Project Name - Play Store App Review Analysis

Project_Type - EDA

Contribution - Individual

Name - Ravikant Khandare

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

In [2]:
```
data=pd.read_csv(r"D:\FingerTip's\Panda's Class 1\Play Store Data.csv")
```

In [3]:
```
data.head(5)
```

Out[3]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone

In [4]:
```
data.tail(5)
```

Out[4]:

	App	Category	Rating	Reviews	Size	Installs	Type	Pr
10836	Sya9a Maroc - FR	FAMILY	4.5	38	53M	5,000+	Free	
10837	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3.6M	100+	Free	
10838	Parkinson Exercices FR	MEDICAL	NaN	3	9.5M	1,000+	Free	
10839	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	Varies with device	1,000+	Free	
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19M	10,000,000+	Free	

In [5]:
```
# to check the info of play_store_data
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  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: 1.1+ MB
```

In [6]:
```
len(data[data.duplicated()])
```

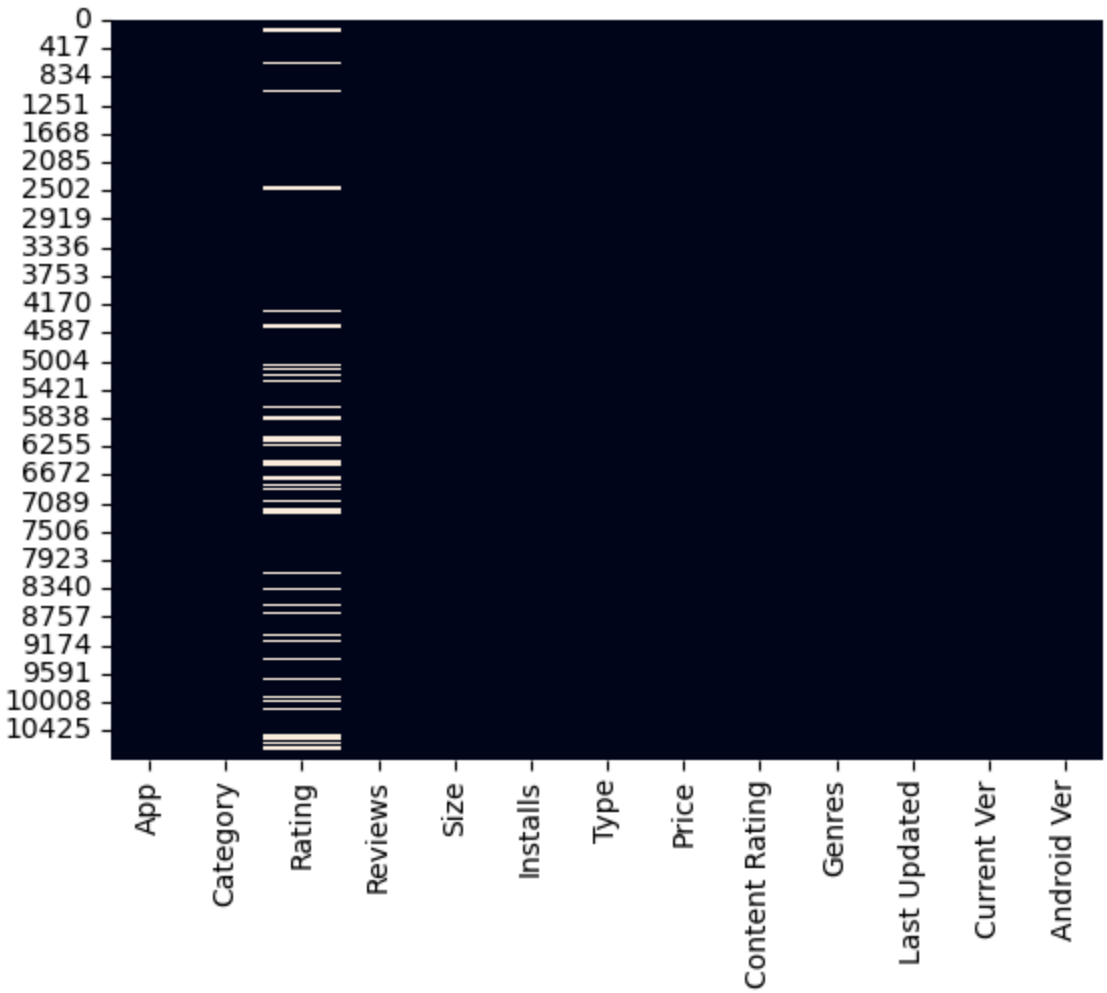
Out[6]: 483

In [7]:
```
# counting the missing value/null value of play_store_data
print(data.isnull().sum())
```

```
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
```

In [8]:
```
# noe visualizing the missing value of play_store_date using seaborn heatmap
sns.heatmap(data.isnull(),cbar=False)
```

Out[8]: <Axes: >



In [9]:
```
data.columns
```

Out[9]:
```
Index(['App', 'Category', 'Rating', 'Reviews', 'Size', 'Installs', 'Type',
       'Price', 'Content Rating', 'Genres', 'Last Updated', 'Current Ver',
       'Android Ver'],
      dtype='object')
```

In [10]:
```
data.describe()
```

Out[10]:

	Rating
count	9367.000000
mean	4.193338
std	0.537431
min	1.000000
25%	4.000000
50%	4.300000
75%	4.500000
max	19.000000

In [11]:
```
# now chwck the unique value for each variable in Play_store_data
for i in data.columns.tolist():
    print("unique value in",i,"is",data[i].nunique())
```

```
unique value in App is 9660
unique value in Category is 34
unique value in Rating is 40
unique value in Reviews is 6002
unique value in Size is 462
unique value in Installs is 22
unique value in Type is 3
unique value in Price is 93
unique value in Content Rating is 6
unique value in Genres is 120
unique value in Last Updated is 1378
unique value in Current Ver is 2832
unique value in Android Ver is 33
```

In [12]:
```
# creating a new empty dataframe to new variable
data_new = pd.DataFrame(index=data.columns)
```

In [13]:
```
# now adding datatype, not_null, and nullcolumns to data_new
data_new["Data_Type"]=data.dtypes
data_new["not_null"]=data.count()
data_new["null"]=data.isnull().sum()
```

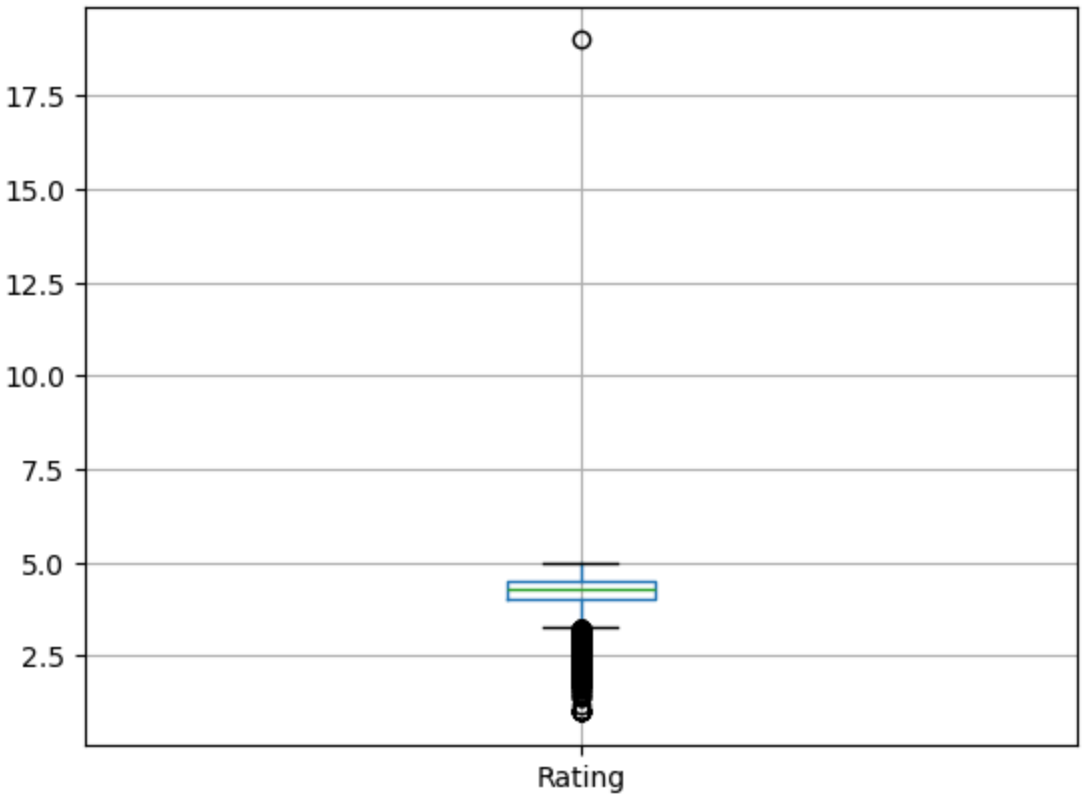
In [14]:
```
data_new
```

Out[14]:

	Data_Type	not_null	null
App	object	10841	0
Category	object	10841	0
Rating	float64	9367	1474
Reviews	object	10841	0
Size	object	10841	0
Installs	object	10841	0
Type	object	10840	1
Price	object	10841	0
Content Rating	object	10840	1
Genres	object	10841	0
Last Updated	object	10841	0
Current Ver	object	10833	8
Android Ver	object	10838	3

In [15]:
```
# checking the rating of data
data.boxplot()
```

Out[15]: <Axes: >



In [16]:
```
data[(data["Rating"]<1) | (data["Rating"]>5)]
```

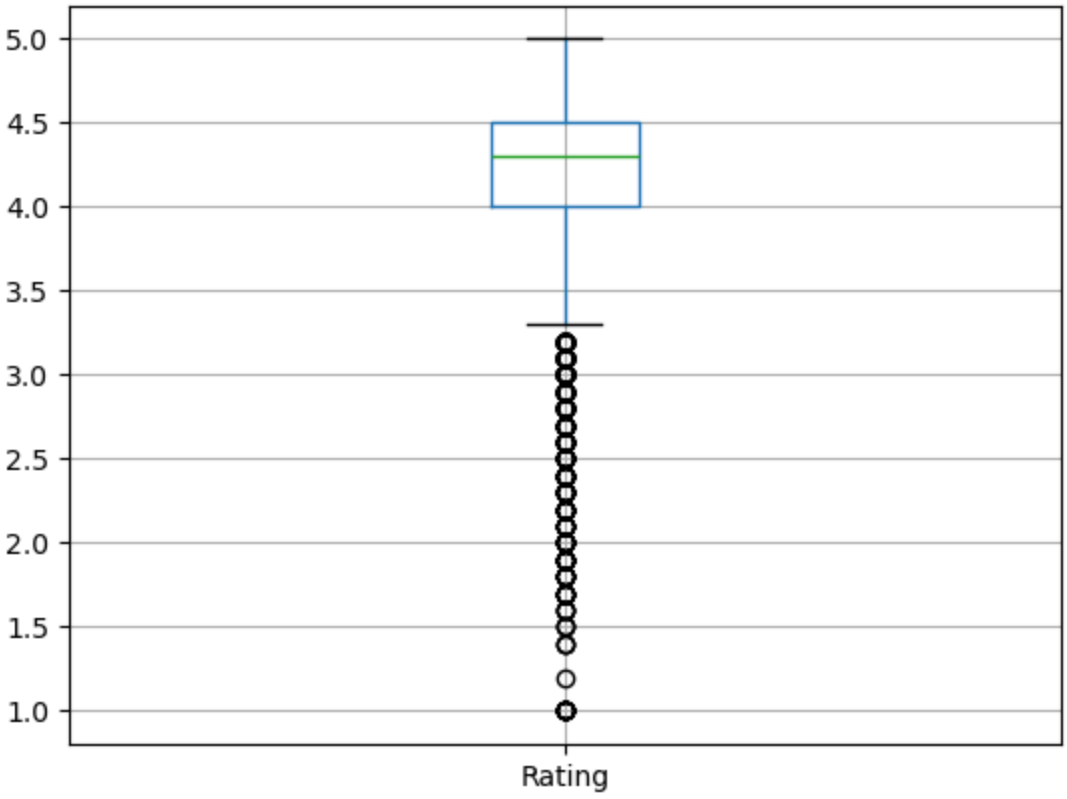
Out[16]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
10472	Life Made WI-Fi Touchscreen Photo Frame	1.9	19.0	3.0M	1,000+	Free	0	Everyone	NaN

In [17]:
```
data.drop(10472,axis=0,inplace=True)
```

In [18]:
```
data.boxplot()
```

Out[18]: <Axes: >



In [19]:
```
# findinng the mean rating column from data
rating_mean=data["Rating"].mean()
print(f"The mean of Rating column is {rating_mean}")

# findinng the median rating column from data
rating_median=data["Rating"].median()
print(f"The median of Rating column is {rating_median}")
```

```
The mean of Rating column is 4.191757420456972
The median of Rating column is 4.3
```

In [20]:
```
# filling all the null value by the median in rating column of data

data["Rating"].fillna(value=rating_median,inplace=True)
```

In [21]:
```
# time to check the null value is filled or not
print(data.isnull().sum())
```

```
App             0
Category        0
Rating          0
Reviews         0
Size            0
Installs        0
Type            1
Price           0
Content Rating  0
Genres          0
Last Updated    0
Current Ver     8
Android Ver     2
dtype: int64
```

In [22]:
```
#checking the type NaN value in type column of data
data[(data["Type"].isnull())]
```

Out[22]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genr
9148	Command & Conquer: Rivals	FAMILY	4.3	0	Varies with device	0	NaN	0	Everyone 10+	Strateg

In [23]:
```
# counting the free and paid version of application in data
data["Type"].value_counts()
```

Out[23]:
```
Type
Free    10039
Paid      800
Name: count, dtype: int64
```

In [24]:
```
# now we replace the NaN value of type
data.loc[9148,"Type"]="Free"
```

In [25]:
```
# Time to check it is fixed or not
data[(data['Type'].isnull())]
```

Out[25]:

App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated

In [26]:
```
# Time to fix the Current Ver's null values in play_store_data
data[(data['Current Ver'].isnull())]
```

Out[26]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price
15	Learn To Draw Kawaii Characters	ART_AND_DESIGN	3.2	55	2.7M	5,000+	Free	0
1553	Market Update Helper	LIBRARIES_AND_DEMO	4.1	20145	11k	1,000,000+	Free	0
6322	Virtual DJ Sound Mixer	TOOLS	4.2	4010	8.7M	500,000+	Free	0
6803	BT Master	FAMILY	4.3	0	222k	100+	Free	0
7333	Dots puzzle	FAMILY	4.0	179	14M	50,000+	Paid	$0.99
7407	Calculate My IQ	FAMILY	4.3	44	7.2M	10,000+	Free	0
7730	UFO-CQ	TOOLS	4.3	1	237k	10+	Paid	$0.99
10342	La Fe de Jesus	BOOKS_AND_REFERENCE	4.3	8	658k	1,000+	Free	0

In [27]:
```
# droping all the NaN values of Current Ver column from play_store_data
data.drop([15,1553,6322,6803,7333,7407,7730,10342],axis=0,inplace=True)
```

In [28]:
```
# Time to check all the Nan values are droped or not
data[(data['Current Ver'].isnull())]
```

Out[28]:

App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated

In [29]:
```
# checking the Android Ver's NaN value which is shown when we used this commmand: p
data[(data['Android Ver'].isnull())]
```

Out[29]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Conte Rati
4453	[substratum] Vacuum: P	PERSONALIZATION	4.4	230	11M	1,000+	Paid	$1.49	Everyo
4490	Pi Dark [substratum]	PERSONALIZATION	4.5	189	2.1M	10,000+	Free	0	Everyo

In [30]:
```
# droping the NaN values of Android Ver column from play_store_data
data.drop([4453,4490],axis=0,inplace=True)
```

In [31]:
```
# Time to check these Nan values are droped or not
data[(data['Android Ver'].isnull())]
```

Out[31]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated

In [32]:
```
# checking the data type in Size column in play_store_data
data['Size']
```

Out[32]:
```
0                     19M
1                     14M
2                    8.7M
3                     25M
4                    2.8M
                ...
10836                 53M
10837                3.6M
10838                9.5M
10839    Varies with device
10840                 19M
Name: Size, Length: 10830, dtype: object
```

In [33]:
```
# The "Size" has different units.

# M means MB
# K means KB
# So We'll fix this using convert them into single unit.
```

In [34]:
```
# checking the data type in Price column in play_store_data
data['Price'].value_counts()
```

Out[34]:
```
Price
0          10033
$0.99        146
$2.99        129
$1.99         73
$4.99         72
           ...
$19.90         1
$1.75          1
$14.00         1
$4.85          1
$1.04          1
Name: count, Length: 92, dtype: int64
```

In [35]:
```
# checking the data type in Installs column in play_store_data
data['Installs'].value_counts()
```

Out[35]:
```
Installs
1,000,000+        1578
10,000,000+       1252
100,000+          1169
10,000+           1052
1,000+             905
5,000,000+         752
100+               718
500,000+           538
50,000+            478
5,000+             476
100,000,000+       409
10+                385
500+               330
50,000,000+        289
50+                205
5+                  82
500,000,000+        72
1+                  67
1,000,000,000+      58
0+                  14
0                    1
Name: count, dtype: int64
```

In [36]:
```
# we're trying to remove +(plus) and ,(comma) from Installs
def remove_from_install(a):
    if type(a) == str:
        a = a.replace(',', '').replace('+', '')
    return a

data['Installs'] = data['Installs'].apply(remove_from_install)
```

In [37]:
```
# we're trying to remove dollar sign from Price
def remove_from_price(b):
    if type(b) == str and '$' in b:
        b = b.replace('$', '')
    return b
```

```
data['Price'] = data['Price'].apply(remove_from_price)
```

In [38]:
```
# now we're converting 'Reviews' to numeric in play_store_data
data['Reviews'] = data['Reviews'].astype(float)
```

In [39]:
```
# cleaning and converting 'Size' to numeric
def clean_size(x):
    if 'Varies with device' in str(x):
        return np.nan
    elif 'k' in str(x):
        return float(str(x).replace('k', '')) / 1024
    else:
        return float(str(x).replace('M', ''))
data['Size'] = data['Size'].apply(clean_size)
```

In [40]:
```
# cleaning and converting 'Installs' to numeric
data['Installs'] = data['Installs'].replace('[^\d]', '', regex=True).astype(float)
```

In [41]:
```
# cleaning and converting 'Price' to numeric
data['Price'] = data['Price'].replace('[^\d\.]', '', regex=True).astype(float)
```

In [42]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10830 entries, 0 to 10840
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   App             10830 non-null  object 
 1   Category        10830 non-null  object 
 2   Rating          10830 non-null  float64
 3   Reviews         10830 non-null  float64
 4   Size            9135 non-null   float64
 5   Installs        10830 non-null  float64
 6   Type            10830 non-null  object 
 7   Price           10830 non-null  float64
 8   Content Rating  10830 non-null  object 
 9   Genres          10830 non-null  object 
 10  Last Updated    10830 non-null  object 
 11  Current Ver     10830 non-null  object 
 12  Android Ver     10830 non-null  object 
dtypes: float64(5), object(8)
memory usage: 1.2+ MB
```

In [43]:
```
data.head(10)
```

Out[43]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159.0	19.0	10000.0	Free	0.0	Everyone
1	Coloring book moana	ART_AND_DESIGN	3.9	967.0	14.0	500000.0	Free	0.0	Everyone
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510.0	8.7	5000000.0	Free	0.0	Everyone
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644.0	25.0	50000000.0	Free	0.0	Teen
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967.0	2.8	100000.0	Free	0.0	Everyone
5	Paper flowers instructions	ART_AND_DESIGN	4.4	167.0	5.6	50000.0	Free	0.0	Everyone
6	Smoke Effect Photo Maker - Smoke Editor	ART_AND_DESIGN	3.8	178.0	19.0	50000.0	Free	0.0	Everyone
7	Infinite Painter	ART_AND_DESIGN	4.1	36815.0	29.0	1000000.0	Free	0.0	Everyone
8	Garden Coloring Book	ART_AND_DESIGN	4.4	13791.0	33.0	1000000.0	Free	0.0	Everyone
9	Kids Paint Free - Drawing Fun	ART_AND_DESIGN	4.7	121.0	3.1	10000.0	Free	0.0	Everyone

In [44]:
```
# checking the values of App from play_store_data
data['App'].value_counts()
```

Out[44]:
```
App
ROBLOX                                                9
CBS Sports App - Scores, News, Stats & Watch Live     8
Candy Crush Saga                                      7
8 Ball Pool                                           7
ESPN                                                  7
                                                     ..
Meet U - Get Friends for Snapchat, Kik & Instagram    1
U-Report                                              1
U of I Community Credit Union                         1
Waiting For U Launcher Theme                          1
iHoroscope - 2018 Daily Horoscope & Astrology         1
Name: count, Length: 9649, dtype: int64
```

In [45]:
```
# calculating the duplicate value in "App" column in play_store_data
data['App'].duplicated().sum()
```

Out[45]:
```
1181
```

In [46]:
```
# dropping duplicates value in "App" in play_store_data
data.drop_duplicates(subset='App',inplace=True)
```

In [47]:
```
# time to check duplicates are removed or not
data['App'].duplicated().sum()
```

Out[47]:
```
0
```

In [48]:
```
# time to check duplicates are removed or not
data['App'].duplicated().sum()
```

Out[48]:
```
0
```

In [49]:
```
# Summary
# All duplicates values removed from dataset.
# All null values are removed or replaced.
# Converted the datatypes of the particular column and also removed all the unwante
# Data Cleaning on User Review dataset
```

Data Cleaning on User Review dataset

In [50]:
```
# now we're importing the User Reviews.csv from Google Drive as user_reviews_data
user_reviews_data = pd.read_csv(r"D:\FingerTip's\Panda's Class 1\User Reviews.csv")
```

In [51]:
```
# using head() to show top 10 rows of user_reviews_data
user_reviews_data.head(10)
```

Out[51]:

	App	Translated_Review	Sentiment	Sentiment_Polarity	Sentiment_Subjectivity
0	10 Best Foods for You	I like eat delicious food. That's I'm cooking ...	Positive	1.00	0.533333
1	10 Best Foods for You	This help eating healthy exercise regular basis	Positive	0.25	0.288462
2	10 Best Foods for You	NaN	NaN	NaN	NaN
3	10 Best Foods for You	Works great especially going grocery store	Positive	0.40	0.875000
4	10 Best Foods for You	Best idea us	Positive	1.00	0.300000
5	10 Best Foods for You	Best way	Positive	1.00	0.300000
6	10 Best Foods for You	Amazing	Positive	0.60	0.900000
7	10 Best Foods for You	NaN	NaN	NaN	NaN
8	10 Best Foods for You	Looking forward app,	Neutral	0.00	0.000000
9	10 Best Foods for You	It helpful site ! It help foods get !	Neutral	0.00	0.000000

In [52]:
```
# using tail() to show last 10 rows of user_reviews_data
user_reviews_data.tail(10)
```

Out[52]:

	App	Translated_Review	Sentiment	Sentiment_Polarity	Sentiment_Subjectivity
64285	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64286	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64287	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64288	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64289	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64290	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64291	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64292	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64293	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64294	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN

In [53]:
```
# checking the shape (number of rows and columns) of user_reviews_data
user_reviews_data.shape
```

Out[53]: (64295, 5)

In [54]:
```
# checking the columns of user_reviews_data
user_reviews_data.columns
```

Out[54]:
```
Index(['App', 'Translated_Review', 'Sentiment', 'Sentiment_Polarity',
       'Sentiment_Subjectivity'],
      dtype='object')
```

In [55]:
```
# The dataset has 5 columns are identified as below:
# App: Title of the application.
# Translated_Review: It contains the English translation of the review.
# Sentiment: It gives the emotion like 'Positive', 'Negative', or 'Neutral'.
# Sentiment_Polarity: It gives the polarity of the review. Its range is [-1,1],
# where 1 means 'Positive statement' and -1 means a 'Negative statement'.
# Sentiment_Subjectivity: This value gives how close a reviewers opinion is to the
# Its range is[0,1].
```

In [56]:
```
# using the info of user_reviews_data
user_reviews_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 64295 entries, 0 to 64294
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   App                     64295 non-null  object 
 1   Translated_Review       37427 non-null  object 
 2   Sentiment               37432 non-null  object 
 3   Sentiment_Polarity      37432 non-null  float64
 4   Sentiment_Subjectivity  37432 non-null  float64
dtypes: float64(2), object(3)
memory usage: 2.5+ MB
```

In [57]:
```
# creating a new empty dataframe to new variable
user_reviews_data_new = pd.DataFrame(index=user_reviews_data.columns)
```

In [58]:
```
# now adding datatype, not_null, and null columns to user_reviews_data_new
user_reviews_data_new["DataType"] = user_reviews_data.dtypes
user_reviews_data_new["not_null"] = user_reviews_data.count()
user_reviews_data_new["null"] = user_reviews_data.isnull().sum()

# printing the dataframe
user_reviews_data_new
```

Out[58]:

	DataType	not_null	null
App	object	64295	0
Translated_Review	object	37427	26868
Sentiment	object	37432	26863
Sentiment_Polarity	float64	37432	26863
Sentiment_Subjectivity	float64	37432	26863

In [59]:
```
# finding the NaN value of Sentiment_Polarity column in user_reviews_data
user_reviews_data[(user_reviews_data['Sentiment_Polarity'].isnull())]
```

Out[59]:

	App	Translated_Review	Sentiment	Sentiment_Polarity	Sentiment_Subjectivity
2	10 Best Foods for You	NaN	NaN	NaN	NaN
7	10 Best Foods for You	NaN	NaN	NaN	NaN
15	10 Best Foods for You	NaN	NaN	NaN	NaN
102	10 Best Foods for You	NaN	NaN	NaN	NaN
107	10 Best Foods for You	NaN	NaN	NaN	NaN
...	...	...	...	...	...
64290	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64291	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64292	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64293	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN
64294	Houzz Interior Design Ideas	NaN	NaN	NaN	NaN

26863 rows × 5 columns

In [60]:
```
# counting the values of Sentiment_Polarity from user_reviews_data
user_reviews_data['Sentiment_Polarity'].value_counts()
```

Out[60]:
```
Sentiment_Polarity
 0.000000    5163
 0.500000    1524
 0.700000     991
 1.000000     959
 0.800000     639
             ...
-0.072024       1
 0.452000       1
 0.076190       1
-0.067256       1
 0.173333       1
Name: count, Length: 5410, dtype: int64
```

In [61]:
```
# droping all null values from user_reviews_data
user_reviews_data.dropna(inplace = True)
```

In [62]:
```
# time to check is there any null values now
user_reviews_data[(user_reviews_data['Sentiment_Polarity'].isnull())]
```

Out[62]:

App	Translated_Review	Sentiment	Sentiment_Polarity	Sentiment_Subjectivity

In [63]:
```
# now time to check the data as little cleaned or not
user_reviews_data.head(20)
```

Out[63]:

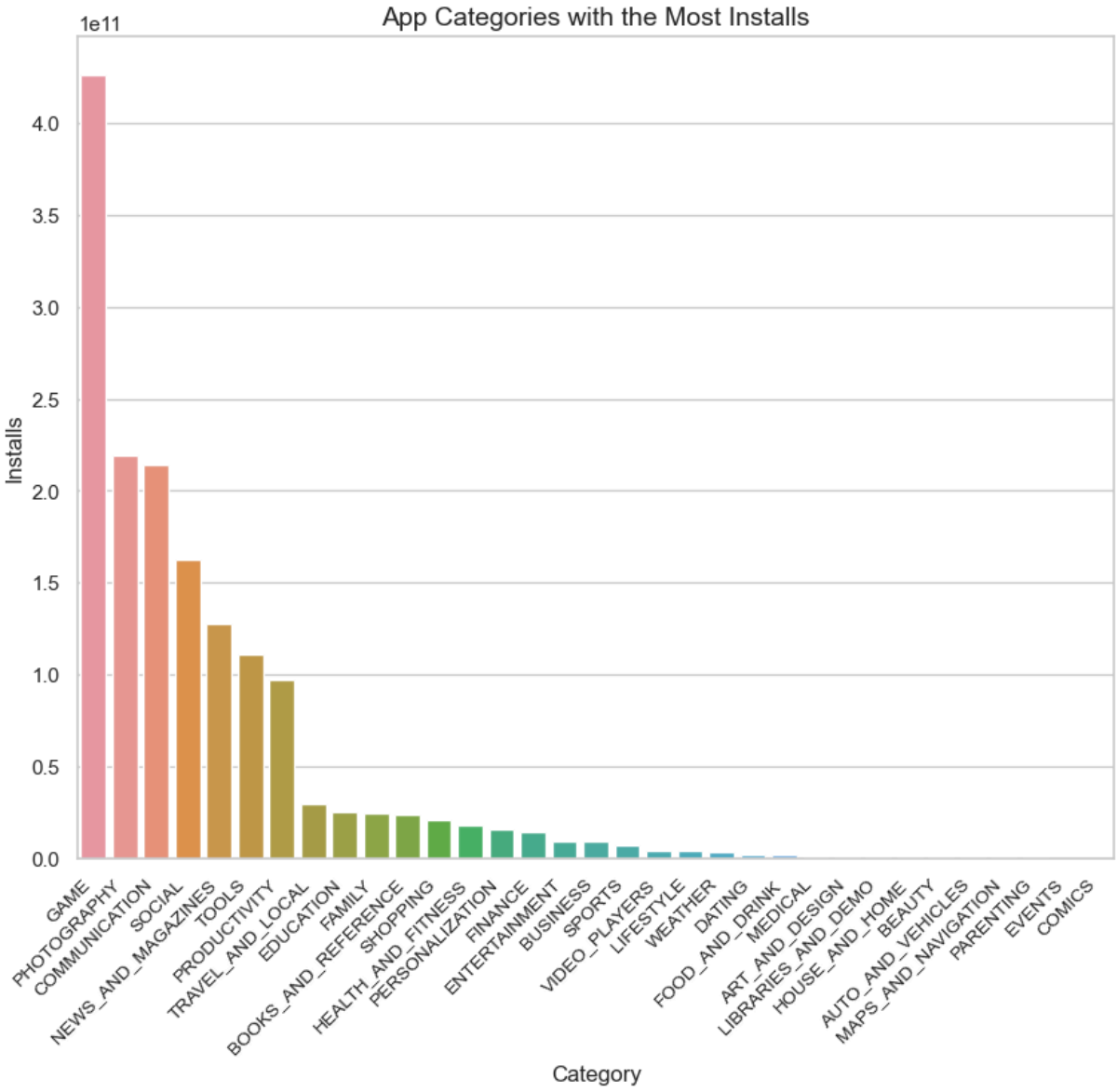
	App	Translated_Review	Sentiment	Sentiment_Polarity	Sentiment_Subjectivity
0	10 Best Foods for You	I like eat delicious food. That's I'm cooking ...	Positive	1.000000	0.533333
1	10 Best Foods for You	This help eating healthy exercise regular basis	Positive	0.250000	0.288462
3	10 Best Foods for You	Works great especially going grocery store	Positive	0.400000	0.875000
4	10 Best Foods for You	Best idea us	Positive	1.000000	0.300000
5	10 Best Foods for You	Best way	Positive	1.000000	0.300000
6	10 Best Foods for You	Amazing	Positive	0.600000	0.900000
8	10 Best Foods for You	Looking forward app,	Neutral	0.000000	0.000000
9	10 Best Foods for You	It helpful site ! It help foods get !	Neutral	0.000000	0.000000
10	10 Best Foods for You	good you.	Positive	0.700000	0.600000
11	10 Best Foods for You	Useful information The amount spelling errors ...	Positive	0.200000	0.100000
12	10 Best Foods for You	Thank you! Great app!! Add arthritis, eyes, im...	Positive	0.750000	0.875000

	App	Translated_Review	Sentiment	Sentiment_Polarity	Sentiment_Subjectivity
13	10 Best Foods for You	Greatest ever Completely awesome maintain heal...	Positive	0.992188	0.866667
14	10 Best Foods for You	Good health..... Good health first priority.....	Positive	0.550000	0.511111
16	10 Best Foods for You	Health It's important world either life . thin...	Positive	0.450000	1.000000
17	10 Best Foods for You	Mrs sunita bhati I thankful developers,to make...	Positive	0.600000	0.666667
18	10 Best Foods for You	Very Useful in diabetes age 30. I need control...	Positive	0.295000	0.100000
19	10 Best Foods for You	One greatest apps.	Positive	1.000000	1.000000
20	10 Best Foods for You	good nice	Positive	0.650000	0.800000
21	10 Best Foods for You	Healthy Really helped	Positive	0.350000	0.350000
22	10 Best Foods for You	God health	Neutral	0.000000	0.000000

Data Vizualization, Storytelling & Experimenting with charts : let's Understand the relationships between variables

Which app categories have the most installs ?

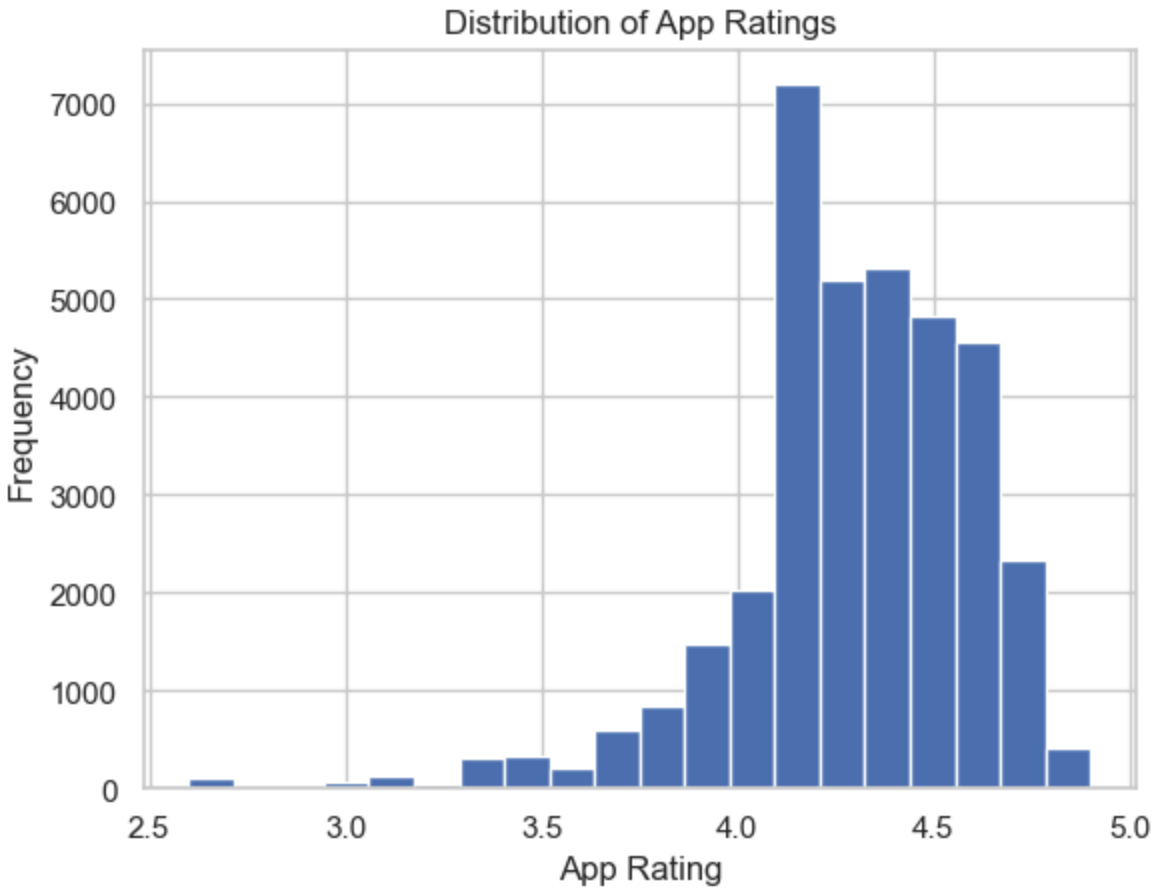
In [64]:
```
# merge the datasets
merged_data = pd.merge(data, user_reviews_data, on='App')

# grouping the data by category and sum the installs
category_installs = merged_data.groupby('Category')['Installs'].sum().reset_index()

# sorting the data by installs in descending order
category_installs = category_installs.sort_values('Installs', ascending=False)

# creating a barplot using seaborn lib
sns.set(style="whitegrid")
fig, ax = plt.subplots(figsize=(10,8))
ax = sns.barplot(x='Category', y='Installs', data=category_installs)
ax.set_xticklabels(ax.get_xticklabels(), rotation=45, ha="right", fontsize=10)
ax.set_xlabel('Category', fontsize=12)
ax.set_ylabel('Installs', fontsize=12)
ax.set_title('App Categories with the Most Installs', fontsize=14)

plt.show()
```



In [65]:
```
# 1. Why did you pick the specific chart?
# A bar chart is a good choice for visualizing the number of installs for different
# app categories because it allows us for easy comparison between categories.

# 2. What is/are the insight(s) found from the chart?
# In this visualization, we can see that the most downloaded applications belong to
# indicating a high demand for GAMES. After GAMES, there is a strong competition be
# and COMMUNICATION applications.

#3. Will the gained insights help creating a positive business impact?
#Are there any insights that lead to negative growth? Justify with specific reason.

# Companies operating in the GAMES category can leverage this high demand to develo
# GAMES more effectively, potentially leading to higher revenue and market share. S
# companies operating in the PHOTOGRAPHY and COMMUNICATION categories can identify
# competition and innovate their products to stand out in the market and capture mo
#This can help them improve their market position and increase their revenue.
```

What is the distribution of app ratings?

In [66]:
```
# merge the data on the app
merged_data = pd.merge(data, user_reviews_data, on='App')

# creating a histogram of the app ratings
plt.hist(merged_data["Rating"], bins=20)

# adding labels and title
plt.xlabel("App Rating")
plt.ylabel("Frequency")
plt.title("Distribution of App Ratings")

# time to print
plt.show()
```

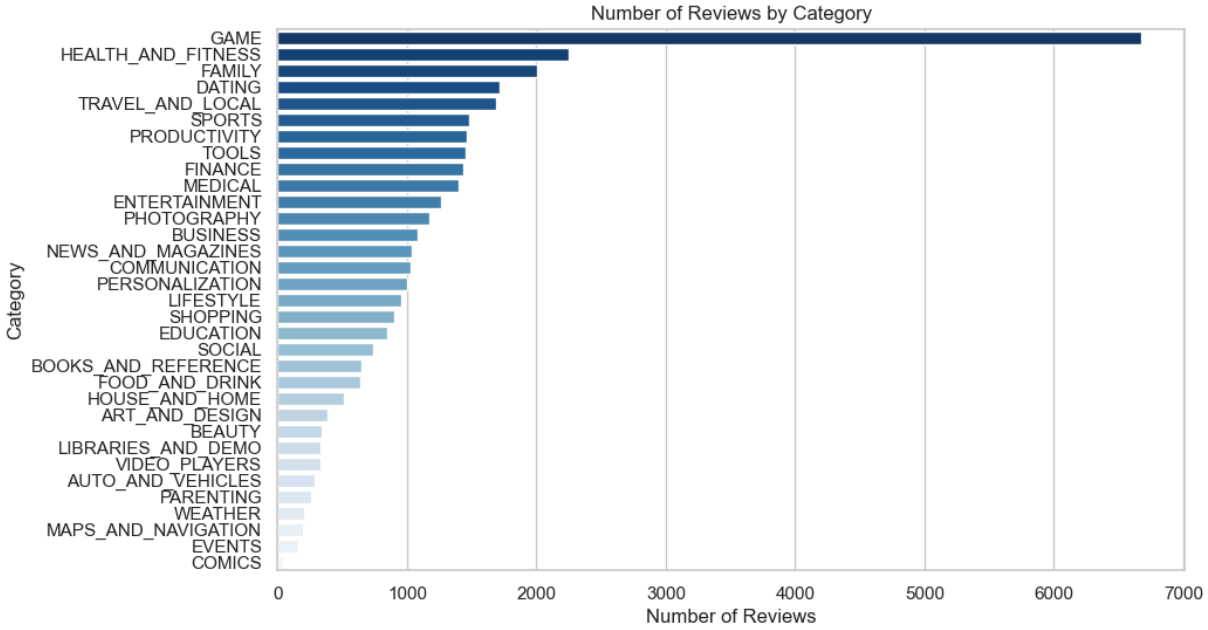


In [67]:
```
#1. Why did you pick the specific chart?
#I chose a bar chart because it is an effective way to display counts or frequencie
#such as app ratings.

#2. What is/are the insight(s) found from the chart?
#Most apps have a rating between 4.0 and 4.5, which means that users are mostly hap
#There are only a few apps with ratings below 3.5.

#3. Will the gained insights help creating a positive business impact?
#Are there any insights that lead to negative growth? Justify with specific reason.
```

```
#Here most apps have high ratings, indicating that users are generally satisfied wi
#quality of apps available on the Play Store. This could result in increased user t
#and retention, which can lead to higher app downloads, increased revenue, and a be
#for the app store.
```

Which categories have the most reviews?

In [68]:
```
# merge the data on the app
merged_data = pd.merge(data, user_reviews_data, on='App')

# calculating the number of reviews by category
reviews_by_category = merged_data.groupby("Category")["Translated_Review"].count().

# creating a horizontal bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x=reviews_by_category.values, y=reviews_by_category.index, palette="Blu
plt.xlabel("Number of Reviews")
plt.ylabel("Category")
plt.title("Number of Reviews by Category")

# time to print
plt.show()
```

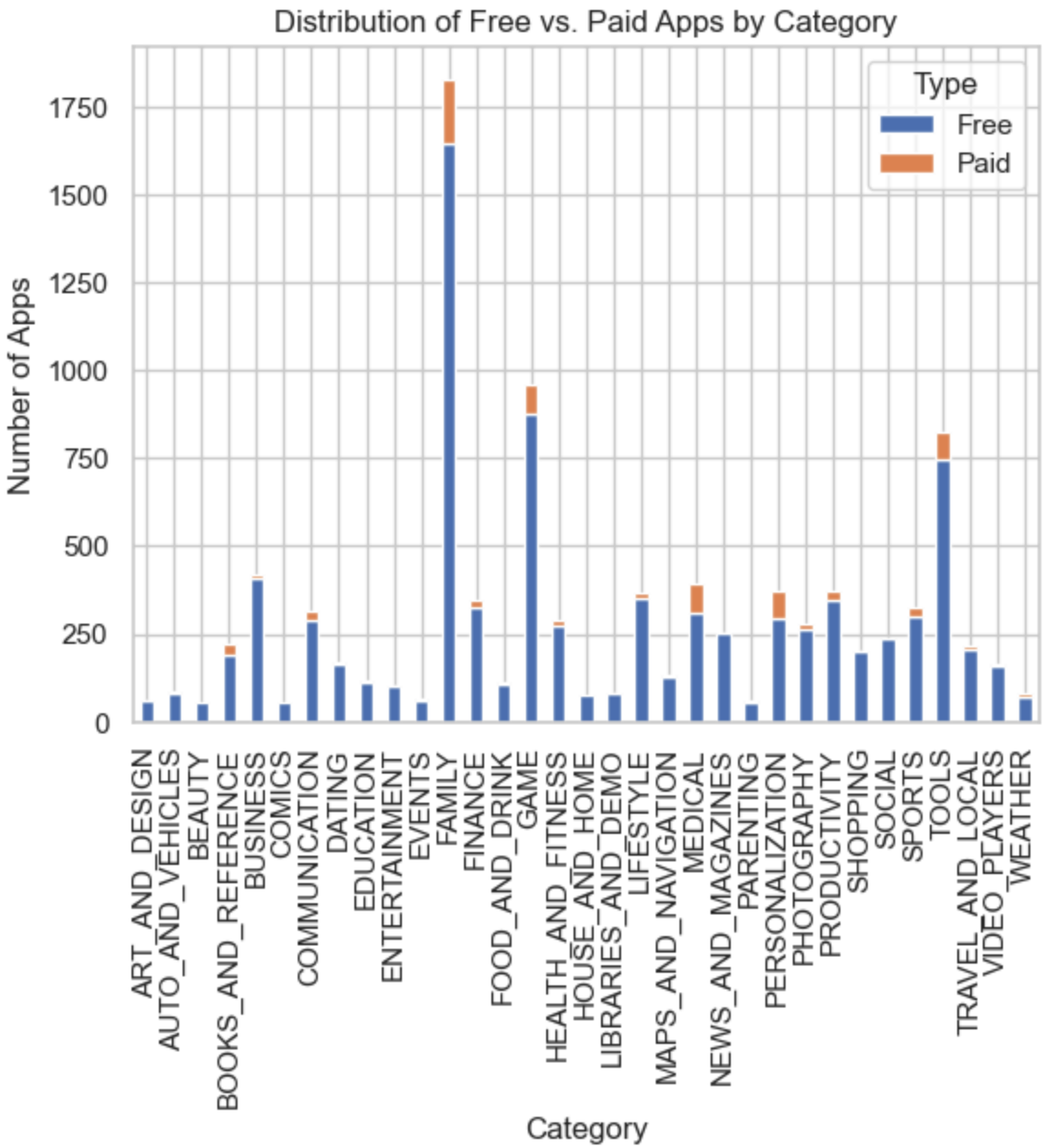


In [69]:
```
# 1. Why did you pick the specific chart?
# I chose a horizontal bar chart to visualize the distribution of reviews across
#different app categories because it allows for easy comparison of the number of re
#for each category. The horizontal layout also makes it easier to read the category

# 2. What is/are the insight(s) found from the chart?
# Users typically leave reviews for an application after using it, and we can see t
#the GAME category has the highest number of reviews. This suggests that the GAME c
#has a large and engaged user base. HEALTH_AND_FITNESS, as well as FAMILY, are the
##highest categories in terms of reviews received. The DATING and TRAVEL_AND_LOCAL
```

```
#have a similar number of reviews from users.

# 3. Will the gained insights help creating a positive business impact?
# Are there any insights that lead to negative growth? Justify with specific reason

# It provides insights into which categories have a large and engaged user base, wh
#could be useful for app developers and marketers to target their efforts towards t
#categories. It also highlights the importance of providing a positive user experie
#order to encourage users to leave reviews, which can ultimately lead to increased
#and downloads for an app.
```

Which categories have the most free vs. paid apps?

In [70]:
```
# grouping data by category and type
category_counts = data.groupby(['Category', 'Type']).count()['App'].unstack()

# ploting stacked bar chart
category_counts.plot(kind='bar', stacked=True)

# title and axis labels
plt.title("Distribution of Free vs. Paid Apps by Category")
plt.xlabel("Category")
plt.ylabel("Number of Apps")

# time to print
plt.show()
```

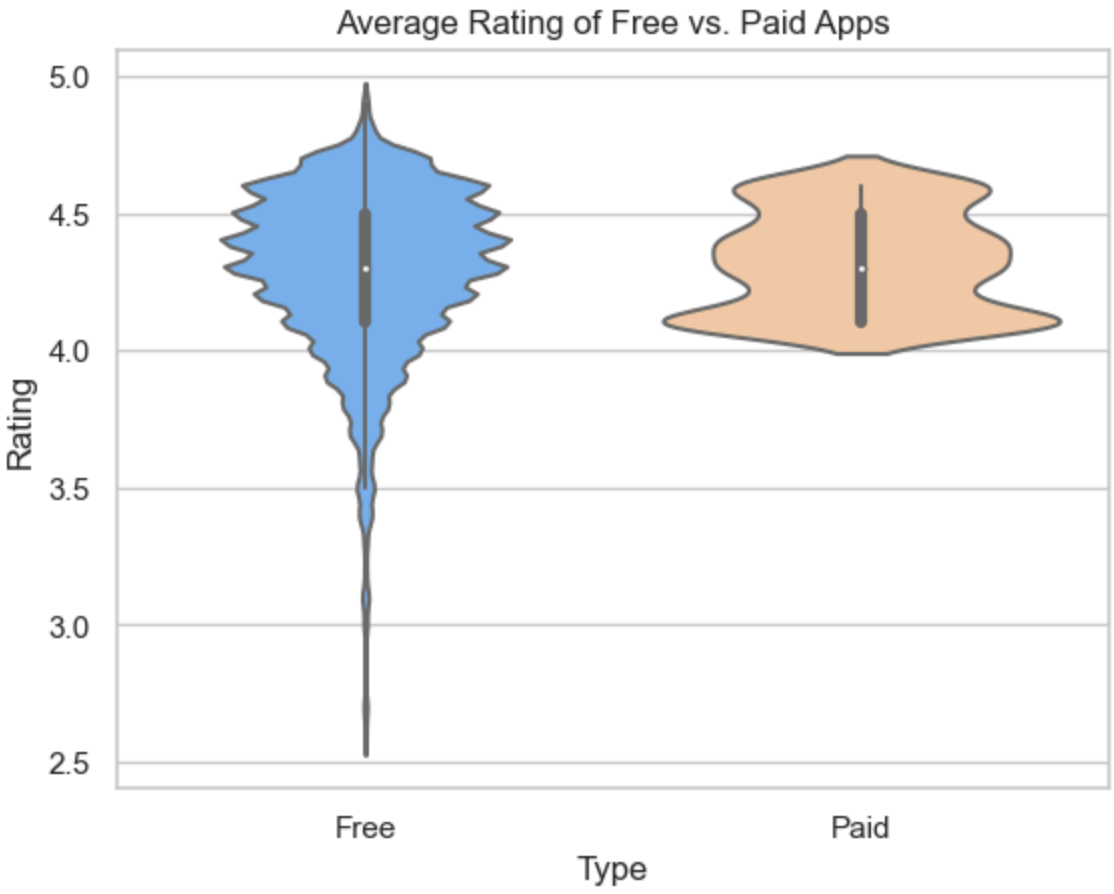


In [71]:
```
# 1. Why did you pick the specific chart?
# I picked the stacked bar chart because it allows for a clear comparison between t
#of free and paid apps in each category, as well as the overall distribution of fre
#apps across all categories.


# 2. What is/are the insight(s) found from the chart?
# Developers and businesses creating apps in popular categories like "FAMILY", "GAM
#and "TOOLS" should explore other revenue options because these categories have a h
#number of free apps, making it challenging to generate income solely through app p

# 3. Will the gained insights help creating a positive business impact?
# Are there any insights that lead to negative growth? Justify with specific reason

# This information can help developers and businesses make better decisions about w
#categories to focus on. If they want to create a paid app, they might want to focu
```

```
#the "Family" or "Game" categories. But if they want to create a free app with less
#they could consider the "Beauty," "Art and Design," and "Comics" categories.
```

What is the average rating of free vs. paid apps?

In [72]:
```
# merge the data on the app
merged_data = pd.merge(data, user_reviews_data, on='App')

free_vs_paid_app = merged_data[merged_data['Type'].isin(['Free', 'Paid'])]

# now using the violin plot
sns.violinplot(x='Type', y='Rating', data=free_vs_paid_app, palette=['#66b3ff', '#f

# title of violin plot
plt.title('Average Rating of Free vs. Paid Apps')

# time to print
plt.show()
```

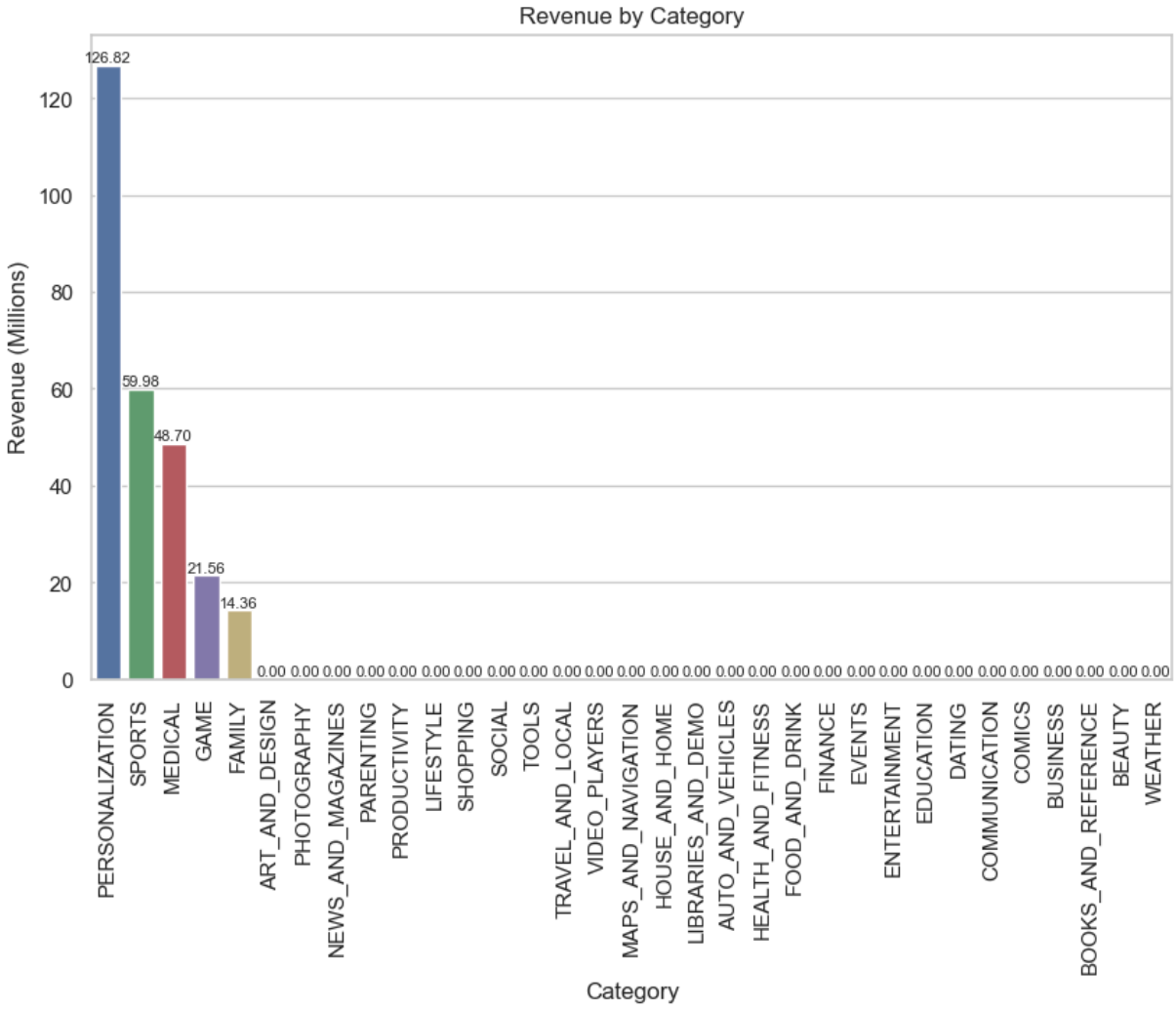


In [73]:
```
# 1. Why did you pick the specific chart?
# I chose to use a violin plot because it can show the distribution of data for bot
#free and paid apps, as well as the average rating of each.

# 2. What is/are the insight(s) found from the chart?
```

```
#Both "FREE" and "PAID" apps have a similar average rating of about 4.3. However, "
#apps have a wider range of ratings than"PAID" apps, meaning there is more variatio
#their ratings. On the other hand, "PAID" apps have a more consistent rating, with
#extreme ratings. This information can be helpful for users when deciding whether t

# 3. Will the gained insights help creating a positive business impact?
# Are there any insights that lead to negative growth? Justify with specific reason

# This information can help a company decide whether to offer their app for "FREE"
 # charge for it. If they want consistent ratings, they may choose to charge for th
# If they want a wider audience, they may choose to offer the app for "FREE", even
# may be more variability in the reviews.#
```

Which categories generate the most revenue?

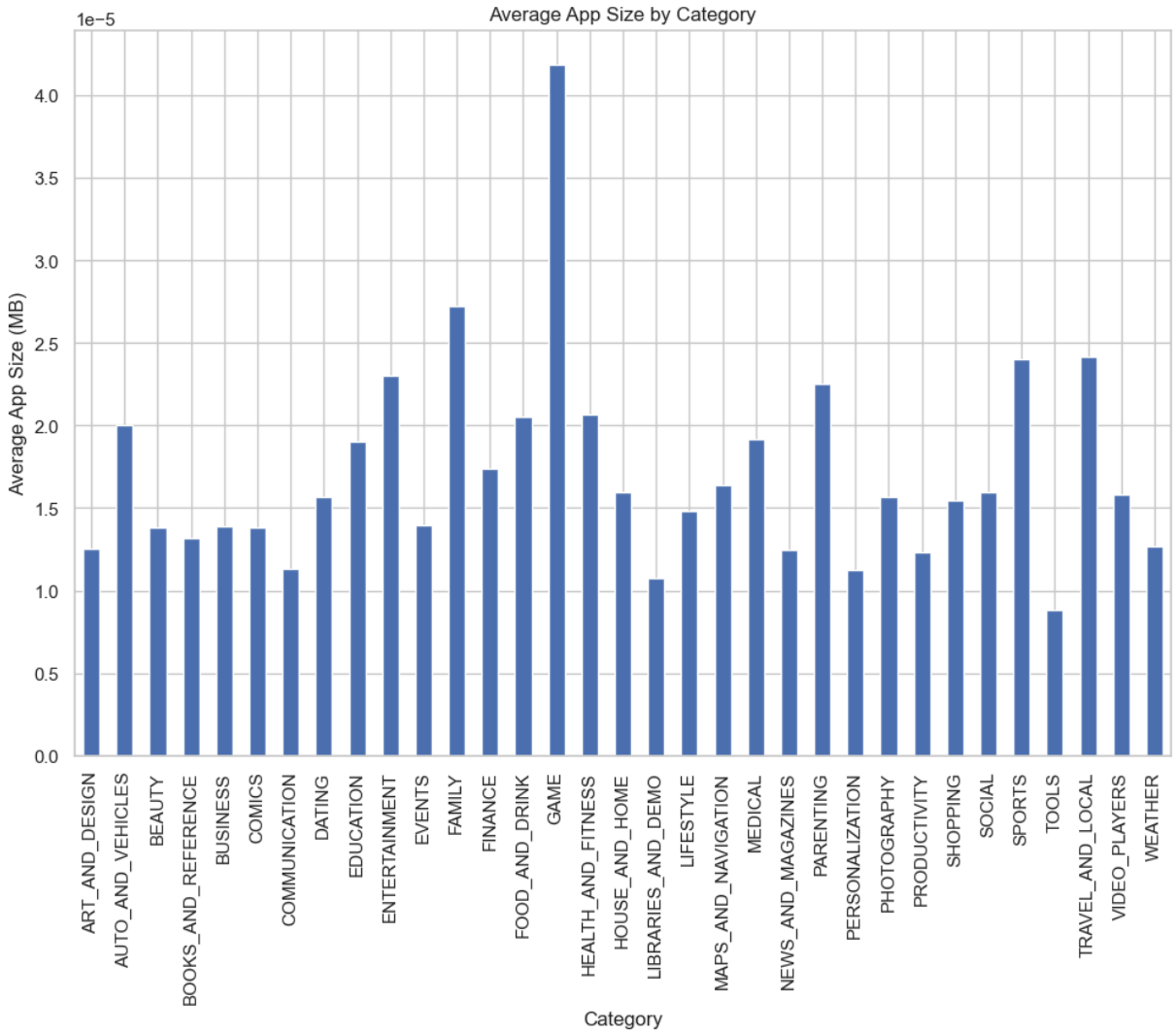
In [74]:
```
# merge the data on the app
merged_data = pd.merge(data, user_reviews_data, on='App')

# calculating the revenue by category
merged_data['Revenue'] = merged_data['Installs'] * merged_data['Price']
revenue_by_category = merged_data.groupby('Category')['Revenue'].sum()

# convert revenue to million
revenue_by_category /= 1000000

# sorting by revenue in descending order
revenue_by_category = revenue_by_category.sort_values(ascending=False)

# adding custom color palette with 6 colors
custom_palette = sns.color_palette(['#4c72b0', '#55a868', '#c44e52', '#8172b2', '#c

# now using bar chart of revenue by category
plt.figure(figsize=(10, 6))
sns.barplot(x=revenue_by_category.index, y=revenue_by_category.values, palette=cust
plt.xticks(rotation=90)
plt.xlabel("Category")
plt.ylabel("Revenue (Millions)")
plt.title("Revenue by Category")

# adding actual revenue values to the bar chart
for i, val in enumerate(revenue_by_category.values):
    plt.annotate("{:.2f}".format(val), (i, val), ha='center', va='bottom', fontsize

# now time to print
plt.show()
```



In [75]:
```
# 1. Why did you pick the specific chart?
# I picked the bar chart because it's an effective way to visualize the revenue
# generated by each category in a clear way.

#  2. What is/are the insight(s) found from the chart?
# It shows that the "Personalization" category generates the highest revenue,
#followed by "Sports" and "Medical". This information can be valuable for companies
#that create apps in these categories as they can prioritize their investments and

# On the other hand, some categories, such as "Weather" and "Beauty", generate lowe
#revenue. This indicates that businesses working in these categories may need to ex
#alternative revenue streams or rethink their business strategy.

# 3. Will the gained insights help creating a positive business impact?
# Are there any insights that lead to negative growth? Justify with specific reason

# Companies can prioritize their investments and efforts in categories that generat
#the highest revenue, such as "Personalization", "Sports", and "Medical". Whereas t
#companies which are operating in categories with lower revenue, such as "Weather"
#may face challenges in generating significant profits solely through the Play Stor
#need to explore other revenue options or
#adjust their business strategy accordingly.
```

What is the average app size in different categories?

In [76]:
```
# grouping play_store_data by category and finding the mean of size
mean_size_by_category = data.groupby("Category")["Size"].mean() / 1000000

# creating bar chart
plt.figure(figsize=(12,8))
mean_size_by_category.plot(kind='bar')
plt.xticks(rotation=90)
plt.title("Average App Size by Category")
plt.xlabel("Category")
plt.ylabel("Average App Size (MB)")

# time to print
plt.show()
```

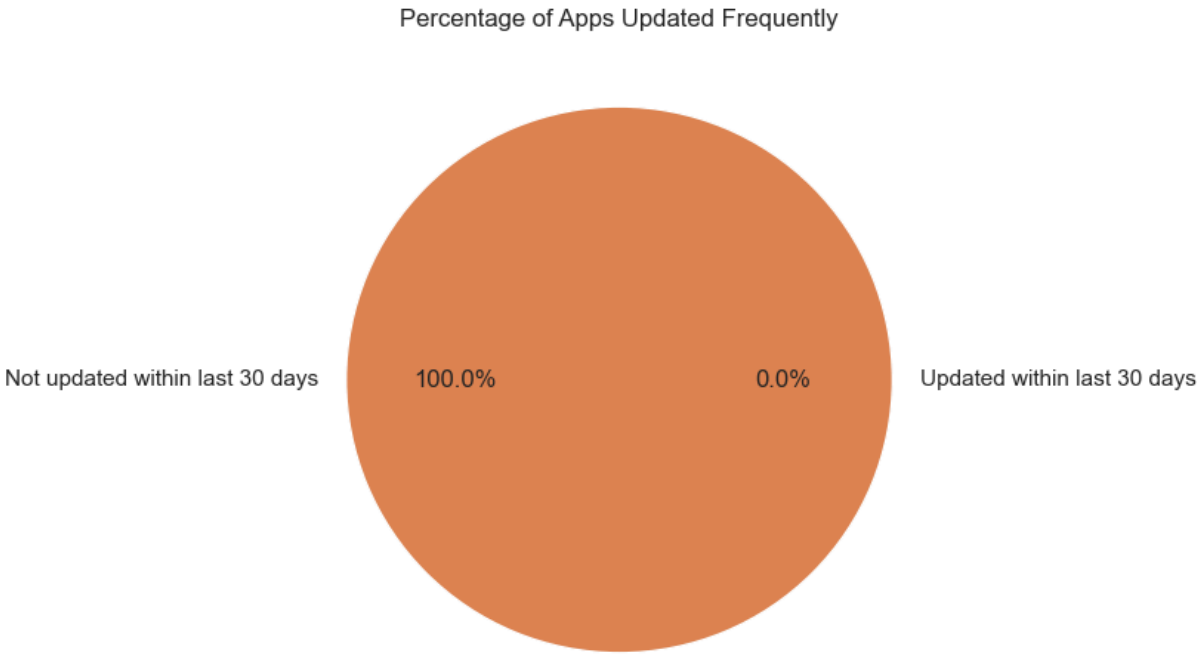


In [77]:
```
# 1. Why did you pick the specific chart?
# I picked a bar chart because it is a simple and effective way to compare the aver
#app size in different categories.
```

```
# 2. What is/are the insight(s) found from the chart?
# Apps in different categories have different average sizes. The biggest apps are i
#the "Game" and "Family" categories, while the smallest apps are in the "Tools" and

# 3. Will the gained insights help creating a positive business impact?
# Are there any insights that lead to negative growth? Justify with specific reason

# It helps app developers to better optimize the size of their app according to
#their target category, as well as give them insights on the size of their
#competitors' apps in the same category. Also, make sure their apps are not too big
#take up too much space on users' devices.
```

What is the percentage of apps that are updated frequently?

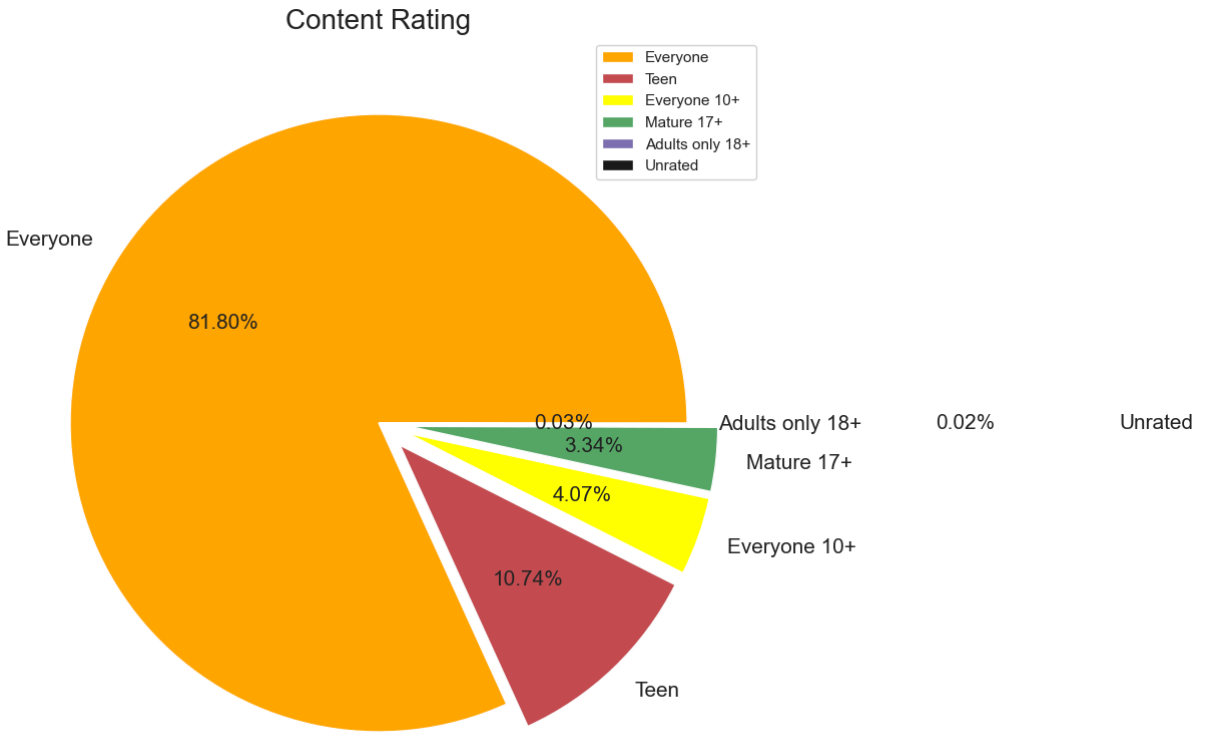
In [78]:
```
# converting "Last Updated" column to datetime format
data["Last Updated"] = pd.to_datetime(data["Last Updated"])

# calculating the percentage of apps that were updated within the last 30 days
last_30_days = data[data["Last Updated"] >= (pd.Timestamp.now() - pd.Timedelta(days
updated_apps_percent = len(last_30_days) / len(data) * 100

# create a pie chart to visualize the percentage of apps that are updated frequentl
plt.figure(figsize=(6,6))
plt.pie([updated_apps_percent, 100-updated_apps_percent], labels=["Updated within l
plt.title("Percentage of Apps Updated Frequently")

# time to print
plt.show()
```



In [79]:
```
# 1. Why did you pick the specific chart?
# I picked a pie chart because it's a great way to visualize the percentage of apps

# 2. What is/are the insight(s) found from the chart?
# No apps were updated in the last 30 days. This suggests that app developers may n

# 3. Will the gained insights help creating a positive business impact?
# Are there any insights that lead to negative growth? Justify with specific reason

# Not updating apps for a long time can be bad for businesses because users may los
```

Which category of Apps from the 'Content Rating' column is found more on the play store?

In [80]:
```
# content rating of the apps
data = data['Content Rating'].value_counts()
labels = ['Everyone', 'Teen', 'Everyone 10+', 'Mature 17+','Adults only 18+', 'Unra

# using pie chart
plt.figure(figsize=(10,10))
explode=(0,0.1,0.1,0.1,0.0,1.3)
colors = ['orange', 'r', 'yellow', 'g', 'm', 'k']
plt.pie(data, labels = labels, colors = colors, autopct='%.2f%%',explode=explode,te
plt.title('Content Rating',size=20,loc='center')
plt.legend();
```



In [81]:
```
# 1. Why did you pick the specific chart?
# A Pie Chart is a great choice for visualizing the distribution of categories in t

# 2. What is/are the insight(s) found from the chart?
# Most of the apps on the play store have a content rating of 'Everyone' (81.80%).
#This is followed by 'Teen' (10.74%), 'Everyone 10+' (4.07%), 'Mature 17+' (3.34%),
#'Adults only 18+' (0.03%), and 'Unrated' (0.02%).

# 3. Will the gained insights help creating a positive business impact?
# Are there any insights that lead to negative growth? Justify with specific reason

# This information can help app developers to know that the majority of apps have a
#'Everyone' rating, which allows for a broad audience. But for businesses targeting
#like 'Mature 17+' or 'Adults only 18+', this information can be useful as there is
```

THANK YOU