ZOMATO DATASET

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [3]:
```
df = pd.read_csv('zomato.csv',encoding='latin-1')
```

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

Out[4]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565443	French, Japanese, Desserts	...	Botswana Pula(P)
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553708	Japanese	...	Botswana Pula(P)
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581404	Seafood, Asian, Filipino, Indian	...	Botswana Pula(P)
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585318	Japanese, Sushi	...	Botswana Pula(P)
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584450	Japanese, Korean	...	Botswana Pula(P)

5 rows × 21 columns

In [5]:
```
df.columns
```

Out[5]:
```
Index(['Restaurant ID', 'Restaurant Name', 'Country Code', 'City', 'Address',
       'Locality', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines',
       'Average Cost for two', 'Currency', 'Has Table booking',
       'Has Online delivery', 'Is delivering now', 'Switch to order menu',
       'Price range', 'Aggregate rating', 'Rating color', 'Rating text',
       'Votes'],
      dtype='object')
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Restaurant ID         9551 non-null   int64
 1   Restaurant Name       9551 non-null   object
 2   Country Code          9551 non-null   int64
 3   City                  9551 non-null   object
 4   Address               9551 non-null   object
 5   Locality              9551 non-null   object
 6   Locality Verbose      9551 non-null   object
 7   Longitude             9551 non-null   float64
 8   Latitude              9551 non-null   float64
 9   Cuisines              9542 non-null   object
 10  Average Cost for two  9551 non-null   int64
 11  Currency              9551 non-null   object
 12  Has Table booking     9551 non-null   object
 13  Has Online delivery   9551 non-null   object
 14  Is delivering now     9551 non-null   object
 15  Switch to order menu  9551 non-null   object
 16  Price range           9551 non-null   int64
 17  Aggregate rating      9551 non-null   float64
 18  Rating color          9551 non-null   object
 19  Rating text           9551 non-null   object
 20  Votes                 9551 non-null   int64
dtypes: float64(3), int64(5), object(13)
memory usage: 1.5+ MB
```

In [7]:
```
df.describe()
```

Out[7]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	Votes
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000
mean	9.051128e+06	18.365616	64.126574	25.854381	1199.210763	1.804837	2.666370	156.909748
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516378	430.169145
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000000	0.000000
25%	3.019625e+05	1.000000	77.081343	28.478713	250.000000	1.000000	2.500000	5.000000
50%	6.004089e+06	1.000000	77.191964	28.570469	400.000000	2.000000	3.200000	31.000000
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700000	131.000000
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000	10934.000000

Let's try to find out if there are any missing values.

In [9]:
```
df.isnull().sum()
```

Out[9]:
```
Restaurant ID           0
Restaurant Name         0
Country Code            0
City                    0
Address                 0
Locality                0
Locality Verbose        0
Longitude               0
Latitude                0
Cuisines                9
Average Cost for two    0
Currency                0
Has Table booking       0
Has Online delivery     0
Is delivering now       0
Switch to order menu    0
Price range             0
Aggregate rating        0
Rating color            0
Rating text             0
Votes                   0
dtype: int64
```

In [12]:
```
[features for features in df.columns if df[features].isnull().sum()>0]
```

Out[12]:
```
['Cuisines']
```

Now let's import the other table too with name 'Country-Code.xlsx'

In [25]:
```
df_country=pd.read_excel('Country-Code.xlsx',engine='openpyxl' )
df_country.head()
```

Out[25]:

	Country Code	Country
0	1	India
1	14	Australia
2	30	Brazil
3	37	Canada
4	94	Indonesia

In [27]:
```
df_country.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Country Code  15 non-null     int64
 1   Country       15 non-null     object
dtypes: int64(1), object(1)
memory usage: 368.0+ bytes
```

In [28]:
```
df_country.describe()
```

Out[28]:

	Country Code
count	15.000000
mean	137.933333
std	80.009345
min	1.000000
25%	65.500000
50%	166.000000
75%	199.500000
max	216.000000

Now let's merge both the tables using a left join and the key column used for the merger will be 'Country Code'

In [29]:
```
final_df=pd.merge(df,df_country,on='Country Code',how='left')
```

In [30]:
```
final_df
```

Out[30]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Has Table booking
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565443	French, Japanese, Desserts	...	Yes
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553708	Japanese	...	Yes
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581404	Seafood, Asian, Filipino, Indian	...	Yes
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585318	Japanese, Sushi	...	No
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584450	Japanese, Korean	...	Yes
...	...	...	...	...	...	...	...	...	...	...	...	...
9546	5915730	NamlÛ± Gurme	208	ÛÁstanbul	Kemanke□ô Karamustafa Pa□ôa Mahallesi, RÛ±htÛ±...	Karakí_y	Karakí_y, ÛÁstanbul	28.977392	41.022793	Turkish	...	No
9547	5908749	Ceviz AÛôacÛ±	208	ÛÁstanbul	Ko□ôuyolu Mahallesi, Muhittin îistí_ndaÛô Cadd...	Ko□ôuyolu	Ko□ôuyolu, ÛÁstanbul	29.041297	41.009847	World Cuisine, Patisserie, Cafe	...	No
9548	5915807	Huqqa	208	ÛÁstanbul	Kuruí_e□ôme Mahallesi, Muallim Naci Caddesi, N...	Kuruí_e□ôme	Kuruí_e□ôme, ÛÁstanbul	29.034640	41.055817	Italian, World Cuisine	...	No
9549	5916112	A□ô□ôk Kahve	208	ÛÁstanbul	Kuruí_e□ôme Mahallesi, Muallim Naci Caddesi, N...	Kuruí_e□ôme	Kuruí_e□ôme, ÛÁstanbul	29.036019	41.057979	Restaurant Cafe	...	No
9550	5927402	Walter's Coffee Roastery	208	ÛÁstanbul	CafeaÛôa Mahallesi, BademaltÛ± Sokak, No 21/B,...	Moda	Moda, ÛÁstanbul	29.026016	40.984776	Cafe	...	No

9551 rows × 22 columns

Now let's perform some basic eda in the processed dataset.

In [39]:
```
country_names=final_df.Country.value_counts().index
```

In [40]:
```
print(country_names)
```

```
Index(['India', 'United States', 'United Kingdom', 'Brazil', 'UAE',
       'South Africa', 'New Zealand', 'Turkey', 'Australia', 'Phillipines',
       'Indonesia', 'Singapore', 'Qatar', 'Sri Lanka', 'Canada'],
      dtype='object')
```

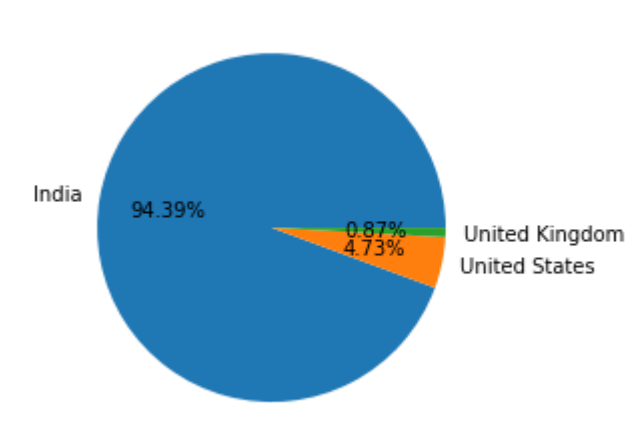
It is quite evident from the above data that most of the transactions of zomato are based primarily in India followed by United States and other countries. Zomato has identified three geographical segments — India, UAE and the rest of the world. The 'Rest of the World' category includes the US, UK, Singapore,and others. The company has decided to focus on India as it is its biggest market and UAE, which is profitable.

In [36]:
```
country_val= final_df.Country.value_counts().values
```

Creating a pie chart to visualise this in a better way.Will consider top 3 countries that use zomato to keep the chart clean

In [51]:
```
plt.pie(country_val[:3],labels=country_names[:3],autopct='%1.2f%%')
```

Out[51]:
```
([<matplotlib.patches.Wedge at 0x1c5be7eeb60>,
  <matplotlib.patches.Wedge at 0x1c5be7ef160>,
  <matplotlib.patches.Wedge at 0x1c5be7ef880>],
 [Text(-1.0829742700952103, 0.19278674827836725, 'India'),
  Text(1.077281715838356, -0.22240527134123297, 'United States'),
  Text(1.0995865153823035, -0.03015783794312073, 'United Kingdom')],
 [Text(-0.590713238233751, 0.10515640815183668, '94.39%'),
  Text(0.5876082086391032, -0.12131196618612707, '4.73%'),
  Text(0.5997744629358018, -0.01644972978715676, '0.87%')])
```



It is observed that 94.39% of zomato transitions take place in india followed by United states(4.73%) and United kingdom(0.87%)

Now let's analyse the ratings received from the customers

In [54]:
```
final_df.groupby(['Aggregate rating','Rating color','Rating text']).size()
```

Out[54]:
```
Aggregate rating  Rating color  Rating text
0.0               White         Not rated      2148
1.8               Red           Poor              1
1.9               Red           Poor              2
2.0               Red           Poor              7
2.1               Red           Poor             15
2.2               Red           Poor             27
2.3               Red           Poor             47
2.4               Red           Poor             87
2.5               Orange        Average         110
2.6               Orange        Average         191
2.7               Orange        Average         250
2.8               Orange        Average         315
2.9               Orange        Average         381
3.0               Orange        Average         468
3.1               Orange        Average         519
3.2               Orange        Average         522
3.3               Orange        Average         483
3.4               Orange        Average         498
3.5               Yellow        Good            480
3.6               Yellow        Good            458
3.7               Yellow        Good            427
3.8               Yellow        Good            400
3.9               Yellow        Good            335
4.0               Green         Very Good       266
4.1               Green         Very Good       274
4.2               Green         Very Good       221
4.3               Green         Very Good       174
4.4               Green         Very Good       144
4.5               Dark Green    Excellent        95
4.6               Dark Green    Excellent        78
4.7               Dark Green    Excellent        42
4.8               Dark Green    Excellent        25
4.9               Dark Green    Excellent        61
dtype: int64
```

In [61]:
```
ratings=final_df.groupby(['Aggregate rating','Rating color','Rating text']).size().reset_index().rename(columns
```

In [62]:
```
ratings
```

Out[62]:

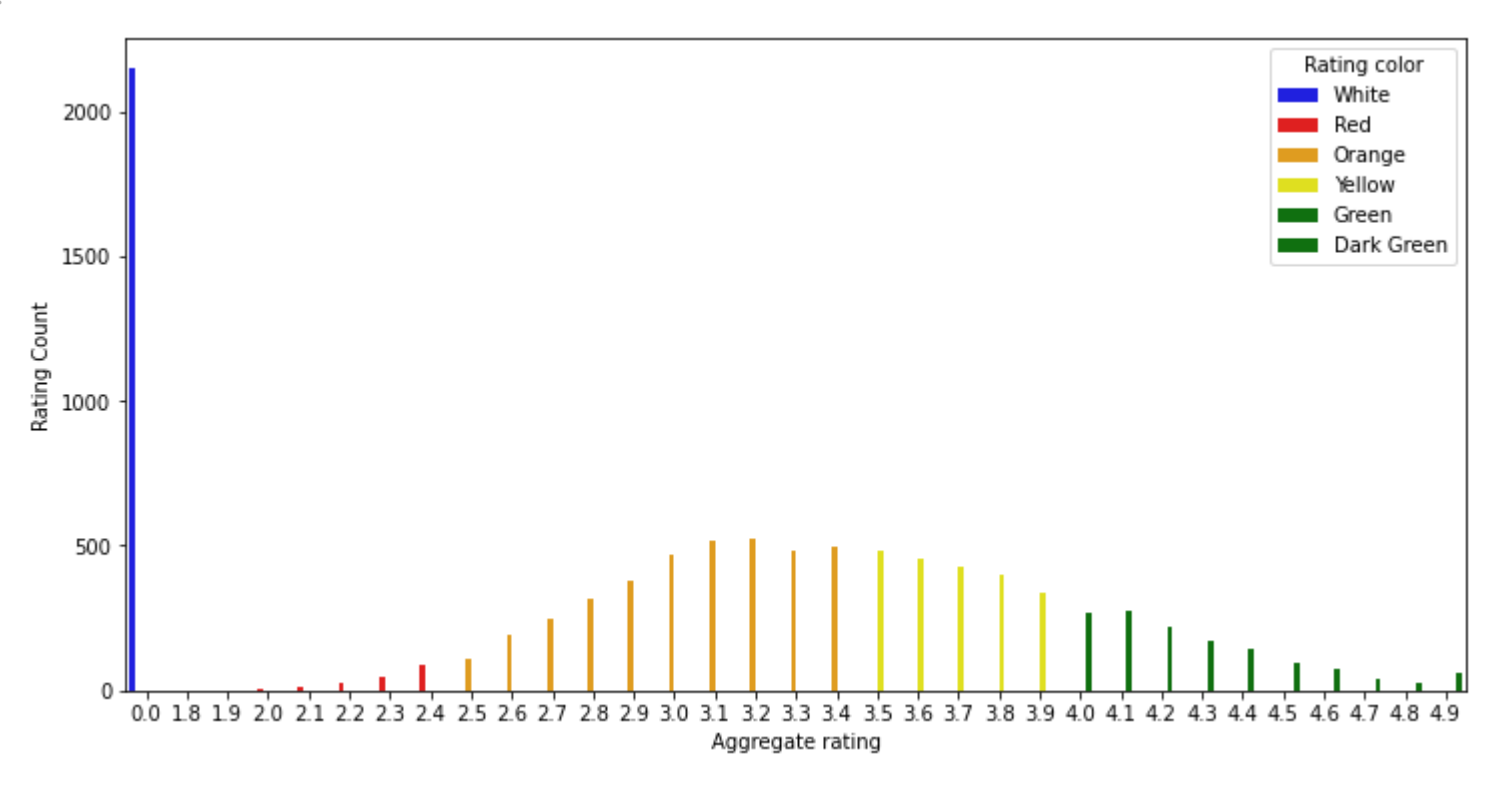
	Aggregate rating	Rating color	Rating text	Rating Count
0	0.0	White	Not rated	2148
1	1.8	Red	Poor	1
2	1.9	Red	Poor	2
3	2.0	Red	Poor	7
4	2.1	Red	Poor	15
5	2.2	Red	Poor	27
6	2.3	Red	Poor	47
7	2.4	Red	Poor	87
8	2.5	Orange	Average	110
9	2.6	Orange	Average	191
10	2.7	Orange	Average	250
11	2.8	Orange	Average	315
12	2.9	Orange	Average	381
13	3.0	Orange	Average	468
14	3.1	Orange	Average	519
15	3.2	Orange	Average	522
16	3.3	Orange	Average	483
17	3.4	Orange	Average	498
18	3.5	Yellow	Good	480
19	3.6	Yellow	Good	458
20	3.7	Yellow	Good	427
21	3.8	Yellow	Good	400
22	3.9	Yellow	Good	335
23	4.0	Green	Very Good	266
24	4.1	Green	Very Good	274
25	4.2	Green	Very Good	221
26	4.3	Green	Very Good	174
27	4.4	Green	Very Good	144
28	4.5	Dark Green	Excellent	95
29	4.6	Dark Green	Excellent	78
30	4.7	Dark Green	Excellent	42
31	4.8	Dark Green	Excellent	25
32	4.9	Dark Green	Excellent	61

It is observed that a major fraction of customers consisting of around 2148 records have not given any ratings. As Ratings play a very important role in making further strategies so Zomato needs to focus more on the customers ratings feedback and encourage more customer participation.

Let's visualize the above observation on a bar chart for better understanding of the stakeholders

In [86]:
```
import matplotlib
matplotlib.rcParams['figure.figsize']=(12,6)
sns.barplot(x='Aggregate rating',y='Rating Count',hue='Rating color',data=ratings,palette=['blue','red','orange
```

Out[86]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>
```

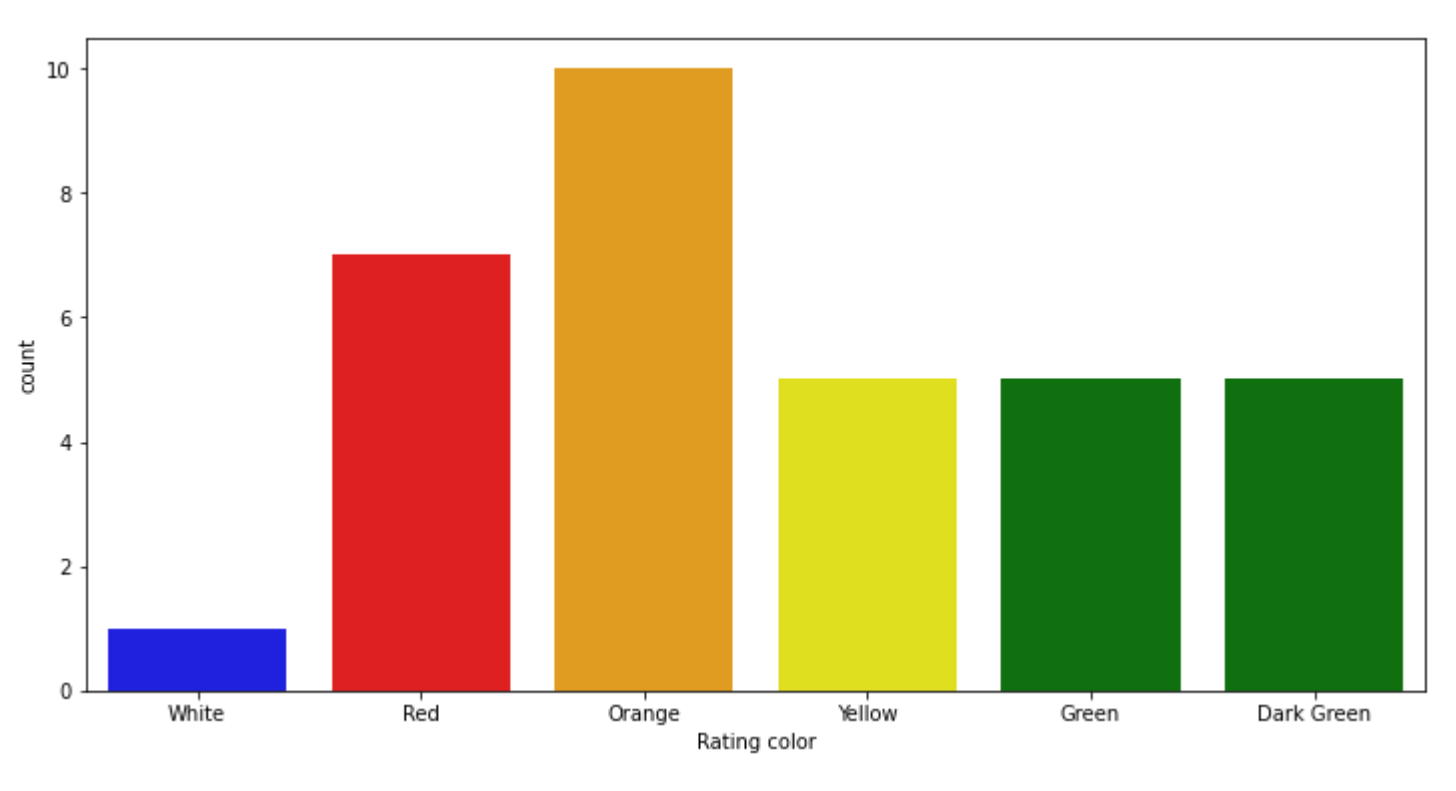


White has been intentionally labeled with blue color to make it visible in the chart

Also, it is evident that maximum user ratings are between 2.5 to 3.4

In [87]:
```
sns.countplot(x='Rating color',data=ratings,palette=['blue','red','orange','yellow','green','green'])
```

Out[87]:
```
<AxesSubplot:xlabel='Rating color', ylabel='count'>
```



Here count is the frequency of occourance of the different colors in the data frame.

Now lets find the count of 0 ratings country wise as special focus on these customers is very important for improving the operations of zomato

In [95]:
```
final_df[final_df['Rating color']=='White'].groupby('Country').size().reset_index()
```

Out[95]:

	Country	0
0	Brazil	5
1	India	2139
2	United Kingdom	1
3	United States	3

It is observed that the maximum number of 0 ratings are from India followed by brazil,UK and US

Now lets find all the countries who are offering online delivery option.

In [112...
```
final_df[final_df['Has Online delivery']=='Yes'].groupby('Country').size().reset_index()
```

Out[112]:

	Country	0
0	India	2423
1	UAE	28

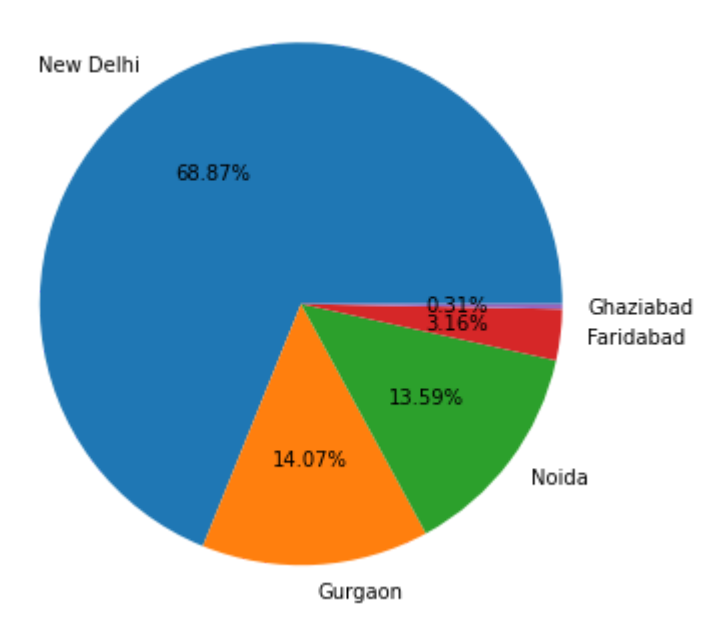
Only India and UAE are giving the option of online delivery to its customers. Zomato can work on this and try to provide the online delivery in other countries as well.

Now let's find the top 5 cities of india with maximum customer base

In [114...
```
city_values=final_df.City.value_counts().values
city_labels=final_df.City.value_counts().index
```

In [117...
```
plt.pie(city_values[:5],labels=city_labels[:5],autopct='%1.2f%%')
```

Out[117]:
```
([<matplotlib.patches.Wedge at 0x1c5c6cd7160>,
  <matplotlib.patches.Wedge at 0x1c5c6cd7760>,
  <matplotlib.patches.Wedge at 0x1c5c6cd7e80>,
  <matplotlib.patches.Wedge at 0x1c5c6cf85e0>,
  <matplotlib.patches.Wedge at 0x1c5c6cf8d00>],
 [Text(-0.6145352824185932, 0.9123301960708633, 'New Delhi'),
  Text(0.0623675251198054, -1.0982305276263407, 'Gurgaon'),
  Text(0.8789045225625368, -0.6614581167535246, 'Noida'),
  Text(1.0922218418223437, -0.13058119407559224, 'Faridabad'),
  Text(1.099946280005612, -0.010871113182029924, 'Ghaziabad')],
 [Text(-0.3352010631374145, 0.497634652402289, '68.87%'),
  Text(0.0340186500653484, -0.5990348332507311, '14.07%'),
  Text(0.47940246685229276, -0.36079533641101336, '13.59%'),
  Text(0.5957573682667329, -0.07122610585941394, '3.16%'),
  Text(0.5999706981848791, -0.005929698099289049, '0.31%')])
```



It is observed that New delhi has maximum number of transactions in the top 5 cities of India followed by Gurgaon and Noida So Zomato should introduce more schemes in these city to increase transaction rates.

linkedin - https://www.linkedin.com/in/nishant-gaurav-4b2753230/

In [ ]: