df['Cost of Trip'].groupby(df['Company']).mean()
```

Out[78]:

```
Company
Pink Cab      248.148682
Yellow Cab    297.922004
Name: Cost of Trip, dtype: float64
```

In [79]:

```
T_Test(df[df['Company'] == 'Yellow Cab']['Cost of Trip'], df[df['Company'] == 'Pink Cab']['
```

```
p-value: 0.0
We are rejecting null hypothesis (H0)
```

H0 = There is no difference in age of Male and Female users

H1 = There is difference in age of Male and Female users

In [80]:

```
df['Age'].groupby(df['Gender']).mean()
```

Out[80]:

```
Gender
Female    35.287608
Male      35.373300
Name: Age, dtype: float64
```

In [81]:

```
T_Test(df[df['Gender'] == 'Male']['Age'], df[df['Gender'] == 'Female']['Age'])
```

```
p-value: 0.864602959280944
We are accepting null hypothesis (H0)
```

H0 = Distance travelled by Male and Female are same

H1 = Distance travelled by Male and Female are not same

In [82]:

```
df['KM Travelled'].groupby(df['Gender']).mean()
```

Out[82]:

```
Gender
Female    22.586388
Male      22.552992
Name: KM Travelled, dtype: float64
```

In [83]:

```
T_Test(df[df['Gender'] == 'Male']['KM Travelled'], df[df['Gender'] == 'Female']['KM Travell
```

```
p-value: 0.9631237409158784
We are accepting null hypothesis (H0)
```

H0 = Distance travelled by Yellow Cab and Pink Cab are same

H1 = Distance travelled by Yellow Cab and Pink Cab are not same

In [84]:

```
df['KM Travelled'].groupby(df['Company']).mean()
```

Out[84]:

```
Company
Pink Cab      22.559917
Yellow Cab    22.569517
Name: KM Travelled, dtype: float64
```

In [85]:

```
T_Test(df[df['Company'] == 'Yellow Cab']['KM Travelled'], df[df['Company'] == 'Pink Cab']['
```

```
p-value: 0.5524789608160832
We are accepting null hypothesis (H0)
```

H0 = Profit is same for weekdays and weekends

H1 = Profit is not same for weekdays and weekends

In [86]:

```
df['Profit'].groupby(df['weekend']).mean()
```

Out[86]:

```
weekend
False    131.877574
True     146.557648
Name: Profit, dtype: float64
```

In [87]:

```
T_Test(df[df['weekend'] == False]['Profit'], df[df['weekend'] == True]['Profit'])
```

```
p-value: 2.7112015576384476e-50
We are rejecting null hypothesis (H0)
```

Chi2 Test

In [88]:

```
def check_relationship(crosstab_table, confidence_interval):
  statistic, p, dof, expected = stat.chi2_contingency(crosstab_table)
  print(f'Chi2 statistic value = {statistic}')
  print(f'p - value = {p}')
  print("Degree of Freedom: ", dof)
  alpha = 1.0 - confidence_interval

  if p <= alpha:
    print('Dependent, Reject Null Hypothesis (H0)')
  else:
    print('Independent, Accept Null Hypothesis (H0)')
```

H0: There is no Gender preference towards cab service provider

H1: There is Gender preference towards cab service provider

In [89]:

```
# Contingency Table
gender_company_ct = pd.crosstab(df['Gender'], df['Company'])
gender_company_ct
```

Out[89]:

Company	Pink Cab	Yellow Cab
Gender		
Female	37480	116000
Male	47231	158681

In [90]:

```
check_relationship(gender_company_ct, 0.95)
```

```
Chi2 statistic value = 107.22063897254299
p - value = 3.982674650131372e-25
Degree of Freedom:  1
Dependent, Reject Null Hypothesis (H0)
```

H0: There is no relationship between city and cab company preference

H1: There is relationship between city and cab company preference

In [91]:

```
# Contingency Table
city_company_ct = pd.crosstab(df['City'], df['Company'])
city_company_ct
```

Out[91]:

Company	Pink Cab	Yellow Cab
City		
ATLANTA GA	1762	5795
AUSTIN TX	1868	3028
BOSTON MA	5186	24506
CHICAGO IL	9361	47264
DALLAS TX	1380	5637
DENVER CO	1394	2431
LOS ANGELES CA	19865	28168
MIAMI FL	2002	4452
NASHVILLE TN	1841	1169
NEW YORK NY	13967	85918
ORANGE COUNTY	1513	2469
PHOENIX AZ	864	1200
PITTSBURGH PA	682	631
SACRAMENTO CA	1334	1033
SAN DIEGO CA	10672	9816
SEATTLE WA	2732	5265
SILICON VALLEY	3797	4722
TUCSON AZ	799	1132
WASHINGTON DC	3692	40045

In [92]:

```
check_relationship(city_company_ct, 0.95)
```

```
Chi2 statistic value = 39825.16829453775
p - value = 0.0
Degree of Freedom:  18
Dependent, Reject Null Hypothesis (H0)
```

H0: There is no relationship between payment mode and cab company

H1: There is relationship between payment mode and cab company

In [93]:

```
# Contingency Table
payment_company_ct = pd.crosstab(df['Payment_Mode'], df['Company'])
payment_company_ct
```

Out[93]:

Company	Pink Cab	Yellow Cab
Payment_Mode		
Card	50719	164785
Cash	33992	109896

In [94]:

```
check_relationship(payment_company_ct, 0.95)
```

```
Chi2 statistic value = 0.3733235887859897
p - value = 0.5411981778304723
Degree of Freedom:  1
Independent, Accept Null Hypothesis (H0)
```

H0: There is no relationship between weekday and cab company

H1: There is relationship between weekday and cab company

In [95]:

```
# Contingency Table
weekday_company_ct = pd.crosstab(df['Weekday'], df['Company'])
weekday_company_ct
```

Out[95]:

Company	Pink Cab	Yellow Cab
Weekday		
0	8700	28167
1	9145	29358
2	9028	29459
3	11251	35839
4	15666	51175
5	16097	52898
6	14824	47785

In [96]:

```
check_relationship(weekday_company_ct, 0.95)
```

```
Chi2 statistic value = 6.9521805581973
p - value = 0.32529218212054056
Degree of Freedom:  6
Independent, Accept Null Hypothesis (H0)
```

H0: There is no relationship between holiday and cab company

H1: There is relationship between holiday and cab company

In [97]:

```
# Contingency Table
holiday_company_ct = pd.crosstab(df['holiday'], df['Company'])
holiday_company_ct
```

Out[97]:

Company	Pink Cab	Yellow Cab
holiday		
False	82055	265338
True	2656	9343

In [98]:

```
check_relationship(holiday_company_ct, 0.95)
```

```
Chi2 statistic value = 14.116272948610183
p - value = 0.0001718506440957376
Degree of Freedom:  1
Dependent, Reject Null Hypothesis (H0)
```

Conclusion

- No duplicate data was found
- People prefer Yellow Cabs over Pink Cabs in every city except these 4:

1. Nashville
2. Pittsburgh
3. Sacramento
4. San Diego

- New York City has the highest Profit per KM while Sacramenyo has the lowest Profit per KM
- Avg distance travelled is 22.5 KM
- Over the weekends: distance travelled increases slightly => Profit increases
- Top 5 cities with highest avg profit (in descending order):

1. New York
2. Dallas
3. Silicon Valley
4. Miami
5. Orange County

- Except Chicago, Yellow Cab has more profit margin in each city.
- Around 30% of the population in Washington DC and Boston use cab services whereas for all other cities it's less than 10%
- Most of the users are aged between 20 to 40 years
- There is no discount for Female customers
- Avg Price Charged per KM for Yellow Cab is 20.3 USD & for Pink Cab is 13.76 USD
- Profit Percentage per Trip for
- Pink Cab is 25.559567
- Yellow Cab is 54.296631
- Mean Profit Percentage per Trip is 47.5%
- Profit is maximum in the weekends

In [ ]:

```

```