Project Description

About:

This dataset is all about the applications available on the Google Play Store. Basically, in this dataset, we will try to understand the trend of the Google Play store market. This analysis will help our team in python ka chilla 2.0 to develop the best application constraints according to the google play store market.

Description:

We have downloaded the dataset from Kaggle named “Playstore.csv”. This dataset contains the following attributes (Columns): 1) App :- Name of the App 2) Category :- Category under which the App falls. 3) Rating :- Application's rating on the play store 4) Reviews :- Number of reviews of the App. 5) Size :- Size of the App. 6) Install :- Number of Installs of the App 7) Type :- If the App is free/paid 8) Price :- Price of the app (0 if it is Free) 9) Content Rating :- Appropriate Target Audience of the App. 10) Genres:- Genre under which the App falls. 11) Last Updated :- Date when the App was last updated 12) Current Ver :- Current Version of the Application 13) Android Ver :- Minimum Android Version required to run the App

Methodology

EDA analysis on google play store dataset.

When we have to install any app from play store, we have to keep in mind the following points to reach a useful app for us: • The app reviews. • Number of people installed the app. • Size of application according to my device capacity. • App is paid or unpaid? • Under what category, this app lies? • The Rating of application.

Step-01: Import Libraries

In [63]:
```
import pandas as pd
import seaborn as sns
import numpy as np
import plotly.express as px
import matplotlib.pyplot as plt
import plotly.graph_objs as go
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
```

Step-02 : Load Dataset

In [4]:
```
app=pd.read_csv('Ali_Hasnain_PlayStore.csv')
```

In [5]:
```
app.head()
```

Out[5]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up

Exploratory Data Analysis (EDA)

Step-03: Shape of Data

In [8]:
```
rows, columns =app.shape
print('The total number of the rows are', rows)
print('The total number of the columns are', columns)
```

```
The total number of the rows are 10841
The total number of the columns are 13
```

Step-04: Data Structure

In [9]:
```
app.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10841 non-null  object 
 1   Category        10841 non-null  object 
 2   Rating          9367 non-null   float64
 3   Reviews         10841 non-null  object 
 4   Size            10841 non-null  object 
 5   Installs        10841 non-null  object 
 6   Type            10840 non-null  object 
 7   Price           10841 non-null  object 
 8   Content Rating  10840 non-null  object 
 9   Genres          10841 non-null  object 
 10  Last Updated    10841 non-null  object 
 11  Current Ver     10833 non-null  object 
 12  Android Ver     10838 non-null  object 
dtypes: float64(1), object(12)
memory usage: 592.9+ KB
```

In [10]:
```
app.describe(include='all').T
```

Out[10]:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
App	10841	9660	ROBLOX	9	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Category	10841	34	FAMILY	1972	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Rating	9367.0	NaN	NaN	NaN	4.193338	0.537431	1.0	4.0	4.3	4.5	19.0
Reviews	10841	6002	0	596	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Size	10841	462	Varies with device	1695	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Installs	10841	22	1,000,000+	1579	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Type	10840	3	Free	10039	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Price	10841	93	0	10040	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Content Rating	10840	6	Everyone	8714	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Genres	10841	120	Tools	842	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Last Updated	10841	1378	August 3, 2018	326	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Current Ver	10833	2832	Varies with device	1459	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Android Ver	10838	33	4.1 and up	2451	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Find the missing values

In [11]:
```
app.isnull().sum()
```

Out[11]:
```
App                 0
Category            0
Rating           1474
Reviews             0
Size                0
Installs            0
Type                1
Price               0
Content Rating      1
Genres              0
Last Updated        0
Current Ver         8
Android Ver         3
dtype: int64
```

Let's check the missing value with percentage

In [12]:
```
percent= app.isnull().sum().sort_values(ascending=False)/app.shape[0]*100
percent
```

Out[12]:
```
Rating            13.596532
Current Ver        0.073794
Android Ver        0.027673
Type               0.009224
Content Rating     0.009224
App                0.000000
Category           0.000000
Reviews            0.000000
Size               0.000000
Installs           0.000000
Price              0.000000
Genres             0.000000
Last Updated       0.000000
dtype: float64
```

Let's clean the dataset

- Creating a variable named 'Clean_Data' and copy the dataset into that variable for data wrangling.

In [13]:
```
Clean_Data = app.copy()
```

In [14]:
```
Clean_Data
```

Out[14]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Curre V
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & Design	January 7, 2018	1.0
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018	1.2
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & Design	June 8, 2018	Varie wi devic
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Art & Design;Creativity	June 20, 2018	1
...	...	...	...	...	...	...	...	...	...	...	...	
10836	Sya9a Maroc - FR	FAMILY	4.5	38	53M	5,000+	Free	0	Everyone	Education	July 25, 2017	1.4
10837	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3.6M	100+	Free	0	Everyone	Education	July 6, 2018	1
10838	Parkinson Exercices FR	MEDICAL	NaN	3	9.5M	1,000+	Free	0	Everyone	Medical	January 20, 2017	1
10839	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	Varies with device	1,000+	Free	0	Mature 17+	Books & Reference	January 19, 2015	Varie wi devic
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19M	10,000,000+	Free	0	Everyone	Lifestyle	July 25, 2018	Varie wi devic

10841 rows × 13 columns

As price column is not consider as a numeric datatype because it contains '$' symbol so we are replacing that symbol.

In [15]:
```
Clean_Data['Price'] = Clean_Data['Price'].str.replace('$','')
Clean_Data['Price'] = Clean_Data['Price'].str.replace('Everyone','0')
Clean_Data['Price'] = Clean_Data['Price'].astype('float')
```

```
c:\users\k.shahzad\appdata\local\programs\python\python37-32\lib\site-packages\ipykernel_launcher.py:1: FutureWa
rning: The default value of regex will change from True to False in a future version. In addition, single charac
ter regular expressions will *not* be treated as literal strings when regex=True.
  """Entry point for launching an IPython kernel.
```

- Check the datatype of price column, does it really change or not?

In [16]:
```
Clean_Data['Price'].dtypes
```

Out[16]:
```
dtype('float64')
```

In reviews column, there is 'M', called million, so we have to replace that 'M'

In [17]:
```
Clean_Data['Reviews'] = Clean_Data['Reviews'].str.replace('M','')
Clean_Data['Reviews'] = Clean_Data['Reviews'].astype('float')
```

In [18]:
```
Clean_Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10841 non-null  object 
 1   Category        10841 non-null  object 
 2   Rating          9367 non-null   float64
 3   Reviews         10841 non-null  float64
 4   Size            10841 non-null  object 
 5   Installs        10841 non-null  object 
 6   Type            10840 non-null  object 
 7   Price           10841 non-null  float64
 8   Content Rating  10840 non-null  object 
 9   Genres          10841 non-null  object 
 10  Last Updated    10841 non-null  object 
 11  Current Ver     10833 non-null  object 
 12  Android Ver     10838 non-null  object 
dtypes: float64(3), object(10)
memory usage: 677.6+ KB
```

In Installs column, there is '+', ',' symbol that is why it is not lying in numeric datatype so replace it

In [19]:
```
Clean_Data['Installs'] = Clean_Data['Installs'].str.replace('+','')
Clean_Data['Installs'] = Clean_Data['Installs'].str.replace(',','')
Clean_Data['Installs'] = Clean_Data['Installs'].str.replace('Free', '0')
Clean_Data['Installs'] = Clean_Data['Installs'].astype('float')
```

```
c:\users\k.shahzad\appdata\local\programs\python\python37-32\lib\site-packages\ipykernel_launcher.py:1: FutureWa
rning: The default value of regex will change from True to False in a future version. In addition, single charac
ter regular expressions will *not* be treated as literal strings when regex=True.
  """Entry point for launching an IPython kernel.
```

In [ ]:
```
Clean_Data
```

In Last Updated column we are having the value of dates but there is a value contain comma so we can change it

In [20]:
```
Clean_Data['Last Updated'] = Clean_Data['Last Updated'].str.replace(',','')
```

Now we can add date and time into 'Last Updated' column

In [21]:
```
from datetime import date
Clean_Data['Last Updated']= pd.to_datetime(Clean_Data['Last Updated'], errors='coerce')
Clean_Data
```

Out[21]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Curre Ve
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159.0	19M	10000.0	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14M	500000.0	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510.0	8.7M	5000000.0	Free	0.0	Everyone	Art & Design	2018-08-01	1.2.
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644.0	25M	50000000.0	Free	0.0	Teen	Art & Design	2018-06-08	Varie wi devic
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2.8M	100000.0	Free	0.0	Everyone	Art & Design;Creativity	2018-06-20	1.
...	...	...	...	...	...	...	...	...	...	...	...	.
10836	Sya9a Maroc - FR	FAMILY	4.5	38.0	53M	5000.0	Free	0.0	Everyone	Education	2017-07-25	1.4
10837	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4.0	3.6M	100.0	Free	0.0	Everyone	Education	2018-07-06	1.
10838	Parkinson Exercices FR	MEDICAL	NaN	3.0	9.5M	1000.0	Free	0.0	Everyone	Medical	2017-01-20	1.
10839	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114.0	Varies with device	1000.0	Free	0.0	Mature 17+	Books & Reference	2015-01-19	Varie wi devic
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307.0	19M	10000000.0	Free	0.0	Everyone	Lifestyle	2018-07-25	Varie wi devic

10841 rows × 13 columns

Now If we look into 'Size' column, there is values in kb's and Mb's represented as 'k' and 'M' respectively,

- so we have to change it and bring into one measure unit that is Mb.

- There is the also text data 'varies with devices' in size column, so we are going to replace it by 0

In [23]:
```
Clean_Data_MB=Clean_Data[Clean_Data['Size'].str.contains('M')]
Clean_Data_KB= Clean_Data[Clean_Data['Size'].str.contains('k')]
Clean_Data_VWD= Clean_Data[Clean_Data['Size']=='Varies with device']
```

In [24]:
```
Clean_Data_KB['Size']
```

Out[24]:
```
58        201k
209        23k
384        79k
450       118k
458       695k
         ...
10763     552k
10764     885k
10798    1020k
10832     582k
10833     619k
Name: Size, Length: 316, dtype: object
```

In [25]:
```
Clean_Data_MB['Size']
```

Out[25]:
```
0         19M
1         14M
2        8.7M
3         25M
4        2.8M
         ...
10835    9.6M
10836     53M
10837    3.6M
10838    9.5M
10840     19M
Name: Size, Length: 8829, dtype: object
```

In [26]:
```
Clean_Data_VWD['Size']
```

Out[26]:
```
37       Varies with device
42       Varies with device
52       Varies with device
67       Varies with device
68       Varies with device
                 ...
10713    Varies with device
10725    Varies with device
10765    Varies with device
10826    Varies with device
10839    Varies with device
Name: Size, Length: 1695, dtype: object
```

In [27]:
```
Clean_Data_MB['Size']=Clean_Data['Size'].str.replace('M','')
Clean_Data_KB['Size']=Clean_Data['Size'].str.replace('k','')
```

```
c:\users\k.shahzad\appdata\local\programs\python\python37-32\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-vi
ew-versus-a-copy)
  """Entry point for launching an IPython kernel.
c:\users\k.shahzad\appdata\local\programs\python\python37-32\lib\site-packages\ipykernel_launcher.py:2: SettingW
ithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-vi
ew-versus-a-copy)
```

With the help of loop, we are placing 0 where 'Varies with devices' are placed

In [28]:
```
Clean_Data_VWD['Size']= [0 for i in range(0,Clean_Data_VWD.shape[0])]
```

```
c:\users\k.shahzad\appdata\local\programs\python\python37-32\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-vi
ew-versus-a-copy)
  """Entry point for launching an IPython kernel.
```

Now we have to convert kb into Mb to have identical unit for all the values

In [29]:
```
Clean_Data_KB['Size']=Clean_Data_KB['Size'].apply(lambda x:str(float(x)/1000))
```

```
c:\users\k.shahzad\appdata\local\programs\python\python37-32\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-vi
ew-versus-a-copy)
  """Entry point for launching an IPython kernel.
```

Now we have done, just combine them together in one column again

In [30]:
```
Clean_Data= pd.concat([Clean_Data_MB,Clean_Data_KB,Clean_Data_VWD],axis=0)
```

In [31]:
```
Clean_Data
```

Out[31]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Curren Ve
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159.0	19	10000.0	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14	500000.0	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510.0	8.7	5000000.0	Free	0.0	Everyone	Art & Design	2018-08-01	1.2.
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644.0	25	50000000.0	Free	0.0	Teen	Art & Design	2018-06-08	Varie wit devic
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2.8	100000.0	Free	0.0	Everyone	Art & Design;Creativity	2018-06-20	1.
...	...	...	...	...	...	...	...	...	...	...	...	..
10713	My Earthquake Alerts - US & Worldwide Earthquakes	WEATHER	4.4	3471.0	0	100000.0	Free	0.0	Everyone	Weather	2018-07-24	Varie wit devic
10725	Posta App	MAPS_AND_NAVIGATION	3.6	8.0	0	1000.0	Free	0.0	Everyone	Maps & Navigation	2017-09-27	Varie wit devic
10765	Chat For Strangers - Video Chat	SOCIAL	3.4	622.0	0	100000.0	Free	0.0	Mature 17+	Social	2018-05-23	Varie wit devic
10826	Frim: get new friends on local chat rooms	SOCIAL	4.0	88486.0	0	5000000.0	Free	0.0	Mature 17+	Social	2018-03-23	Varie wit devic
10839	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114.0	0	1000.0	Free	0.0	Mature 17+	Books & Reference	2015-01-19	Varie wit devic

10840 rows × 13 columns

For better understanding, let's rename the column 'Size' to 'Size_in_MB', that will help to understand values for all

In [32]:
```
Clean_Data = Clean_Data.rename(columns={'Size':'Size_in_MB'})
```

In [33]:
```
Clean_Data['Current Ver']= Clean_Data['Current Ver'].replace('Varies with device',0)
Clean_Data['Android Ver']= Clean_Data['Android Ver'].replace('Varies with device',0)
```

In [34]:
```
Clean_Data.head()
```

Out[34]:

	App	Category	Rating	Reviews	Size_in_MB	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	And
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159.0	19	10000.0	Free	0.0	Everyone	Art & Design	2018-01-07	1.0.0	4 an
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14	500000.0	Free	0.0	Everyone	Art & Design;Pretend Play	2018-01-15	2.0.0	4 an
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510.0	8.7	5000000.0	Free	0.0	Everyone	Art & Design	2018-08-01	1.2.4	4 an
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644.0	25	50000000.0	Free	0.0	Teen	Art & Design	2018-06-08	0	4.2
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2.8	100000.0	Free	0.0	Everyone	Art & Design;Creativity	2018-06-20	1.1	4.4

Changing the datatype accordingly

In [35]:
```
Clean_Data['Size_in_MB'] = Clean_Data['Size_in_MB'].astype('float')
Clean_Data['Type'] = Clean_Data['Type'].astype('category')
Clean_Data['Content Rating'] = Clean_Data['Content Rating'].astype('category')
```

Now, let's check the dataset and it's type

In [36]:
```
Clean_Data.dtypes
```

Out[36]:
```
App                       object
Category                  object
Rating                   float64
Reviews                  float64
Size_in_MB               float64
Installs                 float64
Type                    category
Price                    float64
Content Rating          category
Genres                    object
Last Updated      datetime64[ns]
Current Ver               object
Android Ver               object
dtype: object
```

In [37]:
```
Clean_Data.describe()
```

Out[37]:

	Rating	Reviews	Size_in_MB	Installs	Price
count	9366.000000	1.084000e+04	10840.000000	1.084000e+04	10840.000000
mean	4.191757	4.441529e+05	18.152091	1.546434e+07	1.027368
std	0.515219	2.927761e+06	22.170606	8.502936e+07	15.949703
min	1.000000	0.000000e+00	0.000000	0.000000e+00	0.000000
25%	4.000000	3.800000e+01	2.600000	1.000000e+03	0.000000
50%	4.300000	2.094000e+03	9.200000	1.000000e+05	0.000000
75%	4.500000	5.477550e+04	26.000000	5.000000e+06	0.000000
max	5.000000	7.815831e+07	100.000000	1.000000e+09	400.000000

- Now we are dealing with missing values

In [38]:
```
Clean_Data.isnull().sum()
```

Out[38]:
```
App                 0
Category            0
Rating           1474
Reviews             0
Size_in_MB          0
Installs            0
Type                1
Price               0
Content Rating      0
Genres              0
Last Updated        0
Current Ver         8
Android Ver         2
dtype: int64
```

- Let's look at the null values

In [39]:
```
Clean_Data[Clean_Data.Type.isnull()]
```

Out[39]:

	App	Category	Rating	Reviews	Size_in_MB	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
9148	Command & Conquer: Rivals	FAMILY	NaN	0.0	0.0	0.0	NaN	0.0	Everyone 10+	Strategy	2018-06-28	0	0

- As price is equal to 0 ,so we can fill NaN value with 'free'.

In [40]:
```
Clean_Data["Type"].fillna("Free",inplace=True)
```

In [41]:
```
Clean_Data.dropna(subset=["Current Ver"],inplace=True)
Clean_Data.dropna(subset=["Android Ver"],inplace=True)
```

- For missing values in rating column, we can use backward fill method to handle the missing values

In [42]:
```
Clean_Data['Rating'].fillna(method = 'bfill', axis = 0, inplace=True)
```

In [43]:
```
Clean_Data.isnull().sum()
```

Out[43]:
```
App               0
Category          0
Rating            0
Reviews           0
Size_in_MB        0
Installs          0
Type              0
Price             0
Content Rating    0
Genres            0
Last Updated      0
Current Ver       0
Android Ver       0
dtype: int64
```

As we have done the cleaning, but if we have the duplicate values so we can also remove to them

In [44]:
```
Clean_Data.drop_duplicates(inplace=True)
```

Check the difference b/w cleaned and un cleaned data

In [45]:
```
print('The size of data before cleaning is', app.size)
print('The size of data after cleaning is', Clean_Data.size)
```

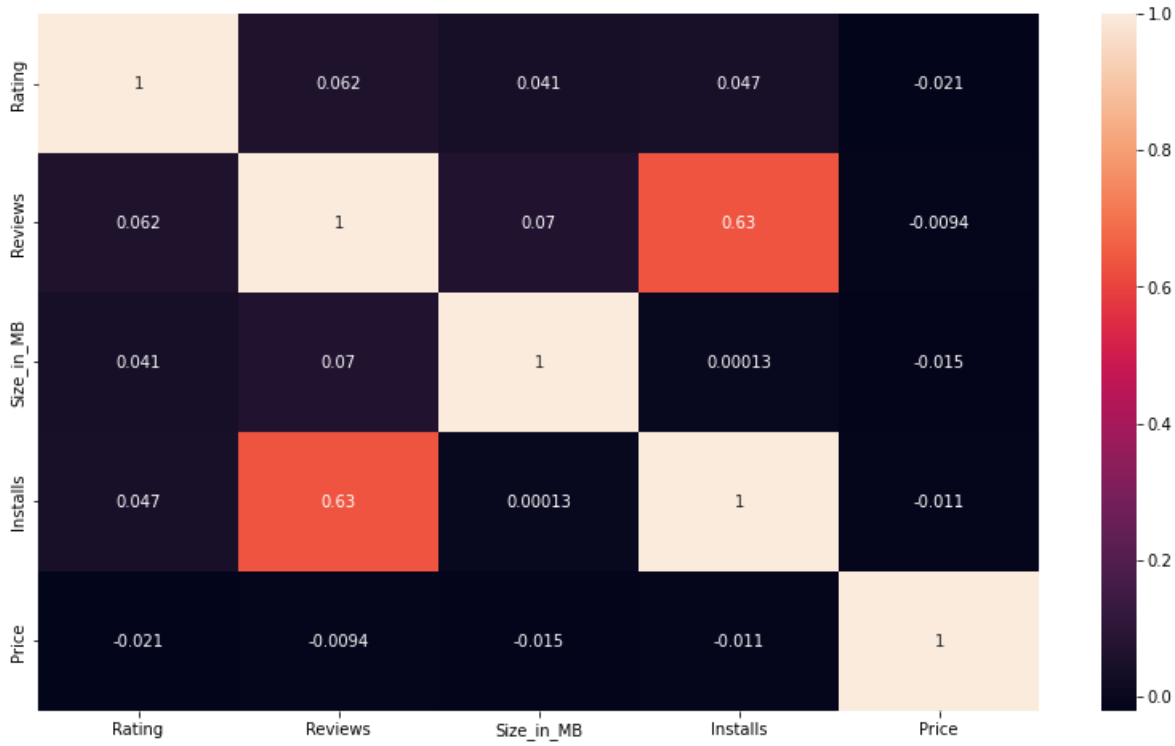
```
The size of data before cleaning is 140933
The size of data after cleaning is 134615
```

Begin With The visualization Now

- Correlation

In [46]:
```
plt.figure(figsize=(14,8))
sns.heatmap(Clean_Data.corr(),annot=True)
```

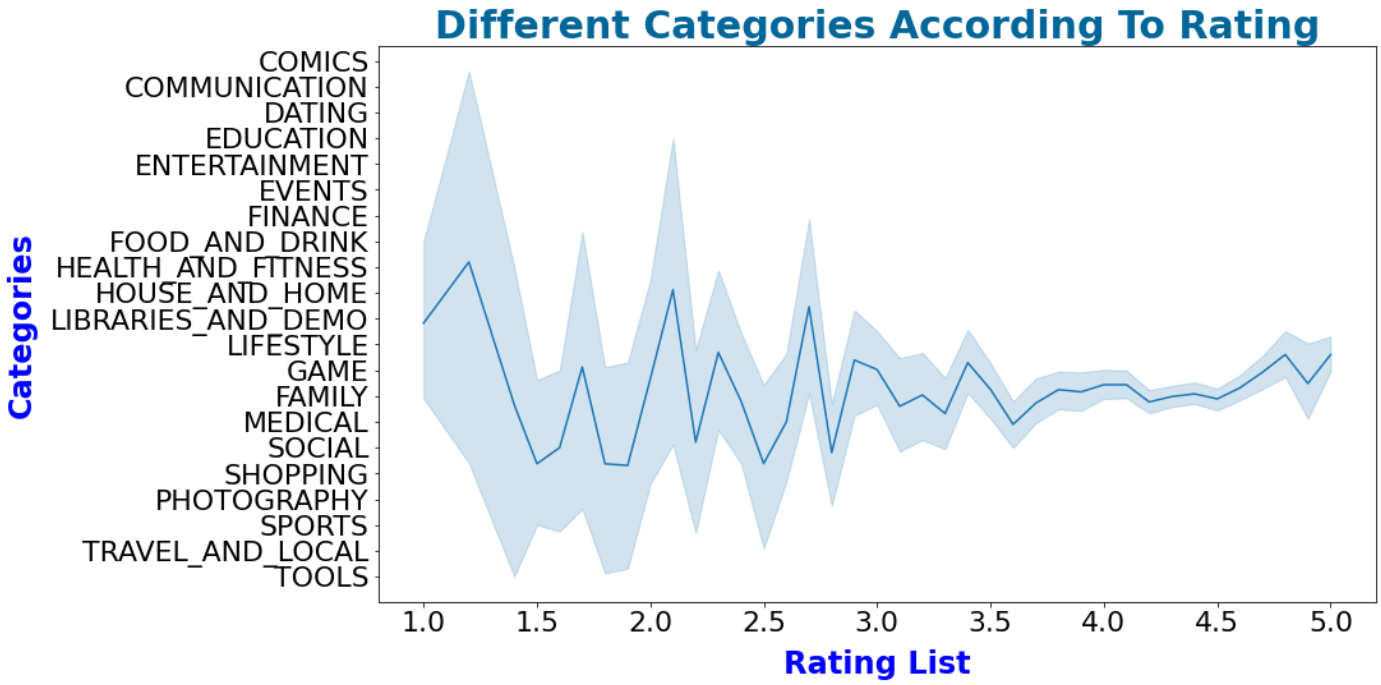
Out[46]: <AxesSubplot:>



- If we look to the graph, we will observe that installs and reviews are highly corelated , as we all know if the review is going to be high it means that installs traffic will also be increase

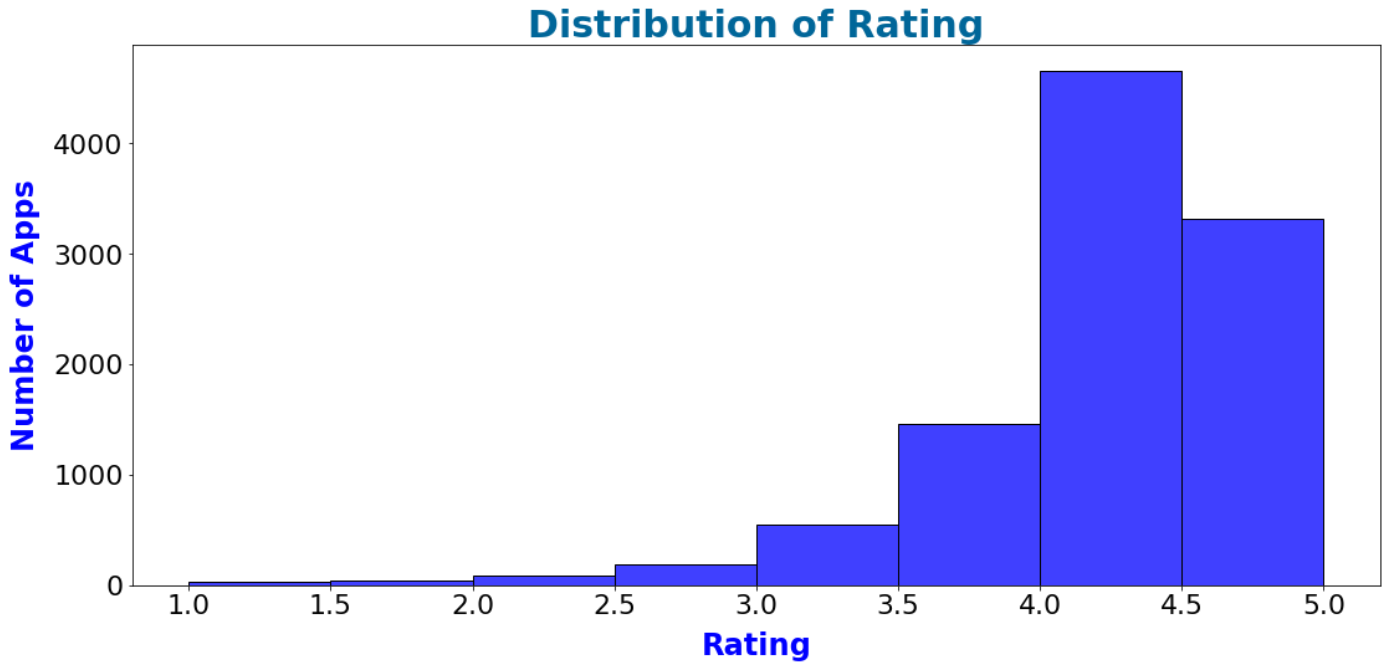
First of all, it is good practice to see the distribution of different categories with respect to rating.

In [47]:
```
plt.figure(figsize=(14,8))
line_fig=sns.lineplot(x='Rating',y='Category', data=Clean_Data)
line_fig.set_title('Different Categories According To Rating', fontsize=30, fontweight='bold', color='#006699
line_fig.set_xlabel('Rating List', fontsize=24, fontweight='bold', color='blue', labelpad=10, rotation=0, ha=
line_fig.set_ylabel('Categories', fontsize=24, fontweight='bold', color='blue', labelpad=10, rotation=90, ha=
line_fig.tick_params(labelsize=22)
```



Here we are analyzing the number of apps with respect to their rating

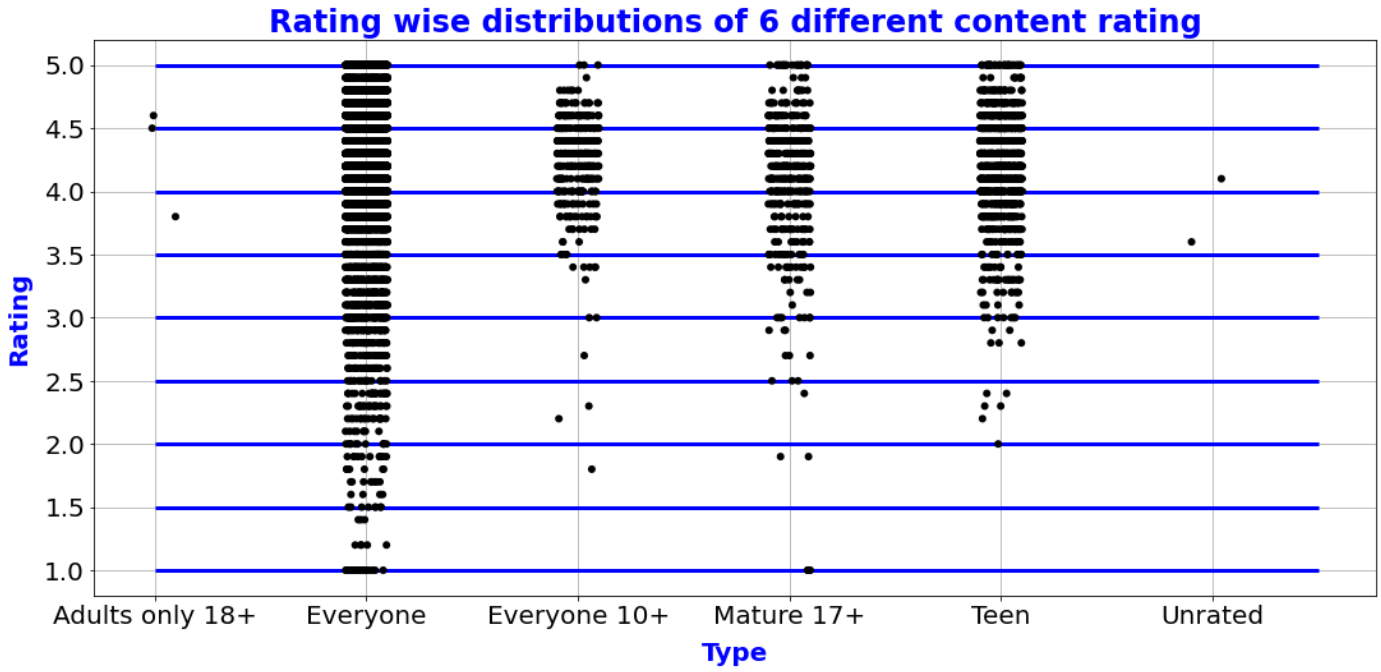
In [48]:
```
plt.figure(figsize=(18,8))
total=sns.histplot(x='Rating', data=Clean_Data, bins=8, color='blue')
total.set_title('Distribution of Rating', fontsize=30, fontweight='bold', color='#006699')
total.set_xlabel('Rating', fontsize=24, fontweight='bold', color='blue', labelpad=10, rotation=0, ha='center'
total.set_ylabel('Number of Apps', fontsize=24, fontweight='bold', color='blue', labelpad=10, rotation=90, ha
total.tick_params(labelsize=22)
```



Now we are analyzing the contents of apps according to their ratings

In [49]:
```
plt.figure(figsize=(18,8))
dist_fig=sns.stripplot(x='Content Rating', y='Rating', data=Clean_Data, jitter=True ,  edgecolor="black", lin
               size=3)
dist_fig.set_title('Rating wise distributions of 6 different content rating', fontsize=24, fontweight='bold',
dist_fig.set_xlabel('Type', fontsize=20, fontweight='bold', color='blue', labelpad=10, rotation=0, ha='center
dist_fig.set_ylabel('Rating', fontsize=20,fontweight='bold', color='blue',labelpad=10, rotation=90, ha='cente
dist_fig.tick_params(labelsize=20)
dist_fig.grid(True)
dist_fig.hlines(y=[5,4.5,4.0,3.5,3.0,2.5,2.0,1.5,1.0], xmin=0, xmax=dist_fig.get_xlim()[1], linewidth=3, colo
```

Out[49]: <matplotlib.collections.LineCollection at 0x592c530>



Let's make group according app type and number of installs

In [50]:
```
group1=Clean_Data.groupby('Type')['Installs'].count()
group1=pd.DataFrame(group1)
group1=group1.reset_index()
group1
```

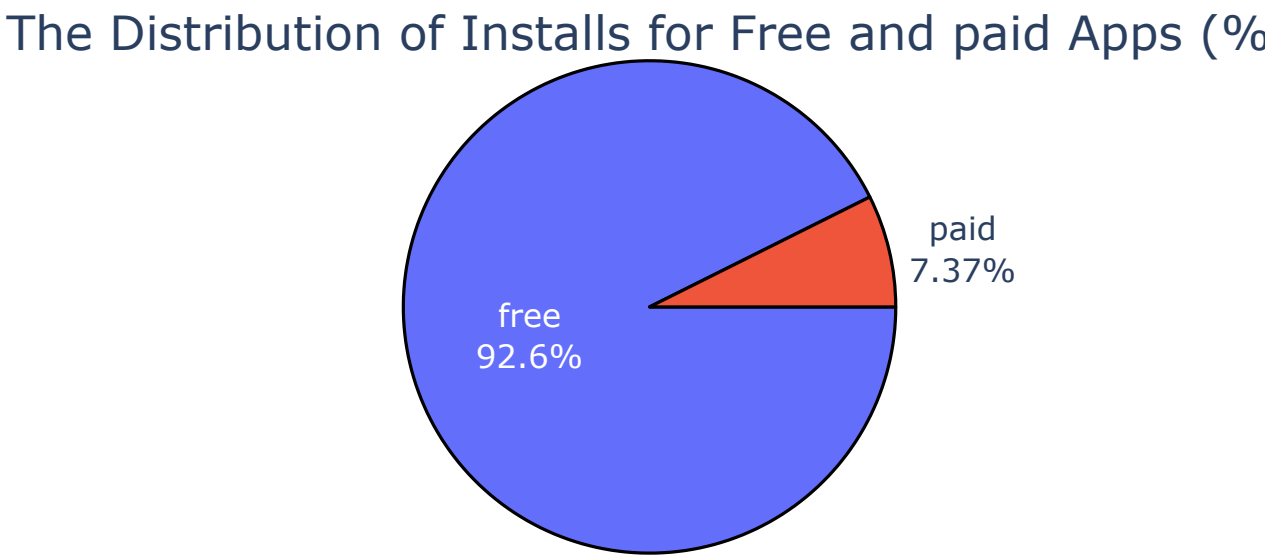
Out[50]:

	Type	Installs
0	Free	9592
1	Paid	763

Let's make it interactive with visualization

In [64]:

```
group= ['free','paid']
color= ['#d32c58','#f7b6d2']
traces = go.Pie(labels=group, values=group1['Installs'], hoverinfo='label+percent',
                textinfo='value', textfont=dict(size=20), marker=dict(colors=color, line=dict(color='#000000'
traces.title='The Distribution of Installs for Free and paid Apps (%)'
traces.titlefont=dict(size=30)
traces.rotation=90
traces.textpositionsrc='relative'
traces.automargin=False
traces.texttemplate='%{label}<br>%{percent}'
traces.marker=dict(line=dict(color='#000000', width=2))
iplot([traces])
```



Now let's see the top 10 Categories according to rating

In [52]:

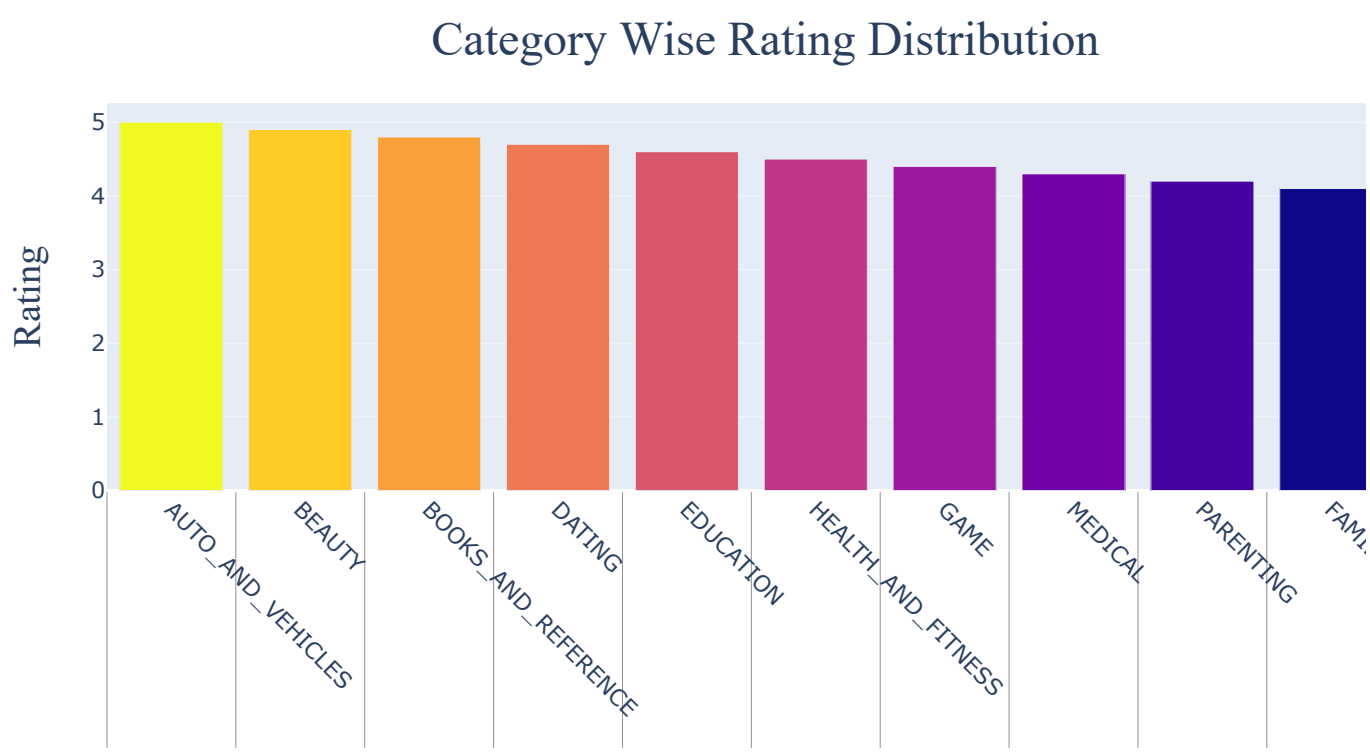
```
fig2=Clean_Data.groupby('Rating')['Category'].unique().sort_index(ascending=False).head(10)
fig2=pd.DataFrame(fig2)
fig2=fig2.reset_index()
fig2
```

Out[52]:

	Rating	Category
0	5.0	[COMICS, DATING, EVENTS, LIBRARIES_AND_DEMO, M...
1	4.9	[AUTO_AND_VEHICLES, BEAUTY, BOOKS_AND_REFERENC...
2	4.8	[ART_AND_DESIGN, AUTO_AND_VEHICLES, BEAUTY, BO...
3	4.7	[ART_AND_DESIGN, BEAUTY, BOOKS_AND_REFERENCE, ...
4	4.6	[ART_AND_DESIGN, AUTO_AND_VEHICLES, BEAUTY, BO...
5	4.5	[ART_AND_DESIGN, AUTO_AND_VEHICLES, BEAUTY, BO...
6	4.4	[ART_AND_DESIGN, AUTO_AND_VEHICLES, BEAUTY, BO...
7	4.3	[ART_AND_DESIGN, AUTO_AND_VEHICLES, BEAUTY, BO...
8	4.2	[ART_AND_DESIGN, AUTO_AND_VEHICLES, BEAUTY, BO...
9	4.1	[ART_AND_DESIGN, BEAUTY, BOOKS_AND_REFERENCE, ...

In [65]:

```
draw = px.bar(
    fig2,
    y='Rating',
    x='Category',
    orientation='v',
    color='Rating',
)
draw.update_layout(title_text='Category Wise Rating Distribution', title_x=0.5,
                   xaxis_title="Category", yaxis_title="Rating",
                   title_font_size=30, title_font_family="Times New Roman",
                    xaxis_title_font_size=24, xaxis_title_font_family="Times New Roman",
                    yaxis_title_font_size=24, yaxis_title_font_family="Times New Roman",
                    xaxis_tickfont_size=14, yaxis_tickfont_size=14, xaxis_tickangle=45
                   )
```



Let's see the top 3 apps who got most reviews

In [54]:
```
fig3= Clean_Data.sort_values(by = 'Reviews',ascending=False).head()
fig3.drop_duplicates(subset=['App'], keep="first", inplace=True)
fig3
```

Out[54]:

	App	Category	Rating	Reviews	Size_in_MB	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver
2544	Facebook	SOCIAL	4.1	78158306.0	0.0	1.000000e+09	Free	0.0	Teen	Social	2018-08-03	0
336	WhatsApp Messenger	COMMUNICATION	4.4	69119316.0	0.0	1.000000e+09	Free	0.0	Everyone	Communication	2018-08-03	0
2604	Instagram	SOCIAL	4.5	66577446.0	0.0	1.000000e+09	Free	0.0	Teen	Social	2018-07-31	0

In [66]:
```
fig_3 = px.bar(
    fig3,
    y='Reviews',
    x='App',
    orientation='v',
    color='App',
    title= 'Apps with Most Reviews'
)
fig_3.update_layout(title_text='Top 3 Apps With Most Reviews', title_x=0.5,
                    xaxis_title="App", yaxis_title="Reviews",
                    title_font_size=30, title_font_family="Times New Roman",
                    xaxis_title_font_size=24, xaxis_title_font_family="Times New Roman",
                    yaxis_title_font_size=24, yaxis_title_font_family="Times New Roman",
                    xaxis_tickfont_size=14, yaxis_tickfont_size=14

                    )
```

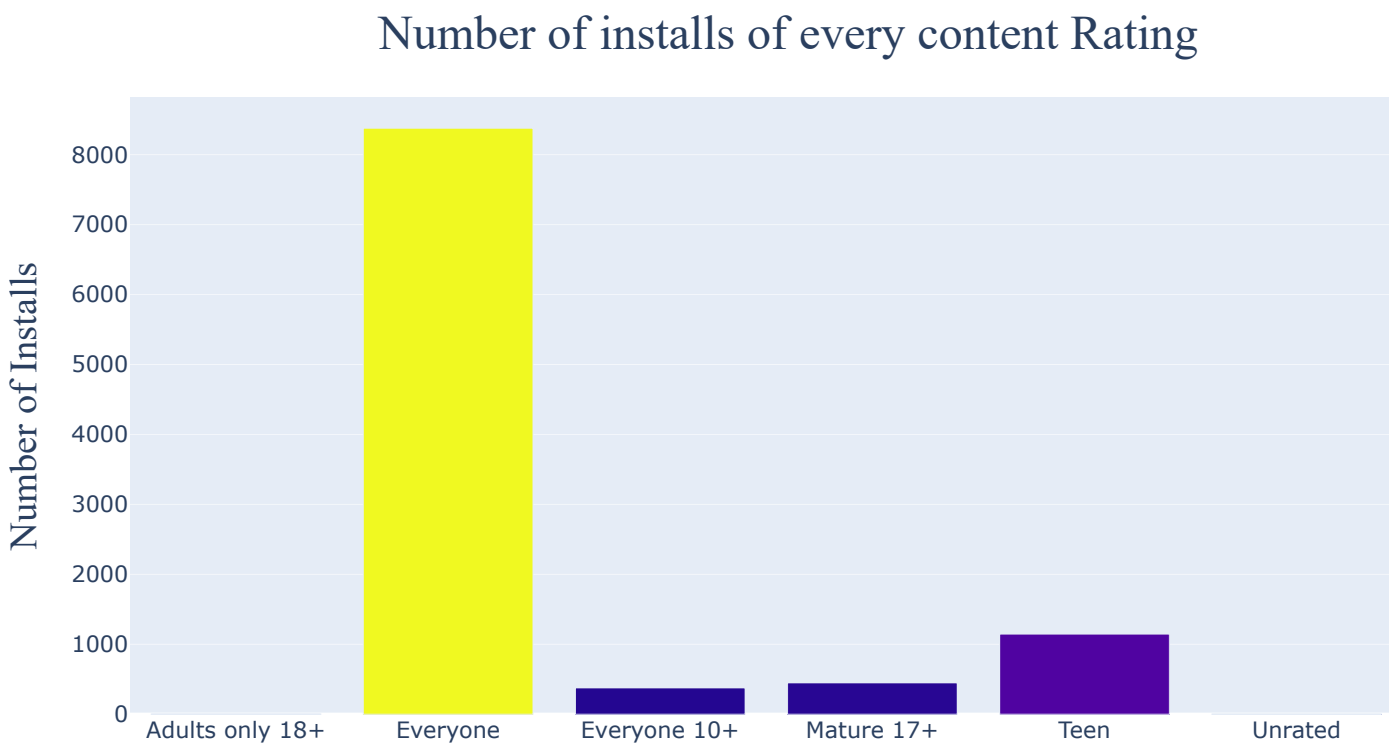
Now let's check the content rating positions according to number of installs

In [56]:
```
checking =Clean_Data.groupby('Content Rating')['Installs'].count()
checking =pd.DataFrame(checking)
checking=checking.reset_index()
checking
```

Out[56]:

	Content Rating	Installs
0	Adults only 18+	3
1	Everyone	8378
2	Everyone 10+	377
3	Mature 17+	449
4	Teen	1146
5	Unrated	2

In [67]:
```
checking_fig= px.bar(
    checking,
    x='Content Rating',
    y='Installs',
    color='Installs'
)
checking_fig.update_layout(title_text='Number of installs of every content Rating ', title_x=0.5,
                           xaxis_title="Content Rating", yaxis_title="Number of Installs",
                           title_font_size=30, title_font_family="Times New Roman",
                           xaxis_title_font_size=24, xaxis_title_font_family="Times New Roman",
                           yaxis_title_font_size=24, yaxis_title_font_family="Times New Roman",
                           xaxis_tickfont_size=14, yaxis_tickfont_size=14
                          )
```



Now we can check, which category contain maximum apps

In [58]:

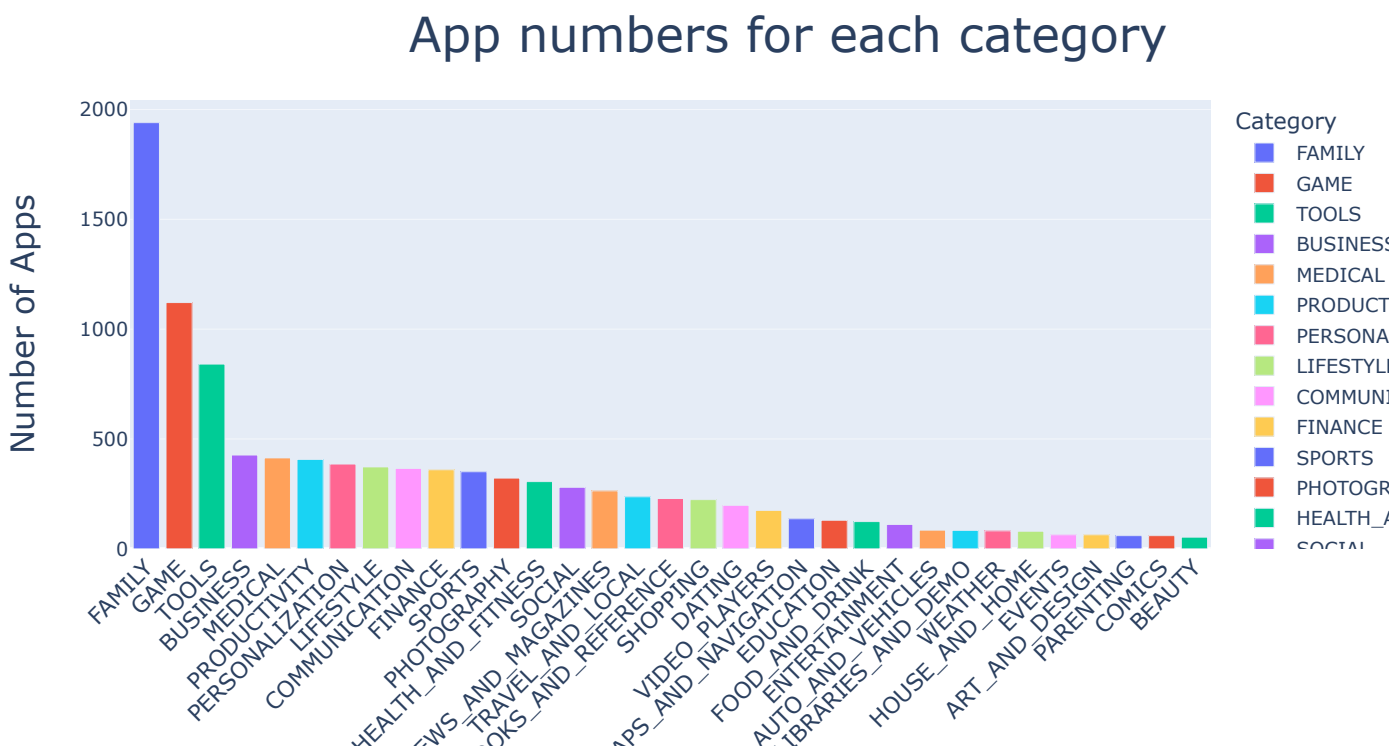
```
ali = Clean_Data.groupby('Category')['App'].count().sort_values(ascending=False)
ali =pd.DataFrame(ali)
ali=ali.reset_index()
ali
```

Out[58]:

	Category	App
0	FAMILY	1940
1	GAME	1121
2	TOOLS	841
3	BUSINESS	427
4	MEDICAL	414
5	PRODUCTIVITY	407
6	PERSONALIZATION	386
7	LIFESTYLE	373
8	COMMUNICATION	366
9	FINANCE	360
10	SPORTS	351
11	PHOTOGRAPHY	322
12	HEALTH_AND_FITNESS	306
13	SOCIAL	280
14	NEWS_AND_MAGAZINES	264
15	TRAVEL_AND_LOCAL	237
16	BOOKS_AND_REFERENCE	229
17	SHOPPING	224
18	DATING	198
19	VIDEO_PLAYERS	175
20	MAPS_AND_NAVIGATION	137
21	EDUCATION	130
22	FOOD_AND_DRINK	124
23	ENTERTAINMENT	111
24	AUTO_AND_VEHICLES	85
25	LIBRARIES_AND_DEMO	84
26	WEATHER	82
27	HOUSE_AND_HOME	80
28	EVENTS	64
29	ART_AND_DESIGN	64
30	PARENTING	60
31	COMICS	60
32	BEAUTY	53

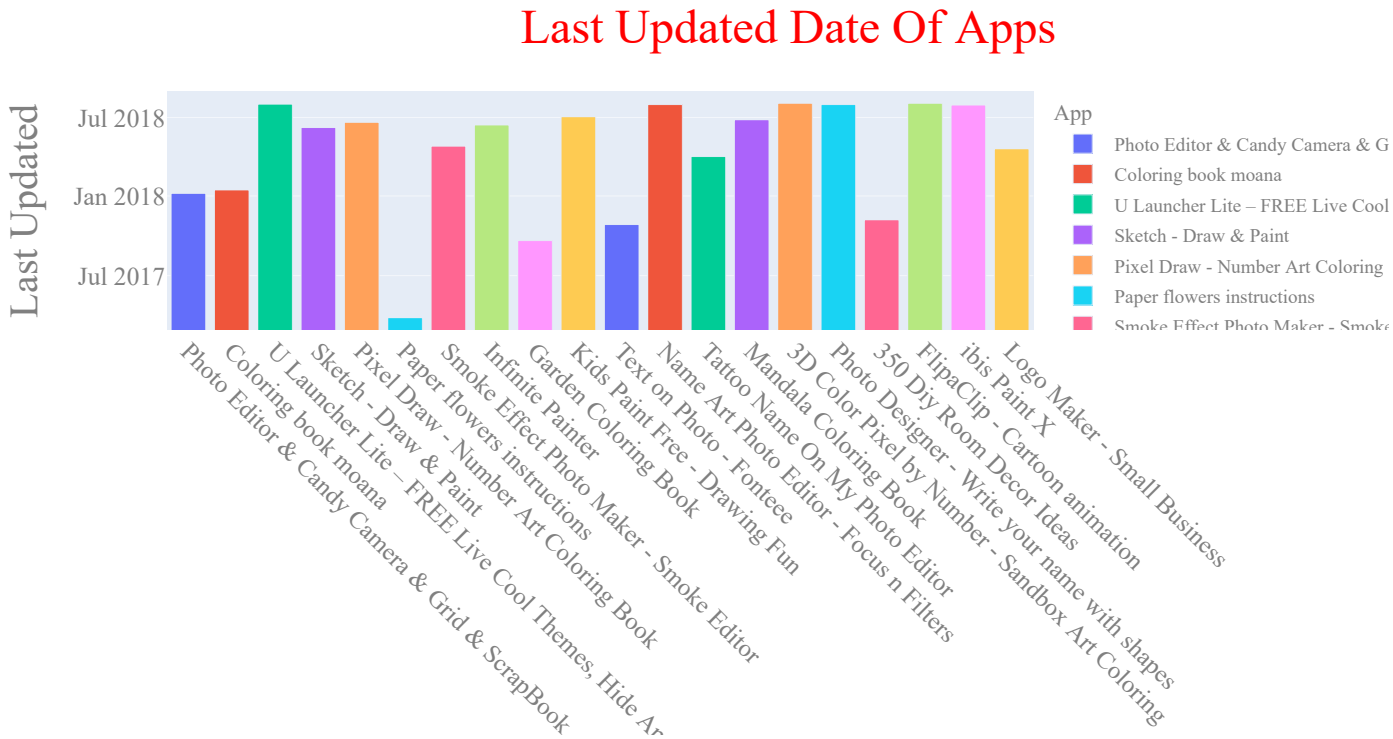
In [68]:

```
arain=px.bar(
    ali,
    x='Category',
    y='App',
    color='Category',
    labels={'App':'Apps_Count'}
)
arain.update_layout(title_text='App numbers for each category', title_x=0.5, title_font_size=30,
                    xaxis_title="Category", yaxis_title="Number of Apps",
                    xaxis_tickangle=-45, xaxis_tickfont_size=14,
                    xaxis_title_font_size=20, yaxis_title_font_size=20,
                    )
```



Now let's have a look of applications according to their respective last updated date.

In [69]:
```
date_fig = px.bar(Clean_Data.head(20), x='App', y='Last Updated',color='App')
date_fig.update_layout(title_text='Last Updated Date Of Apps', title_x=0.5, title_font_size=30,
                       xaxis_title="App", yaxis_title="Last Updated", xaxis_title_font_size=25, yaxis_title_f
                       font_family="Times New Roman", xaxis_tickfont_size=16, yaxis_tickfont_size=16, xaxis_t
    font_color="gray",
    title_font_family="Times New Roman",
    title_font_color="red",
                       )
date_fig.show()
```



Let's see, how many people are using which android version by looking their installed number.

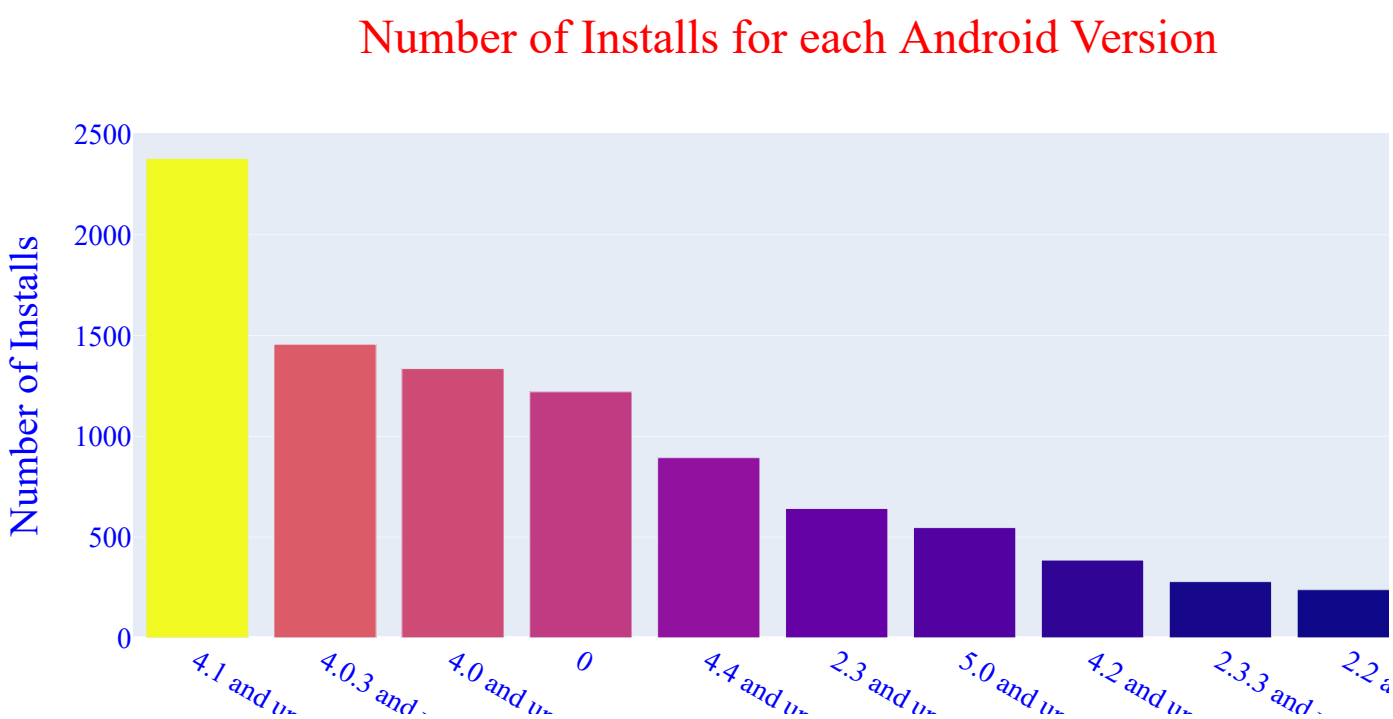
In [70]:
```
version = Clean_Data.groupby('Android Ver')['Installs'].count().sort_values(ascending=False).head(10)
version = pd.DataFrame(version)
version = version.reset_index()
version
```

Out[70]:

	Android Ver	Installs
0	4.1 and up	2379
1	4.0.3 and up	1457
2	4.0 and up	1336
3	0	1221
4	4.4 and up	894
5	2.3 and up	642
6	5.0 and up	547
7	4.2 and up	386
8	2.3.3 and up	279
9	2.2 and up	239

In [71]:

```
l_fig=px.bar(version, x='Android Ver', y='Installs', color='Installs', title='Number of Installs for each And
l_fig.update_layout(
    font_family="Times New Roman",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title_font_size=30,
    title_x=0.5,
    xaxis_title="Android Version",
    xaxis_title_font_size=25,
    legend_title_font_color="green",
    yaxis_title="Number of Installs",
    yaxis_title_font_size=25,
    xaxis_tickfont_size=18,
    yaxis_tickfont_size=18
)
l_fig.show()
```



Conclusion:

After the EDA analysis, there are some results listed below:

- The maximum number of applications are free in google play store, only 7% applications are paid.
- Apps that are free of cost are mostly installed by the users and rated almost 4.5 to 5 stars.
- Top reviews applications in our dataset are Facebook (1st), WhatsApp (2nd), and Instagram (3rd).
- Users are more interested to install those applications that are for everyone not for age limit apps.
- The maximum number of applications come under the family category.

- The android version that are mostly used by the users are latest version like 4.0.3 and up, 4.1 and up, etc.